ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη Αλγορίθμων Εποπτευόμενης Μάθησης, Συστημάτων Βασισμένα σε Κανόνες και Πειραματική Αποτίμηση

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΣΠΥΡΙΔΩΝΑ ΒΕΛΙΑΝΗ
(ΑΕΜ: 2800)

Επιβλέπων : Νικόλαος Δημόκας
Επίκουρος Καθηγητής

Καστοριά Μήνας - 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη Αλγορίθμων Εποπτευόμενης Μάθησης, Συστημάτων Βασισμένα σε Κανόνες και Πειραματική Αποτίμηση

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

του

ΣΠΥΡΙΔΩΝΑ ΒΕΛΙΑΝΗ
(ΑΕΜ: 2800)

Επιβλέπων : Νικόλαος Δημόκας
Επίκουρος Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 28 – 9 - 2022

.....	.....	.....
Νικόλαος Δημόκας	Δόσης Μιχαήλ	Βέργαδος Δημήτριος
Επίκουρος Καθηγητής	Καθηγητής	Επίκουρος Καθηγητής

Καστοριά Σεπτέμβριος - 2022

Περίληψη

Η εργασία χωρίζεται στα κεφάλαια «Εποπτευόμενη Μάθηση» (Supervised Learning), «Μάθηση χωρίς Επίβλεψη» (Unsupervised Learning) που αφορά τους αλγόριθμους μηχανικής μάθησης και στο κεφάλαιο «Συστήματα Βασισμένα σε Κανόνες». Τέλος, αναφέρονται τα «Συμπεράσματα» της εργασίας. Σε κάθε κεφάλαιο χρησιμοποιούνται εικόνες από το λογισμικό που χρησιμοποιείται με την λεζάντα της περιγραφής.

Για τα δυο πρώτα κεφάλαια, χρησιμοποιήθηκε το λογισμικό Weka, το οποίο περιγράφεται εκτενώς στο κεφάλαιο 3. Τα κεφάλαια αυτά, χωρίζονται σε τέσσερις και τρεις ενότητες αντίστοιχα όπου και περιγράφονται ισάριθμοι αλγόριθμοι, θεωρητικά και πειραματικά με το weka και παραθέτονται τα τελικά συμπεράσματα για τον καθένα.

Στο τέταρτο κεφάλαιο, το οποίο αποτελεί το δεύτερο μέρος της εργασίας γίνεται αναφορά στα συστήματα που βασίζονται σε κανόνες. Παράλληλα, παρουσιάζεται μια εφαρμογή – σύστημα βασισμένο σε κανόνες, η οποία υλοποιήθηκε για το σκοπό αυτό.

Τέλος, στο κεφάλαιο «Συμπεράσματα», παραθέτονται κάποια τελικά συμπεράσματα όσον αφορά το σύνολο της εργασίας, τις τεχνικές που παρουσιάστηκαν και την προσωπική εκτίμηση της εφαρμογής τον παραπάνω σε εφαρμογές του πραγματικού κόσμου.

Λέξεις Κλειδιά:

μάθηση, μηχανική, εποπτευόμενη, δεδομένα, πρόβλεψη, απόσταση, σμήνος, κατηγοριοποίηση, σύστημα, κανόνες, μνήμη, μακροχρόνια, βραχυχρόνια.

Abstract

The paper is divided into the chapters "Supervised Learning", "Unsupervised Learning" which concerns machine learning algorithms and the chapter "Rule Based Systems". Finally, the "Conclusions" of the paper are given. In each chapter, images of the software used are used with the description caption.

For the first two chapters, the software used is called Weka, which is described in detail in chapter 3. These chapters are divided into four and three sections respectively where equal numbers of algorithms are described, both theoretically and experimentally with weka, and the final conclusions for each one of them.

Chapter four, which is the second part of the paper, discusses rule-based systems. In this chapter, an application (an expert rule-based system) which was implemented for this purpose, is presented.

Finally, in the chapter "Conclusions", some final conclusions are given regarding the whole thesis, the techniques presented and a personal assessment of the implementation of the above in real world applications.

KeyWords:

Machine learning, algorithm, data, dataset, distance, input, output, prediction, classify, classification, clustering, train set, system, rule-based, long-term, sort-term.

Πίνακας Περιεχομένων

Λίστα Εικόνων

Λίστα Πινάκων

Εισαγωγή

Εδώ και περίπου τέσσερις δεκαετίες, είναι ξακουστός ο όρος «Τεχνητή Νοημοσύνη», Artificial Intelligence(AI εν συντομία). Ένας όρος ο οποίος, όπως και οι περισσότεροι που σχετίζονται με τις επιστήμες της τεχνολογίας, χρησιμοποιείται ευρέως χωρίς αυτό να προϋποθέτει τη βαθειά γνώση της έννοιας, που και πως υλοποιείται, γιατί εφευρέθηκε και κυρίως, που αποσκοπεί.

Το παρόν έγγραφο δεν αποτελεί εργασία έρευνας πάνω στην Τεχνητή Νοημοσύνη, αλλά κυρίως στο τι βρίσκεται πίσω από αυτή, δηλαδή πως και με ποια μέσα «υλοποιείται». Παρόλα αυτά αξίζει εισαγωγικά να γίνει μια σύντομη αναφορά ώστε να αποσαφηνισθούν κάποιες χρήσιμες για το έγγραφο έννοιες, οι οποίες χρησιμοποιούνται εμπειρικά κυρίως από το ευρύ κοινό.

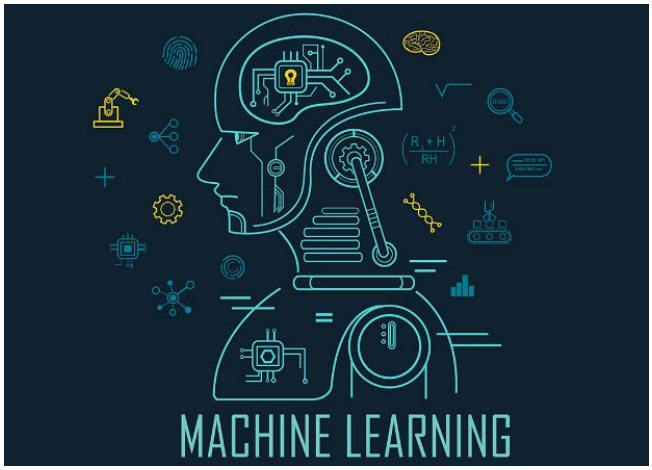
Καταρχάς, για να αναλύσουμε τον όρο «Τεχνητή Νοημοσύνη» λίγο πρόχειρα, θα λέγαμε πως Τεχνητό είναι οτιδήποτε δεν είναι φυσικό, δεν παράγεται δηλαδή με φυσικό τρόπο. Άρα η τεχνητή νοημοσύνη είναι η νοημοσύνη που παράγεται Τεχνητά, από κάποια μηχανή που δημιουργήθηκε από τον άνθρωπο. Από την άλλη για τον όρο «νοημοσύνη», θα μπορούσαμε να πούμε ότι είναι η δυνατότητα «παραγωγής σκέψης», με ό,τι αυτό συνεπάγεται (συλλογισμός, λήψη αποφάσεων, αντίδραση σε κάποιο ερέθισμα, έμπνευση και δημιουργία κλπ.). Ο άνθρωπος είναι ένα από τα όντα του φυσικού κόσμου το οποίο έχει νόηση – νοημοσύνη και λόγω της φυσιολογίας του, έχει τη δυνατότητα να την εξελίσσει, να την αποθηκεύει, να την μεταδίδει και να την μελετά. Οι τρεις τελευταίες έννοιες, της αποθήκευσης, της μετάδοσης και της μελέτης είναι πολύ βασικές για τη συνέχεια. Μάλιστα, μας οδηγούνε σε μία πολύ βασική για τη συνέχεια έννοια, αυτή της Μάθησης(Learning).

Η Μάθηση αποτελεί την πιο βασική και σημαντική ιδιότητα των οργανισμών με νόηση. Είναι η εξελικτική του δύναμη, αυτή που τον κρατά στην ζωή και δεν αφανίζεται. Αυτό που στην ουσία βρίσκεται πίσω από την τεχνητή νοημοσύνη, είναι η απάντηση στο ερώτημα «Πως μπορούμε να δημιουργήσουμε έναν υπολογιστή που θα μαθαίνει;». Ο σκοπός εδώ φυσικά δεν είναι ίδιος με τα δικά μας ένστικτα, αλλά η δυνατότητα να λαμβάνει μια συσκευή αποφάσεις αντί για εμάς, μιας και χειρίζεται τεράστιο όγκο από δεδομένα με τεράστιες ταχύτητες. Αυτό που θέλει η επιστήμη να πετύχει είναι η «Μηχανική Μάθηση» (Machine Learning).

“ *Μηχανική Μάθηση είναι το φαινόμενο κατά το οποίο ένα σύστημα βελτιώνει την απόδοσή του κατά την εκτέλεση μιας συγκεκριμένης εργασίας, χωρίς να υπάρχει ανάγκη να προγραμματιστεί εκ νέου* [1].”

Ως παράδειγμα μάθησης, ένας άνθρωπος στη βρεφική του ηλικία, μπορούμε να πούμε πως δεν έχει ανεπτυγμένη νοημοσύνη και λειτουργεί κατά βάση με τα ένστικτά του. Μεγαλώνοντας και ξεκινώντας να εξερευνά τον κόσμο, έρχεται σε επαφή με πολλά αντικείμενα και πολλά από αυτά που είναι βλαβερά, τα απομακρύνουμε ή τα τοποθετούμε σε ψηλά σημεία που του είναι αδύνατο να τα φτάσει. Μετά από καιρό, αν ένας ενήλικας τοποθετήσει ένα αντικείμενο, π.χ. ένα ξύλινο κουτί ως σκαλοπάτι για να φθάσει ένα αντικείμενο που είναι σε κάποιο ψηλό ράφι και το μικρό παιδί δει αυτή την κίνηση, μπορεί να συσχετίσει αυτό το γεγονός με το δικό του πρόβλημα, και να πάρει π.χ. μια καρέκλα, να τη βάλει κοντά στο ράφι που έχουμε τοποθετήσει το βλαβερό αντικείμενο, να ανέβει πάνω της και να το φτάσει. Βλέπουμε λοιπόν ότι λόγω νοημοσύνης, ένας άνθρωπος μπορεί να παράγει σκέψη, κάποιες φορές μετά από τη μάθηση, όπως σε αυτό το παράδειγμα και άλλες φορές φυσιολογικά γιατί απλά του το έχει χαρίσει η φύση, όπως συνηθίζουμε να λέμε.

Γνωρίζουμε πως ο ανθρώπινος εγκέφαλος μπορεί να αποθηκεύει τεράστιο όγκο δεδομένων και πληροφοριών. Επίσης, γνωρίζουμε πως ένας Ηλεκτρονικός Υπολογιστής μπορεί επίσης να αποθηκεύει τεράστιο όγκο δεδομένων, και ειδικά σήμερα με την εξέλιξη του internet, ο όγκος αυτός συνεχώς πολλαπλασιάζεται. Η ερώτηση, θα μπορούσε να είναι η εξής: «Πως μπορεί ένας Ηλεκτρονικός Υπολογιστής να εκμεταλλευτεί όλα αυτά τα δεδομένα ώστε να μπορεί να τα χρησιμοποιήσει με σκοπό τη μάθηση ώστε να μπορεί π.χ. να ξεχωρίζει αντικείμενα, να λαμβάνει αποφάσεις κλπ;» (προς όφελος δικό μας φυσικά…). Ή, για να φτάσουμε στο δια ταύτα, «Πως πραγματικά λειτουργεί η Μηχανική Μάθηση;».



Εικόνα 1 Machine Learning

Πηγή: https://becominghuman.ai/

Λαμβάνοντας υπόψη το παραπάνω παράδειγμα, για να «παραχθεί» ένα αποτέλεσμα που βασίζεται στη μάθηση χρειαζόμαστε δύο βασικά εργαλεία. Τα δεδομένα, και τον τρόπο αυτά θα μας οδηγούν σε συμπέρασμα. Αυτό που «κρύβεται» πίσω από την τεχνητή νοημοσύνη είναι ένας

τεράστιος όγκος δεδομένων και κάποιοι αλγόριθμοι που χειρίζονται τα δεδομένα και παράγουν κάποια αποτελέσματα, λαμβάνουν κάποιες αποφάσεις και εκτελούν κάποιες ενέργειες.

Για τον όγκο των δεδομένων και τη συλλογή τους είναι υπεύθυνες οι τεχνικές «εξόρυξης δεδομένων» (data mining), οι οποίες χρησιμοποιούν το διαδίκτυο, στατιστικές έρευνες, τους χρήστες του διαδικτύου με τις γνωστές τεχνικές «δεν είμαι ρομπότ» που συλλέγουν χρήσιμες πληροφορίες. Για παράδειγμα, μπορούμε να σκεφθούμε ότι όλες αυτές οι εικόνες με Λεζάντα «Βρες τους φωτεινούς σηματοδότες», στην ουσία τροφοδοτούν βάσεις δεδομένων με πληροφορίες που σκοπό έχουν, πιθανόν, τη δημιουργία ενός AI συστήματος πλοήγησης, ή και αυτόματης οδήγησης. Η εξόρυξη δεδομένων παρόλα αυτά, αν και εμπεριέχει ενδιαφέρουσες τεχνικές, δεν θα αναλυθεί στο παρόν έγγραφο.

Αυτό όμως που θα αναλυθεί είναι κάποιοι Αλγόριθμοι που επεξεργάζονται τα δεδομένα αυτά, γνωστοί και ως «Αλγόριθμοι Μηχανικής Μάθησης» (Machine Learning Algorithms). Αξίζει να πούμε ότι σήμερα η έννοιες αλγόριθμοι – ίντερνετ – διαφημίσεις είναι ένα γνωστό τρίπτυχο, μιας και όλοι πλέον καταλαβαίνουμε ότι από τις προτιμήσεις μας όσον αφορά τις σελίδες που επισκεπτόμαστε, προκύπτουν και τα ανάλογα προϊόντα που μας διαφημίζουν. Δεν είναι κάποια θεωρία συνωμοσίας πίσω από αυτό αλλά το marketing του σήμερα το οποίο εκμεταλλεύεται την εξέλιξη της τεχνητής νοημοσύνης προς όφελός της και πιθανόν προς όφελός μας. Όταν για παράδειγμα ψάχνουμε απεγνωσμένα ένα αγαθό ή μια υπηρεσία, αυτομάτως συλλέγονται αυτές οι πληροφορίες από τις σχετικές πλατφόρμες, οι αλγόριθμοι(όπως θα δούμε στη συνέχεια) συλλέγουν αυτά τα δεδομένα και με έναν «μαγικό τρόπο» ξαφνικά βλέπουμε σε κάθε διαφήμιση παρόμοια προϊόντα με αυτό που επιθυμούμε. Μπορούμε να πούμε ότι αυτό «εξυπηρετεί και τα ηλεκτρονικά καταστήματα γιατί αυξάνουν τις πωλήσεις τους, αλλά και οι καταναλωτές βρίσκουν ευκολότερα τα προϊόντα» [2].

Οι Αλγόριθμοι Μηχανικής Μάθησης οι οποίοι είναι υπεύθυνοι για αυτές τις «επιτυχημένες προβλέψεις», χωρίζονται σε τρεις βασικές κατηγορίες:

- την μάθηση με επίβλεψη(εποπτευόμενη) – supervised learning
- την μάθηση χωρίς επίβλεψη(μη – εποπτευόμενη) – Unsupervised learning
- τη μάθηση με ενίσχυση – reinforcement learning

Η πρώτη κατηγορία της μηχανικής μάθησης είναι η Εποπτευόμενη Μάθηση. Αρχικά, πρέπει να υπενθυμίσουμε πως ό,τι σημαίνει για τον άνθρωπο η έννοια μάθηση, πρέπει με κάποιο τρόπο να σημαίνει το ίδιο και για μια μηχανή. Ο βασικός σκοπός της εποπτευόμενης μάθησης είναι η δημιουργία κάποιων Αλγόριθμων, οι οποίοι θα ενέχουν κάποιο είδος εποπτείας, όπως ένας δάσκαλος επιβλέπει ένα παιδί καθώς μαθαίνει.

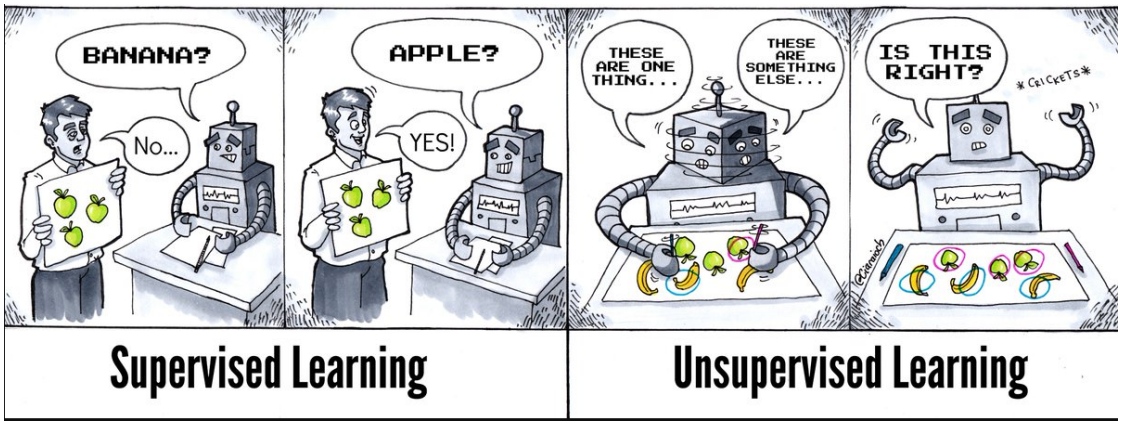
Οι αλγόριθμοι, από τον ορισμό τους, λειτουργούν δεχόμενοι ως είσοδο κάποια δεδομένα και παράγουν ως αποτέλεσμα κάποιο αποτέλεσμα ή κάποια αποτελέσματα. Ένας αλγόριθμος εποπτευόμενης μάθησης, λειτουργεί με αυτό τον τρόπο, μόνο που κάθε παράδειγμα με τα δεδομένα εισόδου, περιέχει και μία επιθυμητή έξοδο.

Η μάθηση με επίβλεψη χρησιμοποιεί ένα σύνολο εκπαίδευσης για να διδάξει τα μοντέλα να αποδίδουν την επιθυμητή έξοδο. Αυτό το σύνολο δεδομένων εκπαίδευσης περιλαμβάνει εισόδους και σωστές εξόδους, οι οποίες επιτρέπουν στο μοντέλο να μαθαίνει με την πάροδο του χρόνου. Ο αλγόριθμος μετράει την ακρίβειά του μέσω της συνάρτησης απωλειών, προσαρμόζοντας την μέχρι να ελαχιστοποιηθεί επαρκώς το σφάλμα.

Στην εποπτευόμενη μάθηση, όλα τα δεδομένα εισόδου είναι labeled, δηλαδή ήταν χαρακτηρισμένα με ένα όνομα-μια ετικέτα. Η έξοδος πάλι είναι επίσης μια αναμενόμενη τιμή από τα δεδομένα εισόδου.[1]

Στη μάθηση χωρίς επίβλεψη ή μη εποπτευόμενη μάθηση(unsupervised learning) ισχύει το αντίθετο. Τα δεδομένα εισόδου δεν έχουν «ετικέτες»(unlabeled) καθώς και η τιμή εξόδου(output) είναι «αβέβαιη». Οπότε στην ουσία, προσπαθούμε να μοντελοποιήσουμε αλγόριθμους οι οποίοι «μαθαίνουν» μοτίβα(καλούπια ή καλύτερα patterns) από τα δεδομένα, ώστε να δίνουν τις απαιτούμενες απαντήσεις.

Είναι αντιληπτό ότι ακόμα και αν ο προγραμματιστής αρχικά εισάγει κάποιο αλγόριθμο, αυτός στη συνέχεια θα τροποποιείται με βάση τα δεδομένα που θα πρέπει να δέχεται, και ως εκ τούτου κάποια στιγμή ο αλγόριθμος θα μας είναι τελείως άγνωστος. Αυτό είναι και το νόημα της μη εποπτευόμενης μάθησης άλλωστε.



Εικόνα 2 Ένα χιουμοριστικό σκίτσο που δείχνει τη διαφορά της εποπτευόμενης με τη μη εποπτευόμενη μάθηση.

[1] Δυστυχώς η μετάφραση στην Ελληνική προκαλεί σύγχυση, οπότε θα αναφέρονται έννοιες και στα αγγλικά παρά την προσπάθεια της πιο εύστοχης μετάφρασης.

Πηγή: https://www.analyticsvidhya.com/

Ας δούμε ένα θεωρητικό παράδειγμα που θα μπορούσε να προσομοιώσει την μη εποπτευόμενη μάθηση, αφού αναφερθούμε στο τι θα γινόταν με την εποπτευόμενη. Το γνωστό σε πολλούς παιγνίδι με τράπουλα «ξερή», έχει βασικούς κανόνες όπου κάθε παίκτης(υποθέτουμε ότι είναι 4 οι παίκτες) παίρνει 6 φύλλα, κάτω αφήνονται 4 και «κόβει» ο παίκτης που είναι η σειρά του να παίξει με βαλέ ή το ίδιο φύλλο που βρίσκεται στην κορυφή, αλλιώς τοποθετεί ένα από τα φύλλα του στην κορυφή των φύλλων που βρίσκονται κάτω. Αυτό διαρκεί 2 «γύρους» και η ομάδα που θα μαζέψει τα περισσότερα φύλλα ή τους περισσότερους πόντους θα κερδίσει το παιγνίδι του γύρου.

Αν υποθέσουμε ότι έχουμε μάθει αυτούς τους κανόνες σε έναν που είναι παρατηρητής του παιγνιδιού και υποτίθεται ότι υποστηρίζει μια ομάδα, τότε είναι φυσικό να γνωρίζει πότε θα χαίρεται ή θα πανηγυρίζει ανάλογα με το αν κερδίζει η ομάδα που υποστηρίζει ή να λυπάται όταν χάνει. Αν υποτίθεται ότι πρέπει το αποτέλεσμα πρέπει να είναι «λυπάμαι» ή «πανηγυρίζω» τότε το παραπάνω παράδειγμα με τα κατάλληλα δεδομένα θα μπορούσε να γίνει ένας αλγόριθμος εποπτευόμενης μάθησης.

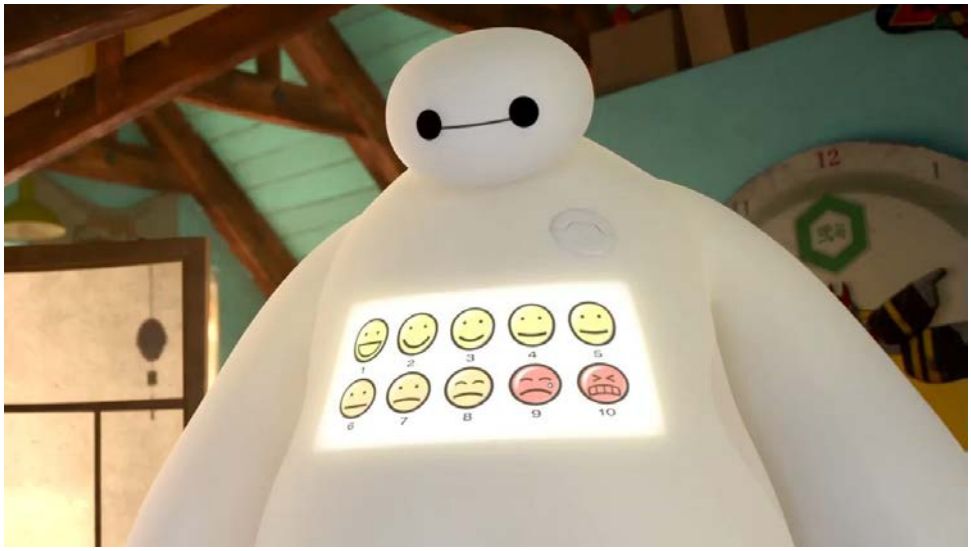
Για έναν παρατηρητή που δε γνωρίζει τους κανόνες παιχνιδιού πρέπει αρχικά να καταλάβει ποιες είναι οι ομάδες. Θα το αντιληφθεί εάν παρατηρήσει ποιοι μαζεύουν τα φύλλα στο ίδιο μέρος ενώ «κόβει» μια ο ένας και μια ο άλλος, άρα πρόκειται για ομαδικό παιγνίδι. Στη συνέχεια θα καταλάβει ότι ο βαλές είναι φύλο που ο παίκτης που το ρίχνει παίρνει και όσα είναι κάτω, ό,τι συμβαίνει και με το ίδιο φύλλο, τι είναι και πως γίνεται η ξερή κλπ. Μετά θα πρέπει να καταλάβει ότι η ομάδα που κερδίζει είναι αυτή που(ενδεχομένως) παίρνει τα περισσότερα φύλλα και(ή) κάνει τις περισσότερες ξερές. Αυτό το παράδειγμα θα μπορούσε να είναι ένας αλγόριθμος ΜΗ εποπτευόμενης μάθησης.

Η μάθηση με ενίσχυση από την άλλη λειτουργεί με έναν διαφορετικό τρόπο που επίσης μπορούμε να πούμε ότι προσομοιώνει δικές μας λογικές συμπεριφορές. Στην κατηγορία αυτή, δημιουργούνται αλγόριθμοι μηχανικής μάθησης οι οποίοι άλλοτε δίνουν ακριβή αποτελέσματα – προβλέψεις και άλλοτε όχι. Συνεπώς, αυτοί που δίνουν σωστές προβλέψεις κατά κάποιο τρόπο «επιβραβεύονται» ή ενισχύονται, ενώ οι υπόλοιποι «τιμωρούνται». Δε θα γίνει καμία περεταίρω αναφορά στην κατηγορία αυτή στη συνέχεια, απλά αναφέρεται σε αυτό το σημείο.

Τα Συστήματα Βασισμένα σε Κανόνες(rule – based systems) από την άλλη, είναι συστήματα που δημιουργούνται πάνω σε ήδη υπάρχουσες γνώσεις. Στην θεωρία μπορούμε να τα παρομοιάσουμε με απλούς αλγόριθμους, αλγόριθμους που για παράδειγμα, με δεδομένα εισόδου Χ θα μας δίνουν πάντα την έξοδο Υ. Αυτό που αποτελεί δομικό στοιχείο της τεχνικής αυτής είναι η γνώση. Σε μια ιατρική εφαρμογή για παράδειγμα, η οποία βασίζεται στην τεχνητή νοημοσύνη,

είναι προφανές ότι μια προσέγγιση με μηχανική μάθηση η οποία δε θα μας δίνει σε γενναίο ποσοστό έγκυρες προβλέψεις, ίσως στοιχίσει έως και τη ζωή ενός ασθενούς. Σε εμπορικές εφαρμογές, ένα ποσοστό σφάλματος δε θα έχει αντίστοιχα σοβαρές συνέπειες.

Τα Έμπειρα Συστήματα Βασισμένα σε Κανόνες (expert rule – based systems), είναι συστήματα που προσομοιώνουν θα λέγαμε έναν ειδικό ο οποίος έχει εμπειρία σε κάποιο γνωστικό αντικείμενο, π.χ. ιατρική, δικηγορία και λοιπά. Σκοπός του δηλαδή, είναι να συμπεριφέρεται σα να είναι ο ειδικός, μια εφαρμογή δηλαδή που θα παίζει το ρόλο του. Όπως στη γνωστή ταινία της Disney “Big Hero 6” όπου ο πρωταγωνιστής προσπαθούσε να υλοποιήσει μια συσκευή ρομπότ, η οποία θα λειτουργούσε σαν ένας ιατρός.



1 Baymax Από την ταινία Big Hero 6

Πηγή: https://www.ign.com/articles/big-hero-6-series-baymax-coming-to-disney

1. Εποπτευόμενη Μάθηση

1.1 Nearest Neighbors & K-Nearest Neighbors

Ο πρώτος αλγόριθμος που θα αναφερθεί στο κεφάλαιο αυτό, είναι ο πιο απλός στο να παρουσιαστεί, εύκολα να περιγραφεί τόσο ως μεθοδολογία όσο και ως προς την κατανόηση γενικότερα της μηχανικής μάθησης. Ανήκει στην κατηγορία των lazy(τεμπέλης) [2] αλγόριθμων και είναι ο αλγόριθμος των κοντινότερων γειτόνων ή Nearest Neighbors Algorithm.

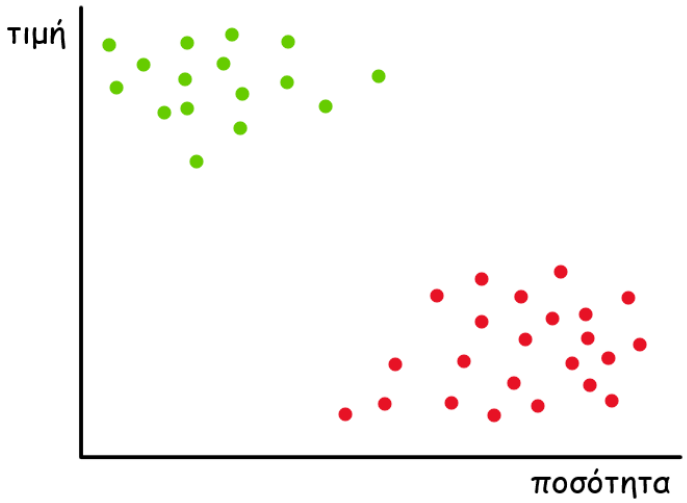
Αξίζει να σημειωθεί ξανά πως ο τρόπος με τον οποίο μια μηχανή μπορεί να αντλεί δεδομένα διαφέρει ανάλογα με το μοντέλο που επιθυμούμε να εξετάσουμε. Αν για παράδειγμα προσπαθούμε να δημιουργήσουμε έναν αλγόριθμο που θα υπολογίζει με ακρίβεια ένα είδος φυτού που όμοιά του υπάρχουν εκατοντάδες και δεδομένα του μπορούμε να αντλήσουμε μόνο από γεωπονικά βιβλία και όχι από το διαδίκτυο, τότε είναι πολύ πιθανό να μην έχουμε το σωστή κατηγοριοποίηση δεδομένων(classification) και ως εκ τούτου αποτυγχάνει ο αλγόριθμος στο να προβλέπει τα αποτελέσματα(να έχει δηλαδή μεγάλο ποσοστό ανακρίβειας). Οπότε θα υποθέσουμε πως τα δεδομένα δεν είναι το φλέγον ζήτημα, μιας και αυτά που ενδιαφέρουν περισσότερο την επιστήμη μας, και για λόγους αγοραστικούς αλλά και γενικότερα βρίσκονται σε πληθώρα στο διαδίκτυο.

1.1.1 Ένα Θεωρητικό Παράδειγμα

Σε ένα σύστημα εποπτευόμενης μάθησης χρειαζόμαστε παραδείγματα με ετικέτες(labeled examples) τα οποία είναι “εκπαιδευμένα” από τον supervisor[2]. Κάθε παράδειγμα μπορεί να χαρακτηριστεί ως πίνακας ή διάνυσμα το οποίο έχει κάποιες ιδιότητες(ως παράμετροι) και είναι συνδεδεμένες με μια έξοδο.

Ας υποθέσουμε ότι το παράδειγμά μας σχετίζεται με πελάτες οι οποίοι αγοράζουν προϊόντα συγκεκριμένης κατηγορίας, π.χ. κοσμήματα και εμείς επιθυμούμε να μελετήσουμε την συμπεριφορά ενός νέου καταναλωτή ώστε με τη βοήθεια αλγόριθμου να τον κατηγοριοποιήσουμε στην κατηγορία αυτών που αγοράζουν λίγα και ακριβά κοσμήματα και αυτών που αγοράζουν φθηνά και περισσότερα. Ας υποθέσουμε ότι με τις πράσινες κουκίδες είναι οι καταναλωτές Α και με τις κόκκινες οι καταναλωτές Β.

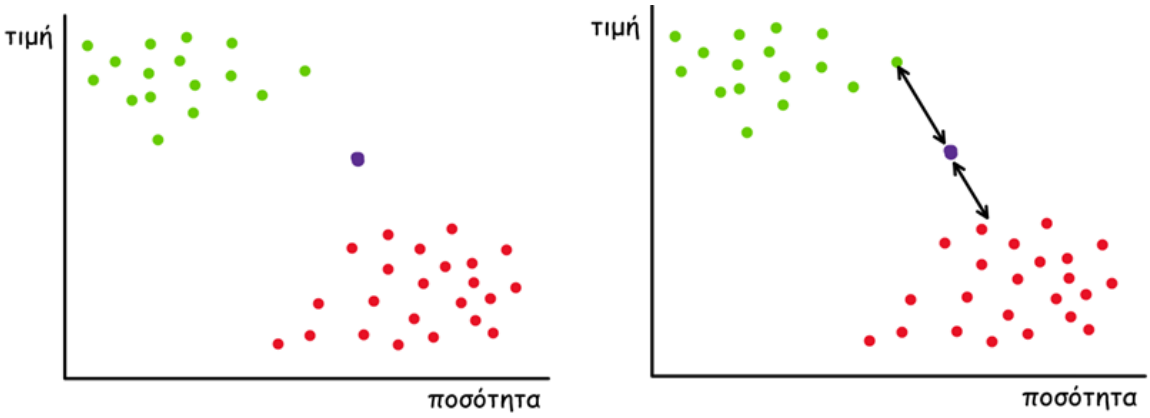
[2] Οι ορολογίες στα ελληνικά δε συνάδουν με τις αγγλικές αλλά θα γίνει μια προσπάθεια προσέγγισης και φυσικά θα αναφέρονται και οι αντίστοιχες αγγλικές.



Εικόνα 1.1 Δύο χαρακτηρίστηκα για να κατηγοριοποιήσουμε έναν καταναλωτή.

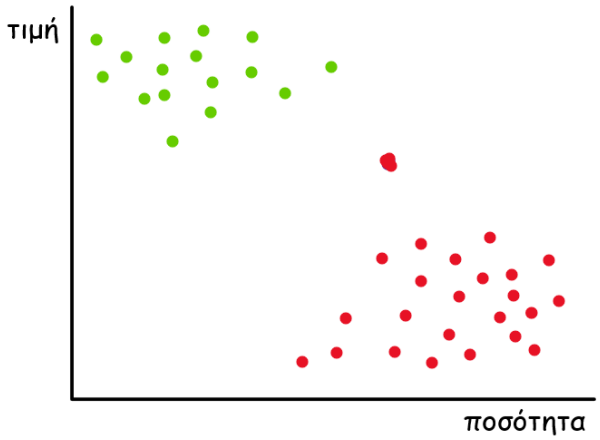
Ο αλγόριθμος των κοντινότερων γειτόνων με απλά λόγια λειτουργεί ως εξής:

Υποθέτουμε ότι στο σύνολο εισάγεται ένα νέο δεδομένο(νέος καταναλωτής) ως κουκίδα και πρέπει ο αλγόριθμος να είναι σε θέση να τον κατηγοριοποιήσει. Ο αλγόριθμος αυτός θα κάνει το πιο απλό. Θα υπολογίσει την κοντινότερη σε απόσταση κουκίδα (π.χ. την ευκλείδεια απόσταση $\sqrt{(x1-x2)^2+(y1-y2)^2}$) και θα κατηγοριοποιήσει τον καταναλωτή με βάση την πιο κοντινή κουκίδα, δηλαδή τον πιο κοντινό γείτονα.



Εικόνα 1.2 Nearest Neighbors - Υπολογισμός Διαστημάτων

Ως αποτέλεσμα έχουμε να κατηγοριοποιηθεί ο καταναλωτής στην κατηγορία Β



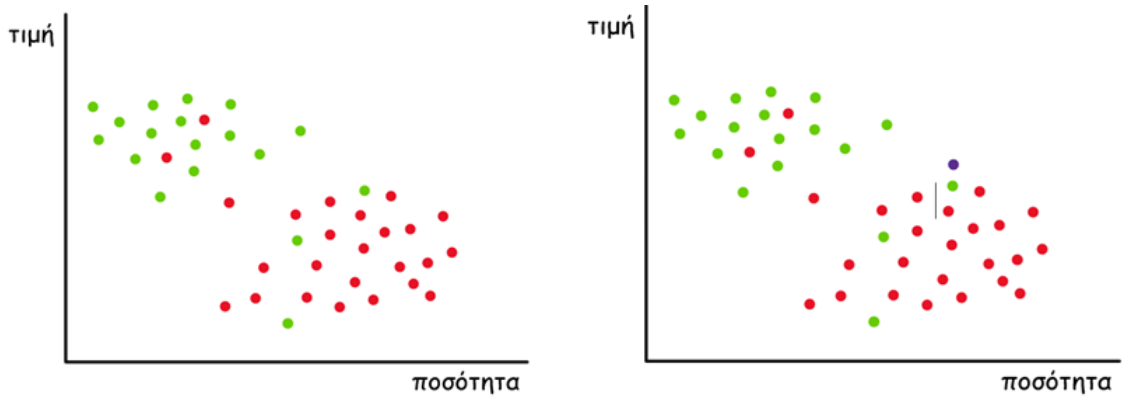
Εικόνα 1.3 Nearest Neighbors, μετά την κατηγοριοποίηση(Classification)

Αν υποθέσουμε πως θα θέλαμε να εμφανίζονται οι κατάλληλες διαφημίσεις ανάλογα με την κατηγορία του καταναλωτή, τότε αυτός ο αλγόριθμος πιθανό να έδειχνε σε έναν χρήστη του διαδικτύου τα επιθυμητά προϊόντα, με σκοπό οι επιχειρήσεις να μεγιστοποιούν τα κέρδη και οι πελάτες να είναι ικανοποιημένοι. Μην ξεχνάμε πως υπάρχει κάποιος σκοπός για όλα αυτά.

Στην ουσία, αυτό που κάνει ο αλγόριθμος είναι να κατηγοριοποιεί το νέο στιγμιότυπο (instance) αναζητώντας στο set ποιο είναι το πιο «ταιριαστό» με αυτό. Τα στιγμιότυπα αντιπροσωπεύουν την υπάρχουσα γνώση. Ο λόγος που ονομάζεται lazy, είναι διότι στην ουσία δεν κάνει τίποτα μέχρις ότου να έρθει η στιγμή να κάνει μια πρόβλεψη.

Στο παράδειγμα που είδαμε το σετ είναι μάλλον αρκετά ιδανικό, δηλαδή ο διαχωρισμός των στιγμιοτύπων - καταναλωτών είναι ξεκάθαρος. Υποθέσαμε δηλαδή πως η κατηγορία Α με τη Β έχουν τόσο μεγάλη διαφορά στην προτίμηση ποιότητας με ποσότητα, που στο ένα μέρος του γραφήματος είναι μόνο οι καταναλωτές τύπου Α και στο άλλο μόνο οι τύπου Β.

Τι θα συμβεί όμως αν το dataset μας δεν είναι τόσο «ιδανικό»; Προφανώς ο αλγόριθμος κοντινότερων γειτόνων θα μας δίνει λανθασμένες προβλέψεις και αυτό διότι μόλις εισέρχεται ένα νέο στιγμιότυπο στο σετ το οποίο είναι σε κοντινότερη απόσταση σε κάποιο που δεν ανήκει στην ίδια κατηγορία με τα υπόλοιπα «κοντά» του. Π.χ.



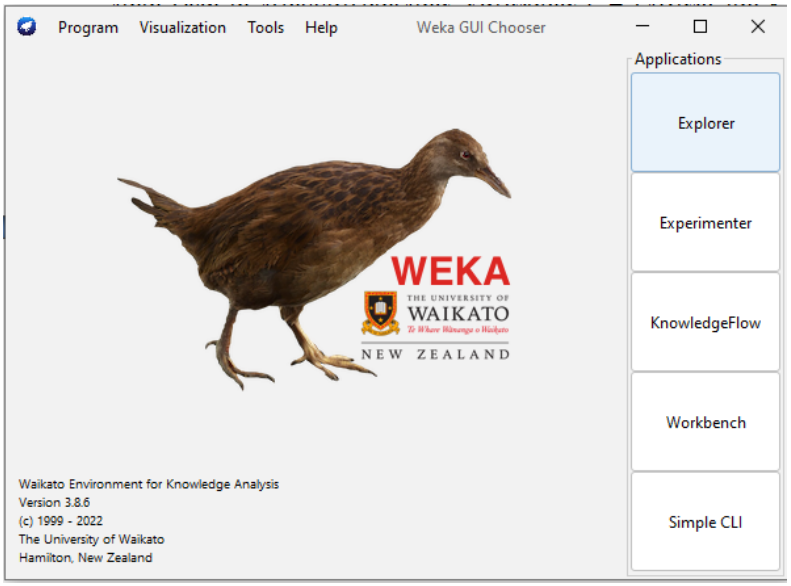
Εικόνα 1.4 Ένα στιγμιότυπο εισέρχεται σε ένα πιο περίπλοκο dataset

Στην εικόνα 4 βλέπουμε πως με βάση τον αλγόριθμο nearest neighbors θα προβλεφθεί ότι το νέο στιγμιότυπο θα χαρακτηριστεί ως τύπου Β(πράσινο) διότι ο κοντινότερος γείτονας είναι ένα στιγμιότυπο τύπου Β.

Για τέτοιες περιπτώσεις όπου το dataset είναι πιο «πολύπλοκο», χρησιμοποιείται μια τεχνική αναζήτησης των Κ-κοντινότερων γειτόνων (k nearest neighbors). Σε αυτή την αλγοριθμική προσέγγιση, αναζητούμε έναν αριθμό κοντινότερων γειτόνων, Κ στον αριθμό, εξ ου και το όνομα του αλγόριθμου. Για παράδειγμα, αναζητούμε τους 5 γείτονες που βρίσκονται πιο κοντά στο νέο στιγμιότυπο και το κατηγοριοποιούμε ανάλογα με το ποιοι είναι οι περισσότεροι(ποιας κατηγορίας). Η επιλογή του Κ, επηρεάζει κατά πολύ την πρόβλεψη χωρίς να υπάρχει κάποιος τρόπος να πούμε με ακρίβεια πόσο πρέπει να είναι το Κ ώστε το αποτέλεσμα να έχει περισσότερη ακρίβεια(accuracy) και όσο το δυνατόν λιγότερο ποσοστό σφάλματος.

1.1.2 Nearest Neighbors with Weka

Για την καλύτερη και πιο ουσιαστική παρουσίαση του αλγόριθμου NN και KNN, θα χρησιμοποιηθεί το λογισμικό Weka, για το οποίο γίνεται εκτενέστερη αναφορά στο 3^{ο} κεφάλαιο. Εδώ για χάρη ευκολίας θα αναφέρονται σύντομα κάποιες πληροφορίες ώστε να γίνονται κατανοητές οι εικόνες. Σκοπός είναι να πειραματιστούμε με δεδομένα ώστε να συγκριθούν τα αποτελέσματα για διάφορες τιμές του Ν και καταλήξουμε σε κάποια βασικά συμπεράσματα σχετικά με τον αλγόριθμο σε πειραματική βάση,



Εικόνα 1.5 Το Λογισμικό Weka

Από τις επιλογές Applications επιλέγουμε το Explorer ώστε να ανοίξει το παράθυρο το οποίο περιέχει τους αλγόριθμους κατηγοριοποίησης(classifiers), φίλτρα επεξεργασίας των χαρακτηριστικών(attributes).

Τα δεδομένα που χρησιμοποιούμε στο παράδειγμα βρίσκονται στο Weka ως datasets σε μια μορφή που χειρίζεται το Weka τα δεδομένα, σε αρχεία δηλαδή που έχουν την επέκταση «.arff». Το αρχείο που θα χρησιμοποιηθεί εδώ είναι το "glass.arff" το οποίο αναφέρεται ως μια «Βάση Δεδομένων Αναγνώρισης Γυαλιού» ώστε να προσδιορίζει πότε ένα γυαλί είναι τύπου "float" και πότε όχι[3] [2] Η δομή του γυαλιού αποτελείται από τα χημικά στοιχεία που είναι και οι μεταβλητές του Dataset:

```
% 7. Attribute Information:
%    1. Id number: 1 to 214
%    2. RI: refractive index
%    3. Na: Sodium (unit measurement: weight percent in
corresponding oxide, as
%                    are attributes 4-10)
%    4. Mg: Magnesium
%    5. Al: Aluminum
%    6. Si: Silicon
%    7. K: Potassium
%    8. Ca: Calcium
%    9. Ba: Barium
%   10. Fe: Iron
```

Εικόνα 1.6 Σχόλια στο αρχείο glass.arff

Πιο συγκεκριμένα το αρχείο glass.arff, αποτελείται από τις ιδιότητες και τα δεδομένα ως εξής:

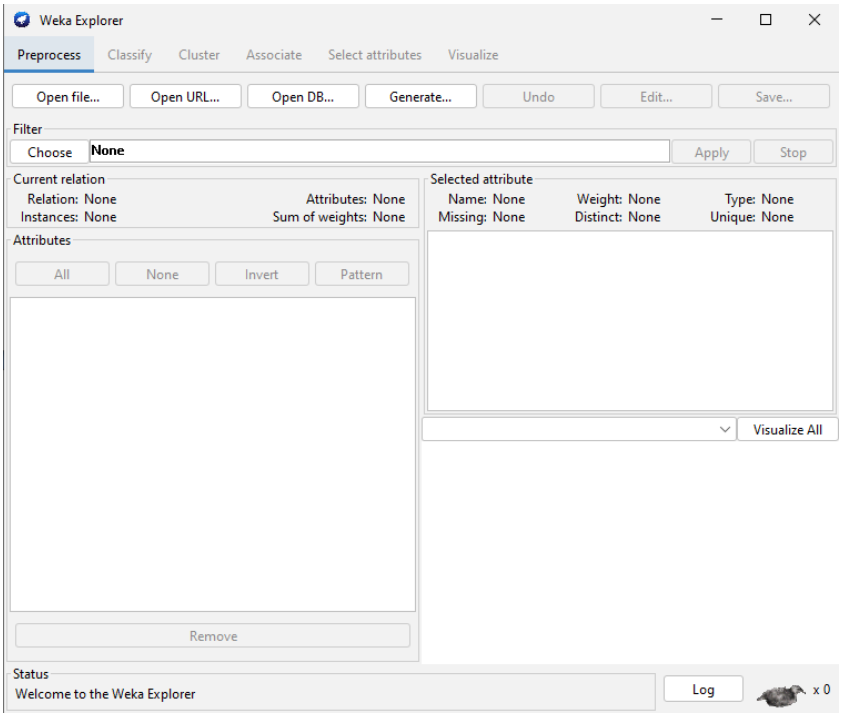
[3] Η μελέτη της ταξινόμησης των τύπων γυαλιού είχε ως κίνητρο εγκληματολογική έρευνα. Στον τόπο του εγκλήματος, το γυαλί που αφήνεται, μπορεί να χρησιμοποιηθεί ως αποδεικτικό στοιχείο... αν ταυτοποιηθεί σωστά!

```
@relation Glass
@attribute 'RI' numeric
@attribute 'Na' numeric
@attribute 'Mg' numeric
@attribute 'Al' numeric
@attribute 'Si' numeric
@attribute 'K' numeric
@attribute 'Ca' numeric
@attribute 'Ba' numeric
@attribute 'Fe' numeric
@attribute 'Type' { 'build wind float', 'build wind non-
float', 'vehic wind float', 'vehic wind non-float',
containers, tableware, headlamps}
@data
1.51793,12.79,3.5,1.12,73.03,0.64,8.77,0,0,'build wind float'
1.51643,12.16,3.52,1.35,72.89,0.57,8.53,0,0,'vehic wind float'
1.51793,13.21,3.48,1.41,72.64,0.59,8.43,0,0,'build wind float'
1.51299,14.4,1.74,1.54,74.55,0,7.59,0,0,tableware
1.53393,12.3,0,1,70.16,0.12,16.19,0,0.24,'build wind non-
float'
1.51655,12.75,2.85,1.44,73.27,0.57,8.79,0.11,0.22,'build wind
non-float'
1.51779,13.64,3.65,0.65,73,0.06,8.93,0,0,'vehic wind float'
1.51837,13.14,2.84,1.28,72.85,0.55,9.07,0,0,'build wind float'
```

1.7 Το αρχείο glass.arff

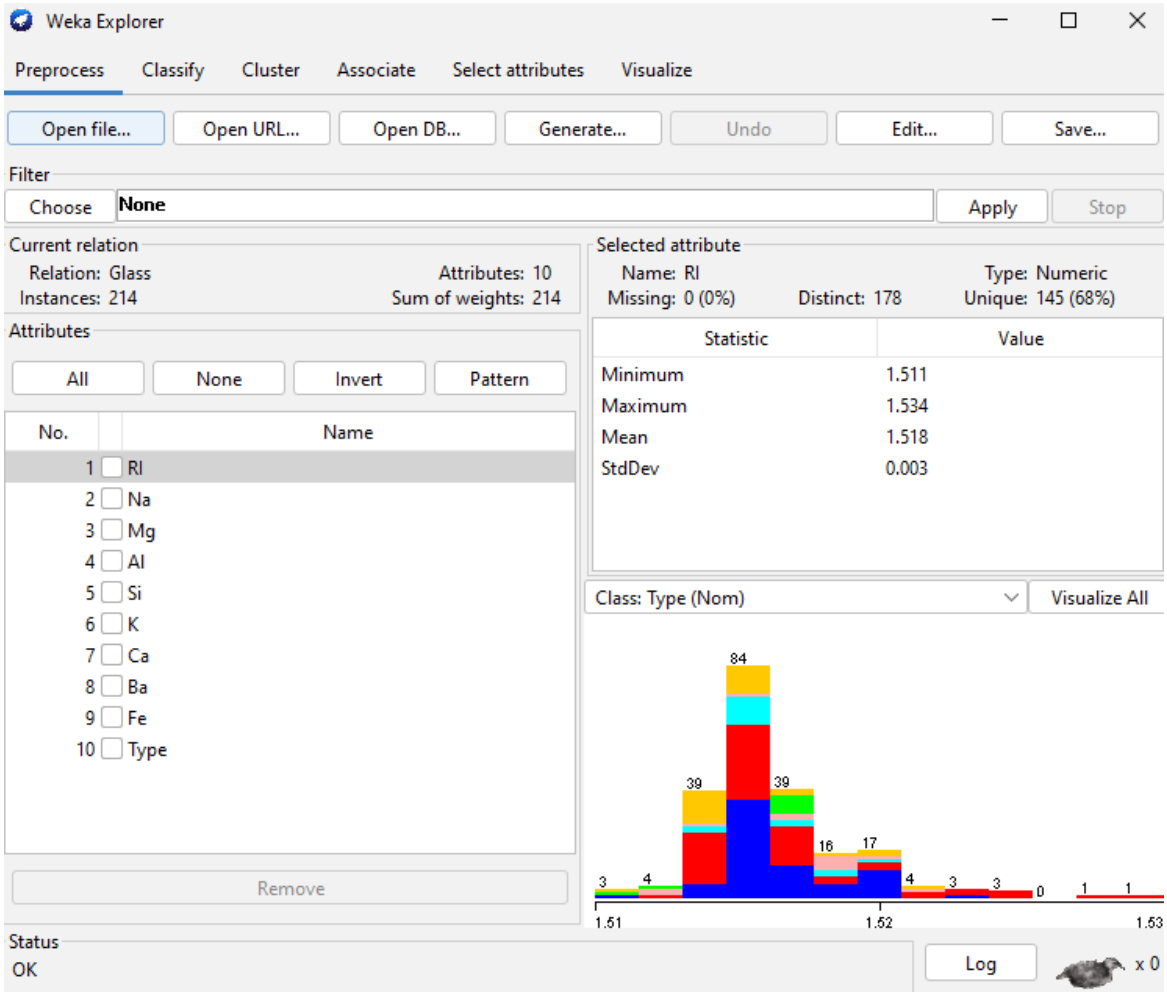
όπου, η τελευταία ιδιότητα μας δίνει την πληροφορία σχετικά με τον σκοπό για τον οποίο κατασκευάστηκε το γυαλί του συγκεκριμένου στιγμιότυπου.

Το σύμβολο @ είναι ένας χαρακτήρας προσδιορισμού στα αρχεία και τοποθετείται πριν από τις ιδιότητες, τη λέξη data από την οποία κι έπειτα παραθέτονται οι τιμές των attributes για κάθε στιγμιότυπο, και πριν από τον τίτλο του Dataset. Οι ιδιότητες στο σετ είναι αριθμητικές(numeric) σε εξαίρεση με την τελευταία που είναι «ονομαστική»(nominal). Ανοίγοντας το weka στο explorer έχουμε αυτό το παράθυρο:



Εικόνα 1.8 Weka Explorer

Η επιλογή που μας ενδιαφέρει(και φυσικά θα θεωρείται αυτονόητη παρακάτω) είναι η Open File, από την οποία θα βρούμε το αρχείο glass.arff και επιλέγοντάς το θα έχουμε την εικόνα αυτή:

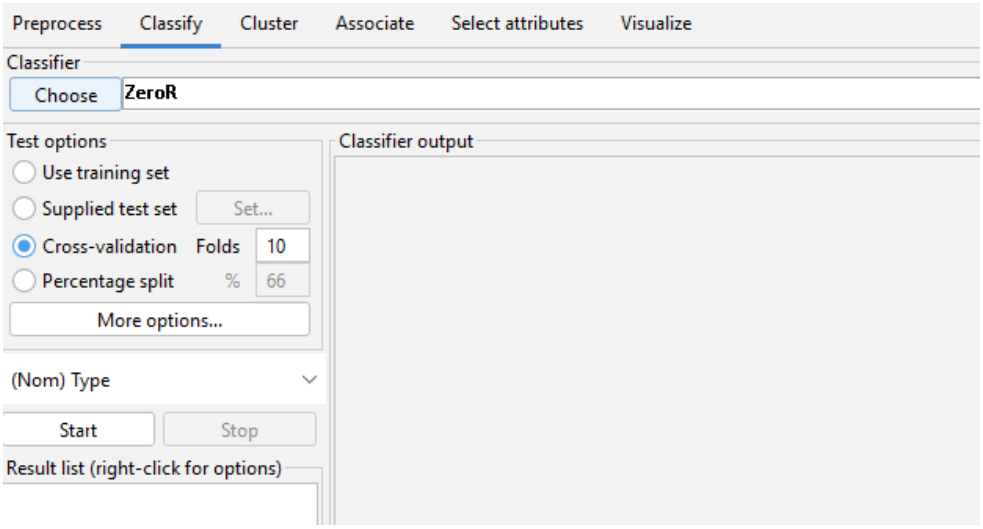


Εικόνα 1.9 Το αρχείο glass.arff στο Weka explorer

Η πρώτη εικόνα είναι αυτή, στην οποία φαίνονται όλα τα attributes, όπου επιλέγοντας ένα συγκεκριμένο, έχουμε κάποια στατιστικά στοιχεία όπως μέγιστη – ελάχιστη τιμή στο σετ, μέσος όρος(mean) και τυπική απόκλιση.

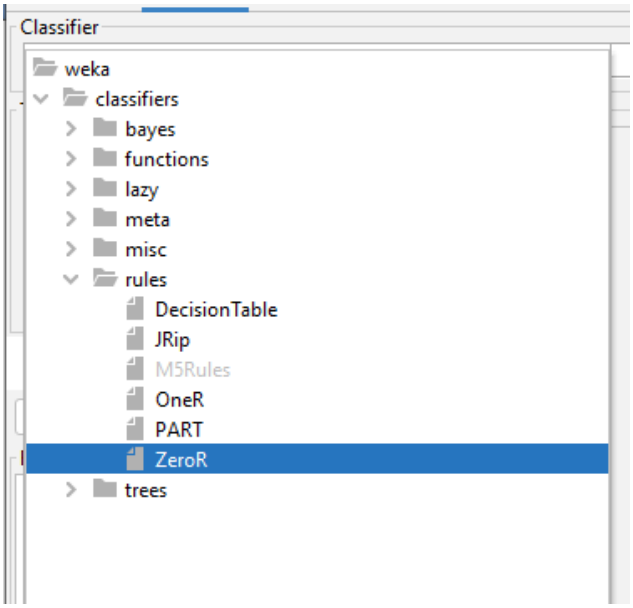
Μπορούμε να «φιλτράρουμε» κάποια attributes, κάτι το οποίο ίσως δούμε αργότερα, Αυτό που μας ενδιαφέρει είναι η επιλογή Classify, η οποία μας οδηγεί στους αλγόριθμους που θα μελετηθούν. Μπορούμε επίσης, να διαγράψουμε attributes σε περίπτωση που για παράδειγμα δεν επηρεάζουν καθόλου την πρόβλεψη.

Από εκεί λοιπόν, αφού επιλέγουμε με κλικ μας ανοίγει το παράθυρο:

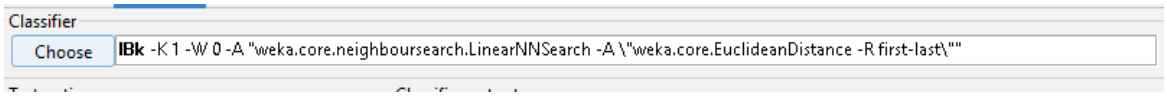


Εικόνα 1.10 Επιλέγοντας Classifier

Η επιλογή Choose, μας ανοίγει ένα αναδυόμενο μενού, από το οποίο επιλέγουμε τους αλγόριθμους κατηγοριοποίησης:

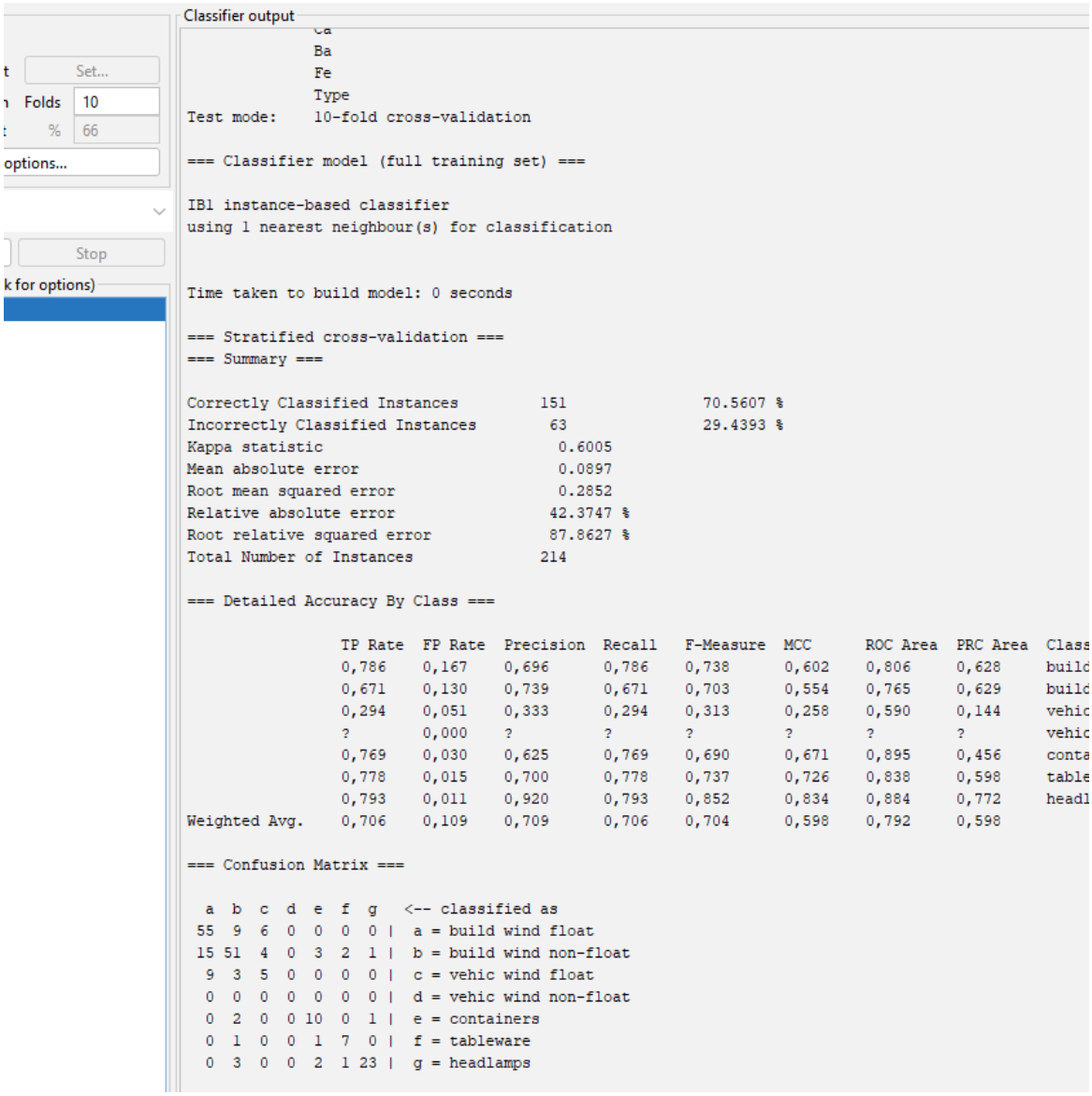


και από την επιλογή Lazy, επιλέγουμε το IBk[4].



όπως φαίνεται, ο υπολογισμός της «απόστασης» γίνεται με την Ευκλείδεια Απόσταση. Πατώντας το “Start” έχουμε:

[4] Από το instance base learning = μάθηση με βάση την περίπτωση. Στο weka ο αλγόριθμος nearest neighbors αναφέρεται με αυτή την ορολογία, όπου δηλαδή, κάθε στιγμιότυπο κατηγοριοποιείται με βάση την περίπτωση στην οποία βρίσκεται όλο το σετ. [2]

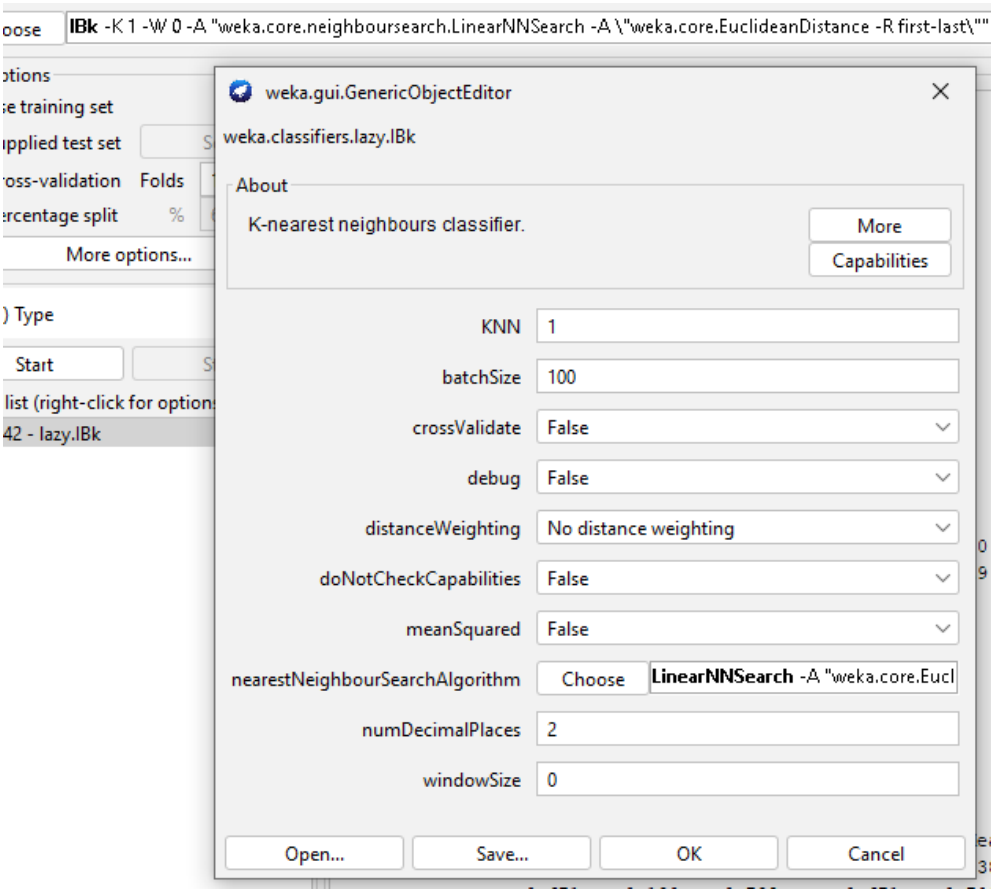


Εικόνα 1.11 Nearest Neighbor Output

Αρχικά ας δούμε την τιμή του Correctly Classified Instances(σωστά κατηγοριοποιημένα στιγμιότυπα) ή αλλιώς accuracy(ακρίβεια) των αποτελεσμάτων. Με ποια ακρίβεια δηλαδή κατηγοριοποιείται ένα στιγμιότυπο. Το 70.5% υποδηλώνει αυτή την ακρίβεια στην πρόβλεψη του αλγόριθμου.

1.1.3 K-Nearest Neighbors

Η προεπιλογή στο παραπάνω παράδειγμα είναι, φυσικά, k = 1, την οποία μπορούμε να αλλάξουμε πατώντας πάνω στο όνομα του αλγόριθμου που έχουμε επιλέξει και μας εμφανίζεται αυτό το μενού:



Εικόνα 1.12 Αλλάζοντας το K

Ας αλλάξουμε το KNN σε 5, ώστε να πειραματιστούμε

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         145               67.757  %
Incorrectly Classified Instances        69               32.243  %
Kappa statistic                          0.5469
Mean absolute error                      0.1085
Root mean squared error                  0.2563
Relative absolute error                 51.243  %
Root relative squared error             78.9576 %
Total Number of Instances              214
```

Το αποτέλεσμα είναι να πάρουμε έναν λιγότερο accurate αλγόριθμο. Αυτό σημαίνει ότι το Set δεν είναι αρκετά noisy. Δηλαδή τα δεδομένα είναι μάλλον περισσότερο όπως στο πρώτο θεωρητικό παράδειγμα παρά προς το 2ο.

Αν το αλλάξουμε σε 10 τότε το ποσοστό μειώνεται στο 66% ενώ για K = 20 κοντά στο 65%, δηλαδή όλο και χαμηλότερο.

Αν είχαμε ένα noisy set τότε θα βλέπαμε βελτίωση σε κάποια αύξηση του k αλλά και πάλι μέχρι ένα συγκεκριμένο όριο. Αν το k πάρει μια υπερβολικά μεγάλη τιμή τότε προφανώς η ακρίβεια των προβλέψεων πάλι θα πέφτει.

Αν πάλι θέσουμε το k κοντά στο μέγεθος του dataset, τότε στην ουσία παίρνουμε την απόσταση όλων των σημείων και βρίσκοντας τον μέσο όρο τους, πιθανόν να έχουμε μια τιμή πολύ κοντά στο baseline accuracy όπου στο weka δίνεται από τον αλγόριθμο που είναι στην προεπιλογή. Αν π.χ. θέσω το k = 100, τότε το accuracy θα είναι κοντά στο 35% όσο περίπου μου δίνει και ο ZeroR αλγόριθμος.

1.1.4 Συμπεράσματα

Στα θετικά του NN και του KNN είναι φυσικά η απλότητά τους και ως λογική και ως υλοποίηση, όπως ειπώθηκε και αρχικά, καθώς επίσης και το ότι συχνά είναι πολύ ακριβής. Το 70% του παραδείγματος δεν είναι καθόλου μικρό ποσοστό, καθώς και τα ποσοστά με k > 1.

Παρόλα αυτά είναι εξαιρετικά αργοί. Απαιτείται σάρωση όλου του training set για κάθε πρόβλεψη, χαρακτηριστικά πιο αργά από άλλες μεθόδους. Επίσης, θεωρεί όλα τα attributes είναι εξίσου σημαντικά. Θα μπορούσαμε να δώσουμε κάποιο διαφορετικό «βάρος» σε χαρακτηριστικά που είναι περισσότερο και λιγότερο σημαντικά. Στα sets που είναι «θορυβώδη», μπορούμε να επιλέξουμε το κατάλληλο k, το οποίο θα μας δίνει το καλύτερο και πιο ακριβές αποτέλεσμα.

1.2 Decision Trees – Δένδρα Απόφασης

Ίσως η πιο «διαδεδομένη» τεχνική υπολογιστικών συστημάτων, αλλά και ειδικότερα των αλγόριθμων μηχανικής μάθησης, είναι τα δένδρα απόφασης. Τα δένδρα στον προγραμματισμό είναι γνωστά ήδη από το Λύκειο, οι υποκατηγορίες τους, αυτές των δυαδικών δένδρων, τα ΔΔΑ(δένδρα δυαδικής αναζήτησης) και τα δένδρα απόφασης που αποτελούν μια τεχνική λήψης αποφάσεων.

Τα δένδρα απόφασης, είναι στην πλειοψηφία τους δυαδικά, χωρίς αυτό να είναι κανόνας, και η δομή τους έχει ως εξής:

- Κάθε κόμβος του δένδρου εκτός από τα φύλλα, είναι μια ερώτηση(συνήθως οι ακμές είναι η απάντηση ναι ή όχι, άλλοτε μια ανίσωση π.χ. >= 10 η δεξιά ακμή και < 10 η αριστερή κλπ που μας οδηγούν σε επόμενο κόμβο). Τα φύλλα είναι η «απάντηση» στις παραπάνω ερωτήσεις η οποία μας οδηγεί και στην τελική απόφαση.

Στο ML, τα δένδρα απόφασης, όπως είναι προφανές, χρησιμοποιούν τα δεδομένα του dataset ώστε να προβλέψουν κάποιο μελλοντικό αποτέλεσμα. Οι κόμβοι είναι attributes, οι οποίοι λειτουργούν σαν ερωτήσεις ώστε να οδηγηθούμε στον επόμενο κόμβο ο οποίος είναι επίσης κάποιο attribute μέχρι να φθάσουμε σε κάποιο φύλλο.

Εδώ πρέπει να γίνουν αντιληπτά δυο βασικά πράγματα:

Το **πρώτο** είναι, όπως είναι λογικό, ότι δεν είναι όλα τα datasets κατάλληλα για δένδρα απόφασης. Θα δούμε σε παράδειγμα ότι θα προκύπτουν δένδρα πολύ μεγάλα, δυσνόητα και μη αποτελεσματικά ή μη αποδοτικά όσον αφορά την ακρίβεια πρόβλεψης. Φυσικά, υπάρχουν τεχνικές «κλαδέματος» (pruning), στις οποίες ουσιαστικά πρέπει να αφαιρεθούν τα λιγότερο σημαντικά attributes από τα δεδομένα μας προς όφελος της ακριβέστερης πρόβλεψης.

Το **δεύτερο** είναι ότι τα δένδρα απόφασης βασίζονται σε κάποιους λογικούς κανόνες. Π.χ. «αν έξω βρέχει πάρε ομπρέλα, αν δε βρέχει μην παίρνεις». Οι κανόνες απόφασης είναι συνυφασμένοι με την λογική, οπότε σε ένα πιο σύνθετο πρόβλημα όπου οι ερωτήσεις πριν από την απόφαση θα είναι περισσότερες, μας οδηγούν σε μια **ιεραρχία** των ερωτήσεων που για τον άνθρωπο είναι αρκετά δύσκολο να χειριστεί «με το χέρι», καθώς επίσης συχνά μας οδηγεί σε τελείως αντιφατικά αποτελέσματα. Αυτό συμβαίνει γιατί το σύνολο των κανόνων που καθορίζουν την ταξινόμηση των ερωτήσεων, είναι πολύ μεγάλο.

“Η ευκολία χειρισμού των δεδομένων από ηλεκτρονικούς υπολογιστές μας δίνει τη δυνατότητα να αυτοματοποιούμε τρόπους που παράγουν μη αντιφατικούς κανόνες πάνω στα δεδομένα[2]”

1.2.1 Χτίζοντας ένα Δένδρο Απόφασης

Η διαδικασία που θα ακολουθήσουμε εδώ θυμίζει τη μέθοδο «διαίρει και βασίλευε»[5], ή εν πάση περιπτώσει τις μεθόδους top-down, που είναι αρκετά διαδεδομένες στους αλγόριθμους αναζήτησης αλλά και ταξινόμησης(binary search & merge sort p.e.). Εδώ βέβαια, αυτό που «απασχολεί» έναν αλγόριθμο μηχανικής μάθησης που «παράγει» προβλέψεις από δένδρα απόφασης είναι η ιεραρχία των attributes, από τη ρίζα μέχρι τα τελικά υποδένδρα., δηλαδή, ποιο attribute είναι το καταλληλότερο ως ρίζα. Τα βήματα είναι τα εξής:

- Επιλέγουμε ένα attribute για ρίζα του δένδρου
  - δημιουργούμε μια ακμή για κάθε πιθανή τιμή του attribute
- Χωρίζουμε τα στιγμιότυπα σε υποδένδρα(ή subsets)
  - ένα για κάθε ακμή που εκτείνεται από τον κόμβο
- Επαναλαμβάνουμε για κάθε ακμή
  - χρησιμοποιούμε στιγμιότυπα τα οποία σχετίζονται με την ακμή
- Σταματάμε όταν όλα τα στιγμιότυπα έχουν την ίδια κλάση[6]

Στην ερώτηση «ποιο attribute να επιλέξουμε ως ρίζα;», η απάντηση είναι απλή:
Στοχεύουμε στο «μικρότερο» δένδρο!

Επιλέγουμε το attribute που παράγει τους "καθαρότερους" κόμβους δηλ. το μεγαλύτερο κέρδος πληροφορίας. Αν για παράδειγμα, καταφέρουμε οι απαντήσεις να είναι χωρισμένες μόνο σε «ναι» και «όχι» χωρίς να παρεμβάλλονται μεταξύ τους, τότε έχουμε κατασκευάσει ένα επιθυμητό δένδρο.

1.2.2 Decision Trees with Weka

Από τα datasets του Weka, θα δουλέψουμε πάνω στο αρχείο weather.nominal.arff. Το συγκεκριμένο σετ έχει αυτή τη μορφή:

[5] Θυμίζουμε ότι τα δένδρα απόφασης ΔΕΝ είναι πάντοτε δυαδικά, επομένως το divide and conquer δεν μας εξυπηρετεί πάντα ως μέθοδος.

[6] Τα class attributes στο Weka είναι καθοριστικής σημασίας για τους αλγόριθμους δένδρων σε αντίθεση με το KNN.

```
@relation weather.symbolic

@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no
```

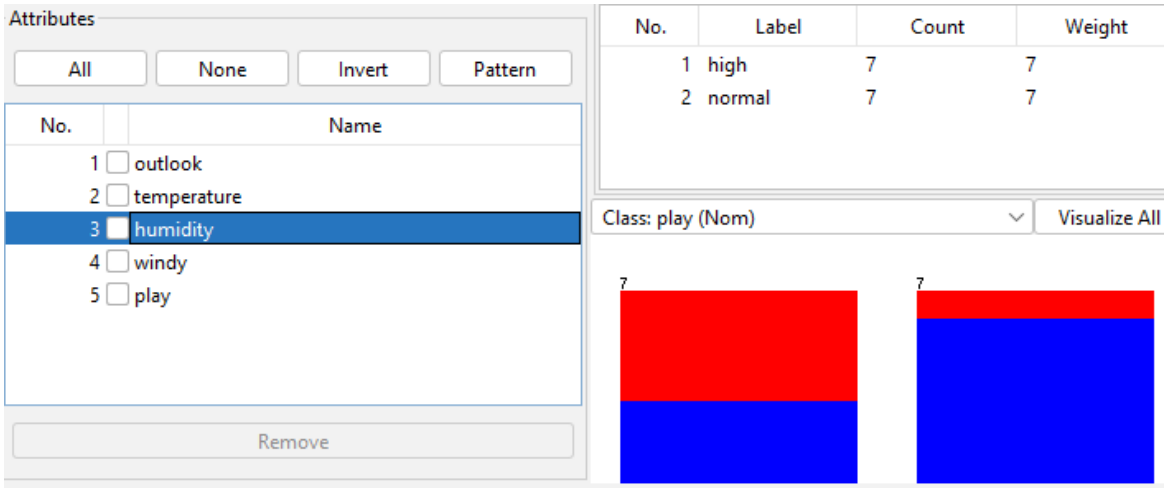
όπου όλες οι μεταβλητές είναι nominal(ονομαστικές) και ο σκοπός είναι να κατασκευαστεί ένα Δένδρο απόφασης, το οποίο θα προβλέπει ανάλογα με τις καιρικές συνθήκες, εάν θα καταλήξει κάποιος σε παιγνίδι ή όχι.

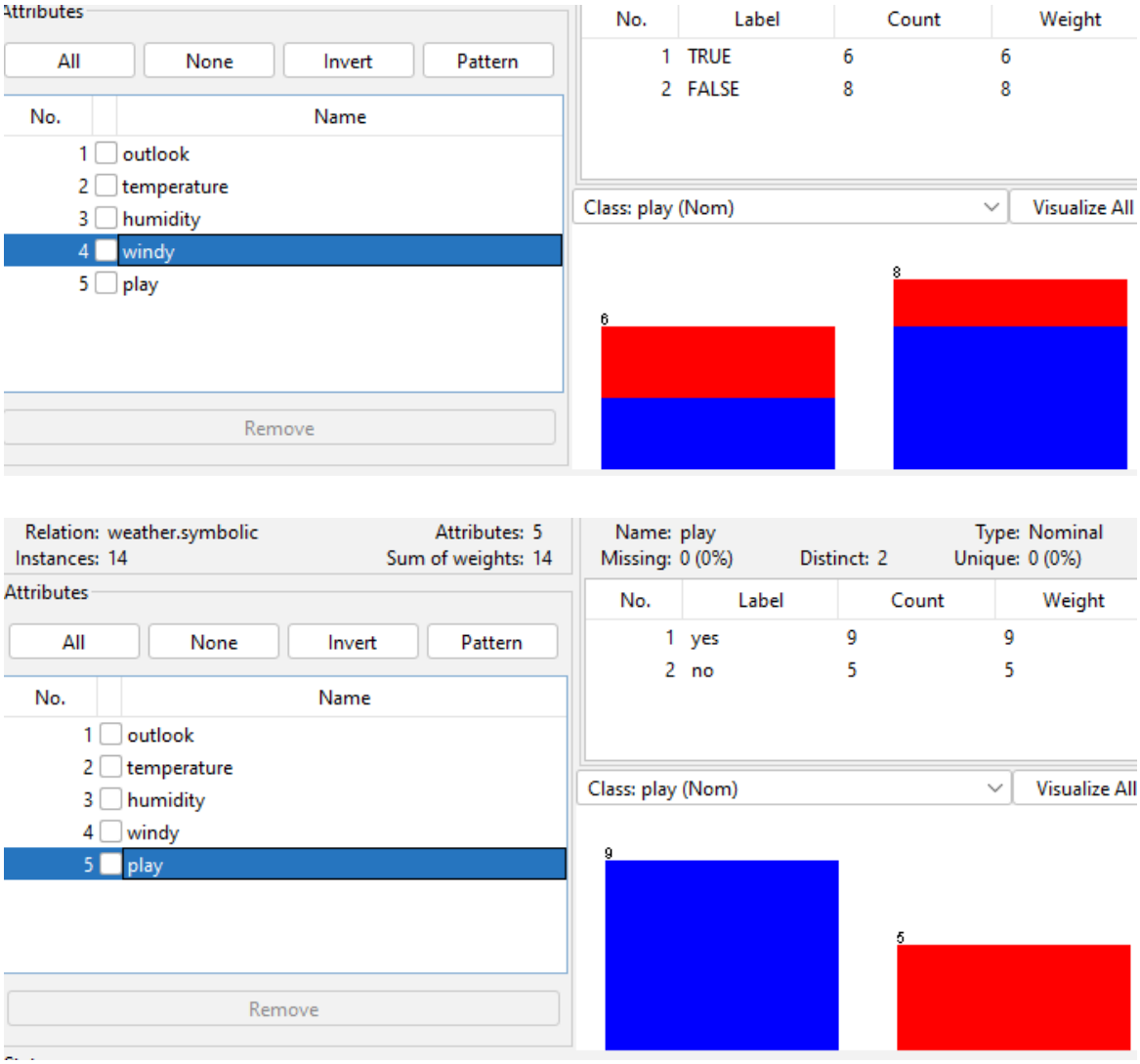
Τα attributes είναι outlook(πως είναι ο καιρός στην όψη) με πιθανές τιμές {ηλιόλουστη, συννεφιασμένη, βροχερή}, temperature(θερμοκρασία) με πιθανές τιμές {θερμή, ήπια, δροσερή}, humidity(υγρασία) με τιμές {υψηλή, κανονική}, windy(αν υπάρχει έντονος άνεμος) με τιμές {true, false} και παιγνίδι με τιμές ναι και όχι. Η τελευταία μεταβλητή-ιδιότητα, όπως θα δούμε στο weka είναι η ιδιότητα κλάσης. Θα πούμε εν συντομία εδώ ότι είναι στην ουσία η απόφαση που πρέπει να καταλήγει κάθε φύλλο του δένδρου(play or don’t play) ανάλογα με τις καιρικές συνθήκες.

Ας δούμε το σετ στο λογισμικό, απ όπου θα βγάλουμε χρήσιμα συμπεράσματα για την «κατανομή» των στιγμιότυπων, προτού δούμε τους αλγόριθμους και τους σχεδιασμούς των δένδρων, όπου το weka μας παρέχει και γραφική απεικόνιση.


Current relation
Relation: weather.symbolic
Instances: 14
Attributes: 5
Sum of weights: 14
Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
Selected attribute
Name: outlook
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)
No.
Label
Count
Weight
1 sunny 5 5
2 overcast 4 4
3 rainy 5 5
Class: play (Nom)
Visualize All
5
4
5


Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
No.
Label
Count
Weight
1 hot 4 4
2 mild 6 6
3 cool 4 4
Class: play (Nom)
Visualize All
4
6
4


Attributes
All
None
Invert
Pattern
No.
Name
1 outlook
2 temperature
3 humidity
4 windy
5 play
Remove
No.
Label
Count
Weight
1 high 7 7
2 normal 7 7
Class: play (Nom)
Visualize All
7
7



Αυτά είναι συγκεντρωτικά τα attributes με τις τιμές σε κάθε στιγμιότυπο.

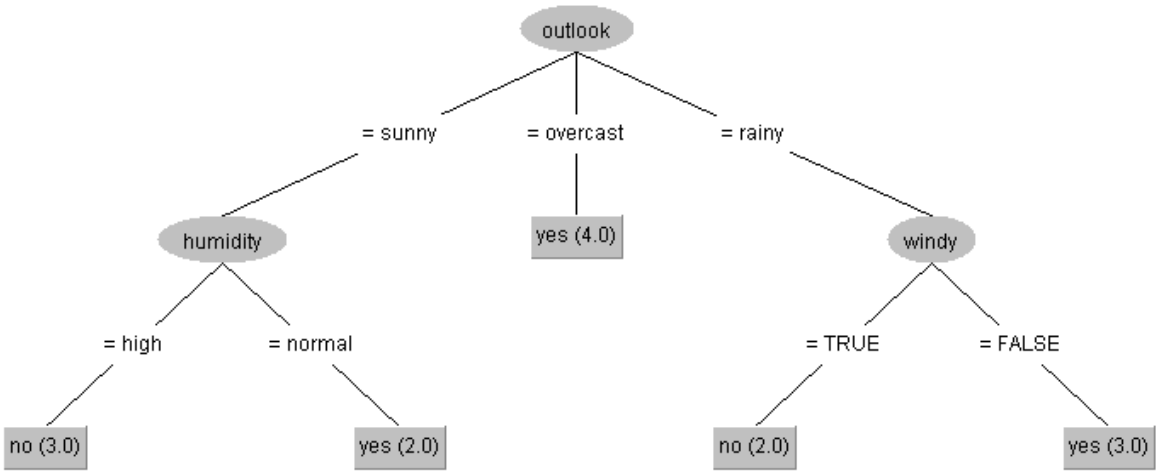
Από τις επιλογές για τους αλγόριθμους κατηγοριοποίησης, επιλέγουμε στα trees τον αλγόριθμο J48. Ας δούμε τα αποτελέσματα:

```
J48 pruned tree
------------------

outlook = sunny
|   humidity = high: no (3.0)
|   humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
|   windy = TRUE: no (2.0)
|   windy = FALSE: yes (3.0)

Number of Leaves  :     5

Size of the tree :      8
```

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           7               50      %
Incorrectly Classified Instances         7               50      %
Kappa statistic                         -0.0426
Mean absolute error                      0.4167
Root mean squared error                  0.5984
Relative absolute error                 87.5    %
Root relative squared error            121.2987 %
Total Number of Instances               14
```

Είναι φανερό ότι δεν είναι και η καλύτερη επιλογή καθώς η ακρίβεια είναι στο 50% και τα errors σε πολύ υψηλά ποσοστά.

Από την επιλογή visualize tree παίρνουμε το δένδρο της εικόνας:



Εικόνα 1.13 J48 algorithm Tree

Παρατηρούμε ότι ως ρίζα του δένδρου επιλέχθηκε το outlook, ως παιδιά(όχι φύλλα) τα humidity και windy. Το δένδρο δεν είναι δυαδικό, αποτελείται από 8 κόμβους εκ των οποίων οι 5 είναι φύλα. Ο σκοπός είναι να φθάνουμε σε μια «καθαρή» απόφαση, ναι ή όχι, και δεδομένου ότι το δένδρο είναι μικρό σε μέγεθος, ο σκοπός έχει επιτευχθεί, όχι βέβαια με μεγάλη ακρίβεια.

1.2.3 Συμπεράσματα

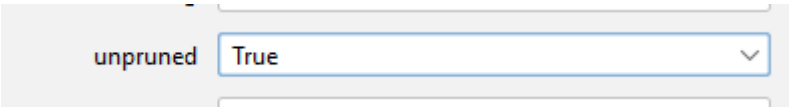
Ο J48, είναι ο πιο διαδεδομένος αλγόριθμος δένδρων στο weka, αρκετά απλός σε σχέση με άλλους αλγόριθμους πιο πολύπλοκους. Στα θετικά είναι ότι παράγει δένδρα τα οποία είναι κατανοητά από την πλειοψηφία, ότι βασίζεται στην θεωρία πληροφοριών και προσφέρει διάφορα κριτήρια για την επιλογή των attributes. Στα αρνητικά, όπως φάνηκε είναι πως η απλότητα αυτή δεν μας οδηγεί σε accurate predictions, και η αλήθεια είναι πως χρειάζεται περεταίρω τροποποιήσεις ώστε να γίνει χρήσιμος στην πράξη.

Οι μέθοδοι αυτές αφορούν κυρίως στο λεγόμενο «κλάδεμα»(pruning), κάτι το οποίο μας επιτρέπει να κάνουμε το Weka στον J48. Στο παραπάνω παράδειγμα αν θέσουμε το pruning στο off έχουμε μια ελάχιστα καλύτερη ακρίβεια.

Αν ανοίξουμε το αρχείο breast-cancer.arff και δοκιμάσουμε τον ίδιο αλγόριθμο τότε έχουμε:

```
Correctly Classified Instances         216               75.5245 %
Incorrectly Classified Instances        70               24.4755 %
Kappa statistic                          0.2826
Mean absolute error                      0.3676
Root mean squared error                  0.4324
Relative absolute error                 87.8635 %
Root relative squared error             94.6093 %
Total Number of Instances              286
```

και ένα δένδρο με 6 συνολικά κόμβους. Αν θέσουμε



τότε έχουμε μείωση στο 69%(όχι κάτι τρομερό) αλλά ένα δένδρο με 179 κόμβους!!! στο σύνολο, κάτι που κάνει τον αλγόριθμο αργό και το δένδρο δυσνόητο.

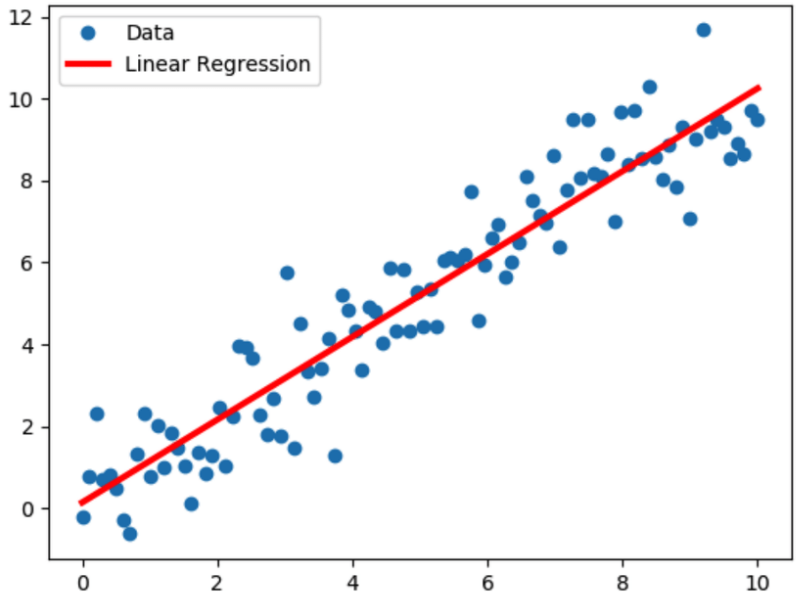
1.3 Linear Regression

Μέχρι τώρα, τα δεδομένα τα οποία θέλαμε να κατηγοριοποιήσουμε στο σύνολο του dataset, ήταν αριθμητικά και ονομαστικά(numeric & nominal) attributes. Τα class attributes όμως ήταν πάντοτε ονομαστικά. Το αποτέλεσμα δηλαδή, ήταν πάντα «ναι ή όχι» (ένας τύπος γυαλιού στο παράδειγμα με το glass.arff κλπ.). Σε αυτό το κεφάλαιο θα μελετηθεί η περίπτωση όπου έχουμε numeric classes κι επομένως η έξοδος του αλγόριθμου θα είναι αριθμητική[7].

1.3.1 Τι είναι η γραμμική παλινδρόμηση

Τα γραμμικά μοντέλα βασίζονται στη γραμμική άλγεβρα, η οποία υπάρχει εδώ και αιώνες, μέσω αλγεβρικών τύπων και θεωρημάτων, και όσον αφορά τη μηχανική μάθηση, υπάρχει μια γραμμική εξάρτηση της εξόδου με τα δεδομένα εισόδου. Επίσης, το μοντέλο είναι απλό και μπορεί σχετικά εύκολα να εκπαιδευτεί, τα υπολογισμένα βάρη σε ένα γραμμικό άθροισμα μας παρέχουν μια άμεση εξήγηση της σημασίας των διαφόρων attributes. Όσο μεγαλύτερη είναι η απόλυτη τιμή του βάρους, τόσο μεγαλύτερη είναι η επίδραση του αντίστοιχου χαρακτηριστικού[8].

Ας δούμε αρχικά ένα θεωρητικό παράδειγμα, σε μορφή διαγραμματική:



1.14 Linear Regression Graph

Πηγή: https://www.researchgate.net/

Στην εικόνα, οι άξονες αντιπροσωπεύουν τιμές, για παράδειγμα ποσότητα – τιμή. Ένα παράδειγμα θα μπορούσε να είναι τα megapixel μιας φωτογραφικής μηχανής σε σχέση με την τιμή της. Όσο περισσότερα τόσο μεγαλύτερη η τιμή.

[7] Linear Regression σε μετάφραση σημαίνει γραμμική παλινδρόμηση. Στην ουσία πρόκειται για αριθμητική πρόβλεψη.
[8] Σκεφτείτε πόσο επηρεάζει έναν μέσο όρο μια πολύ μεγάλη τιμή

Η γραμμική εξίσωση προκύπτει από το άθροισμα των τιμών των χαρακτηριστικών εισόδου, πολλαπλασιασμένα με το βάρος τους:

$$\mathbf{x = w_0 + w_1 a_1 + w_2 a_2 + \ldots + w_k a_k}$$

Η προβλεπόμενη τιμή για το training set για το στιγμιότυπο $a^{(1)}$

$$\sum_{j=0}^{k} w_j a_j$$

Πρέπει να επιλεχθούν τα βάρη που ελαχιστοποιούν το «τετραγωνικό σφάλμα»(squared error) του training data.

$$\sum_{i=1}^{n} \left(x - \sum_{j=0}^{k} w_j a_j \right)^2$$

1.3.2 Linear Regression with Weka

Ως παράδειγμα για το μοντέλο αυτό θα χρησιμοποιηθεί το αρχείο «cpu.arff», στο οποίο όλες οι τιμές των χαρακτηριστικών είναι αριθμητικές.

```
@relation 'cpu'
@attribute MYCT numeric
@attribute MMIN numeric
@attribute MMAX numeric
@attribute CACH numeric
@attribute CHMIN numeric
@attribute CHMAX numeric
@attribute class numeric
@data
125,256,6000,256,16,128,198
29,8000,32000,32,8,32,269
29,8000,32000,32,8,32,220
29,8000,32000,32,8,32,172
29,8000,16000,32,8,16,132
26,8000,32000,64,8,32,318
23,16000,32000,64,16,32,367
23,16000,32000,64,16,32,489
23,16000,64000,64,16,32,636
23,32000,64000,128,32,64,1144
400,1000,3000,0,1,2,38
400,512,3500,4,1,6,40
```

Από το μενού Classify πάλι, επιλέγουμε functions και Linear Regression. Κάνουμε κλικ στο Start και έχουμε το αποτέλεσμα της εικόνας:

```
Linear Regression Model

class =

      0.0491 * MYCT +
      0.0152 * MMIN +
      0.0056 * MMAX +
      0.6298 * CACH +
      1.4599 * CHMAX +
    -56.075

Time taken to build model: 0.43 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.9012
Mean absolute error                     41.0886
Root mean squared error                 69.556
Relative absolute error                 42.6943 %
Root relative squared error             43.2421 %
Total Number of Instances              209
```

Το πρώτο που μας ενδιαφέρει από αυτή την εικόνα είναι η κλάση. Η κλάση έχει προβλεφθεί ως γραμμικό άθροισμα. Οι αριθμοί 0.0491, 0.0152 κλπ είναι τα βάρη των attributes όπως αναφέρθηκαν στην εξίσωση πιο πάνω. Το 56.075 είναι το w_0 , το οποίο δεν τροποποιείται από κάποιο χαρακτηριστικό.
Αυτή είναι μια φόρμουλα υπολογισμού της κλάσης, και όπως βλέπουμε στα αποτελέσματα δεν είναι διόλου άσχημη. Ο συντελεστής συσχέτισης είναι στο 0.9012. Το mean absolute error και το root mean squared error, είναι περίπου στα ίδια επίπεδα με τους προηγούμενους αλγόριθμους που χρησιμοποιήθηκαν στο κεφάλαιο μέχρι στιγμής.

1.3.3 Συμπεράσματα

Στα προβλήματα που προκύπτουν με την τεχνική αυτή είναι τα βασικά προβλήματα των μητρών(matrix problem). Δουλεύει πολύ καλύτερα όταν έχουμε περισσότερα στιγμιότυπα από attributes.

Για τα nominal attributes, πρέπει να τα μετατρέψουμε σε numeric, όπου π.χ. το «ναι» και το «όχι» ή το «true» και «false» θα μετατραπούν σε 0 και 1.

Στα συν είναι το γεγονός ότι υπάρχουν πολλά γραμμικά μοντέλα που μπορούμε να πειραματιστούμε ώστε να καταλήξουμε στο ιδανικό.

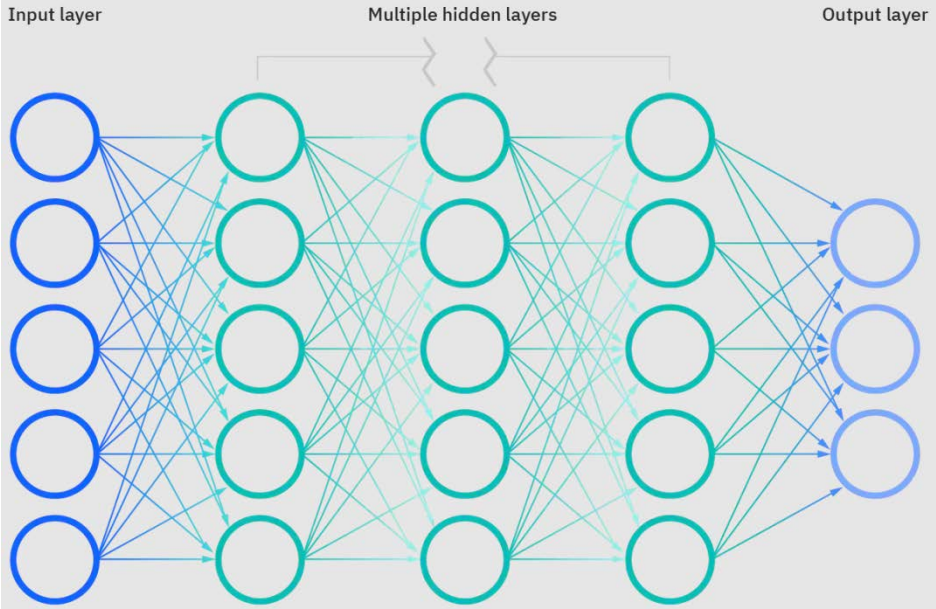
1.4 Neural Networks

Τα Neural Networks ή Νευρωνικά Δίκτυα, είναι ίσως τα πιο διαδεδομένα σε ό,τι αφορά την τεχνητή νοημοσύνη και μάλλον ο λόγος που ισχύει αυτό είναι οι εικόνες 1.15 και 1.16. Στην πρώτη απεικονίζεται στο εσωτερικό του ανθρώπινου εγκεφάλου, ένα δίκτυο με «καλωδιακή τοπολογία», που προφανώς υποδηλώνει πως οι νευρώνες του εγκεφάλου μας επικοινωνούν μέσω ηλεκτρικών σημάτων και μέσω της επικοινωνίας αυτής λαμβάνονται κάποιες αποφάσεις, εκτελούνται κάποιες ενέργειες και λοιπά. Στον εγκέφαλό μας άλλωστε υπάρχουν και οι βιολογικοί μας αισθητήρες, της όρασης, της όσφρησης, της ακοής, της γεύσης και της αφής. Αν δούμε τον άνθρωπο ως μηχανή, τότε π.χ. το αισθητήριο όργανο της όρασης είναι τα μάτια, αλλά ο υπολογιστής που αναλύει και επεξεργάζεται τα δεδομένα εισόδου είναι ο εγκέφαλος και όλες αυτές οι πληροφορίες «ταξιδεύουν» μέσω των νευρώνων σχηματίζοντας ένα δίκτυο.

1.15 Human Mind as a Neural Network

Πηγή: https://www.lifewire.com/what-is-a-neural-network-5181580

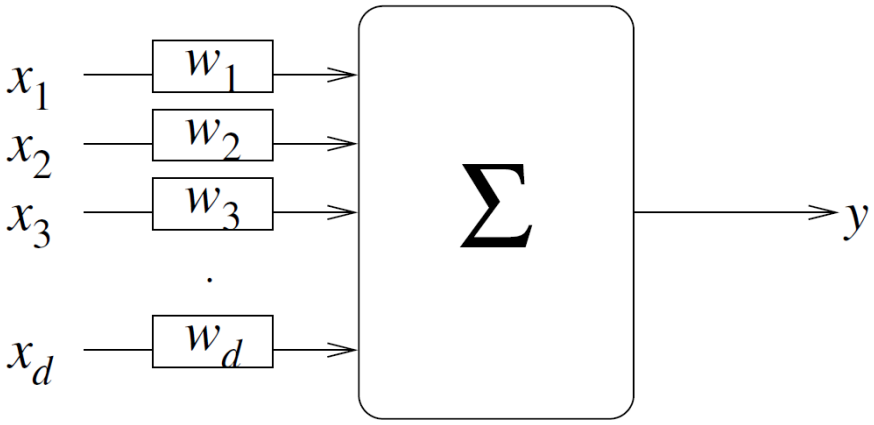
Στην εικόνα 1.16 έχουμε μια γραφική αναπαράσταση νευρωνικού δικτύου, όπως θα μελετηθεί στην ενότητα αυτή. Θα μπορούσαμε να πούμε πως ισχύει αυτό που ειπώθηκε όσον αφορά τη φήμη των νευρωνικών δικτύων μιας και η εικόνα 1.16 μοιάζει εκ πρώτης όψεως πολύπλοκη που γενικά το πολύπλοκο μας δίνει την εντύπωση του δύσκολου, άρα και του εντυπωσιακού, ίσως. Ένα σχήμα με κόμβους και πολλές ακμές που τους ενώνουν, δίνοντας την εικόνα ενός επικοινωνιακού δικτύου σαν κουβάρι, μπορεί αρχικά να περιπλέκει τα πράγματα αλλά μάλλον δεν είναι καθόλου πολύπλοκο.



1.16 Neural Network Graph – "Perceptron"

Πηγή: https://www.ibm.com/cloud/learn/neural-networks

Στην εικόνα 1.16 υπάρχουν τρεις ετικέτες, η Input Layer, η Multiple Hidden Layers και η Output Layer. Αν δηλαδή έχουμε στους αριστερά κόμβους τα δεδομένα εισόδου, στο κέντρο κάποια «κρυμμένα» επίπεδα που σχετίζονται με την επεξεργασία των δεδομένων εισόδου ώστε να πάμε στο δεξιά επίπεδο που είναι η έξοδος. Από κόμβους δηλαδή, οδηγούμαστε με βέλη σε άλλους κόμβους μέχρι που καταλήγουμε στην έξοδο. Άρα, το παραπάνω σχήμα δεν είναι τίποτε παραπάνω από ένα Γραμμικό Μοντέλο, μια Γραμμική Εξίσωση ή μια Γραμμική Συνάρτηση. Οι γραμμικές αυτές συναρτήσεις, όπως αυτή της εικόνας 1.17, ονομάζονται "Perceptrons" για ιστορικούς λόγους[2].



1.17 The Perceptron

Πηγή: The Lion Way [1]

1.4.1 Perceptrons & MLP

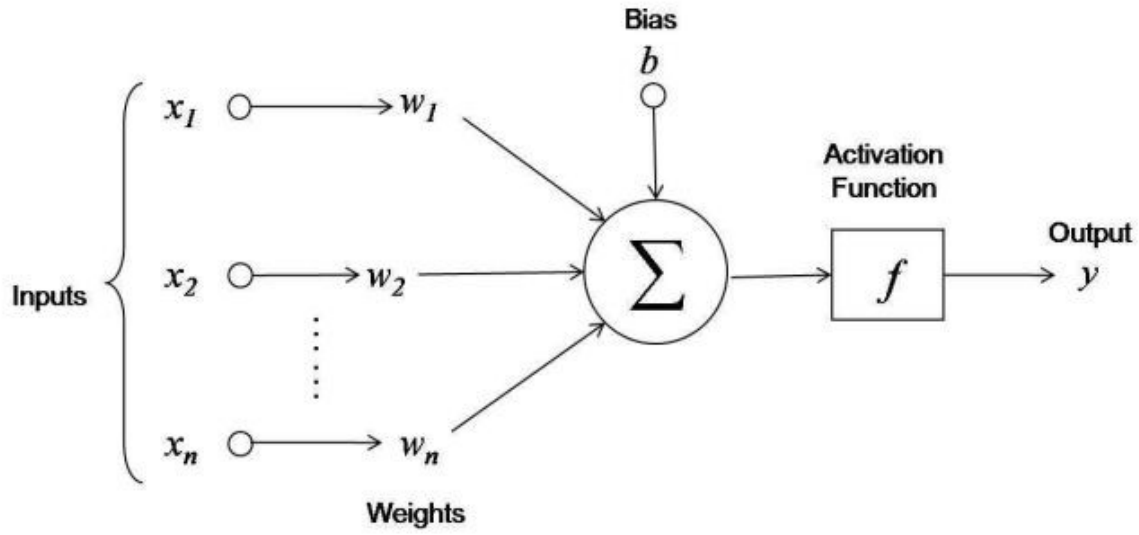
Στην εικόνα 1.17, βλέπουμε μια γραμμική συνάρτηση με d εισόδους ως μεταβλητές x, όπου με όρους machine learning είναι τα attributes ενός Dataset, τα «βάρη» τους ως w, που στην ουσία είναι τα βάρη των ακμών και μια έξοδο y η οποία προκύπτει από το άθροισμα των εισόδων, όπως θα δούμε παρακάτω, που για συντομία να αναφέρουμε ότι προκύπτει από έναν γραμμικό συνδυασμό των attributes ο οποίος θα καθορίσει και την κατηγοριοποίηση τους ως Classifier. Πίσω στην εικόνα **1.16**, υπάρχουν τα 3 Layers, Input, Multiple Hidden Layers και Output. Αυτά τα Layers διαφέρουν σε αριθμό ανάλογα με τα δεδομένα και την υλοποίηση και σχηματίζουν ένα «μοντέλο» που ονομάζεται «multilayer perceptron neural network»(νευρωνικό δίκτυο Perceptron πολλαπλών στρωμάτων) ή MLP. Οι ακμές που ενώνουν τους κόμβους του κάθε επιπέδου έχουν έναν αριθμό που είναι το βάρος(weight) της σύνδεσης. Σε κάθε κόμβο από το $2^{ο}$ επίπεδο και μετά υπολογίζεται το άθροισμα των βαρών αυτών. Οι κόμβοι συχνά ονομάζονται και «νευρώνες» (neurons) ή και μονάδες(units). Λόγω του ότι οι κόμβοι αυτοί στην εικόνα είναι παράλληλοι σε κάθε «επίπεδο», συχνά τα νευρωνικά δίκτυα αναφέρονται στη βιβλιογραφία και ως «παράλληλη κατανεμημένη επεξεργασία»(parallel distributed processing). Το συγκεκριμένο δίκτυο, αυτό δηλαδή στο οποίο δεν προκύπτουν κυκλικές πορείες μέσω των ακμών του, ανήκει στην κατηγορία δικτύων με «προς τα εμπρός τροφοδότηση»(feed-forward networks).

Κάθε μονάδα ή νευρώνας υπολογίζει ένα «σταθμισμένο άθροισμα» (weighted summary) των εισόδων του. Αυτό το άθροισμα δίνεται από τον τύπο:

$$\mathbf{x = w_0 + w_1 a_1 + w_2 a_2 + \ldots + w_n a_n =}$$

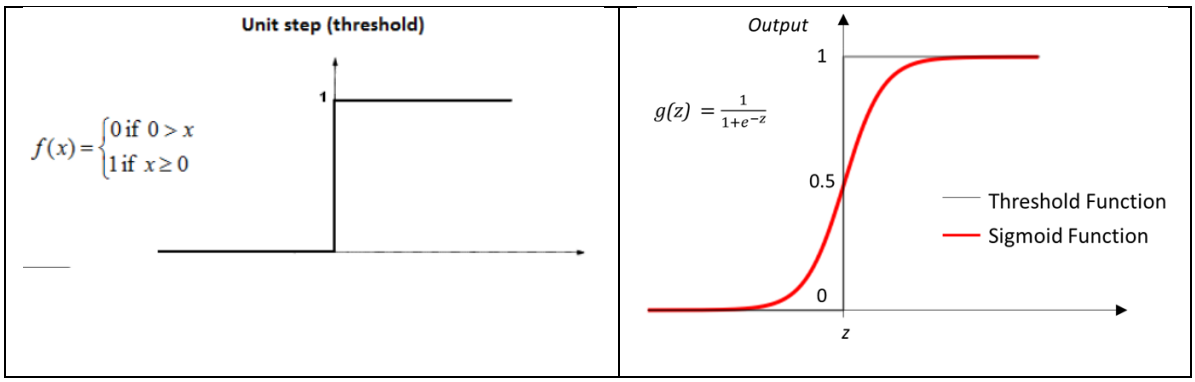
$$\sum_{i=0}^{n} w_i a_i$$

όπου a είναι ένα attribute του στιγμιότυπου και w το βάρος του. Το $\mathbf{w_0}$ έχει έναν ειδικό ρόλο και ονομάζεται «bias weight» ή βάρος πόλωσης και συνδέεται με μια σταθερή είσοδο $\mathbf{a_0 = 0}$. Για ένα a instance από τον παραπάνω υπολογισμό οδηγούμαστε σε classification με τον εξής τρόπο: Αν το x > 0 τότε κατηγοριοποιείται στην class 1, αν x < 0 στην class 2. Το παραπάνω δίκτυο μπορεί να περιγραφεί και ως νευρωνικό δίκτυο ενός επιπέδου (simple neural network ή **single-layer** neural network) όπως στην εικόνα 1.18. Στην εικόνα αυτή υπάρχει και μια «συνάρτηση ενεργοποίησης» η οποία εφαρμόζεται στο άθροισμα με σκοπό να παραχθεί η έξοδος. Η συνάρτηση αυτή είναι σχεδιασμένη έτσι ώστε να καλύπτει δυο περιπτώσεις. Λέμε ότι είναι «ενεργή» στο +1 όταν οι είσοδοι είναι «σωστές» και ανενεργή στο 0 όταν είναι «λανθασμένες» (για την ακρίβεια κοντά στο +1 ή κοντά στο 0). Υπάρχουν 2 επιλογές για τη συνάρτηση αυτή όπως παρουσιάζονται στην εικόνα 1.19: η συνάρτηση κάτω ορίου(κατωφλίου) **threshold** και η «σιγμοειδής» συνάρτηση, **sigmoid function.**



1.18 Single-layer NN

Πηγή: https://towardsdatascience.com/



1.19 Threshold vs Sigmoid function

Πηγή: https://morioh.com/p/576f179d5985

Όπως φαίνεται και από τα δυο σχήματα, οι συναρτήσεις είναι γραμμικές σε κάθε περίπτωση και η λογική των νευρωνικών δικτύων δεν απέχει υπερβολικά από αυτή της γραμμικής παλινδρόμησης της προηγούμενης ενότητας. Προτού προχωρήσουμε στην παρουσίασή τους από το Weka, να αναφέρουμε πως αν οι νευρώνες στο Hidden Layers είναι περισσότεροι από ένα ώστε να σχηματίζονται επίπεδα, τότε μιλάμε για **Multi-Layered** neural network και όχι simple. Κάθε μονάδα θα λαμβάνει είσοδο μόνο από μονάδες του αμέσως προηγούμενου επιπέδου. Στα δίκτυα ενός επιπέδου συχνά δεν ονομάζονται κρυφές οι μονάδες αυτές ενώ στα Multi-layered και ονομάζονται κρυφές επειδή δεν αποτελούν μέρος της τελικής εξόδου [3].

1.4.2 Neural Networks with weka

Στο Weka, ο αλγόριθμος – classifier που θα χρησιμοποιήσουμε πρώτα είναι ο Voted Perceptron Algorithm ο οποίος βασίζεται στον αλγόριθμο των Freund και Schapire ο οποίος με τη σειρά του στηρίζεται στα βάρη που αναφέραμε πιο πάνω. Τα βάρη αυτά προέρχονται από αυτόν τον αλγόριθμο ο οποίος έχει τα εξής βήματα:

1. Θέτουμε όλα τα βάρη ίσα με μηδέν ($w_0 = w_1 = w_2 = \ldots = w_n = 0$)
2. Μέχρι να κατηγοριοποιηθούν(classify) σωστά όλα τα στιγμιότυπα του training set
    3. Για κάθε i-οστό στιγμιότυπο του training set
        Αν κατηγοριοποιείται εσφαλμένα τότε
            Αν ανήκει στην class 1 πρόσθεσέ το στο «διάνυσμα βαρών»
            Αλλιώς αφαίρεσέ το απ' το διάνυσμα[9]

Από το explorer του Weka ανοίγουμε το αρχείο diabetes, το οποίο προέρχεται από σχετική έρευνα για τον διαβήτη. Τα attributes του είναι αυτά της επόμενης εικόνας, τα οποία είναι όλα numeric εκτός του class:

```
%    1. Number of times pregnant
%    2. Plasma glucose concentration a 2 hours in an oral
glucose tolerance test
%    3. Diastolic blood pressure (mm Hg)
%    4. Triceps skin fold thickness (mm)
%    5. 2-Hour serum insulin (mu U/ml)
%    6. Body mass index (weight in kg/(height in m)^2)
%    7. Diabetes pedigree function
%    8. Age (years)
%    9. Class variable (0 or 1)
```

1.20 Diabetes DataSet Attributes

Πάλι από την επιλογή Classify -> Functions -> VotedPerceptron και μετά Start:

Correctly Classified Instances	513	66.7969 %
Incorrectly Classified Instances	255	33.2031 %
Kappa statistic	0.1353	
Mean absolute error	0.3319	
Root mean squared error	0.5752	
Relative absolute error	73.0209 %	
Root relative squared error	120.6751 %	
Total Number of Instances	768	

1.21 diabetes - VotedPerceptron

[9] Ο όρος διάνυσμα ή Vector προτιμάται διότι οι ακμές στο σχήμα είναι κατευθυνόμενες.

Στο documentation του Weka [2], αναφέρεται πως ο αλγόριθμος αυτός υπολογίζει τα διανύσματα βαρών και τα αποθηκεύει. Τα διανύσματα αυτά πρέπει να «ψηφίσουν» από τα test examples και το βάρος τους αλλάζει με βάση το «χρόνο επιβίωσης» στη διαδικασία αυτή.

Στην εικόνα 1.21 βλέπουμε αρκετά μη-έγκυρα κατηγοριοποιημένα στιγμιότυπα. Αν δοκιμάσουμε με κάποιο άλλο dataset όπως το breast-cancer το οποίο ανήκει και στην ίδια κατηγορία, υπό την έννοια ότι προέρχεται από ιατρική στατιστική μελέτη τότε θα πάρουμε ένα ποσοστό ακρίβειας της τάξεως του 71.3%. Στην εικόνα 1.22 βλέπουμε κάτι που δεν αναφέρθηκε στην 1.21 και έχει να κάνει με τον αριθμό των perceptrons. Για το breast-cancer dataset βλέπουμε ότι είναι 117 στον αριθμό ενώ στο diabetes είναι 330. Αυτό προφανώς σημαίνει ότι είναι ποσό ανάλογο με το πόσα instances έχει το dataset, 286 στο breast-cancer ενώ στο diabetes 768. Επίσης, βλέπουμε καλύτερα ποσοστά σε μικρότερο σετ κάτι που μάλλον δεν είναι θετικό.

```
Scheme:       weka.classifiers.functions.VotedPerceptron -I 1 -E 1.0 -S 1 -M 10000
Relation:     breast-cancer
Instances:    286
Attributes:   10
              age
              menopause
              tumor-size
              inv-nodes
              node-caps
              deg-malig
              breast
              breast-quad
              irradiat
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

VotedPerceptron: Number of perceptrons=117

Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         204               71.3287 %
Incorrectly Classified Instances        82               28.6713 %
Kappa statistic                          0.212
Mean absolute error                      0.2848
Root mean squared error                  0.5322
Relative absolute error                 68.0628 %
Root relative squared error            116.4466 %
Total Number of Instances              286
```

1.22 Breast cancer - VotedPerceptron Results

1.4.3 Multilayer Perceptrons with Weka

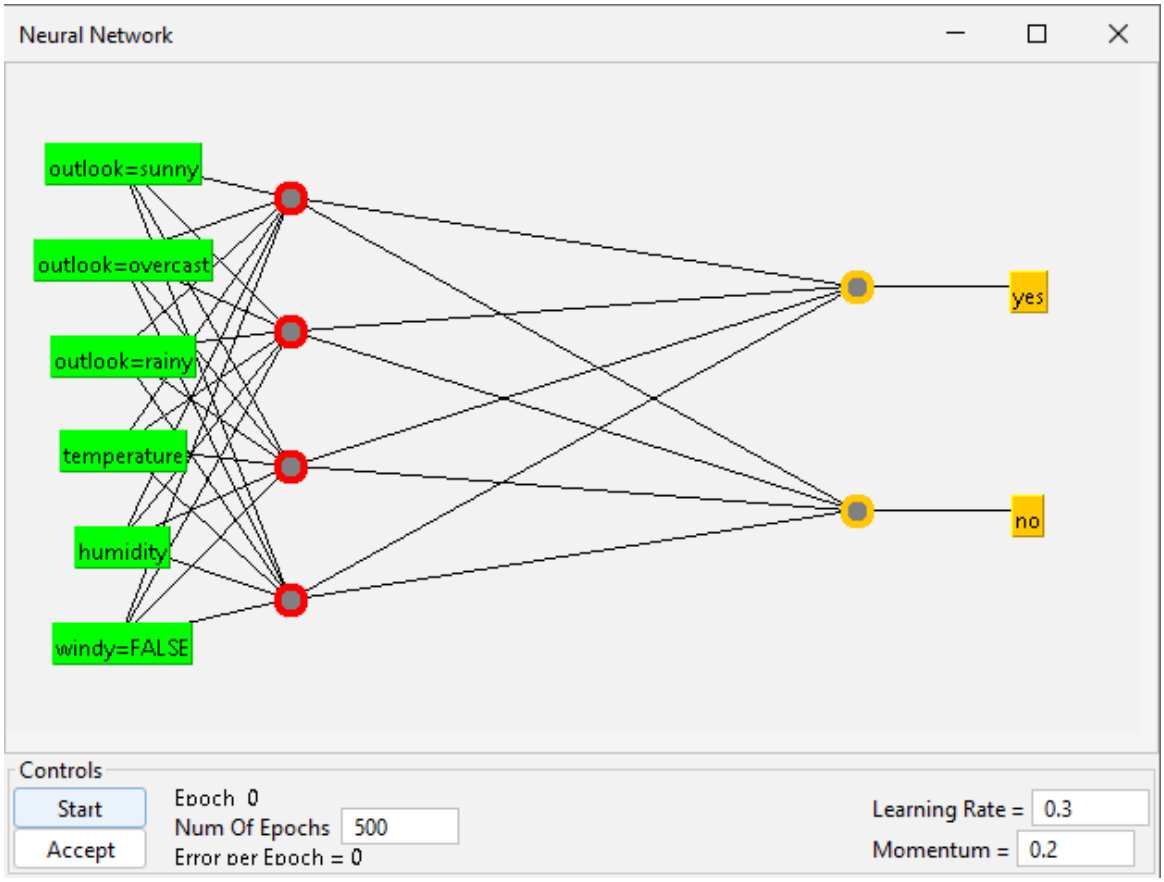
Ο Classifier ο οποίος υλοποιεί τα Perceptrons με πολλά κρυφά επίπεδα όπως στην εικόνα 1.16 βρίσκεται πάλι στην κατηγορία Functions και από εκεί η επιλογή MultilayerPerceptron. Θα επιλέξουμε ξανά το weather-numeric και αρχικά θα τρέξουμε τον classifier με τις προεπιλεγμένες ρυθμίσεις. Το αποτέλεσμα είναι το εξής:

```
Correctly Classified Instances          11               78.5714 %
Incorrectly Classified Instances         3               21.4286 %
Kappa statistic                          0.5116
Mean absolute error                      0.265
Root mean squared error                  0.4627
Relative absolute error                 55.6497 %
Root relative squared error             93.7923 %
Total Number of Instances               14
```

Το ποσοστό είναι στο 78% περίπου, όσο ακριβώς θα παίρναμε και στον nearest neighbors. Πιο πάνω στο output βλέπουμε για κάθε node του τα εξής:

```
Sigmoid Node 0
    Inputs    Weights
    Threshold    -3.2488354416891236
    Node 2    5.706344521860183
    Node 3    2.443270263208691
    Node 4    2.6425576499015655
    Node 5    2.5103414057156117
Sigmoid Node 1
    Inputs    Weights
    Threshold    3.247940047055842
    Node 2    -5.704744057107486
    Node 3    -2.395963544940322
    Node 4    -2.619413415167429
    Node 5    -2.578926745531241
Sigmoid Node 2
    Inputs    Weights
    Threshold    -1.4298110453038173
    Attrib outlook=sunny    1.279607413773088
    Attrib outlook=overcast    2.5993304643376662
    Attrib outlook=rainy    -2.4821894084499005
    Attrib temperature    -0.9917844366897344
    Attrib humidity    -4.1325759725239815
    Attrib windy=FALSE    -0.8030823939514041
Sigmoid Node 3
    Inputs    Weights
    Threshold    -0.7740672340804504
    Attrib outlook=sunny    -1.9100370742566128
    Attrib outlook=overcast    2.382206870768282
    Attrib outlook=rainy    0.23499213125743737
    Attrib temperature    -0.8639638424331714
    Attrib humidity    -0.8117295111072014
    Attrib windy=FALSE    3.092359794678844
```

Συνολικά τα nodes είναι 6. Για να οπτικοποιήσουμε το αποτέλεσμα αυτό, από τις επιλογές του classifier επιλέγουμε στην πρώτη επιλογή, GUI, και από False το αλλάζουμε σε true. Πατάμε στο start και έχουμε αυτό το αποτέλεσμα:

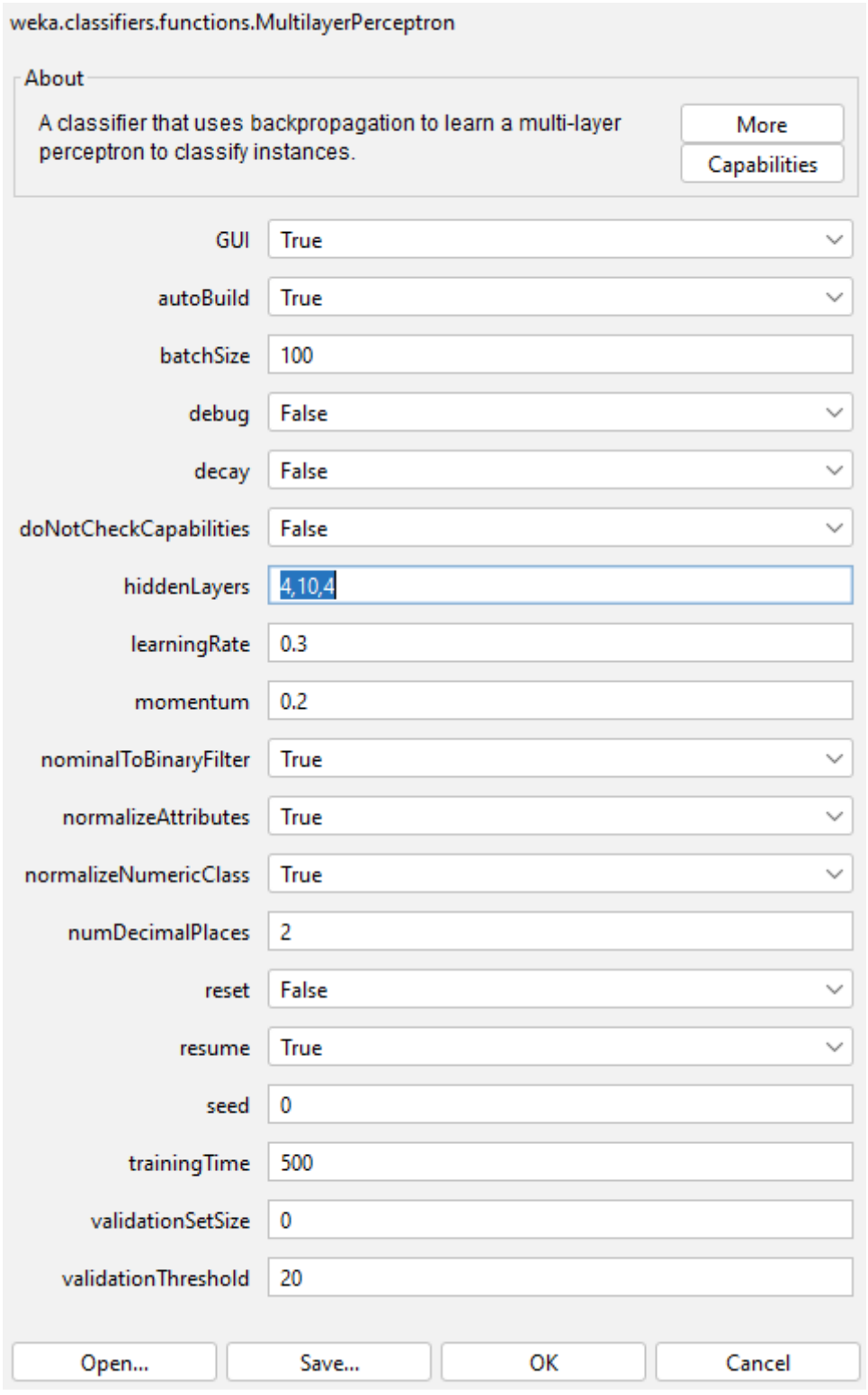


1.23 Neural Network from Weka

Σε αυτή την εικόνα έχουμε ένα επίπεδο hidden layer. Πατώντας το Accept που φαίνεται στην εικόνα 1.23 θα πάρουμε τα αποτελέσματα αυτού του Neural Network Classifier:

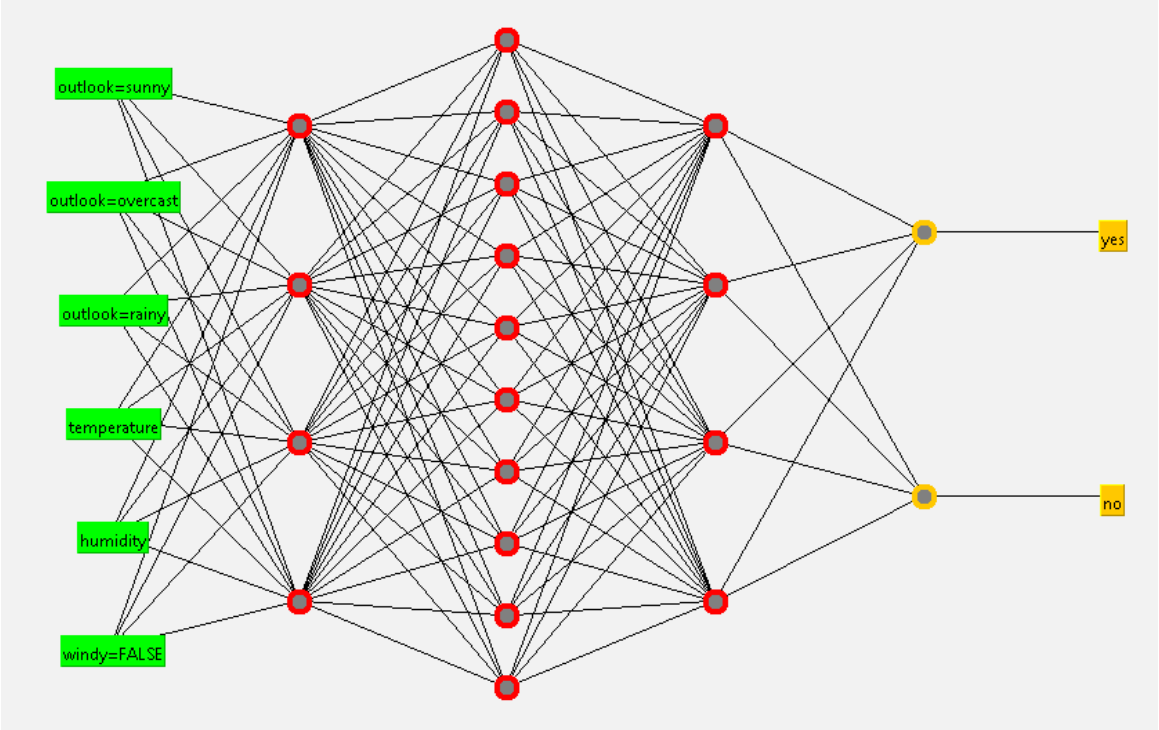
```
=== Summary ===

Correctly Classified Instances           5               35.7143 %
Incorrectly Classified Instances         9               64.2857 %
Kappa statistic                          0
Mean absolute error                      0.5022
Root mean squared error                  0.5022
Relative absolute error                108.1605 %
Root relative squared error            104.7416 %
Total Number of Instances               14
```

Όπως βλέπουμε τα αποτελέσματα δεν είναι τα καλύτερα, άρα ας πειραματιστούμε. Από τις επιλογές του classifier αυτού, μπορούμε να αλλάξουμε τον αριθμό των hidden layers. Μπορούμε μάλιστα να επιλέξουμε πόσα nodes θα έχει το κάθε επίπεδο. Στην επιλογή hiddenLayers, πληκτρολογούμε για δοκιμή 4, 10, 4.



1.24 Multilayer Perceptron options

Με την επιλογή GUI στο True, θα πάρουμε αυτό το γραφικό νευρωνικού δικτύου:



1.25 MultilayerPerceptron 4,10,4

Όσο και να αλλάζουμε τις ρυθμίσεις αυτές τα ποσοστά παραμένουν τα ίδια.

Αν πειραματιστούμε με άλλα datasets τότε θα βρούμε πως οι αλγόριθμοι αυτοί δεν παρουσιάζουν καλύτερα αποτελέσματα από άλλους. Για τον βασικό αλγόριθμο θα λέγαμε ότι τα αποτελέσματα πλησιάζουν σε ποσοστά τον linear regression. Απόλυτα φυσιολογικό μιας και ο τύπος που υπολογίζει το άθροισμα για τα βάρη είναι ακριβώς ο ίδιος. Ο MultilayerPerceptron είναι χειρότερος σε ποσοστό πρόβλεψης σε σχέση με όλους όσους έχουμε δει στην εργασία αυτή αλλά και σε άλλους που δεν έχουν αναφερθεί.

1.4.4 Συμπεράσματα

Τα νευρωνικά δίκτυα αποτελούν μια πολύ καλή προσπάθεια προσομοίωσης των εγκεφαλικών μας νευρώνων. Χρησιμοποιούν μαθηματικές μεθόδους διανυσμάτων που δεν αναφέρθηκαν εκτενώς εδώ (όπως για παράδειγμα το backpropagation(οπισθοδιάδοση) [3]), μηχανισμούς μάθησης για υπολογισμούς βαρών κλπ. Δουλεύουν καλύτερα σε αριθμητικά δεδομένα και σε μερικά σετ όντως τα αποτελέσματα είναι αρκετά ικανοποιητικά. Στα μείον είναι ο χρόνος που απαιτεί ο αλγόριθμος μιας και εκτελεί πολλούς υπολογισμούς για τα βάρη και για να «εκπαιδεύσει» σωστά το training set, καθώς επίσης και στην Multilayer υλοποίηση πρέπει να βρεθεί ο κατάλληλος αριθμός των nodes για να μας δίνει τις πιο σωστές προβλέψεις. Όπως θα αναφερθεί στο τέλος, υπάρχουν πλέον καλύτερες μέθοδοι από αυτή.

2. Μάθηση Χωρίς Επίβλεψη

Σε αυτό το κεφάλαιο θα παραθέσουμε μερικούς σημαντικούς αλγόριθμους αυτής της κατηγορίας της μηχανικής μάθησης αφού πρώτα κάνουμε μερικές αναφορές στους όρους που θα χρησιμοποιηθούν, μιας και ξανά, η μετάφρασή τους στην ελληνική δεν είναι ο καλύτερος τρόπος για να αναφερόμαστε σε αυτές.

- Clustering - cluster είναι το σμήνος, γνωστό από την αστρονομία που υπάρχουν τα star clusters, επομένως το ρήμα clustering θα μπορούσαμε να πούμε ότι είναι η τοποθέτηση σε σμήνος, με κάποια σχετική οριοθέτηση.
- centroid – κέντρο βάρους, ή κεντρικό σημείο βάση ευκλείδειων αποστάσεων

2.1 K-means

Στη σχετική βιβλιογραφία αναφέρονται πέντε τρόποι – μέθοδοι Clustering[10], οι οποίοι δε θα αναλυθούν για λόγους οικονομίας. Ο αλγόριθμος k-means που θα αναλύσουμε εδώ ανήκει στην κατηγορία Prototype-Based, στην οποία ένα cluster είναι ένα σύνολο αντικειμένων όπου κάθε αντικείμενο είναι πιο κοντά(ή πιο όμοιο) με το πρωτότυπο που το «ορίζει» παρά με οποιοδήποτε άλλο πρωτότυπο κάποιου άλλου Cluster. Μια άλλη ονομασία για αυτή την κατηγορία είναι «center-based clusters».

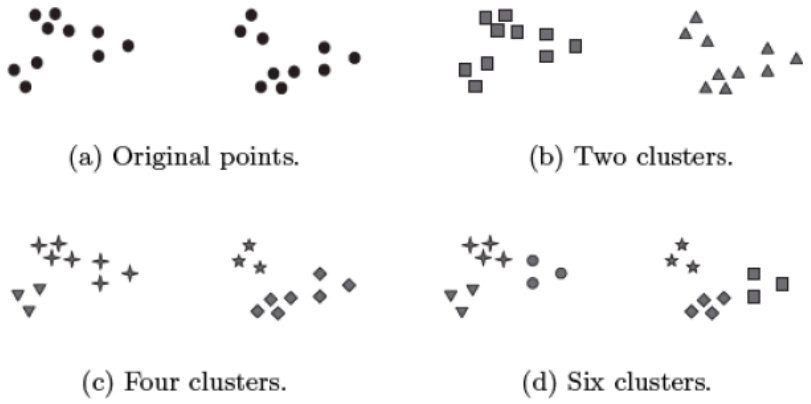
Η πιο απλή υλοποίηση του K-means είναι ο αλγόριθμος Simple K-means, είναι η ίσως πιο απλή υλοποίηση των μεθόδων clustering. Η λογική του είναι η εξής:

- Επιλέγουμε K σημεία ως αρχικά centoids. Ο αριθμός του K ορίζει το πόσα clusters θα δημιουργηθούν και δεν προκαθορίζεται, αντίθετα, επιλέγεται.
- Στη συνέχεια επαναλαμβάνουμε:
    - Σχηματίζουμε K clusters εντάσσοντας κάθε σημείο στο κοντινότερό του centroid.
    - Υπολογίζουμε ξανά τα centroids για κάθε cluster
- Η διαδικασία σταματά όταν τα centroids πάψουν να μεταβάλλονται.

Με λίγα λόγια, κάθε σημείο, θα αντιστοιχηθεί στο cluster του οποίου βρίσκεται πιο κοντά σε αυτό το «κέντρο βάρους». Μετά από κάθε «αντιστοίχιση σημείου» θα πρέπει να ενημερώνεται το κέντρο βάρους του cluster. Αυτό επαναλαμβάνεται μέχρι να ενταχθούν όλα τα σημεία σε clusters και να μην αλλάζουν τα centroids.

[10] Well – Separated, Prototype-Based, Graph – Based, Density – Based, Shared - Property [4]

Να αναφέρουμε εδώ ότι υπάρχουν ομοιότητες με τον αλγόριθμο nearest neighbors, αλλά εδώ δημιουργούμε ένα ομαδοποιημένο σύνολο από unlabeled attributes, ή πιο σωστά προβλέπουμε με τι μοιάζουν περισσότερο και με τι όχι. Στον nearest neighbors προσπαθούσαμε να εντάξουμε ένα νέο χαρακτηριστικό προβλέποντας τι θα είναι. Στον K means ορίζουμε κάποια αρχικά clusters με την έννοια centroid, και στη συνέχεια τα σημεία «ομαδοποιούνται» ανάλογα με την απόσταση και πάλι και διαφοροποιούνται και τα centroids.



2.1 Τέσσερις διαφορετικοί τρόποι Clustering του ίδιου συνόλου σημείων

Πηγή:Introduction to Data Mining [4]

Παρόλο που οι τεχνικές αυτές είναι αρκετά παλιές, χρησιμοποιούνται ακόμα και σήμερα σε δεδομένα όπου μας το «επιτρέπουν». Με τη βοήθεια του λογισμικού θα δούμε πως λειτουργεί ο αλγόριθμος αυτός βήμα – βήμα, πως είναι σχηματικά τα clusters και πότε παίρνουμε πολλά σφάλματα και πότε όχι.

2.1.1 Clustering With Weka

Ας δούμε μερικά παραδείγματα στο Weka ώστε να καταλήξουμε στα συμπεράσματα για τον αλγόριθμο αυτό. Το dataset που θα χρησιμοποιηθεί αρχικά, είναι το γνωστό iris flower. Υπάρχει και στο πασίγνωστο Wikipedia σε αυτή τη μορφή:

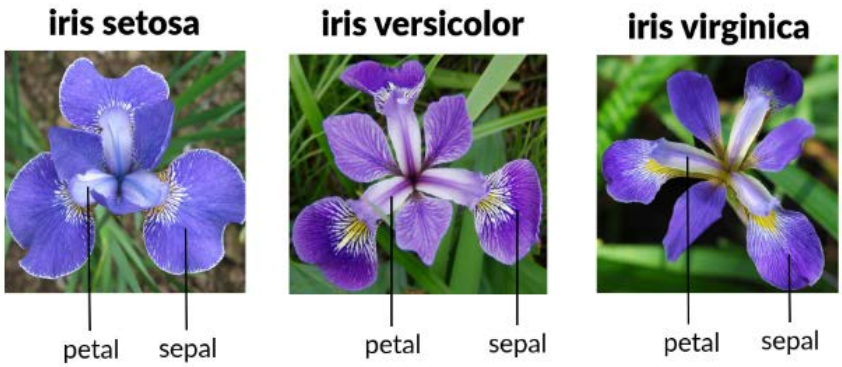
Fisher's *Iris* data [hide]

Dataset order	Sepal length	Sepal width	Petal length	Petal width	Species
1	5.1	3.5	1.4	0.2	*I. setosa*
2	4.9	3.0	1.4	0.2	*I. setosa*
3	4.7	3.2	1.3	0.2	*I. setosa*
4	4.6	3.1	1.5	0.2	*I. setosa*
5	5.0	3.6	1.4	0.3	*I. setosa*
6	5.4	3.9	1.7	0.4	*I. setosa*
7	4.6	3.4	1.4	0.3	*I. setosa*
8	5.0	3.4	1.5	0.2	*I. setosa*
9	4.4	2.9	1.4	0.2	*I. setosa*
10	4.9	3.1	1.5	0.1	*I. setosa*
11	5.4	3.7	1.5	0.2	*I. setosa*
12	4.8	3.4	1.6	0.2	*I. setosa*
13	4.8	3.0	1.4	0.1	*I. setosa*
14	4.3	3.0	1.1	0.1	*I. setosa*
15	5.8	4.0	1.2	0.2	*I. setosa*
16	5.7	4.4	1.5	0.4	*I. setosa*

2.2 Το dataset Iris

Πηγή: https://en.wikipedia.org/wiki/Iris_flower_data_set

Εδώ έχουμε 50 στιγμιότυπα για κάθε ένα από τα 3 είδη. Τα attributes εδώ είναι numerical και εκφράζουν εκατοστά. Αναφέρονται στα χαρακτηριστικά των ανθών του λουλουδιού Iris, μήκος και πλάτος όπως φαίνεται στην εικόνα:



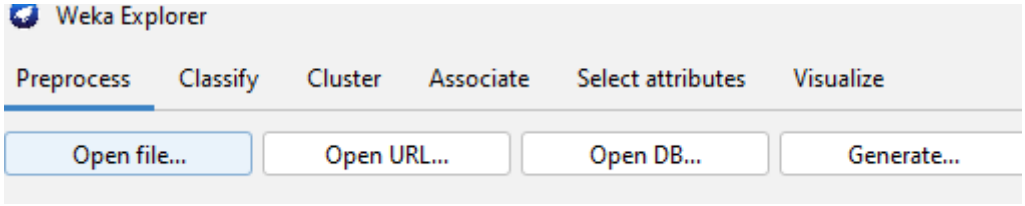
2.3 Iris Flower

Πηγή: https://morioh.com/p/eafb28ccf4e3

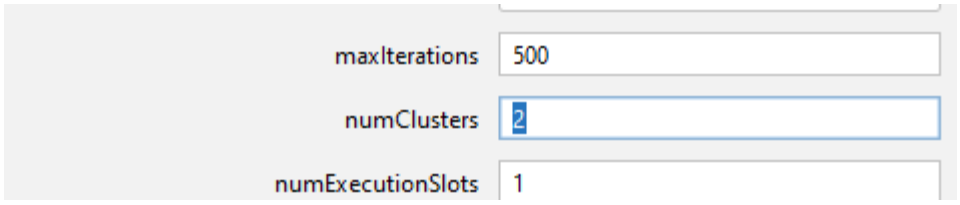
ενώ το αρχείο iris.arff έχει προφανώς αυτή τη μορφή:

```
@RELATION iris

@ATTRIBUTE sepallength    REAL
@ATTRIBUTE sepalwidth     REAL
@ATTRIBUTE petallength    REAL
@ATTRIBUTE petalwidth REAL
@ATTRIBUTE class {Iris-setosa,Iris-
versicolor,Iris-virginica}

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
```

Η διαδικασία είναι η ίδια. Ανοίγουμε το weka, επιλέγουμε το explorer και ανοίγουμε το αρχείο iris.arff μόνο που αυτή τη φορά από το μενού δε θα επιλέξουμε κάποιον classifier από την επιλογή classify, αλλά από την επιλογή cluster:



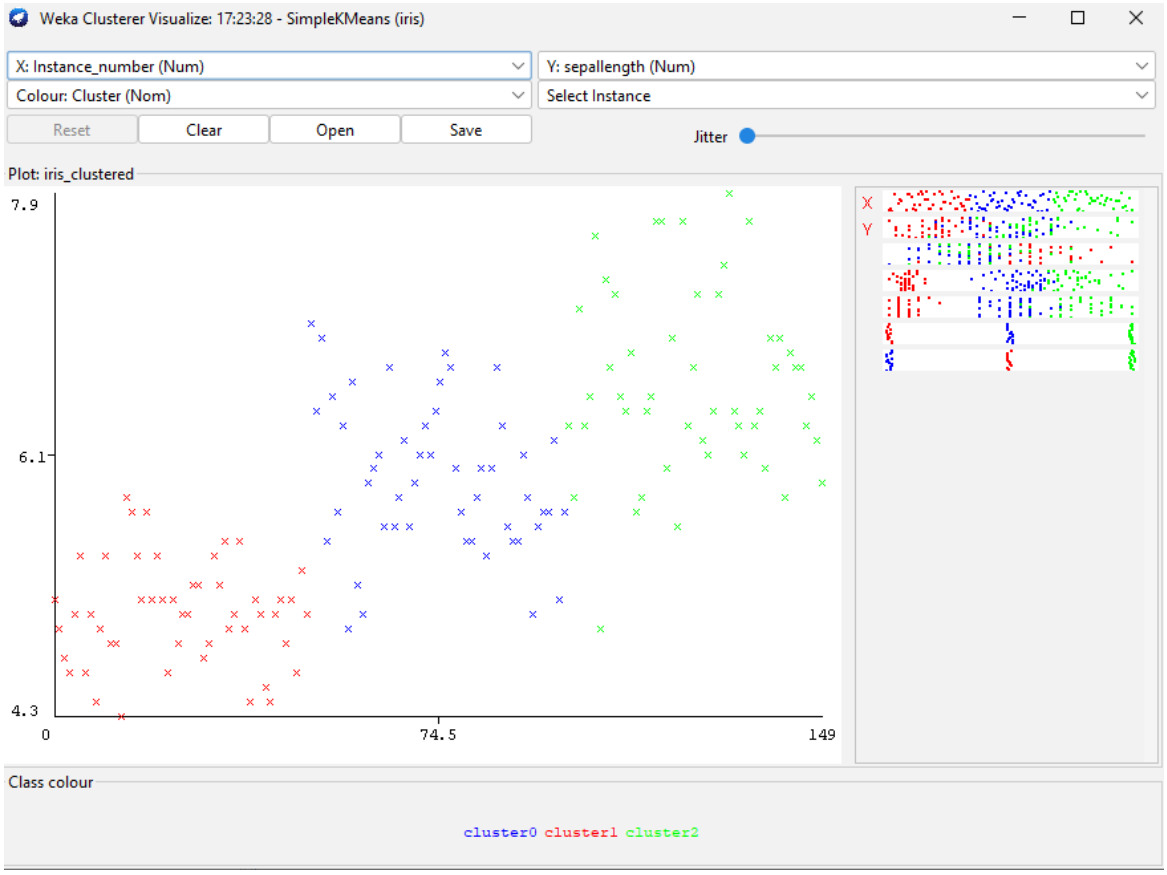
Με τον ίδιο τρόπο που επιλέγαμε τον αλγόριθμο, θα επιλέξουμε τον Simple K means, και μιας και είμαστε προϊδεασμένοι, από τις επιλογές του αλγόριθμου θα αλλάξουμε μια προεπιλογή από 2 clusters σε 3:



κάνουμε κλικ στο start και το αποτέλεσμα είναι αυτό:

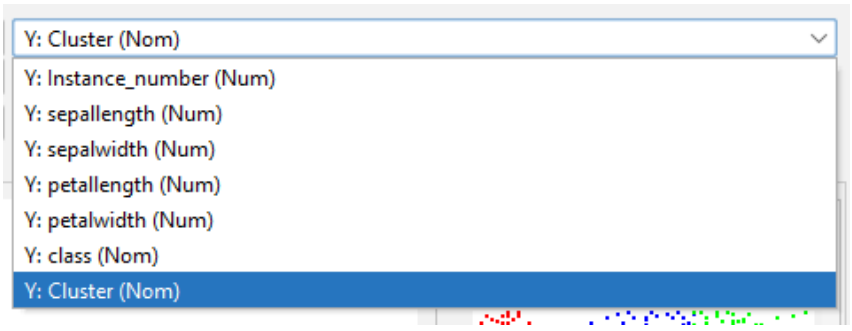
```
kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 7.817456892309574

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4,Iris-versicolor
Cluster 1: 6.2,2.9,4.3,1.3,Iris-versicolor
Cluster 2: 6.9,3.1,5.1,2.3,Iris-virginica

Missing values globally replaced with mean/mode

Final cluster centroids:
                                        Cluster#
Attribute               Full Data             0             1             2
                          (150.0)          (50.0)        (50.0)        (50.0)
================================================================================
sepallength                5.8433           5.936         5.006         6.588
sepalwidth                  3.054            2.77         3.418         2.974
petallength                3.7587            4.26         1.464         5.552
petalwidth                 1.1987           1.326         0.244         2.026
class                 Iris-setosa Iris-versicolor Iris-setosa  Iris-virginica




Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       50 ( 33%)
1       50 ( 33%)
2       50 ( 33%)
```

κάνουμε κι ένα δεξί κλικ στο result ώστε να επιλέξουμε το visualize cluster assignments και να πάρουμε την εικόνα:

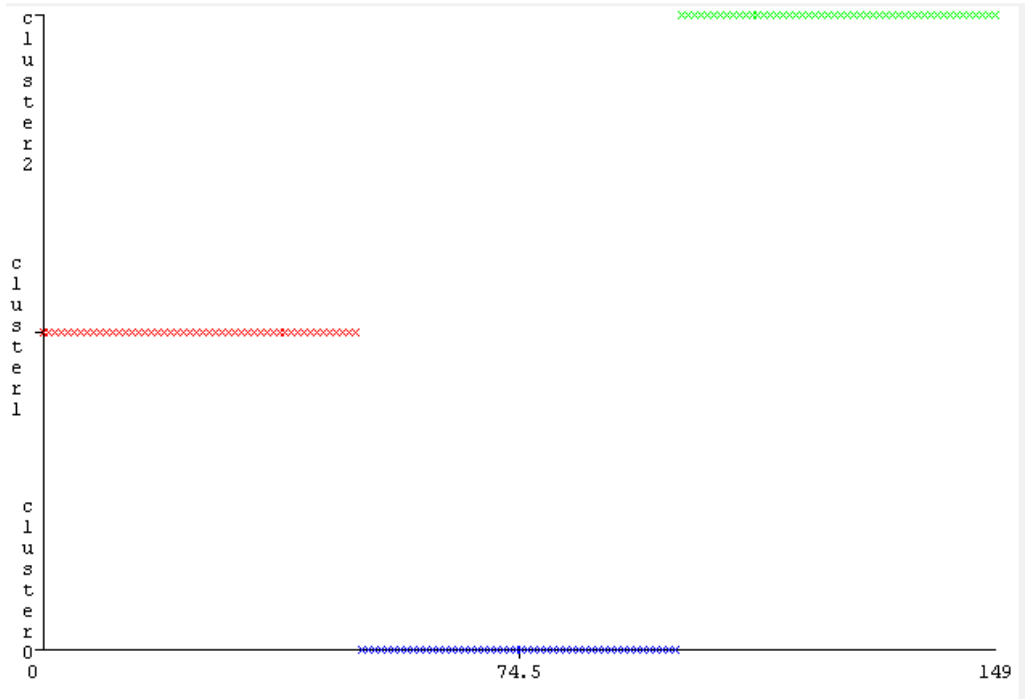


2.4 Clustering With Class Attribute

Ας δούμε τώρα από το σχήμα την επιλογή αυτή:



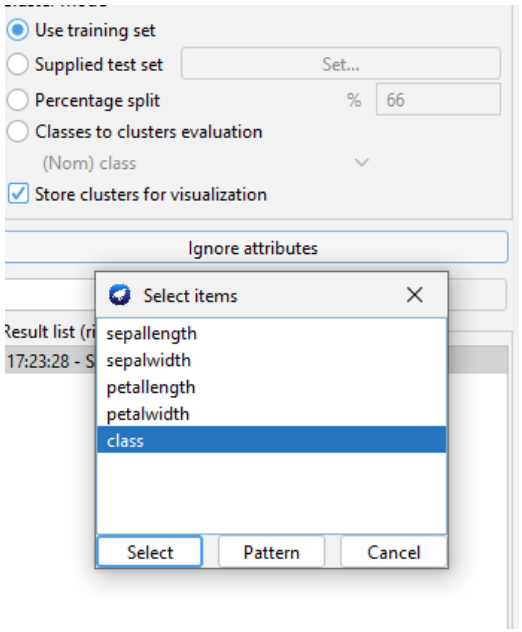
Αυτό θα μας δώσει ως αποτέλεσμα το σχήμα της παρακάτω εικόνας:



2.5 Visualize Clusters

Όπως παρατηρούμε, τα πάντα είναι ιδανικά. Από 33% κάθε cluster που αυτό σημαίνει ότι όλα πήγαν άψογα, και συνήθως στο machine learning, «όταν όλα πηγαίνουν τόσο καλά, τότε μάλλον κάτι δε λειτουργεί και τόσο σωστά» [2].

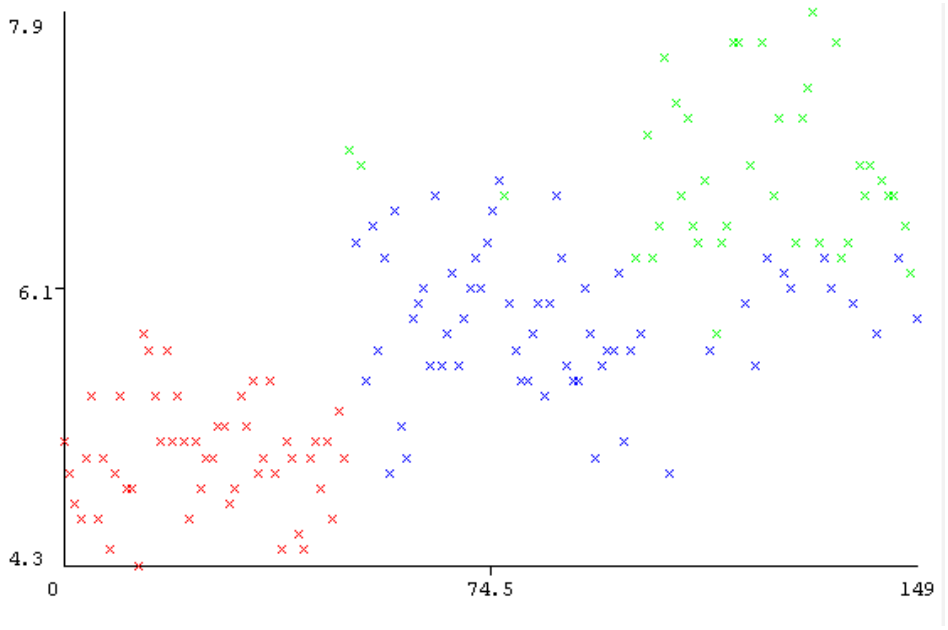
Το πρόβλημα εδώ, αν το καλοσκεφτούμε, είναι το attribute Class. Δεν είναι και τόσο σωστό να συμπεριλαμβάνεται το χαρακτηριστικό αυτό όταν προσπαθούμε να κάνουμε clustering, αλλά και γενικότερα στην μάθηση χωρίς επίβλεψη. Μπορούμε ωστόσο να το παραλείψουμε:



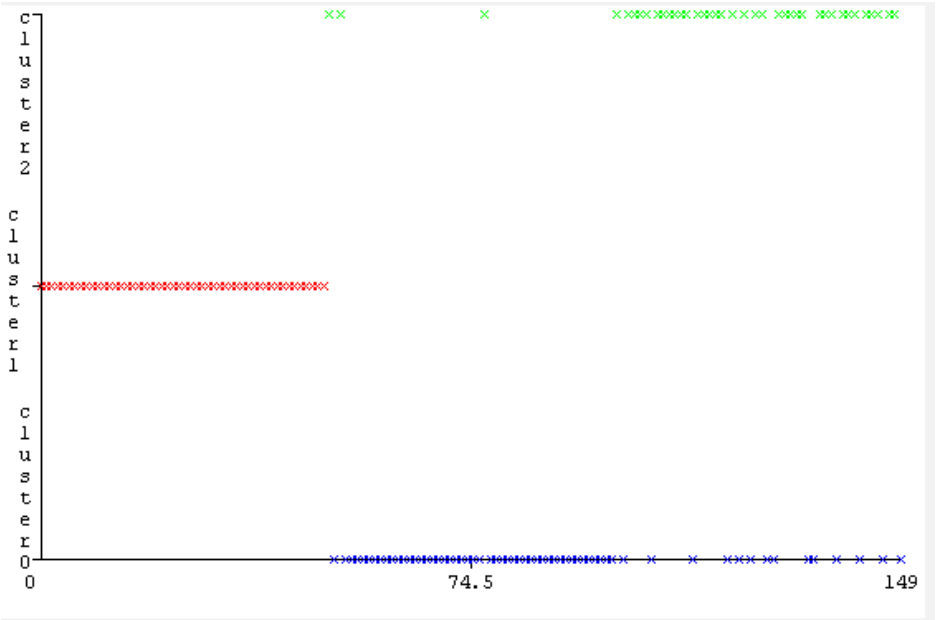
Το αποτέλεσμα είναι να έχουμε ένα αποτέλεσμα που, φυσικά, περιέχει σφάλματα όπως θα ήταν αναμενόμενο:

```
kMeans
======

Number of iterations: 6
Within cluster sum of squared errors: 6.998114004826762

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4
Cluster 1: 6.2,2.9,4.3,1.3
Cluster 2: 6.9,3.1,5.1,2.3

Missing values globally replaced with mean/mode

Final cluster centroids:
                           Cluster#
Attribute      Full Data          0          1          2
                 (150.0)     (61.0)     (50.0)     (39.0)
=========================================================
sepallength       5.8433     5.8885      5.006     6.8462
sepalwidth         3.054     2.7377      3.418     3.0821
petallength       3.7587     4.3967      1.464     5.7026
petalwidth        1.1987      1.418      0.244     2.0795




Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       61 ( 41%)
1       50 ( 33%)
2       39 ( 26%)
```

Και σχηματικά πε την επιλογή visualize:



2.6 Clustering χωρίς το class attribute

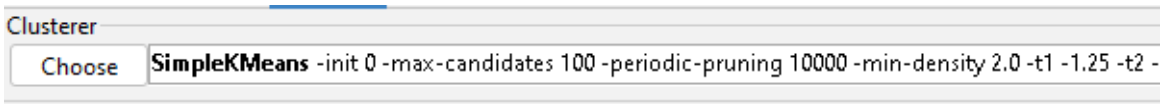


Εδώ φαίνεται ότι το 1ο cluster είναι ιδανικό, το 2ο και το 3ο έχουν κάποια σφάλματα, έχουν δηλαδή δεδομένα από άλλο cluster. Αξίζει να ειπωθεί ότι υπάρχει κι άλλος τρόπος απ το weka να εισάγουμε από τα filters, ένα unsupervised attribute, στην επιλογή AddCluster, και να επιλέξουμε το KMeans, όπου στην ουσία θα κατηγοριοποιήσει τα στιγμιότυπα σε clusters ακριβώς με τον ίδιο τρόπο που είδαμε πριν. Επίσης, αν στην επιλογή του K επιλέξουμε 2 ή 4 για παράδειγμα, θα έχουμε τον αντίστοιχο αριθμό από clusters.

2.1.2 Πειραματισμοί

Λίγο πιο πάνω αναφέρθηκαν τα βήματα του αλγόριθμου. Υπάρχει μια επανάληψη η οποία εκτελείται μέχρι ο αλγόριθμος να «καταλήξει» στα clusters του output. Για να πειραματιστούμε λίγο με τον αλγόριθμο το weka μας δίνει τη δυνατότητα να επιλέξουμε πόσες επαναλήψεις θα γίνουν. Για την ακρίβεια, υπάρχει επιλογή όπου εισάγουμε εμείς τον μέγιστο αριθμό επαναλήψεων που θα γίνει κατά την εκτέλεση.

Κάνοντας κλικ πάνω στον επιλεγμένο αλγόριθμο:



ανοίγει το παράθυρο των επιλογών, όπως είδαμε και πριν, εκεί που επιλέγουμε την τιμή του K, και πηγαίνουμε στο πεδίο «maxIterations» και από την προεπιλογή που είναι 500, το αλλάζουμε σε 1. Στην ουσία αυτό που παίρνουμε είναι το Initialization.

Το αποτέλεσμα είναι αυτό:

```
Number of iterations: 1
Within cluster sum of squared errors: 10.941419169169794

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4
Cluster 1: 6.2,2.9,4.3,1.3
Cluster 2: 6.9,3.1,5.1,2.3

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute      Full Data          0          1          2
                 (150.0)     (40.0)     (79.0)     (31.0)
=========================================================
sepallength       5.8433       6.07     5.3114     6.9065
sepalwidth         3.054       2.83     3.1418     3.1194
petallength       3.7587      4.855     2.3987     5.8097
petalwidth        1.1987     1.6275     0.5987     2.1742
```

```
Clustered Instances

0       65 ( 43%)
1       55 ( 37%)
2       30 ( 20%)
```

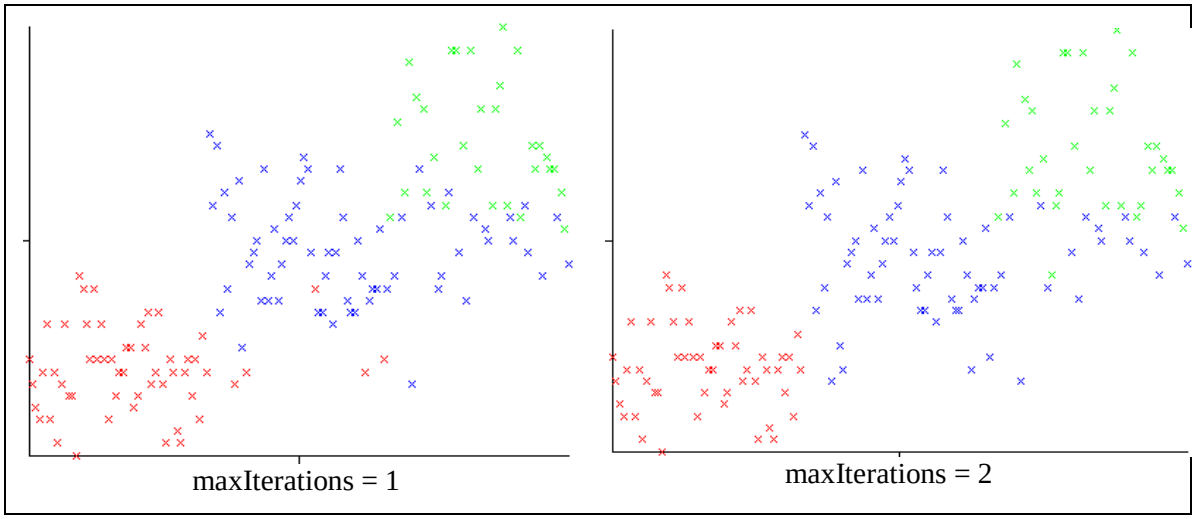
Αλλάζουμε την επιλογή σε 2

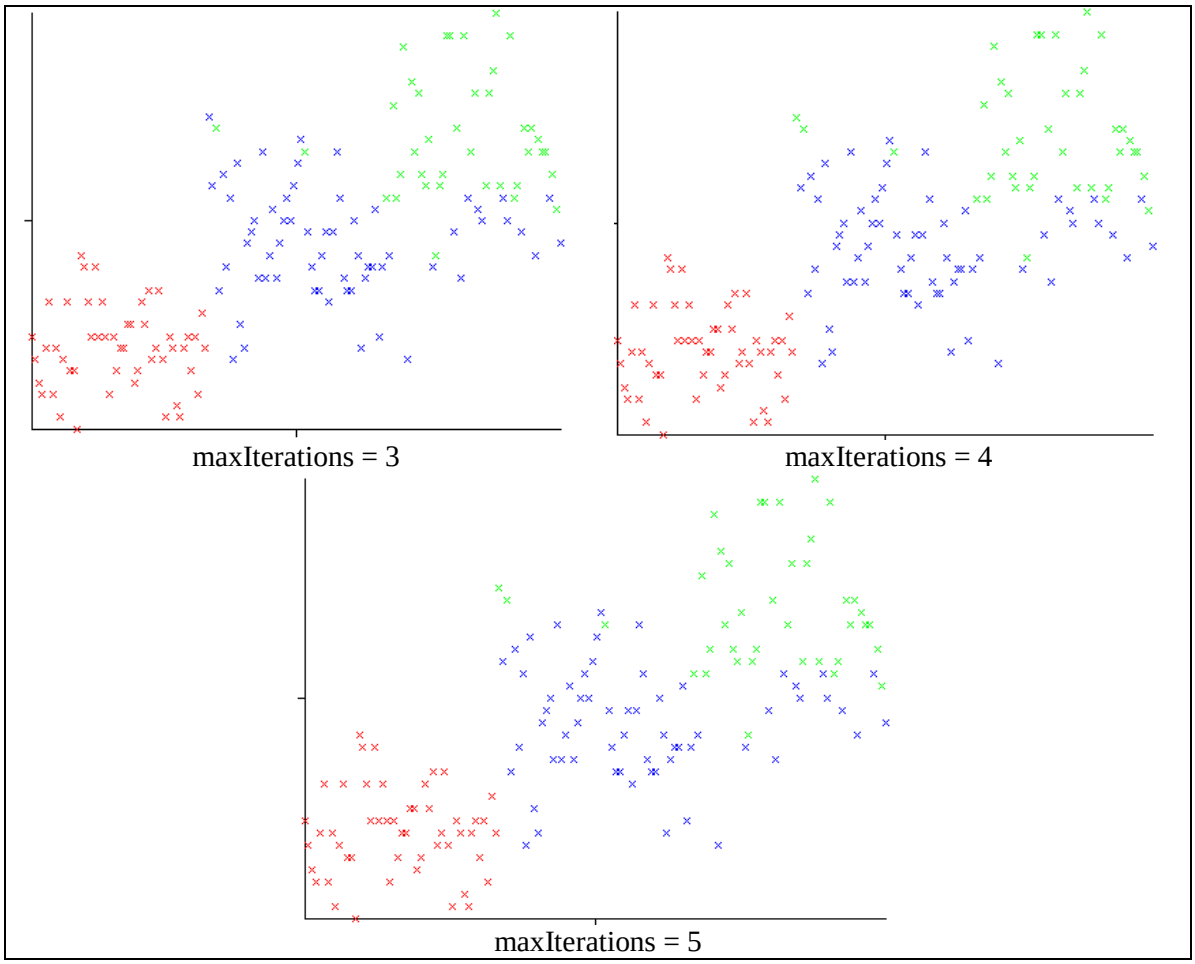
```
Number of iterations: 2
Within cluster sum of squared errors: 7.441610579129841

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4
Cluster 1: 6.2,2.9,4.3,1.3
Cluster 2: 6.9,3.1,5.1,2.3

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute      Full Data          0          1          2
                 (150.0)     (65.0)     (55.0)     (30.0)
=========================================================
sepallength       5.8433     6.0338     5.0182     6.9433
sepalwidth         3.054     2.7923     3.3218       3.13
petallength       3.7587        4.6     1.6327     5.8333
petalwidth        1.1987        1.5     0.3145     2.1667
```

```
=== Model and evaluation on training set ===

Clustered Instances

0       66 ( 44%)
1       50 ( 33%)
2       34 ( 23%)
```

Προφανώς μειώνεται ο συντελεστής των squared errors και αλλάζουν τα ποσοστά των στιγμιότυπων που εντάσσονται στα clusters.

Θα πειραματιστούμε με μερικές τιμές ακόμα και θα παραθέσουμε έναν πίνακα με τα γραφήματα. Να αναφέρουμε ότι στις 5 επαναλήψεις έχουν σταθεροποιηθεί τα centroids. Όταν δηλαδή αλλάξουμε την επιλογή maxIterations από 5 σε 6 δεν υπάρχει καμία αλλαγή.





Πίνακας 1 K-means Clustering by Num of Iterations

2.1.3 Συμπεράσματα

Στα πλεονεκτήματα του αλγόριθμου θα λέγαμε ότι είναι αρχικά γρήγορος[11] και ισχυρός, επίσης πολύ εύκολα κατανοητός, σχετικά αποδοτικός και δίνει πολύ καλά αποτελέσματα όταν το dataset έχει καλά διαχωρισμένα στιγμιότυπα.

Στα μειονεκτήματα, το πρώτο είναι ότι ο αλγόριθμος πρέπει να γνωρίζει από την αρχή τον αριθμό του K, δεν μπορεί δηλαδή να τον «μαντέψει». Είναι φτιαγμένος για γραμμικά δεδομένα, όπου μπορεί να ορισθεί ο μέσος όρος και δεν χειρίζεται σωστά ακραίες τιμές και “noisy” δεδομένα.

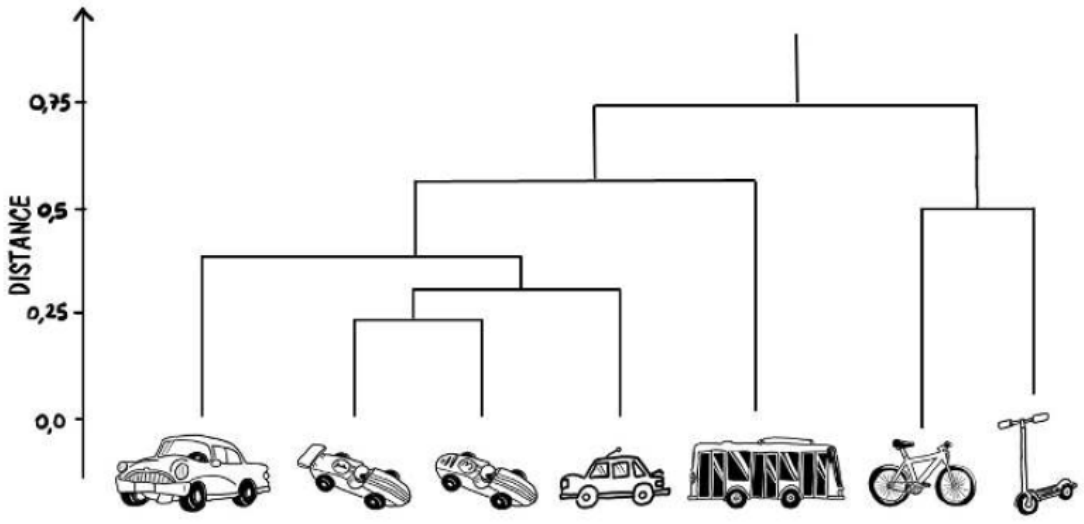
[11] K χρονική πολυπλοκότητα του αναφέρεται ως $O(n^2)$ [4]

2.2 Hierarchical Clustering

Μια επίσης σημαντική κατηγορία αλγόριθμων clustering είναι η Ιεραρχική που είναι εξίσου παλιά τεχνική με την K-means. Στους αλγόριθμους αυτής της κατηγορίας υπάρχουν δυο βασικές προσεγγίσεις [4]:

- η **Agglomerative**[12] προσέγγιση κατά την οποία κάθε σημείο αρχικά λογίζεται ως «ατομικό» cluster και σε κάθε βήμα προσπαθούμε να συγχωνεύσουμε τα πιο κοντινά-εγγύτερα ζεύγη. Αυτό φυσικά απαιτεί να ορίσουμε ως έννοια το τι θεωρούμε ως εγγύτητα όσον αφορά τα clusters.
- η **Divisive**, όπου στην ουσία το σύνολο των σημείων ορίζεται ως ένα cluster και σε κάθε βήμα προσπαθούμε να χωρίζουμε σε περισσότερα. Αυτό πρακτικά πρέπει να συμβαίνει μέχρι τα clusters που θα προκύψουν να αποτελούνται από ένα μόνο σημείο. Στην περίπτωση αυτή θα πρέπει να αποφασίζουμε ποια cluster πρέπει να διασπασθούν σε κάθε βήμα αλλά και πως θα γίνεται η διάσπαση.

Η πρώτη κατηγορία, η οποία μοιάζει και πιο λογική αλλά και μάλλον πιο εύκολη είναι και η πιο διαδεδομένη. Μάλιστα, για την υλοποίησή της, χρησιμοποιούνται και γραφικές μέθοδοι όπως δένδρα ή δενδρογράμματα(dendrograms). Ο σκοπός τους είναι να αναπαρίστανται σε ιεραρχική μορφή τα clusters, μαζί με τα επονομαζόμενα subclusters, με τη μορφή δένδρων – υποδένδρων και να φαίνεται η ιεραρχική σχέση μεταξύ τους.



2.7 Ένα Δενδρόγραμμα - bottom up clustering για οχήματα

Πηγή: The LION way [1]

[12] Η μετάφραση θα μπορούσε να είναι «συσσωρευτική» όπως προκύπτει και από τη λογική των αλγόριθμων της κατηγορίας αυτής.

2.2.1 Ο βασικός Agglomerative Αλγόριθμος

Ας παραθέσουμε τα βήματα του βασικού αλγόριθμου και στη συνέχεια θα πειραματιστούμε με το weka ώστε να καταλήξουμε σε κάποια χρήσιμα συμπεράσματα για τη μέθοδο αυτή.

1. Υπολογίζουμε τον πίνακα εγγύτητας[13], εάν είναι απαραίτητο
2. επαναλαμβάνουμε
    a. Συγχωνεύουμε τα δυο εγγύτερα μεταξύ τους clusters
    b. ενημερώνουμε τον πίνακα εγγύτητας
3. σταματούμε μόλις απομείνει ένα cluster

Προτού να προχωρήσουμε στον πειραματισμό με τα datasets, αξίζει να αναφερθεί ότι η τεχνική αυτή ονομάζεται και Bottom-up, με την έννοια ότι πηγαίνουμε από το ειδικό στο γενικό ή καλύτερα, όπως φαίνεται και στην εικόνα 2. 7, κάθε στιγμιότυπο στο input είναι ένα cluster και το output είναι ένα cluster όλο κι όλο σε αντίθεση με τον K-means που ανήκει στις top-down τεχνικές.

2.2.2 Hierarchical Clustering με το Weka

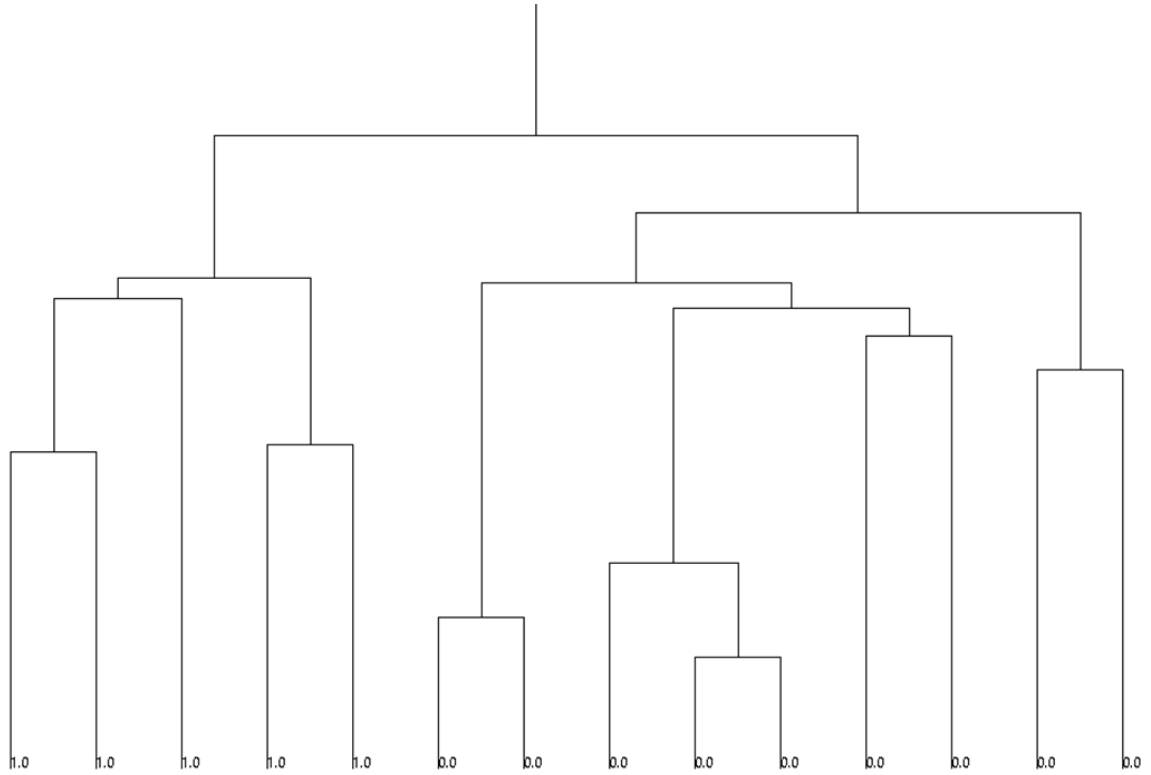
Θα χρησιμοποιήσουμε ξανά ένα dataset που χρησιμοποιήθηκε και στο 1ο κεφάλαιο, διότι είναι προτιμότερο να δουλέψουμε σε ένα σχετικά μικρό dataset, που να έχει δηλαδή λίγα στιγμιότυπα. Το dataset weather.mumeric.arff είναι ιδανικό.

Με τη γνωστή διαδικασία weka-explorer-open file-cluster-choose, επιλέγουμε τον αλγόριθμο Hierarchical Clusterer και προς το παρόν αφήνουμε τις προεπιλογές ως έχουν. Κάνουμε κλικ στο start και παίρνουμε τα παρακάτω αποτελέσματα:

```
=== Clustering model (full training set) ===

Cluster 0
((((1.0:0.94163,1.0:0.94163):0.45776,1.0:1.39939):0.06144,(1.0


Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      14 (100%)
```

[13] Ο τρόπος υπολογισμού των αποστάσεων και της εγγύτητας θα παρουσιαστεί μετά το παράδειγμα.

Η προεπιλογή NumberOfClusters είναι 1, οπότε θα δημιουργηθεί ένα cluster. Αυτό που πραγματικά έχει σημασία να δούμε είναι το dendrogram της επόμενης εικόνας. Για να εμφανιστεί κάνουμε δεξί κλικ στο buffer που «έτρεξε» ο αλγόριθμος και επιλέγουμε το VisualizeTree.



Εδώ γίνεται λίγο πιο ξεκάθαρος ο αλγόριθμος από τη άποψη ότι η ιεράρχηση των clusters από κάτω προς τα πάνω φανερώνει πως εάν μείνουμε μέχρι το σημείο που τα clusters είναι ακόμα 2 το σετ έχει χωριστεί σωστά στα play και don’t play. Θα λέγαμε ότι θυμίζει αρκετά την μέθοδο με επίβλεψη.

Εάν ανοίξουμε το αρχείο iris ώστε να πειραματιστούμε, μιας και είναι dataset με πολύ περισσότερα instances, βλέπουμε αρκετές διακυμάνσεις στα αποτελέσματα όταν επιλέγουμε διαφορετικό linkType. Τα καλύτερα αποτελέσματα τα παίρνουμε με την επιλογή COMPLETE. Το δενδρόγραμμα είναι τεράστιο οπότε δε θα παρουσιαστεί διότι 150 instances είναι δύσκολο να χωρέσουν ώστε να βγάλουμε κάποια χρήσιμα συμπεράσματα.

2.2.3 Υπολογισμός Εγγύτητας

Στην top-down και στην bottom-up συγχώνευση των στιγμιότυπων σε clusters απαιτείται ένας υπολογισμός απόστασης των δεδομένων αυτών, ώστε να ενώνονται με τα εγγύτερα clusters. Η απόσταση ανάμεσα σε δυο clusters γίνεται με διάφορους τρόπους, όπου φυσικά χρησιμοποιούνται μαθηματικοί τύποι(γεωμετρικοί δηλαδή) για τον υπολογισμό της. Κάθε τρόπος υπολογισμού, είναι λογικό να μας οδηγεί και σε διαφορετικά αποτελέσματα. Οι πιο συχνά χρησιμοποιούμενοι τύποι υπολογίζουν 1. την μέση απόσταση, 2. την ελάχιστη απόσταση και 3.

την μέγιστη απόσταση. Στην εικόνα 2.8 δίνονται οι τύποι αυτοί. Τα C, D είναι τα δυο clusters των οποίων θέλουμε να υπολογίσουμε την απόσταση.

$$\begin{aligned} \overline{\delta}_{ave}(C,D) &= \frac{\sum_{x\in C,\, y\in D}\delta(x,y)}{|C|\cdot|D|}; \\ \overline{\delta}_{min}(C,D) &= \min_{x\in C,\, y\in D}\delta(x,y); \\ \overline{\delta}_{max}(C,D) &= \max_{x\in C,\, y\in D}\delta(x,y). \end{aligned}$$

2.8 Τρεις τύποι υπολογισμού απόστασης

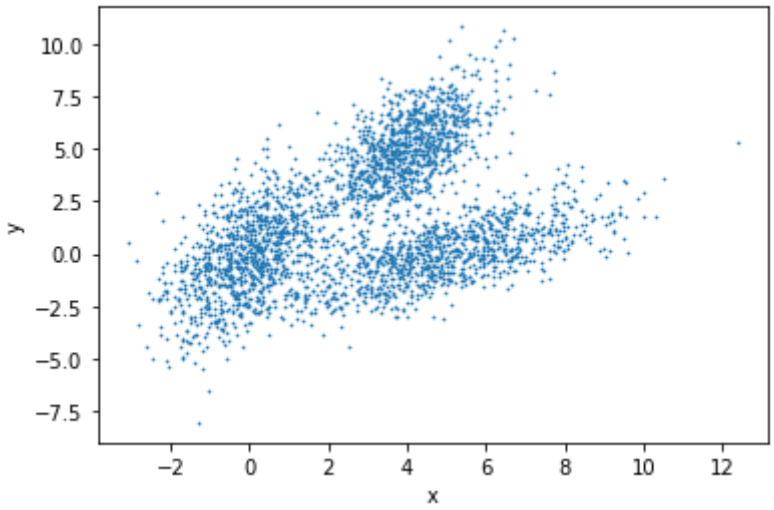
Πηγή: The Lion Way [1]

2.2.4 Συμπεράσματα

Οι Hierarchical Algorithms φαίνεται πως συχνά παράγουν πιο αξιόπιστα αποτελέσματα ή «καλύτερης ποιότητας clusters». Χρησιμοποιούνται σε εφαρμογές που απαιτούν ταξινόμηση η οποία απαιτεί ιεράρχηση. Όπως και να χει, απαιτούν πολύπλοκες πράξεις και αρκετό χώρο για τα δεδομένα καθώς και υπάρχει πάντα το πρόβλημα όλες αυτές οι συγχωνεύσεις να προκαλέσουν αυτό που ονομάζαμε «θόρυβο». Το δενδρόγραμμα παρόλα αυτά, μας βοηθά να βγάλουμε πολύ χρήσιμα συμπεράσματα για την μέθοδο της μάθησης χωρίς επίβλεψη.

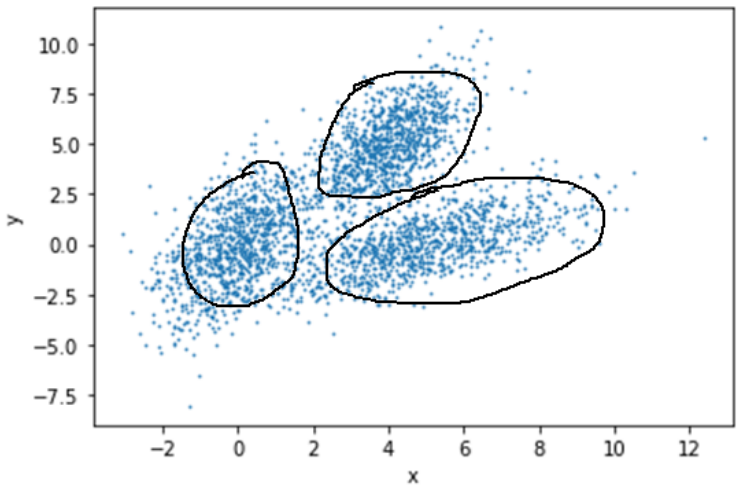
2.3 Density – Based clustering

Η τελευταία τεχνική clustering που θα μελετηθεί είναι αυτή που βασίζεται στην «πυκνότητα» των σημείων. Λαμβάνοντας για παράδειγμα ένα γράφημα όπως αυτό της εικόνας 2.9, όπως συμβαίνει στα περισσότερα παραδείγματα που είδαμε σήμερα, υπάρχουν σημεία με υψηλή και σημεία με χαμηλή πυκνότητα. Για το ανθρώπινο μάτι είναι μάλλον προφανές ότι το ιδανικό για το γράφημα αυτό θα ήταν ο αλγόριθμος που θα χρησιμοποιηθεί να «καταλήξει» σε τρία clusters όπως στην εικόνα 2.10.



2.9 Data before Clustering

Πηγή: https://training.galaxyproject.org/



2.10 After clustering

Αν κάθε αραιό σημείο του γραφήματος είναι ικανό ώστε να μπορεί να διαχωρίζει τα clusters μεταξύ τους, και κάθε πυκνό ικανό ώστε να δημιουργεί ένα, τότε πρέπει να εξετάσουμε πως υπολογίζεται η πυκνότητα αυτή. Όπως και σε προηγούμενα παραδείγματα, πρέπει κάποια σημεία του γραφήματος να επιλέγονται με τέτοιο τρόπο ώστε να μας εξυπηρετούν στο να βρίσκουμε π.χ.

γειτονικά σημεία εύκολα. Στην υλοποίηση που βασίζεται στη συχνότητα, αυτό που θέλουμε να βρούμε είναι ένα «κεντρικό σημείο». Ένα σημείο που θα βρίσκετε σε μια πυκνή «γειτονιά» και θα είναι σε μια κεντρική θέση, όπου κεντρική εννοούμε ότι θα έχει όσο το δυνατόν περισσότερα σημεία σε όσο το δυνατό μικρότερη ακτίνα. Η τεχνική αυτή ονομάζεται center-based density, δηλαδή πυκνότητα βασισμένη στα κέντρα, ή καλύτερα, στα κεντρικά σημεία. Τα σημεία αυτά ονομάζονται κέντρα γιατί πρακτικά γύρο από την ακτίνα τους θα ομαδοποιούνται τα υπόλοιπα στα clusters. Άρα, προς το παρόν κρατάμε δυο βασικά πράγματα τα οποία θα αναφερθούν και στα συμπεράσματα. Πρώτο, πρέπει να γίνεται υπολογισμός των κεντρικών αυτόν σημείων, ή core points, και δεύτερο, πρέπει να λαμβάνεται μια απόφαση σχετικά με το μέγεθος της ακτίνας. Και τα δυο είναι πολύ σημαντικά για την έκβαση του τελικού αποτελέσματος αλλά φυσικά και σχετίζονται μεταξύ τους. Για τον υπολογισμό για παράδειγμα τριών core points μπορούμε να ορίσουμε ένα συγκεκριμένο μέγεθος ακτίνας και να επιλεχθούν τα τρία σημεία τα οποία έχουν τους περισσότερους γείτονες. Αυτό βέβαια θέλει προσοχή ώστε αυτά τα τρία σημεία, π.χ. Α, Β και Γ να μην ανήκουν στην ίδια γειτονιά, γιατί αν ανήκουν τότε είναι σχεδόν βέβαιο ότι θα πρόκειται για σημεία που είναι πολύ κοντά το ένα στο άλλο, κάτι που δε μας εξυπηρετεί.

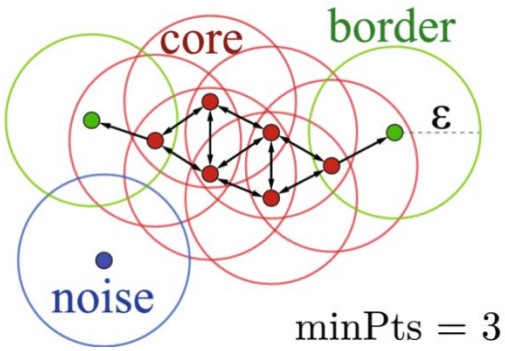
Το μέγεθος της ακτίνας πρέπει να είναι «ιδανικό» διότι, αν είναι πολύ μεγάλο τότε κάθε core point θα έχει όλα τα υπόλοιπα σημεία εντός της ακτίνας του, ενώ αν είναι υπερβολικά μικρό θα έχει μόνο τον εαυτό του. Στο σημείο αυτό είναι φανερό πως όσο απλό και αν φαίνεται σαν λογική τόσο δύσκολο είναι στην υλοποίησή του. Απλά να θυμίσουμε ότι στη μάθηση χωρίς επίβλεψη αυτή ήταν εξαρχής η δυσκολία. Μια καλή προσέγγιση είναι να υπολογίζεται η ακτίνα με κριτήριο το πόσα σημεία θέλουμε να βρίσκονται εντός της ακτίνας αυτής. Με βάση τον υπολογισμό των κεντρικών σημείων και το μέγεθος της ακτίνας προκύπτουν οι τρεις παρακάτω έννοιες που αφορούν τα σημεία ενός γραφήματος [4]:

- σημεία πυρήνα ή κεντρικά σημεία(core points), είναι τα σημεία τα οποία πρέπει να ικανοποιούν κάποιο ή κάποια κριτήρια ώστε να είναι κεντρικά, για παράδειγμα να έχουν το λιγότερο 10 γειτονικά σημεία εντός μιας δοσμένης ακτίνας.(το 10 είναι τυχαίος αριθμός εδώ)
- γειτονικά σημεία(border points) ή σημεία γειτονίας καλύτερα, είναι τα σημεία που δεν είναι κεντρικά, δεν είναι σημεία της τελευταίας κατηγορίας και βρίσκονται εντός της ακτίνας κεντρικών. Σε ακριβέστερη μετάφραση λέγονται συνοριακά σημεία, αλλά μάλλον το γειτονικά είναι πιο σωστό εδώ.
- μη-γειτονικά σημεία(noise points), τα σημεία που δεν ανήκουν σε καμία από τις παραπάνω κατηγορίες. Αναφέρονται και ως σημεία θορύβου ή μη συνοριακά[14]. Με απλά

[14] Ενίοτε αναφέρονται και ως ακραία σημεία.

λόγια, δεν έχουν τον ικανό αριθμό ελάχιστων γειτόνων αλλά ούτε και ανήκουν σε κάποια πυκνή γειτονιά.

Στην εικόνα 2.11 βλέπουμε κάποια κεντρικά σημεία, κάποια γειτονικά και ένα noise point. Ικανή συνθήκη για να είναι ένα σημείο κεντρικό είναι να έχει τουλάχιστον τρεις γείτονες εντός της προκαθορισμένης ακτίνας.



2.11 Core, border, noise points

Πηγή: https://www.researchgate.net/

2.3.1 DBSCAN algorithm

Για τον υπολογισμό όλων των παραπάνω καθώς και για την δημιουργία των clusters απαιτείται ως συνήθως μια αλγοριθμική μέθοδος. Εδώ θα παρουσιάσουμε τον αλγόριθμο που βασίζεται σε αυτό που ονομάσαμε πυκνότητα και ο αλγόριθμος αυτός ονομάζεται DBSCAN[15]. Ο αλγόριθμος αυτός πρέπει να υπολογίζει και να θέτει τα σημεία – στιγμιότυπα ως κεντρικά, γειτονίας ή θορύβου και να τα τοποθετεί στα κατάλληλα clusters. Στον DBSCAN αλγόριθμο, αν δυο σημεία είναι κεντρικά αλλά είναι και γειτονικά μεταξύ τους, δηλαδή είναι εντός της καθορισμένης ακτίνας, τότε πρέπει να ενσωματωθούν στο ίδιο cluster. Το πρόβλημα που αρχικά προκύπτει είναι: αν ένα σημείο είναι κεντρικό αλλά ανήκει και στη γειτονιά άλλων δυο κεντρικών σημείων τότε ποιο σημείο θα παραμείνει κεντρικό και ποιο θα γίνει γειτονικό; Ή ακόμα κι αν ένα σημείο δεν είναι κεντρικό αλλά ανήκει επίσης στη γειτονιά δυο κεντρικών τότε σε ποιο cluster θα ενσωματωθεί; Στον αλγόριθμο υπάρχει μία ακόμη έννοια, αυτή της προσέγγισης με βάση την πυκνότητα. Στην εικόνα 2.11 βλέπουμε κάποιες ακμές ανάμεσα σε όλα τα κεντρικά σημεία(με κόκκινο χρώμα) αλλά και σε δυο σημεία γειτονίας(με πράσινο χρώμα). Αυτά τα δυο σημεία λέμε ότι είναι προσεγγίσιμα. Προσεγγίσιμα σημεία με βάση την πυκνότητα, λέγονται δυο σημεία Α και Β εάν μεταξύ τους υπάρχει ένα μονοπάτι αποτελούμενο από κεντρικά σημεία που είναι ταυτόχρονα και γειτονιά, δηλαδή εντός της ακτίνας.

[15] Density-based spatial clustering of applications with noise

Συνοπτικά μπορούμε να πούμε πως ο αλγόριθμος DBSCAN ακολουθεί τα παρακάτω βήματα [4]:

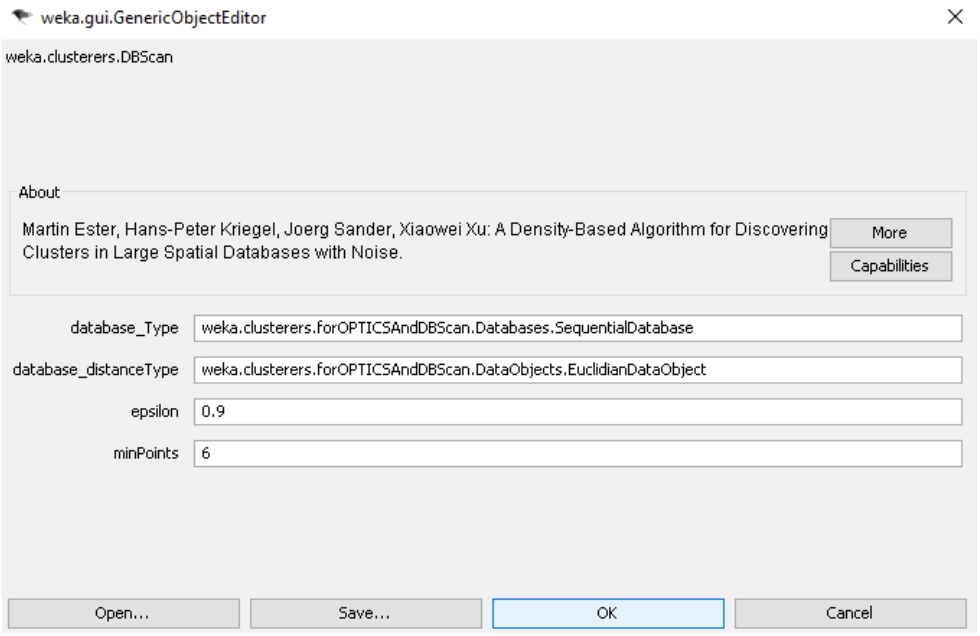
- Βρίσκουμε όλα τα γειτονικά σημεία για κάθε σημείο ξεχωριστά ώστε να χαρακτηριστεί κάθε σημείο ως κεντρικό, γειτονίας ή θορύβου. Αυτό που πρέπει να «γνωρίζει» ο αλγόριθμος είναι το μήκος της ακτίνας(Eps) αλλά και ο ελάχιστος ικανός αριθμός γειτόνων(MinPts) ώστε ένα σημείο να θεωρηθεί κεντρικό.
- Δημιουργούμε τα clusters όπου στην ουσία κάθε cluster αποτελείται από το κεντρικό σημείο και τα γειτονικά του. Δηλαδή σε αυτό το βήμα κάθε γειτονικό σημείο ενσωματώνεται σε ένα cluster.
- Αν δυο ή περισσότερα clusters περιέχουν πάνω από ένα κεντρικό σημείο, τότε συνενώνονται σε ένα cluster το οποίο περιέχει όλα τα κεντρικά και γειτονικά σημεία των clusters που ενώνονται αλλά και όλα τα προσεγγίσιμα σημεία.
- Η διαδικασία του clustering επαναλαμβάνεται ακόμη και για τα σημεία που αρχικά είχαν οριστεί ως σημεία θορύβου.
- Κάθε σημείο που παραμένει σημείο θορύβου δεν ενσωματώνεται σε κανένα cluster.

2.3.2 DBSCAN with Weka

Για της ανάγκες αυτού του αλγόριθμου χρησιμοποιήθηκε μια παλιά έκδοση του Weka και πιο συγκεκριμένα η 3.6.0 διότι οι νεότερες εκδόσεις δεν περιέχουν τον αλγόριθμο αυτόν. Το περιβάλλον της παλαιότερης έκδοσης είναι ίδιο, οπότε πάλι από το open file στον explorer του weka επιλέγουμε το dataset με το οποίο θα πειραματιστούμε. Αρχικά θα επιλέξουμε το αρχείο iris, και από την επιλογή cluster επιλέγουμε με τη γνωστή διαδικασία τον DBScan. Αρχικά δεν θα αλλάξουμε καμία από τις προεπιλεγμένες ρυθμίσεις(εννοείται πως έχουμε αφαιρέσει από τα attributes το class). Το αποτέλεσμα είναι το εξής:

```
Scheme:       weka.clusterers.DBScan -E 0.9 -M 6 -I weka.clusterers.forOPTICSAndDBScan.Databases.Seque
Relation:     iris-weka.filters.unsupervised.attribute.Remove-R5
Instances:    150
Attributes:   4
              sepallength
              sepalwidth
              petallength
              petalwidth
Test mode:    evaluate on training data

=== Model and evaluation on training set ===

DBScan clustering results
==========================================================================================

Clustered DataObjects: 150
Number of attributes: 4
Epsilon: 0.9; minPoints: 6
Index: weka.clusterers.forOPTICSAndDBScan.Databases.SequentialDatabase
Distance-type: weka.clusterers.forOPTICSAndDBScan.DataObjects.EuclidianDataObject
Number of generated clusters: 1
Elapsed time: ,02

(  0.) 5.1,3.5,1.4,0.2                                                  -->  0
(  1.) 4.9,3,1.4,0.2                                                    -->  0
(  2.) 4.7,3.2,1.3,0.2                                                  -->  0
(  3.) 4.6,3.1,1.5,0.2                                                  -->  0
(  4.) 5,3.6,1.4,0.2                                                    -->  0
(  5.) 5.4,3.9,1.7,0.4                                                  -->  0
(  6.) 4.6,3.4,1.4,0.3                                                  -->  0

                        Clustered Instances

0       150 (100%)
```

Αρχικά βλέπουμε πως η προεπιλογή είναι (ακτίνα)Epsilon: 0.9 και MinPts: 6. Δημιουργήθηκε ένα cluster και είναι μάλλον απόλυτα λογικό όπως θα εξηγήσουμε στη συνέχεια. Αυτό που μας ενδιαφέρει σε πρώτη φάση είναι να πειραματιστούμε με τις τιμές της ακτίνας και των ελάχιστων γειτονικών σημείων. Κάνουμε κλικ στον αλγόριθμο για να δούμε τις επιλογές:



Μετά από αρκετούς πειραματισμούς, τα καλύτερα αποτελέσματα δίνονται όταν οι επιλογή της ακτίνας είναι 0.4 με 0.6 και το MinPts κοντά στο 5.

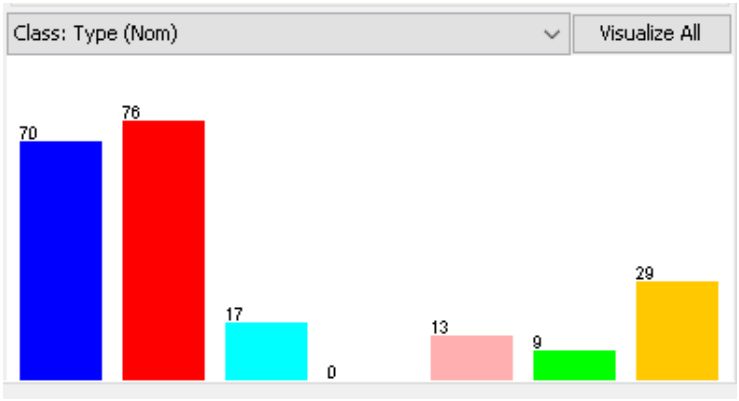

```
Clustered Instances

0       50 ( 34%)
1       99 ( 66%)

Unclustered instances : 1
```

Είναι φανερό ότι ο αλγόριθμος λειτουργεί όπως ήταν αναμενόμενο. Στα συμπεράσματα θα εξηγήσουμε το λόγο για τον οποίο τα αποτελέσματα για αυτό το σετ είναι τόσο απογοητευτικά.

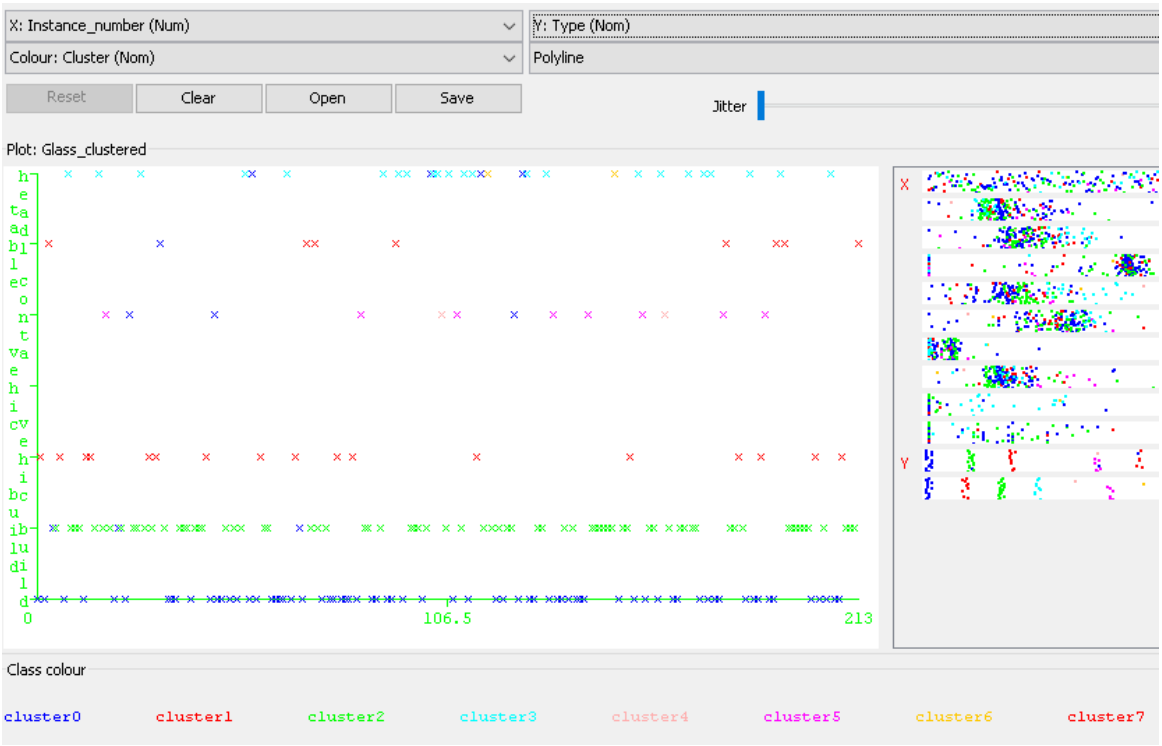
Ας δοκιμάσουμε με άλλο dataset το οποίο παρουσιάζει στο γράφημά του περιοχές με μεγαλύτερη πυκνότητα από αυτή του iris. Αυτό που παρουσιάζει ιδιαίτερο ενδιαφέρον να δούμε είναι το glass dataset που έχει χρησιμοποιηθεί και προηγουμένως. Αφήνουμε αρχικά την ακτίνα και τα ελάχιστα σημεία ως έχουν. Το αποτέλεσμα είναι συνοπτικά:

```
Clustered Instances

0       70 ( 33%)
1       17 (  8%)
2       75 ( 36%)
3       29 ( 14%)
4       10 (  5%)
5        9 (  4%)

Unclustered instances : 4
```

κάτι που δεν απέχει πολύ από το επιθυμητό:



Αυτό που έχει μεγάλη σημασία να δούμε είναι το γράφημα των πέντε clusters που πήραμε από την υλοποίηση αυτή:

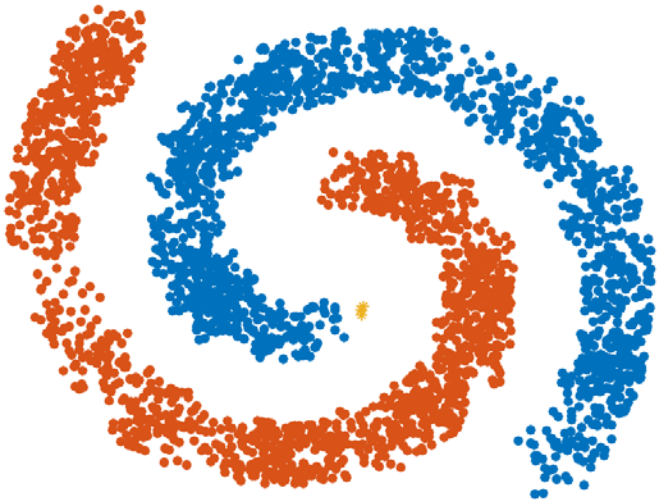


2.12 DBSCAN clustering in glass.arff

Σε κανέναν από τους προηγούμενους αλγόριθμους δεν είδαμε τα clusters να «εισχωρούν» στον χώρο του άλλου. Με βάση όμως το πώς λειτουργεί ο αλγόριθμος αυτός είναι μάλλον φυσικό να έχουμε κάτι τέτοιο. Αν πειραματιστούμε αλλάζοντας τις προεπιλογές μειώνοντας και την ακτίνα και τα ελάχιστα σημεία τότε παίρνουμε περισσότερα clusters αλλά και περισσότερα σημεία εκτός αυτών. Ενδεικτικά, για ακτίνα στο 0.5 και ελάχιστα σημεία στο 2 παίρνουμε 7 clysters και 11 σημεία θορύβου.

2.3.3 Συμπεράσματα

Όπως όλοι οι αλγόριθμοι που μελετήθηκαν στην εποπτευόμενη μάθηση αλλά και στη μάθηση χωρίς επίβλεψη έχουν τα πλεονεκτήματα και τα μειονεκτήματά τους, είτε όσο αφορά μόνο το αποτέλεσμα που παράγουν, αν είναι δηλαδή ικανοποιητικό ή όχι, είτε σε σύγκριση με άλλους αλγόριθμους. Μερικοί είναι πιο αποτελεσματικοί γιατί απλά είναι «εξυπνότεροι» ενώ άλλοι γιατί πολύ απλά τους «βολεύει» κάποιο dataset. Ο λόγος που εδώ εξετάστηκαν τα δυο Dataset αυτά είναι γιατί ο DBSCAN σε αντίθεση με τον K-means για παράδειγμα, μπορεί να σχηματίζει clusters που δεν έχουν απαραίτητα κυκλικό σχήμα. Είναι φανερό ότι για να πετύχει αυτό πρέπει το γράφημα να έχει πυκνές και αραιές περιοχές. Στο iris όσο και να πειραματιστούμε δεν θα πάρουμε τα τρία clusters που επιθυμούμε ιδανικά όπως και με τον K-means στο αρχείο glass δε θα παίρναμε τα clusters τόσο αποτελεσματικά όσο στον DBSCAN. Αυτό καθιστά τον αλγόριθμο πολύ πιο αποτελεσματικό σε γραφήματα όπως αυτό της εικόνας 2.13.



2.13 After DBSCAN

Πηγή: https://www.mathworks.com/

Στα θετικά του DBSCAN να αναφέρουμε πως λόγο των σχηματισμών των clusters με τον τρόπο που αναφέρθηκε δεν επηρεάζεται από τα ακραία σημεία, αλλά τα θεωρεί θόρυβο. Ως πιο θετικό συμπέρασμα θα λέγαμε πως αν και πρέπει πάλι να προσδιορίσουμε δυο τιμές, της ακτίνας και των ελάχιστων σημείων, παρόλα αυτά, σε σχέση και πάλι με άλλους αλγόριθμους clustering δεν χρειάζεται να προεπιλέξουμε τον αριθμό των clusters. Ως μειονέκτημα θα αναφέρουμε πως δεν είναι καθόλου αποτελεσματικός σε datasets τα οποία δεν παρουσιάζουν περιοχές αραιές και πυκνές και συχνά οδηγούν σε λάθος σχηματισμούς clusters. Σε κάθε περίπτωση είναι μια πολύ ενδιαφέρουσα τεχνική.

3. Το λογισμικό Weka

Το Weka το οποίο χρησιμοποιήθηκε εκτενώς στα προηγούμενα δυο κεφάλαια, είναι μια εφαρμογή η οποία σχεδιάστηκε από το Πανεπιστήμιο του Waikato στη Νέα Ζηλανδία. Το όνομα Weka προκύπτει από τα αρχικά των λέξεων «Waikato Environment for Knowledge Analysis» και όπως είδαμε, είναι ένα περιβάλλον εργασίας με σκοπό τη μελέτη των αλγόριθμων μηχανικής μάθησης πάνω σε δεδομένα. Η εφαρμογή χρονολογείται από το 1999 και η έκδοση που χρησιμοποιήθηκε είναι η 3.8.6, εκτός από τον αλγόριθμο BDSCAN ο οποίος υπάρχει σε παλιότερες εκδόσεις μόνο. Για τον αλγόριθμο αυτό χρησιμοποιήθηκε η 3.6.0.

Το περιβάλλον του είναι παραθυρικό και εξαιρετικά εύκολο στην εκμάθηση και τη χρήση του μιας και δεν απαιτεί την εκ βάθος γνώση όλων των αλγόριθμων που εμπεριέχει και ως εκ τούτου είναι φιλικό ακόμα και στον χρήστη που ενδεχομένως να μην έχει ως σκοπό την εκμάθησή τους. Είναι γραμμένο με την γλώσσα προγραμματισμού Java, που σημαίνει ότι πρέπει στον Η/Υ να έχουμε εγκατεστημένη την απαραίτητη έκδοση της Java, ανάλογα με το ποια έκδοση του Weka έχουμε, όπως φαίνεται στον πίνακα της εικόνας που ακολουθεί:

Weka	Java 1.4	Java 5	Java 6	Java 7	Java 8 or later
<3.4.0	☑	☑	☑	☑	☑
3.4.x	☑	☑	☑	☑	☑
3.5.x	<3.5.3	☑	☑	☑	☑
3.6.x		☑	☑	☑	☑
3.7.x		3.7.0	<3.7.14	☑	☑
3.8.x				<3.8.2	☑
3.9.x				<3.9.2	☑

3.1 Πίνακας επιλογής κατάλληλης Έκδοσης Java

Πηγή: https://waikato.github.io/weka-wiki/requirements/

3.1 Τα Αρχεία .arff

Τα datasets που χρησιμοποιήθηκαν στα κεφάλαια 1 και 2, ήταν ενσωματωμένα στο Weka και είχαν όλα την επέκταση .arff. Όπως για παράδειγμα τα αρχεία του excel πρέπει να έχουν την

επέκταση .xls ή .xlsx, τα αρχεία του word να έχουν την επεκταση .doc, έτσι υπάρχει και εδώ το κατάλληλο αναγνωριστικό.

3.1.1 Η Δομή ενός Αρχείου .arff

Ας δούμε ένα μικρό αρχείο που έχει ήδη χρησιμοποιηθεί. Το αρχείο έχει τον τίτλο "weather.numeric.arff" και για να δούμε τη δομή και τα δεδομένα του μπορούμε να το ανοίξουμε σαν αρχείο WordPad. Παρεμπιπτόντως, αυτό αυτόματα σημαίνει πως είναι σχετικά εύκολο και απλό να δημιουργήσουμε ένα δικό μας αρχείο .arff το οποίο θα μπορούμε να το ανοίγουμε από το Weka και να δουλέψουμε τους αλγόριθμους με τα δεδομένα του.

```
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature numeric
@attribute humidity numeric
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes
sunny,72,95,FALSE,no
sunny,69,70,FALSE,yes
rainy,75,80,FALSE,yes
sunny,75,70,TRUE,yes
overcast,72,90,TRUE,yes
overcast,81,75,FALSE,yes
rainy,71,91,TRUE,no
```

3.2 Το αρχείο weather.numeric.arff από το WordPad

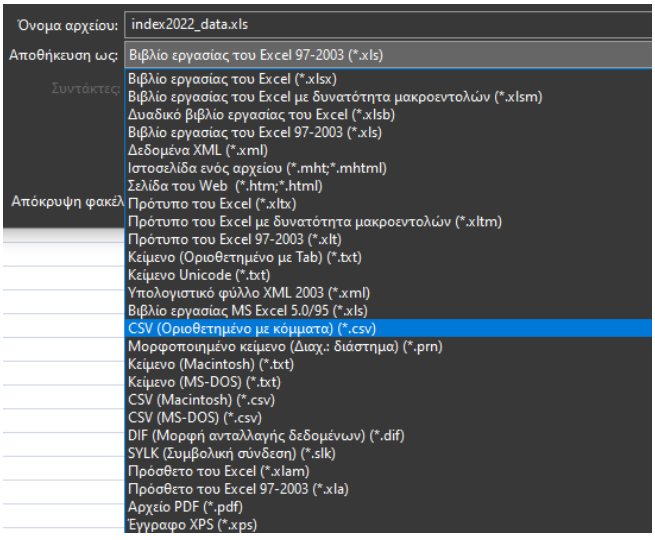
Στην πρώτη γραμμή του αρχείο, μετά τον ειδικό χαρακτήρα «**@**» υπάρχει η λέξη **relation** την οποία ακολουθεί ο τίτλος του αρχείου. Ακολουθεί ένα τμήμα, όμοιο με δήλωση μεταβλητών σε γλώσσα προγραμματισμού. Πάλι ο ειδικός χαρακτήρας «παπάκι», ακολουθεί η λέξει attribute(ιδιότητα ή χαρακτηριστικό» καθώς και ο «τύπος» του χαρακτηριστικού αυτού. Μετά τη λέξη **@data** ακολουθούν τα δεδομένα του dataset με μορφή στιγμιότυπων(instances), δηλαδή, το πρώτο instance, "sunny, 85, 85, FALSE, no", όπου στην ουσία η «τιμή» sunny αναφέρεται στο 1^{ο} attribute, το 85 στο 2ο attribute, temperature και ούτω ο καθεξής. Τα δεδομένα δηλαδή, ακολουθούν αυτή τη σειρά.

Τα δεδομένα που περιλαμβάνει ένα data set όπως το παραπάνω, μπορούν να είναι είτε αριθμητικά(numeric)[16] είτε εικονικά-ονομαστικά(nominal), θα αναφέρονται δηλαδή με κάποιο όνομα σε άγκιστρα, ακόμα κι αν είναι αριθμοί από ένα συγκεκριμένο κλειστό σύνολο. Πιο συγκεκριμένα, ένα numeric attribute μπορεί να πάρει οποιαδήποτε αριθμητική τιμή ενώ ένα nominal μόνο τις τιμές που υπάρχουν στη «δήλωση» του attribute μέσα στα άγκιστρα, π.χ. {yes, no} ή {2, 4, 6, 8} κλπ.. Οι παραπάνω κατηγορίες χωρίζονται σε υποκατηγορίες, όπου για τα αριθμητικά δεδομένα έχουμε τις διακριτές τιμές και τις συνεχόμενες(π.χ. ένα δεδομένο μπορεί να πάρει κάποιες συγκεκριμένες τιμές, προφανώς για τα numeric δεδομένα). Αυτό καθορίζεται επίσης στο σημείο που «δηλώνονται» τα attributes του dataset, εκεί καθορίζεται δηλαδή ο τύπος τους.

3.1.2 Δημιουργία αρχείου .arff

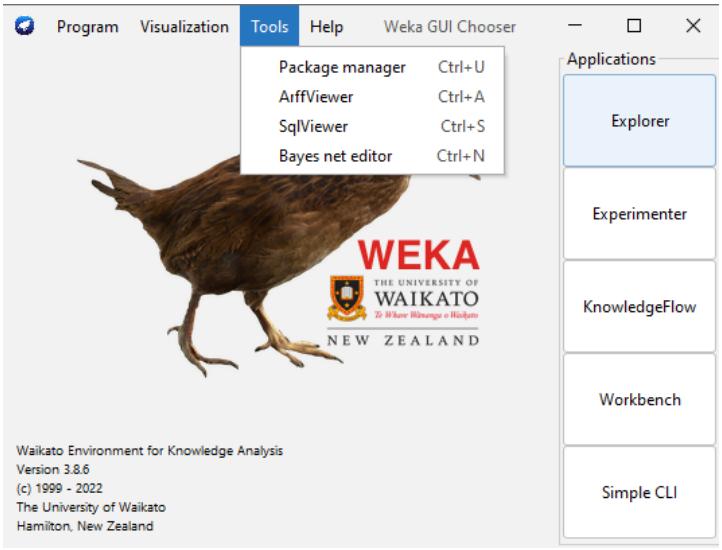
Αν υποθέσουμε ότι έχουμε συλλέξει κάποια στατιστικά δεδομένα, που αφορούν για παράδειγμα κάποια οικονομικά στοιχεία και θέλουμε να δουλέψουμε πάνω σε αυτά με το Weka, τότε πρέπει αρχικά να δημιουργήσουμε ένα αρχείο .arff στην παραπάνω μορφή. Τα δεδομένα που συλλέγονται από εταιρίες που μελετούν στατιστικά δεδομένα και ασχολούνται με τη στατιστική ανάλυση, site όπως η ΕΛΣΤΑΤ, η EUROSTAT κλπ, από τις οποίες μπορούμε να «κατεβάσουμε» δεδομένα σε αρχεία η μορφή των οποίων είναι κατά βάση αρχεία excel. Για να μετατρέψουμε ένα αρχείο excel σε αρχείο arff υπάρχουν κατά βάση δυο τρόποι, ο εύκολος μέσω του weka και ο δύσκολος μέσω ενός κειμενογράφου.

Ο εύκολος τρόπος έχει την εξής ακολουθία βημάτων: Αρχικά πρέπει να μετατρέψουμε το αρχείο από αρχείο με κατάληψη «.xls» σε αρχείο «.csv». Αυτό είναι σχετικά απλό, αρκεί από την επιλογή αποθήκευση ως, να πάμε στην επιλογή «άλλες μορφές αποθήκευσης» και να επιλέξουμε το «CSV οριοθετημένο με κόμματα».



[16] Στα αρχεία .arff του weka, τα αριθμητικά δεδομένα μπορούν να «δηλώνονται» και ως real, integer κλπ.

Στη συνέχεια, ανοίγουμε το Weka και από την επιλογή Tools στο μενού επιλέγουμε το ArffViewer.



3.3 Weka Tools

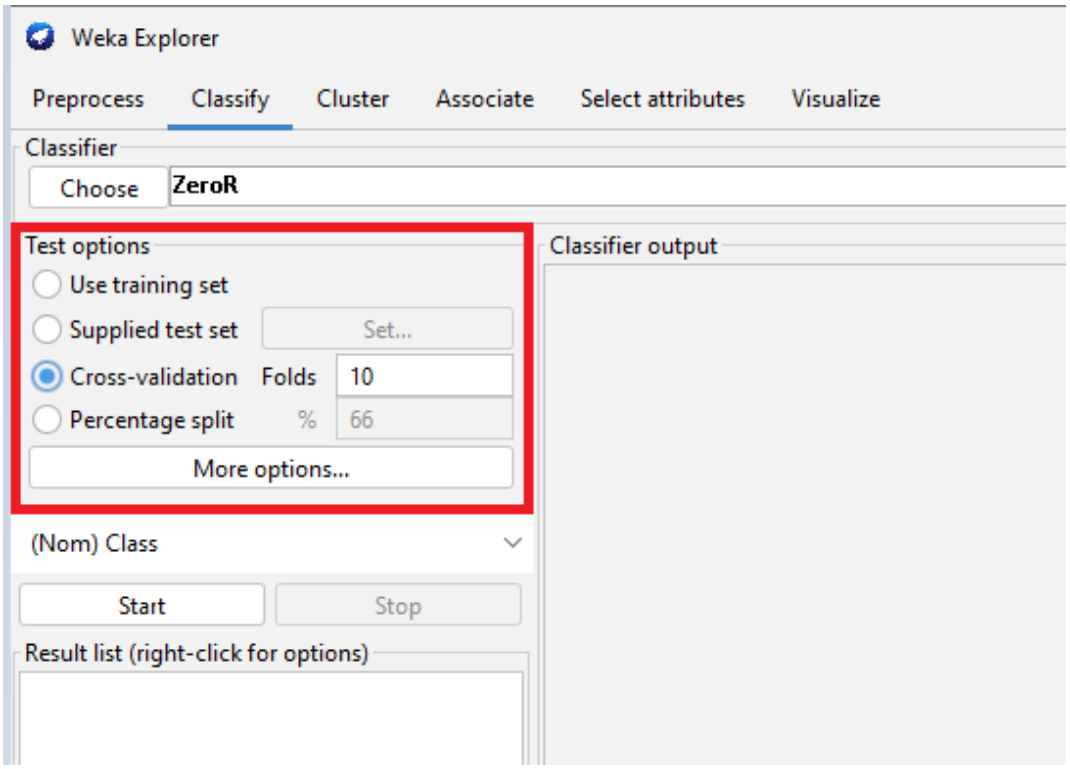
Στη συνέχεια, από το νέο παράθυρο που θα ανοίξει επιλέγουμε το File->Open και φορτώνουμε το αρχείο .csv από τον φάκελο που αποθηκεύτηκε. Στη συνέχεια το αποθηκεύουμε ως αρχείο arff και είναι έτοιμο προς επεξεργασία με το weka. Το όνομα κάθε στήλης γίνεται attribute και τα περιεχόμενα των στηλών δεδομένα.

Ο δεύτερος τρόπος, σε περίπτωση που το αρχείο για παράδειγμα είναι .xlsx και δεν επιτρέπεται η μετατροπή σε CSV, τότε πρέπει να αντιγράψουμε όλα τα αρχεία σε ένα πρόγραμμα κειμένου, πχ το MS word, και να το τροποποιήσουμε έτσι ώστε να έρθει σε μορφή ίδια με τα αρχεία που έχουμε δουλέψει. Αυτό μπορεί να γίνει με πολλαπλή αντικατάσταση των tabs με κόμματα, με δήλωση των attributes κλπ. Δε θα παρουσιαστεί εδώ ο χρονοβόρος αυτός τρόπος, απλά τον αναφέρουμε κυρίως για το τέλος του κεφαλαίου όπου θα δούμε κάποια συμπεράσματα.

3.2 Data Classification

Στο πρώτο κεφάλαιο ασχοληθήκαμε με τους αλγόριθμους εποπτευόμενης μάθησης, στους οποίους τα δεδομένα εισόδου είναι τα attributes και τα δεδομένα σε μορφή instances και σκοπός είναι να προβλέπεται ή να κατηγοριοποιείται η έξοδος. Αφού επιλέγαμε το αρχείο arff με το οποίο θέλαμε να πειραματιστούμε και το ανοίγαμε στο explorer του weka, επιλέγαμε από το μενού στο Classify και στη συνέχεια στο κουμπί Choose επιλέγαμε τον αλγόριθμο που επιθυμούσαμε.

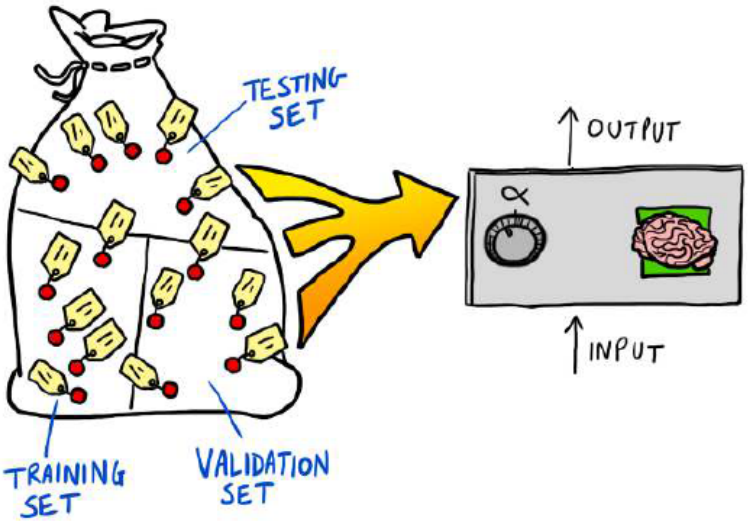
Κάτω από την επιλογή του αλγόριθμου υπάρχει ένα πλαίσιο στο οποίο μπορούμε να επιλέξουμε ανάμεσα σε training set, test set, cross-validation και Percentage split.



3.4 Classify Options

Αν υποθέσουμε ότι το dataset αποτελείται από μεγάλο όγκο δεδομένων τότε καταλαβαίνουμε πως η διαδικασία της μάθησης θα είναι όλο και δυσκολότερη. Αν σκεφτούμε ξανά πως για την εποπτευόμενη μάθηση απαιτούνται δεδομένα τα οποία πρέπει να αντιστοιχούνται σε κατάλληλες εξόδους, τότε το να «εκπαιδεύσουμε» τον αλγόριθμό μας με όλα μας τα δεδομένα θα ήταν μάλλον ανώφελο. Η βασική ιδέα είναι να χωρίζουμε τα δεδομένα μας σε τρεις ενότητες-υποσύνολα. Η ενότητα **training**, όπως προκύπτει από το ρήμα που χρησιμοποιείται, είναι για «προπόνηση» ή καλύτερα θα λέγαμε «εκπαίδευση» του Classifier που θα χρησιμοποιηθεί. Η ενότητα **Cross-validation** χρησιμοποιείται για την «επικύρωση». Στην ουσία πρέπει κατά κάποιο τρόπο τα αποτελέσματά του να συμπίπτουν με αυτά του training αλλά και του test set. Το **test** set είναι ανεξάρτητο από το training set και χρησιμοποιείται ώστε να συγκρίνεται με το training set ως προς το πόσο ταιριάζουν. Εάν ταιριάζουν τότε λέμε ότι έχουμε ελαχιστοποιήσει το Over-fitting ενώ μεγιστοποιείται στην αντίθετη περίπτωση. Η έννοια του Over-fitting ή της υπερπροσαρμογής, είναι μια μαθηματική έννοια που εδώ σχετίζεται με το κατά πόσο κάποια δεδομένα είναι χρήσιμα ώστε να παράγουν πληροφορίες ή με το κατά πόσο είναι περιττά ώστε να δημιουργούν αυτό που παραπάνω ονομάσαμε «θόρυβο»(noise). Τέλος, στο weka υπάρχει η επιλογή **percentage split** η οποία προφανώς μας δίνει τη δυνατότητα να επιλέξουμε το ποσοστό

επί τοις εκατό από το σύνολο των δεδομένων που επιθυμούμε για το classification. Η επόμενη εικόνα περιγράφει τη λογική της διαδικασίας αυτής.

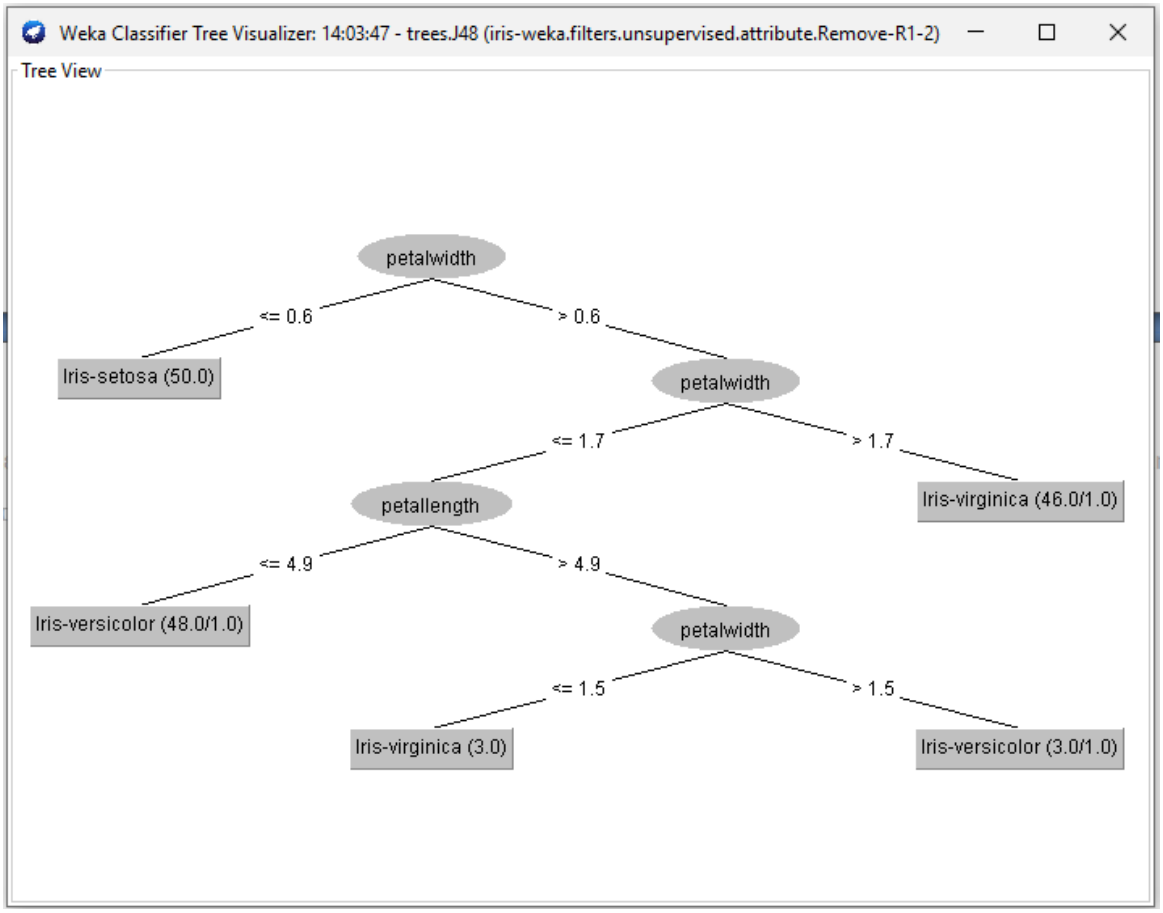


3.5 Διαχωρισμός του DataSet σε Training, Validation & Testing Set

Πηγή: The LION Way [1]

Για την επιλογή Classifier, αρκεί να γνωρίζουμε τα βασικά των αλγόριθμων ώστε να έχουμε μια γενική εικόνα αρχικά, αλλά και μια πιο ειδική γνώση για την περεταίρω πειραματική ανάλυση. Για παράδειγμα, στο πρώτο κεφάλαιο είδαμε πως αλλάζουμε τον αλγόριθμο Nearest Neighbors σε K-Nearest Neighbors, κάνοντας απλά κλικ στον επιλεγμένο αλγόριθμο και από τις επιλογές που εμφανίζονται αλλάζουμε την τιμή του K.

Στην περίπτωση των Αλγόριθμων για τη σχεδίαση δένδρων απόφασης, αφού «τρέξουμε» τον αλγόριθμο επιλέγοντάς τον, μπορούμε από την επιλογή Visualize να εμφανίσουμε και την δενδρική δομή, όπως φαίνεται στην εικόνα 3.6. Εδώ έχει επιλεγεί το dataset iris2D, και ως classifier το J48. Το λογισμικό μας δίνει τη γραφική απεικόνιση του δένδρου και μάλιστα υπάρχει και η δυνατότητα προσαρμογής στο μέγεθος του παραθύρου. Ειδικά για τις δενδρικές δομές όπου η αναπαράσταση του δένδρου είναι ιδιαίτερα χρήσιμη για τον πειραματισμό, όταν π.χ. θέλουμε να συγκρίνουμε έναν αλγόριθμο δένδρων απόφασης με έναν άλλο, το weka αποτελεί πολύ σημαντικό εργαλείο.

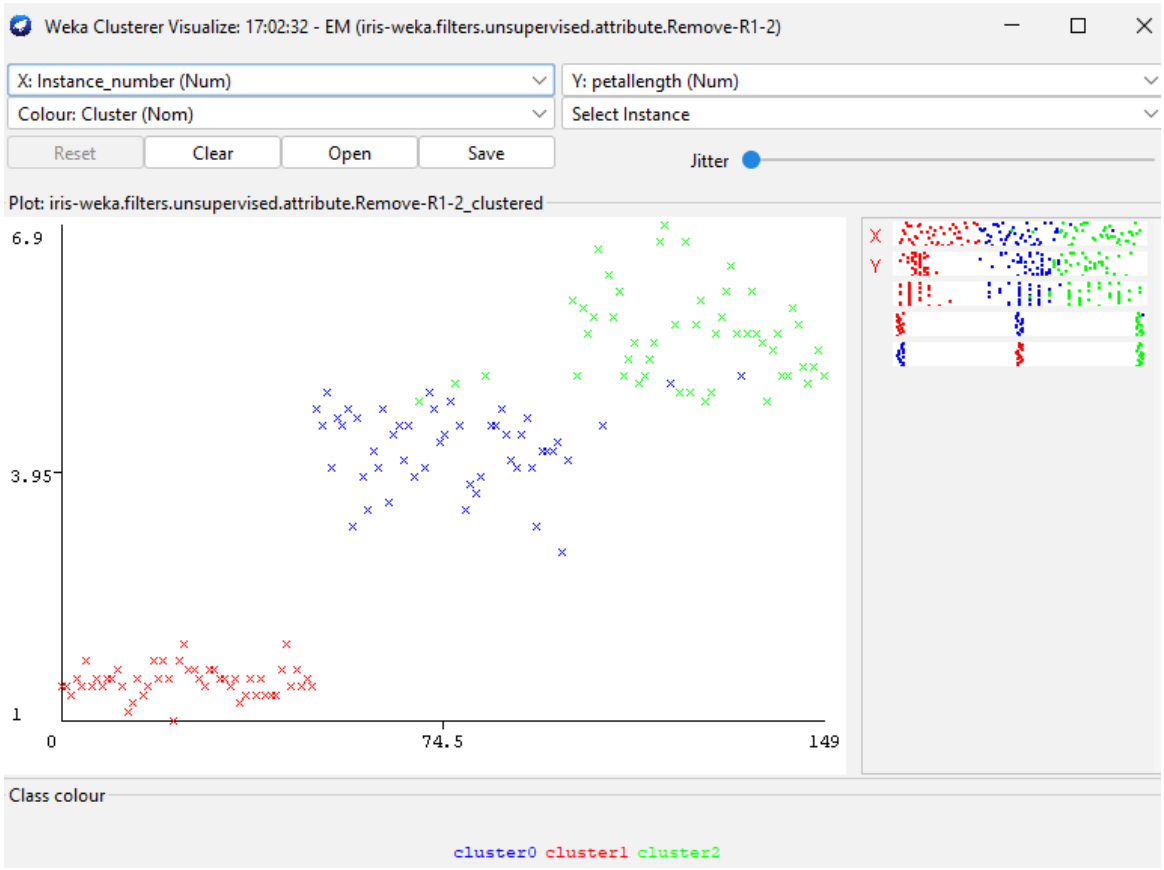


3.6 J48 visualize tree(Iris2D)

3.3 Data Clustering

Στη μάθηση χωρίς επίβλεψη, χρησιμοποιήσαμε την επιλογή Cluster δίπλα από την επιλογή Classify στο Weka Explorer. Στη συνέχεια επιλέγουμε τον Clusterer, δηλαδή τον αλγόριθμο που επιθυμούμε να δουλέψουμε. Να αναφέρουμε ξανά πως πρέπει στα έτοιμα datasets του weka θα πρέπει να επιλέγουμε τη διαγραφή του Class attribute, ή αφού επιλέξουμε τον Clusterer να κάνουμε κλικ στο “ignore attributes” πριν να τρέξουμε τον αλγόριθμο.

Στον χρήστη δίνεται η επιλογή να επιλέγει τον αριθμό των clusters που θα δημιουργούνται κατά την εκτέλεση καθώς και τον αριθμό των επαναλήψεων(iterations) εκτέλεσής του ώστε να μπορεί να πειραματιστεί. Φυσικά κι εδώ υπάρχει η δυνατότητα της οπτικοποίησης του Cluster από την επιλογή Visualize Cluster Assignments. Η οπτικοποίηση γίνεται σε ένα περιβάλλον δυο διαστάσεων, μάλιστα τα clusters διαχωρίζονται με διαφορετικό χρώμα ώστε να είναι πιο εύκολο να αντιληφθούμε τον τρόπο λειτουργίας των αλγόριθμων που χρησιμοποιούμε.



3.7 Cluster Visualization

3.4 Συμπεράσματα

Για κάποιον που επιθυμεί να ασχοληθεί με την μηχανική μάθηση, σε θεωρητικό και πειραματικό επίπεδο, το weka είναι ένα πολύ εύχρηστο εργαλείο. Ακόμα και σε εκπαιδευτικά βίντεο της google τα οποία αναφέρονται στη μηχανική μάθηση, οι προγραμματιστές που τα παρουσιάζουν αναφέρουν πως το πρώτο εργαλείο που χρησιμοποίησαν είναι το weka. Είναι λογισμικό που μετρά πάνω από 20 χρόνια ζωής, που σημαίνει ότι συντηρείται και χρησιμοποιείται ευρέως εδώ και αρκετά χρόνια. Συνοψίζοντας, θα πρέπει να καταλήξουμε σε κάποια θετικά και κάποια αρνητικά στοιχεία του weka, τα οποία συμπεραίνουμε μετά και από την εκπόνηση της εργασίας.

Στα θετικά, ειπώθηκε η εύκολη εκμάθηση και χρήση του λογισμικού, χάρη στο περιβάλλον εργασίας του weka, η ευκολία στην επιλογή των αλγόριθμων είτε εποπτευόμενης είτε μη εποπτευόμενης μάθησης καθώς και ένα πολύ σημαντικό που δεν αναφέραμε πιο πάνω, τα έτοιμα datasets που είναι εγκατεστημένα στο λογισμικό και μπορούμε να τα «φορτώνουμε» στο weka και να πειραματιζόμαστε με τους αλγόριθμους. Οι αλγόριθμοι ουσιαστικά στο weka μας δίνονται έτοιμοι, κάτι που σημαίνει πως εάν κάποιος δεν τους γνωρίζει καθόλου και παρόλα αυτά

πειραματιστεί μαζί τους, μπορεί να βγάλει συμπεράσματα για αυτούς και μελλοντικά εάν πρέπει να πειραματισθεί σε κάποια πραγματικά δεδομένα, να έχει μια εμπειρική γνώση ώστε να επιλέξει τον κατάλληλο Classifier ή Clusterer ανάλογα με την περίπτωση. Η δυνατότητα του visualization είναι φυσικά πολύ δυνατό εργαλείο ειδικά σε αλγόριθμους που το output τους είναι δενδρικές δομές, όπως φυσικά και τα clusters.

Ως αρνητικά συμπεράσματα θα λέγαμε αρχικά την δυσκολία της δημιουργίας των arff αρχείων. Όπως είδαμε, το να δημιουργηθεί ένα αρχείο dataset για το Weka, πρέπει να ακολουθηθεί μια όχι πάντοτε εύκολη διαδικασία. Αν συνυπολογίσουμε και την περίπτωση του class attribute, τότε ίσως να χρειάζεται και μια εξτρά επεξεργασία π.χ. από το excel, τέτοια ώστε να προκύπτει μια τέτοια ιδιότητα στο dataset. Ο πειραματισμός δηλαδή με κάποια δικά μας δεδομένα, θα καθυστερούσε σημαντικά την πειραματική διαδικασία, χωρίς αυτό να σημαίνει ότι θα ήταν απαγορευτικό. Αρχικά η ιδέα ήταν να πειραματιστούμε με κάποια οικονομικά δεδομένα από στατιστικές μελέτες που έχουν δημοσιευθεί στο διαδίκτυο και ως εκ τούτου, υπήρχε το πρόβλημα της μετατροπής τους σε αρχεία arff. Πέρα από αυτό, ένα άλλο αρνητικό του Weka είναι η «αδυναμία» του στο να χειρίζεται datasets με πολλά instances. Επίσης, σε κάθε περίπτωση το weka δεν έχει κάποιους από τους πιο καινούργιους αλγόριθμους μηχανικής μάθησης, κάτι που δεν επηρεάζει όμως την εργασία αυτή.

4. Συστήματα Βασισμένα σε Κανόνες

Στο κεφάλαιο αυτό θα μελετηθούν τα συστήματα που βασίζονται σε κανόνες. Πιο συγκεκριμένα, θα υλοποιηθεί και θα παρουσιαστεί ένα έμπειρο σύστημα βασισμένο σε κανόνες ή Expert Rule-Based System. Τα συστήματα αυτά αποτελούν μια διαφορετική προσέγγιση στην Τεχνητή Νοημοσύνη από την Μηχανική Μάθηση και όπως το μαρτυρά ο τίτλος, βασίζονται σε υπάρχουσα γνώση και εμπειρία. Με πολύ απλά λόγια, μπορούμε να πούμε πως ένα τέτοιο σύστημα έχει σκοπό να προσομοιώσει κάποιον έμπειρο – ειδικό με κύριο στόχο τη λήψη αποφάσεων. Όπως αναφέρθηκε και στην εισαγωγή του παρόντος εγγράφου, η λήψη αποφάσεων για ένα ιατρικό ζήτημα ή για ένα ζήτημα πρόβλεψης οικολογικής καταστροφής που είναι και επίκαιρο(αν και με τη συχνότητα που συμβαίνουν τα τελευταία χρόνια μόνο επίκαιρο δεν είναι…), την μελέτη κάποιον αστροφυσικών δεδομένων και ούτω ο καθεξής, τότε η μηχανική μάθηση δε θα ήταν σε καμία περίπτωση η βέλτιστη επιλογή. Άλλο το να προσπαθείς να προβλέψεις αν ένας πελάτης που αναζητά ρούχα στο διαδίκτυο και πιθανόν να τον δελεάσουν και μερικά ζευγάρια υποδημάτων και άλλο να προσπαθείς να προβλέψεις αν ένας ασθενής παρουσιάζει ένα χρόνιο σύμπτωμα ή ένα περιοδικό.

Ο τρόπος σχεδιασμού ενός Συστήματος Βασισμένο σε Κανόνες παρουσιάζει σημαντικές διαφορές σε σχέση με τους αλγόριθμους μηχανικής μάθησης κυρίως ως προς τη φιλοσοφία τους. Στην μηχανική μάθηση αυτό που χρειαζόμασταν ήταν μια πληθώρα δεδομένων, κυρίως στατιστικών, τα οποία συνέθεταν το dataset. Τα δεδομένα τα επεξεργαζόταν ένας αλγόριθμος ο οποίος κατά βάση χρησιμοποιεί μαθηματικά μοντέλα υπολογισμών ώστε να καταλήγει σε κάποιο συμπέρασμα ή κάποια πρόβλεψη. Αυτό που μελετήθηκε στο προηγούμενο κεφάλαιο είναι η «ικανότητα» τους να προβλέπουν με ακρίβεια ή όχι, η βέλτιστη επιλογή με βάση τη δομή του dataset και ο πειραματισμός. Αν τα αφήσουμε όλα αυτά στην άκρη, τότε μπορούμε να πούμε πως τα συστήματα που βασίζονται σε κανόνες δεν βασίζονται στα δεδομένα για να παράγουν γνώση αλλά βασίζονται κυρίως στη γνώση για να παράγουν δεδομένα. Αν βασιστούμε στα δεδομένα και στα ποσοστά για να προβλέψουμε αν π.χ. ένας ασθενής πάσχει από μια πολύ σπάνια ασθένεια από την οποία πάσχει συνολικά ένα 0.0001%, τότε ένας αλγόριθμος μηχανικής μάθησης για ένα dataset 1000 ατόμων, θα προέβλεπε με ακρίβεια 100% ότι ένας νέος ασθενής δεν έχει την σπάνια ασθένεια αυτή. Ένα σύστημα όμως βασισμένο στη γνώση και σε κανόνες, μάλλον αδιαφορεί εντελώς για το ποσοστό νοσούντων και ενδιαφέρεται για τα απτά δεδομένα, δηλαδή τα συμπτώματα(ίσως), το ιστορικό του ασθενούς κ.λπ.

4.1 Δομώντας Ένα Expert Rule – Based System

Όταν επισκεπτόμαστε έναν ειδικό, έναν εμπειρογνώμονα, για παράδειγμα έναν μηχανικό αυτοκινήτων, του αναφέρουμε τα προβλήματα που αντιμετωπίζουμε με το αυτοκίνητό μας, πραγματοποιεί έναν διαγνωστικό έλεγχο και ο ίδιος και μας ενημερώνει για την βλάβη ή την πιθανή βλάβη, γιατί υπάρχει πάντοτε και η πιθανότητα εσφαλμένου συμπεράσματος. Στη συνέχεια λαμβάνει αποφάσεις σχετικά με την επιδιόρθωσή της, έτσι ώστε το όχημα να είναι και πάλι λειτουργικό. Όλη αυτή η διαδικασία που περιγράφηκε, προφανώς προϋποθέτει τη γνώση και την εμπειρία του μηχανικού, την διαδικασία της διάγνωσης και του ελέγχου όπου είναι πολύ σημαντική εδώ και, στο τέλος, τη λήψη της απόφασης η οποία εξαρτάται από το συμπέρασμα που βγήκε από τη διάγνωση. Ας κρατήσουμε τους όρους δεδομένα - γνώση - πληροφορία. Ο τρόπος που ένας ειδικός φθάνει στην επίλυση ενός προβλήματος είναι και ο τρόπος με τον οποίο δομείται ένα έμπειρο, βασισμένο σε κανόνες σύστημα.

4.1.1 Long-term & sort-term Memory

Οι έννοιες μακροχρόνια και βραχυχρόνια μνήμη στην καθημερινότητά μας έχουν να κάνουν με πληροφορίες που αποθηκεύονται στη μνήμη μας και παραμένουν εκεί για χρόνια ή και δεκαετίες, αλλά και σε αυτές που παραμένουν για σύντομο χρονικό διάστημα. Αυτό που καθορίζει τη διάρκεια αυτή είναι το πόσο σημαντική είναι η πληροφορία αυτή και πόσο καθοριστικό ρόλο θα παίξει στο μέλλον. Για παράδειγμα ένα φαγητό που δοκιμάσαμε σε ένα εστιατόριο μπορεί να το ξεχάσουμε σε λίγες ημέρες αλλά μια συνταγή μαγειρικής ενός φαγητού που θα το μαγειρέψουμε εκατοντάδες φορές θα την αποστηθίσουμε ίσως και την πρώτη φορά. Μπορούμε να πούμε πως η μακροχρόνια μνήμη είναι η γνώση που αποκτήσαμε ενώ η βραχυχρόνια είναι οι πληροφορίες που λαμβάνουμε από κάποια γεγονότα, ή ακόμα και όταν χρησιμοποιούμε τη γνώση μας για να παράξουμε πληροφορίες.

Για ένα έμπειρο σύστημα οι δυο αυτές έννοιες είναι βασικές. Σε όρους πληροφορικής η βραχυχρόνια μνήμη (sort-term memory) ή αλλιώς και «μνήμη εργασίας [5]» είναι αυτή η οποία αναπαρίσταται με μια βάση δεδομένων. Οι βάσεις δεδομένων είναι εύκολα διαχειρίσιμες με τα εργαλεία που θα παρουσιασθούν στη συνέχεια και «αλληλοεπιδρούν» με το έμπειρο σύστημα όπως φαίνεται στην παρακάτω εικόνα(4.1). Αντίθετα, η μακροχρόνια μνήμη (long – term memory) είναι στην ουσία το σύστημα κανόνων. Αναφέρεται και ως «παραγωγική μνήμη» ή «βάση κανόνων» και ως εκ τούτου αποτελείται από τους κανόνες του συστήματος. Εδώ προφανώς δεν υπάρχει αλληλεπίδραση με το σύστημα, υπό την έννοια ότι το σύστημα δεν επηρεάζει τη μνήμη αυτή, δεν αλλάζει κάποιους κανόνες και δεν αποθηκεύει σε αυτή κάποια συμπεράσματα γιατί εννοείται ότι τα συμπεράσματα αυτά προϋπάρχουν(δεν κάνουμε μηχανική μάθηση, υπενθυμίζω).

4.1.2 Μηχανή Εξαγωγής Συμπερασμάτων

Η δύο προαναφερθείσες μνήμες επικοινωνούν με έναν μηχανισμό ο οποίος είναι υπεύθυνος για την εξαγωγή Συμπερασμάτων. Ας μην ξεχνάμε πως τα έμπειρα συστήματα έχουν αυτό το στόχο, το να εξαγάγουν συμπεράσματα παρόμοια με έναν εμπειρογνώμονα. Στην εικόνα 4.1 παρουσιάζεται γραφικά ένα σύστημα βασισμένο σε κανόνες, όπου φαίνεται και ο ρόλος που έχουν οι μνήμες που αναφέραμε σχετικά με τον μηχανισμό που εξάγει τα συμπεράσματα. Παρατηρούμε πως από την sort – term memory υπάρχουν κατευθυνόμενα βέλη από και προς αυτήν ενώ από την long – term memory μόνο από αυτή προς τη «Συμπερασματική Μηχανή» [5]. Κανένα γεγονός δεν πρέπει να αλλάζει τους κανόνες σε ένα έμπειρο σύστημα.

4.1 Ένα Expert Rule-based System

Πηγή: http://repfiles.kallipos.gr/html_books/93/05a-main.html#_idTextAnchor123

Οι κανόνες από την πλευρά τους είναι το αποτέλεσμα της γνώσης και σε ένα σύστημα όπως το παραπάνω, αποτελούνται από δύο «συστατικά» ή δύο μέρη ώστε να παραχθεί το συμπέρασμα. Το πρώτο μέρος είναι η συνθήκη, το ΑΝ, γνωστό από την εισαγωγή ενός μαθητή – φοιτητή στον κόσμο του προγραμματισμού. Οι συνθήκες είναι λογικές εκφράσεις όπως «πονάει το πόδι», «έχει άσχημο καιρό», «η αρτηριακή πίεση είναι υψηλή» κλπ. ακολουθούμενες από έναν υποθετικό σύνδεσμο (Αν, όσο, μέχρι κλπ.). Το δεύτερο μέρος είναι η ενέργεια που πρέπει να ακολουθηθεί, δηλαδή το ΤΟΤΕ. Για παράδειγμα ΑΝ «έξω βρέχει» ΤΟΤΕ «πάρε ομπρέλα». Η λογική έκφραση είναι το «έξω βρέχει», η συνθήκη είναι η «ΑΝ έξω βρέχει» και η ενέργεια είναι το «ΤΟΤΕ πάρε ομπρέλα».

Είναι προφανές ότι εδώ προσομοιώνουμε τον ανθρώπινο συλλογισμό απλά και κατανοητά και μάλιστα με έναν τρόπο που είναι γνώριμος από τα πρώτα βήματα εκμάθησης του προγραμματισμού υπολογιστών και της αλγοριθμικής σκέψης. Εν συντομία δηλαδή και έχοντας

ως παράδειγμα την εικόνα 4.1, με βάση τα δεδομένα που εισάγονται στη «μνήμη εργασίας» και τους κανόνες που υφίστανται στη «βάση κανόνων», η μηχανή είναι υπεύθυνη να εξαγάγει συμπεράσματα με τη λογική του συλλογισμού IF … THEN … Με την υλοποίηση ενός απλού expert rule – based system που παρουσιάζεται παρακάτω, γίνεται αντιληπτό ότι η δυσκολία δεν έγκειται στη δημιουργία του συστήματος, της μνήμης ή καλύτερα του τρόπου αποθήκευσης των δεδομένων(εισαγωγής και εξαγωγής, δηλαδή των συμπερασμάτων), αλλά αυτό που είναι απαραίτητο είναι η γνώση ενός ειδικού με βάση την οποία θα σχεδιαστούν οι κανόνες και τα μοντέλα AN … TOTE.

Φυσικά, υπάρχουν πολύ πιο πολύπλοκα συστήματα από αυτό που θα παρουσιαστεί. Συστήματα που πρέπει να διαχειρίζονται περιπτώσεις αβεβαιότητας, περιπτώσεις ασάφειας όσον αφορά τα δεδομένα αλλά και τους κανόνες, τα οποία είναι σαφώς πιο δύσκολα στην υλοποίηση αλλά και στην ανεύρεση κανόνων. Σκοπός της ενότητας αυτής, αλλά και της εργασίας στο σύνολό της, είναι μια εισαγωγή στους μηχανισμούς που βρίσκονται πίσω από την τεχνητή νοημοσύνη και στην «απομυθοποίησή» της.

4.2 Δημιουργία ενός απλού Expert Rule Based System

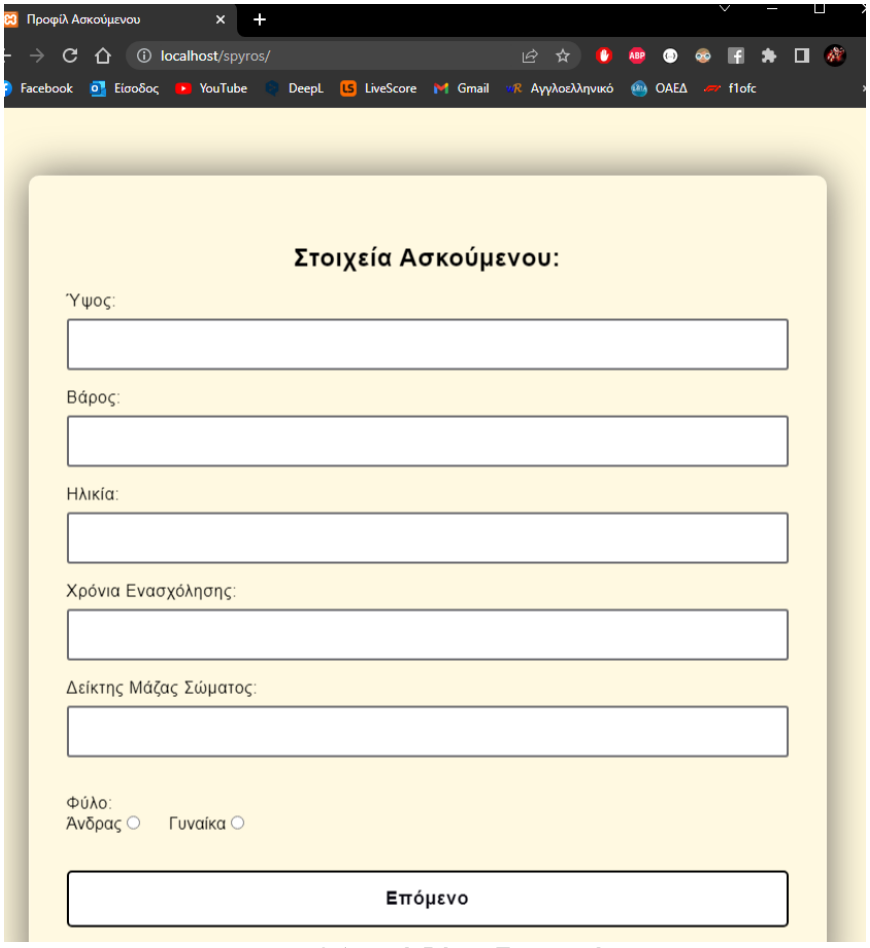
Το έμπειρο, βασισμένο σε κανόνες σύστημα που υλοποιήθηκε για την εργασία, είναι μια web εφαρμογή η οποία σχεδιάστηκε με τα εργαλεία λογισμικού xampp, visual code και τον browser chrome της google και θα παρουσιαστούν στην επόμενη ενότητα. Το σύστημα αυτό έχει στόχο την λήψη απόφασης και την ενημέρωση του χρήστη – αθλητή, στην περίπτωση κάποιου μυοσκελετικού πόνου ή ενόχλησης σε κάποιον μυ του σώματος. Ο χρήστης εισάγει αρχικά κάποιες πληροφορίες σχετικά με αυτόν και μετά απαντά σε κάποιες ερωτήσεις μέσω επιλογών ή radio buttons. Σκοπός του συστήματος αυτού είναι να εξάγει το συμπέρασμα σχετικά με δυο έννοιες, την έννοια του ευερέθιστου και της δριμύτητας. Δηλαδή, συμπεραίνει σχετικά με το αν ο μυϊκός πόνος προκύπτει λόγο κάποιας ευαισθησίας του χρήστη – ασθενή και επίσης κάποιο συμπέρασμα που σχετίζεται με τη δριμύτητα του πόνου, πόσο ισχυρός δηλαδή είναι και αν επηρεάζει και άλλες δραστηριότητες.

Με τον συνδυασμό των παραπάνω, το σύστημα ενημερώνει – υποδεικνύει στον χρήστη σχετικά με το τι πρέπει να κάνει, πως πρέπει να δράσει, αν πρέπει να διακόψει την άσκηση, αν πρέπει να επισκεφθεί ειδικό και λοιπά. Προγραμματιστικά, με τη χρήση της php όλα αυτά είναι ένας απλός κώδικας με συνθήκες if που υλοποιούν το έργο αυτό, όπως παρουσιάστηκε στην ενότητα 4.1.2. Μπορούμε να πούμε πως η συμπερασματική μηχανή υλοποιήθηκε με κώδικα php και πως η sort term memory είναι τα στοιχεία που εισάγει ο χρήστης τα οποία αποθηκεύονται στη βάση δεδομένων που υλοποιήθηκε για την εφαρμογή αυτή. Η long term memory μάλλον δεν έχει

τη θέση που θα έπρεπε, καθώς όπως περιγράφεται στην εικόνα 4.1, θα ήταν πιο σωστό να υπάρχουν σε μια βάση δεδομένων. Όπως και να χει, σε περίπτωση αλλαγής κάποιων κανόνων, θα απαιτούταν το χέρι κάποιου προγραμματιστή για να γίνει η αλλαγή αυτή, άρα θα αρκεστούμε σε αυτή την υλοποίηση που είναι λίγο πιο απλή και οι κανόνες αποθηκεύονται σε μεταβλητές συμβολοσειρών στην php οι οποίες ενημερώνονται σχετικά με τους κανόνες υπολογισμών για το ευερέθιστο και τη δριμύτητα.

4.2.1 Παρουσίαση της Εφαρμογής

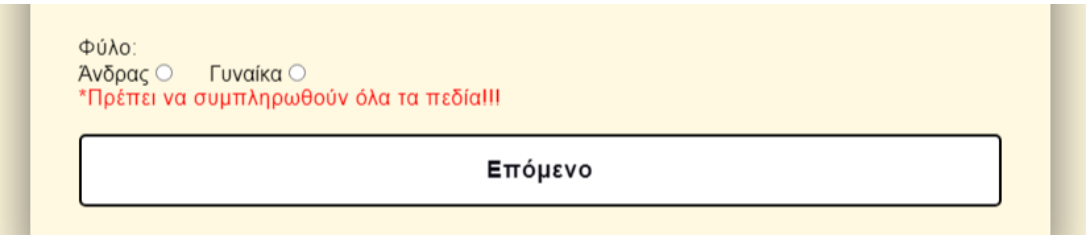
Όπως αναφέρθηκε, η εφαρμογή υλοποιήθηκε με τη βοήθεια του λογισμικού xampp, το οποίο εγκαθιστά έναν τοπικό server στον υπολογιστή μας που σημαίνει πως η εφαρμογή «τρέχει» τοπικά. Η παρουσίαση λεπτομερειών θα γίνει σε επόμενη ενότητα, οπότε εδώ αρκεί να αναφέρουμε ότι στη διεύθυνση localhost ανοίγοντας τον browser μας και με το xampp σε λειτουργία έχουμε αυτή τη φόρμα:



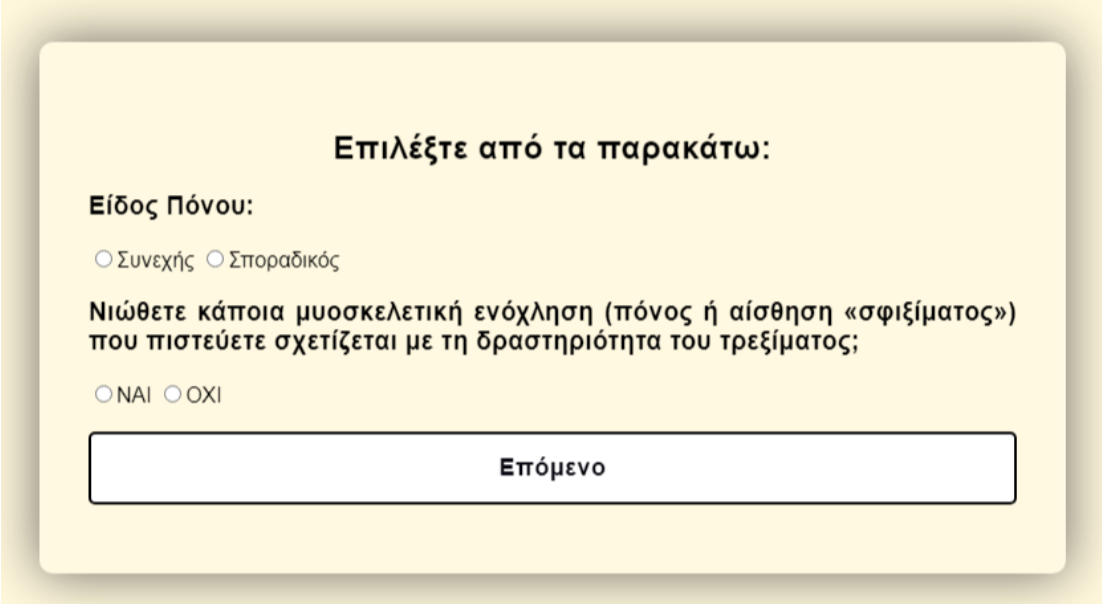
4.2 Αρχική Φόρμα Εφαρμογής

Ο χρήστης – ασκούμενος, πρέπει να εισάγει κάποια βασικά στοιχεία σχετικά με αυτόν, όπως φαίνονται στην εικόνα 4.2. Σε περίπτωση που κάποιο από τα πεδία δεν έχει συμπληρωθεί και

επιχειρήσει να πατήσει στο Επόμενο, εμφανίζεται ένα προειδοποιητικό μήνυμα το οποίο τον προτρέπει να συμπληρώσει όλα τα πεδία. Προειδοποιητικό μήνυμα υπάρχει σε όλες τις σελίδες της εφαρμογής.

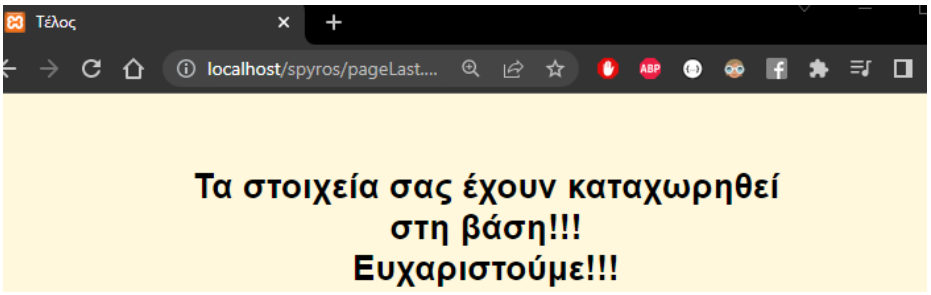


Σε περίπτωση που όλα τα πεδία έχουν συμπληρωθεί, τότε πατώντας στο επόμενο, κατευθυνόμαστε στη δεύτερη σελίδα, όπου πλέον ο χρήστης εισάγει τα σχετικά με τον πόνο ή ενόχληση που νιώθει.



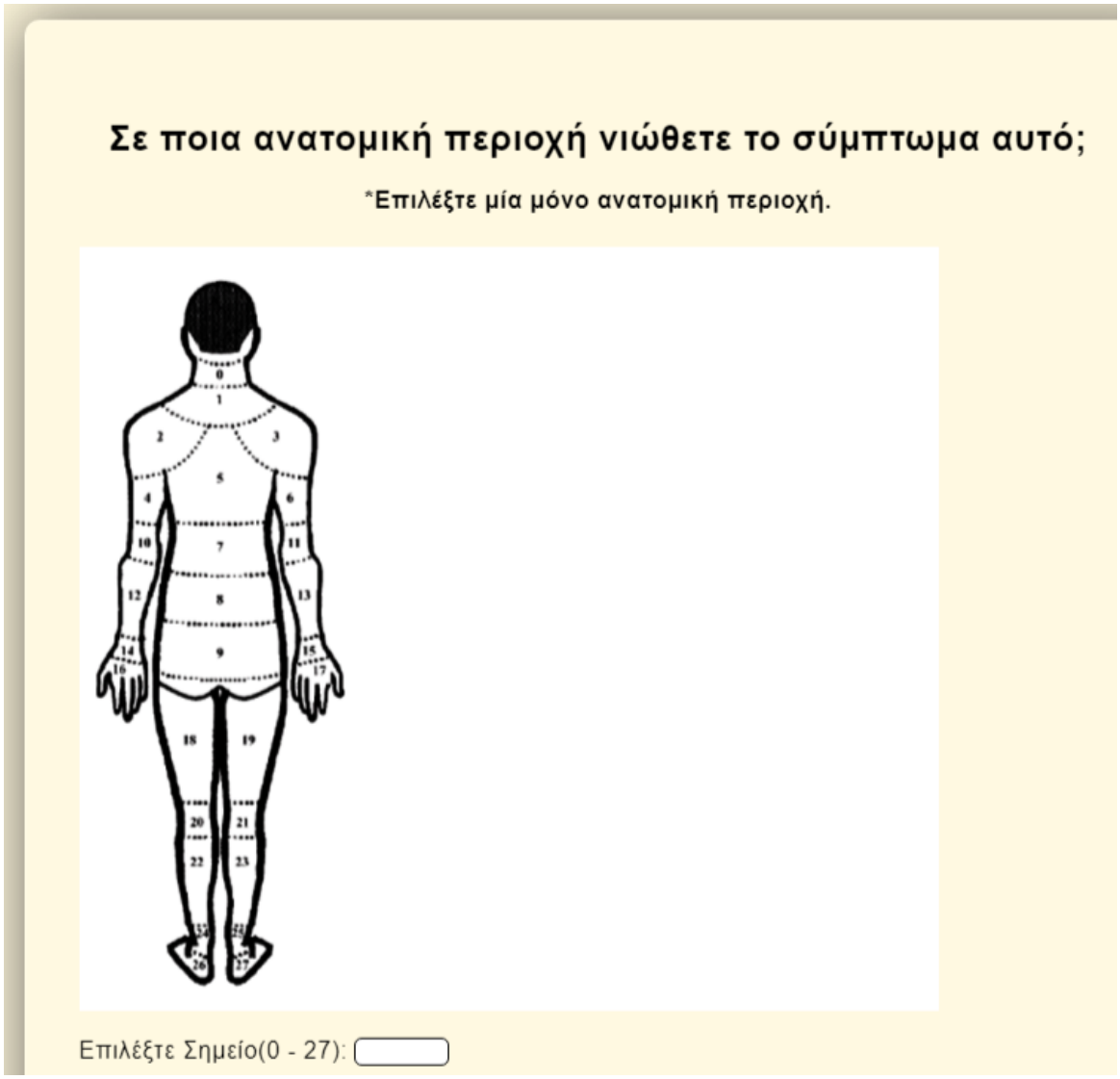
4.3 Δεύτερη σελίδα της Εφαρμογής

Αρχικά ο χρήστης επιλέγει σχετικά με το αν ο πόνος είναι συνεχής ή σποραδικός και μετά, το πιο σημαντικό, αν σχετίζεται με την αθλητική του δραστηριότητα. Το πιο σημαντικό από αυτά τα δυο radio buttons είναι το δεύτερο. Εάν ο χρήστης – ασκούμενος επιλέξει το όχι, η εφαρμογή φθάνει στο τέλος της. Αυτό σημαίνει πως το expert rule-based system σχετίζεται με πόνους που προκύπτουν από κάποια αθλητική δραστηριότητα. Σε περίπτωση που δεν υπάρχει αυτή η συσχέτιση ή, πιο απλά, δεν υπάρχει κάποιος πόνος. Στην περίπτωση που υπάρχει, τότε προφανώς πρόκειται για κάποιο είδος πόνου που σχετίζεται με κάποιο άλλο παθολογικό αίτιο και όχι με την όποια αθλητική δραστηριότητα υπάρχει. Η σελίδα στην οποία οδηγείται η εφαρμογή στην περίπτωση επιλογής του ΟΧΙ στη δεύτερη ερώτηση είναι αυτή της εικόνας 4.4.



4.4 Τελική σελίδα 1

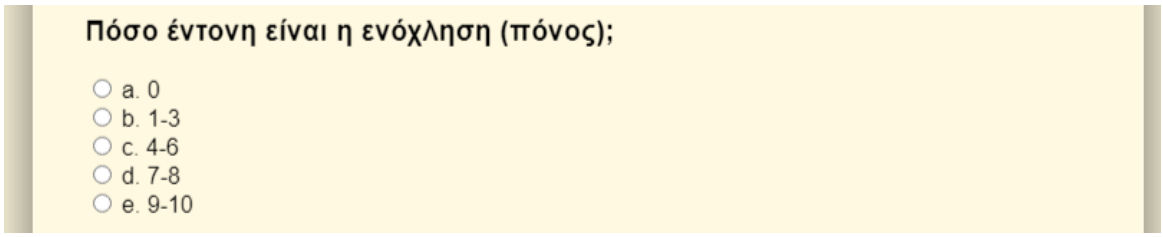
Τα στοιχεία του ασκούμενου, σε κάθε περίπτωση πρέπει να αποθηκεύονται σε μια βάση δεδομένων. Περισσότερα για τη βάση αυτή θα αναλυθούν στη συνέχεια. Το μήνυμα της σελίδας αυτής ενημερώνει το χρήστη σχετικά με τον τερματισμό της, μιας και δεν υπάρχει κάποιο προειδοποιητικό μήνυμα προς αυτόν σε αντίθεση με αυτό που θα συνέβαινε εάν είχε επιλέξει το ΝΑΙ στη δεύτερη ερώτηση. Στην περίπτωση αυτή η εφαρμογή οδηγείται σε επόμενη σελίδα με φόρμα επιλογών, όπως φαίνεται στην εικόνα 4.5



4.5 Επιλογή σημείου όπου παρουσιάζεται ο πόνος

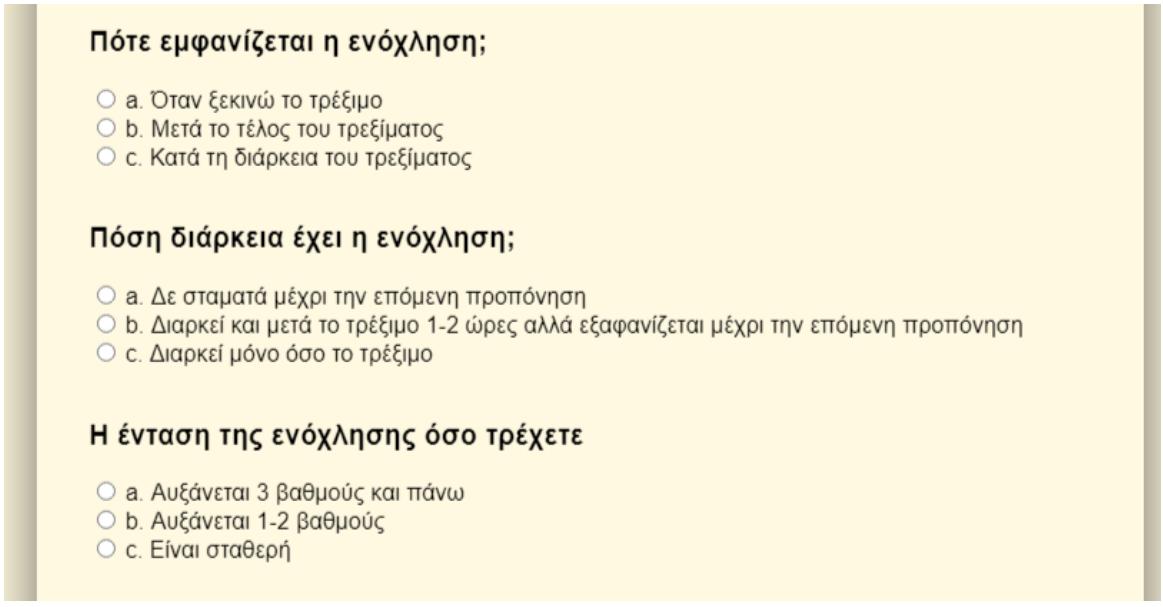
Αρχικά ο χρήστης πρέπει να επιλέξει το σημείο που εμφανίζεται ο πόνος. Αυτό μπορεί να το κάνει με δυο τρόπους, ο πρώτος είναι να βάλει έναν αριθμό από το 0 έως και το 27 στο σημείο κάτω από την εικόνα και δίπλα από τη λεζάντα και ο δεύτερος είναι να επιλέξει με απλό κλικ μια περιοχή πάνω στην εικόνα. Δεν επιτρέπεται να εισαχθεί αριθμός μικρότερος του μηδενός ή μεγαλύτερος του 27. Σε τέτοια περίπτωση, εμφανίζεται ένα μήνυμα το οποίο προτρέπει τον χρήστη να εισάγει ξανά νέο αριθμό.

Η επόμενη ερώτηση της φόρμας, η οποία είναι και αρκετά σημαντική για τους κανόνες υπολογισμού είναι αυτή της εικόνας 4.6 και αντιπροσωπεύει τον συντελεστή της έντασης του πόνου. Η ένταση περιγράφεται από μια κλίμακα, από το 0 μέχρι και το 10, όπου 0 είναι η μικρότερη ενόχληση και 10 η πιο ισχυρή. Εμείς θεωρούμε τις επιλογές a και b ως πόνο ΧΑΜΗΛΗΣ έντασης, την επιλογή c ως ΜΕΤΡΙΑΣ έντασης και τις επιλογές d, e ως ΥΨΗΛΗΣ. Η ένταση του πόνο σχετίζεται κατά πολύ με τους κανόνες υπολογισμού του συστήματός μας και φυσικά με την εξαγωγή των συμπερασμάτων που θα παρουσιαστούν στον χρήστη.



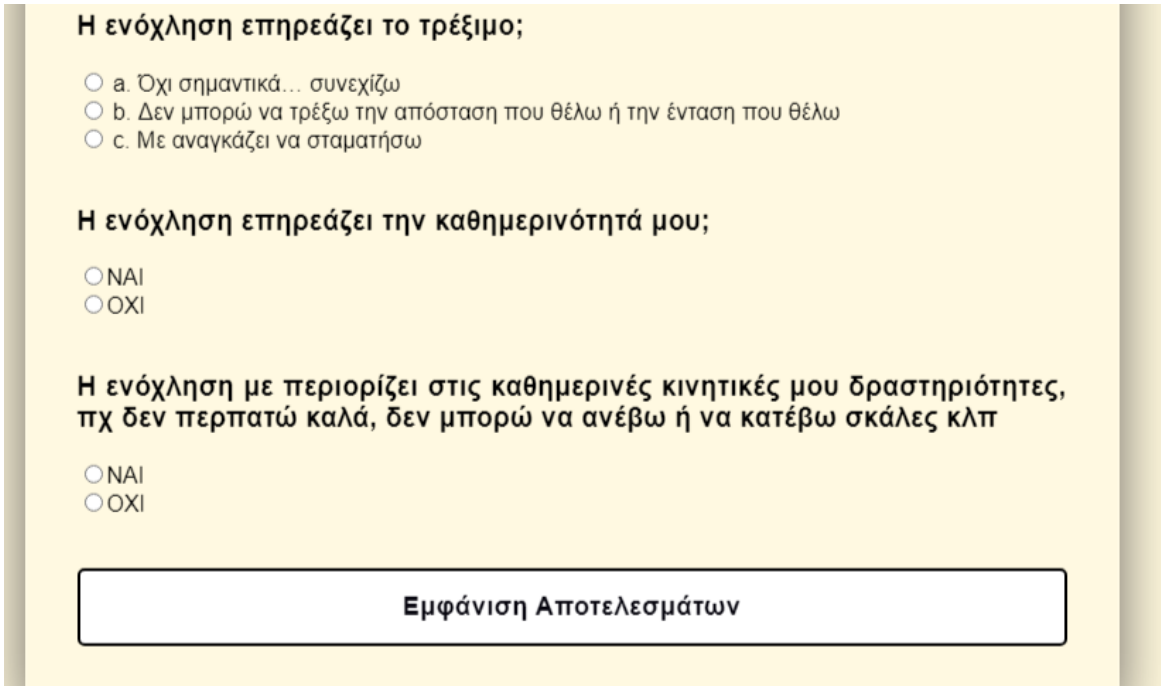
4.6 Ερώτηση που σχετίζεται με την ένταση του πόνου

Οι επόμενες τρεις ερωτήσεις σχετίζονται με τον κανόνα υπολογισμού μιας παραμέτρου που την ονομάζουμε ΕΥΕΡΕΘΙΣΤΟ, δηλαδή με το πόσο εύκολα ερεθίζεται, προκαλείται ο πόνος – ενόχληση στην συγκεκριμένη περιοχή.



4.7 Ερωτήσεις για τον κανόνα υπολογισμού ΕΥΕΡΕΘΙΣΤΟ

Αφού ο χρήστης επιλέξει την κατάλληλη απάντηση, μένουν τρεις ακόμα ερωτήσεις ο οποίες σχετίζονται με τον κανόνα υπολογισμού της ΔΡΙΜΥΤΗΤΑΣ. Αφού επιλεχθούν όλα, μπορεί να υποβάλει τη φόρμα πατώντας στο κουμπί «Εμφάνιση Αποτελεσμάτων».



4.8 Ερωτήσεις για τον κανόνα υπολογισμού ΔΡΥΜΙΤΗΤΑ

Όπως και στις προηγούμενες σελίδες, έτσι κι εδώ ο χρήστης πρέπει να συμπληρώσει το πεδίο που αντιστοιχεί στο σημείο πόνου και να επιλέξει μια επιλογή από όλες τις ερωτήσεις διότι σε διαφορετική περίπτωση θα εμφανιστεί μήνυμα που θα τον προτρέψει να συμπληρώσει όλα τα πεδία για να προχωρήσει στα αποτελέσματα.

Η εμφάνιση των αποτελεσμάτων είναι στην ουσία αυτό που έπεται μετά το “IF … THEN”. Δηλαδή, οι κανόνες υπολογισμού των παραμέτρων ΕΥΕΡΕΘΙΣΤΟ και ΔΡΙΜΥΤΗΤΑ που υπολογίζονται με βάση το τι θα επιλέξει ο χρήστης από τη φόρμα, είναι αυτές που οδηγούν το σύστημα στο να εμφανίζει το κατάλληλο αποτέλεσμα. Οι κανόνες υπολογισμών για τα παραπάνω παρουσιάζονται στον επόμενο πίνακα:

Κανόνες Υπολογισμών για ΕΥΕΡΕΘΙΣΤΟ και ΔΡΙΜΥΤΗΤΑ	
ΕΥΕΡΕΘΙΣΤΟ	**ΔΡΙΜΥΤΗΤΑ**
- Εάν ο χρήστης απαντήσει σε όλα **Α** τότε είναι **ΥΨΗΛΟ** - Εάν ο χρήστης απαντήσει σε όλα **C** τότε είναι **ΧΑΜΗΛΟ** - Εάν κάνει οποιοδήποτε άλλο συνδυασμό τότε είναι **ΜΕΤΡΙΟ**	- Εάν ο χρήστης απαντήσει **Α** στη πρώτη ερώτηση και **ΟΧΙ**, **ΟΧΙ** στις άλλες δύο τότε είναι **ΧΑΜΗΛΟ** - Εάν ο χρήστης απαντήσει **Β** στη πρώτη ερώτηση και **ΟΧΙ**, **ΟΧΙ** στις άλλες δύο τότε είναι **ΜΕΤΡΙΟ** - Εάν κάνει οποιοδήποτε άλλο συνδυασμό τότε είναι **ΥΨΗΛΟ**

Πίνακας 2 Κανόνες Υπολογισμών

Όλοι αυτοί οι υπολογισμοί πρέπει να συνδυαστούν έτσι ώστε να αποτυπωθεί και να εμφανισθεί στον χρήστη το τελικό αποτέλεσμα. Η προτροπή – συμβουλή δηλαδή που προκύπτει από τα παραπάνω και προσομοιώνει τη συμβουλή ενός ειδικού. Ο τρόπος υπολογισμού του αποτελέσματος είναι παρόμοιος με τον τρόπο που υπολογίζονται οι παράμετροι ΕΥΕΡΕΘΙΣΤΟ και ΔΡΙΜΥΤΗΤΑ, δηλαδή βάσει κανόνων με την λογική "IF… THEN". Υπάρχουν 7 συνδυαστικοί κανόνες και παρουσιάζονται στον πίνακα που ακολουθεί.

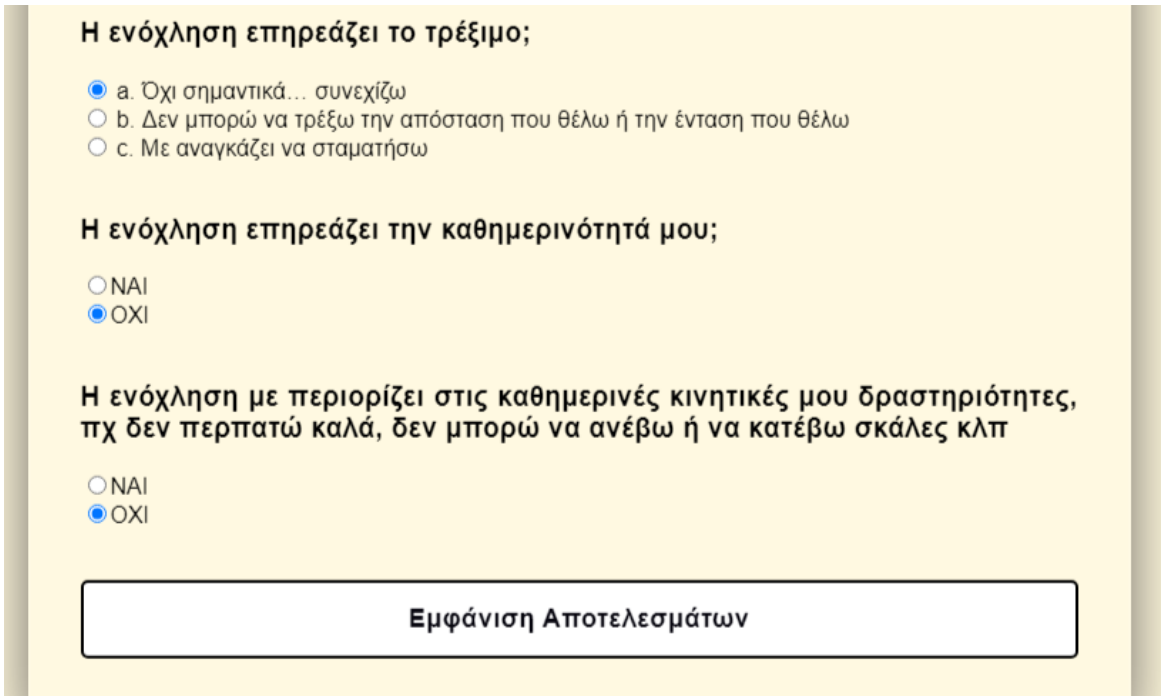
ΣΥΝΔΥΑΣΤΙΚΟΙ ΚΑΝΟΝΕΣ ΑΞΙΟΛΟΓΗΣΗΣ
1. Εάν **ΠΟΝΟΣ** ΧΑΜΗΛΟ και **ΔΡΙΜΥΤΗΤΑ** ΧΑΜΗΛΟ τότε Μειώστε τον φόρτο άσκησης για μία εβδομάδα
2. Εάν **ΠΟΝΟΣ** ΧΑΜΗΛΟ και **ΔΡΙΜΥΤΗΤΑ** ΜΕΤΡΙΟ τότε Συνίσταται διακοπή για τουλάχιστον μία εβδομάδα και στη συνέχεια Ασκήσεις
3. Εάν **ΠΟΝΟΣ** ΧΑΜΗΛΟ και **ΔΡΙΜΥΤΗΤΑ** ΥΨΗΛΟ τότε Συνίσταται διακοπή για τουλάχιστον δύο εβδομάδες
4. Εάν **ΠΟΝΟΣ** ΜΕΤΡΙΟ και **ΔΡΙΜΥΤΗΤΑ** ΧΑΜΗΛΟ τότε Μειώστε τον φόρτο άσκησης για δύο εβδομάδες
5. Εάν **ΠΟΝΟΣ** ΜΕΤΡΙΟ και **ΔΡΙΜΥΤΗΤΑ** ΜΕΤΡΙΟ τότε Συνίσταται διακοπή για τουλάχιστον δύο εβδομάδες και στη συνέχεια Ασκήσεις
6. Εάν **ΠΟΝΟΣ** ΜΕΤΡΙΟ και **ΔΡΙΜΥΤΗΤΑ** ΥΨΗΛΟ τότε Συνίσταται διακοπή για τουλάχιστον τέσσερις εβδομάδες
7. Εάν **ΠΟΝΟΣ** ΥΨΗΛΟ τότε Σταματήστε όλες τις ασκήσεις και ζητήστε ιατρική συμβουλή

Πίνακας 3 Συνδυαστικοί Κανόνες Υπολογισμού Αποτελέσματος

Όλα τα παραπάνω υπολογίζονται με απλές εντολές στην php, με τη χρήση της δομής if και μπορούμε να πούμε πως αποτελεί την συμπερασματική μηχανή στην εφαρμογή μας. Μετά από πολλές δοκιμές ώστε να βεβαιωθούμε πως η εφαρμογή λειτουργεί σωστά και σύμφωνα με τις προδιαγραφές και μετά από σχετικές διορθώσεις, θα παρουσιάσουμε μερικά αποτελέσματα και στη συνέχεια θα αναφερθούμε στον κώδικα αλλά και στη βάση δεδομένων.

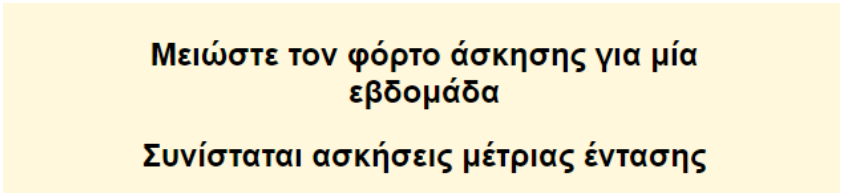
Παράδειγμα $1^{ο}$: Επιθυμούμε να εμφανισθεί το μήνυμα «Μειώστε τον φόρτο άσκησης για μία εβδομάδα» τότε θα πρέπει ο συντελεστής ΠΟΝΟΣ να είναι ΧΑΜΗΛΟ όπως επίσης και η ΔΡΙΜΥΤΗΤΑ χωρίς ακόμα να μας ενδιαφέρει το ΕΥΕΡΕΘΙΣΤΟ το οποίο θα παρουσιαστεί στη συνέχεια. Για να εμφανισθεί το μήνυμα αυτό θα πρέπει αρχικά στη δεύτερη σελίδα ο χρήστης να έχει επιλέξει το ΝΑΙ στην ερώτηση «Νιώθετε κάποια μυοσκελετική ενόχληση (πόνος ή αίσθηση «σφιξίματος») που πιστεύετε σχετίζεται με τη δραστηριότητα του τρεξίματος;» ώστε να τον οδηγήσει στην Τρίτη σελίδα και στη συνέχεια στην ερώτηση «Πόσο έντονη είναι η ενόχληση (πόνος);» ο χρήστης να επιλέξει ένα από τα a και b. Για να προκύψει ΧΑΜΗΛΟ στη

ΔΡΙΜΥΤΗΤΑ πρέπει στις δυο τελευταίες ερωτήσεις να επιλέξει Α, ΟΧΙ, ΟΧΙ όπως φαίνεται στην εικόνα 4.9.



4.9 Παράδειγμα 1ο

Μετά το πάτημα για την επιβεβαίωση της φόρμας «Εμφάνιση αποτελεσμάτων» το μήνυμα που εμφανίζεται στην τελική πλέον σελίδα είναι αυτό της εικόνας 4.10.



4.10 Παράδειγμα 1ο – αποτελέσματα

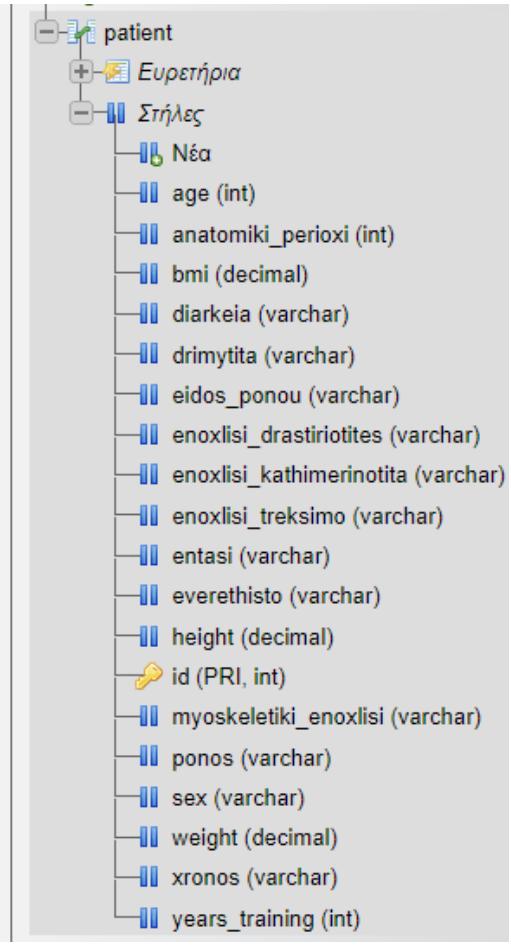
Όλα λειτουργούν σύμφωνα με τις προδιαγραφές της εφαρμογής και το μήνυμα που εμφανίζεται είναι αυτό που πρέπει. Παράλληλα εμφανίζεται ένα δεύτερο μήνυμα προς τον χρήστη το οποίο σχετίζεται με το ασκησιολόγιο του ασκούμενου και πιο συγκεκριμένα με την ένταση των ασκήσεων που κάνει. Το μήνυμα αυτό εμφανίζεται ΜΟΝΟ στην περίπτωση που ο συνδυαστικός κανόνας αξιολόγησης προτείνει και ασκήσεις. Στον επόμενο πίνακα παρουσιάζονται οι κανόνες υπολογισμού για το μήνυμα αυτό:

1. Εάν **ΕΥΕΡΕΘΙΣΤΟ** ΧΑΜΗΛΟ και **ΔΡΙΜΥΤΗΤΑ** ΧΑΜΗΛΟ τότε Συνίσταται ασκήσεις **υψηλής** έντασης
2. Εάν **ΕΥΕΡΕΘΙΣΤΟ** ΜΕΤΡΙΟ και **ΔΡΙΜΥΤΗΤΑ** ΧΑΜΗΛΟ τότε Συνίσταται ασκήσεις **μέτριας** έντασης
3. Εάν **ΕΥΕΡΕΘΙΣΤΟ** ΥΨΗΛΟ και **ΔΡΙΜΥΤΗΤΑ** ΜΕΤΡΙΟ τότε Συνίσταται ασκήσεις **χαμηλής** έντασης

Πίνακας 4 Κανόνας υπολογισμού έντασης ασκήσεων

Όπως είναι φανερό, για την ένταση των ασκήσεων, όταν αυτές είναι εφικτές, στην εξίσωση μπαίνει και η παράμετρος ΕΥΕΡΕΘΙΣΤΟ. Αν οι επιλογές στη ένταση του πόνου είναι d ή e και ΝΑΙ στις δύο τελευταίες τότε η εφαρμογή ζητά από τον ασκούμενο να απευθυνθεί σε ειδικό καθώς «θεωρεί» το πρόβλημα πολύ σοβαρό για να λυθεί απλά με μείωση της άσκησης.

Για την υλοποίηση της sort term memory, τα στοιχεία κάθε ασκούμενου αποθηκεύονται στη βάση δεδομένων, με κατάλληλες εντολές php. Θέλουμε να αποθηκεύονται όλες οι πληροφορίες οι οποίες αφορούν τον εκάστοτε ασκούμενο που θα χρησιμοποιήσει την εφαρμογή. Επομένως, πρέπει κάθε φορά που υπάρχει η μετάβαση από την πρώτη σελίδα έως και την τελευταία να «μεταφέρονται» και οι μεταβλητές ώστε η καταχώρηση στη βάση να γίνεται στον τερματισμό της εφαρμογής, δηλαδή στη σελίδα όπου εμφανίζεται το τελικό αποτέλεσμα – συμβουλή. Στη σελίδα http://localhost/phpmyadmin/ ο xampp server μας δίνει τη δυνατότητα να δημιουργούμε τη δική μας βάση στην οποία θα αποθηκεύονται τα στοιχεία που επιθυμούμε. Στη βάση δημιουργούμε έναν πίνακα με το όνομα patient και εισάγουμε τις στήλες που επιθυμούμε. Για την εφαρμογή απαιτούνται 18 στήλες για κάθε επιλογή – καταχώρηση του χρήστη, συν μία που λογίζεται ως κλειδί και βοηθά στο να μετρούμε τις καταχωρήσεις, ειδικά όταν ελέγχουμε την ορθότητα του κώδικα της εφαρμογής.



4.11 Οι στήλες του πίνακα της Βάσης Δεδομένων

Στην εικόνα 4.11 παρουσιάζονται οι στήλες του πίνακα patient της βάσης δεδομένων. Η ονομασία της κάθε στήλης είναι τέτοια ώστε να γίνεται αντιληπτό σε πιο πεδίο αντιστοιχεί. Μετά από μια καταχώρηση δεδομένων στην εφαρμογή οι τιμές στις στήλες αυτές είναι όπως παρουσιάζονται(ενδεικτικά) στην εικόνα 4.12.

id	height	weight	age	bmi	years_training	sex	eidos_ponou	anatomiki_perioxi	myoskeletiki_enoxlisi	ponos	xronos	diarkeia	entasi	enoxlisi_treksimo	enoxlisi_kathimerinotita	enoxlisi_drasti
222	188	85	45	3	15	male	sporadikos	0	oxi							
223	222	150	46	5	12	male	sporadikos	25	nai	c	b	b	a	c	nai	oxi
224	168	70	25	2	4	male	synexis	5	nai	c	a	b	b	a	oxi	oxi
225	168	70	25	2	4	male	synexis	5	nai	e	a	b	b	a	nai	nai

4.12 Οι στήλες του πίνακα patient

4.2.2 Τεχνικά χαρακτηριστικά της Εφαρμογής

Για την ανάπτυξη της εφαρμογής, και κυρίως για την υλοποίηση των χαρακτηριστικών του έμπειρου συστήματος χρησιμοποιήθηκε κώδικας σε php, ενσωματωμένος φυσικά σε αρχεία html. Για να γίνει πιο ευπαρουσίαστη η εφαρμογή, χρησιμοποιήθηκε κώδικας CSS, ώστε οι φόρμες συμπλήρωσης να αποκτήσουν την μορφή που φαίνεται στις εικόνες παρουσίασης της. Πιο συγκεκριμένα, για την μετάβαση των τιμών των μεταβλητών από μία σελίδα σε μια άλλη χρησιμοποιήθηκε η τεχνική που παρουσιάζεται στο παράρτημα κώδικα.

Για την υλοποίηση των υπολογισμών κανόνων, οι δομές if, elseif & else είναι ιδανικές για να υπολογισθούν σε μεταβλητές, οι οποίες σε νέες δομές if, θα υπολογίσουν τιμές οι οποίες θα χρειαστούν για τον υπολογισμό των συνδυαστικών κανόνων αξιολόγησης. Οι μεταβλητές αυτές μεταβιβάζουν τις τιμές από σελίδα σε σελίδα έως και την τελική, όπου εν τέλει αποθηκεύονται και στη βάση δεδομένων, αφού εμφανισθεί το τελικό αποτέλεσμα – συμβουλή προς τον ασκούμενο.

Ο τρόπος που επιλέχθηκε για την επιλογή του σημείου πόνου από την εικόνα στην οποία παρουσιάζονται οι ανατομικές περιοχές είναι με την μέθοδο map html, με χρήση πολυγώνων για το σχηματισμό της περιοχής επιλογής και, με μια απλή συνάρτηση σε javascript η οποία μεταφέρει σε μεταβλητή την επιλογή της περιοχής(από 0 έως και 27), στη συνέχεια την τιμή αυτή στο πεδίο τύπου number της html και τέλος, από εκεί σε μεταβλητή της php για τον υπολογισμό των κανόνων.

4.2.3 Συμπέρασμα

Για την υλοποίηση ενός έμπειρου συστήματος, βασισμένου σε κανόνες, απαιτείται πρώτιστος η γνώση και οι κανόνες οι οποίοι θα χρειαστούν για να υλοποιηθεί. Οι γνώσεις πηγάζουν από την εμπειρία και τη γνώση ενός(ή και περισσότερων) επιστημονικού πεδίου το οποίο δεν είναι απλά

αποτέλεσμα μιας απλής έρευνας. Η υλοποίηση μιας εφαρμογής, εν κατακλείδι, εξαρτάται περισσότερο από την δυνατότητά μας να συλλέξουμε δεδομένα από τους ειδικούς, από έρευνες και αποτελέσματα, παρά από το να γνωρίζουμε προχωρημένες τεχνικές σε επίπεδο συγγραφής κώδικα, κάτι που για έναν προγραμματιστή π.χ. φέρνει τα πάνω κάτω όσον αφορά στο τι έχει εκπαιδευτεί να κάνει. Παρόλα αυτά, είναι ένα ενδιαφέρον και πολύ χρηστικό περιβάλλον πάνω στο οποίο καλείται η σύγχρονη επιστήμη να Δώσει βάση για το μέλλον.

Συμπεράσματα

Για την υλοποίηση αλγόριθμων μηχανικής μάθησης, είτε με επίβλεψη είναι χωρίς, αυτό που απαιτείται είναι μια συλλογή δεδομένων (και μάλιστα με δομή παρόμοια με τα datasets στο Weka, τα οποία με τη σειρά τους έχουν ομοιότητες με πίνακες από αρχεία excel) και αλγόριθμοι ο οποίοι θα εκμεταλλεύονται τα δεδομένα αυτά ώστε να οδηγούν σε έξοδο – πρόβλεψη, μειώνοντας όσο το δυνατό τα λάθη και αυξάνοντας την ακρίβεια. Στους αλγόριθμους που παρουσιάστηκαν είναι φανερό ότι κάθε ένας από αυτούς λειτουργεί με διαφορετική ακρίβεια σε διαφορετικά datasets κάτι που πρέπει να λαμβάνεται υπόψη στην επιλογή. Οι τεχνικές υλοποίησης των αλγόριθμων βασίζονται στη λογική και στον τρόπο με τον οποίο σκέφτεται και δρα ο άνθρωπος σε συνδυασμό με μαθηματικά μοντέλα για τους υπολογισμούς. Όντως δηλαδή, το έγγραφο αυτό απομυθοποιεί σε ένα βαθμό τον θόρυβο που υπάρχει γύρο από τη μηχανική μάθηση και την Τεχνητή Νοημοσύνη γενικότερα.

Στην υλοποίηση ενός έμπειρου συστήματος λαμβάνονται υπόψη τα δεδομένα που έχουν συλλεχθεί και τεκμηριωθεί σε βάθος χρόνου, ικανά ώστε να οδηγήσουν σε γνώση. Μια εφαρμογή η οποία θα υλοποιεί ένα έμπειρο σύστημα λειτουργεί με πιο απλές μεθόδους προγραμματισμού γιατί τα αποτελέσματα που θα εξάγει θα βασίζονται σε κανόνες και όχι σε στατιστική, πιθανότητες και προβλέψεις. Είναι προφανές ότι η τεχνητή νοημοσύνη θα πρέπει να εκμεταλλεύεται όλες τις προϋπάρχουσες γνώσεις των ειδικών για τον σχεδιασμό εφαρμογών συστημάτων βασισμένα σε κανόνες ώστε στο μέλλον να είμαστε σε θέσει να εξυπηρετούμαστε από την τεχνολογία και σε τομείς που απαιτείται το «μάτι» ενός ειδικού.

Οι διαφορά της μηχανικής μάθησης από τα συστήματα βασισμένα σε κανόνες έγκειται στο γεγονός αυτό. Οι πιο γνωστές μηχανές αναζήτησης προσφέρουν στις εταιρίες τη δυνατότητα να διαφημίζονται πιο στοχευμένα, στους χρήστες μεγαλύτερη αξιοπιστία συναλλαγών μιας και μπορούν να «ανακαλύπτουν» ιστοτόπους με κακόβουλο περιεχόμενο, ηλεκτρονικής απάτης, παράνομο υλικό κλπ, καθώς και να εξελίσσονται συστήματα πλοήγησης, προβλέψεων καιρού και πολλά άλλα χρήσιμα για τον άνθρωπο.

Οι επιστήμονες πλέον έχουν στη διάθεσή τους έναν ατελείωτο ωκεανό από δεδομένα και πληροφορίες και διαρκώς θα εφευρίσκουν τρόπους να τα εκμεταλλεύονται είτε προς όφελος της επιστήμης και της τεχνολογίας, είναι προς όφελος του κέρδους το οποίο θα δημιουργεί νέες θέσεις εργασίας με καινούργια μέσα και προοπτικές εξέλιξης. Οι μέθοδοι που παρουσιάστηκαν στην εργασία, της μηχανικής μάθησης και των έμπειρων συστημάτων, έχουν ούτως ή άλλως περιθώρια εξέλιξης, ανάπτυξης νέων τεχνικών και αλγορίθμων και μελέτη αυτών. Ελπίζουμε το μέλλον να είναι εξίσου συναρπαστικό ή και ακόμη περισσότερο…

Βιβλιογραφία

[1] Roberto Battiti - Mauro Brunato, *The LION Way, Machine Learning plus Intelligent Optimization*, 2013th ed. Los Angeles, USA: LIONsolver, 2013.

[2] Ian Witten. (2013) www.cs.waikato.ac.nz. [Online]. https://www.cs.waikato.ac.nz/ml/weka/courses.html

[3] Κατερίνα Γεωργούλη, *Τεχνιτή Νοημοσύνη - Μια Εισαγωγική Προσέγγιση*, Σωκράτης Κατσικάς, Ed.: Σύνδεσμος Ελλήνων Ακαδημαϊκών Βιβλιοθηκών, 2015.

[4] Michael Steinbach, Vipin Kumar Pang-Ning Tan, *Introduction to Data Mining*. USA: PEARSON Addison Wesley, 2006.