ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Εκπαίδευση και εξοικείωση με τα λογισμικά WEKA και RapidMiner. Συγκριτική αξιολόγηση των μεθόδων τους.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΤΟΥ: ΠΕΤΡΟΥ ΣΚΟΡΔΑ
Π2015070

ΕΠΙΒΛΕΠΟΥΣΑ ΚΑΘΗΓΗΤΡΙΑ: ΚΑΤΙΑ - ΛΗΔΑ ΚΕΡΜΑΝΙΔΟΥ

ΚΕΡΚΥΡΑ 2019

ΕΥΧΑΡΙΣΤΙΕΣ

Θεωρώ υποχρέωση μου να ευχαριστήσω την επιβλέπουσα καθηγήτρια Κάτια - Λήδα Κερμανίδου για την πολύτιμη καθοδήγηση της. Θέλω να ευχαριστήσω θερμά αλλά και να αφιερώσω την πτυχιακή μου εργασία στην οικογένεια μου, που μου συμπαραστάθηκαν όλα τα χρόνια της φοίτησης μου στο τμήμα Πληροφορικής του Ιονίου Πανεπιστημίου της Κέρκυρας.

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία έχει ως θέμα την εκπαίδευση και εξοικείωση του χρήστη με τα λογισμικά weka και rapidminer. Όσων αφορά την χρήση τους, ειδικεύονται στους κλάδους της εξόρυξης δεδομένων και της μηχανικής μάθησης, με κύριο στόχο την επεξεργασία συνόλων δεδομένων, ώστε να διεξαχθεί πολύτιμη πληροφορία. Συγκεκριμένα, παρουσιάζεται μέσα από πολλά διαφορετικά παραδείγματα με σύνολα δεδομένων, ο τρόπος με τον οποίο μπορούμε να τα διαχειριστούμε είτε με το ένα είτε με το άλλο λογισμικό. Τα αποτελέσματα από τα πειράματα μπορούν να δώσουν στον χρήστη σημαντικές πληροφορίες για μελλοντικές προβλέψεις. Αναλυτικότερα, μέσα από την εργασία μπορεί κανείς να μάθει όλες τις βασικές αλλά και επιπρόσθετες λειτουργίες τόσο του weka όσο και του rapidminer, ξεκινόντας από τα πιο απλά εκπαιδευτικά μαθήματα και πηγαίνοντας σε πιο εξειδικευμένα. Πέρα από τα πρακτικά μέρη της εργασίας, τα οποία περιγράφονται μέσα από την παρουσίαση ενδεικτικών εικόνων, υπάρχει και η αντίστοιχη θεωρία που χρειάζεται να γνωρίζει κανείς για να κατανοήσει πως ακριβώς λειτουργούν οι αλγόριθμοι που έχουν υλοποιηθεί. Η εργασία ολοκληρώνεται έπειτα από μία συγκριτική αξιολόγηση των δύο λογισμικών, όπου φαίνονται τα σημεία στα οποία υπερτερεί το λογισμικό rapidminer έναντι του weka, χάρη στην ευκολία χρήσης, τις δυνατότητες που προσφέρει αλλά και τον χρόνο που μπορεί να εξοικονομήσει κάποιος επιλέγοντας το είτε για προσωπική είτε για επαγγελματική χρήση.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: weka, rapidminer, εξόρυξη δεδομένων, μηχανική μάθηση, αλγόριθμοι, εκπαιδευτικά μαθήματα, συγκριτική αξιολόγηση

ABSTRACT

The present thesis focuses on the training and familiarization of the user with weka and rapidminer software. In their use, they specialize in data mining and machine learning. The main objective of using them is to process data sets in order to carry out valuable information. In particular, it is presented through many different examples of data sets, how we can manage them either with the one software or the other. The results of the experiments can give the user important information for future predictions. In more detail, through this thesis one can learn all the basic and additional functions of both weka and rapidminer, starting with the simplest training courses and going to the more specialized. Beyond the practical parts of this thesis, which are described through the presentation of court pictures, there is also the corresponding theory that one needs to know in order to understand exactly how the algorithms that have been implemented work. The thesis is completed after a comparative evaluation of the two software, where the points of the rapidminer software over the weka are shown, thanks to its ease of use, the possibilities it offers but also the time one can save by choosing it for either personal or professional use.

KEY WORDS: weka, rapidminer, data mining, machine learning, algorithms, training courses, comparative evaluation

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΕΙΣΑΓΩΓΗ

Η πτυχιακή εργασία έχει ως στόχο να παρουσιάσει με εκπαιδευτικό τρόπο τα λογισμικά Weka και RapidMiner σε φοιτητές που πρόκειται να τα χρησιμοποιήσουν κατά τις σπουδές τους σε τμήματα Πληροφορικής ή σε χρήστες που επιθυμούν να μάθουν τις ικανότητες των δύο αυτών εργαλείων και να τα αξιοποιήσουν είτε προσωπικά είτε επαγγελματικά. Αρχικά γίνεται γνωστή στους αναγνώστες η έννοια της εξόρυξης δεδομένων, ο στόχος της και τα βήματα που ακολουθούνται κατά την υλοποίηση της. Στη συνέχεια, ο αναγνώστης αποκτά τις βασικές γνώσεις που χρειάζεται για να μπορεί να ορίσει την μηχανική μάθηση, να δει τον στόχο της και να είναι σε θέση να διακρίνει τις κατηγορίες μηχανικής μάθησης αλλά και τους τρόπους εκμάθησης. Με την ολοκλήρωση των εννοιών -εξόρυξη δεδομένων και μηχανική μάθηση- θα μεταφερθεί στο πρακτικό, εργαστηριακό κομμάτι που αφορά τα δύο λογισμικά. Τα βήματα είναι δομημένα με τέτοιο τρόπο ώστε να υπάρχει τόσο η απαραίτητη θεωρία για να γίνονται κατανοητές οι έννοιες, όσο και η πράξη για την εξοικείωση με αυτά. Στο λογισμικό Weka γίνεται μία πρώτη γνωριμία με αυτό και μετά παρουσιάζεται ο τρόπος εγκατάστασης του. Αφού έχει πλέον ολοκληρωθεί αυτό το κομμάτι, στη συνέχεια αναλύονται ορισμένες από τις δυνατότητες του λογισμικού, όπως η χρήση αλγορίθμων και η οπτικοποίηση των αποτελεσμάτων από κάποια πειράματα, που καλύπτουν το μεγαλύτερο μέρος του και έχουν αρκετό ενδιαφέρον. Ίδια λογική υιοθετείται και για την εκπαίδευση και εξοικείωση με το λογισμικό RapidMiner όπου μετά την ολοκλήρωση της εγκατάστασης πηγαίνουμε στην αξιοποίηση του. Αφού ολοκληρώνεται το πρακτικό μέρος των δύο λογισμικών, ακολουθεί η συγκριτική αξιολόγηση τους όπου θα μπορείτε να αποφασίσετε ευκολότερα ποιο εργαλείο καλύπτει καλύτερα τις ανάγκες σας. Οι λόγοι για τους οποίους έχω επιλέξει να παρουσιάσω αυτή την εργασία είναι τρεις. Πρώτων, γιατί και τα δύο αυτά λογισμικά είναι πολύ χρήσιμα και θα αποτελούν πλεονέκτημα για κάποιον που είναι σε θέση να τα χρησιμοποιεί σε μία εποχή όπου κυριαρχεί η εξόρυξη πληροφορίας, η μηχανική μάθηση και τα δεδομένα έχουν πολύ μεγάλη αξία για τις επιχειρήσεις. Δεύτερον, γιατί είναι πολύ σημαντικό για κάθε φοιτητή Πληροφορικής να μάθει να αξιοποιεί ένα τέτοιο εργαλείο, το οποίο σίγουρα θα ενισχύσει τις ικανότητές του και θα μπορέσει να τον βοηθήσει στο μέλλον και σε επαγγελματικό επίπεδο. Τρίτον, επειδή κατά την διδασκαλία στο τμήμα πληροφορικής, κατάφερα να εντοπίσω ένα κομμάτι με το οποίο θέλω να ασχοληθώ περισσότερο μετά την απόκτηση του πτυχίου, όπου η εξόρυξη πληροφοριών αποτελεί πολύ σημαντικό κομμάτι.

1. ΕΞΟΡΥΞΗ ΔΕΔΟΜΕΝΩΝ

1.1 Ορισμός της εξόρυξης δεδομένων

Εξόρυξη δεδομένων[1] είναι η διαδικασία ανακάλυψης προτύπων ή πληροφοριών μέσα σε τεράστια σύνολα από δεδομένα, που πραγματοποιείται με τη συμμετοχή της μηχανικής μάθησης, της στατιστικής και των συστημάτων βάσεων δεδομένων. Η εξόρυξη δεδομένων βρίσκεται στο διεπιστημονικό πεδίο της επιστήμης των υπολογιστών και της στατιστικής, με κύριο στόχο την εξαγωγή της πληροφορίας (με έξυπνες μεθόδους), από μεγάλα σύνολα δεδομένων και τη μετατροπή της πληροφορίας σε κατανοητή δομή για περαιτέρω χρήση. Πέραν του βήματος της ανάλυσης δεδομένων, περιλαμβάνει επίσης βάσεις δεδομένων και τρόπους διαχείρισης τους, προ-επεξεργασία δεδομένων, συμπεράσματα των μοντέλων, μετρήσεις ενδιαφέροντος, θεωρίες πολυπλοκότητας, μετα-επεξεργασία προτύπων που έχουν ανακαλυφθεί, οπτικοποίηση και ενημέρωση. Η διαφορά ανάμεσα στην ανάλυση και στην εξόρυξη των δεδομένων είναι ότι στην πρώτη συνοψίζεται το ιστορικό ενός γεγονότος όπως για παράδειγμα μια αποτελεσματική στρατηγική μάρκετινγκ ενώ στη δεύτερη δίνεται έμφαση στη χρήση συγκεκριμένων μοντέλων μηχανικής μάθησης και στατιστικής για μελλοντικές προβλέψεις και ανακάλυψη πληροφοριών.

1.2 Στόχος της εξόρυξης δεδομένων

Ο βασικός στόχος της εξόρυξης δεδομένων είναι η ημι-αυτόματη ή αυτόματη ανάλυση τεράστιων ποσοτήτων με δεδομένα για την εξαγωγή πρώην άγνωστης, σημαντικής και ενδιαφέρουσας πληροφορίας. Διακρίνεται σε ομάδες από αρχεία δεδομένων (ομαδοποιημένη ανάλυση), ασυνήθιστα αρχεία (εντοπισμός ανωμαλίας), και εξαρτήσεις (εξόρυξη με κανόνες συσχετίσεων). Συνήθως περιλαμβάνει τεχνικές που χρησιμοποιούνται στις βάσεις δεδομένων. Αυτά τα πρότυπα ή πληροφορίες που εξάγονται, μπορούν στη συνέχεια να αξιοποιηθούν σε περαιτέρω ανάλυση, όπως για παράδειγμα στη μηχανική μάθηση και στα αναλυτικά στοιχεία πρόβλεψης. Για παράδειγμα, το στάδιο της εξόρυξης της πληροφορίας θα αποκαλύψει εκείνα τα στοιχεία που είναι σημαντικά για μία πρόβλεψη μεγάλης ακρίβειας, που δίνεται στη συνέχεια από τα συστήματα υποστήριξης αποφάσεων.

1.3 Βήματα στην εξόρυξη δεδομένων

1. Επιλογή των δεδομένων
2. Προ-επεξεργασία
3. Μετατροπή ή προσαρμογή
4. Εξόρυξη πληροφορίας
5. Εκτίμηση των αποτελεσμάτων

[1] https://en.wikipedia.org/wiki/Data_mining

Ωστόσο υπάρχουν και τα στάδια εξόρυξης όπως έχουν δοθεί από την Cross Industry Standard Process for Data Mining, η οποία έχει δημιουργηθεί από ειδικούς στον τομέα της εξόρυξης και είναι ευρέως χρησιμοποιούμενο το πρότυπο της.

1. Κατανόηση της επιχείρησης
2. Κατανόηση των δεδομένων
3. Προετοιμασία των δεδομένων
4. Μοντελοποίηση
5. Εκτίμηση
6. Ανάπτυξη

1.4 Οι 6 τάξεις της εξόρυξης δεδομένων

1.Εντοπισμός ανωμαλίας.

Είναι ο εντοπισμός αρχείων που περιέχουν ασυνήθιστα δεδομένα ή που μπορεί να υπάρχουν σφάλματα στα δεδομένα και που απαιτούν μεγαλύτερη διερεύνηση.

2.Μάθηση με κανόνες συσχέτισης.

Είναι η αναζήτηση σχέσεων που μπορεί να εμφανίζονται μέσα στις μεταβλητές.

Για παράδειγμα η συλλογή πληροφοριών για έναν πελάτη που ψωνίζει στο σούπερ μάρκετ. Με τη χρήση κανόνων συσχέτισης, το κατάστημα μπορεί να αναγνωρίσει ποια προϊόντα αγοράζονται συχνά μαζί και να εφαρμόσει κατάλληλες στρατηγικές μαρκετινγκ.

3.Ομαδοποίηση

Είναι η διαδικασία ανακάλυψης ομάδων μέσα στα δεδομένα, δηλαδή στοιχεία τα οποία κατά κάποιο τρόπο είναι όμοια.

4.Ταξινόμηση

Είναι η διαδικασία εφαρμογής μιας ήδη γνωστής γενικευμένης δομής σε νέα δεδομένα. Για παράδειγμα ένα πρόγραμμα μπορεί να ταξινομήσει ένα email αν είναι να τοποθετηθεί στα εισερχόμενα ή στα ανεπιθύμητα.

5.Οπισθοδρόμηση

Προσπάθεια εύρεσης μιας συνάρτησης ώστε τα δεδομένα να έχουν όσο το δυνατό λιγότερα σφάλματα για να γίνει σωστή εκτίμηση των σχέσεων τους ή σωστή εκτίμηση του συνόλου των δεδομένων.

6.Σύνοψη

Είναι η προβολή των δεδομένων με πιο συμπαγή παρουσίαση, περιλαμβάνοντας την οπτικοποίηση και τη δημιουργία αναφορών.

2 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

2.1 Ορισμός της μηχανικής μάθησης

Μηχανική μάθηση[2] σύμφωνα με τον Άρθουρ Σάμουελ το 1959 ορίζεται ως το πεδίο μελέτης που δίνει στους υπολογιστές την ικανότητα να μαθαίνουν, χωρίς να έχουν ρητά προγραμματιστεί. Αποτελεί ένα κλαδί της τεχνητής νοημοσύνης που βασίζεται στην ιδέα ότι τα συστήματα μπορούν να μαθαίνουν μέσα από δεδομένα, να ανακαλύψουν πρότυπα και να λάβουν αποφάσεις με ελάχιστη παρέμβαση του ανθρώπινου παράγοντα. Η μηχανική μάθηση διερευνά τη μελέτη και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά. Είναι στενά συνδεδεμένη και συχνά συγχέεται με υπολογιστική στατιστική, ένας κλάδος, που επίσης επικεντρώνεται στην πρόβλεψη μέσω της χρήσης υπολογιστών. Εφαρμόζεται σε μία σειρά από υπολογιστικές εργασίες, όπου τόσο ο σχεδιασμός όσο και ο ρητός προγραμματισμός των αλγορίθμων είναι ανέφικτος. Επιπλέον, συνδέεται και με την εξόρυξη δεδομένων που επικεντρώνεται στην ανάλυση των δεδομένων.

2.2 Στόχος της μηχανικής μάθησης

Όλες οι τεχνικές που έχουμε δει μέχρι στιγμής μας επιτρέπουν τη δημιουργία έξυπνων συστημάτων. Τα ευφυή συστήματα έχουν την ικανότητα να παρατηρούν το περιβάλλον και να μαθαίνουν από αυτό. Αυτού του είδους η νοημοσύνη καλείται να αντιμετωπίσει και να επιλύσει νέα προβλήματα καθώς και να μάθει μέσα από λάθη. Ο στόχος της μηχανικής μάθησης είναι να ξεπεράσει τους περιορισμούς των απλών εφαρμογών τεχνητής νοημοσύνης και να μπορεί να επιλύσει σημαντικά προβλήματα με τέτοιο τρόπο ώστε να μη χρειάζεται η παρέμβαση του ανθρώπινου παράγοντα ή ο επαναπρογραμματισμός των προγραμμάτων. Δεν είναι δυνατό να γνωρίζουμε εκ των προτέρων όλες τις πιθανές λύσεις ή τα προβλήματα αλλά μέσω υπολογιστικών συστημάτων που είναι ικανά να μαθαίνουν με κατάλληλη εκπαίδευση, θα μπορέσουμε να έχουμε σημαντικές προβλέψεις και λύσεις σε κρίσιμα ζητήματα.

2.3 Κατηγορίες της μηχανικής μάθησης

1.Επιτηρούμενη μάθηση (επιβλεπόμενη μάθηση)
   Το υπολογιστικό πρόγραμμα δέχεται τις παραδειγματικές εισόδους καθώς και τα επιθυμητά αποτελέσματα από έναν άνθρωπο και ο στόχος είναι να μάθει έναν γενικό κανόνα προκειμένου να αντιστοιχίσει τις εισόδους με τα αποτελέσματα.
2.Μη επιτηρούμενη μάθηση (μάθηση χωρίς επίβλεψη)
   Χωρίς να παρέχεται κάποια εμπειρία στον αλγόριθμο μάθησης, πρέπει να βρει τη δομή των δεδομένων εισόδου.
3.Ενισχυτική μάθηση
   Ένα πρόγραμμα υπολογιστή αλληλεπιδρά με ένα δυναμικό περιβάλλον στο οποίο πρέπει να επιτευχθεί ένας συγκεκριμένος στόχος. Ένα παράδειγμα είναι

[2] https://en.wikipedia.org/wiki/Machine_learning

να μάθει να παίζει ένα παιχνίδι εναντίον κάποιου αντιπάλου.

4.Αναπτυξιακή μάθηση

Έχει αναπτυχθεί για την εκμάθηση από ρομπότ, δημιουργεί τη δική της ακολουθία μαθησιακών καταστάσεων, ώστε το ρομπότ συσσωρευτικά αποκτά ποικιλία δεξιοτήτων μέσω της αυτόνομης αυτοεξερεύνησης και της κοινωνικής αλληλεπίδρασης με ανθρώπους. Χρησιμοποιεί μηχανισμούς καθοδήγησης, όπως η ενεργητική μάθηση, η ωρίμανση και η μίμηση.

2.5 Τρόποι εκμάθησης

1. Εκμάθηση με δέντρο απόφασης
2. Εκμάθηση με κανόνες συσχέτισης
3. Τεχνητά νευρωτικά δίκτυα
4. Βαθιά μάθηση
5. Επαγωγικός λογικός προγραμματισμός
6. Μηχανές διανυσμάτων υποστήριξης
7. Ομαδοποίηση
8. Δίκτυα Bayes
9. Ενισχυτική μάθηση
10. Εκμάθηση με μέτρο ομοιότητας
11. Γενετικοί αλγόριθμοι

Πολλοί από τους παραπάνω τρόπους εκμάθησης βρίσκονται μέσα στα λογισμικά WEKA και RapidMiner. Εδώ δεν θα αναλύσουμε τον κάθε τρόπο αφού υπάρχει πολύ μεγάλος όγκος πληροφορίας. Στα βήματα εκμάθησης που θα ακολουθήσουν θα δούμε σημαντικά παραδείγματα αυτών των τρόπων και θα μάθουμε να χρησιμοποιούμε καλά και τα δύο αυτά εργαλεία.[3]

[3] https://www.sas.com/en_id/insights/analytics/machine-learning.html

3 Λογισμικό Εξόρυξης Δεδομένων σε Java

Το Waikato Environment for Knowledge Analysis (WEKA)[4] είναι ένα λογισμικό εξόρυξης δεδομένων γραμμένο σε Java, που περιέχει αλγορίθμους μηχανικής μάθησης. Δημιουργήθηκε στο πανεπιστήμιο Waikato, στη Νέα Ζηλανδία το 1993 και είναι ένα δωρεάν, διαθέσιμο λογισμικό με άδεια της GNU General Public License. Η ανάπτυξη της ολοκληρωμένης έκδοσης, που βασίστηκε σε Java, ξεκίνησε το 1997 συμπεριλαμβάνοντας την εγκατάσταση αλγορίθμων μοντελοποίησης και πλέον χρησιμοποιείται σε πολλές διαφορετικές περιοχές, τόσο για εκπαιδευτικούς όσο και για ερευνητικούς σκοπούς. Μέσα σε αυτό υπάρχουν εργαλεία για προετοιμασία, ταξινόμηση, παλινδρόμηση, ομαδοποίηση, οπτικοποίηση και κανόνες συσχέτισης των δεδομένων. Όλες οι τεχνικές του WEKA βασίζονται στη παραδοχή ότι τα δεδομένα είναι διαθέσιμα μέσα σε ένα αρχείο, όπου κάθε σημείο μέσα στο αρχείο περιγράφεται από έναν σταθερό αριθμό χαρακτηριστικών (αριθμητικά ή ονοματικά χαρακτηριστικά καθώς και μερικοί ακόμα τύποι χαρακτηριστικών υποστηρίζονται). Επιπλέον το WEKA παρέχει πρόσβαση σε Java Database Connectivity και μπορεί να επεξεργαστεί το αποτέλεσμα που προέρχεται από ένα ερώτημα σε βάση δεδομένων. Τέλος το WEKA δίνει τη δυνατότητα πρόσβασης σε βαθιά μάθηση μέσω της βιβλιοθήκης Deeplearing4j, που έχει γραφτεί σε Java.[5][6]

[4] https://www.cs.waikato.ac.nz/ml/weka/
[5] https://en.wikipedia.org/wiki/Weka_(machine_learning)
[6] https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

4 ΕΓΚΑΤΑΣΤΑΣΗ ΛΟΓΙΣΜΙΚΟΥ WEKA

Αυτή τη χρονική περίοδο είναι διαθέσιμες οι δύο πιο πρόσφατες εκδόσεις του WEKA. Η έκδοση 3.8 είναι η τελευταία πιο ολοκληρωμένη, πιο σταθερή και η πιο κατάλληλη για εγκατάσταση στον υπολογιστή. Η έκδοση 3.9 περιλαμβάνει ένα σύστημα διαχείρισης πακέτων που επιτρέπει στην κοινότητα του WEKA να προσθέτει νέα λειτουργικότητα. Σε αυτή την έκδοση όμως μπορεί να υπάρχουν ορισμένα προβλήματα που αποκαλούνται "bugs" γιατί δεν είναι ακόμα σταθερή. Σίγουρα η βέλτιστη επιλογή για εγκατάσταση είναι η έκδοση 3.8 αλλά αν κάποιος επιθυμεί μπορεί να κάνει λήψη προηγούμενων εκδόσεων. Το WEKA διατίθεται για τα λειτουργικά συστήματα Windows, Mac OS X, Linux και άλλα. Ο σύνδεσμος που θα σας οδηγήσει στην επίσημη ιστοσελίδα για επιλογή του πακέτου προς εγκτατάσταση είναι εδώ: https://www.cs.waikato.ac.nz/ml/weka/downloading.html

ΕΙΚΟΝΑ 1

Εκτελέσιμα αρχεία της έκδοσης 3.8

- **Windows**

  Click **here** to download a self-extracting executable for 64-bit Windows that includes Oracle's 64-bit Java VM 1.8
  (weka-3-8-3jre-x64.exe; 120.3 MB)

  Click **here** to download a self-extracting executable for 64-bit Windows without a Java VM
  (weka-3-8-3-x64.exe; 51 MB)

  Click **here** to download a self-extracting executable for 32-bit Windows that includes Oracle's 32-bit Java VM 1.8
  (weka-3-8-3jre.exe; 113.4 MB)

  Click **here** to download a self-extracting executable for 32-bit Windows without a Java VM
  (weka-3-8-3.exe; 51 MB)

  These executables will install Weka in your Program Menu. Download the version without the Java VM if you already have Java 1.8 (or later) on your system.

- **Mac OS X**

  Click **here** to download a disk image for OS X that contains a Mac application including Oracle's Java 1.8 JVM
  (weka-3-8-3-oracle-jvm.dmg; 143.1 MB)

- **Other platforms (Linux, etc.)**

  Click **here** to download a zip archive containing Weka
  (weka-3-8-3.zip; 51.4 MB)

  First unzip the zip file. This will create a new directory called weka-3-8-3. To run Weka, change into that directory and type

  ```
  java -jar weka.jar
  ```

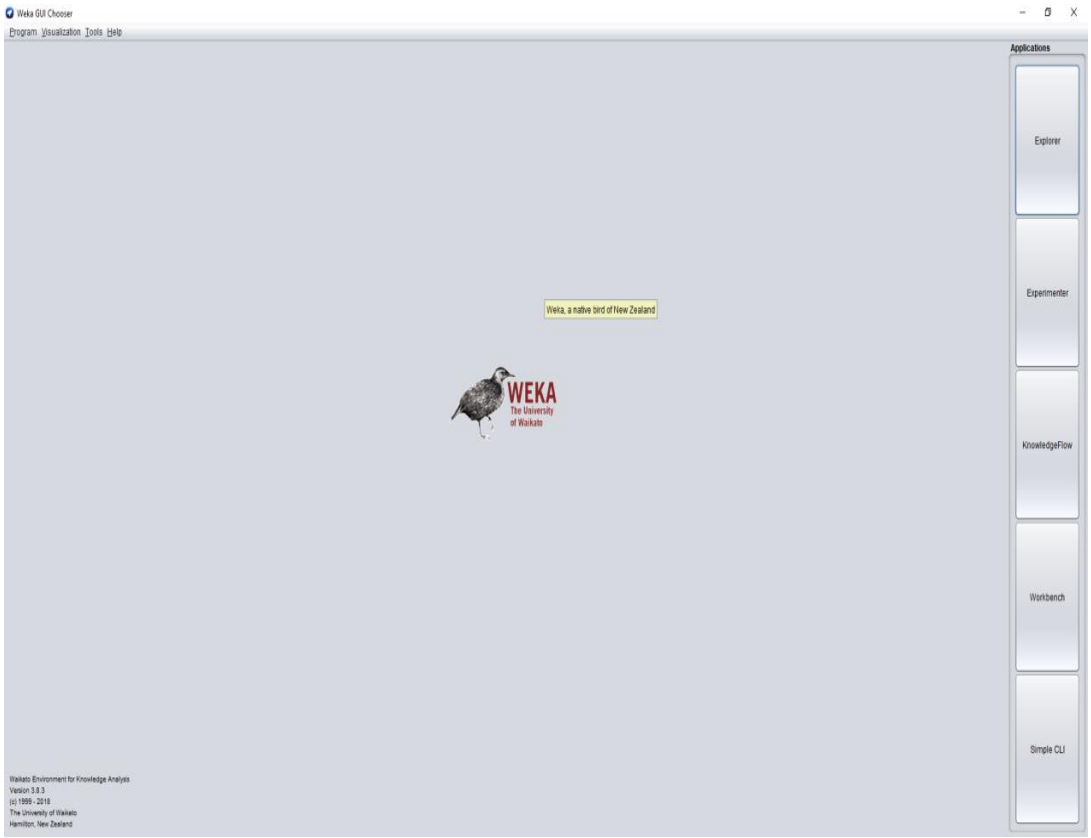
  Note that Java needs to be installed on your system for this to work. Also note, that using `-jar` will override your current CLASSPATH variable and only use the `weka.jar`.

'Οπως φαίνεται στην ΕΙΚΟΝΑ1 μπορούμε να επιλέξουμε είτε ένα πακέτο με το WEKA και το Java Virtual Machine είτε μόνο με το WEKA. Εάν κάποιος χρήστης δεν έχει εγκαταστήσει πρόσφατη έκδοση της Java θα πρέπει να το κάνει αυτό ώστε το

WEKA να μπορέσει να λειτουργήσει κανονικά μετά τη διαδικασία της εγκατάστασης. Έστω ότι επιλέγουμε να εγκαταστήσουμε την 64 bit Windows χωρίς το VM τότε κάνουμε κλικ και η λήψη θα ξεκινήσει μετά από λίγο χωρίς επιπλέον βήματα. Αφού ολοκληρωθεί η λήψη, το κομμάτι της εγκατάστασης είναι απλό καθώς πρέπει μόνο να ακολουθήσουμε τα βήματα και να αφήσουμε τις προεπιλογές. Ανοίγοντας για πρώτη φορά την εφαρμογή θα δούμε ακριβώς αυτό που φαίνεται στην ΕΙΚΟΝΑ2 και στη συνέχεια θα αναφερθούμε πιο αναλυτικά στις λειτουργίες του και στο άνοιγμα των αρχείων.

ΕΙΚΟΝΑ 2

Το περιβάλλον του WEKA



5 ΑΡΧΕΙΑ ARFF, CSV ΚΑΙ ΕΙΣΑΓΩΓΗ ΔΕΔΟΜΕΝΩΝ

5.1 Στόχος

Ο χρήστης μετά την ολοκλήρωση του θεωρητικού και του πρακτικού μέρους να μπορεί να διακρίνει τις διαφορές ανάμεσα στα αρχεία CSV, Excel και ARFF. Επιπλέον να είναι σε θέση να ορίζει η να περιγράφει τα αρχεία CSV και ARFF. Τελικός στόχος είναι να μετατρέπει το αρχείο από CSV σε ARFF και να το φορτώνει και με τις δύο καταλήξεις στο WEKA όπου θα μπορέσει στη συνέχεια να κάνει διάφορους πειραματισμούς με τα δεδομένα.

5.2 Θεωρία για τα αρχεία

Στο πρώτο μέρος για το λογισμικό WEKA παρουσιάζονται οι τύποι των αρχείων που χρησιμοποιούνται από αυτό. Συγκεκριμένα το WEKA δέχεται αρχεία δεδομένων με κατάληξη .arff (attribute relation file format) και .csv (comma separated values). Ένα σύνολο δεδομένων τοποθετείται κατά τον συνήθη τρόπο σε αρχείο xls (Microsoft Excel), το οποίο στη συνέχεια αποθηκεύεται ως αρχείο .csv και μπορεί να μετατραπεί μέσω κατάλληλης επεξεργασίας και αποθήκευσης σε αρχείο .arff.

5.3 ARFF

Ένα αρχείο ARFF[7] (Attribute-Relation File Format) είναι ένα ASCII αρχείο κειμένου το οποίο περιγράφει μία λίστα από παραδείγματα που μοιράζονται ένα σύνολο χαρακτηριστικών. Τα αρχεία ARFF δημιουργήθηκαν από το τμήμα επιστήμης των υπολογιστών του πανεπιστημίου Waikato για να χρησιμοποιούνται από το WEKA.
Το αρχείο ARFF χωρίζεται σε δύο μέρη τα οποία είναι η κεφαλίδα (Header) στο πάνω μέρος του αρχείου και τα δεδομένα (Data) που ακολουθούν. Η κεφαλίδα (Header) περιλαμβάνει το όνομα του (relation) καθώς και τα χαρακτηριστικά (attributes), όπως επίσης και τους τύπους αυτών των χαρακτηριστικών. Ακολουθεί ένα παράδειγμα αρχείου ARFF.

```
@relation iris

@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute class {Iris-setona, 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
```

[7] https://www.cs.waikato.ac.nz/ml/weka/arff.html?fbclid=IwAR0DVTsDzb_dt1BWxA-CSPjUxS7GMJHkGGrMgfJg2nR7jrYuqzaUsasiheE

5.0,3.4,1.5,0.2,Iris-setosa

Τα @relation, @attribute και @data πρέπει να γράφονται ακριβώς έτσι, γιατί αν χρησιμοποιηθούν πεζά και κεφαλαία γράμματα τότε δεν θα αναγνωρίζονται σωστά.
Το όνομα του @relation είναι ένα string όπου σε περίπτωση που υπάρχουν κενά τότε πρέπει να τοποθετείται μέσα σε εισαγωγικά (quoted).
Το κάθε χαρακτηριστικό @attribute έχει το δικό του όνομα που πρέπει να ξεκινάει με αλφαβητικό χαρακτήρα. Αν χρειάζεται να συμπεριληφθούν κενά στο όνομα τότε πρέπει και αυτό να τοποθετείται σε εισαγωγικά (quoted). Επιπλέον το χαρακτηριστικό έχει δεξιά από το όνομα του, έναν τύπο που αντιστοιχεί σε όλη την στήλη των δεδομένων, που απευθύνεται αυτό το χαρακτηριστικό. Για παράδειγμα το πρώτο @attribute sepallength numeric αντιστοιχεί στη πρώτη στήλη αριστερά δηλαδή στα δεδομένα τα οποία έχουν την τιμή numeric. Αντίστοιχα συμβαίνει και με τα υπόλοιπα attributes. Οι τιμές που μπορεί να πάρει είναι numeric(real, integer), <nominal-specification> που αναφέρεται σε συγκεκριμένες τιμές που μπορεί να πάρει ένα χαρακτηριστικό, boolean (true, false) και date[<date-format>] δηλαδή στη μορφή "yyyy-MM-dd'T'HH:mm:ss". Το χαρακτηριστικό @attribute class {Iris-setona, Iris-versicolor, Iris-virginica} αναφέρεται στις τιμές εξόδου του αρχείου.
Το @data καθορίζει το μέρος έναρξης των δεδομένων μέσα στο αρχείο. Τα δεδομένα είναι τοποθετημένα σε γραμμές όπου σε κάθε γραμμή υπάρχει δεδομένο για το κάθε χαρακτηριστικό. Τα δεδομένα χωρίζονται με κόμμα (,) προκειμένου να φορτώνεται σωστά το αρχείο μέσα στο λογισμικό.

5.4 CSV

Ενα αρχείο με κατάληξη .csv[8] ονομάζεται Comma Separated Values file και το περιεχόμενο του αποτελείται από μία λίστα ή σύνολο δεδομένων. Τέτοια αρχεία χρησιμοποιούνται για την ανταλλαγή δεδομένων μέσα σε διαφορετικές εφαρμογές. Για παράδειγμα οι βάσεις δεδομένων και οι κατάλογοι επαφών συχνά υποστηρίζουν αρχεία της μορφής .csv.
Τα αρχεία αυτά μερικές φορές μπορεί να ονομαστούν και ως Character Separated Values ή Comma Delimited files. Περιέχουν το σύμβολο της υποδιαστολής (,) για να ξεχωρίσουν τα δεδομένα αλλά είναι πιθανό να χωρίζεται σε ορισμένα αρχεία και με το σύμβολο του ελληνικού ερωτηματικού (;). Εξυπηρετούν τον χρήστη που θέλει να εξάγει σύνθετα δεδομένα μέσα από μία εφαρμογή σε αρχείο CSV ώστε στη συνέχεια να τα εισάγει μέσα σε κάποια άλλη εφαρμογή.
Αν για παράδειγμα διαθέτουμε μερικές επαφές από έναν κατάλογο επαφών και εξάγουμε τα δεδομένα σε ένα αρχείο CSV τότε η μορφή του θα είναι η ακόλουθη.

Name,Email,PhoneNumber,Address
Bob Smith, bob@example.com , 123-456-7890, 123 Fake Street
Mike Jones, mike@example.com ,098-765-4321, 321 Fake Avenue

Αυτή είναι η βασική μορφή την οποία μπορεί να έχει ένα τέτοιο αρχείο κειμένου, σαφώς όμως μπορούν να έχουν πολύ πιο σύνθετο περιεχόμενο που να αποτελείται από χιλιάδες γραμμές και μεγάλα strings που περικλείονται με εισαγωγικά.
Ο λόγος που είναι τόσο απλή η δομή τους είναι για να μπορεί ο χρήστης να το διαβάζει εύκολα και να είναι διακριτές οι τιμές του. Κάνοντας άνοιγμα ενός τέτοιου αρχείου με

[8] https://www.howtogeek.com/348960/what-is-a-csv-file-and-how-do-i-open-it/

το Notepad ή το Excel φαίνεται ξεκάθαρα το περιεχόμενο του. Υπάρχει περίπτωση, αν το περιεχόμενο του αρχείο είναι πολύ μεγάλο τότε να μην ανοίγει με το Notepad, τότε προτείνεται το εργαλείο Notepad++ το οποίο το ανοίγει κανονικά και επιτρέπει την επεξεργασία του.

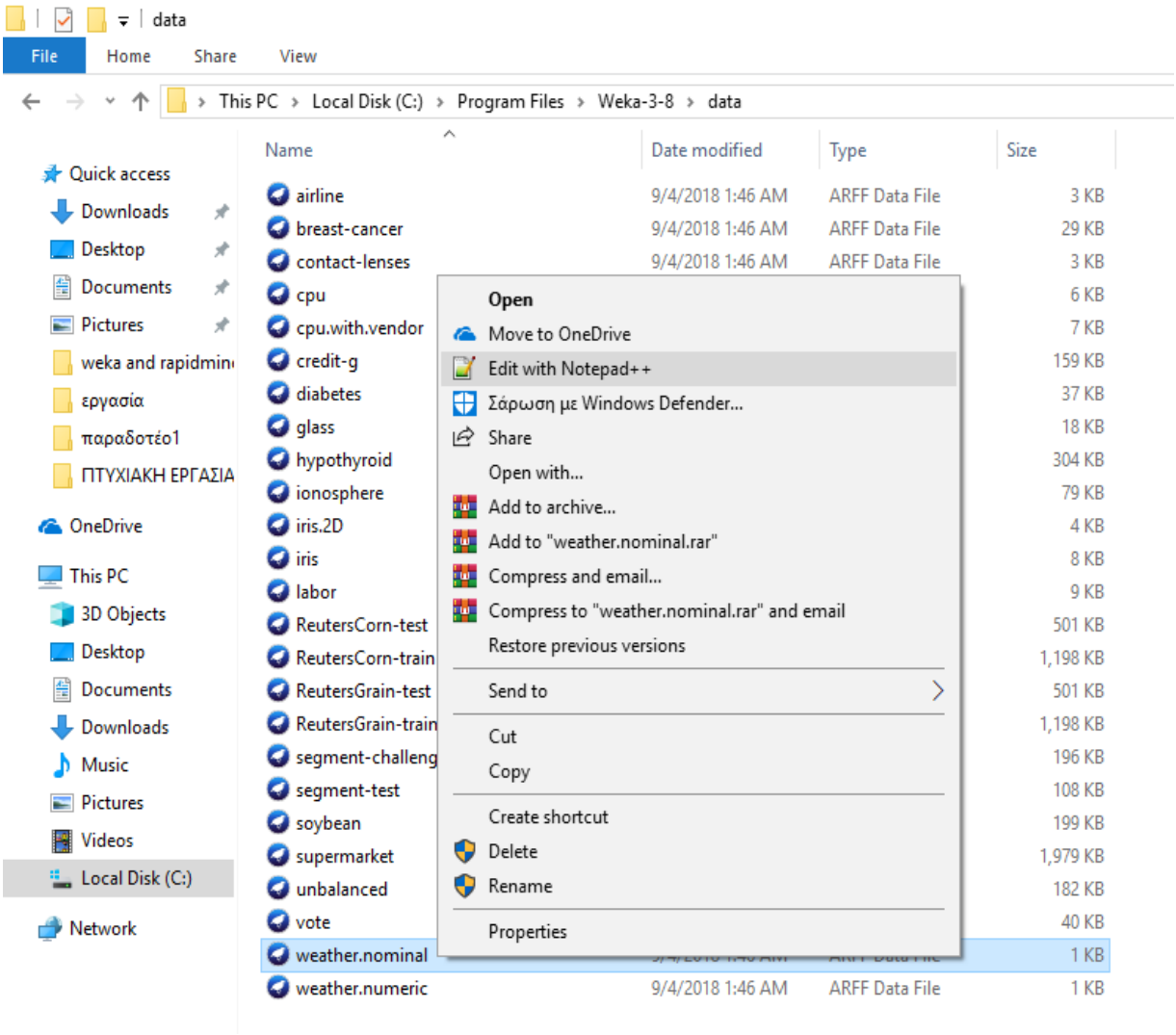
5.5 Πρώτα παραδείγματα στο λογισμικό WEKA

Στο πρώτο πρακτικό μέρος για το λογισμικό WEKA θα παρουσιαστεί η διαδικασία που πρέπει να ακολουθήσει ο χρήστης για να μπορέσει να ανοίξει ένα αρχείο CSV, να δει τον τρόπο με τον οποίο αναπαρίστανται τα δεδομένα μέσα σε αυτό και στη συνέχεια να το μετατρέψει και να το αποθηκεύσει σε αρχείο ARFF. Παρουσιάζονται επίσης σημεία που απαιτούν προσοχή ώστε το αρχείο να μπορέσει να φορτωθεί χωρίς την εμφάνιση error στο WEKA. Σε αυτό παρέχονται έτοιμα τα δεδομένα από τον φάκελο με τα αρχεία ARFF που εγκαθίστανται στον υπολογιστή καθώς πραγματοποιείται η εγκατάσταση του λογισμικού WEKA.

Ανοίγοντας τον τοπικό δίσκο (Local Disk (C:)) και ακολουθόντας το μονοπάτι Program Files > Weka 3.8 > data βρίσκονται έτοιμα αρχεία ARFF με δεδομένα. Στην παρακάτω εικόνα (ΕΙΚΟΝΑ3) παρουσιάζεται ο τρόπος ανοίγματος και επεξεργασίας του αρχείου weather.nominal με το εργαλείο Notepad++ και στην ΕΙΚΟΝΑ4 προβάλλεται το περιεχόμενο του με τη δομή που έχουν τα αρχεία ARFF .

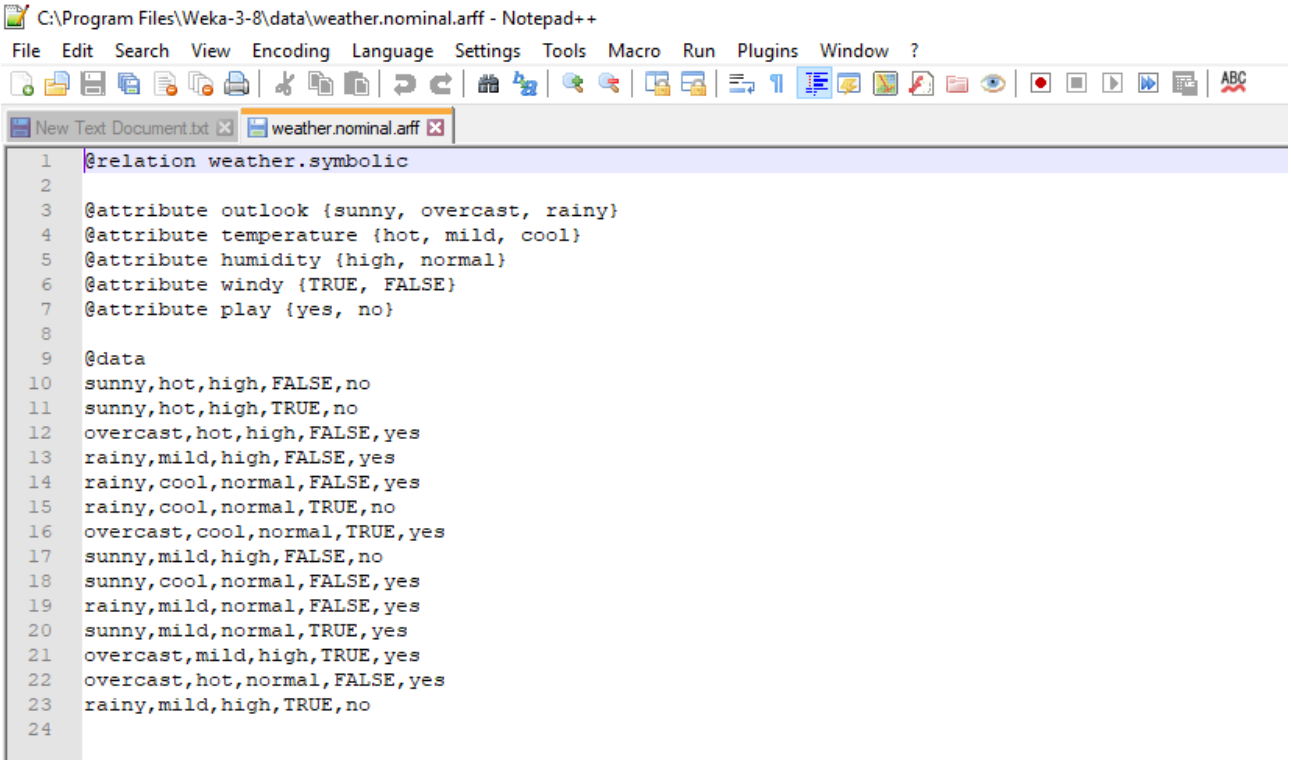
ΕΙΚΟΝΑ 3

Άνοιγμα αρχείου ARFF με το Notepad++



ΕΙΚΟΝΑ4

Περιεχόμενο αρχείου σε δομή ARFF

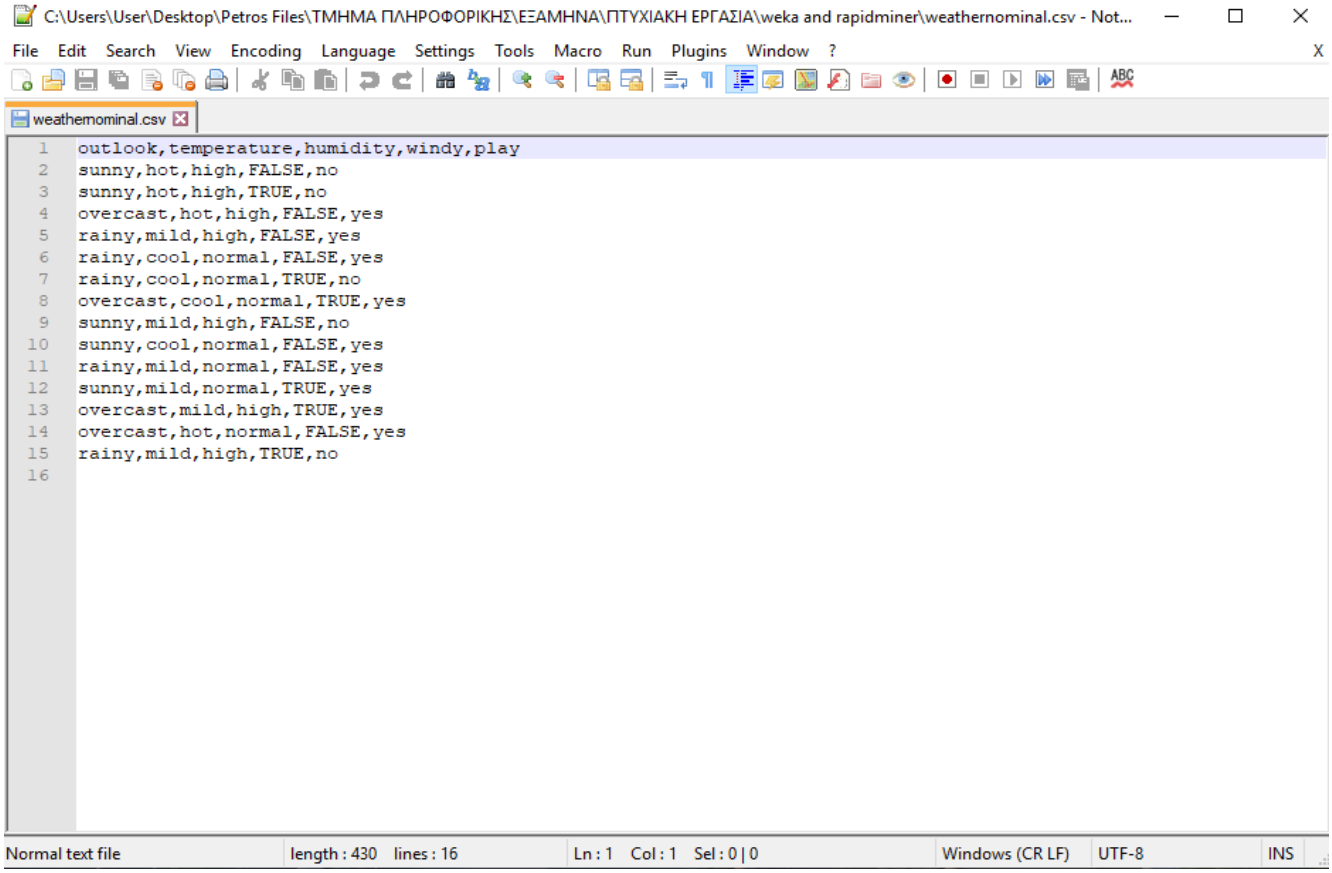


Το περιεχόμενο ενός αρχείου ARFF πρέπει να είναι δομημένο ακριβώς όπως στην ΕΙΚΟΝΑ4. Παρατηρούμε ότι στην πρώτη γραμμή, χρησιμοποιώντας το @relation έχουμε δώσει το όνομα και λίγο πιο κάτω τοποθετούμε τα χαρακτηριστικά ,όπου πρώτα γράφουμε τη λέξη @attribute και στη συνέχεια το όνομα του χαρακτηριστικού και τις τιμές που αυτό δέχεται. Στο τέλος με τον συμβολισμό @data ξεχωρίζουμε τα δεδομένα μέσα στο αρχείο και όπως είναι προφανές κάθε στήλη των δεδομένων αντιστοιχεί στις τιμές ενός χαρακτηριστικού.Το τελευταίο χαρακτηριστικό play{yes,no}είναι η class που δίνει το αποτέλεσμα. Αυτό σημαίνει ότι ανάλογα με την κάθε περίπτωση, όπου εδώ αφορά τον καιρό, θα εξάγεται το αποτέλεσμα yes ή no, αν δηλαδή μπορεί να παίξει κάποιος έξω ή όχι.

Ένα αρχείο CSV είναι στο εσωτερικό του όπως φαίνεται στην ΕΙΚΟΝΑ5 και ανοίγει με την επιλογή (edit with Notepad++). Τα δεδομένα χωρίζονται με το σύμβολο (,) και η πρώτη γραμμή μας δείχνει τα χαρακτηριστικά του αρχείου, ενώ από τη δεύτερη γραμμή και ύστερα είναι τα δεδομένα. Πρόκειται για μία αρκετά απλή δομή συγκριτικά με αυτή των αρχείων ARFF.

ΕΙΚΟΝΑ 5

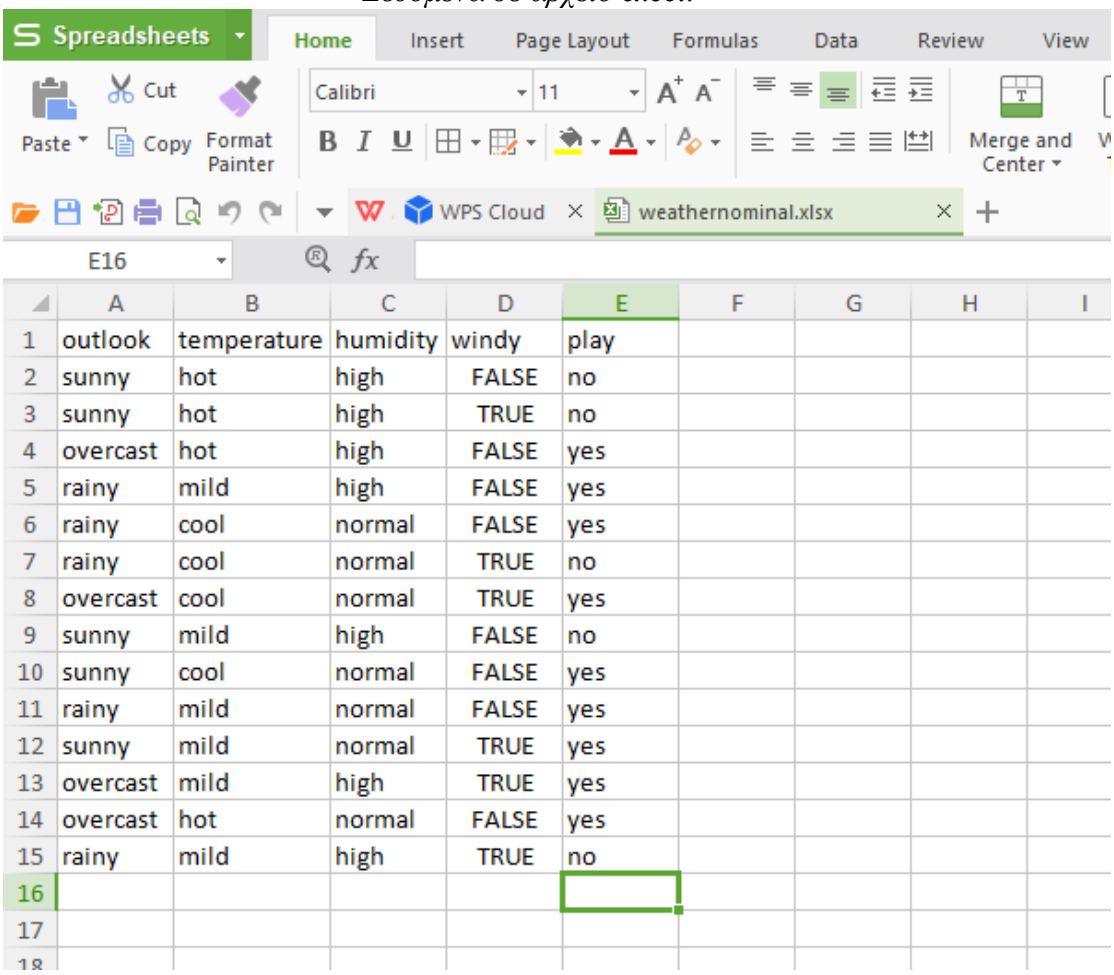
Το περιεχόμενο του αρχείου CSV



Επομένως αν έχουμε ένα αρχείο excel, τα δεδομένα μας χωρίζονται σε στήλες με κάθε στήλη να έχει στην πρώτη γραμμή το χαρακτηριστικό που την αντιπροσωπεύει και από κάτω τα δεδομένα που ανήκουν σε αυτήν. Αποθηκεύουμε ένα αρχείο excel με δεξί κλικ αποθήκευση ως CSV αλλά **προσέχουμε** κατά την αποθήκευση να **μην** το αποθηκεύσουμε σε UTF-8 CSV γιατί δεν θα μπορούμε να το φορτώσουμε στο λογισμικό WEKA για περαιτέρω χρήση. Το αρχείο CSV θα είναι όπως δείξαμε στην ΕΙΚΟΝΑ5, το οποίο επεξεργαζόμαστε για να το φτιάξουμε με την σωστή δομή ΕΙΚΟΝΑ4, που έχουν τα ARFF και το αποθηκεύουμε βάζοντας στο τέλος του ονόματος του την κατάληξη **.arff** . Παρακάτω ΕΙΚΟΝΑ6 φαίνονται τα δεδομένα του παραδείγματος weather.nominal όπως τοποθετούνται μέσα σε ένα αρχείο Excel.

ΕΙΚΟΝΑ 6

Δεδομένα σε αρχείο excel.

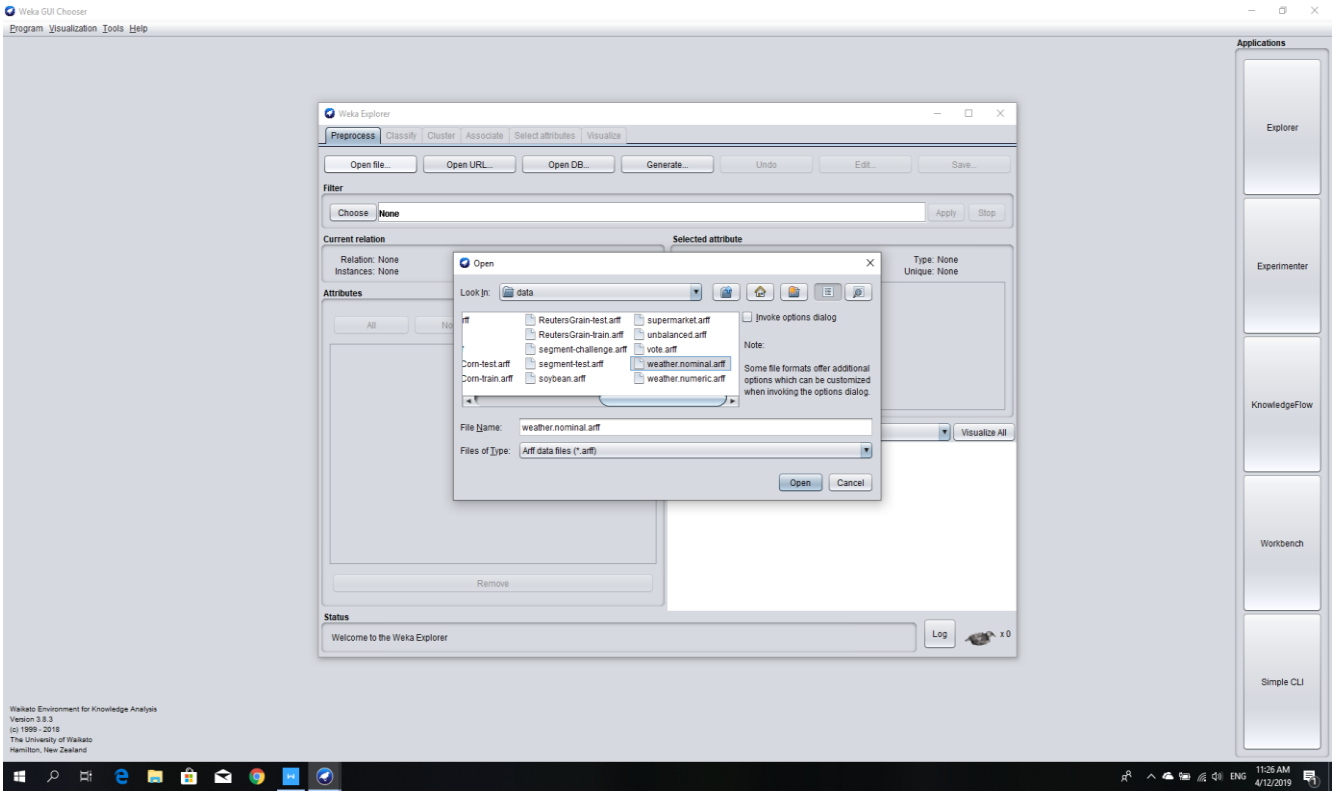


	A	B	C	D	E
1	outlook	temperature	humidity	windy	play
2	sunny	hot	high	FALSE	no
3	sunny	hot	high	TRUE	no
4	overcast	hot	high	FALSE	yes
5	rainy	mild	high	FALSE	yes
6	rainy	cool	normal	FALSE	yes
7	rainy	cool	normal	TRUE	no
8	overcast	cool	normal	TRUE	yes
9	sunny	mild	high	FALSE	no
10	sunny	cool	normal	FALSE	yes
11	rainy	mild	normal	FALSE	yes
12	sunny	mild	normal	TRUE	yes
13	overcast	mild	high	TRUE	yes
14	overcast	hot	normal	FALSE	yes
15	rainy	mild	high	TRUE	no
16					
17					

Αφού έχουμε ολοκληρώσει όλα τα παραπάνω βήματα, τώρα απομένει να φορτώσουμε ένα αρχείο ARFF στο λογισμικό WEKA και να δούμε πως παρουσιάζονται τα χαρακτηριστικά (attributes), και τα δεδομένα (data) του αρχείου weather.nominal, μέσα σε αυτό. Αρχικά ανοίγουμε το WEKA, γράφοντας στην αναζήτηση το όνομα του και κάνουμε κλικ στην επιλογή Explorer. Στη συνέχεια, πατάμε κλικ στο πάνω αριστερό μέρος με όνομα (Open file), για να εντοπισουμε και να ανοίξουμε το αρχείο με το οποίο θα πειραματιστούμε. Στην παρακάτω εικόνα (ΕΙΚΟΝΑ7) παρουσιάζεται ο τρόπος αυτός. Έχοντας πλέον εισάγει το αρχείο, παρατηρούμε ότι στην αριστερή πλευρά εμφανίζονται τα χαρακτηριστικά (attributes). Κάνοντας κλικ σε ένα από τα outlook, temperature, humidity, windy, play, βλέπουμε ότι όλα έχουν 14 instances. Στην δεξιά πλευρά, εμφανίζονται τα δεδομένα που αντιστοιχούν σε κάθε χαρακτηριστικό. Για παράδειγμα, για το attribute outlook, έχουμε τις τιμές sunny (5), overcast (4) και rainy (5). Τα διαγράμματα με κόκκινο και μπλε χρώμα στο κάτω δεξί μέρος κάνουν μία οπτικοποίηση για το πόσο πιθανό είναι το yes και το no ανάλογα με την κάθε περίπτωση. Στην ΕΙΚΟΝΑ8 φαίνονται όλα τα παραπάνω.

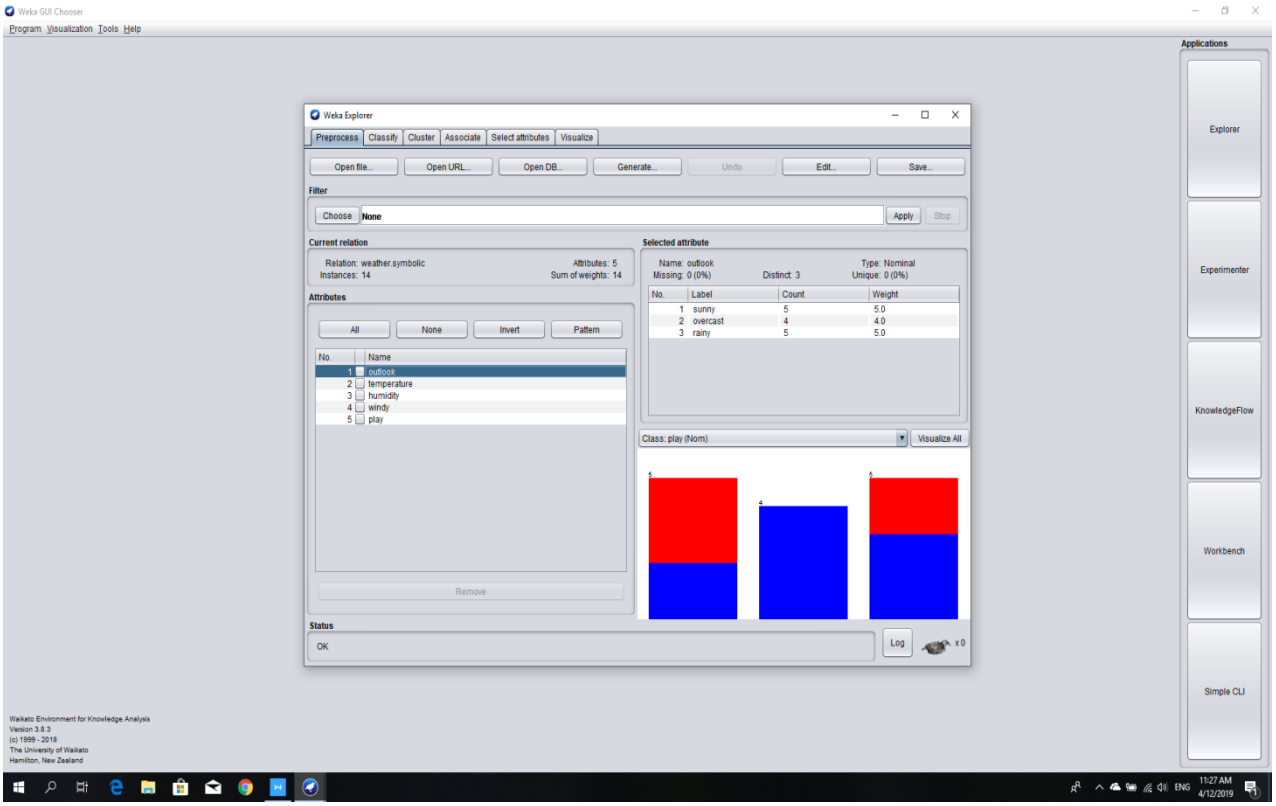
ΕΙΚΟΝΑ7

Φόρτωση αρχείου ARFF στο WEKA



ΕΙΚΟΝΑ8

Παρουσίαση χαρακτηριστικών του weather.nominal



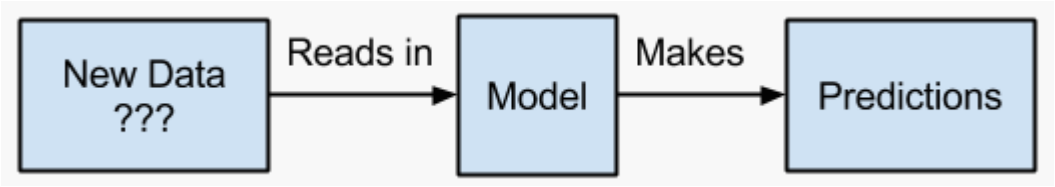
6 ΠΑΡΟΥΣΙΑΣΗ ΔΥΝΑΤΟΤΗΤΩΝ ΤΟΥ WEKA

6.1 Στόχος

Σε αυτό το σημείο, γίνεται λεπτομερής περιγραφή των δυνατοτήτων του λογισμικού WEKA, πρώτα όμως ο χρήστης θα πρέπει να γνωρίζει ορισμένες, σημαντικές έννοιες που θα ακολουθήσουν. Σκοπός του δεύτερου μέρους είναι να μπορεί κανείς να εξοικειωθεί με τη χρήση του εργαλείου αυτού, μαθαίνοντας πολύ δημοφιλείς αλγορίθμους της εξόρυξης δεδομένων. Αναλυτικότερα, να έχει τη δυνατότητα ο χρήστης, να μάθει την λογική που χρησιμοποιείται από τους αλγορίθμους μηχανικής μάθησης και εξόρυξης και πώς γίνονται οι προβλέψεις, να διακρίνει τις διαφορές των όρων Classification και Regration και να κάνει διαφορετικά πειράματα. Για κάθε κατηγορία αλγορίθμων εξηγείται η λογική που χρησιμοποιεί ο αλγόριθμος.

6.2 Προγνωστική μοντελοποίηση (Predictive modeling)

Ο όρος predictive modeling[9] σημαίνει, η δυνατότητα να προβλέψουμε με όσο το δυνατόν μεγαλύτερη ακρίβεια, ένα αποτέλεσμα, έχοντας ως εφόδιο ένα σύνολο από δεδομένα. Αν για παράδειγμα μέσα σε ένα αρχείο έχουμε αποθηκευμένα τα στοιχεία διαφορετικών φυτών, τότε θέλουμε μέσα από αυτά τα χαρακτηριστικά να μπορεί ο αλγόριθμος να προβλέψει το είδος του φυτού. Το πρόβλημα αυτό ονομάζεται supervised learning και ο στόχος είναι να πάρουμε κάποια δεδομένα που σχετίζονται μεταξύ τους για να δημιουργήσουμε ένα μοντέλο μέσα από τις σχέσεις τους. Συγκεκριμένα μέσα στο WEKA εισάγουμε νέα δεδομένα ή αλλιώς ένα αρχείο ARFF, έπειτα με την χρήση κάποιου αλγορίθμου γίνονται υπολογισμοί και στο τέλος εξάγεται ένα αποτέλεσμα που αποκαλούμε ως προβλεπόμενο αποτέλεσμα.



6.3 Η λογική των αλγορίθμων μηχανικής μάθησης

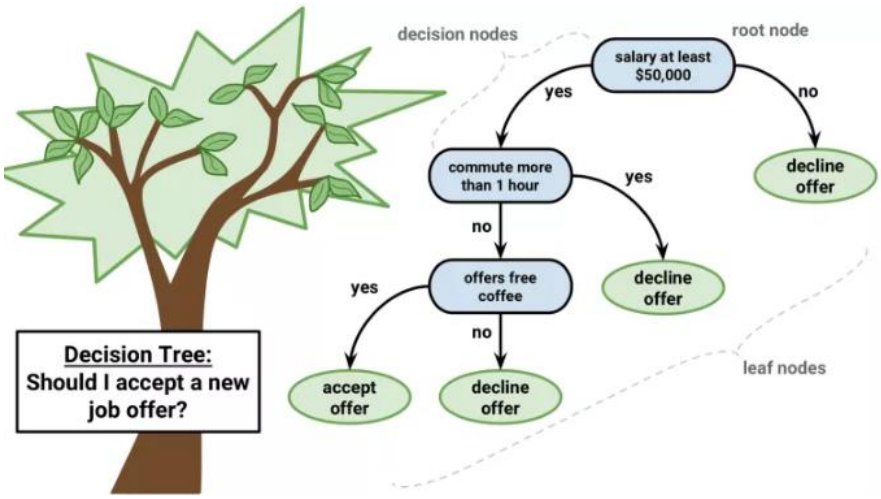
Οι αλγόριθμοι μηχανικής μάθησης[10] προσπαθούν να ‘μάθουν’ μία συνάρτηση στόχος f, η οποία δέχεται μεταβλητές x και προσπαθεί να τις αντιστοιχίσει σε μεταβλητές εξόδου y, δηλαδή y = f(x). Γενικότερα θεωρούμε ότι η y αναφέρεται στο μέλλον, το οποίο θέλουμε να προβλέψουμε δίνοντας ως είσοδο νέες τιμές x. Δεν γνωρίζουμε εκ των προτέρων την συνάρτηση f, γιατί αν το ξέραμε αυτό τότε δεν θα χρειαζόταν καν να γίνονται οι προβλέψεις, αλλά πρέπει επίσης να συμπεριλάβουμε και τα λάθη που μπορούν πολλές φορές να συμβούν. Τα λάθη αυτά προκύπτουν επειδή δεν είναι πάντα δυνατό, να γίνεται τέλεια αντιστοίχιση των x με τα y από τους αλγορίθμους μηχανικής μάθησης. Επομένως ο τελικός τύπος που περιγράφει τη λογική αυτή είναι y = f(x) + e, όπου το e = error.

[9] https://machinelearningmastery.com/gentle-introduction-to-predictive-modeling/
[10] https://machinelearningmastery.com/how-machine-learning-algorithms-work/

6.4 Δέντρα αποφάσεων (Decision tree algorithms)

Οι αλγόριθμοι της κατηγορίας δέντρα αποφάσεων[11] ανήκουν στην οικογένεια αλγορίθμων supervised learning και μπορούν να χρησιμοποιηθούν για την επίλυση προβλημάτων ταξινόμησης και οπισθοδρόμησης. Ο γενικότερος σκοπός για να χρησιμοποιήσει κανείς ένα δέντρο απόφασης είναι για την δημιουργία ενός μοντέλου εκπαίδευσης, το οποίο είναι αναγκαίο για την πρόβλεψη των αποτελεσμάτων ενός σετ με δεδομένα, τα οποία πρόκειται μετά την χρήση του αλγορίθμου να μας δώσουν σημαντική πληροφορία. Η κατανόηση του τρόπου λειτουργίας ενός αλγορίθμου από την ομάδα των δέντρων αποφάσεων είναι μία απλή διαδικασία σε σύγκριση με αλγορίθμους ταξινόμησης άλλων ομάδων. Κάθε δέντρο αποτελείται από τη ρίζα, τους κόμβους, τα κλαδιά και τα φύλλα. Οι εσωτερικοί κόμβοι αντιστοιχούν σε χαρακτηριστικά (attributes) του συνόλου δεδομένων μας, ενώ τα φύλλα αντιστοιχούν στις τιμές της class. Για να γίνει αυτό πιο κατανοητό με ένα πολύ απλό παράδειγμα, όπως φαίνεται στην επόμενη ΕΙΚΟΝΑ9.

ΕΙΚΟΝΑ9
Ένα παράδειγμα δέντρου απόφασης



Σε αυτό το δέντρο απόφασης η τιμή της class είναι accept offer, decline offer και εμφανίζεται στα φύλλα του. Οι εσωτερικοί κόμβοι έχουν κάποια χαρακτηριστικά και η ρίζα έρχεται με το όνομα root node. Συγκεκριμένα στην πρώτη φάση αν η επιλογή είναι no θα οδηγηθούμε σε φύλλο ενώ αν είναι yes, συνεχίζει η εκτέλεση προς τα κάτω από αριστερά, όπου επαναλαμβάνεται η ίδια διαδικασία.
Κατά την χρήση ενός δέντρου απόφασης, πρέπει να αναγνωρίζουμε ποια χαρακτηριστικά θα θεωρήσουμε ως το root node σε κάθε επίπεδο του δέντρου. Αν ένα σετ δεδομένων περιλαμβάνει (n) χαρακτηριστικά, το να αποφασίσουμε πιο από αυτά θα τοποθετήσουμε στη ρίζα και ποια στους εσωτερικούς κόμβους είναι ένα πολύπλοκο βήμα. Η τυχαία επιλογή ενός κόμβου, ως ρίζα του δέντρου δεν είναι η λύση και μπορεί να οδηγήσει σε ανεπιθύμητα αποτελέσματα και χαμηλή ακρίβεια (accuracy). Για να γίνει αυτό με ορθό τρόπο, έχουν αναπτυχθεί δύο γνωστές μέθοδοι που αποκαλούνται

[11] http://dataaspirant.com/2017/01/30/how-decision-tree-algorithm-works/

information gain και gini index. Αυτές μέσω των κατάλληλων υπολογισμών των τιμών, τις ταξινομούν και έτσι τα attributes εισάγονται στο δέντρο με σωστή σειρά.

6.5 Information gain

Χρησιμοποιώντας το κριτήριο information gain προσπαθούμε να εκτιμήσουμε την πληροφορία, η οποία περιέχεται σε κάθε χαρακτηριστικό. Από την 'Θεωρία Πληροφορίας' είναι γνωστό πως η τυχαιότητα ή αβεβαιότητα μίας μεταβλητής (X) ορίζεται από την 'Εντροπία'. Αν όλα τα παραδείγματα του σετ δεδομένων μας είναι θετικά ή αν όλα είναι αρνητικά τότε η εντροπία θα ισούται με 0. Αν τα μισά από τα δεδομένα μας έχουν θετική class, ενώ τα υπόλοιπα αρνητική τότε η εντροπία ισούται με 1. Επομένως, υπολογίζοντας το μέτρο της εντροπίας κάθε attribute, τότε μπορεί να υπολογιστεί και το information gain του. Αυτή είναι η μία από τις δύο μεθόδους λειτουργίας των δέντρων απόφασης για την ορθή τοποθέτηση των κόμβων και την καλύτερη ακρίβεια των αποτελεσμάτων. Ο τύπος για τον υπολογισμό της εντροπίας είναι ο ακόλουθος.

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

6.6 Gini index

Το Gini index είναι ένα μέτρο που μας εμφανίζει πόσο συχνά ένα στοιχείο που έχει επιλεχθεί με τυχαίο τρόπο, έχει αναγνωριστεί λάθος (incorrectly identified). Αυτό σημαίνει ότι ένα χαρακτηριστικό με χαμηλότερο Gini index θα πρέπει να προτιμάται. Έστω ότι έχουμε ένα χαρακτηριστικό με όνομα A και μία class. Οι τιμές τους φαίνονται παρακάτω.

A	class
4.8	positive
5	positive
5	negative
5.2	positive
5.2	positive
4.7	negative

Αρχικά ομαδοποιούμε τις τιμές του A σε >=5 και σε <5. Από αυτές, οι 4/6 τιμές είναι >=5, ενώ οι άλλες 2/6 είναι <5. Στη συνέχεια, >=5 & class = positive είναι οι 3/4, ενώ >=5 & class = negative 1/4. Υπολογίζουμε το gini index ως εξής:
Gini(3,1) = 1 - ((3/4)2 + (1/4)2) = ... Με τον ίδιο τρόπο κάνουμε και για τα στοιχεία που είναι <5. Gini(1,1) = 1 - ((1/2)2 + (1/2)2) = …
Τελικός τύπος για το A είναι gini(Target,A) =(4/6) * gini(3,1) + (2/6) * gini (1,1) = …
Το attribute με το μικρότερο gini index γίνεται η ρίζα του ΄δεντρου.
Οι αλγόριθμοι δέντρων αποφάσεων αποτελούν μία χρήσιμη μέθοδο για την μηχανική μάθηση και εξόρυξη πληροφορίας, η οποία είναι πολύ απλή στην εξήγηση της. Τα δέντρα απόφασης ακολουθούν την ίδια λογική σειρά βημάτων που χρησιμοποιούν και οι άνθρωποι για να λαμβάνουν αποφάσεις στην καθημερινότητα τους. Ακόμα και ένα πιο σύνθετο δέντρο μπορεί να απλοποιηθεί μέσω των οπτικοποιήσεών του. Παρόλα αυτά, σε ορισμένες περιπτώσεις οι υπολογισμοί μπορεί να είναι πάρα πολλοί, όταν

στα δεδομένα μας έχουμε πολλά class labels. Τέλος, είναι πολύ πιθανό να γίνει υπερφόρτωση σε ένα δέντρο και τότε χρησιμοποιούμε τεχνικές κλαδέματος (pre-pruning, post pruning) για να αντιμετωπιστεί το πρόβλημα, που θα δείξουμε παρακάτω.

6.7 J48

Ο αλγόριθμος J48[12] αποτελεί επέκταση του αλγορίθμου ID3. Τα πρόσθετα χαρακτηριστικα του J48 είναι το κλάδεμα δέντρων αποφάσεων, η καταγραφή ελλειπουσών τιμών, η παραγωγή κανόνων κ.α. Μέσα στο εργαλείο WEKA μπορούμε να χρησιμοποιήσουμε τον J48, όπου αυτός δημιουργεί κανόνες μέσω των οποίων, παράγεται συγκεκριμένη ταυτότητα των δεδομένων. Ο στόχος του αλγορίθμου είναι σταδιακά η γενίκευση ενός δέντρου απόφασης μέχρι να αποκτήσει ισορροπία, ευελιξία και ακρίβεια. Σε περίπτωση που τα στοιχεία ανήκουν στην ίδια class τότε το δέντρο αναπαριστά ένα φύλλο έτσι ώστε το φύλλο να ταυτίζεται με την ίδια class. Η πιθανότητα της πληροφορίας υπολογίζεται για κάθε χαρακτηριστικό (attribute), κάνοντας ένα τεστ στο συγκεκριμένο attribute. Τότε υπολογίζεται το information gain, το οποίο θα είναι το αποτέλεσμα του τεστ. Όπως, έχει ήδη αναφερθεί, το attribute, με το μεγαλύτερο information gain, που υπολογίζεται σύμφωνα με τον τύπο της εντροπίας, θα γίνει η ρίζα του δέντρου. Ωστόσο, ορισμένες εμφανίσεις υπάρχουν σε όλα τα σύνολα των δεδομένων, που δεν είναι καλά καθορισμένα, με αποτέλεσμα να έχουμε υπερβολικές τιμές. Γι' αυτό τον λόγο γίνεται το κλάδεμα, μειώνοντας τα σφάλματα ταξινόμησης που παράγονται από το σετ εκπαίδευσης. Συνοπτικά ο αλγόριθμος λειτουργεί με 3 βασικά βήματα.

- Μία τιμή επιλέγεται ως οριακή τιμή (threshold value) ή αλλιώς κατώφλι. Αυτή η τιμή διαχωρίζει το σύνολο δεδομένων σε τιμές η οποίες είναι πάνω από το όριο και σε τιμές που βρίσκονται κάτω από αυτό
- Ο αλγόριθμος διαχειρίζεται και τις τιμές που παραλήφθηκαν στο σετ εκπαίδευσης
- Μετά την πλήρη κατασκευή του δέντρου, ο αλγόριθμος εκτελεί το κλάδεμα του δέντρου. Στο τέλος προσπαθεί να αφαιρέσει εκείνα τα κλαδιά τα οποία δεν βοηθούν στο να φτάσει σε κόμβους φύλλα.

Φτάνοντας σε αυτό το σημείο, μπορούμε να κατανοήσουμε πως λειτουργούν τα δέντρα αποφάσεων, να διακρίνουμε τις διαφορές του information gain και του gini index και πως αυτά υπολογίζονται. Επιπρόσθετα, είδαμε ότι ο αλγόριθμος J48 βασίζεται στην λογική του information gain και ότι χρησιμοποιεί pruning στο σετ εκπαίδευσης μετά την κατασκευή του δέντρου. Στη συνέχεια της εργασίας θα παρουσιαστεί αυτή η διαδικασία μέσα στο WEKA.

[12] Improved J48 Classification Algorithm for the Prediction of Diabetes (PDF)

6.8 Εκτέλεση αλγορίθμων διαφορετικών οικογενειών

Αφού έχουμε ήδη δείξει τον τρόπο με τον οποίο ένα σετ δεδομένων μπορεί να εισαχθεί στο WEKA, τώρα θα παρουσιάσουμε λεπτομερώς, ορισμένες δυνατότητες που μας προσφέρει το λογισμικό. Συγκεκριμένα, στο πάνω αριστερό μέρος υπάρχει μία λίστα με επιλογές (Preprocess, Classify, Cluster,κ.α.), όπου εμείς κάνουμε κλικ στο Classify, αφού πρώτα έχουμε τοποθετήσει ένα αρχείο arff. Μέσα στο αριστερό κόκκινο περίγραμμα ΕΙΚΟΝΑ10, φαίνονται οι διαφορετικοί τρόποι που μπορούμε να χρησιμοποιήσουμε για να εξάγουμε αποτελέσματα στα πειράματά μας.

Αναλυτικά:

- Use training set: Χρησιμοποιείται το αρχείο που έχουμε εισάγει στο εργαλείο και μαζί με έναν αλγόριθμο της επιλογής μας, εξάγεται το ανάλογο αποτελέσματα
- Supplied test set: Εκτός του σετ εκπαίδευσης, εισάγουμε και ένα τεστ σετ ώστε να δούμε πόσο υψηλή ακρίβεια πετυχαίνει το λογισμικό δίνοντας του ένα τέτοιο σετ.
- Cross validation: Κατά τη διαδικασία cross validation, το αρχείο που έχουμε εισάγει χωρίζεται σε κομμάτια. Στην περίπτωση της ΕΙΚΟΝΑ10, σε 10 κομμάτια όπου το 1 από αυτά θα χρησιμοποιηθεί ως test set, ενώ τα άλλα 9 θα χρησιμοποιηθούν ως training set. Αυτή η διαδικασία θα επαναληφθεί μέχρι όλα τα κομμάτια του συνόλου να περάσουν από την θέση του test set. Στο WEKA μπορούμε να θέσουμε εμείς σε πόσα κομμάτια θα χωριστεί το σετ μας.
- Percentage split: Σε αυτή την περίπτωση, ένα ποσοστό του συνόλου χρησιμοποιείται ως training set ενώ το υπόλοιπο ποσοστό ως test set. Στην ΕΙΚΟΝΑ10 το 66% αναφέρεται στο training set.

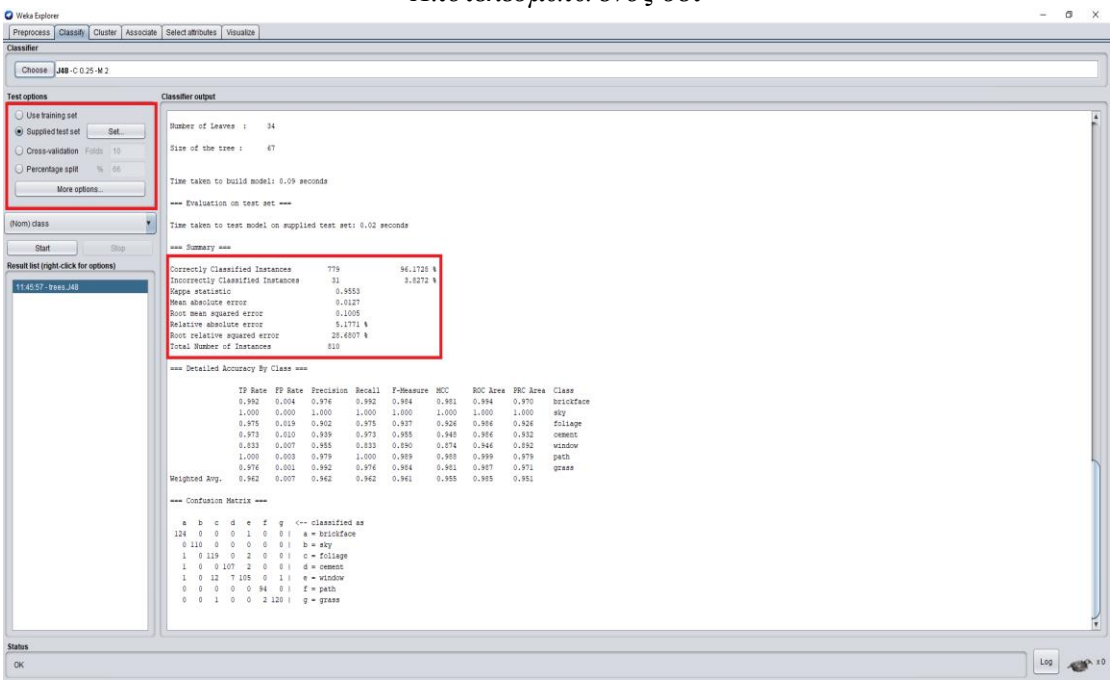
Για να μπορέσουμε να ‘τρέξουμε’ το πείραμα, θα πρέπει πρώτα να εισάγουμε έναν από τους αλγορίθμους που υπάρχουν στο WEKA. Πάνω από το κόκκινο πλαίσιο ΕΙΚΟΝΑ10, εμφανίζεται το κουμπί Choose, και πατώντας σε αυτό ανοίγουν οι διαφορετικές ομάδες αλγορίθμων. Στον ομάδα των δέντρων (trees), βρίσκεται ο αλγόριθμος J48 που είχαμε εξηγήσει πιο πάνω στην ενότητα “Δέντρα αποφάσεων”. Επιλέγοντας αυτόν τον αλγόριθμο και πατώντας στο Supplied test set, μπορούμε πηγαίνοντας πάλι στον φάκελο με τα data του WEKA, που παρέχονται με την εγκατάσταση, να πάρουμε ένα αρχείο arff ως test set. Στο παράδειγμα ΕΙΚΟΝΑ10 ως training set έχει τοποθετηθεί το segment-challenge.arff και ως test set το segment-test.arff. Στη συνέχεια, κάνοντας κλικ στο κουμπί Start, ‘τρέχει’ το πείραμα μας και στο κάτω αριστερό πλαίσιο φαίνεται πότε εκτελέστηκε.

Στο δεξί μέρος με όνομα Classifier output, μπορούμε να δούμε το ποσοστό των στοιχείων που ταξινομήθηκαν σωστά από τον αλγόριθμο και το ποσοστό των στοιχείων που ταξινομήθηκαν λάθος, τα οποία αντίστοιχα είναι τα Correctly classified instances και τα Incorrectly classified instances μέσα στο κόκκινο πλαίσιο ΕΙΚΟΝΑ10. Λίγο πιο κάτω (Detailed accuracy by class), φαίνεται αναλυτικά για κάθε class, πόση ακρίβεια έχουμε στο True Positive Rate, στο False Positive καθώς και στις ακόλουθες στήλες. Στο τέλος, φαίνεται ο πίνακας Confusion Matrix που μας δείχνει για την τιμή κάθε class πόσα στοιχεία έχουν βρεθεί σωστά και πόσα λάθος. Για παράδειγμα για το class brickface, που έχει 0.992 TP rate, βλέπουμε ότι στην στήλη a του confusion matix είναι τα 124 σωστά ενώ στην στήλη e βρίσκεται το ένα

λάθος ταξινομημένο στοιχείο. Αυτό είναι λογικό αφού υπάρχει 0.004 FP rate για το ένα λάθος στοιχείο. Αντίστοιχα, αυτό ισχύει και για τις υπόλοιπες στήλες και τα class.

ΕΙΚΟΝΑ10

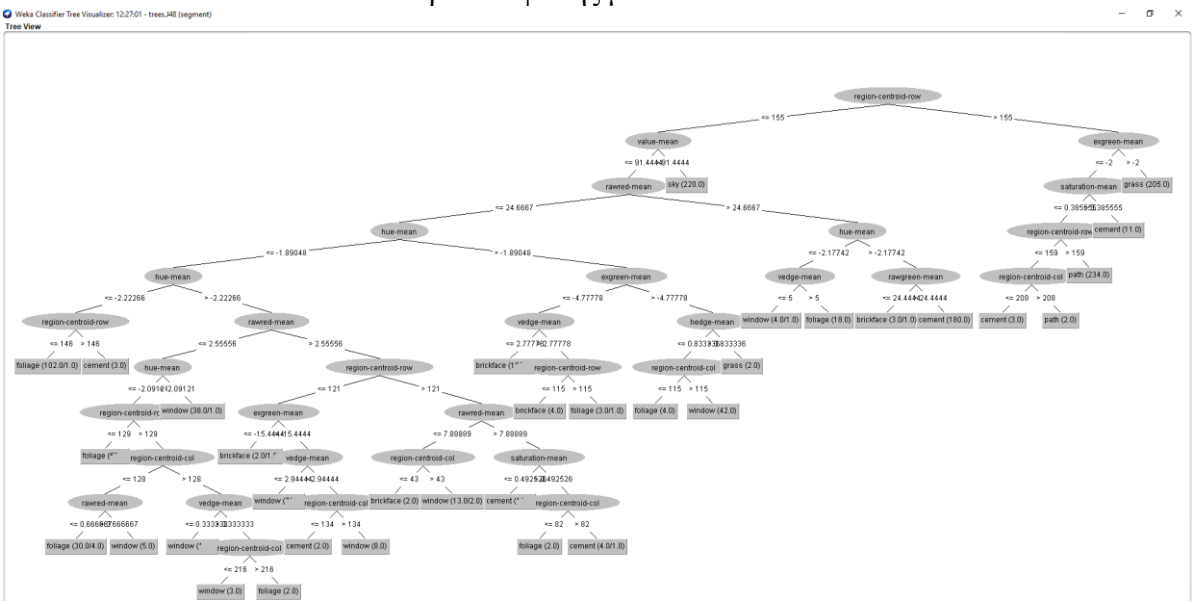
Αποτελέσματα ενός σετ



6.9 Οπτικοποίηση του δέντρου και των στοιχείων με λάθος ταξινόμηση

Για να δούμε το δέντρο που έχει παράγει ο αλγόριθμος J48, κάνουμε δεξί κλικ πάνω στο σημείο που φαίνεται η χρονική στιγμή που τρέξαμε το πείραμα και πατάμε την επιλογή visualize tree. Στην ΕΙΚΟΝΑ10 το σημείο αυτό είναι με σκούρο μπλε χρώμα. Έπειτα, εμφανίζεται το δέντρο, και πατάμε δεξί κλικ και την επιλογή Fill to screen για να μπορούμε να το δούμε σε πλήρη οθόνη. Όπως έχει αναφερθεί, ο αλγόριθμος J48 λειτουργεί με το information gain για να ορίσει τον κόμβο ρίζα του δέντρου, τους εσωτερικούς κόμβους και να υλοποιήσει την διαδικασία της ταξινόμησης. Στο παράδειγμα που έχουμε επιλέξει οι τιμές στα class αναφέρονται σε δοκιμά υλικά, τα οποία μπορούμε να δούμε στα φύλλα του δέντρου ΕΙΚΟΝΑ11. Τα φύλλα είναι τα ορθογώνια πλαίσια ενώ οι κόμβοι έχουν το σχήμα της έλλειψης.

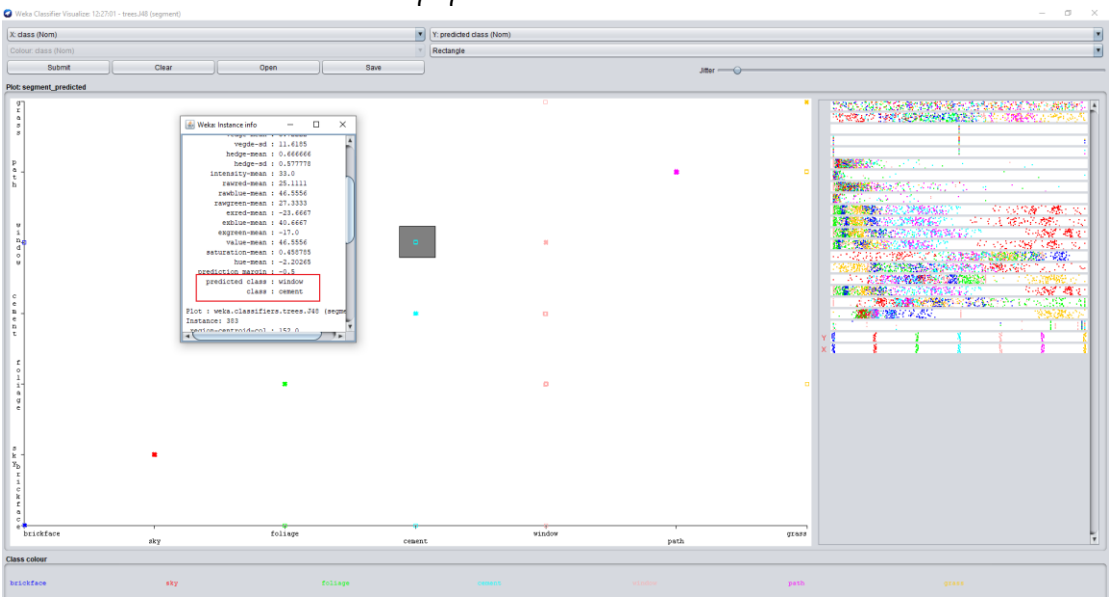
ΕΙΚΟΝΑ11
Δέντρο απόφασης με τον J48



Η οπτικοποίηση των στοιχείων που έχουν ταξινομηθεί λάθος από τον αλγόριθμο J48 μπορεί να γίνει με διαφορετικούς τρόπους. Ένας από αυτούς είναι να κάνουμε κλίκ ξανά στο σκούρο μπλε πλαίσιο που είχαμε κάνει και πριν για το tree visualization, μόνο που αυτή την φορά επιλέγουμε visualize classifier errors. Στην ΕΙΚΟΝΑ12 παρουσιάζονται 2 άξονες x,y με τα class και τα predicted class του παραδείγματος. Κάθε class αντιστοιχεί σε ένα χρώμα και κάθε σημείο μέσα στον χώρο είναι ένα απλό ή ένα πολλαπλό instance. Τα πολλαπλά instances έχουν λίγο πιο σκούρο χρώμα. Για να μπορέσουμε να διακρίνουμε καλύτερα τα instances αυξάνουμε λίγο προς τα δεξιά την μπάρα που ονομάζεται Jitter στο πάνω δεξί μέρος. Αυτό το κάνουμε γιατί κάποια instances ‘πέφτουν’ το ένα πάνω στο άλλο και δεν ξεχωρίζονται εύκολα με το μάτι. Στο πάνω μέρος, στο κέντρο μπορούμε να επιλέγουμε ένα από τα σχήματα rectangle, polygon, polyline για να περικυκλώσουμε την περιοχή που μας ενδιαφέρει να δούμε πιο αναλυτικά. Επιπλέον κάνοντας δεξί κλικ πάνω σε κάποιο instance, εμφανίζονται λεπτομερώς οι πληροφορίες του. Στην ΕΙΚΟΝΑ12 έχουμε ανοίξει τις πληροφορίες ενός πολλαπλού instance όπου μπορούμε να δούμε μέσα στο κόκκινο πλαίσιο ότι το predicted class είναι window ενώ το κανονικό class είναι cement. Τα στοιχεία που έχουν ταξινομηθεί λάθος έχουν το σχήμα από ένα μικρό τετράγωνο.

ΕΙΚΟΝΑ12

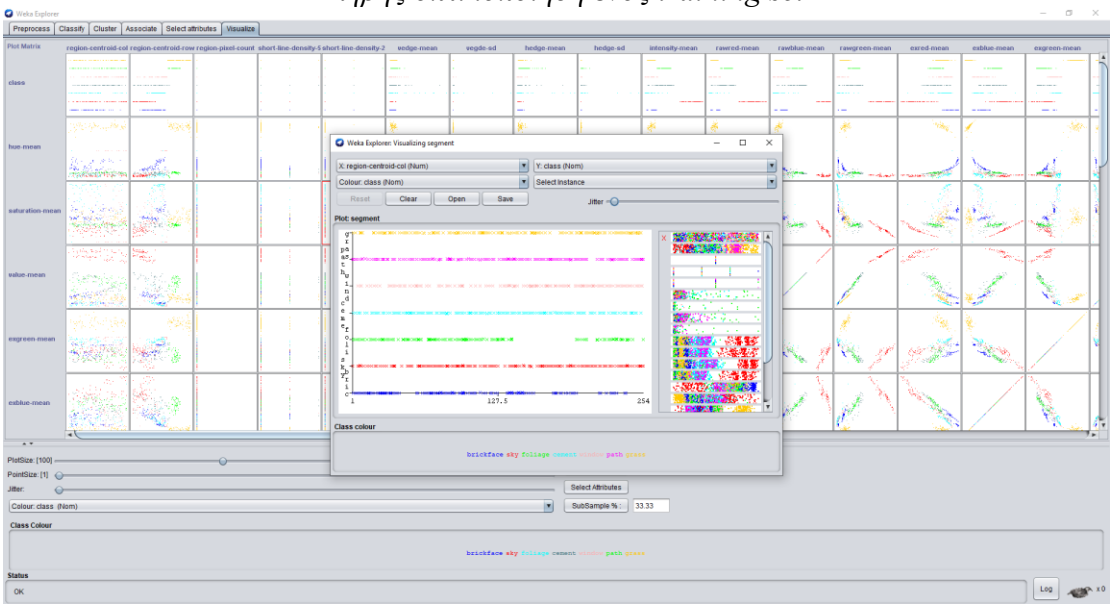
Οπτικοποίηση των instances και των errors



Ένας ακόμα τρόπος για την οπτικοποίηση που αφορά την οπτικοποίηση κάθε attribute γίνεται με την επιλογή Visualize που βρίσκεται στην λίστα με τις επιλογές (Preprocess, Classify, … , Visualize). Με αυτόν τον τρόπο μπορούμε να δούμε, που έχουν ταξινομηθεί τα instances σε κάθε περίπτωση. Αν θυμηθούμε το πρώτο, απλό παράδειγμα που αφορούσε τον καιρό, εκεί θα μπορούσαμε να δούμε για παράδειγμα ποια instances τοποθετούνται στον χώρο στην class yes στην no και ποια είναι η τιμή του χαρακτηριστικού outlook (sunny, windy, rainy). Η οπτικοποίηση γίνεται πολύ αναλυτικά με αυτή την επιλογή και παρουσιάζει όλους τους συνδιασμούς που μπορούν να πραγματοποιηθούν. Η ΕΙΚΟΝΑ13 παρουσιάζει την οπτικοποίηση του παραδείγματος που χρησιμοποιήσαμε με τα σετ segment-challenge και segment-test.

ΕΙΚΟΝΑ13

Πλήρης οπτικοποίηση ενός training set



6.10 Classification και Regression

Πριν περάσουμε στο κομμάτι των επιπρόσθετων αλγορίθμων του WEKA, είναι σημαντικό να αφιερώσουμε λίγο χρόνο στις διαφορές που βρίσκονται ανάμεσα στους όρους classification και regression[13] predictive modeling. Η έννοια του classification απευθύνεται σε μία συνάρτηση f που προσπαθεί να αντιστοιχίσει τις τιμές εισόδου x σε διακριτές τιμές εξόδου y, οι οποίες (y) αποκαλούνται συνήθως ως labels. Για παράδειγμα, ένα email μπορεί να ταξινομηθεί ως 'spam' ή not 'spam'. Ένα πρόβλημα ταξινόμησης, απαιτεί παραδείγματα τα οποία μπορούν να ταξινομηθούν σε μία ή σε περισσότερες classes. Είναι σύνηθες, για τα μοντέλα ταξινόμησης να προβλέπουν μία τιμή ως την πιθανότητα ενός δοσμένου παραδείγματος που ανήκει σε κάθε output class. Μια πιθανότητα πρόβλεψης μπορεί να μετατραπεί σε μία τιμή της class, η οποία (class) έχει την υψηλότερη πιθανότητα πρόβλεψης. Για να γίνει πιο σαφές, αν ένα κείμενο ενός email έχει τις πιθανότητες 0.1 για 'spam' και 0.9 για 'not spam', τότε, αυτές μετατρέπονται σε class labels και επιλέγεται από τον αλγόριθμο το label 'not spam' ως η υψηλότερη προβλεπόμενη πιθανότητα. Όσον αφορά την έννοια regression, εδώ μία συνάρτηση f προσπαθεί να αντιστοιχίσει τις τιμές εισόδου x σε συνεχή τιμή y. Μία συνεχής τιμή είναι συνήθως μία real ή integer value όπως για παράδειγμα ποσότητα ή μέγεθος. Ωστόσο, ένα πρόβλημα regression απαιτεί την πρόβλεψη μιας ποσότητας, γεγονός που σημαίνει ότι όταν εκκινήσουμε τον αλγόριθμο τότε πρέπει να εξάγεται ένα αποτέλεσμα (report) που να αναφέρεται ως error. Για παράδειγμα στο WEKA το RMSE(Root Mean Squared Error) μας δίνει την προβλεπόμενη τιμή. Έστω ότι το predictive model έχει κάνει 2 προβλέψεις, από τις οποίες στην πρώτη 'βγάζει' 1.5 ενώ κανονικά η τιμή είναι 1.0 και στην δεύτερη 3.3 ενώ κανονικά η τιμή είναι 3.0, τότε το RMSE = sqrt(((1.0-1.5)^2 + (3.0 - 3.3)^2) / 2). RMSE = 0.412. Οι αλγόριθμοι που μπορούν να υπολογίσουν το regression, ονομάζονται regression algorithms, ενώ οι αλγόριθμοι ταξινόμησης classification algorithms.

6.11 Αλγόριθμος ZeroR στο WEKA της κατηγορίας rules

Ο αλγόριθμος που θέτει ένα όριο ή κατώφλι στα προβλήματα ταξινόμησης και παλινδρόμησης, ονομάζεται Zero Rule[14] algorithm και είναι ο default αλγόριθμος που χρησιμοποιείται στο WEKA. Για ένα πρόβλημα παλινδρόμησης, όπου πρόκειται να προβλεφθεί μία αριθμητική τιμή, ο αλγόριθμος ZeroR προβλέπει τον μέσο όρο του σετ εκπαίδευσης. Αν για παράδειγμα έχουμε ένα σετ με τιμές σπιτιών σε χιλιάδες ευρώ ο αλγόριθμος αυτός θα εξάγει ένα αποτέλεσμα που θα είναι ο μέσος όρος του σετ με τις τιμές και ένα RMSE. Ωστόσο για ένα πρόβλημα ταξινόμησης, όπου οι τιμές δεν είναι αριθμητικές, ο αλγόριθμος θα προβλέψει την τιμή που έχει τις περισσότερες εμφανίσεις μέσα στο σετ εκπαίδευσης. Ας χρησιμοποιήσουμε ένα σετ από τον φάκελο data που μας δίνει το WEKA μαζί με την εγκατάσταση. Το σετ αυτό είναι το diabetes.arff και για να δούμε περισσότερες πληροφορίες για το περιεχόμενο του μπορούμε να το ανοίξουμε με το Wordpad. Συνολικά αποτελείται από 9 attributes και οι τιμές που παίρνει το class είναι tested positive και tested negative. Εδώ αναφερόμαστε σε ένα classification πρόβλημα.

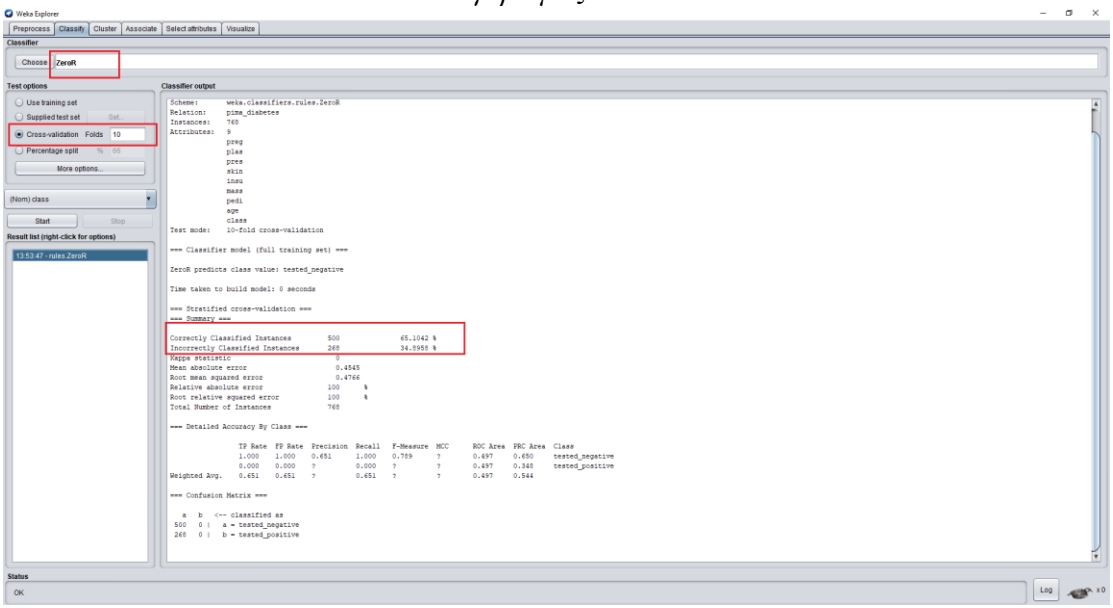
---

13 https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/

14 https://machinelearningmastery.com/estimate-baseline-performance-machine-learning-models-weka/

ΕΙΚΟΝΑ14

Αλγόριθμος ZeroR



Με την χρήση του ZeroR και την μέθοδο cross validation που το χωρίζει σε 10 κομμάτια, παρατηρούμε στην ΕΙΚΟΝΑ14 ότι ο αλγόριθμος προβλέπει την τιμή της class tested negative για όλα τα instances αφού αυτή αποτελεί την πλειοψηφία και πετυχαίνει ακρίβεια 65.1%. Σε οποιοδήποτε πείραμα κάνουμε, είναι πολύ σημαντικό κάθε φορά να χρησιμοποιούμε στην αρχή αυτόν τον αλγόριθμο για να έχουμε ένα κάτω όριο. Δεν μπορούμε να γνωρίζουμε από την αρχή ποιος αλγόριθμος θα είναι ο πιο αποτελεσματικός γι αυτό και δημιουργούμε το κατώφλι. Στη συνέχεια προσπαθούμε να βρούμε με διάφορες δοκιμές των αλγορίθμων του WEKA, τον καλύτερο αλγόριθμο, δηλαδή αυτόν που θα έχει την υψηλότερη ακρίβεια. Τέλος μπορεί μερικές φορές από έναν άλλον αλγόριθμο να δούμε 'χειρότερα' αποτελέσματα στο πείραμα μας απ' ότι αυτά του ZeroR. Σε αυτή την περίπτωση πρέπει να συνεχίσουμε να ψάχνουμε για αλγόριθμο με μεγαλύτερη ακρίβεια.

Αλγόριθμος IBk στο WEKA της κατηγορίας nearest neighbor

Οι αλγόριθμοι της κατηγορίας nearest neighbor[15] αποτελούν μία χρήσιμη τεχνική, που μας επιτρέπει να χρησιμοποιήσουμε παλαιότερα δεδομένα με γνωστές τιμές εξόδου για να προβλέψουμε την τιμές εξόδου νέων δεδομένων. Παρόλο που αυτή η τεχνική φαίνεται να μοιάζει με τις προηγούμενες (classification, regression), διαφέρει από αυτές, αφού δεν παράγει κάποιο δέντρο όπως η πρώτη αλλά ούτε και χρησιμοποιείται μόνο για αριθμητικές τιμές εξόδου όπως η δεύτερη. Η παραγωγή ενός δέντρου απόφασης είναι πολλές φορές σημαντικό πρόβλημα για τεράστια σύνολα δεδομένων όπως για παράδειγμα της εταιρείας Amazon, γιατί αν έπρεπε να σχηματιστεί ένα δέντρο που αφορά τις επιλογές των πελατών της για προϊόντα, τότε θα ήταν τεράστιο και η ακρίβεια του αρκετά χαμηλή. Για τον λόγο αυτό, οι αλγόριθμοι nearest neighbor επιλύουν τέτοιου είδους ζητήματα χρησιμοποιώντας την 'κανονικοποιημένη Ευκλείδεια απόσταση'.

[15] https://www.ibm.com/developerworks/library/os-weka3/index.html

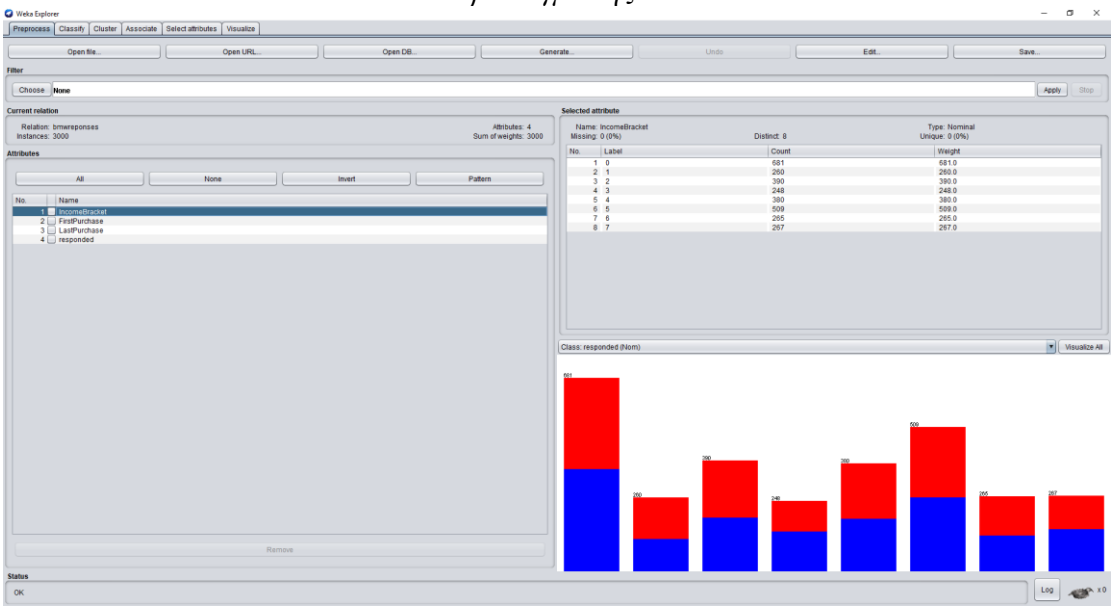
d(X,Y) = ||X-Y|| = sqrt((X1-Y1)^2 + (X2-Y2)^2 + … + (Xn - Yn)^2)

Αν υποθέσουμε ότι έχουμε μία λίστα πελατών της Amazon και θέλουμε να προβλέψουμε τι θα αγοράσει ένας νέος πελάτης, για τον οποίον έχουμε μερικές πληροφορίες, τότε ο αλγόριθμος IBk στο WEKA υπολογίζοντας την Ευκλείδεια απόσταση και εντοπίζοντας ποιος είναι ο κοντινότερος γείτονας-πελάτης στον συγκεκριμένο πελάτη, θα μπορεί να προβλέψει την συμπεριφορά του και τελικά τι είναι αυτό που πιθανόν να αγοράσει. Οι δυνατότητες αυτής της κατηγορίας αλγορίθμων δεν τελειώνουν σε αυτό το σημείο, αφού μπορούμε να υπολογίσουμε και την απόσταση μεταξύ του πελάτη με περισσότερους και έτσι να προβλέψουμε τα 2 ή 3 ή Κ προϊόντα που πιθανόν να αγοράσει ο πελάτης. Πέρα από τα προϊόντα, ο αλγόριθμος θα μπορούσε να χρησιμοποιηθεί και για απαντήσεις του τύπου 'YES, NO' όπου για παράδειγμα αν έχουμε περισσότερες απαντήσεις 'YES' και λιγότερες 'NO' τότε μπορεί να υπολογιστεί και πάλι η Ευκλείδεια απόσταση. Έτσι γίνεται δυνατό να βρούμε αν ο γείτονας βρίσκεται κοντά σε περισσότερες τιμές 'YES' και τότε θα σήμαινει ότι και η πρόβλεψη για αυτόν θα έβγαινε ως 'YES'. Το ερώτημα είναι πόσους γείτονες θα πρέπει να χρησιμοποιούμε στο μοντέλο κάθε φορά. Αυτό εξαρτάται από τα δεδομένα και πάνω σε αυτά κάνουμε διάφορες δοκιμές μέχρι να έχουμε το βέλτιστο αποτέλεσμα.

Ολοκληρώνοντας το μαθηματικό υπόβαθρο που χρησιμοποιούν οι αλγόριθμοι της κατηγορίας αυτής, θα περάσουμε σε ένα παράδειγμα του αλγορίθμου IBk που υπάρχει μέσα στο λογισμικό WEKA. Το σετ που θα χρειαστούμε αφορά την αντιπροσωπεία της BMW και την διαφημιστική καμπάνια για να πουλήσει μια διετή επέκταση εγγύησης στους προηγούμενους πελάτες. Το αρχείο ονομάζεται bmw-training.arff και βρίσκεται στον σύνδεσμο:
https://www.ibm.com/developerworks/apps/download/index.jsp?contentid=494038&filename=os-weka3-Example.zip&method=http&locale=
Αρχικά ανοίγουμε το αρχείο και παρατηρούμε στην EIKONA15 ότι έχουμε συνολικά 4 attributes και 3000 instances. Τα attributes αναφέρονται σε εισόδημα, τον χρόνο/μήνα που αγοράστηκε το πρώτο BMW, τον χρόνο/μήνα που αγοράστηκε το πιο πρόσφατο BMW και αν ανταποκρίθηκαν στην επέκταση εγγύησης αντίστοιχα.

ΕΙΚΟΝΑ15

Το παράδειγμα της BMW

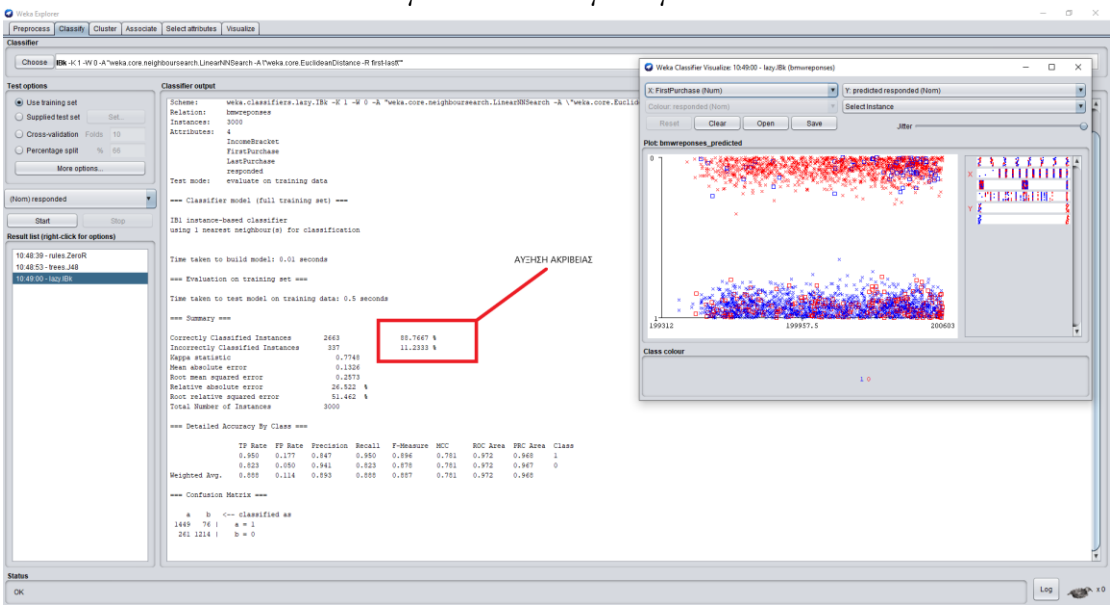


Πηγαίνοντας στην επιλογή Classify και στην συνέχεια Choose Classifier, επιλέγουμε για αρχή τον ZeroR, χρησιμοποιώντας ως επιλογή το 'Use training set' για να μας δώσει το κάτω όριο, το οποίο είναι 50.8%, σχεδόν 51% ακρίβεια. Αν τώρα δοκιμάσουμε να εκτελέσουμε έναν αλγόριθμο της κατηγορίας των δέντρων (J48) που είχαμε αναλύσει, θα παρατηρήσουμε ότι το αποτέλεσμα εξάγει χαμηλή ακρίβεια, μόνο 59.1%, δηλαδή λίγο καλύτερα από τον ZeroR. Επομένως, καταλαβαίνουμε ότι δεν είναι κατάλληλος αυτός ο αλγόριθμος για το συγκεκριμένο σύνολο δεδομένων. Επιπλέον αν επιλέξουμε την οπτικοποίηση των σφαλμάτων με δεξί κλικ στην εκτέλεση του J48 και visualize classifier errors, αυξάνοντας το Jitter θα δούμε ότι έχουν γίνει πολλές λάθος ταξινομήσεις των στοιχείων. Άρα σε αυτή την περίπτωση, θέλουμε έναν αλγόριθμο της κατηγορίας lazy, που είναι ο IBk και ανήκει στους αλγορίθμους nearest neighbor ώστε να υπολογίσει την Ευκλείδεια απόσταση.

Εκτελώντας τον αλγόριθμο IBk, παρατηρούμε ότι η ακρίβεια έχει αυξηθεί πάρα πολύ ΕΙΚΟΝΑ16 και βρίσκεται στο 88.7%, δηλαδή σχεδόν στο 90% που είναι ένα καλό ποσοστό. Τα αποτελέσματα του μοντέλου μας δείχνουν ότι έχουμε 76 false positives(2.5%) στον confusion matrix και ότι έχουμε 261 false negatives (8.7%). Το μοντέλο μας εδώ, έχει προβλέψει ότι ο πελάτης θα αγοράσει μία επέκταση εγγύησης ενώ στην πραγματικότητα δεν το έκανε (2.5%), και για το ποσοστό (8.7%) το μοντέλο προβλέπει ότι ο πελάτης δεν θα αγόραζε μία επέκταση εγγύησης αλλά στην πραγματικότητα το έκανε. Αν υποθέσουμε ότι το κάθε φυλλάδιο κοστίζει 3 ευρώ και ότι το συνολικό κέρδος που θα λάβει ο αντιπρόσωπος της BMW είναι 400 ευρώ τότε από την οπτική cost/benefit μπορούμε να κάνουμε τον εξής υπολογισμό: 400 - (2.5% * 3) - (8.7% * 400) = 365 ευρώ, το οποίο είναι αρκετά κερδοφόρο για την αντιπροσωπία. Αν αυτόν τον υπολογισμό το κάνουμε ξανά με τον αλγόριθμο που μας δίνει 59% ακρίβεια θα δούμε ότι το κέρδος μειώνεται γιατί τα ποσοστά των FP και FN είναι πιο υψηλά. Τέλος στο WEKA μπορούμε κάνοντας κλικ στο όνομα του αλγορίθμου να αλλάξουμε το 'KNN' που αναφέρεται στους γείτονες και να πάρουμε διαφορετικά αποτελέσματα.

ΕΙΚΟΝΑ 16

Βελτιωμένα αποτελέσματα με τον IBk



7 Σύνοψη λογισμικού WEKA

Συνοψίζοντας όλα τα παραπάνω, το λογισμικό WEKA είναι ένα εργαλείο που μας βοηθάει να κάνουμε μελλοντικές προβλέψεις. Αυτές οι προβλέψεις απευθύνονται σε μεγάλη κλίμακα από παραδείγματα όπως, η αναγνώριση του είδους ενός λουλουδιού έχοντας σαν δεδομένα τα χαρακτηριστικά πολλών λουλουδιών, αν ο καιρός είναι κατάλληλος για να παίξει έξω ένα παιδί, αν ένας άνθρωπος ανάλογα με τα συμπτώματα πάσχει από κάποια ασθένεια, ποιο ή ποια προϊόντα είναι πιθανό να αγοράσει ένας νέος πελάτης του ηλεκτρονικού καταστήματος, αν ένα email κατατάσσεται ως spam κ.α. Όλα αυτά τα προβλήματα επιλύονται μέσω των μαθηματικών υπολογισμών που εκτελούν οι αλγόριθμοι του WEKA. Αρχικά ο χρήστης θα πρέπει να μάθει καλά την δομή των αρχείων ARFF, ώστε να καταλαβαίνει πως διακρίνονται τα δεδομένα, τα χαρακτηριστικά και η class. Επόμενο βήμα είναι να μπορέσει να εισάγει ένα τέτοιο αρχείο στο λογισμικό και να οπτικοποιήσει τα δεδομένα. Αξίζει να σημειωθεί ότι, κάθε σύνολο χαρακτηριστικών μπορεί να αφορά είτε ένα πρόβλημα ταξινόμησης είτε ένα πρόβλημα παλινδρόμησης και για τον λόγο αυτό θα πρέπει να είμαστε προσεκτικοί και να ξέρουμε τι δεδομένα θα χειριστούμε. Είναι καλή πρακτική, να χρησιμοποιούμε στα πειράματα μας τον αλγόριθμο ZeroR για να μας δίνει το κάτω όριο είτε σε ποσοστό ακρίβειας είτε σε αριθμητική τιμή και έπειτα να κάνουμε δοκιμές με άλλους αλγορίθμους, διαφορετικά δεν θα μπορούμε να μάθουμε αν είναι καλύτεροι. Είδαμε επίσης ότι, εκτελώντας έναν αλγόριθμο όπως είναι ο J48 σε ένα πρόβλημα classification, έχουμε αρκετά καλά αποτελέσματα, μπορούμε ακόμα να δούμε πως αναπαρίστανται το δέντρο και εξηγήσαμε την λογική πίσω από αυτό. Σε κάποια άλλα προβλήματα όμως με πολύ περισσότερα δεδομένα τα δέντρα αποφάσεων δεν είναι η πιο σωστή επιλογή, γι αυτό και το WEKA είναι εξοπλισμένο με αλγορίθμους της κατηγορίας nearest neighbor που λειτουργούν με την Ευκλείδεια απόσταση. Τέλος, υπάρχουν και άλλες επιπρόσθετες λειτουργίες και αλγόριθμοι του WEKA που οδηγούν σε πολλές λεπτομέρειες και ξεπερνούν το βασικότερο επίπεδο. Με εφόδιο όλες τις παραπάνω γνώσεις, βρισκόμαστε σε ένα καλό επίπεδο όπου μπορούμε να επιλύσουμε αρκετά προβλήματα πρόβλεψης.

8 Λογισμικό εξόρυξης δεδομένων rapidminer

Το rapidminer[16] είναι μία ηλεκτρονική πλατφόρμα, ιδανική για ομάδες αναλυτών, η οποία προσφέρει τον συνδυασμό και την συνένωση της προετοιμασίας δεδομένων, της μηχανικής μάθησης και της ανάπτυξης μοντέλων προβλέψεων. Πιο συγκεκριμένα, το λογισμικό rapidminer είναι εφοδιασμένο με πάρα πολλές συναρτήσεις, βιβλιοθήκες και αλγορίθμους της μηχανικής μάθησης, καθιστώντας δυνατή την προετοιμασία και την επεξεργασία των συνόλων δεδομένων που τοποθετεί ο αναλυτής. Μέσα σε αυτό βρίσκονται πολλά διαθέσιμα παραδείγματα με έτοιμα δεδομένα, τα οποία μπορεί να χρησιμοποιήσει ο χρήστης για να εξοικειωθεί με το περιβάλλον και τις διάφορες λειτουργίες. Εκτός από τον κλασικό τρόπο, όπου ο αναλυτής θα πρέπει να κάνει το μεγαλύτερο μέρος της επεξεργασίας των δεδομένων, σε αυτή την πλατφόρμα υπάρχει λειτουργία που ονομάζεται AutoModel και χρησιμοποιεί αυτοματοποιημένη μηχανική μάθηση. Μέσω της χρήσης υπερ παραμέτρων και αυτόματης λειτουργίας παράγονται τα βέλτιστα μοντέλα προβλέψεων χωρίς ο χρήστης να αφιερώσει πολύ χρόνο για την εκτέλεση τέτοιων διαδικασιών. Είναι σημαντικό να αναφερθεί πως το rapidminer στην επίσημη ιστοσελίδα αλλά και μέσα στο ίδιο το λογισμικό παρέχει βήματα για να μάθει να το χρησιμοποιεί ακόμα και ένας εντελώς αρχάριος χρήστης. Στην ιστοσελίδα υπάρχουν πολλά βίντεο που παρουσιάζουν και εξηγούν τις λειτουργίες του περιβάλλοντος ενώ μέσα στο λογισμικό μετά την εγκατάσταση μπορεί κανείς να εργαστεί περισσότερο. Αυτό σημαίνει ότι, περιλαμβάνονται βήματα για την χρήση του λογισμικού και επιπλέον δίνονται ασκήσεις προαιρετικά που βοηθούν στην ανάπτυξη καλύτερων δεξιοτήτων. Τέλος, το rapidminer αποτελεί ένα ισχυρό εργαλείο εξόρυξης πληροφορίας η οποία μπορεί να πραγματοποιηθεί με πάρα πολλούς τρόπους μέσα από αυτό και να αναπαρασταθεί με διαγράμματα για περισσότερη ανάλυση και διάκριση. Είναι καλό πριν επιλέξουμε να κατεβάσουμε το rapidminer, να αφιερώσουμε λίγο χρόνο εξερευνώντας την σελίδα γιατί έχει πολλές χρήσιμες πληροφορίες, απαντήσεις σε ερωτήματα και μπορούμε να δούμε τα πρόσωπα των ατόμων που συνέβαλαν στην δημιουργία του λογισμικού και μας παρέχουν πολύτιμη βοήθεια.

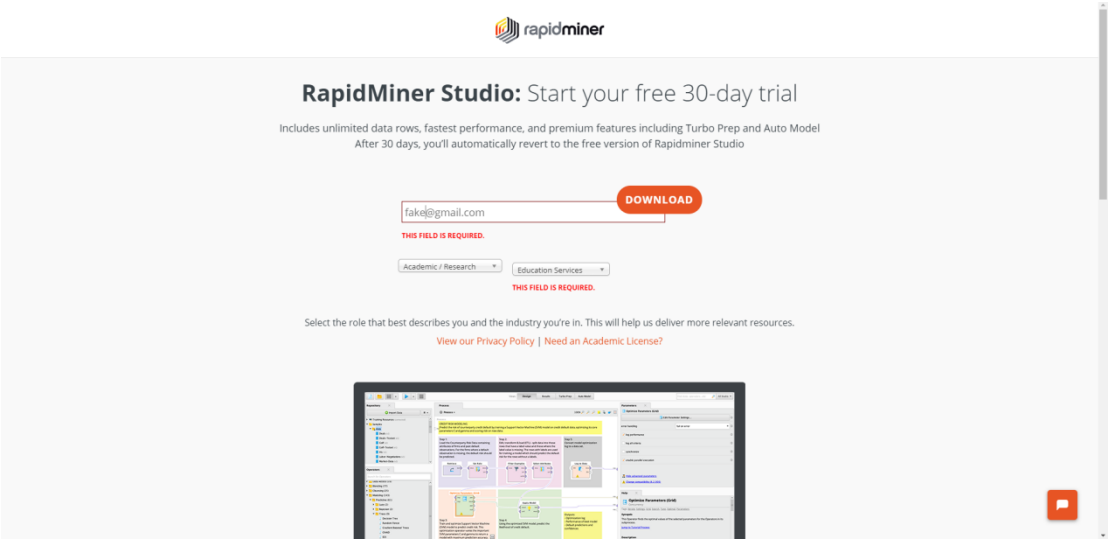
[16] https://rapidminer.com/

9 Διαδικασία λήψης και εγκατάστασης του λογισμικού rapidminer

Για την εγκατάσταση[17] θα χρειαστούν μερικά απλά βήματα στην επίσημη ιστοσελίδα του λογισμικού rapidminer. Αρχικά, ο χρήστης θα πρέπει να πατήσει στο κουμπί get started, ώστε να μεταφερθεί στην επόμενη σελίδα. Εκεί θα παρατηρήσει ότι το rapidminer είναι διαθέσιμο δωρεάν για 30 ημέρες, όμως αυτό αφορά το πλήρες περιεχόμενο του λογισμικού. Στο πλήρες πακέτο συμπεριλαμβάνονται πολλά παραδείγματα με σύνολα δεδομένων, επιπρόσθετα χαρακτηριστικά του εργαλείου όπως το AutoModel και το TurboPrep και εκτελούνται πιο γρήγορα οι διαδικασίες. Έπειτα από αυτό το χρονικό διάστημα, ο χρήστης επιστρέφει στην απλή μορφή του εργαλείου που παρόλα αυτά είναι αρκετά ικανοποιητική. Ωστόσο, υπάρχει η δυνατότητα, αν κάποιος είναι είτε φοιτητής είτε καθηγητής, να ζητήσει άδεια από την εταιρεία, συμπληρώνοντας τα απαιτούμενα στοιχεία που αποδεικνύουν τη θέση του και θα μπορέσει, αφού γίνει αποδεκτός, να χρησιμοποιήσει το πλήρες πακέτο του λογισμικού για 1 χρόνο. Για την εγκατάσταση θα ζητηθεί ένας λογαριασμός email και η τοποθέτηση στοιχείων που περιγράφουν καλύτερα τον χρήστη όπως για παράδειγμα η επιλογή business analyst. Ο χρήστης θα λάβει ένα email επιβεβαίωσης στον λογαριασμό του και στη συνέχεια θα μπορέσει να ξεκινήσει την λήψη του λογισμικού. Με την ολοκλήρωση αυτού του βήματος το rapidminer βρίσκεται πλέον στον υπολογιστή όπου και θα γίνει η εγκατάσταση. Η εγκατάσταση δεν απαιτεί πολλά βήματα παρά μόνο να κάνουμε διπλό κλικ στο rapidminer-studio-9.2.1-win64-install το οποίο θα βρίσκεται στον φάκελο με τις λήψεις αρχείων. Έπειτα θα πρέπει να περιμένουμε για ένα χρονικό διάστημα να φορτώσει επειδή το λογισμικό έχει πολύ μεγάλο περιεχόμενο και αργεί να ανοίξει. Την πρώτη φορά εισάγουμε τα στοιχεία που έχουμε δώσει στην σελίδα του rapidminer (email, password) για να συνδεθούμε και μπορούμε να επιλέξουμε να παραμείνουμε συνδεδεμένοι. Στην ΕΙΚΟΝΑ18 φαίνεται το σημείο λήψης όπου αν μεταβούμε πιο κάτω μπορούμε να βρούμε και την άδεια για φοιτητές και καθηγητές.

ΕΙΚΟΝΑ18

Λήψη του rapidminer



[17] https://rapidminer.com/get-started/

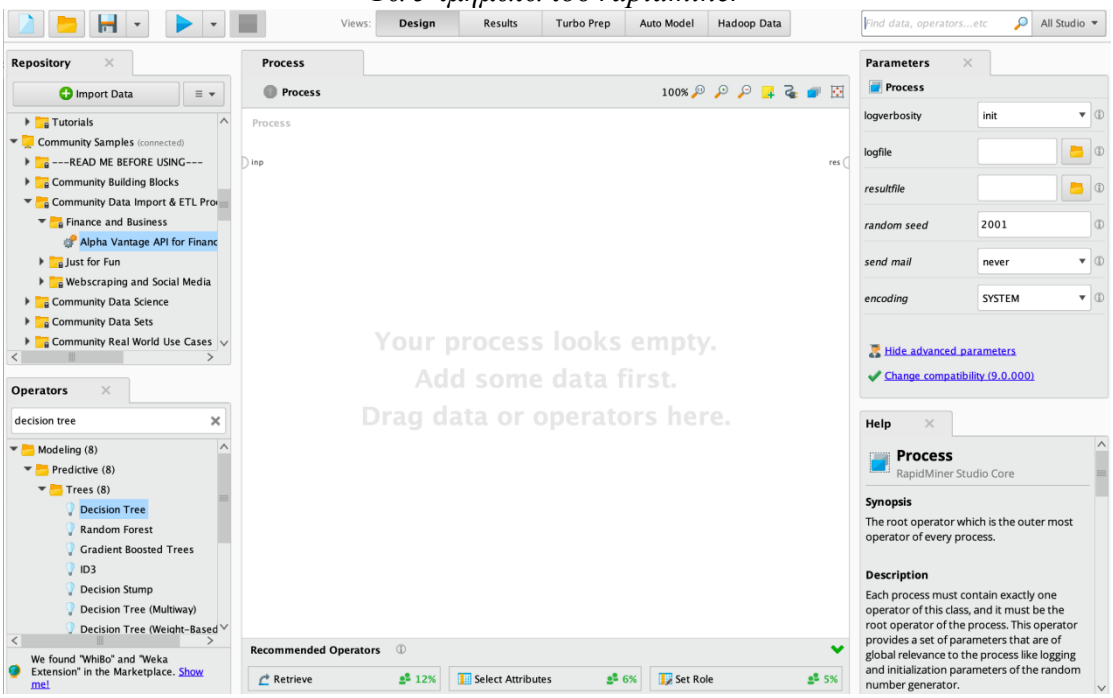
10 Περιβάλλον εργασίας του rapidminer

10.1 Γραφικό περιβάλλον και τα 5 βασικά τμήματα

Το rapidminer αποτελείται από πέντε βασικά τμήματα, τα οποία είναι Repository, Process, Operators, Parameters και Help. Το Repository είναι το πρώτο τμήμα στην αριστερή πλευρά και είναι υπεύθυνο για την εισαγωγή συνόλων δεδομένων σε μορφή αρχείων XLS και CSV. Στο Repository μπορεί να βρει και να χρησιμοποιήσει ο αναλυτής έτοιμα σετ που παρέχει το rapidminer με την εγκατάσταση, τα οποία είναι μέσα στον φάκελο Samples. Επιπλέον, στο Repository βρίσκεται το Local Repository, όπου αποθηκεύονται οι εργασίες που κάνει κάποιος τοπικά, ενώ το Cloud Repository αφορά εργασίες που κάνουμε στο Cloud όμως αυτή η δυνατότητα θα αφαιρεθεί από το λογισμικό στο μέλλον. Το Repository αποθηκεύει ακόμα και εργασίες που είναι ομαδικές, εκτελούνται από την κοινότητα του rapidminer και αφορούν συνήθως μεγαλύτερα έργα. Το δεύτερο τμήμα στην αριστερή πλευρά είναι το Operators, μέσα στο οποίο υπάρχουν αλγόριθμοι μηχανικής μάθησης, μαθηματικές συναρτήσεις, αλγόριθμοι ταξινόμησης, συνδέσεις συνόλων δεδομένων, πράξεις και πολλά άλλα χρήσιμα χαρακτηριστικά με τα οποία μπορούμε να επεξεργαστούμε τα δεδομένα. Είναι εύκολο να βρούμε κάποιον αλγόριθμο η συνάρτηση, πληκτρολογώντας το όνομα του στην αναζήτηση κάτω από την λέξη Operators. Στο κέντρο του περιβάλλοντος του λογισμικού βρίσκεται το τμήμα Process ή αλλιώς ονομάζεται root operator και είναι ο χώρος μέσα στον οποίο εργαζόμαστε. Εκεί μπορούμε να τοποθετήσουμε το σετ δεδομένων, όπως θα δείξουμε στη συνέχεια, να επιλέξουμε αλγόριθμο, φίλτρο, ακόμα και δύο σετ δεδομένων να τα συνδέσουμε μεταξύ τους και να ‘τρέξουμε’ στη συνέχεια τη διαδικασία για να δούμε τα αποτελέσματα. Επιπρόσθετα πάνω από το τμήμα Process υπάρχουν τα Views: Design, Results, TurboPrep, AutoModel τα οποία είναι πολύ σημαντικά για την ανάλυση θα εξηγηθούν παρακάτω. Στο δεξί μέρος βρίσκεται το τέταρτο τμήμα Parameters, το οποίο χρησιμοποιείται τη στιγμή που εισάγουμε έναν Operator στην επιφάνεια του τμήματος Process. Με το τμήμα Parameters μπορούμε να κάνουμε συγκεκριμένες αλλαγές στα δεδομένα μας αλλάζοντας κάποια χαρακτηριστικά. Για παράδειγμα μπορούμε σε ένα σύνολο δεδομένων με στοιχεία πελατών και σε ένα άλλο με στοιχεία προϊόντων να τροποποιήσουμε τον αλγόριθμο ταξινόμησης που βρίσκεται στους Operators, θέτοντας κατάλληλες Parameters και να βρούμε ποιοι πελάτες αγόρασαν τα περισσότερα προϊόντα. Τέτοιου είδους αλλαγές καλύπτει το συγκεκριμένο τμήμα. Το πέμπτο και τελευταίο βασικό τμήμα του GUI είναι το Help που δεν είναι τίποτα άλλο από το να μας δίνει πληροφορίες για ότι χρησιμοποιούμε στο rapidminer. Όλα αυτά παρουσιάζονται στην ΕΙΚΟΝΑ19 που βρίσκεται στην επόμενη σελίδα.

ΕΙΚΟΝΑ19

Τα 5 τμήματα του rapidminer



10.2 Βασικές τεχνικές της επιστήμης των δεδομένων με το rapidminer

Σε αυτό το tutorial[18] θα χρησιμοποιήσουμε το rapidminer για βασικές τεχνικές του data science. Τέτοιες θα είναι η πρόσβαση στα σετ δεδομένων, η μετατροπή τους και η κατασκευή στατιστικών μοντέλων. Αρχικά, θα ανοίξουμε το σετ που περιέχει στοιχεία για τους επιβάτες του Τιτανικού, το οποίο βρίσκεται στο τμήμα Repository, στον φάκελο Samples, έπειτα στον φάκελο data και έχει το όνομα Titanic training. Παίρνουμε το σετ κρατώντας το αριστερό κλικ πατημένο και το μεταφέρουμε στην επιφάνεια του τμήματος Process, όπου συνδέουμε την έξοδο του με όνομα 'out' με την πύλη 'res' που βρίσκεται στην δεξιά πλευρά. Αφού ολοκληρώσουμε την σύνδεση κάνουμε κλικ στο γαλάζιο κουμπί 'Run button' ώστε να δούμε τα αποτελέσματα. Παρατηρούμε ότι στα Views, ήμασταν στο Design ενώ τώρα έχουμε μεταφερθεί στο Results. Εδώ φαίνονται τα δεδομένα του data set ενώ αν κάνουμε κλικ στο κουμπί Statistics παίρνουμε σημαντικές πληροφορίες όπως για παράδειγμα το πόσοι επιβάτες είχαν επιζήσει.

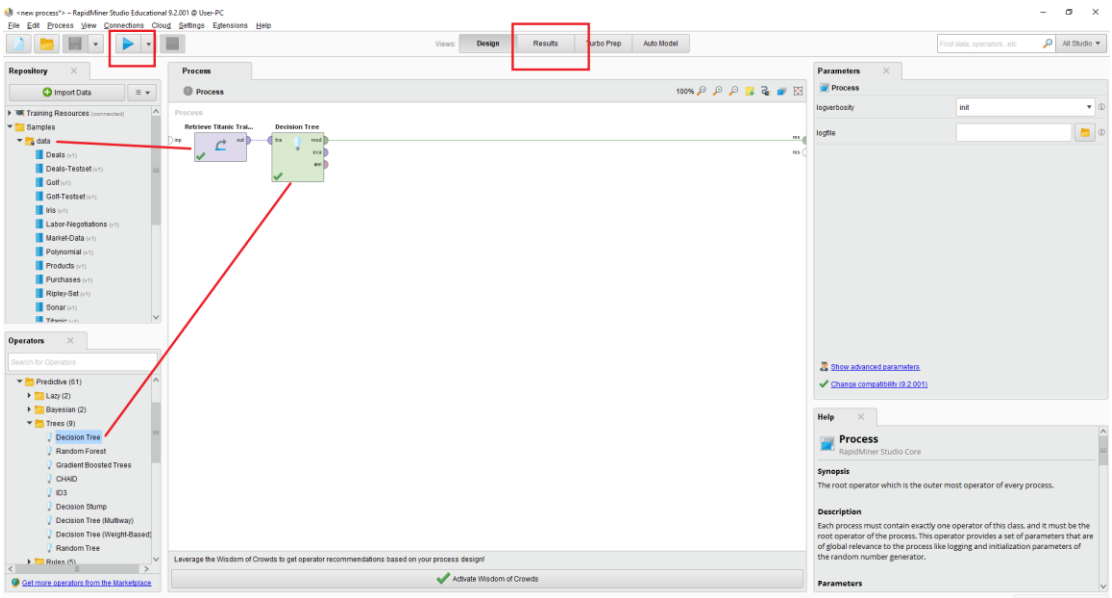
Για να εντοπίσουμε κρυμμένα μονοπάτια στο σετ δεδομένων, θα επιλέξουμε έναν αλγόριθμο της ομάδας δέντρων αποφάσεων (decision trees)). Στο τμήμα Operators ανοίγουμε τον φάκελο Modeling και στη συνέχεια τον φάκελο Predictive, μετά Trees και παίρνουμε τον Operator Decision Tree με ίδιο τρόπο όπως και το dataset και τον αφήνουμε στην επιφάνεια του Process. Δεν πρέπει να ξεχάσουμε να ενώσουμε το σετ μας με τον αλγόριθμο και μετά να ενώσουμε την πύλη εξόδου 'mod' του αλγορίθμου με την πύλη 'res'. Έχοντας κάνει τις σωστές συνδέσεις μπορούμε να 'τρέξουμε' και να δούμε το παραγόμενο αποτέλεσμα. Εδώ εμφανίζεται ένα δέντρο που μας δίνει την

[18] https://academy.rapidminer.com/learning-paths/get-started-with-rapidminer-and-machine-learning

πληροφορία ότι οι γυναίκες με μικρές σχετικά οικογένειες και ακριβό εισιτήριο είχαν μεγάλη πιθανότητα να επιζήσουν. Όλα αυτά φαίνονται στις εικόνες 20 και 21 αντίστοιχα.

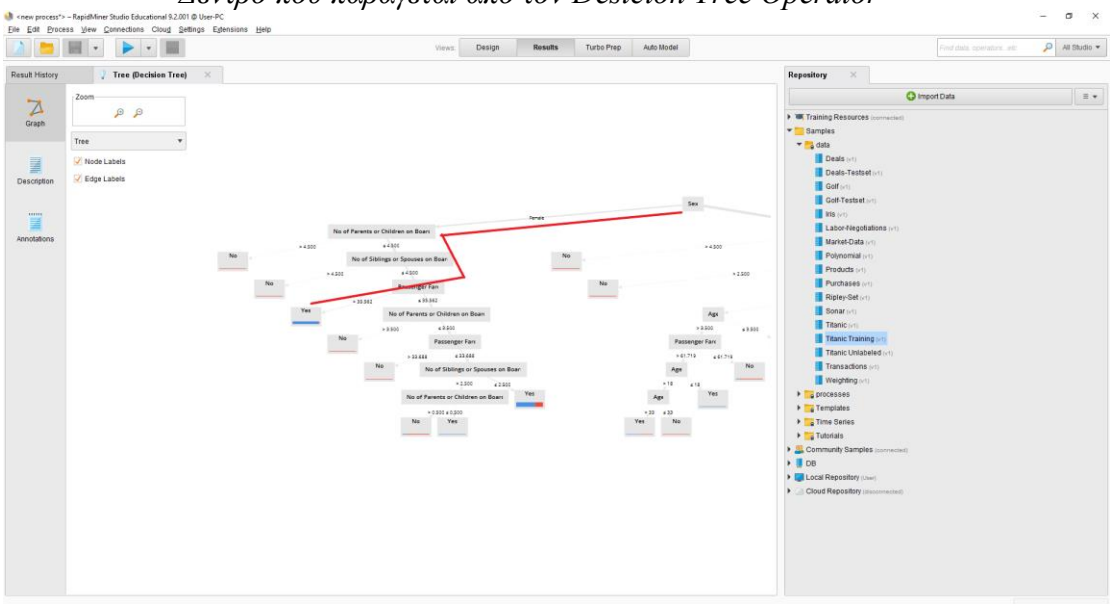
ΕΙΚΟΝΑ20

Σετ δεδομένων και αλγόριθμος Desicion Tree στο rapidminer



ΕΙΚΟΝΑ21

Δέντρο που παράγεται από τον Desicion Tree Operator



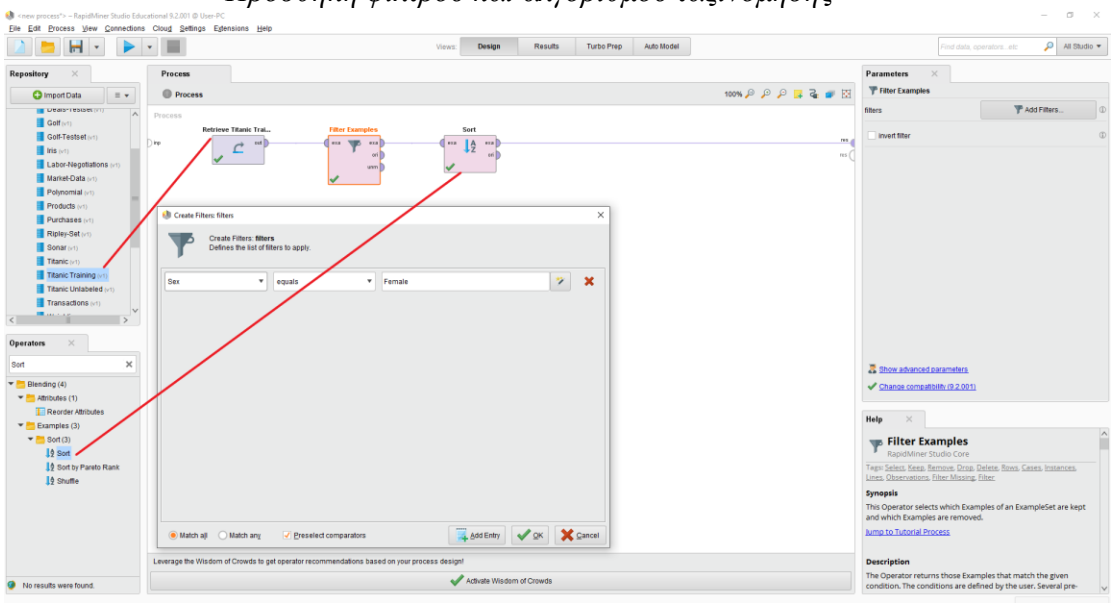
Με κόκκινη γραμμή αποκαλύπτεται το μονοπάτι που μας δείχνει ότι οι γυναίκες μικρών οικογενειών (Number of parents, siblings κ.α.) και με ακριβό εισιτήριο (Passenger Fare) έχουν πολύ αυξημένο το “YES” δηλαδή την πιθανότητα επιβίωσης που φαίνεται με γαλάζιο χρώμα στην ΕΙΚΟΝΑ21. Σε περίπτωση που θέλουμε να εισάγουμε ένα σετ δεδομένων πηγαίνουμε στο κουμπί ‘import data’ στο Repository και από εκεί ακολουθούμε τα βήματα My Computer, select the data location, όπου επιλέγουμε το μονοπάτι με το αρχείο XLS ή CSV που θα βάλουμε μέσα στο

rapidminer. Στο διαδίκτυο μπορούμε να βρούμε μεγάλα σετ με δεδομένα να τα περάσουμε σε μορφή αρχείου XLS και να τα επεξεργαστούμε στο rapidminer.

Ας εφαρμόσουμε τώρα ένα φίλτρο στο σετ δεδομένων με όνομα Titanic αυτή τη φορά, για να ξεχωρίσουμε τις γυναίκες επιβάτες του πλοίου. Ο λόγος που το κάνουμε αυτό είναι για να εντοπίσουμε τα πιο ακριβά εισιτήρια που αγοράστηκαν από τις γυναίκες αφού προηγουμένως είχαμε δει πως η πιθανότητα να σωθούν ήταν πολύ υψηλή. Επιλέγουμε τον φάκελο Samples στο Repository, μετά το dataset Titanic και το τοποθετούμε στον χώρο του Process ΕΙΚΟΝΑ22. Αυτή τη φορά θα χρησιμοποιήσουμε ένα φίλτρο το οποίο θα ξεχωρίσει τις γυναίκες και θα αφαιρέσει τους άντρες από το σετ μας. Για να το κάνουμε αυτό, πληκτρολογούμε στο πλαίσιο αναζήτησης του τμήματος Operators 'Filter Examples', παίρνουμε το φίλτρο και το ενώνουμε με τα δεδομένα τραβόντας μία γραμμή από την πύλη εξόδου των δεδομένων προς την πύλη εισόδου του φίλτρου. Παρατηρούμε ότι στο δεξί μέρος του rapidminer μπορούμε στο τμήμα Parameters να πατήσουμε το κουμπί Add Filters. Αυτό μας επιτρέπει να κρατήσουμε μόνο τις γυναίκες γράφοντας στα κενά πλαίσια 'Sex equals Female'. Σε περίπτωση που δεν έχουμε ενώσει τα δεδομένα με το φίλτρο τότε δεν μπορούμε να αλλάξουμε το σετ, αφού το φίλτρο δεν γνωρίζει εκ των προτέρων σε ποιο σετ αναφέρεται. Για να ολοκληρώσουμε την διαδικασία θα προσθέσουμε έναν αλγόριθμο ταξινόμησης που ονομάζεται 'Sort' και τοποθετείται και αυτός μέσα στο τμήμα Operators. Επομένως, βρίσκουμε τον αλγόριθμο και τον ενώνουμε με το φίλτρο με την ίδια λογική όπως κάναμε και με τα δεδομένα. Κάνοντας κλικ πάνω στον 'Sort' μπορούμε να αλλάξουμε την κατεύθυνση της ταξινόμησης σε φθίνουσα σειρά 'descending' και είναι απαραίτητο να θέσουμε την στήλη στην οποία αναφερόμαστε, όπου εδώ στο attribute name γράφουμε 'Passenger Fare'. Τέλος, τρέχουμε το πείραμα και μπορούμε να δούμε στο Results View τις γυναίκες με τα πιο ακριβά εισιτήρια. Μια τελευταία σημαντική πληροφορία που λαμβάνουμε από αυτό είναι πως στη στήλη 'Survived' υπάρχουν μόνο τιμές 'Yes' στα εισιτήρια με τη μεγαλύτερη τιμή.

ΕΙΚΟΝΑ22

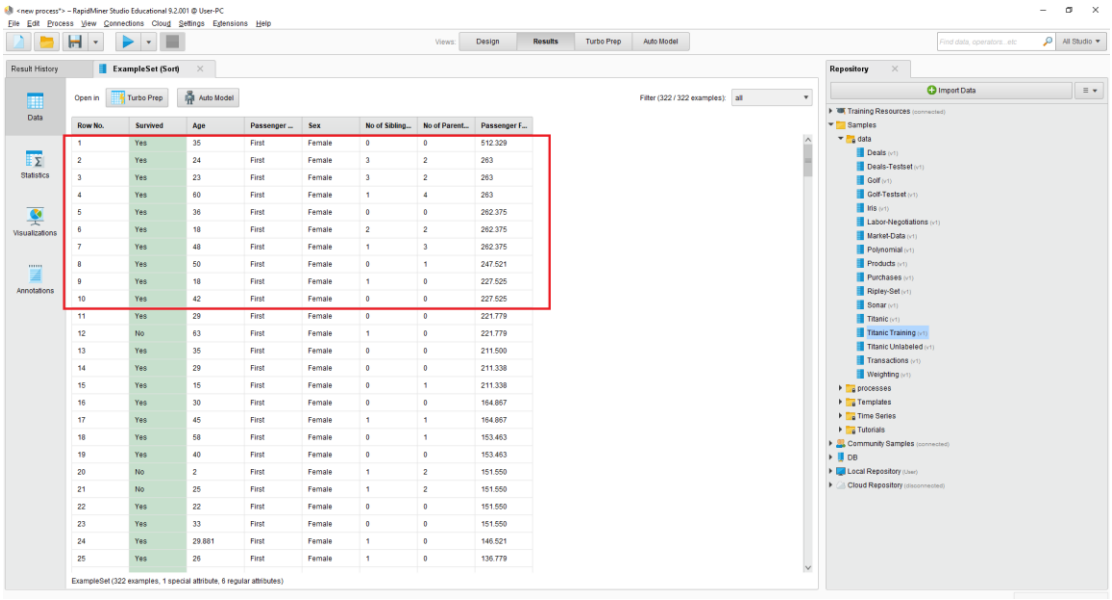
Προσθήκη φίλτρου και αλγόριθμου ταξινόμησης



Στην εικόνα παρατηρούμε πως μπορούμε να προσθέσουμε το κατάλληλο φίλτρο για να έχουμε μέσα στα δεδομένα μόνο τις γυναίκες που θέλουμε να εξετάσουμε

ΕΙΚΟΝΑ23

Ταξινομημένη αναπαράσταση των στοιχείων του σετ μας

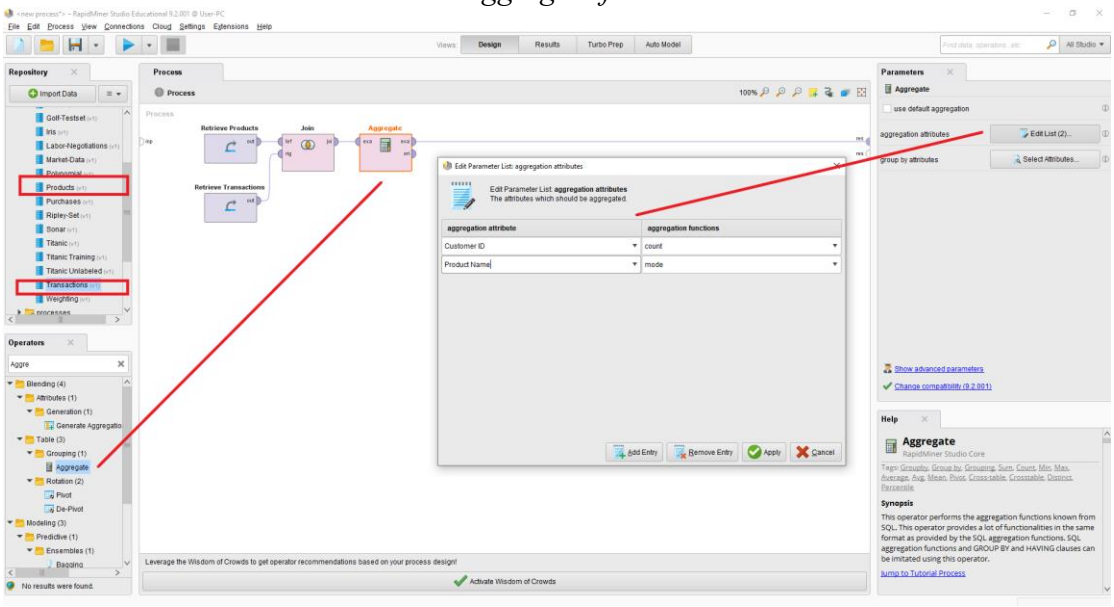


Στην εικόνα βλέπουμε ότι οι πρώτες 10 γυναίκες (στήλη 'Sex') που είχαν αγοράσει τα πιο ακριβά εισιτήρια (στήλη 'Passenger Fare') είχαν καταφέρει να σωθούν όλες, το οποίο φαίνεται στη στήλη 'Survived' με όλες τις τιμές να είναι 'Yes'.

Στο επόμενο tutorial θα δείξουμε τον τρόπο με τον οποίο μπορούμε να συνδυάσουμε δύο σύνολα δεδομένων. Το πρώτο data set αναφέρεται σε προϊόντα που έχουν πουληθεί από κάποιον οργανισμό και το δεύτερο περιέχει πληροφορίες για τους πελάτες και τα προϊόντα που έχουν αγοράσει. Αρχικά, θα επιλέξουμε από το τμήμα Repository τον φάκελο Samples -> data και τοποθετούμε τα σετ Products και Transactions στην επιφάνεια του Process, το ένα σετ κάτω από το άλλο. Δεύτερο βήμα είναι να αναζητήσουμε τον κατάλληλο Operator που θα μας επιτρέψει να συνδέσουμε τα δεδομένα μέσω μιας κοινής στήλης. Ο Operator σε αυτή την περίπτωση είναι ο Join, τον οποίο θα συνδέσουμε με τις πύλες εξόδου των σετ δεδομένων. Στη συνέχεια θα χρειαστεί να πατήσουμε στο κουμπί Edit List, που φαίνεται στο τμήμα Parameters και στο παράθυρο που ανοίγει να θέσουμε το Product ID δεξιά και αριστερά, αφού αυτή είναι η κοινή στήλη των δύο σετ. Επιπρόσθετα κάνουμε αναζήτηση στο τμήμα Operators για την συνάρτηση με όνομα Aggregate. Είναι μία συνάρτηση τύπου group-by την οποία θα χρησιμοποιήσουμε για να ταξινομήσουμε τα δεδομένα με βάση το προϊόν. Η συνάρτηση υπολογίζει τον αριθμό κάθε προϊόντος και το όνομα του για να περιγράψει το προϊόν. Για να γίνει αυτό πρώτα πρέπει να ενώσουμε την Aggregate με το Join και μετά να αλλάξουμε τα Parameters της. Πατάμε στο κουμπί 'Select Attributes' και στο πλαίσιο που ανοίγει, παίρνουμε το 'Product ID' , το τοποθετούμε στο δεξί μέρος και πατάμε 'Apply'. Επιπλέον στο κουμπί Έdit List' τοποθετούμε στα αριστερά το 'Customer ID' και στα δεξιά την 'count' function, ενώ από κάτω πατώντας στο 'Add Entry' θα προσθέσουμε αριστερά το 'Product name' και δεξιά την επιλογή 'mode'. Στο τέλος συνδέουμε με την πύλη res και εκτελούμε το πείραμα μας. Η ολοκληρωμένη σύνδεση φαίνεται στην ΕΙΚΟΝΑ24.

EIKONA24

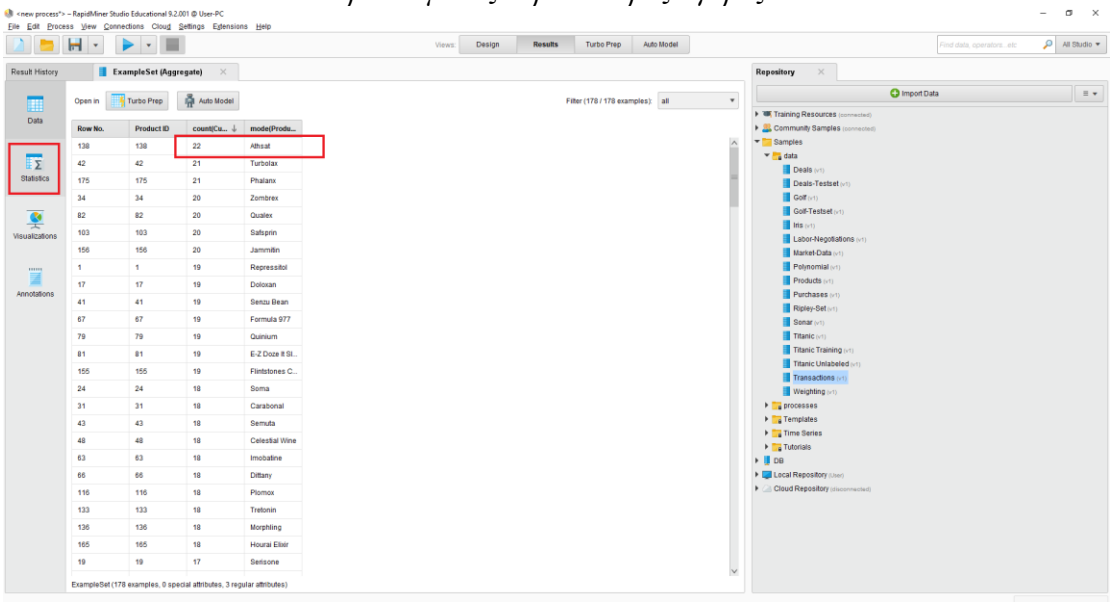
Join και Aggregate function



Όταν βρισκόμαστε στο ‘Results view’ μπορούμε να κάνουμε ταξινόμηση χωρίς να χρειάζεται ο αλγόριθμος ‘Sort’ που είχαμε χρησιμοποιήσει σε προηγούμενο παράδειγμα γι αυτή τη διαδικασία. Αν για παράδειγμα θέλουμε να ταξινομήσουμε τα προϊόντα για να δούμε ποιο πέτυχε τις περισσότερες πωλήσεις κάνουμε κλικ στην στήλη ‘count(Customer ID)’ και το βέλος πρέπει να δείχνει προς τα κάτω για την φθίνουσα σειρά. Στην EIKONA25 παρατηρούμε ότι 22 πελάτες αγόρασαν το προϊόν Athsat. Είναι επίσης ενδιαφέρον να δούμε τον μέσο όρο των συναλλαγών, πηγαίνοντας στην επιλογή Statistics θα εμφανιστούν τρεις γραμμές. Κάνοντας κλικ πάνω στο ‘count(Customer ID)’ φαίνεται το ιστόγραμμα συχνοτήτων. Για να προβληθεί καλύτερα κάνουμε κλικ στο ‘open visualizations’ και με αυτόν τον τρόπο παρατηρούμε στο διάγραμμα ότι οι συχνότερες συναλλαγές ήταν 9 - 11 πελάτες ανά είδος προϊόντος όπως φαίνεται στην EIKONA 26.

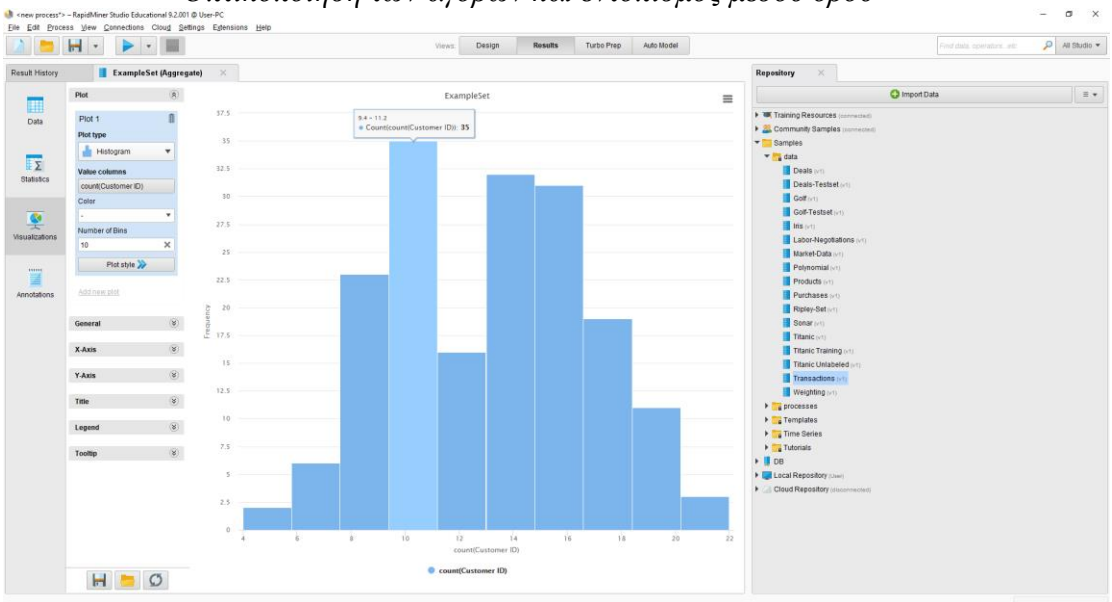
ΕΙΚΟΝΑ25

Το προϊόν με τις περισσότερες αγορές



ΕΙΚΟΝΑ26

Οπτικοποίηση των αγορών και εντοπισμός μέσου όρου

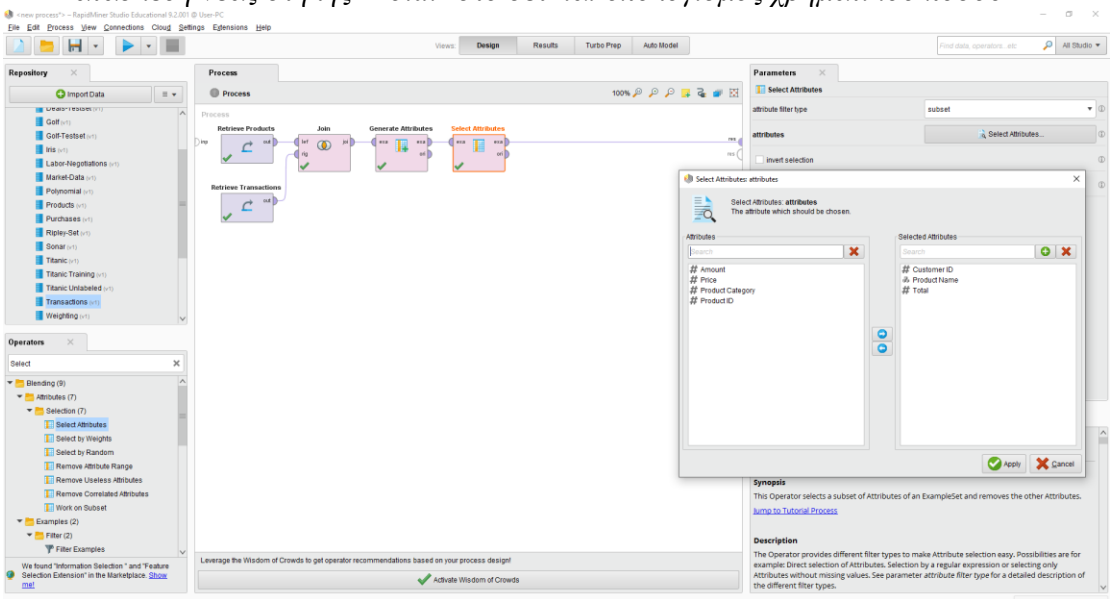


Όπως και πριν, έτσι και εδώ θα επιλέξουμε τα δύο σετ που ήδη χρησιμοποιήσαμε, δηλαδή τα Products και Transactions. Εδώ όμως θα αξιοποιήσουμε έναν διαφορετικό Operator που ονομάζεται ‘Generate Attributes’, και είναι αρμόδιος για την δημιουργία μιας νέας στήλης στα δεδομένα. Θα κατασκευάσουμε μία στήλη με όνομα Total, η οποία θα παρουσιάζει την ποσότητα επί την τιμή κάθε είδους προϊόντος. Για να γίνει αυτό εφικτό, πρέπει να συνδέσουμε τον Operator ‘Generate Attributes’ με τον Operator ‘Join’ που συνδυάζει τα δύο σετ μας. Στη συνέχεια, πατάμε στο κουμπί ‘Edit List’ που βρίσκεται στο τμήμα Parameters, για να θέσουμε ως attribute name ‘Total’ στα αριστερά, ενώ δεξιά γράφουμε Amount*Price για να υπολογιστεί η τιμή που θέλουμε και πατάμε το κουμπί ‘Apply’. Ένας άλλος τρόπος είναι να χρησιμοποιήσουμε τον Calculator που περιέχει μεγάλη ποικιλία

συναρτήσεων και πράξεων για να εκχωρίσουμε την πράξη ‘Amount*Price’. Επιπρόσθετα, θα υλοποιήσουμε έναν ακόμα Operator με όνομα ‘Select Attributes’. Τον ενώνουμε με τον προηγούμενο Operator και τροποποιούμε τα Parameters. Αλλάζουμε το attribute filter σε ‘subset’ έτσι ώστε να κρατήσουμε μόνο τις στήλες που μας ενδιαφέρουν σε αυτό το πείραμα. Για να καθορίσουμε ποιες στήλες θα παραμείνουν και ποιες θα αφαιρεθούν από το σετ, θα πατήσουμε στο ‘Select Attributes’ και στο παράθυρο που ανοίγει, θα μεταφέρουμε στη δεξιά πλευρά τις στήλες ‘Customer ID, Product Name, Total’ και τέλος θα εκτελέσουμε με το κουμπί Run αυτή τη διαδικασία. Όπως θα δούμε παρακάτω στην ΕΙΚΟΝΑ27 όλοι οι Operators είναι σωστά συνδεδεμένοι και το πείραμα εκτελέστηκε με επιτυχία. Το αποτέλεσμα που δεχόμαστε εδώ είναι μία στήλη με το όνομα ‘Total’ που είχαμε εισάγει πριν και σε αυτή φαίνονται στο ‘Results view’ τα συνολικά έξοδα που έκανε κάθε πελάτης για ένα συγκεκριμένο προϊόν. Για παράδειγμα στη πρώτη στήλη παρουσιάζεται το προϊόν με όνομα ‘Repressitol’ ο πελάτης με ID 220 και το συνολικό ποσό των 142.160. Για να δούμε τα υψηλότερα ποσά απλά κάνουμε κλικ στο όνομα της στήλης Total ώστε να γίνει αυτόματη ταξινόμηση.

ΕΙΚΟΝΑ27

Κατασκευή νέας στήλης ‘Total’ στο σετ και υπολογισμός χρηματικού ποσού

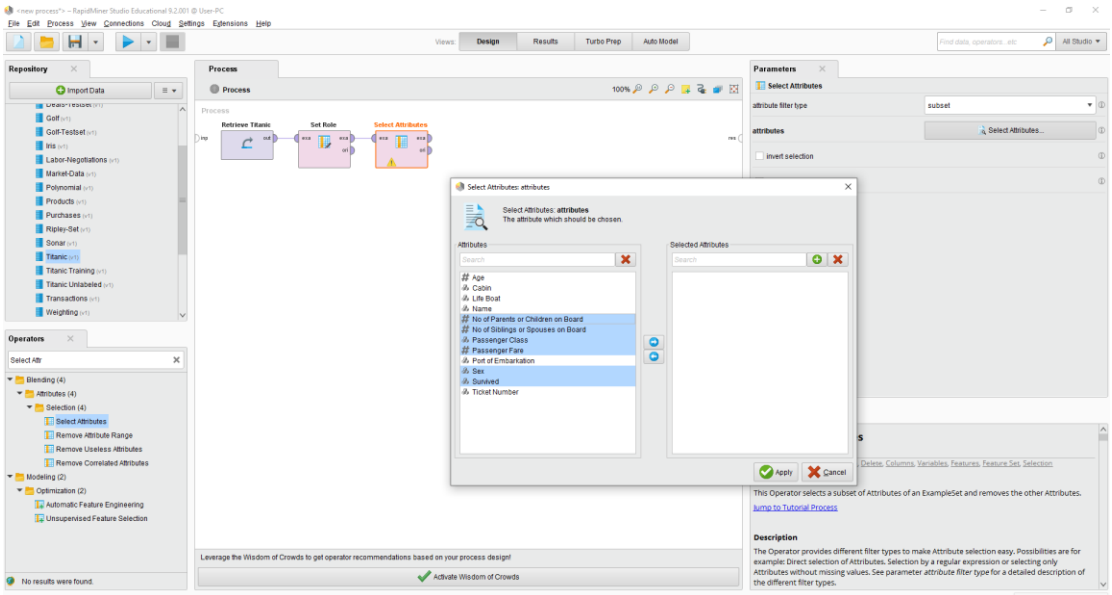


Για να ολοκληρώσουμε το πρώτο εκπαιδευτικό μέρος στο rapidminer, θα κατασκευάσουμε ένα μοντέλο, για οποίο θέλουμε να προβλέψουμε τα άτομα που είχαν την μεγαλύτερη πιθανότητα να επιζήσουν στο γεγονός που είχε συμβεί με τον Τιτανικό. Αρχικά, θα χρειαστούμε το σετ δεδομένων ‘Titanic’ από τον φάκελο data στο τμήμα Repository και το τοποθετούμε στο πλαίσιο του Process. Επόμενο βήμα είναι να αναζητήσουμε τον ‘Set Role’ Operator, ο οποίος μας βοηθάει να θέσουμε έναν ρόλο στην στήλη που πρόκειται να προβλέψει ο αλγόριθμος. Πιο συγκεκριμένα, θα αλλάξουμε το attribute name σε ‘Survived’ από το τμήμα Parameters και ως target role θα επιλέξουμε το ‘label’. Είναι σημαντικό να γίνει αυτή η διαδικασία επειδή, ο ρόλος ενός attribute, δηλαδή μίας στήλης στο σετ μας, περιγράφει πως θα χρησιμοποιηθεί η στήλη από τον αλγόριθμο μηχανικής μάθησης. Τα χαρακτηριστικά που δεν έχουν κάποιον καθορισμένο ρόλο, χρησιμοποιούνται ως είσοδος (input) για

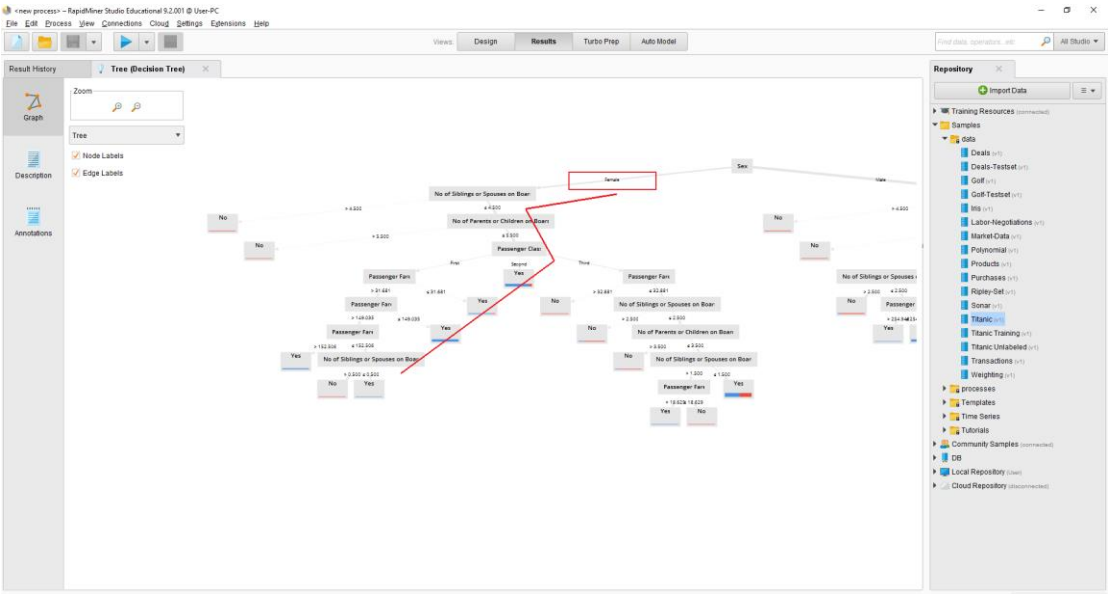
την εκπαίδευση και εμείς εδώ θέλουμε η στήλη Survived να είναι ο στόχος μας, εκείνη για την οποία ο αλγόριθμος θα κάνει την πρόβλεψη. Για τον λόγο αυτό έχουμε θέσει το ‘label’ στον Operator ‘Set Role’. Επιπλέον, χρειαζόμαστε τον ‘Select Attributes’ Operator, προκειμένου να καθορίσουμε τα χαρακτηριστικά που είναι απαραίτητα για να γίνει η πρόβλεψη. Κάνουμε κλικ στο ‘attribute filter type’ και δημιουργούμε ένα subset μέσα στο οποίο βάζουμε τα χαρακτηριστικά που θέλουμε να λάβει ως είσοδο ο αλγόριθμος μας ΕΙΚΟΝΑ28. Θα επιλέξουμε όπως και σε προηγούμενο πείραμα τον ‘Decision Tree’, θα τον συνδέσουμε με τον ‘Select Attributes’ Operator και με την πύλη res για να εκτελέσουμε το πείραμα μας. Παράγεται ένα δέντρο στο οποίο μπορούμε να παρατηρήσουμε ορισμένα μονοπάτια και να λάβουμε σημαντικές πληροφορίες για το περιστατικό του Τιτανικού ΕΙΚΟΝΑ29. Ο αλγόριθμος έδειξε ότι οι γυναίκες που είχαν μικρές οικογένειες εμφανίστηκαν στα φύλλα του δέντρου όπου υπήρξαν οι περισσότεροι επιζώντες. Πέρα από τον παράγοντα ‘οικογένεια’ έπαιξε σημαντικό ρόλο και η τάξη και η τιμή του εισιτηρίου. Το δέντρο αποκαλύπτει ότι οι γυναίκες με τα πιο ακριβά εισιτήρια είχαν μεγάλη πιθανότητα να επιζήσουν και αν μεταφέρουμε τον κέρσορα πάνω σε ένα από αυτά τα φύλλα του δέντρου, βλέπουμε ακριβώς των αριθμό των ατόμων με τιμή Yes και No.

ΕΙΚΟΝΑ28

Καθορισμός των απαραίτητων χαρακτηριστικών για την πρόβλεψη



ΕΙΚΟΝΑ29
Παραγόμενο δέντρο



Η κόκκινη γραμμή δείχνει το μονοπάτι με τις γυναίκες που είχαν μικρό αριθμό οικογένειας (γονείς, παιδιά, κ.α.), την τάξη της καμπίνας, δηλαδή αν ήταν first, second, third class καθώς και το που κυμαίνεται η τιμή του εισιτηρίου.

Έχουμε ολοκληρώσει το πρώτο μέρος της εκπαίδευσης χρήστη με το λογισμικό rapidminer και υλοποιήσαμε έως εδώ αρκετούς και μάλιστα πολύ σημαντικούς Operators για τα διάφορα σύνολα δεδομένων, όπως επίσης και κάποιους αλγορίθμους που είχαμε αναλύσει τον τρόπο με τον οποίο λειτουργούν σε προηγούμενα κεφάλαια

10.3 Προετοιμασία και επεξεργασία συνόλων δεδομένων

Η ορθή προετοιμασία των δεδομένων είναι ίσως το κομμάτι στο οποίο ο αναλυτής αφιερώνει τον περισσότερο χρόνο. Αυτό συμβαίνει γιατί, υπάρχουν δύο βασικές μέθοδοι επεξεργασίας και κατάλληλης διαχείρισης των δεδομένων, οι οποίες είναι η ανάμιξη (blending) και ο καθαρισμός (cleansing). Η πρώτη αναφέρεται στην μετατροπή ενός συνόλου δεδομένων από μία αρχική κατάσταση σε μία άλλη ή επίσης μπορεί να σημαίνει συνδυασμός διαφορετικών σετ. Η δεύτερη είναι μια διαδικασία κατά την οποία βελτιώνουμε το περιεχόμενο των δεδομένων λαμβάνοντας έτσι καλύτερης ή υψηλότερης ακρίβειας αποτελέσματα στα πειράματα μας.

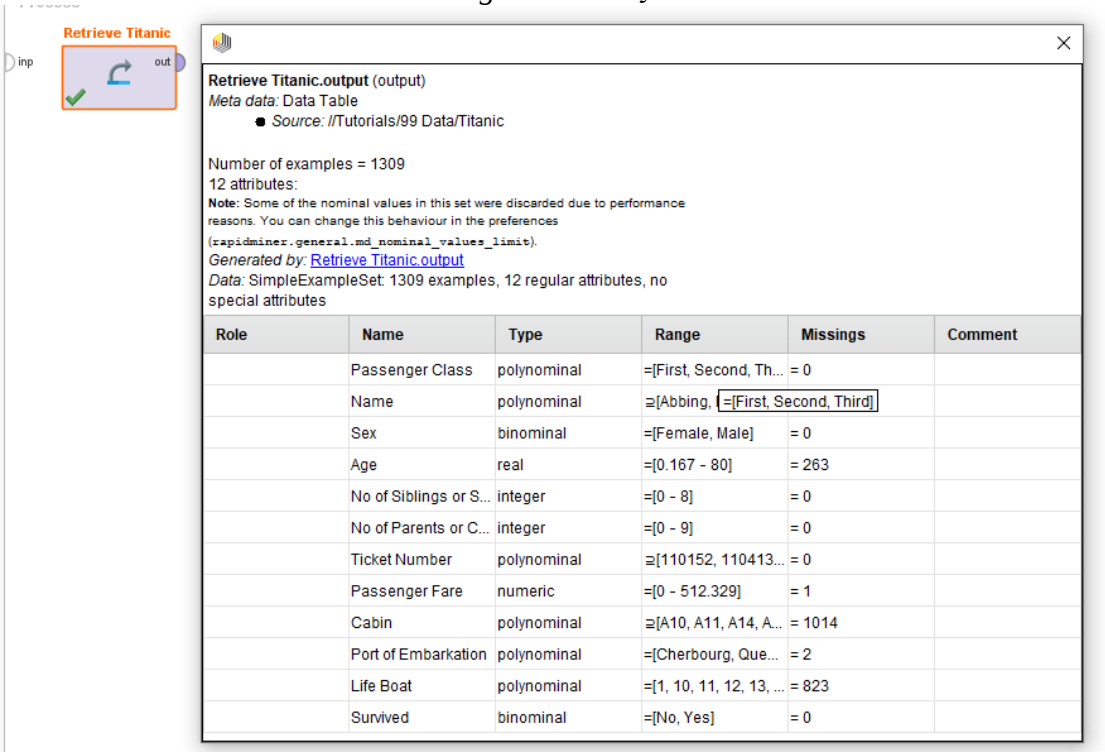
Στο λογισμικό rapidminer μπορούμε να δούμε τις στήλες ενός σετ, οι οποίες έχουν απώλεια τιμών, τα κελιά τους συμβολίζονται με (?) και φαίνονται στο Results view. Αυτές οι ελλείψεις τιμών των στηλών, αποτελούν πρόβλημα για την προετοιμασία και για τις διαδικασίες μοντελοποίησης.

Για ακόμα μία φορά θα μελετήσουμε το σετ δεδομένων Titanic, που έχουμε δει στο πρώτο εκπαιδευτικό μέρος. Αν αφήσουμε τον κέρσορα πάνω στην πύλη εξόδου του Titanic, θα ανοίξει ένα μικρό παράθυρο και για να παραμείνει ανοιχτό πρέπει να πατήσουμε F3. Τώρα μπορούμε να διακρίνουμε ότι στην στήλη ‘Missing’, υπάρχουν ορισμένα χαρακτηριστικά που εμφανίζουν έλλειψη τιμών. Αυτά τα πέντε attributes είναι (‘Age, Passenger Fare, Cabin, Port of Embartation, Life Boat’) και εμείς θα πρέπει να αντιμετωπίσουμε το πρόβλημα με αυτές τις τιμές. Στην ΕΙΚΟΝΑ30 παρουσιάζεται

στα δεξιά η στήλη ‘Missings’ με όλα τα χαρακτηριστικά και σε όσα δεν έχουμε (=0), υπάρχει απώλεια των τιμών.

ΕΙΚΟΝΑ30

Τα missing values ενός σετ

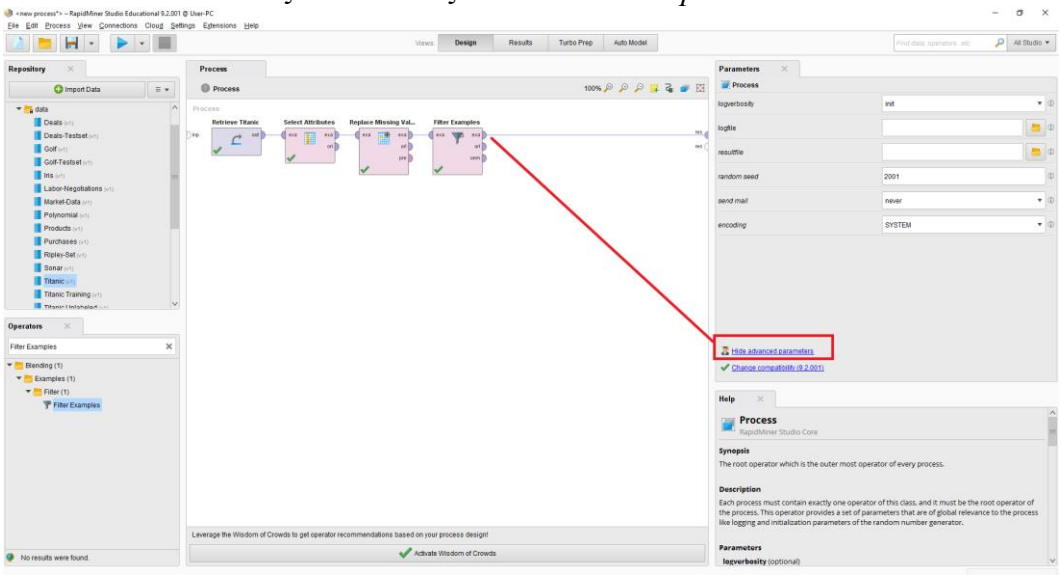


Role	Name	Type	Range	Missings	Comment
	Passenger Class	polynominal	=[First, Second, Th...	= 0	
	Name	polynominal	≥[Abbing, =[First, Second, Third]		
	Sex	binominal	=[Female, Male]	= 0	
	Age	real	=[0.167 - 80]	= 263	
	No of Siblings or S...	integer	=[0 - 8]	= 0	
	No of Parents or C...	integer	=[0 - 9]	= 0	
	Ticket Number	polynominal	≥[110152, 110413...	= 0	
	Passenger Fare	numeric	=[0 - 512.329]	= 1	
	Cabin	polynominal	≥[A10, A11, A14, A...	= 1014	
	Port of Embarkation	polynominal	=[Cherbourg, Que...	= 2	
	Life Boat	polynominal	=[1, 10, 11, 12, 13, ...	= 823	
	Survived	binominal	=[No, Yes]	= 0	

Ως πρώτο βήμα, θα συνδέσουμε τον ‘Select Attributes’ Operator με το σετ και μετά θα δημιουργήσουμε ένα ‘subset’ μέσα στο οποίο κρατάμε όλα τα χαρακτηριστικά εκτός των ‘Cabin’ και ‘Life Boat’. Αυτά τα δύο έχουν πάρα πολύ μεγάλες απώλειες και δεν είναι χρήσιμα αυτή τη στιγμή, επομένως τα αφαιρέσαμε. Ως δεύτερο βήμα, θα προσθέσουμε τον ‘Replace Missing Values’ Operator και στο τμήμα παραμέτρων θα θέσουμε ως ‘single’ το attribute filter και ως attribute το Age. Ο λόγος που το κάνουμε αυτό στο χαρακτηριστικό ‘Age’ είναι ότι έχει 263 ελλείψεις τιμών, αλλά με τη χρήση αυτού του Operator θα καλυφθούν όλα τα κενά εισάγοντας τον μέσο όρο της ηλικίας. Ως τρίτο και τελευταίο βήμα θα αναζητήσουμε το φίλτρο ‘Filter Examples’ στους Operators. Πρώτα το συνδέουμε με τον προηγούμενο Operator από την μία πλευρά και με την έξοδο από την άλλη. Στις παραμέτρους αυτού, υπάρχει μία επιλογή ως ‘Show advanced parameters’ και αφού κάνουμε κλικ εκεί θα βάλουμε στο condition class την τιμή ‘no_missing_attributes’. Έτσι εξαλείφονται και οι λίγες ελλείψεις τιμών που είχαν απομείνει και θα εκτελέσουμε πλέον το πείραμα κανονικά.

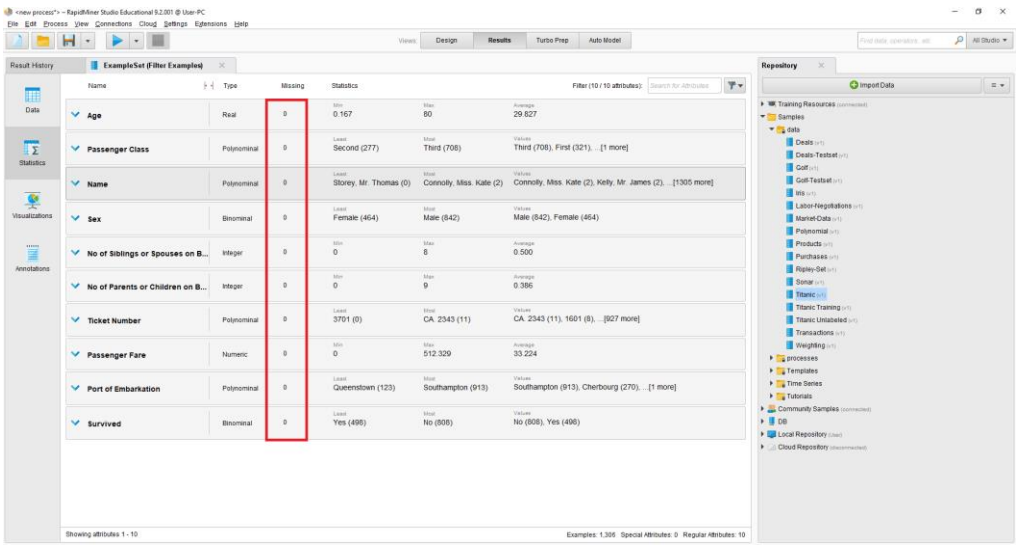
ΕΙΚΟΝΑ31

Όλες οι συνδέσεις και τα advanced parameters



ΕΙΚΟΝΑ32

Στήλη χωρίς απώλειες τιμών



Ένα άλλο σημαντικό βήμα για τον καθαρισμό των δεδομένων είναι να εντοπίσουμε ασυνήθιστες περιπτώσεις και να τις αφαιρέσουμε από το σετ. Σε κάποιες καταστάσεις η ανίχνευση τέτοιων περιπτώσεων μπορεί να παρέχει τις πιο ενδιαφέρουσες πληροφορίες. Ένα παράδειγμα αποτελούν οι απατηλές συναλλαγές που γίνονται με τις πιστωτικές κάρτες. Ωστόσο, αυτές οι περιπτώσεις, τις περισσότερες φορές είναι αποτελέσματα από λάθος μετρήσεις και θα πρέπει να καταργηθούν μέσα από το σετ.

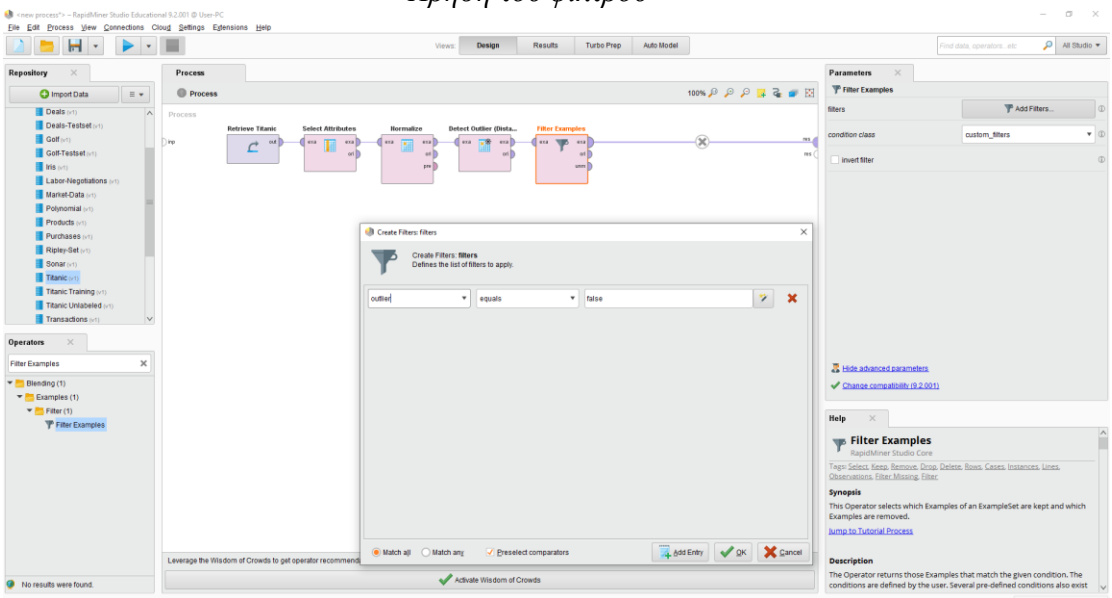
Θα χρησιμοποιήσουμε το Titanic σετ και θα το ενώσουμε πρώτα με τον ‘Select Attributes’ Operator. Στο τμήμα Parameter δημιουργούμε και πάλι ένα subset μέσα στο οποίο θα κρατήσουμε όλα τα χαρακτηριστικά εκτός των ‘Cabin’, ‘Life Boat’, ‘Name’ και ‘Ticket Number’. Το αποτέλεσμα θα είναι ένα σύνολο με χαρακτηριστικά που

συμβάλλουν στον εντοπισμό των ‘outliers’, δηλαδή των περιπτώσεων που εξηγήσαμε στην προηγούμενη παράγραφο. Για να πραγματοποιηθεί αυτό, θα υλοποιήσουμε έναν αλγόριθμο που υπολογίζει την Ευκλείδεια απόσταση στα σημεία των δεδομένων και σημαδεύει τα σημεία που βρίσκονται σε μακρινές αποστάσεις ως ‘outliers’. Θα πρέπει να εξασφαλίσουμε ότι τα χαρακτηριστικά είναι σε παρόμοιες αποστάσεις μεταξύ τους. Αυτή η μετατροπή που πραγματοποιούμε ονομάζεται ‘Normalization’, έτσι θα αναζητήσουμε τον ‘Normalize’ Operator και μετά θα τον συνδέσουμε με τον ‘Select Attributes’, ώστε όλα τα χαρακτηριστικά να βρίσκονται σε μία ίδια κλίμακα μετά την εξομάλυνση και να μπορούν να συγκριθούν μεταξύ τους. Στη συνέχεια γράφουμε στην αναζήτηση του τμήματος Operators ‘Detect Outlier(Distances)’ και ενώνουμε με τα προηγούμενα. Είναι ένας Operator, ο οποίος by default χρήση, εντοπίζει 10 παραδείγματα και τα επισημειώνει ως ‘outliers’. Επιπλέον παράγει μία νέα στήλη με όνομα ‘outlier’, η οποία έχει την τιμή ‘true’ για όλα τα ‘outliers’ που βρέθηκαν, ενώ τιμή ‘false’ για τα υπόλοιπα παραδείγματα. Στο τέλος προσθέτουμε το φίλτρο ‘Filter Examples’ και μέσα στο πλαίσιο εκχωρούμε την πράξη ‘outlier equals false’ με τον ίδιο τρόπο που είχαμε παρουσιάσει στο πρώτο εκπαιδευτικό μέρος του rapidminer. Ένα ακόμα ενδιαφέρον σημείο για αυτό το πείραμα είναι να πάμε στον Operator ‘Detect Outlier(Distances)’, να πατήσουμε δεξί κλίκ και να θέσουμε σε αυτόν την επιλογή ‘Breakpoint after’. Το ‘Breakpoint after’ είναι ένα χρήσιμο εργαλείο για debugging. Εδώ αν τρέξουμε το πείραμα χωρίς αυτή την προσθήκη θα δούμε στο Results view και μετά στο Statistics tab ότι αφαιρέθηκαν τα 10 ‘outliers’, και έχουμε συνολικά 1299 παραδείγματα από τα 1309 που ήταν. Αν όμως το εκτελέσουμε με την προσθήκη αυτή, τότε θα μπορούμε να δούμε στα Results και τα 10 ‘outliers’.

Στην επόμενη σελίδα παρατηρούμε αρχικά στην ΕΙΚΟΝΑ33 πως χρησιμοποιούμε το φίλτρο. Στην ΕΙΚΟΝΑ34 παρουσιάζεται ο τρόπος με τον οποίο μπορούμε να θέσουμε ένα ‘Breakpoint after’ καθώς και η τελική σύνδεση των Operators. Στην ΕΙΚΟΝΑ35 φαίνονται τα αποτελέσματα μετά την εκτέλεση αυτού του πειράματος.

ΕΙΚΟΝΑ33

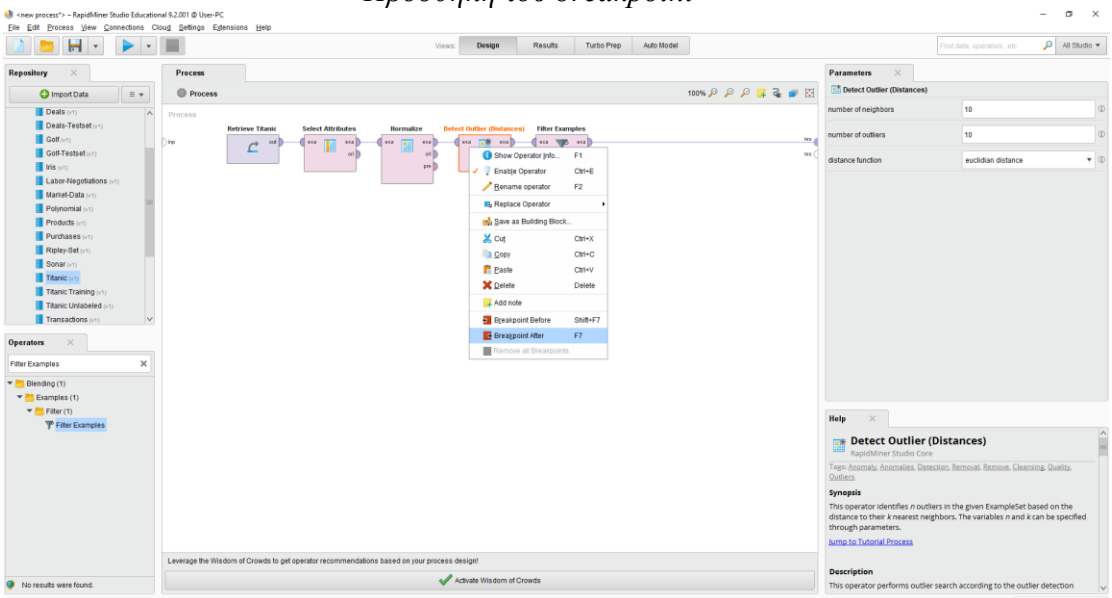
Χρήση του φίλτρου



Η πράξη ‘outlier equals false’ της εικόνας 33, τοποθετείται στο σετ για να προσδιορίσει τα παραδείγματα που θέλουμε να αφαιρεθούν από το δείγμα μας.

ΕΙΚΟΝΑ34

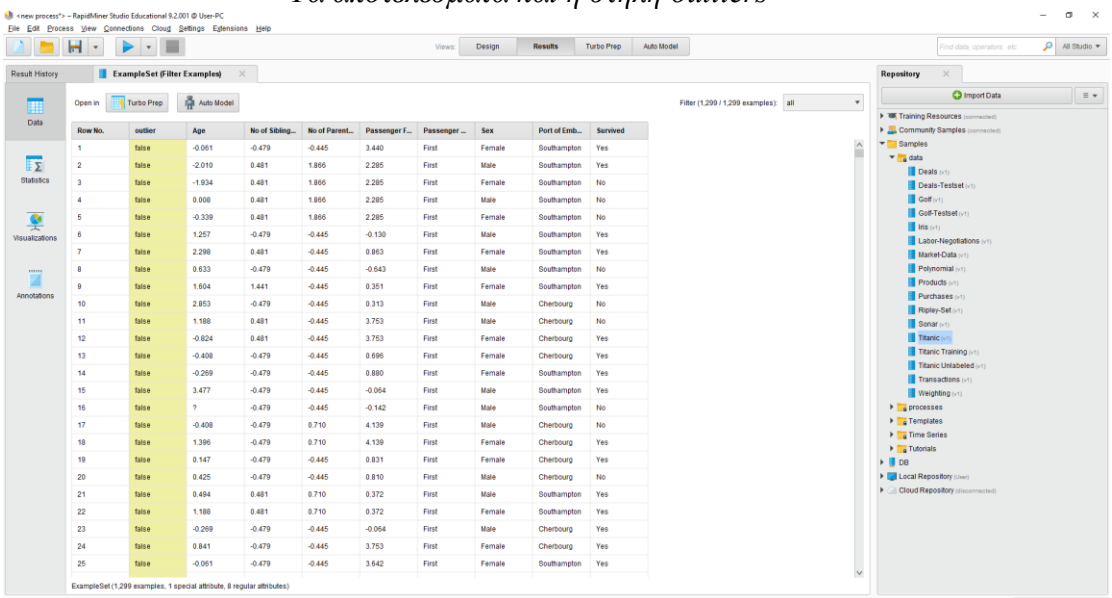
Προσθήκη του breakpoint



Το ‘Breakpoint after’ εισάγεται στον Operator ώστε να μπορούμε να δούμε τα ‘outliers’ που έχουν επισημειωθεί ως ‘προβληματικά’ πριν αυτά διαγραφούν από το δείγμα.

ΕΙΚΟΝΑ35

Τα αποτελέσματα και η στήλη outliers



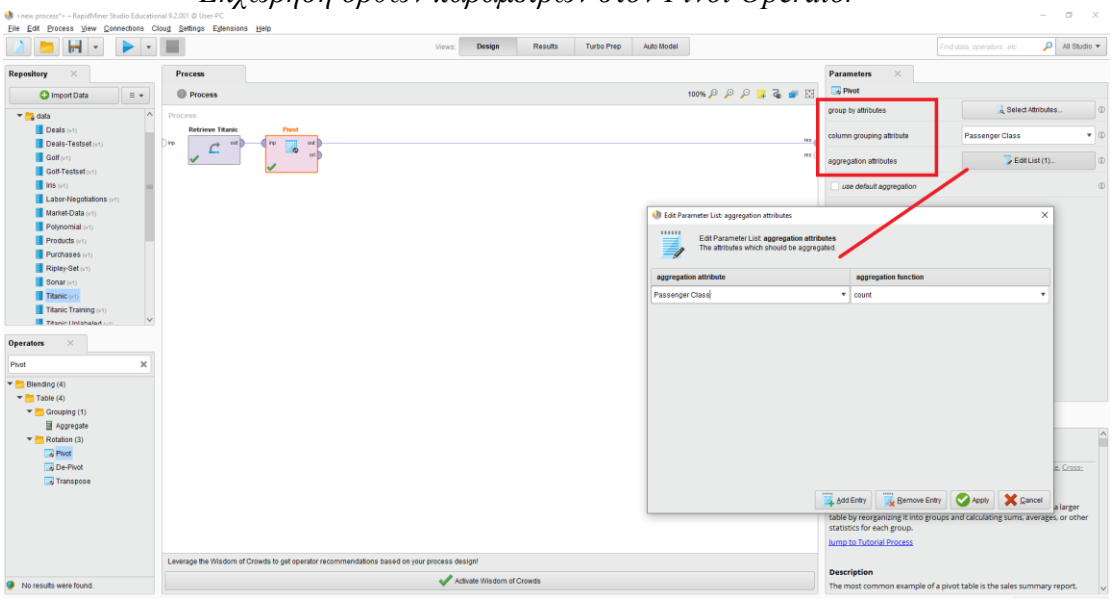
Στο ακόλουθο παράδειγμα θα παρουσιάσουμε μία ακόμα πολύ σημαντική τεχνική της προετοιμασίας των δεδομένων η οποία ονομάζεται Pivoting. Αν έχουμε για

παράδειγμα έναν πίνακα και στο περιεχόμενο του ένα μόνο χαρακτηριστικό με πολλά διαφορετικά παραδείγματα, τότε θέλουμε να μετατρέψουμε αυτόν τον πίνακα σε έναν με πολλά χαρακτηριστικά τα οποία θα έχουν ένα παράδειγμα το καθένα. Αυτή η τροποποίηση είναι εξαιρετικά χρήσιμη για την συγκέντρωση πληροφοριών σε δύο η περισσότερες διαστάσεις και την προετοιμασία των δεδομένων αυτών για μηχανική μάθηση. Τα μοντέλα μηχανική μάθησης χρειάζονται τα δεδομένα να είναι αποθηκευμένα σε πίνακες ευρείας μορφής (wide format) και γι αυτό θα δείξουμε πως γίνεται μία τέτοια μετατροπή στο rapidminer.

Θα κατασκευάσουμε έναν πίνακα που θα περιέχει του επιβάτες κάθε class (first class, second class, third class) και θα είναι χωρισμένοι ανάλογα με το φύλλο και την τάξη στην οποία ανήκουν. Αρχικά τοποθετούμε το σετ Titanic μέσα στην επιφάνεια του τμήματος Process. Αναζητούμε τον Operator 'Pivot' και τον συνδέουμε με το Titanic σετ. Στο τμήμα των Parameters που αφορούν αυτόν τον Operator, προσθέτουμε το φύλλο 'Sex' στο πεδίο 'group by attributes', ενώ στο πεδίο 'column grouping attribute' επιλέγουμε το 'Passenger Class' για να μοιραστούν τα άτομα ανάλογα με το φύλλο και την τάξη τους όπως φαίνεται στην EIKONA36 στην επόμενη σελίδα. Επίσης στο πεδίο 'aggregation attributes' τοποθετούμε το 'Passenger Class' με την συνάρτηση count για να δούμε στο τέλος τον συνολικό αριθμό επιβατών ανά κατηγορία. Το αποτέλεσμα που προκύπτει είναι ένας πίνακας με δύο γραμμές που αναφέρονται στο φύλλο (Male, Female) και τρεις στήλες για τα διαφορετικές τάξεις των επιβατών. Αυτή τη στιγμή αν εκτελέσουμε τη διαδικασία θα παραχθεί ένας πίνακας που στις στήλες αναγράφεται η πράξη που έγινε π.χ (count(Passenger Class)_First). Για τον λόγο αυτό θα δώσουμε ένα πιο αντιπροσωπευτικό όνομα σε κάθε στήλη, παρά να βλέπουμε την πράξη που έγινε σε αυτά τα δεδομένα, γιατί μερικές φορές είναι δύσκολο να καταλάβουμε πως έχει γίνει ο διαχωρισμός.

EIKONA36

Εκχώρηση ορθών παραμέτρων στον Pivot Operator



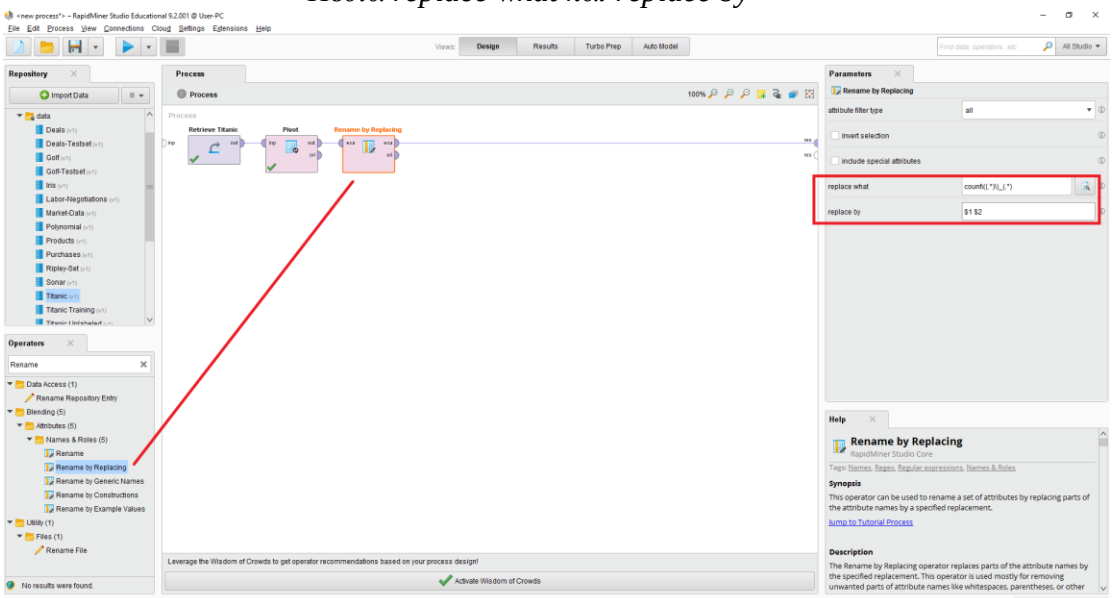
Με τον Rename Operator μπορούμε να μετονομάσουμε την κάθε στήλη για τα τρία αυτά χαρακτηριστικά, όμως αυτός δεν είναι ο καλύτερος τρόπος. Υπάρχουν τρόποι να

κάνουμε μαζική μετονομασία γιατί σε κάποιες περιπτώσεις μπορεί να έχουμε εκατοντάδες χαρακτηριστικά στο σετ μας.

Αναζητούμε τον Operator ‘Rename by Replacing’, τον προσθέτουμε στο τμήμα Process και τον συνδέουμε με τα προηγούμενα. Στις παραμέτρους αυτού γράφουμε αρχικά στο πεδίο ‘replace what’ την έκφραση ή πράξη count\((.*)\)_(.*) ακριβώς όπως είναι εδώ και στο πεδίο ‘replace by’ γράφουμε $1 $2 με το κενό ανάμεσα τους. Η πρώτη πρόκειται για ένα regular expression που σημαίνει ότι ψάχνουμε να βρούμε κάτι ανάμεσα στο count και το σύμβολο (_) και στη συνέχεια ότι ψάχνουμε κάτι μετά το σύμβολο (_). Η δεύτερη αναφέρεται στα δύο τμήματα, δηλαδή το $1 στο Passenger Class και το $2 στα τρία διαφορετικά class (‘First’, ‘Second’, ‘Third’).

EIKONA37

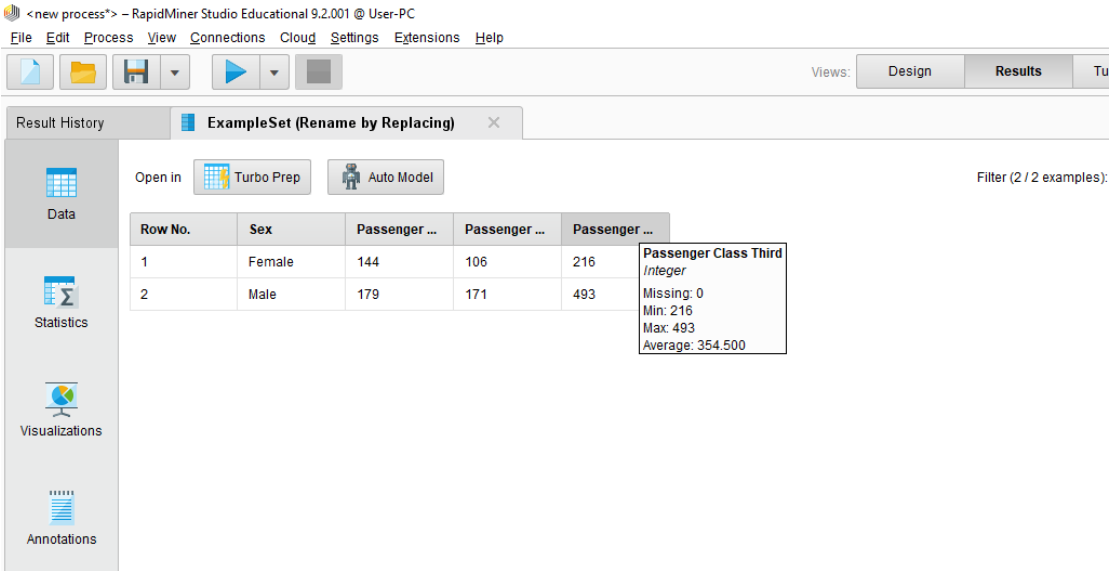
Πεδία replace what και replace by



Τώρα αν εκτελέσουμε τη διαδικασία θα δούμε στο Results view ότι οι στήλες έχουν αλλάξει και τα ονόματα τους είναι πλέον ‘Passenger Class First’, ‘Passenger Class Second’ , ‘Passenger Class Third’. Έτσι έχουμε καταφέρει να οργανώσουμε με καλύτερο τρόπο το σετ δεδομένων πριν περάσουμε στο κομμάτι της μοντελοποίησης και χρησιμοποιήσουμε αλγορίθμους μηχανικής μάθησης.

ΕΙΚΟΝΑ38

Η μετονομασία των Passenger Class



Έως τώρα έχουμε παρουσιάσει πολλούς τρόπους διαχείρισης των δεδομένων, όμως είναι πιθανό να έχουμε το ερώτημα, πώς τελικά εξάγουμε το σετ δεδομένων από το λογισμικό rapidminer. Είτε πρόκειται για κάποια συνεργασία, είτε για πιο εξειδικευμένη ανάλυση ή για άλλους σκοπούς, είναι πολύ βασικό να γνωρίζουμε με ποιον τρόπο γίνεται αυτό εφικτό. Έτσι μέσα από αυτό το tutorial, το οποίο θα είναι και το τελευταίο tutorial του δεύτερου εκπαιδευτικού μέρους, θα παρουσιάσουμε τη διαδικασία με την οποία εξάγουμε τα δεδομένα από το rapidminer.

Επιλέγουμε το Titanic dataset και το εισάγουμε στο τμήμα Process. Στο πλαίσιο αναζήτησης των Operators, πληκτρολογούμε 'Write' και εμφανίζονται διάφορες επιλογές. Για την ώρα θα πάρουμε τον 'Write Excel' Operator και θα τον ενώσουμε με το σετ μας. Ωστόσο πρέπει στο τμήμα Parameters να προσδιορίσουμε το σημείο που επιθυμούμε να αποθηκεύσουμε το αρχείο XLS. Για να το κάνουμε αυτό πατάμε στο κουμπί με το εικονίδιο του φακέλου, επιλέγουμε την επιφάνεια εργασίας για παράδειγμα και αφού δώσουμε ένα όνομα στον φάκελο τότε πατάμε στο κουμπί Open. Πριν προχωρήσουμε παρακάτω, μπορούμε να δοκιμάσουμε έχοντας συνδέσει και με την πύλη εξόδου τον Operator, να εκτελέσουμε τη διαδικασία. Θα δούμε ότι στην επιφάνεια εργασίας έχει σχηματιστεί ένας νέος φάκελος με το όνομα που δώσαμε.

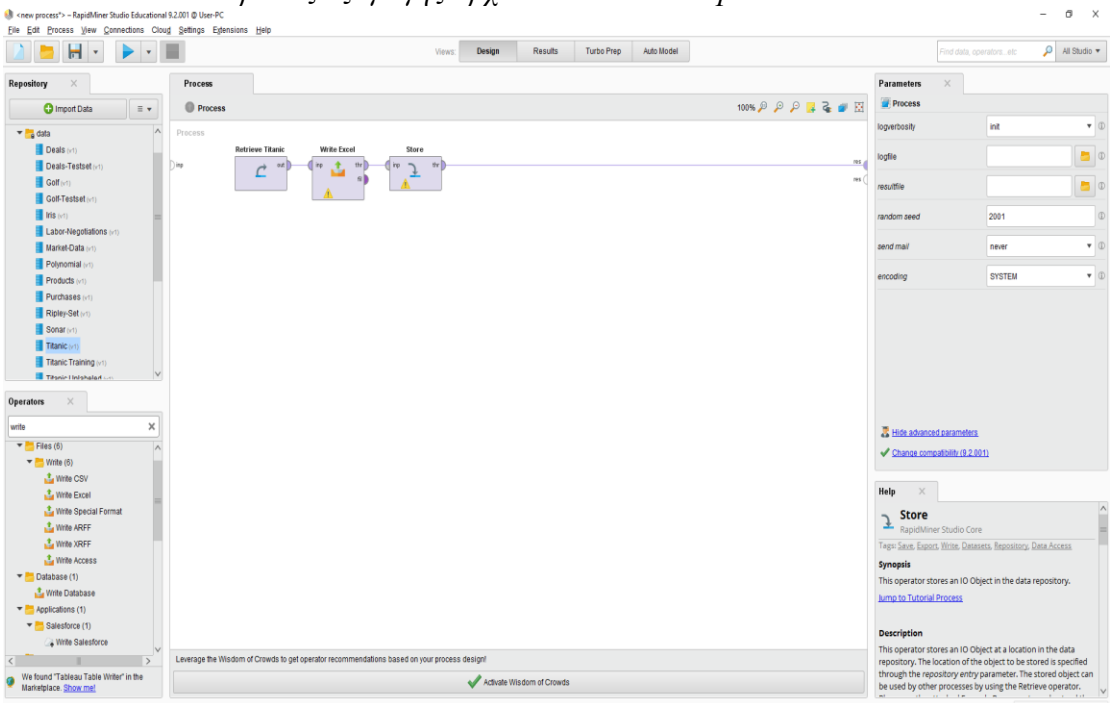
Στο rapidminer υπάρχει μεγάλη ποικιλία από Operators που είναι υπεύθυνοι για την εξαγωγή αρχείων σε διαφορετικά format. Μπορούμε να διακρίνουμε Operators για database formats, καθώς και για business applications που αφορούν CRM και ERP συστήματα. Μέσα σε μία επιχείρηση θα χρειαστεί να χρησιμοποιήσουμε πολλές φορές τα μοντέλα πρόβλεψης και γι' αυτό η σύνδεση του με άλλα συστήματα είναι εξαιρετικής σημασίας. Αν θέλουμε να βρούμε περισσότερα file formats από αυτά που υπάρχουν ήδη μέσα στους Operators, μπορούμε να ανοίξουμε το Marketplace που βρίσκεται στο κάτω μέρος των Operators για την παροχή επιπλέον formats.

Για να αποθηκεύσουμε μέσα στο τοπικό (local) Repository αυτή τη διαδικασία, βρίσκουμε και τοποθετούμε τον Operator 'Store' στο Process. Στο τμήμα των παραμέτρων αυτού, ορίζουμε στο πεδίο 'repository entry' ότι θα αποθηκευτεί στο Local Repository και στο τέλος εκτελούμε με το κουμπί Run. Αν κοιτάξουμε πάλι στην επιφάνεια εργασίας, μέσα στον φάκελο που είχαμε δημιουργήσει πριν, τώρα θα υπάρχει εκεί και το αρχείο Excel, το οποίο μπορούμε να ανοίξουμε και στο περιεχόμενο

του θα βρίσκονται όλα τα δεδομένα του Titanic. Με τον ίδιο ακριβώς τρόπο εργαζόμαστε αν θέλουμε να εξάγουμε ένα αρχείο σε μορφή CSV, το οποίο εύκολα να εισάγουμε και στο λογισμικό WEKA όπως είχαμε δείξει εκεί. Στην ΕΙΚΟΝΑ39 παρουσιάζονται όλες οι συνδέσεις που πρέπει να γίνουν εδώ για την εξαγωγή του αρχείου στον τοπικό υπολογιστή.

ΕΙΚΟΝΑ39

Τρόπος εξαγωγής αρχείου Excel στο rapidminer



Συμπεριλαμβάνοντας όλα τα παραπάνω που αναλύθηκαν στο δεύτερο μέρος, έχουμε δει πως πρέπει να επεξεργαστούμε τα δεδομένα πριν κάνουμε οποιαδήποτε άλλη δράση. Είναι σημαντικό να αφαιρούμε τμήματα του σετ τα οποία φαίνεται να είναι από την αρχή ‘προβληματικά’ όπως για παράδειγμα στήλες με πάρα πολλές απώλειες τιμών (missing values). Υπάρχουν περιπτώσεις που μπορούμε να καλύψουμε αυτά τα κενά, αν δεν είναι πολύ μεγάλα με διάφορους τρόπους, ένας εκ των οποίων είναι η συμπλήρωση των κενών κελιών με τον μέσο όρο των τιμών του συγκεκριμένου χαρακτηριστικού. Πέρα από αυτό το βήμα, για να έχουμε τα βέλτιστα αποτελέσματα στις προβλέψεις, είναι καλό να τροποποιούμε έναν πίνακα με τέτοιο τρόπο ώστε να έχουμε περισσότερα χαρακτηριστικά με λιγότερα παραδείγματα, δηλαδή να τον επεκτείνουμε σε πλάτος για να γίνει καλύτερη και πιο αναλυτική η διάκριση του περιεχομένου. Τέλος παρουσιάσαμε κάτι το οποίο πρέπει να γνωρίζει κάθε χρήστης του λογισμικού αυτού και είναι ο τρόπος με τον οποίο εξάγουμε ένα αρχείο σε οποιοδήποτε format που διαθέτουν οι Operators του rapidminer.

10.4 Κατασκευή μοντέλων προβλέψεων

Μέχρι αυτό το σημείο, εκτελέσαμε τα πιο σημαντικά βήματα για την διαχείριση των δεδομένων, τον συνδυασμό τους, την προετοιμασία τους και το πως παράγουμε νέες και ευέλικτες διαδικασίες. Σε αυτό το κομμάτι, θα αφιερώσουμε χρόνο στο να κατασκευάσουμε μοντέλα προβλέψεων, τα οποία θα χρησιμοποιήσουμε για την επίτευξη συγκεκριμένων στόχων και τέλος θα επικυρώσουμε το πόσο καλά ανταποκρίνονται αυτά τα μοντέλα σε νέες διαδικασίες και συνθήκες.

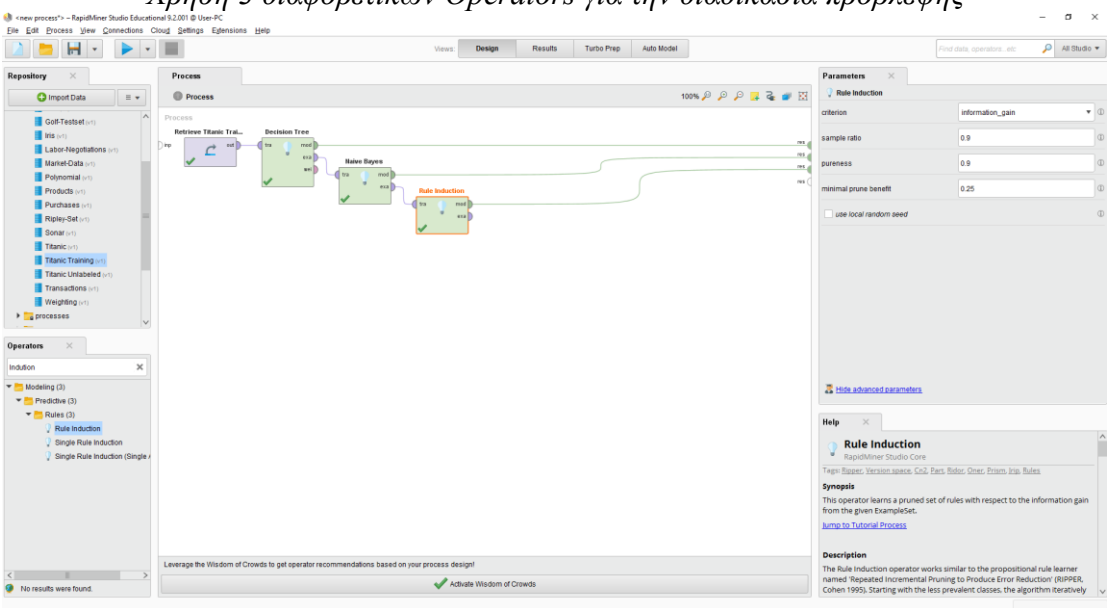
Ο όρος 'Predictive Modeling' αναφέρεται σε ένα σύνολο αποτελούμενο από τεχνικές της μηχανικής μάθησης, με τις οποίες αναζητούμε μονοπάτια μέσα σε πολύ μεγάλα σετ δεδομένων, ώστε να πραγματοποιηθούν προβλέψεις για νέες καταστάσεις η συνθήκες. Αυτές οι προβλέψεις διακρίνονται σε κατηγορικές και πρόκειται για τις μεθόδους ταξινόμησης και σε αριθμητικές που αφορούν την παλινδρόμηση. Αυτά τα μοντέλα μας βοηθούν στο να κατανοήσουμε καλύτερα τον τρόπο με τον οποίο οδηγηθήκαμε σε συγκεκριμένα αποτελέσματα.

Στο πρώτο tutorial του τρίτου μέρους θα δημιουργήσουμε τρία διαφορετικά μοντέλα ταξινόμησης για το σετ δεδομένων Titanic. Αυτά θα είναι, ένα δέντρο απόφασης, ένα σετ κανόνων και ένα μοντέλο Bayes. Ας εξερευνήσουμε τις τρεις μεθόδους για να δούμε αν τελικά μπορούμε να πάρουμε περισσότερες πληροφορίες για το ατύχημα και επίσης ποιος είχε την υψηλότερη πιθανότητα να σωθεί.

Θα επιλέξουμε το Titanic Training σετ και θα το τοποθετήσουμε στην επιφάνεια του Process. Πιο συγκεκριμένα, αυτό το σύνολο δεδομένων είναι ήδη επεξεργασμένο και έχει περάσει το στάδιο της προετοιμασίας που έχουμε ήδη αναλύσει. Μέσα σε αυτό δεν υπάρχουν απώλειες τιμών (missing values), και έχει καθοριστεί το label εκ των προτέρων. Το label αυτό είναι η στήλη 'Survived' για την οποία θέλουμε να γίνει πρόβλεψη. Άρα, διαθέτοντας το ορθό σετ μπορούμε να το εισάγουμε στη μέθοδο μηχανικής μάθησης που θα αναπτύξουμε. Στην ΕΙΚΟΝΑ40 φαίνονται οι Operators που θέλουμε να υλοποιήσουμε, ενώ στην επόμενη σελίδα βρίσκεται η περιγραφή για το πως τους συνδέουμε.

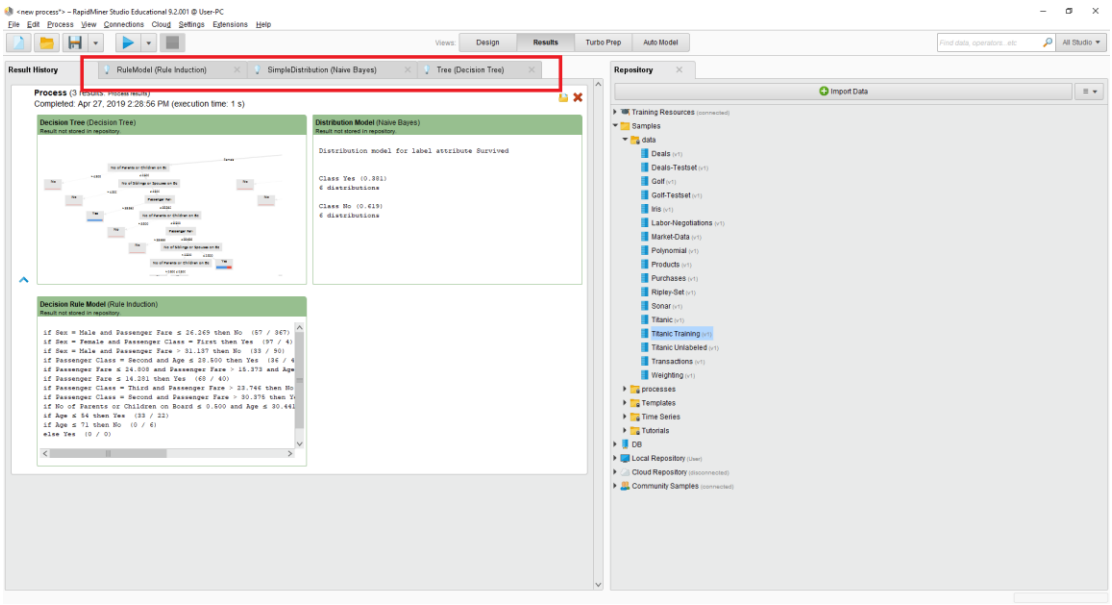
ΕΙΚΟΝΑ40

Χρήση 3 διαφορετικών Operators για την διαδικασία πρόβλεψης



Αρχικά παίρνουμε από τους Operators τον Decision Tree και τον ενώνουμε με το σετ. Δεύτερος Operator θα είναι ο Naive Bayes τον οποίο θα συνδέσουμε με την πύλη εξόδου ‘exa’ του προηγούμενου Operator. Τρίτος Operator θα είναι ο Rule Induction για τον οποίο θα επαναλάβουμε τον ίδιο τρόπο σύνδεσης, μόνο που τον ενώνουμε με τον Naive Bayes σύμφωνα με την ΕΙΚΟΝΑ40. Στη συνέχεια πρέπει να συνδέσουμε την κάθε πύλη εξόδου ‘mod’ αυτών των τριών με κάθε μία από τις πύλες ‘res’ που θα εμφανιστούν στο δεξί μέρος. Αφού γίνει αυτό εκτελούμε κανονικά το πείραμα.

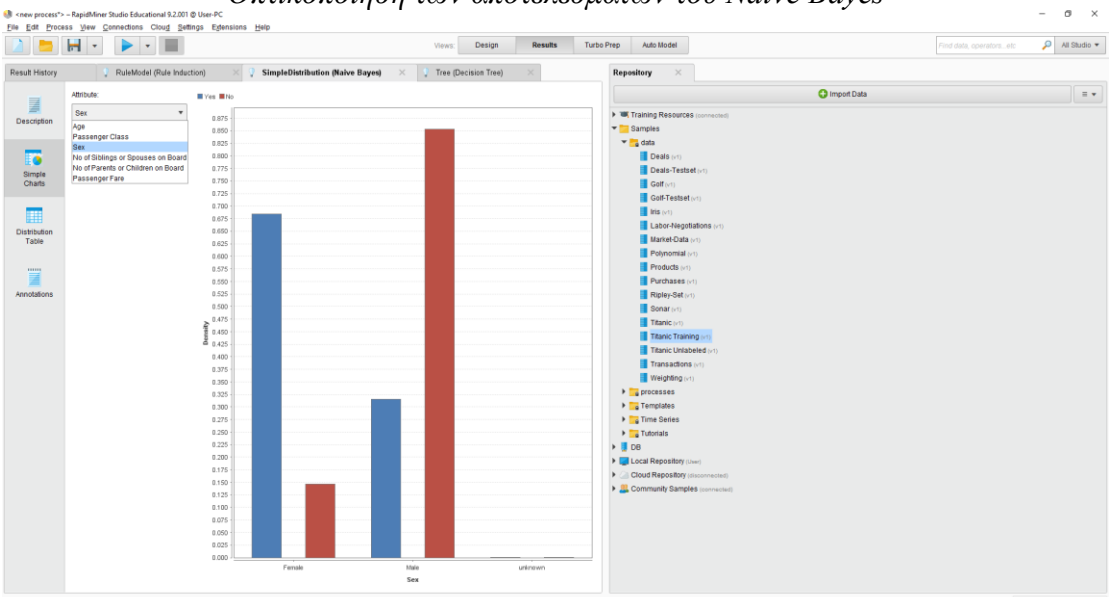
ΕΙΚΟΝΑ41
Αποτελέσματα των 3 Operators



Στα αποτελέσματα, δηλαδή στο Results view φαίνεται ξεκάθαρα στο δέντρο απόφασης ότι το μέγεθος της οικογένειας έχει περισσότερη σημασία απ’ ότι η τάξη του επιβάτη όσον αφορά το γυναικείο φύλο. Ωστόσο το μονοπάτι μιας τέτοιας συμπεριφοράς δεν ισχύει και για του άντρες. Γενικότερα οι άντρες είχαν μικρότερη πιθανότητα να σωθούν. Αν μετακινηθούμε στα αποτελέσματα του Naive Bayes και κάνουμε κλικ στο ‘Simple Charts’ βλέπουμε κάποιο γράφημα. Για να δούμε και τα υπόλοιπα, πατάμε στο πλαίσιο που βρίσκεται αριστερά κάτω από την λέξη ‘Attribute’. Παρόλο που αυτός ο αλγόριθμος συνήθως δεν έχει τόσο υψηλή ακρίβεια, εδώ μας παρέχει μία καλή διάκριση των δεδομένων και είναι πιο εύκολο να τα παρατηρήσουμε. Αν κοιτάξουμε με μεγαλύτερη προσοχή τα αποτελέσματα των τριών αλγορίθμων, γίνεται φανερό ότι και στα τρία το ‘attribute’ για την διάκριση είναι το ‘Sex’ που σημαίνει ο΄τι παρέχει το μεγαλύτερο information gain. Είχαμε δείξει πως υπολογίζεται το information gain σε προηγούμενο κεφάλαιο της εργασίας. Ας υπενθυμίσουμε ότι χάρη σε αυτόν τον υπολογισμό ο αλγόριθμος εντοπίζει το χαρακτηριστικό του σετ το οποίο περιλαμβάνει την περισσότερη πληροφορία. Στην ΕΙΚΟΝΑ42 αναπαρίστανται σε μορφή ραβδογράμματος, με μπλε χρώμα οι τιμές ‘Yes’, ενώ με κόκκινο οι τιμές ‘No’ τόσο για ανδρικό όσο και για το γυναικείο φύλο. Είναι σαφέστατα αντιληπτό ότι υπάρχει μεγάλη διαφορά ανάμεσα τους και με αυτό το αποτέλεσμα ολοκληρώνουμε το παρόν tutorial.

ΕΙΚΟΝΑ42

Οπτικοποίηση των αποτελεσμάτων του Naive Bayes

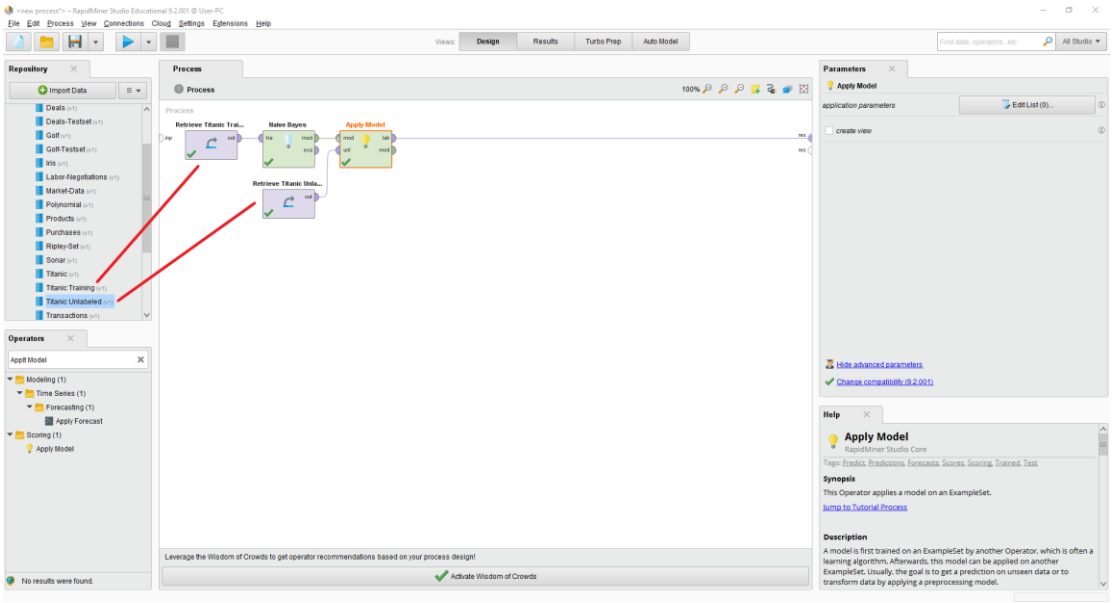


Στο προηγούμενο tutorial παρουσιάσαμε τον τρόπο με τον οποίο ένα μοντέλο πρόβλεψης μπορεί να αποκαλύψει πληροφορίες για τα δεδομένα. Σε αυτό το tutorial θα δείξουμε πως είναι δυνατό να χρησιμοποιήσουμε αυτές τις πληροφορίες του σετ εκπαίδευσης για να προβλέψουμε ορισμένα συμβάντα. Πιο συγκεκριμένα θα αξιοποιήσουμε τον 'Naive Bayes' Operator για να προβλέψουμε το 'Survived class' για τον κάθε επιβάτη και να βρούμε το ποσοστό 'confindence' κάθε γραμμής του πίνακα. Με τον όρο 'confidence' αναφερόμαστε στην βεβαιότητα ή αβεβαιότητα ενός αποτελέσματος σε κλίμακα από 0 έως 1.

Τα βήματα που θα χρειαστεί να κάνουμε εδώ, είναι η τοποθέτηση του Titanic Training μέσα στο Process και η σύνδεση του με τον 'Naive Bayes' ο οποίος δημιουργεί τις προβλέψεις για ένα καινούργιο σύνολο δεδομένων, το οποίο όμως δεν διαθέτει label στην στήλη 'Survived' που θέλουμε να προβλέψουμε. Εκτός από τον Operator χρειαζόμαστε φυσικά το test set και τον 'Apply Model' Operator. Το σετ αυτό στην περίπτωση μας είναι το Titanic Unlabeled και ονομάζεται έτσι γιατί η στήλη Survived δεν έχει επισημανθεί με label. Θα ενώσουμε τους δύο Operators με τις πύλες 'mod', ενώ την πύλη 'out' του Titanic Unlabeled με την πύλη 'unl' του 'Apply Model' Operator. Για να ολοκληρώσουμε συνδέουμε τον Operator με την πύλη 'res' και εκτελούμε το πείραμα κανονικά. Στην επόμενη σελίδα, η ΕΙΚΟΝΑ43 παρουσιάζει όλους τους Operators και τις μεταξύ τους απαραίτητες συνδέσεις.

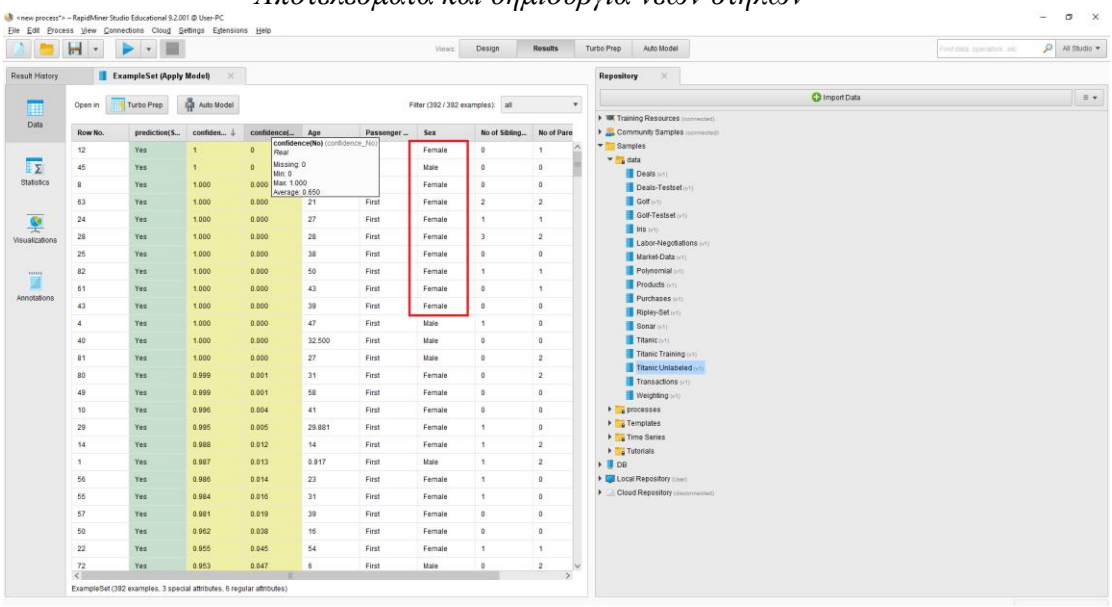
ΕΙΚΟΝΑ43

Χρήση ενός νέου 'test set' για την εξαγωγή προβλέψεων



ΕΙΚΟΝΑ44

Αποτελέσματα και δημιουργία νέων στηλών

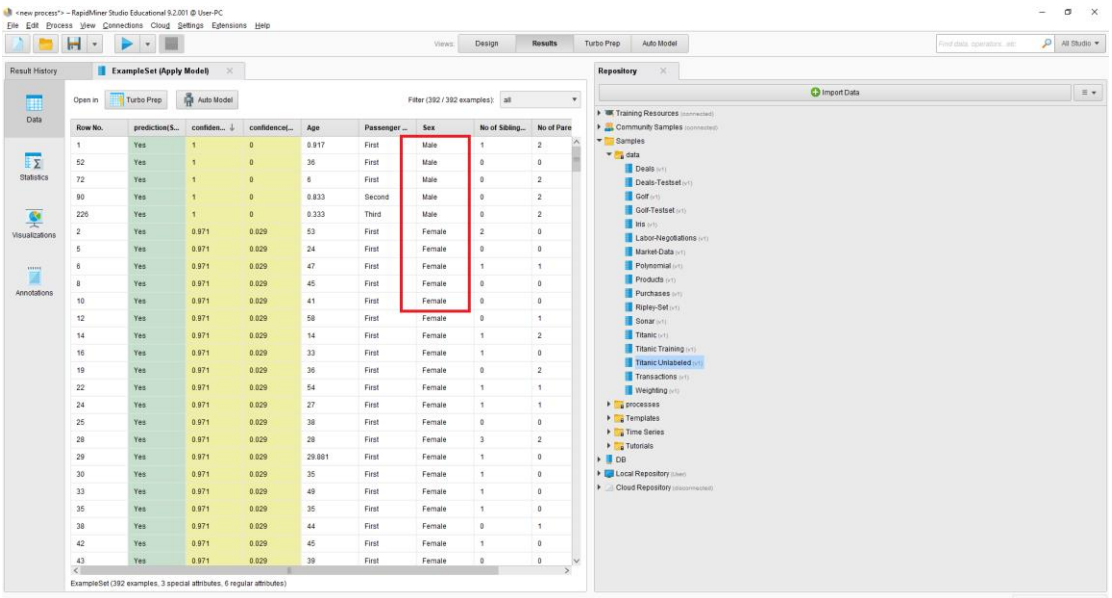


Έχουμε πλέον μεταφερθεί στην όψη των αποτελεσμάτων, όπου σχηματίστηκαν 3 στήλες με τα ονόματα prediction(Survived), confidence(Yes), confidence(No) στην ΕΙΚΟΝΑ44. Η πρώτη στήλη αναπαριστά τις προβλέψεις με τιμές Yes|No για τον κάθε επιβάτη αντίστοιχα σε κάθε γραμμή του πίνακα. Η δεύτερη όπως και η τρίτη στήλη δείχνουν την βεβαιότητα για το αποτέλεσμα σε κλίμακα 0 έως 1. Αν κάνουμε κλικ στο όνομα της στήλης confidence(Yes) για ταξινομηθούν τα αποτελέσματα σε φθίνουσα σειρά, θα διαπιστώσουμε ότι 9 από τα 10 άτομα για τις πρώτες σειρές του πίνακα που είχαν 'Yes' ήταν οι γυναίκες, αυτό φαίνεται στο κόκκινο πλαίσιο. Αν δοκιμάσουμε να εκτελέσουμε το πείραμα ξανά, όμως να αντικαταστήσουμε τον 'Naive Bayes' με τον

‘Decision Tree’ θα παρατηρήσουμε ότι υπάρχει διαφορά σε αυτές τις πρώτες 10 γραμμές του πίνακα.

ΕΙΚΟΝΑ45

Διαφορά των αποτελεσμάτων με την εκτέλεση του Decision Tree



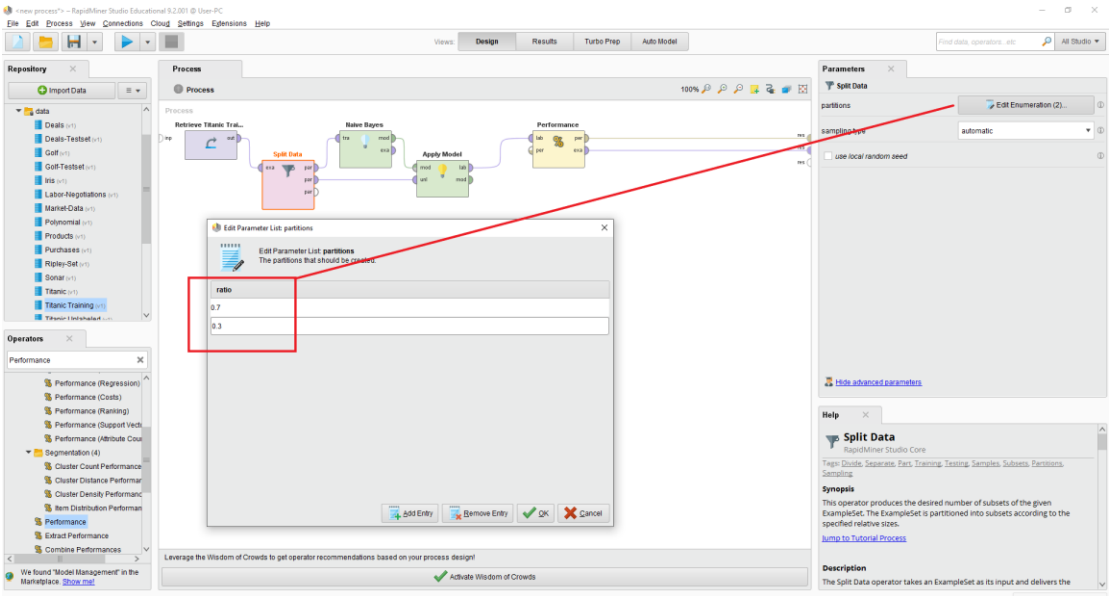
Σε αυτό το tutorial, θα εξετάσουμε πόσο καλά ανταποκρίνεται το μοντέλο, το οποίο πρόκειται να φτιάξουμε. Θα δούμε αν αυτό είναι ιδανικό για μελλοντικά σενάρια και πόσο υψηλή είναι η ακρίβεια των αποτελεσμάτων του.

Αρχικά, εισάγουμε το ‘Titanic Training’ σετ μέσα στο τμήμα του Process και έπειτα αναζητούμε τον ‘Split Data’ Operator. Αυτός ο Operator παίρνει το σετ και το διαιρεί σε δύο κομμάτια. Σε αυτή την περίπτωση θέλουμε να διαιρέσουμε το σετ μας σε 70% και σε 30%. Για να το κάνουμε αυτό, θα πρέπει πρώτα να συνδέσουμε τον ‘Split Data’ Operator με το ‘Titanic Training’ σετ. Στη συνέχεια, θα κάνουμε κλικ στο ‘Edit Enumeration(0)’, το οποίο βρίσκεται στο τμήμα Paramaters του ‘Split Data’. Στο παράθυρο που ανοίγει, πατάμε το κουμπί ‘Add Entry’ δύο φορές και γράφουμε 0.7 στην πρώτη γραμμή και 0.3 στην δεύτερη. Με αυτόν τον τρόπο, το 70% του σετ θα χρησιμοποιηθεί ως ‘training set’ ενώ το 30% ως ‘test set’. Ο λόγος που κάνουμε αυτή την διαίρεση είναι ότι το 70-30% δίνει συχνά σε τέτοια πειράματα τα καλύτερα αποτελέσματα.

Πέρα από τα παραπάνω, θα πρέπει να προσθέσουμε τον αλγόριθμο ‘Naive Bayes’ και να τον ενώσουμε με τον ‘Split Data’ Operator. Επιπλέον, τοποθετούμε τον ‘Apply Model’ Operator και τον συνδέουμε με τον ‘Naive Bayes’ μέσω των πυλών ‘mod’, ενώ με τον ‘Split Data’ μέσω της πύλης ‘unl’. Στο τέλος, εισάγουμε τον ‘Performance’ Operator, τον οποίο συνδέουμε με τον ‘Apply Model’ από αριστερά μέσω της πύλης ‘mod’, ενώ από δεξιά με δύο πύλες εξόδου ‘res’ και εκτελούμε το μοντέλο. Στην ΕΙΚΟΝΑ46, στην επόμενη σελίδα, αναπαρίστανται η παραπάνω διαδικασία σύνδεσης.

ΕΙΚΟΝΑ46

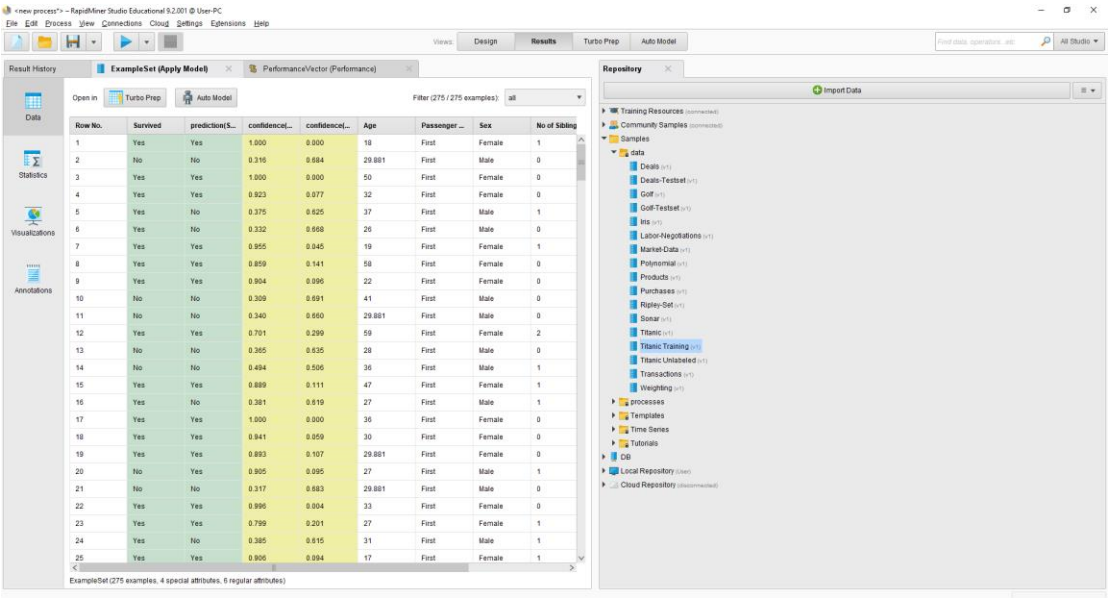
Χρήση 70-30% για training και testing sets



Αφού εκτελέσαμε το μοντέλο, στο Results View μπορούμε να διακρίνουμε δύο αποτελέσματα με τα ονόματα ‘ExampleSet(Apply Model)’ και ‘Performance Vector’ αντίστοιχα. Στο πρώτο από αυτά, φαίνονται συνολικά 275 παραδείγματα, τα οποία ανήκουν στο ‘test set’. Μέσα στην στήλη ‘Survived’ έχουμε την πραγματική τιμή, ενώ μέσα στην στήλη ‘Prediction’, έχουμε την τιμή πρόβλεψης ΕΙΚΟΝΑ47. Αν μετακινηθούμε στο δεύτερο αποτέλεσμα ΕΙΚΟΝΑ48, παρατηρούμε έναν πίνακα ‘confusion matrix’ με ακρίβεια ‘accuracy: 80.36%’. Η ακρίβεια υπολογίζεται από το άθροισμα της διαγωνίου του πίνακα, εδώ είναι (76 + 145), προς το άθροισμα όλων των τιμών του πίνακα, εδώ είναι (76 + 25 + 29 + 145). Δηλαδή, (76 + 145) / (76 + 25 + 29 + 145) = 0.8036 = 80.36%.

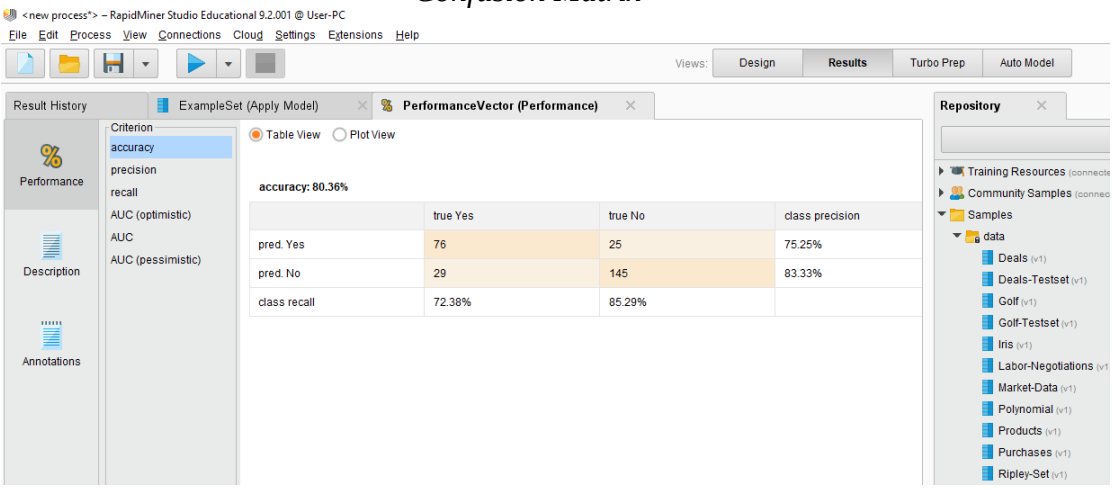
ΕΙΚΟΝΑ47

Πραγματική τιμή και τιμή πρόβλεψης



ΕΙΚΟΝΑ48

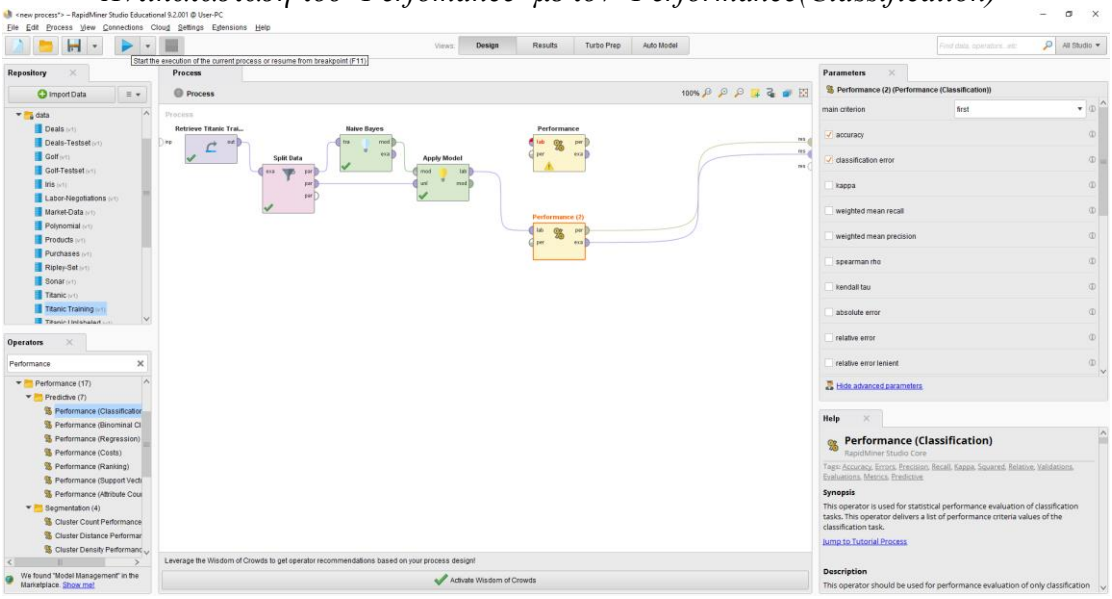
Confusion Matrix



Ο πίνακας φανερώνει για πόσες τιμές έχει γίνει σωστή πρόβλεψη και για πόσες λανθασμένη. Ένας τρόπος για να το υπολογίσουμε αυτό είναι να προσθέσουμε τον αριθμό 29 (true Yes, pred No) με τον αριθμό 25 (true No, pred Yes). Δηλαδή, wrong values = 29 + 25 = 54 λάθος από τα 275 παραδείγματα. Ένας πιο απλός και καλύτερος τρόπος για να δούμε τα λάθος αποτελέσματα, είναι να χρησιμοποιήσουμε το φίλτρο 'wrong_predictions', το οποίο βρίσκεται στο άνω δεξί μέρος των αποτελεσμάτων, ΕΙΚΟΝΑ47 Filter(275/275 Examples) και το αλλάζουμε. Τέλος αν θέλουμε να δούμε πέρα από την ακρίβεια, το 'classification error' και το 'root mean squared error', μπορεί να αντικαταστήσουμε τον 'Performance' Operator με τον 'Performance(Classification)' και στο τμήμα των παραμέτρων, να επιλέξουμε τα κουτάκια 'accuracy', 'classification error', 'root mean squared error'. Η ΕΙΚΟΝΑ49 παρουσιάζει αυτή τη διαδικασία. Αν εκτελέσουμε ξανά το μοντέλο με τον νέο Operator θα δούμε ότι οι προσθήκες που κάναμε, εμφανίζονται στα αποτελέσματα του 'Confusion Matrix'.

ΕΙΚΟΝΑ49

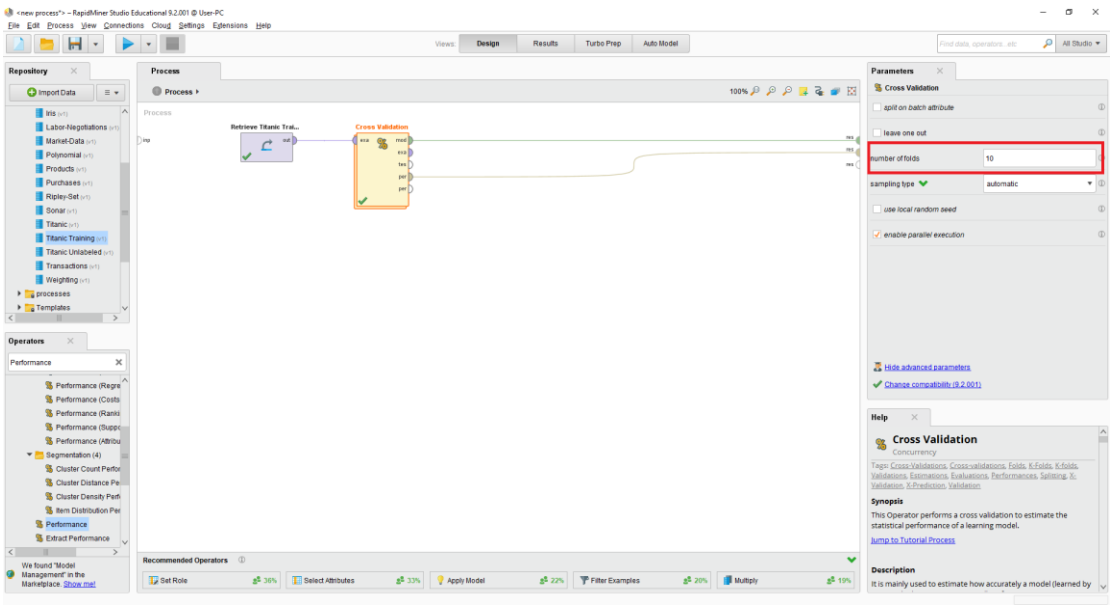
Αντικατάσταση του 'Perfomance' με τον 'Performance(Classification)'



Στο προηγούμενο tutorial είχαμε δει πως μπορούμε να κάνουμε μία εκτίμηση για την ακρίβεια του μοντέλου μας, μοιράζοντας τα δεδομένα σε ‘training’ και ‘testing’ σύνολα. Όμως, σε κάποια σύνολα δεδομένων, τα οποία χωρίζονται με αυτόν τον τρόπο, μπορεί να υπάρχουν σημαντικές διαφορές στα κομμάτια και αυτές οι διαφορές να επηρεάσουν την τελική ακρίβεια με ανεπιθύμητο τρόπο. Επομένως, σε αυτό το tutorial θα χρησιμοποιήσουμε την μέθοδο ‘Cross Validation’ ώστε να εξασφαλίσουμε ότι κάθε κομμάτι των δεδομένων μας, αξιοποιείται και ως ‘training’ και ως ‘testing’. Αυτή η τεχνική ‘Cross Validation’ διαιρεί το σύνολο σε ίσα μέρη και περιστρέφεται γύρω από το κάθε μέρος, όπου κάθε φορά χειρίζεται το ένα κομμάτι ως ‘test set’ και όλα τα υπόλοιπα ως ‘training set’. Στο τέλος ο μέσος όρος από όλες τις ακρίβειες που έχουν υπολογιστεί, δίνεται ως το τελικό αποτέλεσμα. Αυτή η μέθοδος, είναι καλό να καθιερώνεται από τον χρήστη του λογισμικού γιατί είναι ιδανική για τον υπολογισμό της ακρίβειας.

Αρχικά, θα τοποθετήσουμε το ‘Titanic Training’ μέσα στην επιφάνεια του τμήματος Process. Στη συνέχεια, θα συνδέσουμε τον ‘Cross Validation’ Operator με το σετ μας. Αυτός ο Operator χρειάζεται ένα ‘labeled’ σετ προκειμένου να λειτουργήσει κανονικά, γι αυτό και χρησιμοποιούμε το ‘Titanic Training’ και όχι κάποιο άλλο. Φυσικά θα μπορούσαμε να ορίσουμε ‘label’ στο σετ μας με τον τρόπο που είχαμε παρουσιάσει σε ένα από τα προηγούμενα tutorials. Το ‘Cross Validation’, στην ‘by default’ χρήση του, θα μοιράσει το σύνολο δεδομένων σε δέκα ίσα μέρη και αυτό φαίνεται στις παραμέτρους του στο δεξί μέρος ‘number of folds’ όπως στην ΕΙΚΟΝΑ50.

ΕΙΚΟΝΑ50
Cross Validation 10-folds

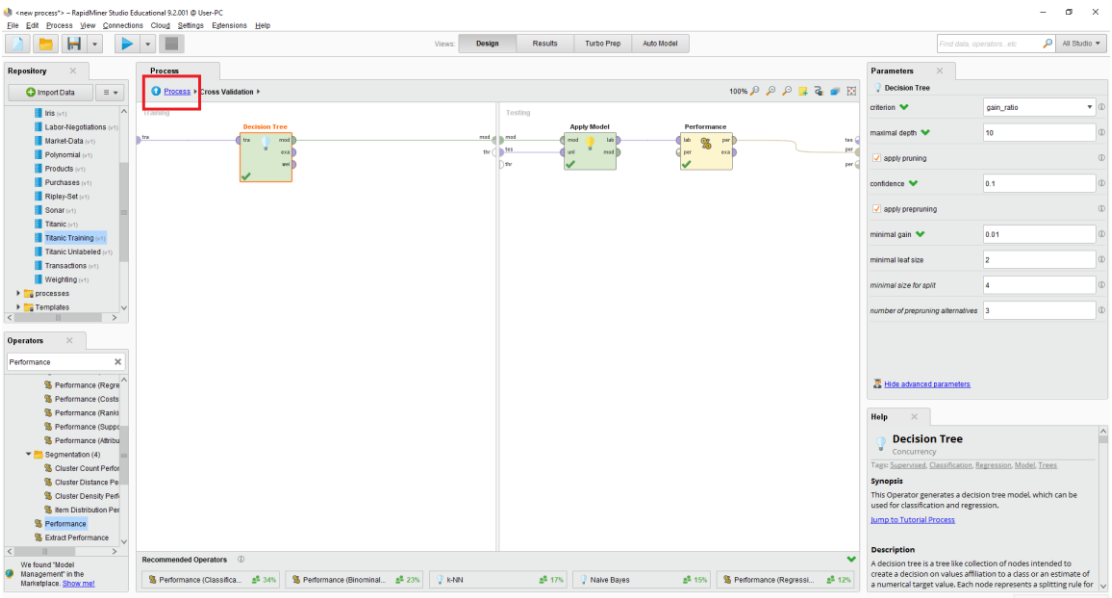


Κάνοντας διπλό κλικ πάνω στον ‘Cross Validation’ Operator, μεταφερόμαστε σε μία επιφάνεια που ονομάζεται ‘sub-process’. Αυτή η επιφάνεια ανήκει στον συγκεκριμένο Operator, όπου το αριστερό τμήμα έρχεται με το όνομα ‘Training’, ενώ το δεξί τμήμα είναι το ‘Testing’. Μπορούμε και εδώ να εργαστούμε κανονικά όπως και στο κεντρικό τμήμα Process. Έτσι, θα προσθέσουμε στην αριστερή πλευρά τον Decision Tree Operator και θα τον συνδέσουμε με την πύλη εισόδου ‘tra’ όπως

επίσης και με την πύλη εξόδου ‘mod’. Στην δεξιά πλευρά, θα τοποθετήσουμε τον ‘Apply Model’ Operator και θα τον συνδέσουμε αριστερά με τις πύλες ‘mod’ προς ‘mod’ και ‘tes’ προς ‘unl’. Στη συνέχεια, θα εισάγουμε τον ‘Performance’ Operator, τον οποίο ενώνουμε μέσω της πύλης ‘lab’ με τον προηγούμενο, ενώ την πύλη ‘per’ αυτού με την πύλη εξόδου ‘per’. Για να ολοκληρώσουμε τις συνδέσεις, θα επιστρέψουμε στην αρχική επιφάνεια του Process, κάνοντας κλικ στο όνομα ‘Process’ που εμφανίζεται με γαλάζια γράμματα. Όλές οι εσωτερικές διαδικασίες του ‘Cross Validation’ φαίνονται παρακάτω στην ΕΙΚΟΝΑ51. Με κόκκινο περίγραμμα τονίζεται το πώς επιστρέφουμε στην αρχική επιφάνεια του Process. Έπειτα, συνδέουμε τις πύλες του ‘Cross Validation’ Operator με τις πύλες εξόδου όπως στην ΕΙΚΟΝΑ50 και εκτελούμε κανονικά το μοντέλο.

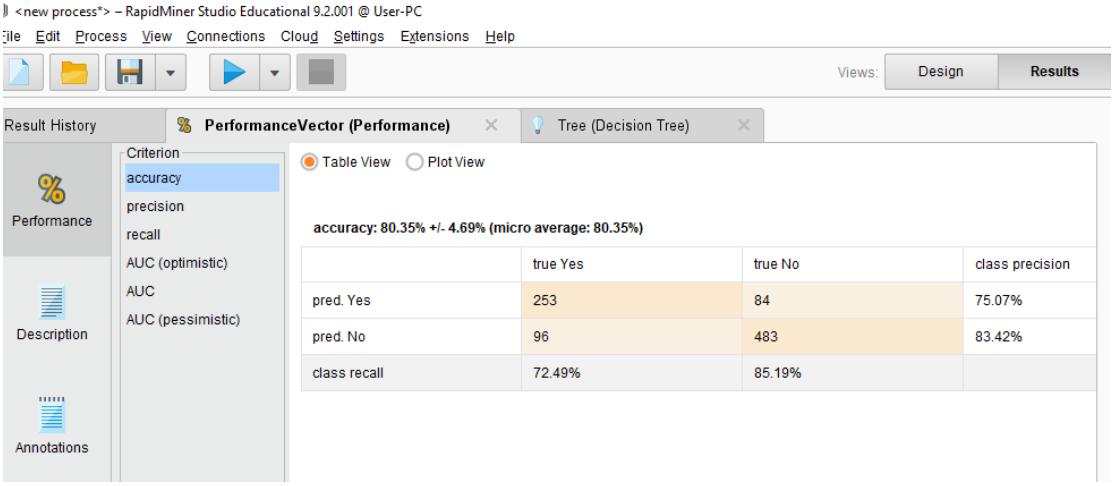
ΕΙΚΟΝΑ51

Εσωτερικές διαδικασίες στον Cross Validation Operator



ΕΙΚΟΝΑ52

Ακρίβεια και απόκλιση



	true Yes	true No	class precision
pred. Yes	253	84	75.07%
pred. No	96	483	83.42%
class recall	72.49%	85.19%	

Όντας πλέον στην επιφάνεια των αποτελεσμάτων, παρατηρούμε ότι η ακρίβεια είναι accuracy:80.35% +/- 4.69% ΕΙΚΟΝΑ52. Το (+/-) αναφέρεται στην απόκλιση που μπορεί να έχει η ακρίβεια ενός μοντέλου. Όσο μικρότερη είναι η απόκλιση τόσο το καλύτερο για εμάς. Τέλος για να κάνουμε πιο ενδιαφέρον αυτό το tutorial, ας δοκιμάσουμε να εκτελέσουμε το πείραμα και με τους Operators 'Naive Bayes' και 'Rule Induction' ώστε να δούμε ποιος τελικά από τους τρεις πετυχαίνει την υψηλότερη ακρίβεια. Για να γίνει αυτό θα αντικαταστήσουμε τον Decision Tree του 'sub-process' με έναν της επιλογής μας και θα εκτελέσουμε ξανά. Στον παρακάτω πίνακα έχουν καταγραφεί τα αποτελέσματα των τριών Operators.

ΠΙΝΑΚΑΣ1 (*Αλγόριθμος και η ακρίβεια του*)

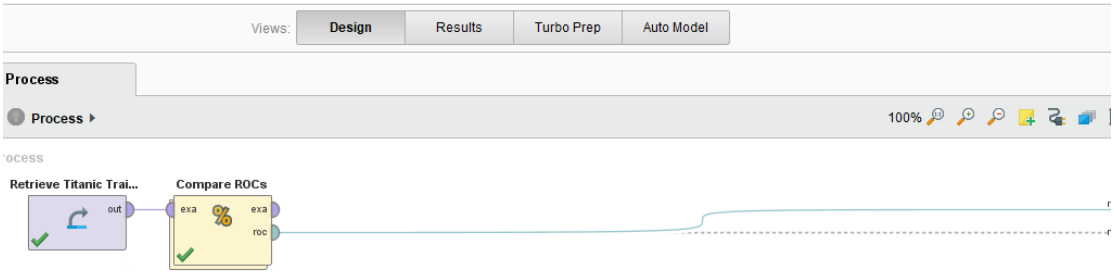
OPERATOR	ACCURACY
Decision Tree	80.35 +/- 4.69%
Rule Induction	78.93% +/- 3.33%
Naive Bayes	78.17% +/- 4.83%

Μεταβαίνουμε στο τελευταίο tutorial για το λογισμικό rapidminer, αφού έχουμε καλύψει τόσο τις απλές και πολύ βασικές λειτουργίες του όσο και τις πιο σύνθετες, που είναι απαραίτητες και μεγάλης σημασίας για έναν αναλυτή. Σε αυτό το tutorial, θα προσθέσουμε μία τελική λεπτομέρεια, η οποία μας είναι χρήσιμη όταν θέλουμε να συγκρίνουμε τους αλγορίθμους και να εντοπίζουμε τον πιο αποδοτικό χωρίς να χρειάζεται να εκτελούμε το πείραμα πολλές φορές αλλά μόνο μία.

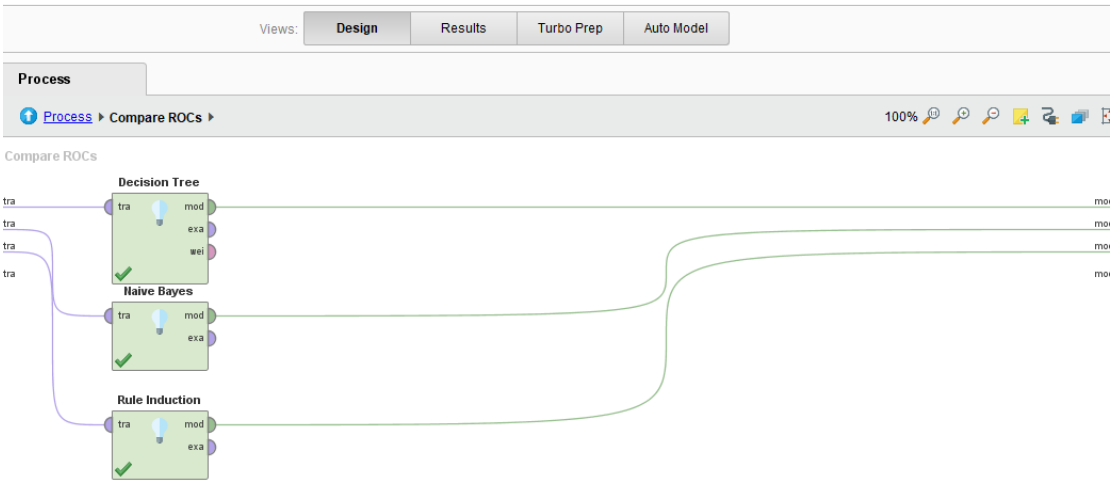
Ο Operator που μας παρέχει αυτή τη δυνατότητα ονομάζεται 'Compare ROCs' όπου το ROC = Receiver Operating Characteristics. Ουσιαστικά με την εκτέλεση του πειράματος σχηματίζεται μία καμπύλη για κάθε αλγόριθμο στο Results view. Αν αυτή είναι ευθεία γραμμή κάθετη στον άξονα x τότε σημαίνει ότι το μοντέλο είναι τέλειο. Σε όλες τις άλλες περιπτώσεις, όσο πιο κάθετη κατά κάποιο τρόπο είναι η καμπύλη και πιο κοντά προς τον άξονα y, τόσο καλύτερο είναι το μοντέλο. Στη συνέχεια θα προβάλουμε με εικόνα την παραπάνω εξήγηση για να γίνει πιο κατανοητή η λειτουργία του ROC.

Ως πρώτο βήμα, θα τοποθετήσουμε το 'Titanic Training' σετ και τον 'Compare ROCs' Operator μέσα στην επιφάνεια του τμήματος Process. Τα ενώνουμε μεταξύ τους και μετά κάνουμε διπλό κλικ στον 'Compare ROCs' ώστε να μετακινηθούμε στο 'sub-process' του. Μέσα σε αυτό, θα εισάγουμε τους τρεις αλγορίθμους του ΠΙΝΑΚΑΣ1 και θα συνδέσουμε τον καθένα από αριστερά με μία πύλη 'tra' και από δεξιά με μία πύλη 'mod'. Ωστόσο, πρίν εκτελέσουμε, θα πρέπει να συνδέσουμε την πύλη 'roc' του 'Compare ROCs' με την πύλη εξόδου 'res'. Στις επόμενες δύο εικόνες φαίνονται οι συνδέσεις του Process και του sub-process αντίστοιχα.

ΕΙΚΟΝΑ53
Χρήση του Compare ROCs Operator



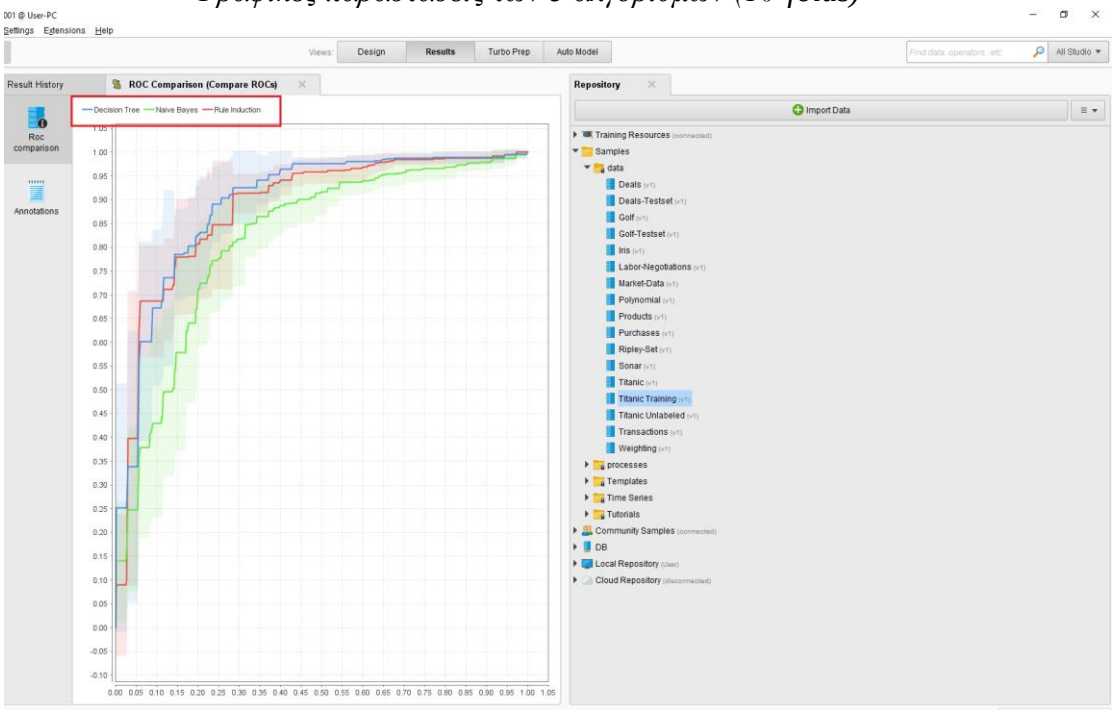
ΕΙΚΟΝΑ54
Συνδέσεις στο Sub-process του Compare Rocs



Στα αποτελέσματα ΕΙΚΟΝΑ55, εμφανίζονται τρεις καμπύλες, με μπλε, πράσινο και κόκκινο χρώμα για τους αλγορίθμους Decision Tree, Naive Bayes και Rule Induction αντίστοιχα. Οι καμπύλες αυτές αναπαρίστανται με πολλές κάθετες γραμμές και σε μία κοντινή περιοχή, επεκτείνεται το χρώμα της κάθε καμπύλης με πιο αχνό χρωματισμό το οποίο αφορά την απόκλιση (+/-) που είχαμε εξηγήσει νωρίτερα. Αν κάποιος αλγόριθμος είχε σχηματίσει μία κάθετη γραμμή στον άξονα x τότε θα σήμαινε ότι έχει την τέλεια ακρίβεια, δηλαδή 100%, όμως εδώ δεν γίνεται αυτό. Ανάμεσα στους τρεις, φαίνεται ότι ο Decision Tree είχε την καλύτερη απόδοση, αφού η καμπύλη βρίσκεται πιο ψηλά από τις άλλες δύο και πιο κοντά στον άξονα y. Αυτό είναι και το αναμενόμενο αποτέλεσμα, αφού το ίδιο πείραμα είχαμε δοκιμάσει προηγουμένως και είχαμε αποθηκεύσει τις τιμές στον ΠΙΝΑΚΑΣ1. Εδώ απεικονίζεται με γραφικές παραστάσεις και μόνο με μία εκτέλεση. Σε περίπτωση που αλλάξουμε τις παραμέτρους του ‘Compare ROCs’ Operator και θέσουμε διαφορετικό k-fold για training και για testing, για παράδειγμα 5-folds, τότε θα έχουμε διαφορετικό αποτέλεσμα ΕΙΚΟΝΑ56.

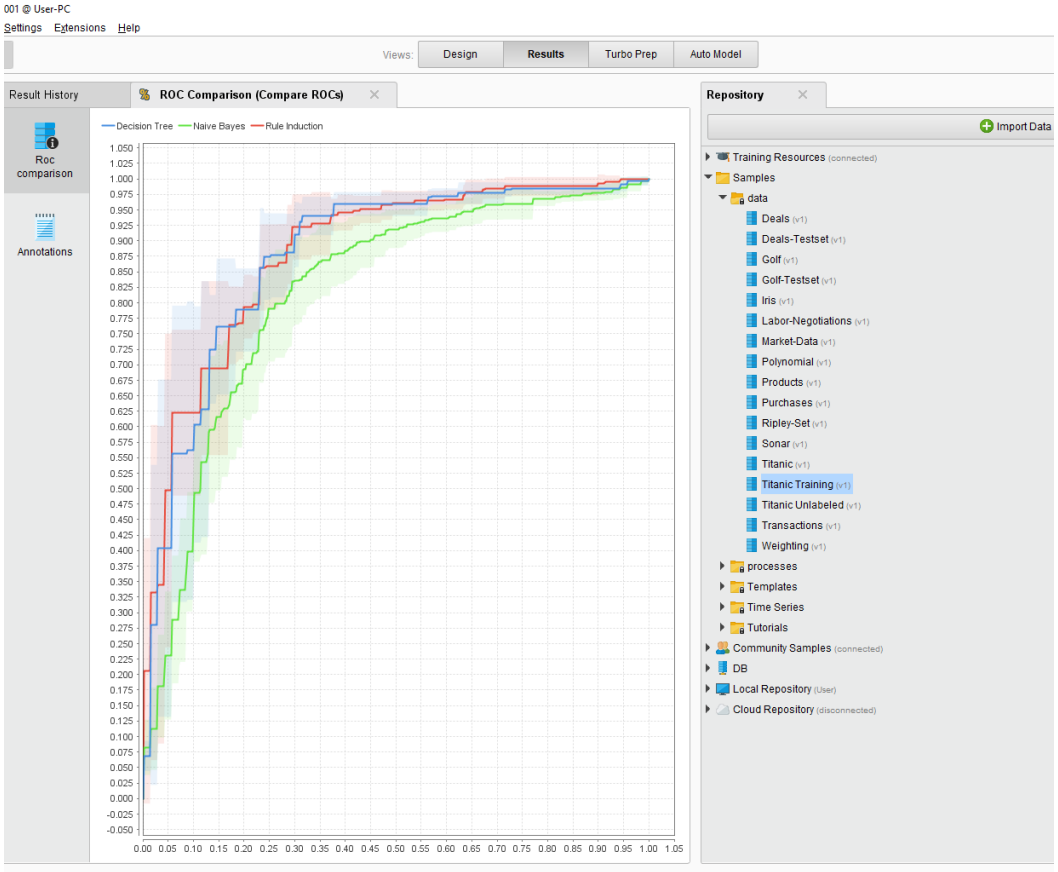
ΕΙΚΟΝΑ55

Γραφικές παραστάσεις των 3 αλγορίθμων (10-folds)



ΕΙΚΟΝΑ56

Γραφικές παραστάσεις των 3 αλγορίθμων (5-folds)

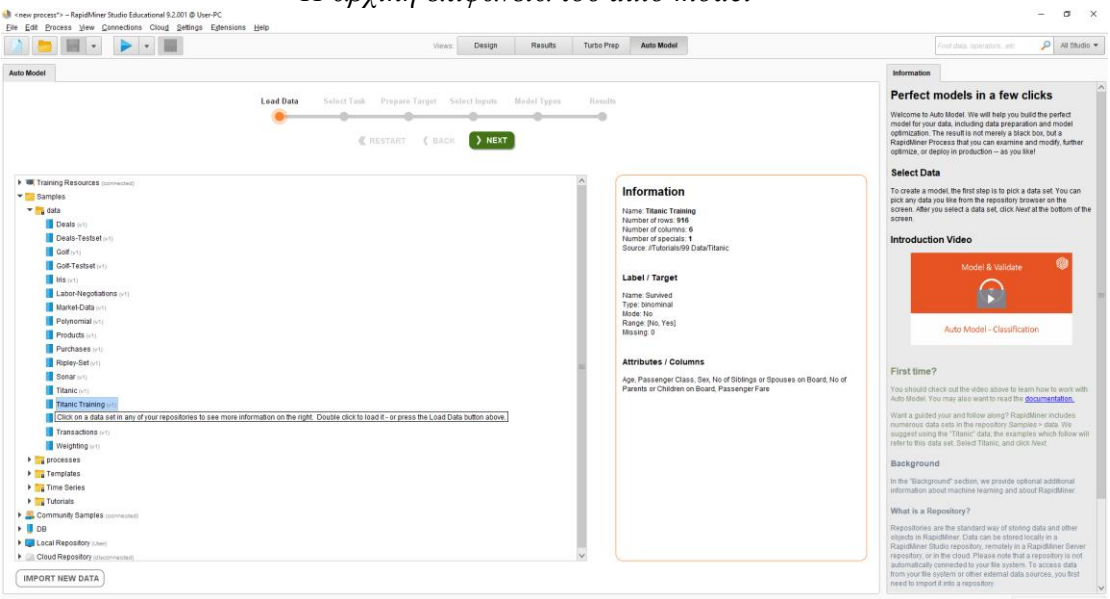


10.5 Rapidminer λειτουργία auto model

Το rapidminer auto model αποτελεί μια επιπρόσθετη λειτουργία του λογισμικού. Αυτή η λειτουργία βρίσκεται στο ίδιο σημείο με τις επιλογές Process, Results κ.α. Το auto model έχει δημιουργηθεί για να διευκολύνει τον χρήστη, μέσω των αυτοματοποιημένων διαδικασιών που προσφέρει και κάνει την ανάλυση πιο ευχάριστη και εύκολη μέσα από λίγα μόνο κλικ. Συγκεκριμένα στην EIKONA57, φαίνεται η αρχική επιφάνεια του auto model. Στην δεξιά πλευρά εξηγείται αναλυτικά, τι πρέπει να κάνει ο χρήστης για να προχωρήσει στο επόμενο βήμα. Στο κέντρο υπάρχει μία γραμμή με 6 κουκίδες, η οποία παρουσιάζει όλα τα στάδια, αρχίζοντας από την εισαγωγή ενός σετ δεδομένων μέχρι και τα αποτελέσματα μετά την εκτέλεση. Το μόνο που χρειάζεται σε αυτό το σημείο είναι η επιλογή ενός σετ και στη συνέχεια να πατήσει ο χρήστης το START.

EIKONA57

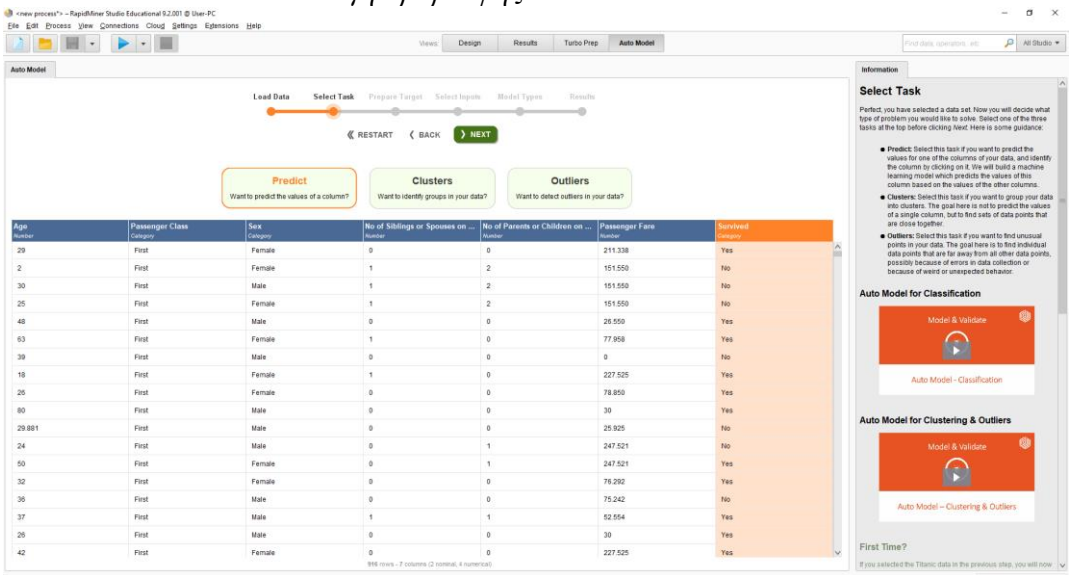
Η αρχική επιφάνεια του auto model



Ως επόμενο βήμα, ο χρήστης πρέπει να προσδιορίσει αν θέλει να χρησιμοποιήσει το σετ για πρόβλεψη, ομαδοποίηση ή για τον εντοπισμό ‘missing values’ αυτού. Έστω ότι κάνουμε κλικ στην πρώτη επιλογή ‘Predict’ EIKONA58 και ως στήλη για την οποία θέλουμε να γίνει η πρόβλεψη την ‘Survived’. Έπειτα θα ολοκληρώσουμε και την υπόλοιπη σειρά των βημάτων με πολύ απλό τρόπο μέχρι να φτάσουμε στο σημείο (προ-τελευταία κουκίδα) όπου μπορούμε να κρατήσουμε όσους αλγορίθμους θέλουμε για την εκτέλεση της πρόβλεψης, από αυτούς που μας προτείνει το auto model. Στην EIKONA59 παρουσιάζεται η εκτέλεση όλων των αλγορίθμων, υποδεικνύοντας την ακρίβεια του καθενός, την απόκλιση (+/-), καθώς και τον χρόνο εκτέλεσης. Στον οριζόντιο άξονα φαίνονται όλοι οι αλγόριθμοι ενώ ο κάθετος άξονας έχει τα ποσοστά από 0 έως 100%.

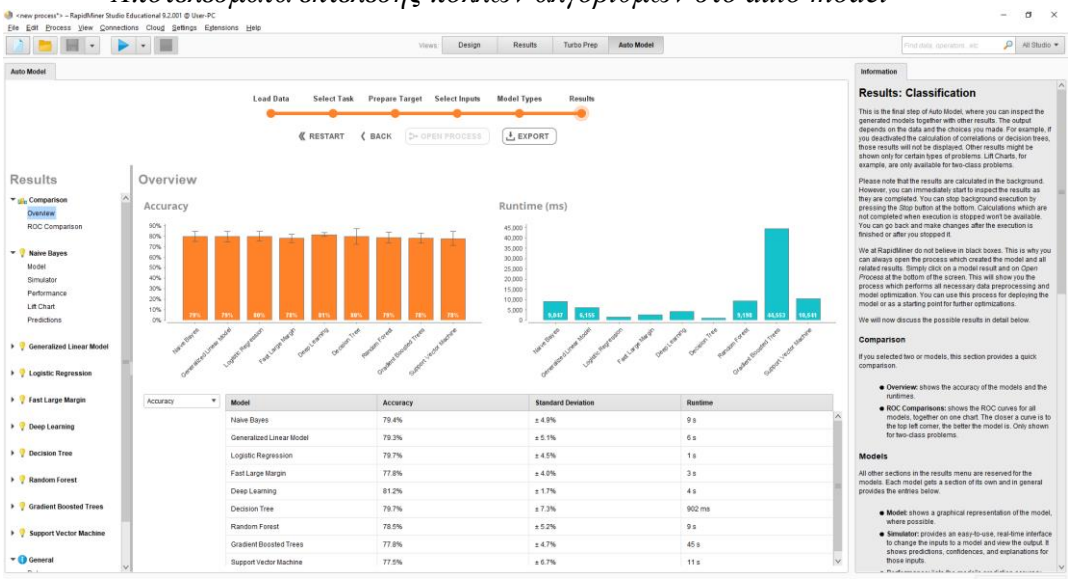
ΕΙΚΟΝΑ58

Επιλογή πρόβλεψης στο auto model



ΕΙΚΟΝΑ59

Αποτελέσματα εκτέλεσης πολλών αλγορίθμων στο auto model



10.6 Rapidminer turbo prep

Όπως το auto model έτσι και το turbo prep έχει δημιουργηθεί για να διευκολύνει τον χρήστη με τα σύνολα δεδομένων. Η διαφορά σε αυτή τη λειτουργία βρίσκεται στην επεξεργασία του σετ. Με το turbo prep μπορούμε για παράδειγμα να κάνουμε καθαρισμό (cleanse) στα δεδομένα, δηλαδή να αφαιρέσουμε στήλες οι τιμές που δεν θέλουμε. Επιπλέον, δίνεται η δυνατότητα να επεξεργαστούμε μία μόνο στήλη και να την τροποποιήσουμε ή απλά να την κάνουμε μετονομασία. Τέλος μπορούμε να προσθέσουμε με λίγα μόνο κλίκ νέες στήλες στο σύνολο μας για τις οποίες καθορίζουμε σε τι μας εξυπηρετούν. Για παράδειγμα, μπορούμε να εισάγουμε μία στήλη που να είναι κάποιο άθροισμα τιμών ή κάποια διαίρεση ανάλογα κάθε φορά με τα δεδομένα. Έτσι ο χρήστης δεν χρειάζεται να ψάξει όλους αυτούς τους Operators είχαμε παρουσιάσει στα tutorial για να ετοιμάσει τα δεδομένα του.

10.7 Σύνοψη του λογισμικού rapidminer

Το λογισμικό εξόρυξης δεδομένων και μηχανικής μάθησης rapidminer αποτελεί μία σύγχρονο, ισχυρό και μεγάλου εύρους δυνατοτήτων εργαλείο. Ένα αρχικό πλεονέκτημά του, είναι η δωρεάν εγκατάσταση που παρέχει καθώς και η δυνατότητα ακαδημαϊκής χρήσης ή οποία περιλαμβάνει το πλήρες πακέτο. Η ιστοσελίδα είναι πολύ καλά οργανωμένη ώστε να παρέχει εύκολη πλοήγηση και να καλή διεπαφή με τον χρήστη. Ακόμα και κάποιος που δεν έχει χρησιμοποιήσει άλλο λογισμικό εξόρυξης δεδομένων, η δεν γνωρίζει γενικότερα την έννοια μπορεί μέσω του rapidminer να μάθει να χειρίζεται και να επεξεργάζεται σύνολα δεδομένων σε σύντομο χρονικό διάστημα. Η ιστοσελίδα περιλαμβάνει αρκετά βοηθητικά βίντεο που είναι εύχρηστα για έναν αρχάριο χρήστη, ενώ το εργαλείο που εγκαθίστανται στον υπολογιστή είναι ιδανικό για την υλοποίηση αυτών.

Το rapidminer έχει πολύ μεγάλη ποικιλία από έτοιμα σύνολα δεδομένων, συναρτήσεις, αλγορίθμους, φίλτρα κ.α, που συμβάλλουν στην εκτέλεση τόσο μικρών όσο και μεγάλων πειραμάτων. Αποτελείται από τα τμήματα Repository, Operators, Process, Parameters και Help, τα οποία αντίστοιχα αντιπροσωπεύουν τα δεδομένα, αλγορίθμους ή άλλες συναρτήσεις, την επιφάνεια που εργάζεται ο χρήστης, τις παραμέτρους, δηλαδή αλλαγή τιμών και τέλος την βοήθεια που εξηγεί το κάθε στοιχείο πιο αναλυτικά. Επιπλέον το λογισμικό είναι εφοδιασμένο με μία σειρά από βήματα που βοηθούν τον χρήστη να μάθει να το χρησιμοποιεί πλήρως.

Με το rapidminer μπορούμε να προετοιμάσουμε ένα σύνολο δεδομένων το οποίο είχαμε πάρει από κάπου αλλού και πιθανόν να μην ήταν καλά οργανωμένο ή να εμφανίζει πολλές απώλειες τιμών. Μέσω των κατάλληλων Operators, διαγράφουμε ή προσθέτουμε στοιχεία, ταξινομούμε τα ήδη υπάρχοντα στοιχεία, καλούμε συναρτήσεις που μας επιτρέπουν να κάνουμε πολλούς υπολογισμούς και έτσι το προηγουμένως ανοργάνωτο σετ να γίνεται πλέον ένα ιδανικό σετ που θα αξιοποιηθεί για μελλοντικές προβλέψεις.

Το γεγονός ότι στο rapidminer έχουμε τη δυνατότητα να εκτελέσουμε πολλούς αλγορίθμους ταυτόχρονα στο ίδιο πείραμα και να συγκρίνουμε τα αποτελέσματα τους, αυτό αποτελεί ακόμα ένα πλεονέκτημα του λογισμικού. Ο αναλυτής κερδίζει χρόνο και δεν χρειάζεται να επαναλαμβάνει το πείραμα περισσότερες φορές. Η γραφική αναπαράσταση των αποτελεσμάτων γίνεται με τέτοιο τρόπο που εύκολα μπορεί να την μελετήσει και να την κατανοήσει ο χρήστης, όπως επίσης και να συγκρίνει αυτά αποτελέσματα μέσω των διαφορετικών χρωμάτων με τα οποία εμφανίζονται είτε σαν ραβδογράμματα, είτε σαν γραφικές καμπύλες αναλόγως το πείραμα.

Η μεγάλη ποικιλία των Operators είναι πολύ σημαντική γιατί ο χρήστης μπορεί να αντιμετωπίσει πολλά ζητήματα σχετικά με τα δεδομένα. Αν και δεν είναι δυνατό στην αρχή αλλά και μετά από κάποιο διάστημα χρήσης του λογισμικού, να μάθουμε όλους τους Operators και τις συναρτήσεις, αυτό δεν αποτελεί πρόβλημα, γιατί το εργαλείο είναι εφοδιασμένο με τις επιλογές auto model και turbo prep. Αυτές μπορούν να μας καλύψουν πλήρως τόσο για την παραγωγή μοντέλων πρόβλεψης όσο και για την προετοιμασία των δεδομένων, την διόρθωση, την τροποποίηση, την συμπλήρωση και την ομαδοποίηση. Επομένως, το rapidminer είναι ένα πολύ ικανό και εύχρηστο λογισμικό για την εξόρυξη της πληροφορίας.

11 Σύγκριση weka - rapidminer

Σε αυτό το κεφάλαιο, θα πραγματοποιήσουμε σύγκριση των δύο λογισμικών εξόρυξης δεδομένων, έχοντας ως στόχο να εντοπίσουμε που υστερεί και υπερτερεί το ένα έναντι του άλλου. Η σύγκριση διαιρείται σε πέντε στάδια τα οποία είναι "ως προς το γραφικό περιβάλλον χρήστη (GUI), ως προς την ταχύτητα εκτέλεσης αλγορίθμων, ως προς την ευκολία διαχείρισης αρχείων, ως προς την αναπαράσταση των αποτελεσμάτων, ως προς το εύρος δυνατοτήτων. Κάθε ένα από τα στάδια παρουσιάζεται αναλυτικά στη συνέχεια.

11.1 Ως προς το γραφικό περιβάλλον χρήστη (GUI)

Όταν ένας χρήστης ανοίγει για πρώτη φορά το λογισμικό rapid miner, θα παρατηρήσει ένα σύνθετο "με την πρώτη ματιά" περιβάλλον, το οποίο όμως φαίνεται να είναι πολύ καλά δομημένο. Όπως έχουμε ήδη αναφέρει, το λογισμικό αποτελείται από πέντε βασικά τμήματα με ονόματα (repository, process, operators, parameters, help). Αν υποθέσουμε ότι ο χρήστης δεν έχει χρησιμοποιήσει ποτέ ξανά ένα λογισμικό εξόρυξης δεδομένων, το rapidminer έχει σχεδιαστεί με τέτοιο τρόπο ώστε είτε μέσω της ιστοσελίδας είτε μέσω της εφαρμογής που έχουμε εγκαταστήσει, να κατευθύνει τον χρήστη. Εξηγώντας του με σύντομο τρόπο όλα τα βασικά μέρη του λογισμικού και φυσικά το πώς μπορεί να χρησιμοποιήσει έτοιμα σύνολα δεδομένων που βρίσκονται μέσα σε αυτό.

Σε αντίθεση με τα παραπάνω έρχεται το γραφικό περιβάλλον του λογισμικού weka. Ανοίγοντας το, δεν θα μεταφερθούμε άμεσα στον χώρο, όπου πρόκειται να ασχοληθούμε με την ανάλυση δεδομένων, αλλά θα δούμε ένα αρχικό GUI με 5 επιλογές, οι οποίες είναι (Explorer, Experimenter, Knowledge Flow, Workbench, Simple CLI). Ο χρήστης θα πρέπει σε αυτή την περίπτωση να γνωρίζει εκ των προτέρων, ότι πρέπει να επιλέξει τον Explorer προκειμένου να αρχίσει να κάνει ορισμένα πειράματα. Αφού γίνει η παραπάνω ενέργεια, είναι πλέον σε θέση να επιλέξει ένα σετ δεδομένων, το οποίο θα πρέπει και εδώ να γνωρίζει πως γίνεται αυτό και στη συνέχεια θα μπορέσει να χρησιμοποιήσει πιο ουσιαστικά το weka.

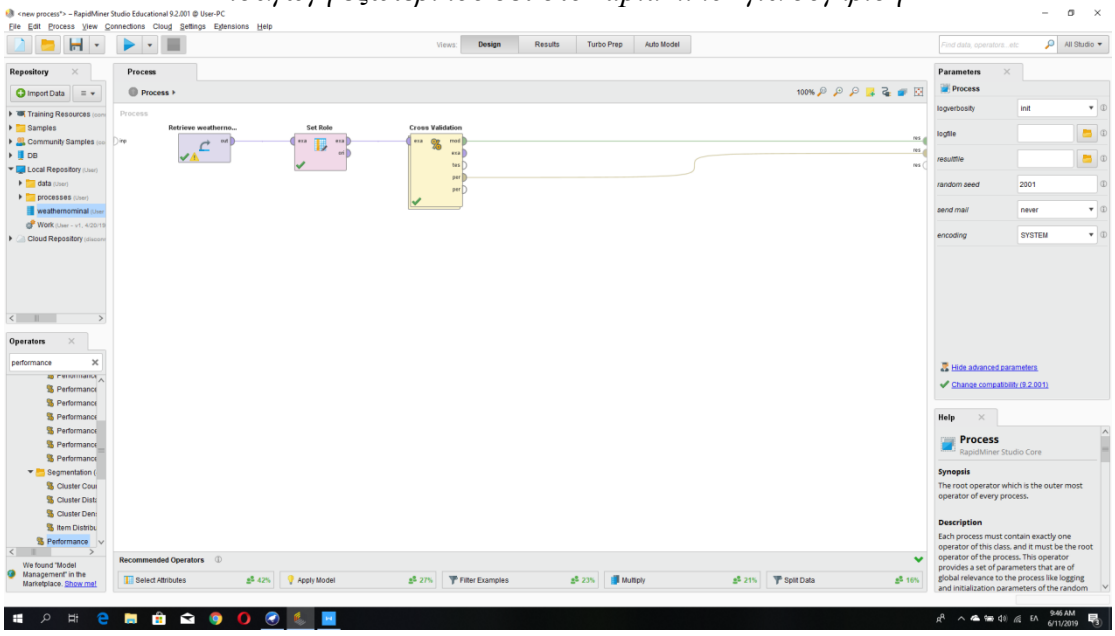
Με βάση αυτές τις αρχικές διαφορές, συμπεραίνουμε ότι το rapidminer έχει καλύτερη δομή και κατεύθυνση του χρήστη, ενώ το λογισμικό weka, προκειμένου να χρησιμοποιηθεί, θα πρέπει να έχει γίνει μία πρώτη μελέτη στο διαδίκτυο για τις βασικές του λειτουργίες ή να υπάρχει κάποιος με πιο εξειδικευμένες γνώσεις που θα βοηθήσει τον χρήστη στην πρώτη του επαφή με αυτό. Επιπλέον το rapidminer έχει ένα πιο φιλικό περιβάλλον, το οποίο επιτυγχάνεται με τα χρώματα που χρησιμοποιούνται σε αυτό, ενώ στο λογισμικό weka κυριαρχεί η μονοτονία σε γκρι χρώμα, γεγονός το οποίο φαίνεται να απωθεί ορισμένους χρήστες. Αν και αυτό δεν έχει καμία σημασία ως προς την απόδοση και τα αποτελέσματα των πειραμάτων, είναι σημαντικό για την πρώτη εντύπωση που μας δίνει.

11.2 Ως προς την ταχύτητα εκτέλεσης και ακρίβεια αλγορίθμων

Προκειμένου να πραγματοποιηθεί ισάξια αυτή η σύκγριση, θα πρέπει να επιλέξουμε ένα σύνολο δεδομένων, το οποίο θα χρησιμοποιήσουμε και στα δύο λογισμικά, εκτελλόντας έναν ή περισσότερους ίδιους αλγορίθμους και να δούμε αν υπάρχει διαφορά στην ταχύτητα εκτέλεσης, επειδή βρισκόμαστε μέσα σε διαφορετικό περιβάλλον. Για να παραμείνει απλή η διαδικασία θα διαλέξουμε ένα απλό σετ όπως για παράδειγμα το weather nominal, το οποίο βρίσκεται αποθηκευμένο στον φάκελο data του weka και αυτό θα φορτωθεί με μορφή arff στο weka, ενώ ως xls στο rapidminer. Πέρα από την ταχύτητα θέλουμε να προσέξουμε αν είναι ίδια η όχι και η ακρίβεια των αποτελεσμάτων, χρησιμοποιώντας φυσικά τις ίδιες παραμέτρους. Δηλαδή, αν για παράδειγμα επιλέξουμε cross validation με 10 folds για το weka τότε ακριβώς ίδιες παραμέτους θα θέσουμε και στο rapidminer. Να επισημάνουμε ότι το συγκεκριμένο σύνολο δεν είναι labeled και για τον λόγο αυτό, όταν το εισάγουμε στο τμήμα process του rapidminer, θα πρέπει να χρησιμοποιήσουμε τον operator “Set Role” και να θέσουμε την στήλη “play” ως labeled, γιατί αυτή θέλουμε να προβλέψουμε. Στη συνέχεια τοποθετούμε τον operator “Decision Tree” και όπως και σε προηγούμενο παράδειγμα του rapidminer, έτσι και εδώ θα εισάγουμε μέσα σε αυτόν από αριστερά τον αλγόριθμο “Desicion Tree” και απο δεξιά τους operators “Apply Model” και “Performance” για να δούμε την τελική ακρίβεια του αλγορίθμου, καθώς και τα λάθη. Παρακάτω φαίνονται οι διαδικασίες που περιγράφηκαν εδώ.

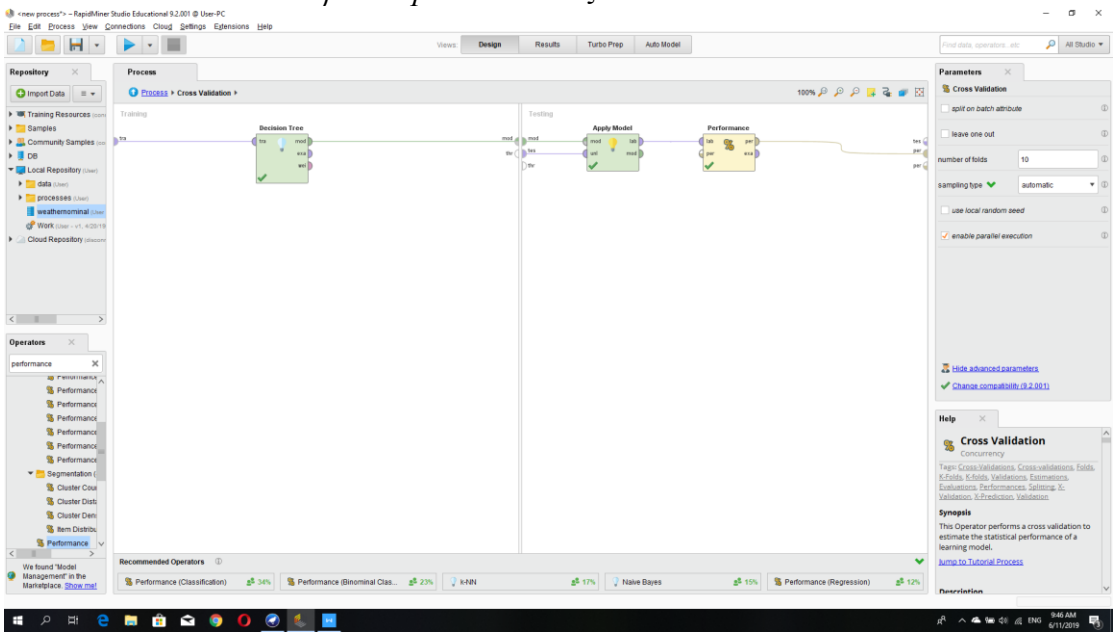
ΕΙΚΟΝΑ 60

Εισαγωγή εξωτερικού σετ στο rapidminer για σύγκριση



ΕΙΚΟΝΑ 61

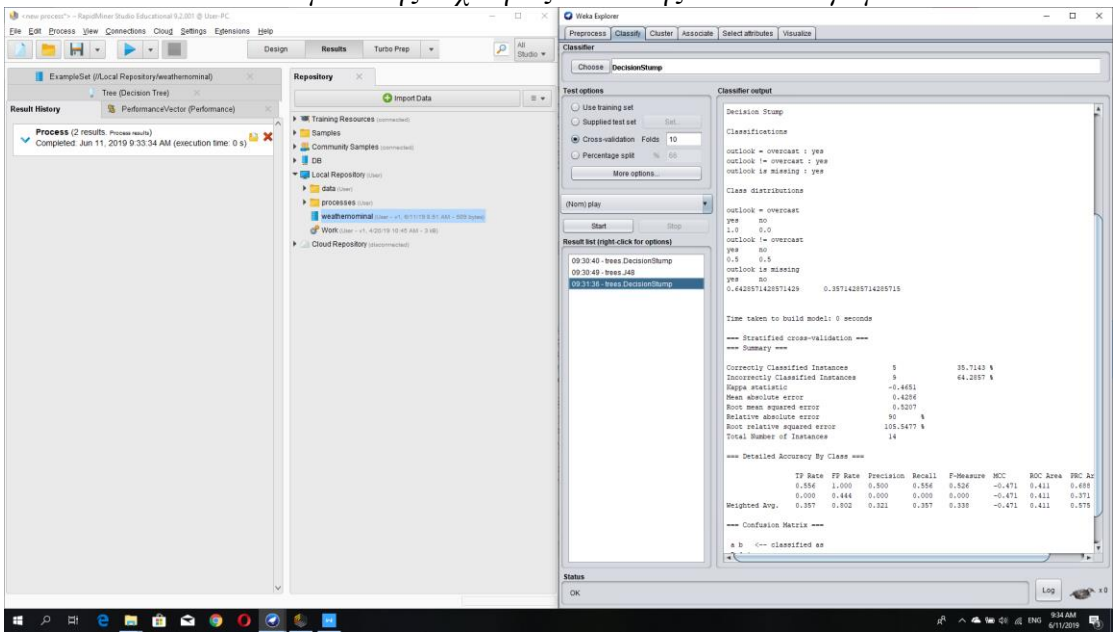
Εσωτερικοί operators εντός του Cross Validation



Επειδή έχει ήδη περιγραφεί αναλυτικά η παραπάνω διαδικασία σε προηγούμενο tutorial, θα συνεχίσουμε παρακάτω για τα αποτελέσματα της σύγκρισης. Για να υπενθυμίσουμε όμως ότι αυτό το βήμα απαιτεί απαραίτητη προϋπόθεση για την σωστή εκτέλεση της σύγκρισης πρέπει να το εισάγουμε εδώ πριν προχωρήσουμε.

ΕΙΚΟΝΑ62

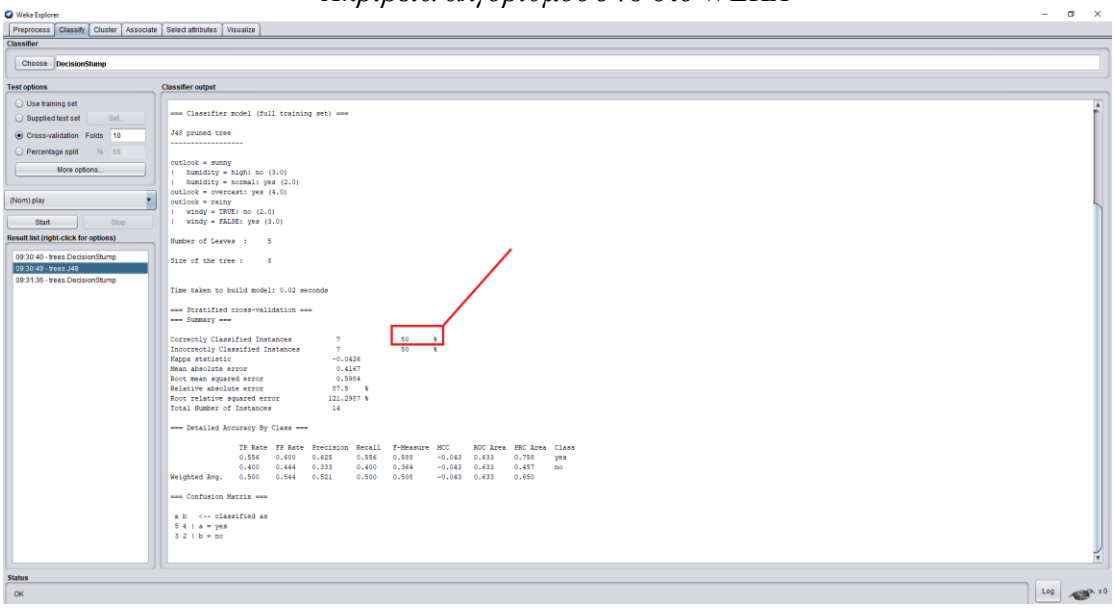
Αποτελέσματα της ταχύτητας εκτέλεσης στα δύο λογισμικά



Παρατηρούμε ότι στο συγκεκριμένο πείραμα, όπου έχουμε μικρό σύνολο δεδομένων, η ταχύτητα εκτέλεσης και στα δύο λογισμικά είναι η ίδια, δηλαδή 0 δευτερόλεπτα το οποίο ήταν αναμενόμενο φυσικά. Δεν χρειάζεται να αναφέρουμε κάτι άλλο εδώ, επομένως πηγαίνουμε στις επόμενες εικόνες, όπου παρουσιάζονται τα αποτελέσματα της ακρίβειας των αλγορίθμων.

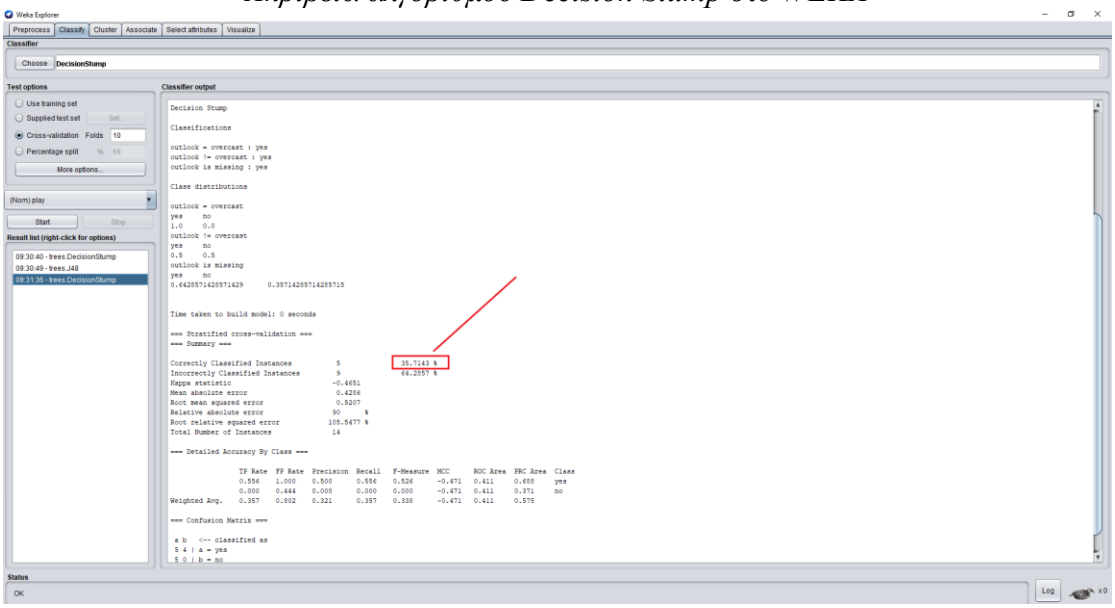
ΕΙΚΟΝΑ63

Ακρίβεια αλγορίθμου J48 στο WEKA



ΕΙΚΟΝΑ64

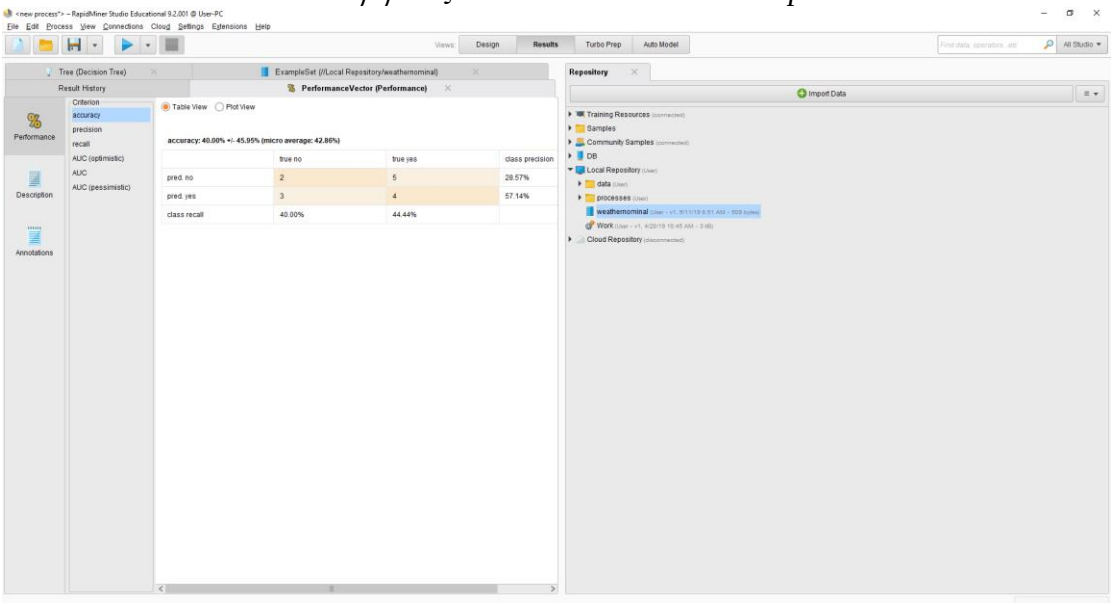
Ακρίβεια αλγορίθμου Decision Stump στο WEKA



Στις δύο προηγούμενες εικόνες αναπαρίστανται τα αποτελέσματα τις ακρίβειας με τιμές 50% και 35.7% για τους J48 και Decision Stump αντίστοιχα. Η ακρίβεια είναι πολύ χαμηλή και στις δύο περιπτώσεις, αλλά ας δούμε στη συνέχεια τι αποτέλεσμα λαμβάνουμε από το rapidminer.

ΕΙΚΟΝΑ 65

Ποσοστό ακρίβειας του Decision Tree στο rapidminer



Και σε αυτό το λογισμικό δεν υπάρχει ιδιαίτερη διαφορά, κάτι το οποίο είναι λογικό αφού χρησιμοποιούμε τα ίδια σύνολα δεδομένων και τους ίδιους αλγορίθμους όπου εδώ η ακρίβεια έρχεται με ποσοστό από 40% έως 45.95%.

Συμπερασματικά, για την πρώτη σύγκριση των δύο λογισμικών με θέμα την ταχύτητα εκτέλεσης και τα ποσοστά ακρίβεια ίδιων αλγορίθμων, παρατηρούμε ότι δεν υπάρχει σχεδόν καμία διαφορά, εκτός φυσικά από τον αλγόριθμο Decision Stump, ο οποίος έδειξε 10% μειωμένη ακρίβεια σε σχέση με τους άλλους. Σε αυτή την περίπτωση θα προτιμήσουμε την χρήση του λογισμικού WEKA, γιατί η διαδικασία εκτέλεσης του πειράματος απαιτεί λιγότερα βήματα και μπορεί να γίνει ακόμα και από έναν σχετικά νέο χρήστη. Αντιθέτως, όπως είδαμε στο rapidminer, χρειάζονται μερικές λεπτομέρειες, όπως είναι οι εσωτερικοί operators του “Cross Validation”, κάτι το οποίο κάνει την διαδικασία λίγο πιο σύνθετη. Φυσικά αυτό δεν είναι το μοναδικό κριτήριο για να επιλέξουμε το WEKA, επομένως στη συνέχεια θα δούμε περισσότερες συγκρίσεις για να καταλήξουμε στην βέλτιστη επιλογή.

11.3 Ως προς την ευκολία διαχείρισης αρχείων

Το WEKA δέχεται αρχεία .arff και .csv τα οποία πρέπει να είναι ορθά γραμμένα, δηλαδή τα δεδομένα να είναι ομαδοποιημένα, να υπάρχουν κλάσεις και άλλες λεπτομέρειες. Το rapidminer δέχεται αρχεία .csv και αρχεία xls για τα οποία ισχύουν σχεδόν οι ίδιες προδιαγραφές όπως και στο WEKA. Η διαφορά στο rapidminer είναι ότι καθώς εισάγουμε ένα νέο σύνολο, έστω ένα αρχείο xls, το rapidminer μας ρωτάει για τους τύπους των δεδομένων που βρίσκονται ανά στήλη και εμείς μπορούμε να θέσουμε την κάθε μία, ανάλογα με τα δεδομένα που περιέχει (numerical, nominal, boolean, κτλ.). Επιπλέον, το γεγονός ότι στο rapidminer παρέχονται οι επιπρόσθετες λειτουργίες turbo prep και auto-model, το καθιστά πολύ εύχρηστο και εύκολο στη χρήση εργαλείο, αφού μπορούμε κατευθείαν με την εκτέλεση μερικών απλών βημάτων να πετύχουμε αυτοματοποιημένη διαχείριση των δεδομένων μας και να λάβουμε τα ανάλογα αποτελέσματα. Με βεβαιότητα θα λέγαμε ότι είναι καλύτερο από το WEKA σε αυτό το κομμάτι, δηλαδή της διαχείρισης γιατί είναι πολύ δομημένο, έχει καλύτερα οργανωμένο GUI (Graphical User Interface) και σε κάθε βήμα υπάρχουν βοηθητικές πληροφορίες που μας εξηγούν τι ακριβώς κάνουμε και πιο θα είναι το επόμενο βήμα. Με αυτό τον τρόπο, όχι μόνο μαθαίνουμε καλύτερα να χρησιμοποιούμε το λογισμικό, αλλά συγχρόνως ξέρουμε τι ακριβώς είναι αυτό που κάνουμε. Στο WEKA, ίσως σε ορισμένα σημεία να έχετε λάβει ένα αποτέλεσμα ή να κάνατε κάτι, για το οποίο δεν μπορούσατε να καταλάβετε περί τίνος πρόκειται. Συνήθως ένα αποτέλεσμα εκτέλεσης αλγορίθμων ή μία αναπαράσταση δεδομένων.

Πέρα από τα παραπάνω, στο rapidminer, οι πολλοί και διαφορετικοί operators μας επιτρέπουν να οργανώσουμε τα δεδομένα μας πολύ καλά και με εύκολο τρόπο. Για κάθε operator, υπάρχει μία περιγραφή του πως δουλεύει, η οποία βρίσκεται στο τμήμα “Help” και έτσι δεν χρειάζεται ο χρήστης να ψάχνει στο διαδίκτυο μέχρι να βρει αυτό που θέλει.

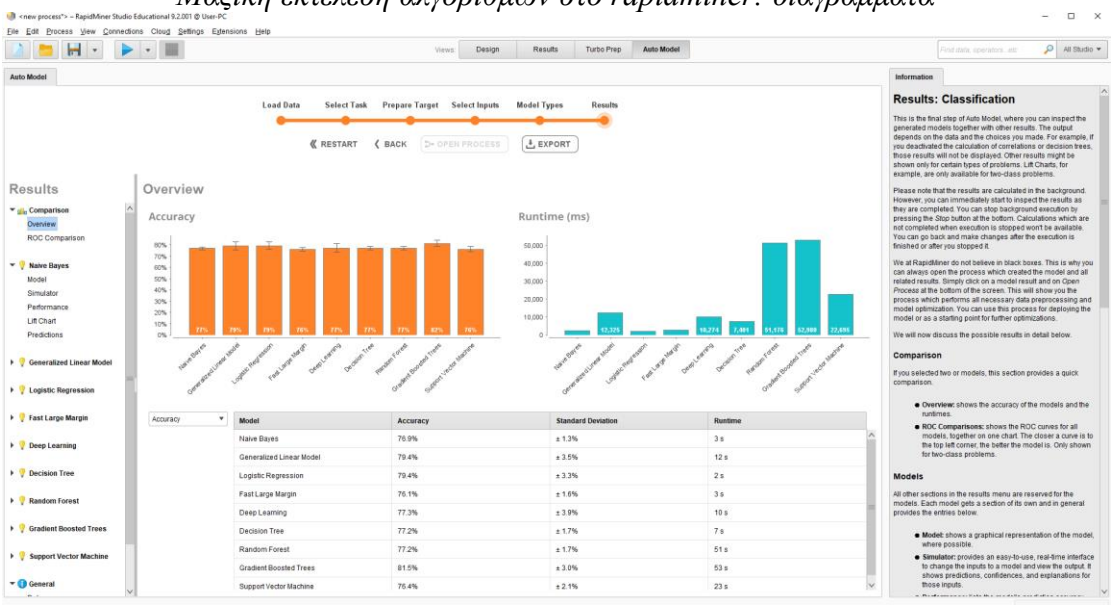
11.4 Ως προς την αναπαράσταση των αποτελεσμάτων

Ξεκινώντας με το rapidminer, τα αποτελέσματα κάθε πειράματος, όπως έχουμε ήδη αναφέρει, παρουσιάζονται στο τμήμα “Results view”. Το ξεχωριστό αυτό τμήμα, το οποίο είναι εξίσου καλά οργανωμένο, παρέχει γραφήματα, περιγραφές, ιστορικό, διαγράμματα, σχόλια, τα οποία μπορεί ο χρήστης να προσθέσει και να αποθηκεύσει, απεικόνιση δέντρων και άλλες αναπαραστάσεις ή πληροφορίες ανάλογα με το πείραμα που έχουμε εκτελέσει εκείνη τη στιγμή. Στο WEKA, όταν εισάγουμε τα δεδομένα και βρισκόμαστε στο τμήμα “Preprocess” εμφανίζονται αμέσως κάποιες πληροφορίες που μας δείχνουν το σύνολο των χαρακτηριστικών του αρχείου, τα συνολικά “instances” και γίνεται επίσης ένα count για κάθε κατηγορία ώστε να ξέρουμε τις ποσότητες ανά κατηγορία. Επιπλέον υπάρχει απεικόνιση με τη μορφή διαγραμμάτων που χωρίζονται με κόκκινο και μπλε χρώμα. Όταν πρόκειται να εκτελέσουμε ένα πείραμα με χρήση κάποιου αλγορίθμου, τα αποτελέσματα εμφανίζονται στο σημείο της εκτέλεσης, δηλαδή στο τμήμα “Classify” σε μία σειρά και είναι χωρισμένα με τίτλους όπως (Summary, Confusion Matrix). Για να δούμε την απεικόνιση ενός δέντρου πατάμε δεξί κλικ στο “result list” και επιλέγουμε “visualize tree”. Στο rapidminer αυτό γίνεται αυτόματα και μάλιστα η απεικόνιση του δέντρου είναι αρκετά καλύτερη και πιο λεπτομερής. Για παράδειγμα σε κάθε φύλλο αναγράφεται ο συνολικός αριθμός των “items” και πόσα από αυτά είχαν τιμή “yes” ή “no”.

Ένα άλλο σημαντικό σημείο του “visualization” πέρα από τα δέντρα, είναι τα διαγράμματα, οι γραφικές παραστάσεις και η ακρίβεια. Παρακάτω παρουσιάζονται δύο εικόνες από τη λειτουργία “Auto Model” του rapidminer όπου καλύπτουν όλα όσα έχουμε αναφέρει εδώ. Στην ΕΙΚΟΝΑ 66 φαίνεται η μαζική αναπαράσταση των αποτελεσμάτων ακρίβειας όλων των αλγορίθμων που εκτελέστηκαν αυτόματα για ένα σύνολο δεδομένων. Με πορτοκαλί χρώμα σημειώνεται η ακρίβεια του καθενός, ενώ με γαλάζιο ο χρόνος εκτέλεσης. Ακριβώς από κάτω εμφανίζεται η ακρίβεια ξεχωριστά για τον κάθε αλγόριθμο καθώς και πόση απόκλιση (+.-) είχε.

ΕΙΚΟΝΑ 66

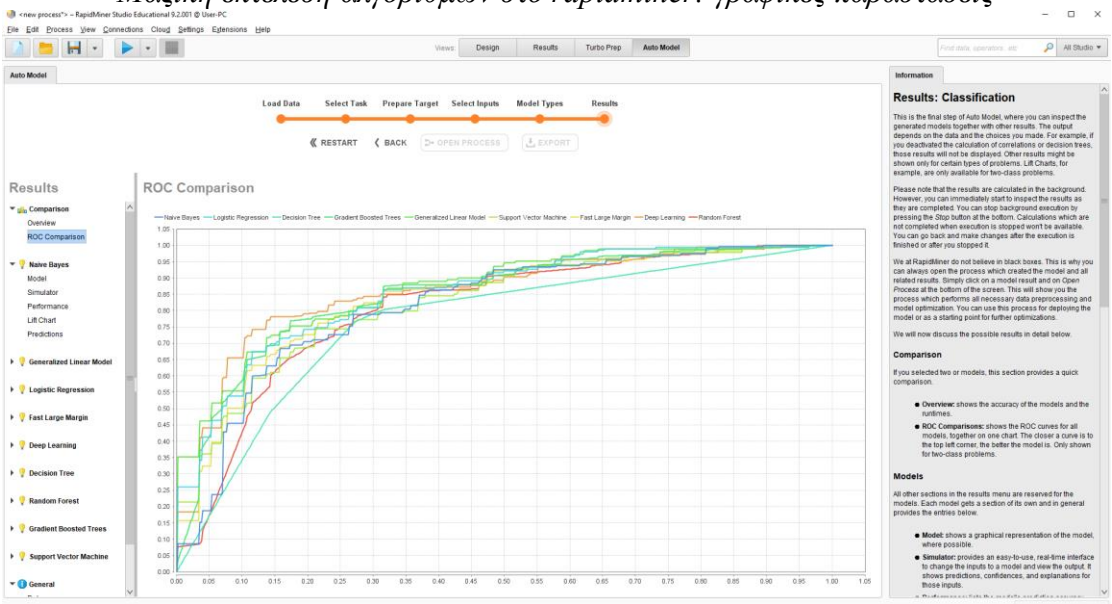
Μαζική εκτέλεση αλγορίθμων στο rapidminer: διαγράμματα



Στην ΕΙΚΟΝΑ 67 έχουμε μία διαφορετική όψη, δηλαδή είναι η επιλογή “ROC” όπου εδώ η γραφική παράσταση που βρίσκεται πιο αριστερά και προς τα πάνω αντιστοιχεί στον πιο αποδοτικό αλγόριθμο. Κάθε αλγόριθμος έχει διαφορετικό χρώμα στην καμπύλη για να διακρίνεται από τους υπόλοιπους. Επιπλέον στην αριστερή στήλη βρίσκονται τα ονόματα των αλγορίθμων και μπορούμε εύκολα από εκεί να επιλέξουμε κάποιον για να δούμε πιο αναλυτικά τα αποτελέσματά του.

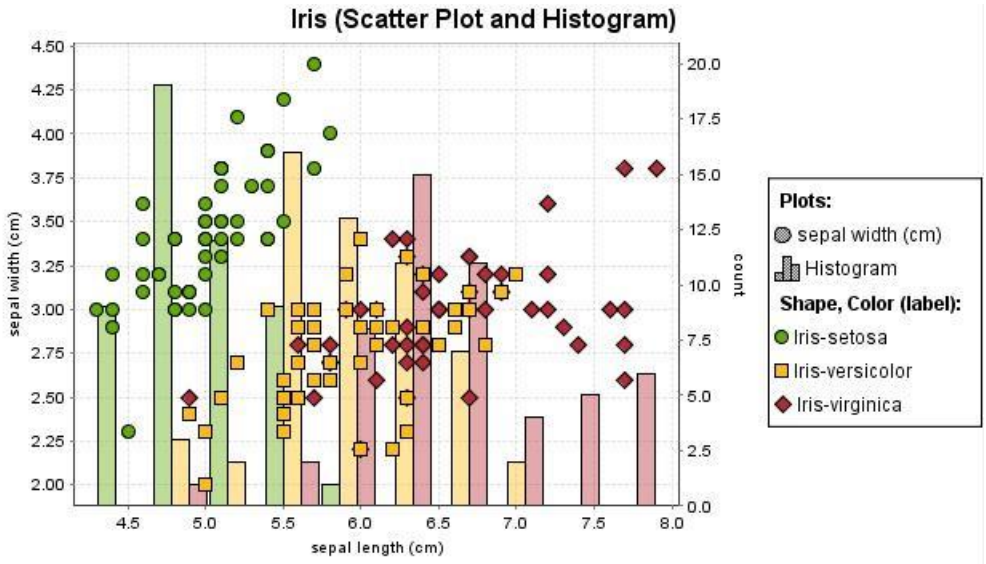
ΕΙΚΟΝΑ 67

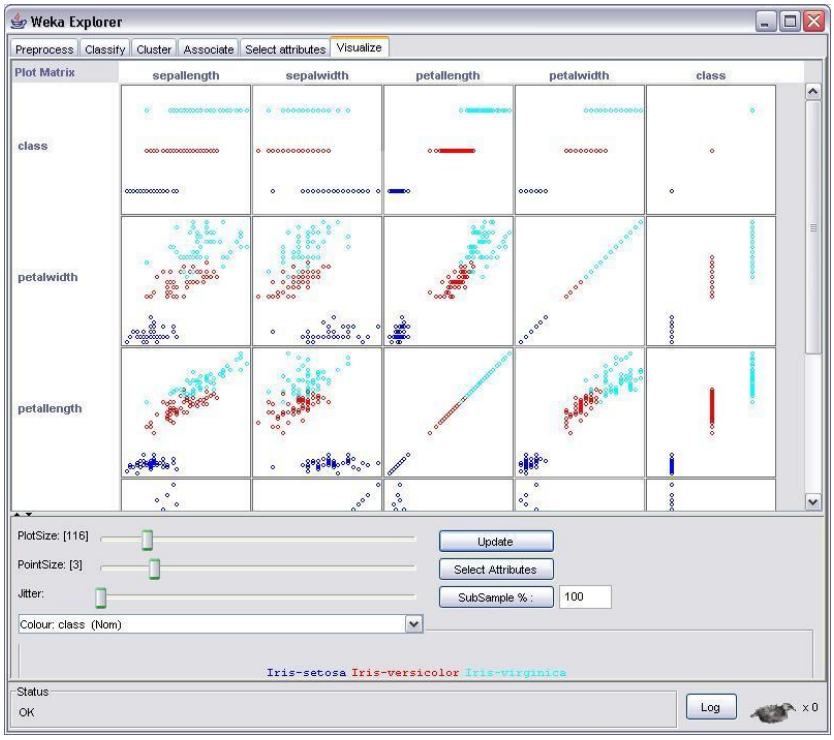
Μαζική εκτέλεση αλγορίθμων στο rapidminer: γραφικές παραστάσεις



Στο WEKA η οπτικοποίηση διαφέρει αρκετά με αυτή του rapidminer και μάλιστα είναι πιο δύσκολο να ξεχωρίσουμε με εντελώς απλό τρόπο τις πληροφορίες που λαμβάνουμε. Κάτι πολύ σημαντικό είναι ότι στο WEKA δεν μπορούμε να εκτελέσουμε ταυτόχρονα πολλούς αλγορίθμους, που σημαίνει ότι ο χρήστης θα χρειάζεται να επιλέγει μεμονωμένα τον κάθε αλγόριθμο, να το εκτελεί και στη συνέχεια να βλέπει τα αποτελέσματα του ενός και έπειτα του άλλου. Επομένως, όταν θέλουμε να συγκρίνουμε 2 ή και περισσότερους αλγορίθμους ταυτόχρονα, θα προτιμήσουμε το rapidminer.

Το WEKA έχει πολύ αναλυτική οπτικοποίηση και μάλιστα μπορούμε να δούμε συγκεκριμένα για κάθε στοιχείο του συνόλου των δεδομένων μας τι συμβαίνει. Για παράδειγμα μπορούμε να εντοπίσουμε αν έγινε λάθος πρόβλεψη με τιμή “yes” για κάποιο instance που θα στη πραγματικότητα ήταν “no”. Στις παρακάτω δύο εικόνες φαίνονται διαφορές στην οπτικοποίηση ίδιων δεδομένων, με το rapidminer να έχει σχηματίσει ιστόγραμμα και στη συνέχεια το weka με τα διαγράμματα διασποράς.


Iris (Scatter Plot and Histogram)
sepal width (cm)
sepal length (cm)
count
4.50
4.25
4.00
3.75
3.50
3.25
3.00
2.75
2.50
2.25
2.00
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Plots:
sepal width (cm)
Histogram
Shape, Color (label):
Iris-setosa
Iris-versicolor
Iris-virginica


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Plot Matrix
sepallength
sepalwidth
petallength
petalwidth
class
PlotSize: [116]
PointSize: [3]
Jitter:
Update
Select Attributes
SubSample % :
100
Colour: class (Nom)
Iris-setosa Iris-versicolor Iris-virginica
Status
OK
Log
x 0

11.5 Ως προς το εύρος δυνατοτήτων

Στο τελικό αυτό κομμάτι θα συγκρίνουμε τα δύο λογισμικά, ώστε να δούμε ποιο είναι τελικά αυτό που θα προτιμήσει ο χρήστης, ανάλογα με τις δυνατότητες που παρέχει το καθένα. Μία πολύ βασική διαφορά ανάμεσα τους είναι η δυνατότητα του rapidminer να εξηγεί στον χρήστη αναλυτικά τι είναι το κάθε στοιχείο που χρησιμοποιεί μέσα από το λογισμικό, κάτι το οποίο δεν υπάρχει στο weka. Δηλαδή, αν για παράδειγμα ο χρήστης επιλέξει έναν "operator" στο rapidminer, τότε θα εμφανιστούν στην άκρη του δεξιού μέρους πληροφορίες για τον συγκεκριμένο "operator". Αν ο χρήστης έκανε κάτι παρόμοιο με έναν αλγόριθμο στο weka, εκεί δεν θα εμφανίζονταν πληροφορίες για τον αλγόριθμο. Επομένως, έχουμε ήδη ένα πρώτο επιχείρημα που μας δίχνει ότι το rapidminer έχει μεγαλύτερο εύρος δυνατοτήτων. Μία δεύτερη βασική διαφορά είναι η δυνατότητα για online tutorials στην επίσημη ιστοσελίδα του λογισμικού. Το rapidminer έχει μία πολύ καλά οργανωμένη ιστοσελίδα στο διαδίκτυο, που περιλαμβάνει αρκετά tutorials με χρήση βίντεο, ώστε αν κάποιος νέος χρήστης δεν νιώθει έτοιμος να κατεβάσει και να χρησιμοποιήσει μόνος του το λογισμικό εξ αρχής, να μπορεί να διδαχθεί τα βασικά από την ιστοσελίδα. Επιπλέον, σε περίπτωση που έχει ήδη κάνει λήψη και εγκατάσταση, μπορεί μέσα από το rapidminer, να μεταβεί στον ιστότοπο και να μάθει αρχικά από εκεί. Αυτό δεν συμβαίνει στο weka, επομένως ο χρήστης θα πρέπει να αναζητήσει στο διαδίκτυο άλλες ιστοσελίδες ή βίντεο με tutorials για το λογισμικό αυτό. Τρίτο σημαντικό σημείο για την σύγκριση είναι η επεξεργασία των συνόλων δεδομένων. Σε όλη την πορεία της εργασίας έχουμε δείξει πολλές εικόνες για το πώς επεξεργάζονται τα δεδομένα τα δύο αυτά λογισμικά. Και τα δύο έχουν παρόμοιες δυνατότητες όπως και αποτελέσματα όταν εκτελούνται τα ίδια πειράματα, αλλά το rapidminer μπορεί να γίνει πολύ πιο εύκολο και εύχρηστο σε αρκετές περιπτώσεις. Αυτό επιτυγχάνεται από τις επιπρόσθετες λειτουργίες "auto model" και "turbo prep" που έχουμε εξηγήσει προηγουμένως, όπου όλες σχεδόν οι ενέργειες γίνονται αυτοματοποιημένα και ο χρήστης κάνει μόνο μερικά κλικ. Ίσως κάποιος να προτιμά να κάνει όλα τα βήματα μόνος του για την ανάλυση και εξόρυξη δεδομένων, από την οργάνωση του αρχικού σετ μέχρι την εκτέλεση αλγορίθμου, αλλά αυτό απαιτεί περισσότερο χρόνο και προσοχή ώστε να μην γίνουν λάθη. Επειδή πολλές φορές οι ρυθμοί είναι απαιτητικοί και ο χρόνος είναι ο βασικός παράγοντας της επιτυχίας, το rapidminer είναι σαφώς προτιμότερο για να τελειώσει κάποιος γρήγορα την ανάλυση δεδομένων. Σε έναν οργανισμό η ταχύτητα είναι ζωτικής σημασίας και τα δεδομένα που θα εξαχθούν από ένα σετ, θα χρησιμοποιηθούν στη συνέχεια για έναν άλλο σκοπό. Αν η ανάλυση είναι μια χρονοβόρα διαδικασία, τότε και οι υπόλοιπες διαδικασίες μέσα στον οργανισμό θα καθυστερήσουν να ολοκληρωθούν.

12 ΣΥΜΠΕΡΑΣΜΑΤΑ

Συμπερασματικά, ο κλάδος της εξόρυξης δεδομένων αποτελεί ακρογωνιαίο λίθο για τις μεγάλες εταιρείες της σημερινής εποχής και για τον ίδιο τον άνθρωπο. Με την απόκτηση των πληροφοριών των πελατών, μπορεί να γίνει καλύτερη διάκριση των επιθυμιών και των αναγκών τους και έτσι οι εταιρείες να παράγουν πιο αποτελεσματικά προϊόντα. Τόσο για την υγεία, την διασκέδαση, την εγκληματικότητα, την εξέλιξη και την καινοτομία, η πληροφορία που μπορούμε να λάβουμε μέσα από τα δεδομένα είναι εξαιρετικής σημασίας. Προκειμένου όμως να μπορούν να υλοποιηθούν όλα τα παραπάνω και να υπάρξει βελτίωση και πρόοδος, η εξόρυξη δεδομένων συνδιάζεται με τα κατάλληλα λογισμικά. Για τον λόγο αυτόν στην παρούσα έρευνα αναλύθηκαν με αρκετά παραδείγματα τα λογισμικά weka και rapidminer, τα οποία αποσκοπούν στην απόκτηση της πληροφορίας και στην αναπαράσταση της, με τέτοιο τρόπο που ο αναλυτής θα λάβει ότι είναι σημαντικό για την έρευνα του.

Σύμφωνα με όλα τα παραδείγματα της παρούσας εργασίας και έπειτα από την αναλυτική σύγκριση, συμπεραίνουμε ότι το λογισμικό rapidminer μπορεί να χρησιμοποιηθεί πιο αποτελεσματικά από κάποιον χρήστη, που δεν είχε προηγούμενη εμπειρία με την εξόρυξη δεδομένων και την μηχανική μάθηση. Οφείλουμε να τονίσουμε ότι τα σύνολα δεδομένων πρέπει να είναι εκ των προτέρων ορθά δομημένα και σε μορφή αρχείου arff, ώστε να τα εισάγουμε στο weka. Αντιθέτως στο rapidminer, αρκεί να υπάρχει ένα αρχείο xls ή csv το οποίο, με πολύ απλό τρόπο γίνεται να το χειριστούμε μέσα στο λογισμικό. Το rapidminer παρέχει πλήρη υποστήριξη για να βοηθήσει τον χρήστη να μάθει να το χρησιμοποιεί τόσο για απλά όσο και για πιο σύνθετα προβλήματα, με τα οποία ερχόμαστε αντιμέτωποι και σε πραγματικές συνθήκες εργασίας. Από την άλλη πλευρά, για να επιλύσει ο χρήστης προβλήματα με το weka, θα πρέπει να αφιερώσει αρκετό χρόνο, ώστε να μάθει τις βασικές λειτουργίες του. Κατά την αναζήση μου στο διαδίκτυο για tutorials που σχετίζονται με το weka, δεν είχα καταφέρει να βρω αρκετές πληροφορίες ή κάποια καλά δομημένη ιστοσελίδα με αναλυτικά παραδείγματα. Αυτό που με βοήθησε αρκετά ήταν ορισμένα μαθήματα μέσα από βίντεο.

Όσων αφορά το κομμάτι των δυνατοτήτων, πράγματι και τα δύο λογισμικά είναι αξιόπιστα για την επίλυση προβλημάτων είτε ταξινόμησης είτε παλινδρόμησης, όπως έχουμε ήδη αναφέρει και μάλιστα σε αρκετές περιπτώσεις πετυχαίνουμε υψηλή ακρίβεια. Αυτό σημαίνει ότι μπορούμε να τα χρησιμοποιήσουμε χωρίς αμφιβολία για διάφορα πειράματα. Όμως, τόσο η οπτικοποίηση όσο και η συνολική δομή που έχει ένα αποτέλεσμα πειράματος, φαίνεται καλύτερα στο rapidminer. Αν το ζήτημα σε ένα πρόβλημα είναι η ταχύτητα εκτέλεσης, τότε πρέπει να αναφέρουμε ότι η εκτέλεση με ίδια δεδομένα ολοκληρώνεται και στα δύο λογισμικά με την ίδια ταχύτητα. Αν θέλουμε να εξετάσουμε, να συγκρίνουμε και να «τρέξουμε» πολλούς αλγορίθμους ταυτόχρονα τότε αυτό επιτυγχάνεται μέσω του rapidminer, τόσο με τη χρήση κατάλληλου “Operator” όσο και με την λειτουργία auto model, που κάνει την διαδικασία πολύ πιο απλή. Αυτή η δυνατότητα για μαζική εκτέλεση αλγορίθμων δεν υπάρχει στο weka. Εκεί πρέπει ο αναλυτής να επαναλαμβάνει την διαδικασία με έναν προς έναν και να συγκρίνει τα αποτελέσματα, αφού αυτά αποθηκεύονται στο ιστορικό εκτέλεσης αλγορίθμων.

Από την συνολική έρευνα προκύπτει το γενικότερο συμπέρασμα ότι το rapidminer είναι πιο κατάλληλο για χρήση, ειδικότερα αν ο χρήστης δεν είχε προηγούμενες γνώσεις, αλλά οι δυνατότητες και των δύο λογισμικών δεν σταματούν σε αυτό το σημείο. Πιο συγκεκριμένα, έπειτα από το κεφάλαιο της σύγκρισης των δύο λογισμικών, είδαμε για ποιους λόγους το rapidminer φαίνεται να ξεπερνάει το weka. Η εργασία θα μπορούσε να επεκταθεί ακόμα περισσότερο στην ανάλυση του weka, αφού αυτό περιλαμβάνει και άλλους αλγορίθμους. Αυτό θα έκανε την συγκριτική αξιολόγηση ακόμα εκτενέστερη. Παρόλα αυτά, χρησιμοποιώντας κανείς το rapidminer, θα ανακαλύψει ότι στο τμήμα των "Operators" υπάρχουν πάρα πολλοί αλγόριθμοι. Έτσι θα ήταν και πάλι δύσκολο να πούμε ότι το weka θα μπορούσε να ξεπεράσει το rapidminer.

Εξαρτάται πάντα από τον ίδιο τον χρήστη ή τον αναλυτή πιο θεωρεί εκείνος το πιο εξυπηρετικό για έργο του. Αν μπορεί εύκολα και γρήγορα να χειρίζεται ένα σύνολο δεδομένων, να του δίνει την ορθή δομή αρχείου arff και να γνωρίζει εκ των προτέρων ποιους αλγορίθμους θέλει να εκτελέσει για το συγκεκριμένο πείραμα, τότε μπορεί να κάνει όλη αυτή τη διαδικασία πολύ εύκολα με το weka. Αν όμως θέλει να μάθει ακριβώς πως λειτουργεί το κάθε στοιχείο (αλγόριθμος, παράμετροι, φίλτρα κτλ.) μέσα στο πείραμα και δεν έχει τις απαραίτητες γνώσεις, τότε είναι καλύτερο το rapidminer. Επιπλέον, η κοινότητα του rapidminer είναι πάντα διαθέσιμη για επικοινωνία, συζήτηση και επίλυση οποιασδήποτε απορίας. Τόσο η επικοινωνία μέσω email με κάποιον ειδικό, όσο και η ανάρτηση κάποιας ερώτησης στην ιστοσελίδα του λογισμικού, το καθιστά άμεσα πιο εξυπηρετικό. Επιπροσθέτως, οι χρήστες του rapidminer μπορούν μέσα στο λογισμικό να δημιουργούν κοινά project και να εργάζονται πάνω σε αυτά από απόσταση, απαιτείται όμως η σύνδεση στο διαδίκτυο. Αυτός είναι ένας πολύ καλός τρόπος για να μπορέσουν συγκεκριμένα φοιτητές της Πληροφορικής να εξοικειωθούν στο κομμάτι της συνεργασίας, της ομαδικής δουλειάς και να αποκτήσουν μία εμπειρία, αντίστοιχη με αυτή που πιθανόν να συναντήσουν στη συνέχεια, στον επαγγελματικό τομέα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1]Εξόρυξη δεδομένων γενικές πληροφορίες
https://en.wikipedia.org/wiki/Data_mining

[2]Μηχανική μάθηση γενικές πληροφορίες
https://en.wikipedia.org/wiki/Machine_learning

[3]Μηχανική μάθηση επιπλέον υλικό
https://www.sas.com/en_id/insights/analytics/machine-learning.html

[4]Επίσημη σελίδα του WEKA
https://www.cs.waikato.ac.nz/ml/weka/

[5]Πληροφορίες για το WEKA και μηχανική μάθηση
https://en.wikipedia.org/wiki/Weka_(machine_learning)

[6]Επιστημονικά άρθρα που σχετίζονται με τη Πληροφορική
http://www.jmlr.org/papers/

[7]Πλήρης οδηγός για το WEKA
https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

[8]Τα αρχεία μορφής .arff
https://www.cs.waikato.ac.nz/ml/weka/arff.html?fbclid=IwAR0DVTsDzb_dt1BWxA-CSPjUxS7GMJHkGGrMgfJg2nR7jrYuqzaUsasiheE

[9]Τα αρχεία μορφής .csv
https://www.howtogeek.com/348960/what-is-a-csv-file-and-how-do-i-open-it/

[10]Predictive Modeling
https://machinelearningmastery.com/gentle-introduction-to-predictive-modeling/

[11]Πως λειτουργούν οι αλγόριθμοι μηχανικής μάθησης
https://machinelearningmastery.com/how-machine-learning-algorithms-work/

[12]Πως λειτουργούν τα δέντρα απόφασης
http://dataaspirant.com/2017/01/30/how-decision-tree-algorithm-works/

[13]Αλγόριθμος J48
Improved J48 Classification Algorithm for the Prediction of Diabetes (PDF)

[14]ZeroR Algorithm & Baseline Performance
https://machinelearningmastery.com/estimate-baseline-performance-machine-learning-models-weka/

[15]Διαφορές Classification και Regression Predictive Modeling
https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/

[16]Περισσότερες πληροφορίες για weka algorithms
https://www.ibm.com/developerworks/library/os-weka3/index.html

[17]Το λογισμικό rapidminer
https://rapidminer.com/

[18]Μέρος εγκατάστασης του λογισμικού
https://rapidminer.com/get-started/

[19]Χρήσιμα βίντεο για το rapidminer
https://academy.rapidminer.com/learning-paths/get-started-with-rapidminer-and-machine-learning