Πανεπιστήμιο Αιγαίου
Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Αποθήκες Δεδομένων και Εξόρυξη Γνώσης από Δεδομένα

Ομαδική Εργασία

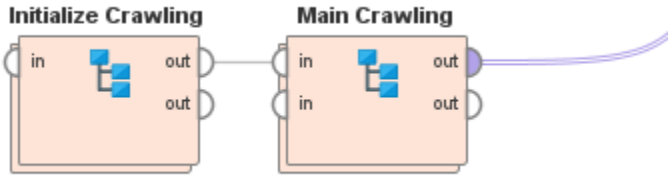
321/2014134 Μπόνης Αθανάσιος
321/2013187 Τσιρίδης Γεώργιος

Δευτέρα 04/06/2018

Περιεχόμενα

Υποεργασία 1η

Αρχικά όσον αφορά το κομμάτι υλοποίησης του RapidMiner, σε γενικό πλαίσιο, η διαδικασία crawling ξεκινά με ένα subprocess το οποίο κάνει Initialize, δηλαδή αντλεί πληροφορίες από μια χώρα και τις αποθηκεύει στο CSV. Στη συνέχεια, κάνει ακριβώς το ίδιο με τη μόνη διαφορά ότι το κάνει σε loop για κάθε χώρα κάνοντας append to file για το CSV ώστε να κρατάμε και τα προηγούμενα δεδομένα.



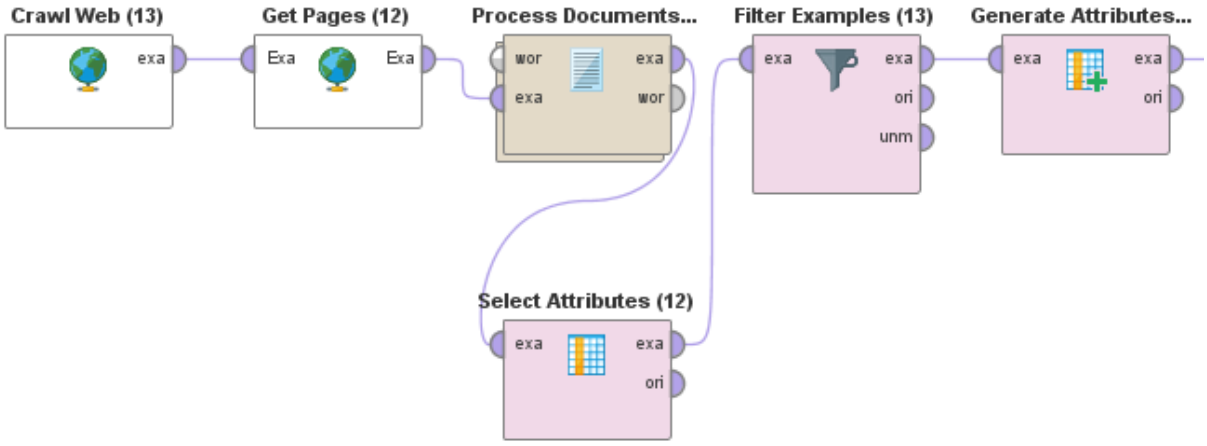
Το Initialize Crawling, ξεκινά με τον Operator Crawl Web, ο οποίος δέχεται ένα Link και στη συνέχεια του κάνει crawling. Του έχουμε θέσει έναν Crawling Rule, ώστε να αντλήσει όλα τα Link τα οποία μας ενδιαφέρουν από το Page, δηλαδή τα archives. Αφού έχει αντλήσει τους συνδέσμους λοιπόν, χρησιμοποιούμε τον Operator GetPages ώστε να αντλήσει τις σελίδες τους.

Παρακάτω, χρησιμοποιούμε Process Documents to Data για να δημιουργήσουμε Vectors από τα δεδομένα. Μέσα στον operator, χρησιμοποιούμε Extract Information, ώστε να πάρουμε ακριβώς το δεδομένο που μας ενδιαφέρει. Χρησιμοποιώντας XPath, προσδιορίζουμε ακριβώς την παράγραφο που θέλουμε με τον παρακάτω τρόπο μέσω Query Expression.

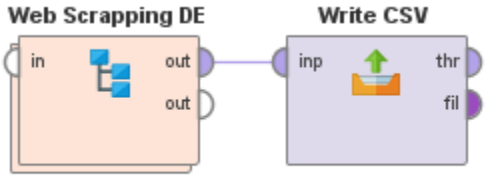
attribute name	query expression
mainheader	//h:h2[@id='infocollect']/following-sibling::h:p/text()

Αφού έχουμε αντλήσει την πληροφορία, χρησιμοποιούμε Select Attributes, για να επιλέξουμε το attribute με την παράγραφο που θέλουμε και του προσθέτουμε Filter Examples ώστε να μην πάρει δεδομένα

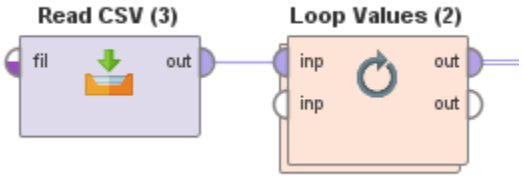
που είναι κενά. Στη συνέχεια, ακολουθεί ένας Operator Generate Attributes, ώστε να δημιουργήσουμε ένα label το οποίο θα προσδιορίζει τι γλώσσα θα είναι το κείμενο. Η τελική μορφή του πριν γράψει τα δεδομένα στο CSV αρχείο είναι η παρακάτω.



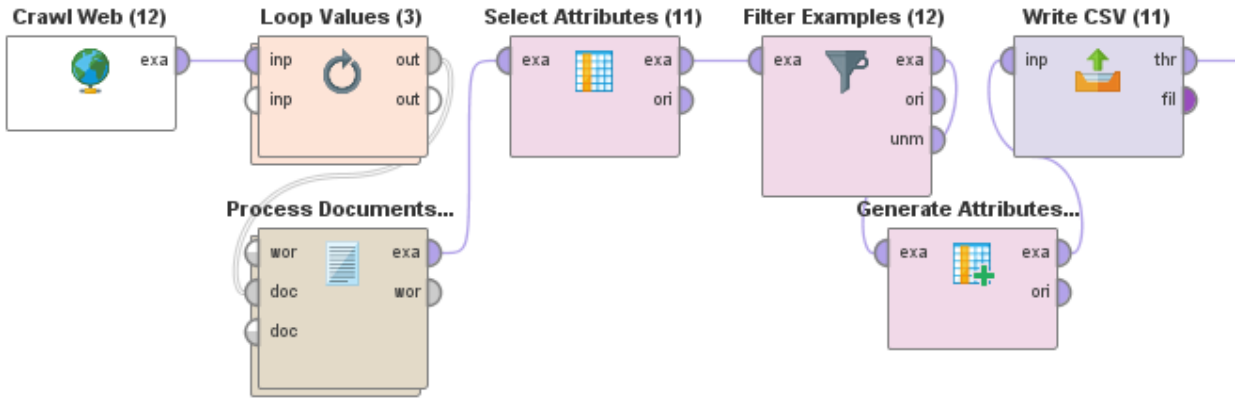
Τέλος, γράφουμε τα δεδομένα στο CSV αρχείο ώστε να συνεχίσει η διαδικασία. Οι παραπάνω Operators βρίσκονται μέσα σε ένα Subprocess οπότε ως τελικό αποτέλεσμα έχουμε το παρακάτω σχήμα.



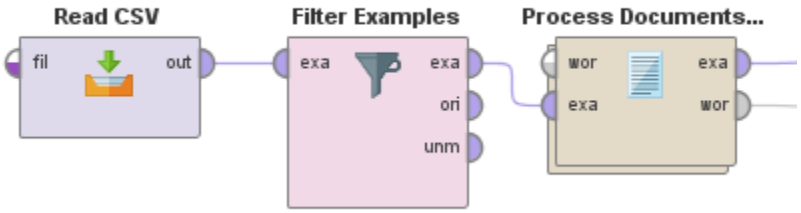
Εφόσον έχει τελειώσει το Initialize Crawling, ακολουθεί η διαδικασία του Main Crawling. Στη συγκεκριμένη περίπτωση, έχουμε δημιουργήσει ένα CSV το οποίο περιέχει όλα τα links με την κάθε μία χώρα που θέλουμε να αντλήσουμε τα δεδομένα. Χρησιμοποιώντας Operator Loop Values τρέχουμε ουσιαστικά την διαδικασία που περιγράψαμε παραπάνω, με την μόνη διαφορά ότι το κάνει για κάθε τιμή που έχει η Loop δηλαδή για κάθε Link μιας χώρας, κάνοντας append to file στο CSV.



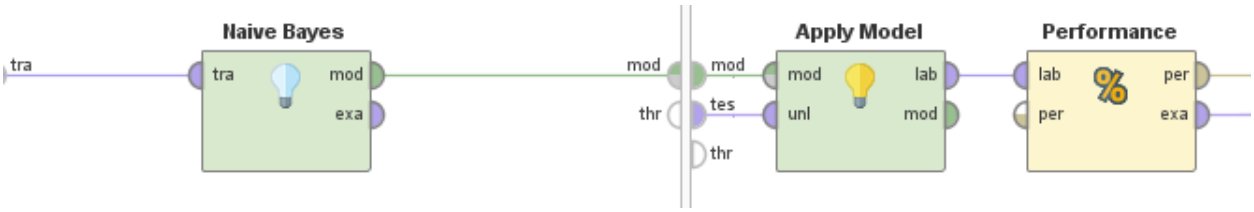
Παρακάτω βλέπουμε την δομή της Loop Values που περιγράψαμε παραπάνω. Όπως βλέπουμε λειτουργεί με ακριβώς τον ίδιο τρόπο, τον οποίο περιγράψαμε στο Initialize Crawling.



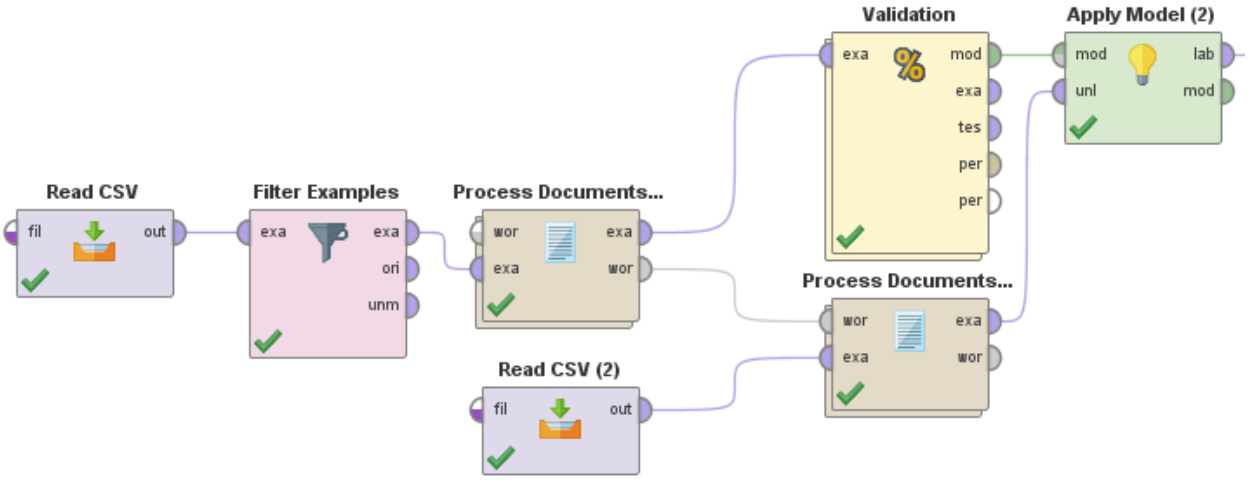
Αφού λοιπόν έχει κάνει το Scrapping από τις σελίδες και το έχει αποθηκεύσει στο CSV αρχείο, απενεργοποιούμε τα δυο συγκεκριμένα subprocesses ώστε να τρέξουμε το train και test κομμάτι της υποεργασίας. Αρχικά, χρησιμοποιούμε έναν Operator Read CSV για να διαβάσουμε τα δεδομένα μας για το train κομμάτι περνώντας τα δεδομένα από ένα Filter Examples ώστε να μην έχουμε Missing Values και στη συνέχεια δημιουργούμε Vectors με το Process Documents from Data.



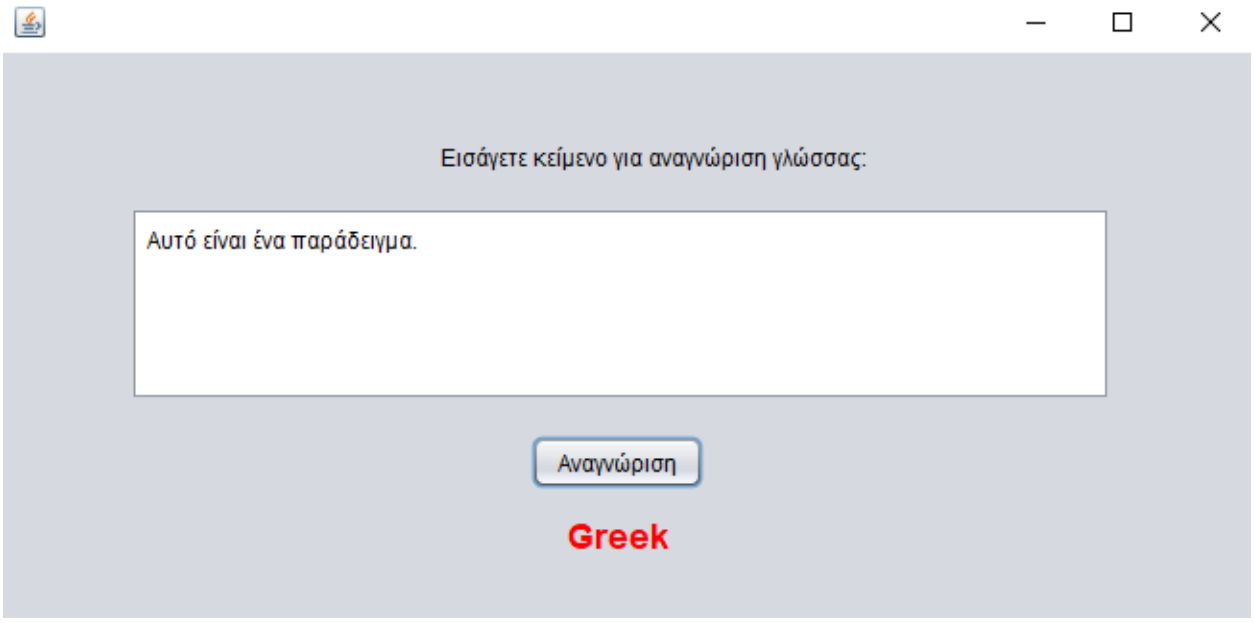
Στη συνέχεια, χρησιμοποιούμε Operator Validation για να κάνουμε train με τα δεδομένα, χρησιμοποιώντας εσωτερικά Naive Bayes για καλύτερο αποτέλεσμα.



Παράλληλα, διαβάζουμε ένα παράδειγμα κειμένου από ένα CSV δημιουργώντας και σε αυτό Vectors με τον ίδιο ακριβώς τρόπο, ώστε να έχουμε το Test κομμάτι που χρειαζόμαστε. Τέλος, χρησιμοποιούμε Apply Model για να παραγάγουμε το αποτέλεσμα της πρόβλεψης.

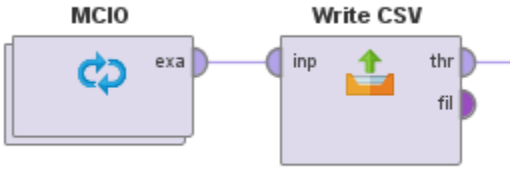


Παρακάτω, βλέπουμε το τελικό αποτέλεσμα μέσω της εφαρμογής η οποία υλοποιήθηκε σε Java μέσω του περιβάλλον ανάπτυξης NetBeans.



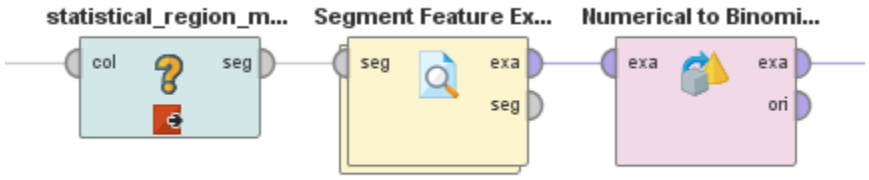
Υποεργασία 2η

Η συγκεκριμένη υποεργασία ασχολείται με το Image Mining, συγκεκριμένα με την αναγνώριση ποδοσφαιρικών γηπέδων από δορυφορικές εικόνες. Αρχικά χρειαστήκαμε ένα Custom Extension το οποίο υπάρχει ως jar αρχείο και το εισαγάγαμε στο Rapid Miner. Το γενικό πλάνο περιέχει έναν Multiple Color Image Opener(MCIO) το οποίο αφού αναλύσει όλες τις εικόνες, τις οποίες τις παίρνει από έναν φάκελο τον οποίο ορίζουμε στα χαρακτηριστικά του operator, γράφει τα δεδομένα σε ένα CSV αρχείο.



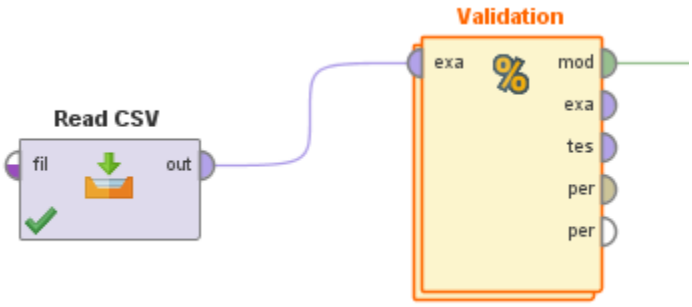
Ο Operator MCIO λοιπόν, περιέχει έναν Statistical Region Merging Operator ο οποίος ταξινομεί την εικόνα σε κομμάτια, χρησιμοποιώντας breakpoint αφού ολοκληρώσει την διαδικασία του, ώστε να επιλέξουμε το κομμάτι που μας ενδιαφέρει πάνω στην εικόνα, θέτοντας την τιμή q του στο 204, στο οποίο επιλέγουμε σε πόσα κομμάτια να χωρίσει την εικόνα. Έπειτα, ο Operator Segment Feature Extractor αντλεί τα δεδομένα από την επεξεργασία της εικόνας δημιουργώντας ένα ExampleSet.

Τέλος, χρησιμοποιούμε έναν Operator Numerical to Binominal ώστε να αλλάξουμε τα δεδομένα από αριθμητικά σε διωνυμικά.

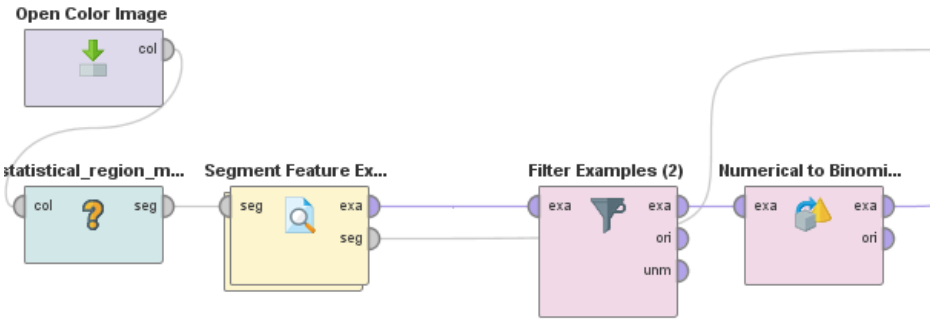


Αφού λοιπόν έχουμε δημιουργήσει τα δεδομένα που θα χρησιμοποιήσουμε, προχωρήσαμε στην υλοποίηση του Train και Test κομμάτι της υποεργασίας. Το γενικό πλάνο του Train και Test περιέχει, ένα Subprocess για το Train, ένα για το Test, ένα Apply Model το οποίο θα μας παραγάγει το αποτέλεσμα της πρόβλεψης, ένα Filter Examples το οποίο μας βοηθά να επιλέξουμε το σωστό μέγεθος περιοχής και ένα Segment Filter by ExampleSet για να μας παραγάγει στην οθόνη την εικόνα την οποία θα έχει σχεδιασμένο με κόκκινο περίγραμμα, το αποτέλεσμα.

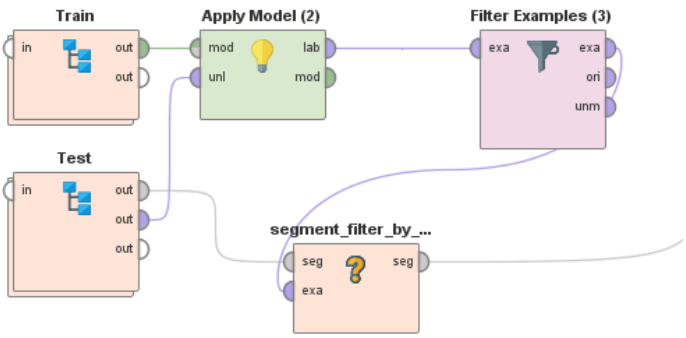
Η διαδικασία Train αρχικά, ξεκινά διαβάζοντας τα δεδομένα των γηπέδων τα οποία έχουν αποθηκευτεί σε ένα CSV, μέσω του Operator Read CSV. Στη συνέχεια, κάνει Validation για να παραγάγει το μοντέλο πρόβλεψης, το οποίο θα πάρει το Apply Model.



Η διαδικασία Test θα περιέχει όλη την ανάλυση της εικόνας που θέλουμε να δοκιμάσουμε. Ξεκινώντας, χρησιμοποιούμε έναν Operator Open Color Image (ο οποίος είναι αντίστοιχος του MCIO απλά για μόνο μία εικόνα) και εφαρμόζουμε ακριβώς την ίδια λογική την οποία εφαρμόσαμε στη δημιουργία των δεδομένων. Αυτό παρατηρείται εύκολα, στην παρακάτω εικόνα.

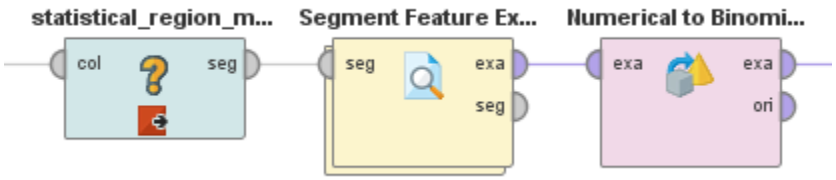


Εφόσον έχουμε δημιουργήσει και το αποτέλεσμα του Test Process το εφαρμόζουμε στο Apply Model για να παραγάγουμε αποτέλεσμα πρόβλεψης. Παράλληλα, έχουμε πάρει από το Segment Feature Extractor την εικόνα που πήραμε στο Test και την στέλνουμε σε Segment Filter μαζί με το αποτέλεσμα της πρόβλεψης ώστε να σημειωθεί στην εικόνα με κόκκινη γραμμή τα σημεία που έχει βρει ως γήπεδα. Παρακάτω βλέπουμε το γενικό πλάνο της διαδικασίας.

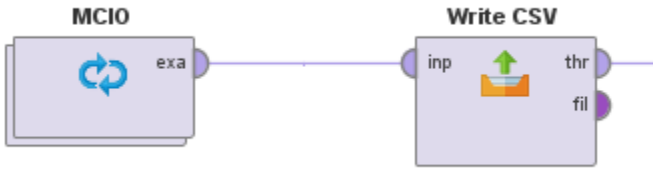


Υποεργασία 3η

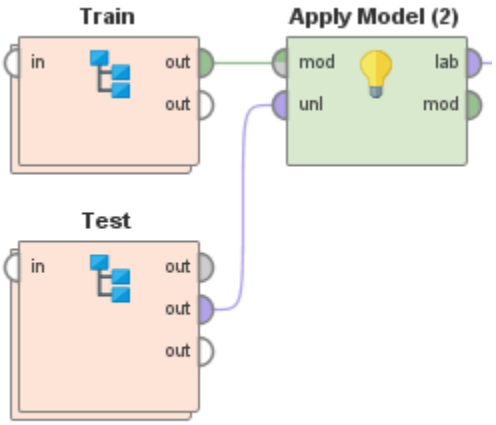
Η τρίτη υποεργασία ασχολείται με την πρόβλεψη αναγνώριση πίνακα ενός ζωγράφου. Η διαδικασία η οποία θα παράγει τα δεδομένα, γίνεται ακριβώς με τον ίδιο τρόπο όπως η υποεργασία 2. Συγκεκριμένα, έχουμε πάλι έναν Multiple Color Image Opener, ο οποίος περιέχει αρχικά, Statistical Region Merging, με την διαφορά ότι στο q αυτή την φορά θα δώσουμε την τιμή 1 ώστε να πάρει ως σχήμα όλη την εικόνα. Επίσης, περιέχει Segment Feature Extractor και Numerical to Binominal, όπως ακριβώς στην υποεργασία 2.



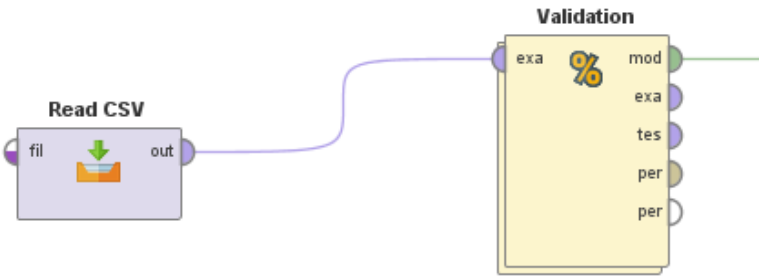
Αφού ολοκληρώσει την επεξεργασία των εικόνων, γραφεί τα δεδομένα σε ένα CSV ώστε να τα χρησιμοποιήσουμε για την δημιουργία μοντέλου.



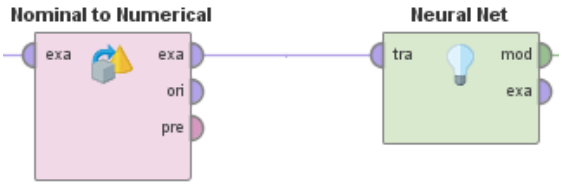
Παρακάτω, θα αναλύσουμε την διαδικασία παραγωγής αποτελέσματος πρόβλεψης. Σε αρχικό στάδιο, έχουμε μια διαδικασία εκπαίδευσης (Train) και μια διαδικασία Test, οι οποίες καταλήγουν σε Apply Model ώστε να μας παράγει το αποτέλεσμα πρόβλεψης.



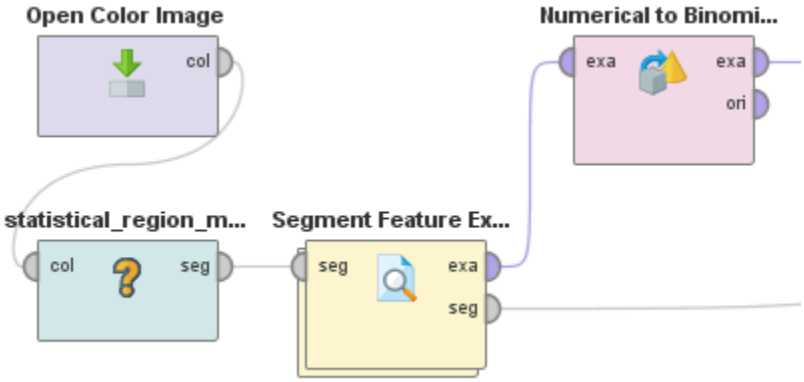
Η διαδικασία Train εφαρμόζεται με ακριβώς τον ίδιο τρόπο με την υποεργασία 2. Δηλαδή, διαβάζουμε ένα CSV αρχείο και κάνουμε Validation. Η μόνη βασική διαφορά είναι ότι, χρησιμοποιούμε Νευρωνικά Δίκτυα για καλύτερο αποτέλεσμα στο Validation.



Εσωτερικό Operator Validation



Στη διαδικασία Test, όπως παρατηρήσουμε στην παρακάτω εικόνα έχουμε πάλι, έναν Open Color Image, ένα Statistical Region Merging με q=1, ένα Segment Feature Extractor και ένα Numerical to Binominal Operator.



Τέλος, το αποτέλεσμα που παράγει η διαδικασία Test καταλήγει όπως αναφέραμε παραπάνω στο Apply Model, το οποίο μας δίνει το αποτέλεσμα της πρόβλεψης.