



**ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**ΤΕΧΝΟΛΟΓΙΕΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΣΤΗΝ ΙΑΤΡΙΚΗ ΚΑΙ ΤΗ ΒΙΟΛΟΓΙΑ**

**Αναγνώριση Προτύπων**

## **Πρακτική Άσκηση 2**

Επιβλέπων: **Περαντώνης Σταύρος**, Διευθυντής Ερευνών, ΕΚΕΦΕ-Δημόκριτος

**ΑΘΗΝΑ**

**ΜΑΡΤΙΟΣ 2013**



## **Αναγνώριση Προτύπων**

### **Πρακτική Άσκηση 2**

**Κωνσταντόπουλος Γ. Δημήτριος**  
**Μπεγέτης Ι. Νικόλαος**

ΑΜ: ΠΙΒ0112 / ΠΙΒ0111

#### **ΕΠΙΒΛΕΠΩΝ :**

**Περαντώνης Σταύρος** , Διευθυντής Ερευνών, ΕΚΕΦΕ-Δημόκριτος



# ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

<b>1</b>	<b>Πρακτική Άσκηση 2</b>	<b>11</b>
1.1	Θέμα 1 . . . . .	12
1.2	Θέμα 2 . . . . .	23
<b>2</b>	<b>Σύντομα Αποτελέσματα Ταξινομητών</b>	<b>25</b>
2.1	Iris Plant DataBase . . . . .	25
2.2	Promoter Gene Sequence DataBase . . . . .	29
<b>3</b>	<b>Υλοποιήσεις σε Matlab</b>	<b>33</b>
3.1	Θέμα 2 . . . . .	33
<b>4</b>	<b>Ενδεικτικές Εικόνες</b>	<b>39</b>
<b>5</b>	<b>Κατακλείδα</b>	<b>49</b>



# ΛΙΣΤΑ ΕΙΚΟΝΩΝ

1.1	Iris plant Database loaded in WEKA . . . . .	13
1.2	Promoter Gene Sequence Database loaded in WEKA . . . . .	13
1.3	Iris parametrization for classification using NB . . . . .	14
1.4	Iris parametrization for classification using SVM RBF . . . . .	15
1.5	Iris parametrization for classification using SVM Polykernel . . . . .	15
1.6	Iris parametrization for classification using RBFnetworks . . . . .	16
1.7	Iris output of Naive Bayes classification . . . . .	17
1.8	Iris output of SVM classification using RBFkernel . . . . .	18
1.9	Iris output of SVM classification using Polykernel . . . . .	19
1.10	Iris output of RBFnetworks classification . . . . .	20
1.11	Gathered Iris output from all classifiers . . . . .	21
1.12	Gathered Iris output from all classifiers (continue) . . . . .	21
4.1	Iris plant Database loaded in WEKA . . . . .	39
4.2	Promoter Gene Sequence Database loaded in WEKA . . . . .	40
4.3	Iris parametrization for classification using NB . . . . .	40
4.4	Iris parametrization for classification using SVM RBF . . . . .	41
4.5	Iris parametrization for classification using SVM Polykernel . . . . .	41
4.6	Iris parametrization for classification using RBFnetworks . . . . .	42
4.7	Iris output of Naive Bayes classification . . . . .	43

4.8	Iris output of SVM classification using RBFkernel . . . . .	44
4.9	Iris output of SVM classification using Polykernel . . . . .	45
4.10	Iris output of RBFnetworks classification . . . . .	46
4.11	Gathered Iris output from all classifiers . . . . .	47
4.12	Gathered Iris output from all classifiers (continue) . . . . .	47







# ΚΕΦΑΛΑΙΟ 1

## Πρακτική Άσκηση 2

Η Αναγνώριση Προτύπων (Pattern Recognition) είναι μία επιστημονική περιοχή που έχει στόχο την απόδοση κάποιας τιμής ή διακριτικού στοιχείου σε εισαγόμενα δεδομένα. Οι άνθρωποι και τα άλλα όντα έχουν την ικανότητα να ταυτοποιούν πραγματικά δεδομένα χρησιμοποιώντας τις αισθήσεις τους και την αντιληπτική τους ικανότητα (cognition) προκειμένου να λάβουν τις κατάλληλες αποφάσεις ώστε να επιβιώσουν στο περιβάλλον τους.

Μία μηχανή, όπως ένας ηλεκτρονικός υπολογιστής, πρέπει να εκπαιδευθεί κατάλληλα ώστε να αναγνωρίζει πρότυπα (patterns) και να τα κατηγοριοποιεί αυτόματα σε κατηγορίες. Ανάλογα με την εφαρμογή γίνεται κατάταξη των αντικειμένων σε κλάσεις με τη βοήθεια αλγορίθμων ταξινόμησης.

Το ερευνητικό ενδιαφέρον για αυτά τα ζητήματα ξεκίνησε από τη δεκαετία του 1960, κατά την πρώτη περίοδο της ανάπτυξης της επιστήμης των υπολογιστών. Βασισμένη στο θεωρητικό υπόβαθρο που παρείχε η επιστήμη της Στατιστικής, η πρώιμη έρευνα επικεντρώθηκε στην ανάπτυξη θεωρητικών μεθόδων. Ήδη από το 1970 γίνονταν προσπάθειες για την καλύτερη κατεύθυνση των προσπαθειών και το 1976 ιδρύεται η Παγκόσμια Ένωση για την Αναγνώριση Προτύπων (IARP). Σε πολλά επιστημονικά πεδία αξιοποιούνται εφαρμογές της αναγνώρισης προτύπων, όπως στην Ιατρική (υποβοηθούμενη από Η/Υ διάγνωση, ανάλυση δεδομένων DNA και άλλες εφαρμογές της βιοπληροφορικής) και την επιστήμη υπολογιστών (υπολογιστική όραση, αναγνώριση χαρακτηρισμών ή φωνής, νευρωνικά δίκτυα, εξόρυξη δεδομένων και ανάκτηση γνώσης, τεχνητή νοημοσύνη και μηχανική μάθηση, συστήματα υποστήριξης αποφάσεων). Στον σύγχρονο κόσμο, πολλές βιομηχανικές εφαρμογές ενσωματώνουν ανάλογα συστήματα για την αποδοτική και αυτόματη επεξεργασία πληροφοριών<sup>1</sup>.

Στη δεύτερη εργασία, στα πλαίσια του μεταπτυχιακού μαθήματος της Αναγνώρισης Προτύπων, κληθήκαμε να χρησιμοποιήσουμε ταξινομητές όπως οι Naive Bayes, SVM και RBF network (δίκτυο ακτινικής βάσης) χρησιμοποιώντας το λογισμικό του weka για να κάνουμε χρήση της μεθόδου ten crossvalidation, για την αποτίμηση των αποτελεσμάτων. Επίσης, στο δεύτερο μέρος αυτής της εργασίας δημιουργήσαμε ένα νευρωνικό δίκτυο και χρησιμοποιήσαμε τους αλγόριθμους backpropagation, conjugate gradient και Levenberg-Marquardt.

---

<sup>1</sup>[http://el.wikipedia.org/wiki/Αναγνώριση\\_προτύπων](http://el.wikipedia.org/wiki/Αναγνώριση_προτύπων)

## 1.1 Θέμα 1

Στα πλαίσια της πρώτης Πρακτικής Άσκησης στο μάθημα της Αναγνώρισης Προτύπων, κληθήκαμε να χρησιμοποιήσουμε, να πειραματιστούμε και να εκτιμήσουμε τρεις ταξινομητές, (α) τον απλό ταξινομητή Bayes(Naive Bayes), (β) τον ταξινομητή SVM και (γ) το δίκτυο ακτινικής βάσης RBF network. Για την διεξαγωγή των πειραμάτων κάναμε χρήση του λογισμικού του weka<sup>2</sup> και χρησιμοποιήσαμε την μέθοδο ten crossvalidation, για την αποτίμηση των αποτελεσμάτων. Εφαρμόσαμε τους παραπάνω ταξινομητές πάνω σε 2 πολύ γνωστά από την βιβλιογραφία datasets, το Iris plant Database<sup>3</sup> (ταξινόμηση φυτών Iris σε τρία είδη) και το Promoter Gene Sequence Database<sup>4</sup> (ταξινόμηση ακολουθιών γονιδίων).

Λίγα λόγια για το λογισμικό που χρησιμοποιήσαμε. Το WEKA αποτελεί ένα πακέτο λογισμικού υλοποιημένο σε Java, το οποίο αναπτύχθηκε στο πανεπιστήμιο "Waikato" της Νέας Ζηλανδίας. Το Weka αποτελεί λογοπαίγνιο με το ομώνυμο σπάνιο πτηνό που μπορεί να βρεθεί στα νησιά της Νέας Ζηλανδίας και για αυτό το λόγο αποτελεί και το λογότυπο του λογισμικού. Στην πραγματικότητα όμως το WEKA αποτελεί αρκτικόλεξο του Waikato Environment for Knowledge Analysis. Μέχρι στιγμής, το WEKA έχει διανεμηθεί σε τρεις μακροχρόνιες εκδόσεις και παρέχει πολλούς διαφορετικούς αλγορίθμους για μηχανική μάθηση, εξόρυξη δεδομένων, αναγνώριση προτύπων και νευρωνικά δίκτυα. Τέλος, μαζί με τη διανομή του WEKA διανέμεται και το GUI περιβάλλον, το οποίο και μας ζητήθηκε να χρησιμοποιήσουμε στην άσκηση μας, μιας και είναι και πιο user friendly.

Ακολουθώντας λοιπόν τις οδηγίες της εκφώνησης για την πρώτη εργασία, εγκαταστήσαμε και μελετήσαμε το λογισμικό και κατεβάσαμε από το online UCI Machine Learning Repository τα datasets που θα χρησιμοποιήσουμε, τα οποία αναφέραμε προηγουμένως.

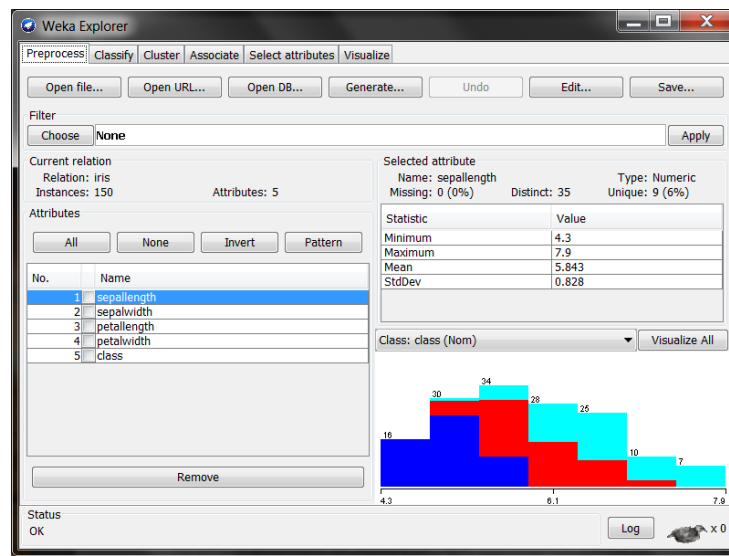
Για να μπορέσουμε να κάνουμε χρήση στο WEKA αυτών των datasets μετατρέψαμε αρχικά τα ονόματα των βάσεων του dataset που αφορούσε τους promoters σε αριθμητικές μεταβλητές (με τη μορφή a=0, c=1, g=2, t=3) όπως προτείνεται και στην εκφώνηση της εργασίας. Στη συνέχεια, μετατρέψαμε τα αρχεία τύπου .data, στη δομή που πρέπει να συμβιβαστούμε ώστε να εξάγουμε δεδομένα τύπου .arff, που δίνονται ως είσοδος στο WEKA.

---

<sup>2</sup><http://www.cs.waikato.ac.nz/ml/weka/>

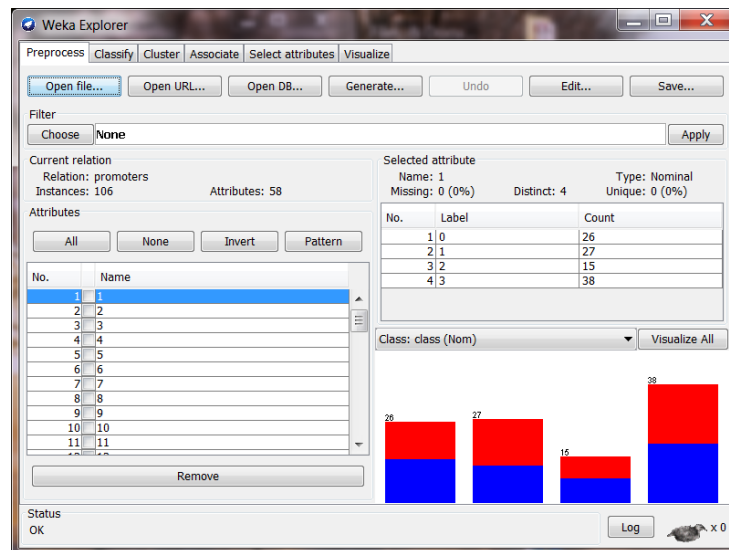
<sup>3</sup><ftp://ftp.ics.uci.edu/pub/machine-learning-databases/iris/>

<sup>4</sup><ftp://ftp.ics.uci.edu/pub/machine-learning-databases/molecular-biology/promoter-gene-sequences/>



Σχήμα 1.1: *Iris plant Database loaded in WEKA using ARFF file format*

Έχοντας λοιπόν πλέον, έτοιμα τα input datasets εκκινήσαμε το WEKA και επιλέξαμε από τα "Applications" τον "Explorer". Στο αρχικό tab που εμφανίστηκε στο παράθυρο του λογισμικού φορτώσαμε διαδοχικά τα αρχεία ARFF (Σχήμα 1.1, 1.2), όπως υποδείχθηκε από την εκφώνηση, και στη συνέχεια προχωρήσαμε στο κυρίως θέμα της πρώτης εργασίας που αφορά την ταξινόμηση των δύο προβλημάτων με χρήση των ταξινομητών Naive Bayes, SVM (αναφέρεται ως SMO στις επιλογές που παρέχει το πρόγραμμα), και δίκτυο ακτινικής βάσης (RBFnetwork). Η αποτίμηση των αποτελεσμάτων για όλους τους ταξινομητές διεκπεραιώθηκε με τη χρήση της μεθόδου tenfold crossvalidation.

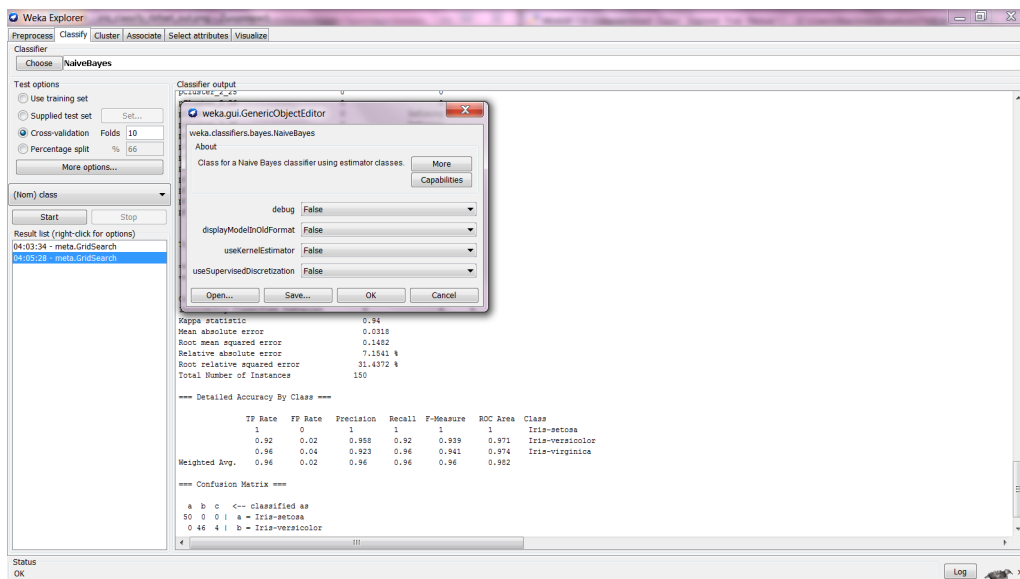


Σχήμα 1.2: *Promoter Gene Sequence Database loaded in WEKA using ARFF file format*

Η λειτουργία της μεθόδου *tenfold crossvalidation* είναι να χωρίζει τα δοθέντα δεδομένα σε 10 ισόποσα μέρη, *folds*. Σε κάθε βήμα του ελέγχου, τα δεδομένα των 9 *folds*, *training fold*, θα αποτελούν τα δεδομένα εκπαίδευσης του ταξινομητή που υλοποιείται, ενώ τα δεδομένα του 1 *fold*, *validation fold*, θα αποτελούν αυτά που είναι προς ταξινόμηση. Πραγματοποιώντας επαναληπτικά την διαδικασία, όλα τα *folds*, με την σειρά θα έχουν από μία φορά τον ρόλο του *validation fold*, και 9 φορές θα αποτελούν μέρος του *training fold*. Στη συνέχεια, από τα αποτελέσματα που θα πάρουμε από τα πειράματα χρησιμοποιώντας την παραπάνω μέθοδο, θα μπορούμε να αξιολογήσουμε τους ταξινομητές. Η αξιολόγηση των ταξινομητών γίνεται με βάση την επίδοσή τους, και πιο συγκεκριμένα με το ποσοστό πραγματοποίησης ορθής ταξινόμησης. Μετά από κάθε υλοποίηση των ταξινομητών, συγκρίνονται τα αποτελέσματα ταξινόμησης που προέκυψαν με αυτά που είναι καταχωρημένα στα *dataset* που έχουν δοθεί και έτσι υπολογίζεται αυτόματα από το λογισμικό *weka* το παρακάτω ποσοστό.

$$\frac{Numberofgoodclassifications}{Numberofclassifications} \times 100$$

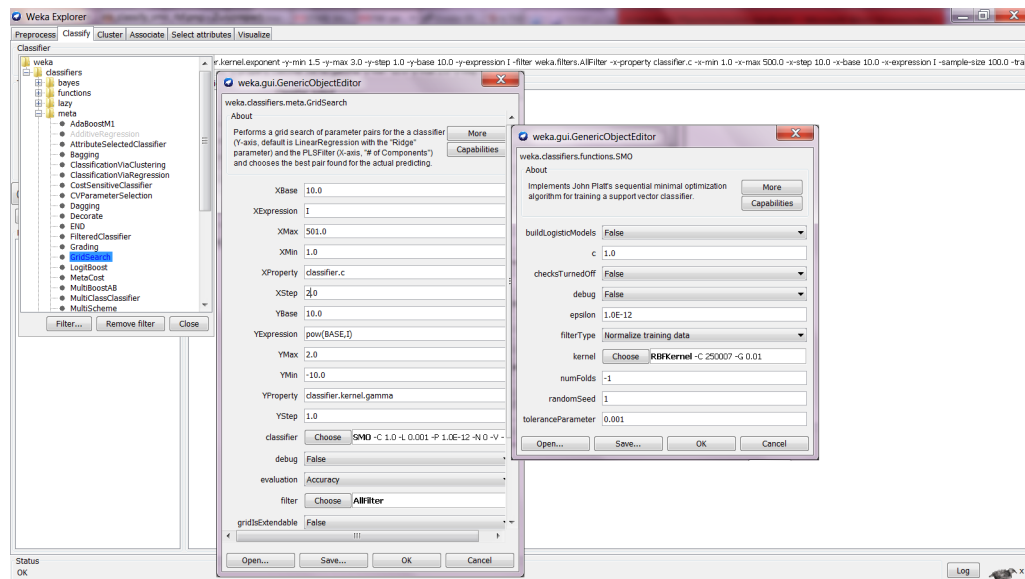
Συνεπώς, για να επιτύχουμε τα καλύτερα δυνατά αποτελέσματα, δηλαδή την καλύτερη δυνατή γενίκευτική ικανότητα, όπως μετριέται από την ποσότητα "*Correctly classified instances*", για τους ταξινομητές *Naive Bayes*, *SVM* και *RBFnetwork* είναι πιθανό να πρέπει να παραμετροποιηθούν κατάλληλα πριν τη χρήση τους. Πράγματι, ενώ τον ταξινομητή *Naive Bayes* δεν τον αλλάξαμε (Σχήμα 1.3), στους άλλους δύο ταξινομητές χρειάστηκε να γίνουν οι κατάλληλες παραμετροποιήσεις.



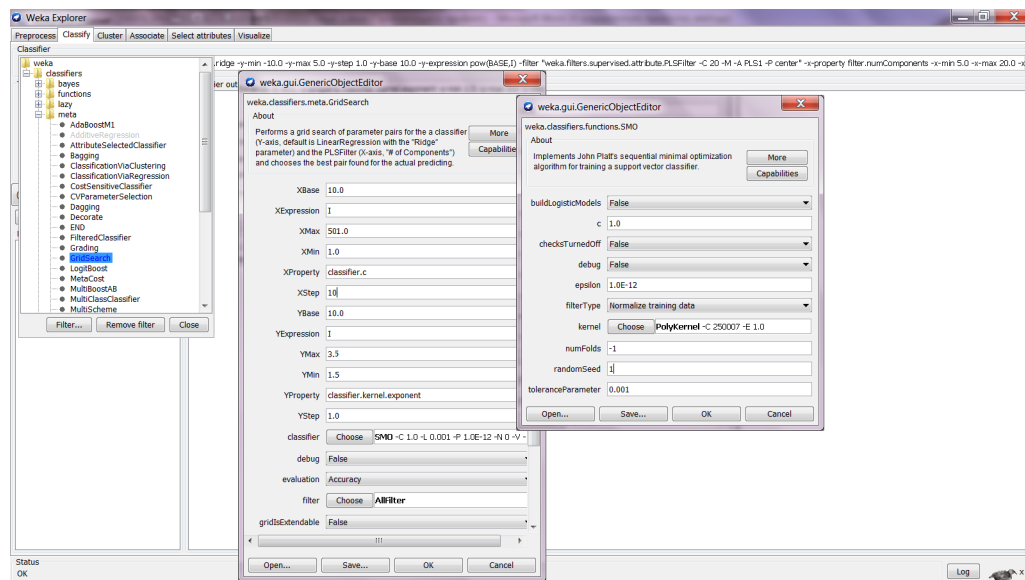
Σχήμα 1.3: *Iris parametrization for classification using Naive Bayes*

Πιο συγκεκριμένα, όπως ζητείται και από την εκφώνηση στην περίπτωση του SMO οι βασικές παράμετροι τις οποίες και παραμετροποιήσαμε είναι ο τύπος του πυρήνα (γραμμικός, πολυωνυμικός, RBF), καθώς και η σταθερά *C* που ρυθμίζει την έκταση του περιθωρίου. Όσον αφορά

τους πυρήνες, όπως διασαφηνίστηκε ο γραμμικός πυρήνας προκύπτει από τον πολυωνυμικό (Polykernel) με επιλογή της παραμέτρου  $E$  ίση με 1. Εμείς πειραματιστήκαμε και με καθαρά πολυωνυμικούς πυρήνες με  $E$  ίσο με 2 και 3. Επίσης, στην περίπτωση του ΡΒΦ πυρήνα, η παράμετρος  $\Gamma$  αναφέρθηκε ότι ρυθμίζει την τυπική απόκλιση των γκαουσιανών συναρτήσεων και αυτή την ορίσαμε ως 0.01. Το Σχήμα 1.4, δείχνει την παραμετροποίηση με τη χρήση του RBF kernel, ενώ το σχήμα 1.5 δείχνει την παραμετροποίηση με χρήση του Polykernel.

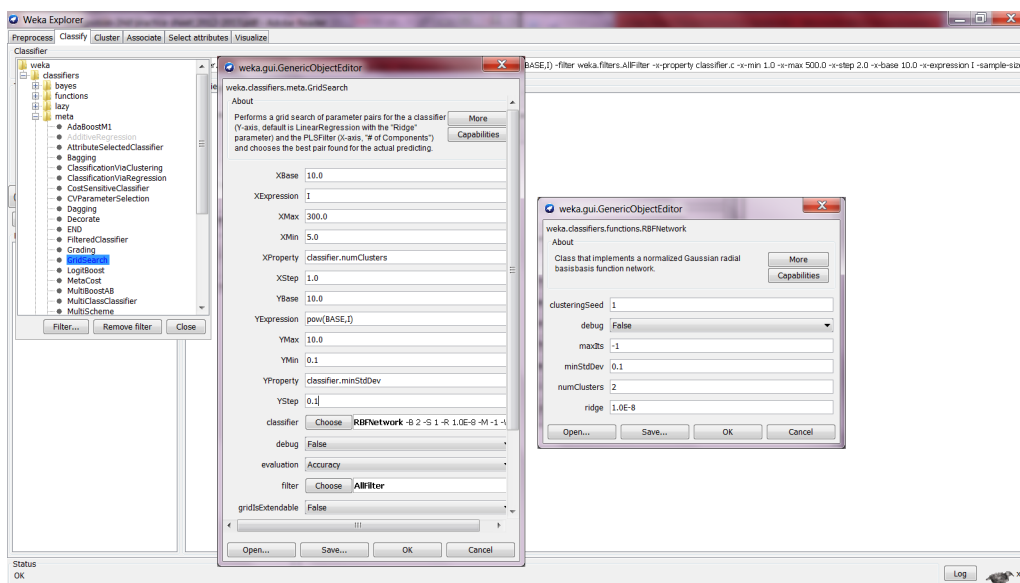


Σχήμα 1.4: *Iris parametrization for classification using SVM RBF*



Σχήμα 1.5: *Iris parametrization for classification using SVM Polykernel*

Στην περίπτωση του RBFnetwork, όπως αναφέρεται και στην εκφώνηση οι βασικές παράμετροι είναι το πλήθος των ενδιάμεσων νευρώνων (numClusters) και η ελάχιστη τυπική απόκλιση των γκαουσιανών συναρτήσεων (minStdDev), τα οποία παραμετροποιήσαμε ανάλογα όπως φαίνεται και στο Σχήμα 1.6.



Σχήμα 1.6: *Iris parametrization for classification using RBFnetworks*

Στη συνέχεια, έχοντας κάνει την παραπάνω παραμετροποίηση ορίζουμε στο παράθυρο του explorer ό,τι επιθυμούμε να γίνει crossvalidation με 10 folds και πατάμε start. Αργότερα, όταν τελειώσει το build του μοντέλου προκύπτουν τα αποτελέσματα, όπως φαίνεται ενδεικτικά στα Σχήματα 1.7, 1.8, 1.9 και 1.10, για τους ταξινομητές Naive Bayes, SVM-RBFkernel SVM-Polykernel και RBFnetworks αντίστοιχα.



```

34 petallength
35     mean          1.4694          4.2452          5.5516
36     std. dev.      0.1782          0.4712          0.5529
37     weight sum      50             50             50
38     precision      0.1405          0.1405          0.1405
39
40 petalwidth
41     mean           0.2743          1.3097          2.0343
42     std. dev.      0.1096          0.1915          0.2646
43     weight sum      50             50             50
44     precision      0.1143          0.1143          0.1143
45
46
47
48 Time taken to build model: 0.01 seconds
49
50 === Stratified cross-validation ===
51 === Summary ===
52
53 Correctly Classified Instances      144          96      %
54 Incorrectly Classified Instances     6           4      %
55 Kappa statistic                     0.94
56 Mean absolute error                 0.0342
57 Root mean squared error             0.155
58 Relative absolute error             7.6997 %
59 Root relative squared error        32.8794 %
60 Total Number of Instances          150
61
62 === Detailed Accuracy By Class ===
63
64      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
65      1         0         1          1         1          1      Iris-setosa
66      0.96      0.04      0.923      0.96      0.941      0.992    Iris-versicolor
67      0.92      0.02      0.958      0.92      0.939      0.992    Iris-virginica
68 Weighted Avg.  0.96      0.02      0.96      0.96      0.96      0.994
69
70 === Confusion Matrix ===
71
72  a  b  c  <-- classified as
73  50  0  0 | a = Iris-setosa
74   0 48  2 | b = Iris-versicolor
75   0  4 46 | c = Iris-virginica
76
77

```

Normal te: length : 2865 lines : 77 Ln : 77 Col : 1 Sel : 0 Dos\Windows ANSI INS

Σχήμα 1.7: *Iris output of Naive Bayes classification*

```

88 - 325 + <0.555556 0.541667 0.627119 0.625 > + X]
89 + 325 + <0.472222 0.416667 0.644068 0.708333 > + X]
90 - 325 + <0.361111 0.416667 0.59322 0.583333 > + X]
91 - 325 + <0.472222 0.375 0.59322 0.583333 > + X]
92 - 325 + <0.611111 0.333333 0.610169 0.583333 > + X]
93 + 0.299
94
95 Number of support vectors: 29
96
97 Number of kernel evaluations: 3206 (83.448% cached)
98
99
100
101 Time taken to build model: 862.97 seconds
102
103 === Stratified cross-validation ===
104 === Summary ===
105
106 Correctly Classified Instances 143 95.3333 %
107 Incorrectly Classified Instances 7 4.6667 %
108 Kappa statistic 0.93
109 Mean absolute error 0.2326
110 Root mean squared error 0.2906
111 Relative absolute error 52.3333 %
112 Root relative squared error 61.6441 %
113 Total Number of Instances 150
114
115 === Detailed Accuracy By Class ===
116
117 TP Rate FP Rate Precision Recall F-Measure ROC Area Class
118 1 0 1 1 1 1 Iris-setosa
119 0.9 0.02 0.957 0.9 0.928 0.945 Iris-versicolor
120 0.96 0.05 0.906 0.96 0.932 0.963 Iris-virginica
121 Weighted Avg. 0.953 0.023 0.954 0.953 0.953 0.969
122
123 === Confusion Matrix ===
124
125 a b c <-- classified as
126 50 0 0 | a = Iris-setosa
127 0 45 5 | b = Iris-versicolor
128 0 2 48 | c = Iris-virginica
129
130

```

Normal te: length : 5250 lines : 130 Ln : 1 Col : 1 Sel : 0 Dos\Windows ANSI INS

Σχήμα 1.8: Iris output of SVM classification using RBFkernel

```

74 + 23.363 * <0.166667 0.208333 0.593222 0.666667 > * X]
75 + 10.6215 * <0.805556 0.416667 0.813559 0.625 > * X]
76 + 31 * <0.472222 0.416667 0.644068 0.708333 > * X]
77 - 5.5186 * <0.361111 0.416667 0.593222 0.583333 > * X]
78 - 7.227 * <0.611111 0.333333 0.610169 0.583333 > * X]
79 - 7.882
80
81 Number of support vectors: 16
82
83 Number of kernel evaluations: 2495 (80.74% cached)
84
85
86
87 Time taken to build model: 15.57 seconds
88
89 === Stratified cross-validation ===
90 === Summary ===
91
92 Correctly Classified Instances      144      96      %
93 Incorrectly Classified Instances    6        4      %
94 Kappa statistic                    0.94
95 Mean absolute error                0.2311
96 Root mean squared error            0.288
97 Relative absolute error            52      %
98 Root relative squared error        61.101 %
99 Total Number of Instances          150
100
101 === Detailed Accuracy By Class ===
102
103      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
104      1        0        1          1        1          1      Iris-setosa
105      0.9      0.01      0.978      0.9      0.938      0.945    Iris-versicolor
106      0.98      0.05      0.907      0.98      0.942      0.971    Iris-virginica
107 Weighted Avg.  0.96      0.02      0.962      0.96      0.96      0.972
108
109 === Confusion Matrix ===
110
111  a  b  c  <-- classified as
112  50  0  0 | a = Iris-setosa
113  0  45  5 | b = Iris-versicolor
114  0  1  49 | c = Iris-virginica
115
116

```

Normal te: length: 4375 lines: 116 Ln: 1 Col: 1 Sel: 0 Dos\Windows ANSI INS

Σχήμα 1.9: Iris output of SVM classification using Polykernel

```

252 pCluster_2_25      0      0
253 pCluster_2_26      0      0
254 pCluster_2_27      0      Infinity
255 pCluster_2_28      0      Infinity
256 pCluster_2_29      0      Infinity
257 pCluster_2_30      0      0
258 pCluster_2_31      0      Infinity
259 pCluster_2_32      0      0
260 pCluster_2_33      0      Infinity
261 pCluster_2_34      0      0
262 pCluster_2_35      1.0649    1.1345
263 pCluster_2_36      0      0
264
265
266 Time taken to build model: 1663.5 seconds
267
268 === Stratified cross-validation ===
269 === Summary ===
270
271 Correctly Classified Instances      144      96      %
272 Incorrectly Classified Instances     6      4      %
273 Kappa statistic                     0.94
274 Mean absolute error                 0.0218
275 Root mean squared error             0.1482
276 Relative absolute error             7.1541 %
277 Root relative squared error        31.4372 %
278 Total Number of Instances          150
279
280 === Detailed Accuracy By Class ===
281
282      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
283      1      0      1      1      1      1      Iris-setosa
284      0.92    0.02    0.958    0.92    0.939    0.971    Iris-versicolor
285      0.96    0.04    0.923    0.96    0.941    0.974    Iris-virginica
286 Weighted Avg.  0.96    0.02    0.96    0.96    0.96    0.982
287
288 === Confusion Matrix ===
289
290      a  b  c  <-- classified as
291  50  0  0 | a = Iris-setosa
292   0 46  4 | b = Iris-versicolor
293   0  2 48 | c = Iris-virginica
294
295

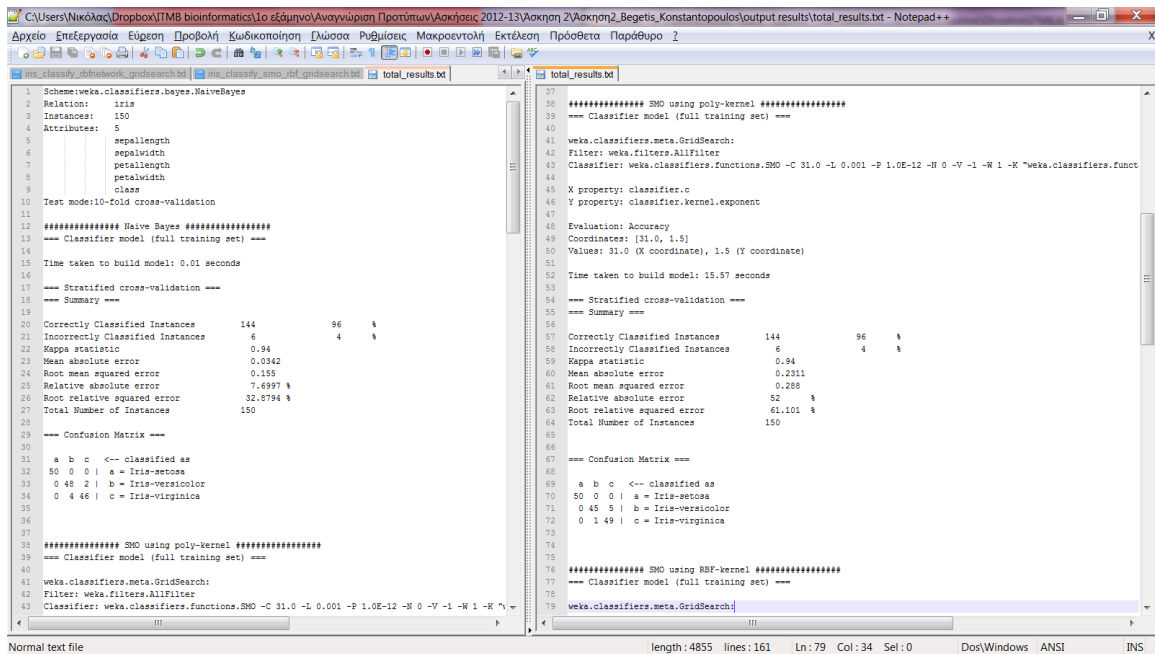
```

Normal te: length: 16398 lines: 295 Ln: 295 Col: 1 Sel: 0 Dos\Windows ANSI INS

Σχήμα 1.10: *Iris output of RBFnetworks classification*

Στα αρχεία total\_results\_iris.txt και total\_results\_promoters.txt έχουμε μαζέψει όλα τα αποτελέσματα από όλους τους classifiers για κάθε ένα από τα datasets. Τα σχήματα 1.11 και 1.12 δείχνουν τα αποτελέσματα από το πρώτο αρχείο.

## Αναγνώριση Προτύπων: Πρακτική Άσκηση 2



```
1 Schema: weka.classifiers.bayes.NaiveBayes
2 Relation: iris
3 Instances: 150
4 Attributes: 5
5   sepalwidth
6   sepalwidth
7   petalwidth
8   petalwidth
9   class
10 Test mode: 10-fold cross-validation
11
12 ===== Naive Bayes =====
13 --- Classifier model (full training set) ---
14
15 Time taken to build model: 0.01 seconds
16
17 --- Stratified cross-validation ---
18 --- Summary ---
19
20 Correctly Classified Instances      144      96 %
21 Incorrectly Classified Instances      6      4 %
22 Kappa statistic                    0.94
23 Mean absolute error                 0.0342
24 Root mean squared error             0.135
25 Relative absolute error             7.4997 %
26 Root relative squared error         32.8794 %
27 Total Number of Instances          150
28
29 --- Confusion Matrix ---
30
31 a b c <-- classified as
32 50 0 0 | a = Iris-setosa
33 0 45 2 | b = Iris-versicolour
34 0 4 46 | c = Iris-virginica
35
36
37
38 ===== SMO using poly-kernel =====
39 --- Classifier model (full training set) ---
40
41 weka.classifiers.meta.GridSearch:
42 Filter: weka.filters.AllFilter
43 Classifier: weka.classifiers.functions.SMO -C 31.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "
44
45 X property: classifier.c
46 Y property: classifier.kernel.exponent
47
48 Evaluation: Accuracy
49 Coordinates: [31.0, 1.5]
50 Values: 31.0 (X coordinate), 1.5 (Y coordinate)
51
52 Time taken to build model: 15.57 seconds
53
54 --- Stratified cross-validation ---
55 --- Summary ---
56
57 Correctly Classified Instances      144      96 %
58 Incorrectly Classified Instances      6      4 %
59 Kappa statistic                    0.94
60 Mean absolute error                 0.2311
61 Root mean squared error             0.288
62 Relative absolute error             52 %
63 Root relative squared error         61.101 %
64 Total Number of Instances          150
65
66 --- Confusion Matrix ---
67
68 a b c <-- classified as
69 50 0 0 | a = Iris-setosa
70 0 45 5 | b = Iris-versicolour
71 0 1 49 | c = Iris-virginica
72
73
74
75 ===== SMO using RBF-kernel =====
76 --- Classifier model (full training set) ---
77
78 weka.classifiers.meta.GridSearch:
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
255
```

μητές για να ταξινομήσουν τα αποτελέσματα ήταν 0.01 δ., 15.57 δ., 862.97 δ. και 1663.5 δ., αντίστοιχα για τους Naive Bayes, SVM -polykernel, SVM -rbfkernel, και δίκτυο ακτινικής βάσης (RBFnetwork). Βέβαια τα 3 τελευταία, όπως ζητήθηκε τα τρέξαμε σε όλες τις πιθανές τιμές χρησιμοποιώντας την GridSearch ταξινόμηση.

Επιπλέον, από το όνφυσιον Ματριξ (αναλύουμε των 3 ταξινομητών που σκόραραν υψηλότερα) διαπιστώσαμε ότι 50/50 δείγματα της κλάσης a (Iris-setosa) έχουν ταξινομηθεί σωστά στην κλάση αυτή, 48 της κλάσης b (Iris-versicolor) έχουν ταξινομηθεί σωστά και 2 λανθασμένα στη κλάση c (Iris-virginica) ενώ 4 πρότυπα της κλάσης c έχουν λανθασμένα ταξινομηθεί στη κλάση b.

Αντίστοιχα, όπως προέκυψε από όλους τους ταξινομητές ήταν 96,98,97,89 instances από τα 106 είναι σωστά ταξινομημένα με ποσοστό επιτυχία 90.566%, 92.4528 %, 91.5094%, 83.9623% αντίστοιχα. Οι χρόνοι που έκαναν οι ταξινομητές για να ταξινομήσουν τα αποτελέσματα ήταν 0.02 δ., 11.18 δ., 138.33 δ. και 3405.62 δ., αντίστοιχα για τους Naive Bayes, SVM -polykernel, SVM -rbfkernel, και δίκτυο ακτινικής βάσης (RBFnetwork). Βέβαια τα 3 τελευταία, όπως ζητήθηκε τα τρέξαμε σε όλες τις πιθανές τιμές χρησιμοποιώντας την GridSearch ταξινόμηση.

Επιπλέον, από το όνφυσιον Ματριξ (αναλύουμε του Naive Bayes, τα άλλα προκύπτουν αντίστοιχα) διαπιστώσαμε ότι 46/53 δείγματα της κλάσης a έχουν ταξινομηθεί σωστά στην κλάση αυτή, ενώ τα υπόλοιπα 6 ταξινομήθηκαν στην κλάση b. 49/53 της κλάσης b έχουν ταξινομηθεί σωστά και 4 λανθασμένα στη κλάση a.

Η παραμετροποίηση που χρησιμοποιήσαμε στην GridSearch ταξινόμηση φαίνεται από τα printscreen που παραθέσαμε ανωτέρω. Αυτή έγινε με βάση τις default τιμές που έδιναν εξ αρχής οι ταξινομητές και ανεβάζοντας και κατεβάζοντας λίγο τα min και max γύρω από αυτές τις τιμές.

## 1.2 Θέμα 2

Στα πλαίσια της δεύτερης Πρακτικής Άσκησης στο μάθημα της Αναγνώρισης Προτύπων, κληθήκαμε να δημιουργήσουμε νευρωνικά δίκτυα που θα προσεγγίζουν τη συνάρτηση  $y(x)=\sin?(6\pi x_i)+er_i$  στο διάστημα  $[0,1]$ . Όπως προτάθηκε και στην περιγραφή της άσκησης, χρησιμοποιήσαμε το Neural Network Toolbox του Matlab.

- ◇ α) Δημιουργήσαμε ένα σύνολο 200 εκπαιδευτικών προτύπων για την εκπαίδευση κάθε δικτύου που μας ζητείται να εκπαιδεύσουμε. Η δημιουργία των προτύπων με την χρήση της συνάρτησης `rand` και οι αντίστοιχες έξοδοι δημιουργήθηκαν από τον τύπο  $y(x)=\sin?(6\pi x_i)+er_i$ . Παράλληλα δημιουργήσαμε ένα Test set 500 προτύπων, χρησιμοποιώντας την συνάρτηση `rand`, ενώ για τις αντίστοιχες εξόδους, χρησιμοποιήσαμε την συνάρτηση  $y(x)=\sin?(6\pi x_i)$ .
- ◇ β-γ) Με τα 200 εκπαιδευτικά πρότυπα που δημιουργήσαμε, εκπαιδεύσαμε τα 3 διαφορετικά δίκτυα που μας ζητήθηκαν, τα οποία χρησιμοποιούν τους αλγόριθμους `Backpropagation` με όρο ορμής, την μέθοδο συζυγών κλίσεων και την μέθοδο `Levenberg-Marquardt` αντίστοιχα. Ως κριτήρια τερματισμού χρησιμοποιήσαμε με βαθμό προτεραιότητας :

1. Τον αριθμό επαναλήψεων, `trainParam.epochs = 100`
2. Την τιμή του σφάλματος, `trainParam.goal = 10(-5)`
3. Την τιμή της κλίσης, `trainParam.min_grad = 10(-3)`
4. Την τιμή, `trainParam.lr = 0.05`

Η εκπαίδευση έγινε για 5, 10, 15 και 20 ενδιάμεσους νευρώνες, και 20 φορές για κάθε αριθμό ενδιάμεσων νευρώνων. Για κάθε αλγόριθμο εκπαίδευσης, για κάθε αριθμό ενδιάμεσων νευρώνων, για τις 20 επαναλήψεις που έγινε η εκπαίδευση, υπολογίσαμε τον μέσο όρο του τετραγωνικού σφάλματος .

1. Για το αλγόριθμο `Backpropagation`: 0.4647, 0.0932, 0.0713, 0.0731 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
2. Για την μέθοδο συζυγών κλίσεων: 0.1381, 0.0054, 0.0050, 0.0040 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
3. Για την μέθοδο `Levenberg-Marquardt`: 0.1096, 0.0035, 0.0030, 0.0027 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.

Για την αποτίμηση της ικανότητας γενίκευσης των δικτύων που εκπαιδεύσαμε, υπολογίσαμε επίσης ως κριτήριο το μέσο τετραγωνικό σφάλμα.

1. Για το αλγόριθμο `Backpropagation`: 8.5593, 2.4305, 1.5989, 1.9727 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
2. Για την μέθοδο συζυγών κλίσεων: 1.0970, 0.2744, 0.2468, 0.2587 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.

3. Για την μέθοδο Levenberg-Marquardt: 2.1221, 0.2167, 0.2173, 0.2201 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
- ◇ δ) Για να χωρίσουμε το σύνολο των 200 προτύπων σε 150 πρότυπα εκπαίδευσης, θέσαμε `divideParam.trainRatio = 0.75` και `divideParam.testRatio = 0.25` και ακολουθήσαμε την παραπάνω διαδικασία. Τα αποτελέσματα θα είναι:
  1. Για το αλγόριθμο Backpropagation: 8.5593, 2.4305, 1.5989, 1.9727 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
  2. Για την μέθοδο συζυγών κλίσεων: 0.2379, 0.0096, 0.0105, 0.0058 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
  3. Για την μέθοδο Levenberg-Marquardt: 0.1501, 0.0176, 0.0030, 0.0029 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.

Για την αποτίμηση της ικανότητας γενίκευσης των δικτύων που εκπαιδεύσαμε, υπολογίσαμε επίσης ως κριτήριο το μέσο τετραγωνικό σφάλμα.

1. Για το αλγόριθμο Backpropagation: 16.3015, 26.2724, 29.4385, 38.6151 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
2. Για την μέθοδο συζυγών κλίσεων: 2.9565, 0.4703, 0.3933, 0.4038 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.
3. Για την μέθοδο Levenberg-Marquardt: 2.8303, 0.4181, 0.2350, 0.2453 για 5, 10, 15, 20 ενδιάμεσους νευρώνες αντίστοιχα.



## ΚΕΦΑΛΑΙΟ 2

# Σύντομα Αποτελέσματα Ταξινομητών

### 2.1 Iris Plant DataBase

```
1 Scheme:weka.classifiers.bayes.NaiveBayes
2 Relation:      iris
3 Instances:     150
4 Attributes:    5
5               sepallength
6               sepalwidth
7               petallength
8               petalwidth
9               class
10 Test mode:10-fold cross-validation
11
12 ##### Naive Bayes #####
13 === Classifier model (full training set) ===
14
15 Time taken to build model: 0.01 seconds
16
17 === Stratified cross-validation ===
18 === Summary ===
19
20 Correctly Classified Instances      144           96      %
21 Incorrectly Classified Instances     6            4      %
22 Kappa statistic                     0.94
23 Mean absolute error                  0.0342
24 Root mean squared error              0.155
25 Relative absolute error              7.6997 %
26 Root relative squared error          32.8794 %
27 Total Number of Instances           150
28
29 === Confusion Matrix ===
30
```

```

31  a  b  c  <-- classified as
32  50  0  0 |  a = Iris-setosa
33  0 48  2 |  b = Iris-versicolor
34  0  4 46 |  c = Iris-virginica
35
36
37
38 ##### SMO using poly-kernel #####
39 === Classifier model (full training set) ===
40
41 weka.classifiers.meta.GridSearch:
42 Filter: weka.filters.AllFilter
43 Classifier: weka.classifiers.functions.SMO -C 31.0 -L 0.001 -P 1.0E-12 -N ...
         0 -V -l -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel ...
         -C 250007 -E 1.5"
44
45 X property: classifier.c
46 Y property: classifier.kernel.exponent
47
48 Evaluation: Accuracy
49 Coordinates: [31.0, 1.5]
50 Values: 31.0 (X coordinate), 1.5 (Y coordinate)
51
52 Time taken to build model: 15.57 seconds
53
54 === Stratified cross-validation ===
55 === Summary ===
56
57 Correctly Classified Instances      144           96      %
58 Incorrectly Classified Instances      6           4      %
59 Kappa statistic                    0.94
60 Mean absolute error                 0.2311
61 Root mean squared error             0.288
62 Relative absolute error              52      %
63 Root relative squared error         61.101  %
64 Total Number of Instances          150
65
66
67 === Confusion Matrix ===
68
69  a  b  c  <-- classified as
70  50  0  0 |  a = Iris-setosa
71  0 45  5 |  b = Iris-versicolor
72  0  1 49 |  c = Iris-virginica
73
74
75
76 ##### SMO using RBF-kernel #####
77 === Classifier model (full training set) ===
78

```

```

79 weka.classifiers.meta.GridSearch:
80 Filter: weka.filters.AllFilter
81 Classifier: weka.classifiers.functions.SMO -C 325.0 -L 0.001 -P 1.0E-12 -N ...
    0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.RBFKernel -C ...
    250007 -G 0.01"
82
83 X property: classifier.c
84 Y property: classifier.kernel.gamma
85
86 Evaluation: Accuracy
87 Coordinates: [325.0, -2.0]
88 Values: 325.0 (X coordinate), 0.01 (Y coordinate)
89
90 SMO
91
92 Kernel used:
93   RBF kernel:  $K(x,y) = e^{-(0.01 * \langle x-y, x-y \rangle^2)}$ 
94
95 Time taken to build model: 862.97 seconds
96
97 === Stratified cross-validation ===
98 === Summary ===
99
100 Correctly Classified Instances      143      95.3333 %
101 Incorrectly Classified Instances    7      4.6667 %
102 Kappa statistic                    0.93
103 Mean absolute error                 0.2326
104 Root mean squared error             0.2906
105 Relative absolute error             52.3333 %
106 Root relative squared error         61.6441 %
107 Total Number of Instances          150
108
109 === Confusion Matrix ===
110
111   a  b  c   <-- classified as
112  50  0  0 |  a = Iris-setosa
113   0 45  5 |  b = Iris-versicolor
114   0  2 48 |  c = Iris-virginica
115
116
117 ##### RBFnetworks #####
118 === Classifier model (full training set) ===
119
120 weka.classifiers.meta.GridSearch:
121 Filter: weka.filters.AllFilter
122 Classifier: weka.classifiers.functions.RBFNetwork -B 37 -S 1 -R 1.0E-8 -M ...
    -1 -W 199526.23149688786
123
124 X property: classifier.numClusters
125 Y property: classifier.minStdDev

```

```
126
127 Evaluation: Accuracy
128 Coordinates: [37.0, 5.3]
129 Values: 37.0 (X coordinate), 199526.23149688786 (Y coordinate)
130
131 Radial basis function network
132 (Logistic regression applied to K-means clusters as basis functions):
133
134 Logistic Regression with ridge parameter of 1.0E-8
135 Coefficients...
136
137 TTime taken to build model: 1663.5 seconds
138
139 === Stratified cross-validation ===
140 === Summary ===
141
142 Correctly Classified Instances      144           96      %
143 Incorrectly Classified Instances    6             4      %
144 Kappa statistic                    0.94
145 Mean absolute error                 0.0318
146 Root mean squared error            0.1482
147 Relative absolute error             7.1541 %
148 Root relative squared error        31.4372 %
149 Total Number of Instances          150
150
151
152 === Confusion Matrix ===
153
154  a  b  c  <-- classified as
155  50  0  0 | a = Iris-setosa
156   0 46  4 | b = Iris-versicolor
157   0  2 48 | c = Iris-virginica
```

## 2.2 Promoter Gene Sequence DataBase

```

1 cheme:weka.classifiers.meta.GridSearch -E ACC -y-property ...
  classifier.kernel.gamma -y-min -10.0 -y-max 2.0 -y-step 1.0 -y-base ...
  10.0 -y-expression pow(BASE,I) -filter weka.filters.AllFilter ...
  -x-property classifier.c -x-min 1.0 -x-max 501.0 -x-step 10.0 -x-base ...
  10.0 -x-expression I -sample-size 100.0 -traversal COLUMN-WISE ...
  -log-file "C:\\Program Files\\Weka-3-6" -S 1 -W ...
  weka.classifiers.functions.SMO -- -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V ...
  -1 -W 1 -K "weka.classifiers.functions.supportVector.RBFSKernel -C ...
  250007 -G 0.01"
2 Relation:      promoters
3 Instances:     106
4 Attributes:    58
5               1
6               2
7               3
8               .....
9               .....
10              56
11              57
12              class
13 Test mode:10-fold cross-validation
14
15 ##### Naive Bayes #####
16 === Classifier model (full training set) ===
17
18
19 Time taken to build model: 0.02 seconds
20
21 === Stratified cross-validation ===
22 === Summary ===
23
24 Correctly Classified Instances          96           90.566 %
25 Incorrectly Classified Instances        10           9.434 %
26 Kappa statistic                        0.8113
27 Mean absolute error                    0.1149
28 Root mean squared error                0.2705
29 Relative absolute error                 22.9633 %
30 Root relative squared error             54.0623 %
31 Total Number of Instances              106
32
33
34 === Confusion Matrix ===
35
36  a  b  <-- classified as
37 47  6 |  a = +
38  4 49 |  b = -

```

```

39
40
41
42 ##### SMO using poly-kernel #####
43 === Classifier model (full training set) ===
44
45 weka.classifiers.meta.GridSearch:
46 Filter: weka.filters.AllFilter
47 Classifier: weka.classifiers.functions.SMO -C 31.0 -L 0.001 -P 1.0E-12 -N ...
    0 -V -l -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel ...
    -C 250007 -E 1.5"
48
49 X property: classifier.c
50 Y property: classifier.kernel.exponent
51
52
53 Evaluation: Accuracy
54 Coordinates: [501.0, 2.5]
55 Values: 501.0 (X coordinate), 2.5 (Y coordinate)
56
57 Time taken to build model: 11.18 seconds
58
59 === Stratified cross-validation ===
60 === Summary ===
61
62 Correctly Classified Instances          98           92.4528 %
63 Incorrectly Classified Instances        8           7.5472 %
64 Kappa statistic                        0.8491
65 Mean absolute error                    0.0755
66 Root mean squared error                0.2747
67 Relative absolute error                 15.0855 %
68 Root relative squared error            54.9104 %
69 Total Number of Instances              106
70
71 === Confusion Matrix ===
72
73   a  b   <-- classified as
74  46  7 |   a = +
75   1 52 |   b = -
76
77
78 ##### SMO using RBF-kernel #####
79 === Classifier model (full training set) ===
80
81 weka.classifiers.meta.GridSearch:
82 Filter: weka.filters.AllFilter
83 Classifier: weka.classifiers.functions.SMO -C 325.0 -L 0.001 -P 1.0E-12 -N ...
    0 -V -l -W 1 -K "weka.classifiers.functions.supportVector.RBFKernel -C ...
    250007 -G 0.01"
84

```

```

85 X property: classifier.c
86 Y property: classifier.kernel.gamma
87
88 Evaluation: Accuracy
89 Coordinates: [501.0, -3.0]
90 Values: 501.0 (X coordinate), 0.001 (Y coordinate)
91
92 SMO
93
94 Kernel used:
95   RBF kernel:  $K(x,y) = e^{-(0.001 * \langle x-y, x-y \rangle^2)}$ 
96
97
98 Time taken to build model: 138.33 seconds
99
100 === Stratified cross-validation ===
101 === Summary ===
102
103 Correctly Classified Instances          97           91.5094 %
104 Incorrectly Classified Instances         9           8.4906 %
105 Kappa statistic                        0.8302
106 Mean absolute error                     0.0849
107 Root mean squared error                 0.2914
108 Relative absolute error                 16.9712 %
109 Root relative squared error             58.2412 %
110 Total Number of Instances              106
111
112
113 === Confusion Matrix ===
114
115   a  b   <-- classified as
116  47  6 |   a = +
117   3 50 |   b = -
118
119
120 ##### RBFnetworks #####
121 === Classifier model (full training set) ===
122
123 weka.classifiers.meta.GridSearch:
124 Filter: weka.filters.AllFilter
125 Classifier: weka.classifiers.functions.RBFNetwork -B 37 -S 1 -R 1.0E-8 -M ...
126               -1 -W 199526.23149688786
127
128 X property: classifier.numClusters
129 Y property: classifier.minStdDev
130
131 Evaluation: Accuracy
132 Coordinates: [20.0, 10.0]
133 Values: 20.0 (X coordinate), 1.0E10 (Y coordinate)

```

```
134 Radial basis function network
135 (Logistic regression applied to K-means clusters as basis functions):
136
137 Logistic Regression with ridge parameter of 1.0E-8
138 Coefficients...
139
140 Time taken to build model: 3405.62 seconds
141
142 === Stratified cross-validation ===
143 === Summary ===
144
145 Correctly Classified Instances      89           83.9623 %
146 Incorrectly Classified Instances   17           16.0377 %
147 Kappa statistic                    0.6792
148 Mean absolute error                 0.1628
149 Root mean squared error            0.3978
150 Relative absolute error            32.5454 %
151 Root relative squared error        79.5109 %
152 Total Number of Instances         106
153
154
155 === Confusion Matrix ===
156
157 === Confusion Matrix ===
158
159   a  b  <-- classified as
160  51  2 |  a = +
161  15 38 |  b = -
```



## ΚΕΦΑΛΑΙΟ 3

### Υλοποιήσεις σε Matlab

Στο παρόν Κεφάλαιο παραθέτουμε τις υλοποιήσεις σε κώδικα Matlab για το δεύτερο θέμα της δεύτερης πρακτικής άσκησης στην Αναγνώριση Προτύπων.

#### 3.1 Θέμα 2

```
1
2 clc
3 clear all
4
5 %(a)
6 e = 0.2;
7 Input = rand([1,200]);
8 Target = zeros(1,200);
9 for i=1:200
10     if rand(2) == 1
11         Sign = -1;
12     else
13         Sign = 1;
14     end
15     Target(1,i) = sin(6*pi*Input(1,i)) + e*Sign*rand;
16 end
17
18 TestInput = rand(1,500); %To Test Set mou
19 TestTarget = sin(6*pi*TestInput); %To Test Target
20
21 % (b)
22 %
23 % %To 20 ekfrazai ton ari8mo tw n nevrwnwn, 8a dokimasw gia 5, 10, 15, 20
24 %
25 % (i) BackPropagation me oro ormhs
26 Avg_MSE_GDM_ij = zeros(4,20);
```

```

27 for i = 5:5:20
28     mean_square = 0;
29     my_net = fitnet(i); %Orizw to diktuo mou
30     GDM_MSE = 0;
31     for j=1:1:20
32         my_net = init(my_net);
33         my_net.trainFcn = "traingdm";
34         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun
35         my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...
            na stamathsei default 0
36         my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...
            megaluterh suglish default 1e-10 den prepei na ginei polu ...
            mikrh h timh gia na apofugoume overfitting
37         my_net.trainParam.lr = 0.05; %?? default 0.01, otan to kanw 0.05, ...
            oso afksanontai ta layers, toso meiwnetai to best ...
            performance(tetragwniko error)
38         my_net.divideParam.trainRatio = 1;
39         my_net.divideParam.valRatio = 0;
40         my_net.divideParam.testRatio = 0;
41         [my_net tr] = train(my_net,Input,Target);
42         %To meso oro tw'n meswn tetragwnikwn errors gia ka8e ari8mo ...
            neuronwn, 5,10,15,20
43         Avg_MSE_GDM_ij((i/5),j) = tr.best_perf;
44
45         % (c)
46         my_error = my_net(TestInput) - TestTarget;
47         my_mse = sum(my_error.^2) / length(my_error);
48         GDM_MSE = GDM_MSE + my_mse; %meso tetragwniko sfalma gia test set
49         % (c)
50
51     end
52     GDM_MSE
53 end
54 Avg_MSE_GDM = mean("Avg_MSE_GDM_ij");
55
56 % (ii) Conjugate Gradient
57 Avg_MSE_CG_ij = zeros(4,20);
58 for i = 5:5:20
59     mean_square = 0;
60     my_net = fitnet(i); %Orizw to diktuo mou
61     CG_MSE = 0;
62     for j=1:1:20
63         my_net = init(my_net);
64         my_net.trainFcn = "traincgp";
65         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun
66         my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...
            na stamathsei default 0
67         my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...
            megaluterh suglish default 1e-10 den prepei na ginei polu ...
            mikrh h timh gia na apofugoume overfitting

```

```

68     my_net.trainParam.lr = 0.05;
69     my_net.divideParam.trainRatio = 1;
70     my_net.divideParam.valRatio = 0;
71     my_net.divideParam.testRatio = 0;
72     [my_net tr] = train(my_net, Input, Target);
73     %To meso oro twm meswn tetragwnikwn errors gia ka8e ari8mo ...
       neuronwn, 5,10,15,20
74     %tr.best_perf einai ousiastika to meso tetragwniko kostos
       Avg_MSE_CG_ij((i/5),j) = tr.best_perf;
75     % (c)
76     my_error = my_net(TestInput) - TestTarget;
77     my_mse = sum(my_error.^2) / length(my_error);
78     CG_MSE = CG_MSE + my_mse;
79     % (c)
80
81     end
82     CG_MSE
83 end
84 Avg_MSE_CG = mean("Avg_MSE_CG_ij");
85
86 % %(iii) Levenberg-Merquardt
87 Avg_MSE_LM_ij = zeros(4,20);
88 for i = 5:5:20
89     mean_square = 0;
90     my_net = fitnet(i); %Orizw to diktuo mou
91     LM_MSE = 0;
92     for j=1:1:20
93         my_net = init(my_net);
94         my_net.trainFcn = "trainlm";
95         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun
96         my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...
           na stamathsei default 0
97         my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...
           megaluterh suglish default 1e-10 den prepei na ginei polu ...
           mikrh h timh gia na apofugoume overfitting
98         my_net.trainParam.lr = 0.05; %?? default 0.01
99         my_net.divideParam.trainRatio = 1;
100        my_net.divideParam.valRatio = 0;
101        my_net.divideParam.testRatio = 0;
102        [my_net tr] = train(my_net, Input, Target);
103        %To meso oro twm meswn tetragwnikwn errors gia ka8e ari8mo ...
           neuronwn, 5,10,15,20
104        %tr.best_perf einai ousiastika to meso tetragwniko kostos
           Avg_MSE_LM_ij((i/5),j) = tr.best_perf;
105        % (c)
106        my_error = my_net(TestInput) - TestTarget;
107        my_mse = sum(my_error.^2) / length(my_error);
108        LM_MSE = LM_MSE + my_mse;
109        % (c)
110
111    end
112    LM_MSE

```

```

113 end
114 Avg_MSE_LM = mean("Avg_MSE_LM_ij");
115
116 % (d)
117 % (i) BackPropagation με oro ormhs
118 Avg_MSE_GDM_ij = zeros(4,20);
119 for i = 5:5:20
120     mean_square = 0;
121     my_net = fitnet(i); %Orizw to diktuo mou
122     GDM_MSE = 0;
123     for j=1:1:20
124         my_net = init(my_net);
125         my_net.trainFcn = "traingdm";
126         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun
127         my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...
            na stamathsei default 0
128         my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...
            megaluterh suglish default 1e-10 den prepei na ginei polu ...
            mikrh h timh gia na apofugoume overfitting
129         my_net.trainParam.lr = 0.05; %?? default 0.01, otan to kanw 0.05, ...
            oso afksanontai ta layers, toso meiwnetai to best ...
            performance(tetragwniko error)
130         my_net.divideParam.trainRatio = 0.75;
131         my_net.divideParam.valRatio = 0.125;
132         my_net.divideParam.testRatio = 0.125;
133         [my_net tr] = train(my_net,Input,Target);
134         %To meso oro twm meswn tetragwnikwn errors gia ka8e ari8mo ...
            neuronwn, 5,10,15,20
135         Avg_MSE_GDM_ij((i/5),j) = tr.best_perf;
136
137         % (c)
138         my_error = my_net(TestInput) - TestTarget;
139         my_mse = sum(my_error.^2) / length(my_error);
140         GDM_MSE = GDM_MSE + my_mse;
141         % (c)
142     end
143     GDM_MSE
144 end
145 Avg_MSE_GDM = mean("Avg_MSE_GDM_ij");
146
147 % (ii) Conjugate Gradient
148 Avg_MSE_CG_ij = zeros(4,20);
149 for i = 5:5:20
150     mean_square = 0;
151     my_net = fitnet(i); %Orizw to diktuo mou
152     CG_MSE = 0;
153     for j=1:1:20
154         my_net = init(my_net);
155         my_net.trainFcn = "traincgp";
156         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun

```

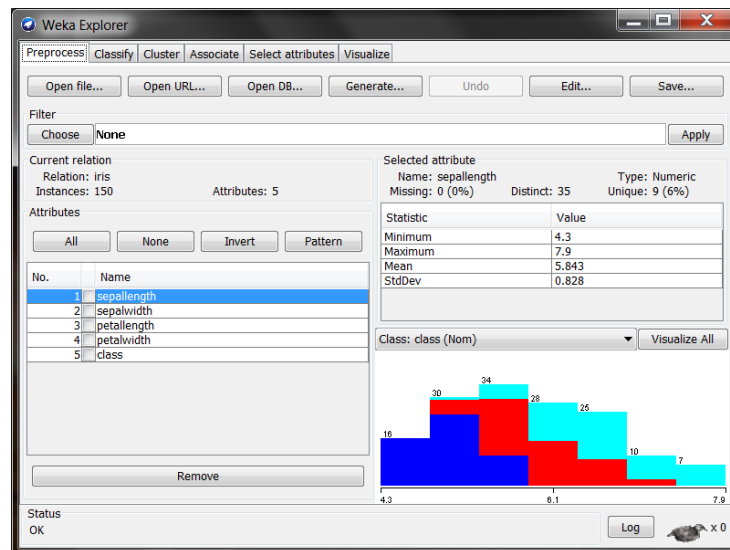
```
157     my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...  
        na stamathsei default 0  
158     my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...  
        megaluterh suglish default 1e-10 den prepei na ginei polu ...  
        mikrh h timh gia na apofugoume overfitting  
159     my_net.trainParam.lr = 0.05; %?? default 0.01  
160     my_net.divideParam.trainRatio = 0.75;  
161     my_net.divideParam.valRatio = 0.125;  
162     my_net.divideParam.testRatio = 0.125;  
163     [my_net tr] = train(my_net, Input, Target);  
164     %To meso oro twm meswn tetragwnikwn errors gia ka8e ari8mo ...  
        neuronwn, 5,10,15,20  
165     %tr.best_perf einai ousiastika to meso tetragwniko kostos  
        Avg_MSE_CG_ij((i/5),j) = tr.best_perf;  
166     % (c)  
167     my_error = my_net(TestInput) - TestTarget;  
168     my_mse = sum(my_error.^2) / length(my_error);  
169     CG_MSE = CG_MSE + my_mse;  
170     % (c)  
171  
172     end  
173     CG_MSE  
174 end  
175 Avg_MSE_CG = mean("Avg_MSE_CG_ij");  
176  
177 % %(iii) Levenberg-Merquardt  
178 Avg_MSE_LM_ij = zeros(4,20);  
179 for i = 5:5:20  
180     mean_square = 0;  
181     my_net = fitnet(i); %Orizw to diktuo mou  
182     LM_MSE = 0;  
183     for j=1:1:20  
184         my_net = init(my_net);  
185         my_net.trainFcn = "trainlm";  
186         my_net.trainParam.epochs = 100; %Poses epanalhpseis 8a ginoun  
187         my_net.trainParam.goal = 1e-5; %Poso 8elw na ftasei to error gia ...  
            na stamathsei default 0  
188         my_net.trainParam.min_grad = 0.001; %Oso mikroterh, toso ...  
            megaluterh suglish default 1e-10 den prepei na ginei polu ...  
            mikrh h timh gia na apofugoume overfitting  
189         my_net.trainParam.lr = 0.05; %?? default 0.01  
190         my_net.divideParam.trainRatio = 0.75;  
191         my_net.divideParam.valRatio = 0.25;  
192         my_net.divideParam.testRatio = 0;  
193         [my_net tr] = train(my_net, Input, Target);  
194         %To meso oro twm meswn tetragwnikwn errors gia ka8e ari8mo ...  
            neuronwn, 5,10,15,20  
195         %tr.best_perf einai ousiastika to meso tetragwniko kostos  
            Avg_MSE_LM_ij((i/5),j) = tr.best_perf;  
196         % (c)  
197         my_error = my_net(TestInput) - TestTarget;
```

```
199         my_mse = sum(my_error.^2) / length(my_error);  
200         LM_MSE = LM_MSE + my_mse;  
201         % (c)  
202     end  
203     LM_MSE  
204 end  
205 Avg_MSE_LM = mean("Avg_MSE_LM_ij");
```

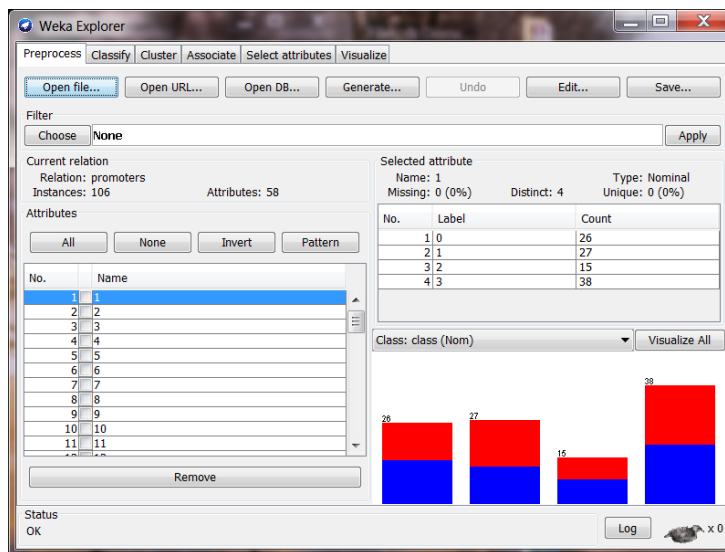
# ΚΕΦΑΛΑΙΟ 4

## Ενδεικτικές Εικόνες

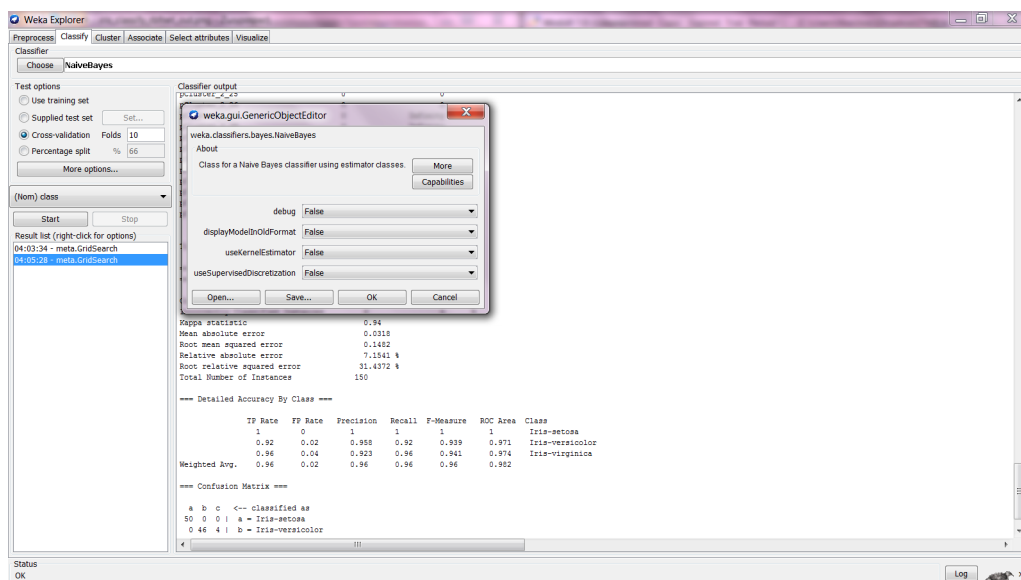
Σε αυτό το κεφάλαιο παραθέτουμε κάποιες ενδεικτικές εικόνες που δεν παρουσιάσαμε στην εργασία νωρίτερα.



Σχήμα 4.1: *Iris plant Database loaded in WEKA using ARFF file format*

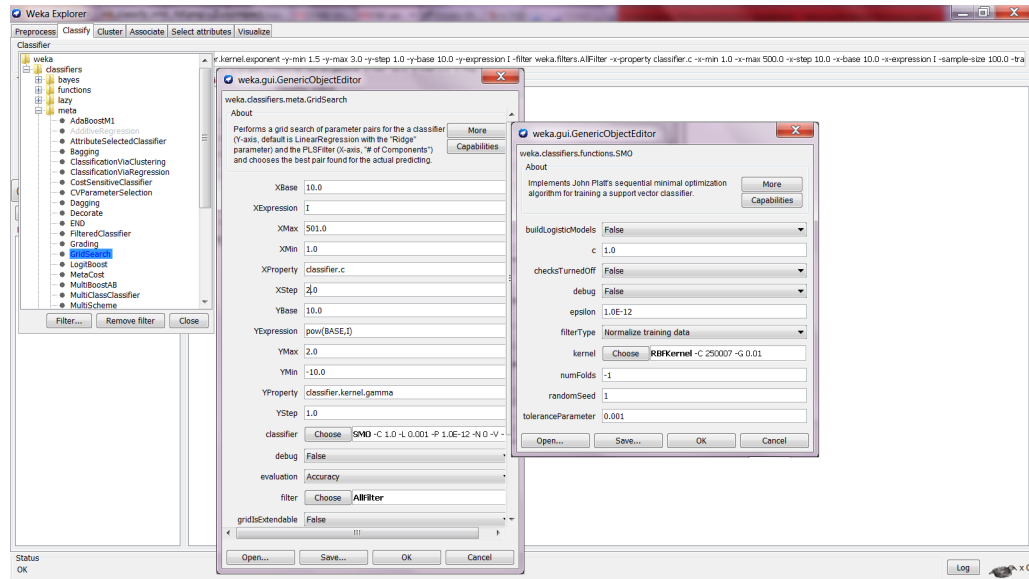


Σχήμα 4.2: *Promoter Gene Sequence Database loaded in WEKA using ARFF file format*

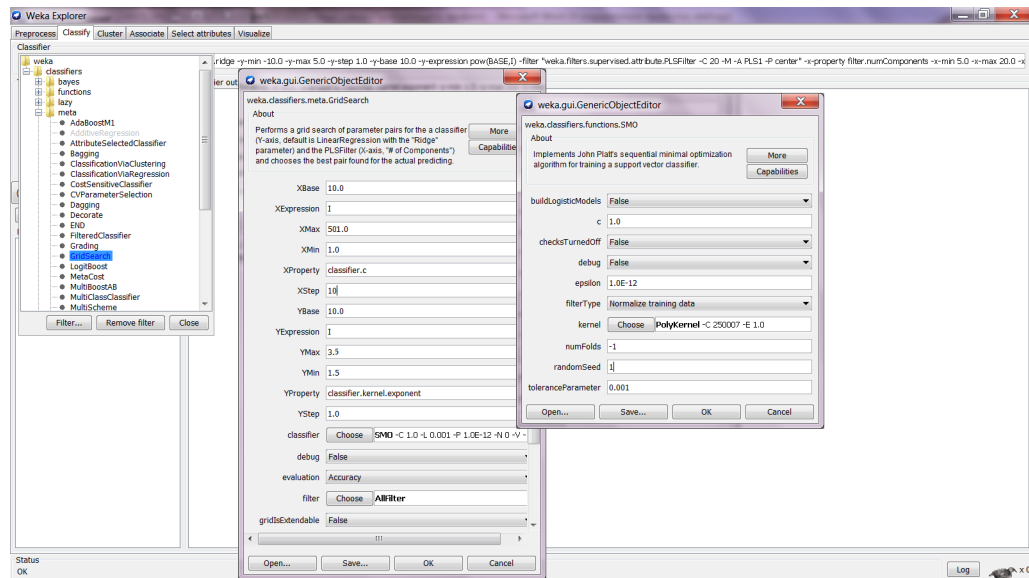


Σχήμα 4.3: *Iris parametrization for classification using Naive Bayes*

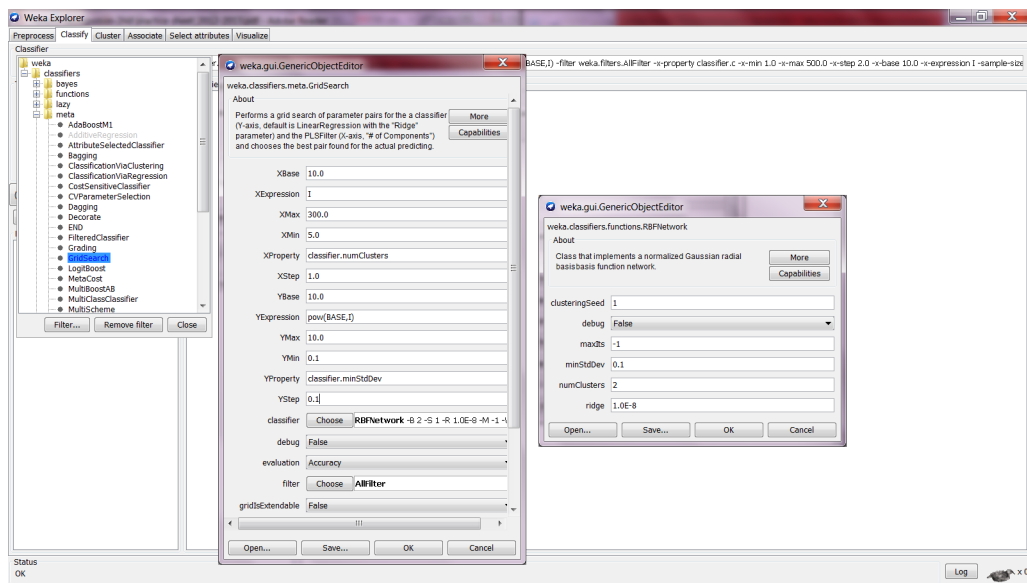




Σχήμα 4.4: *Iris parametrization for classification using SVM RBF*



Σχήμα 4.5: *Iris parametrization for classification using SVM Polykernel*



Σχήμα 4.6: *Iris parametrization for classification using RBFnetworks*

```

34 petallength
35     mean          1.4694          4.2452          5.5516
36     std. dev.      0.1782          0.4712          0.5529
37     weight sum      50            50            50
38     precision      0.1405          0.1405          0.1405
39
40 petalwidth
41     mean           0.2743          1.3097          2.0343
42     std. dev.      0.1096          0.1915          0.2646
43     weight sum      50            50            50
44     precision      0.1143          0.1143          0.1143
45
46
47
48 Time taken to build model: 0.01 seconds
49
50 === Stratified cross-validation ===
51 === Summary ===
52
53 Correctly Classified Instances      144          96      %
54 Incorrectly Classified Instances     6            4      %
55 Kappa statistic                     0.94
56 Mean absolute error                 0.0342
57 Root mean squared error             0.155
58 Relative absolute error             7.6997 %
59 Root relative squared error        32.8794 %
60 Total Number of Instances          150
61
62 === Detailed Accuracy By Class ===
63
64      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
65      1        0        1          1        1          1      Iris-setosa
66      0.96     0.04     0.923     0.96     0.941     0.992    Iris-versicolor
67      0.92     0.02     0.958     0.92     0.939     0.992    Iris-virginica
68 Weighted Avg.  0.96     0.02     0.96     0.96     0.96     0.994
69
70 === Confusion Matrix ===
71
72  a  b  c  <-- classified as
73  50  0  0 | a = Iris-setosa
74  0  48  2 | b = Iris-versicolor
75  0  4  46 | c = Iris-virginica
76
77

```

Normal te: length : 2865 lines : 77 Ln : 77 Col : 1 Sel : 0 Dos\Windows ANSI INS

Σχήμα 4.7: *Iris output of Naive Bayes classification*

```

88 - 325 + <0.555556 0.541667 0.627119 0.625 > + X]
89 + 325 + <0.472222 0.416667 0.644068 0.708333 > + X]
90 - 325 + <0.361111 0.416667 0.59322 0.583333 > + X]
91 - 325 + <0.472222 0.375 0.59322 0.583333 > + X]
92 - 325 + <0.611111 0.333333 0.610169 0.583333 > + X]
93 + 0.299
94
95 Number of support vectors: 29
96
97 Number of kernel evaluations: 3206 (83.448% cached)
98
99
100
101 Time taken to build model: 862.97 seconds
102
103 === Stratified cross-validation ===
104 === Summary ===
105
106 Correctly Classified Instances 143 95.3333 %
107 Incorrectly Classified Instances 7 4.6667 %
108 Kappa statistic 0.93
109 Mean absolute error 0.2326
110 Root mean squared error 0.2906
111 Relative absolute error 52.3333 %
112 Root relative squared error 61.6441 %
113 Total Number of Instances 150
114
115 === Detailed Accuracy By Class ===
116
117 TP Rate FP Rate Precision Recall F-Measure ROC Area Class
118 1 0 1 1 1 1 Iris-setosa
119 0.9 0.02 0.957 0.9 0.928 0.945 Iris-versicolor
120 0.96 0.05 0.906 0.96 0.932 0.963 Iris-virginica
121 Weighted Avg. 0.953 0.023 0.954 0.953 0.953 0.969
122
123 === Confusion Matrix ===
124
125 a b c <-- classified as
126 50 0 0 | a = Iris-setosa
127 0 45 5 | b = Iris-versicolor
128 0 2 48 | c = Iris-virginica
129
130

```

Normal text: length: 5250 lines: 130 Ln: 1 Col: 1 Sel: 0 Dos\Windows ANSI INS

Σχήμα 4.8: Iris output of SVM classification using RBFkernel

```

74 + 23.363 * <0.166667 0.208333 0.593222 0.666667 > * X]
75 + 10.6215 * <0.805556 0.416667 0.813559 0.625 > * X]
76 + 31 * <0.472222 0.416667 0.644068 0.708333 > * X]
77 - 5.5186 * <0.361111 0.416667 0.593222 0.583333 > * X]
78 - 7.227 * <0.611111 0.333333 0.610169 0.583333 > * X]
79 - 7.882
80
81 Number of support vectors: 16
82
83 Number of kernel evaluations: 2495 (80.74% cached)
84
85
86
87 Time taken to build model: 15.57 seconds
88
89 === Stratified cross-validation ===
90 === Summary ===
91
92 Correctly Classified Instances      144          96  %
93 Incorrectly Classified Instances    6           4  %
94 Kappa statistic                    0.94
95 Mean absolute error                 0.2311
96 Root mean squared error            0.288
97 Relative absolute error             52  %
98 Root relative squared error        61.101  %
99 Total Number of Instances          150
100
101 === Detailed Accuracy By Class ===
102
103      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
104      1        0        1          1          1          1      Iris-setosa
105      0.9      0.01      0.978      0.9      0.938      0.945    Iris-versicolor
106      0.98      0.05      0.907      0.98      0.942      0.971    Iris-virginica
107 Weighted Avg.  0.96      0.02      0.962      0.96      0.96      0.972
108
109 === Confusion Matrix ===
110
111  a  b  c  <-- classified as
112  50  0  0 | a = Iris-setosa
113  0  45  5 | b = Iris-versicolor
114  0  1  49 | c = Iris-virginica
115
116

```

Normal te: length: 4375 lines: 116 Ln: 1 Col: 1 Sel: 0 Dos\Windows ANSI INS

Σχήμα 4.9: Iris output of SVM classification using Polykernel

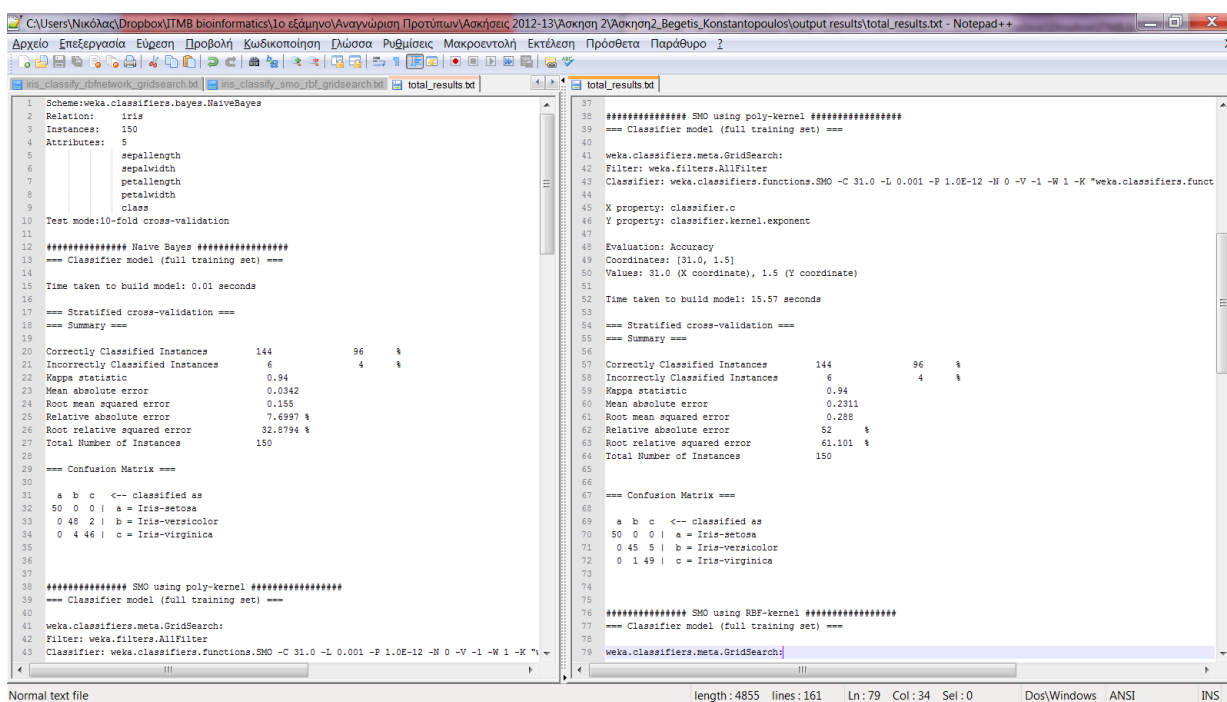
```

252 pCluster_2_25          0          0
253 pCluster_2_26          0          0
254 pCluster_2_27          0      Infinity
255 pCluster_2_28          0      Infinity
256 pCluster_2_29          0      Infinity
257 pCluster_2_30          0          0
258 pCluster_2_31          0      Infinity
259 pCluster_2_32          0          0
260 pCluster_2_33          0      Infinity
261 pCluster_2_34          0          0
262 pCluster_2_35          0      1.0649      1.1345
263 pCluster_2_36          0          0
264
265
266 Time taken to build model: 1663.5 seconds
267
268 === Stratified cross-validation ===
269 === Summary ===
270
271 Correctly Classified Instances      144          96      %
272 Incorrectly Classified Instances     6           4      %
273 Kappa statistic                    0.94
274 Mean absolute error                 0.0318
275 Root mean squared error             0.1482
276 Relative absolute error             7.1541 %
277 Root relative squared error        31.4372 %
278 Total Number of Instances          150
279
280 === Detailed Accuracy By Class ===
281
282      TP Rate   FP Rate   Precision   Recall   F-Measure   ROC Area   Class
283      1         0         1           1         1           1         Iris-setosa
284      0.92      0.02      0.958       0.92      0.939       0.971     Iris-versicolor
285      0.96      0.04      0.923       0.96      0.941       0.974     Iris-virginica
286 Weighted Avg.  0.96      0.02      0.96        0.96      0.96        0.982
287
288 === Confusion Matrix ===
289
290  a  b  c  <-- classified as
291  50  0  0 | a = Iris-setosa
292   0 46  4 | b = Iris-versicolor
293   0  2 48 | c = Iris-virginica
294
295

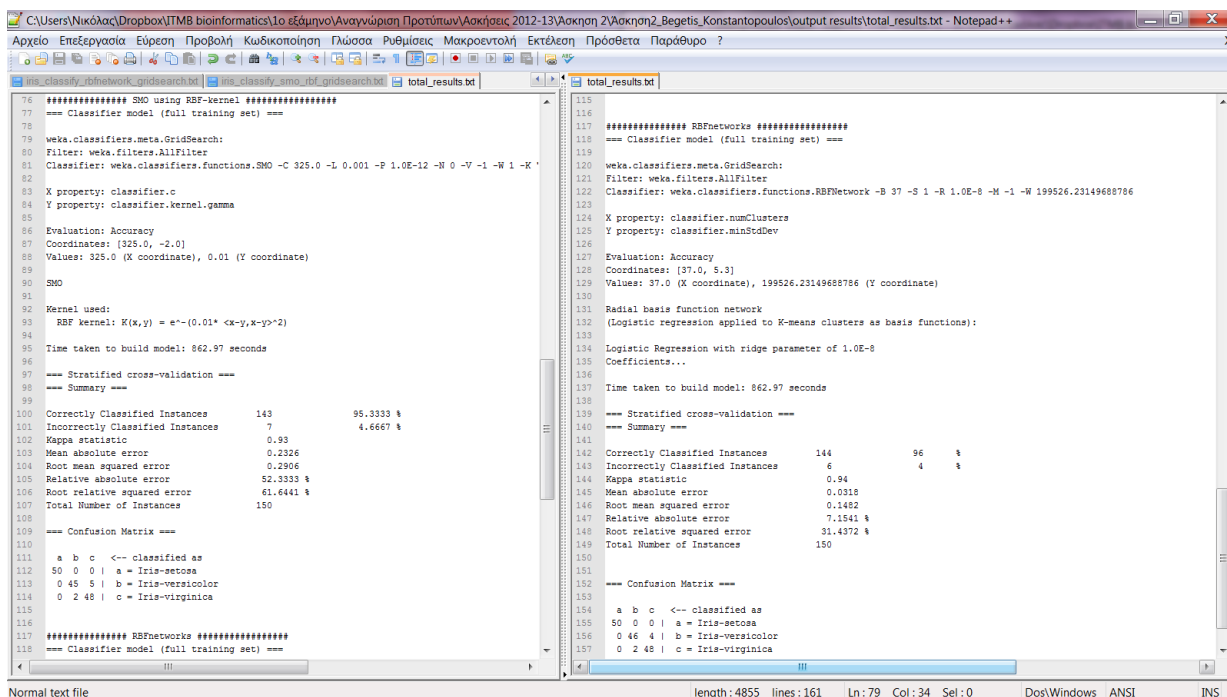
```

Normal text: length : 16398 lines : 295 Ln : 295 Col : 1 Sel : 0 Dos\Windows ANSI INS

Σχήμα 4.10: *Iris output of RBFnetworks classification*



Σχήμα 4.11: *Gathered Iris output from all classifiers*



Σχήμα 4.12: *Gathered Iris output from all classifiers (continue)*





## ΚΕΦΑΛΑΙΟ 5

### Κατακλείδα

Στη δεύτερη εργασία, στα πλαίσια του μεταπτυχιακού μαθήματος της Αναγνώρισης Προτύπων, κληθήκαμε να χρησιμοποιήσουμε ταξινομητές όπως οι Naive Bayes, SVM και RBF network (δίκτυο ακτινικής βάσης) χρησιμοποιώντας το λογισμικό του weka για να κάνουμε χρήση της μεθόδου ten crossvalidation, για την αποτίμηση των αποτελεσμάτων. Παρουσιάσαμε τα αποτελέσματα και τα συμπεράσματά μας μέσω εικόνων και τα παραθέσαμε και στο παραδοτέοcd ώστε να είναι εύκολο για κάποιον να ανατρέξει στο αντίστοιχο αρχείο αποτελεσμάτων του κάθε ταξινομητή και να μπορέσει να τα χρησιμοποιήσει για τον ίδιο ή για άλλο συμπερασμό ταξινόμησης. Επίσης, στο δεύτερο μέρος αυτής της εργασίας δημιουργήσαμε ένα νευρωνικό δίκτυο και χρησιμοποιήσαμε τους αλγόριθμους backpropagation, conjugate gradient και Levenberg-Marquardt. Παραθέσαμε σε ξεχωριστή ενότητα τον κώδικα MatLab που γράψαμε καθώς επίσης και στο παραδοτέοcd ώστε να μπορεί να ελεγχθεί και να τροποποιηθεί από όποιον θέλει να τον χρησιμοποιήσει.