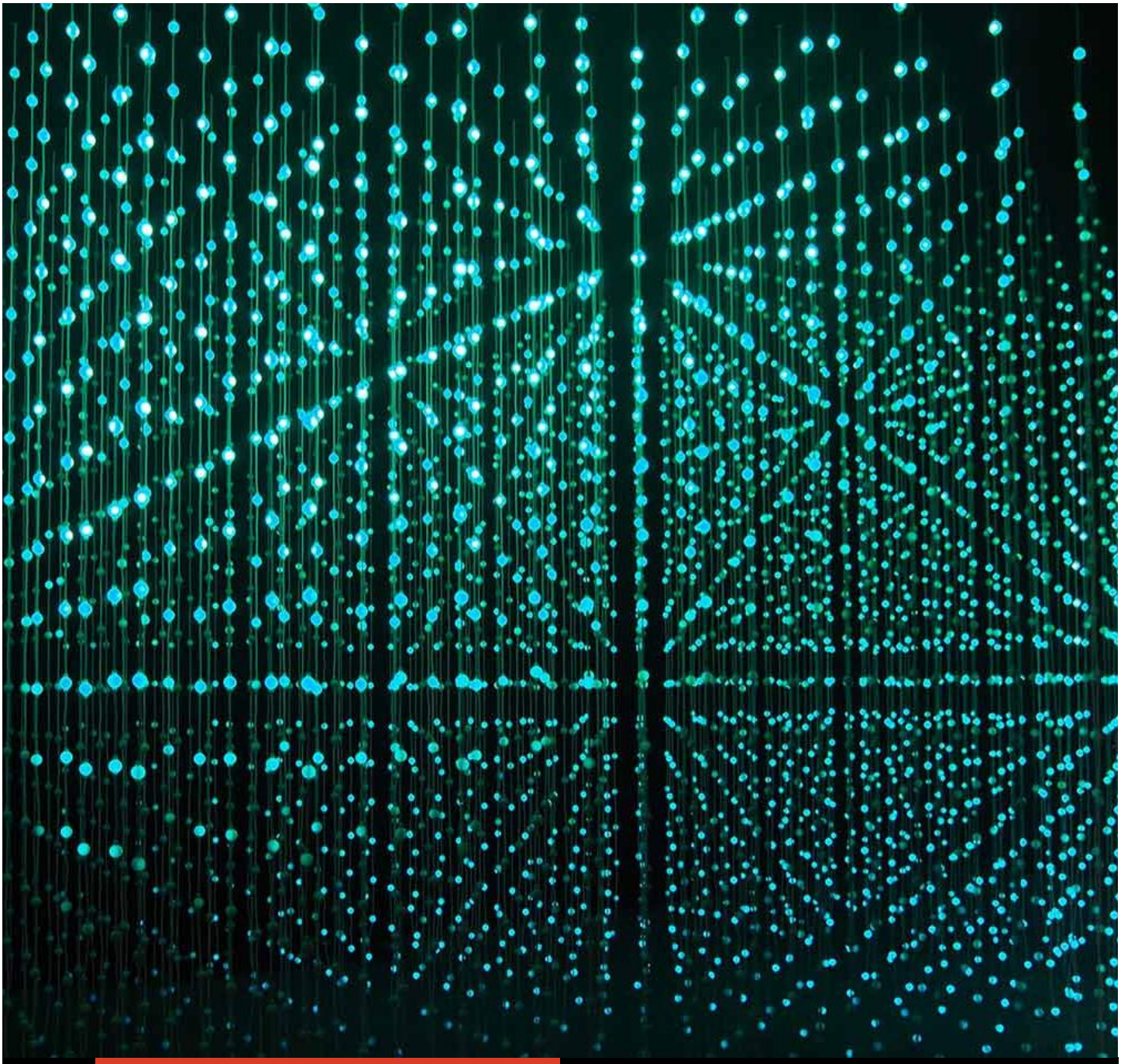
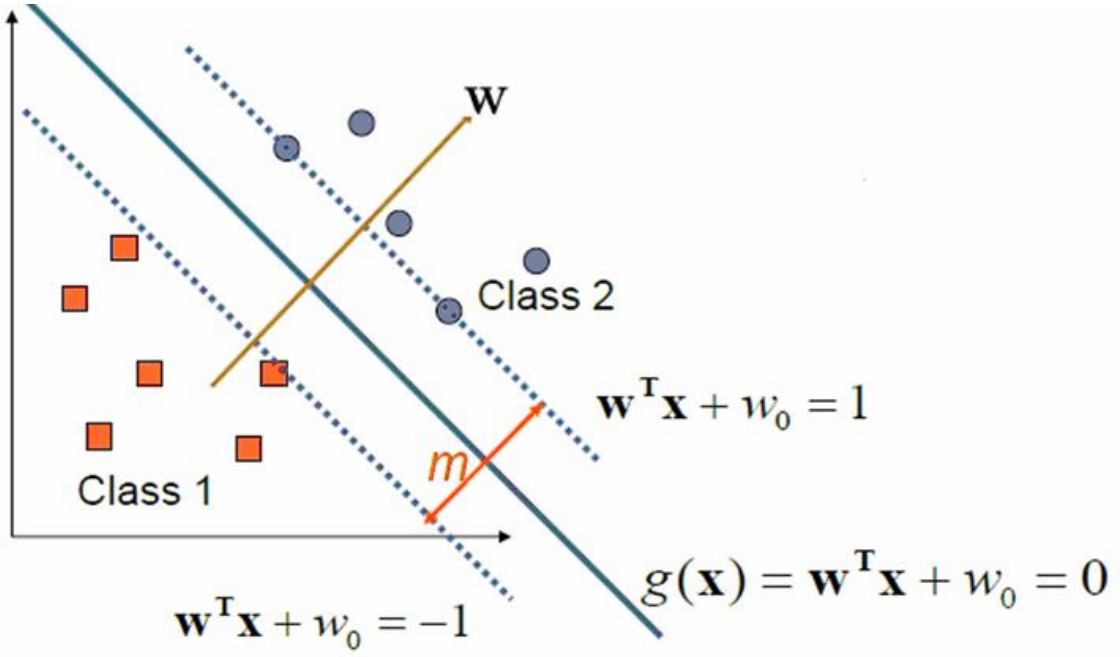


ΕΡΓΑΣΙΑ 6^{Η}

SUPPORT VECTOR MACHINES

ΚΑΡΑΠΕΠΕΡΑ ΕΛΠΙΔΑ 57423

ΑΝΑΓΝΩΡΙΣΗ ΠΡΟΤΥΠΩΝ



SVM

Στόχος ενός SVM είναι μεγιστοποίηση, αρχικά, του περιθωρίου m και, στη συνέχεια, η εύρεση της συνάρτησης διάκρισης, η οποία αφήνει το μέγιστο περιθώριο και από τις 2 κατηγορίες. Με τον τρόπο αυτό διασφαλίζουμε τον καλύτερο δυνατό γραμμικό διαχωρισμό των 2 κλάσεων. Το μέτρο του margin υπολογίζεται από την f(x,y) και άρα αυτή προσπαθούμε να μεγιστοποιήσουμε, με την συνθήκη όμως g(x,y)=c (=1 το πλησιέστερο στοιχείο στην κλάση 1 και -1 το πλησιέστερο στη 2).

Για να επιτευχθούν τα παραπάνω εισάγουμε μία καινούρια μεταβλητή λ (Lagrange multiplier).

Βρίσκουμε, λοιπόν τους multipliers από τη σχέση: L(x,y,λ)=f(x,y)-λ(g(x,y)-c). Οι πολλαπλασιαστές Lagrange μπορεί ναι είναι είτε 0 είτε θετικοί. Εμείς επιλέγουμε τους θετικούς. Αυτοί αποτελούν τα διανύσματα στήριξης.

Ορίζονται τα διανύσματα στήριξης (support vectors) που είναι τα πλησιέστερα διανύσματα (δεδομένα), από κάθε κατηγορία στον ταξινομητή. Είναι, συνεπώς, εμφανές ότι, ακόμη και αν αλλάξουμε τα σημεία που βρίσκονται πιο μακριά από τον ταξινομητή από ότι τα sv, τα sv θα παραμείνουν αναλλοίωτα.

Τα διανύσματα Lagrange που είναι 0 είτε βρίσκονται έξω από την ζώνη διαχωρισμού των κλάσεων, είτε πάνω σε ένα από τα υπερεπίπεδα.

Στη συνέχεια, το w είναι ένας γραμμικός συνδυασμός των παραπάνω διανυσμάτων.

Το υπερεπίπεδο του ταξινομητή που βρίσκουμε από μία support vector machine είναι μοναδικό. Παρ' όλο ότι η λύση είναι μοναδική, οι Lagrange multipliers δεν είναι.

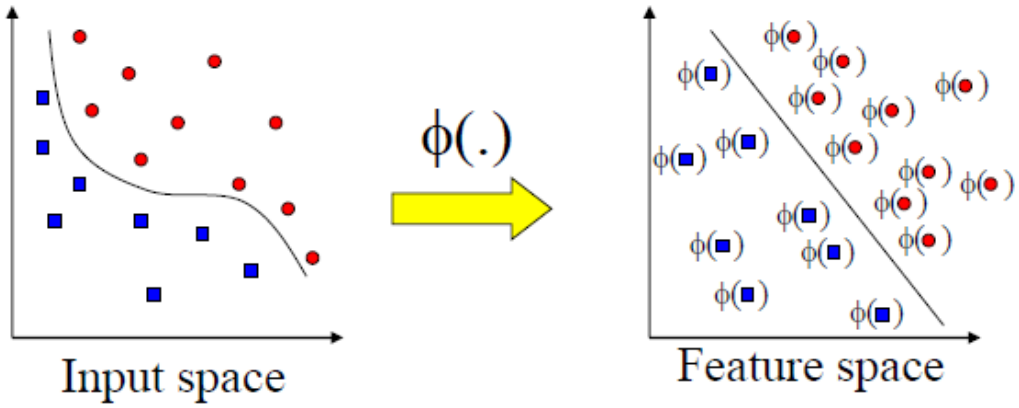
Αν οι κλάσεις δεν είναι γραμμικά διαχωρίσιμες επιτρέπουμε λάθη στην ταξινόμηση και ελαχιστοποιούμε ένα μέτρο το οποίο μεγιστοποιεί το margin και ταυτόχρονα ελαχιστοποιεί τον αριθμό των λαθών.

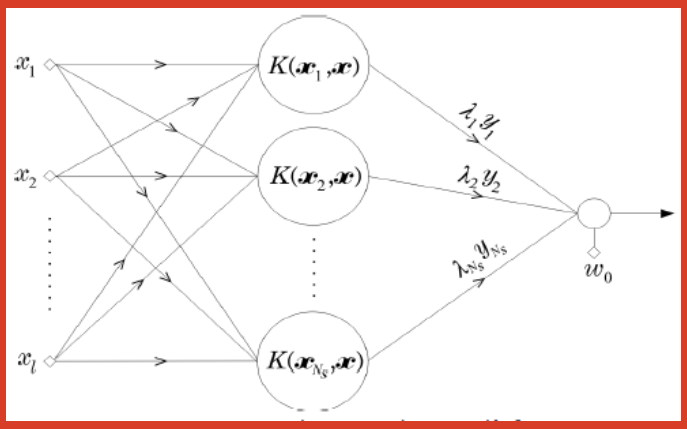
Άλλη μία τακτική είναι να μετασχηματίσουμε τα δεδομένα ώστε να γίνουν γραμμικά διαχωρίσιμα.

Όταν προσπαθούμε να ταξινομήσουμε δεδομένα σε περισσότερες από 2 κλάσεις υπάρχουν 2 τρόποι αντιμετώπισης:

A. One vs one: Βρίσκουμε τους γραμμικούς ταξινομητές που ταξινομούν τα στοιχεία ανάμεσα σε 2 κλάσεις (όλες τις κλάσεις ανά 2) και στη συνέχεια βρίσκουμε το max των gk(x).

B. One vs all: Βρίσκουμε τους γραμμικούς ταξινομητές που ταξινομούν τα στοιχεία ανάμεσα σε κάθε κλάση ενάντια σε όλες τις άλλες. Επιλέγουμε και πάλι την κλάση της οποίας η συνάρτηση gk(x) είναι μέγιστη.





Ένα παράδειγμα μετασχηματισμού των δεδομένων είναι η προσθήκη ενός επιπλέον βάρους. Εισάγουμε ένα δεύτερο χαρακτηριστικό, το οποίο είναι το τετράγωνο του πρώτου. Με τον τρόπο αυτό καταφέρνουμε να κάνουμε τα δεδομένα γραμμικά διαχωρίσιμα. Τα νέα πρότυπα υπάρχουν σε ένα δισδιάστατο, πλέον, χώρο, όπου διαχωρίζονται γραμμικά, ανήκουν όμως σε μια μονοδιάστατη καμπύλη του χώρου, αφού τα δεδομένα όλα προέρχονται ουσιαστικά από 1 μόνο χαρακτηριστικό.

Συνολικά, το SVM:

- Εύκολο στην εκπαίδευση
- Scalable για μεγάλο όγκο δεδομένων
- Μπορεί να ελεγχθεί εύκολα το tradeoff ανάμεσα σε πολυπλοκότητα και ανεκτικότητα σε σφάλμα
- Μπορεί να διαχειριστεί διάφορους τύπους δεδομένων, όπως strings και trees, εκτός από διανύσματα
- Χρειάζεται, ωστόσο, προσοχή στην επιλογή της συνάρτησης Kernel που θα χρησιμοποιηθεί για την εκπαίδευσή του

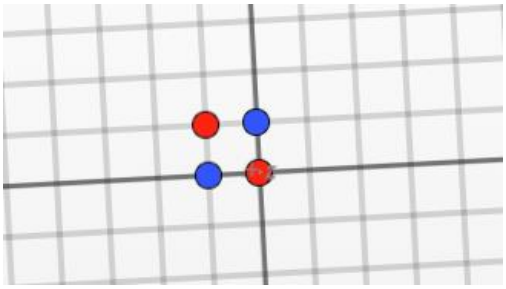
6.1 ΠΡΟΒΛΗΜΑ XOR

Πρόβλημα

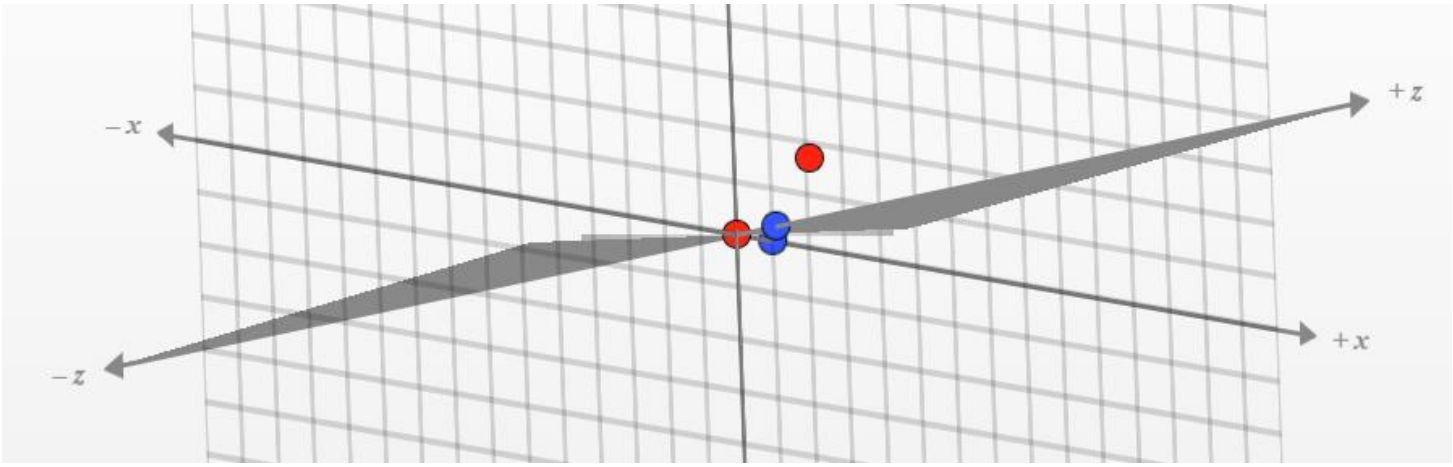
A/A	X1	X2	t
1	0	0	-1
2	1	0	1
3	0	1	1
4	1	1	-1

$\omega_1 = [(0,0),(1,1)]$ $\omega_2 = [(0,1),(1,0)]$ $\Phi(x) = x_1^2 + \sqrt{2}x_1x_2 + x_2^2$
$K(x,y) = \Phi(x)\Phi(y) = (x_1y_1 + x_2y_2)^2 = (xy)^2$

$\boldsymbol{x} = \begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \end{bmatrix}$, $\boldsymbol{y} = [\mathbf{-1}\ \mathbf{1}\ \mathbf{1} - \mathbf{1}]$

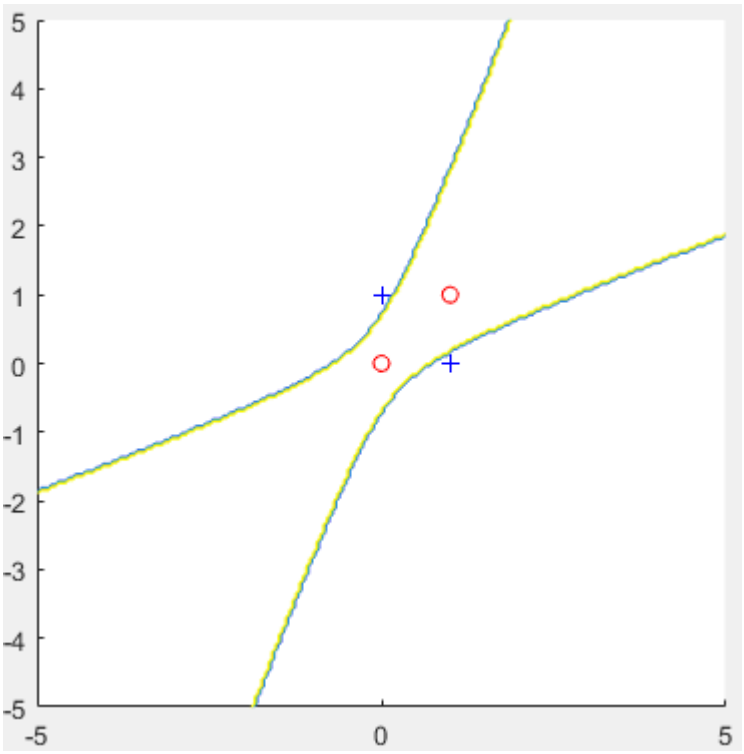


Παρατηρούμε ότι το πρόβλημα δεν είναι γραμμικά διαχωρίσιμο.
Για να γίνει γραμμικά διαχωρίσιμο επιλέγουμε το μετασχηματισμό $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}$*. Πλέον, φαίνεται από την παρακάτω εικόνα ότι τα στοιχεία είναι γραμμικά διαχωρίσιμα.*



Το αποτέλεσμα της επίλυσης με της χρήση MATLAB είναι ο διαχωρισμός των σημείων ως εξής:
Το ω, από το οποίο προκύπτει η συνάρτηση διαχωρισμού είναι ω = [-2, 4.2426, -2] και ω0 = 1. Από αυτά προκύπτει το επίπεδο διαχωρισμού στον τριδιάστατο χώρο.

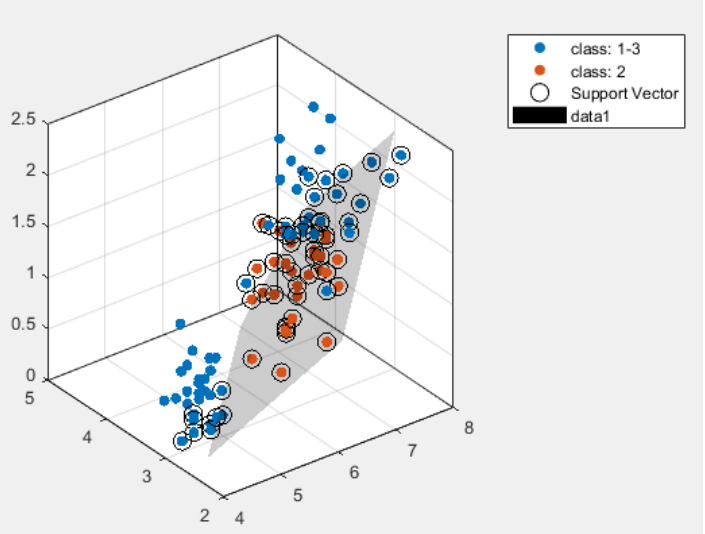
Το αποτέλεσμα είναι error = 0, με 4 sv>0 και ο επιτυχής διαχωρισμός παρουσιάζεται και σχηματικά στην παρακάτω εικόνα:


5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
0
5

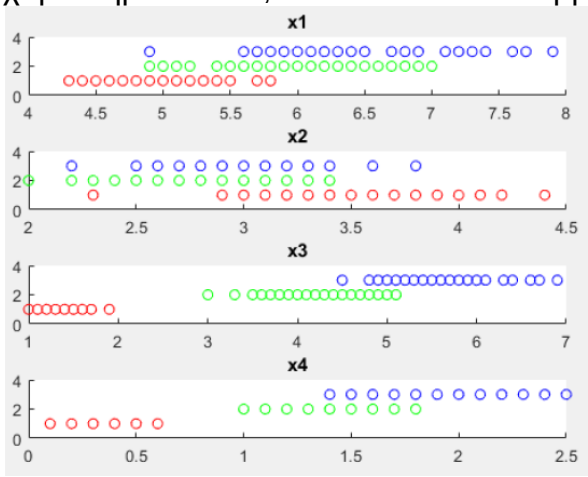
IRIS FLOWERS CLASSIFICATION

A.

Στην ταξινόμηση με τη χρήση τριών χαρακτηριστικών με τη χρήση γραμμικής συνάρτησης στο SVM ο διαχωρισμός ήταν ανεπιτυχής, καθώς το Iris-versicolor δεν είναι γραμμικά διαχωρίσιμο από τα άλλα 2 λουλούδια κρίνοντας από τα χαρακτηριστικά 1, 2 και 4. Το σφάλμα ήταν 27.78% για την ταξινόμηση του ίδιου του training set, 30% για την ταξινόμηση του validation set και 33.33% για την ταξινόμηση του test set.



Αυτό είναι λογικό αφού, όπως ειπώθηκε και στην εργασία 5 και φαίνεται στην αριστερή εικόνα, το Iris-versicolor (πράσινο) είναι γραμμικά διαχωρίσιμο από τα άλλα δύο (σχεδόν) στο χαρακτηριστικό x3, το οποίο εδώ δε λαμβάνεται υπ' όψιν.



Η απεικόνιση της προσπάθειας διαχωρισμού φαίνεται στην πάνω εικόνα:

Χρησιμοποιώντας και τα τέσσερα χαρακτηριστικά, ο διαχωρισμός δεν βελτιώνεται πολύ, αλλά εξακολουθεί να είναι απογοητευτικός, με σφάλμα 28.89% για την ταξινόμηση του ίδιου του training set. 26.67% για την ταξινόμηση του validation set και 33.33% για την ταξινόμηση του test set.

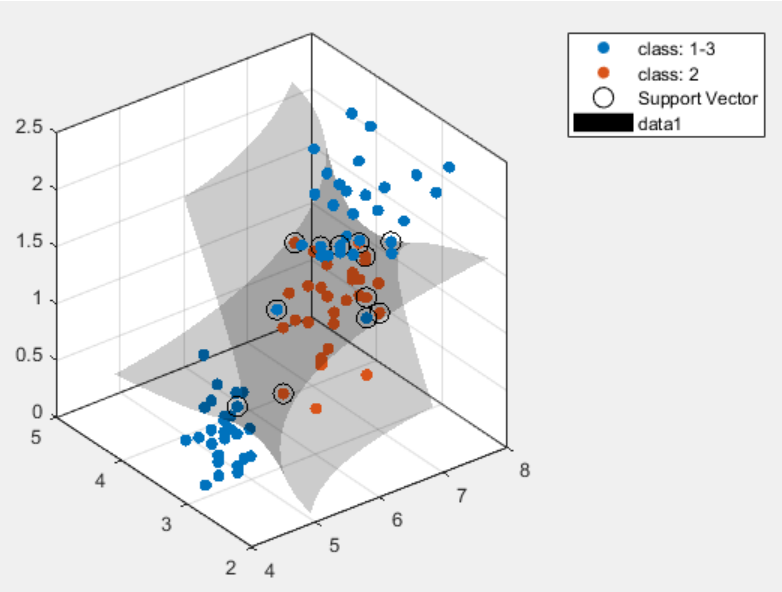
Το αποτέλεσμα βγάζει νόημα καθώς όντως δε φαίνεται να είναι καθαρά γραμμικά διαχωρίσιμη η κατηγορία που εξετάζουμε σε σχέση με τις άλλες δύο, καθώς υπάρχουν σημεία που αλληλοκαλύπτονται από δύο τουλάχιστον κατηγορίες.

Στην εργασία 5 για το διαχωρισμό του Iris-versicolor από τα άλλα δύο λουλούδια χρησιμοποιήθηκε η μέθοδος των ελαχίστων τετραγώνων με χρήση του ψευδοαντιστρόφου (LS) και με την επαναληπτική μέθοδο του Ho-Kashyap. Τα αποτελέσματα εκεί ήταν πολύ καλύτερα, αφού καταφέραμε να επιτύχουμε μόλις 1.33% σφάλμα στην ταξινόμηση του training set με τη χρήση και των τεσσάρων χαρακτηριστικών. Η διαφορά οφείλεται πιθανότατα στο ότι οι αλγόριθμοι που μόλις αναφέρθηκαν επιτρέπουν ένα σφάλμα, σε βαθμό που εμείς καθορίζουμε, και έτσι μπορεί το επίπεδο να κάνει πιο σωστό fit χωρίζοντας τα δεδομένα και χωρίς να λαμβάνει υπόψιν του μερικά στοιχεία που μπορεί να έχουν λίγο πιο ακραίες τιμές σε μία κλάση από τα υπόλοιπα στοιχεία της.

B.

Χρησιμοποιώντας την επιλογή ‘OptimizeHyperparameters’ μπορώ να ελέγξω όλους τους δυνατούς συνδυασμούς που μπορώ να κάνω και να επιλέξω τον βέλτιστο.

Για τρία χαρακτηριστικά η βέλτιστη απόδοση έχει σφάλμα 2.22% για την ταξινόμηση του ίδιου του training set, 6.67% για την ταξινόμηση του validation set και 0% για την ταξινόμηση του test set.




```
Best estimated feasible point (according to models):
    BoxConstraint    KernelScale    KernelFunction    PolynomialOrder
    _____________    ___________    ______________    _______________

    0.00297          NaN            polynomial        4

Estimated objective function value = 0.054903
Estimated function evaluation time = 0.19095


Train_Error =

    0.0222


Validation_Error =

    0.0667


Test_Error =

     0
```

Για τέσσερα χαρακτηριστικά η βέλτιστη απόδοση έχει σφάλμα 0% για την ταξινόμηση του ίδιου του training set, 6.67% για την ταξινόμηση του validation set και 0% για την ταξινόμηση του test set.

```
Best estimated feasible point (according to models):
    BoxConstraint    KernelScale    KernelFunction    PolynomialOrder
    _____________    ___________    ______________    _______________

    0.1086           NaN            polynomial        2

Estimated objective function value = 0.01217
Estimated function evaluation time = 0.10133


Train_Error =

     0


Validation_Error =

    0.0667


Test_Error =

     0
```

Συγκρίνοντας την ταξινόμηση των τεσσάρων χαρακτηριστικών με την ταξινόμηση των λουλουδιών χρησιμοποιώντας τις μεθόδους που προαναφέρθηκε ότι χρησιμοποιήθηκαν στην εργασία 5, χρησιμοποιώντας αντίστοιχα και τα τέσσερα χαρακτηριστικά για να διαχωρίσουμε το Iris-versicolor από τα δύο άλλα λουλούδια, παρατηρούμε ότι πλέον το svm ταξινομεί τα δεδομένα καλύτερα. Είναι λογικό, καθώς μία μη γραμμική μέθοδος ταξινόμησης σίγουρα μπορεί να πετύχει μικρότερο σφάλμα από μία που είναι αυστηρά γραμμική.

Όπως περιμέναμε, τα αποτελέσματα της non-linear ταξινόμησης είναι πολύ καλύτερα απ' ότι αυτά της linear.

Παρατηρούμε επίσης ότι χρησιμοποιώντας όλα τα διαθέσιμα χαρακτηριστικά ο σφάλμα μειώνεται, όπως και θα περιμέναμε, αφού χρησιμοποιείται περισσότερη πληροφορία, άρα υπάρχουν περισσότεροι τρόποι να χωριστούν τα δεδομένα.

Γ.

Στην ταξινόμηση με τη χρήση τριών χαρακτηριστικών με τη χρήση γραμμικής συνάρτησης στο SVM το σφάλμα ήταν 5.56% για την ταξινόμηση του ίδιου του training set, 6.67% για την ταξινόμηση του validation set και 3.33% για την ταξινόμηση του test set.

Χρησιμοποιώντας και τα τέσσερα χαρακτηριστικά, ο διαχωρισμός έχει σφάλμα 0% για την ταξινόμηση του ίδιου του training set. 6.67% είναι το σφάλμα ταξινόμησης του validation set και 0% για την ταξινόμηση του test set.

Συγκρίνοντας την ταξινόμηση των τεσσάρων χαρακτηριστικών με την ταξινόμηση των λουλουδιών χρησιμοποιώντας τις μεθόδους που προαναφέρθηκε ότι χρησιμοποιήθηκαν στην εργασία 5, παρατηρούμε ότι πλέον το svm ταξινομεί τα δεδομένα λίγο χειρότερα. Είναι λογικό, καθώς γνωρίζουμε ότι όλες οι μέθοδοι που εξετάζονται είναι γραμμικές και τα δεδομένα είναι γραμμικά διαχωρίσιμα. Φαίνεται ότι στη συγκεκριμένη περίπτωση οι μέθοδοι της εργασίας 5 ήταν λίγο πιο αποδοτικές, χωρίς κάποια έντονη ωστόσο διαφορά.

Δ.

Χρησιμοποιώντας την επιλογή ‘OptimizeHyperparameters’ μπορώ να ελέγξω όλους τους δυνατούς συνδυασμούς που μπορώ να κάνω και να επιλέξω τον βέλτιστο.

Για τρία χαρακτηριστικά η βέλτιστη απόδοση έχει σφάλμα 2.22% για την ταξινόμηση του ίδιου του training set, 6.67% για την ταξινόμηση του validation set και 3.33% για την ταξινόμηση του test set.

```
Best estimated feasible point (according to models):
    BoxConstraint    KernelScale    KernelFunction    PolynomialOrder
    _____________    ___________    ______________    _______________

    999.79           NaN            linear            NaN

Estimated objective function value = 0.033272
Estimated function evaluation time = 0.24212


Train_Error =

    0.0222


Validation_Error =

    0.0667


Test_Error =

    0.0333
```

Για τέσσερα χαρακτηριστικά η βέλτιστη απόδοση έχει σφάλμα 0% για την ταξινόμηση του ίδιου του training set, 6.67% για την ταξινόμηση του validation set και 3.33% για την ταξινόμηση του test set.

```
Best estimated feasible point (according to models):
    BoxConstraint    KernelScale    KernelFunction    PolynomialOrder
    _____________    ___________    ______________    _______________

    963.76           0.55321        gaussian          NaN

Estimated objective function value = 0.022552
Estimated function evaluation time = 0.25576


Train_Error =

     0


Validation_Error =

    0.0667


Test_Error =

    0.0333
```

Συγκρίνοντας την ταξινόμηση με βάση τρία χαρακτηριστικά με αυτή με βάση τέσσερα, παρατηρούμε ότι όσο περισσότερα χαρακτηριστικά δίνουμε τόσο καλύτερα είναι τα αποτελέσματα.

Συγκρίνοντας την ταξινόμηση των τεσσάρων χαρακτηριστικών με την ταξινόμηση των λουλουδιών χρησιμοποιώντας τις μεθόδους που προαναφέρθηκε ότι χρησιμοποιήθηκαν στην εργασία 5, παρατηρούμε ότι πλέον το svm ταξινομεί τα δεδομένα με περίπου το ίδιο σφάλμα και ίσως πάλι ελάχιστα χειρότερα. Τα δεδομένα είναι όπως γνωρίζουμε γραμμικά διαχωρίσιμα οπότε ίσως σε αυτή την περίπτωση αυτός να είναι ο λόγος που οι linear αλγόριθμοι της εργασίας 5 είναι πιο αποδοτικοί.

Όπως περιμέναμε, τα αποτελέσματα της non-linear ταξινόμησης είναι πολύ καλύτερα απ’ ότι αυτά της linear.