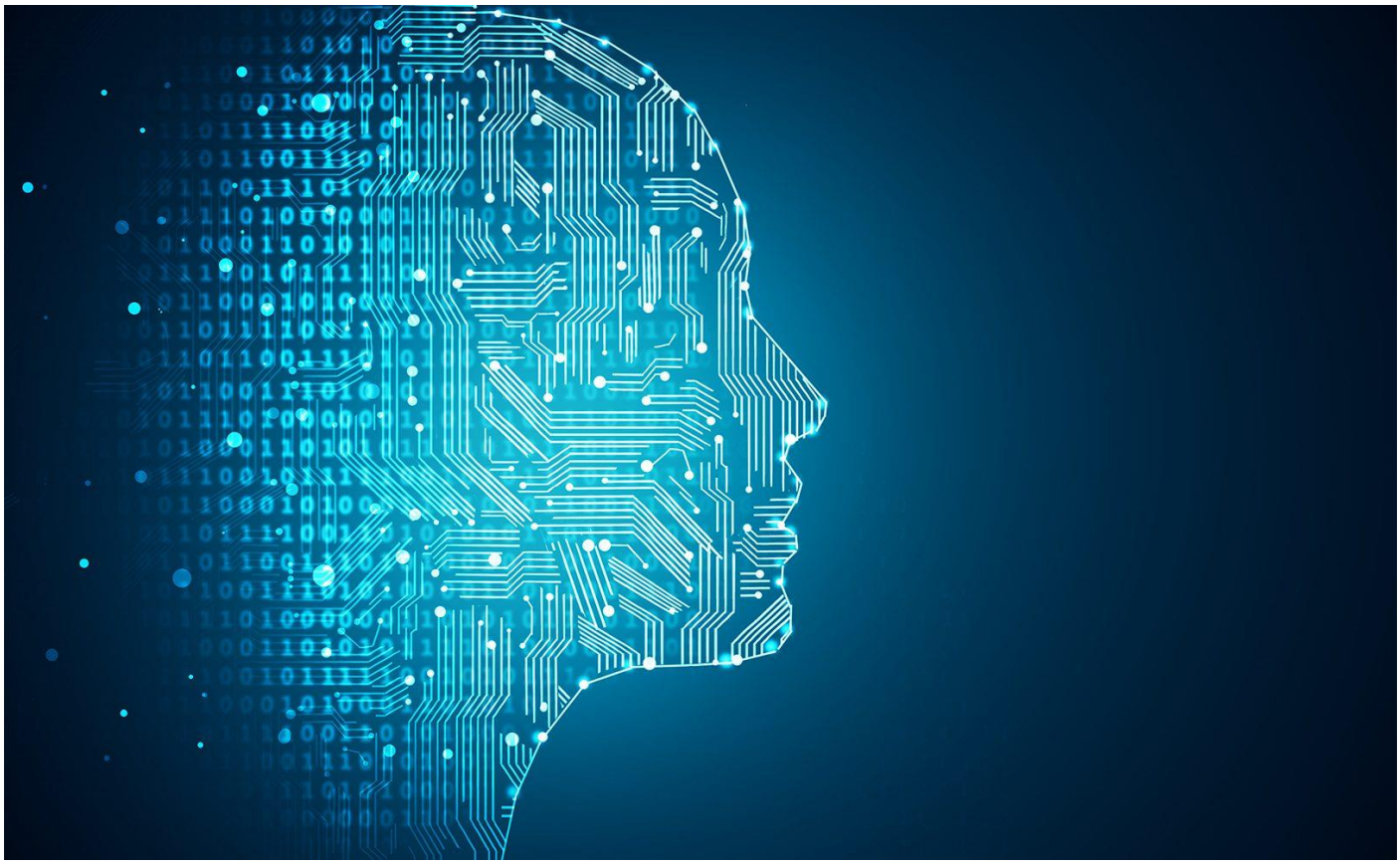


Αλβανάκη Παρασκευή
Α.Μ 57286

# Αναγνώριση Προτύπων
# Report 5

## ΕΚΦΩΝΗΣΗ

Το IRIS data set (δες http://en.wikipedia.org/wiki/Iris_flower_data_set) περιέχει μετρήσεις της μορφής: (μήκος σέπαλου, πλάτος σέπαλου, μήκος πετάλου, πλάτος πετάλου) σε cm για 150 φυτά iris (είδος κρίνου, αγριόκρινο). Από αυτά τα 150 φυτά, 50 είναι Iris Setosa (ω1 ), 50 είναι Iris Versicolour (ω2 ) και 50 είναι Iris Virginica (ω3 ). Γνωρίζουμε ότι μόνο η μία (Iris Setosa ) από τις άλλες δυο κλάσεις είναι γραμμικά διαχωρίσιμη.

A. Να βρεθεί ένας γραμμικός ταξινομητής που να χωρίζει την Iris Setosa από τις άλλες 2 κατηγορίες με το batch perceptron (αλγόριθμος 3), και με το batch relaxation with margin (αλγόριθμος 6).

B. Να βρεθεί ένας γραμμικός ταξινομητής που να χωρίζει την Iris Setosa (ω1 ) από τις άλλες 2 κατηγορίες (ω2, ω3) χρησιμοποιώντας την μέθοδο των ελαχίστων τετραγώνων με χρήση του ψευδοαντιστρόφου, καθώς και με την επαναληπτική μέθοδο LMS (Windrow-Hopf) (αλγόριθμος 8).

C. Να βρεθεί ένας γραμμικός ταξινομητής που να χωρίζει την Iris Versicolour (ω2 ) από την Iris Virginica (ω3 ) χρησιμοποιώντας την μέθοδο των ελαχίστων τετραγώνων με χρήση του ψευδοαντιστρόφου (LS) καθώς και με την επαναληπτική μέθοδο του Ho-Kashyap (αλγόριθμος 9).

D. Να βρείτε τους γραμμικούς ταξινομητές και των τριών κατηγοριών και των 3 κατηγοριών χρησιμοποιώντας την μέθοδο των ελαχίστων τετραγώνων με χρήση του ψευδοαντιστρόφου (LS) και όλα τα χαρακτηριστικά (1,2,3,4)

E. Επαναλάβατε το D για τους χώρους (1,2,3) και (2,3,4) (1=μήκος σέπαλου, 2=πλάτος σέπαλου, 3=μήκος πετάλου,4= πλάτος πετάλου) και δείξτε τα υπερεπίπεδα διαχωρισμού στον χώρο που έχετε καλύτερα αποτελέσματα.

F. Προσπαθήστε να βρείτε τους γραμμικούς ταξινομητές και των τριών κατηγοριών, χρησιμοποιώντας την δομή Kesler.

## A)

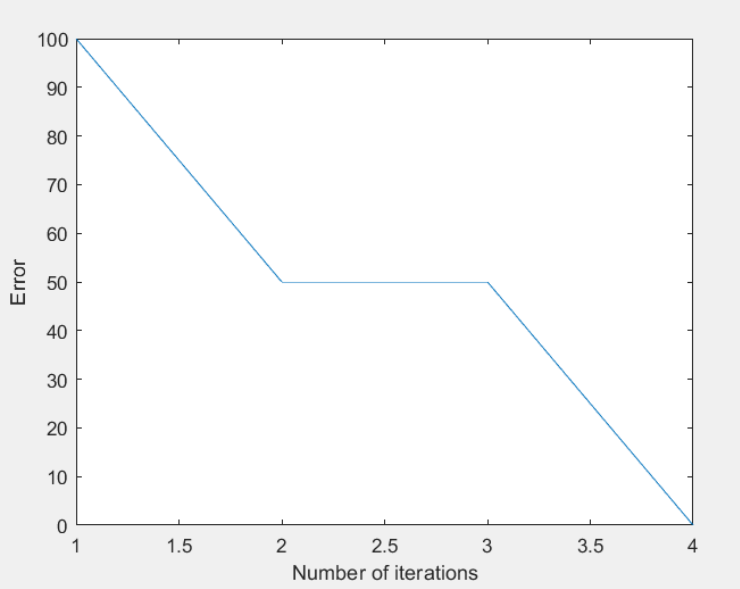
Η διαδικασία που ακολουθείται είναι η εξής:

Αρχικοποιούμε τα βάρη σε μια αρχική τιμή. Κάθε φορά που κάποιο παράδειγμα είναι misclassified κάνουμε update το gradient για να ξέρουμε προς ποια "κατεύθυνση" θα κινηθούμε. Αφού έχουμε κάνει iterate όλα τα παραδείγματα κάνουμε update τα βάρη με βάση το gradient που υπολογίστηκε κινούμενοι στην κατεύθυνση αρνητικής κλίσης κατά μία ποσότητα η(κ) (learning rate) και η διαδικασία επαναλαμβάνεται μέχρι ή να φτάσουμε το μέγιστο βαθμό επαναλήψεων ή όλα τα παραδείγματα να είναι σωστά ταξινομημένα.

Με τον **Batch Perceptron** χωρίσαμε τα δεδομένα σε 2 κατηγορίες και έτσι με βάση τα 4 χαρακτηριστικά δημιουργείται ένα υπερεπίπεδο που διαχωρίζει τις 2 κλάσεις(Iris Setosa και non-Iris Setosa) , με το w να ορίζει το σημείο διαχωρισμού για κάθε χαρακτηριστικό. Ως αναδρομική σχέση έχουμε την $a(k+1) = a(k) + n(k)\Sigma y$ με $y_k$ το σύνολο των δειγμάτων που δεν έχουν ταξινομηθεί σωστά από το ακ.

Όταν το σφάλμα μηδενίστηκε πήραμε το εξής w:1.0000 ⇒w0 , -0.2560 ⇒w1,1.5460⇒w2, - 2.4420 ⇒w3, -0.4320⇒w4 .
Ή σε μορφή διανύσματος:
w= [ 1.0000 -0.2560 1.5460 -2.4420 -0.4320];

Το plot των λάθος ταξινομημένων παραδειγμάτων ως προς τα iterations είναι το παρακάτω:




Συνολικά ο αλγόριθμος εκτελείται 4 φορές και προφανώς ο αριθμός των misclassified elements είναι 0 αφού ο αλγόριθμος τερματίζει όταν δεν υπάρχουν λάθος ταξινομημένα παραδείγματα.

Με τον αλγόριθμο **Batch Relaxation with Margin** ως αναδρομική σχέση έχουμε την $a(k+1) = a(k) + n(k)\Sigma \frac{b-a^t y}{\|y\|^2} y$ και πάλι κανουμε iterate όλα τα πρότυπα καταχωρούμε αυτά που έχουν ταξινομηθεί λανθασμένα στο Υκ κάνουμε update το α και επαναλαμβάνουμε τη διαδικασία. Ουσιαστικά με το batch relaxation with margin και συγκεκριμένα με την συνάρτηση κριτηρίου του καταφέρνουμε και να μην χρειάζεται να περιμένουμε την gradient να φτάσει ακριβώς το boundary point a=0 και να μην κυριαρχείται η συνάρτηση κριτηρίου από το μεγαλύτερο sample of vectors.

Παρατηρούμε πως με αυτό τον αλγόριθμο το πρόβλημα λύνεται γρηγορότερα όπως είναι και αναμενόμενο. Συγκεκριμένα γίνεται classification σε 1 επανάληψη.

## Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Α

```
clear;
clc;
read_the_dataset();
X=charact;
X=[ones(1,150);X];

y=ones(1,150);
y(51:150)=-1;
```

```
w=[1 1 1 1 1]';
rho=0.1;
[w1,iter,mis_class,wrong_class]=perce_with_plot(X,y,w,rho);
figure('Name','Perceptron misclassifications');
plot([1:iter],wrong_class);
xlabel('Iterations');
ylabel('Misclassifications');
disp("Weights when error reaches 0");
disp(w1);
disp("Number of iterations for the batch perceptron");
disp(iter);
disp("Misclassified elements ");
disp(mis_class);
X=charact;
y = [ones(1,50) (-1)*ones(1,50) (-1)*ones(1,50)];

[l,N]=size(X);
[test_targets, a,iter] = Relaxation_BM(X,y, X, [2000 1 0.01]);
disp('Misclassified elements with batch relaxation with margin')
mis_clas=N-sum(test_targets)
disp('Number of iterations');
iter
```

## B)

**Μέθοδος Ελαχίστων Τετραγώνων**

Η μέθοδος των ελαχίστων τετραγώνων μετατρέπει το πρόβλημα επίλυσης ενός συνόλου γραμμικών ανισοτήτων σε πρόβλημα επίλυσης γραμμικών εξισώσεων. Προσπαθούμε ουσιαστικά να ελαχιστοποιήσουμε μια συνάρτηση κριτηρίου που βασίζεται στα ελάχιστα τετράγωνα Πλέον, αντί να ψάχνουμε να βρούμε α τέτοια ώστε $a\ tyi > 0$ για κάθε πρότυπο $yi$ , θέλουμε να βρούμε α τέτοια ώστε $a\ tyi = bi$ . Η μέθοδος των ελαχίστων τετραγώνων στηρίζεται στο γεγονός ότι τα πρότυπα n είναι περισσότερα από τις διαστάσεις d+1, με αποτέλεσμα το σύστημα να μην έχει ακριβή λύση. Επομένως προσπαθούμε να ελαχιστοποιήσουμε το τετράγωνο του μήκους του διανύσματος σφάλματος e=Ya-b. Η συνάρτηση κριτηρίου που προσπαθούμε να ελαχιστοποιήσουμε είναι της μορφής $J = \sum_{i=1}^{n}(a^t y_i - b_i)^2$ . Πρέπει οπότε να ισχύει $\nabla Js\ (a) = 0 \rightarrow Y^t Ya = Y^t b$. Αν $Y^t Y$είναι ομαλός τότε γίνεται: $a$ =( $Y^t Y)^{-1}\ Y^t b$.Στη μέθοδο ελαχίστων τετραγώνων δεν συνεισφέρουν μόνο τα misclassified πρότυπα αλλά όλα τα πρότυπα.

Ως αποτέλεσμα παίρνουμε error=0 και συνεπώς γίνεται ο διαχωρισμός της κλάσης Iris Setosa από τις άλλες 2 κατηγορίες.

**Επαναληπτική μέθοδος LMS (Windrow-Hopf)**

Η συνάρτηση κριτηρίου μπορεί να ελαχιστοποιηθεί μέσω αναδρομικών αλγορίθμων χωρίς να απαιτείται η αντιστροφή πινάκων. Αυτό ουσιαστικά μας γλυτώνει τον υπολογισμό του ψευδοαντιστρόφου και αποφεύγει το πρόβλημα που προκύπτει αν υπάρξει det=0 στον ψευδοαντιστροφο.Τέλος χρησιμοποιώντας ένα δείγμα σε κάθε βήμα προκύπτει ο αλγόριθμος Windrow-Hopf.Με τον αλγόριθμο αυτό χρειάστηκαν 253 iterations και ταξινομήθηκαν σωστά 147 πρότυπα. Αυτό συμβαίνει καθώς το πρόβλημα είναι μη γραμμικά διαχωρίσιμο και συνεπώς δεν έχουμε 100%επιτυχια στο classification.

## Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Β

```
clear;
clc;
read_the_dataset();

X=charact;
X=[X;ones(1,150)];
y=ones(1,150);
y(51:150)=-1;
[M,N]=size(X);
C=0.0001;

a=inv(X*X'+C*eye(M))*(X*y');

mis_clas=sum(((a'*X).*y)<0);
disp('LS error');
disp(mis_clas);
disp('LMS');
Xtrain=[X(:,1:45) X(:,51:95) X(:,101:145)];
Xtest=[X(:,46:50) X(:,96:100) X(:,146:150)];
y=zeros(1,135);
y(46:135)=1;
[test_targets, a,
updates]=LMS_cc(Xtrain(2:5,:),y',Xtest(2:5,:),10000,0.01,0.001);
disp(a);
cla=150-sum(test_targets);
disp('Correctly classified samples');
disp(cla);
```

## Γ)

Η μέθοδος ελαχίστων τετραγώνων περιεγράφηκε εκτενώς στο προηγούμενο ερώτημα .Στο διαχωρισμό του Iris Versicolour από την Iris Virginica η παραπάνω μέθοδος μας ταξινομεί λάθος 3 πρότυπα.

Ως αποτέλεσμα παίρνουμε error=0 και συνεπώς γίνεται ο διαχωρισμός της κλάσης Iris Setosa από τις άλλες 2 κατηγορίες.

**Επαναληπτική μέθοδος Ho-Kashyap**

Το perceptron και οι τεχνικές ελαχίστων τετραγώνων βρίσκουν διαχωριστικά διανύσματα αν τα δείγματα είναι γραμμικά διαχωρίσιμα ωστόσο δε συγκλίνουν για μη διαχωρίσιμες κλασεις. Ο αλγόριθμος Ho-Kashyap φροντίζει το β να μην συγκίνει στο 0 θέτοντας όλες τις θετικές συνιστώσες της κλίσης ίσες με 0.Συνεπώς δεν θέτουμε εμείς χειροκίνητα το β αλλά εξάγεται από το ίδιο το αποτέλεσμα.

Το αποτέλεσμα που παίρνουμε από τον παραπάνω είναι και πάλι 4 λάθος ταξινομήσεις.

## Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Γ

```
clear;
clc;
read_the_dataset();
X=charact(:,51:150);
X=[ones(1,100);X];
y=ones(1,100);
y(51:100)=-1;
```

```
C=0.01;
[M,N]=size(X);
a=inv(X*X'+C*eye(M))*(X*y');

mis_clas=sum(((a'*X).*y)<0);
disp('LS misclassifications');
disp(mis_clas);

[a,b]=Ho_Kashyap_cc(X,y, 0, 100,10,0.1);
mis_clas=sum(((a'*X).*y)<0);
disp('Kashyap misclassifications');
disp(mis_clas);
```

## Δ)

Η μέθοδος ελαχίστων τετραγώνων περιεγράφηκε εκτενώς στο προηγούμενο ερώτημα .Στο ερώτημα αυτό ουσιαστικά έχουμε να κάνουμε με multiclass classification.Οπότε για αν εφαρμόσουμε LS ουσιαστικά κάνουμε one versus all για κάθε κλάση και ταξινομούμε το κάθε πρότυπο σε αυτή με τη μεγαλύτερη τιμή. Η παραπάνω μέθοδος μας δίνει τα εξής αποτελέσματα

```
LS wrong classification for 1st class
     0

LS wrong classification for 2nd class
    39

LS wrong classification for 3rd class
    11

Total Error
    0.8467
```

Ως αποτέλεσμα παίρνουμε συνολικά error=0,08467 και παρατηρούμε πως η μέθοδος έχει 0 λάθος ταξινομημένα πρότυπα στην 1η κλάση που είναι γραμμικά διαχωρίσιμη από τις υπόλοιπες.

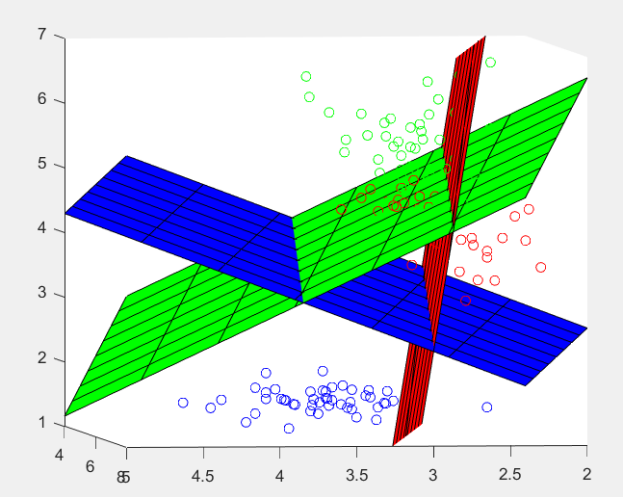
### Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Δ

```
clear;
clc;
read_the_dataset();
X=charact;
X=[ones(1,150);X];
y = [ones(1,50) ones(1,50).*(2) ones(1,50).*(3)];
figure(1), plot3(X(1,y==1),X(2,y==1),X(3,y==1),'r.',...
    X(1,y==2),X(2,y==2),X(3,y==2),'g.',...
    X(1,y==3),X(2,y==3),X(3,y==3),'b.')
axis equal; hold on; axis(axis); XA=axis;
```

```
y1=ones(1,150);
y2=ones(1,150);
y3=ones(1,150);
y1(51:150)=-1;
y2([1:50,101:150])=-1;
y3(1:100)=-1;
C=0.01;
[M,N]=size(X);
a1=inv(X*X'+C*eye(M))*(X*y1');

mis_clas1=sum(((a1'*X).*y1)<0);
disp('LS wrong classification for 1st class');
disp(mis_clas1);
a2=inv(X*X'+C*eye(M))*(X*y2');

mis_clas2=sum(((a2'*X).*y2)<0);
disp('LS wrong classification for 2nd class');
disp(mis_clas2);

a3=inv(X*X'+C*eye(M))*(X*y3');
mis_clas3=sum(((a3'*X).*y3)<0);
disp('LS wrong classification for 3rd class');
disp(mis_clas3);
[max,class_est]=max([a1'*X;a2'*X;a3'*X]);
correct_sse_class=sum(class_est==y)/150;
disp('Total Error');
disp(correct_sse_class);
```

## E)

- Το υπερεπίπεδο για το χώρο (1,2,3) είναι το εξής :



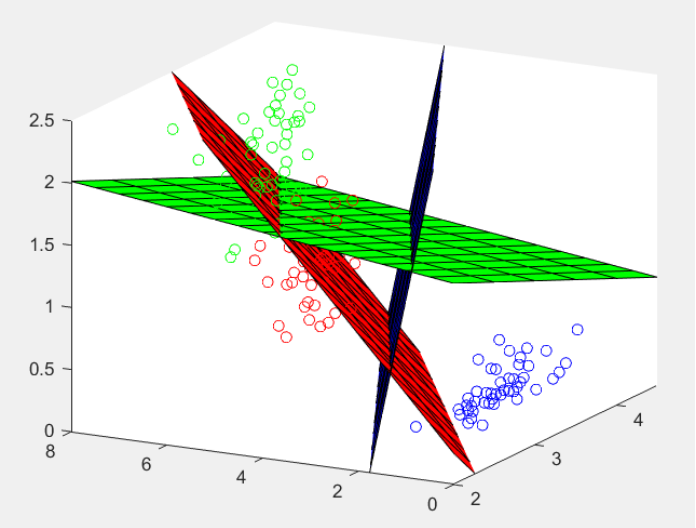

### Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Ε1

```
clear; clc;
close all

load fisheriris.mat
```

```
% Minimum Square Error - MSE
% 1-2
y = [ones(1,150)' meas(1:150,1:3)];
b1 = [ones(1,50) (-1)*ones(1,100)]';
% a12=pinv(y'*y)*y'*b12;
a1=SSErr(y',b1',0.00001);
mis_clas=sum((a1'*(y'.*b1')<0))
a1

% 2-3
% y = [ones(1,100)' meas(51:end,1:3)];
b2 = [(-1)*ones(1,50) ones(1,50) (-1)*ones(1,50)]';
% a23=pinv(y'*y)*y'*b23;
a2=SSErr(y',b2',0.00001);
mis_clas=sum((a2'*(y'.*b2')<0))
a2

% 1-3
% y2 = [ones(1,150)' [meas(1:50,1:3);meas(101:end,1:3)]];
b3 = [ones(1,100) (-1)*ones(1,50)]';
% a13=pinv(y'*y)*y'*b13;
a3=SSErr(y',b3',0.00001);
mis_clas=sum((a3'*(y'.*b3')<0))
a3


train = meas(:,1:3)';
labels=[ones(1,50) 2*ones(1,50) 3*ones(1,50)];
% 1. Plot X1, where points of different classes are denoted by
different colors,
figure;
plot3(train(1,labels==1),train(2,labels==1),train(3,labels==1),'bo'
,...
train(1,labels==2),train(2,labels==2),train(3,labels==2),'ro',...
train(1,labels==3),train(2,labels==3),train(3,labels==3),'go')
 axis(axis);

figure;
plot3(train(1,labels==1),train(2,labels==1),train(3,labels==1),'bo'
,...
train(1,labels==2),train(2,labels==2),train(3,labels==2),'ro',...
train(1,labels==3),train(2,labels==3),train(3,labels==3),'go')
hold on; axis(axis);
y=[ones(1,150) ;train];

[x1,x2] = meshgrid(0:.5:8);
x3=(-a1(1)-a1(2)*x1-a1(3)*x2)/a1(4);
surf(x1,x2,x3,'FaceColor','b');
x3=(-a3(1)-a3(2)*x1-a3(3)*x2)/a3(4);
surf(x1,x2,x3,'FaceColor','g');
x3=(-a2(1)-a2(2)*x1-a2(3)*x2)/a2(4);
surf(x1,x2,x3,'FaceColor','r');  hold off
```

Παρατηρούμε ότι σε αυτά τα 3 χαρακτηριστικά ξεκάθαρα η $1^{η}$ κλάση είναι linearly seperable ως προς τις άλλες 2 ωστόσο μεταξύ τους η $2^{η}$ και η $3^{η}$ δεν είναι και για αυτό είχαμε και αρκετά misclassifications σε πρότυπα.

- Το υπερεπίπεδο για το χώρο ((2,3,4) είναι το εξής :




Παρατηρούμε ότι και πάλι η 1η κλάση είναι γραμμικά διαχωρίσιμη από τις άλλες 2.Επιπλέον η 2η κλάση είναι περισσότερο "μη γραμμική" από την 3η καθώς περισσότερα πρότυπα της 2ης είναι ταξινομημένα στην 3η.

## Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ Ε2

```
clear; clc;
close all

load fisheriris.mat

% Minimum Square Error - MSE
% 1-2
y = [ones(1,150)' meas(1:150,1:3)];
b1 = [ones(1,50) (-1)*ones(1,100)]';
% a12=pinv(y'*y)*y'*b12;
a1=SSErr(y',b1',0.00001);
mis_clas=sum((a1'*(y'.*b1')<0))
a1

% 2-3
% y = [ones(1,100)' meas(51:end,1:3)];
b2 = [(-1)*ones(1,50) ones(1,50) (-1)*ones(1,50)]';
% a23=pinv(y'*y)*y'*b23;
a2=SSErr(y',b2',0.00001);
mis_clas=sum((a2'*(y'.*b2')<0))
a2

% 1-3
% y2 = [ones(1,150)' [meas(1:50,1:3);meas(101:end,1:3)]];
b3 = [ones(1,100) (-1)*ones(1,50)]';
% a13=pinv(y'*y)*y'*b13;
a3=SSErr(y',b3',0.00001);
mis_clas=sum((a3'*(y'.*b3')<0))
a3
```

```
train = meas(:,1:3)';
labels=[ones(1,50) 2*ones(1,50) 3*ones(1,50)];
% 1. Plot X1, where points of different classes are denoted by
different colors,
figure;
plot3(train(1,labels==1),train(2,labels==1),train(3,labels==1),'bo'
,...
train(1,labels==2),train(2,labels==2),train(3,labels==2),'ro',...
train(1,labels==3),train(2,labels==3),train(3,labels==3),'go')
 axis(axis);

figure;
plot3(train(1,labels==1),train(2,labels==1),train(3,labels==1),'bo'
,...
train(1,labels==2),train(2,labels==2),train(3,labels==2),'ro',...
train(1,labels==3),train(2,labels==3),train(3,labels==3),'go')
hold on; axis(axis);
y=[ones(1,150) ;train];

[x1,x2] = meshgrid(0:.5:8);
x3=(-a1(1)-a1(2)*x1-a1(3)*x2)/a1(4);
surf(x1,x2,x3,'FaceColor','b');
x3=(-a3(1)-a3(2)*x1-a3(3)*x2)/a3(4);
surf(x1,x2,x3,'FaceColor','g');
x3=(-a2(1)-a2(2)*x1-a2(3)*x2)/a2(4);
surf(x1,x2,x3,'FaceColor','r');  hold off
```

## F)

**Kesler**

Στόχος του Kesler είναι ο γραμμικός διαχωρισμός πολλαπλών κλάσεων. Μας επιτρέπει να μετατρέπουμε multiclass error-correction διαδικασίες σε two-category διαδικασίες. Ουσιαστικά τα μεμονωμένα g(x) one-versus all classification που εφαρμόσαμε πριν μπορούν να γραφούν ως $a_i^t \ y - a_j^t \ y > 0$ για κάθε j=2…c.Αυτό το σετ των c-1 ανισοτήτων μπορεί να γραφεί ως (c-1)cd-dimensional weight vectors, $n_{1j}$ με το να σπάσουμε το n σε cd-dimentional subvectors με το i-οστο subvector να είναι y το j-οστο να είναι -y και όλα τα αλλα να είναι 0.

Με Kesler έχουμε error 0.0533 , 8 misclassifications και βάρη :

10.3000 -0.6000 -9.7000

28.7000 -11.7000 -17.0000

-46.3000 14.3000 32.0000

-24.2000 6.1000 18.1000

6.0000 17.0000 -23.0000

Ο ΚΩΔΙΚΑΣ ΜΕ ΤΟΝ ΟΠΟΙΟ ΥΛΟΠΟΙΗΘΗΚΕ ΤΟ F

```
clear; clc;
close all

load fisheriris.mat
```

```
X = meas';
[l,N1]=size(X);
X1=[X; ones(1,N1)];
y1=[ones(1,50) 2*ones(1,50) 3*ones(1,50)];
%--------- Multi class perceptron with Kesler structure -----------
data.X=X;  data.y=y1; %iputs data in format for STPRtool
options.tmax=1000; % max number of iterations
modelKESLER = mperceptron( data, options );
% figure; ppatterns( data ); pboundary( modelKESLER );
% calculates the error on the trainning set
for i=1:l
    w_allKESLER(i,:)=modelKESLER.W(i,:);
end
w_allKESLER(l+1,:)=modelKESLER.b';
[vali,class_est]=max(w_allKESLER'*X1);
errKESLER=sum(class_est~=y1)/N1
mis_clas=sum(class_est~=y1)
```