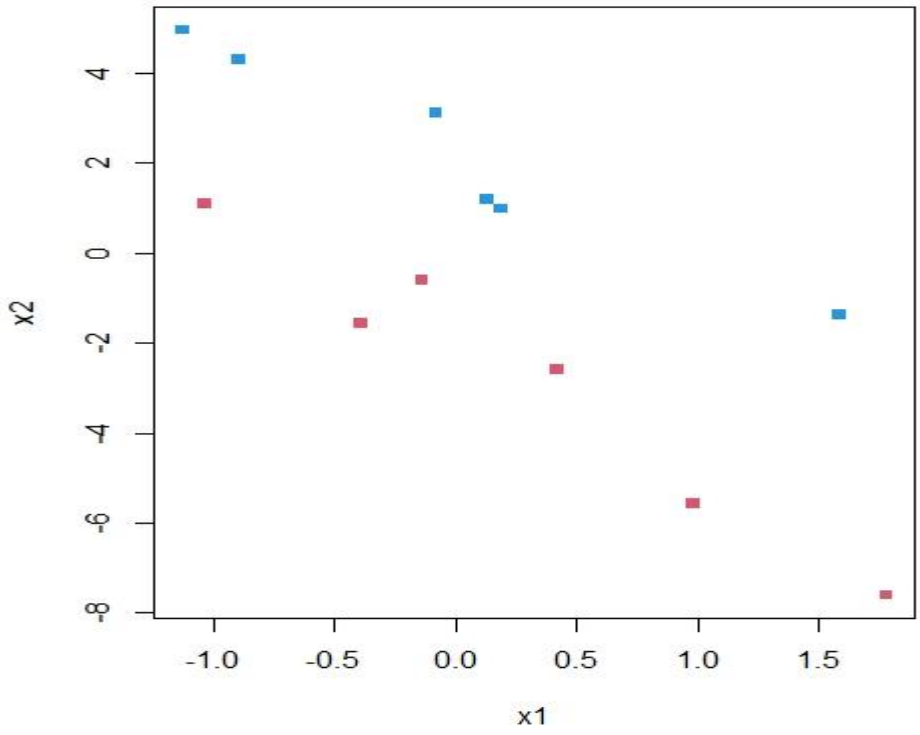
ΧΑΤΖΟΠΟΥΛΟΣ ΓΕΡΑΣΙΜΟΣ

SVM CODE

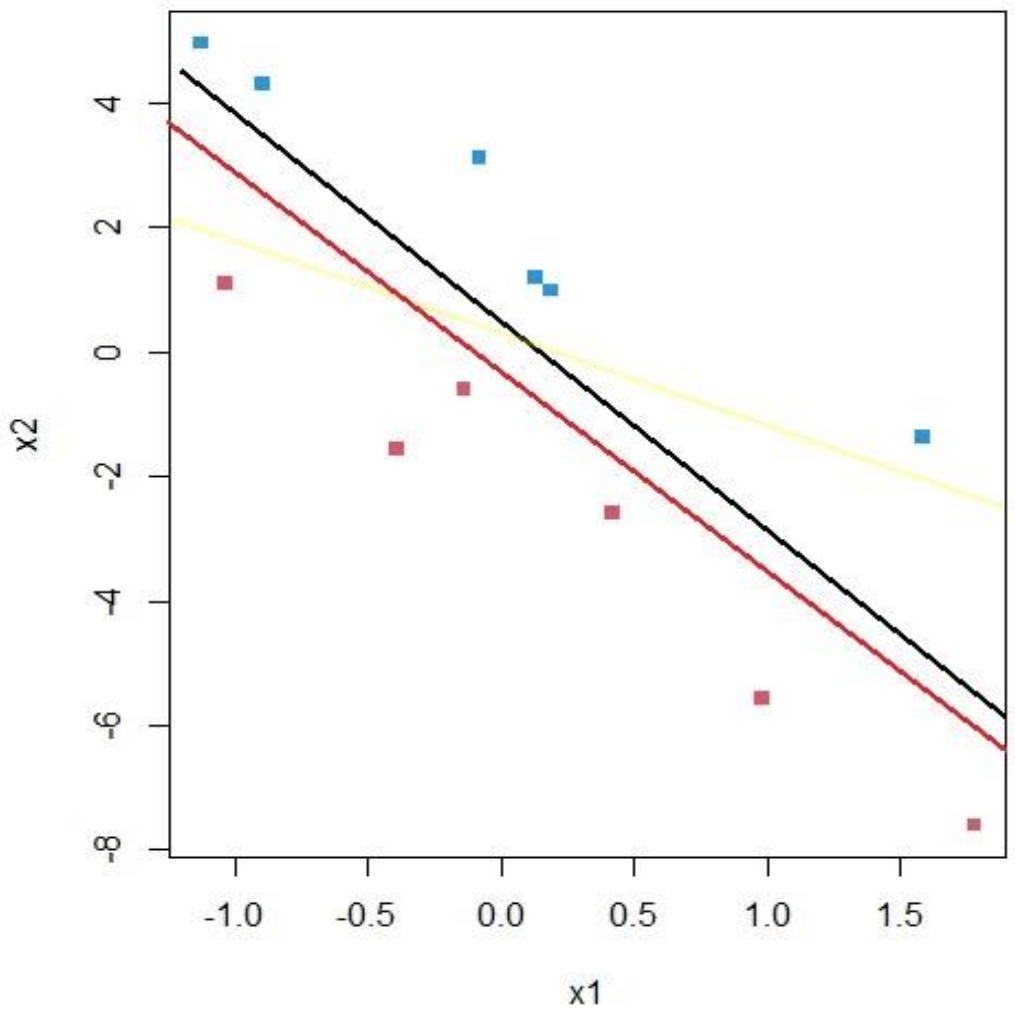
Πριν αρχίσουμε ας δούμε τι είναι το support vector machine

Το Support Vector Machine είναι ένας αλγόριθμος μπορεί να χρησιμοποιηθεί τόσο για προβλήματα ταξινόμησης όσο και για παλινδρόμηση. Ακολουθεί μια τεχνική για τη μετατροπή των δεδομένων και με βάση αυτούς τους μετασχηματισμούς, βρίσκει ένα βέλτιστο όριο μεταξύ των πιθανών σημείων. Δηλαδή κάνει όποιους μετασχηματισμούς δεδομένων για να καταλάβει πώς να διαχωρίσει τα δεδομένα με βάση τις καθορισμένες τιμές των dataset. Ένα παράδειγμα είναι

Ας υποθέσουμε ότι έχουμε ένα σύνολο δεδομένων που έχει δύο ετικέτες (πράσινο και μπλε) και το σύνολο δεδομένων έχει x1 και x2. Θέλουμε έναν classifier που μπορεί να ταξινομήσει το ζεύγος (x1, x2) συντεταγμένων σε πράσινο ή μπλε

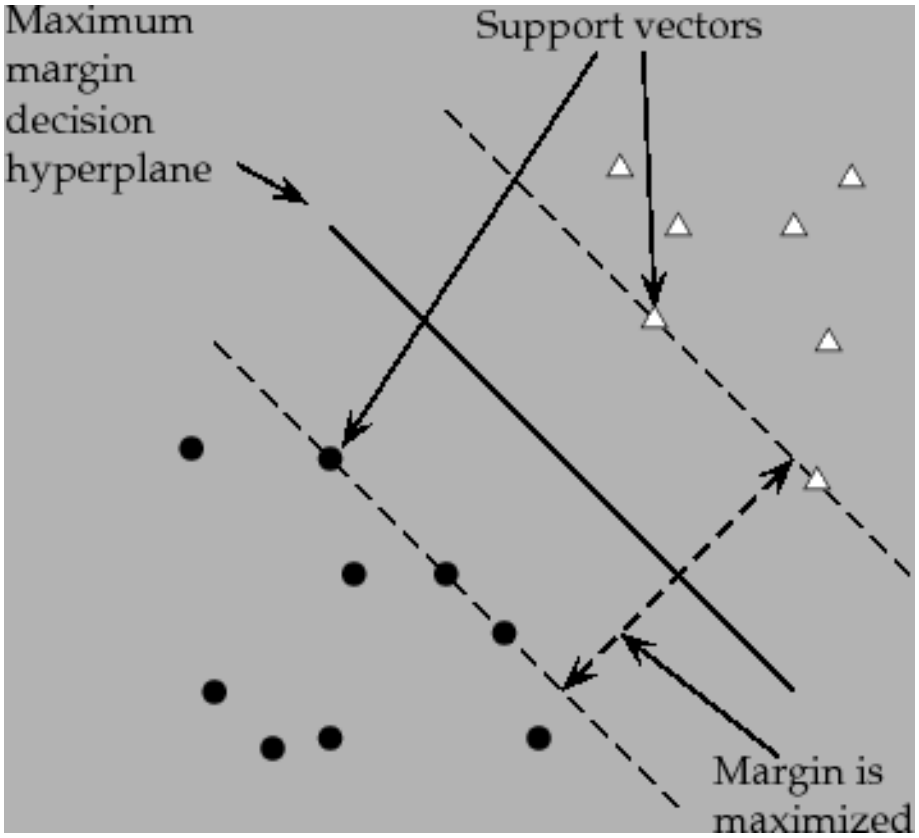


Για να χωρίσουμε τα δεδομένα μπορούμε να τραβήξουμε άπειρες γραμμές πάνω στο σχήμα δηλαδή



Εμείς πρέπει να δημιουργήσουμε έναν αλγόριθμο SVM που θα βοηθά στην εύρεση της καλύτερης γραμμής ή του ορίου αποφάσεων. αυτό το καλύτερο όριο ή περιοχή ονομάζεται hyperplane. Ο αλγόριθμος SVM βρίσκει το πλησιέστερο σημείο των γραμμών και από τις δύο κλάσεις. Η απόσταση μεταξύ των διανυσμάτων και του hyperplane ονομάζεται περιθώριο. Και ο στόχος του SVM μας είναι να μεγιστοποιήσει αυτό το περιθώριο.

Δηλαδή το svm θα μας βοηθήσει να πετύχουμε αυτό το αποτέλεσμα του παρακάτω σχήματος



Γενικώς το svm μπορεί να κάνει classification για τα πάντα π.χ να ξεχωρίσουμε τις φωτογραφίες τους ανθρώπους από τα υπόλοιπα φαινόμενα τριγύρω η ακόμα να διαχωρίζουμε τους ανθρώπους σε μια φωτογραφία αναλόγως με ποιοι είναι λυπημένοι και ποιοι χαρούμενοι κλπ

.

.

.

.

.

.

.

Πριν ξεκινήσουμε με τον αλγόριθμο πρέπει να βρούμε το gradient descent toy svm

Ξέρω ότι το SVM πρόβλημα με το hinge loss με weight είναι,

$$J(\mathbf{w}, \mathbf{b}) = a\sum_{i=1}^{m} \max\left(0,1 - y_i(\mathrm{w} \cdot \mathbf{x}_i + b)\right) + \frac{1}{2}\mathrm{w} \cdot \boldsymbol{w^{\tau}}$$

α=regularization parameter, $\sum_{i=1}^{m} \mathbf{max}\left(\mathbf{0}, \mathbf{1} - \boldsymbol{y_i}(\mathbf{w} \cdot \mathbf{x_i} + \boldsymbol{b})\right) =$
$\boldsymbol{empirical\ loss\ L}$ $(\boldsymbol{how\ we\ fit\ training\ data})$

Θέλουμε να minimize αυτή την ποσότητα. Άρα η 1η παραγωγός ως προς w:

$$\frac{\partial J(\mathbf{w}, \boldsymbol{b})}{\partial \mathbf{w}} = a\sum_{i=1}^{m} [y_i(w \cdot \mathbf{x}_i + b) < 1] \cdot (-y_i \cdot \mathbf{x}_i) + \mathbf{w}$$

Η πρώτη παράγωγος ως προς το β:

$$\frac{\partial J(\mathbf{u}, b)}{\partial b} = a\sum_{i=1}^{m} [y_i(\mathrm{w} \cdot \boldsymbol{x}_i + b) < 1] \cdot (-y_i \cdot b)$$

Αρα W(i+1)=Wi-α*$\sum_{i=1}^{m}$ (y_i*(w $\cdot$ $\boldsymbol{x}_i + b$)<1) * ($-y_i \cdot b$)

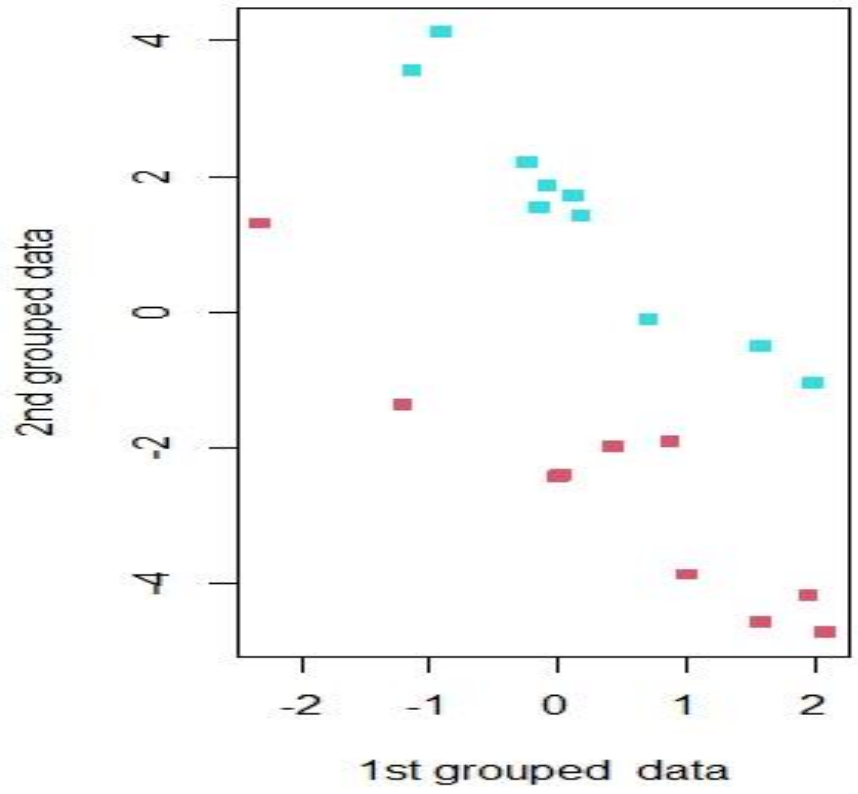
Αρα W(i+1)=W+α*$\sum_{\mathrm{i=1}}^{\mathrm{m}}$ (y_i*(w $\cdot$ $x_i + b$)<1) * $y_i \cdot b$

(εμείς δεν έχουμε weight)

W(i+1)=W+α*$\sum_{\mathrm{i=1}}^{\mathrm{m}}$ (y_i*(w $\cdot$ x_i)<1) * $y_i \cdot b$

MANIPULATED DATA(linear svm)

```
set.seed(2)

n = 10

#φτιαχνω τα δεδομενα μου

a1 = rnorm(n)

a2 = 1 - 2*a1 + 2* runif(n)

b1 = rnorm(n)

b2 = -1 - b1 - 2*runif(n)

x = rbind(matrix(cbind(a1,a2),,2),matrix(cbind(b1,b2),,2))

y <- matrix(c(rep(1,n),rep(-1,n)))

plot(x,col=ifelse(y>0,5,2),pch=".",cex=7,xlab = "1st grouped  data",ylab = "2nd grouped data")
#διαγραμμα
```



```
#εύκολη δουλειά για τον αλγόριθμο με βάση το σχήμα, για να δούμε

df<- cbind(x,y)

colnames(df) <- c("1st grouped  data ","2nd grouped data ","y(label)")  #ας δουμε πως ειναι τα δεδομενα μας

df

X<- cbind(1,x) #make design matrix

n <- nrow(X)  #number of sample

p <- ncol(X) #bias

w_intial <- rnorm(p,-2,1)

w <- w_intial

a <- 0.1

R <- 10 #number of iteration

W <- matrix(w_intial ,nrow = R+1,ncol = p,byrow = T) #matrix put intial guess and the procedure to do gradient descent

X%*%w_intial

indicator<-function(condition) ifelse(condition,1,0)

a <- 3

indicator( a<1 )

(a<4)*1

########################################

###θέλω για κάθε στοιχείο του πινάκα

###w_intial[1]+eta*sum(   ((y*(X%*%W[1,]))<1)*1 * y * X[,1] )

##w_intial[2]+eta*sum(   ((y*(X%*%W[1,]))<1)*1 * y * X[,2] )

##w_intial[3]+eta*sum(   ((y*(X%*%W[1,]))<1)*1 * y * X[,3] )

##########################################

##########################################
```

```
for(i in 1:R){
   for(j in 1:p){
      W[i+1,j]<- W[i,j]+a*sum(((y*(X%*%W[i,]))<1) * y * X[,j] )
   }
}
W
print(   "  first guess  ")
X%*%W[1,]
          [,1]
 [1,] -3.5573228
 [2,] -2.8701595
 [3,] -3.4666078
 [4,] -2.6513306
 [5,] -2.8399145
 [6,] -3.0659239
 [7,] -2.3426885
 [8,] -2.8845726
 [9,] -3.6328706
[10,] -2.4526033
[11,]  1.4258601
[12,] -0.9753591
[13,]  0.8983661
[14,]  0.6136737
[15,] -0.1556877
[16,] -0.4203525
[17,]  2.0234087
[18,]  0.2862600
[19,] -0.6829125
```

[20,] 0.9712251

#να δούμε πως θα κάνει το classification μετά από χ επαναλήψεις ,μετά από δοκιμές
#παρατήρησα ότι γύρω στις 5-6 επαναλήψεις τα ξεχωρίζει άριστα

```
print(   "  the result after 10 times R  ")
```

```
X%*%W[nrow(W),]
```

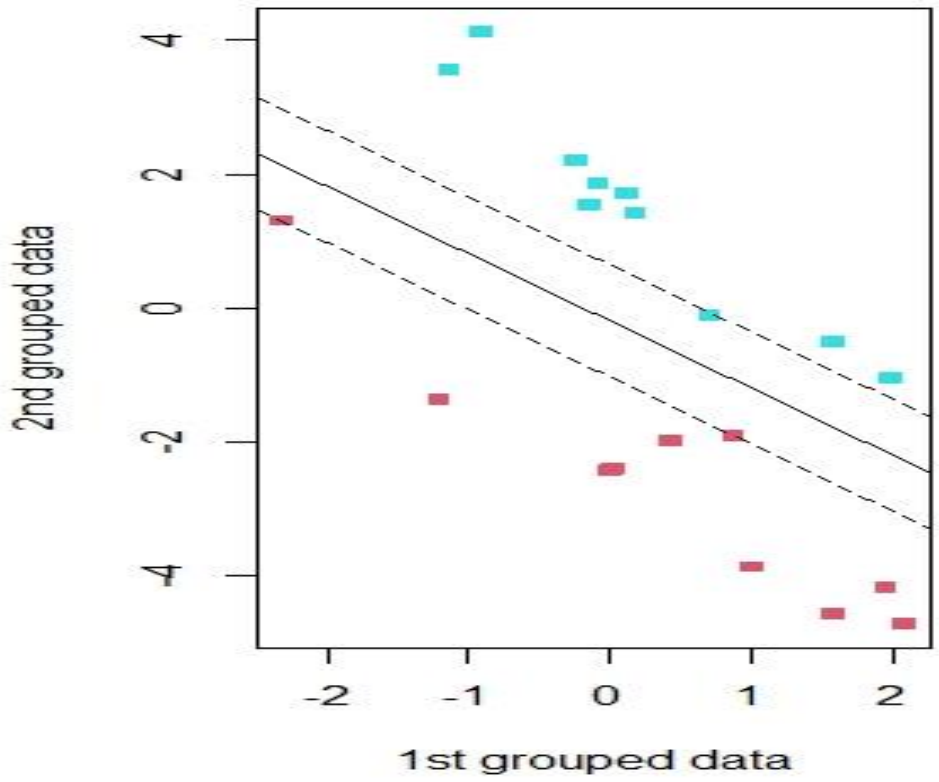
```
          [,1]
 [1,]  4.897806
 [2,]  2.540327
 [3,]  1.985238
 [4,]  3.633129
 [5,]  2.783915
 [6,]  2.927865
 [7,]  1.072008
 [8,]  3.036143
 [9,]  1.824354
[10,]  2.197426
[11,] -1.913767
[12,] -1.417569
[13,] -3.633272
[14,] -4.239620
[15,] -2.300587
[16,] -3.697643
[17,] -4.153136
[18,] -4.329565
[19,] -3.104734
[20,] -3.721542
```

#Παρατηρώ ότι το μοντέλο μ τα ξεχώρισε τέλεια χωρίς κανένα λάθος (προφανώς όμως τα
#έκανα manipulate σε real data θα περιμένω μεγάλη διαφορά)

ΜΠΟΡΩ ΤΩΡΑ ΝΑ ΒΡΩ ΤΟ HYPERPLANE ΜΑΖΙ ΜΕ ΤΟ ΠΕΡΙΘΩΡΙΟ(margin maximized)ΣΤΟ ΣΧΗΜΑ ΠΟΥ ΕΔΕΙΞΑ ΣΤΗΝ ΕΚΦΩΝΗΣΗ

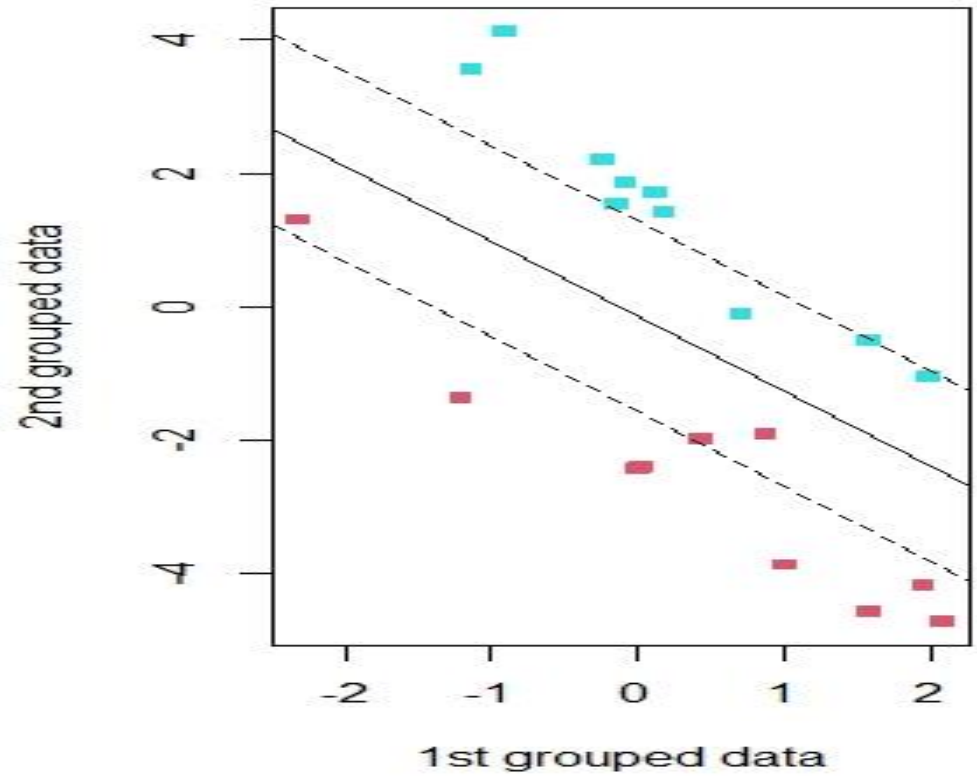
```
gradsvm<- function(x,alpha=0.001,R=750){
  X<- cbind(1,x)#matrix
  n <- nrow(X)  #number of observations
  p <- ncol(X) #bias
  w_intial <- rep(0,p)
  W <- matrix(w_intial ,nrow = R+1,ncol = p,byrow = T) #matrix put intial guess and the
#procedure to do gradient descent
for(i in 1:R){
      for(j in 1:p)
      {
          W[i+1,j]<- W[i,j]+alpha*sum(((y*(X%*%W[i,]))<1) * y * X[,j] ) #gradient descent
      }
  }
  return(W)
}
svm <- function(x){
  result<- gradsvm(x)[nrow(gradsvm(x)),]
  return(result )
}
set.seed(2)
n = 10
a1 = rnorm(n)
a2 = 1 - 2*a1 + 2* runif(n)                    #φτιάχνω τα δεδομένα μου
```

```
b1 = rnorm(n)

b2 = -1 - b1 - 2*runif(n)

x = rbind(matrix(cbind(a1,a2),,2),matrix(cbind(b1,b2),,2))

y <- matrix(c(rep(1,n),rep(-1,n)))

plot(x,col=ifelse(y>0,5,2),pch=".",cex=7,xlab = "1st grouped data",ylab = "2nd grouped data")

result<- svm(x)

abline(-result[1]/result[3],-result[2]/result[3])

abline((1-result[1])/result[3],-result[2]/result[3],lty=5)

abline((-1-result[1])/result[3],-result[2]/result[3],lty=5)
```



ΠΑΡΑΤΗΡΩ ΟΤΙ ΣΤΙΣ 750 ΕΠΑΝΑΛΗΨΕΙΣ ΜΠΟΡΕΙ ΝΑ ΞΕΧΩΡΙΣΕΙ ΤΑ ΔΕΔΟΜΕΝΑ ΜΟΥ ΑΡΙΣΤΑ

ΑΜΑ ΕΚΑΝΑ 150 ΠΑΡΑΤΗΡΗΣΕΙΣ ΘΑ ΜΟΥ ΕΒΓΑΙΝΕ ΤΟ ΠΑΡΑΚΑΤΩ ΣΧΗΜΑ ΠΟΥ ΔΕΝ ΕΙΝΑΙ ΙΚΑΝΟΠΟΙΗΤΙΚΟ



ΕΥΧΑΡΙΣΤΩ