Αλγόριθμοι τεχνητής νοημοσύνης Από τον Perceptron στα νευρωνικά δίκτυα

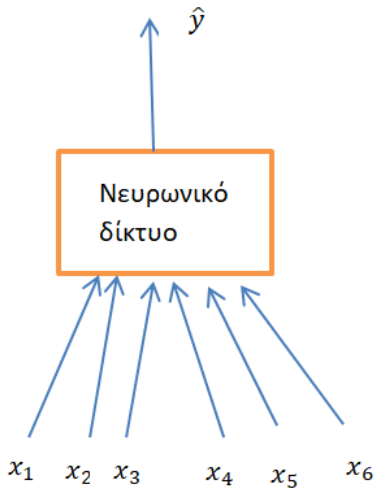
Τσαμπούκα Πετρούλα
ΣΕ Μαθηματικών Δυτικής Θεσσαλονίκης

Πρόβλημα: Διαχωρισμός συμβολοσειρών (μήκους 6 δυαδικών ψηφίων) σε εκείνες που είναι συμμετρικές ως προς το κέντρο τους και σε μη συμμετρικές.

Κλασική αλγοριθμική προσέγγιση (Python)

```
x=[]
for i in range(6):
    s=input("Give a value of 1 or 0:\n"
    x.append(s)
l='true'
for i in range(6):
   if x[i]!=x[5-i]:
      l='false'
      break
if l=='true':
  print("The string is symmetric")
else:
  print("The string is non-symmetric")
```

Επίλυση με τη βοήθεια νευρωνικού δικτύου



Το δίκτυο δέχεται εισόδους που είναι συμμετρικές και μη συμμετρικές συμβολοσειρές και υπόκειται σε μεταβολή των παραμέτρων που καθορίζουν την εσωτερ του κατάσταση όταν η έξοδος του, που παίρνε την τιμή -1 ή 1, διαφοροποιείται από την τιμή εξόδου που έχουμε αποδώσει ανάλογα με την κατηγορία στην οποία ανήκει κάθε συμβολοσειρά, πχ. 1 για τη συμμετρική και -1 για τη μη συμμετρική.

Ref: Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, & the PDP Research Group, Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations. Cambridge: MIT Press/Bradford Books,

Μάθηση υπό επιτήρηση (supervised learning)

Θέλουμε να εκπαιδεύσουμε έναν αλγόριθμο ο οποίος θα δέχεται κάποιες εισόδους, π.χ. ένα σύνολο δεδομένων (ηλικία, BMI, πίεση, κ.λ.π.) που θεωρούμε ότι συνδέονται με κάποια ασθένεια και να παράγει κάποια έξοδο $\hat{y}$. Επιθυμούμε π.χ. όταν το άτομο δεν είναι διαβητικό η έξοδος να είναι -1 ενώ όταν είναι διαβητικό η έξοδος να είναι 1.

Μέχρι να εκπαιδευτεί ο αλγόριθμος να αναγνωρίζει τους ασθενείς με βάση τα δεδομένα που τον τροφοδοτούμε οι έξοδοι που παράγει μπορεί να είναι αντίθετοι σε σχέση την πραγματικότητα, π.χ. 1 για τον μη διαβητικό και -1 για τον διαβητικό. Επειδή κάθε σύνολο δεδομένων συνοδεύεται από την ετικέτα του y, -1 ή 1 που έχει προκύψει από εξετάσεις αίματος (-1 για τον μη διαβητικό και 1 για τον διαβητικό), το σφάλμα του αλγορίθμου μεταξύ αυτού που πρόβλεψε $\hat{y}$ και αυτού που ισχύει y (ετικέτα) οδηγεί σε διόρθωση των τιμών κάποιων παραμέτρων που ορίζουν την εσωτερική κατάσταση του αλγορίθμου ώστε να περιοριστούν αν όχι να μηδενιστούν τα σφάλματα μεταξύ πρόβλεψης και ετικέτας. Στόχος είναι τελικά ο αλγόριθμος να προβλέπει σωστά τις κατηγορίες (κλάσεις) σε άγνωστα δεδομένα.

Είσοδοι: $x \in X \subset \boldsymbol{R}^n$

Έξοδοι: $y \in Y = \{-1,1\}$ για ταξινόμηση

Το σύνολο εκπαίδευσης αποτελείται από ζεύγη σημείων-ετικετών : $S = \{(\vec{x}_1, y_1), \dots, (\vec{x}_i, y_i), \dots\}$

Τα σημεία έχουν διανυσματική αναπαράσταση σε χώρο με εσωτερικό γινόμενο.

Ο αλγόριθμος Perceptron

Ο γραμμικός διαχωρισμός του χώρου των σημείων σε δύο ημιχώρους (halfspaces) γίνεται με τη συνάρτηση

$$f(\vec{x}) = \vec{w} \cdot \vec{x} + b,$$

όπου $\vec{w}$ είναι το διάνυσμα βαρών που προσδιορίζει ως κάθετο σ' αυτό, ευθεία γραμμή ή γενικά υπερεπίπεδο (hyperplane) που διαχωρίζει τα σημεία σε 2 κατηγορίες (κλάσεις).

Το κατώφλι (bias) b καθορίζει την απόσταση της ευθείας ή γενικά του υπερεπιπέδου από την αρχή των αξόνων.

Η $f(\vec{x})$ απεικονίζει κάθε σημείο σε έναν πραγματικό αριθμό (functional margin)

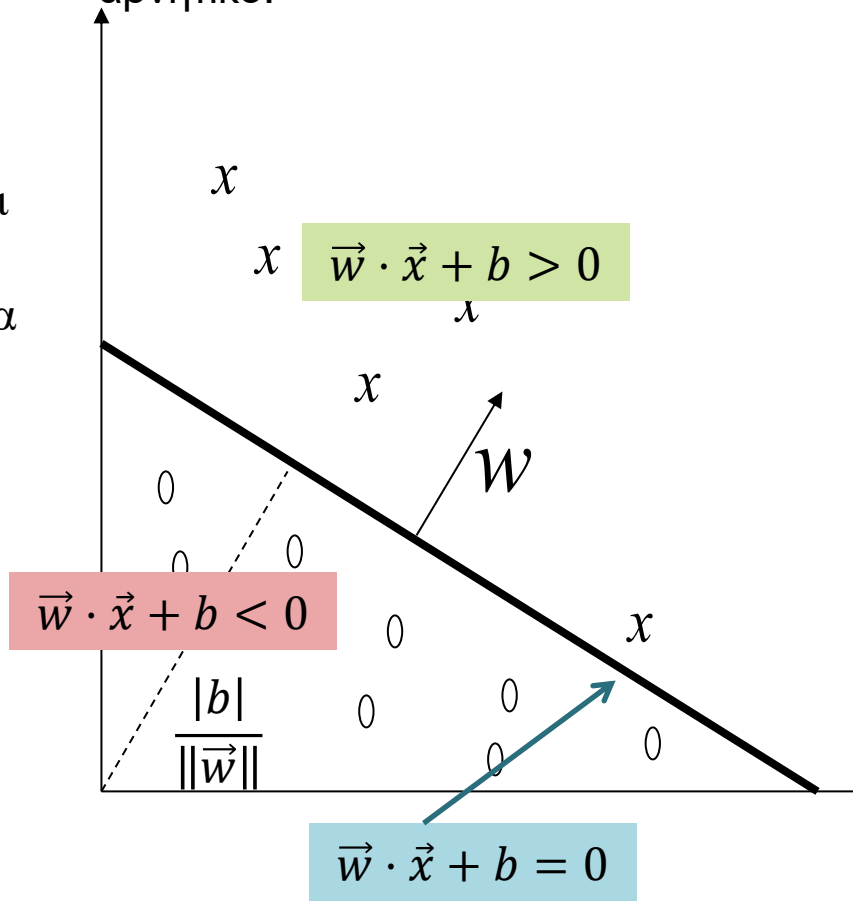
Συνάρτηση απόφασης: $h = sign(f(\vec{x}))$

$h > 0 \Rightarrow \hat{y}(x) = 1 \quad h < 0 \Rightarrow \hat{y}(x) = -1$

Προϋπόθεση: Για να συγκλίνει ο Perceptron πρέπει τα σημεία να είναι γραμμικώς διαχωρίσιμα.

Ο Perceptron παράγει το διάνυσμα βαρών $\vec{w}$ και το (bias) b.

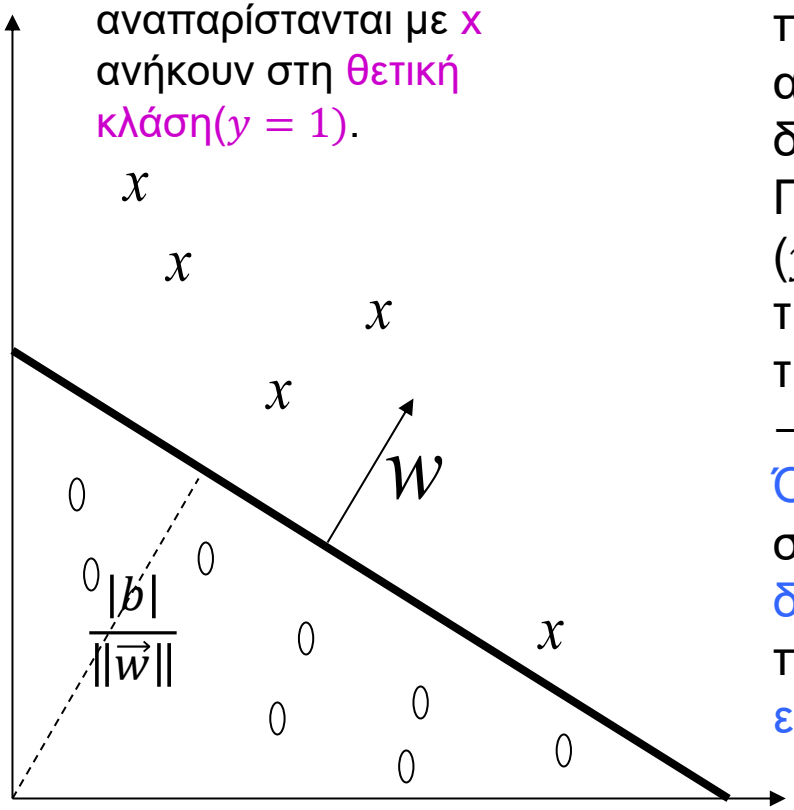
Το $\vec{w}$ δείχνει πάντα κατά τον διαχωρισμό στη θετική κλάση ($y = 1$). Το b στην εικονιζόμενη περίπτωση είναι αρνητικό.



Το $\|\vec{w}\|$ παριστάνει το μέτρο του $\vec{w}$, ενώ το $|b|$ την απόλυτη τιμή του b.

Επεξήγηση του ρόλου της $f(\vec{x}) = \vec{w} \cdot \vec{x} + b$

Τα σημεία που αναπαρίστανται με x ανήκουν στη θετική κλάση($y = 1$).



Η συνάρτηση $f(\vec{x}) = \vec{w} \cdot \vec{x} + b$ παριστάνει ένα είδος απόστασης από την ευθεία γραμμή που διαχωρίζει τα σημεία σε 2 κλάσεις. Για τα σημεία της θετικής κλάσης $(y = 1)$ που διαχωρίζονται σωστά η τιμή είναι θετική ενώ για τα σημεία της αρνητικής κλάσης $(y = -1)$ είναι αρνητική.
Όσο πιο μακριά βρίσκεται ένα σημείο της θετικής κλάσης από τη διαχωριστική ευθεία στον ημιχώρο που δείχνει το $\vec{w}$ τόσο πιο μεγάλη είναι η τιμή της συνάρτησης.

Τα σημεία που αναπαρίστανται με 0 ανήκουν στην αρνητική κλάση $(y = -1)$.

Για την εύρεση της πραγματικής απόστασης θα πρέπει να διαιρέσουμε με το μέτρο $\|\vec{w}\|$ του $\vec{w}$ και να θεωρήσουμε το μοναδιαίο στη διεύθυνσή του. (Θα εξεταστεί στην πορεία).

Ο αλγόριθμος Perceptron (αρχική μορφή)

initialize $\mathbf{w}_0 \leftarrow \mathbf{0}, b_0 \leftarrow 0, k \leftarrow 0$

repeat

error ⟵ false

for i=1..l

if $y_i(\boldsymbol{w}_k \cdot \boldsymbol{x}_k + b_k) \leq 0$ **then**

Αποκαλείται Perceptron update

$\mathbf{w}_{k+1} \leftarrow \mathbf{w}_k + y_i \mathbf{x}_i$

$b_{k+1} \leftarrow b_k + y_i$

$k \leftarrow k+1$

error ⟵ true

end if

end for

until (error==false)

return k, $(\mathbf{w}_k, b_k)$ όπου *k* είναι το πλήθος των λαθών

Εντός του βρόχου for ο αλγόριθμος διατρέχει όλα τα σημεία του συνόλου, πλήθους l και όταν παρουσιάζεται λάθος διαχωρισμού προχωρά σε ανανέωση του διανύσματος βάρους και του bias.

Αν συμβεί σφάλμα εντός του βρόχου for επαναλαμβάνεται η ανακύκλωση των σημείων έως ότου δεν εντοπιστεί κανένα σφάλμα και τότε θεωρούμε ότι ο αλγόριθμος συνέκλινε. Αυτό εξασφαλίζεται από τον βρόχο repeat.

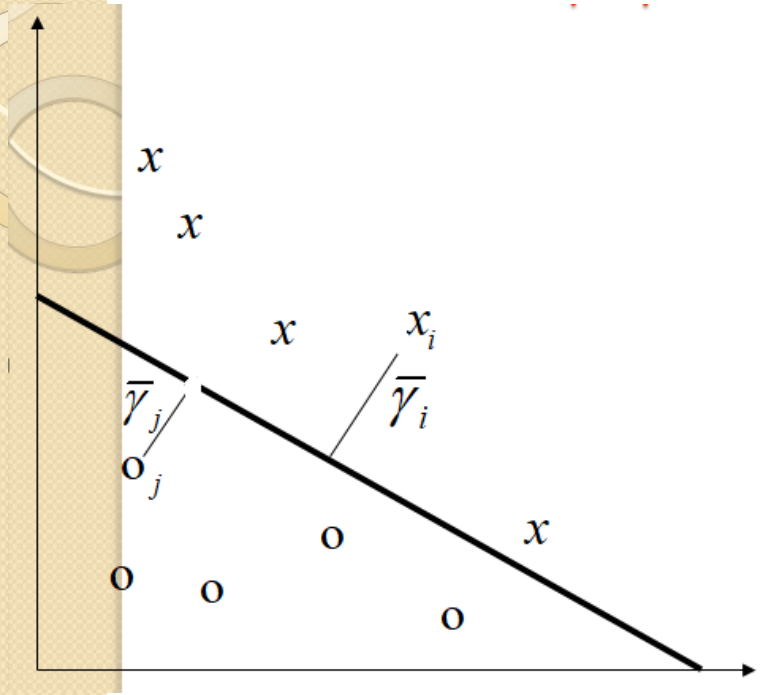
Λειτουργία του Perceptron

- Ο Perceptron αθροίζει τα λάθος ταξινομημένα θετικά σημεία και αφαιρεί τα λάθος ταξινομημένα αρνητικά σημεία από το διάνυσμα βαρών (weight vector).
- Όταν δει μία φορά όλα τα σημεία συνεχίζουμε ξαναπαρουσιάζοντας το σύνολο των σημείων όσες φορές χρειαστεί μέχρι να μη συμβεί κανένα λάθος κατά τη διάρκεια παρουσίασης του συνόλου. Κάθε τέτοια επανάληψη παρουσίασης του συνόλου των δεδομένων καλείται εποχή (epoch). Η σειρά με την οποία βλέπει τα σημεία μπορεί επιλέγεται με τυχαίο τρόπο.
- Το διάνυσμα βαρών γράφεται σαν γραμμικός συνδυασμός των σημείων

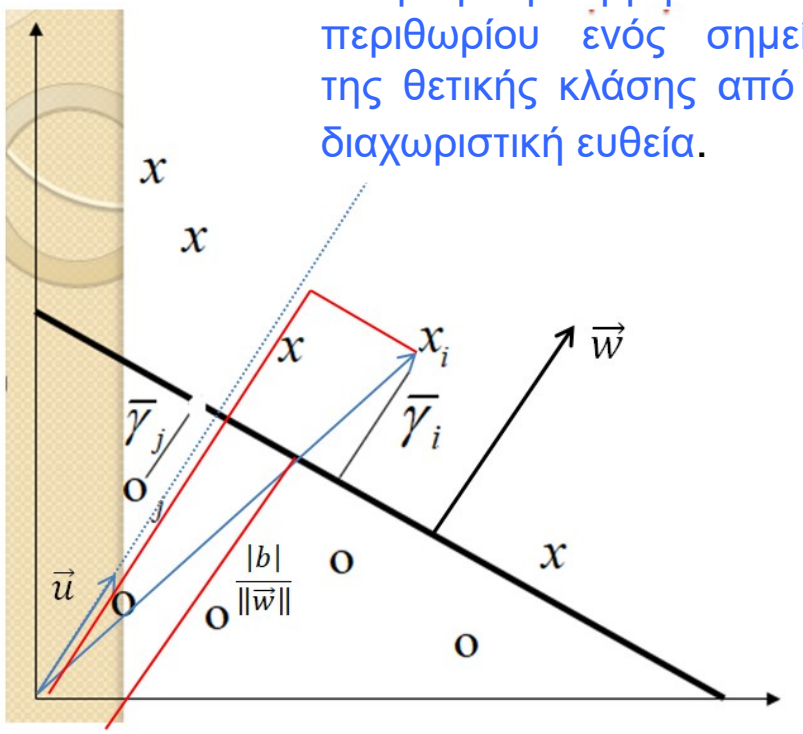
$$\vec{w} = \sum_{i=1}^{l} a_i y_i \vec{x}_i$$

- Το a_i είναι μη αρνητικό και αντιπροσωπεύει τον αριθμό των λανθασμένων ταξινομήσεων του σημείου $\vec{x}_i$.

Γεωμετρικό περιθώριο (margin)



Γεωμετρική ερμηνεία του περιθωρίου ενός σημείου της θετικής κλάσης από τη διαχωριστική ευθεία.



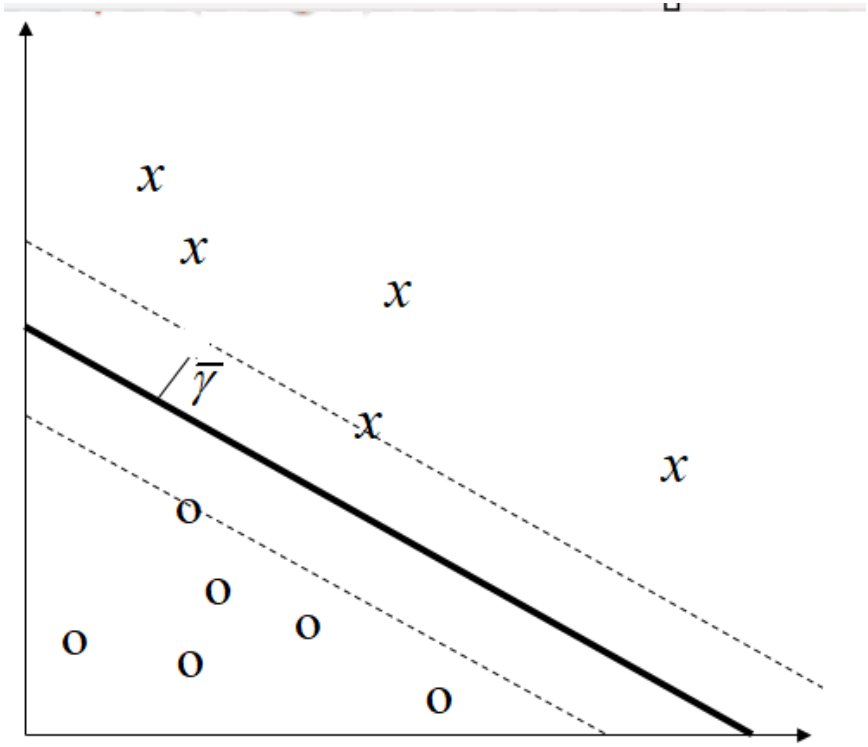
Γεωμετρικό περιθώριο (margin) ενός σημείου

$$\bar{\gamma}_j = y_j \left(\vec{u} \cdot \vec{x}_j + \frac{b}{\|\vec{w}\|} \right),$$

όπου $\vec{u}$ το μοναδιαίο στη διεύθυνση του $\overrightarrow{w}$.

Για τη συγκεκριμένη ευθεία όπως θα εξηγήσουμε στην πορεία $b < 0$. Άρα ο όρος $\frac{|b|}{\|\vec{w}\|}$ αφαιρείται από τον όρο $\vec{u} \cdot \vec{x}_j$.

Γεωμετρικό περιθώριο του συνόλου των σημείων ως προς δεδομένη διαχωριστική ευθεία ή γενικά υπερεπίπεδο

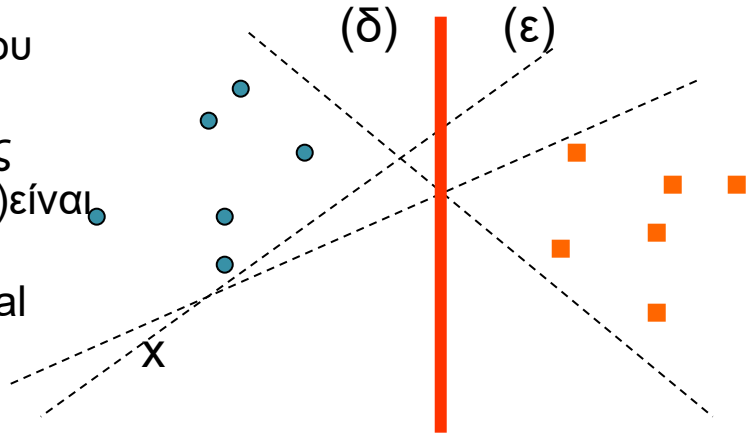


Γεωμετρικό περιθώριο (margin) του συνόλου των σημείων

$$\bar{\gamma} = min_j \bar{\gamma}_j$$

- Το margin του συνόλου των σημείων είναι το μέγιστο γεωμετρικό margin ως προς όλα τα δυνατά υπερεπίπεδα που μπορούν να τα διαχωρίσουν.
- Ένα υπερεπίπεδο που πραγματοποιεί αυτόν το διαχωρισμό ονομάζεται υπερεπίπεδο μέγιστου περιθωρίου

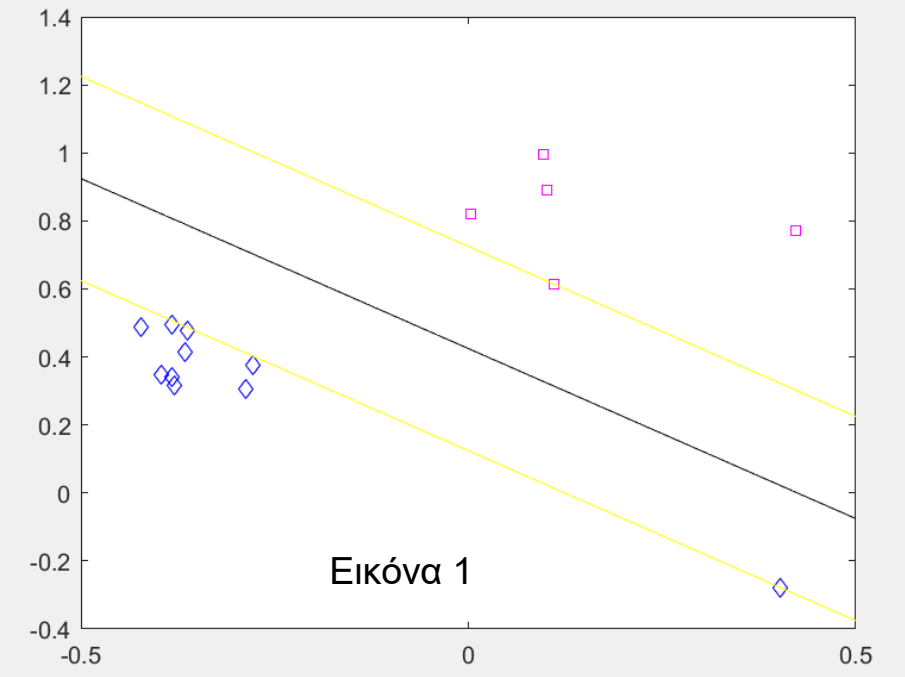
Στόχος των αλγορίθμων που καλούνται Support Vector Machines (SVMs) (μηχανές υποστήριξης διανυσμάτων)είναι η επίτευξη ταξινόμησης με μέγιστο περιθώριο (maximal margin).



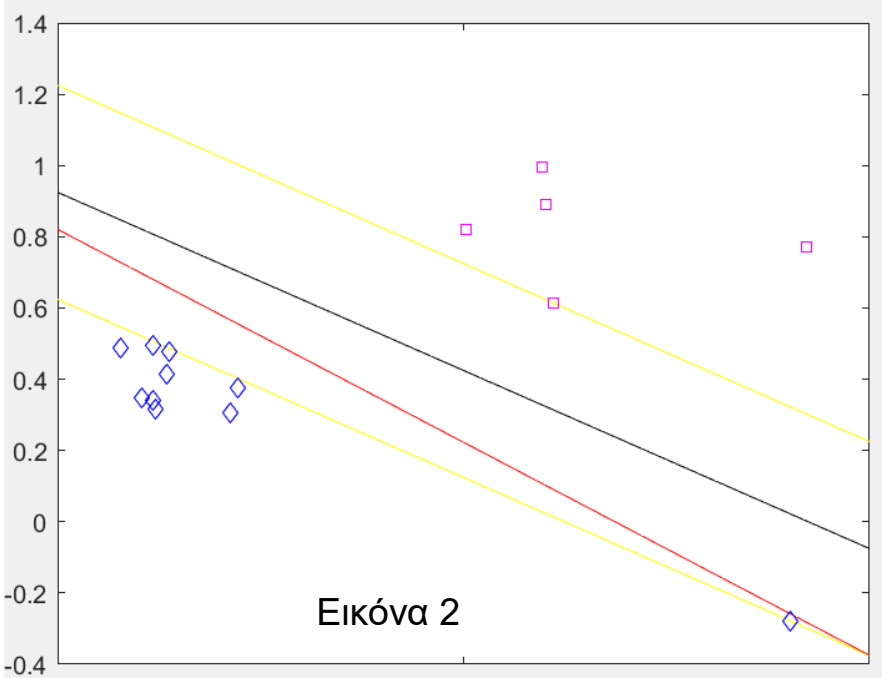
Τα διανύσματα υποστήριξης (support vectors) είναι τα πιο κοντινά σημεία στο υπερεπίπεδο. Ονομάζονται έτσι γιατί υποστηρίζουν τη θέση του υπερεπιπέδου που διαχωρίζει με μέγιστο περιθώριο.

Ένας διαχωρισμός με μέγιστο περιθώριο επιτυγχάνει καλύτερη γενίκευση σε άγνωστα δεδομένα. Για παράδειγμα αν ο διαχωρισμός με την ολοκλήρωση της εκπαίδευσης γίνει με την ευθεία (ε) το σημείο x που ανήκει στην κατηγορία των μπλε θα ταξινομηθεί λανθασμένα ως κόκκινο, παρ' όλο που βρίσκεται πιο κοντά στην κλάση των μπλε σημείων, το οποίο δε θα συνέβαινε αν ο αλγόριθμος κατέληγε στην ευθεία (δ).

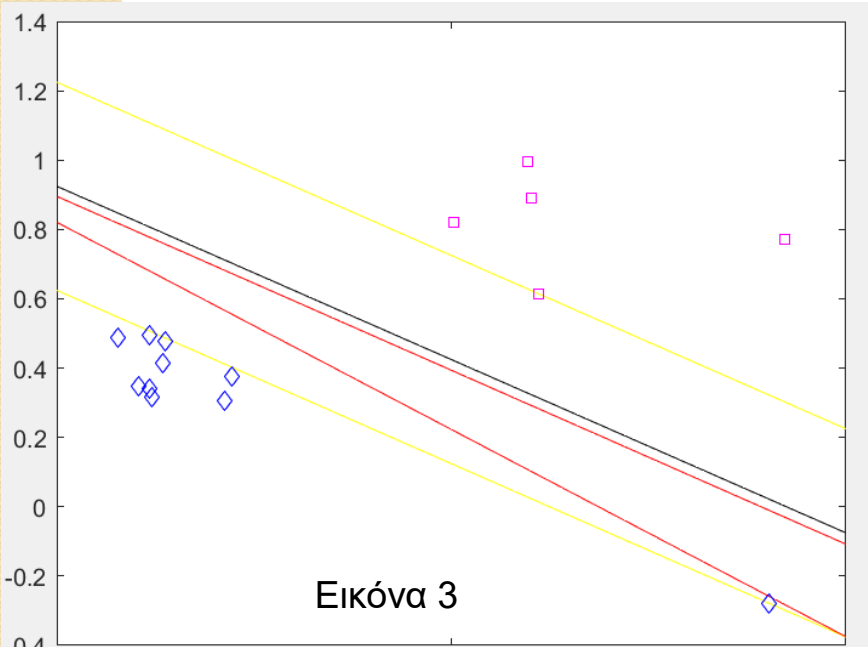
Ο Perceptron σε γραμμικώς διαχωρίσιμα σύνολα



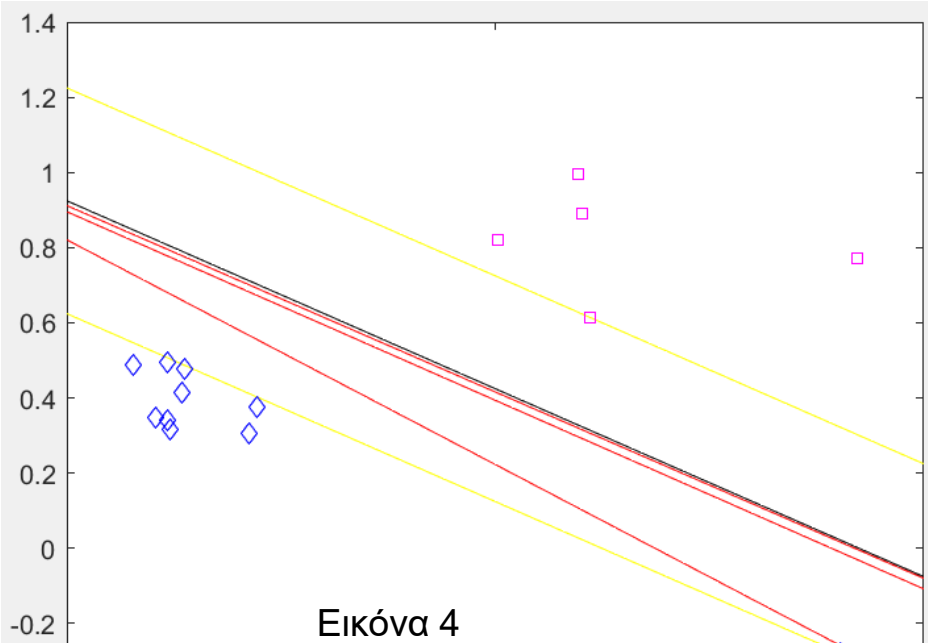
Εικόνα 1



Εικόνα 2

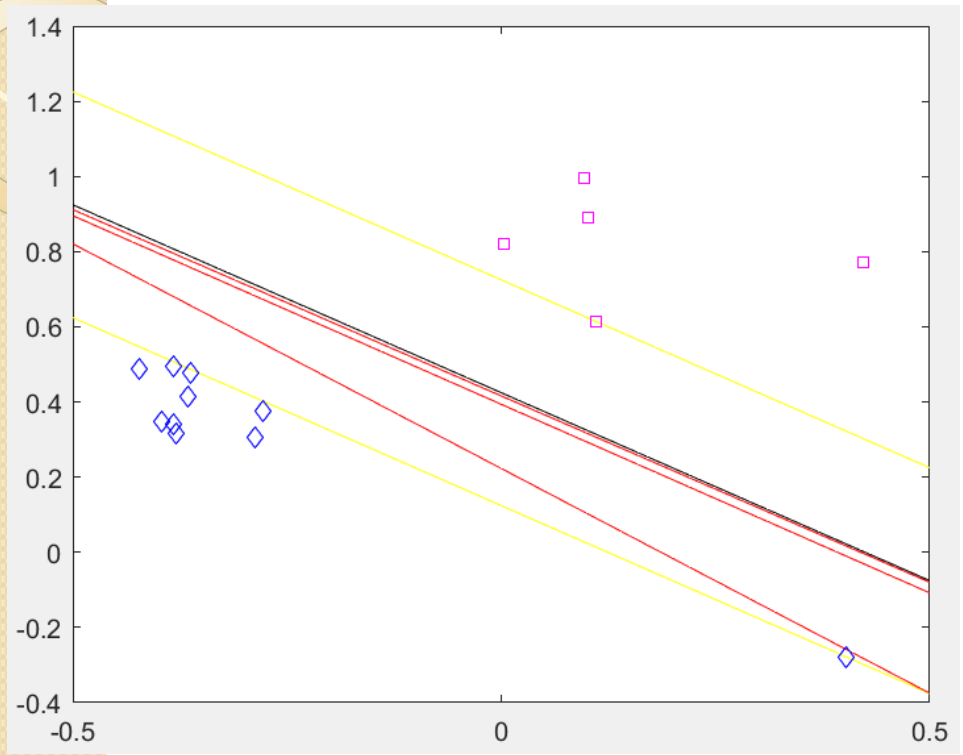


Εικόνα 3



Εικόνα 4

Πώς μπορεί ο Perceptron να επιτύχει μεγαλύτερο περιθώριο;



Ο Perceptron στην απλή του μορφή δεν μπορεί να εξασφαλίσει διαχωρισμό με περιθώριο, αλλά απλώς διαχωρισμό.

Στις Εικόνες 2,3,4 ο Perceptron διαχωρίζει τα σημεία των 2 κατηγοριών με όλο και μεγαλύτερο περιθώριο.

Για να επιτύχουμε τα αποτελέσματα αυτά μεταβάλλαμε την κλασική συνθήκη με την οποία ένα σημείο x_i θεωρείται λάθος κατηγοριοποιημένο

$$y_i((\vec{w}_k \cdot \vec{x}_i) + b_k) \leq 0.$$

σε συνθήκη με την οποία επιβάλλουμε μεγαλύτερο περιθώριο

$$y_i((\vec{w}_k \cdot \vec{x}_i) + b_k) \leq B.$$

Το B είναι μία παράμετρος την οποία θέτουμε σε σταθερή τιμή. Αποδεικνύεται ότι όταν $B \to +\infty$ ο αλγόριθμος καταλήγει σε υπερεπίπεδο που διαχωρίζει τα σημεία με περιθώριο τουλάχιστον το μισό του μέγιστου περιθωρίου. Στην πράξη διαχωρίζει με πολύ περισσότερο θέτοντας το B σε τιμές που είναι αρκετά πολλαπλάσια του τετραγώνου της μέγιστης απόστασης των σημείων από την αρχή των αξόνων.

Δυική αναπαράσταση (Dual Representation)

- Η $f(\vec{x})$ ξαναγράφεται σαν

$$f(\vec{x}) = \vec{w} \cdot \vec{x} + b = \sum_{j=1}^{l} \alpha_j y_j (\vec{x}_j \cdot \vec{x}) + b$$

- Η συνθήκη (condition) και ο κανόνας ανανέωσης (update rule) του διανύσματος βαρών ξαναγράφονται σαν

$$\text{if } \; y_i \left(\sum_{j=1}^{l} \alpha_j y_j (\vec{x}_j \cdot \vec{x}_i) + b \right) \leq 0 \quad \text{then} \quad a_i \leftarrow a_i + 1$$

με το $\vec{x}_i$ να είναι το σημείο που ελέγχεται.

Απεικόνιση στο χώρο των χαρακτηριστικών (feature space)

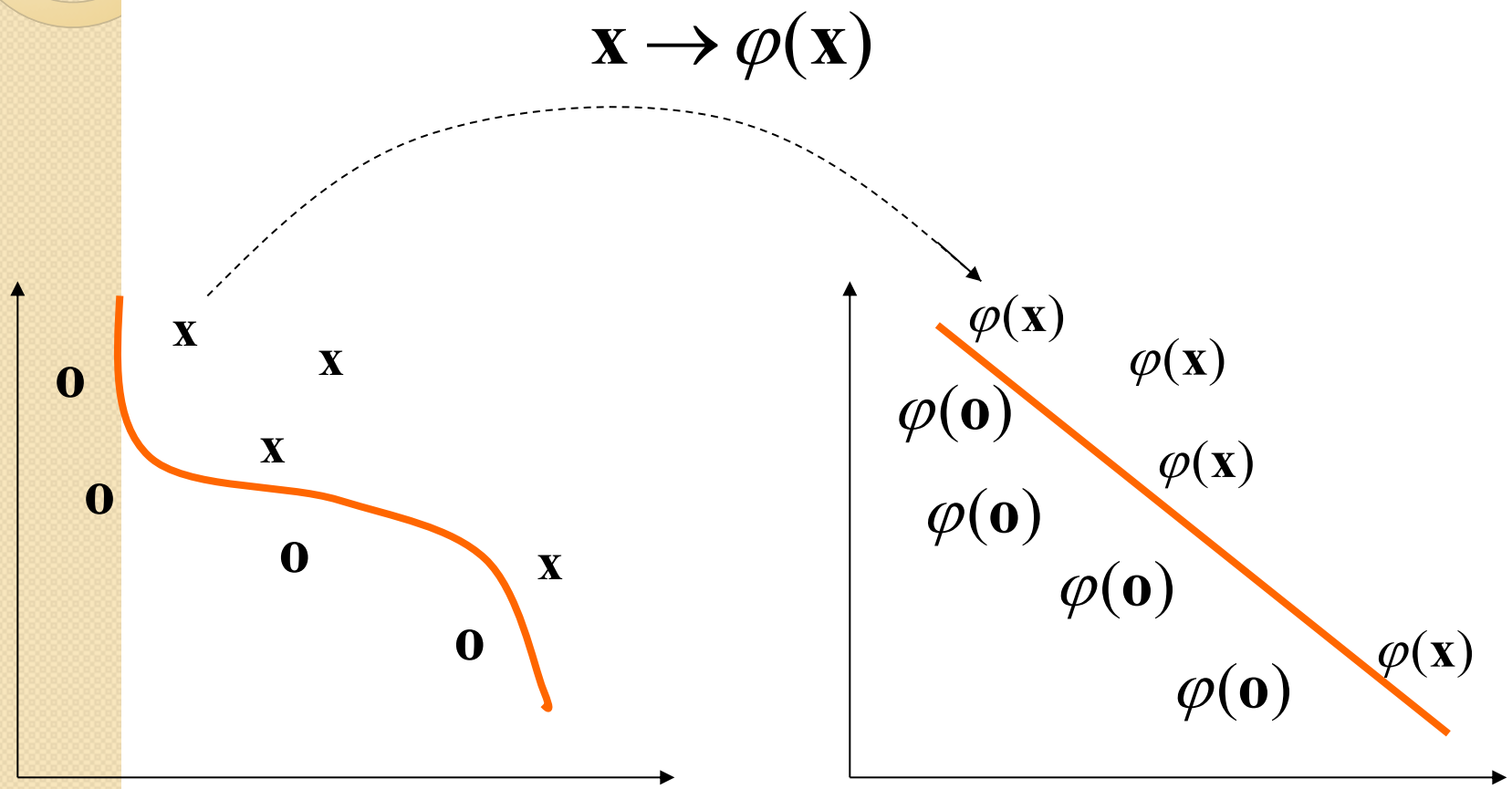
- Επιτρέπει την εξάλειψη συνιστωσών μη σχετικών με το πρόβλημα της ταξινομήσης των σημείων (irrelevant attributes)

- Ενσωματώνει τη γνώση μας σχετικά με το πρόβλημα που αναπαρίσταται με τον κατάλληλο αριθμό χαρακτηριστικών

- Ενδεχομένως επιτρέπει τον **γραμμικό διαχωρισμό (attributes → features)**

Απεικόνιση στο χώρο των χαρακτηριστικών (feature space)

Ο χώρος X ονομάζεται χώρος των εισόδων ενώ ο $F = \{\varphi(\vec{x}) : \vec{x} \in X\}$ χώρος των χαρακτηριστικών



Παράδειγμα:

Έστω ότι η πληροφορία για ένα πρόβλημα περιλαμβάνεται σε μονώνυμα 2ου βαθμού που σχηματίζονται από συνιστώσες του $\vec{x}$.

Η γνώση μας κωδικοποιείται με την απεικόνιση

$$\vec{x} = (x_1, x_2) \rightarrow \varphi(x_1, x_2) = ({x_1}^2, {x_2}^2, x_1 x_2)$$

Αν η διάσταση του αρχικού χώρου είναι n και μεταβαίνουμε σε χώρους μονωνύμων διαστάσεων d η διάσταση N του χώρου των χαρακτηριστικών είναι

$$N = \binom{n+d-1}{d} = \frac{(n+d-1)!}{d!(n-1)!}$$

Για μεγάλο n και d οδηγούμαστε σε έκρηξη των διαστάσεων.

- Θα εισάγουμε Kernels για να αντιμετωπίσουμε το πρόβλημα των πολλών διαστάσεων.
- Μπορούμε να κάνουμε απεικονίσεις σε χώρους απείρων διαστάσεων.

Για να μάθουμε μη γραμμικές σχέσεις χρησιμοποιούμε απεικονίσεις σε μη γραμμικά χαρακτηριστικά. Το διάνυσμα βαρών μετά από απεικόνιση στον χώρο των χαρακτηριστικών γράφεται ως

$$\vec{w} = \sum_{i=1}^{l} \alpha_i y_i \varphi(\vec{x}_i)$$

Στη δυική αναπαράσταση τα απεικονισμένα σημεία συμμετέχουν εντός της συνάρτησης απόφασης σε εσωτερικά γινόμενα της μορφής

$$\sum_{i=1}^{l} a_i y_i \left(\varphi(\vec{x}_i) \cdot \varphi(\vec{x})\right) + b$$

Kernels

Το kernel είναι μια συνάρτηση που δέχεται σαν ορίσματα δύο σημεία $\vec{x}$ και $\vec{z}$ και επιστρέφει έναν πραγματικό αριθμό που μετρά την ομοιότητα των εικόνων των σημείων στο χώρο των χαρακτηριστικών

$$K(\vec{x}, \vec{z}) = \varphi(\vec{x}) \cdot \varphi(\vec{z})$$

Kernel matrix

K(1,1)	K(1,2)	K(1,3)	…	K(1,m)
K(2,1)	K(2,2)	K(2,3)	…	K(2,m)
…	…	…	…	…
K(m,1)	K(m,2)	K(m,3)	…	K(m,m)

Παραδείγματα kernels

- **Πολυωνυμικό βαθμού d**

$$K(\vec{x}, \vec{z}) = \left((\vec{x} \cdot \vec{z}) + 1\right)^d$$

- **Gaussian RBF**

$$K(\vec{x}, \vec{z}) = \boldsymbol{e}^{-\|\vec{x}-\vec{z}\|^2/2\boldsymbol{\sigma}^2}$$

Έμμεση απεικόνιση στο χώρο των χαρακτηριστικών (Implicit feature mapping)

Χρησιμοποιώντας τη δυική αναπαράσταση και την αντικατάσταση του εσωτερικού γινομένου με Kernel ο Perceptron επιτυγχάνει μη γραμμικούς διαχωρισμούς στον αρχικό χώρο ως εξής:

$$\text{if} \quad y_i\left(\sum_{j=1}^{l} a_j y_j K\left(\vec{x}_j, \vec{x}_i\right) + b\right) \leq 0 \quad \text{then} \quad a_i \leftarrow a_i + 1$$

Ο αλγόριθμος Perceptron στη δυική αναπαράσταση

Αρχικός χώρος

initialize $\mathbf{w}_0 \leftarrow \mathbf{0}, b_0 \leftarrow 0, k \leftarrow 0$

repeat

error ← false

for i=1..l

if $y_i(\boldsymbol{w}_k \cdot \boldsymbol{x}_k + b_k) \le 0$ **then**

$\mathbf{w}_{k+1} \leftarrow \mathbf{w}_k + y_i \mathbf{x}_i$

$b_{k+1} \leftarrow b_k + y_i$

$k \leftarrow k+1$

error ← true

end if

end for

until (error==false)

return k, $(\mathbf{w}_k, b_k)$ όπου k είναι το πλήθος των λαθών

Ο 2ος αλγόριθμος είναι γραμμένος σε προγραμματιστική γραφή για να μην εισάγουμε 2ο δείκτη στα α_i.Το αντίστοιχο στον 1ο αλγόριθμο θα ήταν να γράψουμε $\boldsymbol{w} \leftarrow \boldsymbol{w} + y_i \boldsymbol{x_i}$.

Δυικός χώρος

initialize $\alpha_i \leftarrow 0, i = 1..l, b_o \leftarrow 0, k \leftarrow 0$

repeat

error← $false$

for i=1..l

if $y_i\left(\sum_{j=1}^{l} a_j y_j K(\vec{x}_j, \vec{x}_i) + b\right) \le 0$ **then**

$a_i \leftarrow a_i + 1$

$b \leftarrow b + y_i$

$k \leftarrow k + 1$

error← $true$

end if

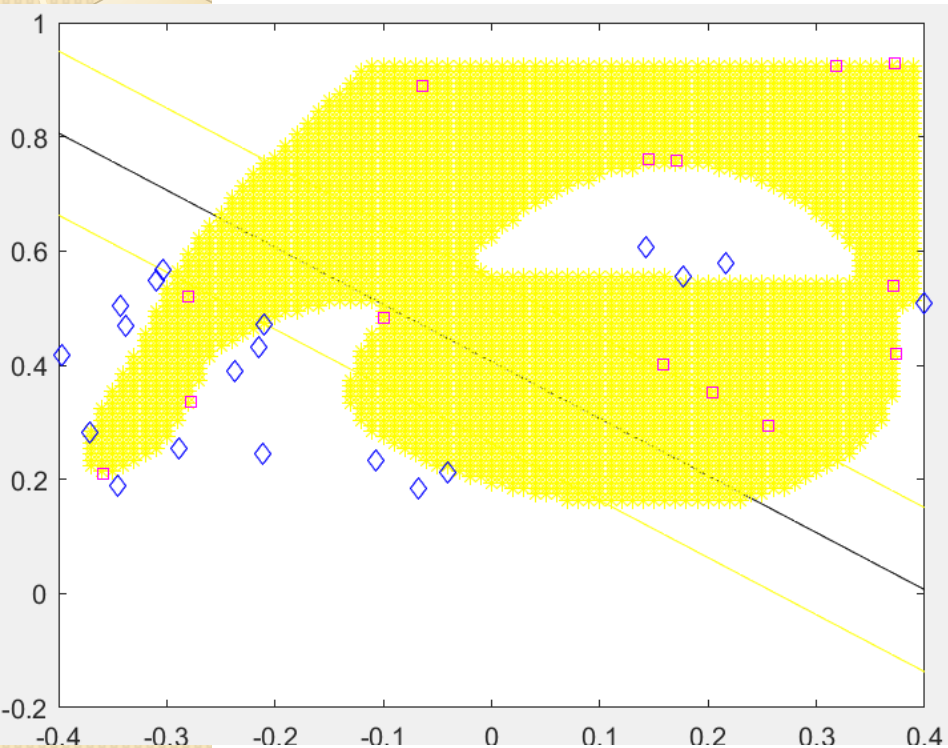
end for

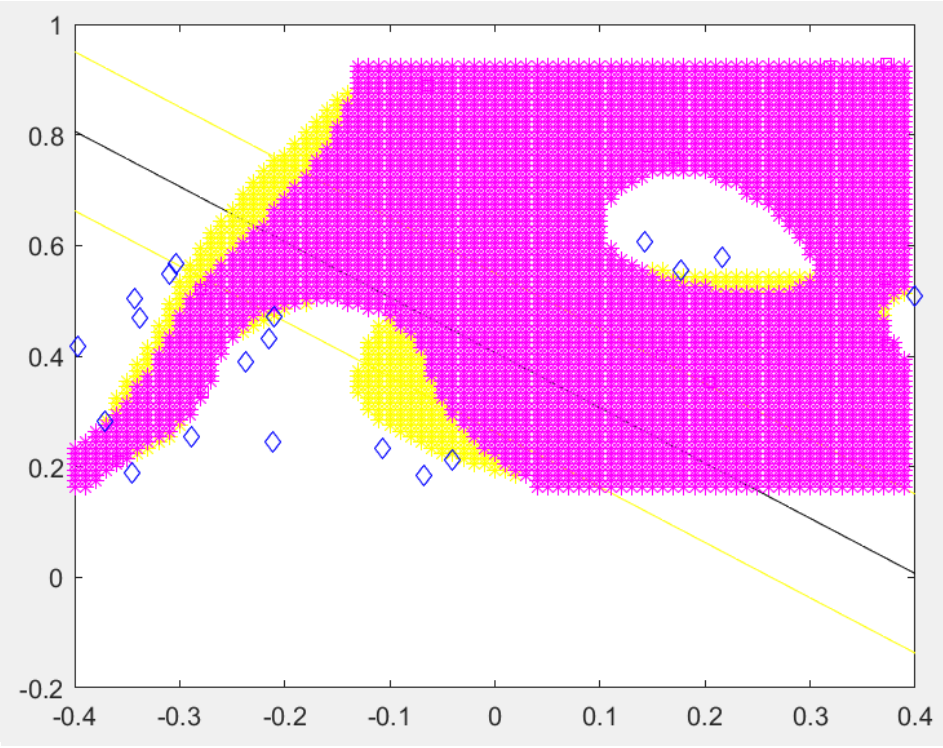
until (error==false)

return k, $a_i, i = 1..l, b$

Παρατηρούμε ότι δεν εμφανίζεται πουθενά μόνο του το μετασχηματισμένο σημείο $\varphi(x_i)$ που ελέγχεται από τη συνθήκη αλλά εντός του εσωτερικού γινομένου το οποίο αντικαθίσταται με το Kernel. Στη θέση του διανύσματος βάρους εμφανίζονται τα α_ι τα οποία είναι όσα και τα σημεία και μετράν το πλήθος των λάθος ταξινομήσεων.

Εφαρμογή του Perceptron σε μη γραμμικά διαχωρίσιμα σύνολα με χρήση kernel





Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το Gaussian rbf με τυπική απόκλιση $\sigma = 0.5$

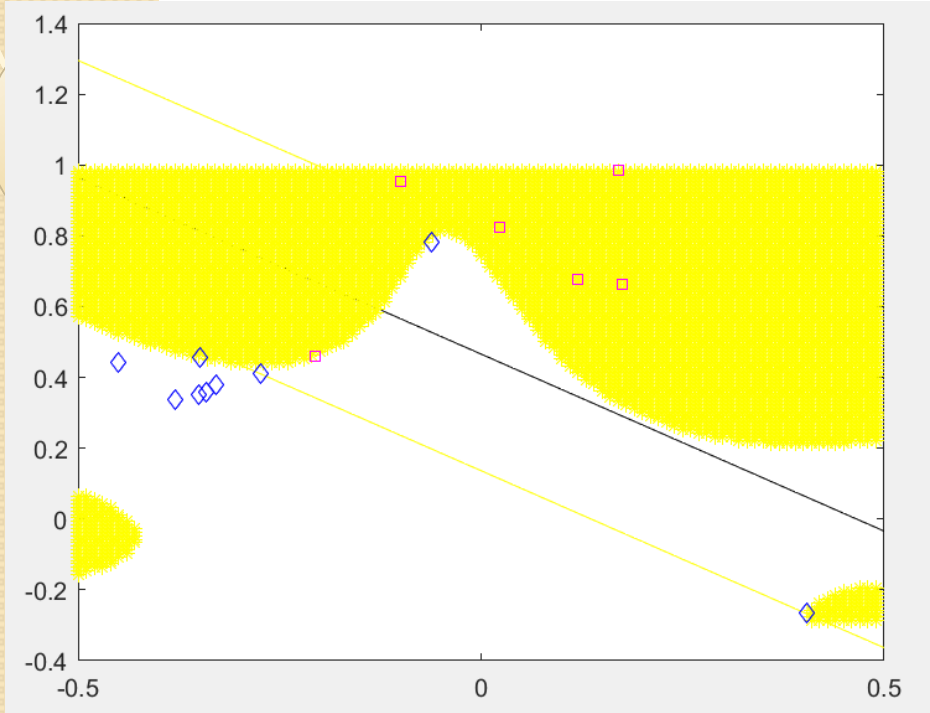
Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το Gaussian rbf με τυπική απόκλιση $\sigma = 0.2$

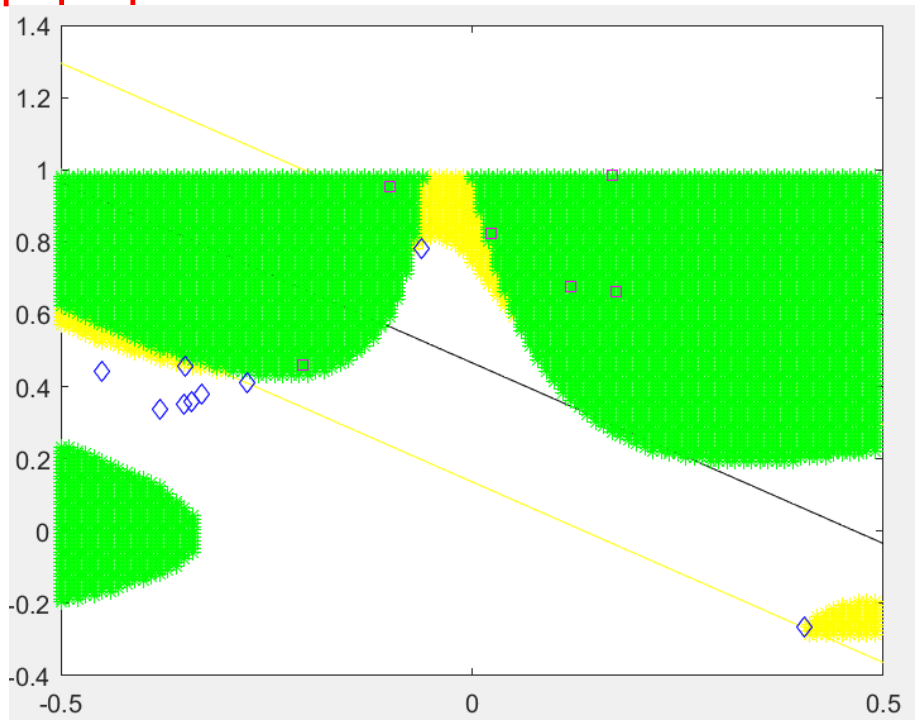
Μη γραμμικώς διαχωρίσιμα σύνολα δεδομένων

- Ο αλγόριθμός Perceptron συγκλίνει, δηλαδή οδηγείται σε αποτέλεσμα που είναι το κάθετο διάνυσμα στο υπερεπίπεδο που διαχωρίζει τα δεδομένα, σε πεπερασμένο αριθμό επαναλήψεων μόνο όταν τα δεδομένα είναι γραμμικώς διαχωρίσιμα, είτε στον αρχικό χώρο των εισόδων X είτε στο χώρο των χαρακτηριστικών $F = \{\varphi(\vec{x}) : \vec{x} \in X\}$ όπως για παράδειγμα μέσω της απεικόνισης $\boldsymbol{x} = (x_1, x_2) \rightarrow \boldsymbol{\varphi}(x_1, x_2) = (x_1^2, x_2^2, x_1 x_2)$

- Δεν μπορεί όμως να μας εξασφαλίσει κανείς ότι τα δεδομένα θα έχουν γίνει διαχωρίσιμα στο χώρο των χαρακτηριστικών παρ' όλο που μεταβήκαμε σε ένα χώρο μεγαλύτερων διαστάσεων.

Εφαρμογή του Perceptron σε μη γραμμικά διαχωρίσιμα σύνολα με χρήση kernel





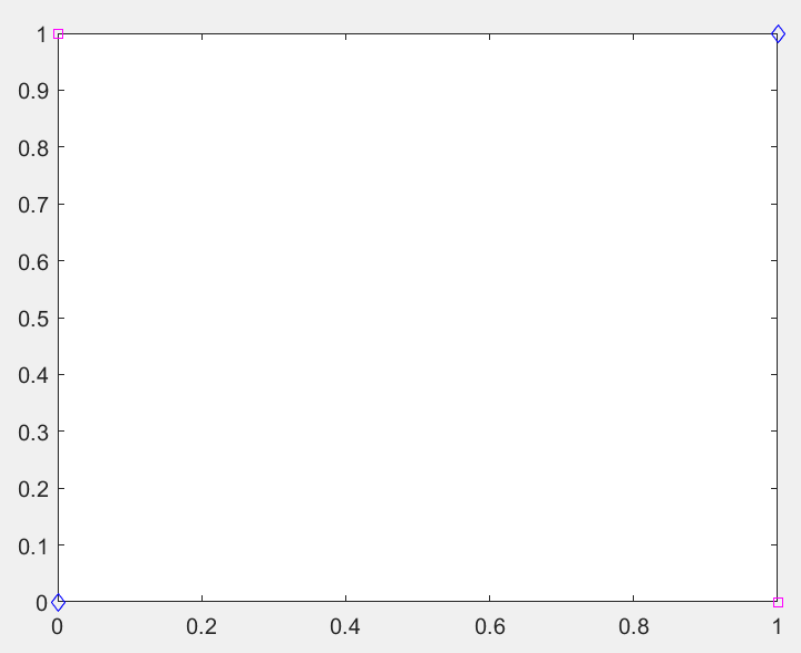
Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το πολυωνυμικό 5ου βαθμού με ελάχιστο περιθώριο $7{,}05 \cdot 10^{-6}$. Όλα τα πολυώνυμα με μικρότερο βαθμό δεν μπόρεσαν να διαχωρίσουν τα δεδομένα.
Ο Perceptron έτρεξε με παράμετρο $\mathrm{B} = 0$

Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το πολυωνυμικό 5ου βαθμού με ελάχιστο περιθώριο $6{,}2 \cdot 10^{-4}$. Όλα τα πολυώνυμα με μικρότερο βαθμό δεν μπόρεσαν να διαχωρίσουν τα δεδομένα.
Ο Perceptron έτρεξε με παράμετρο $\mathrm{B} = 2R^2$, όπου $R = max_i \|x_i\|$

Το πρόβλημα της πύλης XOR

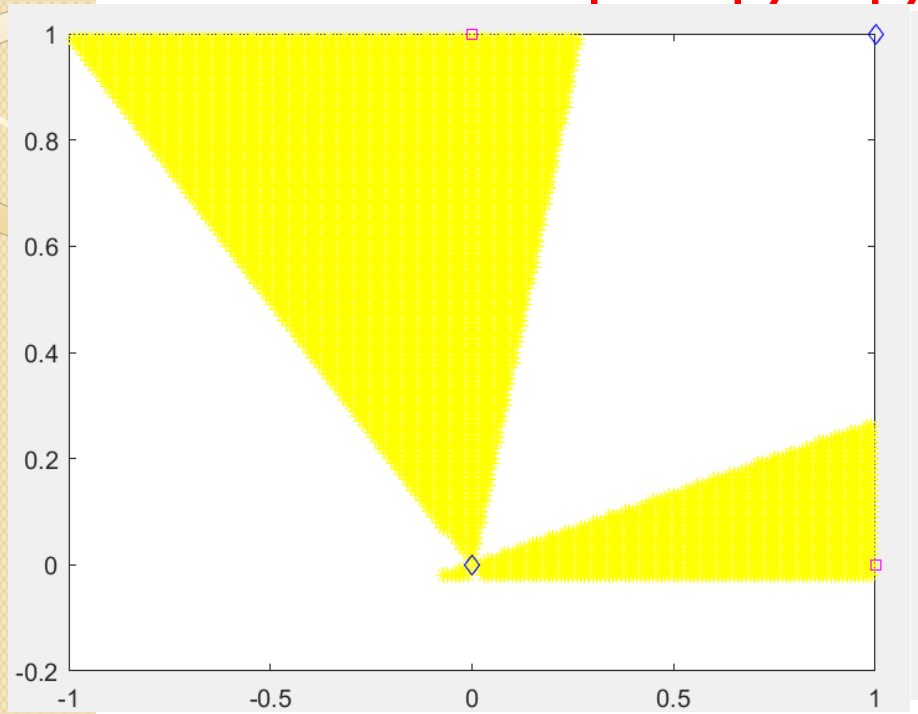
Ο Perceptron συναντά πρόβλημα στην εύρεση υπερεπιπέδων στην περίπτωση που έχουμε μη γραμμικά διαχωρίσιμα σύνολα, όπως στην περίπτωση των δεδομένων που προκύπτουν από τη λειτουργία της πύλης XOR που περιγράφεται από τον παρακάτω πίνακα αλήθειας

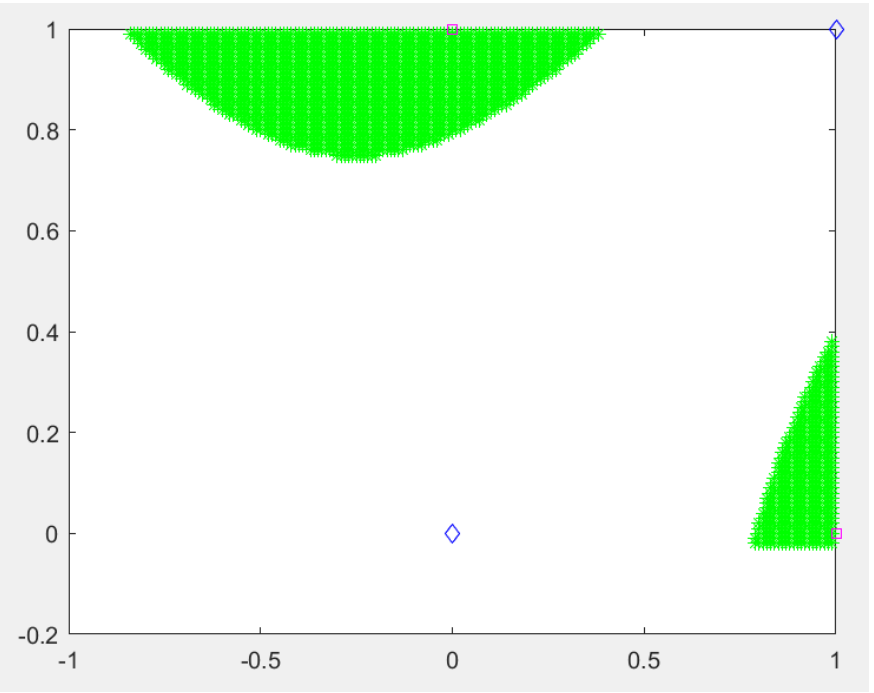
x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0



Γραφική αναπαράσταση των αποκρίσεων της πύλης XOR στα σημεία (0,0), (0,1), (1,0) και (1,1)

Εφαρμογή του Perceptron στα δεδομένα της απόκρισης της πύλης XOR

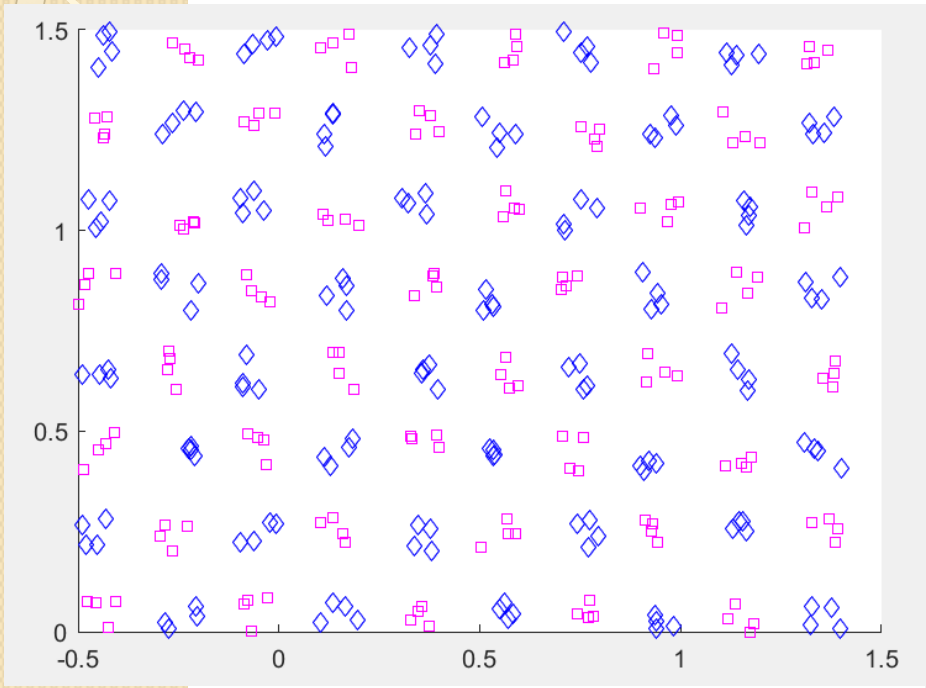


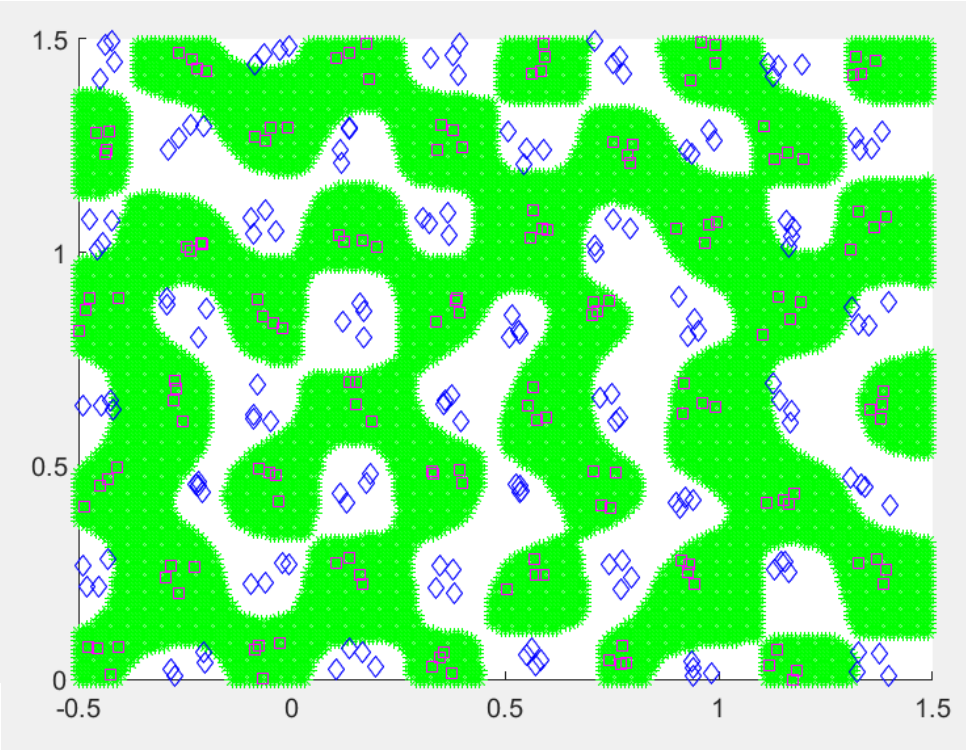


Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το πολυωνυμικό 3ου βαθμού. Ελάχιστο περιθώριο $4{,}9 \cdot 10^{-18}$. Όλα τα πολυώνυμα με μικρότερο βαθμό δεν μπόρεσαν να διαχωρίσουν τα δεδομένα.
Ο Perceptron έτρεξε με παράμετρο $\mathrm{B} = 0$

Το kernel που χρησιμοποιήθηκε για τον διαχωρισμό είναι το πολυωνυμικό 3ου βαθμού. Ελάχιστο περιθώριο $0{,}3033$. Όλα τα πολυώνυμα με μικρότερο βαθμό δεν μπόρεσαν να διαχωρίσουν τα δεδομένα.
Ο Perceptron έτρεξε με παράμετρο $\mathrm{B} = 4R^2$

Εφαρμογή του Perceptron σε δεδομένα τύπου σκακιέρας

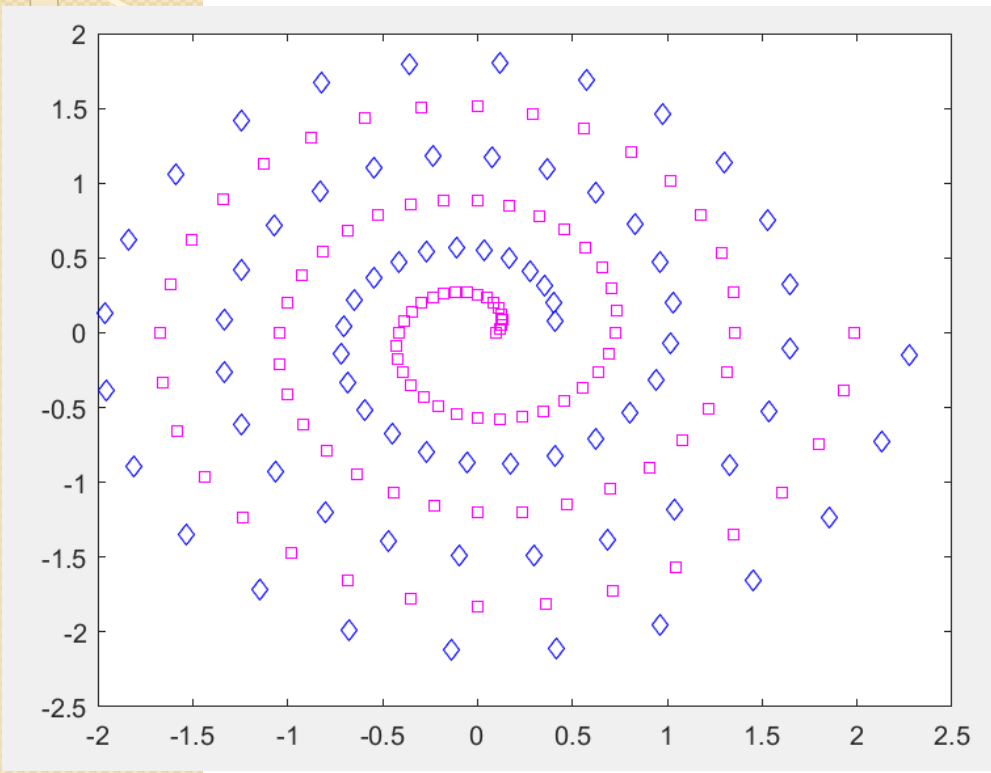


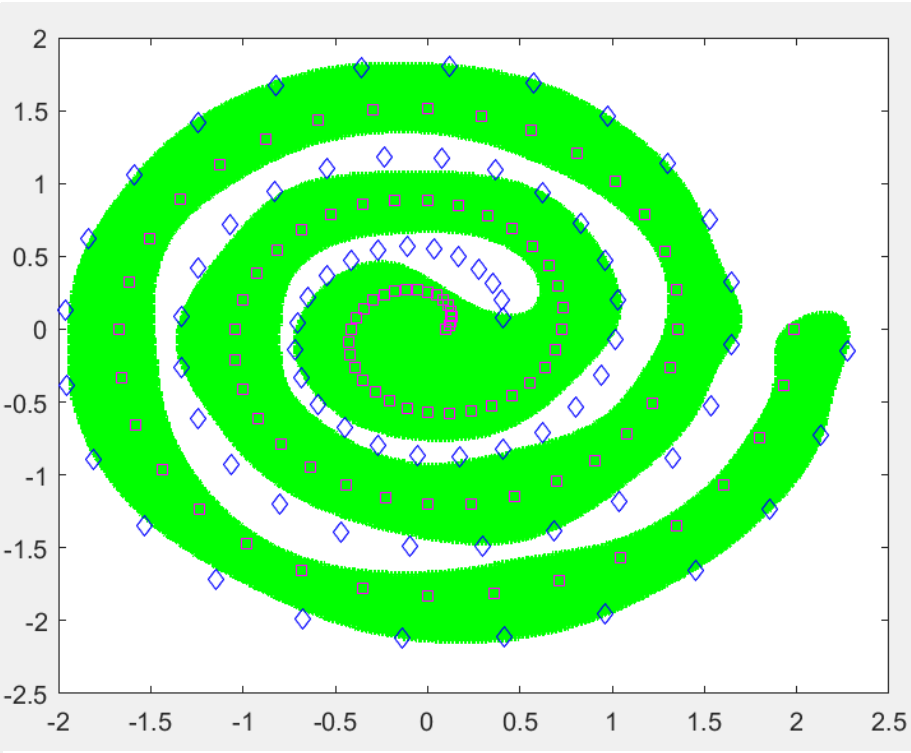


Εκτέλεση αλγορίθμου με Gaussian kernel με τυπική απόκλιση $\sigma = 0{,}2$.

Ο αλγόριθμος πέτυχε ελάχιστο περιθώριο στο χώρο των χαρακτηριστικών $5{,}99 \cdot 10^{-4}$.

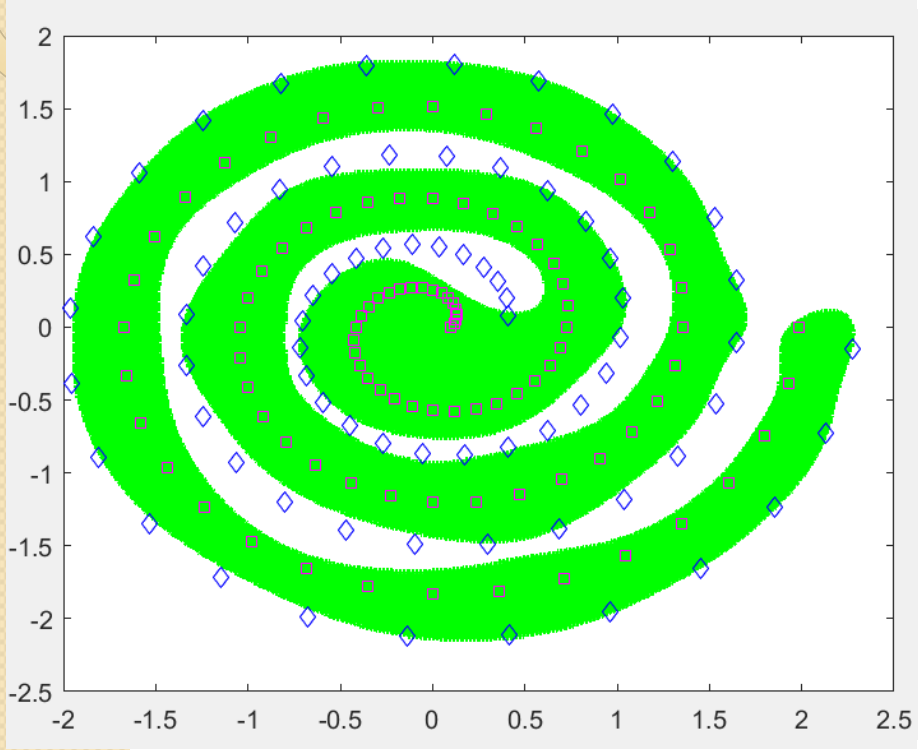
Εφαρμογή του Perceptron σε δεδομένα τύπου σπείρας

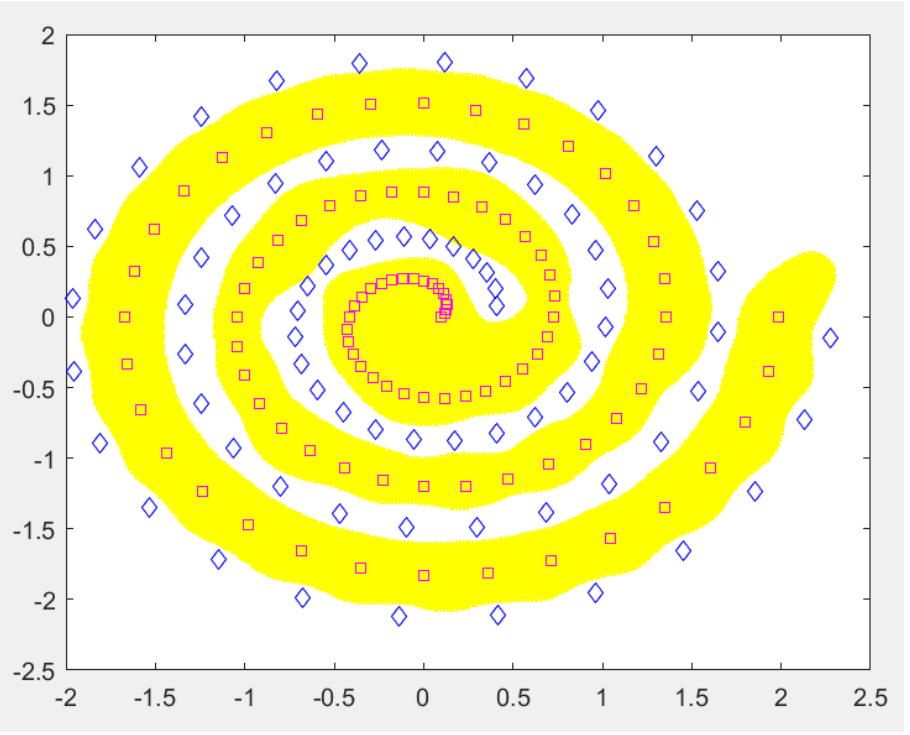




Εκτέλεση αλγορίθμου με Gaussian kernel με τυπική απόκλιση $\sigma = 0{,}5$.
Ο αλγόριθμος πέτυχε ελάχιστο περιθώριο στο χώρο των χαρακτηριστικών $1{,}53 \cdot 10^{-4}$.

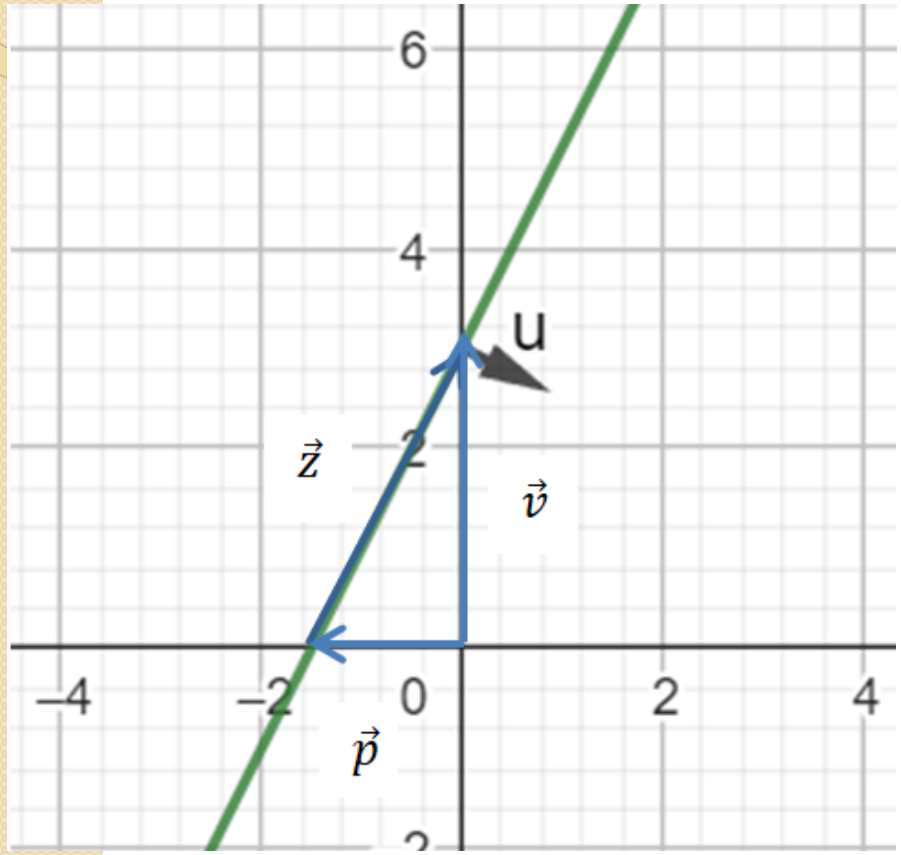
Εφαρμογή του Perceptron σε δεδομένα τύπου σπείρας





Εκτέλεση αλγορίθμου με Gaussian kernel με τυπική απόκλιση $\sigma = 0{,}2$.
Ο αλγόριθμος επέτυχε ελάχιστο περιθώριο στο χώρο των χαρακτηριστικών $0{,}0141$.

Ορισμός της ευθείας με τη βοήθεια του κάθετου διανύσματος



$$y = ax + b_1$$

$$\vec{z} = \vec{v} - \vec{p}$$

Έστω η ευθεία με εξίσωση

$$y = 2x + 3$$

Για $x = 0$ $y = 3$ και για $y = 0$ $x = -\frac{3}{2}$

Επομένως $\vec{v} = (0,3)$ και $\vec{p} = (-\frac{3}{2}, 0)$

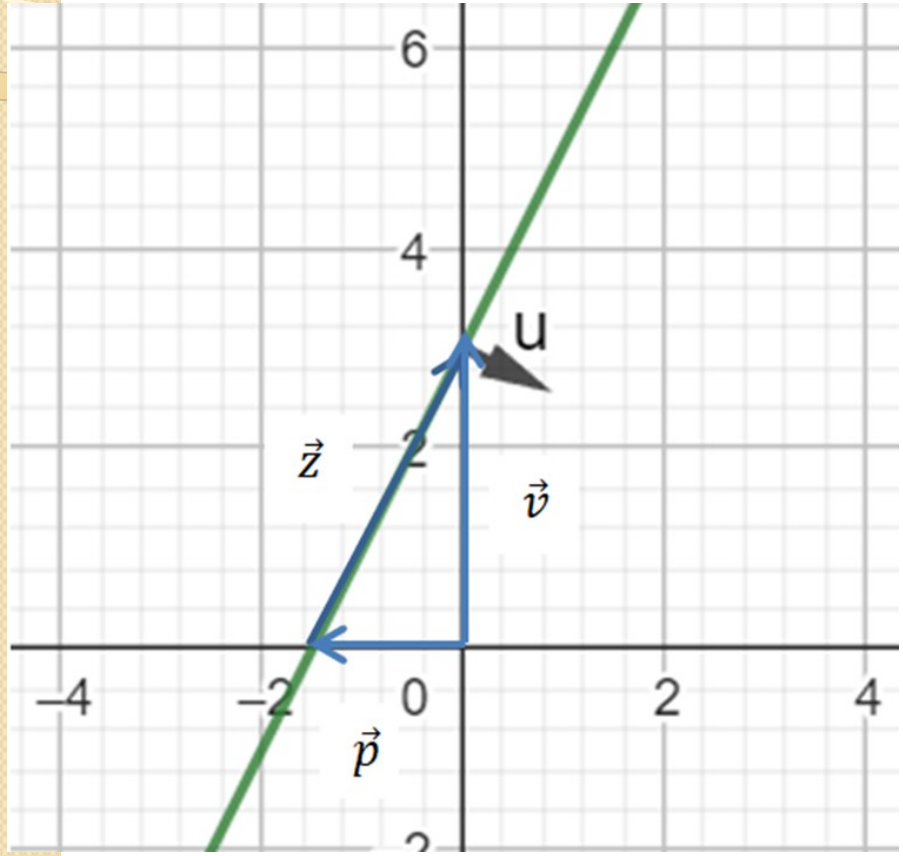
$$\vec{v} - \vec{p} = (\frac{3}{2}, 3)$$

Θέλουμε να φέρουμε την εξίσωση στη μορφή

$$\vec{w} \cdot \vec{x} + b = 0,$$

όπου $\vec{w}$ είναι το κάθετο διάνυσμα στην ευθεία.

Προσδιορισμός του κάθετου διανύσματος και του bias b



Επειδή $\vec{w}$ κάθετο διάνυσμα στην ευθεία ισχύει

$$\vec{w} \cdot (\vec{v} - \vec{p}) = 0 \Rightarrow$$

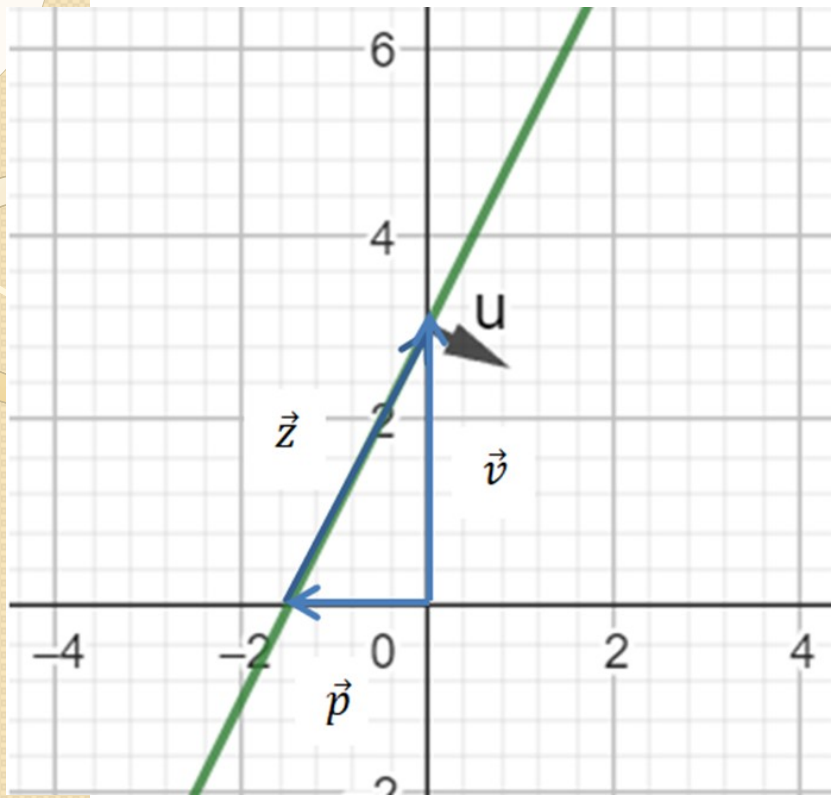
$$(w_1, w_2) \cdot \left(\frac{3}{2}, 3\right) = 0 \Rightarrow$$

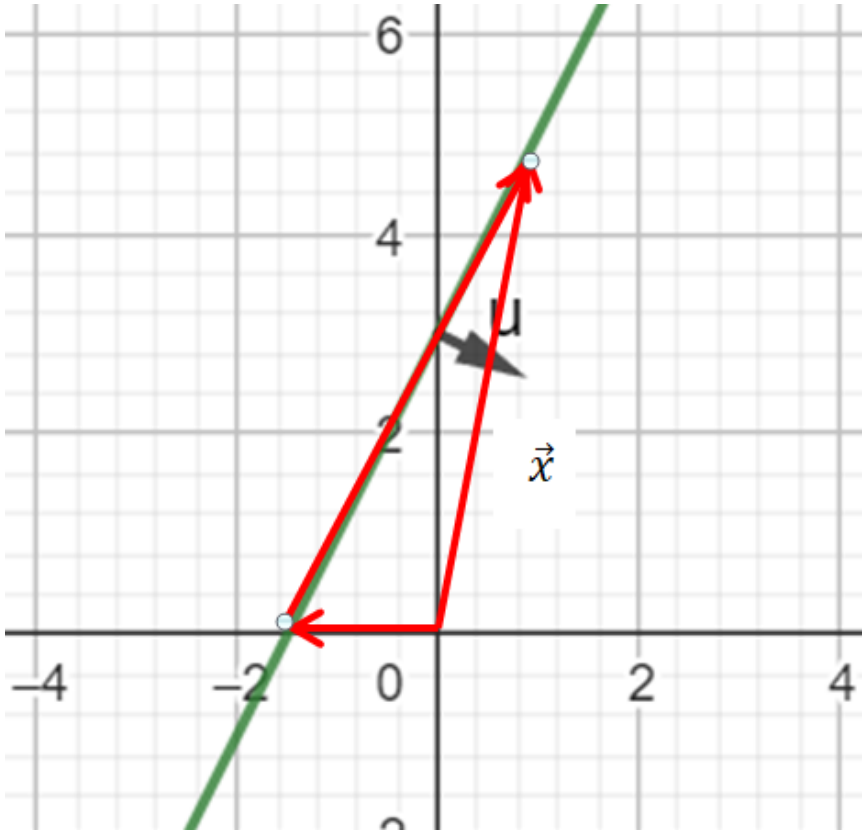
$$\vec{w} = (-3, \frac{3}{2})$$

Από τη στιγμή που είναι γνωστό το $\vec{w}$ το b θα προσδιοριστεί έτσι ώστε

$\vec{w} \cdot \vec{x} + b = 0$, όπου το διάνυσμα $\vec{x}$ ανήκει στην ευθεία.

Στην εικόνα απεικονίζεται το μοναδιαίο κάθετο διάνυσμα στην ευθεία γραμμή το οποίο όμως έχει αντίθετη κατεύθυνση σε σχέση με το $\vec{w}$.
Το μοναδιαίο μπορεί να έχει ίδια κατεύθυνση με το εικονιζόμενο $\vec{u}$ είτε την αντίθετη του.





Κάθε σημείο $\vec{x}$ της ευθείας $y = ax + b_1$ γράφεται σαν

$$\vec{x} = \left(-\frac{b_1}{a}, 0\right) + \lambda\vec{z},$$

όπου $\lambda \in \mathbb{R}$ και $\vec{z}$ διάνυσμα παράλληλο στην ευθεία.

Θα πρέπει $\vec{w} \cdot \vec{x} + b = 0$

Οπότε $\left(-3, \frac{3}{2}\right) \cdot \left(-\frac{b_1}{a}, 0\right) + \lambda\vec{w} \cdot \vec{z} + b = 0 \Rightarrow -\frac{3}{2}(-3) + 0 + b = 0 \Rightarrow$

$$b = -\frac{9}{2}$$

Η εξίσωση $y = ax + b_1$ μπορεί γενικά να γραφεί στη μορφή

$$\mathrm{Ax} + \mathrm{By} + \Gamma = 0 \ (1)$$

Για την $y = 2x + 3$ βρήκαμε ότι $\vec{w} = \left(-3, \frac{3}{2}\right)$ $\kappa\alpha\iota$ $b = -\frac{9}{2}$

Μπορεί να γραφεί επίσης ως

$$2x - y + 3 = 0, \text{ όπου } \mathrm{A} = 2, \mathrm{B} = -1, \Gamma = 3$$

Το κάθετο διάνυσμα σε μία ευθεία της μορφής (1) δίνεται γενικά από τη σχέση

$$\vec{w} = \left(\frac{\Gamma}{\mathrm{B}}, \frac{\Gamma}{\mathrm{A}}\right)$$

και το bias από τη σχέση $b = \frac{\Gamma^2}{\mathrm{AB}}$

Επειδή για τα σημεία που ανήκουν στην ευθεία ισχύει

$$\vec{w} \cdot \vec{x} + b = 0 \Rightarrow \left(\frac{\Gamma}{\mathrm{B}}, \frac{\Gamma}{\mathrm{A}}\right) \cdot \vec{x} + \frac{\Gamma^2}{\mathrm{AB}} = 0,$$

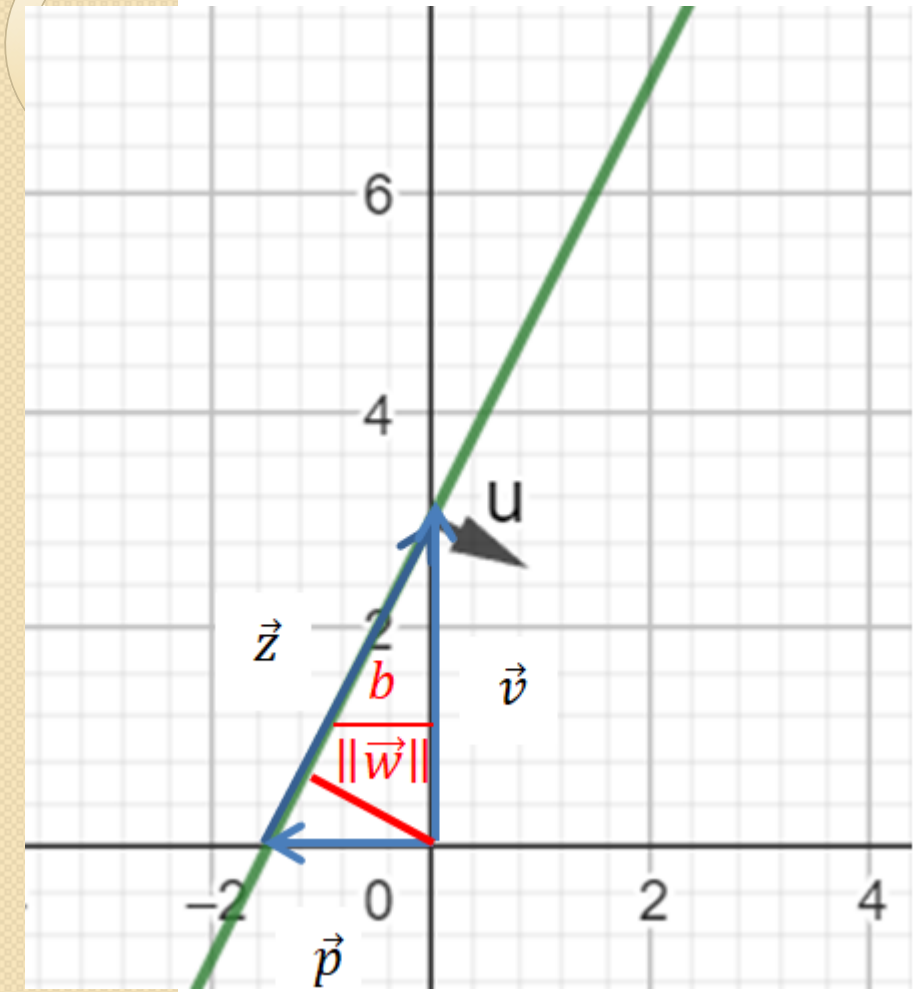
μπορούμε να πολλαπλασιάσουμε την εξίσωση με $\frac{\mathrm{AB}}{\Gamma}$ και να διατηρηθεί η ισότητα με το μηδέν.

$$\frac{\mathrm{AB}}{\Gamma}\left(\frac{\Gamma}{\mathrm{B}}, \frac{\Gamma}{\mathrm{A}}\right) \cdot \vec{x} + \frac{\Gamma^2}{\mathrm{AB}} = 0 \Rightarrow (\mathrm{A}, \mathrm{B}) \cdot \vec{x} + \Gamma = 0.$$

Επομένως $\vec{w} = (\mathrm{A}, \mathrm{B})$ $\kappa\alpha\iota$ $b = \Gamma$

Μετά τον πολλαπλασιασμό της εξίσωσης της ευθείας με $\frac{\mathrm{AB}}{\Gamma}$ το $\vec{w}$ αποκτά την ίδια φορά με το εικονιζόμενο $\vec{u}$. Το b στην περίπτωση αυτή γίνεται θετικό.

Ποιος είναι ο ρόλος του b ή του Γ όσον αφορά τη θέση της ευθείας στο επίπεδο



Έστω η ευθεία με εξίσωση $\mathrm{Ax} + \mathrm{By} + \Gamma = 0$

Η απόσταση ενός σημείου D=(x_1, y_1) από την ευθεία δίνεται από τη σχέση

$$d_x = \frac{|Ax_1 + Bx_2 + \Gamma|}{\sqrt{\mathrm{A}^2 + \mathrm{B}^2}}$$

Αν $\vec{w} = (w_1, w_2)$ είναι το κάθετο διάνυσμα στην ευθεία η απόσταση του σημείου $\vec{x}$ γράφεται ως

$$d_x = \frac{|w_1 x_1 + w_2 x_2 + b|}{\|\vec{w}\|}$$

Η απόσταση της αρχής (0,0) από την ευθεία είναι $\frac{|b|}{\|\vec{w}\|}$.

Γενίκευση της σχέσης για επίπεδα και υπερεπίπεδα (hyperplane)

Ποιο είναι το πλεονέκτημα της χρήσης του κάθετου διανύσματος αντί του διανύσματος που καθορίζει τη διεύθυνση της ευθείας;

Στην περίπτωση της παλινδρόμησης και όταν τα δεδομένα μας περιγράφονται από μία ευθεία ο αλγόριθμος (perceptron) προσπαθεί να μάθει τη γραμμική σχέση που συνδέει τη συνιστώσα y έχοντας ως είσοδο τις τιμές του x. Επομένως παράγει ευθείες με στόχο να ελαχιστοποιήσει μία συνάρτηση απώλειας L (loss function). Στην περίπτωση όμως που η έξοδος εξαρτάται από ένα ζεύγος εισόδων τις οποίες μπορούμε να εκφράσουμε σαν διάνυσμα

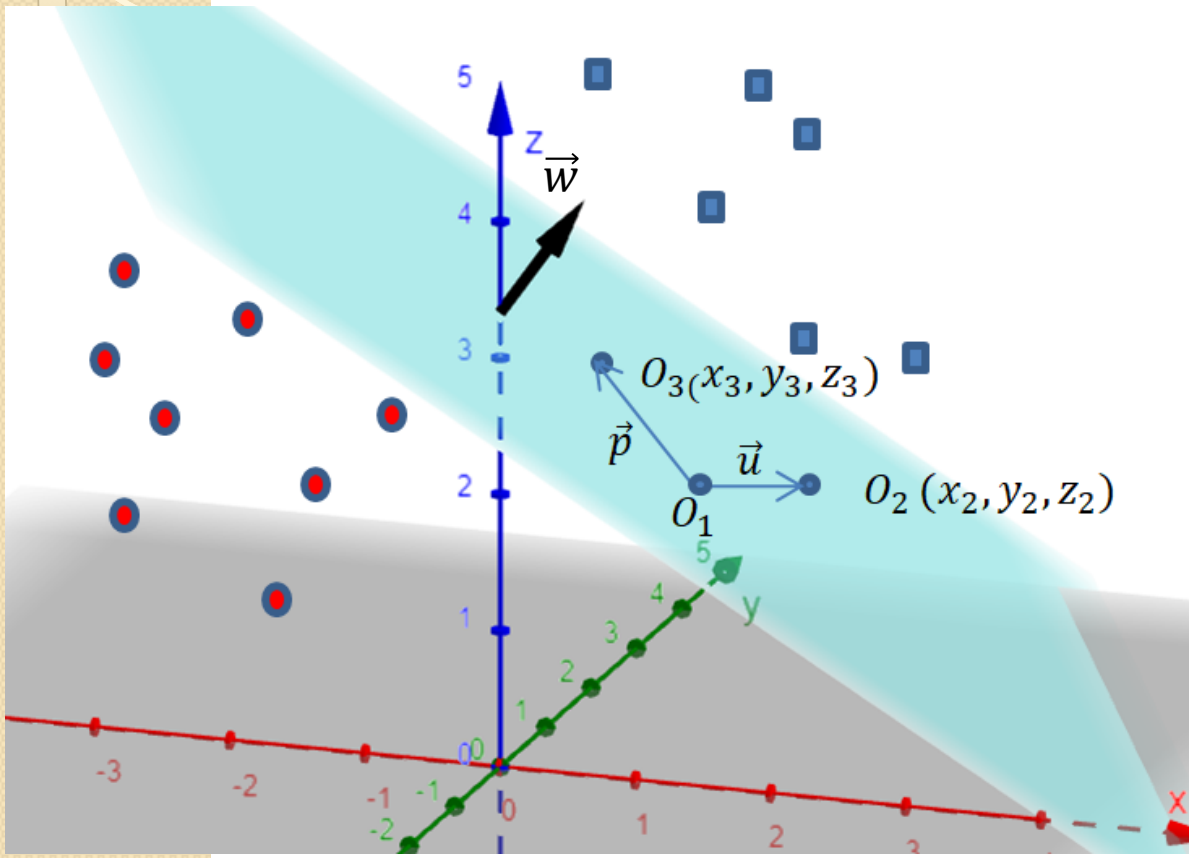
$$\vec{x} = (x_1, x_2)$$

τότε ο αλγόριθμός παράγει επίπεδα σε έναν τριδιάστατο χώρο με στόχο να ελαχιστοποιήσει πάλι την ίδια συνάρτηση κόστους.

Για παράδειγμα θα μπορούσαμε να θεωρήσουμε ένα πρόβλημα στο οποίο αναζητούμε τη σχέση που συνδέει την τιμή πώλησης ενός σπιτιού σε σχέση με την παλαιότητά του και το εμβαδόν του.

Προσδιορισμός του κάθετου διανύσματος στις 3 διαστάσεις

Έστω το επίπεδο που δίνεται από τη σχέση $Ax + By + \Gamma z + \Delta = 0$



$$\vec{u} = \overrightarrow{OO_2} - \overrightarrow{OO_1}$$

$$\vec{p} = \overrightarrow{OO_3} - \overrightarrow{OO_1}$$

$$\vec{w} \cdot \vec{u} = 0 \Rightarrow$$
$$\vec{w} \cdot \overrightarrow{OO_2} - \vec{w} \cdot \overrightarrow{OO_1} = 0 \quad (1)$$

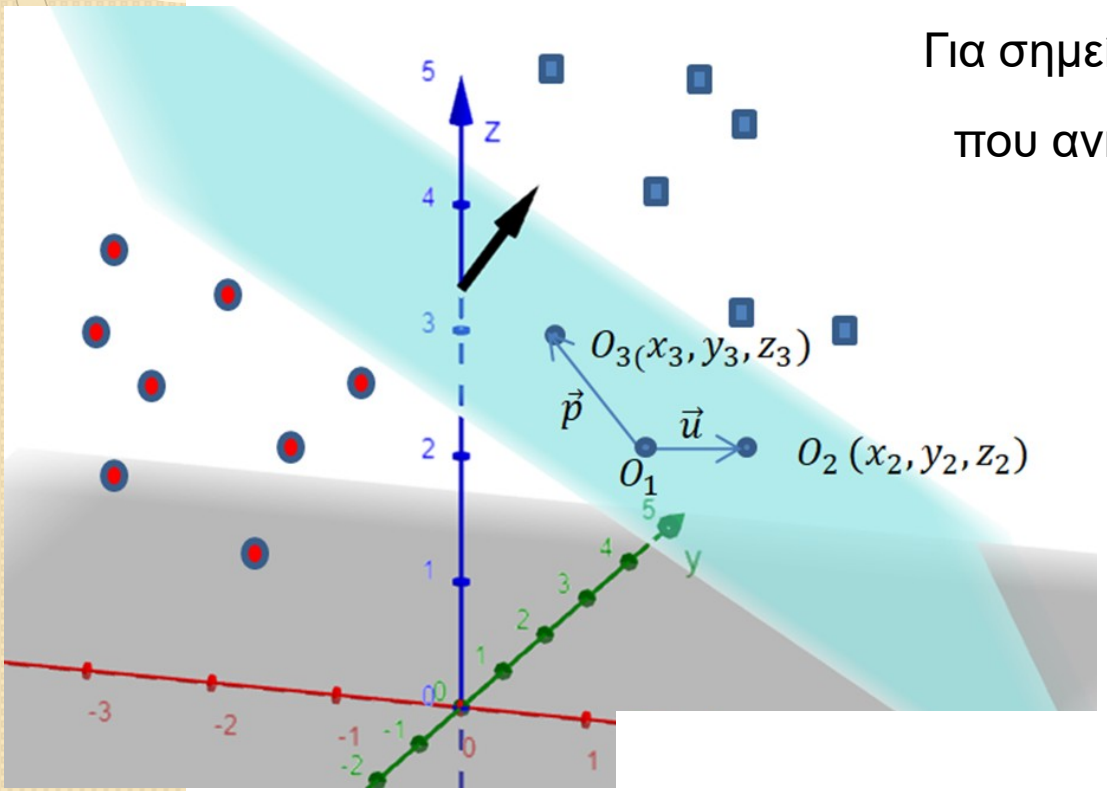
$$\vec{w} \cdot \vec{p} = 0 \Rightarrow$$
$$\vec{w} \cdot \overrightarrow{OO_3} - \vec{w} \cdot \overrightarrow{OO_1} = 0 \quad (2)$$

Αφαιρώντας τις (1) και (2) κατά μέλη προκύπτει

$$\vec{w} \cdot \left(\overrightarrow{OO_2} - \overrightarrow{OO_3}\right) = 0 \Rightarrow$$
$$\vec{w} \cdot \left((x_2, y_2, z_2) - (x_3, y_3, z_3)\right) = 0$$

$$\vec{w} \cdot (x_2 - x_3, y_2 - y_3, z_2 - z_3) = 0$$

Προσδιορισμός του κάθετου διανύσματος στις 3 διαστάσεις (συνέχεια)



Για σημεία (x_2, y_2, z_2) και (x_3, y_3, z_3)

που ανήκουν στο επίπεδο ισχύει

$$Ax_2 + By_2 + \Gamma z_2 + \Delta = 0 \ (3)$$
$$Ax_3 + By_3 + \Gamma z_3 + \Delta = 0 \ (4)$$

Αφαιρώντας κατά μέλη τις (3) και (4) παίρνουμε

$$A(x_2 - x_3) + B(y_2 - y_3) + \Gamma(z_2 - z_3) = 0$$

Ήδη έχουμε αποδείξει ότι

$$\vec{w} \cdot (x_2 - x_3, y_2 - y_3, z_2 - z_3) = 0$$

Συνδυάζοντας τις 2 προηγούμενες σχέσεις προκύπτει ότι

$\vec{w} \cdot (x_2 - x_3, y_2 - y_3, z_2 - z_3) = (A, B, \Gamma) \cdot ((x_2 - x_3, y_2 - y_3, z_2 - z_3)$

Επομένως $\vec{w} = (A, B, \Gamma)$

Παράδειγμα συνόλου δεδομένων που είναι γραμμικά διαχωρίσιμο

Σαν παράδειγμα κάποιων πολύ κοινών συνόλων δεδομένων που έχουν χρησιμοποιηθεί για την εκπαίδευση νευρωνικών δικτύων και είναι ελεύθερα διαθέσιμα (UCI dataset) μπορούμε να αναφερθούμε στο sonar dataset.

Τα δεδομένα σ' αυτήν την περίπτωση ανήκουν σε δύο κατηγορίες (binary classification). Το σύνολο περιλαμβάνει 111 πρότυπα (patterns) που είτε ανήκουν σε σήματα υπό διάφορες γωνίες που ανακλώνται από ένα μεταλλικό κύλινδρο είτε από πέτρινο κύλινδρο. Για το διαχωρισμό των σημάτων σε 2 κατηγορίες χρησιμοποιούνται 60 τιμές στο διάστημα από 0.0 μέχρι 1.0 που αντιπροσωπεύουν την ενέργεια που συγκεντρώνεται σε μία ζώνη συχνοτήτων για μία συγκεκριμένη χρονική περίοδο.

.

Μη γραμμικώς διαχωρίσιμα σύνολα δεδομένων

Στην περίπτωση που τα δεδομένα δεν είναι γραμμικώς διαχωρίσιμα ένας αλγόριθμος θα μπορούσε να λειτουργήσει στη λογική της ελαχιστοποίησης μίας συνάρτησης κόστους που μπορεί να είναι η συνάρτηση τετραγωνικού σφάλματος και εξαρτάται από το περιθώριο (margin) των σημείων σε σχέση με το υπερεπίπεδο που έχει διαμορφωθεί μέχρι εκείνη τη στιγμή)

$$\sum_j (y_j(\vec{w} \cdot \vec{x}_j + b) - 1)^2$$

Δεν εξασφαλίζεται ότι με τη διαδικασία αυτή που αντιστοιχεί σε παλινδρόμηση θα διαχωριστούν σωστά τα γραμμικώς διαχωρίσιμα σημεία.

Παρ' ολ' αυτά αποδεικνύεται ότι ανεξάρτητα από το αν υπάρχει χώρος των χαρακτηριστικών στον οποίο μπορώ να μεταβώ και τα σημεία μου να είναι γραμμικώς διαχωρίσιμα ότι υπάρχει κατάλληλη απεικόνιση που τα καθιστά σε κάθε περίπτωση γραμμικώς διαχωρίσιμα.

Μη γραμμικώς διαχωρίσιμα σύνολα δεδομένων (Συνέχεια)

Στην απεικόνιση αυτή επεκτείνεται ο αρχικός χώρος των εισόδων κατά τόσες διαστάσεις όσες είναι και το πλήθος των σημείων και τοποθετείται κάθε διάνυσμα εισόδου στην ίδια απόσταση Δ από την αρχή των αξόνων στην αντίστοιχη διάσταση

$\vec{x}_i^{\,ext} = (\overrightarrow{x_i}, \Delta\delta_{1i}, \Delta\delta_{2i}, \dots, \Delta\delta_{li})$, όπου l το πλήθος των σημείων

και δ_{ij} το δέλτα του Kronecker, δηλ $\delta_{ij} = \begin{cases} 1, \alpha\nu\ i = j \\ 0, \alpha\nu\ i \neq j \end{cases}$.

Η διαδικασία της εύρεσης ενός υπερεπιπέδου που διαχωρίζει τα δεδομένα στο χώρο αυτό οδηγεί πάντα σε διαχωρισμό στον αρχικό χώρο στην περίπτωση που τα σημεία είναι εξαρχής γραμμικά διαχωρίσιμα.

Ένας αλγόριθμος όπως ο Perceptron που δρα με στόχο να διαχωρίσει τα σημεία στο χώρο επιχειρεί ελαχιστοποίηση μιας συνάρτησης κόστους J (αντικείμενη συνάρτηση-objective function)

$$J = \min_{\vec{u},\gamma} \left(\frac{1}{\gamma^2} + \frac{\Delta^{-2}}{\gamma^2} \sum_{i=1}^{l} (\max(0, \gamma - y_i \vec{u} \cdot \vec{z}_i))^2\right.$$

που σταθμίζει το μέτρο του περιθωρίου με τα σφάλματα περιθωρίου που οφείλονται στην αδυναμία κάποιων σημείων να διαχωριστούν με ένα δεδομένο περιθώριο που καθορίζεται σε εκείνο το βήμα από το διάνυσμα βάρους. Έχουμε ενσωματώσει το b στο $\vec{w}$ αυξάνοντας την διάσταση του $\vec{x}$ κατά ένα και θέτοντας στη συνιστώσα αυτή την τιμή 1. Το $\vec{z}_i$ είναι η επαυξημένη εκδοχή του $\vec{x}_i$ και $\vec{u}$ το μοναδιαίο του επαυξημένου $\overrightarrow{w}$.

Ο Perceptron στην απλή του μορφή δεν μπορεί να εξασφαλίσει διαχωρισμό με περιθώριο, αλλά απλώς διαχωρισμό.

Στην επέκταση του αρχικού χώρου τα διανύσματα των σημείων έχουν τη μορφή:

$$\vec{x}_1^{\,ext} = (\overrightarrow{x_1}, \Delta, 0, \dots, 0),$$
$$\vec{x}_2^{\,ext} = (\overrightarrow{x_2}, 0, \Delta, \dots, 0),$$
$$\dots$$
$$\vec{x}_l^{\,ext} = (\overrightarrow{x_l}, 0, 0, \dots, \Delta)$$

$$J = \min_{\vec{u},\gamma} \left(\frac{1}{\gamma^2} + \frac{\Delta^{-2}}{\gamma^2} \sum_{\iota=1}^{l} (\max(0, \gamma - y_i \vec{u} \cdot \vec{z}_i))^2\right.$$

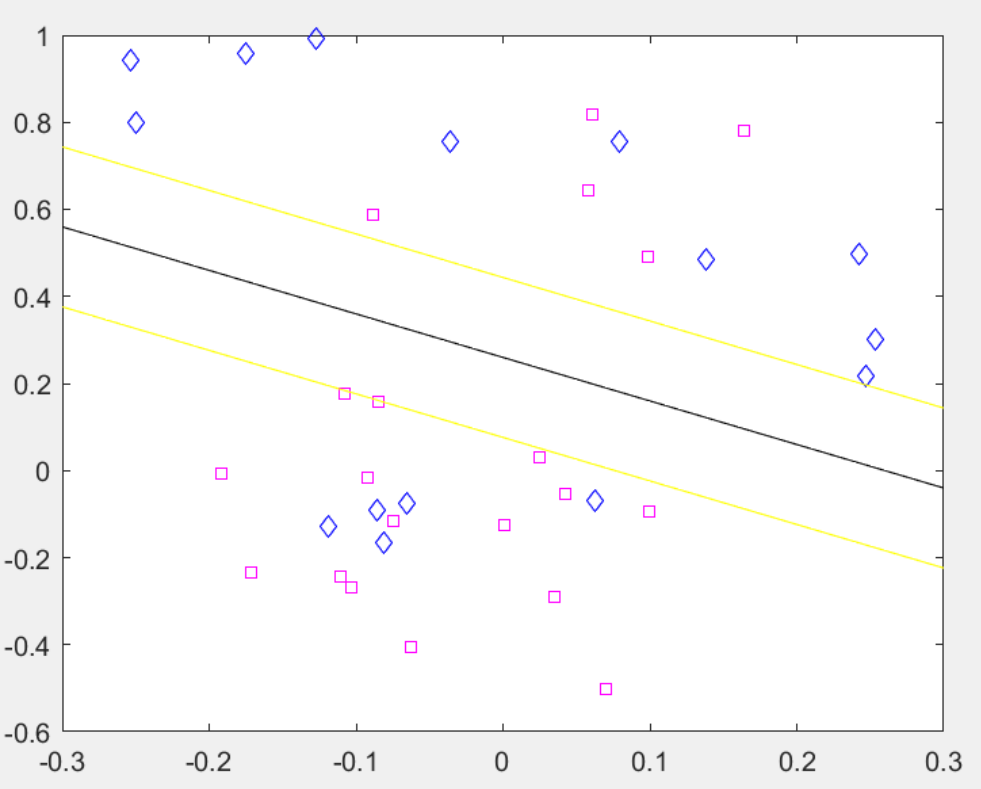
Όταν η τιμή της παραμέτρου Δ είναι μεγάλη υποβαθμίζεται η συμβολή του όρου του σφάλματος $\sum_{\iota=1}^{l}(\max(0, \gamma - y_i \vec{u} \cdot \vec{z}_i))^2$ και προκρίνονται λύσεις με μεγάλο περιθώριο στον εκτεταμένο χώρο και οδηγούμαστε σε μεγαλύτερα περιθώρια γ.
Όταν η τιμή της παραμέτρου Δ είναι μικρή δίνεται έμφαση στη μείωση των σφαλμάτων περιθωρίου. Κατ' επέκταση οδηγούμαστε σε μικρότερο περιθώριο γ.

Ο Perceptron στα μη γραμμικώς διαχωρίσιμα σύνολα

Θα εφαρμόσουμε την απεικόνιση σε χώρο που επαυξάνει τις διαστάσεις του αρχικού όσα και τα σημεία και τα τοποθετεί στην ίδια απόσταση στην επιπλέον διάσταση.

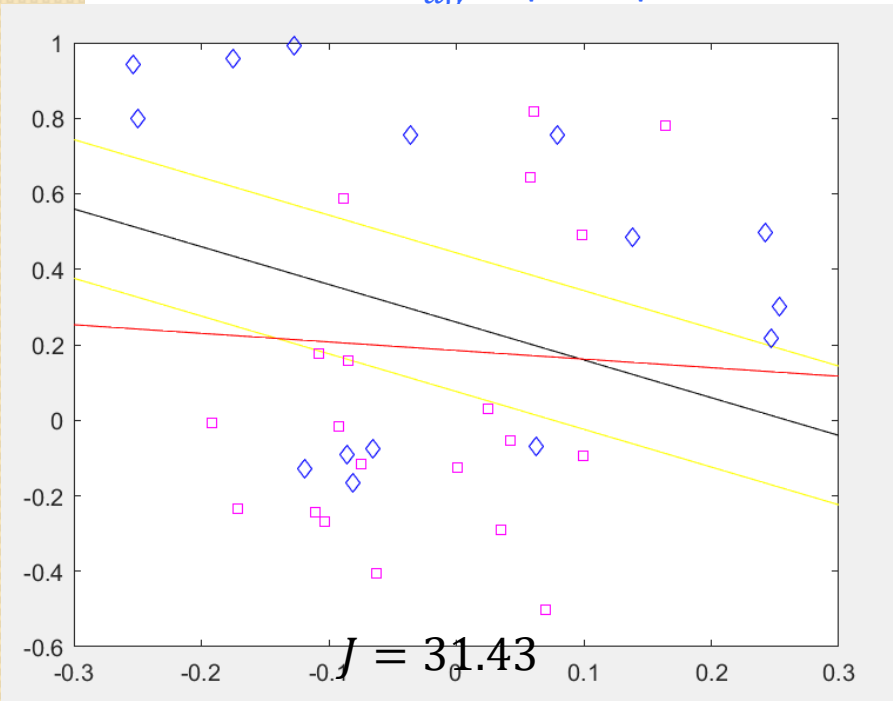
Σε όλα τα επόμενα παραδείγματα επειδή αποσκοπούσαμε σε όσο το δυνατόν μεγαλύτερο περιθώριο στο χώρο που μεταβήκαμε για να καταστήσουμε διαχωρίσιμα τα δεδομένα μας εκτελέσαμε τον Perceptron με παράμετρο

$$\mathrm{B} = 10R^2$$



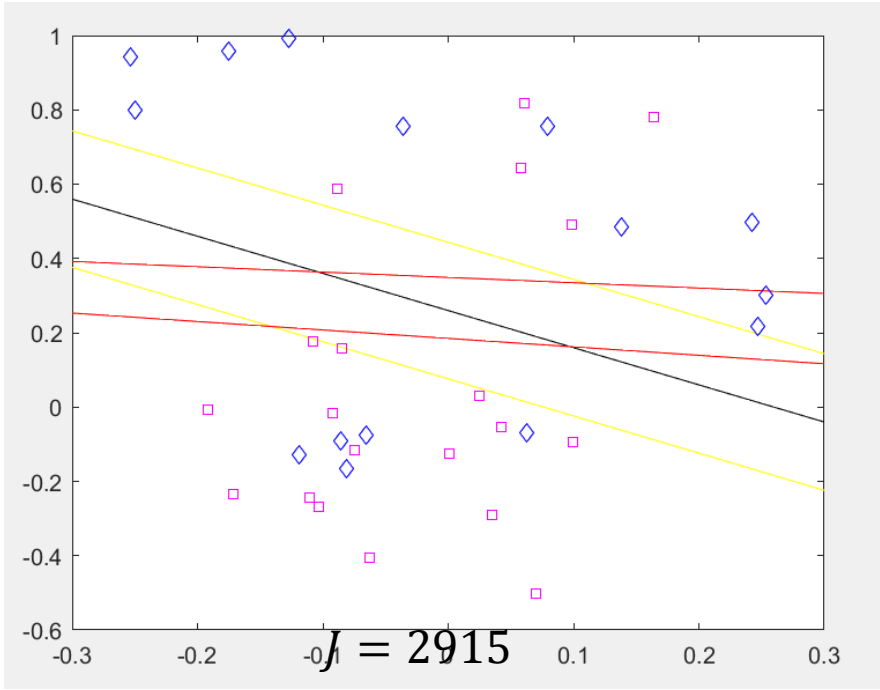
Θα εφαρμόσουμε την απεικόνιση σε χώρο που επαυξάνει τις διαστάσεις του αρχικού όσα και τα σημεία και τα τοποθετεί στην ίδια απόσταση στην επιπλέον διάσταση.

$$J = \min_{\vec{u},\gamma} \left(\frac{1}{\gamma^2} + \frac{\Delta^{-2}}{\gamma^2}\sum_{i=1}^{l}(\max(0, \gamma - y_i\vec{u}\cdot\vec{z}_i)\right)^2$$



$J = 31.43$

Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Τα σφάλματα περιθωρίου είναι 27.5063. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.95.



$J = 2915$

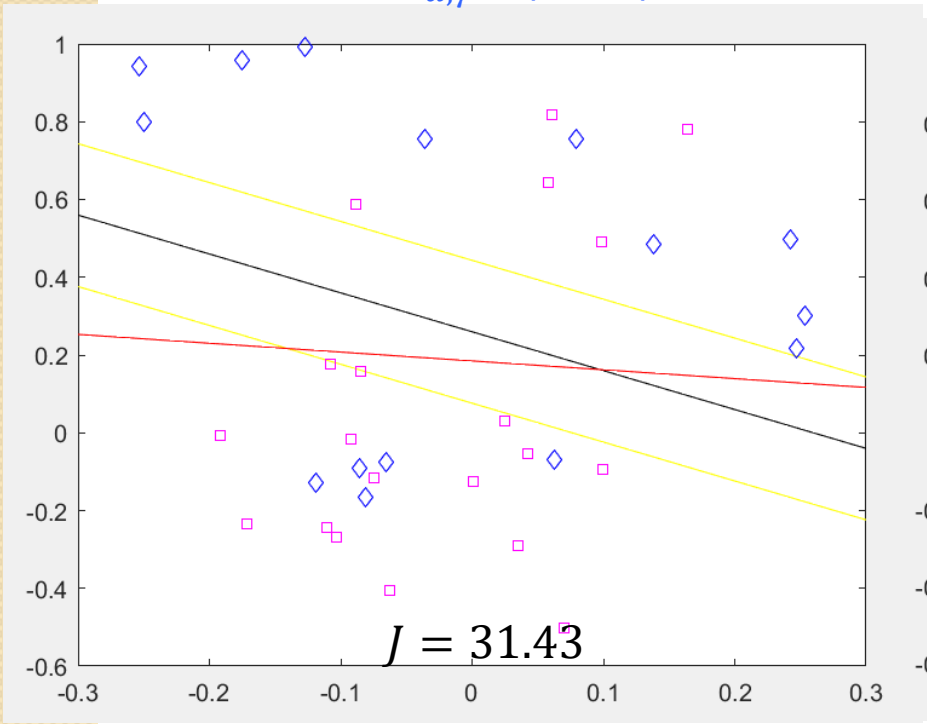
Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 0.1$. Δίνεται μεγαλύτερη έμφαση στη μείωση του σφάλματος και λιγότερο στο περιθώριο που έχει επιτευχθεί στο χώρο με επιπλέον διαστάσεις. Τα σφάλματα περιθωρίου είνα 2.69. Το περιθώριο ως προς το οποίο υπολογίζοντα τα σφάλματα περιθωρίου είναι 0.305.

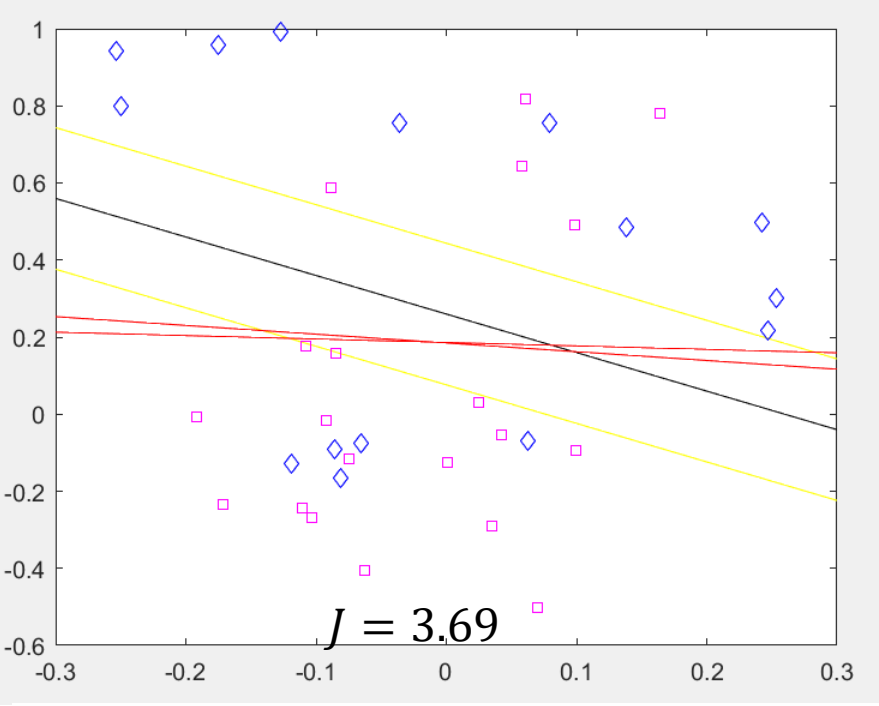
Η παράμετρος Δ θα προσδιοριστεί με τη βοήθεια ενός ανεξάρτητου dataset (validation set)με στόχο τη μείωση των λανθασμένων ταξινομήσεων.

Οι 2 J δεν είναι ευθέως συγκρίσιμες μεταξύ τους επειδή αναφέρονται σε διαφορετική παράμετρο Δ. (Θα σχολιαστεί στη συνέχεια)

Ο αλγόριθμος τρέχει σε όλες τις περιπτώσεις με $B = 10R^2$. Στόχος είναι η ελαχιστοποίηση της παρακάτω συνάρτησης κόστους

$$J = \min_{\vec{u},\gamma} \left(\frac{1}{\gamma^2} + \frac{\Delta^{-2}}{\gamma^2}\sum_{i=1}^{l}(\max(0, \gamma - y_i\vec{u}\cdot\vec{z}_i)\right)^2$$

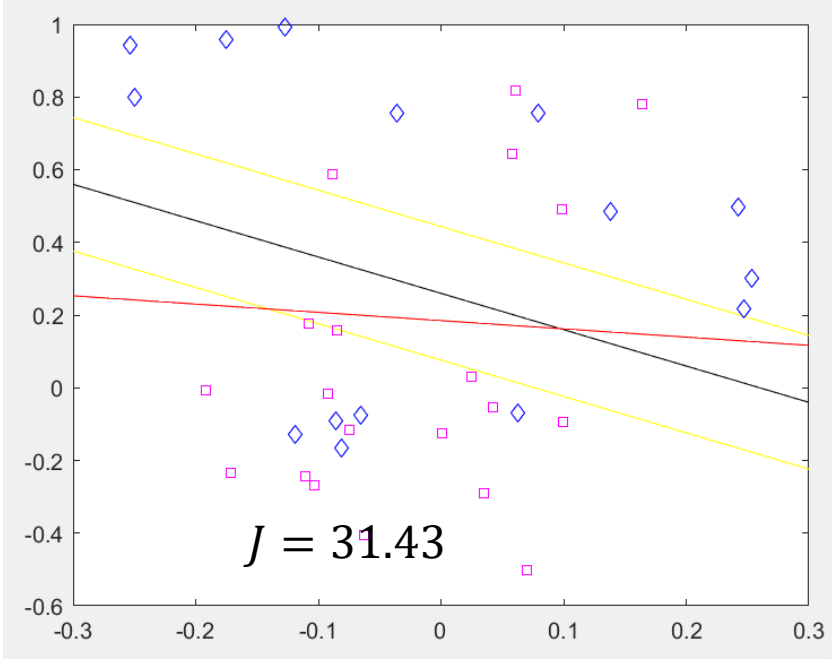


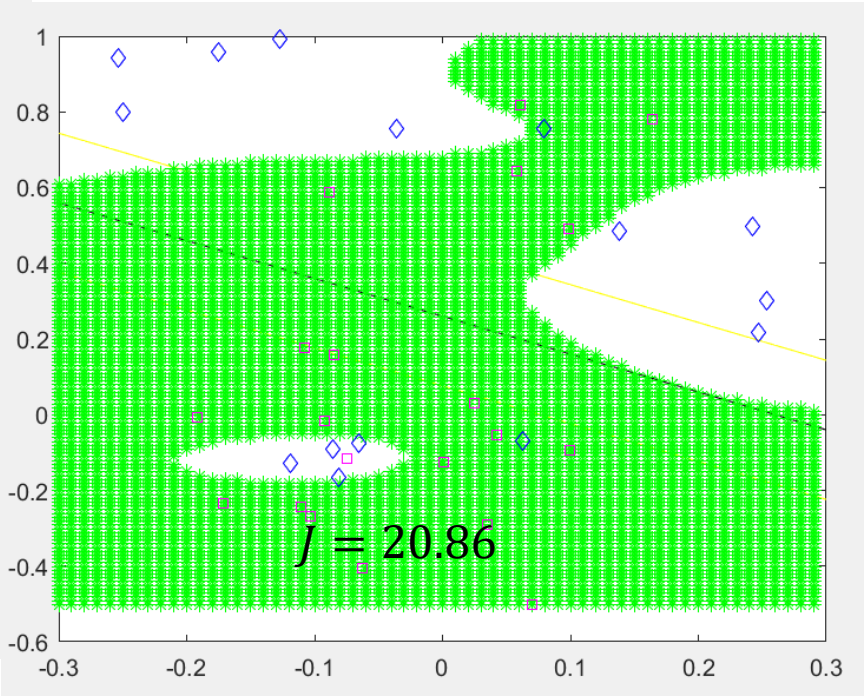


Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Τα σφάλματα περιθωρίου είναι 27.5063. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.95.

Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 3$. Δίνεται μεγαλύτερη έμφαση στο περιθώριο που έχει επιτευχθεί στο χώρο με επιπλέον διαστάσεις και λιγότερο στη μείωση του σφάλματος. Τα σφάλματα περιθωρίου είναι 223.88. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα είναι 2.65.

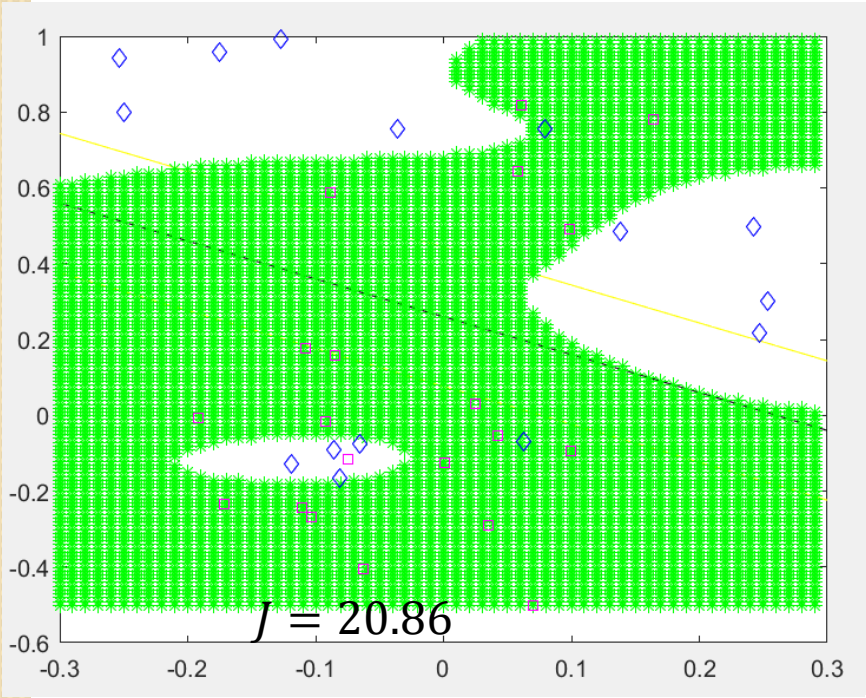
Για να προκύψει η δεξιά εικόνα χρησιμοποιήθηκε το $K(\vec{x},\vec{z}) = e^{-\|\vec{x}-\vec{z}\|^2/2\sigma^2}$ για όλα τα $\vec{x} \neq \vec{z}$, ενώ για $\vec{x} = \vec{z}$ παίρνει την τιμή $K(\vec{x},\vec{z}) = 1 + \Delta^2$, καθώς λόγω της επέκτασης του χώρου των σημείων, στον kernel matrix του αρχικού 'rbf' kernel προστίθεται ο όρος Δ^2 στα διαγώνια στοιχεία του.

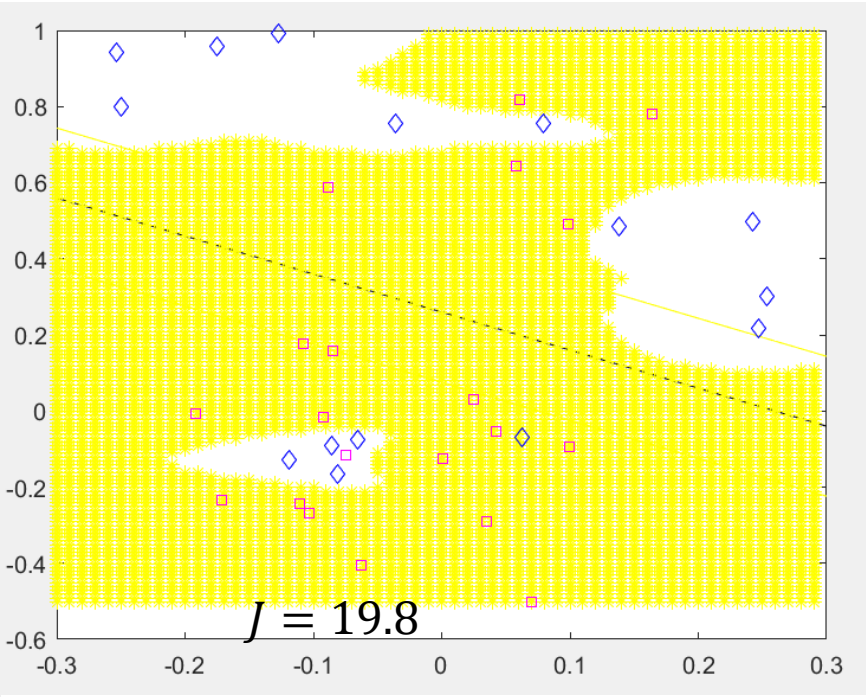




Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Τα σφάλματα περιθωρίου είναι 27.5063. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.95.

Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Εφαρμόστηκε 'rbf' kernel με τυπική απόκλιση σ=0.1. Τα σφάλματα περιθωρίου είναι 2.1. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.39.





Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Εφαρμόστηκε ‘rbf’ kernel με τυπική απόκλιση σ=0.1. Τα σφάλματα περιθωρίου είναι 2.1. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.39.

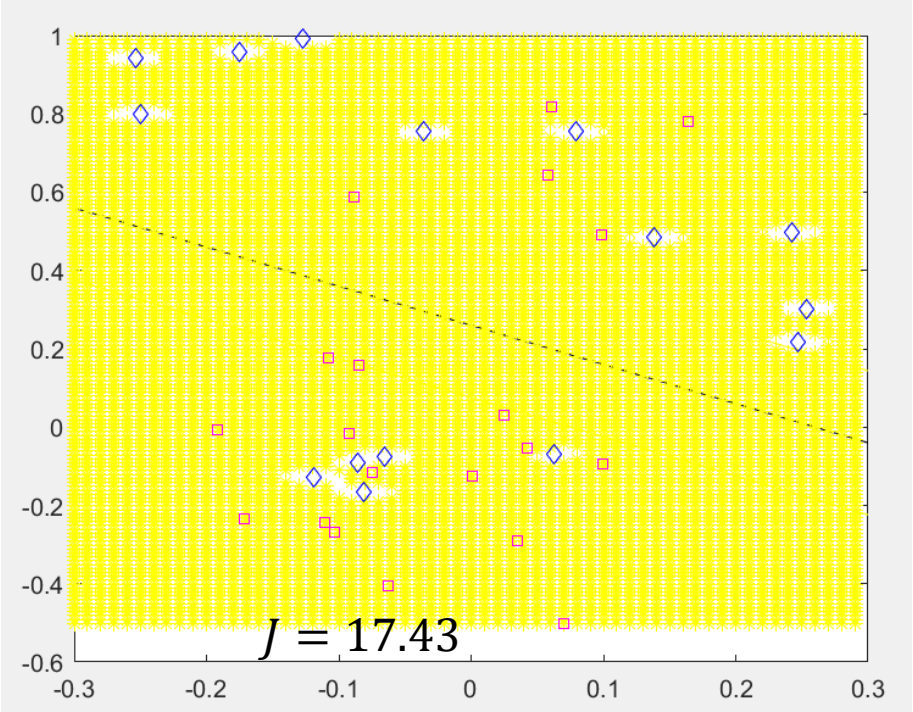
Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Εφαρμόστηκε ‘rbf’ kernel με τυπική απόκλιση σ=0.05. Τα σφάλματα περιθωρίου είναι 1.52. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.35.

Υπερμοντελοποίηση (Overfitting) και Γενίκευση (Generalisation)

Παρατηρούμε ότι το 'rbf' kernel εντάσσει όλο το χώρο στη θετική κλάση εκτός από τα σημεία της αρνητικής κλάσης. Το μοντέλο αυτό έχει πολύ μεγάλη πολυπλοκότητα και δεν αποτελεί καλή επιλογή.

Χρησιμοποιούμε μοντέλα που μπορούν να γενικεύσουν σε δεδομένα που δεν είναι γνωστά (testing data). Για να επιλέξουμε μοντέλο που δίνει μεγάλη ακρίβεια σε άγνωστα δεδομένα που προέρχονται από την ίδια κατανομή απομονώνουμε κάποιο σύνολο δεδομένων για επαλήθευση (validation data) από τα δεδομένα που έχουμε στη διάθεση μας για εκπαίδευση.

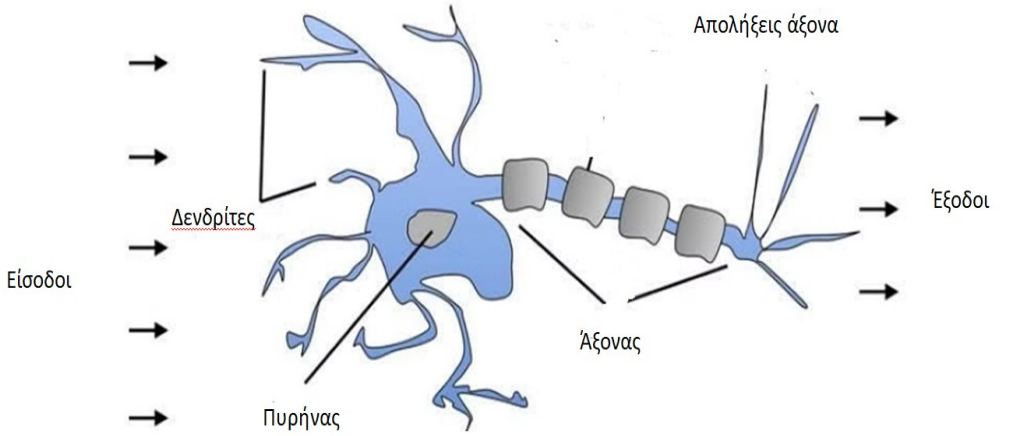
Ελέγχουμε την ακρίβεια που επιτυγχάνει ο αλγόριθμος σε σχέση με το σύνολο επαλήθευσης. Με αυτόν τον τρόπο θα γίνει επιλογή της πολυπλοκότητας, δηλ. γραμμικό, πολυωνυμικό ή Gaussian kernel καθώς και της τιμής υπερπαραμέτρων όπως του Δ.

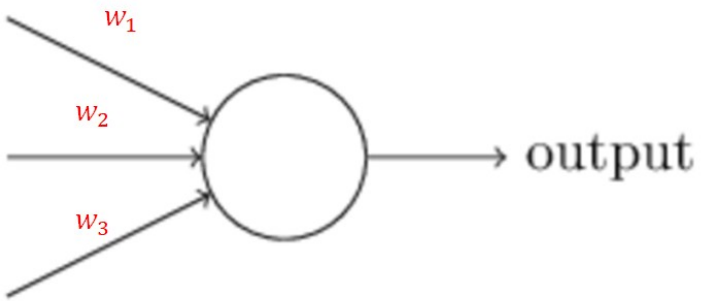


Τα σημεία τοποθετήθηκαν στην επιπλέον διάσταση σε απόσταση $\Delta = 1$. Εφαρμόστηκε 'rbf' kernel με τυπική απόκλιση σ=0.01. Τα σφάλματα περιθωρίου είναι 0.85. Το περιθώριο ως προς το οποίο υπολογίζονται τα σφάλματα περιθωρίου είναι 0.32.

Perceptron

Ένας τύπος τεχνητού νευρώνα είναι ο Perceptron που προσομοιάζει τη λειτουργία του βιολογικού νευρώνα.





Δέχεται εισόδους και παράγει μία έξοδο η οποία είναι δυαδικού τύπου, δηλαδή είτε 0 είτε 1
Ο Perceptron αναπτύχθηκε τη δεκαετία του 1950 από τον Rosenblatt ο οποίος βασίστηκε στη δουλειά των McCulloch και Pitts.

Οι είσοδοι συνδυάζονται με βάρη ως εξής:

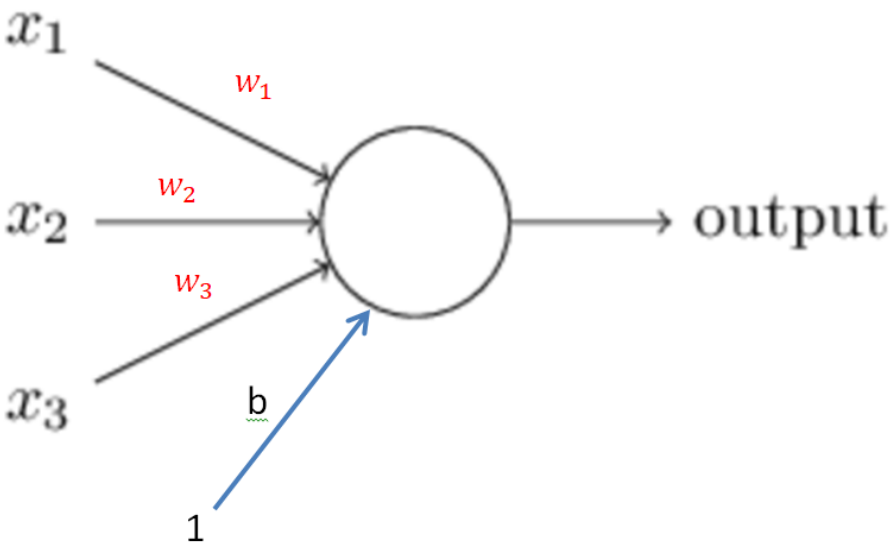
$$w_1x_1 + w_2x_2 + w_3x_3 = \sum_j w_jx_j = \vec{w} \cdot \vec{x} \ (5)$$

ώστε να καθορίσουν τη σημασία που έχει κάθε είσοδος στη διαμόρφωση της τελικής εξόδου.

Αν θεωρήσουμε ότι τα $\vec{w}$ και $\vec{x}$ είναι πίνακες στήλης τότε η (5) γράφεται σαν

$$\sum_j w_jx_j = \vec{w}^T\vec{x}$$

Προσθήκη του bias



Επιβάλλεται να προσθέτουμε στις εισόδους του Perceptron και μία επιπλέον είσοδο b (bias) που δεν εξαρτάται από τις εισόδους και μας παρέχει μεγαλύτερη ευελιξία στην εκπαίδευσή του.
Όταν λέμε εκπαίδευση εννοούμε τον προσδιορισμό μέσα από βήματα των παραμέτρων του νευρωνικού δικτύου που μπορούν να μαθευτούν. Τέτοιες είναι τα
βάρη w_j και το bias b.

Μετά την προσθήκη του bias η σχέση (5) γίνεται

$$w_1x_1 + w_2x_2 + w_3x_3 + b = \sum_j (w_jx_j + b) = \vec{w} \cdot \vec{x} + b$$

Αν θεωρήσουμε ότι τα $\vec{w}$, $\vec{x}$ είναι διανύσματα στήλης σε μορφή πινάκων η προηγούμενη σχέση γράφεται ως

$$\sum_j (w_jx_j + b) = \vec{w}^T\vec{x} + \mathrm{b}$$

Συνάρτηση ενεργοποίησης και εκπαίδευση του Perceptron για τη δυαδική κατηγοριοποίηση (binary classification)

Στη συνέχεια εφαρμόζεται στο άθροισμα των σταθμισμένων εισόδων μία συνάρτηση ενεργοποίησης σ της οποίας η τιμή αποτελεί την έξοδο του νευρωνικού δικτύου. Για κάθε είσοδο $\vec{x}_j$ υπολογίζεται η έξοδος από τη σχέση που αποτελεί και την πρόβλεψή μας για την κατηγορία που ανήκει

$$\sigma(\vec{w}\cdot\vec{x}_j+b)=\begin{cases}1, & \alpha\nu\ \vec{w}\cdot\vec{x}_j+\mathrm{b}>0\\ -1, & \alpha\nu\ \vec{w}\cdot\vec{x}_j+\mathrm{b}\leq 0\end{cases} \quad (6)$$

Καθώς τα δεδομένα χωρίζονται σε 2 κατηγορίες αν το αποτέλεσμα μας οδηγήσει στον πρώτο κλάδο της (6), εκτιμούμε ότι η είσοδος ανήκει στη θετική κλάση με ετικέτα $\hat{y}_j=1$, ενώ όταν οδηγούμαστε στον δεύτερο κλάδο εκτιμούμε ότι η είσοδος ανήκει στην αρνητική κλάση με με ετικέτα $\hat{y}_j=1$.

Τα δεδομένα οδηγούνται διαδοχικά στον Perceptron και κάθε φορά που υπάρχει διαφοροποίηση μεταξύ της εκτίμησης και της πραγματικής ετικέτας $\hat{y}_j\neq y_j$ οδηγούμαστε σε ανανέωση τον τιμών των παραμέτρων (βάρη και bias).

Όταν εξαντληθούν τα δεδομένα ανακυκλώνονται από την αρχή. Κάθε τέτοια ανακύκλωση καλείται εποχή. Η σειρά με την οποία βλέπει τα δεδομένα ο αλγόριθμος μπορεί να είναι τυχαία.

Συνάρτηση απώλειας (loss function)

Για να αξιολογήσουμε την απόδοση του Perceptron μπορούμε να χρησιμοποιήσουμε μία συνάρτηση εκτίμησης των αστοχιών του Perceptron που καλούμε συνάρτηση απώλειας (loss function)
Μία λογική συνάρτηση απώλειας για τον Perceptron είναι

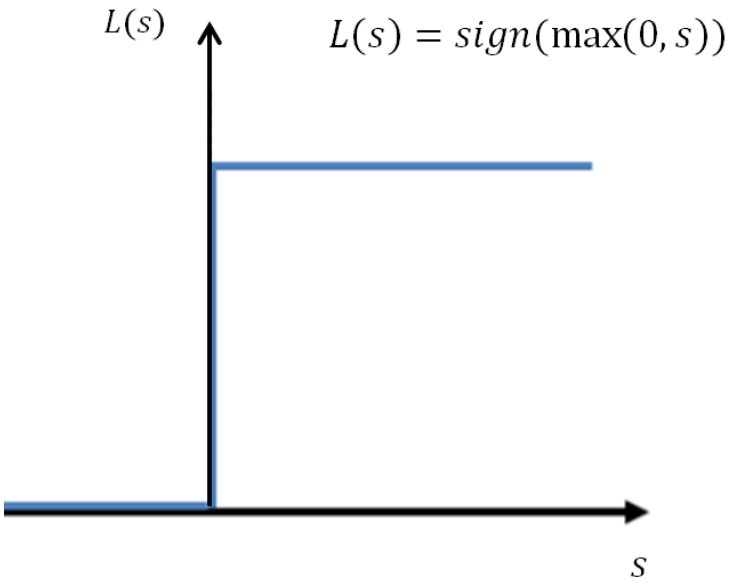
$$L = \sum_j \text{sign}(\max(0, -y_j(\vec{w} \cdot \vec{x}_j + b))) \quad (7)$$

Εξετάζοντας την (7) διαπιστώνουμε ότι όταν η πρόβλεψη είναι η θετική κλάση, δηλαδή $\vec{w} \cdot \vec{x}_j + b > 0$ και η ετικέτα είναι αρνητική τότε $-y_j(\vec{w} \cdot \vec{x}_j + b) > 0$, οπότε το συγκεκριμένο σημείο είναι λάθος κατηγοριοποιημένο και συνεισφέρει με μία μονάδα στη συνάρτηση απώλειας.
Το ίδιο συμβαίνει όταν η πρόβλεψη είναι η αρνητική κλάση, δηλαδή $\vec{w} \cdot \vec{x}_j + b < 0$ και η ετικέτα είναι θετική τότε $-y_j(\vec{w} \cdot \vec{x}_j + b) > 0$, οπότε και πάλι το συγκεκριμένο σημείο συνεισφέρει μία μονάδα στη συνάρτηση απώλειας.
Συμπερασματικά μπορούμε να πούμε ότι η συνάρτηση απώλειας καταγράφει το πλήθος των σημείων που κατηγοριοποιούνται λανθασμένα.
Η παραπάνω συνάρτηση χαρακτηρίζεται και ως συνάρτηση απώλειας 0-1, λόγω των 2 δυνατών τιμών που λαμβάνει.

Χαρακτηριστικά της συνάρτησης απώλειας 0-1

Στόχος κατά την εκπαίδευση του αλγορίθμου είναι η ελαχιστοποίηση της συνάρτησης απώλειας όσον αφορά τα δεδομένα με τα οποία τροφοδοτείται ο αλγόριθμος.
Μία διαδομένη μέθοδος βελτιστοποίησης και εν προκειμένω ελαχιστοποίησης είναι η κατάβαση κλίσης της συνάρτησης (gradient descent).
Συνάρτηση απώλειας 0-1

$$L_{0-1} = \sum_j \text{sign}(\max(0, -y_j(\vec{w} \cdot \vec{x}_j + b)))$$

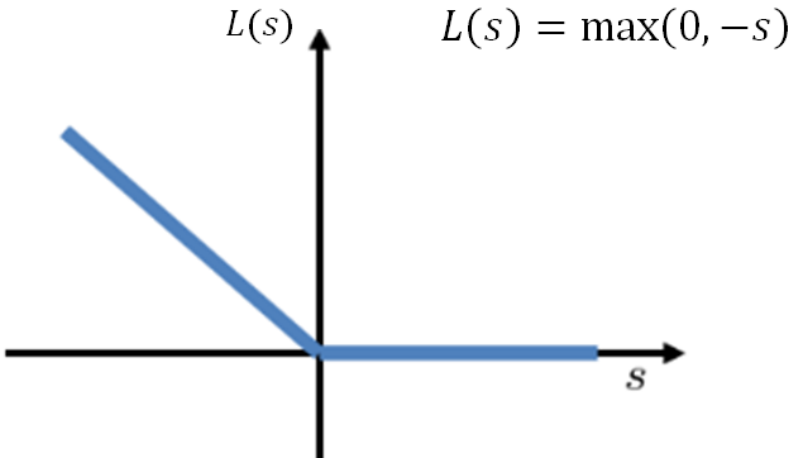


Η μέθοδος κατάβασης κλίσης δεν μπορεί να εφαρμοστεί για τη συνάρτηση απώλειας 0-1 γιατί δεν είναι διαφορίσιμη.
Επιπλέον δεν είναι κυρτή (convex) γεγονός που σημαίνει ότι υπάρχει πληθώρα τιμών των παραμέτρων του αλγορίθμου για τις οποίες ο αλγόριθμος συγκλίνει όταν οι 2 κλάσεις είναι γραμμικώς διαχωρίσιμες.

Άλλες συναρτήσεις απώλειας που επιτρέπουν την εφαρμογή της κατάβασης κλίσης

Για να ξεπεράσουμε το πρόβλημα ορίζουμε τη συνάρτηση απώλειας Perceptron η οποία δεν τιμωρεί με τον ίδιο τρόπο όλα τα λάθος κατηγοριοποιημένα σημεία αλλά αποδίδει διαφορετικό μέτρο ανάλογα με το πόσο λάθος είναι.

$$L_p = \sum_j \max\left(0, -y_j(\vec{w} \cdot \vec{x}_j + b)\right)$$



Το πλεονέκτημα είναι ότι είναι διαφορίσιμη εκτός από το μηδέν.

Επιπλέον είναι κυρτή που σημαίνει ότι υπάρχει μοναδική λύση.

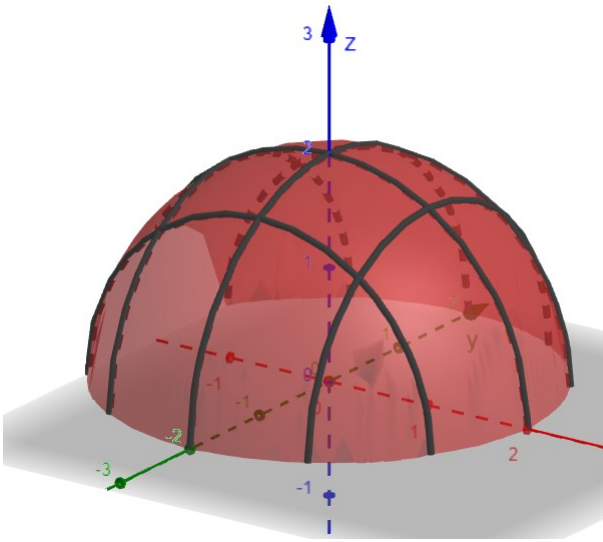
Παρατηρώντας τη συνάρτηση απώλειας διαπιστώνουμε ότι μπορεί να μηδενιστεί για $\vec{w} = 0$ και $b = 0$.

Τελεστής ανάδελτα , βαθμίδα ή κλίση∇-Γενίκευση της παραγώγου στην περίπτωση που η συνάρτηση έχει περισσότερες από μία μεταβλητές

Αν θεωρήσουμε τη συνάρτηση (βαθμωτό πεδίο) $L: \mathbb{R}^2 \to \mathbb{R}$

L= $\sqrt{4-x^2-y^2}$ υπάρχουν καμπύλες πάνω της που αν κινηθώ κατά μήκος τους θα έχω μεταβολή της L.

Μας ενδιαφέρει το μέγεθος αυτής της μεταβολής που το σημειώνουμε με $\frac{dL}{dS}$, όπου το ds είναι μία στοιχειώδης μετατόπιση πάνω στην καμπύλη στη διεύθυνση του μοναδιαίου $\vec{u} = a\vec{\imath} + \beta\vec{\jmath},\ a^2 + \beta^2 = 1$



Οι (παραμετρικές) εξισώσεις της ευθείας που διέρχεται από το σημείο $P_o(x_o, y_o)$ και είναι παράλληλη στο $\vec{u}$ είναι:

$x = x_o + as,\ y = y_o + \beta s$

Αν κινούμαστε πάνω σ' αυτήν την ευθεία, η L γράφεται

$L(x, y) = L(x_o + as, y_o + \beta s)$

$$\frac{\partial \mathrm{L}(P_o)}{\partial s} = \frac{\partial L}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial L}{\partial y}\frac{\partial y}{\partial s} = \frac{\partial L}{\partial x}a + \frac{\partial L}{\partial y}\beta = \left(\frac{\partial L}{\partial x}\vec{\imath} + \frac{\partial L}{\partial y}\vec{\jmath}\right)\cdot\left(a\vec{\imath} + \beta\vec{\jmath}\right)$$

$$\frac{\partial \mathrm{L}(P_o)}{\partial s} = \nabla L(P_o)\cdot\vec{u}$$ Παράγωγος κατά κατεύθυνση

Μεγιστοποιείται όταν η κατεύθυνση κίνησης συμπίπτει με το $\nabla L(P_o)$.

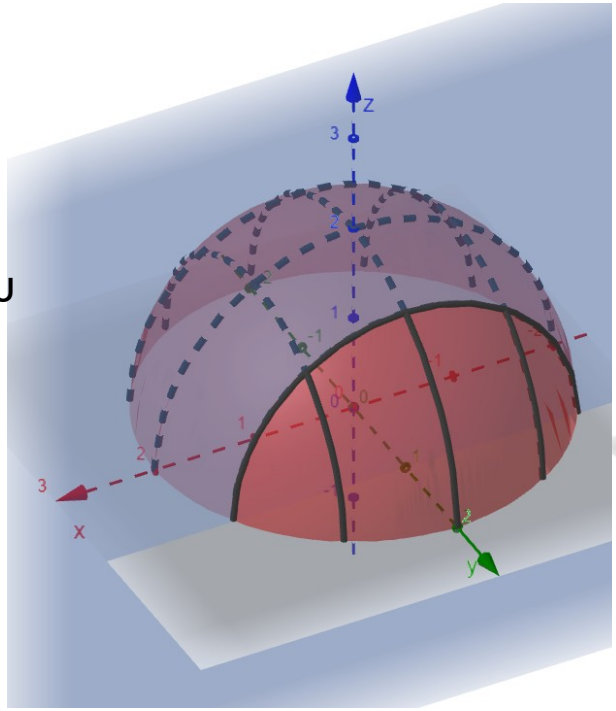
Για συνεχείς μερικές παραγώγους η ∇f ονομάζεται κλίση του βαθμωτού πεδίου L(x,y,) στο $P_o(x_o, y_o)$ και ισούται με

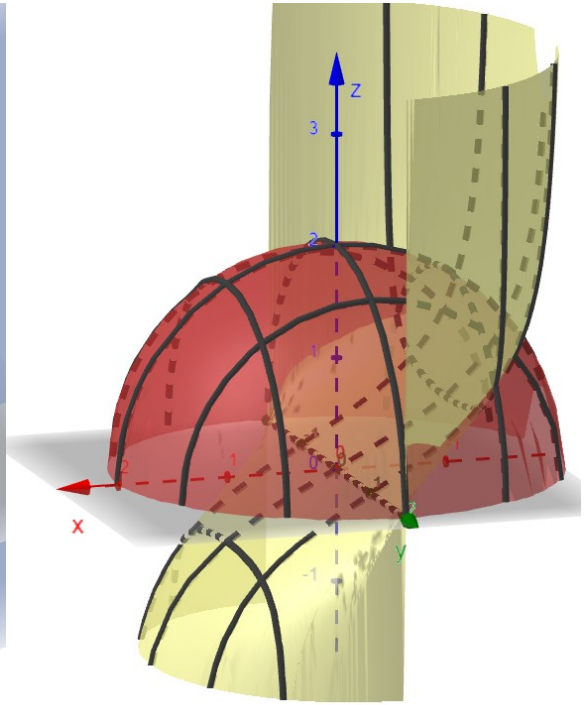
$$\nabla f(\mathrm{P}_O) = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y})|_{P_o}$$

Είναι διανυσματική συνάρτηση.

Η πρώτη συνιστώσα της $\nabla f(\mathrm{P}_O)$ αντιπροσωπεύει την κλίση της καμπύλης πάνω στην επιφάνεια που σχηματίζεται από την τομή της επιφάνειας με το επίπεδο $y = y_o$ παίρνοντας μια μικρή μεταβολή στο σημείο $P_o(x_o, y_o)$ στη διεύθυνση του άξονα των x.

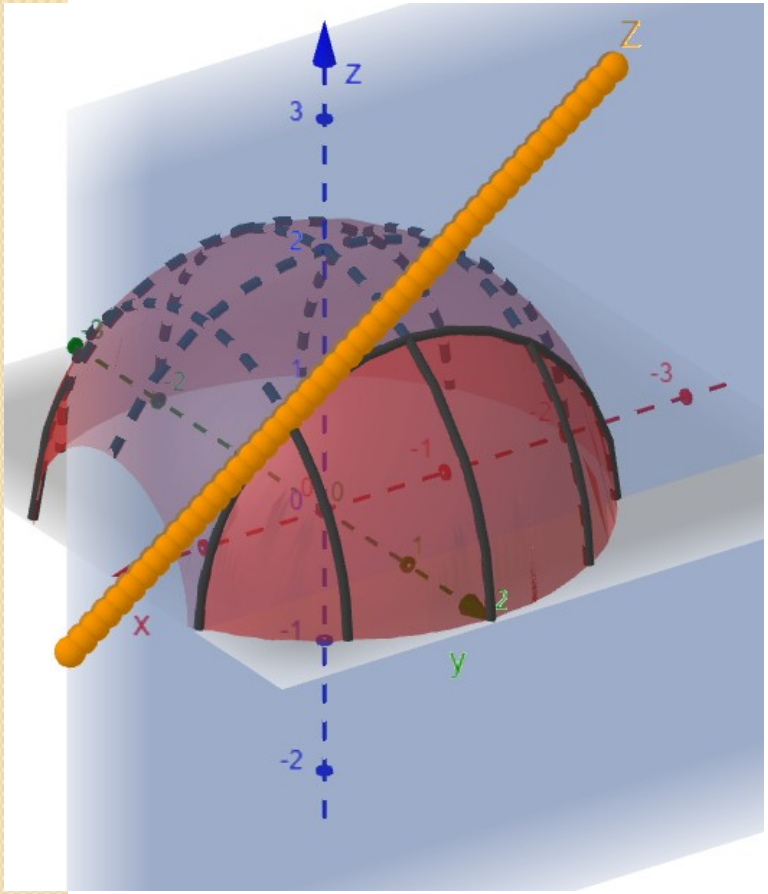
Το επίπεδο y=1 τέμνει την επιφάνεια του ημισφαιρίου



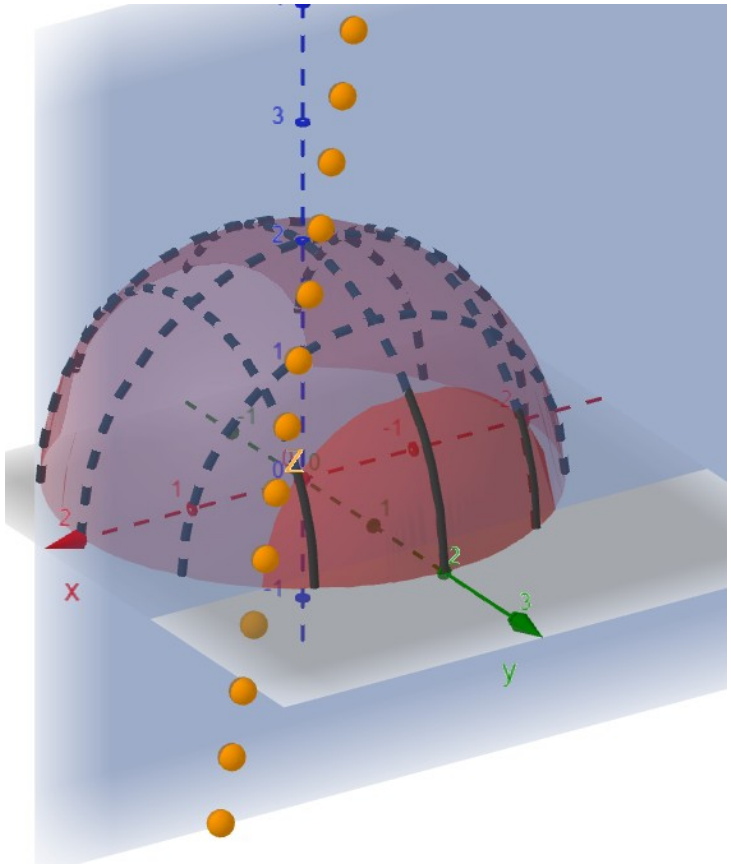


Η κίτρινη επιφάνεια αντιστοιχεί στις τιμές της συνιστώσας $\frac{\partial f}{\partial x}$ για τιμές των (x,y)

Εφαπτόμενες σε καμπύλες που βρίσκονται πάνω σε επίπεδο με σταθερό y

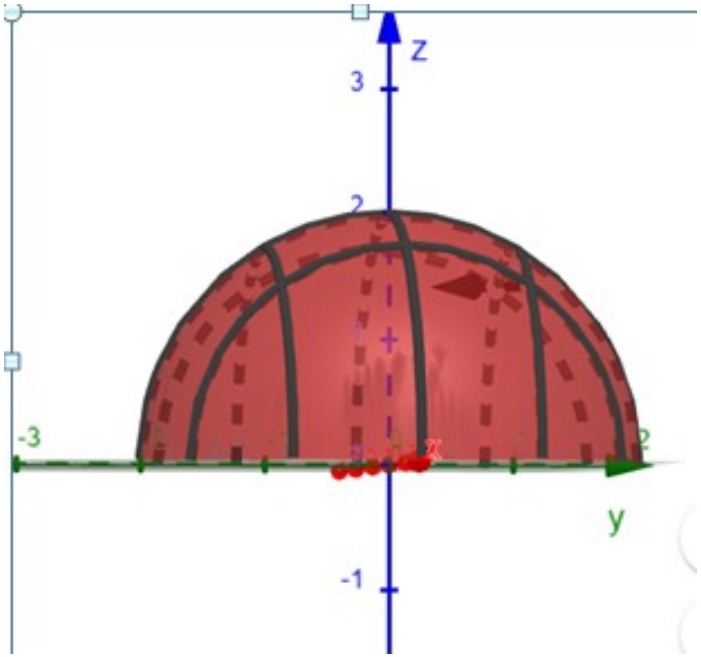


Εφαπτομένη σε καμπύλη πάνω στο ημισφαίριο που ορίζεται από την τομή της επιφάνειας με επίπεδο y=1.
Η εφαπτομένη λαμβάνεται στο σημείο (x,y)=(1,1).



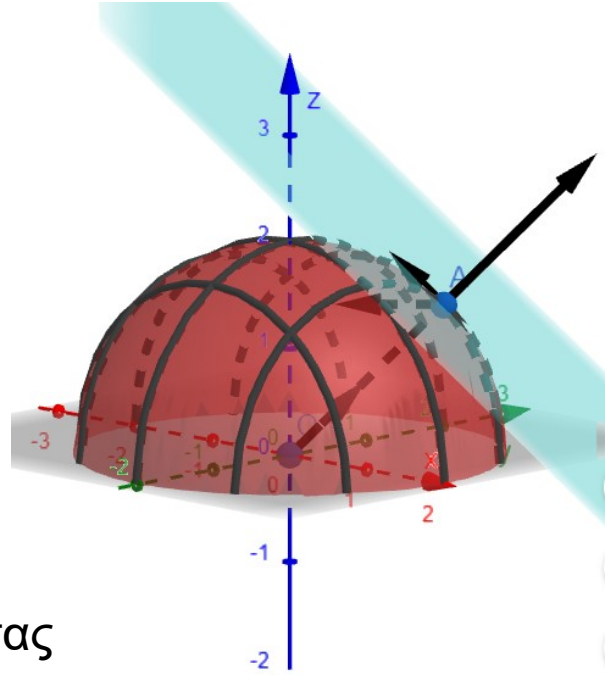
Εφαπτομένη σε καμπύλη πάνω στο ημισφαίριο που ορίζεται από την τομή της επιφάνειας με επίπεδο y=1.5.
Η εφαπτομένη λαμβάνεται στο σημείο (x,y)=(1.3,1.5).

Βαθμίδα ή κλίση ∇



$$\nabla f(\mathrm{P}_O) = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y})|_{P_o}$$

Είναι κάθετη σε ισοσταθμικές καμπύλες $x^2 + y^2 = c$ που είναι κύκλοι (οι παράλληλοι της σφαίρας).



Για να βρούμε την κατεύθυνση της μέγιστης αύξησης προβάλλουμε το διάνυσμα του $\nabla f(\mathrm{P}_O)$ στο επίπεδο που είναι εφαπτόμενο στη σφαίρα στο σημείο P_o. Η κατεύθυνση της μεγαλύτερης μείωσης είναι στην αντίθετη φορά.

Για να προσδιορίσουμε το εφαπτόμενο επίπεδο βρίσκουμε το κάθετο διάνυσμα σ' αυτό λαμβάνοντας το grad στη σφαίρα $x^2 + y^2 + z^2 = 4$

$\nabla(x^2 + y^2 + z^2) = (2\mathrm{x}, 2\mathrm{y}, 2\mathrm{z})$ (Η σφαίρα είναι ισοσταθμική επιφάνεια.)

Συνάρτηση απώλειας hinge loss-Εφαρμόζοντας την τεχνική κατάβασης κλίσης για τον προσδιορισμό του κανόνα ανανέωσης

Μία παραλλαγή της προηγούμενης συνάρτησης απώλειας είναι η hinge loss

$$L_h = \sum_j \max\left(0, 1 - y_j(\vec{w} \cdot \vec{x_j} + b)\right)$$

Για να βρούμε πως θα μεταβληθούν οι παράμετροι παίρνουμε την κλίση της L_p ή L_h ως προς τις συνιστώσες του $\vec{w}$ $(w_1, w_2, \dots, w_n)$ και το bias b και προκύπτει η διεύθυνση της μέγιστης αύξησης της L. n είναι η διάσταση του χώρου των εισόδων.
Επειδή στόχος μας είναι να ελαχιστοποιηθεί η L μεταβάλλουμε τα βάρη και το bias στην αντίθετη κατεύθυνση, μόνο κάθε φόρα που το σημείο x_j κατατάσσεται στην λάθος κατηγορία, δηλαδή $-y_j(\vec{w} \cdot \vec{x_j} + b) > 0$ για την L_p και $y_j(\vec{w} \cdot \vec{x_j} + b) < 1$ για L_h.
Ο κανόνας της ανανέωσης (update rule) του βάρους και του bias όταν συμβεί το $k+1$ σφάλμα είναι:

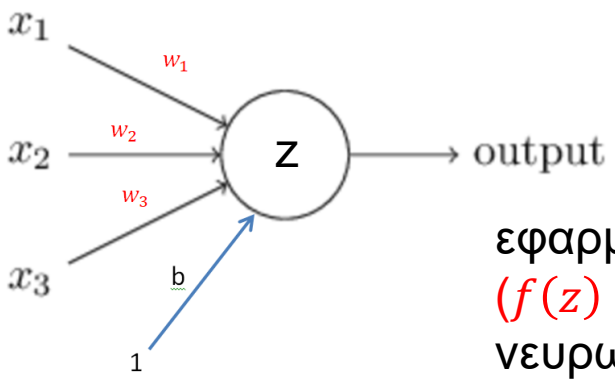
$$\vec{w}_{k+1} = \vec{w}_k + y_j \vec{x}_j \ (\alpha)$$
$$b_{k+1} = b_k + y_j \ (\beta)$$

$\nabla L_h = -y_j(\vec{x}_j, 1)$ αν $y_j(\vec{w} \cdot \vec{x_j} + b) \leq 1$
διαφορετικά μηδέν
Στις σχέσεις (α) και (β) προστίθενται το grad με – (μείον) γιατί θέλουμε να ακολουθήσουμε την κατεύθυνση της μέγιστης μείωσης και όχι αύξησης.

Στον κανόνα ανανέωσης μπορούμε να κάνουμε χρήση ενός ρυθμού εκμάθησης η διαφορετικού του 1.

$$\vec{w}_{k+1} = \vec{w}_k + \eta y_j \vec{x}_j$$
$$b_{k+1} = b_k + \eta y_j$$

Επέκταση των feed forward νευρωνικών δικτύων σε περισσότερα στρώματα

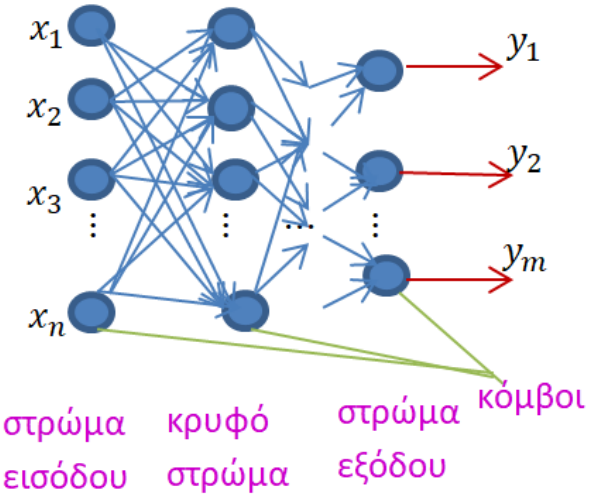


Αν στη συνάρτηση των σταθμισμένων εισόδων

$$z = h(\vec{w}) = \vec{w} \cdot \vec{x}_j + \mathrm{b}$$

εφαρμόσουμε γραμμική συνάρτηση ενεργοποίησης f $(f(z) = z)$, δηλαδή στην έξοδο πάρουμε αμετάβλητη την z, το νευρωνικό δίκτυο δεν έχει τη δυνατότητα να διαχωρίσει χωρίς λάθη μη γραμμικά διαχωρίσιμα δεδομένα.

Επεκτείνουμε το νευρωνικό δίκτυο σε περισσότερα στρώματα και κόμβους.
Αφού εισάγουμε στο δίκτυο ένα σημείο, το σήμα που αντιστοιχεί σ΄αυτό οδεύει αποκλειστικά προς τα εμπρός μέχρι να καταλήξει στην έξοδο



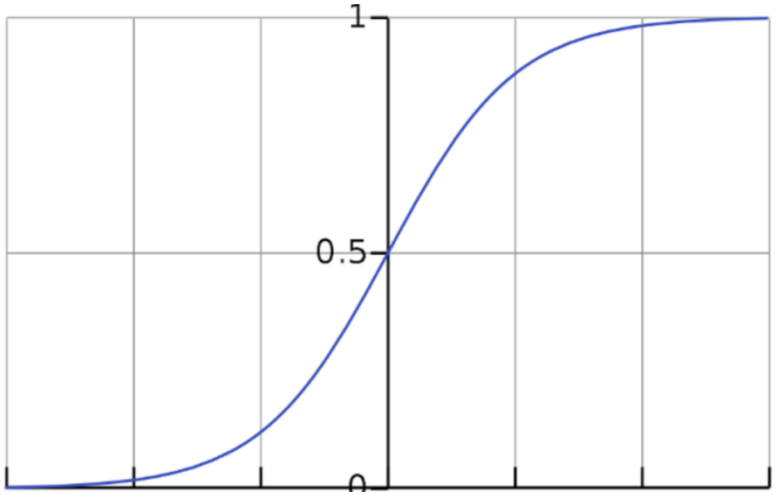
Στο στρώμα εισόδου το νευρωνικό δίκτυο δέχεται τις συνιστώσες του διανύσματος που αντιπροσωπεύει κάθε σημείο, δεν εφαρμόζεται συνάρτηση ενεργοποίησης.

Αν στους κόμβους των κρυφών στρωμάτων η ενεργοποίηση είναι γραμμική το νευρωνικό δίκτυο μπορεί να απλοποιηθεί σε ένα δίκτυο που έχει μόνο στρώμα εισόδου και εξόδου

Για τις εισόδους στο νευρωνικό δε χρειάζεται να θεωρήσουμε ξεχωριστό στρώμα εισόδου.

Συναρτήσεις ενεργοποίησης του νευρώνα σε κάποια κοινά προβλήματα

- Για την περίπτωση της δυαδικής κατηγοριοποίησης (binary classification) εφαρμόζεται στον κόμβο εξόδου η σιγμοειδής συνάρτηση ενεργοποίησης



$$f(z) = \frac{1}{1 + e^{-z}}$$

Αν η τιμή είναι πάνω από 0,5, δηλ. $z > 0$ αποδίδεται η ετικέτα 1 ενώ διαφορετικά η ετικέτα -1.

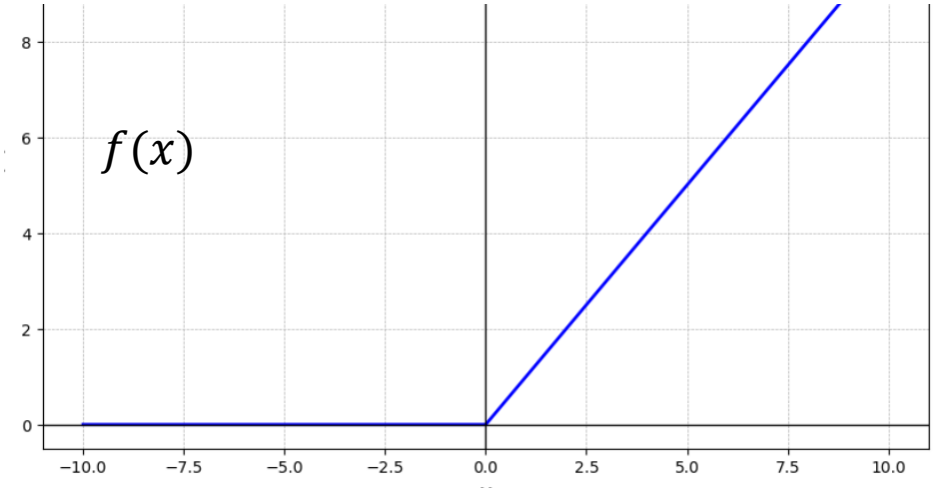
- Για την περίπτωση της κατηγοριοποίησης σε πολλαπλές κατηγορίες (multiclass classification) K το στρώμα εξόδου αποτελείται από τόσους κόμβους όσες και οι κατηγορίες. Σε κάθε κόμβο εξόδου εφαρμόζεται η softmax συνάρτηση ενεργοποίησης

$$f_i(\vec{z}) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Η softmax εξασφαλίζει ότι οι έξοδοι των κόμβων αθροίζονται στη μονάδα. Επομένως αντιπροσωπεύουν την πιθανότητα το σημείο να ανήκει στη συγκεκριμένη κατηγορία. Επιλέγεται η κατηγορία εκείνη με τη μεγαλύτερη πιθανότητα.

Η συνάρτηση ενεργοποίησης «Ανορθωμένης Γραμμικής Μονάδας» (Rectified Line Unit-RELU)

$$f(x) = \max(0, x)$$



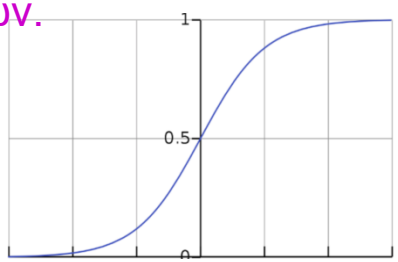
Χρησιμοποιείται ευρέως στα μοντέλα βαθιάς μάθησης δηλαδή νευρωνικά δίκτυα με μεγάλο αριθμό κρυφών στρωμάτων.

ΠΛΕΟΝΕΚΤΗΜΑΤΑ

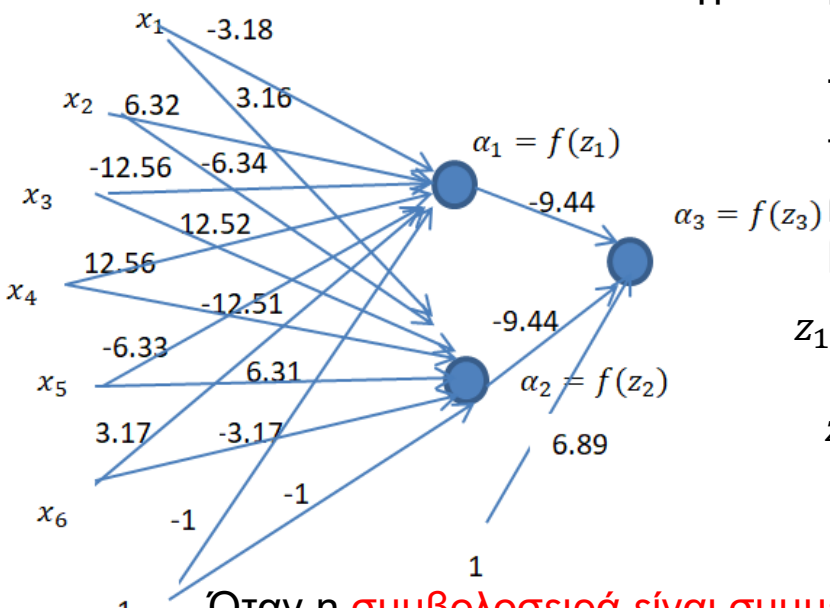
- Εισάγει μη γραμμικότητα γεγονός που επιτρέπει την επίλυση προβλημάτων που δεν είναι γραμμικώς διαχωρίσιμα
- Εισάγει απλότητα στον υπολογισμό του gradient descent ιδίως όταν εμπλέκονται εκατομμύρια παραμέτρων σε βαθιά νευρωνικά δίκτυα. Όταν η είσοδος είναι αρνητική το αποτέλεσμα είναι 0 και όταν είναι θετική το αποτέλεσμα είναι 1. Επιλύει ως ένα βαθμό το πρόβλημα της gradient που τείνει σε μηδενισμό (vanishing gradient) οδηγώντας σε πρόβλημα εφαρμογής του αλγορίθμου εκπαίδευσης νευρωνικών backpropagation
- Εισάγει αραιότητα (sparsity) γεγονός που σημαίνει ότι μόνο ένα μέρος των νευρώνων είναι ενεργοποιημένο κάθε φορά.

Το πρόβλημα της συμμετρίας

Προσθέσαμε ένα κρυφό στρώμα με μη γραμμική συνάρτηση ενεργοποίησης για να εξασφαλίσουμε τον διαχωρισμό μη γραμμικά διαχωρίσιμων προτύπων.



Ένα κρυφό στρώμα με συνάρτηση ενεργοποίησης f τη σιγμοειδή.



Το νευρωνικό δίκτυο εκπαιδεύτηκε παρουσιάζοντας τις 64 διαφορετικές συμβολοσειρές 1208 φορές με ρυθμό εκμάθησης $\eta = 0.1$ και προσδιορίστηκαν τα βάρη που εμφανίζονται στη διπλανή εικόνα.

$$z_1 = -3.18x_1 + 6.32x_2 - 12.56x_3 + 12.56x_4 - 6.33x_5 + 3.17x_6 - 1$$

$$z_2 = 3.16x_1 - 6.34x_2 + 12.52x_3 - 12.51x_4 + 6.31x_5 - 3.17x_6 - 1$$

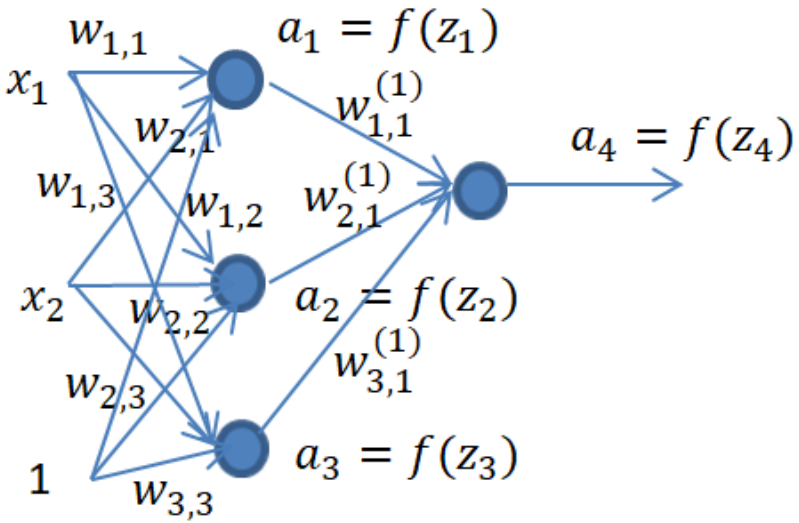
$$z_3 = -9.44a_1 - 9.44a_2 + 6.89$$

Όταν η συμβολοσειρά είναι συμμετρική τότε $x_1 = x_6,\ x_2 = x_5,\ x_3 = x_4$. Σ' αυτήν την περίπτωση $z_1 = -1, z_2 = -1$. Άρα $\alpha_1 \approx 0, \alpha_2 \approx 0$. Επομένως $z_3 = 6.89$ και $\alpha_3 \approx 1$.

Στην περίπτωση που η συμβολοσειρά είναι μη συμμετρική πρέπει να προσέξουμε ότι τα βάρη μπροστά από τα x_1, x_2, x_3 και αντίστοιχα για τα x_4, x_5, x_6 χαρακτηρίζονται από λόγο 1:2:4. Επομένως παράγεται διαφορετική τιμή για το άθροισμα του αριστερού μισού των παραγόντων στα z_1, z_2σε σχέση με το άθροισμα του δεξιού μισού των παραγόντων. Αυτό σημαίνει ότι είτε $\alpha_1 \approx 1$ είτε $\alpha_2 \approx 1$. Επομένως $z_3 \approx -2.55$ και $\alpha_3 \approx 0$.

Επίλυση προβλημάτων που δεν είναι γραμμικά διαχωρίσιμα-το πρόβλημα της XOR

Στην περίπτωση που έχουμε ένα νευρωνικό δίκτυο με έναν νευρώνα στην έξοδο δεν είναι δυνατός ο διαχωρισμός μη γραμμικώς διαχωρίσιμων δεδομένων. Απαιτείται η προσθήκη κρυφού στρώματος με μη γραμμική ενεργοποίηση στους κόμβους.



Για την εκπαίδευση στην XOR χρησιμοποιούμε ένα κρυφό στρώμα με 3 κόμβους. Και στους 4 κόμβους η συνάρτηση ενεργοποίησης είναι η σιγμοειδής. Στόχος κατά την εκπαίδευση είναι η ελαχιστοποίηση της συνάρτησης απώλειας που έχουμε επιλέξει.

$$z_1 = w_{1,1}x_1 + w_{2,1}x_2 + w_{3,1}$$
$$z_2 = w_{1,2}x_1 + w_{2,2}x_2 + w_{3,2}$$
$$z_3 = w_{1,3}x_1 + w_{2,3}x_2 + w_{3,3}$$
$$z_4 = {w_{1,1}}^{(1)}a_1 + {w_{2,1}}^{(1)}a_2 + {w_{3,1}}^{(1)}a_3$$

Μέθοδος εκπαίδευσης μέσω διάχυσης προς τα πίσω (backpropagation)

Εφαρμόζουμε τη μέθοδο κατάβασης της κλίσης και τον κανόνα της αλυσίδας για να προσδιορίσουμε τη μεταβολή στα βάρη

$$\frac{\partial \mathrm{L}}{\partial {w_{1,1}}^{(1)}} = \frac{\partial L}{\partial a_4}\frac{\partial a_4}{\partial z_4}\frac{\partial z_4}{\partial {w_{1,1}}^{(1)}} \quad (1)$$

Ως συνάρτηση ενεργοποίησης α_4 επιλέγουμε την $\mathrm{f}(z_4) = \frac{1}{1+e^{-z_4}}$.

Επομένως $\frac{\partial a_4}{\partial z_4} = f'(\mathrm{z}_4) = (1 - a_4)a_4.$

Επειδή $z_4 = {w_{1,1}}^{(1)}a_1 + {w_{2,1}}^{(1)}a_2 + {w_{3,1}}^{(1)}a_3$ προκύπτει $\frac{\partial z_4}{\partial {w_{1,1}}^{(1)}} = a_1$

Αντικαθιστώντας στην (1) παίρνουμε

$$\frac{\partial \mathrm{L}}{\partial {w_{1,1}}^{(1)}} = \frac{\partial L}{\partial a_4}(1 - a_4)a_4\alpha_1 (2)$$

Ως συνάρτηση απώλειας θεωρούμε τη συνάρτηση του τετραγωνικού σφάλματος $\frac{1}{2}\sum_j \left(a_4 - y_j\right)^2$, όπου y_j η ετικέτα του σημείου $\vec{x}_j$. Ως προς ένα σημείο η συνάρτηση απώλειας γίνεται $\frac{1}{2}(y_i - a_4)^2$.

Αντικαθιστώντας στη (2) παίρνουμε

$$\frac{\partial \mathrm{L}}{\partial {w_{1,1}}^{(1)}} = -(y_i - a_4)(1 - a_4)a_4\alpha_1$$

Προσδιορισμός των βαρών από το κρυφό στρώμα στο στρώμα εξόδου

$$\frac{\partial \mathrm{L}}{\partial {w_{1,1}}^{(1)}} = -(y_i - a_4)(1 - a_4)a_4\alpha_1 = -(y_i - a_4)f'(\mathrm{z}_4)a_1 \quad (3)$$

Αν θέσουμε $e_1 = y_i - a_4$ και $\delta_1 = e_1 f'(\mathrm{z}_4)$ η (3) γίνεται

$$\frac{\partial \mathrm{L}}{\partial {w_{1,1}}^{(1)}} = -\delta_1 \alpha_1$$

Αφού το σήμα από την είσοδο οδηγηθεί στην έξοδο και προκύψει η απόκλιση από την επιθυμητή τιμή η τιμή του βάρους θα ανανεωθεί. Η νέα τιμή του βάρους είναι

$${w_{1,1}}^{(1)} = {w_{1,1}}^{(1)} + \eta\delta_1\alpha_1,$$

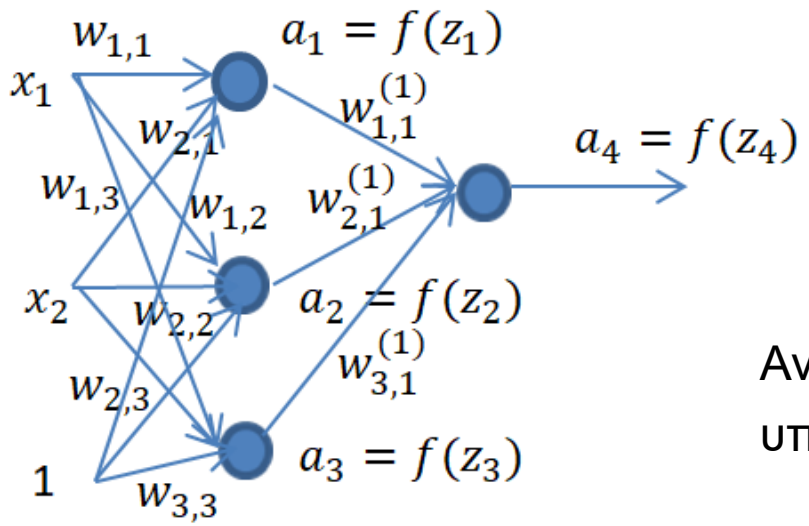
όπου η ο ρυθμός μάθησης (learning rate) που καθορίζεται από τον χρήστη. Η επιλογή του η παίζει σημαντικό ρόλο στη σύγκλιση καθώς για μικρή τιμή μπορεί η σύγκλιση να είναι αργή ενώ για πολύ μεγάλη να μην έχω σύγκλιση λόγω αναπηδήσεων στις επιφάνεια των τοπικών ελαχίστων (overshoot).

Αντίστοιχα υπολογίζονται και οι νέες τιμές των βαρών στους κλάδους από το κρυφό στρώμα προς το στρώμα εξόδου.

$${w_{2,1}}^{(1)} = {w_{2,1}}^{(1)} + \eta\delta_1\alpha_2$$
$${w_{3,1}}^{(1)} = {w_{3,1}}^{(1)} + \eta\delta_1\alpha_3$$

Για να βρούμε πως θα μεταβληθούν τα βάρη στους κλάδους από το στρώμα εισόδου στο κρυφό στρώμα υπολογίζουμε τη μέγιστη κλίση για την αύξηση της συνάρτησης απώλειας

Προσδιορισμός των βαρών στους κλάδους από το στρώμα εισόδου στο κρυφό στρώμα



$$\frac{\partial \mathrm{L}}{\partial w_{1,1}} = \frac{\partial L}{\partial a_1}\frac{\partial a_1}{\partial z_1}\frac{\partial z_1}{\partial w_{1,1}} \quad (4)$$

$$\frac{\partial L}{\partial a_1} = \frac{\partial L}{\partial a_4}\frac{\partial a_4}{\partial z_4}\frac{\partial z_4}{\partial a_1} = -\delta_1 \frac{\partial z_4}{\partial \alpha_1} \quad (5)$$

$$z_4 = {w_{1,1}}^{(1)} a_1 + {w_{2,1}}^{(1)} a_2 + {w_{3,1}}^{(1)} a_3$$

Αντικαθιστώντας την (5) στην (4) και υπολογίζοντας το $\frac{\partial z_4}{\partial \alpha_1}$ προκύπτει

$$\frac{\partial \mathrm{L}}{\partial w_{1,1}} = -\delta_1 {w_{1,1}}^{(1)} f'(z_1)\frac{\partial z_1}{\partial w_{1,1}}$$

Επειδή $z_1 = w_{1,1}x_1 + w_{2,1}x_2 + w_{3,1}$, προκύπτει

$$\frac{\partial \mathrm{L}}{\partial w_{1,1}} = -\delta_1 {w_{1,1}}^{(1)} f'(z_1)x_1 = -\delta_1 {w_{1,1}}^{(1)}(1-a_1)a_1 x_1$$

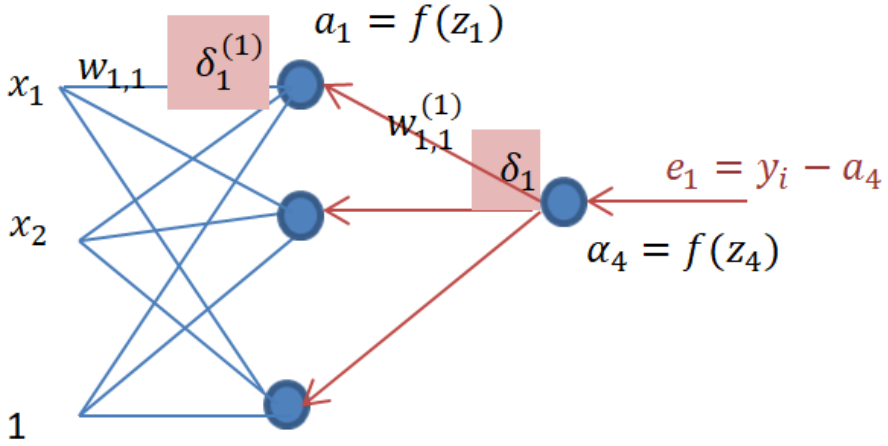
Κάποιες από τις νέες τιμές των βαρών είναι

$$w_{1,1} = w_{1,1} + \eta\delta_1 {w_{1,1}}^{(1)} f'(z_1)x_1$$
$$w_{1,2} = w_{1,2} + \eta\delta_1 {w_{2,1}}^{(1)} f'(z_2)x_1$$
$$w_{2,1} = w_{2,1} + \eta\delta_1 {w_{1,1}}^{(1)} f'(z_1)x_2$$
$$w_{3,3} = w_{3,3} + \eta\delta_1 {w_{3,1}}^{(1)} f'(z_3)$$

Όδευση του σφάλματος προς τα πίσω (backpropagation)

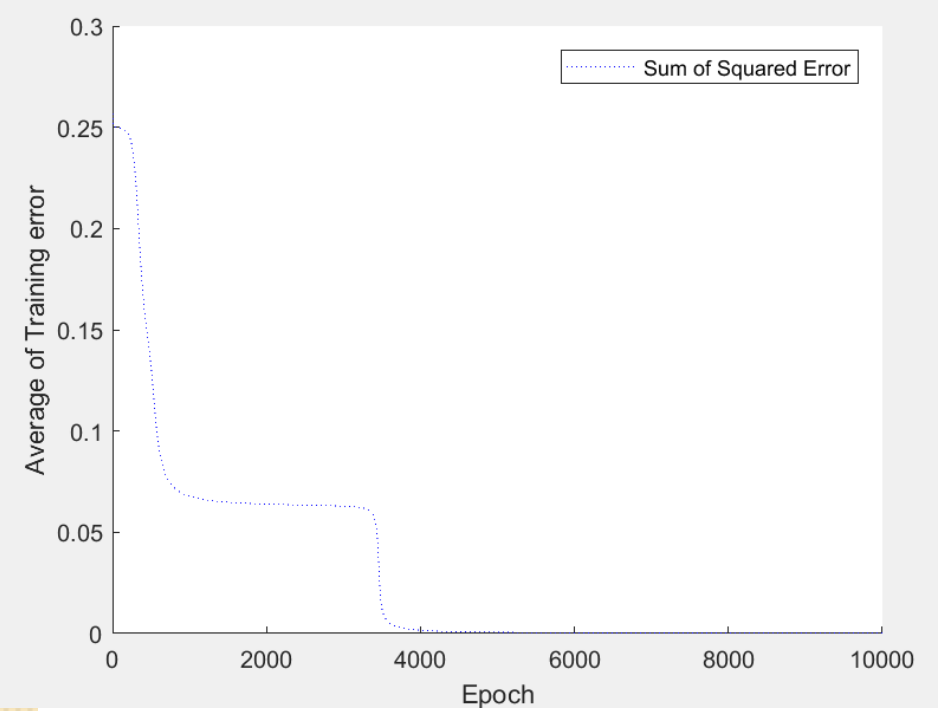
Όλη αυτή τη διαδικασία που την είδαμε μέσα από το πρίσμα της κατάβασης κλίσης (gradient descent) μπορούμε γραφικά να την αντιληφθούμε ως την όδευση του σφάλματος προς τα πίσω.

$$w_{1,1}{}^{(1)} = w_{1,1}{}^{(1)} + \eta\delta_1\alpha_1$$
$$\delta_1 = e_1 f'(z_4)$$



$$\delta_1{}^{(1)} = \delta_1 w_{1,1}{}^{(1)} f'(z_1)$$
$$w_{1,1} = w_{1,1} + \eta\delta_1{}^{(1)} x_1$$

Εκπαίδευση του νευρωνικού δικτύου που περιγράψαμε με δεδομένα της XOR και συνάρτηση απώλειας το τετραγωνικό σφάλμα



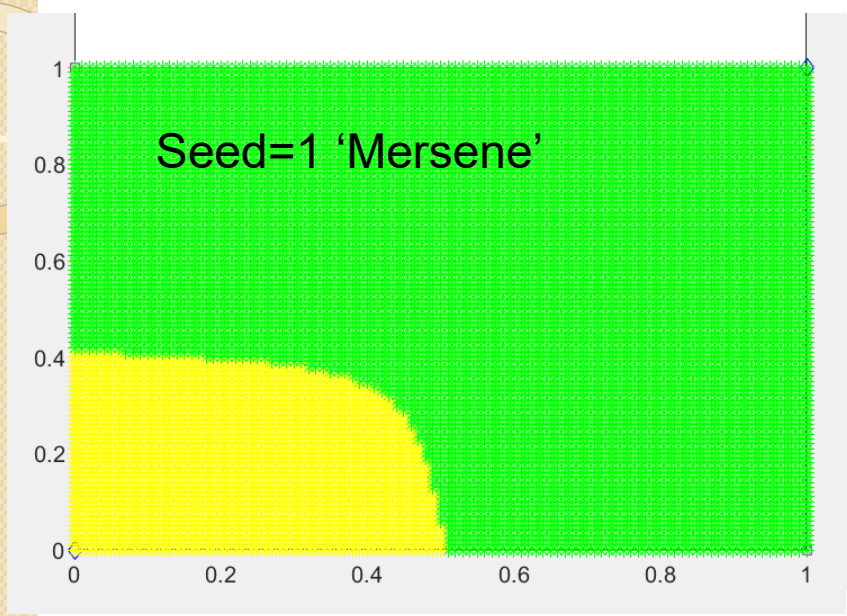
Τροφοδοτούμε το νευρωνικό δίκτυο με τα δεδομένα της XOR και παρακολουθούμε πως εξελίσσεται το μέσο τετραγωνικό σφάλμα

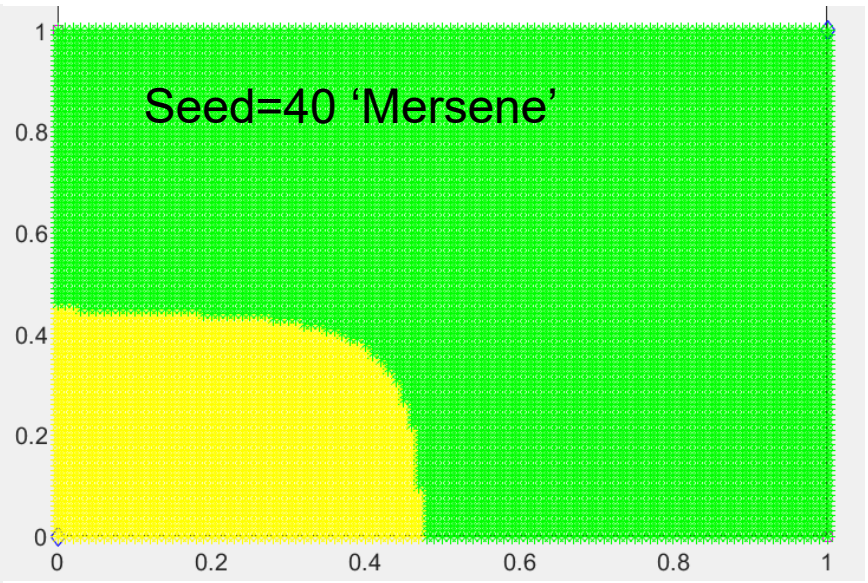
$$\frac{1}{\mathrm{N}}\sum(y_i - a_4)^2$$

κατά τη διάρκεια 10000 εποχών. Ο ρυθμός εκμάθησης που επιλέχθηκε ήταν η=0,9.

Τα σημεία με τα οποία τροφοδοτούμε το νευρωνικό είναι $\mathrm{X} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ και οι ετικέτες $\Upsilon = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$. Η τρίτη συνιστώσα του X αντιστοιχεί στο bias. Το αποτέλεσμα μετά την εκπαίδευση είναι $\hat{\Upsilon} = \begin{bmatrix} 0.0091 \\ 0.9860 \\ 0.9888 \\ 0.0135 \end{bmatrix}$. Παρατηρούμε πόσο κοντά είναι το διάνυσμα $\hat{\Upsilon}$ στο διάνυσμα των ετικετών Y.

Αποτελέσματα μετά την εκπαίδευση του νευρωνικού με διαφορετική αρχικοποίηση.





Μετά από εκπαίδευση 1000 εποχών με διαφορετική αρχικοποίηση (αλγόριθμος 'Mersene' με διαφορετικό seed) των βαρών και του bias.

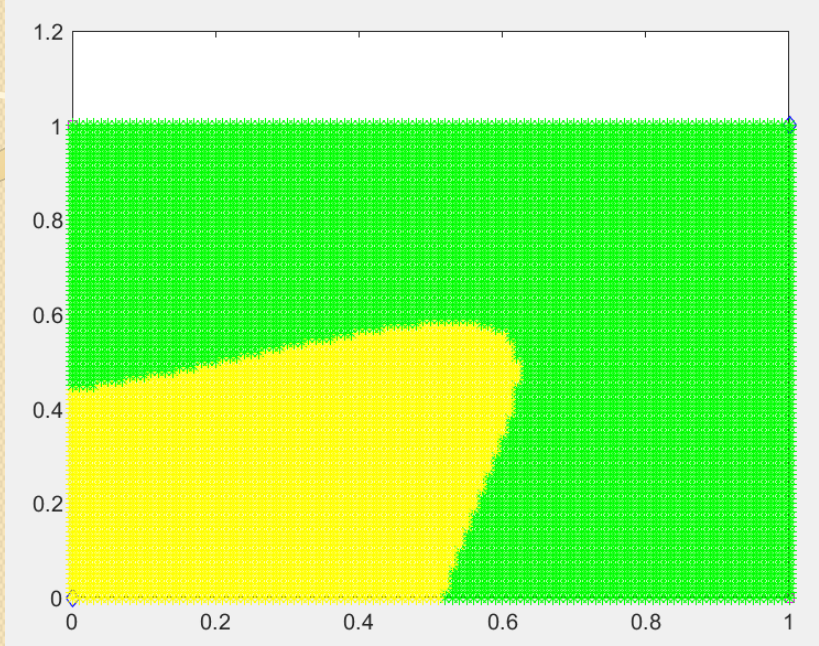
Το διάνυσμα των ετικετών για τη 2η εικόνα που πρόβλεψε το νευρωνικό είναι

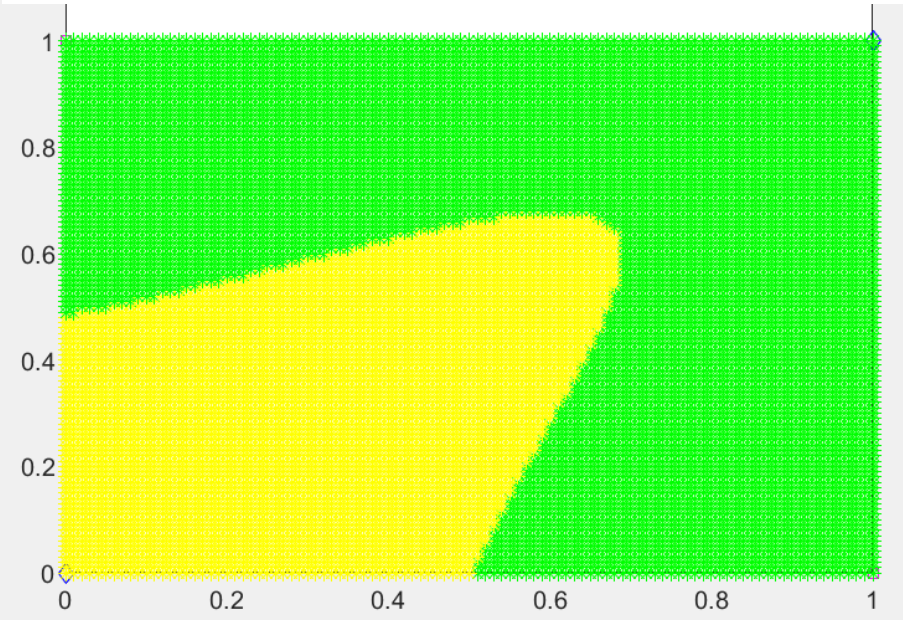
$\hat{\Upsilon} = \begin{bmatrix} 0.0462 \\ 0.9281 \\ 0.9312 \\ 0.5118 \end{bmatrix}$. Είναι εμφανές ότι το σημείο (1,1) ταξινομείται σε λάθος κατηγορία.

Ακρίβεια στην ταξινόμηση $\frac{1}{\pi\lambda\eta\theta o\varsigma\ \sigma\eta\mu\varepsilon\iota\omega\nu} * \alpha\rho\iota\theta\mu o\varsigma\ \sigma\omega\sigma\tau\omega\nu\ \tau\alpha\xi\iota\nu o\mu\eta\sigma\varepsilon\omega\nu =$

$$\frac{1}{4} * 3 = 75\%$$

Αποτελέσματα μετά την εκπαίδευση του νευρωνικού με διαφορετική αρχικοποίηση.



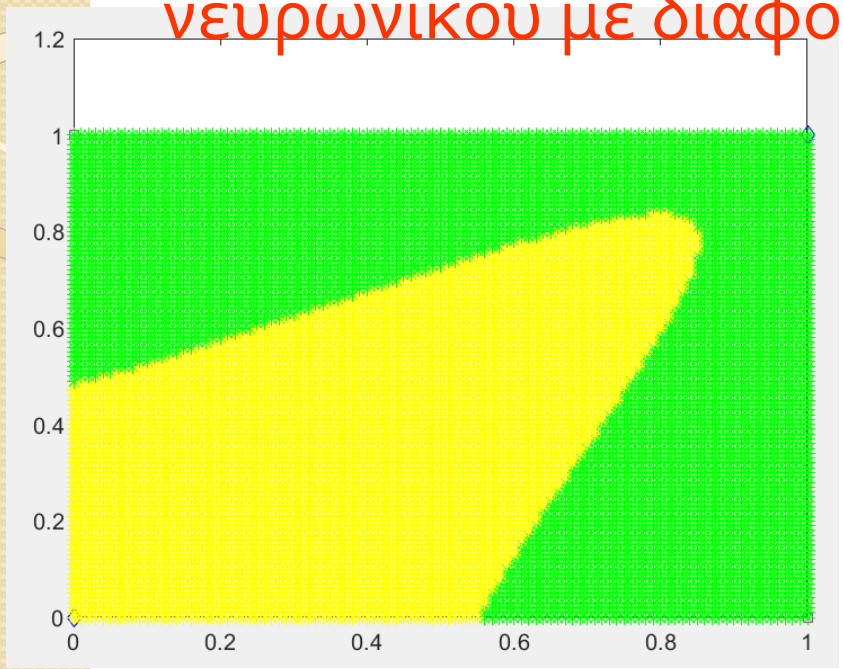


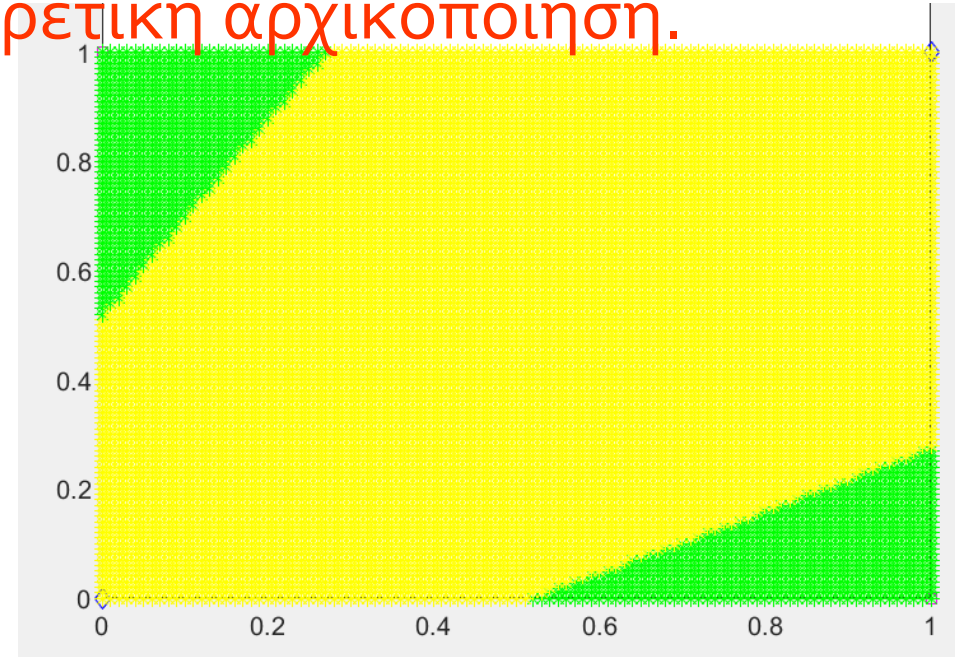
Μετά από εκπαίδευση 4000 εποχών με διαφορετική αρχικοποίηση των βαρών και του bias.

Το διάνυσμα των ετικετών για τη 2η εικόνα που πρόβλεψε το νευρωνικό είναι

$\hat{\Upsilon} = \begin{bmatrix} 0.0134 \\ 0.9781 \\ 0.9785 \\ 0.5013 \end{bmatrix}$. Είναι εμφανές ότι το σημείο (1,1) ταξινομείται σε λάθος κατηγορία.

Αποτελέσματα μετά την εκπαίδευση του νευρωνικού με διαφορετική αρχικοποίηση.



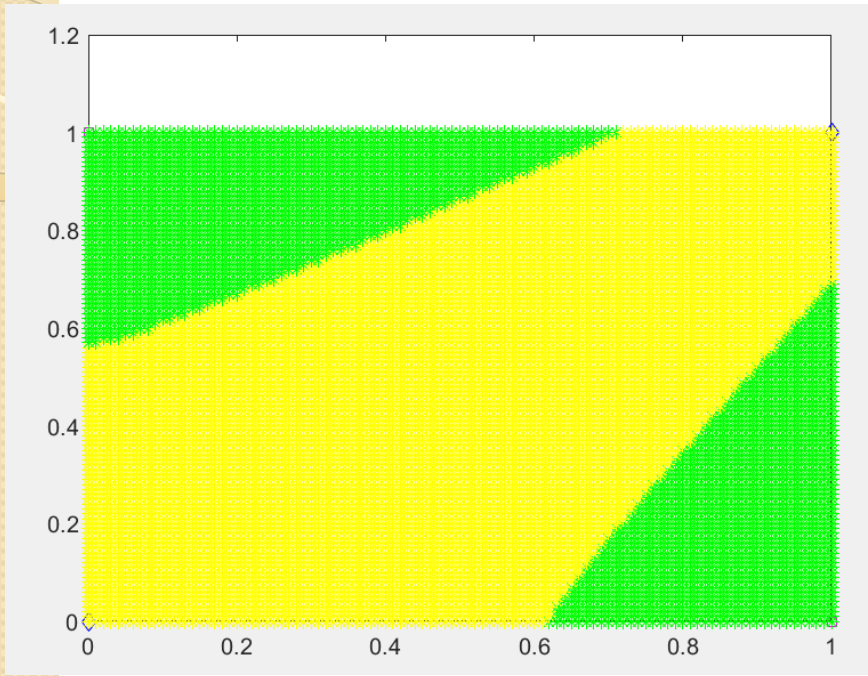


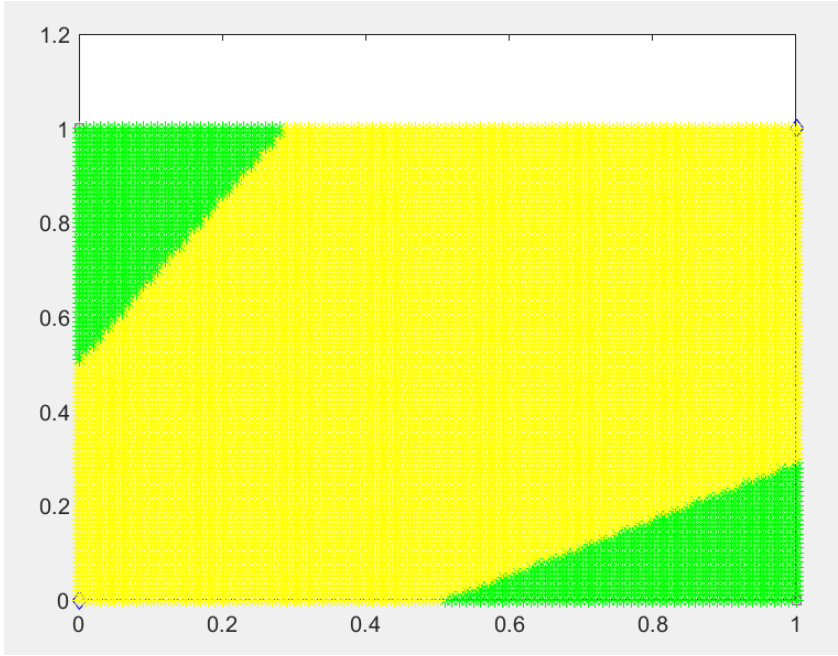
Μετά από εκπαίδευση 8000 εποχών με διαφορετική αρχικοποίηση των βαρών και του bias.

Το διάνυσμα των ετικετών για τη 2η εικόνα που πρόβλεψε το νευρωνικό είναι

$\hat{\Upsilon} = \begin{bmatrix} 0.0379 \\ 0.9612 \\ 0.9607 \\ 0.0391 \end{bmatrix}$. Το σημείο (1,1) ταξινομείται λάθος με την πρώτη αρχικοποίηση και σωστά με τη δεύτερη μετά από 8000 εποχές.

Αποτελέσματα μετά την εκπαίδευση του νευρωνικού με διαφορετική αρχικοποίηση.





Μετά από εκπαίδευση 10000 εποχών με διαφορετική αρχικοποίηση των βαρών και του bias.

Το διάνυσμα των ετικετών που πρόβλεψε το νευρωνικό για την 1η εικόνα είναι

$\hat{Y} = \begin{bmatrix} 0.0065 \\ 0.9847 \\ 0.9843 \\ 0.4993 \end{bmatrix}$. Όλα τα σημεία ταξινομούνται σωστά και με τις 2 αρχικοποιήσεις μετά από 10000 εποχές.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Duda, R.O. and Hart, P.E. (1973) Pattern Classification and Scene Analysis. Wiley, New York.

Vapnik, V. (1998) Statistical Learning Theory. John Wiley & Sons, Chichester.

Cristianini, N. and Shawe-Taylor, J. (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge.

Duda, P. E. Hart and D. G. Stork, “Pattern Classification,” 2nd Edition, John Wiley and Sons, Inc., New York, 2001

Schölkopf, B., & Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.

Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press.

Zhang, A., Lipton, Z. C., Li M., Smola, A. J. (2023) Dive Into Deep Learning.