Αναγνώριση Προτύπων

Ταξινόμηση

SVMs / Ensembles

Ανδρέας Λ. Συμεωνίδης

Αν. Καθηγητής

Τμήμα Ηλεκτρολόγων Μηχ/κών
&
Μηχ/κών Υπολογιστών, Α.Π.Θ.

Email: asymeon@eng.auth.gr

issel
Intelligent Systems
& Software Engineering
Labgroup

ARISTOTLE
UNIVERSITY OF
THESSALONIKI

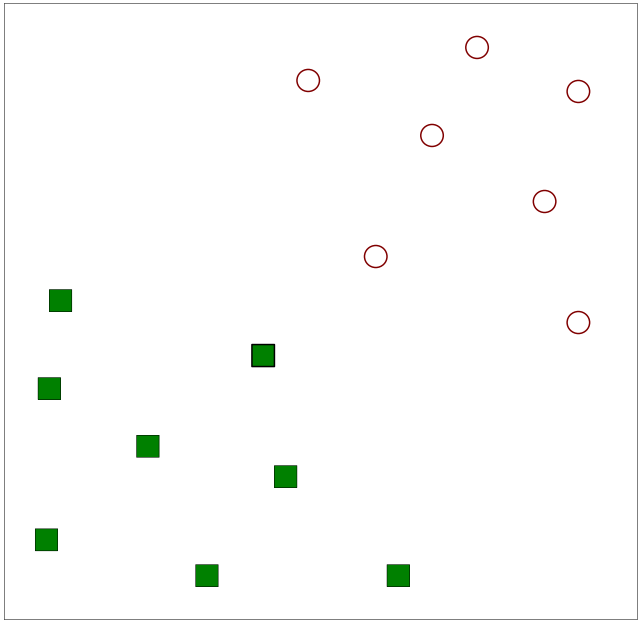
Διάρθρωση διάλεξης

- Γραμμικά Support Vector Machines
- Μη γραμμικά Support Vector Machines
- Τεχνικές σύνθεσης μοντέλων
  - Bootstrappping
  - Bagging
  - Boosting

Support Vector Machines

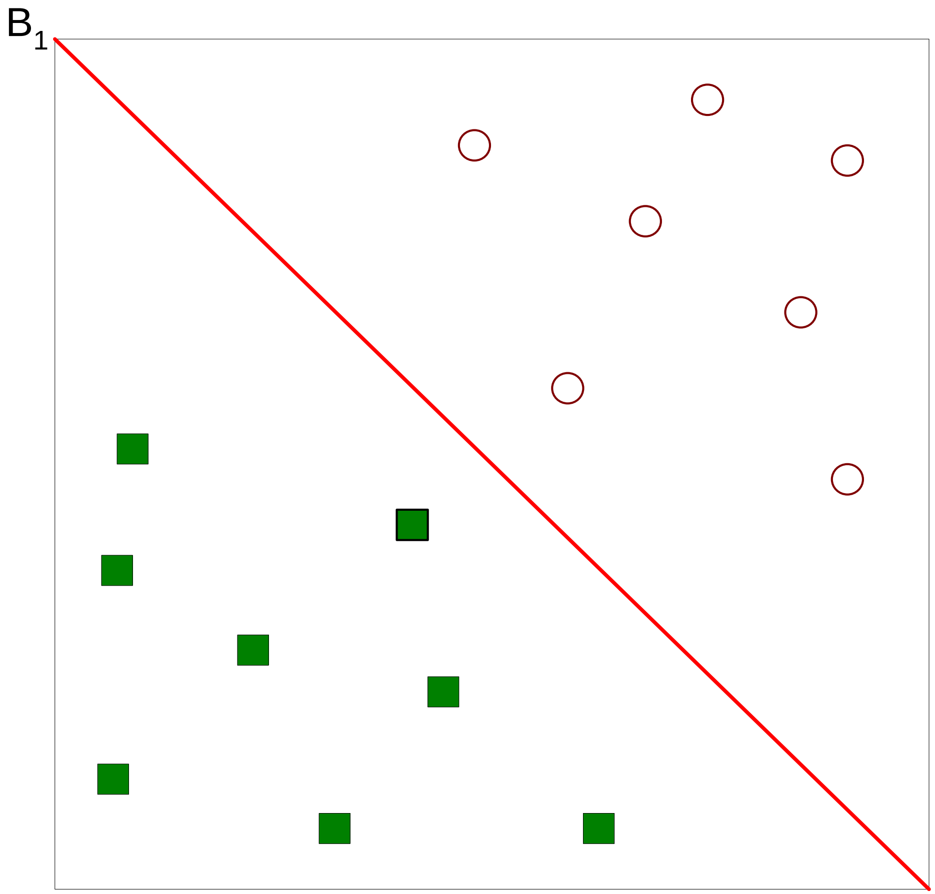
- Πρακτικός/Μαθηματικός ορισμός
- Στόχος βελτιστοποίησης
- Γραμμικά/μη-γραμμικά Support Vector Machines
- Kernels

Support Vector Machines



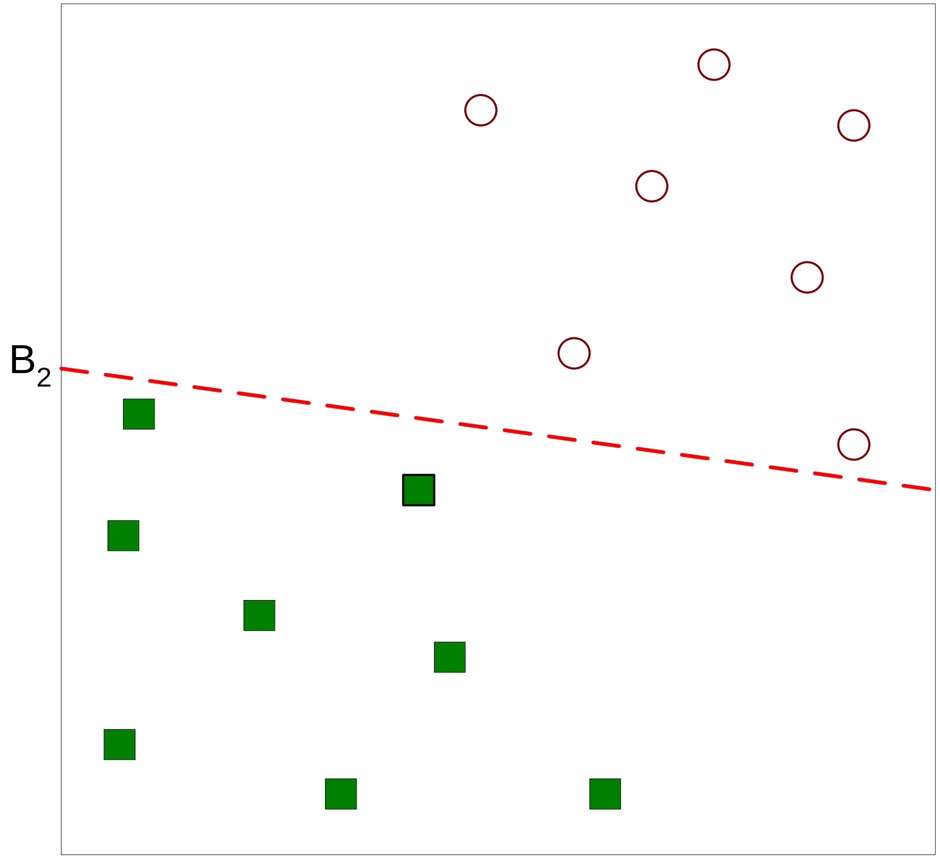
- Εύρεση ενός γραμμικού υπερεπίπεδου (σύνορο απόφασης) το οποίο θα διαχωρίσει τα δεδομένα

Support Vector Machines



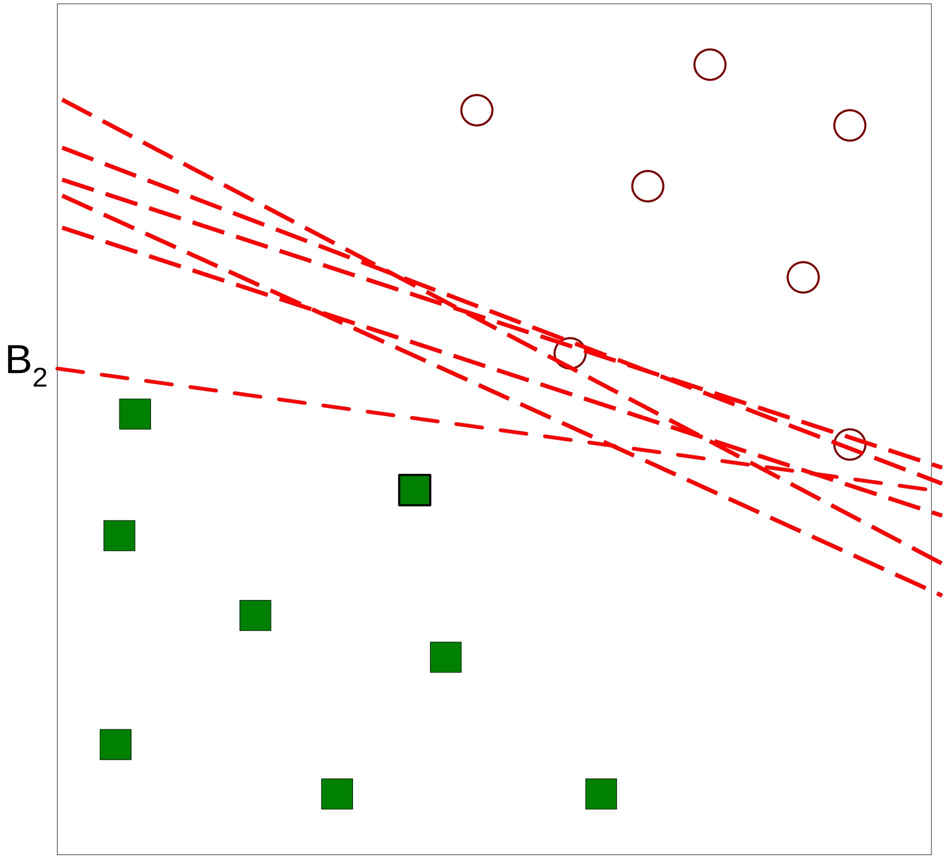
- Μια πιθανή λύση

Support Vector Machines



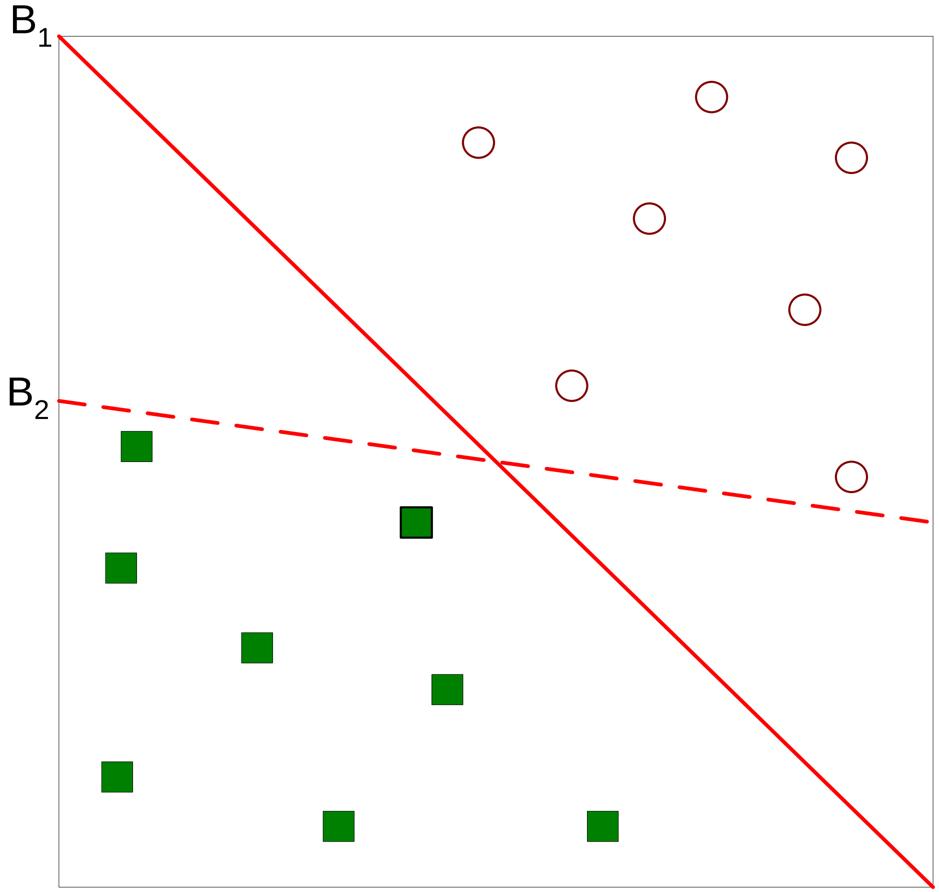
- Μια άλλη πιθανή λύση

Support Vector Machines



- Άλλες πιθανές λύσεις

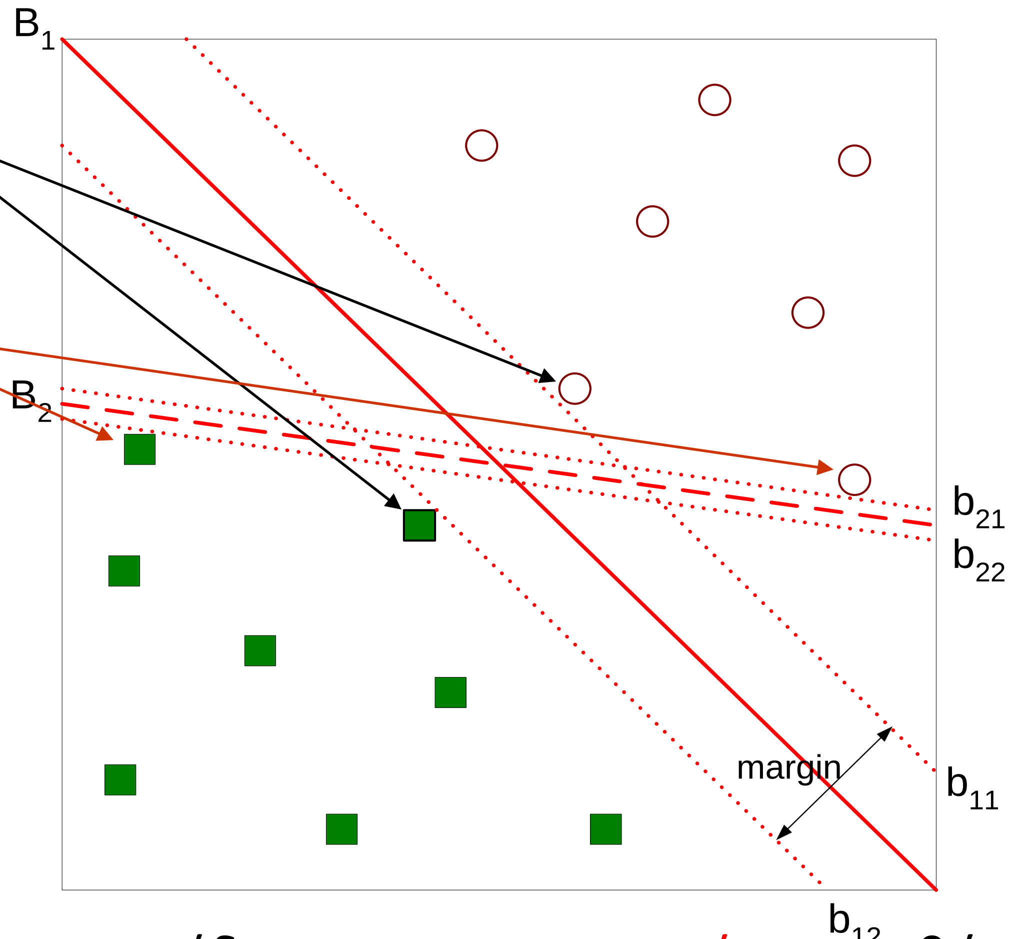
Support Vector Machines



- Ποια είναι καλύτερη; Η B1 ή η B2;
- Πώς ορίζεται το «καλύτερη»;

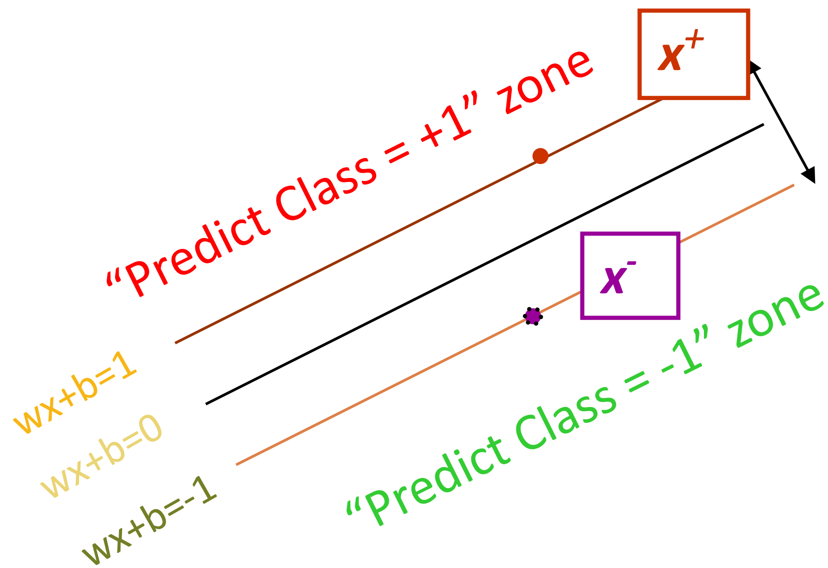
Support Vector Machines

Support Vectors είναι εκείνα τα σημεία που «βρίσκουν» το margin



- Εύρεση του υπερεπιπέδου που μεγιστοποιεί το περιθώριο (margin) => η B1 είναι καλύτερη από τη B2

Πιο συγκεκριμένα

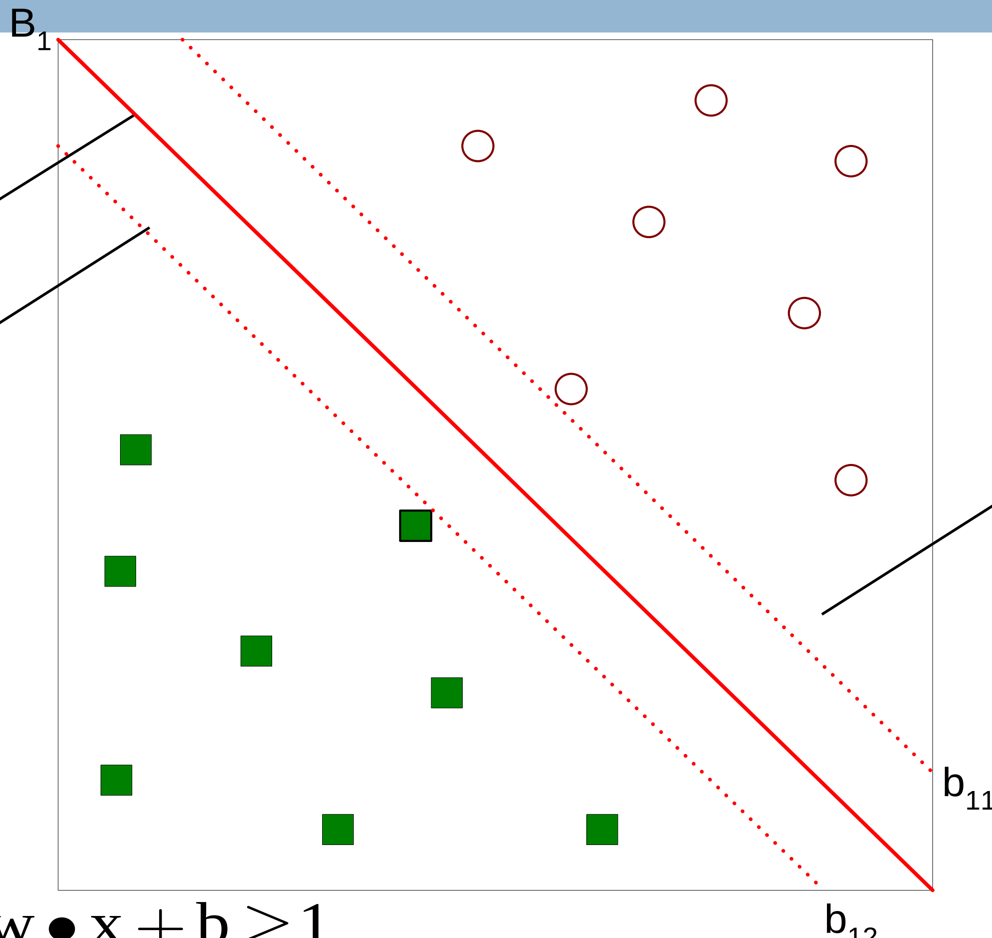


M=Margin Width

Μαθηματικά ορισμένα

$$w \bullet x + b = 0$$

$$w \bullet x + b = -1$$



$$w \bullet x + b = +1$$

$$f(x) = \begin{cases} 1 & \text{if } \mathrm{w} \bullet \mathrm{x} + \mathrm{b} \geq 1 \\ -1 & \text{if } \mathrm{w} \bullet \mathrm{x} + \mathrm{b} \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\| w \|^2}$$

Στόχος βελτιστοποίησης

- Θέλουμε να μεγιστοποιήσουμε: $\text{Margin} = \dfrac{2}{\| w \|^2}$
  - Ισοδύναμο του να ελαχιστοποιήσουμε: $L(w) = \dfrac{\| w \|^2}{2}$
  - Αλλά δεδομένων των παρακάτω περιορισμών:

$$f(x_i) = \begin{cases} 1 & \text{if } \mathrm{w} \bullet \mathrm{x_i} + \mathrm{b} \geq 1 \\ -1 & \text{if } \mathrm{w} \bullet \mathrm{x_i} + \mathrm{b} \leq -1 \end{cases}$$

  - Αυτό είναι ένα πρόβλημα βελτιστοποίησης υπό περιορισμούς
  - Επίλυση με αριθμητικές μεθόδους (π.χ., τετραγωνικός προγραμματισμός)

Το πρόβλημα βελτιστοποίησης

- Η λύση έχει τη μορφή:

$$\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i} \qquad b = y_k - \mathbf{w}^T\mathbf{x_k} \text{ for any } \mathbf{x_k} \text{ such that } \alpha_k \neq 0$$

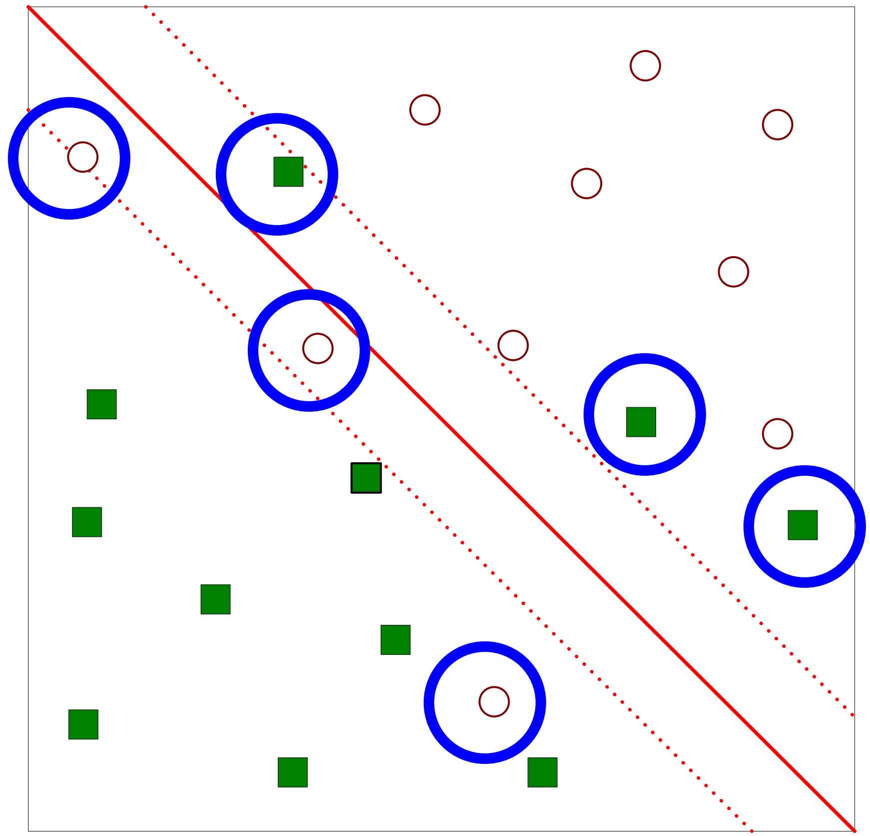
- Κάθε μη-μηδενικό α_i συνεπάγεται ότι το αντίστοιχο $\mathbf{x_i}$ είναι support vector.
- Η συνάρτηση ταξινόμησης έχει τη μορφή:

$$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^T\mathbf{x} + b$$

- Βασίζεται στο εσωτερικό γινόμενο ανάμεσα στο σημείο ελέγχου $\mathbf{x}$ και τα support vectors $\mathbf{x_i}$.
- Επίλυση του προβλήματος βελτιστοποίησης εμπλέκει τον υπολογισμό όλων των εσωτερικών γινομένων $\mathbf{x_i}^T\mathbf{x_j}$ ανάμεσα σε όλα τα σημεία εκπαίδευσης.

Μη γραμμικό πρόβλημα

- Τι συμβαίνει όταν το πρόβλημα δε λύνεται γραμμικά;



Μεταβλητές χαλαρότητας

- Μη γραμμική λύση εφικτή
- Εισαγωγή μεταβλητών χαλαρότητας
  - Ανάγκη ελαχιστοποίησης: $$L(w)=\frac{\|w\|^2}{2}+C\left(\sum_{i=1}^{N}\xi_i^k\right)$$
  - Δεδομένων των παρακάτω περιορισμών:

$$f(x_i)=\begin{cases}1 & \text{if } \mathrm{w}\bullet\mathrm{x_i}+\mathrm{b}\geq 1\text{-}\xi_\mathrm{i}\\ -1 & \text{if } \mathrm{w}\bullet\mathrm{x_i}+\mathrm{b}\leq -1+\xi_\mathrm{i}\end{cases}$$

Αυστηρό vs Χαλαρό περιθώριο

- Το αυστηρό περιθώριο:

Find **w** and *b* such that

$\mathbf{f}(\mathbf{w}) = \frac{1}{2}\,\mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i\,(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

- Το χαλαρό περιθώριο:

Find **w** and *b* such that

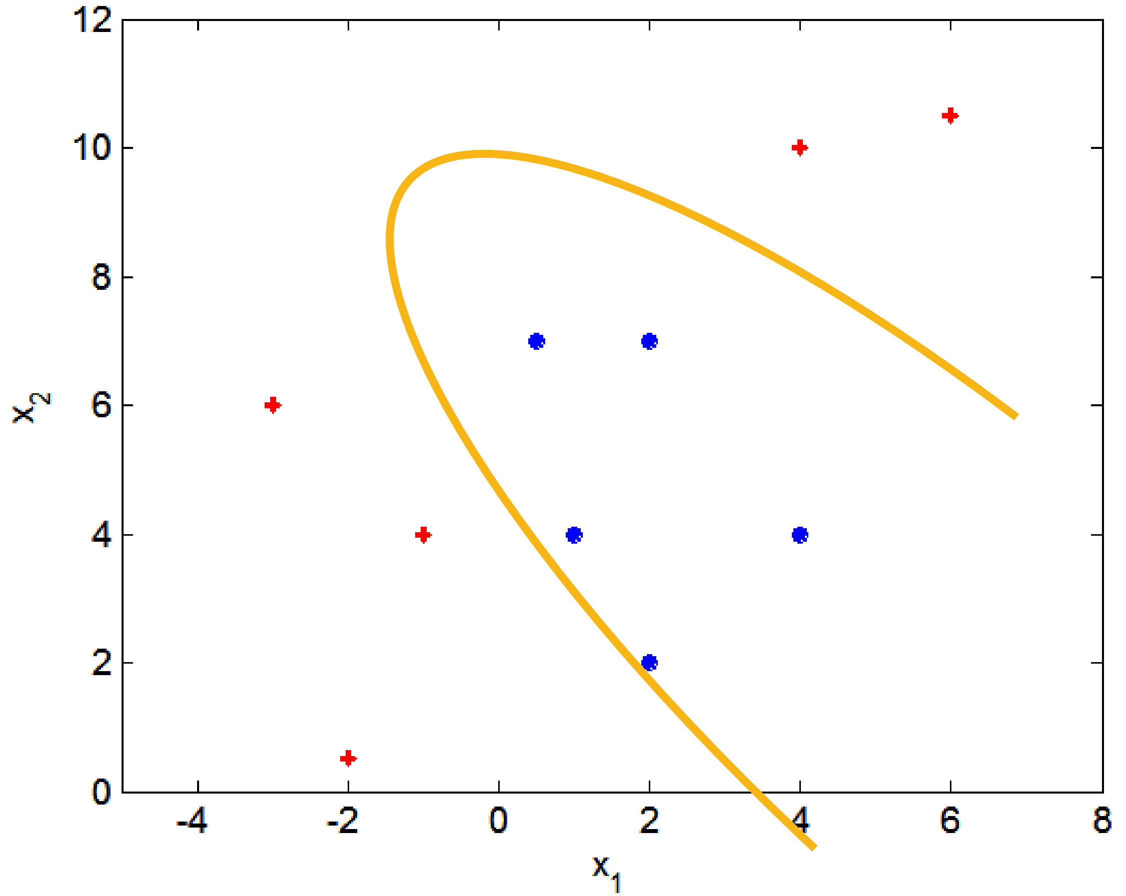
$\mathbf{f}(\mathbf{w}) = \frac{1}{2}\,\mathbf{w}^T\mathbf{w} + C\Sigma\xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i\,(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

- Η παράμετρος *C* μπορεί να χρησιμοποιηθεί για τον έλεγχο υπερεκπαίδευσης (overfitting).

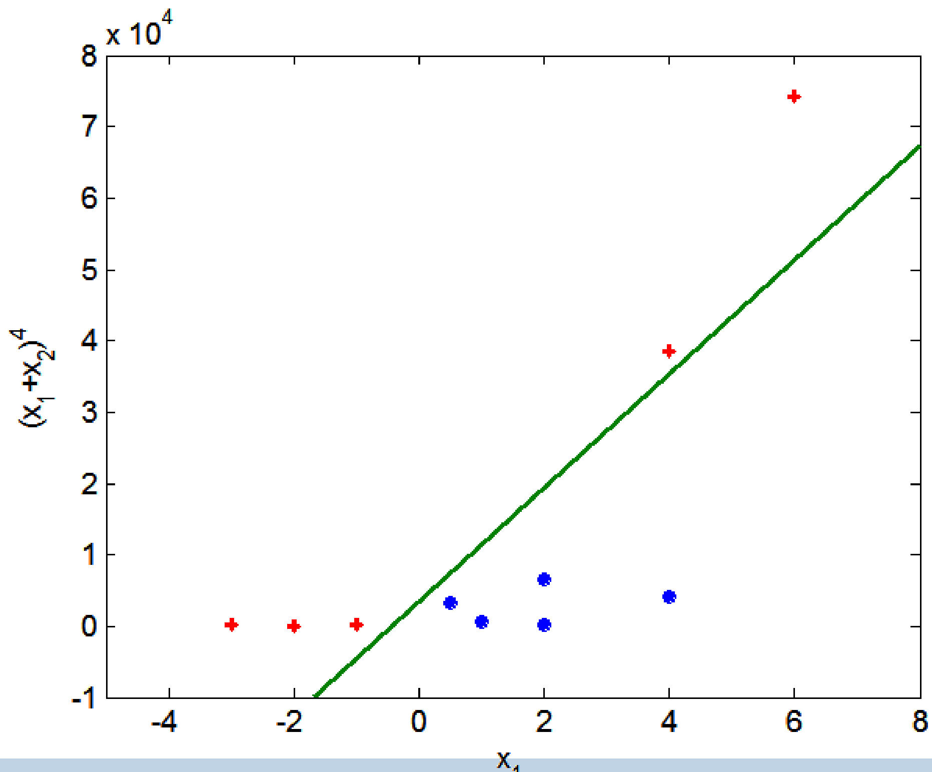
Μη γραμμικά Support Vector Machines

- Τι συμβαίνει αν ο χώρος αποφάσεων δεν είναι γραμμικός;



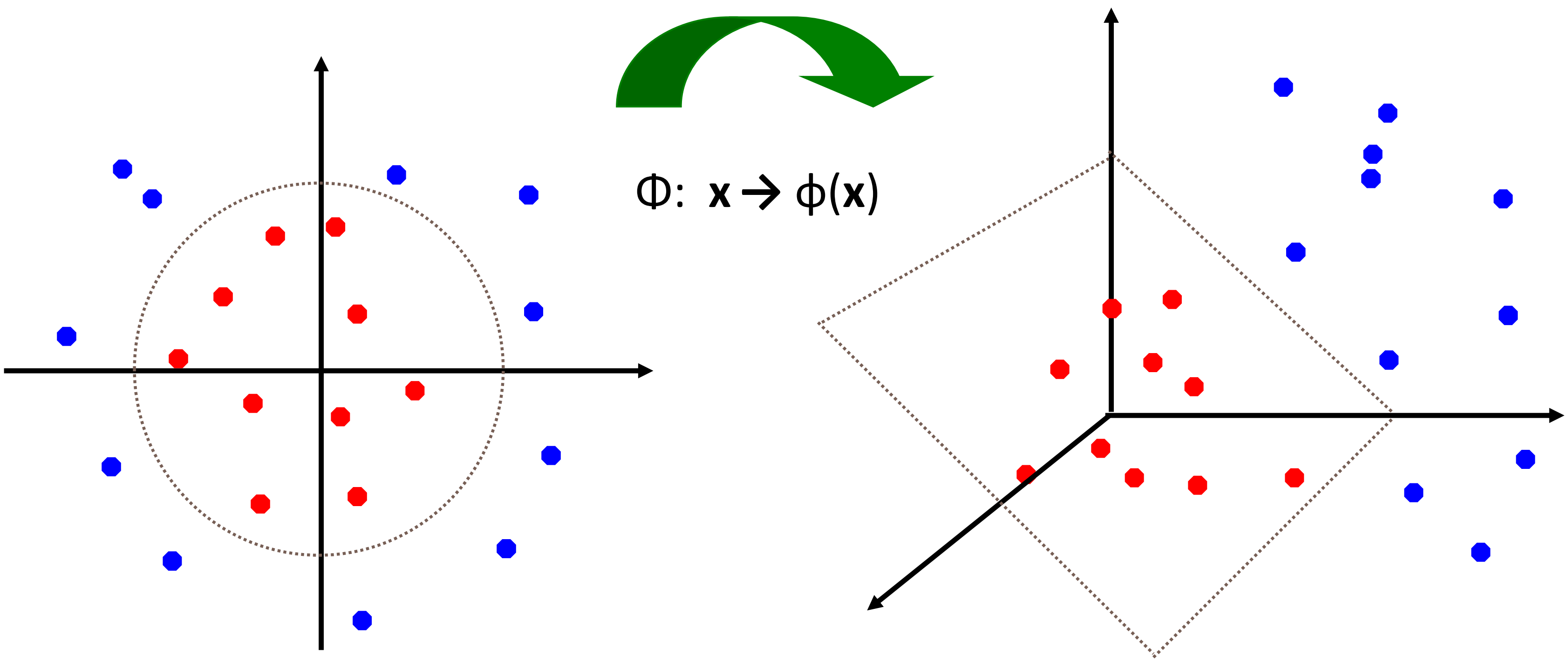
Μη γραμμικά Support Vector Machines

- Μετασχηματισμός των δεδομένων σε δεδομένα υψηλότερης διάστασης
- Χρήση Kernels



Μη-γραμμικά SVMs: Διάσταση του σετ εκπαίδευσης

- Γενική ιδέα: αντιστοίχιση του σετ εκπαίδευσης σε χώρο ανώτερης διάστασης, ώστε το σετ εκπαίδευσης να είναι διαχωρίσιμο:



Η χρήση “Kernels”

- Ο γραμμικός ταξινομητής βασίζεται στο εσωτερικό γινόμενο των διανυσμάτων $K(x_i,x_j)=x_i^T x_j$
- Εάν κάθε σημείο αντιστοιχιστεί σε κάποιο υψηλότερης διάστασης χώρο μέσω ενός μετασχηματισμού Φ: $x \rightarrow \phi(x)$, το γινόμενο γίνεται:

$$K(x_i,x_j)= \phi(x_i)^T \phi(x_j)$$

- Μια συνάρτηση *kernel* είναι μια τέτοια.
- Παράδειγμα:

2-διάστατα διανύσματα $x=[x_1 \;\; x_2]$; Έστω $K(x_i,x_j)=(1 + x_i^T x_j)^2$,

Πρέπει να δείξουμε ότι $K(x_i,x_j)= \phi(x_i)^T \phi(x_j)$

$$K(x_i,x_j)=(1 + x_i^T x_j)^2 = 1+ x_{i1}^2 x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2} + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$$

$$= [1 \;\; x_{i1}^2 \;\; \sqrt{2}\, x_{i1}x_{i2} \;\; x_{i2}^2 \;\; \sqrt{2}x_{i1} \;\; \sqrt{2}x_{i2}]^T [1 \;\; x_{j1}^2 \;\; \sqrt{2}\, x_{j1}x_{j2} \;\; x_{j2}^2 \;\; \sqrt{2}x_{j1} \;\; \sqrt{2}x_{j2}]$$

$$= \phi(x_i)^T \phi(x_j), \quad \text{όπου } \phi(x) = [1 \;\; x_1^2 \;\; \sqrt{2}\, x_1x_2 \;\; x_2^2 \;\; \sqrt{2}x_1 \;\; \sqrt{2}x_2]$$

Ποιες συναρτήσεις είναι Kernels;

- Για ορισμένες συναρτήσεις $K(x_i,x_j)$ ο έλεγχος $K(x_i,x_j)= \phi(x_i)^T\phi(x_j)$ είναι δύσκολος.
- Το θεώρημα του Mercer:
  - Κάθε ημι-θετική συμμετρική συνάρτηση είναι kernel
  - Οι ημι-θετικές συμμετρικές συναρτήσεις αντιστοιχούν σε έναν ημι-θετικό συμμετρικό Gram πίνακα:
- K =

$K(\mathbf{x}_1,\mathbf{x}_1)$	$K(\mathbf{x}_1,\mathbf{x}_2)$	$K(\mathbf{x}_1,\mathbf{x}_3)$	…	$K(\mathbf{x}_1,\mathbf{x}_N)$
$K(\mathbf{x}_2,\mathbf{x}_1)$	$K(\mathbf{x}_2,\mathbf{x}_2)$	$K(\mathbf{x}_2,\mathbf{x}_3)$		$K(\mathbf{x}_2,\mathbf{x}_N)$
…	…	…	…	…
$K(\mathbf{x}_N,\mathbf{x}_1)$	$K(\mathbf{x}_N,\mathbf{x}_2)$	$K(\mathbf{x}_N,\mathbf{x}_3)$	…	$K(\mathbf{x}_N,\mathbf{x}_N)$

Παραδείγματα από συναρτήσεις Kernel

- Γραμμικός: $K(\mathbf{x_i},\mathbf{x_j}) = \mathbf{x_i}^T\mathbf{x_j}$
- Πολυωνυμικός τάξης p: $K(\mathbf{x_i},\mathbf{x_j}) = (1+ \mathbf{x_i}^T\mathbf{x_j})^p$
- Γκαουσιανός (radial-basis function network):

$$K(\mathbf{x_i},\mathbf{x_j}) = \exp\left(-\frac{\left\|\mathbf{x_i}-\mathbf{x_j}\right\|^2}{2\sigma^2}\right)$$

- Σιγμοειδής: $K(\mathbf{x_i},\mathbf{x_j}) = \tanh(\beta_0\mathbf{x_i}^T\mathbf{x_j} + \beta_1)$

Μη γραμμικά SVMs - Μαθηματικά

- Ορισμός του διπλού προβλήματος:

Βρες $\alpha_1 \ldots \alpha_N$ τέτοια ώστε:

$Q(\alpha) = \Sigma\alpha_i - ½\Sigma\Sigma\alpha_i\alpha_j y_i y_j K(x_i, x_j)$ μεγιστοποιείται και

(1) $\Sigma\alpha_i y_i = 0$

(2) $\alpha_i \geq 0$ για όλα τα α_i

- Η λύση είναι: $f(x) = \Sigma\alpha_i y_i K(x_i, x_j) + b$

- Οι τεχνικές βελτιστοποίησης για την εύρεση των α_i's παραμένουν οι ίδιες!

Ιδιότητες των SVMs

- Ευελιξία στην επιλογή της συνάρτησης ομοιότητας
- Στο διαχωρισμό μεγάλων σετ δεδομένων η λύση είναι «αραιή»
  - Μόνο τα support vectors χρησιμοποιούνται για να ορίσουν το υπερ-επίπεδο διαχωρισμού
- Δυνατότητα διαχείρισης δεδομένων με πολλών διαστάσεις
  - Η πολυπλοκότητα δεν εξαρτάται από τη διάσταση του χώρου των δεδομένων
- Η υπερεκπαίδευση μπορεί να ελεγχθεί με την εφαρμογή της προσέγγισης χαλαρού ορίου
- Βολική μαθηματική ιδιότητα: ένα απλό κυρτό πρόβλημα βελτιστοποίησης το οποίο εγγυάται σύγκλιση σε μοναδική, ολική λύση

Εφαρμογές των SVMs

- Επιτυχημένη εφαρμογή τους σε προβλήματα

  - ταξινόμησης κειμένου (και υπερκειμένου)

  - ταξινόμησης εικόνων

  - βιοπληροφορικής (ταξινόμηση πρωτεϊνών, ταξινόμηση ασθενειών)

  - αναγνώρισης χειρόγραφου κειμένου

Συχνά θέματα

- Επιλογή του kernel
  - Gaussian ή πολυωνυμικός kernel συνήθως
  - Εάν το αποτέλεσμα δεν είναι καλό, χρησιμοποιούνται πιο πολύπλοκοι kernels
- Επιλογή των παραμέτρων του kernel
  - Το σ στον Gaussian kernel
  - Το σ είναι η απόσταση ανάμεσα στα κοντινότερα σημεία με διαφορετική ταξινόμηση
  - Εάν δεν υπάρχουν αξιόπιστα κριτήρια, οι εφαρμογές βασίζονται σε σετ επικύρωσης ή cross-validation.
- Κριτήριο βελτιστοποίησης – Αυστηρό vs Χαλαρό περιθώριο
  - Εφαρμογή μιας εκτεταμένης σειράς πειραμάτων

Μέθοδοι σύνθεσης μοντέλων (Ensemble Methods)

- Ομάδες ταξινομητών
- Τεχνικές σύνθεσης μοντέλων
  - Bootstrappping
  - Bagging
  - Boosting

Ιστορική αναδρομή

Επαναδειγματοληψία για τον υπολογισμό στατιστικών μετρικών

Bootstrapping

Bagging

Boosting (Schapire 1989)

Adaboost (Schapire 1995)

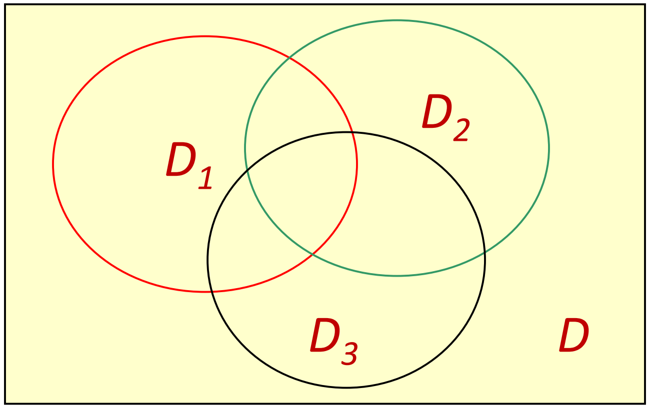
Επαναδειγματοληψία για τη σχεδίαση ταξινομητών

Bootstrap εκτίμηση

- Επανηλειμμένα επέλεξε *n* δείγματα από το σετ δεδομένων
- Για κάθε σετ δειγμάτων, υπολόγισε τη στατιστική μετρική που σε ενδιαφέρει
- Η bootstrap εκτίμηση είναι η μέση τιμή των επιμέρους εκτιμήσεων
- Χρησιμοποιείται για τον υπολογισμό μιας παραμέτρου και της διακύμανσής της

Bagging – Aggregate Bootstrapping

- Για $i = 1 .. M$
  - Επέλεξε $n^*<n$ δείγματα από το σετ δεδομένων με αντικατάσταση
  - Δημιούργησε τον ταξινομητή C_i
- Ο τελικός ταξινομητής είναι μια ψηφοφορία των $C_1 .. C_M$
- Αυξάνει τη σταθερότητα του ταξινομητή/μειώνει τη διακύμανση



Bagging

- Δειγματοληψία με αντικατάσταση

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Δημιουργία ενός ταξινομητή σε κάθε δείγμα εκκίνησης
- Κάθε σημείο έχει πιθανότητα να επιλεγεί $1/n$
- Για ένα σημείο που δεν έχει επιλεγεί μετά από n φορές, η πιθανότητα είναι $(1 – 1/n)^n$ (όταν το n είναι μεγάλο, προσεγγίζει το $1/e$)
- Για ένα σημείο που επιλέγεται μετά από n φορές, η πιθανότητα είναι $1 – 1/e = 0.632$
- Ένα δείγμα εκκίνησης (bootstrap sample) περιέχει 63% του αρχικού σετ δεδομένων

Boosting

- Μια επαναληπτική διαδικασία που προσαρμόζει την κατανομή των δεδομένων εκπαίδευσης, με το να επικεντρώνεται σε πρότερα λανθασμένα ταξινομημένες εγγραφές
  - Αρχικά, όλα τα *n* σημεία παίρνουν το ίδιο βάρος
  - Σε αντίθεση με το bagging, τα βάρη μπορεί να αλλάξουν στο τέλος του γύρου boosting.

Boosting

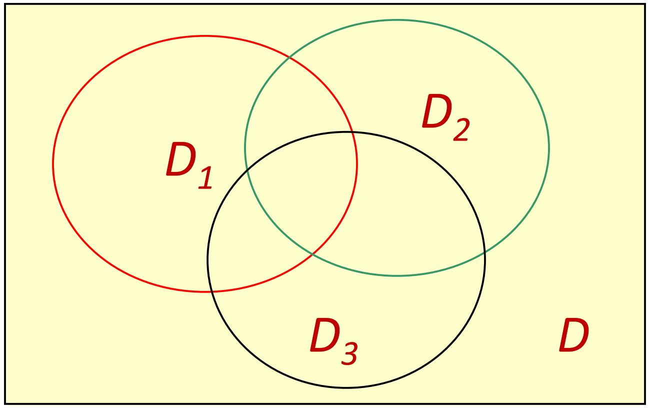
- Δεδομένα που ταξινομούνται λανθασμένα αυξάνουν το βάρος τους.
- Δεδομένα που ταξινομούνται σωστά μειώνουν το βάρος τους.

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Το σημείο 4 είναι δύσκολο να ταξινομηθεί, γι’ αυτό και αυξάνεται το βάρος του, ώστε να είναι πιο πιθανή η επιλογή του στους επόμενους γύρους

Boosting (Schapire 1989)

- Θεωρήστε ότι πρέπει να εκπαιδεύσετε τρεις ταξινομητές για ένα πρόβλημα δυο κλάσεων με τη χρήση boosting.
- Τυχαία επιλέξτε $n_1 < n$ δείγματα από το σετ δεδομένων D χωρίς αντικατάσταση για να δημιουργήσετε το D_1
  - Εκπαιδεύστε το μοντέλο C_1
- Επιλέξτε $n_2 < n$ δείγματα από το σετ δεδομένων D με τα μισά από τα δείγματα που ταξινομήθηκαν λάθος από το C_1 για να δημιουργήσετε το D_2
  - Εκπαιδεύστε το μοντέλο C_2
- Επιλέξτε όλα τα υπόλοιπα δείγματα από το σετ δεδομένων D, για τα οποία τα C_1 και C_2 διαφωνούν
  - Εκπαιδεύστε το μοντέλο C_3



- Ο τελικός ταξινομητής είναι η ψήφος των τριών μοντέλων ταξινόμησης

Adaboost - Adaptive Boosting

- Αντί για επαναδειγματοληψία, προσαρμόζει τα βάρη του σετ εκπαίδευσης
  - Κάθε δείγμα στο σετ εκπαίδευσης χρησιμοποιεί ένα βάρος που καθορίζει την πιθανότητά του να επιλεγεί σε ένα σετ εκπαίδευσης.
- Ο AdaBoost κατασκευάζει έναν “ισχυρό” ταξινομητή $H(x)$, με τον γραμμικό συνδυασμό πιο απλών, “αδύναμων” ταξινομητών $h_t(x)$
- Η τελική απόφαση ταξινόμησης βασίζεται σε μια ψηφοφορία με βάρη των αδύναμων ταξινομητών

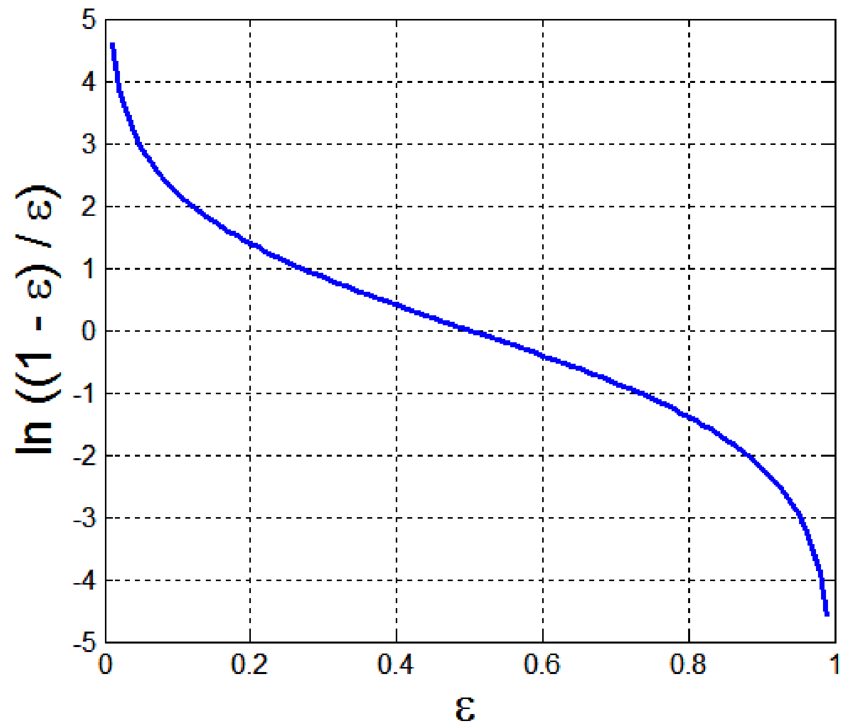
Η ορολογία του AdaBoost

- $h_i(x)$: “αδύναμος” ταξινομητής ή ταξινομητής βάσης
- Ταξινομητές βάσης: $h_1, h_2, \ldots, h_K$
- Ρυθμός σφάλματος:

$$\varepsilon_i = \frac{1}{N}\sum_{j=1}^{N} w_j \delta\ \ h_i(x_j) \neq y_j$$

- Σημασία ενός ταξινομητή:

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$



Η ορολογία του AdaBoost

- Προσαρμογή βάρους:

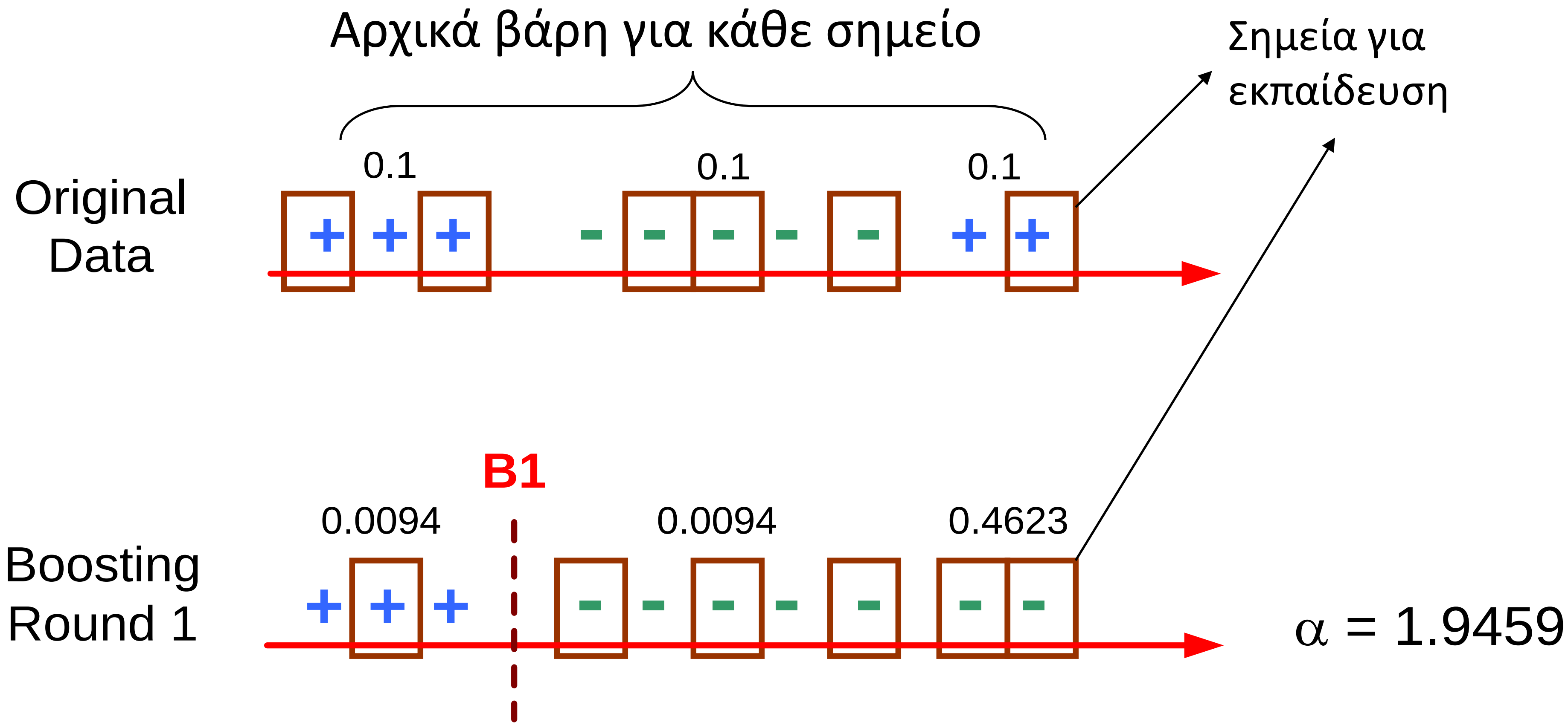
$$w_p^{(j+1)} = \frac{w_p^{(j)}}{Z_j} \begin{cases} \exp^{-\alpha_j} & \text{if } \mathrm{h}_t(x_p) = y_p \\ \exp^{\alpha_j} & \text{if } \mathrm{h}_t(x_p) \neq y_p \end{cases}$$

όπου Z_j ο παράγοντας κανονικοποίησης

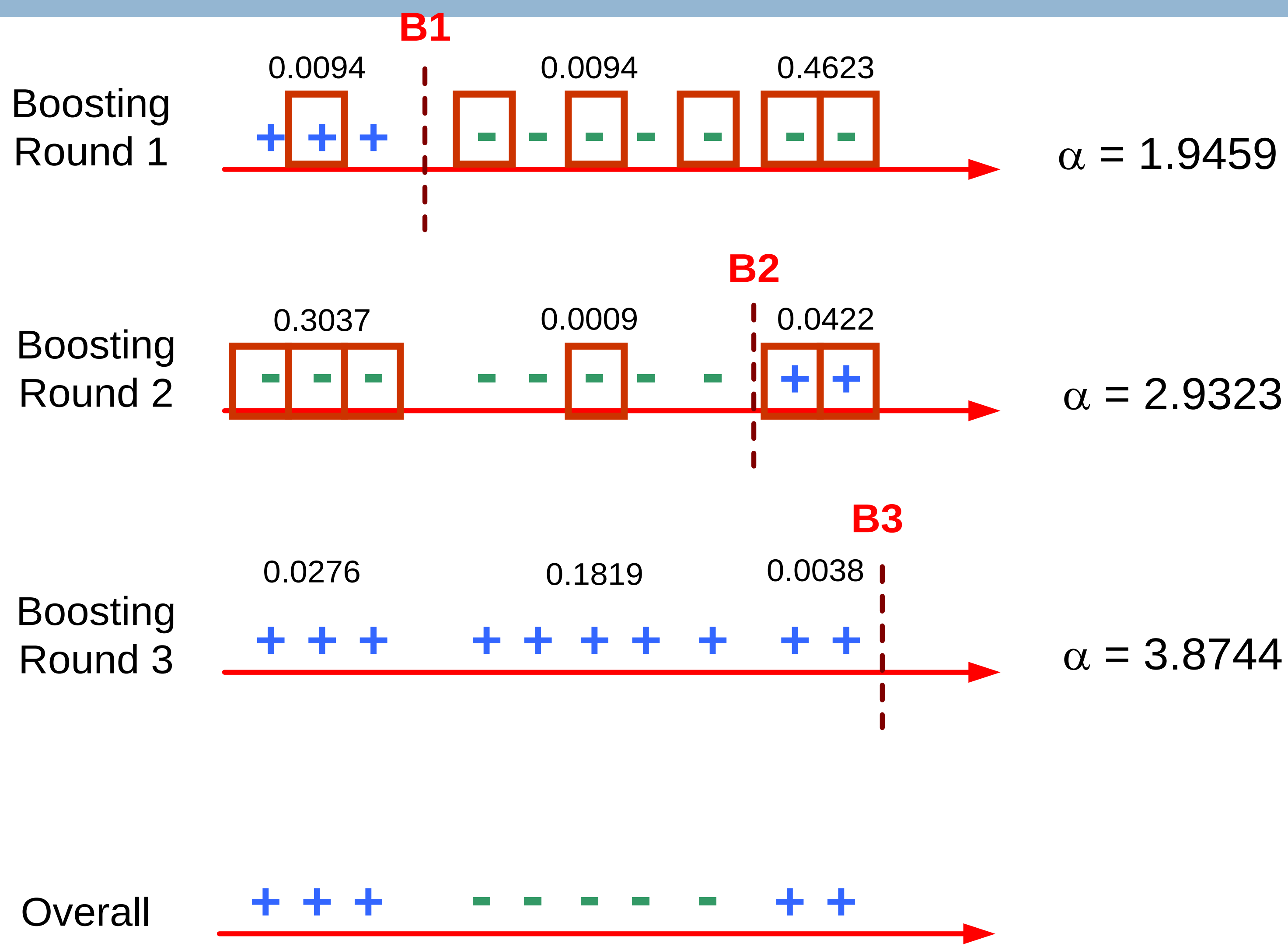
- Εάν κατά τους ενδιάμεσους γύρους, βρεθεί ασθενής ταξινομητής, τα βάρη επανακαθορίζονται στο *1/n* και δειγματοληψία με αντικατάσταση επαναλαμβάνεται.
- Ταξινόμηση:

$$H(x) = \arg\max_y \sum_{i=1}^{T} \alpha_i \delta \; h_i(x) = y$$

Επίδειξη AdaBoost



Επίδειξη AdaBoost (συν.)

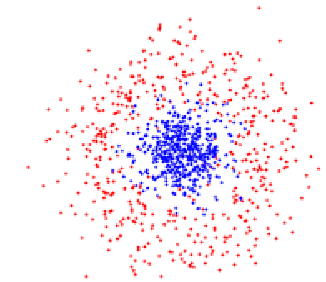


Τι είναι το τόσο καλό με τον AdaBoost

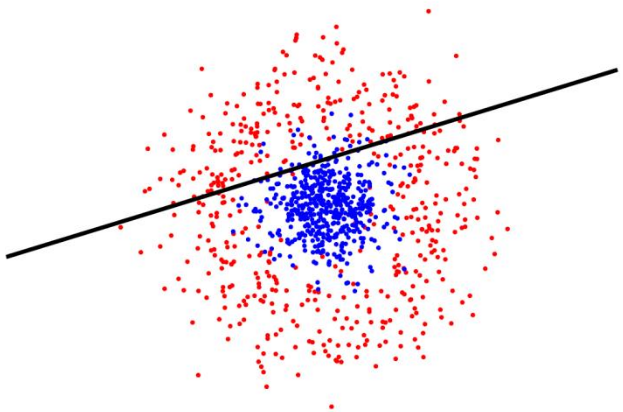
- Μπορεί να χρησιμοποιηθεί με πολλούς διαφορετικούς ταξινομητές
- Βελτιώνει την ακρίβεια ταξινόμησης
- Είναι απλός στην υλοποίηση
- Δεν είναι επιρρεπής σε υπερεκπαίδευση

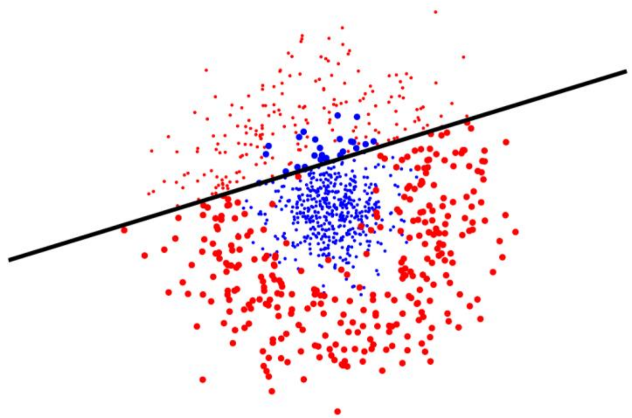
Εύρεση ασθενούς ταξινομητή

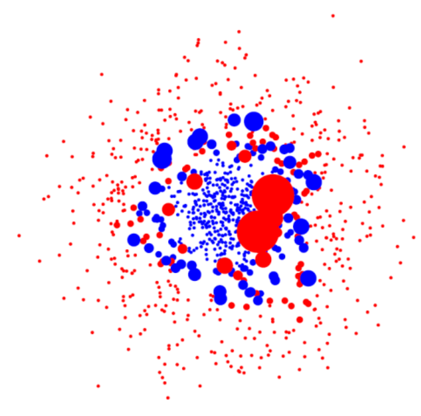
Training set



$\bullet \sim N(0,1)$ $\quad$ $\bullet \sim \frac{1}{2\pi}e^{-1/2(r-4)^2}$

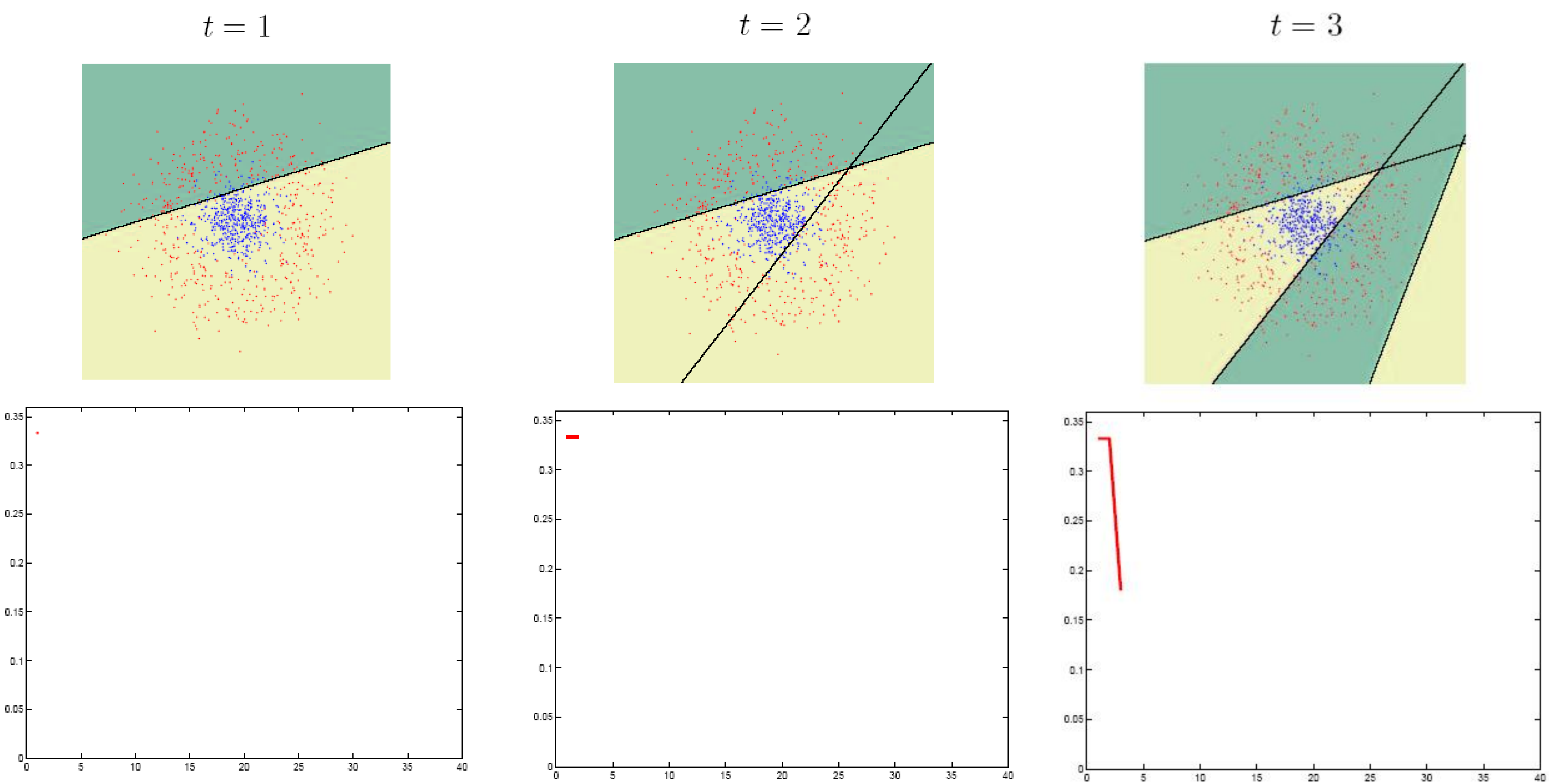




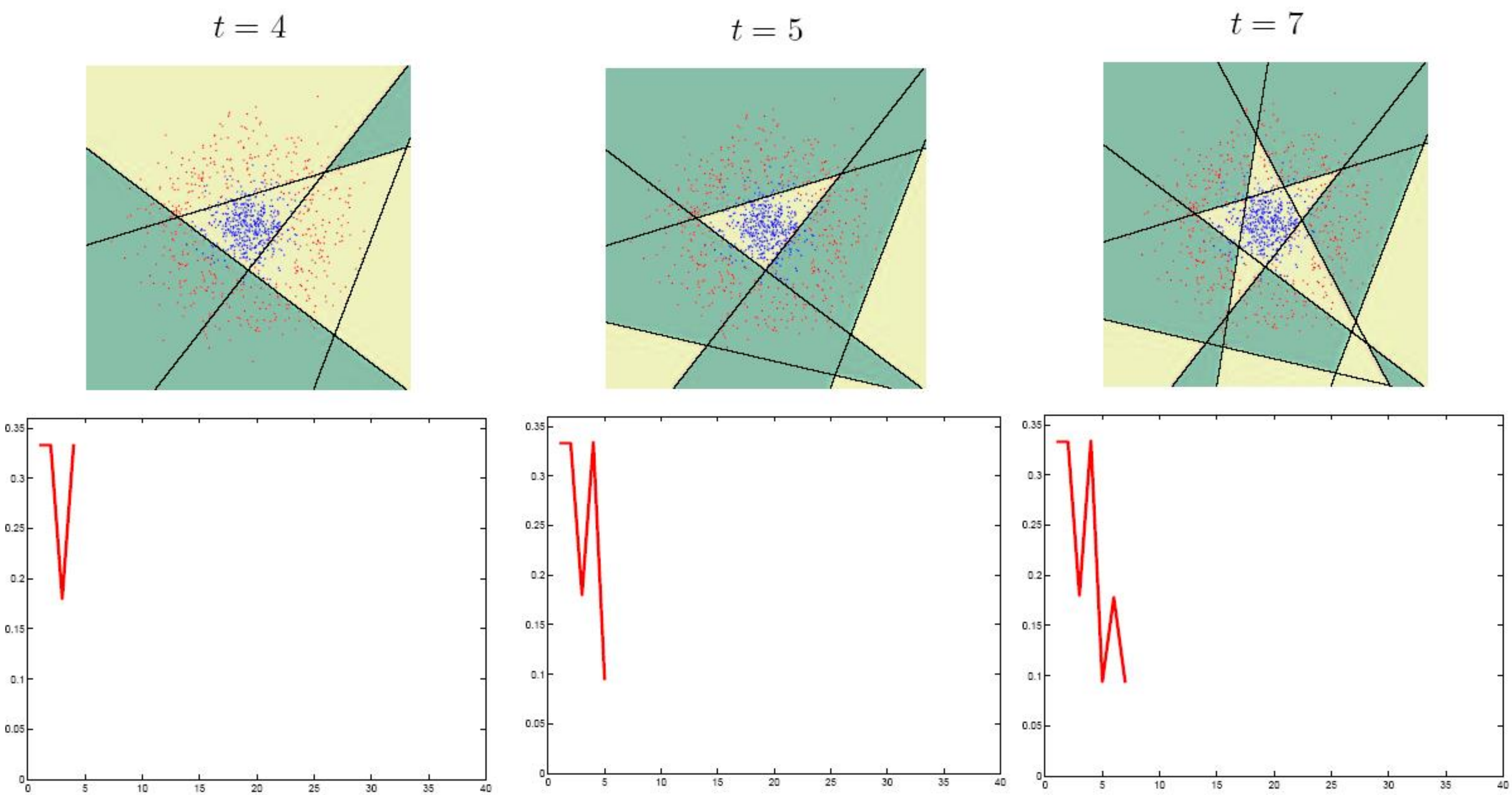


- Με τον τρόπο αυτό ο AdaBoost "επικεντρώνεται" στις "δύσκολες" περιοχές ταξινόμησης

Σταδιακά ο AdaBoost…

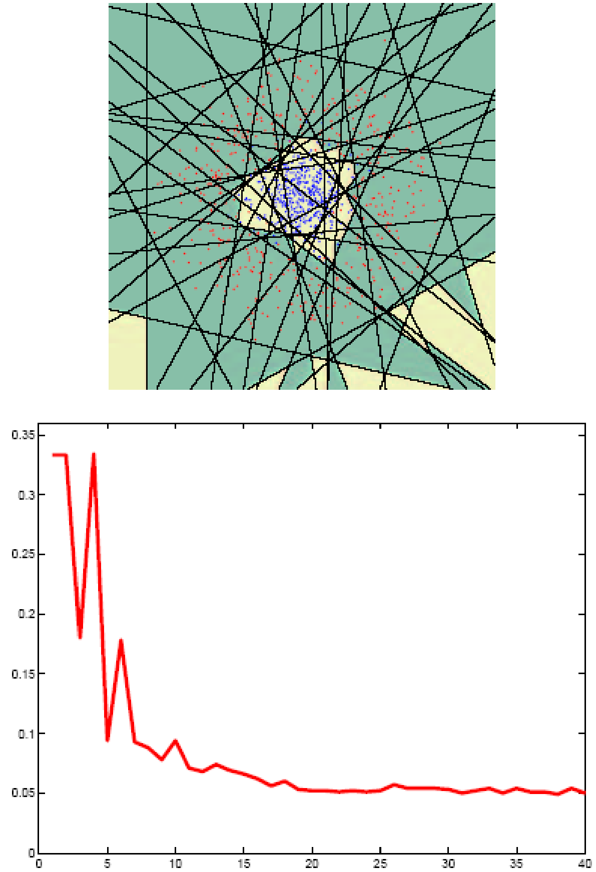


Σταδιακά ο AdaBoost (συν.)



Τελικά ο AdaBoost…

$t = 40$



Υπερ και κατά του AdaBoost

Πλεονεκτήματα

- Πολύ απλός στην υλοποίηση
- Κάνει επιλογή χαρακτηριστικών σε οδηγεί σε σχετικά απλούς ταξινομητές
- Γενικεύει σχετικά καλά

Μειονεκτήματα

- Υποβέλτιστη λύση
- Ευαίσθητος σε θόρυβο και εξωκείμενες τιμές

Παραλλαγές του Adaboost

- LogitBoost
  - Επιλύει $$\min_{f(x)} E_{w(x)} \left(F(x) + \frac{1}{2} \frac{y^* - p(x)}{p(x)(1-p(x))} - (F(x) + f(x)) \right)^2$$
  - Θέλει προσοχή ώστε να αποφευχθούν αριθμητικά προβλήματα

- GentleBoost
  - Η ανανέωση είναι $f_m(x) = P(y=1|x) - P(y=0|x)$ αντί για
  $$f_m(x) = \frac{1}{2} \log \frac{P_w(y=1|x)}{P_w(y=-1|x)}$$
  - Στο πεδίο [0 1]

LogitBoost

LogitBoost (2 classes)

1. Start with weights $w_i = 1/N \; i = 1, 2, \ldots, N$, $F(x) = 0$ and probability estimates $p(x_i) = \frac{1}{2}$.
2. Repeat for $m = 1, 2, \ldots, M$:
   (a) Compute the working response and weights
   $$\begin{aligned} z_i &= \frac{y_i^* - p(x_i)}{p(x_i)(1 - p(x_i))} \\ w_i &= p(x_i)(1 - p(x_i)) \end{aligned}$$
   (b) Fit the function $f_m(x)$ by a weighted least-squares regression of z_i to x_i using weights w_i.
   (c) Update $F(x) \leftarrow F(x) + \frac{1}{2} f_m(x)$ and $p(x) \leftarrow \frac{e^{F(x)}}{e^{F(x)} + e^{-F(x)}}$.
3. Output the classifier $\text{sign}[F(x)] = \text{sign}[\sum_{m=1}^{M} f_m(x)]$

GentleBoost

Gentle AdaBoost

1. Start with weights $w_i = 1/N$, $i = 1, 2, \ldots, N$, $F(x) = 0$.
2. Repeat for $m = 1, 2, \ldots, M$:
   (a) Fit the regression function $f_m(x)$ by weighted least-squares of y_i to x_i with weights w_i.
   (b) Update $F(x) \leftarrow F(x) + f_m(x)$
   (c) Update $w_i \leftarrow w_i e^{-y_i f_m(x_i)}$ and renormalize.
3. Output the classifier $\text{sign}[F(x)] = \text{sign}[\sum_{m=1}^{M} f_m(x)]$

Πηγές

- Introduction to Data Mining, Tan, Steinbach, Kumar.
- Y. Freund, and R. Shapire, “A decision-theoretic generalization of on-line learning and an application to boosting", Proceedings of the Second European Conference on Computational Learning Theory, 1995, pp. 23-37.
- P. A. Viola, M.J. Jones, “Robust Real-Time Face Detection", ICCV 2001, Vol. 2, pp. 747.
- T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer-Verlag, New York, 2001.

Ανακεφαλαίωση

- Βασικές έννοιες ταξινόμησης
- Διαφορετικοί τύποι ταξινόμησης
  - Δένδρα απόφασης
  - Πιθανοτικοί αλγόριθμοι – Naïve bayes, Bayesian Belief Networks
  - Γραμμικοί ταξινομητές – Perceptron, Support Vector Machines
  - Μη γραμμικοί ταξινομητές – Perceptron, Support Vector Machines
- Πηγές:
  - Introduction to Data Mining, Tan, Steinbach, Kumar.
  - Pattern recognition, Theodoridis & Koutroumbas.
  - SVM tutorial , Prof. Moore, CMU.
  - Bayesian Networks, Weng-Keen Wong, Oregon State University.