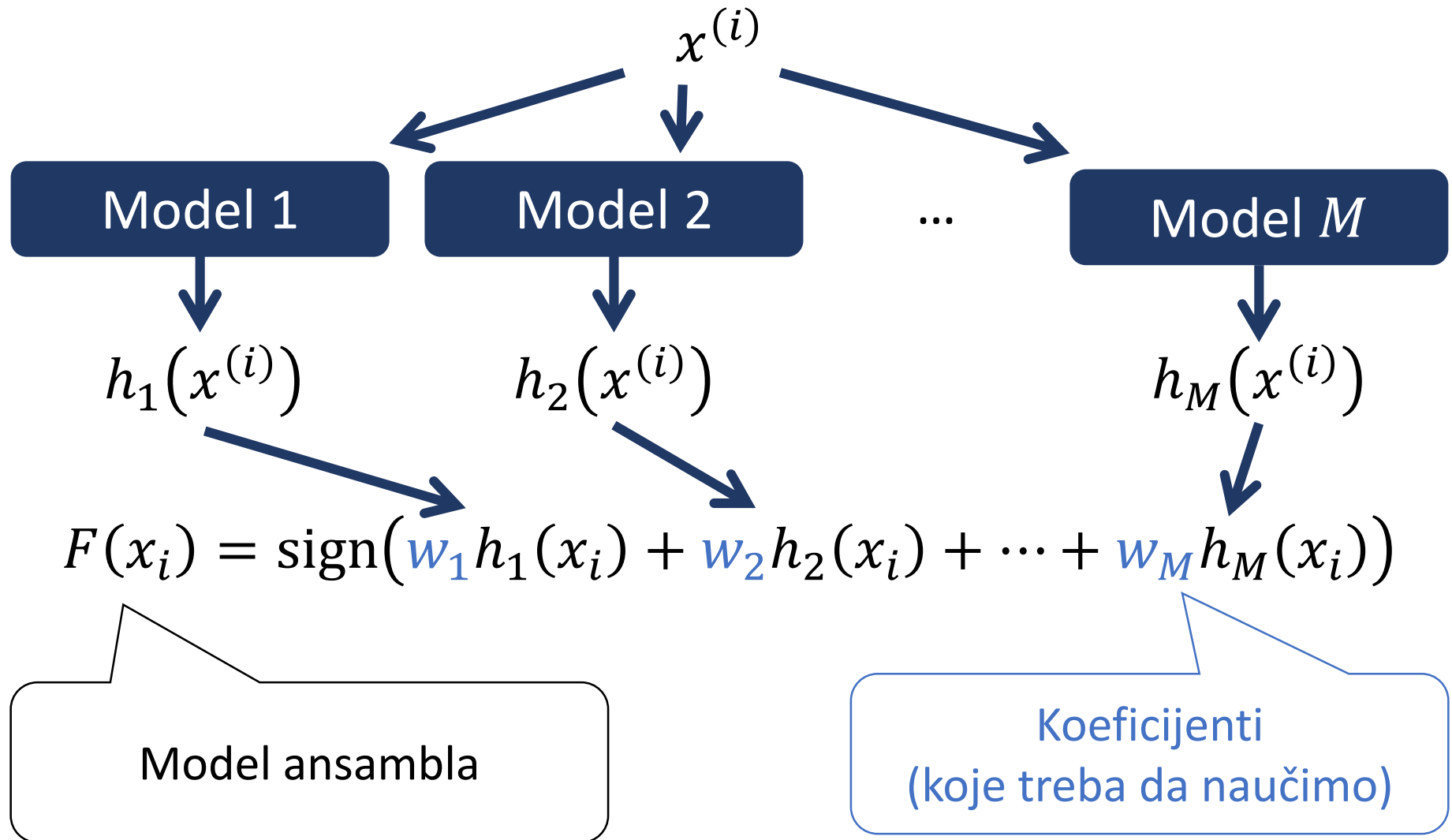


Boosting

Boosting: ideja

- Umesto da obučimo jedan (slab) klasifikator,
- obučićemo **mnogo** (slabih) klasifikatora
- koji su dobri **na različitim delovima ulaznog prostora**

Boosting: ideja



Ansambl (grupa) prediktora

- Obučavanje modela ansambla:
 - Članovi ansambla (klasifikatori): $h_1(x), h_2(x), \dots, h_M(x)$
 - Koeficijenti: w_1, w_2, \dots, w_M

- Predikcija:

$$y = \text{sign} \left(\sum_{m=1}^M w_m h_m(x) \right)$$

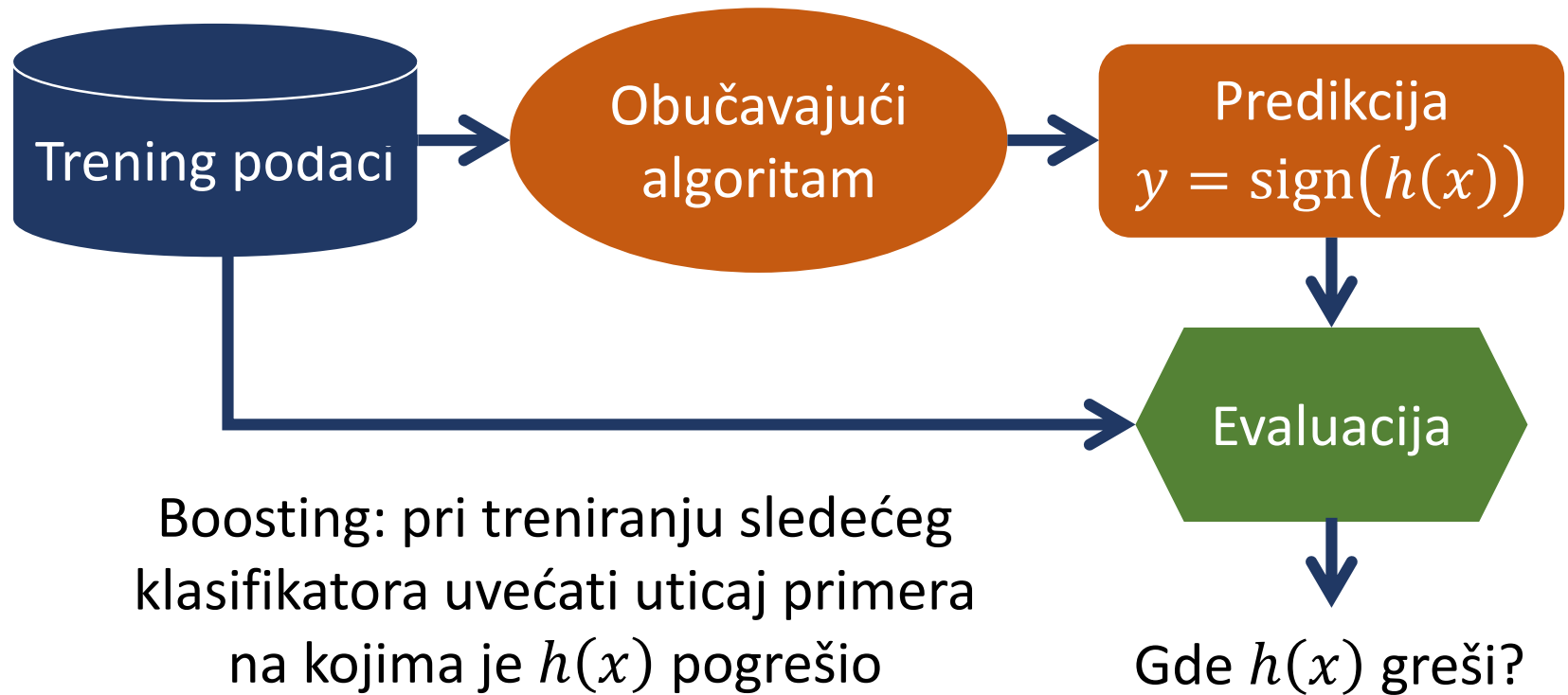
- Kako:
 1. Prinuditi klasifikatore da nauče različite delove ulaznog prostora?
 2. Dodeliti težinu različitim klasifikatorima (w_i)?

Kako prinuditi klasifikatore da nauče različite delove ulaznog prostora?

Fokusiraćemo se na „teške“ primere
(one na kojima su prethodni klasifikatori pogrešili)

Fokusiranje na „teške“ primere

- Krenuti od slabog prediktora (jedini uslov je da radi malo bolje od slučajnosti)
- Poboljšati prediktor fokusiranjem na teške slučajeve



Kako uvećati uticaj određenog primera?

- Sve instance su jednake, ali neke instance su jednakije od drugih
- Svakom primeru $x^{(i)}$ iz trening skupa dodelićemo težinu α_i
 - Dajemo veću težinu „teškim“ ili „važnim“ primerima
- Obučavanje na otežinjenim podacima:
 - Primer i se računa kao α_i primera
 - Npr. $\alpha_i = 2 \rightarrow$ računaj instancu i dva puta
 - Ako bismo „resamplovali“ podatke, dobili bismo više „teških“ primera

Učenje na otežinjenim podacima

1. Ponovo uzorkovati skup podataka (*resample*)

- Iz skupa podataka (sa N instanci) na slučajan način uzorkovati N instanci
- Teži primeri (sa većim α) imaju veću verovatnoću selekcije i mogu biti selektovani više puta

Učenje na otežinjenim podacima

2. Koristiti obučavajući algoritam koji može direktno da koristi težine primera

- Logistička regresija

$$\theta_j^{(t+1)} \leftarrow \theta_j^{(t)} + \eta \sum_{i=1}^N \alpha_i h_j(x_i) \left((h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \right)$$

- Naivni Bajes (ili bilo koji MLE metod) – Redefinisati $\text{count}(y = c)$ da bude otežinjen broj

Neotežinjeni podaci

$$\text{count}(y = c) = \sum_{i=1}^N \mathbb{I}(y^{(i)} = c)$$

Otežinjeni podaci

$$\text{count}(y = c) = \sum_{i=1}^N \alpha_i \mathbb{I}(y^{(i)} = c)$$

Boosting

- Boosting je pohlepan (*greedy*) algoritam za učenje ansambla iz podataka
 - Za algoritam kažemo da je pohlepan ako u svakom koraku bira lokalno optimalno rešenje (nadajući se da će na taj način doći do globalnog optimuma)
 - Odluka se donosi inkrementalno u malim koracima
 - Odluka u svakom koraku vodi ka popravljaju trenutnog stanja i pomeranju ka cilju
 - Ne vodi se računa o posledicama ranijih loših izbora

AdaBoost

- Jedan konkretan boosting algoritam
- Jedan od ranih boosting algoritama *Freund and Schapire 1999*
- Veoma efektivan, a jednostavan za implementaciju

AdaBoost

Ulaz	<ul style="list-style-type: none">• $T = \{(x^{(i)}, y^{(i)}), i = 1, \dots, N, y \in \{-1, +1\}\}$ – trening skup• M – broj slabih prediktora ?
Postupak	<ul style="list-style-type: none">• Inicijalizovati težine primera tako da budu uniformne $\alpha_i = \frac{1}{N}, i = 1, \dots, N$• for $m = 1, \dots, M$<ul style="list-style-type: none">• Obučiti klasifikator $h_m(x) \rightarrow \{+1, -1\}$ koristeći težine α_i• Odrediti težinu klasifikatora w_m ?• Rekalkulisti težine instanci α_i
Izlaz	$\hat{y} = \text{sign} \left(\sum_{m=1}^M w_m h_m(x) \right)$

Problem 1: kako odrediti koeficijente w_m ?

- Koliko verujemo klasifikatoru h_m ?
 - Ako je klasifikator h_m „dobar“ on ima nisku trening grešku
- Kako da merimo grešku na otežinjenim podacima?
 - Ako smo dodelili veću težinu nekim primerima (zato što su teški) želimo da klasifikator dobro klasifikuje te primere
 - Za grešku ćemo uzeti otežinjen broj pogrešno klasifikovanih primera

$y^{(i)}$	$h_m(x^{(i)})$	α_i
+1	+1	1.2
-1	-1	1
+1	-1	0.5
-1	+1	1.1
+1	+1	0.8

Težina korektno klasifikovanih primera	$1.2 + 1 + 0.8$
Težina grešaka	$0.5 + 1.1$

Problem 1: kako odrediti koeficijente w_m ?

- Ukupna težina grešaka:

$$\sum_{i=1}^N \alpha_i \mathbb{I}(h_m(x^{(i)}) \neq y^{(i)})$$

- Ukupna težina instanci:

$$\sum_{i=1}^N \alpha_i$$

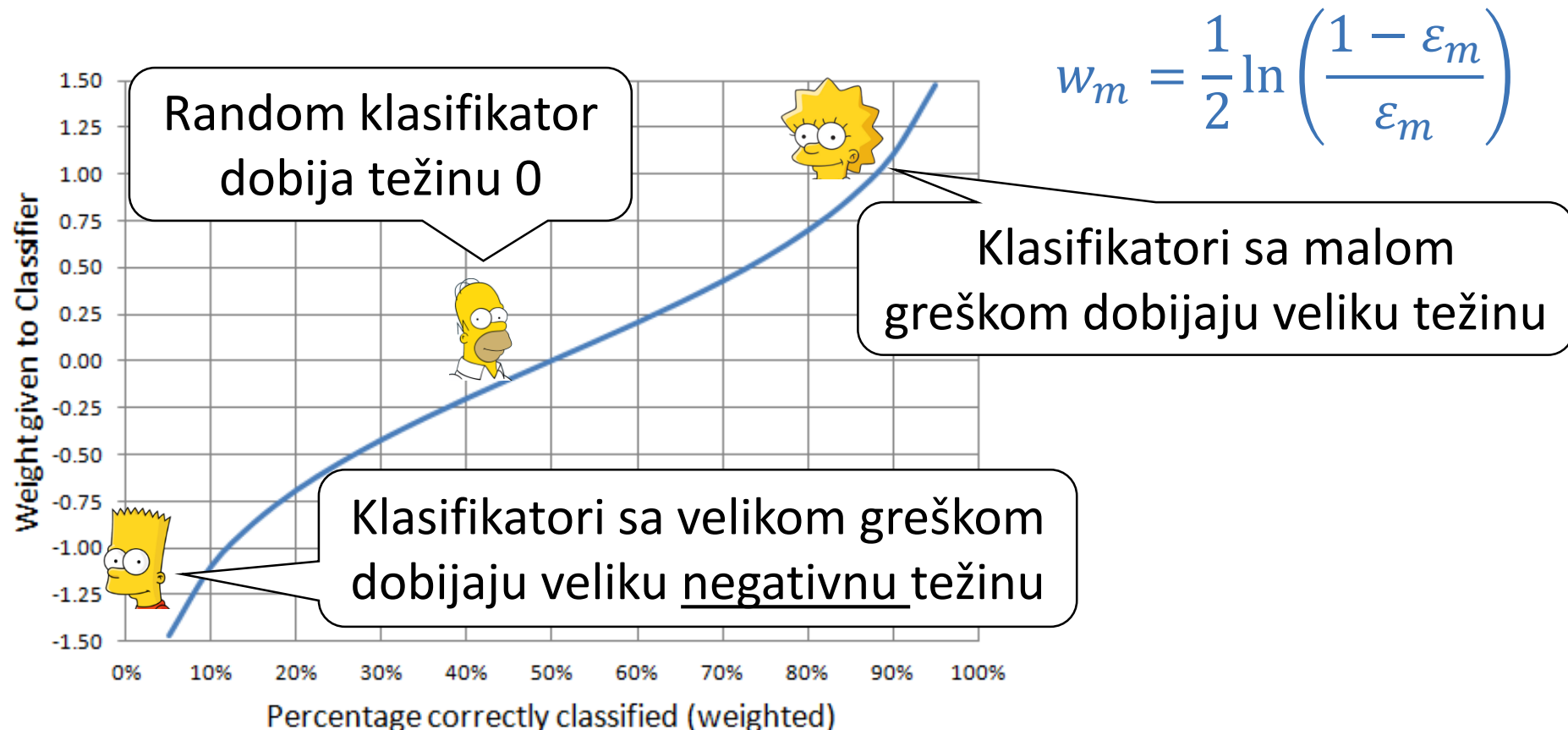
- Otežinjena greška:

$$\varepsilon = \frac{\text{ukupna težina grešaka}}{\text{ukupna težina instanci}} \in [0, 1]$$

- Najbolja vrednost ε je 0
- Najgora vrednost ε je 0.5

Problem 1: kako odrediti koeficijente w_m ?

- AdaBoost: računanje koeficijenta w_m za klasifikator $h_m(x)$ za koji je otežinjena greška ε_m

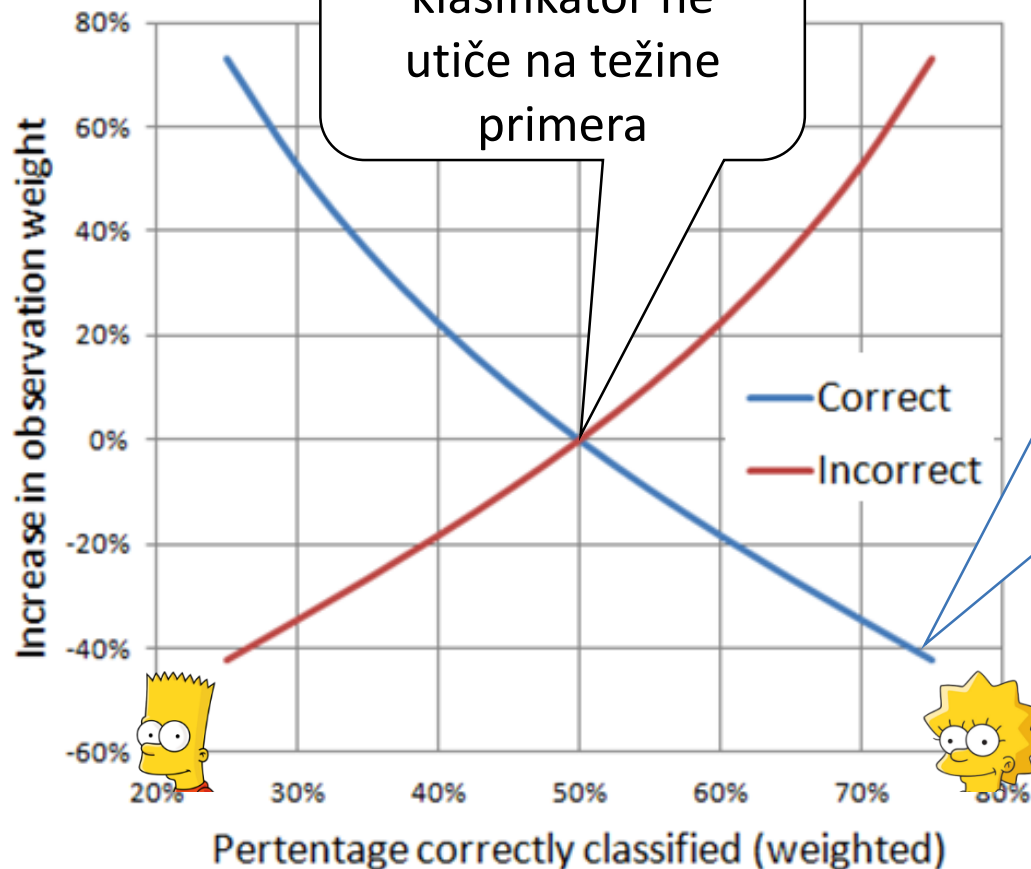


Problem 2: kako odrediti težine primera α_i ?

- AdaBoost: ažuriranje težina α_i bazirano na tome gde $h_m(x)$ pravi greške:

$$\alpha_i \leftarrow \begin{cases} \alpha_i e^{-w_m}, & h_m(x^{(i)}) = y^{(i)} \\ \alpha_i e^{w_m} & h_m(x^{(i)}) \neq y^{(i)} \end{cases}$$

Random
klasifikator ne
utiče na težine
primera



Ako je klasifikator dobar (veliko pozitivno w_m)

- smanjujemo težinu primera koje korektno klasifikuje
- povećavamo težinu onih na kojima greši

Normalizacija težina α_i

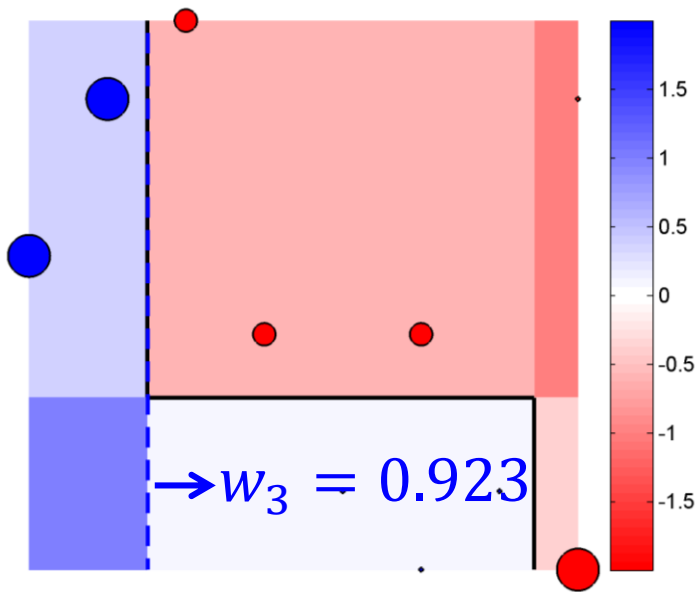
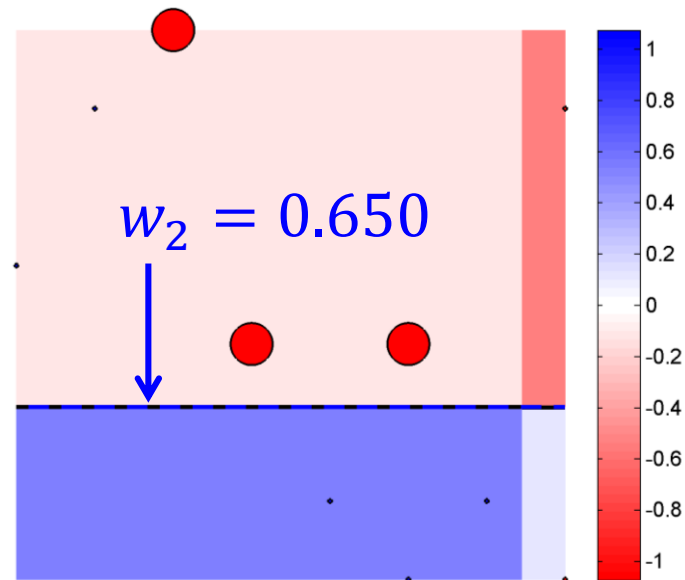
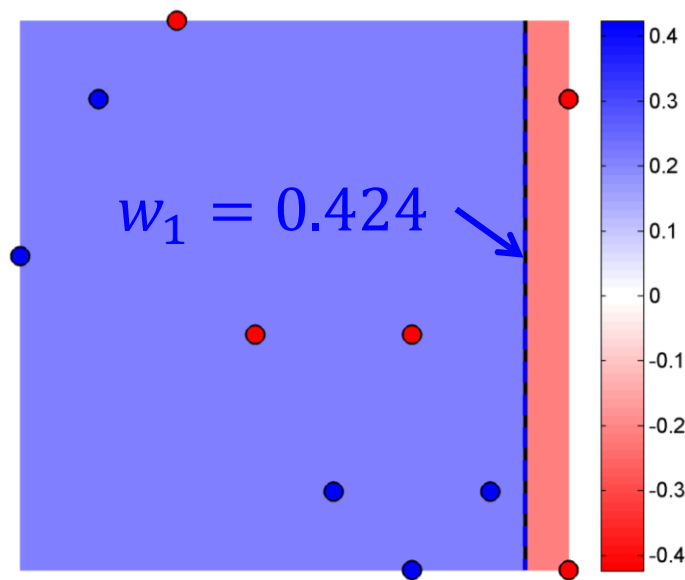
- Ako je primer $x^{(i)}$ „težak“
 - Mnogo klasifikatora greši na njemu
 - U svakoj iteraciji se težina primera uvećava
 - **Težina α_i postaje veoma velika**
- Ako je primer $x^{(i)}$ „lak“
 - Mnogi klasifikatori ga dobro klasifikuju
 - U svakoj iteraciji se težina primera smanjuje
 - **Težina α_i postaje veoma mala**
- Ovo može dovesti do numeričke nestabilnosti nakon mnogo iteracija
- Rešenje je da u svakoj iteraciji normalizujemo težine primera da se sabiraju na 1:

$$\alpha_i \leftarrow \frac{\alpha_i}{\sum_{j=1}^N \alpha_j}$$

AdaBoost sumaryzacija

Ulaz	<ul style="list-style-type: none">$\{(x^{(i)}, y^{(i)}), i = 1, \dots, N, y \in \{-1, +1\}\}$ – trening skupM – broj slabih prediktora ?
Postupak	<ul style="list-style-type: none">Inicijalizovati težine primera tako da budu uniformne $\alpha_i = \frac{1}{N}, i = 1, \dots, N$for $m = 1, \dots, M$<ul style="list-style-type: none">Obučiti klasifikator $h_m(x)$ koristeći težine α_iIzračunati koeficijent w_m $\rightarrow w_m = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_m}{\varepsilon_m} \right)$Rekalkulisti težine α_i $\rightarrow \alpha_i \leftarrow \begin{cases} \alpha_i e^{-w_m}, & h_m(x_i) = y_i \\ \alpha_i e^{w_m} & h_m(x_i) \neq y_i \end{cases}$Normalizovati težine α_i $\rightarrow \alpha_i \leftarrow \frac{\alpha_i}{\sum_{j=1}^N \alpha_j}$
Izlaz	$\hat{y} = \text{sign} \left(\sum_{t=1}^T w_t f_t(x) \right)$

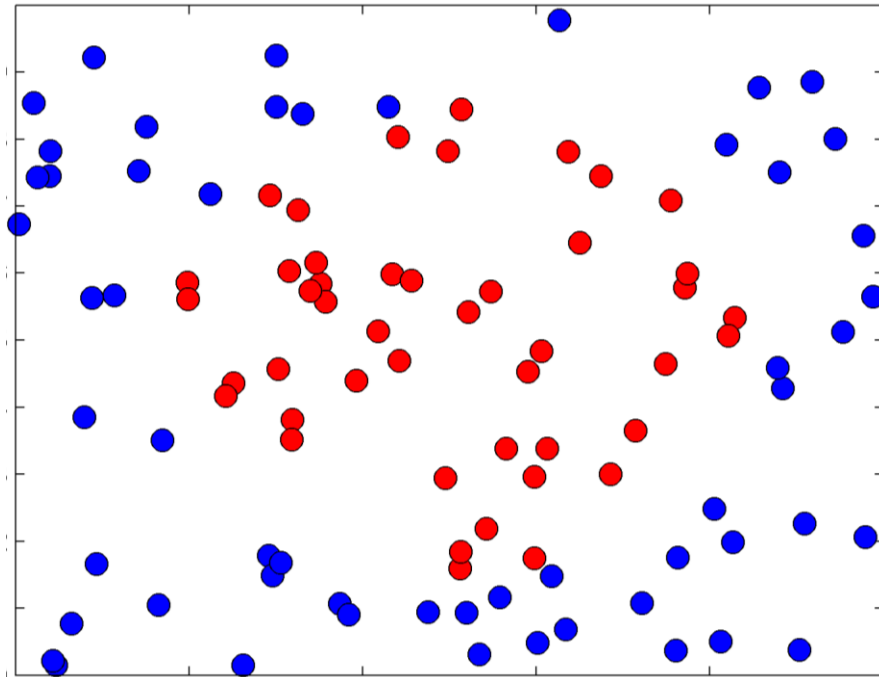
AdaBoost primer



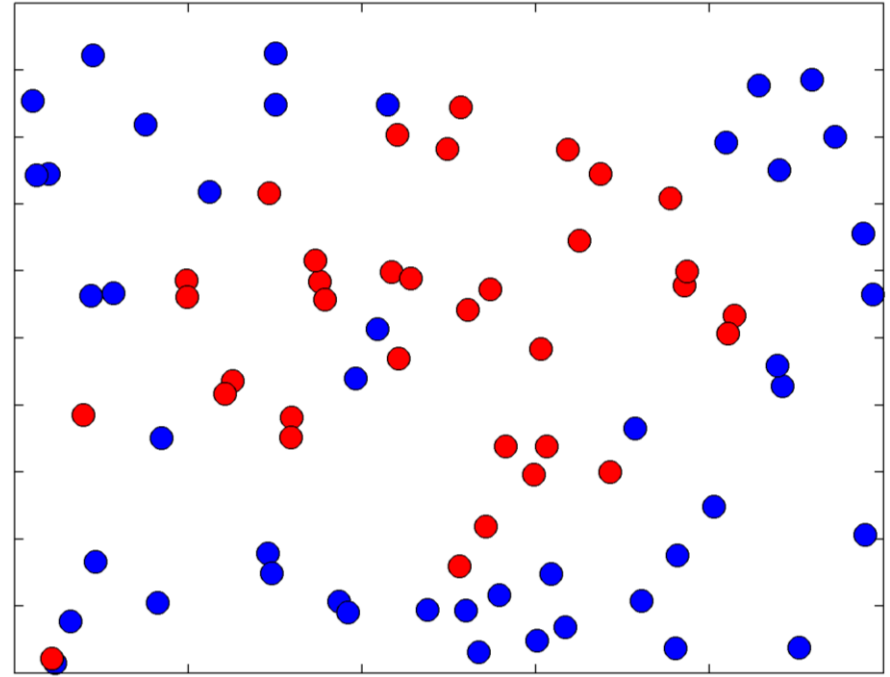
$$\begin{aligned} F(x) &= 0.424 \cdot h_1(x) \\ &+ 0.650 \cdot h_2(x) \\ &+ 0.923 \cdot h_3(x) \end{aligned}$$

AdaBoost primer 2

Svi podaci (trening + test skup)

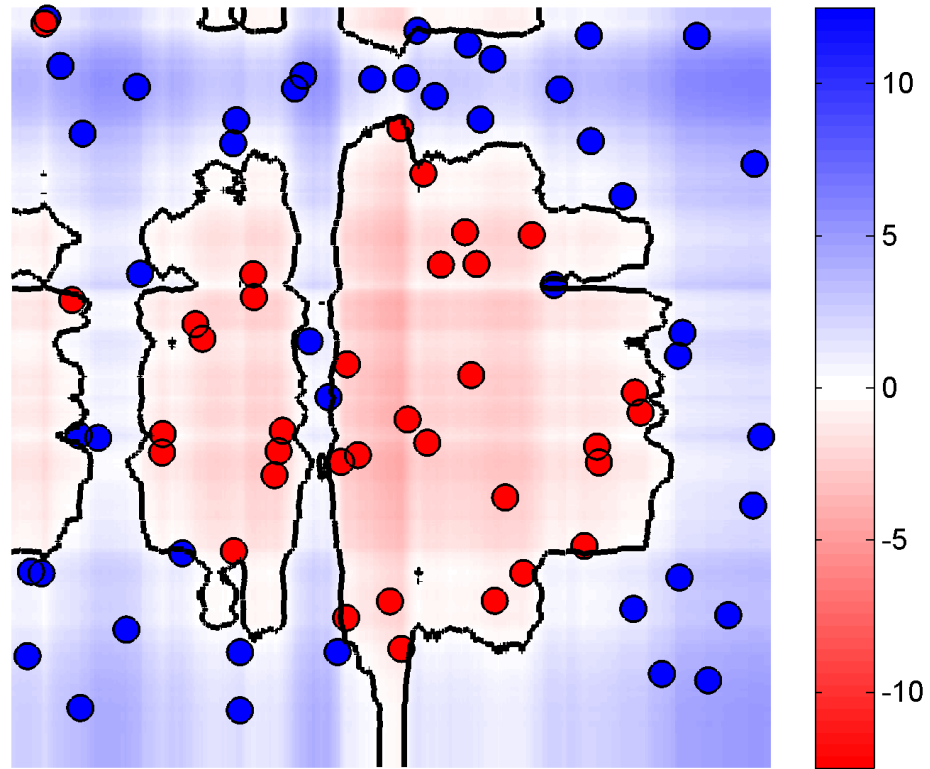
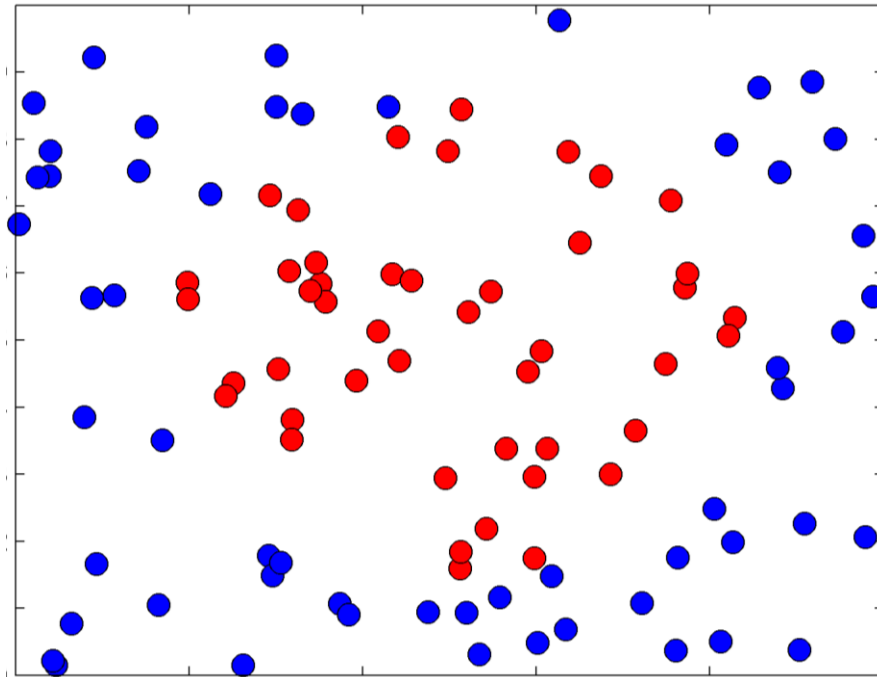


Trening skup (sa šumom)



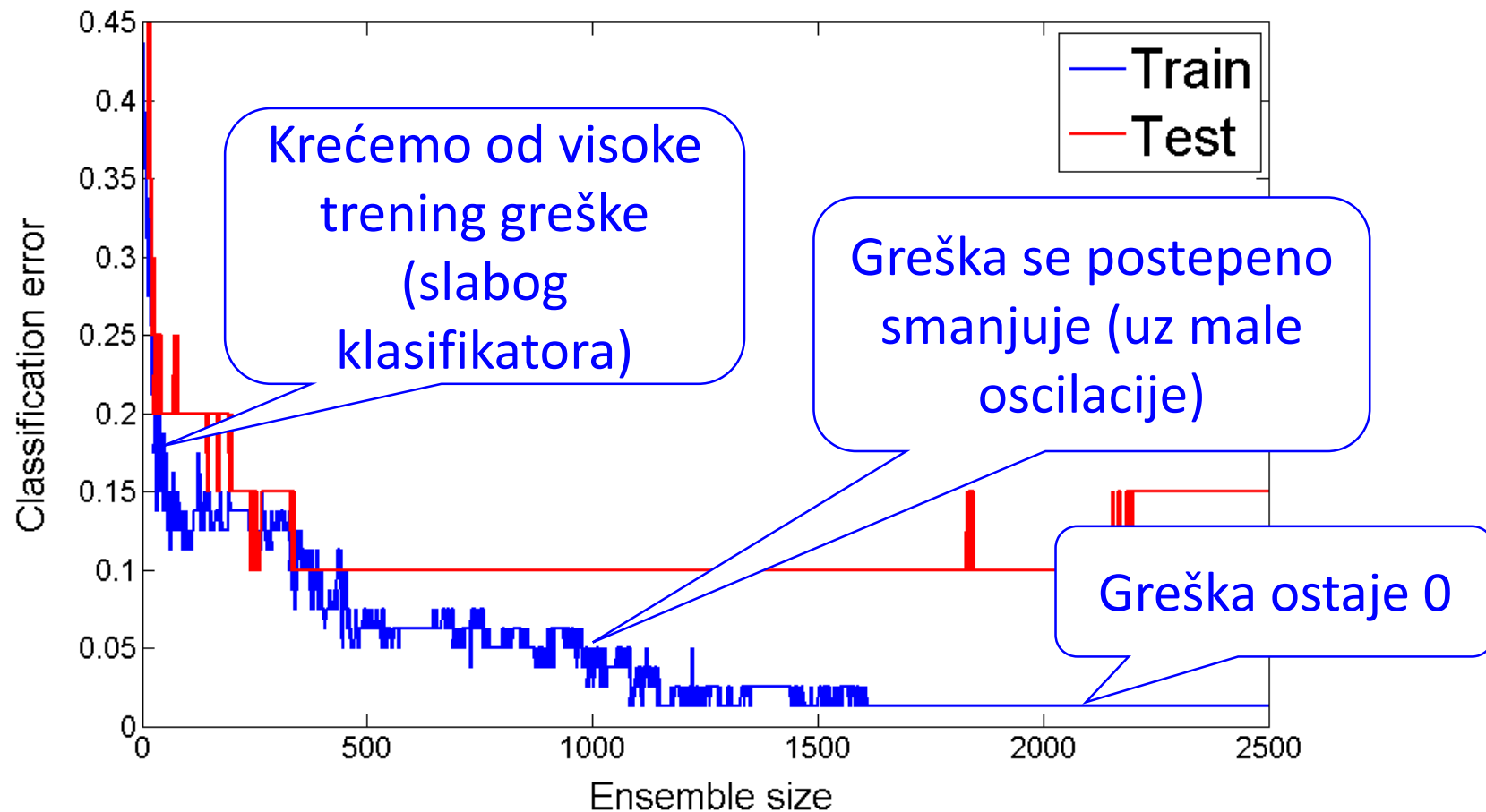
AdaBoost primer 2

Svi podaci (trening + test skup)



- Najbolja trening tačnost pojedinačnog klasifikatora: 0.66
- Konačna trening tačnost boostinga: 1
- Najbolja test tačnost pojedinačnog klasifikatora: 0.65
- Konačna test tačnost boostinga: 0.85

AdaBoost greška na trening skupu




Ovo je tipična *trening* greška koju ćemo uočiti kod boostinga

AdaBoost teorema

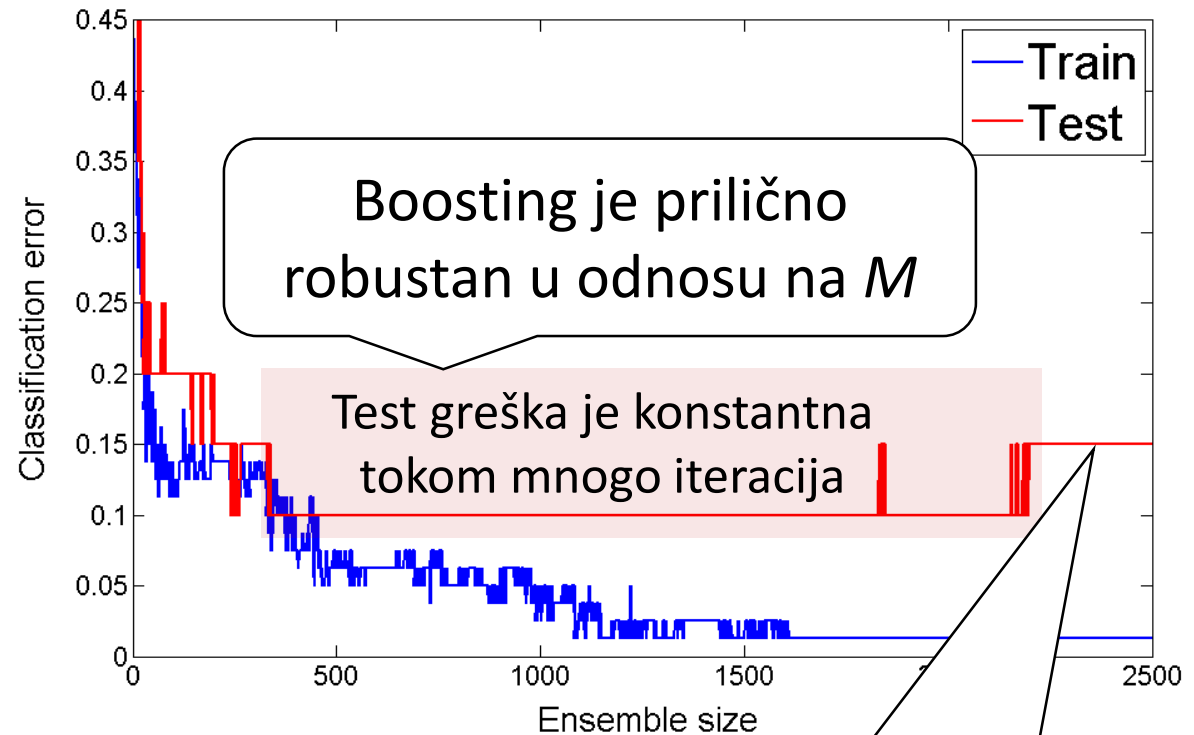
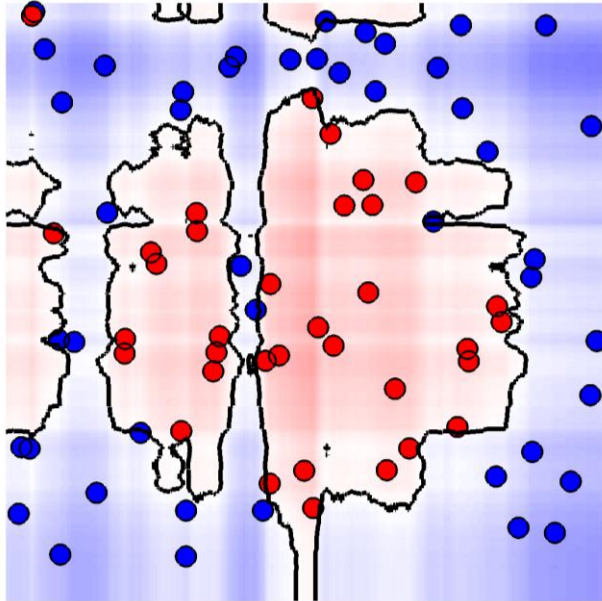
$$E_{train} \rightarrow 0 \text{ kada } M \rightarrow \infty$$

- Pod uslovom: u svakoj iteraciji možemo pronaći (slab) klasifikator sa otežinjenom greškom manjom od 0.5

$$\varepsilon(h_m) < 0.5$$

- Ovaj uslov nije uvek moguće ispuniti
 - Npr. ne postoji klasifikator koji može razdvojiti 
 - Bez obzira na ovo, boosting često postiže veoma nisku grešku na trening skupu kada broj iteracija teži beskonačnosti (iako možda ne tačno 0)

AdaBoost greška na test skupu



Ali, ipak može doći do overfittinga

Problem 3: određivanje M

- M je hiper-parametar koji pravi nagodbu između prilagođavanja podacima i kompleksnosti modela

