Osztályozókról még pár dolog

Csima Judit

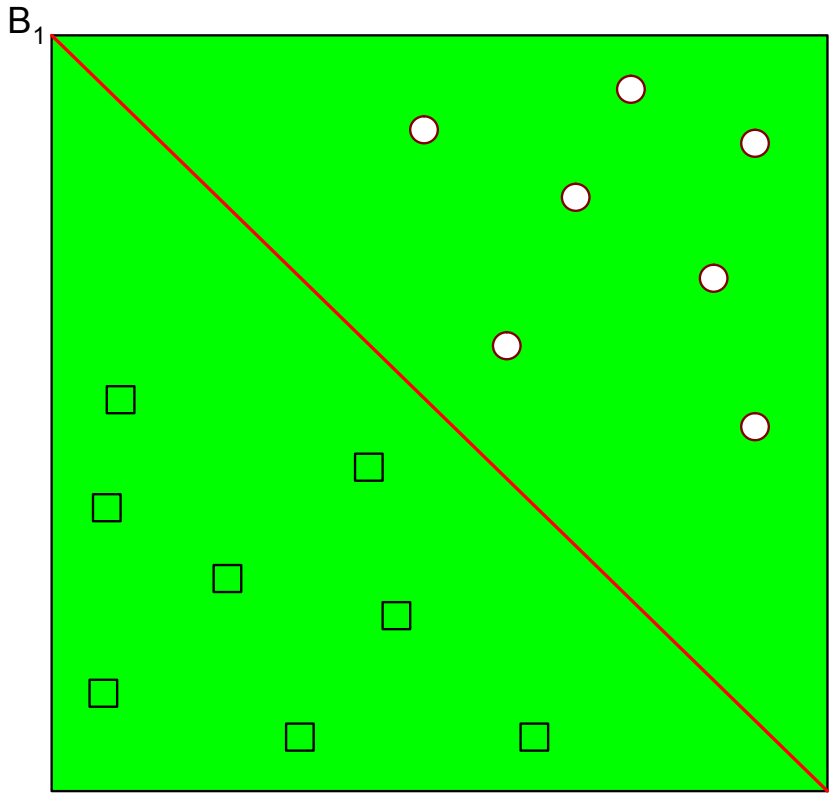
BME, VIK,
Számítástudományi és Információelméleti Tanszék

2015. április 8.

SVM (support vector machine)

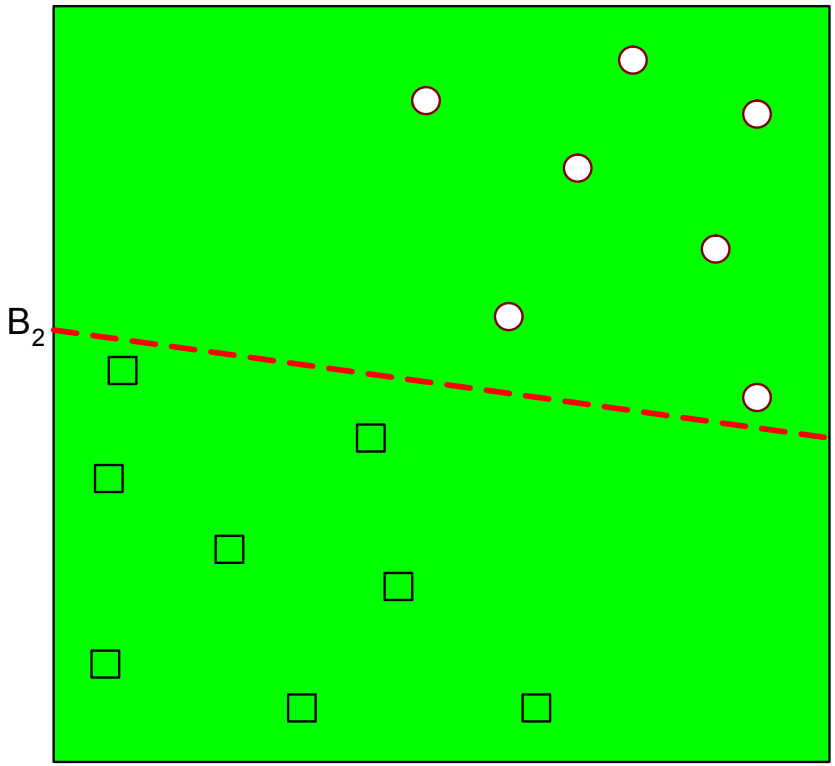
- ez is egy osztályozó
- ha lineárisan szeparálható a két osztály: cél az, hogy az őket elválasztó "senki földje" közepén történjen a szeparálás
- alapfogalom a margin: legalább ekkora távolságra van minden training pont a szeparáló hipersíktól (térben sík, síkon egyenes)

Support Vector Machines



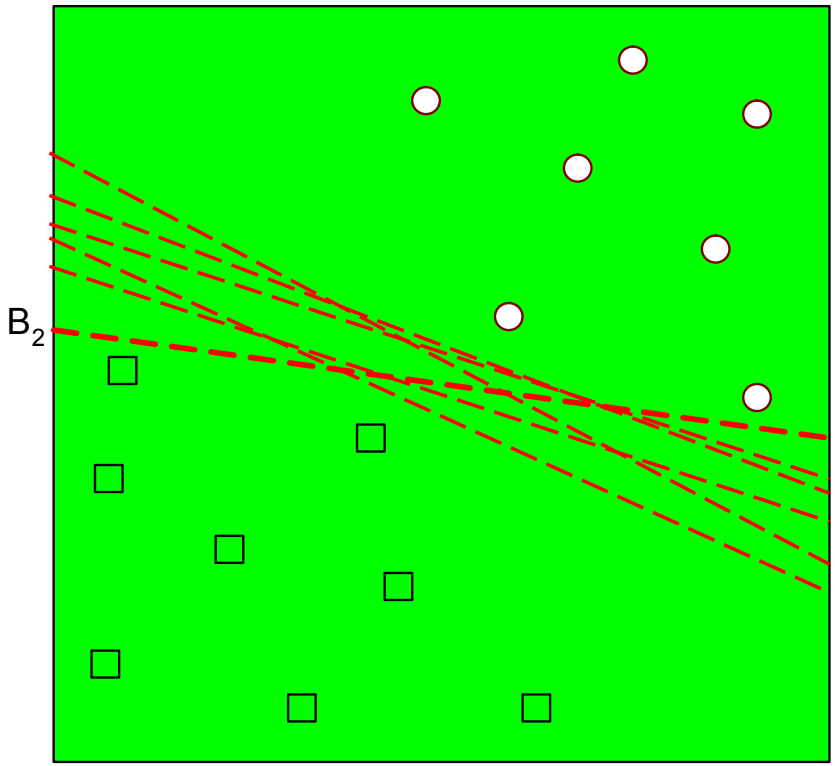
- One Possible Solution

Support Vector Machines



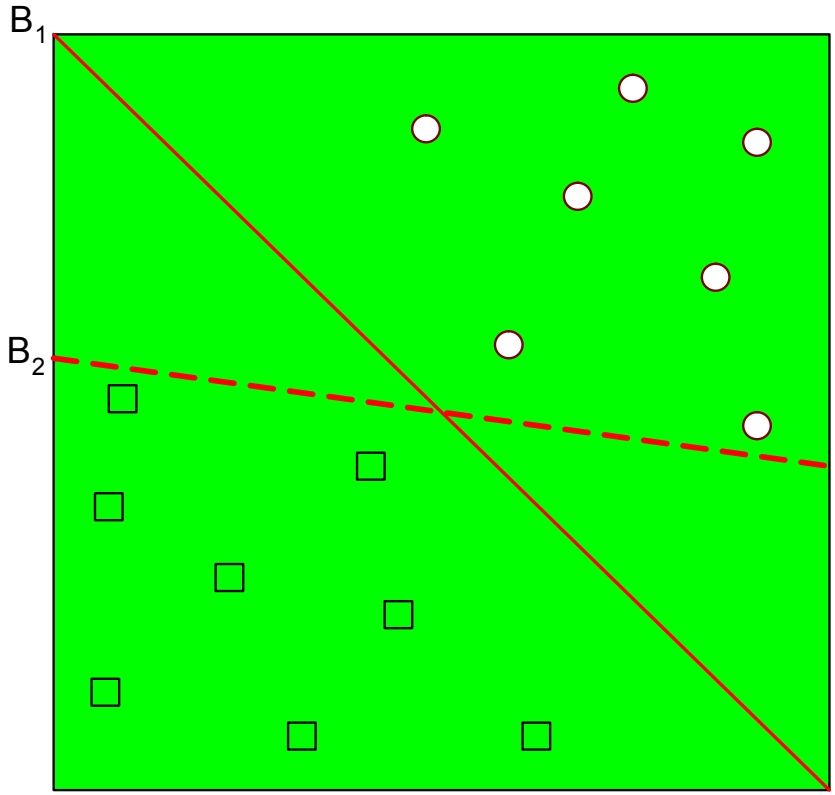
- Another possible solution

Support Vector Machines



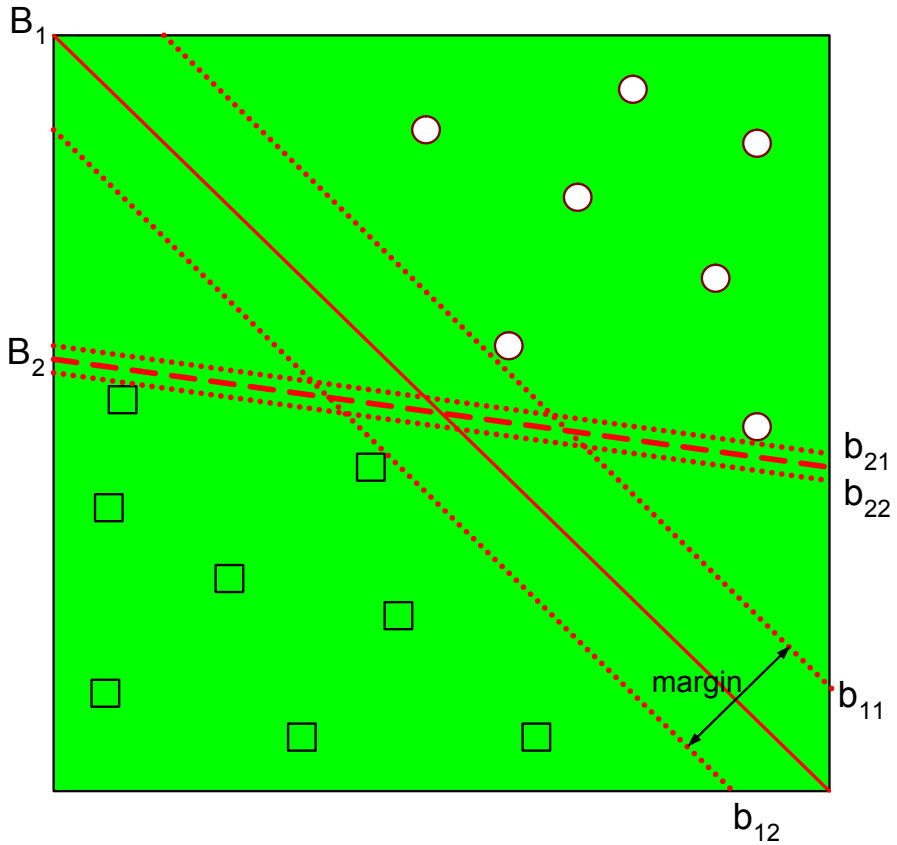
- Other possible solutions

Support Vector Machines



- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines

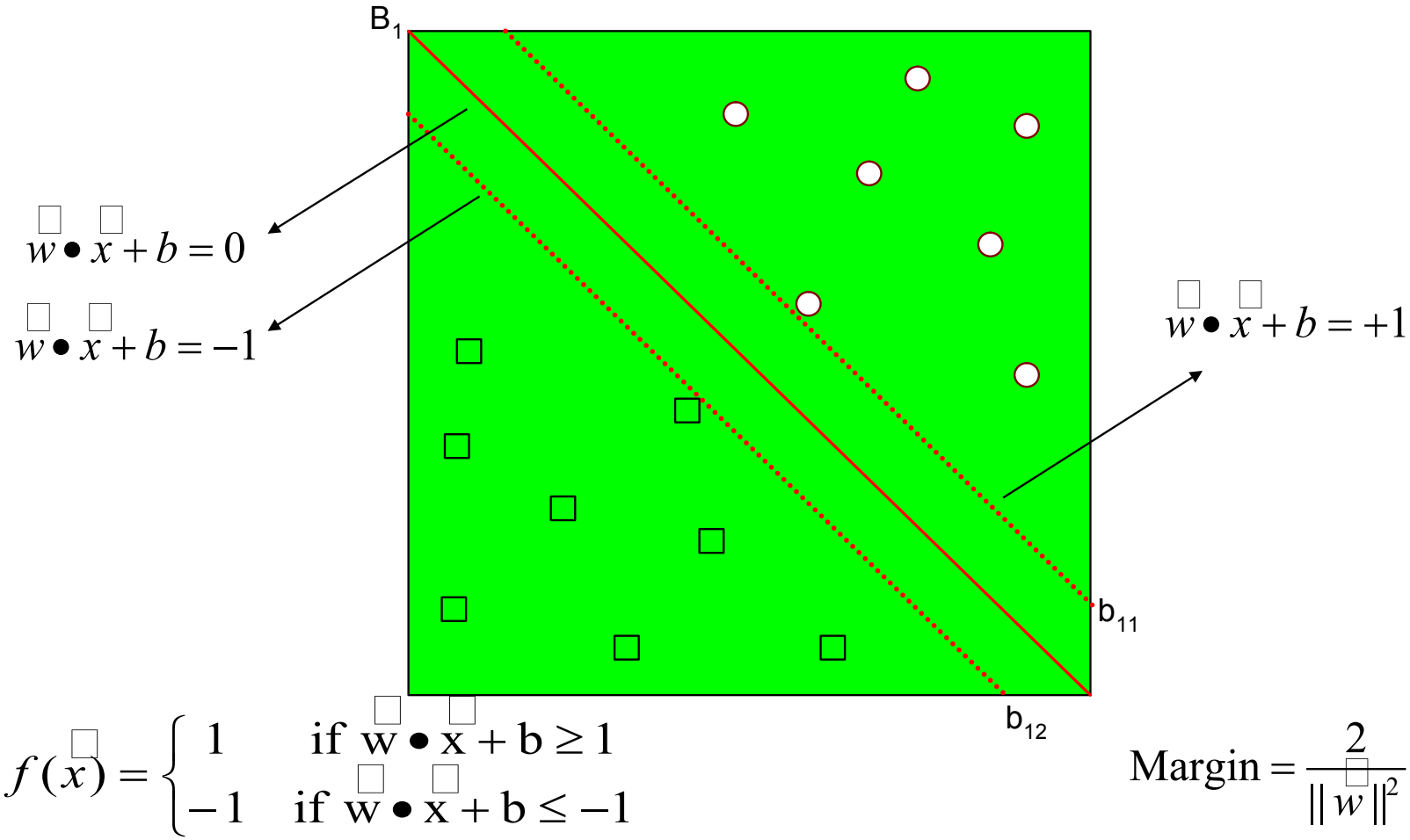


- Find hyperplane maximizes the margin => B1 is better than B2

Miért akarunk nagy margin-t?

- jobban általánosítódik az ilyen osztályozó: jobban viselkedik az új adatokon
- kisebb az esélye az overfittingnek: nincs túlságosan rászabva a training adatokra

Support Vector Machines



Support Vector Machines

- We want to maximize: $\text{Margin} = \frac{2}{\| w \|^2}$
  - Which is equivalent to minimizing: $L(w) = \frac{\| w \|^2}{2}$
  - But subjected to the following constraints:

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

    - This is a constrained optimization problem
      - Numerical approaches to solve it (e.g., quadratic programming)

SVM-ről még pár dolog

- ez csak az intuíció volt, hogy mit akarunk, részletesen nem nézzük, hogy hogyan kell megtalálni a legjobb szeparálást
- lineárisan nem szeparálható esetekre is van elmélet
- R-ben package e1071 (ugyanebben a package-ben van naív Bayes osztályozó is és még sok minden más)

Több osztályozó használata egyszerre

- Eddig egy osztályozót építettünk és azzal címkéztük az új eseteket
- Hatékonyabb, ha több osztályozónk van és ezeket egyszerre használjuk:
  - base classifier-eket építek
  - ezeket egymástól függetlenül lefuttatom
  - a többségi címkét választom

Why does it work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

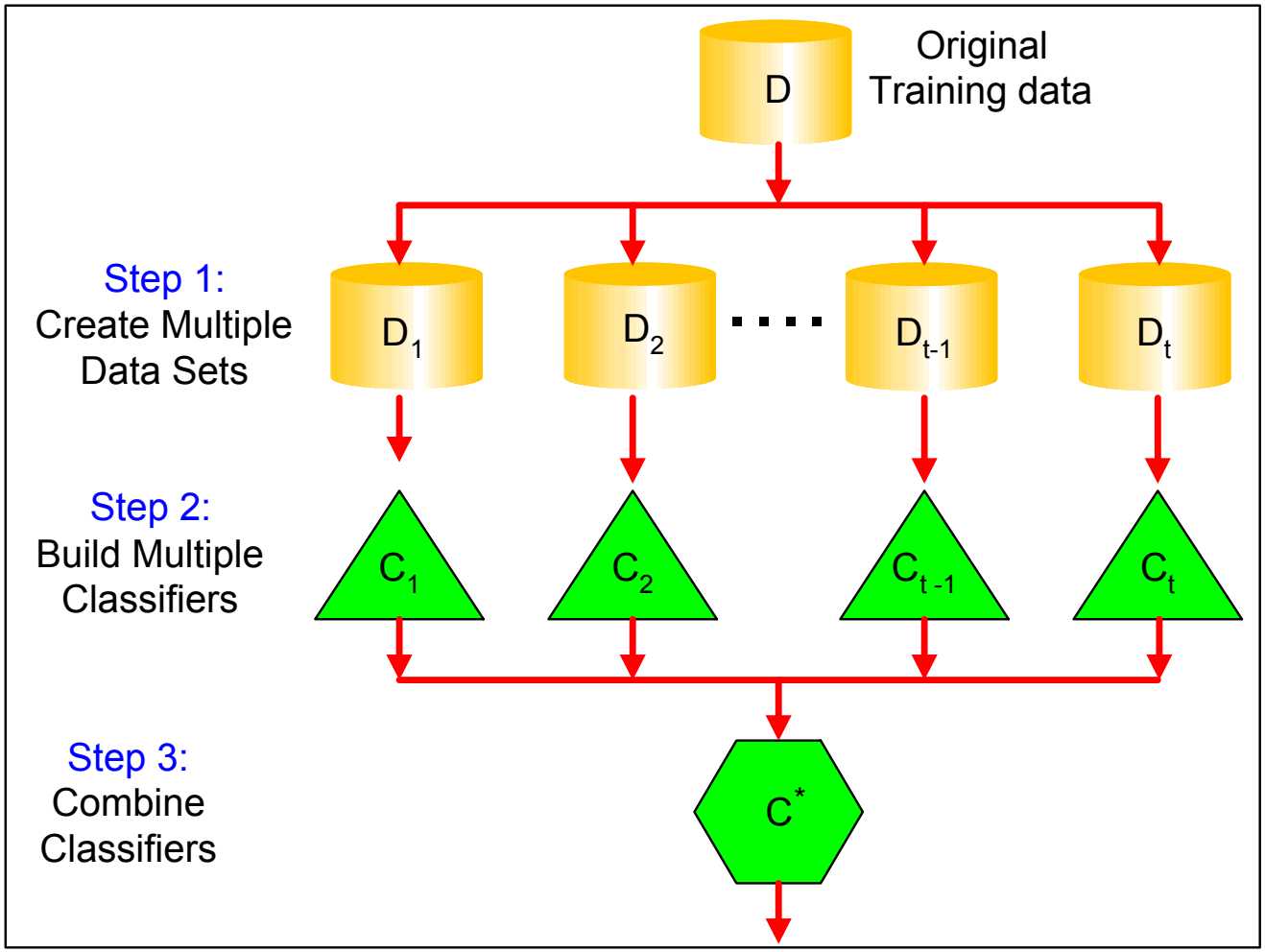
Szükséges feltételek

- az osztályozók függetlenek legyenek (ez nehezen teljesíthető, de ennek ellenére segít a több osztályozó használata)
- a base classifier-ek hibája 0.5-nél kisebb legyen (különben együtt még rosszabbak, mint külön-külön)

Hogy lesz több osztályozónk?

- az input változók változtatásával
  - ezek részhalmazait figyelembe véve építek osztályozókat
  - pl. random forest, ha döntési fákról van szó
- címkék manipulásával
  - ha sok címke van, akkor ezeket kettéosztom és minden (néhány) kettéosztáshoz készítek osztályozót
  - új sor osztályozásánál minden osztályozó egy szavazatot generál azoknak a címkéknek, amik az általa választott részhalmazba esnek
  - a végén a legtöbb szavazatot kapó címke nyer
- az osztályozót előállító algo módosításával
  - pl. ANN-nél a hálózat topológiája vagy kezdő Θ változtatása
  - döntési fáknál: nem a legjobb vágást veszem, hanem a legjobb k közül egyet véletlenszerűen
- training set darabolásával (erről mindjárt)

General Idea



Egy training setből több osztályozó

- ha nagyon sok training adat van: szétbontás diszjunkt részekre és mindből egy osztályozó
- bagging:
  - visszatevéses mintavételezés, n elemű mintákat állítok elő úgy, hogy újra és újra húzok visszatevéssel az eredeti training setből
  - egymástól függetlenül több, azonos elemszámú mintát készítek így
  - minden mintában ugyanakkora esélye vanegy rekordnak a bekerülésre
- boosting:
  - egymás után készítem a mintákat, egy minta kiválasztása függ az előző mintán felépített osztályozó teljesítményétől
  - az egyes rekordok bekerülési valószínűségei változnak az egyes mintáknál
  - az előző körben nem jól osztályozott rekordok nagyobb vgel kerülnek be a következő körbe

Bagging

- Sampling with replacement

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

- Build classifier on each bootstrap sample

- Each sample has probability $(1 - 1/n)^n$ of being selected

Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

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

• Example 4 is hard to classify

• Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds