Machine Learning

Contents

8.1. Why?

- Different learners use different
  - Algorithms
  - Hyper-parameters
  - Representations (Modalities)
  - Training sets
  - Subproblems
- No Free Lunch Theorem
  - There is no single algorithm that is always the most accurate
  - Each learners assumes a certain model (assumptions, inductive bias)
  - Even if we fine tune the algorithm, there are instances the algorithm is not accurate – there may be another algorithm that is accurate on those
- Generate a group of base learners which when combined has higher accuracy

8.2. Voting

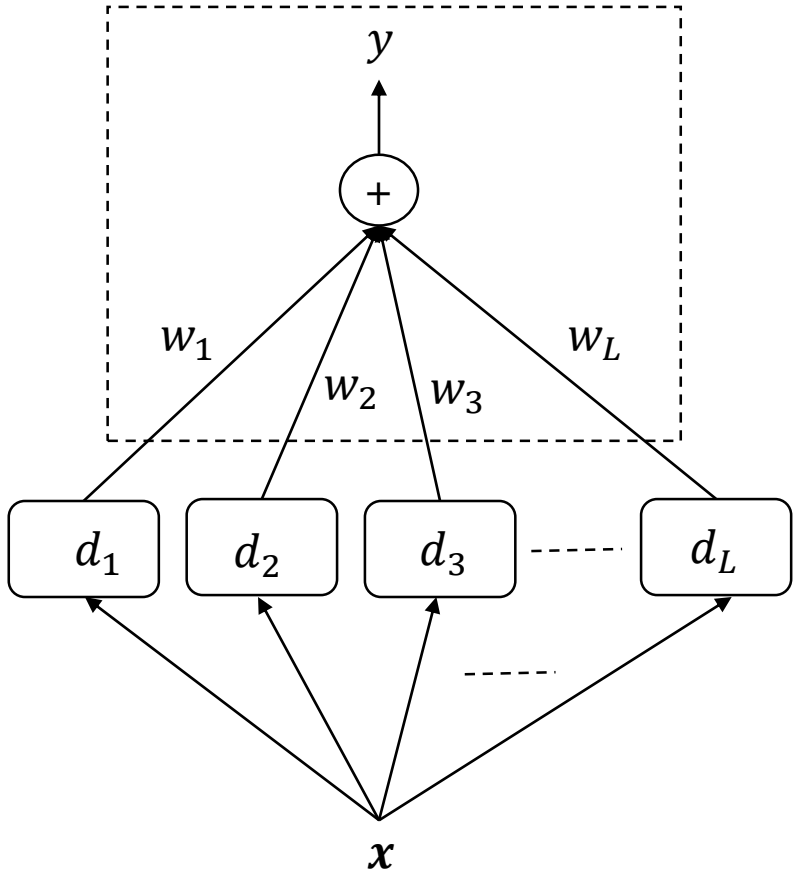
- Linear combination for output

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$

- Classification

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$



  - Simple voting: equal weight
  - Plurality voting: class with max vote is the winner
  - Majority voting (2 class problem)

- Bayesian perspective (Bayesian model combination):

$$P(C_i|x) = \sum_{\text{all models } \mathcal{M}_j} P(C_i|x, \mathcal{M}_j)P(\mathcal{M}_j)$$

- If d_j are i.i.d.(independent, identically distributed)

$$E[y] = E\left[\sum_j \frac{1}{L} d_j\right] = \frac{1}{L} L E[d_j] = E[d_j]$$

$$Var[y] = Var\left[\sum_j \frac{1}{L} d_j\right] = \frac{1}{L^2} Var[\sum_j d_j] = \frac{1}{L^2} L Var[d_j] = \frac{1}{L} Var[d_j]$$

  - Bias does not change, variance decrease by factor *L*
- Suppose base learner = discriminant/regression function + random noise
  - If noises are uncorrelated with 0 mean, we are averaging over noise

Given an input value $\boldsymbol{x}$ and L base learners, seek the weights such that

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} E[(r^* - \sum_{j=1}^{L} w_j d_j(\boldsymbol{x}))^2]$$

- Expectation done with regard to the true input distribution
- Note $E[(r^* - \sum_{j=1}^{L} \hat{w}_j d_j(\boldsymbol{x}))^2] \leq E[(r^* - \sum_{j=1}^{L} \frac{1}{L} d_j(\boldsymbol{x}))^2]$

Input distribution is not available in practice

- Use empirical distribution from training dataset

참조: 독립 확률변수의 분산

N개의 독립 확률변수 $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$가 있으며 이들의 평균과 분산을 각각 $m_i = E[\boldsymbol{x}_i]$와 $\sigma_i^2 = E[(\boldsymbol{x}_i - m_i)^2]$이라고 하자. 이를 합한 것을 z라고 하면

$$z = \sum_{i=1}^{N} \boldsymbol{x}_i \tag{8.2.9}$$

이다. 그러면, 평균은

$$E[z] = \sum_{i=1}^{N} E[\boldsymbol{x}_i] = \sum_{i=1}^{N} m_i \tag{8.2.10}$$

이다. 분산은

$$\sigma_z^2 = E[(z - E[z])^2] = E[(\sum_{i=1}^{N} (\boldsymbol{x}_i - m_i))^2] \tag{8.2.11}$$

이다. 여기서, $\boldsymbol{x}_i$와 $\boldsymbol{x}_j (i \neq j)$는 독립이므로

$$E[(\boldsymbol{x}_i - m_i)(\boldsymbol{x}_j - m_j)] = E[(\boldsymbol{x}_i - m_i)]E[(\boldsymbol{x}_j - m_j)] = 0 \tag{8.2.12}$$

인 것을 이용하면 식 (8.2.11)는

$$\sigma_z^2 = \sum_{i=1}^{N} E[(\boldsymbol{x}_i - m_i)^2] = \sum_{i=1}^{N} \sigma_i^2 \tag{8.2.13}$$

이 된다.

이제 $\boldsymbol{x}_i$의 가중치 합에 의해

$$y = \sum_{i=1}^{N} w_i \boldsymbol{x}_i \tag{8.2.14}$$

로 주어진 경우의 평균을 구하면

$$E[y] = E[\sum_{i=1}^{N} w_i \boldsymbol{x}_i] = \sum_{i=1}^{N} w_i E[\boldsymbol{x}_i] = \sum_{i=1}^{N} w_i m_i \tag{8.2.15}$$

이고, 분산은 식 (8.2.13)을 이용하면

$$\sigma_y^2 = E[(\sum_{i=1}^{N} (w_i \boldsymbol{x}_i - w_i m_i))^2] = \sum_{i=1}^{N} E[(w_i \boldsymbol{x}_i - w_i m_i)^2] = w_i^2 \sum_{i=1}^{N} \sigma_i^2 \tag{8.2.16}$$

으로 구해진다.

참조: 확률변수의 상관관계 계수

2 개의 확률변수 $\boldsymbol{x}$와 $\boldsymbol{y}$ 사이의 상관관계 계수(correlation coefficient)는

$$r_{xy} = \frac{E[\boldsymbol{xy}] - E[\boldsymbol{x}]E[\boldsymbol{y}]}{\sigma_x \sigma_y} \tag{8.2.17}$$

로 정의된다. 여기서,

$$C_{xy} = E[\boldsymbol{xy}] - E[\boldsymbol{x}]E[\boldsymbol{y}] \tag{8.2.18}$$

를 공분산이라고 한다. $r_{xy} = 0$이면 $\boldsymbol{x}$와 $\boldsymbol{y}$ 는 상관관계가 없다고 한다.

8.3. Bagging

- Bootstrapping method: generate training sets, train one base-learner with each, and combine them
  - From $\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$, generate $\mathcal{X}_l, l = 1, \ldots, L$ with $|\mathcal{X}_l| = N$ by sampling $1/N$ with replacement,
    - $\mathcal{X}_l$ are called bootstrap samples
  - Train L base-learners for $\mathcal{X}_l \Rightarrow g_l(\mathbf{x})$
  - Use voting (average or median with regression)

$$g_{\text{bag}}(x) = \frac{1}{L}\sum g_l(\mathbf{x})$$

- Why do we want to do this?
  - Simplified analysis: Let $g^*(\mathbf{x})$ be the best base-learner

$$\begin{aligned} E[(r^* - g^*(x))^2] &= E[(r^* - g_{\text{bag}}(x) + g_{\text{bag}}(x) - g^*(x))^2] \\ &= E[(r^* - g_{\text{bag}}(x))^2] + E[(g^* - g_{\text{bag}}(x))^2] \\ &\geq E[(r^* - g_{\text{bag}}(x))^2] \end{aligned}$$

    - Expectation done with regard to the input distribution
  - Unstable algorithms profit from bagging

Boosting

- Another way to make a non-linear classifier, this time by voting or averaging an ensemble or "committee" of very simple (or not-so-simple) classifiers. The strength of this approach is that the simple classifiers only need to have an error rate < 0.5 ; we can systematically train and combine a group of them to have low error.

- Basic algorithm is AdaBoost.m1. The idea is to train a classifier on the initial data set, notice which examples are misclassified, then train another weak classifier with the previously misclassified points emphasized in the training set. The job of the new classifier is to "handle" those points in some sense. This process continues, with new classifiers trained on a continually reweighted data set. The classifiers are then combined, with weights related to their performance.

AdaBoost: Algorithm

- Sequence of weak classifiers (or hypotheses)

$$G_k(x) \in \{-1, +1\} \quad k = 1, ..., L$$

- Final classifier is

$$F_M(x) = \sum_{m=1}^{M} \alpha_m G_{k_m}(x) \quad \rightarrow \hat{y} = \text{sign}(F_M(x)) \quad k_m \in \{1, ..., L\}$$

- Training algorithm

$$\text{Data} = \{< x_1, y_1 >, ..., < x_N, y_N >\} \quad (y \in \{-1, +1\})$$

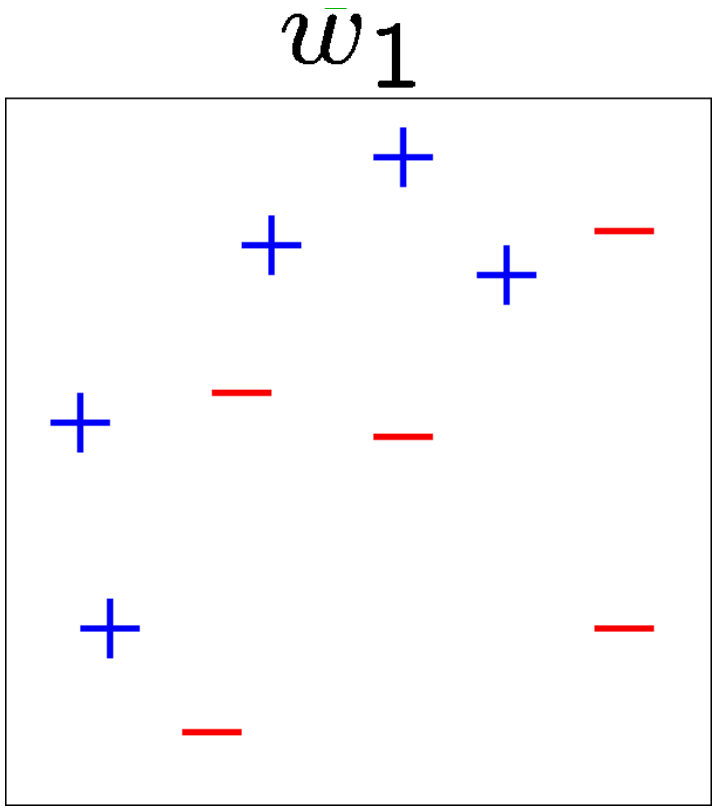
  - Initialize observation weights : $w_{1,i} = 1/N$
  - For $m = 1$ to M
    - Fit (or Select) a weak classifier $G_{k_m}(x)$ with small error

$$\text{err}_m = \sum_i w_{m,i} I(y_i \neq G_{k_m}(x_i)) \qquad \text{(should be } < 0.5\text{)}$$

    - Let $\alpha_m = \frac{1}{2} \log\left(\frac{1-\text{err}_m}{\text{err}_m}\right)$
      - Large positive where err_m is small.
      - Shouldn't be negative.
    - Set $w_{m+1,i} = \frac{w_{m,i} \cdot e^{-\alpha_m y_i G_{k_m}(x_i)}}{Z_m}$ for all i

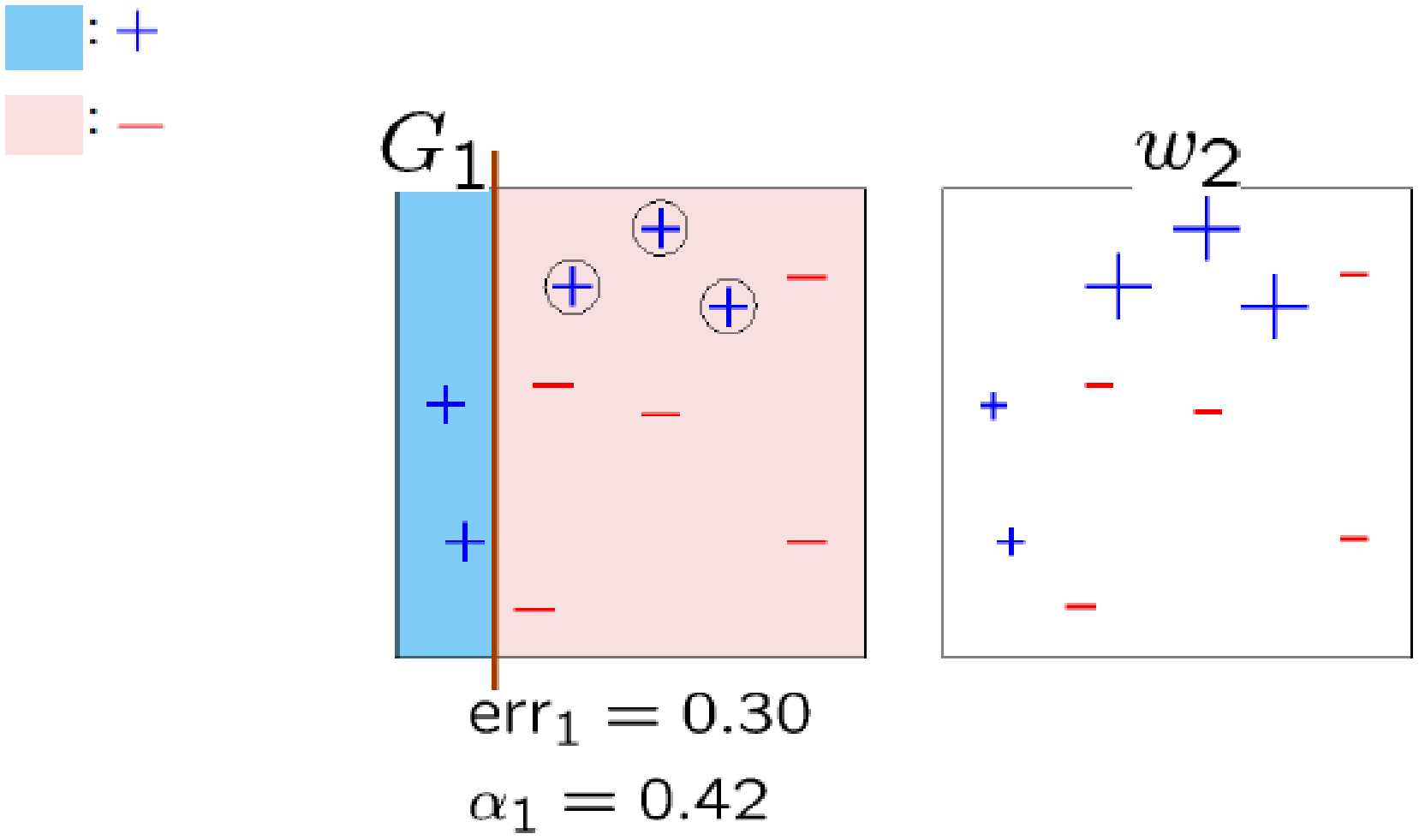
where $Z_m = \sum_i w_{m,i} \cdot e^{-\alpha_m y_i G_{k_m}(x_i)}$: normalization factor.

Toy Example

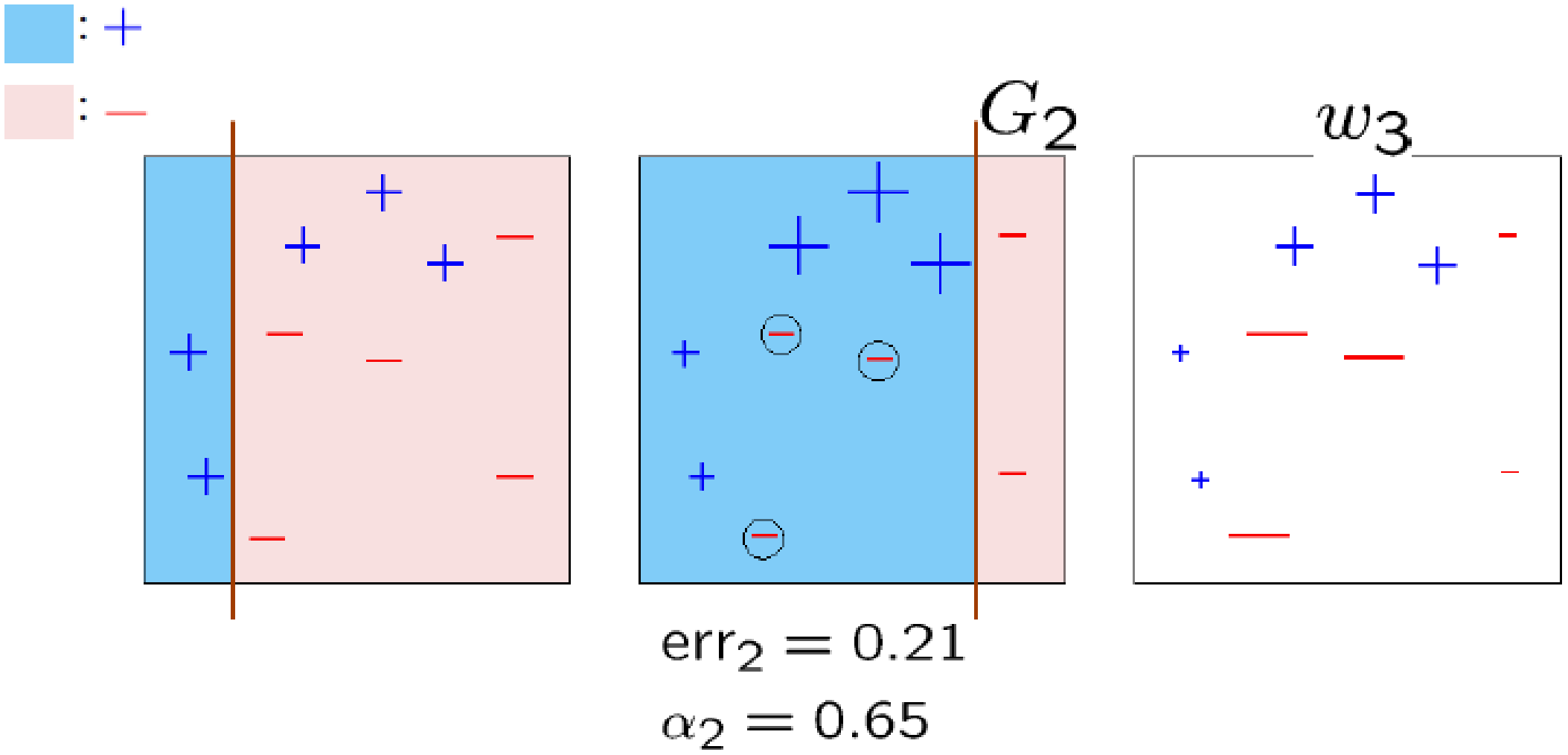


weak classifiers = vertical or horizontal half-planes

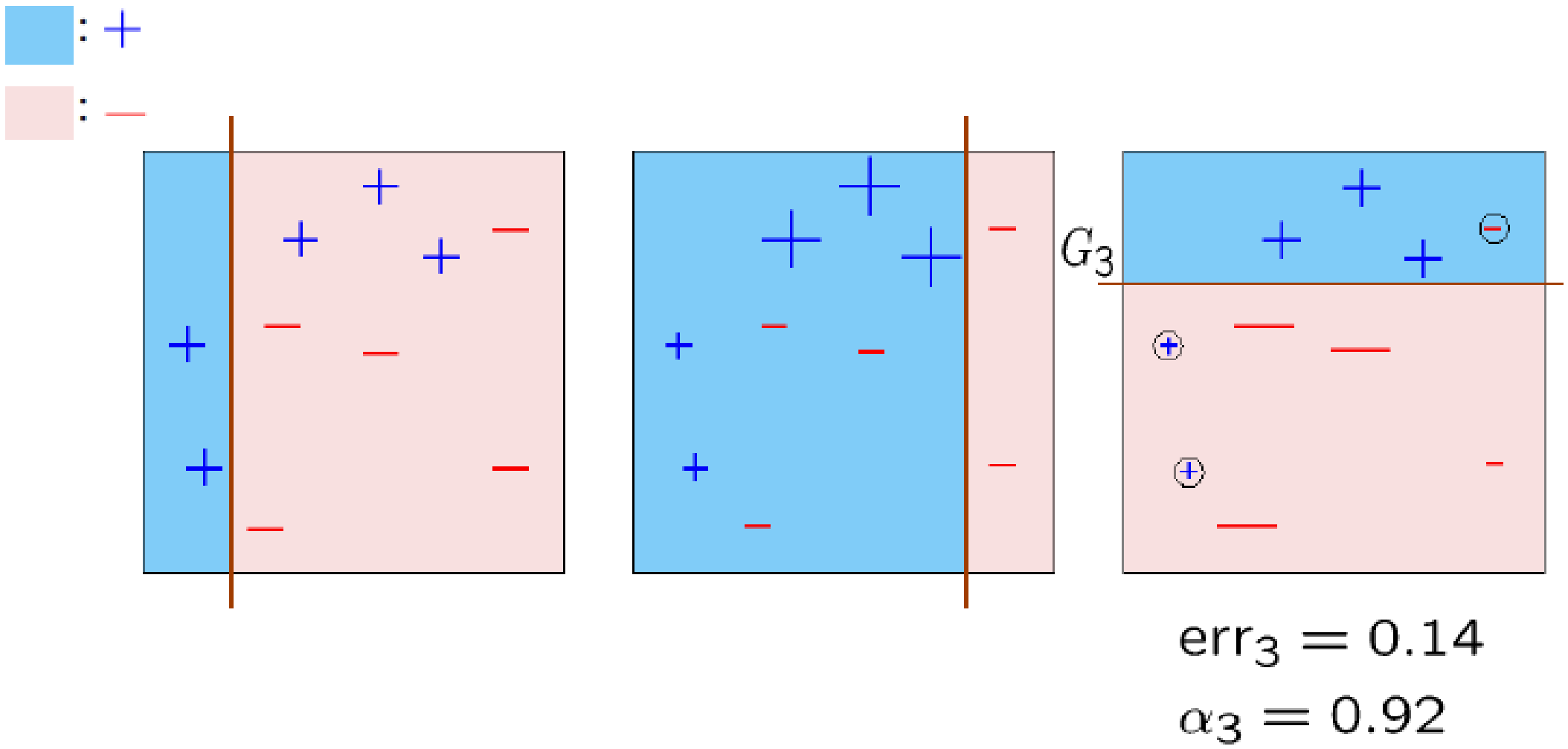
Round 1



Round 2



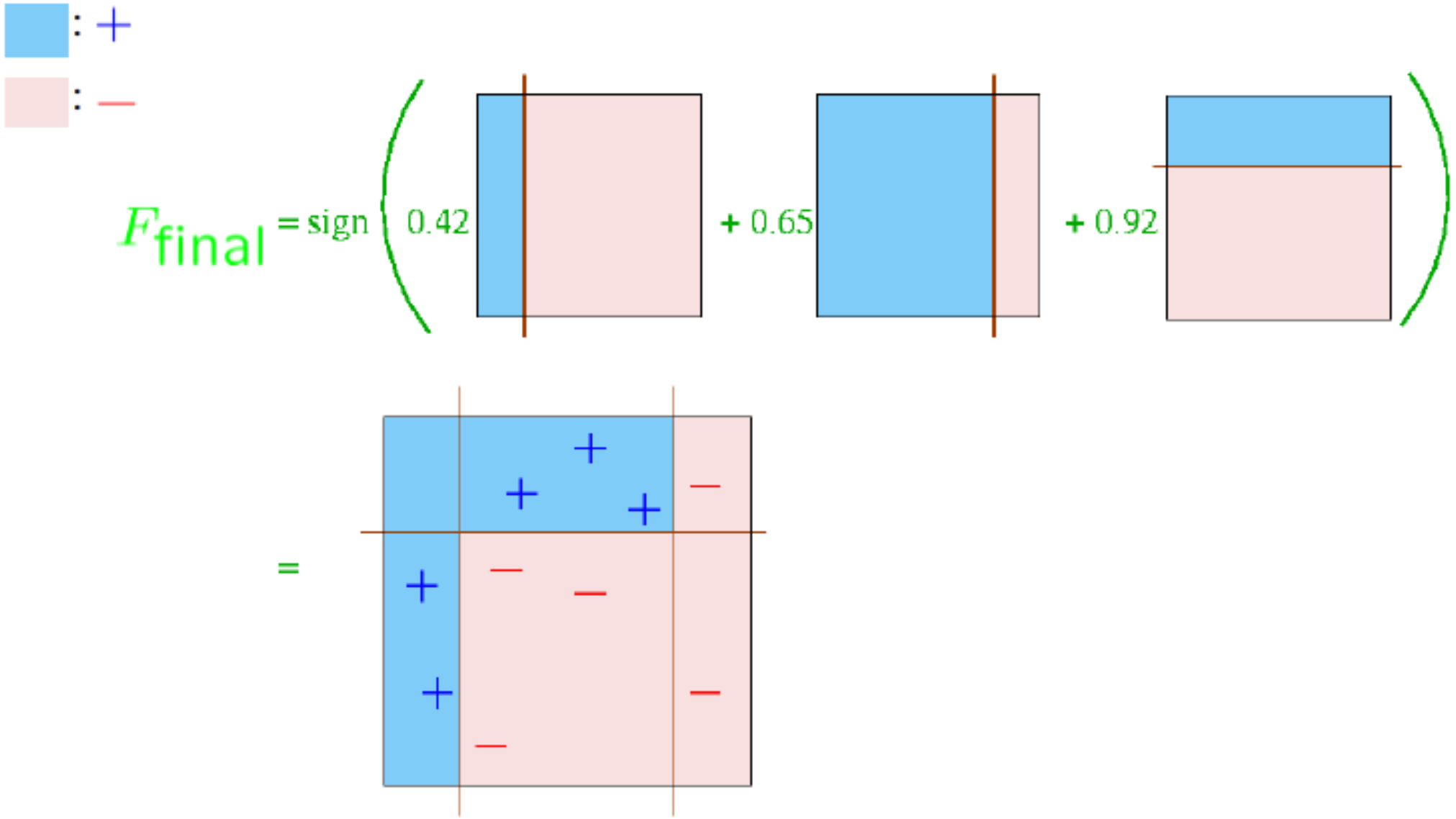
Round 3



$\text{err}_3 = 0.14$

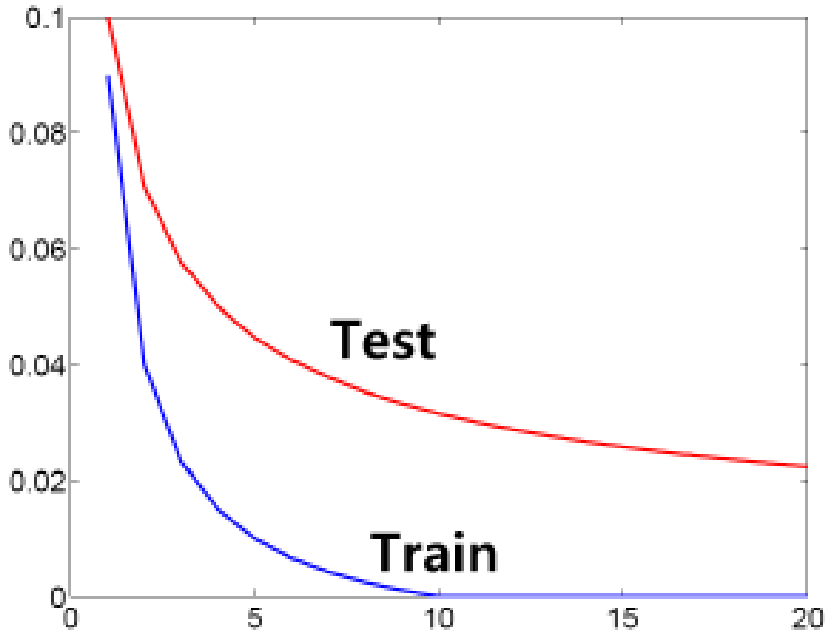
$\alpha_3 = 0.92$

Final Classifier



Test Error Behavior

- When should you stop? You can use cross validation or a held-out validation set. But experience shows that boosting usually doesn't overfit. Test error tends to keep decreasing as a function of m, even after training error has gone to 0!



Practical Advantages of AdaBoost

- Fast
- Simple and easy to program
- No parameters to tune (except M)
- Flexible – can combine with any learning algorithm
- No prior knowledge needed about weak learner
- Probably effective, provided can consistently find rough rules of thumb
  - Shift in mind set – goal now is merely to find classifiers barely better than random guessing
- Versatile
  - Can use with data that is textual, numeric, discrete, etc.
  - Has been extended to learning problems well beyond binary classification

8.4. Evaluation of Classifiers

- **"Accuracy" or "Recognition Ratio"**

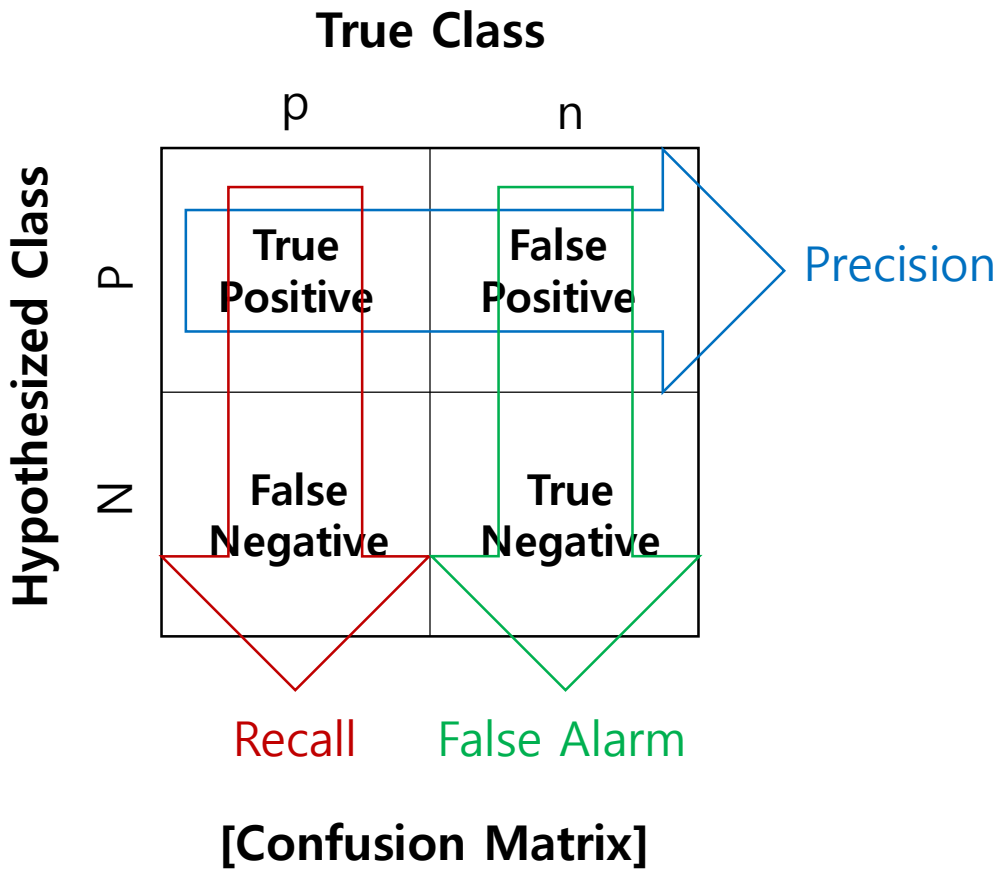
$$Accuracy = \frac{No.(Correctly\ Classified\ Samples)}{No.(Presented\ Samples)} \tag{8.4.1}$$

- **"Precision and Recall"**
  - TP : True Positive
  - FP : False Positive
  - TN : True Negative
  - FN : False Negative Positive

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$



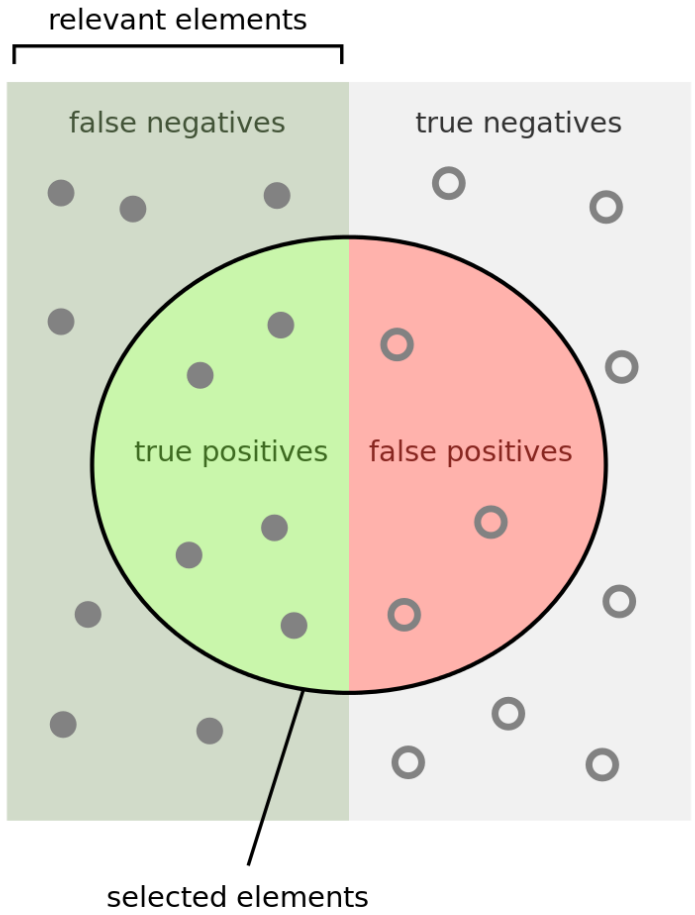
[Confusion Matrix]

- A precision score of 1.0 for a class C means that every item labeled as belonging to class C does indeed belong to class C (but says nothing about the number of items from class C that were not labeled correctly).

- A recall of 1.0 means that every item from class C was labeled as belonging to class C (but says nothing about how many other items were incorrectly also labeled as belonging to class C)

- An inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other.

- Brain surgery provides an illustrative example of the tradeoff. Consider a brain surgeon tasked with removing a cancerous tumor from a patient's brain. The surgeon needs to remove all of the tumor cells since any remaining cancer cells will regenerate the tumor. Conversely, the surgeon must not remove healthy brain cells since that would leave the patient with impaired brain function.
- The surgeon may be more liberal in the area of the brain he removes to ensure he has extracted all the cancer cells. This decision increases recall but reduces precision. On the other hand, the surgeon may be more conservative in the brain he removes to ensure he extracts only cancer cells. This decision increases precision but reduces recall.
- That is to say, greater recall increases the chances of removing healthy cells (negative outcome) and increases the chances of removing all cancer cells (positive outcome). Greater precision decreases the chances of removing healthy cells (positive outcome) but also decreases the chances of removing all cancer cells (negative outcome).



How many selected items are relevant?

$$\text{Precision} = \frac{\text{green}}{\text{green + red}}$$

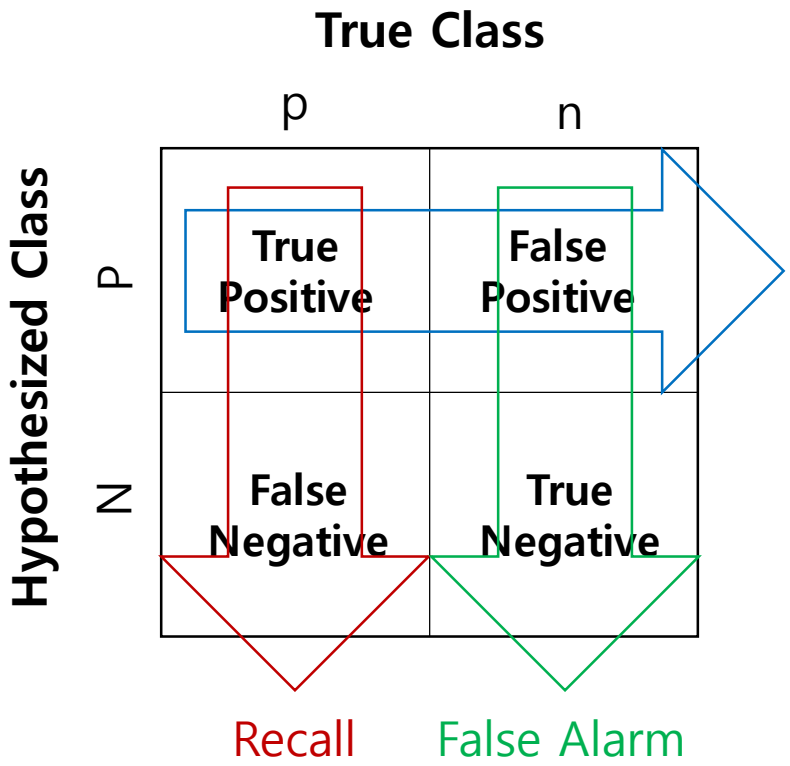
How many relevant items are selected?

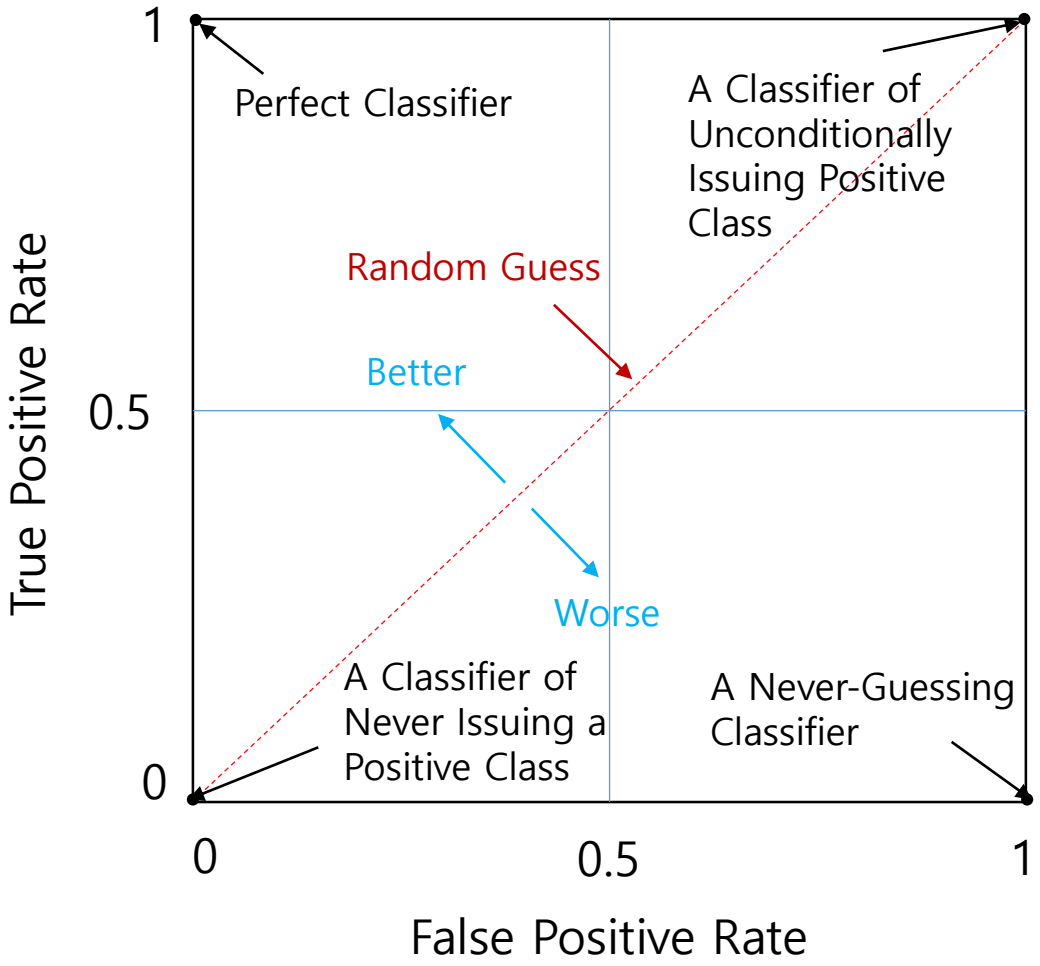
$$\text{Recall} = \frac{\text{green}}{\text{green + dark}}$$

[from Wikipedia]

- **ROC(Receiver Operating Characteristic)**
  - a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.
  - plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

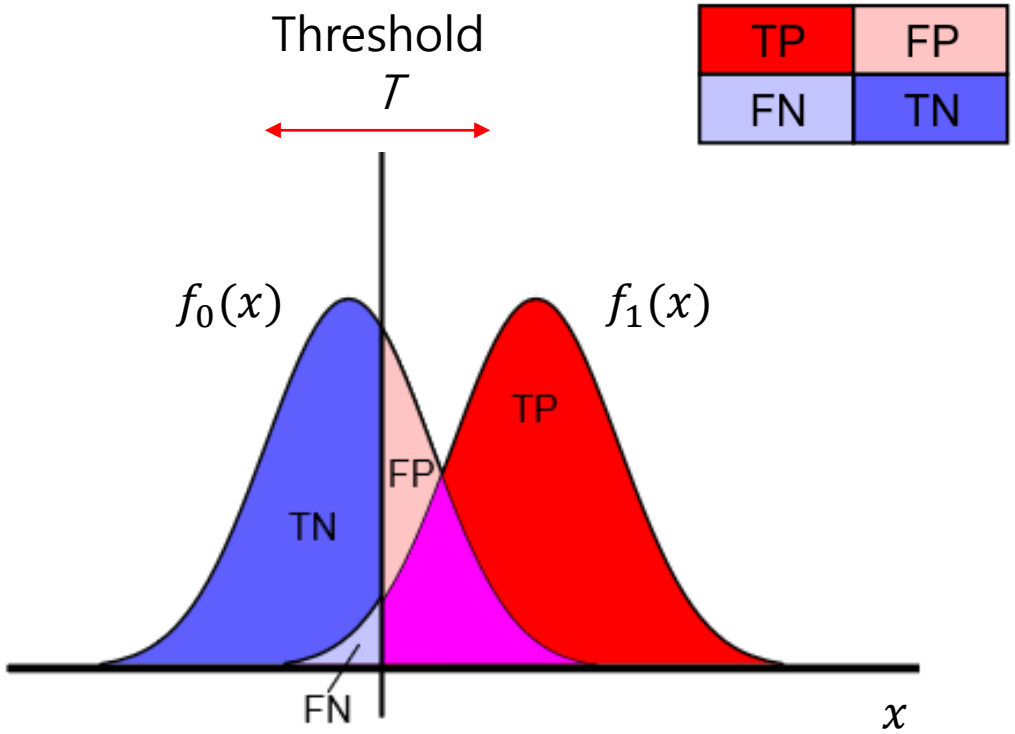
$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}$$

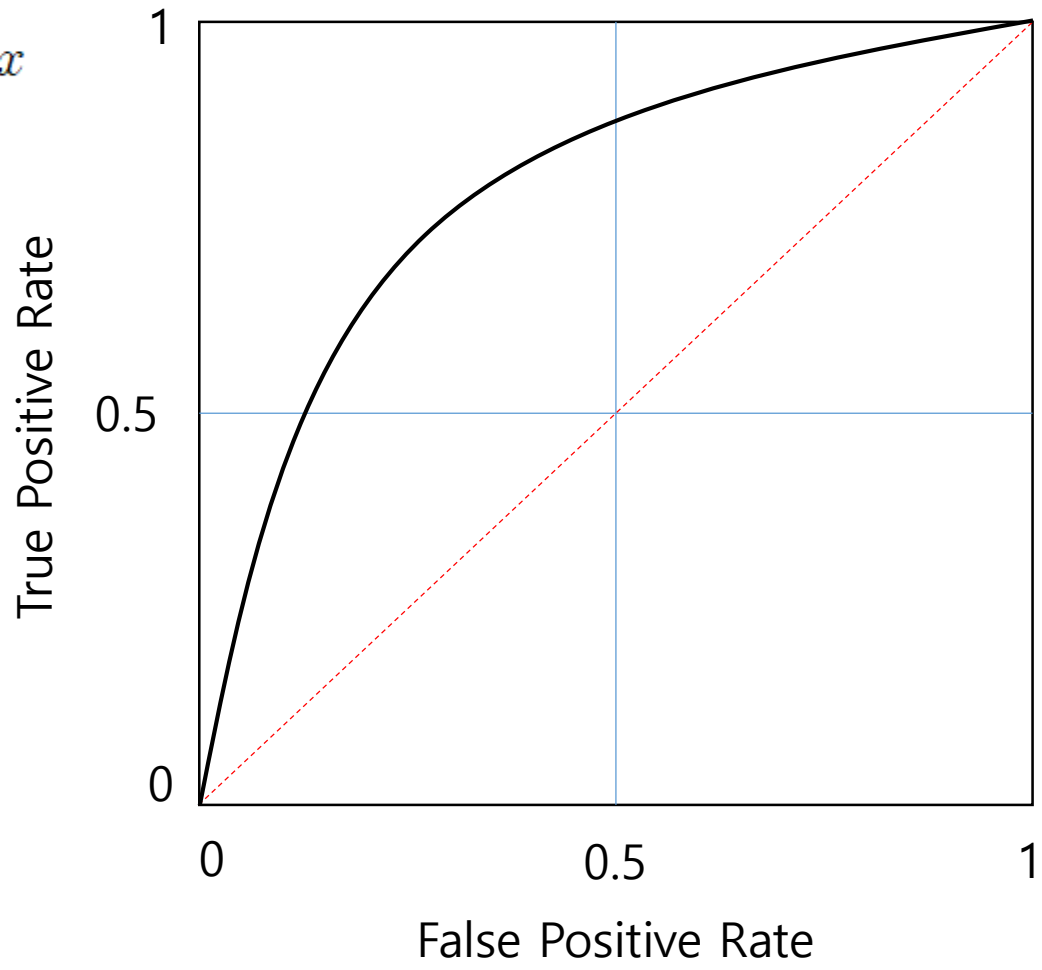




- **Curves in ROC Space : Useful for Imbalanced Data!**
  - Given a threshold T, the instance is classified as "positive" if $X > T$, and "negative" otherwise.
  - X follows a probability density $f_1(x)$ if the instance actually belongs to class "positive", and $f_0(x)$ if otherwise.

$$TPR(T) = \int_T^{\infty} f_1(x)dx \qquad FPR(T) = \int_T^{\infty} f_0(x)dx$$





예제 8.4-1

혼동행렬이 아래와 같이 주어진 경우에 정확도, 정밀도와 상기율, TPR, FPR을 계산하여라. 그리고, (FPR,TPR)을 ROC 공간상에 점으로 표시하라.

TP=80	FP=10
FN=20	TN=5

풀이

식 (8.4.4)에 의해 정확도는

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{85}{115} = 0.739$$

이다. 정밀도와 상기율은 식 (8.4.2)와 (8.4.3)에 의해

$$Precision = \frac{TP}{TP+FP} = \frac{80}{90} = 0.89, \quad Recall = \frac{TP}{TP+FN} = \frac{80}{85} = 0.94$$

이다. TPR은 식 (8.4.5)에서와 같이 상기율과 같은 값이고, FPR은 식 (8.4.6)에 의해

$$FPR = \frac{FP}{FP+TN} = \frac{10}{30} = 0.33$$

이다. 이를 ROC 공간상에 표시하면