# Boosting and Adaboost

CS 584 Data Mining (Spring 2022)

Prof. Sanmay Das

George Mason University

Slides are adapted from the available book slides developed by Tan, Steinbach, Karpatne, and Kumar

# Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of each boosting round

- Originally arose as a solution to a theoretical puzzle:
  - Given a *weak* base learner (one that can only do a little better than chance on the training data), can we *boost* it into a strong learner?

# 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

# AdaBoost

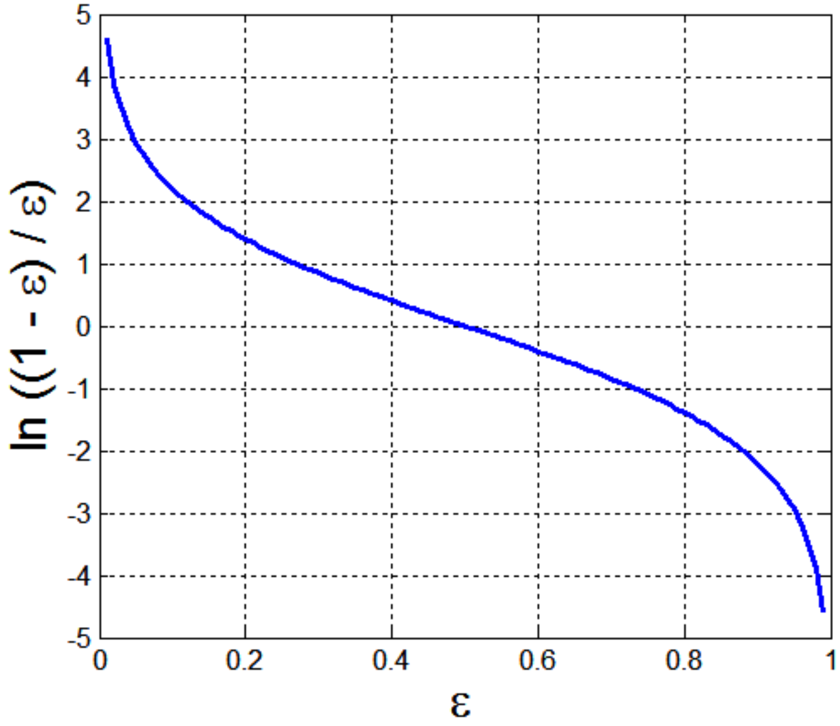
- Base classifiers: $C_1, C_2, \ldots, C_T$

- Error rate:

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

- Importance of a classifier:

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



# AdaBoost Algorithm

- Weight update:

$$w_j^{(i+1)} = \frac{w_j^{(i)}}{Z_i} \begin{cases} \exp^{-\alpha_i} & \text{if } C_i(x_j) = y_j \\ \exp^{\alpha_i} & \text{if } C_i(x_j) \neq y_j \end{cases}$$

$$\text{where } Z_i \text{ is the normalization factor}$$

- If any intermediate rounds produce error rate higher than 50%, the weights are reverted back to 1/n and the resampling procedure is repeated
- Classification:

$$C^*(x) = \underset{y}{\operatorname{argmax}} \sum_{i=1}^{T} \alpha_i \delta\big(C_i(x) = y\big)$$

# AdaBoost Algorithm

**Algorithm 5.7** AdaBoost Algorithm

1: $\mathbf{w} = \{w_j = 1/n \mid j = 1, 2, \cdots, n\}$. {Initialize the weights for all $n$ instances.}
2: Let $k$ be the number of boosting rounds.
3: **for** $i = 1$ to $k$ **do**
4: Create training set $D_i$ by sampling (with replacement) from $D$ according to $\mathbf{w}$.
5: Train a base classifier $C_i$ on $D_i$.
6: Apply $C_i$ to all instances in the original training set, $D$.
7: $\epsilon_i = \frac{1}{n}\left[\sum_j w_j \; \delta\big(C_i(x_j) \neq y_j\big)\right]$ {Calculate the weighted error}
8: **if** $\epsilon_i > 0.5$ **then**
9: $\mathbf{w} = \{w_j = 1/n \mid j = 1, 2, \cdots, n\}$. {Reset the weights for all $n$ instances.}
10: Go back to Step 4.
11: **end if**
12: $\alpha_i = \frac{1}{2} \ln \frac{1-\epsilon_i}{\epsilon_i}$.
13: Update the weight of each instance according to equation (5.88).
14: **end for**
15: $C^*(\mathbf{x}) = \arg\max_y \sum_{j=1}^{T} \alpha_j \delta\big(C_j(\mathbf{x}) = y\big)\big)$.

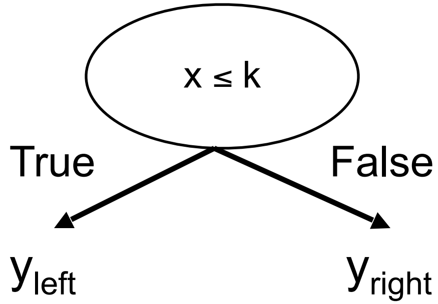
# AdaBoost Example

- Consider 1-dimensional data set:

**Original Data:**

| x | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| y | 1 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 |

- Classifier is a decision stump
  - Decision rule: $x \leq k$ versus $x > k$
  - Split point k is chosen based on entropy



# AdaBoost Example

- Training sets for the first 3 boosting rounds:

Boosting Round 1:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| x | 0.1 | 0.4 | 0.5 | 0.6 | 0.6 | 0.7 | 0.7 | 0.7 | 0.8 | 1 |
| y | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 |

Boosting Round 2:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| x | 0.1 | 0.1 | 0.2 | 0.2 | 0.2 | 0.2 | 0.3 | 0.3 | 0.3 | 0.3 |
| y | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Boosting Round 3:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| x | 0.2 | 0.2 | 0.4 | 0.4 | 0.4 | 0.4 | 0.5 | 0.6 | 0.6 | 0.7 |
| y | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

- Summary:

| Round | Split Point | Left Class | Right Class | alpha |
|---|---|---|---|---|
| 1 | 0.75 | -1 | 1 | 1.738 |
| 2 | 0.05 | 1 | 1 | 2.7784 |
| 3 | 0.3 | 1 | -1 | 4.1195 |

# AdaBoost Example

- Weights

| Round | x=0.1 | x=0.2 | x=0.3 | x=0.4 | x=0.5 | x=0.6 | x=0.7 | x=0.8 | x=0.9 | x=1.0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| 2 | 0.311 | 0.311 | 0.311 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| 3 | 0.029 | 0.029 | 0.029 | 0.228 | 0.228 | 0.228 | 0.228 | 0.009 | 0.009 | 0.009 |

- Classification

| Round | x=0.1 | x=0.2 | x=0.3 | x=0.4 | x=0.5 | x=0.6 | x=0.7 | x=0.8 | x=0.9 | x=1.0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| Sum | 5.16 | 5.16 | 5.16 | -3.08 | -3.08 | -3.08 | -3.08 | 0.397 | 0.397 | 0.397 |
| Sign | 1 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 |

Predicted Class

# Recent Developments

- State of the art is *gradient boosting*, which is motivated in the same way (adding weak models together), but can also be thought of as an additive model
- Gradient boosting and random forests are the two off-the-shelf classifiers to try first on almost anything!