Data 2060 Final Presentation
by AI Alchemist

AdaBoosting

Agenda

01

Math behind Machine Learning algorithm

02

Reproducing Previous Work

03

Pseudo-code for Algorithm

04

Our Implementation

01

Math behind Machine Learning algorithm

Representation

The final prediction combines weighted outputs of all decision stumps:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Where:

1. T: Total number of decision stumps (weak classifiers).
2. α□: Weight of the t-th decision stump
3. h□(x): Prediction of the t-th decision stump
4. x: Sample from training set

Representation

The weight of each weak learner is calculated as

$$\alpha_t = 0.5 * \log((1 - \varepsilon_t)/\varepsilon_t)$$

Where:

ε□ is the error rate of weak learner at iteration t

$$\varepsilon_t = \Sigma\left[w_i * I(\, h_t(x_i) \neq y_i)\right]$$

Loss

The exponential loss is minimized iteratively by focusing on misclassified samples

$$L = \Sigma \exp\left(-y_i * H\left(x_i\right)\right)$$

Where:

1. y_i: True label of sample i
2. $H(x_i)$: Combined prediction from all weak learners.

Optimizer

Weights for each sample are updated to emphasize misclassified points:

$$\mathrm{w_i}(\mathrm{t}+1)=\left[\mathrm{w_i}(\mathrm{t}) * \exp\left(-\alpha_{\mathrm{t}} * \mathrm{y_i} * \ \mathrm{h_t}\left(\mathrm{x_i}\right)\right)\right]/\mathrm{z_t}$$

Where:

Z□: Normalization factor to maintain a valid distribution

02

Pseudo-code for Algorithm

2. Pseudo-code for Algorithm

Input: Training set S = {(x1, y1), (x2, y2), ..., (xm, ym)}, weak learner WL, number of rounds T
Output: Final hypothesis H(x)

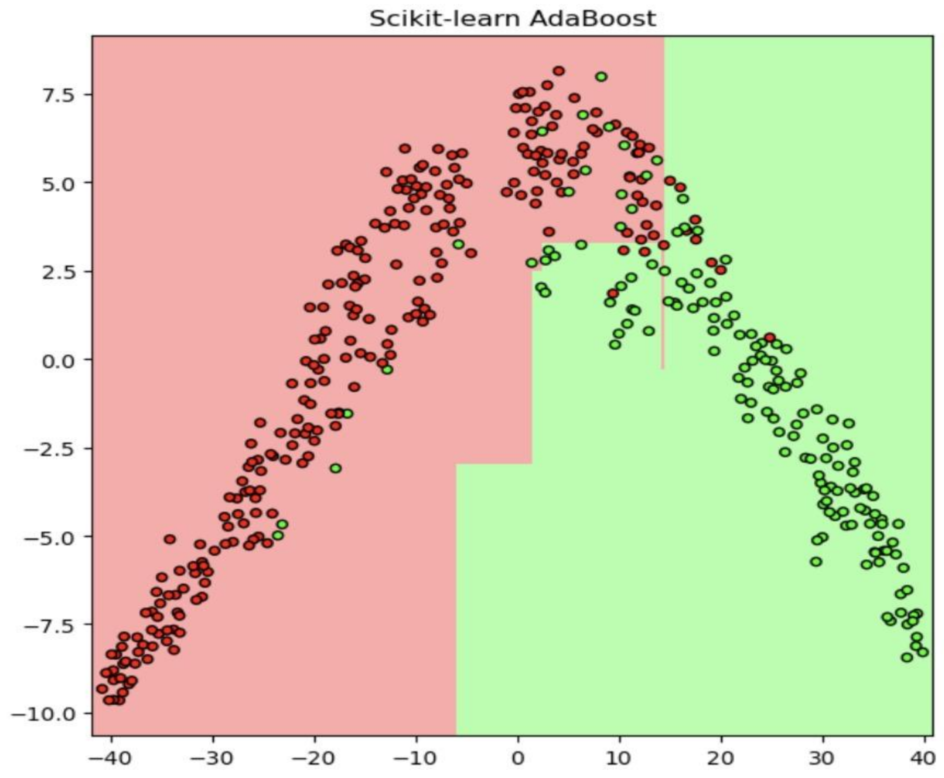
```
Initialize distribution D1(i) = 1/m for all i = 1 to m

for t = 1 to T:
    1. Train weak learner h_t using distribution Dt
    2. Calculate error ε_t = sum(Dt(i) * [h_t(x_i)  !=  y_i]) for all i
    3. Compute α_t = 0.5 * log((1 - ε_t) / ε_t)
    4. Update distribution:
       Dt+1(i) = Dt(i) * exp(-α_t * y_i * h_t(x_i))
       Normalize Dt+1 to maintain a probability distribution

Output final hypothesis:
    H(x) = sign(sum(α_t * h_t(x) for t = 1 to T))
```

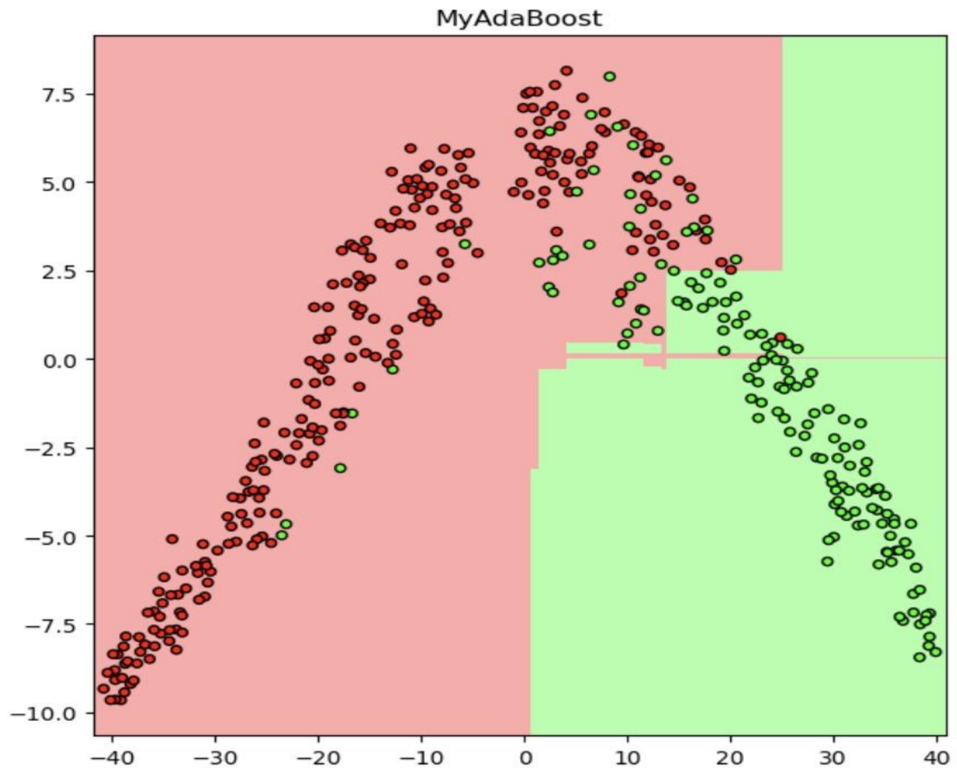
03

Reproducing Previous Work

3. Reproducing Previous Work

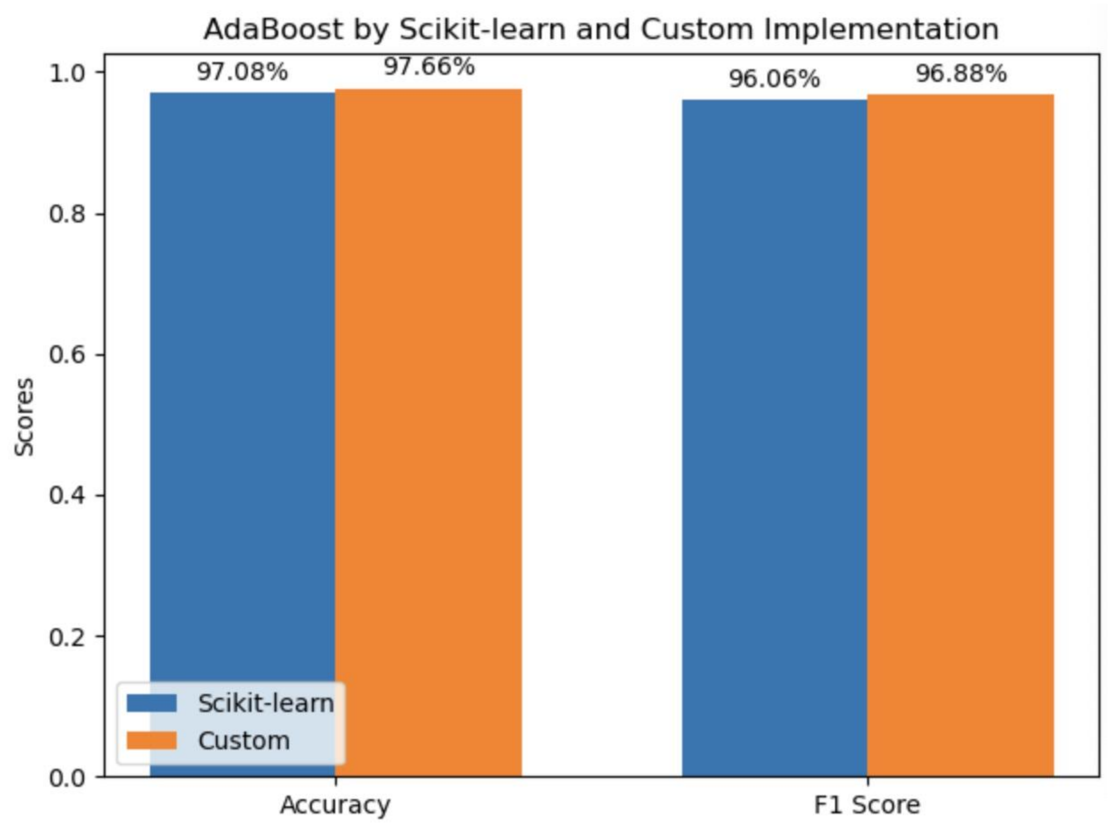
1. **Breast Cancer Wisconsin (Breast) Dataset**
2. **MyAdaBoost vs Scikit-learn's AdaBoost**
3. **Decision boundary visualized by t-SNE**





3. Reproducing Previous Work

1. **Breast Cancer Wisconsin (Breast) Dataset**
2. **MyAdaBoost vs Scikit-learn's AdaBoost**



MyAdaBoost: Slightly higher accuracy and F1 score

04

Our Implementation

4. Our Implementation - Overview

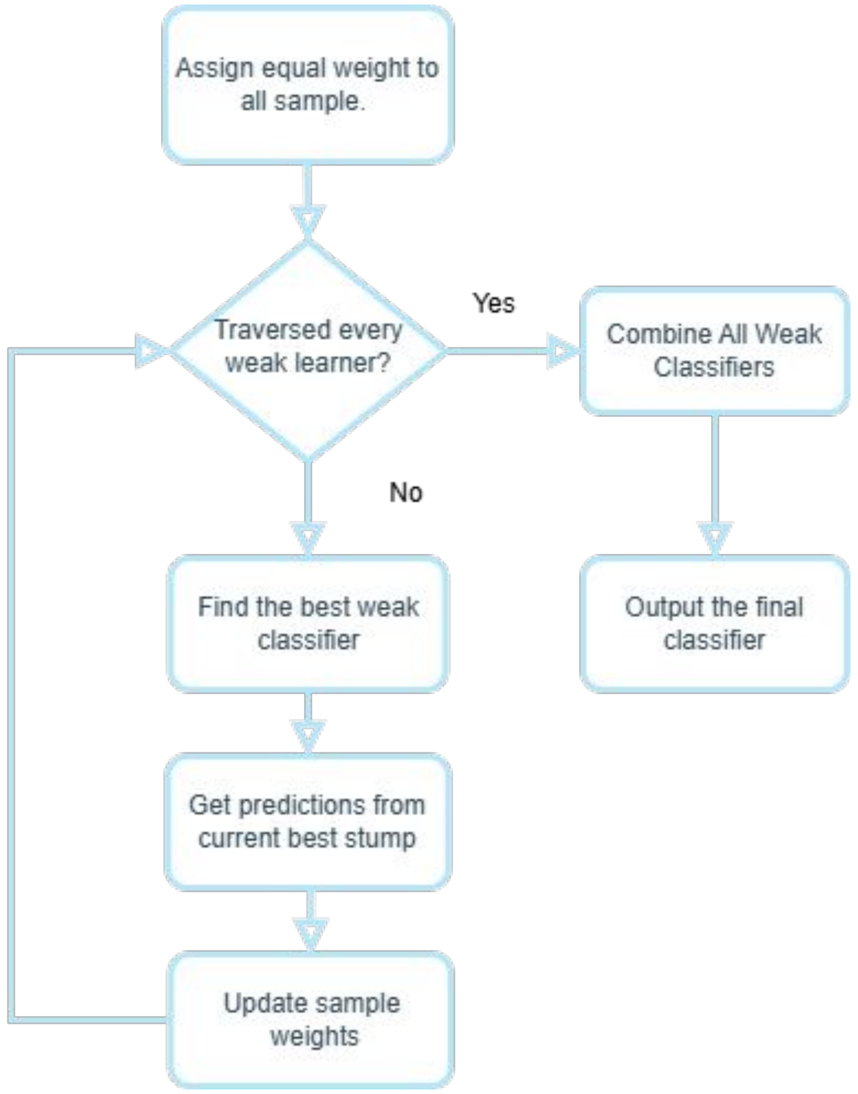
1. Input the dataset and weak

2. Assign equal weights

3. Traverse every weak learner
    (1) Find best classifier
    (2) Decision stump prediction
    (3) Update Sample weights

4. Combine all weak learners

5. Output the Final Classifier


Assign equal weight to all sample.
Traversed every weak learner?
Yes
Combine All Weak Classifiers
No
Find the best weak classifier
Output the final classifier
Get predictions from current best stump
Update sample weights

4. Our Implementation - Interesting Things

1. Dynamic Weight Adjustment:

- Adapts sample weights to focus on harder-to-classify instances.

2. Error-Based Weighting:

- Weak classifiers are weighted by their performance

3. Iterative Learning Process:

- Sequentially reduces error by emphasizing misclassified samples.

4. Our Implementation - Challenges

1. Weight Updates and Normalization:

- **Challenge:** Updating sample weights involves exponential operations, which can result in numerical instability.
- **Solution:** Ensure a valid distribution by normalizing the weights through division by the total sum of the sample weights.

2. Error Handling in α Calculation:

- **Challenge:** When error comes to 0 or 1, the calculation of α (classifier weight) can lead to dividing by zero problems.
- **Solution:** Introducing a small epsilon (1×10^{-10})

Thank You!