Introduction to Data Mining

Lecture10 Adaboost

Jun Huang

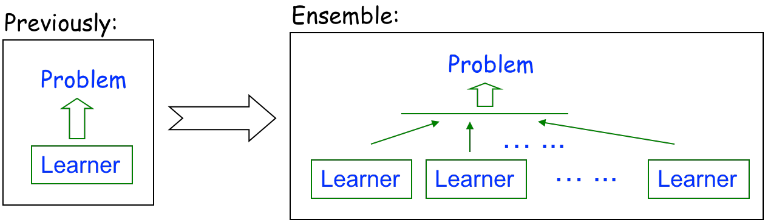
Anhui University of Technology

Spring 2018

huangjun_cs@163.com

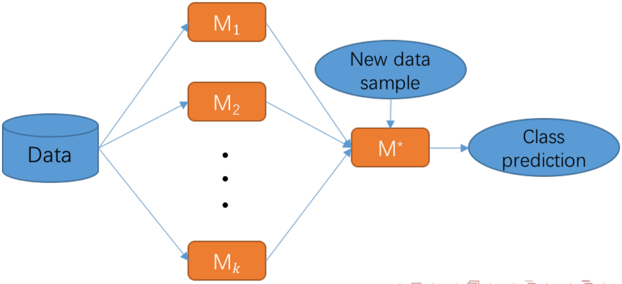
How to improve the generalization ability of machine learning learners?

- **Ensemble learning**
- A machine learning paradigm where multiple learners are used to solve the problem
- The generalization ability of the ensemble is usually significantly better than that of an individual learner
- Boosting is one of the most important families of ensemble methods



Ensemble Methods:Increasing the Accuracy
Bagging and Boosting

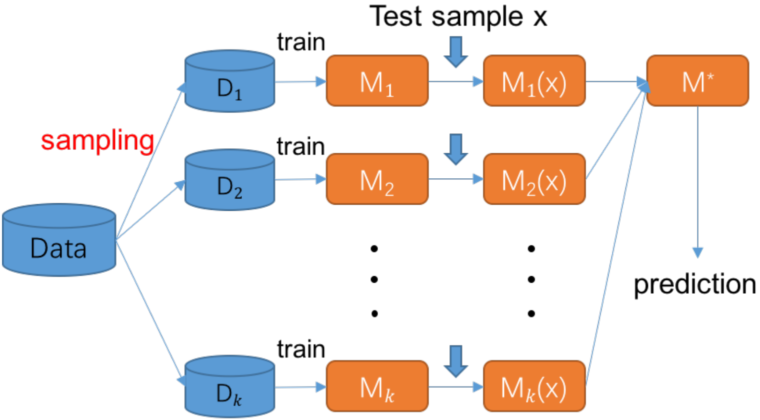
- Ensemble methods
  - Use a combination of models to increase accuracy
  - Combine a series of k learned models, M_1, M_2, ...,M_k, with the aim of creating an improved model M^*
- Popular ensemble methods
  - Bagging
  - Boosting



Bagging: Bootstrap Aggregation

- Analogy:Diagnosis based on multiple doctors' majority vote
- Training
  - Give a data set $\mathcal{D}$ of N samples, at each iteration i, a training set $\mathcal{D}_i$ is sampled with replacement from $\mathcal{D}$
  - A classifier model M_i is learned for each training set $\mathcal{D}_i$
- Classification: classify an unknown data sample X
  - Each classifier M_i returns its class prediction
  - The bagged classifier M^* counts the votes and assigns the class with the most votes to X

Bagging: Bootstrap Aggregation



- $M^*(x) = \text{maxcount}_t M_t(x)$

Bagging: Bootstrap Aggregation

- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
  - Often significant better than a single classifier derived from $\mathcal{D}$
  - For noise data: not considerably worse, more robust
  - Proved improved accuracy in prediction

Exercise

Following is a data set to construct a bagging classifier

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	1	1	1	-1	-1	-1	-1	1	1	1

Examples chosen for training in each round are shown below:

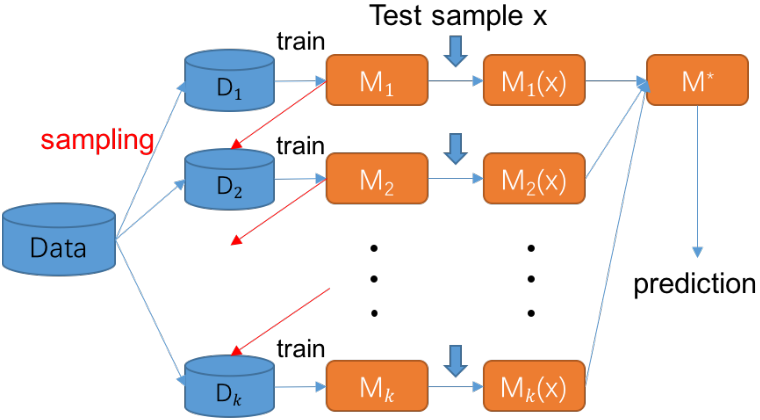
x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	-1	- 1
Classifier: ① x <= 0.35 -> y=1, ② x>0.35 -> y = -1										
x	0.1	0.2	0.3	0.5	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1
Classifier: ① 0.4<= x <=0.55 -> y=-1, ② x>0.55 -> y=1, ③ x<0.4 -> y=1										
x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1
Classifier: ① x<=0.35 -> y=1, ② 0.35<=x<=0.75 -> y=-1, ③ x>0.75 -> y=1										

Please predict the class label for x = 0.38 ?

Boosting

- Analogy: Consult several doctors, based on a combination of weighted diagnoses - weight assigned based on the previous diagnosis accuracy
- How boosting works?
  - After a classifier M_i is learned, the weights are updated to allow the subsequent classifier M_{i+1} pay more attention to the training tuples that were misclassified by M_i
  - A series of k classifiers are iteratively learned
  - The final M^* combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy

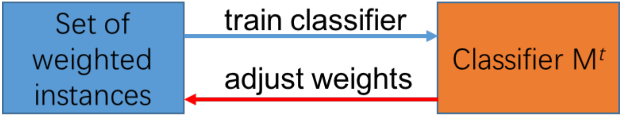
Boosting



- $\mathrm{M}^*(\mathrm{x}) = \mathrm{argmax}_{M_c} \sum_t^k w_t M_t(x)$

Boosting

- The boosting algorithm can be extended for the prediction of continuous values
- Comparing with bagging: boosting tends to achieve greater accuracy, but it also risks overfitting the model to the misclassified data



Bagging vs. Boosting

- Model training:
  - Bagging: random sampling, independent classifiers
  - Boosting: subsequent classifier M_{i+1} pay more attention to the training tuples that were misclassified by M_i
- Model usage:
  - Bagging: equal weight
  - Boosting: different weights assigned

Boosting

- **Significant advantageous**:
  - Solid theoretical foundation
  - Very accurate prediction
  - Very simple ("just 10 lines of code" [R. Schapire])
  - Wide and successful applications
  - …… …
- R. Schapire and Y. Freund won **the 2003 Godel Prize** (one of the most prestigious awards in theoretical computer science)
  - Prize winning paper (which introduced AdaBoost): "A decision theoretic generalization of on-line learning and an application to Boosting, "Journal of Computer and System Sciences, 1997, 55: 119-139.

Random Forest

Tree bagging

- Given a training set $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$, $x_i \in \mathbb{R}^d$, and y_i is the corresponding class label.
- The procedures of Tree bagging is summarized as following:
  1. For $b = 1$ *to* B
  2. Sample, with replacement, n training examples from $\mathcal{D}$, call $\mathcal{D}_b = \{x_i, y_i\}_{i=1}^n$;
  3. Train a classification or regression tree f_b on $\mathcal{D}_b$;
  4. End
- Predictions for unseen samples x' can be made by taking the majority vote in the case of classification trees.
- or by averaging the predictions from all the individual regression trees on x'

$$\hat{f} = \frac{1}{B} f_b(x')$$

Random Forest

- Random forests differ in only one way from Tree Bagging
  - They use a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features.

1. For $b = 1$ *to* B
2. Sample, with replacement, n training examples **with p features** from $\mathcal{D}$, call $\mathcal{D}_b = \{x_i, y_i\}_{i=1}^n, x_i \in \mathbb{R}^p$;
3. Train a classification or regression tree f_b on $\mathcal{D}_b$;
4. End

- Typically, for a classification problem with d features, $\sqrt{d}$ (rounded down) features are used in each split.
- For regression problems the inventors recommend $d/3$ (rounded down) with a minimum node size of 5 as the default. (The Elements of Statistical Learning, 2nd ed.)

Random Forest

- Decision trees are a popular method for various machine learning tasks. Tree learning comes closest to meeting the requirements for serving as an off-the-shelf procedure for data mining
- It is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features
- Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks
- Random decision forests correct for decision trees' habit of overfitting to their training set

A formal decription of Boosting

- given training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m)\}$
- $y_i \in \{0, 1\}$ correct label of instance $\mathbf{x}_i \in \mathcal{X}$
- for $t = 1, ..., T$
  - construct distribution $\mathcal{D}_t$ on $\{1, ..., m\}$
  - find weak classifier

$$h_t : \mathcal{X} \to \{-1, +1\}$$

  - with samll error ϵ_t on $\mathcal{D}_t$

$$\epsilon_t = \mathsf{Pr}_{i\sim\mathcal{D}_t}[h_t(\mathbf{x}_i) \neq y_i]$$

- output final classifier H_{final}

AdaBoost

- constructing $\mathcal{D}_t$:
  - $\mathcal{D}_1(i) = \frac{1}{m}$
  - given $\mathcal{D}_t$ and h_t:

$$\begin{aligned}\mathcal{D}_{t+1}(i) &= \frac{\mathcal{D}_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & \text{if } y_i = h_t(\mathbf{x}_i) \\ e^{-\alpha_t}, & \text{if } y_i \neq h_t(\mathbf{x}_i) \end{cases} \\ &= \frac{\mathcal{D}_t(i)}{Z_t} e^{-\alpha_t y_i h_t(\mathbf{x}_i)}\end{aligned}$$

  where

$$\begin{aligned} Z_t &= \text{normalization constant} \\ \alpha_t &= \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) > 0 \end{aligned}$$

- final classifier

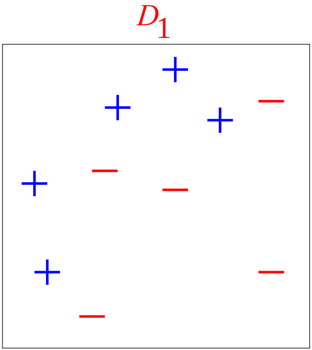
$$H_{\text{final}}(\mathbf{x}) = \text{sign}\left(\sum_t \alpha_t h_t(\mathbf{x})\right)$$

Adaboost (Adaptive Boost) Algorithm

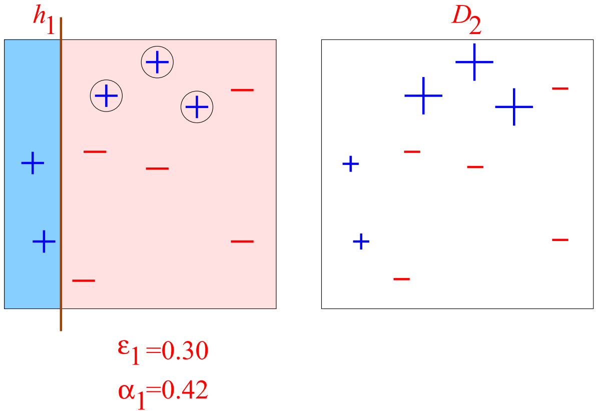
1. **Input**: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m)\}$, , T rounds, base learner $\mathcal{L}$
2. **Output**: $H_{\text{final}}(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right)$
3. $\mathcal{D}_1(i) = \frac{1}{m}, 1 \leq i \leq m$
4. **for** $t = 1, ..., T$
5. $h_t = \mathcal{L}(\mathcal{D}, \mathcal{D}_t)$
6. $\epsilon_t = \Pr_{i \sim \mathcal{D}_t}[h_t(\mathbf{x}_i) \neq y_i]$
7. **if** $\epsilon_t > 0.5$, **then** break
8. $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
9. $$\mathcal{D}_{t+1}(i) = \frac{\mathcal{D}_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & \text{if } y_i = h_t(\mathbf{x}_i) \\ e^{-\alpha_t}, & \text{if } y_i \neq h_t(\mathbf{x}_i) \end{cases} = \frac{\mathcal{D}_t(i)}{Z_t} e^{-\alpha_t y_i h_t(\mathbf{x}_i)}$$
10. **end**

Toy Example

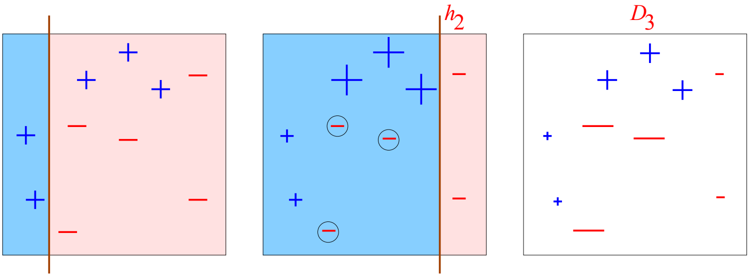
- weak classifiers = vertical or horizontal half-planes



Round1



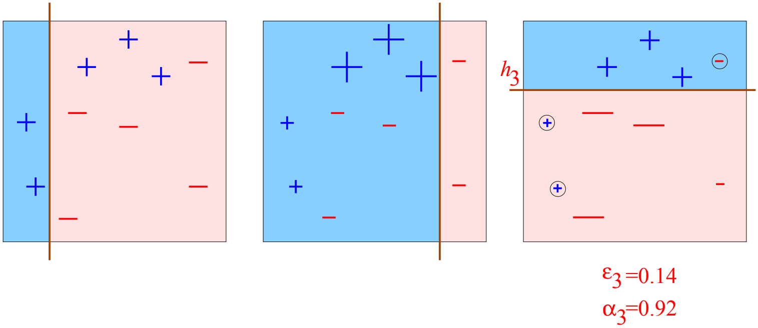
Round2



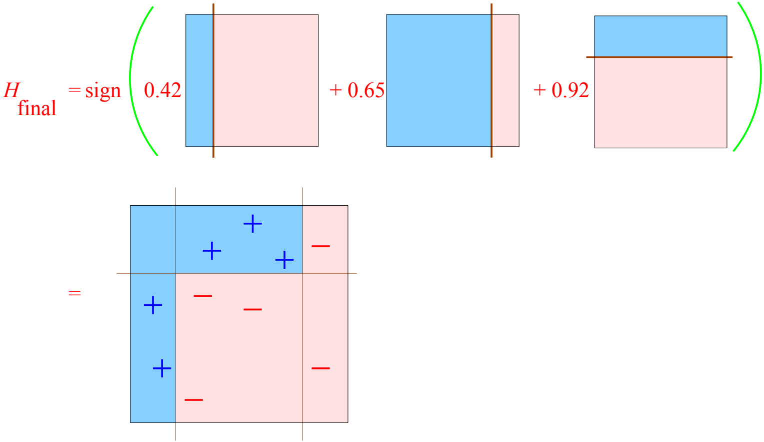
$\varepsilon_2=0.21$

$\alpha_2=0.65$

Round3



Final Classifier



Analyzing the training error

- Theorem:
  - write ϵ_t as $\frac{1}{2} - \gamma_t$
  - then

$$
\begin{aligned}
\text{training error}(H_{\text{final}}) &\leq \prod_t \left[2\sqrt{\epsilon_t(1-\epsilon_t)}\right] \\
&= \prod_t \sqrt{1-4\gamma_t^2} \\
&\leq \exp\left(-2\sum_t \gamma_t^2\right)
\end{aligned}
$$

- so, if $\forall t : \gamma_t \geq \gamma > 0$, then training error$(H_{\text{final}}) \leq e^{-2\gamma^2 T}$
- AdaBoost is adaptive:
  - does not need to know γ or T apriori
  - can exploit $\gamma_t \gg \gamma$

Proof

- Let $f(\mathbf{x}) = \sum_t \alpha_t h_t(\mathbf{x}) \Rightarrow H_{\text{final}}(\mathbf{x}) = \text{sign}(f(\mathbf{x}))$
- Step 1: unwrapping recurrence:

$$\begin{aligned}\mathcal{D}_{\text{final}}(i) &= \frac{1}{m}\frac{\exp(-y_i \sum_t \alpha_t h_t(\mathbf{x}_i))}{\prod_t Z_t} \\ &= \frac{1}{m}\frac{\exp(-y_i f(\mathbf{x}_i))}{\prod_t Z_t}\end{aligned}$$

Proof (cont.)

- Step 2: training error$(H_{\text{final}}) \leq \prod_t Z_t$
- proof:

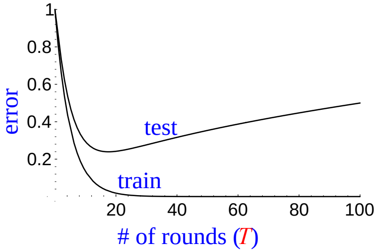
$$
\begin{aligned}
\text{training error}(H_{\text{final}}) &= \frac{1}{m}\sum_i \begin{cases} 1, & \text{if } y_i \neq H_{\text{final}(\mathbf{x}_i)} \\ 0, & \text{else} \end{cases} \\
&= \frac{1}{m}\sum_i \begin{cases} 1, & \text{if } y_i f(\mathbf{x}_i) \leq 0 \\ 0, & \text{else} \end{cases} \\
&\leq \frac{1}{m}\sum_i \exp(-y_i f(\mathbf{x}_i)) \\
&= \sum_i \mathcal{D}_{\text{final}}(i) \prod_t Z_t \\
&= \prod_t Z_t
\end{aligned}
$$

Proof (cont.)

- Step 3: $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$
- Proof:

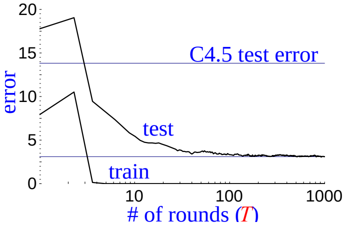
$$
\begin{aligned}
Z_t &= \sum_i \mathcal{D}_t(i) \exp(-\alpha_t y_i h_t(\mathbf{x}_t)) \\
&= \sum_{i: y_i \neq h_t(\mathbf{x}_t)} \mathcal{D}_t(i) e^{\alpha_t} + \sum_{i: y_i = h_t(\mathbf{x}_t)} \mathcal{D}_t(i) e^{\alpha_t} \\
&= \epsilon_t e^{\alpha_t} + (1-\epsilon_t) e^{-\alpha_t} \\
&= 2\sqrt{\epsilon_t(1-\epsilon_t)}
\end{aligned}
$$

How Will Test Error Behave? (A first Guess)



- Expect:
  - training error to continue to drop (or reach zero)
  - test error to increase when Hfinal becomes "too complex"
    - "Occam's razor"
      - overfitting
      - hard to know when to stop training

Actual Typical Run



(boosting C4.5 on "letter" dataset)

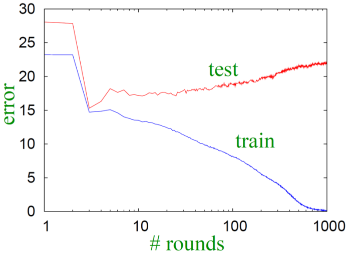
- test error does not increase, even after 1000 rounds
  - (total size $>$ 2,000,000 nodes)
- test error continues to drop even after training error is zero!

	# rounds		
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1

- Occam's razor wrongly predicts "simpler" rule is better

Overfitting

- Overfitting:
  - the data size is too small
  - the base learner is too weak



(boosting "stumps" on heart-disease dataset)

Conclusions

- **Boosting is a practical tool for classification and other learning problems**
  - grounded in rich theory
  - performs well experimentally
  - often (but not always!) resistant to overfitting
  - many applications and extensions
- **Many ways to think about boosting**
  - none is entirely satisfactory by itself, but each useful in its own way
  - considerable room for further theoretical and experimental work

References

- Freund – "An adaptive version of the boost by majority algorithm"
- Freund – "Experiments with a new boosting algorithm"
- Freund, Schapire – "A decision-theoretic generalization of on-line learning and an application to boosting"
- Friedman, Hastie, etc – "Additive Logistic Regression: A Statistical View of Boosting"
- Jin, Liu, etc (CMU) – "A New Boosting Algorithm Using Input-Dependent Regularizer"
- Li, Zhang, etc – "Floatboost Learning for Classification"
- Opitz, Maclin – "Popular Ensemble Methods: An Empirical Study"
- Ratsch, Warmuth – "Efficient Margin Maximization with Boosting"

References

- Schapire, Freund, etc – "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods"
- Schapire, Singer – "Improved Boosting Algorithms Using Confidence-Weighted Predictions"
- Schapire – "The Boosting Approach to Machine Learning: An overview"
- Zhang, Li, etc – "Multi-view Face Detection with Floatboost"