Model Combination

24 Novembre 2009

Plan

Model selection versus model combination

For a given learning task, we typically train several models.

- **Model selection**
  Choose the model that maximizes some performance criterion on a test set.
- **Model combination**
  Combine several models into a single aggregate model (aka ensemble or committee)

Why combine models?

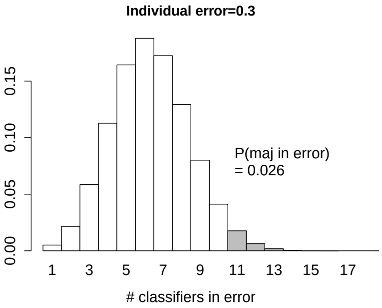
The error of an ensemble $<$ the error of an individual model provided that:

1. the models are diverse or independent;
2. each model performs at least slightly better than chance : $err < 0.5$.

Justification: The error probability of a set of J models

1. follows a binomial distribution $\longrightarrow$ assumes independent models;
2. equals the probability that at least $J/2$ models are wrong.

Ex. An ensemble of 21 models is in error when ≥ 11 make an error.



- $p < 0.5 \Rightarrow$ *P(ensemble error)* $<$ *P(individual error)*
- $p > 0.5 \Rightarrow$ *P(ensemble error)* $>$ *P(individual error)*

Phases of model combination

1. **Diversification** : choose diverse models to cover different regions of your instance space
2. **Intégration** : combine these models to maximize the performance of the ensemble.

Model diversity

- The diversity of 2 models h_1 et h_2 is quantified in terms of their error (non)correlation
- Different definitions, e.g. the error correlation of 2 models is the probability that both make the same error given that one of them makes an error:

$$C_{err}(h_1, h_2) = P(h_1(x_i) = h_2(x_i) | h_1(x_i) \neq y_i \vee h_2(x_i) \neq y_i)$$

- Many other measures of diversity have been proposed [Kuncheva, 2003].

Diversification techniques

- **Resampling** : vary the data used to train a given algorithm
  - Bagging : "bootstrap aggregation"
  - Random Forests : bagging + "variable resampling"
  - Boosting : resampling through adaptive weighting
- **Hybrid learning** : vary the algorithms trained on a given dataset
  - Stacking : "stacked generalization"
  - So-called multistrategy models

Integration techniques

- **Static** : integration procedure is fixed, e.g., vote and retain the majority or mean/median of the individual predictions
- **Dynamic** : base predictions are combined using an adaptive procedure, e.g. meta-learning

Plan

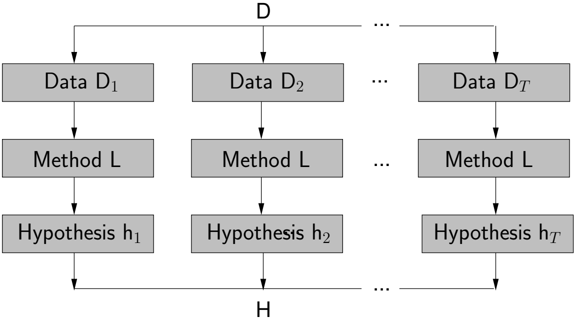
1. Principles of model combination
2. **Resampling methods**
   - **Bagging**
   - **Random Forests**
   - **Boosting**
3. Hybrid methods
   - Stacking
   - Generic algorithm for multistrategy learning

Bagging

- Bagging = "bootstrap aggregation"
- Diversification via resampling
  - Create T bootstrap replicates of the dataset D (samples of size $|D|$ using random draws with replacement)
  - Apply a given learning algorithm to the T replicates to produce T models or hypotheses
- Static integration
  - Ensemble prediction = mean (regression) or uniform vote (classification) of the T models

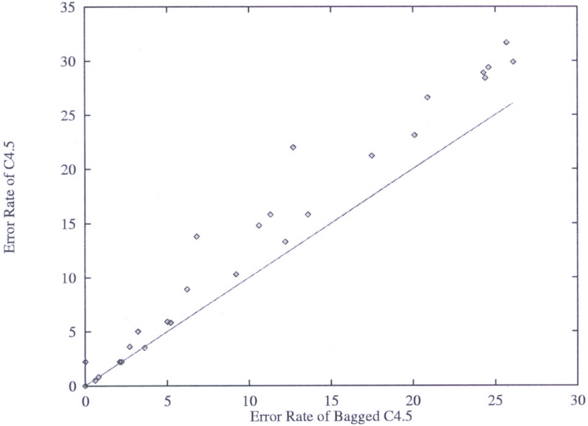
Bagging schema

Diversification : Create bootstrap replicates of D



Integration : Final hypothesis $H=f(h_1, h_2, \ldots, h_T)$

C4.5 with and without bagging



Bagging: summary

- Bootstrap resampling yields highly overlapping training sets
- Reduces error due to variance: improves highly unstable or high-variance algorithms, i.e., where small data variations lead to very different models (e.g. decision trees, neural networks)
- "Reasonable" values of T
  $\sim$ 25 for regression
  $\sim$ 50 for classification

[Breiman, 1996]

Plan

1. Principles of model combination
2. Resampling methods
   - Bagging
   - **Random Forests**
   - Boosting
3. Hybrid methods
   - Stacking
   - Generic algorithm for multistrategy learning

Random Forests (RF)

- RF combines random instance selection (bagging) with random variable selection
- Base learning algorithm: decision trees (CART).
- Forest = an ensemble of T trees.

[Breiman, 2001]

RF algorithm

- TRN contains n examples and p variables
- Hyperparameters : $T = \#$ of trees to build; $m = \#$ of variables to choose at each test node, $m \ll p$
- To build each tree
  1. Create a bootstrap replicate TRN_i of TRN
  2. Create a CART tree with the following modifications:
     - At each tree node, randomly choose m candidate variables
     - Do not prune the tree
- Integration method: uniform vote of the T trees

Optimizing RF

- The ensemble error ϵ_F depends on two factors :
  - correlation co among trees of the forest: if $co \nearrow$ then $\epsilon_F \nearrow$
  - strength s_i of individual trees. Strength $\approx$ prediction accuracy: if $s_i \nearrow$ then $\epsilon_F \searrow$
- Impact of m, the # of randomly drawn variables
  - Increasing m increases both co ($\epsilon_F \nearrow$) and s ($\epsilon_F \searrow$) : co vs s trade-off
  - Choose m by estimating ϵ_i on $VAL_i = \{x_j \in TRN | x_j \notin TRN_i\}$
- Impact of T, the number of trees
  - T can be increased without risk of overfitting [Breiman, 2001]

Plan

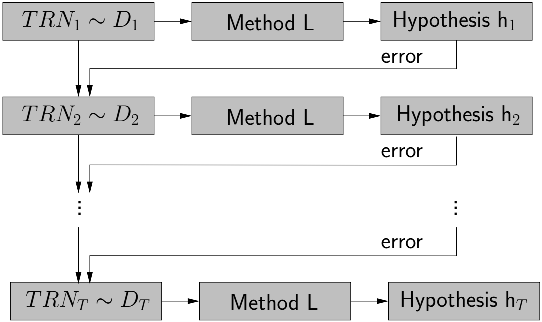
1. Principles of method combination
2. Resampling methods
   - Bagging
   - Random Forests
   - **Boosting**
3. Hybrid methods
   - Stacking
   - Generic algorithm for multistrategy learning

Boosting : the basic idea

- Diversification: sequential adaptive resampling by instance weighting
  - Initially: all instances have equal weights: $1/|TRN|$
  - At each of the T iterations,
    - apply the algorithm and estimate the resubsitution error
    - increase/decrease weights of misclassified/correctly classified cases (focus learning on difficult cases)
- Integration by weighted voting :
  - Apply the T base models to the current instance
  - Return the weighted majority prediction (base classifiers' weights $\propto$ their predictive power).

Boosting schema

Diversification through instance weighting



Integration : Final hypothesis H=f(h_1, h_2,. . ., h_T)

Bagging versus boosting

	Bagging	Boosting
Diversification : resampling		
1	in parallel	sequentially
2	random draws	adaptive weighting
Integration : vote		
3	uniform	weighted by $1 - \epsilon(h_j)$

AdaBoost algorithm

Input: $TRN = \{(x_i, y_i)\}, y_i \in \{-1, 1\}$, algo $\mathcal{L}$

Initialization: $D_1(i) = 1/N$ uniform distribution

For $t = 1$ to T

1. $h_t \leftarrow \mathcal{L}(TRN, D_i)$
2. $\epsilon_t \leftarrow \sum_{i|h_t(x_i)\neq y_i} D_t(i)$ (weighted error)
3. if $\epsilon_t > 0.5$ then $T \leftarrow t - 1$; exit
4. Compute D_{t+1} by adjusting weights of training cases

Result: $H_{final}(x) \leftarrow$weighted combination of h_t's

Freund & Schapire, 1996-1998

Algorithmic details

$\mathcal{L}$ a weak learner ($\epsilon < 0.5$)

Computing D_{t+1}

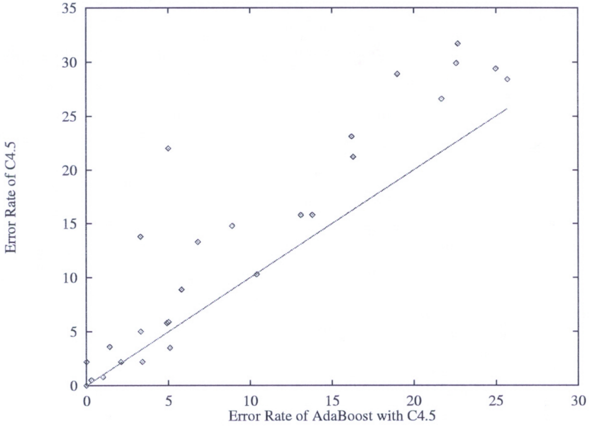
- $\alpha_t \leftarrow \frac{1}{2}\, ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$ (since $\epsilon_t \leq 0.5$, $\Rightarrow \alpha_t \geq 0 \Rightarrow e^{\alpha_t} \geq 1$)
- $D_{t+1}(i) \leftarrow \frac{D_t(i)}{Z_t} \cdot \begin{cases} e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \Rightarrow \text{weight } \nearrow \\ e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \Rightarrow \text{weight } \searrow \end{cases}$

$$\leftarrow \frac{D_t(i)}{Z_t} exp(-\alpha_t y_i h_t(x_i))$$

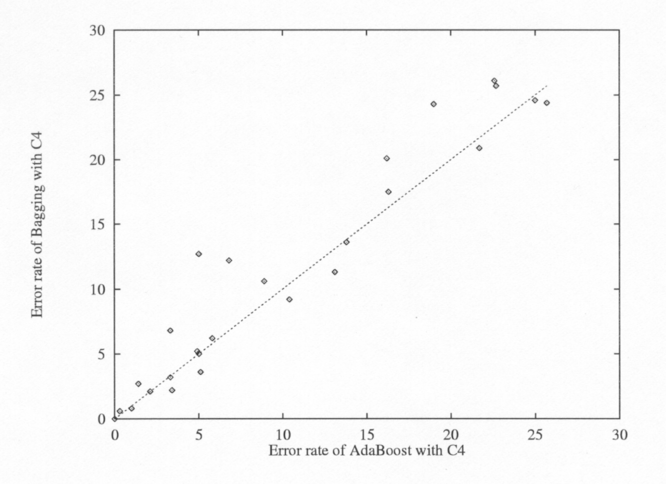
Z_t normalization factor $\rightarrow \sum_i D_{t+1}(i) = 1$

$H_{final}(x) \leftarrow sgn\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

C4.5 with and without boosting



Performance of bagging vs boosting



Boosting: summary

- The power of boosting comes from adaptive resampling
- Like bagging, boosting reduces variance
- Also reduces bias by obliging the learner to focus on hard cases
  $\longrightarrow$ combined hypothesis more flexible
- Fast convergence
- Sensitive to noise : when base learners misclassify noisy examples
  $\Rightarrow$ weights increase $\Rightarrow$ overfitting to noise

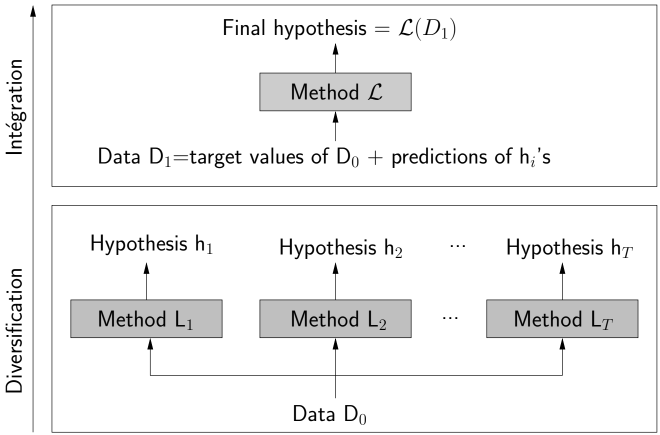
Plan

1. Principles of method combination
2. Resampling methods
   - Bagging
   - Random Forests
   - Boosting
3. **Hybrid methods**
   - **Stacking**
   - **Generic algorithm for multistrategy learning**

Stacking

- "Stacked generalization" [Wolpert 1992]
- Level 0 : diversification through the use of different learning algorithms
- Level 1 : integration through meta-learning

Stacking schema



Level 0 : Diversification

- Input : $D_0 = \{(\mathbf{x}_i, y_i)\}$, $i = 1..N$, J learning algorithms L_j
- Diversification by K-fold cross-validation :

```
for k = 1 to K do
   for j = 1 to J
      h_kj ← L_j(D_0^{-TST_k})
      foreach x_i ∈ TST_k
             pred_ij ← h_kj(x)
```

$$\texttt{for}\ k = 1\ \texttt{to}\ K\ \texttt{do}$$
$$\texttt{for}\ j = 1\ \texttt{to}\ J$$
$$h_{kj} \leftarrow L_j(D_0^{-TST_k})$$
$$\texttt{foreach}\ \mathbf{x_i} \in TST_k$$
$$pred_{ij} \leftarrow h_{kj}(\mathbf{x})$$

- Result: J predictions for each instance of D_0

Level 1 meta-data

- Notation : $i = 1 \ldots N$ examples; $j = 1 \ldots J$ models
- Input : $D_1 = \{(\mathbf{z}_{i,} y_i)\}$, $\mathbf{z}_i \in \mathcal{R}^J$ are the base predictions, y_i the target values
  - regression : $\mathbf{z}_i = [z_{i1}, \ldots, z_{kJ}]$, where z_{ij} = approximation of y_i by model h_j
  - classification :
    - discrete classifier :
      $\mathbf{z}_i = [z_{i1}, \ldots, z_{iJ}]$, with $z_{ij} \in \mathcal{C}$ = class predicted by model h_j for example i
    - probabilistic classifier :
      $\mathbf{z}_i = [\mathbf{z}_{i1}, \ldots, \mathbf{z}_{iJ}]$, with $\mathbf{z}_{ij} = (z_{ij1}, z_{ij2}, \ldots, z_{ij|\mathcal{C}|})$ = class probability distribution

Examples de meta-data

- Level 0 (base-level) : Iris. $L_1 =$J48, $L_2 =$IB1, $L_3 =$NB
- Level 1 : discrete classifiers

#	J48	IB1	NB	Target
1	setosa	setosa	versicolor	setosa
...				...
150	virginica	setosa	virginica	virginica

- Level 1 : probabilistic classifiers

#	J48			IB1			NB			Target
	p1	p2	p3	p1	p2	p3	p1	p2	p3	
1	0.58	0.22	0.2	0.74	0.20	0.06	0.37	0.59	0.04	setosa
...										...
150	0.02	0.16	0.82	0.45	0.39	0.16	0.36	0.23	0.41	virginica

Level 1 : Integration through meta-learning

- The level 1 algorithm depends on the variables of D_1
  - real-valued predictions (regression ou probabilistic classification) : $\mathcal{L}_{\mathcal{M}} \in$ {trees, IBk, MLP, SVM, linear regressors, ...}
  - discrete predictions (discrete classification) : $\mathcal{L}_{\mathcal{M}} \in$ {trees, IBk, Naïve Bayes, ...}
- The combined model $H \leftarrow \mathcal{L}_{\mathcal{M}}(D_1)$

Using stacked models

- After creation of thecombined model H, create the final base models h_j by training the J algorithmes on D_0.
- In production mode: 2-step prediction given a new case $\mathbf{x} \in \mathcal{R}^d$
  - Level 0 : generate a meta-example $\mathbf{z} \in \mathcal{R}^J$: For $j = 1$ to J, $z_j \leftarrow h_j(\mathbf{x})$
  - Level 1: combined prediction $\leftarrow H(\mathbf{z})$

Stacking : summary

- improves considerably over cross-validated model selection
- on average, 3 hypotheses play a significant role in the combined hypothesis
- stacking is just one example of hybrid model combination, many other examples have been proposed

Generic multistrategy learning algorithm

- Diversification
  - Train M learning algorithms on a base-level dataset and measure their performance on a test set
  - Select J models with minimal correlation error
- Integration
  - Static :
    - continuous predictions : compute the mean, median, linear combination, etc.
    - discrete prédictions : uniform or weighted vote
  - Dynamic : use meta-learning

References

- L. Breiman (1996). Bagging Predictors. *Machine Learning* 24(2): 123-140.
- L. Breiman (2001). Random Forests. *Machine Learning* 45: 5-32.
- T. G. Dietterich (1998). Machine Learning Research: Four Current Directions. *AI Magazine* 18(4): 97–136.
- Y. Freund, R. E. Schapire (1996). Experiments with a new boosting algorithm. *ICML-96*.
- L. Kuncheva, C. J. Whitaker (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning* 51: 181–207.
- D. Wolpert (1992). Stacked Generalization, *Neural Networks* 5: 241-259.