

# Stacking Algorithm for Ensemble Modelling

Frederik Schreck

Course: Numerical Introductory Course

Lecturer: Prof. Dr. Brenda López Cabrera

Humboldt-University Berlin



## Motivation - The wisdom of the crowd

- The aggregation of individual guesses in groups is often superior to individual guesses - even to experts
- BUT: Only fulfilled under certain criteria [Surowiecki, 2005]
  - ▶ Variation of guesses
  - ▶ Independence of guesses
  - ▶ Decentralization
  - ▶ Algorithm



# Outline

1. Motivation ✓
2. Background: Decision Tree
3. Ensemble Learning
4. Stacking Algorithms and Ensemble Modelling
  - 4.1 Bagging and Random Forest
  - 4.2 Boosting and Gradient Boosting
  - 4.3 Stacked Generalization
5. Potentials and Problems of Ensemble Learning
6. Sources



## Decision Tree

- Idea: Use a set of splitting rules to recursively partition the dataset
- Classification trees:
  - ▶ Minimize impurity within nodes
- Regression trees:
  - ▶ Minimize variance of the response variable within nodes



## Decision Tree for classification

- Choice of splitting rule: maximizing information gain (IG) by decreasing node impurity ( $I$ )

$$IG_n = I(n) - p_{n_1} \times I(n_1) - p_{n_2} \times I(n_2), \quad (1)$$

for node  $n$  with branching nodes  $n_1$  and  $n_2$ , and  $p_{n_i}$  as the fraction of cases in branching node  $n_i$ ,  $i \in \{1, 2\}$



## Decision Tree for classification

- Choice of stopping rule:

A fully grown tree has pure leaf nodes and may overfit the data. However, a small tree may not capture all relevant structure of the data

- ▶ Pre-pruning
- ▶ Post-pruning



## Ensemble Learning - Terminology

### Machine Learning

- Part of computer science that uses statistical techniques to train models on data
- Typically used for prediction purposes

### Stacking and Ensemble Learning

- Idea is to combine hypotheses of multiple learning algorithms (base learners)
- Goal is to obtain a better predictive performance than with each of the single algorithms alone
- Mainly used in supervised learning
- Very flexible method



## Ensemble Learning

Which models to combine?

- Effective ensembling builds on diverse and little correlated models [Kuncheva and Whitaker, 2003]
- Typically best to use strong base learners
- Choice of algorithm decisive

Compare with the criteria mentioned in the Motivation!





# Ensemble Learning

Which models to combine?

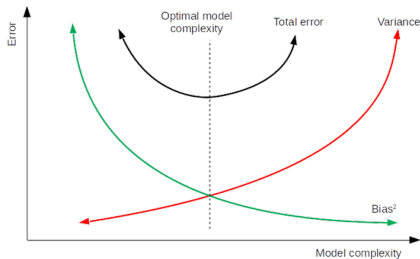


Figure 1: The bias-variance-trade-off.

- Combining complex classifiers may reduce variance
- Combining simple classifiers may reduce bias

Stacking Algorithm for Ensemble Modelling



## Bagging (= Bootstrap Aggregating)

- Proposed by Leo Breiman [Breiman, 1996]
- Meta-algorithm, designed to
  - ▶ improve accuracy of base algorithms
  - ▶ reduce MSE by reducing variance
  - ▶ avoid overfitting problems
  - ▶ obtain smoother prediction boundaries
- Can be applied to all kinds of base learners
- However best to use unstable methods that tend to have high variance, like trees



## Bagging algorithm

Suppose we have training data  $\{(x_1, y_1), \dots, (x_N, y_N)\}$

---

- 1 for base learner  $m$  in  $\{1, 2, \dots, M\}$
  - 2        uniformly draw sample  $D_m$  from dataset  $D$  (with repl.)
  - 3        build model  $T_m$  on dataset  $D_m$  to obtain hypothesis  $h_m(x)$
  - 4 combine hypotheses
- 

- ▣ Combining by averaging in regression problems
- ▣ Combining by majority vote in classification problems



## Random Forest

- Also proposed by Leo Breiman [Breiman, 2001]
- Random forests combine bagging with random subspace approach [Ho, 1998]
- Random subspace approach randomly samples features from set of all features for each learner (with replacement)
  - ▶ Reduces the correlation between estimators
  - ▶ Thus decreases variance in the ensemble learner
- Random feature sampling happens at tree level or at split level
- Random Forest only possible with tree-based base learners



## Random Forest algorithm for classification

Suppose we have training data  $\{(x_1, y_1), \dots, (x_N, y_N)\}$

---

- 1 for base learner  $m$  in  $\{1, 2, \dots, M\}$
  - 2        uniformly draw sample  $k_m$  of size  $L$  from features  $\{1, 2, \dots, K\}$  (with repl.)
  - 3        uniformly draw sample  $D_m$  from dataset  $D$  (with repl.)
  - 4        build model  $T_m$  on dataset  $D_m$  using feature set  $k_m$
  - 5     $\hat{C}_{rf}^{L,N}(x) = \text{majority vote}\{\hat{T}_m\}_1^M$
- 



## Random Forest

### Random Forest vs. single Tree

Random Forest	Single Tree
<ul style="list-style-type: none"><li>— higher computational costs</li><li>— blackbox</li><li>— many parameter choices to make</li><li>+ easy to tune parameters</li><li>+ smaller prediction variance</li><li>+ scalability</li></ul>	<ul style="list-style-type: none"><li>+ computationally simple</li><li>+ insights into decision rules</li><li>+ easy to tune parameters</li><li>— tends to overfit and have high variance</li></ul>



## Boosting

- ▣ Method proposed by Freund & Shapire [Freund et al., 1996]
- ▣ Original idea only applies to classification problems
- ▣ Idea: Simple learners are easier to find. Combining many simple learners can produce a powerful learner
- ▣ The ensemble first considers only one base learner. Then we iteratively enlarge it by another base learner that aims to correct the error of the current ensemble



## The Adaboost algorithm

Suppose we have training data  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ ,  
initialize weightings  $d_i^{(1)} = \frac{1}{N}, \forall i \in \{1, \dots, N\}$

---

- 1 for base learner  $m$  in  $\{1, 2, \dots, M\}$
  - 2     train base learner according to weighted data  $d^{(m)}$   
      and obtain hypothesis  $h_m : \mathbf{x} \mapsto \{-1, +1\}$
  - 3     calculate weighted classifier error  
       $\epsilon_m = \sum_{i=1}^N d_i^m I(y_i \neq h_m(x_i))$
  - 4     calculate hypothesis weighting  $\beta_m = \frac{1}{2} \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$
  - 5     update data weighting, e.g. by  
       $h_m(x_i) = y_i : d_i^{m+1} = d_i^m \exp(-\beta_m)$   
       $h_m(x_i) \neq y_i : d_i^{m+1} = d_i^m \exp(\beta_m)$
  - 6      $\hat{y}(x) = H_{final}(x) = \frac{1}{M} \sum_{m=1}^M \beta_m h_m(x)$
- 





## Gradient Boosting

- ▣ Developed by Friedman [Friedman, 2001]
- ▣ Extended boosting to regression and other problem types
- ▣ Shortcomings of current ensemble is identified by gradients instead of weightings of data
- ▣ In each stage  $m \in \{1, 2, \dots, M\}$ , a new learner improves the current ensemble  $H_{m-1}$  and is fitted to  $(x_i, y_i - H_{m-1}(x_i)), \forall i \in \{1, 2, \dots, N\}$



## Stochastic Gradient Boosting

- Advancement of Gradient Boosting, again by Friedman [Friedman, 2002]
- Utilizes ideas from Bagging:
  - ▶ Using trees as base learners
  - ▶ Fit trees to negative gradient of random sample of dataset
- Less prone to overfitting



## Gradient Boosting

### Random Forest vs. single Tree vs. Gradient Boosting

Random Forest	Single Tree	Gradient Boosting
<ul style="list-style-type: none"><li>— higher computational costs</li><li>— blackbox</li><li>+ easy to tune parameters</li><li>+ smaller prediction variance</li><li>+ scalability</li><li>— many parameter choices to make</li></ul>	<ul style="list-style-type: none"><li>+ computationally simple</li><li>+ insights into decision rules</li><li>+ easy to tune parameters</li><li>— tends to overfit and have high variance</li></ul>	<ul style="list-style-type: none"><li>+ relatively fast to train</li><li>+ insights by feature importance and partial dependence plots</li><li>+ one of the best of-the-shelf methods</li><li>— tends to overfit</li><li>— parallelization difficult</li><li>— many tunable parameters</li></ul>



## Stacked Generalization

- Heterogeneous ensemble model
- Second stage model: Combines predictions of different classifiers
- Multistep-Approach:
  1. Compare predictive accuracy of individual classifiers
  2. Select the best individual classifier
  3. Iteratively add one more classifier and combine the predictions, e.g. by averaging
  4. Stop when predictive accuracy of the ensemble does not increase anymore



## The Multistep-Approach

An example:

Iteration	y	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$
1	0	0.53	0.17	0.07
	0	0.62	0.61	0.95
	1	0.76	0.41	0.31
Brier Score:		0.24	0.25	0.46

Iteration	y	$(\hat{y}_1, \hat{y}_1)$	$(\hat{y}_1, \hat{y}_2)$	$(\hat{y}_1, \hat{y}_3)$
2	0	0.53	0.35	0.30
	0	0.62	0.62	0.79
	1	0.76	0.59	0.54
Brier Score:		0.24	0.22	0.31

Iteration	y	$(\hat{y}_1, \hat{y}_2, \hat{y}_1)$	$(\hat{y}_1, \hat{y}_2, \hat{y}_2)$	$(\hat{y}_1, \hat{y}_2, \hat{y}_3)$
2	0	0.41	0.29	0.26
	0	0.62	0.61	0.73
	1	0.64	0.53	0.49
Brier Score:		0.225	0.228	0.280



## Stacked Generalization

- Most important: Diversity of first-stage models!
- Diversity can be enhanced by using
  - ▶ different algorithms
  - ▶ different parameter settings
  - ▶ different feature subsets
  - ▶ different training sets
- Overfitting problem. Can be evaded by
  - ▶ using cross-validation in second-stage model training
  - ▶ using regularization
  - ▶ using a combiner algorithm that is not sensible to multicollinearity, like Regularized Logistic Regression, Random Forest or hill-climbing methods.



## Stacked Generalization

Advantages	Disadvantages
<ul style="list-style-type: none"><li>+ often high predictive accuracy and robustness (most successful approach in data science competitions on <i>kaggle</i>)</li><li>+ scalable, especially parallelizable</li></ul>	<ul style="list-style-type: none"><li>— complex to implement</li><li>— high amount of computational resources needed</li><li>— different (!) first-stage models must be built</li></ul>



## Potentials of Ensemble Learning

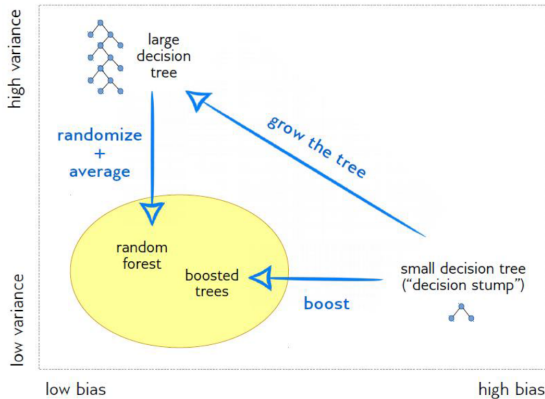


Figure 2: How Gradient Boosting and Random Forest improve performance.





# Potentials and Problems of Ensemble Learning

Potentials	Problems
<ul style="list-style-type: none"><li>+ currently best predictive methods available</li><li>+ ensembling can decrease variance and bias</li><li>+ often scalable</li></ul>	<ul style="list-style-type: none"><li>— needs high computational resources</li><li>— blackbox problems</li><li>— many parameters to tune</li><li>— lack of proven statistical properties</li></ul>



## References |

- Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.
- Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Bari, Italy.
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.
- Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.



## References II

- Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844.
- Kuncheva, L. I. and Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207.
- Surowiecki, J. (2005). *The Wisdom of Crowds*. Anchor.

