Multiple Models

João Gama

LIAAD-INESC Porto, University of Porto, Portugal
jgama@fep.up.pt

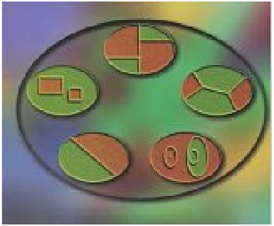
December 2018

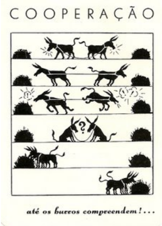
Outline

Multiple Models

How to take advantage of these differences?

Would be possible to obtain an ensemble of classifiers with a performance better than each individual classifier?





Observation: There is no overall better algorithm.

- Experimental results from Statlog and Metal project;
- Theoretical Results: *No free lunch* theorems.

Multiple Models

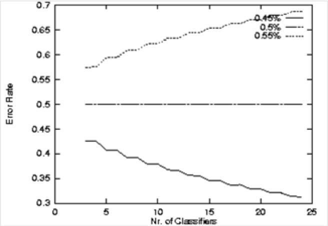
A simulation study:

- Consider a decision problem with two equi-probable classes: $P(Class_1) = P(Class_2)$
- The number of classifiers in the ensemble varies between [3, ..., 25].
- All classifiers have the same probability of error. Assume $P_{error}(Classifier_i) = \{0.45; 0.5; 0.55\}$

Multiple Model: aggregate the predictions of individual classifiers

- For each example
  - Each classifier predicts a class label.
  - Count the votes for each class
  - Predict the most voted class: *uniform voting*.

Multiple Models: Simulation



Study how the error varies when varying the number of classifiers in the ensemble. Probability of error of each classifier:

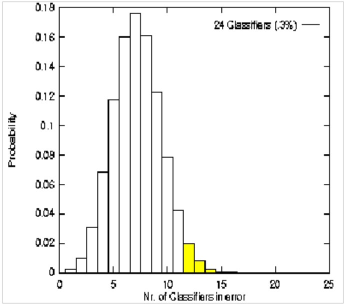
- $P = 0.5$ (random choice)
  The error of the ensemble is constant: 0.5
- $P > 0.5$
  The error of the ensemble increases linearly with the number of classifiers.
- $P < 0.5$
  The error of the ensemble **decreases** linearly with the number of classifiers.

Another Necessary Condition

Necessary Condition

The error of the ensemble decreases, with respect to each individual classifier, iif each individual classifier has a performance better than a random choice.

Multiple Models: Simulation



Assume an ensemble of 23 classifiers:

- probability of error of each classifier: 30%;
- aggregation by uniform vote.

Given a test example:

- the ensemble will be in error iif 12 or more classifiers are in error.
- The probability of error in the ensemble is given by the area under the curve of a binomial distribution;
- In this case this area is 0.026.
- Much less than each individual classifier

Necessary Conditions

To achieve higher accuracy the models should be diverse and each model must be quite accurate Ali & Pazzani 96

Necessary Conditions

Classifiers in the ensemble, should have:

- performance better than random guess;
- non-correlated errors;
- errors in different regions of the instance space.

Multiple Models

- Combining Outputs
  - Voting Methods
  - Fusion of Classifiers
  - Model Applicability
- Perturbing the set of training examples
  - Homogeneous Classifiers
    - Bagging
    - Boosting
  - Heterogeneous Classifiers
    - Cascading
    - Stacking
- Perturbing the set of attributes
- Perturbing test examples

Outline

Using different distributions of examples

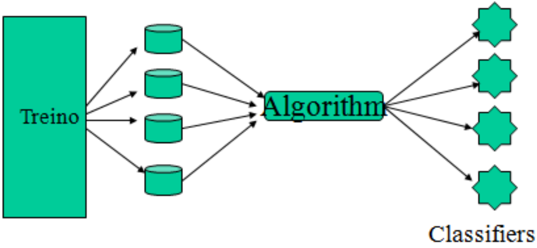
- Bagging, L. Breiman 92
- Boosting, R. Schapire and Y. Freund, 94

Boostrap Aggregation - Bagging

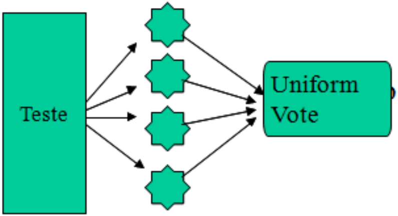
- Learning:
  - Obtain N replicas of the training set, with reposition;
  - All the samples with the same number of examples of the training set;
  - Learn a classifier for each sample.
- Testing
  - For each test example;
  - All classifiers classify the test example;
  - Predictions are aggregated by uniform vote.

Bagging

- **Learning:**



- **Test**



Bagging

Given a dataset D with n examples, bagging generates m new training sets D_i, each of size n', by sampling from D uniformly and with replacement.

- By sampling with replacement, some observations may be repeated in each D_i.
- When drawing with replacement n' values out of a set of n (different and equally likely), the expected number of unique draws is $1 - (1 - \frac{1}{n})^n$
- For large n, this probability is $1 - 1/e$, where e is the base of natural logaritms
- On average, each replica will contain 36.8% of duplicates

Why Bagging Works ?

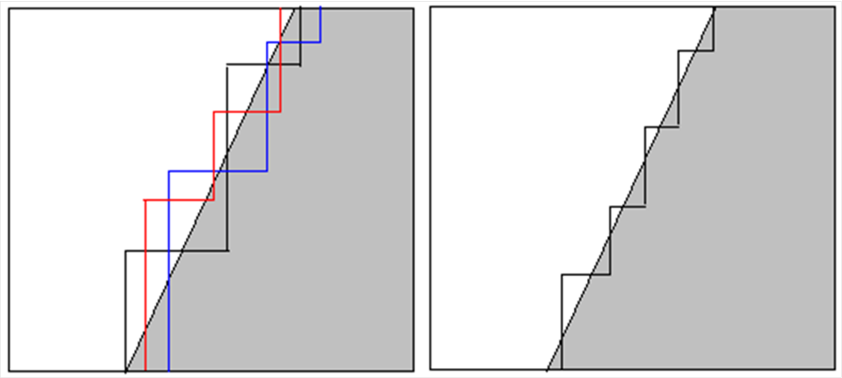
Choosing the majority vote over several classifiers reduces the randomness associated with individual models.

Example: decision trees

Decision trees use greedy algorithms. The training set can influence too much in:

- The choice of attributes for splitting-tests;
- The choice of *cut_points*

Why Bagging Works ?



Bagging

Properties:

- Requires unstable algorithms (greedy like)
- Algorithms sensible to small perturbations of the training set;
  - Decision trees, Rule learners, Neural Networks, etc.
- Easy to implement with any algorithm;
- Easy to implement in parallel environments.

The bias-variance argument:

Error decreases due to reduction in the variance component.

Random Forests

Breiman, *Random Forests*, MLJ 2001;

A variant of Bagging;

- Repeat k times
  - Training set = Draw with replacement N examples;
  - Built a decision tree
    - Choose (without replacement) i features
    - Choose best of these i as the root of this (sub)tree
  - Do NOT prune

where N is the nr. of examples, F nr. of features, and i some number $<< F$.

Boosting

- Can a set of *weak learners* create a single *strong learner*?
- A weak learner is defined to be a classifier which is only slightly correlated with the true classification.
- A strong learner is a classifier that is arbitrarily well-correlated with the true classification.

Rob Schapire, *Strength of Weak Learnability* Journal of Machine Learning Vol. 5, pages 197-227. 1990

Boosting

Theoretical framework

- Given:
  - A confidence level δ, so high as desired;
  - An error bound ϵ, so small as desired;
- Is it possible to design an algorithm that with probability δ generates an hypothesis with error ϵ for *any* distribution of examples generated for a given problem?

Boosting is one of such algorithms!

Boosting

Characteristics

- Boosting is an iterative algorithm;
- Associates a weight with each example;
- The weight indicates the probability of the example being select in a uniform sampling;

Boosting

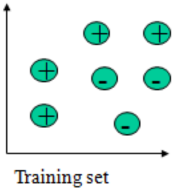
Base Algorithm

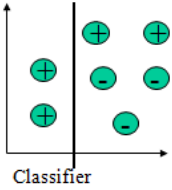
- Input:
  - *weak-learner* algorithm that generates a classifier better than a random guess;
  - Training set.
- Initialize uniform weights of examples, sum equal to one;
- For i in 1 ... N
  - Generate a classifier using the actual distribution of the examples;
  - The weight of the examples misclassified increases;
  - The weight of the examples correctly classified decreases;
- The classifiers generated in all iterations are aggregated using weighted voting.

Boosting: Example

Weak learner – generate an hyper-plane perpendicular to one of the axis

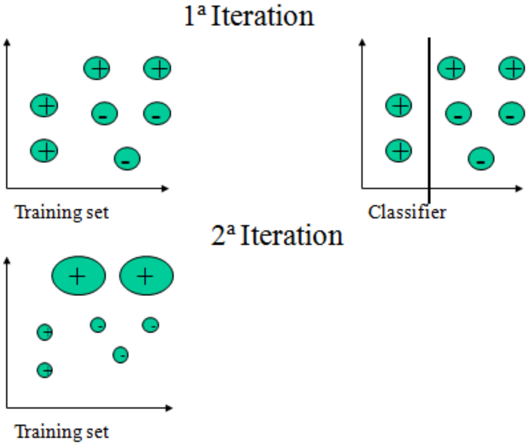
1ª Iteration





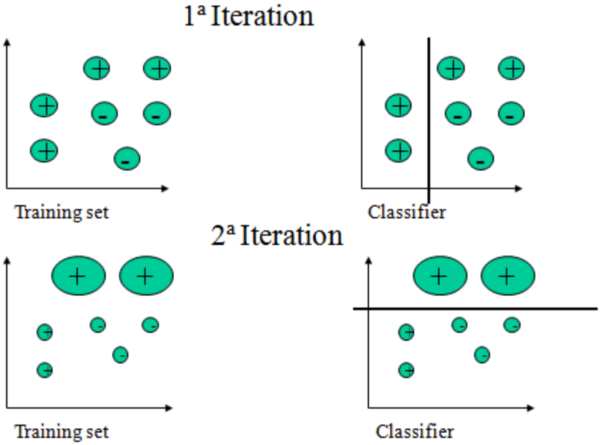
Boosting: Example

Weak learner – generate an hyper-plane perpendicular to one of the axis



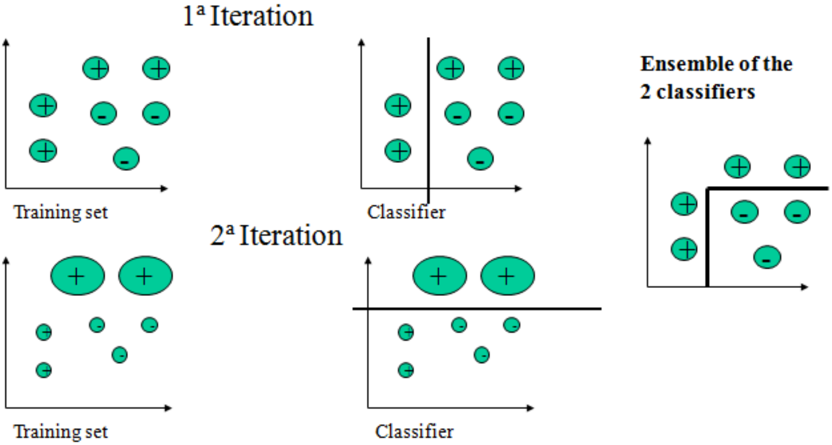
Boosting: Example

Weak learner – generate an hyper-plane perpendicular to one of the axis



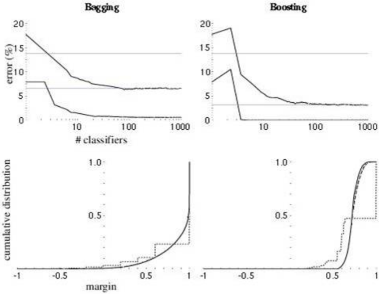
Boosting: Example

Weak learner – generate an hyper-plane perpendicular to one of the axis



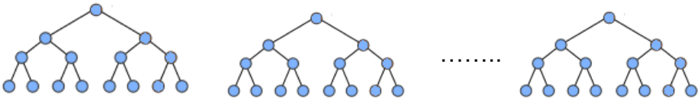
Comparison between Bagging & Boosting

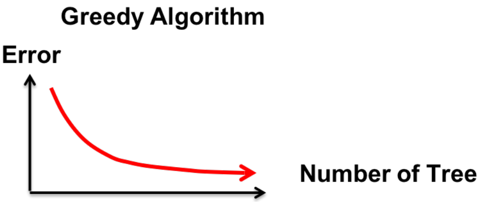
- Bagging
  - Error reduction due to reduction in Variance;
  - Effective with unstable classifiers;
  - Not reported increase of error;
- Boosting
  - Error reduction due to reduction in bias and variance;
  - risky in problems with noise (increase of the error);



XGBoost - Extreme Gradient Boosting Tree

- Additive tree model: add new trees that complement the already-built ones

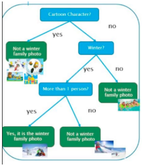




XGBoost: Learning

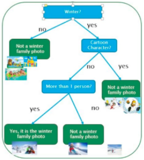
First tree:

e_1	$\mathbf{x_1}$	y_1
e_2	$\mathbf{x_2}$	y_2
	$\dots$	
e_n	$\mathbf{x_n}$	y_n

h_1: 

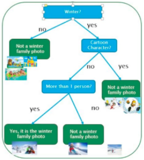
Second tree:

e_1	$\mathbf{x_1}$	$y_1 - h_1(e_1)$
e_2	$\mathbf{x_2}$	$y_2 - h_1(e_2)$
	$\dots$	
e_n	$\mathbf{x_n}$	$y_n - h_1(e_n)$

h_2: 

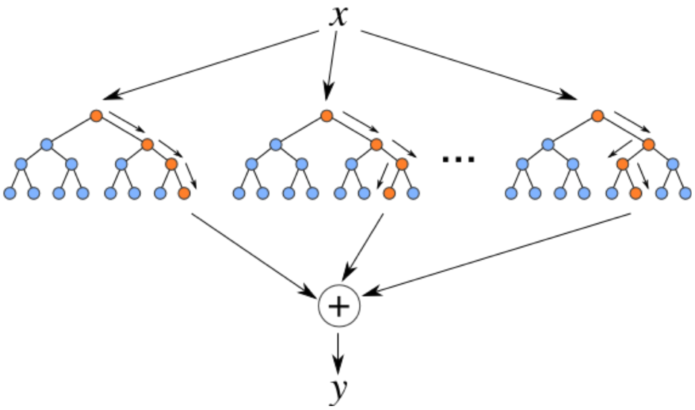
Second tree:

e_1	$\mathbf{x_1}$	$y_1 - h_1(e_1) - h_2(e_1)$
e_2	$\mathbf{x_2}$	$y_2 - h_1(e_2) - h_2(e_2)$
	$\dots$	
e_n	$\mathbf{x_n}$	$y_n - h_1(e_n) - h_2(e_n)$

h_3: 

XGBoost - Extreme Gradient Boosting Tree

- Response is the optimal linear combination of all decision trees



Stacking Generalization

Wolpert, *Stacking Generalization*, Neural Networks, Nr. 5, 1992

Layered Learning

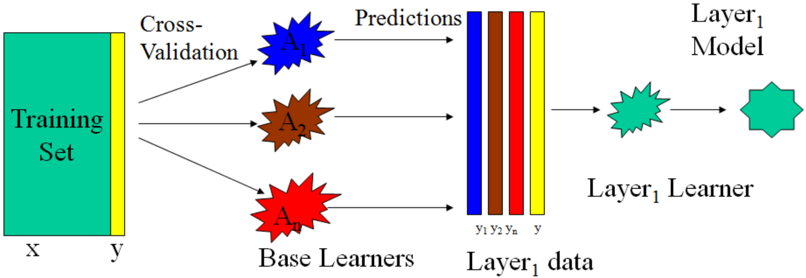
The output of an ensemble of trained classifiers is used as input to the next-layer of classifiers.

Stacked Generalization with 2 layers

- $Layer_0$
  - Data : is original training set;
  - Models : classifiers trained from the $layer_0$ data;
- $Layer_1$
  - Data : the predictions of $layer_0$ classifiers on $layer_0$ data using cross-validation;
  - Models : classifier trained from the $layer_1$ data;

Stacking Generalization

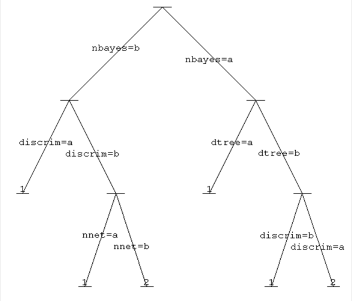
Learning $Layer_1$ Model



Stacking Generalization: Example

Base models: naive Bayes, neural net, decision tree, linear discriminant (LDA);

layer$_1$ model: decision tree



layer$_1$ model: LDA

```
> lda(observed~.,df)
Call:
lda.formula(observed ~ ., data = df)

Prior probabilities of groups:
   1    2
0.51 0.49

Group means:
   discrim2   nbayes2     nnet2    dtree2
1 0.4509804 0.5686275 0.5098039 0.3725490
2 0.4285714 0.4693878 0.4489796 0.4693878

Coefficients of linear discriminants:
                LD1
discrim2 -0.3424098
nbayes2  -1.3950698
nnet2    -1.0185849
dtree2    1.0547212
```

Analysis

Main Goal

Layer$_1$ classifier search for the best bias between *layer*$_0$ classifiers.

Stacking Generalization: when it works?, Ting & Witten, IJCAI-97,

- Which Classifier for *layer*$_1$?
  - Linear discriminant (LDA):
    weighted vote of predictions of each base classifier.
- Which Attributes for *layer*$_1$?
  - Class probability distribution of base classifiers

Effectiveness

Stacking is effective in reduction of error's bias component

Cascade Generalization

Gama, Brazdil; *Cascade Generalization*, Machine Learning, 2000

- Layered Learning: Sequential composition of classifiers,
- A each layer:
  - Learn a classifier
  - Extend the training set with new attributes
  - The new attributes are the predictions of classifier learnt at this layer
  - The new attributes might be:
    - The class label predicted by the classifier;
    - Class distribution given by each base classifier;

Cascade Generalization

Sequential composition of a naive-Bayes and a Decision Tree:

Dataset Original

```
3,4,3,4,B
4,1,4,1,B
4,2,2,1,L
5,2,5,3,R
2,5,4,4,R
2,3,4,3,R
5,1,4,5,R
4,3,2,5,L
3,3,2,5,R
1,3,4,5,R
```

Dataset Extendido

```
3,4,3,4,0.461183,0.077635,0.461183,B
4,1,4,1,0.413818,0.172365,0.413818,B
4,2,2,1,0.838750,0.089446,0.071804,L
5,2,5,3,0.307441,0.089143,0.603416,R
2,5,4,4,0.283686,0.104362,0.611952,R
2,3,4,3,0.213796,0.070892,0.715312,R
5,1,4,5,0.072916,0.075340,0.851744,R
4,3,2,5,0.505602,0.094848,0.399550,L
3,3,2,5,0.391624,0.080813,0.527563,R
1,3,4,5,0.030005,0.043305,0.926691,R
```

Arvore de Decisao (dataset extendido)

```
File stem <balnew>
Read 625 cases (7 attributes) from balnew.data
Decision Tree:
p3 > 0.471812 : R (288.0)
p3 <= 0.471812 :
|   p1 <= 0.471812 : B (49.0)
|   p1 > 0.471812 : L (288.0)
Tree saved
```

How to Use?

- Use algorithms with different bias-variance profiles
- At the beginning of the sequence use low-variance algorithms
- At the end of the sequence use low-bias algorithms

Outline

Summary

Well designed ensembles of classifiers allow improve performance over their individual elements.

Necessary Conditions

- Variability between elements;
- Low Error correlation;
- Each individual classifier must be better than a random choice.

Bibliography

- Ali and Pazzani, *Error Reduction through learning multiple descriptions*, Machine Learning,23, 1996
- Breiman, L. Stacking Predictors, Machine Learning, 25, 1997
- Breiman,L, Bagging Predictors, Machine Learning,24, 1997
- Freund,Y. and Schapire Experiments with a new boosting algorithm, ICML96
- Gama,J. Cascade Generalization, Machine Learning, 2000
- Wolpert,D. Stacked Generalization, Neural Networks, N.5