Group 97: A boosting approach to the Higgs Boson classification challenge

Laurent Lejeune, Tatiana Fountoukidou, Guillaume de Montauzon

I. Introduction

The Higgs Boson machine learning challenge was organized in 2014 by the CERN. The ATLAS simulated detector was used to provide training and validation data sets to the scientific community. In the frame of this work, we explore basic approaches that tackles the binary classification of the Higgs Boson.

II. Data pre-processing and exploration

A. Data clean-up

About 70% of the samples contain missing values. Replacing missing values by the expectation over the valid samples leads to a distortion of the variable's distribution, which tends to introduce severe biases in learning procedures. The strategies that were implemented and tested are now introduced.

1) Least-squares regression
2) Attribute removal Among the 30 attributes given in the datasets, 11 contain missing values. The most straight-forward appraoch is to remove those attributes from the sets and thus reduce the dimensionality of the problem.
3) Analysis of missing values The class probabilities given the presence or absence of missing values are computed on the training set. The results show that uncomplete samples carry significant prior information on the class label. In this case, such samples are likely to belong to the negative class.
   - $P(Y = 1|X \text{ is uncomplete }) \approx 0.30$
   - $P(Y = -1|X \text{ is uncomplete}) \approx 0.70$
   - $P(Y = 1|X \text{is complete}) \approx 0.47$
   - $P(Y = -1|X \text{is complete}) \approx 0.53$
4) K-Nearest-Neighbors regression A variant of the K-Nearest-Neighbors algorithm was implemented to fill-in missing values [1]. To alleviate the computational cost, a random uniform sampling of the valid samples (samples without missing values) was performed prior to the nearest neighbors search. The missing values are replaced by the weighted average value over its K nearest neighbors using Euclidean distance.
   -

III. Methods

Both linear least-squares and logistic regression were implemented and tested. They are used as baselines for the evaluation of our boosting approach.

A. Linear Least-squares regression

B. AdaBoost with decision stumps

The idea of adding weak learners (learner that predict slightly better than random guessing) in a stage-wise manner to produce a strong classifier is commonly referred to as boosting. Discrete AdaBoost, described in [2], consists in adapting the weights of training samples based on the missclassification error. The goal is to penalize missclassified samples and updated the weights for the computation of the next stage. The exponential loss, written $e^{-\boldsymbol{y}_i \alpha_t \boldsymbol{h}_t(x)}$, takes values higher than 1 (weight is increased) when the response $h_t(x_i)$ (stage t on sample x_i) has an opposite sign as the ground-truth y_i. Conversely, when the classification is correct (signs are the same), the weight is decreased.

Algorithm 1 Discrete AdaBoost

Start with weights $w_i = \frac{1}{N}, i = 1, ..., N$
for $t = 1, 2, ..., T$ **do**
    Fit the classifier $h_t(\boldsymbol{x}) \in \{-1, 1\}$ using weights w_i
    Compute $\boldsymbol{e}_t = \sum_{i=1}^{N} \boldsymbol{w}_i, t$, where $h_t(x_i) \neq y_i$
    Choose $\alpha_t = \frac{1}{2} \log \frac{1-\boldsymbol{e}_t}{\boldsymbol{e}_t}$
    Add to ensemble: $\boldsymbol{F}_t(\boldsymbol{x}) = \boldsymbol{F}_{t-1}(\boldsymbol{x}) + \alpha_t h_t(x)$
    Update weights: $\boldsymbol{w}_{i,t+1} = \boldsymbol{w}_{i,t} e^{-\boldsymbol{y}_i \alpha_t \boldsymbol{h}_t(x_i)}$
    Renormalize $\boldsymbol{w}_{i,t+1}$ such that $\sum_i \boldsymbol{w}_{i,t+1} = 1$
end for

A very simple version of AdaBoost, named Discrete Adaboost, was implemented following algorithm 1. It selects optimal decision stumps as weak-learners, that is, thresholding function that best separates the positive and negative classes with respect to a single attribute (feature). A given weak-learner can thus be described using three parameters: The feature, the threshold value, and the polarity. This last parameter can be either "less or equal", in which case the positive label is given to samples less or equal to the threshold, or "greater than".

IV. Results

A 10-fold cross-validation is performed on our methods. In the case of Adaboost, the model complexity is given by the number of stages (or number of iterations). As for linear least-squares and logistic regressions, the

regularization coefficient λ expresses the smoothness constraint. As metric, the missclassification rate is used.

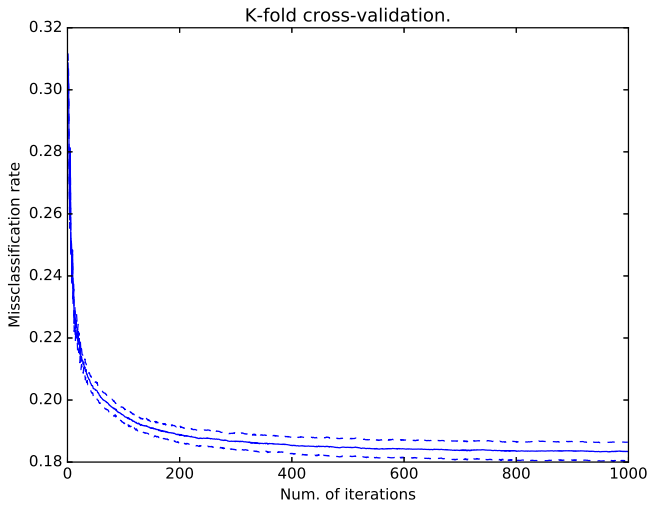

Figure 1: Missclassification rate with respect to number of stages. 10-fold cross-validation

REFERENCES

[1] M. R. Malarvizhi and A. S. Thanamani, "K-nearest neighbor in missing data imputation," *International Journal of Engineering Research and Development*, vol. 5, no. 1, pp. 5–7, 2012.

[2] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Annals of Statistics*, vol. 28, p. 2000, 1998.