swissQuant
Mastering Complexity

Mathematical Challenge May 2017

Model boosting

References

- [1] Chen, Tianqi, and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." *arXiv preprint arXiv:1603.02754* (2016).
- [2] Freund, Yoav, Robert Schapire, and N. Abe. "A short introduction to boosting." *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999): 1612.
- [3] Hastie, Trevor, et al. "The elements of statistical learning: data mining, inference and prediction." *The Mathematical Intelligencer* 27.2 (2005): 83-85.
- [4] Friedman, Jerome H. "Stochastic gradient boosting." *Computational Statistics & Data Analysis* 38.4 (2002): 367-378
- [5] Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.
- [6] Mallat, Stéphane G., and Zhifeng Zhang. "Matching pursuits with time-frequency dictionaries." *Signal Processing, IEEE Transactions on* 41.12 (1993): 3397-3415.

Description

Model boosting is a very successful and widely used technique to build classifiers [1]. Essentially, it allows to obtain a powerful ensemble classifier from a “weak” base classifier, whose error rate is slightly better than random guessing. The main idea is to build a committee of sequentially trained classifiers, where each classifier is trained on a data set modified according to the performance of the previous classifiers, with the aim of improving it. The final classifier is then usually obtained through a weighted majority vote

$$C^M(x) = \sum_{m=1}^{M} \alpha_m C_m(x)$$

A popular boosting algorithm is called *AdaBoost* (Adaptive Boosting) [2], where the classifier $C_m(x)$ is trained on the data (x_i, y_i), modified through a reweighting schema which increases the importance of previously misclassified instances

$$w_i^{(m+1)} = w_i^{(m)} \exp(\alpha_m I(y_i \neq C_m(x_i)))$$

The vote of each classifier is finally weighted according to its accuracy

$$\alpha_m = log\left(\frac{1-err_m}{err_m}\right) \text{ with } err_m = \frac{\sum w_i I(y_i \neq C_m(x_i))}{\sum w_i}$$

This algorithm can be conveniently reformulated [3] as a forward stagewise additive modeling using the loss function

$$L\big(y, C(x)\big) = \exp(-y \cdot C(x))$$

This means that AdaBoost is a greedy approximate solution of the problem of fitting an additive model, through the minimization problem:

$$\min_{\alpha_m, C_m} \sum_i \exp(-y_i \sum_m \alpha_m C_m(x))$$

This reformulation makes clear why AdaBoost is sensitive to outliers, indeed those receive a weight exponentially growing with the negative margin, and suggests how the algorithm could be modified. This can be achieved considering a more robust loss function, as the support vector loss $(1 - y \cdot C(x))^+$ for example, changing the way how the minimization problem is approximated or by regularizing the target function.

The second suggestion leads indeed to a very important family of boosting algorithms, called (Stochastic) *Gradient Boosting* [4]. In these algorithms the idea is to approximate (stochastic) gradient descents techniques by fitting the classifier $C_m(x)$ to the negative gradient of the loss function at the point $\sum_{j=0}^{m-1} \alpha_j C_j(x)$.

The third suggestion would consist in adding a $L_{p=1,2}$ penalization on the model parameter or in the introduction of a small learning rate ν

$$C^M(x) = C^{M-1}(x) + \nu C_M(x)$$

Historically, boosting algorithms were mostly applied to simple classification trees for classification tasks, however the same principle can be applied to regression problems, where for example the base model can be a single variable regression. In this case (using a small learning rate) we obtain an algorithm very similar to the Lasso regression [3].

Questions:

- Analogously to the classification case, in the regression settings more robust loss functions could be used instead of OLS (a.o. Huber loss function). Provide an example where the change of loss function relevantly improve the performance of a boosted regression algorithm.
- Following [5, Ch 6.3] try to motivate from a theoretical perspective why boosting “does not” (slowly) overfit.
- Consider situations where a classifier has low bias but high variance or high bias and low variance. Assess the improvement provided by boosting in each case and compare it with the ones obtained by using other ensemble techniques like bagging or stacking [3].
- Boosting, or the closely related matching pursuit approach [6], have been applied to different models (dictionaries) mainly for signal processing and machine learning purposes. Could you envision an application to econometric modeling?

We look forward to your opinions and insights.

Best Regards,

swissQuant Group Leadership Team