Avisek Gupta (avisek003@gmail.com)

Machine Learning

22 – AdaBoost, Random Forests

December 02, 2022

Ensemble Learners

Main Idea: Combine a set of base models that have been trained on the same original task, to obtain a composite model that is more accurate than the base models.

Ensemble Learning: A model $f(y|\mathbf{x})$ is learnt as a weighted combination of M number of base models $f_i(y|\mathbf{x})$.

$$f(y|\mathbf{x}) = \sum_{m=1}^{M} \alpha_m f_m(y|\mathbf{x})$$

Ensemble Learners

Main Idea: Combine a set of base models that have been trained on the same original task, to obtain a composite model that is more accurate than the base models.

Ensemble Learning: A model $f(y|\mathbf{x})$ is learnt as a weighted combination of M number of base models $f_i(y|\mathbf{x})$.

$$f(y|\mathbf{x}) = \sum_{m=1}^{M} \alpha_m f_m(y|\mathbf{x})$$

Example of Ensemble Learning - **Stacking** (Stacked Generalization): Define a loss function L and optimize for w.

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} L(y_i, \sum_{m=1}^{M} \alpha_m f_m(\mathbf{x}_i))$$

Cross-Validation (k-fold or leave-one-out) can be used to prevent the ensemble model from overfitting.

Ensemble Learnering with weak learners

The **Hypothesis Boosting problem** (Kearns and Valiant, 1989): Can there exist a method that combines lower accuracy (weaker) learning models to form a model that has (theoretically guaranteed) higher (*boosted*) accuracies.[1]

Schapire (1990)[2] proved the existence of such methods.

A consequence of his proof was that as long as the weak learners were better than random guessing (i.e., accuracy $> 50\%$ for binary classification problems), the combined model can be have arbitrary small error.

---

[1] M. Kearns and L. G. Valiant, Crytographic limitations on learning Boolean formulae and finite automata, in Proc. 21st Annu. ACM Symp. Theory Comput. (STOC), pp. 433-444, 1989.

[2] Schapire, R.E. The strength of weak learnability. Mach Learn 5, 197-227, 1990.

AdaBoost Classification

The AdaBoost algorithm[3] involves training M weak learners sequentially in M number of steps.

In each step, one weak learner is trained on a weighted version of the data.

At step i, larger weights are given data instances that were misclassified several times in the previous $1, ..., (i-1)$ rounds.

[3] Yoav Freund, Robert E Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences, Volume 55, Issue 1, Pages 119-139, 1997.

AdaBoost Classification

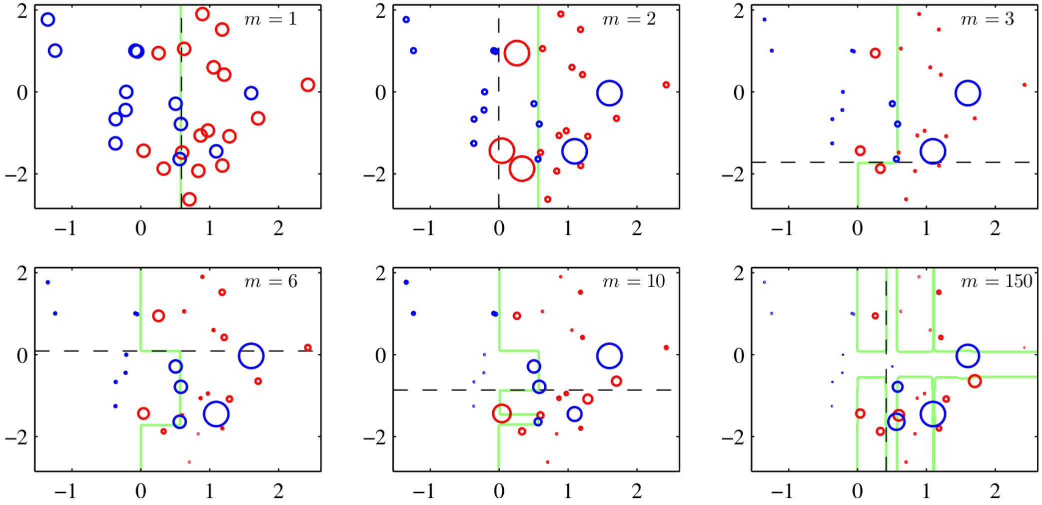

Image Source: Christopher M. Bishop, Chapter 14 Combining Models, *Pattern Recognition and Machine Learning*, Springer, 2006.

AdaBoost Classification

The AdaBoost algorithm involves training M weak learners $k_1, ..., k_M$ sequentially in M number of steps.

$$\hat{f}(\mathbf{x}_i) = \alpha_1 k_1(\mathbf{x}_i) + ... + \alpha_M k_M(\mathbf{x}_i)$$

In each step, one weak learner is trained on a weighted version of the data.

$$\text{After step } m-1\text{:}\ \hat{f}^{(m-1)}(\mathbf{x}_i) = \alpha_1 k_1(\mathbf{x}_i) + ... + \alpha_{m-1} k_{m-1}(\mathbf{x}_i)$$

$$\text{At step } m\text{:}\ \hat{f}^{(m)}(\mathbf{x}_i) = \hat{f}^{(m-1)}(\mathbf{x}_i) + \alpha_m k_m(\mathbf{x}_i)$$

After M steps, $\hat{f} = \hat{f}^{(M)}$

AdaBoost Classification

AdaBoost learns **instance weights** $w_1, ..., w_n$ so that at step i, larger weights are given to data instances that were misclassified several times in the previous $(i-1)$ steps.

Let us define an error at the m-th step as,

$$E = \sum_{i=1}^{N} e^{-y_i(\hat{f}^{(m-1)}(\mathbf{x}_i)+\alpha_m k_m(\mathbf{x}_i))}$$

The instance weights that will be learnt from the m-th step are:

$$w_i^{(m)} = e^{-y_i \hat{f}^{(m-1)}(x_i)} \ , \quad i = 1, ..., n$$

The error can then be written as,

$$E = \sum_{\substack{i=1 \\ y_i = k_m(\mathbf{x}_i)}}^{N} w_i^{(m)} e^{-\alpha_m} + \sum_{\substack{i=1 \\ y_i \neq k_m(\mathbf{x}_i)}}^{N} w_i^{(m)} e^{\alpha_m}$$

AdaBoost Classification

$$E = \sum_{\substack{i=1 \\ y_i = k_m(\mathbf{x}_i)}}^{N} w_i^{(m)} e^{-\alpha_m} + \sum_{\substack{i=1 \\ y_i \neq k_m(\mathbf{x}_i)}}^{N} w_i^{(m)} e^{\alpha_m} = \omega_c e^{-\alpha_m} + \omega_\epsilon e^{\alpha_m}$$

How can we find suitable values for α_m?

$$\begin{aligned} \frac{\partial}{\partial \alpha_m} E &= -\omega_c e^{-\alpha_m} + \omega_\epsilon e^{\alpha_m} = 0 \\ \Longrightarrow & -\omega_c + \omega_\epsilon e^{2\alpha_m} = 0 \\ \Longrightarrow \alpha_m &= \frac{1}{2} \ln\left(\frac{\omega_c}{\omega_\epsilon}\right) = \frac{1}{2} \ln\left(\frac{\omega - \omega_\epsilon}{\omega_\epsilon}\right) = \frac{1}{2} \ln\left(\frac{1 - r_m}{r_m}\right) \end{aligned}$$

Here the sum of weights is $\omega = \omega_c + \omega_\epsilon$, and $r_m = \omega_\epsilon / \omega$.

The AdaBoost Algorithm

1. Select one classifier from the pool of M classifiers, which minimizes

$$\omega_\epsilon = \sum_{y_i \neq k_m(\mathbf{x}_i)} w_i^{(m)}$$

2. Set $\alpha_m = \frac{1}{2} \ln\left(\frac{1 - r_m}{r_m}\right)$, where $r_m = \frac{\omega_\epsilon}{\omega}$

3. Update the weights of the data instances.

If $k_m(\mathbf{x}_i)$ is a missclassification, then set:

$$w_i^{(m+1)} = w_i^m e^{\alpha_m} = w_i^m \sqrt{\frac{1 - r_m}{r_m}}$$

Othwerise if $k_m(\mathbf{x}_i)$ is correct, then set:

$$w_i^{(m+1)} = w_i^m e^{-\alpha_m} = w_i^m \sqrt{\frac{r_m}{1 - r_m}}$$

Weak Learners: Decision Stumps

An example of a weak learner is a decision stump $k(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$ that has output $\{+1, -1\}$.

$$k(\mathbf{x}) = s(\mathbf{x}_{(p)} > c)$$

To learn a decision stump, three parameters need to be learnt:

$$(i) c \in \mathbb{R} \qquad (ii) p \in \{1, ..., d\} \qquad (iii) s \in \{-1, +1\}$$

Random Forests

Bootstrap Aggregating (Bagging): From a dataset X with n instances $\mathbf{x}_1, ..., \mathbf{x}_n$, M new training sets are created by sampling with replacement:

$$X_i = \left[\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \mathbf{x}_{i_{n_i}}\right], \quad i = 1, ..., M$$

For classification problems, bagging is generally used to train an ensemble of models $k_1, ..., k_M$, where each model captures the general trends in a dataset and does not overfit it. The models are trained to obtain the final ensemble:

$$\hat{f}(\mathbf{x}) = \max\ k_m(\mathbf{x})$$

The individual models trained using bagging tend to be highly correlated with each other. Random Forests aim to reduce this correlation, by training each model on a random subset of the features.

Random Forests

Random Forests approach: Train M weak learners (usually decision trees) on:

1. A random subset of training instances (bagging)
2. A random subset of features

Predictions are obtained as the majority predicted class from the trained weak learners.

$$\hat{f}(\mathbf{x}) = \max\ k_m(\mathbf{x})$$

Wainberg, Michael, Babak Alipanahi, and Brendan J. Frey. "Are random forests truly the best classifiers?." The Journal of Machine Learning Research 17.1 (2016): 3837-3841.

References

[1] Raul Rojas. AdaBoost and the Super Bowl of Classifiers - A Tutorial Introduction to Adaptive Boosting. Freie Universit¨at Berlin. 2009.

[2] Aaron Hertzmann, David J. Fleet and Marcus Brubaker. Lecture 18 Adaboost. CSC 411 / CSC D11 / CSC C11. University of Toronto. 2015.

[3] Ibomoiye Domor Mienye, Yanxia Sun. A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects. IEEE Access. Vol 10, 2022.

[4] B. V. Dasarathy and B. V. Sheela. A composite classifier system design: Concepts and methodology. Proc. IEEE. Vol 67, no. 5, pp. 708-713, 1979.

[5] Lior Rokach. Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. Computational Statistics and Data Analysis. Vol 53, pp. 4046-4072, 2009.

References

[6] M. Kearns and L. G. Valiant, Crytographic limitations on learning Boolean formulae and finite automata, in Proc. 21st Annu. ACM Symp. Theory Comput. (STOC), pp. 433-444, 1989.

[7] Schapire, R.E. The strength of weak learnability. Mach Learn 5, 197-227, 1990.

[8] Yoav Freund, Robert E Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences, Volume 55, Issue 1, Pages 119-139, 1997.

[9] Kervin Murphy. Chapter 16. Machine Learning A Probabilistic Perspective. MIT Press. 2012.