ADABOOST

1. Giới thiệu

AdaBoost (Adaptive Boosting) là một thuật toán Machine Learning thuộc nhóm bài toán Boosting, nằm trong nhóm các phương pháp Ensemble Learning, nghĩa là kết hợp nhiều mô hình học máy độc lập với nhau để cho ra kết quả cuối cùng. Thuật toán này có thể sử dụng cho cả hai bài toán Classification và Regression, nhưng thường là Classification, dùng để giải quyết các bài toán thuộc nhóm “Học có giám sát” (Supervised Learning).

Điểm đặc biệt của thuật toán này đó là nó tập trung vào những điểm dữ liệu khó (bị dự đoán sai) từ mô hình trước để cải tiến mô hình sau đó nhằm dự đoán được những điểm dữ liệu này). Việc học tuần tự qua nhiều mô hình cơ sở (base model hoặc weak-learner) có độ quan trọng khác nhau sẽ tạo thành một mô hình dự đoán tốt (strong-learner) cuối cùng.

1. Chi tiết thuật toán

Nguyên lý hoạt động của AdaBoost sẽ gồm các bước như sau:

1. Gán trọng số cho tất cả điểm dữ liệu

Đầu tiên, thuật toán sẽ gán cho mỗi điểm dữ liệu trong bộ dữ liệu một trọng số (gọi là sample weights) với giá trị độ lớn như nhau theo công thức:

trong đó: N là số điểm dữ liệu

Với công thức trên thì giá trị trọng số của mỗi sample sẽ nằm trong khoảng (0, 1) và tổng độ lớn của chúng luôn bằng 1.

1. Khởi tạo bộ dự đoán (mô hình) để dự đoán dữ liệu

Đối với thuật toán AdaBoost, mỗi mô hình nhỏ (weak-learner) thường sẽ là những mô hình đơn giản (nhưng vẫn có thể dự đoán tốt hơn dự đoán ngẫu nhiên) và độc lập nhau như: Linear Regression, Decision Tree, … Nhưng thường dùng nhất vẫn là Decision Tree với chỉ 1 node gốc và 2 node lá (độ sâu của cây là 1) được gọi là stump (tạm dịch: gốc). Lý do vì sao lại chọn cấu trúc như này sẽ được giải thích ở phần cuối.

Để chọn được giá trị và thuộc tính phân chia trong stump, tương tự như trong mô hình Decision Tree, ta có thể dùng các giá trị Entropy – Information Gain hoặc Gini Index để chọn được thuộc tính phân chia dữ liệu tốt nhất. Nếu sử dụng Gini Index thì ta sẽ chọn thuộc tính có Gini nhỏ nhất để làm thuộc tính phân loại.

1. Tính độ lỗi (total\_error hay r) của mô hình:

Giá trị độ lỗi của mô hình chính là tổng trọng số (sample weights) của những điểm dữ liệu bị dự đoán sai.

1. Tính trọng số cho mô hình (Amount of Say)

Trọng số của mô hình(nhỏ) là giá trị quyết định độ quan trọng của mô hình này đối với mô hình lớn cuối cùng, được tính theo công thức sau:

Trong đó:

* α: Trọng số của mô hình
* : Tốc độ học (learning rate, thuật toán AdaBoost gốc sử dụng giá trị ).
* : Độ lỗi của mô hình

Với trường hợp r = 0, dựa theo công thức ta sẽ không tính toán được, nên công thức thực tế thường được cộng thêm một giá trị để tránh trường hợp chia số 0.

1. Cập nhật trọng số sample weights cho các điểm dữ liệu

Do ta muốn mô hình sau tập trung vào những điểm bị dự đo án sai ở mô hình này, nên ta cần trọng số của những điểm sai sẽ lớn, còn những điểm đúng sẽ nhỏ hơn. Vì vậy ta sử dụng công thức:

Nếu tổng của các trọng số chưa bằng 1, ta cần bước chuẩn hóa các trọng số theo công thức:

1. Khởi tạo mô hình tiếp theo dựa trên các trọng số từ mô hình trước

Mô hình sau sẽ tập trung hơn vào các điểm dữ liệu sai (trọng số lớn) trong mô hình trước bằng 2 cách:

* Một là giữ nguyên bộ dữ liệu, nhưng thay vì tính Gini Index để chọn thuộc tính phân loại cho stump, ta sẽ tính Weighted Gini Index, nghĩa là điểm Gini có kèm trọng số từ mô hình trước.
* Hai là tạo bộ dataset mới (bootstrap dataset) chứa phần lớn là các điểm dữ liệu sai nhờ vào các trọng số.

Cả 2 cách này đều giúp mô hình sau tập trung dự đoán đúng những điểm dữ liệu bị dự đoán sai từ mô hình trước. Và ta sẽ lặp lại tất cả các bước trên cho đến khi không có dòng dữ liệu nào bị dự đoán sai hoặc số mô hình được tạo ra đạt một mốc nhất định.

1. Cách thuật toán AdaBoost dự đoán nhãn một sample mới

Khi dự đoán, dữ liệu sẽ được đưa qua tất cả các mô hình con trong thuật toán và ghi lại kết quả của từng mô hình (ở ví dụ này là có bệnh hoặc không, 0 hoặc 1). Sau đó ta tính tổng trọng số của các mô hình ứng với kết quả mà nó dự đoán, rồi chọn tổng nào lớn hơn thì kết quả sẽ là nhãn đó.

1. Các siêu tham số trong mô hình:

* **base\_estimator**: chính là các mô hình nhỏ mà ta chọn để huẩn luyện mô hình chung. Các mô hình thường được sử dụng là Decision Tree, Linear Regression, SVM, …
* **n\_estimators**: số lượng mô hình con được tạo ra và sử dụng trong AdaBoost.
* **learning\_rate**: tốc độ học của từng mô hình con. Nếu learning\_rate nhỏ thì n\_estimators lớn và ngược lại.

Lưu ý rằng ở mỗi mô hình con cũng sẽ có các siêu tham số tương ứng ta cần chọn. Để tránh trường hợp quá khớp (overfitting), người ta thường chọn các mô hình con có cấu trúc đơn giản, hoặc điều chỉnh cấu trúc nếu mô hình con quá phức tạp, hoặc giảm số lượng mô hình cơ sở trong quá trình huấn luyện.

1. Đánh giá thuật toán:

**Ưu điểm:**

* Dễ sử dụng và ít điều chỉnh tham số.
* Sử dụng base model là Decision Tree (độ sâu = 1) sẽ mô hình ít bị overfitting hơn do mỗi mô hình chỉ có 1 node gốc và 2 node lá (lí giải lý do chọn cấu trúc này).
* Cho ra kết quả tốt hơn các mô hình cơ bản, riêng lẻ như Linear, Logistic Regression, Decision Tree, … do đặc tính cải thiện khuyết điểm qua từng mô hình.
* Hiện nay đã được phát triển để phân loại hình ảnh và văn bản.

Nhược điểm:

* Phụ thuộc nhiều vào bộ dữ liệu, đòi hỏi bộ dữ liệu phải sạch và ít điểm gây nhiễu vì ý tưởng chính của AdaBoost là khớp từng điểm dữ liệu một cách hoàn hảo.
* Do sử dụng phương pháp học tuần tự (sequencial), nên ta chỉ có thể huấn luyện bộ dự đoán mới chỉ khi đã hoàn thành huấn luyện và đánh giá bộ dự đoán trước đó. Vì vậy, nhìn chung kĩ thuật này sẽ không mở rộng tốt bằng các thuật toán Ensemble khác (bagging, pasting).
* Đây là thuật toán cơ bản và đầu tiên trong nhánh Boosting nên ngoài nó còn có các thuật toán khác có hiệu quả và tốc độ xử lý nhanh hơn như GradientBoosting hay XGBoost, …

1. Tổng kết

Khi nhắc đến thuật toán Adaptive Boosting (AdaBoost), có 3 đặc điểm chính ta cần nhớ:

* AdaBoost tổng hợp nhiều mô hình con (nhỏ, yếu) lại với nhau để đưa ra kết quả dự đoán. Những mô hình con sẽ có cấu trúc rất đơn giản, thường dùng nhất là DecisionTree với cấu trúc “stump” (gốc cây) với 1 node gốc và 2 node lá.
* Mỗi mô hình con trong Adaboost sẽ có độ ảnh hưởng (Amount of Say) khác nhau đến kết quả cuối cùng.
* Các mô hình con được tạo ra lần lượt, nối tiếp nhau, với mô hình sau sẽ tập trung hơn vào lỗi từ mô hình trước.

<https://www.mygreatlearning.com/blog/adaboost-algorithm/#:~:text=AdaBoost%20algorithm%2C%20short%20for%20Adaptive,assigned%20to%20incorrectly%20classified%20instances.>

<https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/>

<https://viblo.asia/p/adaboost-buoc-di-dau-cua-boosting-gAm5yrGwKdb>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html>