Team Cute
LightGBM

Trung-Trực Trần, Nhung Nguyễn

Ngày 6 tháng 7 năm 2024

1 Giới thiệu LightGBM

LightGBM (viết tắt của Light Gradient Boosting Machine) là một thuật toán học máy tăng cường (boosting algorithm) được sử dụng phổ biến trong các bài toán phân loại và hồi quy. Thuật toán này nổi tiếng bởi tốc độ huấn luyện nhanh, hiệu suất cao và khả năng xử lý dữ liệu lớn.

2 Ý tưởng chính của LightGBM

2.1 Gradient-based One-Side Sampling (GOSS)

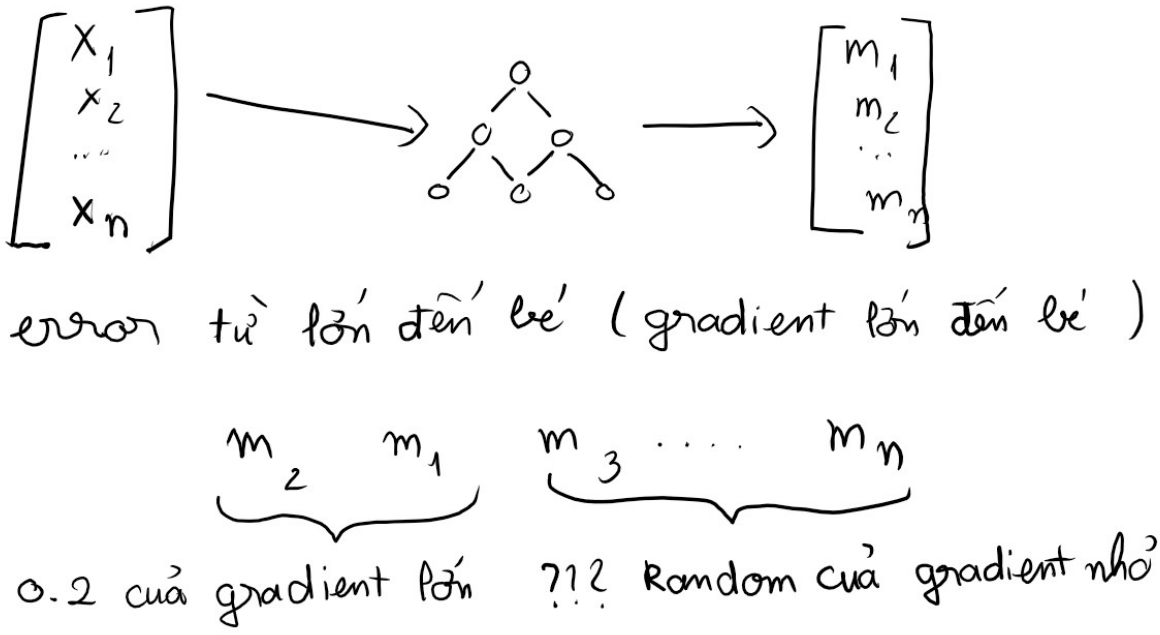
Bối Cảnh

- Gradient-based One-Side Sampling (GOSS) được phát triển để giải quyết vấn đề hiệu suất tính toán khi huấn luyện các mô hình Gradient Boosting Decision Trees (GBDT) với các tập dữ liệu lớn. GOSS giữ lại các mẫu có gradient lớn, vì chúng đóng vai trò quan trọng trong việc cải thiện mô hình, và ngẫu nhiên loại bỏ các mẫu có gradient nhỏ, sau đó điều chỉnh trọng số của các mẫu giữ lại để đảm bảo sự cân bằng. Kỹ thuật này giúp giảm khối lượng tính toán cần thiết mà vẫn duy trì được độ chính xác của mô hình, từ đó nâng cao hiệu suất tổng thể của GBDT.

Ý tưởng chính

- Dựa vào gradient của các mẫu dữ liệu để chọn lọc và giữ lại những mẫu có gradient lớn, vì chúng đóng vai trò quan trọng hơn trong việc tối ưu hóa mô hình. Các mẫu này thường chứa nhiều thông tin hơn và giúp mô hình học được các đặc trưng quan trọng của dữ liệu. Ngược lại, các mẫu có gradient nhỏ ít quan trọng hơn được loại bỏ ngẫu nhiên để giảm số lượng mẫu cần xử lý, giúp giảm khối lượng tính toán mà vẫn đảm bảo hiệu quả huấn luyện. Sau khi chọn mẫu, GOSS điều chỉnh trọng số của các mẫu được giữ lại để chúng phản ánh đúng tỷ lệ và sự đa dạng của toàn bộ dữ liệu ban đầu. Điều này giúp duy trì độ chính xác của việc ước lượng độ lợi thông tin (information gain), đặc biệt khi giá trị của độ lớn thông tin có phạm vi lớn. Nhờ vậy, GOSS không

chỉ tăng tốc độ huấn luyện mà còn duy trì được độ chính xác cao của mô hình, làm cho quá trình huấn luyện GBDT hiệu quả hơn rất nhiều.



Hình 1: Mô tả về GOSS

Lí do random gradient nhỏ

- Để đánh mạnh vào các Gradient lớn (Error lớn) để model tập trung huấn luyện vào tập dữ liệu có gradient lớn, bớt chú ý dữ liệu có gradient nhỏ lại nhằm cho máy có dự đoán tốt hơn vào lần sau, giảm bias(Mô hình sẽ có được độ phức tạp trong tập dữ liệu thông qua gradient lớn và nhỏ =» Mô hình đủ mạnh để đưa ra dự đoán chính xác), tránh mất mát thông tin.

2.2 Exclusive Feature Bundling (EFB)

Bối Cảnh

- Không gian đặc trưng thưa thớt: Trong nhiều ứng dụng thực tế, mặc dù có nhiều đặc trưng (features), nhưng phần lớn các giá trị của các đặc trưng này là không. Ví dụ: trong khai thác văn bản, các từ thường được biểu diễn bằng các vector one-hot, trong đó chỉ có một giá trị là 1 và các giá trị còn lại là 0 [0,1,0,0].

Ý tưởng chính

1. Đặc trưng độc quyền (Exclusive Features): Trong một không gian thưa thớt, nhiều đặc trưng hiếm khi có giá trị khác không đồng thời. Nghĩa là, nếu một đặc trưng có giá trị khác không thì các đặc trưng khác thường có giá trị là không. Ví dụ: trong biểu diễn one-hot, nếu từ "A" có giá trị là 1 thì các từ khác đều có giá trị là 0.

2. Gộp đặc trưng: Do đó, chúng ta có thể gộp các đặc trưng độc quyền lại với nhau thành một đặc trưng mới mà không làm mất thông tin. Đặc trưng mới này sẽ có giá trị khác không nếu và chỉ nếu một trong các đặc trưng ban đầu có giá trị khác không.

Bộ dữ liệu ban đầu

Mẫu	A	B	C
1	1	0	0
2	0	1	0
3	0	0	1
4	1	0	0
5	0	1	0

Gộp đặc trưng độc quyền

Chúng ta sẽ gộp A và B thành một đặc trưng mới gọi là AB, theo nguyên tắc:

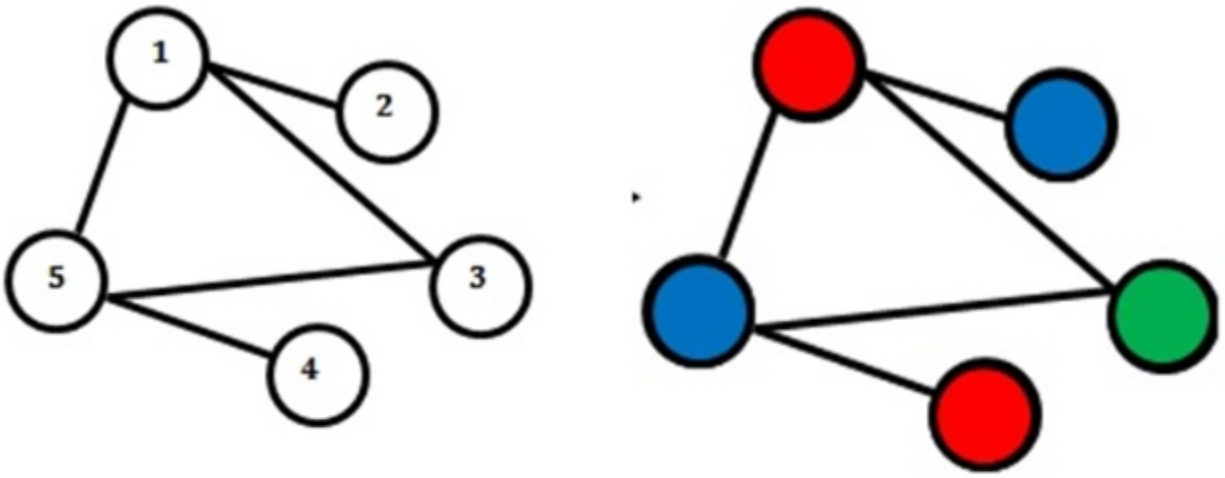
- Nếu A hoặc B có giá trị khác không, AB sẽ nhận giá trị đó.
- Nếu cả A và B đều bằng không, AB sẽ nhận giá trị 0.

Mẫu	AB	C
1	1	0
2	1	0
3	0	1
4	1	0
5	1	0

Hình 2: Ví dụ về gộp đặc trưng

Thuật toán: Bài toán tô màu đồ thị

1. Đồ thị đặc trưng: Xây dựng một đồ thị trong đó mỗi đặc trưng là một đỉnh.
2. Cạnh: Nối hai đỉnh bằng một cạnh nếu hai đặc trưng tương ứng không độc quyền (tức là chúng có thể có giá trị khác không đồng thời).
3. Bài toán tô màu đồ thị: Bài toán gộp đặc trưng được biến đổi thành bài toán tô màu đồ thị, trong đó mục tiêu là tô màu các đỉnh sao cho hai đỉnh nối bởi một cạnh không cùng màu. Mỗi màu sẽ tương ứng với một nhóm đặc trưng được gộp lại.



Bước 1: Ta có đồ thị có 5 đỉnh được đánh số 1, 2, 3, 4, 5 với các bậc tương ứng với từng đỉnh theo thứ tự là 3, 1, 2, 1, 3. Do đó V' ban đầu có thứ tự là [1, 5, 3, 2, 4]. Gán i =1.

Bước 2: Tô màu 1 (red) cho đỉnh 1. Lần lượt duyệt các đỉnh còn lại trong V':

Ta có: Đỉnh 5 kề đỉnh 1 (đỉnh 1 đã tô màu 1 - red) nên chưa tô màu cho đỉnh 5. Tương tự các đỉnh 3, 2 đều kề với đỉnh 1 nên đỉnh 3, 2 cũng chưa được tô màu.

Đỉnh 4 không kề với đỉnh 1, do đó thực hiện tô màu 1 cho đỉnh 4. Đỉnh 4 có màu 1 - red.

Bước 3: Kiểm tra thấy vẫn còn các đỉnh trong V chưa được tô màu nên chuyển sang bước 4.

Bước 4: Loại bỏ các đỉnh 1, 4 đã được tô màu ra khỏi V', sắp xếp lại V' theo thứ tự bậc giảm dần, ta thu được V'= [5, 3, 2]. Ta có i = 2. Thực hiện lặp lại bước 2:

Bước 2(1): Tô màu 2 (blue) cho đỉnh 5. Lần lượt duyệt các đỉnh còn lại trong V'. Ta có: Đỉnh 3 kề đỉnh 5 (đã tô màu 2 - blue) nên chưa tô màu cho đỉnh 3.

Đỉnh 2 không kề với đỉnh 5, do đó thực hiện tô màu 2 cho đỉnh 2. Đỉnh 2 có màu 2 - blue.

Bước 3(1): Kiểm tra thấy vẫn còn đỉnh 3 chưa được tô màu nên chuyển sang bước 4.

Bước 4(1): Loại bỏ các đỉnh 5, 2 đã được tô màu ra khỏi V', V'=[3]. Ta có i = 3. Thực hiện lặp lại bước 2:

Bước 2(2): Tô màu 3 (Green) cho đỉnh 3.

Bước 3(2): Kiểm tra thấy tất cả các đỉnh trong V đã được tô màu, thuật toán dừng lại. Kết luận: Đỉnh 1 và 4 được tô màu 1-red, đỉnh 5 và đỉnh 2 được tô màu 2-blue, đỉnh 3 được tô màu 3-Green. Số màu cần thiết phải sử dụng là i =3 màu.

Hình 3: Ví dụ về thuật toán tô màu đồ thị

Từ hình ảnh trên ta có thể gộp thành 3 nhóm lần lượt là: 1,4; 5,2; 3. Ta có thể thấy mô hình Từ 5 features giảm còn 3 features nghĩa là giảm đi hơn 2/3 số features, tương tự nếu scale up data càng lớn thì thuật toán tô màu đồ thị sẽ càng có lợi thế trong việc giảm thiểu thời gian chạy.

Bias-Variance Tradeoff

- **Bias-Variance Tradeoff** là sự cân bằng giữa bias và variance:
  - **Bias cao, Variance thấp**: Mô hình quá đơn giản, dẫn đến underfitting.
  - **Bias thấp, Variance cao**: Mô hình quá phức tạp, dẫn đến overfitting.

Hình 4: Chú ý về variance và bias

THAT'S IT, THANK YOU FOR READING ♡♡♡