**\*Thuật toán Random Forest**

-Sử dụng Cây quyết định(decision tree).

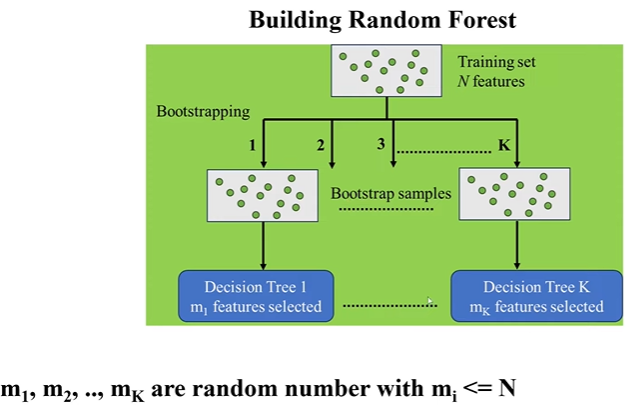
+Ưu điểm:dễ dàng xây dựng và sử dụng và giải thích

+Nhược điểm: rất nhạy cảm với training data(nếu thay đổi training data thì result sẽ khác) => sai số phương sai khá lớn(high Variance)=>Overfitting.

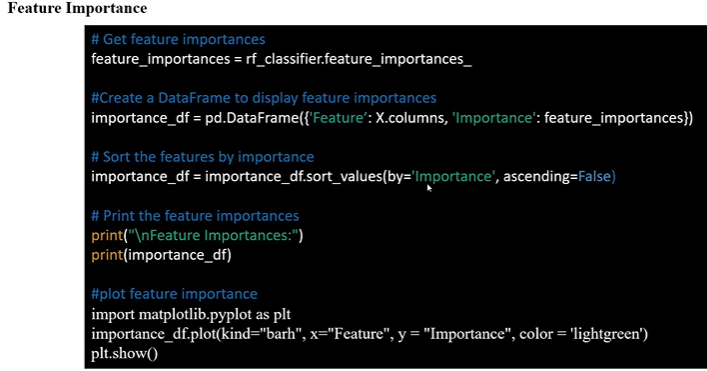
-Random Forest sẽ tận dụng ưu điểm và khắc phục nhược điểm của decision tree.

-Ý tưởng: xây dựng nhiều cây khác nhau trong 1 bộ dữ liệu sau đó tổng hợp tất cả các cây đưa ra predict cuối cùng (bằng vote hoặc lấy trung bình).

+Random các mẫu(bootstrap samples) để thực hiện cho các cây training nhưng các bootstrap samples sẽ được random các features cho mỗi cây=>Random forest làm cho đa dạng dữ liệu và đa dạng cây giúp giảm overfitting và kết quả tốt hơn.



-Có hàm feature\_importances\_ để lấy các feature quan trọng.



-Các tham số:n\_estimation:số cây(100-1000) default= 100.

+Bootstrap:Đây là cách tạo ra mẫu của câyTrue(default=True)

+Max\_features: số feature tối đa của mỗi mẫu có thể là từ 1 đến n-features (default=sqrt(n-features)).

+oob\_score: để đánh giá độ chính xác của mô hình không sử dụng test set (default=False).

+random\_state: được sử dụng để kiểm soát quá trình sinh số ngẫu nhiên. Ex random\_state=42. Đảm bảo rằng mỗi khi mô hình được huấn luyện lại, nó sẽ tạo ra cùng một bộ cây ngẫu nhiên và kết quả sẽ giống nhau, giúp việc so sánh mô hình trở nên nhất quán.

+Max\_depth:10-20(default=None).

+Min\_sample\_split:2-20(default=2).

+min\_samples\_leaf:Số lượng mẫu tối thiểu cho 1 nút lá :1-5(default=1).

+Ba giá trị này chỉ cần setup 1 giá trị là đủ

**\*Thuật toán logistic regression:**

-Thuật toán hồi quy nhưng thuộc nhóm các thuật toán phân lớp.

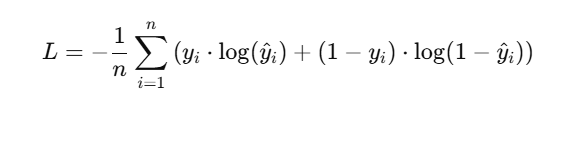
-Sử dụng hàm sigmoid logistic để trả về giá trị xác suất có được ánh xạ từ 2 hoặc nhiều lớp rời rạc-là các lớp phân loại.( khác linear regression).

-Đầu ra:0 -> 1(dựa vào hàm sigmoid).

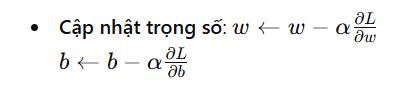
-Sau khi tính xác suất, **ngưỡng 0.5** thường được dùng để phân loại:

Vd p>=0,5 thì là class 1 còn <0,5 là class 0.

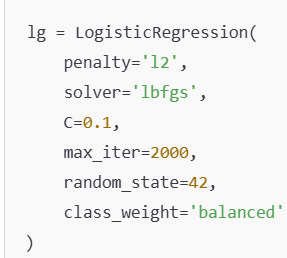
-Hàm lỗi :ở đây k nên dùng MSE mà nên dùng cross-Entropy



-Tối ưu trọng số w và b sử dụng Gradient descent.



-Các tham số cần lưu ý:



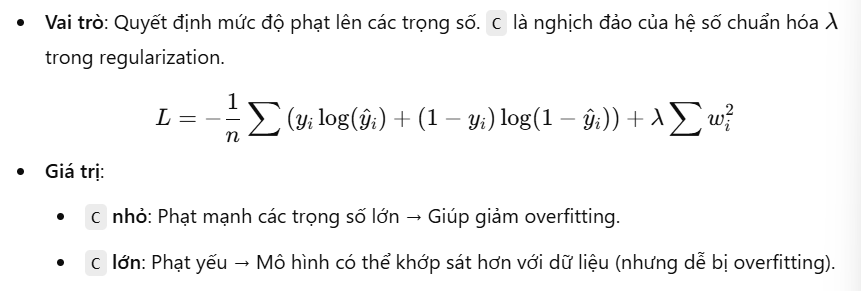
+ penalty: Mặc định: 'l2'. Quy định loại chuẩn hóa (regularization) được áp dụng.Tránh overfitting.

$ 'l2': Ridge Regularization (phổ biến).

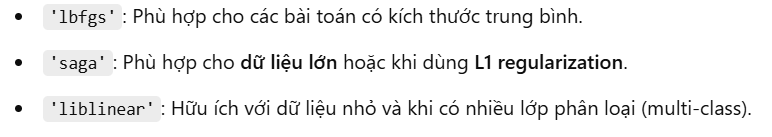
$ 'l1': Lasso Regularization.

+ **C** (Độ mạnh của Regularization):

$Mặc định: 1.0.



+solver (Thuật toán tối ưu):

$ Quyết định cách cập nhật trọng số và hội tụ. Tùy thuộc vào kích thước dữ liệu và loại regularization, bạn cần chọn solver phù hợp. 

+ **max\_iter** (Số lần lặp tối đa): Quy định số lần lặp để thuật toán tối ưu hóa hội tụ.

+ **class\_weight** (Trọng số của các lớp): giải quyết vấn đề dữ liệu mất cân bằng (khi một lớp có nhiều mẫu hơn lớp khác).