AI VIET NAM – COURSE 2022

XGBOOST

Ngày 10 tháng 3 năm 2024

Nguồn dữ liệu (nếu có):	XGBoost Slide, XGBoost Code Example
Từ khóa:	Decision Tree, Ensembles, Boosting, XGBoost
Người tóm tắt:	Bảo-Sơn Trần

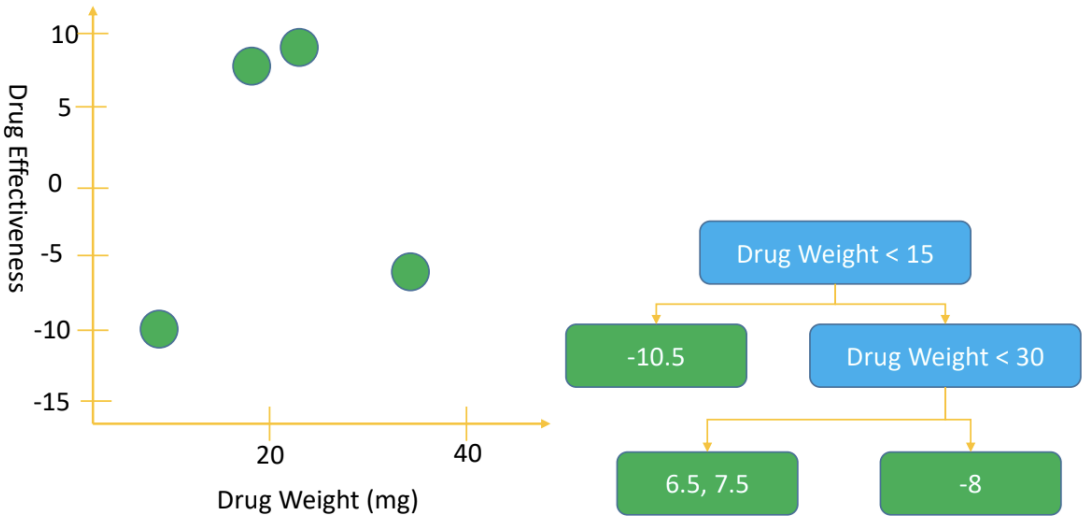
1. **Regularization**

   - Một trong những cải tiến của XGBoost so với Gradient Boost là khắc phục được trường hợp overfit
   - Reguralization là kỹ thuật để tránh trường hợp overfit của model. Kỹ thuật này thêm vào hàm loss một regugalization term để điều chỉnh model độ fit của model sang "just right" fit.
   - **[Phút 23]** Tương tự với trường hợp của linear regression khi thêm vào regularization term, XGBoost cũng áp dụng để tránh nhược điểm overfitting của Gradient Boosting bằng cách đưa vào một tham số λ vào hàm loss function

   Câu hỏi dặt ra ở đây là term λ này được đặt như thế nào vào model để xây dựng mô hình cây cốt hơn?

2. **XGBoost For Regression:**

   - Bài toán đầu tiên về việc áp dụng XGBoost cho bài toán regression. Dưới đây là ví dụ mình kiểm tra xem ứng với liều lượng thuốc nào thì nó hiệu quả trên từng bệnh nhân [1]
   - Các bước dùng để xây dựng mô hình XGBoost:
     - Khởi tạo giá trị dư đoán. Đối với ví dụ này giá trị khởi tạo dự đoán là 0.5, điều này có nghĩa là với bất kì liều lượng thuốc của từng bệnh nhân như thế nào, mô hình luôn dự đoán mức hiệu quả là 0.5
     - Tính lỗi của mô hình với từng giá trị dự đoán ban đầu (Residual Error) bằng công thức $-(Output - Predicted)$. Nhiệm vụ của XGBoost là thu hẹp những lỗi này lại đến khi gần bằng 0 (đối với trường hợp tối ưu)
       - Đối với mô hình Gradient Boost, mô hình sẽ dự đoán mức độ hiệu quả của thuốc. Khác với Gradient Boost, mô hình XGBoost sẽ dự đoán Residual Error và cố gắng minimize error này. Đối với bài toán regression, XGBoost sẽ sử dụng thước đo là Similarity Score (SC) [2]
       - term λ được đưa vào Similarity Score (SC) để giảm sự overfit của mô hình
     - Tương tự với Classification Decision Tree sử dụng Entropy hoặc Gini để split 1 node ra thành 2 leafs. XGBoost sẽ chia Residual Error thành 2 group dựa trên Information Gain từ Similarity Score. Và sẽ chọn giá trị threshold để split ở vị trí mà có Information Gain lớn nhất. Information Gain sẽ được tính bằng công thức



Hình 1: Bài toán đưa ra làm sao xây dựng mô hình XGBoost để dự đoán hiệu quả của thuốc với từng mức liều lượng. Kỳ vọng mô hình đầu ra sẽ như cây ở bên phải

$$InformationGain = LeftSC + RightSC - RootSC$$

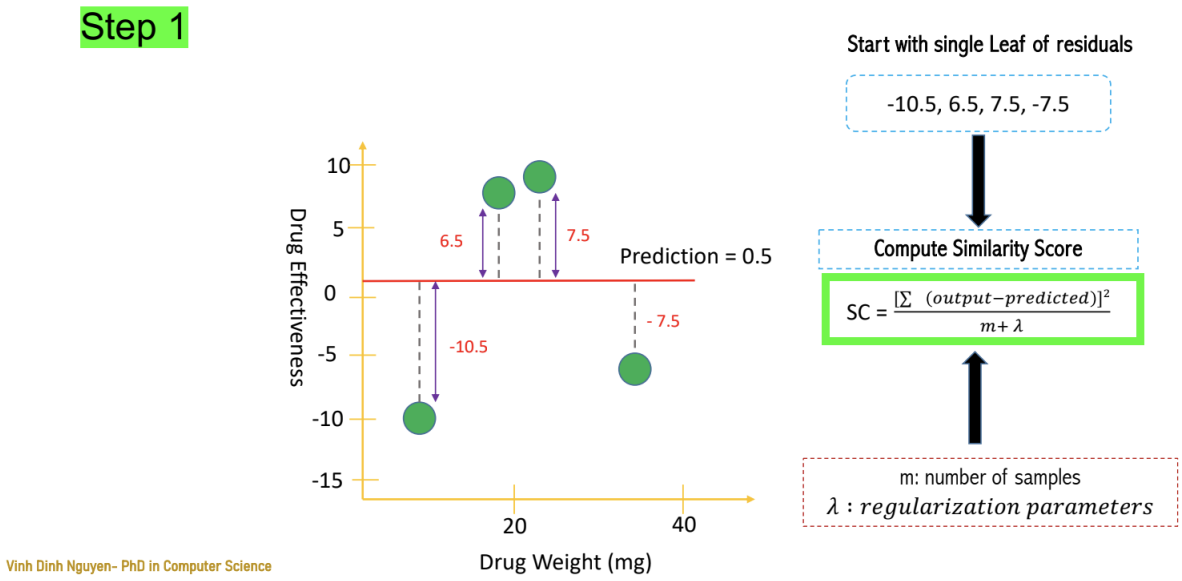
  - Nếu Residual Error rất khác nhau (dương và âm) sẽ triệt tiêu lẫn nhau dẫn đến SC sẽ nhỏ.
  - Residual giống nhau hoặc không triệt tiêu lẫn nhau, nên SC khá lớn.
- Nếu leaf chỉ có một giá trị, thì sẽ không split nữa (Trong ví dụ trên là giá trị -10.5). Nếu leaf có nhiều hơn 1 giá trị, sẽ tiếp tục tính Information Gain để tìm ra threshold tối ưu để split tiếp.
- Mô hình XGBoost ngoài sử dụng tham số λ để tránh overfitting, mô hình này có sử dụng thêm 1 tham số khác là γ cũng để tránh overfitting. Tham số γ này được đưa vào để cắt tỉa (prune) cây. Nguyên tắc hoạt động của γ như sau:
  - Tính $Difference = Gain - \gamma$
  - Nếu $Difference > 0$, sẽ không remove branch
  - Nếu $Difference < 0$, remove branch
- Idea của việc sử dụng tham số λ để tránh overfitting của XGBoost là khi đưa λ vào công thức tính SC, thì SC sẽ bị giảm, tương ứng sẽ làm giảm Information Gain. Đồng thời khi kết hợp với γ sẽ làm giảm vấn đề overfitting tốt hơn
- Sau khi cây đã được tạo thành, Output Value ở từng leaf sẽ được tính như sau

$$OutputValue = \frac{SumOfResiduals}{NumberofResiduals + \lambda}$$

- Prediction Value sẽ được mô tả như hình [3], sẽ tiếp tục xây dựng cây cho tới khi Residual chạm tới 1 mốc predefined threshold hoặc là đã chạm đến số lượng tree tối đa của mô hình

- Một điểm mạnh của XGBoost là có thể sử dụng kỹ thuật Quantile để tìm ra vị trí threshold để split tree dễ dàng hơn so với mô hình tree truyền thống, kỹ thuật này sẽ được đề cập ở bài tiếp theo là SVM

3. **XGBoost For Classification:**

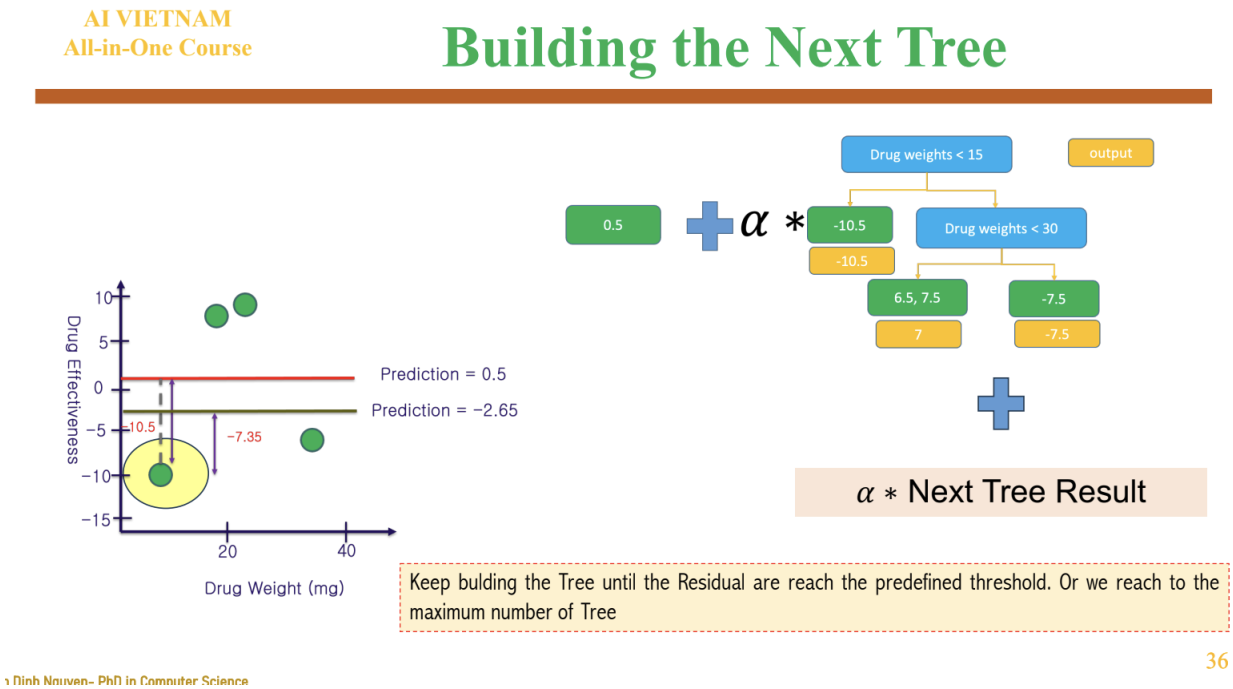


Hình 2: Tính Residual Error đối với từng data point và Similarity Score

- Khác với bài toán regression, khi ta có thể khởi tạo giá trị FirstPrediction là ngẫu nhiên. Đối với bài toán classification, biến đầu ra y chỉ nhận 2 giá trị là 0 và 1. Để mô hình hội tụ nhanh hơn, giá trị khởi tạo của mô hình được tính bằng công thức:

$$FirstPrediction = \frac{number\_of\_observation_having\_y=1}{totalobservation}$$

- Ví dụ minh hoa cho công thức trên ở 4
- Công thức để tính Similarity Score cho bài toán Classification và Regression như ảnh 5
- *PreviousProbability* trong công thức tính SC chính là xác suất ở 1 node mà các observation trong node đó có y=1 (Drug Effectiveness). Công thức tính tương tự như *FirstPrediction*
- Cách thức xây dựng mô hình XGBoost cho bài toán classification cũng giống như bài toán regression. Ta sẽ tính InformationGain ở từng threshold và rẽ nhanh một cái cây dựa trên threshold có InformationGain là lớn nhất. Chi tiết ở 6
- XGBoost có một tham số để giúp xác định việc có nên tiếp tục rẽ nhanh hay không là XGBoot Cover để estimate số lượng minimum Residuals của từng leaf, Nếu XGBoot cover ở 1 node lớn hơn con số mà mình set cho mô hình, thì mô hình sẽ tiếp tục rẽ nhanh cho node đó. By default thì XGBoot Cover được set là 1 7
- Cách XGBoost dự đoán cho bài toán classification
  - Tính Log(Odds) cho giá trị dự đoán khởi tạo (FirstPrediction)
  - Tính OutputValue ở từng Terminal Node theo công thức 8
  - Tính $Log(odds)$ cho data point cần dự đoán 9
  - Tính probability cho data point dự đoán đó 9
  - Sau khi có kết quả dự đoán, ta tìm được New Residual, update thông tin cho Previous-Probability của từng datapoint và tiếp tục xây dựng những cây tiếp theo
- Review Question 1 (2h:22'): Khi nào sẽ dừng việc xây dựng thêm cây?
  - Khi đạt được số cây maximum mà mình đang set
  - Khi Residual Error đã giảm về được mức mong muốn



Hình 3: Mô tả cách thức XGBoost xây dựng giá trị Prediction, và tiếp tục xây dựng những cây tiếp theo để dự đoán cho phần Residual còn lại

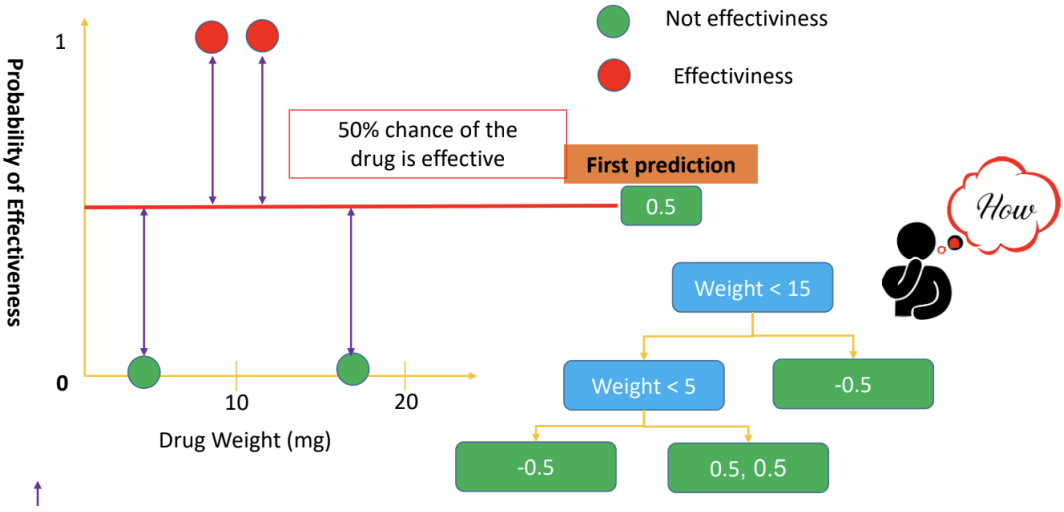
- Review Question 2: Nếu $\lambda > 0$ thì mô hình sẽ underfitting hay overfitting?
  - Nếu set $\lambda > 0$, giải thuật sẽ underfitting
  - Diễn giải chi tiết hơn cho phần này sẽ nằm ở mục Mathematical Explanation bên dưới

4. **XGBoost: Mathematical Explanation**

   Câu hỏi đặt ra ở phần này vì sao XGBoost được xây dựng các Similarity Score và Output Value như vậy?

   - Mô hình XGBoost được xây dựng dựa trên Loss Function 11 và 12
   - Sau khi xây dựng xong hàm loss, XGBoost sẽ xây dựng new tree dựa trên loss function. Mục tiêu là tìm giá trị dự đoán cho mỗi leaf P của cây mới nhằm minimize hàm loss 13
   - Nếu giá trị λ càng lớn, để minimize làm loss, P sẽ tiến dần về 0 và làm cho giá trị hàm loss không còn tối ưu như khi $\lambda = 0$ nữa, điều đó dẫn đến mô hình sẽ underfitting
   - Tương tự như các thuật toán khác, để minimize được hàm loss theo biến là P, chúng ta cần tính đạo hàm của hàm loss tại P. Tuy nhiên việc khai triển hàm loss theo P sẽ rất phức tạp, giải pháp ở đây là sử dụng Second Order Taylor Appriximation để tìm công thức xấp xỉ hàm loss nhưng ở dạng dễ tính đạo hàm hơn 14
   - Tính Output Value P từ đạo hàm của xấp xỉ hàm loss. Giá trị Output value P ở bài toán regression chi tiết ở 15. Giá trị Output value P ở bài toán classification chi tiết ở 16
   - Từ công thức trên ta nhận thấy g_i cũng chính là Residual Error. Đối với bài toán Regression, h_i là 1. Đối với bài toán Classification, $h_i = \sum \bar{y}_i * (1 - \bar{y}_i)$
   - Similarity Score Regression và Classification sẽ được mô tả ở ảnh 17 và 18

5. **How To Fill Missing Values:**



Hình 4: Mô tả cách thức tạo ra fist_prediction đối với bài toán classification của XGBoost, ở ảnh này ta có 2 điểm đỏ (y=1) và 2 điểm xanh (y=0), do đó first_prediction sẽ bằng 2 điểm đỏ chia cho 4 điểm bằng 0.5

- Phần này sẽ được discuss ở đầu buổi bài Support Vector Machine

6. **Example**

- Dataset và Code đính kèm ở đường link màu đỏ đầu bài tóm tắt
- Dataset ở ví dụ sẽ có những columns như ảnh 19
- Một số tham số quan trọng của mô hình
  - **learning_rate** α là tốc hộ học của mô hình, tương tự như learning_rate trong thuật toán gradient descent.
  - **max_depth** là độ sâu tối đa của mỗi cây trong mô hình
  - **subsample** tỉ lệ phần trăm số lượng ngẫu nhiên sample được lấy ra từ tập train của mô hình
  - **reg_lambda** regularization term đi liền với Output value để giảm overfitting cho mô hình
  - **gamma** regularization term dùng để prunning 1 tree
  - **min_child_weight** threshold cho XGBoost Cover để quyết định ở 1 node có nên tiếp tục rẽ nhanh hay không

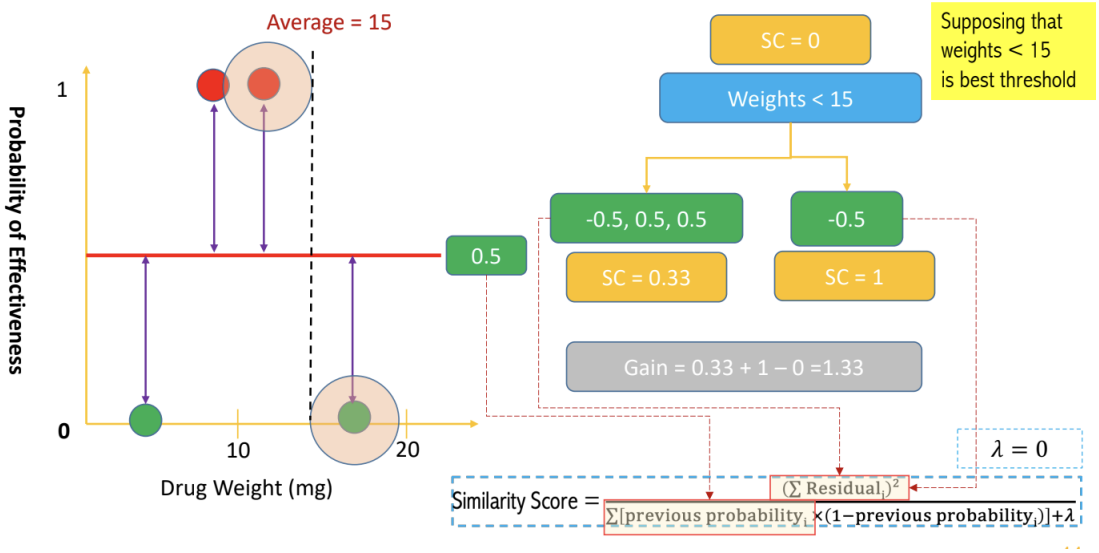
Similarity Score for Classification:

$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\sum[\text{previous probability}_i \times (1 - \text{previous probability}_i)] + \lambda}$$

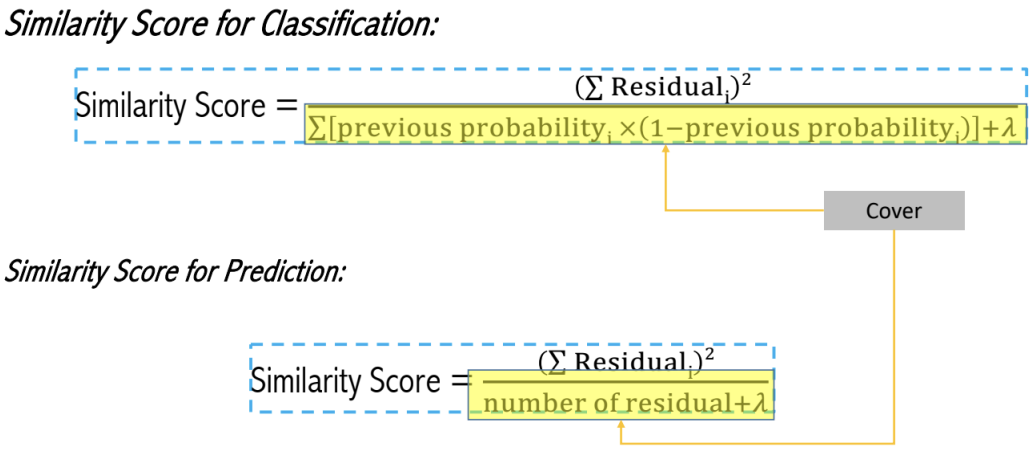
Similarity Score for Prediction (regression):

$$\text{Similarity Score} = \frac{(\sum \text{Residual}_i)^2}{\text{number of residual} + \lambda}$$

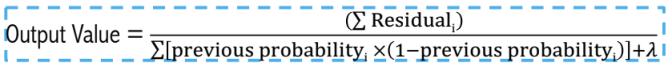
Hình 5: Hình ảnh minh hoạ cách tính Similarity Score cho mô hình XGBoost đối với bài toán classification và bài toán regression



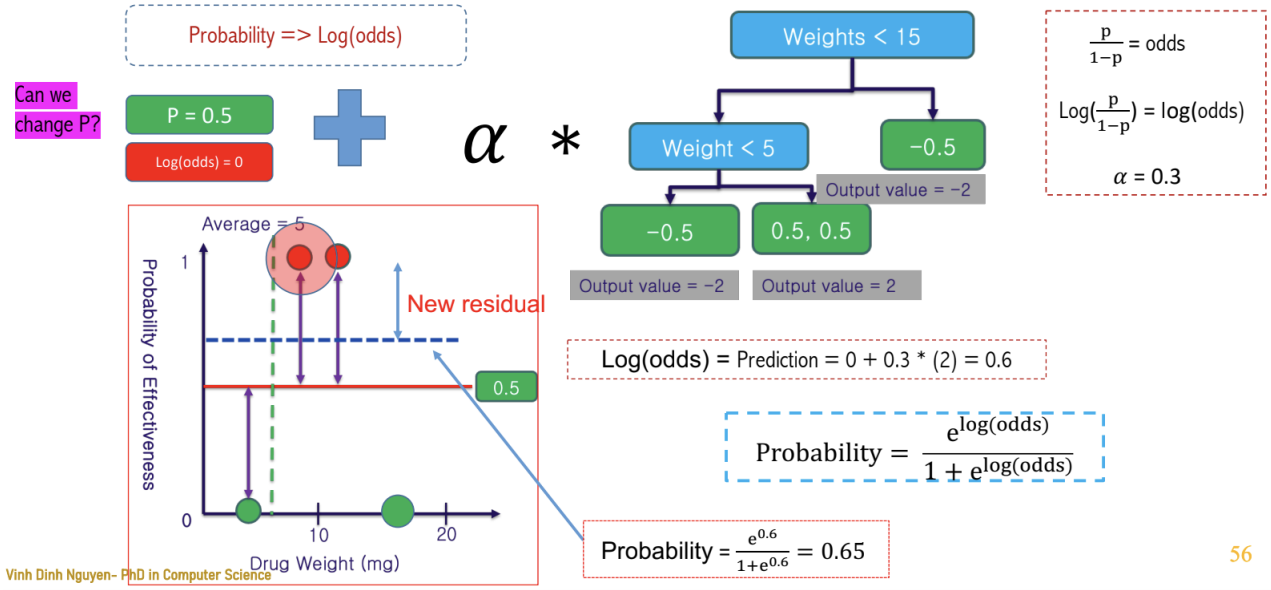
Hình 6: Ta sẽ tính InformationGain ở từng mốc threshold của DrugWeight. PreviousProbability trong node ban đầu cũng chính là FirstPrediction và bằng 0.5. Giả sử trong trường hợp này mức threshold 15 ta sẽ có được InformationGain là lớn nhất, ta sẽ rẽ nhanh cây ở mốc drug weight là 15.



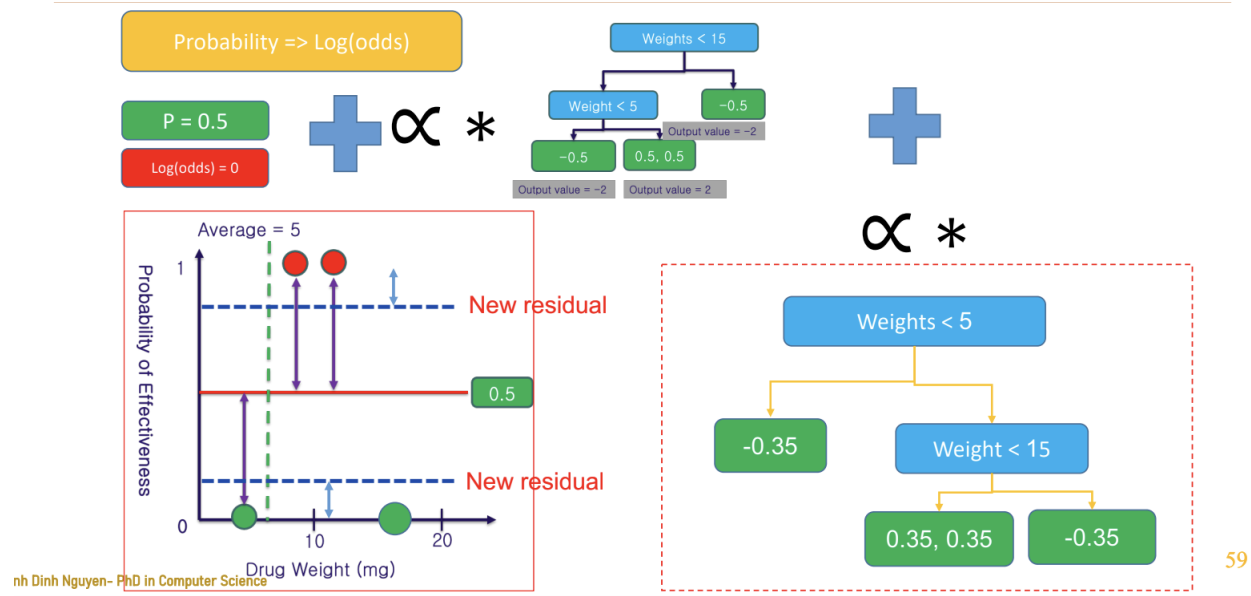
Hình 7: Công thức để tính XGBoot Cover cho từng node cho bài toán Regression và Classification, nếu XGBoot cover ở node đó lớn hơn giá trị mà ta set, thì ta quyết định sẽ tiếp tục rẽ nhanh ở node đó



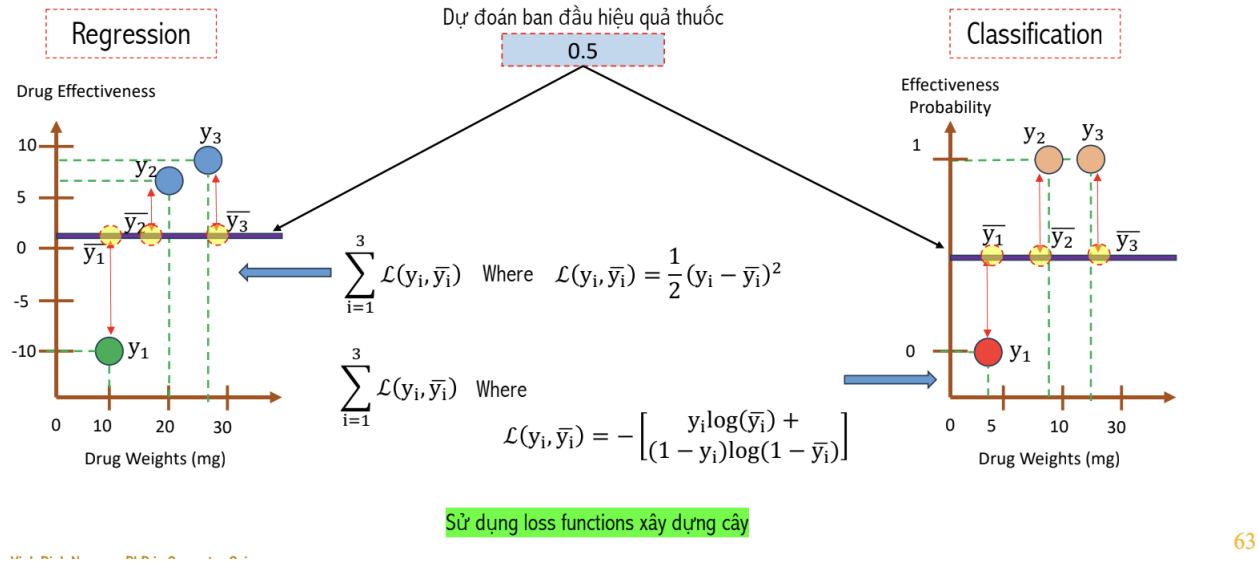
Hình 8: Công thức tính Output value ở từng terminal Node cho bài toán classification khi sử dụng mô hình XGBoost



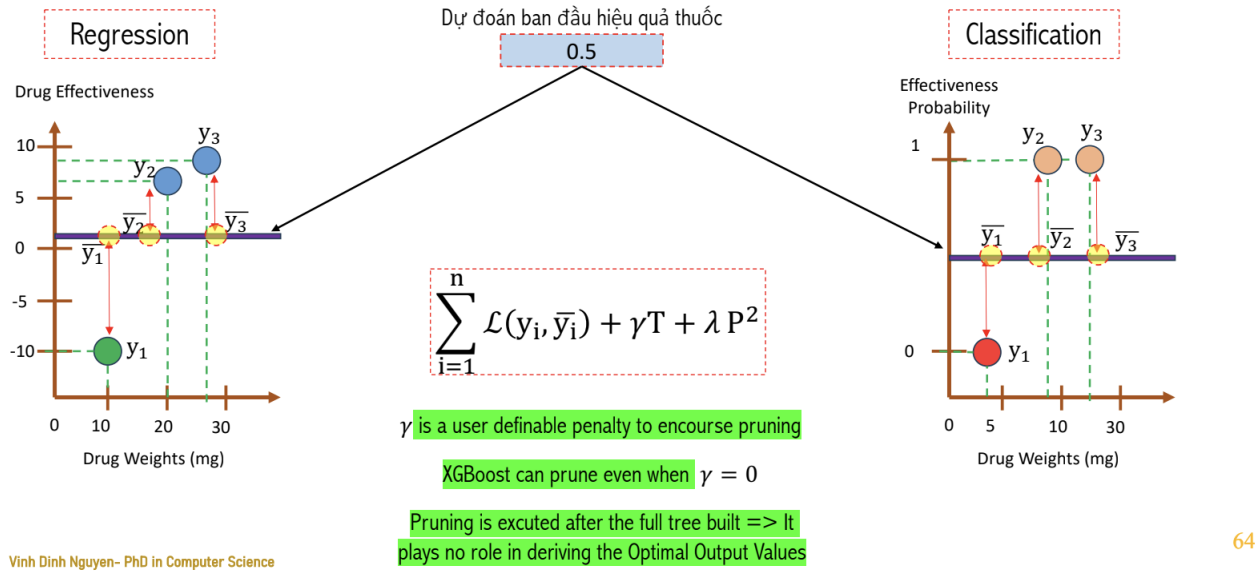
Hình 9: Ví dụ cách thức mô hình XGBoost dự đoán cho bài toán classfication cho data point có DrugWeight < 15 và DrugWeight > 5 (vòng tròn màu đỏ). Với giá trị khởi tạo FirstPrediction là 0.5, Log(Odds) tại FirstPrediction 0.5 là 0. OutputValue tại datapoint đó là 2. Do đó $Log(Odds = Prediction = Log(OddsOfFirstPrediction) + Learningrate(\alpha) * OutputValue = 0 + 0.3 * 2 = 0.6$. Khi Log(Odds) $= 0.6$, probability của drug effectiveness của datapoint bằng hàm logistic tại giá trị Log(Odds) là 0.65. Khi đó ta thấy New Residul đã giảm hơn so với mốc Residual ban đầu



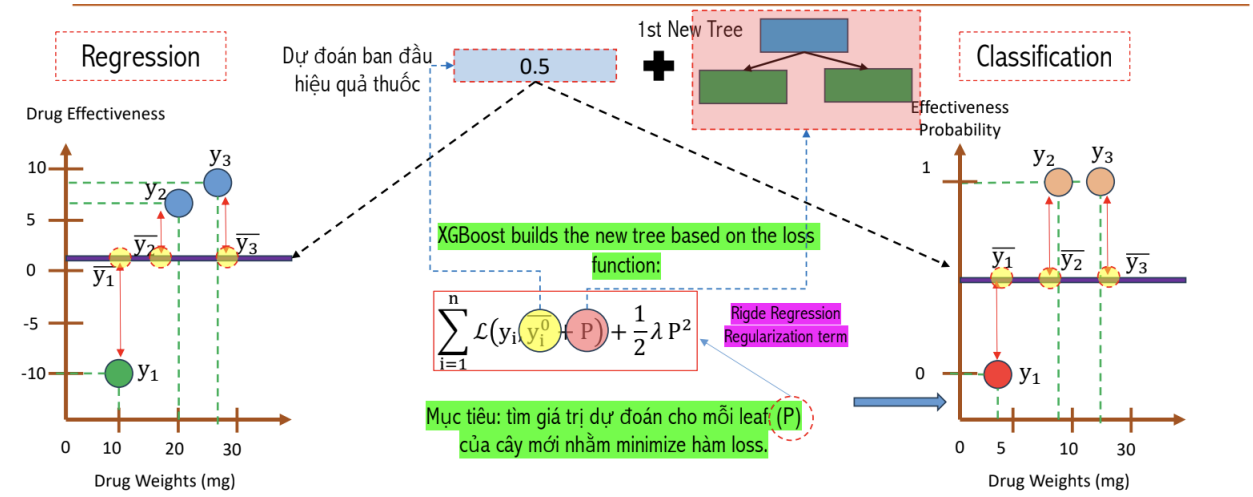
Hình 10: Sau khi build hoàn thành 1 cây, ta sẽ update lại thông tin PreviousPrediction và tính New Residual cho từng data point để xây dựng cây tiếp theo



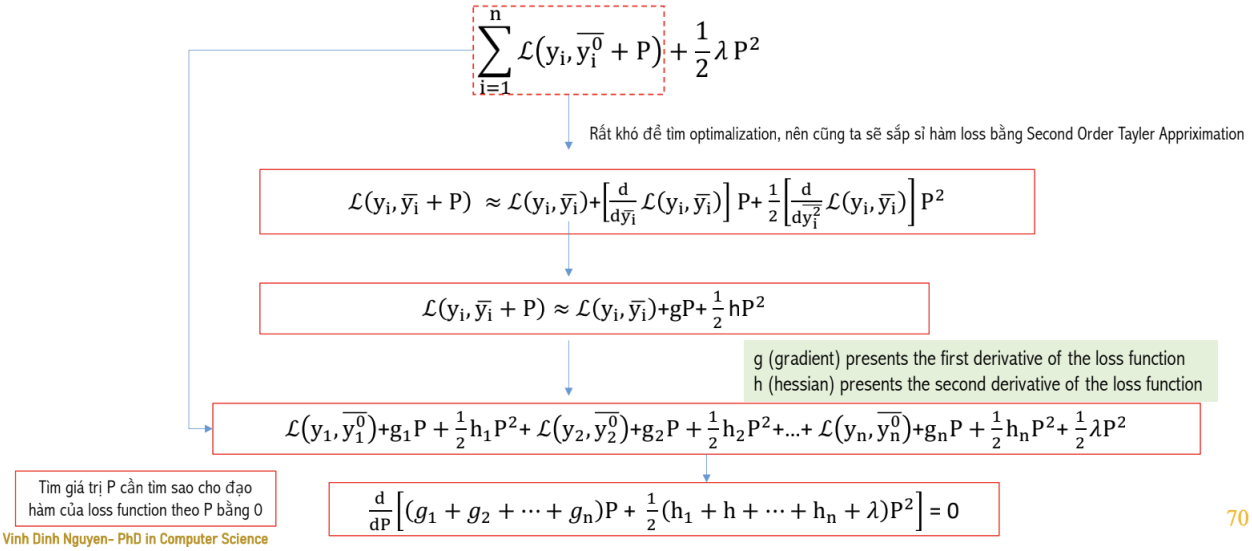
Hình 11: Loss Funtion bên trên là Loss Function đối với bài toán Regression, Loss Function bên dưới là Loss Function được sử dụng trong bài toán Classification. Loss Function của Classification sẽ được đào sâu hơn trong bài Logistic Regression. Một mục tiêu cung khi sử dụng hàm loss function là hàm loss phải là Convex để có thể dễ dàng hội tụ trong bài toán Regression hoặc Classification



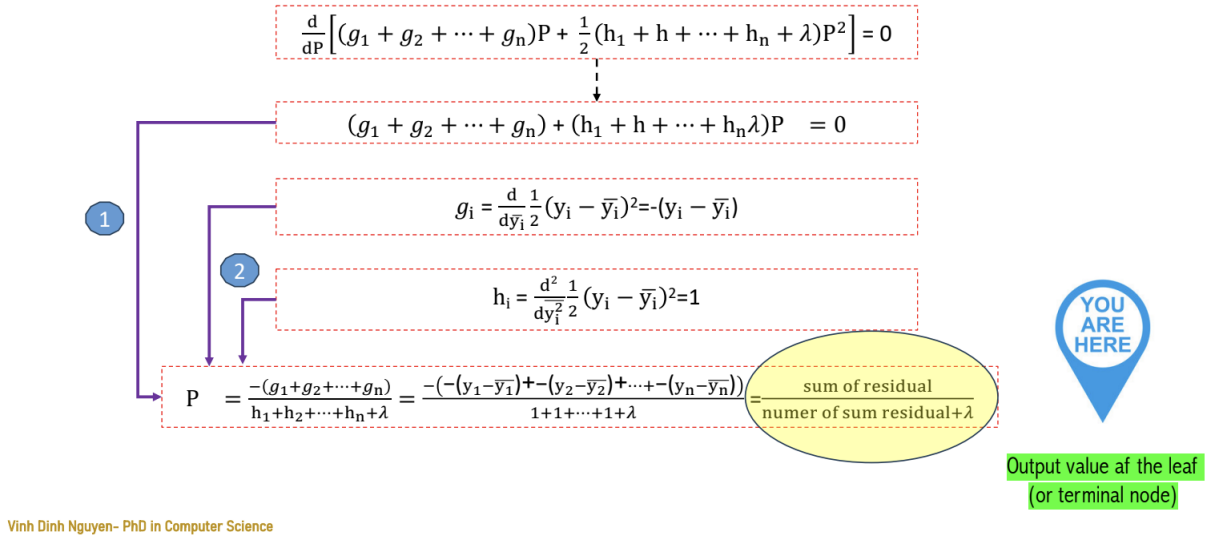
Hình 12: Loss Function được thêm vào các Regularization terms λ và γ. Lưu ý là tham số γ để prunning một cây chỉ được execute khi cây đã được xây dựng hoàn thành và nó sẽ không ảnh hưởng đến cách tìm được Optimal Output Value. T là số lượng leaf đầu ra của cây và P là giá trị output đầu ra của cây. Cây vẫn có thể thực hiện chức năng prunning khi set $\gamma = 0$



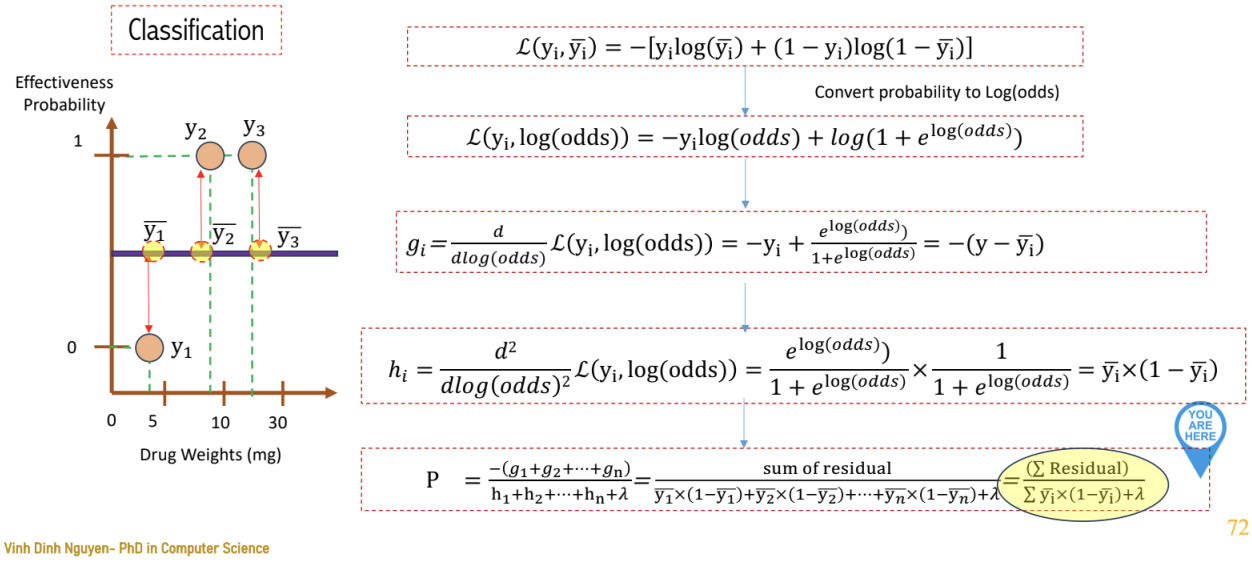
Hình 13: Mục tiêu của việc xây dựng new tree là tìm giá trị output P cho mỗi leaf để tiếp tục minimize hàm loss



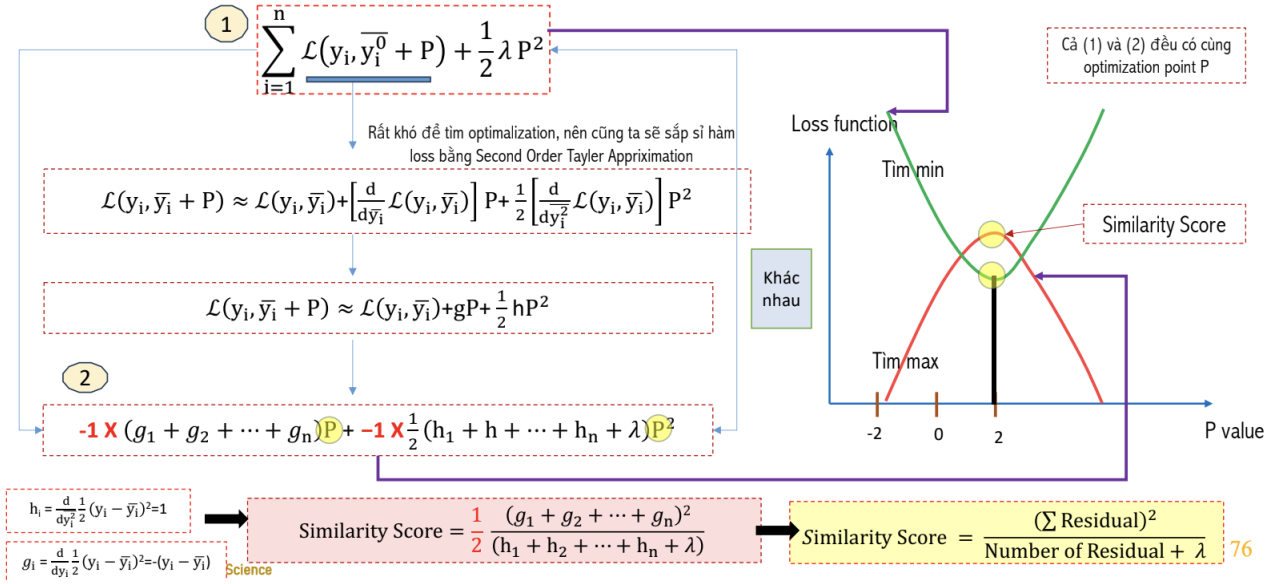
Hình 14: Xây dựng hàm xấp xỉ hàm loss bằng khai triển Taylor bậc 2



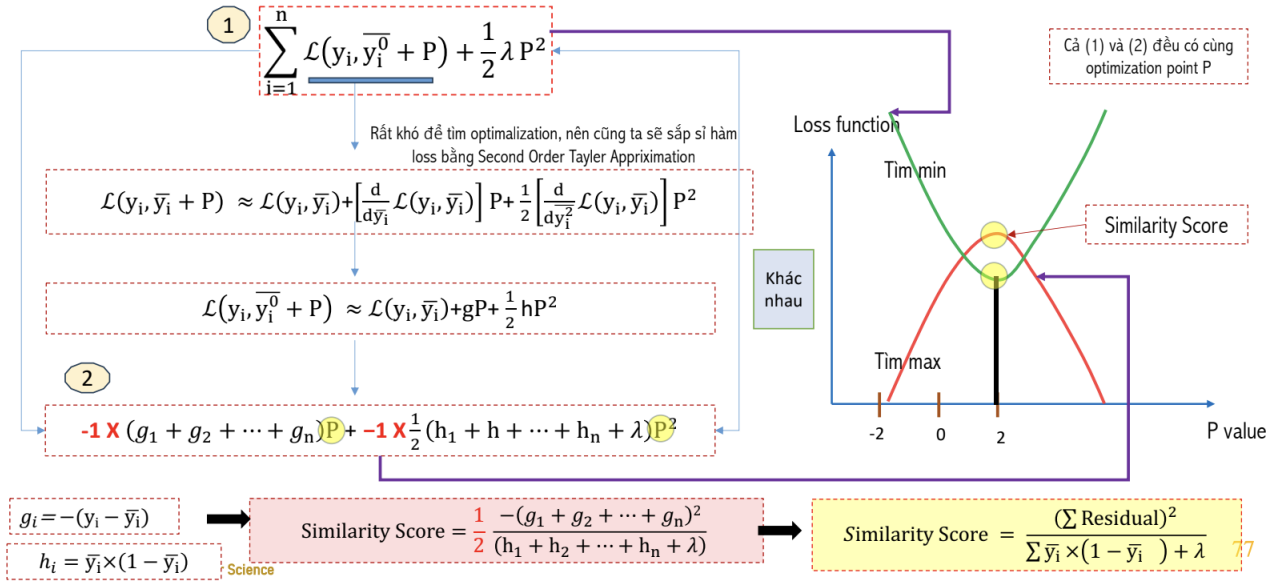
Hình 15: Công thức tính P cho bài toán Regression của XGBoost



Hình 16: Công thức tính P cho bài toán Classification của XGBoost



Hình 17: Phương trình (1) và (2) có cùng optimization point là P. Khi nhân phuương trình (2) với -1 ta được hàm số màu đỏ thể hiện Similarity Score. Do đó giá trị P để Similarity Score đạt cực đại cũng chính là giá trị P để loss function đạt cực tiểu. Khi thay giá trị của P bằng công thức ở 15 ta được Similarity Score như ảnh



Hình 18: Phương trình (1) và (2) có cùng optimization point là P. Khi nhân phưuong trình (2) với -1 ta được hàm số màu đỏ thể hiện Similarity Score. Do đó giá trị P để Similarity Score đạt cực đại cũng chính là giá trị P để loss function đạt cực tiểu. Khi thay giá trị của P bằng công thức ở 16 ta được Similarity Score như ảnh

TV	Radio	Newspaper	Sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	12
151.5	41.3	58.5	16.5
180.8	10.8	58.4	17.9
8.7	48.9	75	7.2
57.5	32.8	23.5	11.8
120.2	19.6	11.6	13.2
8.6	2.1	1	4.8
199.8	2.6	21.2	15.6

Hình 19: Dataset cho ví dụ XGBoost