TÌM HIỂU VẤN ĐỀ OVERFITTING VÀ CÁC PHƯƠNG PHÁP GIẢI QUYẾT VẤN ĐỀ NÀY

I. Overfitting là gì?

Trong machine learning, overfitting là hiện tượng khi một mô hình máy tính học tập quá mức từ tập dữ liệu đào tạo. Điều này dẫn đến việc mô hình hiệu quả quá mức trên tập dữ liệu đào tạo nhưng không thể tổng quát hóa cho các dữ liệu mới hoặc dữ liệu kiểm tra. Cụ thể, overfitting có thể xảy ra khi mô hình trở nên quá phức tạp hoặc quá nhạy cảm đến sự biến thiên trong dữ liệu đào tạo.

II. Các dấu hiệu nhận biết của overfitting.

Dưới đây là một số dấu hiệu nhận biết overfitting:

- Độ chính xác trên tập dữ liệu huấn luyện cao, nhưng độ chính xác trên tập dữ liệu kiểm tra thấp. Đây là dấu hiệu phổ biến nhất của overfitting. Mô hình học quá nhiều chi tiết cụ thể của tập dữ liệu huấn luyện và không thể tổng quát hóa cho các dữ liệu mới.
- Độ phức tạp của mô hình cao. Mô hình càng phức tạp thì càng dễ bị overfitting.
- Dữ liệu huấn luyện nhỏ hoặc nhiễu. Dữ liệu huấn luyện nhỏ hoặc nhiễu sẽ khiến mô hình khó học các mối quan hệ thực tế giữa dữ liệu và có thể bắt đầu học các mối quan hệ ngẫu nhiên hoặc nhiễu trong dữ liệu.

- Mô hình không thể giải thích được. Mô hình càng khó giải thích thì càng dễ bị overfitting. Mô hình không thể giải thích được có thể khó điều chỉnh để tránh overfitting.

Dưới đây là một số ví dụ cụ thể về các dấu hiệu nhận biết overfitting:

- Nếu mô hình của bạn có độ chính xác trên tập dữ liệu huấn luyện là 99%, nhưng độ chính xác trên tập dữ liệu kiểm tra chỉ là 70%, thì có thể mô hình của bạn đang bị overfitting.
- Nếu mô hình của bạn có một mạng thần kinh có hàng triệu tham số, thì có thể mô hình của bạn đang bị overfitting.
- Nếu tập dữ liệu huấn luyện của bạn chỉ có 100 điểm dữ liệu, thì có thể mô hình của bạn đang bị overfitting.
- Nếu mô hình của bạn có thể dự đoán chính xác các điểm dữ liệu trong tập dữ liệu huấn luyện, nhưng không thể dự đoán chính xác các điểm dữ liệu mới, thì có thể mô hình của bạn đang bị overfitting.

III. Các phương pháp giải quyết overfitting.

a. Sử dụng nhiều dữ liệu huấn luyện hơn.

- Đây là cách đơn giản và hiệu quả nhất để giảm thiểu overfitting. Khi có nhiều dữ liệu hơn, mô hình sẽ có nhiều dữ liệu hơn để học và sẽ ít có khả năng học các mối quan hệ ngẫu nhiên hoặc nhiễu trong dữ liệu.
- Một số phương pháp tăng dữ liệu :
    - **Thu thập thêm dữ liệu** : chúng ta phải craw thêm dữ liệu hay tới thực tiễn để thu thập, quay video, chụp ảnh,...Tuy nhiên trong nhiều trường hợp thì

việc thu thập thêm dữ liệu là infeasible nên phương pháp này không được khuyến khích.

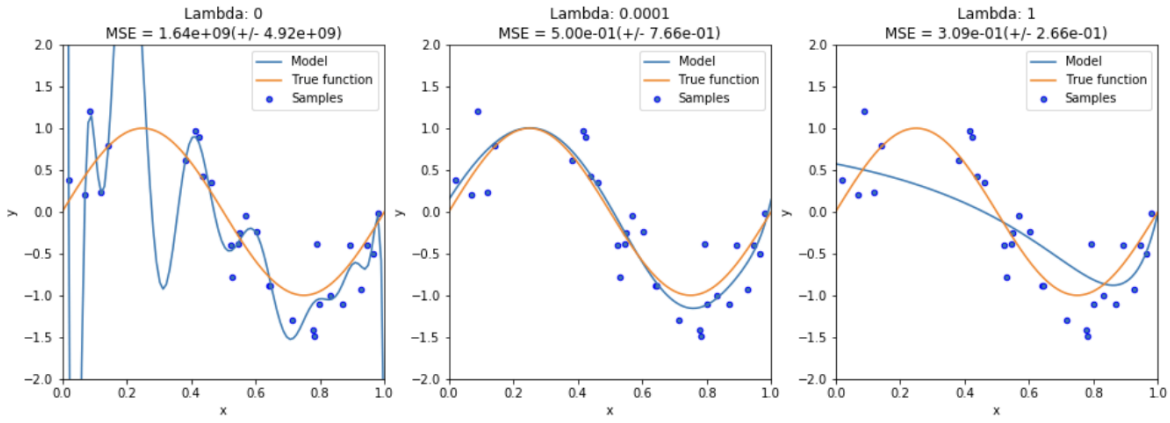
- **Data Augmentation** : Augmentation là 1 phương thức tăng thêm dữ liệu từ dữ liệu có sẵn bằng cách rotation, flip, scale, skew,... images. Phương pháp này được sử dụng rất phổ biến trong xử lý ảnh cho Deep learning.
- **GAN**: GAN (Generative Adverserial Network) là mô hình học không giám sát dùng để sinh dữ liệu từ nhiễu (noise). Nó là sự kết hợp của 2 model: Generative dùng để sinh ảnh từ nhiễu và Discriminator dùng để check ảnh được sinh ra có giống ảnh real hay không? GAN là mô hình hiên nay đang được sử dụng rất phổ biến và tính ứng dụng rất cao. Hiện có rất nhiều mô hình GAN như: CGAN, StyleGAN, StarGAN, CycleGAN...

b. Sử dụng regularization.

- Regularization là một kỹ thuật làm giảm độ phức tạp của mô hình và làm cho nó ít có khả năng overfitting. Có nhiều loại regularization khác nhau, chẳng hạn như L1 regularization, L2 regularization.

$$J(\theta) = E_X(\theta) + \lambda E_\theta(\theta)$$

- $E_X(0)$ là hàm lỗi ban đầu và cụm $\lambda E_0(0)$ mới thêm vào là số hạng chính quy hoá đóng vai trò như một biện pháp phạt lỗi (*penalization*).



- Trong hình trên, biểu thức được biểu diễn là $f(x) = \sin(2\pi X)$. Với $\lambda = 0$ thì được biểu diễn bằng hình bên trái. Với $\lambda = 0.0001$ thì tương ứng với hình chính giữa. Và trường hợp hình bên phải còn lại đại diện cho $\lambda = 1$.

- Trong đó, hệ số chính quy hoá λ được chọn từ trước để cân bằng giữa $E_X(0)$ và $E_0(0)$. λ càng lớn thì ta càng coi trọng $E_0(0)$, ít coi trọng tham số cho hàm lỗi ban đầu hơn, dẫn tới việc các tham số θ ít có ảnh hưởng tới mô hình hơn. Hay nói cách khác là mô hình bớt phức tạp đi giúp ta đỡ việc lỗi *quá khớp*.

- $E_0(0)$ ở đây sẽ không bao gồm độ lệch 0_0 và thường có dạng như sau:

$$E_\theta(\theta) = \frac{1}{p}\|\theta\|_p^p = \frac{1}{p}\sum_{i=1}^{n}|\theta_i|^p$$

- Khi đó, hàm lỗi có thể viết lại như sau:

$$J(\theta) = E_X(\theta) + \lambda \frac{1}{p} \sum_{i=1}^{n} |\theta_i|^p$$

p thường được chọn là 2 (*L2 Norm*) và 1 (*L1 Norm* hay còn được gọi là *Lasso* trong thống kê).

- Với L2, hàm lỗi có dạng:

$$J(\theta) = E_X(\theta) + \frac{\lambda}{2} \theta^\intercal \theta$$

- Với L1, hàm lỗi có dạng:

$$J(\theta) = E_X(\theta) + \lambda \sum_{i=1}^{n} |\theta_i|$$

- Phương pháp chính quy hoá này còn có tên là **cắt trọng số** (*weight decay*) vì nó làm cho các trọng số (tham số θ) bị tiêu biến dần về 0 trong khi học. Còn trong thống kê, phương pháp này có tên là **co tham số** (*parameter shrinkage*) vì nó làm co lại các giá trị tham số dần về 0.

- Công thức chuẩn:
    - Với hàm lỗi của hồi quy tuyến tính thì ta thường chia lấy trung bình của toàn mẫu nên số hạng chính quy hoá cũng sẽ được chia tương tự. Ngoài ra ta cũng thường lấy *L2* để thực hiện việc chính quy hoá, nên:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(\theta^\intercal \phi(\mathbf{x}_i) - y_i \right)^2 + \frac{\lambda}{2m} \theta^\intercal \theta$$

    - Khi đó, công thức chuẩn được viết lại như sau:

$$\hat{\theta} = (\lambda \mathbf{I} + \mathbf{\Phi}^\intercal \mathbf{\Phi})^{-1} \mathbf{\Phi}^\intercal \mathbf{y}$$

- Tính đạo hàm:
    - Đạo hàm khi có số hạng chính quy hóa với:

- $L2: \frac{\partial E_X(\theta)}{\partial \theta_i} + \lambda \theta_i$
- $L1: \frac{\partial E_X(\theta)}{\partial \theta_i} + \lambda \text{sgn}(\theta_i)$

    - đạo hàm này **không** tính cho 0_0. Nói cách khác 0_0 không được thêm số hạng chính quy hoá.

- Trường hợp của bài toán hồi quy tuyến tính:

$$\frac{\partial}{\partial \theta_i} = \frac{1}{m}\sum_{j=1}^{m}(\theta^{\mathsf{T}}\phi(\mathbf{x}_j) - y_j)\mathbf{x_j} + \begin{cases} 0 & \text{for } i = 0 \\ \frac{\lambda}{m}\theta_i & \text{for } i > 0 \end{cases}$$

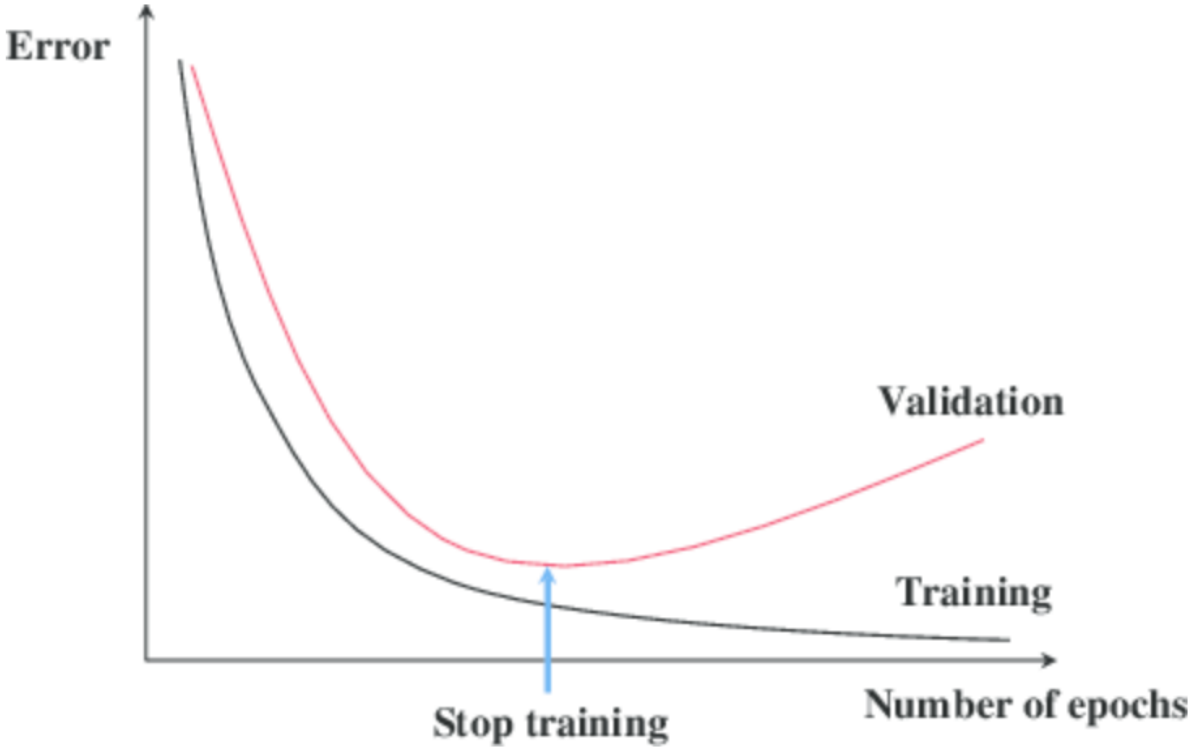
- Gradient có dạng sau:

$$\Delta_\theta J(\theta) = \frac{1}{m}(\theta^{\mathsf{T}}\mathbf{\Phi} - y)\mathbf{\Phi} + \frac{\lambda}{m}\theta$$

- Đương nhiên là khi tính số hạng chính quy hóa ta gắn $0_0 \triangleq 0$ để tiêu biểu số hạng đó đi.

c. Sử dụng early stopping.

- Early stopping là một kỹ thuật dừng quá trình huấn luyện mô hình trước khi nó có cơ hội overfitting. Điều này được thực hiện bằng cách theo dõi hiệu suất của mô hình trên một tập dữ liệu kiểm tra và dừng quá trình huấn luyện khi hiệu suất trên tập dữ liệu kiểm tra bắt đầu giảm.



- Khi training model thì không phải lúc nào (hàm mất mát) loss của tập train và tập test cũng đồng thời giảm, tới một epoch nào đó thì loss của tập train sẽ tiếp tục giảm nhưng loss của tập test không giảm mà tăng trở lại => Đó là hiện tượng overfitting. Vì vậy để ngăn chặn nó, thì ngay tại thời điểm đó người ta sẽ dừng việc training (vì để chương trình tiếp tục training thì cũng không cải thiện được gì mà lại tốn tài nguyên).

d. Sử dụng cross-validation.

- Cross-validation là một kỹ thuật đánh giá hiệu suất của mô hình trên nhiều tập dữ liệu khác nhau. Điều này có thể giúp bạn xác định xem mô hình của mình có bị overfitting hay không.