Tóm tắt sơ lược lý thuyết trong Machine Learning 2022

Contents

----------xxx----------

10 Câu hỏi phỏng vấn thường gặp trong ngành Machine Learning (Học Máy)

https://itguru.vn/blog/10-cau-hoi-phong-van-thuong-gap-trong-nganh-machine-learning-hoc-may-phan-1/

Cơ hội việc làm dành cho các **Machine Learning** đang ngày càng mở rộng với số lượng tăng cao tại các doanh nghiệp. Vì vai trò này rất quan trọng đối với các startups tương lai, việc tuyển chọn các ứng viên Machine Learning phù hợp đòi hỏi tính chọn lọc cao và rất khắt khe.

Nếu bạn là 1 lập trình viên Machine Learning tài năng và đã từng ứng tuyển vào các công việc ở mảng này, bạn có thể đọc tiếp. Chúng ta liệt kê danh sách 10 câu hỏi mà bạn có thể nhận được khi đi phỏng vấn cho vị trí lập trình viên Machine Learning (học máy).

1. Tại sao chúng ta cần một validation set and test set? sự khác biệt giữa chúng là gì?

Khi training một model, chúng ta chia dữ liệu có sẵn thành 3 bộ riêng biệt:

- ***Training dataset*** được sử dụng để phù hợp với các thông số của model. Tuy nhiên, độ chính xác mà chúng ta đạt được trên Training Set là không đáng tin cậy để dự đoán nếu model cũng sẽ chính xác trên sample mới.
- Validation dataset được sử dụng để đo lường mức độ hiệu quả của models trên các ví dụ không phải là một phần của Training dataset. Các số liệu được tính toán trên Validation dataset có thể được sử dụng để điều chỉnh các hyperparameters của model. Tuy nhiên, mỗi khi chúng ta đánh giá Validation dataset và chúng ta đưa ra quyết định dựa trên những điểm số đó, chúng ta leaking thông tin từ Validation dataset vào model. Các đánh giá nhiều lần, càng có nhiều thông tin bị leaking. Vì vậy, chúng ta có thể kết thúc việc ghi đè lên Validation dataset và một lần nữa, validation score sẽ không đáng tin cậy để dự đoán hành vi của model trong thế giới thực.
- Test dataset được sử dụng để đo lường mức độ hiệu quả của model trên các ví dụ không nhìn thấy trước đó. Nó chỉ nên được sử dụng khi chúng ta đã điều chỉnh các thông số bằng cách sử dụng validation set.

Vì vậy, nếu chúng ta bỏ qua test set và chỉ sử dụng validation set, validation score sẽ không phải là ước tính tốt cho việc khái quát hóa model.

2. Stratified cross-validation là gì và khi nào chúng ta nên sử dụng nó?

Cross-validation là một kỹ thuật để chia dữ liệu giữa các training set và validation sets. Trên mỗi Cross-validation điển hình, việc chia tách này được thực hiện ngẫu nhiên. Tuy nhiên, trong Stratified cross-validation, sự phân chia tỷ lệ của các categories trên cả training và validation datasets.

Ví dụ, nếu chúng ta có một tập dữ liệu với 10% của loại A và 90% của loại B, và chúng ta sử dụng stratified cross-validation, chúng ta sẽ có tỷ lệ tương tự training and validation. Ngược lại, nếu chúng ta sử dụng cross-validation đơn giản, trong trường hợp xấu nhất, chúng ta có thể thấy rằng không có mẫu nào của loại A trong validation set.

Có thể áp dụng stratified cross-validation trong các trường hợp sau:

- Trên dataset có nhiều categories.
  Các Dataset càng nhỏ và càng mất cân đối giữa các Categories thì nên sử dụng stratified cross-validation.
- Trên dataset với data phân chia khác nhau.

Ví dụ: trong dataset để điều khiển tự động, chúng ta có thể chụp ảnh vào ban ngày và ban đêm. Nếu chúng ta không đảm bảo rằng cả hai loại đều xuất hiện trong training and validation, thì sẽ nảy sinh một số vấn đề chung khác.

3. Tại sao ensembles thường có điểm số cao hơn các model riêng lẻ?

Ensembles là sự kết hợp của nhiều models để tạo ra một dự đoán duy nhất. Ý tưởng chính để đưa ra dự đoán tốt hơn là các models nên tạo ra các lỗi khác nhau. Bằng cách đó, các lỗi của một model sẽ được bù đắp bằng các dự đoán đúng của các models khác và do đó số điểm của ensembles sẽ cao hơn.

Chúng ta cần các models đa dạng để tạo ra một ensembles. Sự đa dạng có thể đạt được bằng cách:

- Sử dụng các thuật toán ML khác nhau.
  Ví dụ, bạn có thể kết hợp logistic regression, k-nearest neighbors, and decision trees.
- Sử dụng các tập con khác nhau data for training, hay còn được gọi là bagging.
- Đưa ra một trọng lượng khác nhau cho mỗi sample training set. Nếu điều này được thực hiện lặp đi lặp lại, hãy kiểm tra trọng lượng của các samples thông qua lỗi của ensembles, hay còn gọi là boosting.

4. Regularization là gì? Bạn có thể đưa ra một số ví dụ về kỹ thuật regularization không?

Regularization, một cách cơ bản, là thay đổi mô hình một chút để tránh overfitting trong khi vẫn giữ được tính tổng quát của nó (tính tổng quát là tính mô tả được nhiều dữ liệu, trong cả tập training và test). Một cách cụ thể hơn, ta sẽ tìm cách di chuyển nghiệm của bài toán tối ưu hàm mất mát (cost function) tới một điểm gần nó. Hướng di chuyển sẽ là hướng làm cho mô hình ít phức tạp hơn mặc dù giá trị của hàm mất mát có tăng lên một chút.

Một số kỹ thuật Regularization:

- L1 cố gắng giảm thiểu absolute value của các parameters trong models. Nó tạo ra sparse parameters.
- L2 cố gắng giảm thiểu square value của các parameters trong models. Nó tạo ra các parameters với small values.
- Dropout là một kỹ thuật được áp dụng cho các neural networks được đặt một cách ngẫu nhiên và cho ra kết quả đầu ra của các neurons bằng 0 trong quá trình training.
- Early stopping sẽ dừng training khi validation score ngừng cải thiện, ngay cả khi training score có thể được cải thiện. Điều này giúp ngăn chặn overfitting trên training dataset.

5. Dimensionality reduction là gì ? Có cách nào giảm tải tính toán nhưng vẫn giữ được độ chính xác?

Một kỹ thuật khác cùng hướng tiếp cận unsupervised learning đó là giảm số chiều (dimensionality reduction). Dimensionality reduction là một cách để đơn giản hóa dữ liệu, giúp dữ liệu dễ trao đổi, tính toán nhanh hơn, và dễ lưu trữ hơn.

Về mặt ý tưởng, dimensionality reduction nhằm mô tả dữ liệu ngắn gọn hơn.

Ví dụ như điểm GPA. Để đánh giá một sinh viên trong quá trình học, ta cần biết hàng chục lớp học sinh viên đó đã tham gia, hàng trăm bài kiểm tra và hàng ngàn bài tập mà sinh viên đó đã làm. Mỗi bài kiểm tra sẽ cho biết sinh viên này hiểu được nội dung bài giảng đến đâu. Nhưng đối với nhà tuyển dụng việc đọc hết các điểm số này là quá sức. May mắn thay, ta có thể tổng hợp điểm số lại bằng cách lấy trung bình. Ta không cần quan tâm đến hàng đống điểm số vừa rồi mà chỉ cần quan sát điểm GPA để đánh giá lực học của sinh viên đó.

6. Imbalanced dataset là gì? Liệt kê một số cách để xử lý nó?

Imbalanced dataset là tập dữ liệu có tỷ lệ các categories khác nhau.

Ví dụ, một tập dữ liệu với các hình ảnh y tế mà chúng ta phát hiện trong một số bệnh thường sẽ có nhiều mẫu âm tính hơn mẫu dương tính, ví dụ: 98% hình ảnh không có bệnh và 2% hình ảnh bị bệnh.

Có các tùy chọn khác nhau để xử lý các Imbalanced dataset:

- Oversampling hoặc undersampling.
- Data augmentation.

  Chúng ta có thể thêm data vào các categories ít thường xuyên hơn bằng cách sửa đổi data hiện có theo cách được kiểm soát. Trong dataset mẫu, chúng ta có thể lật hình ảnh bị bệnh hoặc thêm nhiễu vào bản sao của hình ảnh theo cách mà bệnh vẫn có thể nhìn thấy được.
- Sử dụng các metrics thích hợp.

  Trong dataset mẫu, nếu chúng ta có một model luôn đưa ra các dự đoán tiêu cực, nó sẽ đạt được độ chính xác 98%. Ngoài ra còn có các metrics khác như độ chính xác, số lần truy cập và F-score để mô tả độ chính xác của một model tốt hơn khi sử dụng Imbalanced dataset.

7. Giải thích sự khác biệt giữa supervised, unsupervised và reinforcement learning?

- **Supervised learning** là thuật toán dự đoán đầu ra (outcome) của một dữ liệu mới (new input) dựa trên các cặp (input, outcome) đã biết từ trước. Cặp dữ liệu này còn được gọi là (data, label), tức (dữ liệu, nhãn). Trong Supervised learning, chúng ta train một model để tìm hiểu về mối quan hệ giữa input và output data. Chúng ta cần có label data để có thể thực hiện được quá trình supervised learning.
- **Unsupervised learning**, chúng ta chỉ có unlabeled data. Thuật toán unsupervised learning sẽ dựa vào cấu trúc của dữ liệu để thực hiện một công việc nào đó, ví dụ như phân nhóm (clustering) hoặc giảm số chiều của dữ liệu (dimension reduction) để thuận tiện trong việc lưu trữ và tính toán. Unsupervised learning thường được sử dụng để khởi tạo các parameters của model khi chúng ta có rất nhiều unlabeled data và một phần nhỏ labeled data. Trước tiên, chúng ta train một Unsupervised Models và sau đó chúng ta sử dụng trọng số của models để train một supervised model.
- **Reinforcement learning**, model có một số input data và phần thưởng tùy thuộc vào output của model. Reinforcement learning là các bài toán giúp cho một hệ thống tự động xác định hành vi dựa trên hoàn cảnh để đạt được lợi ích cao nhất (maximizing the performance). Hiện tại, Reinforcement learning chủ yếu được áp dụng vào Lý Thuyết Trò Chơi (Game Theory), các thuật toán cần xác định nước đi tiếp theo để đạt được điểm số cao nhất.

8. Một số yếu tố giải thích về sự thành công và sự gia tăng gần đây của Deep Learning là gì?

Sự thành công của Deep Learning trong thập kỷ qua có thể được giải thích bằng ba yếu tố chính:

- Thêm Data

  Sự sẵn có khối lượng lớn những tập dữ liệu đã dán nhãn (labeled datasets) cho phép chúng ta train các models với nhiều parameters hơn và đạt được điểm số vượt trội. Các thuật toán ML khác không scale cũng như Deep Learning khi nói đến kích thước dataset.
- GPU

  Các Training models trên GPU có thể giảm thời gian training theo của magnitude so với việc training trên CPU. Hiện tại, các models tiên tiến được đào tạo trên nhiều GPU hoặc thậm chí trên phần cứng chuyên dụng.
- Cải tiến trong thuật toán

  ReLU activation, dropout, và complex network architectures cũng là những yếu tố rất quan trọng.

9. Data augmentation là gì? Cho một số ví dụ?

Data augmentation là một kỹ thuật tổng hợp data mới bằng cách sửa đổi data hiện có theo cách mà target không thay đổi, hoặc nó được thay đổi theo cách đã biết.

Computer vision là một trong những fields mà việc Data augmentation rất hữu ích. Có nhiều sửa đổi mà chúng ta có thể thực hiện đối với hình ảnh:

- Thay đổi kích thước
- Lật ngang hoặc dọc
- Quay
- Thêm tiếng ồn
- Biến dạng
- Sửa đổi màu sắc
- Mỗi vấn đề cần có một cách optimize Data augmentation tùy chỉnh.
  Ví dụ: trong OCR, việc thực hiện flips sẽ thay đổi văn bản và sẽ không mang lại lợi ích; tuy nhiên, thay đổi kích thước và xoay nhỏ có thể hữu ích.

10. Convolutional Neural Network là gì? Có thể sử dụng chúng ở đâu?

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao. Như hiện nay, hệ thống xử lý ảnh lớn như Facebook, Google hay Amazon đã đưa vào sản phẩm của mình những chức năng thông minh như nhận diện khuôn mặt người dùng, phát triển xe hơi tự lái hay drone giao hàng tự động. CNN được sử dụng nhiều trong các bài toán nhận dạng các object trong ảnh.

Các convolutional layer có các parameter(kernel) đã được học để tự điều chỉnh lấy ra những thông tin chính xác nhất mà không cần chọn các feature.

Trong hình ảnh ví dụ trên, ma trận bên trái là một hình ảnh trắng đen được số hóa. Ma trận có kích thước 5×5 và mỗi điểm ảnh có giá trị 1 hoặc 0 là giao điểm của dòng và cột. Convolution hay tích chập là nhân từng phần tử trong ma trận 3.

Sliding Window hay còn gọi là kernel, filter hoặc feature detect là một ma trận có kích thước nhỏ (3×3). Convolution hay tích chập là nhân từng phần tử bên trong ma trận 3×3 với ma trận bên trái. Kết quả được một ma trận gọi là Convoled feature được sinh ra từ việc nhận ma trận Filter với ma trận ảnh 5×5 bên trái.

Các câu hỏi phỏng vấn học máy của lĩnh vực AI

https://vn.bitdegree.org/huong-dan/hoc-machine-learning/

Các khía cạnh chính của Machine learning

Cách tốt nhất để biết về machine learning là bắt đầu từ các câu hỏi liên quan đến machine learning cơ bản. Bằng việc bắt đầu từ những câu hỏi cơ bản, nhà tuyển dụng muốn xem liệu bạn có tư duy phê phán và hình thành suy nghĩ gắn kết của riêng mình hay không. Đó là lý do tại sao rất nhiều câu hỏi này sẽ được dựa trên các định nghĩa, so sánh, giải thích như vậy.

Câu hỏi 1: Giải thích về 'machine learning'.

Machine learning là gì hay hãy mô tả về machine learning có lẽ là câu hỏi đầu tiên liên quan. bởi:

Trước tiên, người phỏng vấn không thể hỏi bạn các câu hỏi khác về học máy mà chưa biết liệu bạn có hiểu machine learning là gì hay không. Hơn nữa, cách bạn trả lời sẽ cho thấy cách bạn nghĩ về một định nghĩa tốt đến mức nào - hay nói cách khác, bạn có thể giải thích một chủ đề khó một cách dễ hiểu hay không. Nếu bạn chỉ nói ra hơn hai lăm dòng mà bạn dành cả đêm để ghi nhớ từ một bài viết khoa học ngẫu nhiên nào đó, bạn sẽ ghi điểm ít hơn so với việc bạn giải thích thuật ngữ theo cách riêng của mình.

Vậy *machine learning là gì?*

Có lẽ cách dễ dàng và dễ hiểu nhất để mô tả học máy là gọi nó là *một triết lý cụ thể về phát triển trí tuệ nhân tạo AI*. Đây là một lĩnh vực khoa học liên quan đến cách chế tạo máy móc mà chúng sẽ học hỏi từ những thông tin được cung cấp cho chúng, mà không được lập trình để làm điều đó trước đó.

Câu hỏi 2: 'deep learning' là gì?

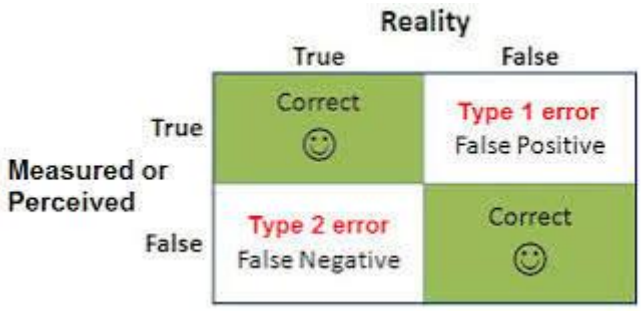
Vì deep learning liên kết chặt chẽ với machine learning, bạn thậm chí có thể được hỏi về các câu hỏi phỏng vấn chéo hay sâu hơn về hai lĩnh vực này. Cho nên nếu bạn đã học machine learning thì đây không phải là câu hỏi quá khó.

Deep learning là *một nhánh của machine learning*. Nhánh khoa học này liên quan đến việc làm cho mạng nơ ron của cỗ máy càng giống với bộ não của con người càng tốt.

Câu hỏi 3: Sự khác nhau giữa lỗi 'type 1' và 'type 2'?

Lỗi loại 1 (type 1) cho rằng một cái gì đó đã xảy ra khi, trong thực tế, nó không thể xảy ra.

Lỗi loại 2 (type) ngược lại - tuyên bố rằng không có gì xảy ra khi nó xảy ra.



Ví dụ: đây là một phương pháp tốt để giúp bạn nhớ sự khác biệt giữa hai loại lỗi: chỉ cần tưởng tượng rằng lỗi loại 1 là khi bạn nói với con chó của mình rằng nó là một con mèo,
trong khi lỗi loại 2 là khi bạn nói với con chó nó là chó không thể sủa.

Câu hỏi 4: 'data augmentation' là gì?

Một trong những câu hỏi phỏng vấn học máy đơn giản hơn, **data augmentation** (tăng dữ liệu) là *cách sửa đổi và tạo dữ liệu mới từ dữ liệu cũ*. Cách thức này được thực hiện bằng cách rời khỏi mục tiêu hoặc đơn giản là thay đổi mục tiêu thành một cái gì đó đã được biết đến. Nếu bạn học machine learning, có lẽ bạn đã biết câu trả lời cho câu hỏi này.

Câu hỏi 5: Tại sao 'naive Bayes' được gọi như vậy?

Naive Bayes được gọi là *naive* theo cách nó nghĩ như vậy. Điều này giả định rằng mọi yếu tố trong một tập dữ liệu đều giống nhau khi nói về tầm quan trọng của chúng. Không cần phải nói, đó là trường hợp hiếm khi xảy ra trong một kịch bản hàng ngày.

Câu hỏi 6: Cái nào tốt hơn – deep networks hay shallow networks?

Đây có thể được coi là một trong những câu hỏi phỏng vấn so sánh khi bạn học machine learning để đánh giá bạn biết gì về hai mạng này và có thể so sánh chúng hay không.

Deep networks thường được coi là một sự thay thế tốt hơn. Điều này đơn giản là vì chúng bao gồm nhiều lớp hơn, phần lớn bị ẩn - điều này giúp các mạng deep networks trích xuất và xây dựng các tính năng tốt hơn.

Câu hỏi 7: 'Fourier transform' là gì?

Phương thức **Fourier transform** được sử dụng để chuyển đổi các hàm đơn giản, chung chung thành các *siêu hàm*. Nếu đây là một trong những câu hỏi phỏng vấn học máy mà bạn muốn mở rộng thêm một chút, bạn có thể so sánh nó với một tình huống mà bạn được cho một chiếc xe để tháo nó ra và xem tất cả các thành phần và bộ phận khác nhau mà nó được tạo ra từ đó.

Câu hỏi 8: 'convolutional network' là gì?

Các mạng thông thường, đơn giản sử dụng các lớp được kết nối để thực hiện các quy trình. Đổi lại, **convolutional network** (mạng chập) là các mạng, thay vì sử dụng các lớp được kết nối, sử dụng các mạng *chập*.

Lý do chính tại sao mọi người thích sử dụng các mạng chập hơn các mạng kết nối tiêu chuẩn, là các mạng chập có một lượng tham số nhỏ hơn nhiều được quy cho chúng.

Câu hỏi 9: Chúng ta biết gì về mối tương quan giữa 'True Positive Rate' và 'Recall'?

Trong việc học machine learning, ban đầu câu hỏi này có vẻ mang tính nâng cao, nhưng câu trả lời khá đơn giản. Cả hai số liệu này đều giống nhau.

Chúng ta có thể thấy điều này bằng cách nhìn vào công thức: **TP/(TP + FN)**

In machine learning, the true positive rate, also referred to sensitivity or recall, is **used to measure the percentage of actual positives**

Câu hỏi 10: 'backpropagation' là gì?

Trong học machine learning, **backpropagation** chỉ đơn giản là một *phương pháp đào tạo các mạng lưới thần kinh nhiều lớp*. Chúng ta huấn luyện mạng với phương pháp này bằng cách lấy 'lỗi' (**error**) từ điểm cuối của mạng và đặt nó vào bên trong mỗi trọng lượng trong mạng. Bằng cách này, máy có cơ hội áp dụng tính toán của nó một cách hiệu quả.

Câu hỏi 11: Điều gì sẽ xảy ra nếu chúng ta chỉ sử dụng một 'validation set', mà không áp dụng 'test set'?

Trong học machine learning và các câu hỏi phỏng vấn, câu hỏi này khó hơn một chút.

Nếu bạn chỉ áp dụng một bộ xác thực (**validation set**), nó sẽ không cung cấp ước tính chính xác cho tất cả các phép đo cho mô hình bạn đang thử nghiệm. Điều này là do bộ kiểm tra (**test set**) được sử dụng để kiểm tra xem mô hình sẽ hoạt động như thế nào trên các ví dụ mà nó không gặp phải đến thời điểm đó. Do đó, nếu bạn loại bỏ test set, bạn sẽ tự động làm suy yếu các kết quả kiểm tra có thể hợp lệ, có thể nói như vậy.

Câu hỏi 12: Sự khác nhau giữa học máy suy diễn và quy nạp là gì?

Sự khác biệt chính là cách chúng bắt đầu.

- Học máy quy nạp bắt đầu với các ví dụ để từ đó đưa ra kết luận.
- Học máy suy diễn bắt đầu bằng kết luận, sau đó học bằng cách suy luận điều gì sai hoặc điều gì đúng về kết luận đó.

Câu hỏi 13: Làm thế nào variance và bias xảy ra trong học máy?

Cả hai đều là lỗi. Không được nhầm lẫn hai lỗi này vì bạn sẽ cần phải nhớ chúng trong các câu hỏi phỏng vấn học máy cũng như học machine learning.

- Variance (Phương sai) là một lỗi là kết quả của sự phức tạp trong thuật toán học máy.
- Bias là một lỗi là do các giả định thiếu sót trong thuật toán học tập.

Câu hỏi 14: Học có giám sát là gì và nó khác với không giám sát như thế nào? -->

- ***Học máy có giám sát*** là một quá trình trong đó các đầu ra được đưa trở lại vào máy tính để phần mềm học hỏi và nhận được kết quả chính xác hơn vào lần tiếp theo. -→???
- Học máy không giám sát có nghĩa là một máy tính sẽ học mà không cần đào tạo ban đầu, đó là một thay thế cho học máy có giám sát, trong đó 'máy, nhận được đào tạo ban đầu để bắt đầu.

Câu hỏi 15: Làm thế nào để bạn chọn một thuật toán cho một vấn đề phân loại? -->

Trong trường hợp này, câu trả lời phụ thuộc vào mức độ chính xác cần thiết và quy mô của tập huấn luyện. Nếu tập huấn luyện nhỏ, nên chọn phân loại sai lệch thấp / sai lệch cao. Nếu tình huống ngược lại, tập huấn luyện lớn, thì bạn nên chọn phân loại sai lệch cao và sai lệch thấp.

Câu hỏi phỏng vấn học máy nâng cao

Nếu bạn học machine learning, bạn dễ dàng nhận ra chúng ta đã trải qua các câu hỏi machine learning cơ bản. Bây giờ hãy cùng tìm hiểu những câu hỏi phỏng vấn nâng cao.

Nhà tuyển dụng sẽ không yêu cầu bạn xây dựng một hệ thống AI đầy đủ hay viết một cuốn sách dài ba trăm trang về tất cả các cách khác nhau mà bạn có thể thực hiện deep learning. Trong bối cảnh này, 'nâng cao' đơn giản có nghĩa là các câu hỏi sẽ trở nên khó hơn một chút bạn có thể được yêu cầu cung cấp giải thích sâu hơn cho câu trả lời của bạn, đưa ra ví dụ, v.v…

Câu hỏi 1: Sự khác nhau giữa mô hình 'generative' và 'discriminative' là gì?

Với việc học machine learning, câu hỏi này nghe có vẻ là câu hỏi mẹo nhưng thực chất nhà tuyển dụng muốn biết cách các mô hình này xử lý dữ liệu.

- **Generative model** (mô hình thế hệ), như tên gọi của nó, là nỗ lực tìm hiểu các loại dữ liệu khác nhau mà nó được cung cấp.
- **Discriminative model** (mô hình phân biệt) sẽ chỉ nghiên cứu sự khác biệt giữa các loại dữ liệu khác nhau.

Các developer và kỹ sư thường thích sử dụng mô hình discriminative, vì nó có xu hướng xử lý các nhiệm vụ nhanh và hiệu quả hơn.

Câu hỏi 2: Giải thích sự khác nhau giữa 'cross-validation' và 'stratified cross-validation'.

- **Cross-validation** (Xác thực chéo) đơn giản được sử dụng để phân tách ngẫu nhiên dữ liệu giữa thời gian đào tạo và bộ xác thực.
- **Stratified cross-validation** (Xác thực chéo phân tầng) thực hiện điều tương tự, *nhưng không có biến ngẫu nhiên* - nó theo dõi và bảo toàn tỷ lệ đào tạo so với kiểm tra xác nhận.

Đây là một trong những câu hỏi phỏng vấn học máy trong học machine learning mà bạn có thể bị nhầm lẫn nên hãy cẩn thận với câu hỏi này!

Câu hỏi 3: Trong trường hợp nào bạn nên sử dụng hồi quy 'Lasso' và 'Ridge'?

Đây là một câu hỏi phỏng vấn machine learning nâng cao chủ yếu vì bạn cần một số kiến thức chuyên sâu liên quan đến cả hai loại hồi quy để cung cấp câu trả lời chính xác.

- **Hồi quy Lasso** có thể thực hiện cả hai chức năng chọn biến và thu nhỏ tham số,
- **Hồi quy Ridge** chỉ có thể được sử dụng để thu nhỏ hàm số mà thôi.

Do vậy, bạn sẽ sử dụng hồi quy Lasso khi bạn chỉ có một vài biến và có ảnh hưởng lớn. Đổi lại, nên sử dụng hồi quy Ridge khi có nhiều biến nhỏ.
Đây là một ví dụ hay về những câu hỏi phỏng vấn học máy mà bạn có thể mở rộng dựa trên câu trả lời của mình, thay vì chỉ đưa ra một câu hỏi chung chung.

Câu hỏi 4: 'F1' là gì?

Không, đây không phải là một phím trên bàn phím để bạn đưa ra câu trả lời.
Điểm **F1 score** là một phép đo xem mô hình của bạn hoạt động tốt như thế nào. Bất cứ cái gì gần với '**1'** được coi là tốt, dưới mức '**0,5**', nên được xử lý.

Câu hỏi 5: Trong hầu hết các trường hợp, cái nào trong hai cái sau có điểm cao hơn – mô hình quần thể (Ensembles) hay mô hình cá nhân?

Quần thể (**Ensembles**) thường cung cấp số điểm lớn hơn. Bởi vì chúng đơn giản là sự kết hợp của nhiều mô hình khác nhau, được thực hiện để dự đoán một kết quả cụ thể. Càng nhiều mô hình, càng nhiều lỗi họ có thể lọc ra - điểm dự đoán cuối sẽ càng tốt.

Câu hỏi 6: Sự khác nhau giữa 'correlation' và 'covariance'?

Đây sẽ là một trong những câu hỏi phỏng vấn học máy nâng cao chỉ khi bạn không biết hai cái này có tương quan như thế nào (*không có việc chơi chữ ở đây*).
Còn nếu bạn biết, câu trả lời khá đơn giản: **covariance** (hiệp phương sai) *trở thành* một **correlation** (hệ số tương quan) một khi nó được chuẩn hóa.

Câu hỏi 7: Mô tả 'imbalanced dataset'.

Imbalanced dataset (Bộ dữ liệu không cân bằng) là một bộ mà sau khi thử nghiệm sẽ mang lại kết quả rằng hơn một nửa toàn bộ thông tin được đặt trong một lớp.
Làm thế nào để tránh điều này?
Vâng, có một vài giải pháp đơn giản - hoặc thực hiện kiểm tra lại bằng thuật toán khác hoặc thử kiểm tra một lượng dữ liệu thậm chí còn lớn hơn để ra kết quả.

Câu hỏi 8: 'data normalization' là gì?

Bạn có nhớ khi chúng ta nói về **'backpropagation'**, trong các câu hỏi phỏng vấn học máy phía trên không?
Vâng, **data normalization** (Chuẩn hóa dữ liệu) được sử dụng để giảm thiểu sự dư thừa dữ liệu trong quá trình backpropagation. Nó cho phép người dùng hủy bỏ các giá trị khác nhau khi anh ta thấy phù hợp, do đó loại bỏ các vấn đề dư thừa có thể xảy ra.

Câu hỏi 9: Bạn có thể nắm bắt mối tương quan giữa biến phân loại và biến liên tục không?

Bạn có thể nhưng sẽ phải sử dụng phương pháp Phân tích hiệp phương sai (**ANCOVA - Analysis of Covariance**). Sử dụng nó, bạn có thể nắm bắt được mối tương quan.

Câu hỏi 10: Chức năng kích hoạt được sử dụng để làm gì?

Chức năng này cho phép bạn đa dạng hóa mạng lưới của mình bằng cách giới thiệu các phương pháp học phi tuyến tính. Nó sẽ giúp máy của bạn học cách xử lý các quy trình khó một cách dễ dàng hơn.

12 câu hỏi phỏng vấn Deep Learning siêu hay không thể bỏ qua

https://viblo.asia/p/ai-interview-12-cau-hoi-phong-van-deep-learning-sieu-hay-khong-the-bo-qua-LzD5djvEZjY

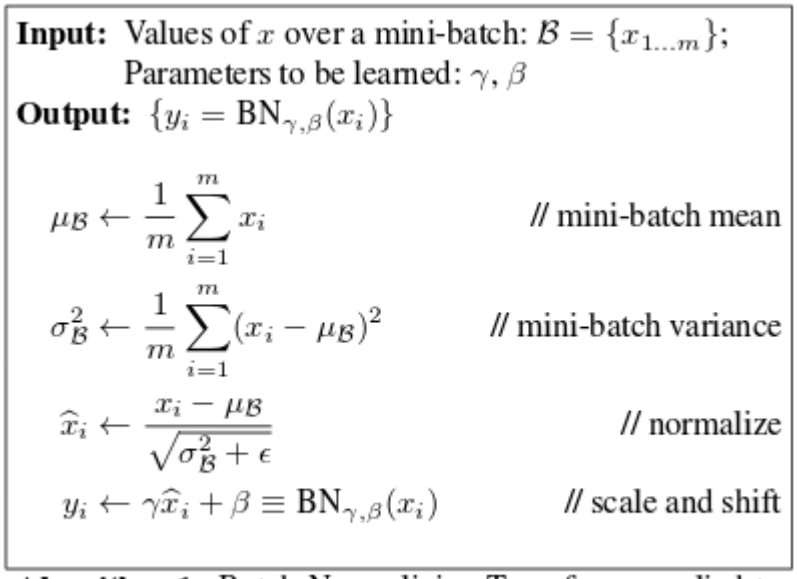
Xin chào các bạn, hôm nay mình sẽ quay lại với các bạn về một chủ đề không mới những chưa bao giờ hết **hot**. Đó chính là **các câu hỏi mà thường được hỏi khi phỏng vấn vị trí AI Engineer là gì?**. Thực ra cũng không phải cuộc phỏng vấn nào cũng cần phải dùng đến hết những câu hỏi ở trong này vì còn tuỳ thuộc vào kinh nghiệm và các dự án mà ứng viên đã từng làm qua nữa. Qua rất nhiều cuộc phỏng vấn, đặc biệt là với các bạn sinh viên mình đúc kết ra tập hợp 12 câu hỏi phỏng vấn tâm đắc nhất trong mảng **Deep Learning** mà ngày hôm nay sẽ chia sẻ lại cho các bạn trong bài viết lần này. Rất mong nhận được nhiều ý kiến đóng góp của các bạn. OK không lan man nữa chúng ta bắt đầu thôi nhé.

1. Trình bày ý nghĩa của Batch Normalization

Đây có thể coi là một câu hỏi rất hay vì nó cover được gần hết các kiến thức mà ứng viên cần phải biết khi làm việc với một mô hình mạng nơ ron. Các bạn có thể trả lời khác nhau nhưng cần làm rõ được các ý chính sau:

Batch Normalization là một phương pháp hiệu quả khi training một mô hình mạng nơ ron.

Mục tiêu của phương pháp này chính là việc muốn chuẩn hóa các feature (đầu ra của mỗi layer sau khi đi qua các activation) về trạng thái **zero-mean** với độ lệch chuẩn 1.
Vậy hiện tượng ngược lại đó là **non-zero mean** có ảnh hưởng như thế nào đến việc training mô hình:



Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

- Thứ nhất có thể hiểu rằng **Non zero mean** là hiện tượng dữ liệu không phân bố quanh giá trị 0, mà dữ liệu có phần nhiều giá trị lớn hơn không, hoặc nhỏ hơn không. Kết hợp với vấn đề high variance khiến dữ liệu trở nên có nhiều thành phần rất lớn hoặc rất nhỏ. Vấn đề này rất phổ biến khi training các mạng nơ ron với số layer sâu. Việc feature không phân phối trong những khoảng ổn định (giá trị to nhỏ thất thường) sẽ có ảnh hưởng đến quá trình tối ưu của mạng. Vì như chúng ta đã biết việc tối ưu một mạng nơ ron sẽ cần phải sử dụng đến tính toán đạo hàm. Giả sử như một công thức tính layer đơn giản là y = (Wx + b) thì đạo hàm của y theo w có dạng: dy = *dWx*. Như vậy giá trị x ảnh hưởng trực tiếp đến giá trị của đạo hàm (tất nhiên khái niệm gradient trong các mô hình mạng nơ ron không thể đơn giản như vậy, tuy nhiên về mặt lý thuyết thì x sẽ có ảnh hưởng đến đạo hàm). Do đó nếu x mang các giá trị thay đổi không ổn định dẫn đến đạo hàm sẽ có thể bị quá lớn, hoặc quá nhỏ dẫn đến việc learning model không được ổn định. Và điều đó cũng đồng nghĩa với việc chúng ta có thể sử dụng các learning rate cao hơn trong quá trình training khi sử dụng Batch Normalization.

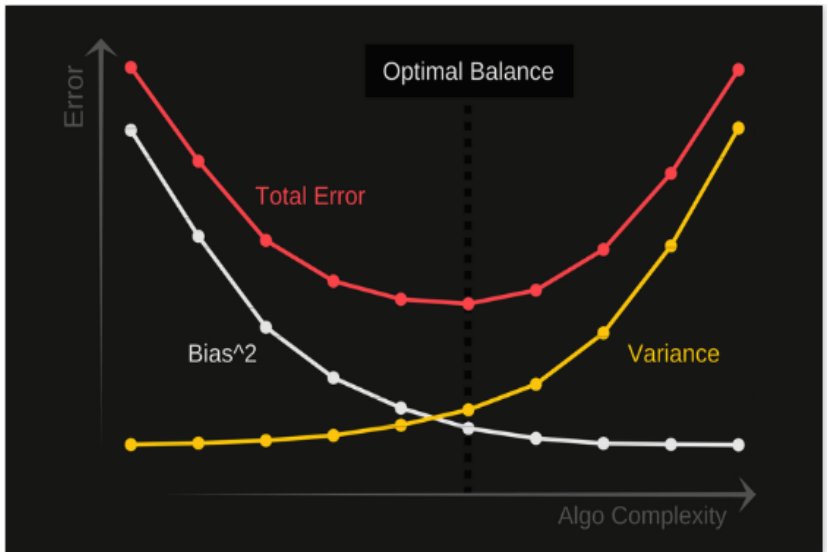
- **Batch normalization (BN)** có thể giúp chúng ta tránh được hiện tượng giá trị của x rơi vào **khoảng bão hòa** sau khi đi qua các hàm kích hoạt phi tuyến. Vậy nên nó đảm bảo rằng không có sự kích hoạt nào bị vượt quá cao hoặc quá thấp. Điều này giúp cho các weights mà khi không dùng **BN** có thể sẽ không bao giờ được học thì nay lại được học bình thường. Điều này giúp chúng ta làm giảm đi sự phụ thuộc vào giá trị khởi tạo của các tham số.
- Batch Normalization còn có vai trò như một dạng của **regularization** giúp cho việc giảm thiểu overfiting. Sử dụng batch normalization, chúng ta sẽ không cần phải sử dụng quá nhiều dropput và điều này rất có ý nghĩa vì chúng ta sẽ không cần phải lo lắng vì bị mất quá nhiều thông tin khi dropout weights của mạng. Tuy nhiên vẫn nên sử dụng kết hợp cả hai kỹ thuật này

2. Trình bày khái niệm và mối quan hệ đánh đổi giữa bias và variance?

- **Bias là gì?**

  Có thể hiểu đơn giản thế này, bias chính là sự khác nhau của giá trị trung bình các dự đoán (average prediction) của mô hình hiện tại với kết quả thực tế mà chúng ta đang cần dự đoán. Mô hình nào có chỉ số bias cao chứng tỏ nó ít tập trung hơn vào dữ liệu huấn luyện. Điều này khiến cho mô hình trở nên quá đơn giản và không đạt được độ chính xác tốt trên cả tập training và tập testing hay hiện tượng này còn được gọi là **underfitting**
- **Variance** Có thể hiểu đơn giản là sự phân tán (hay co cụm) của kết quả đầu ra của mô hình trên một điểm dữ liệu. Nếu như variance càng lớn thì chứng tỏ mô hình đang tập trung chú ý nhiều vào dữ liệu huấn luyện và không mang được tính tổng quát trên dữ liệu chưa gặp bao giờ. Từ đó dẫn đến mô hình đạt được kết quả cực kỳ tốt trên tập dữ liệu huấn luyện, tuy nhiên kết quả rất tệ với tập dữ liệu kiểm thử (testing). Đây chính là hiện tượng **overfitting**

Có thể hình dung sự tương quan qua lại của 2 khái niệm này trong hình sau:



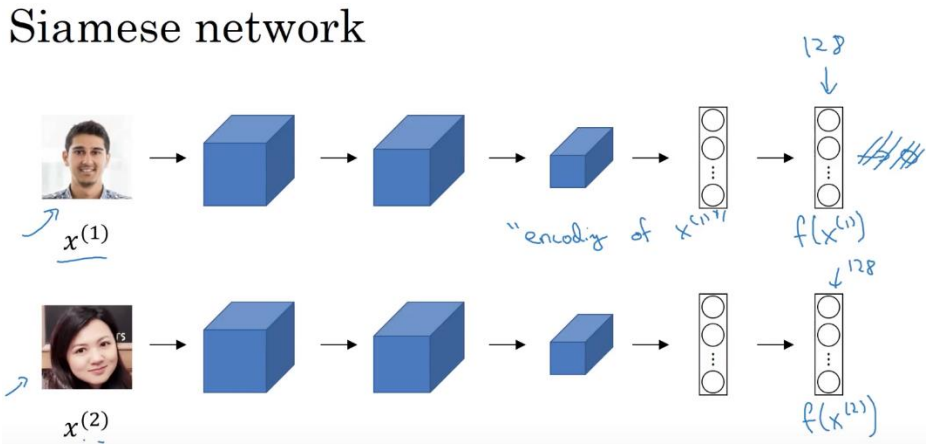
Trong sơ đồ trên, trung tâm của hình tròn là một mô hình dự đoán hoàn hảo các giá trị chính xác. Thực tế là chẳng bao giờ các bạn tìm được một mô hình nào tốt như vậy cả. Khi chúng ta càng xa tâm vòng tròn thì dự đoán của chúng ta ngày càng trở nên tồi tệ hơn. Chúng ta có thể thay đổi mô hình để có thể tăng số lượt mô hình dự đoán rơi vào đúng tâm vòng tròn càng nhiều càng tốt. Việc cân bằng giữa hai giá trị **Bias** và **Variance** rất cần thiết. Nếu mô hình của chúng ta quá đơn giản và có rất ít tham số thì nó có thể có **high bias và low variance.**

Mặt khác, nếu mô hình của chúng ta có số lượng lớn các tham số thì nó sẽ có **high variance và low bias**. Đây chính là cơ sở để chúng ta tính toán đến độ phức tạp của mô hình khi thiết kế thuật toán.

3. Giả sử sau mô hình Deep Learning tìm được 10 triệu vector khuôn mặt. Làm sao tìm query khuôn mặt mới nhanh nhất.

Câu hỏi này thiên về việc ứng dụng các giải thuật Deep Learning trên thực tế, điểm mấu chốt của câu hỏi này là phương pháp indexing dữ liệu. Đây là bước sau cùng của bài toán áp dụng **One Shot Learning** cho nhận

diện khuôn mặt nhưng lại là bước quan trọng nhất giúp cho ứng dụng này có thể triển khai dễ dàng trên thực tế. Về cơ bản thì với câu hỏi này các bạn nên trình bày tổng quan về phương pháp nhận diện khuôn mặt bằng **One Shot Learning** trước. Nó có thể hiểu đơn giản là việc biến mỗi khuôn mặt thành một vector, và việc nhận dạng khuôn mặt mới chính là việc tìm ra các vector có khoảng cách gần nhất (giống nhất) với khuôn mặt đầu vào. Thông thường người ta sẽ sử dụng mô hình deep learning với hàm loss được custom là **triplet loss** để thực hiện điều đó.



Tuy nhiên với số lượng ảnh tăng lên như đầu bài thì việc tính toán khoảng cách đến 10 triệu vector trong mỗi lần nhận diện là một giải pháp không được thông minh. Khiến cho hệ thống bị chậm đi rất nhiều.

Chúng ta cần nghĩ ngay đến các phương pháp indexing dữ liệu trên không gian vector số thực để khiến cho việc truy vấn được thuận lợi hơn.

Tư tưởng chính của các phương pháp này đó chính là phân chia dữ liệu thành các cấu trúc dễ dàng cho việc query dữ liệu mới (có thể tương tự cấu trúc cây). Khi có dữ liệu mới, việc truy vấn trên cây sẽ giúp nhanh chóng tìm ra được vector có khoảng cách gần nhất với thời gian rất nhanh chóng.

Có một số phương pháp có thể sử dụng cho mục đích này như **Locality Sensitive Hashing - LSH**, **Approximate Nearest Neighbors Oh Yeah - Annoy Indexing**, **Faiss** .... có thể được sử dụng cho mục đích này

4. Với bài toán classification, chỉ số accuracy có hoàn toàn tin tưởng được không. Bạn thường sử dụng các độ đo nào để đánh giá mô hình của mình?

Với một bài toán phần lớp thì có nhiều cách đánh giá khác nhau. Riêng đối với accuracy (tức độ chính xác) thì công thức chỉ đơn giản là lấy số điểm dữ liệu dự đoán đúng chia cho tổng số dữ liệu. Điều này nghe qua thì có vẻ hợp lý.

Nhưng trên thực tế đối với những bài toán dữ liệu bị mất cân bằng thì đại lượng này chưa đủ ý nghĩa.

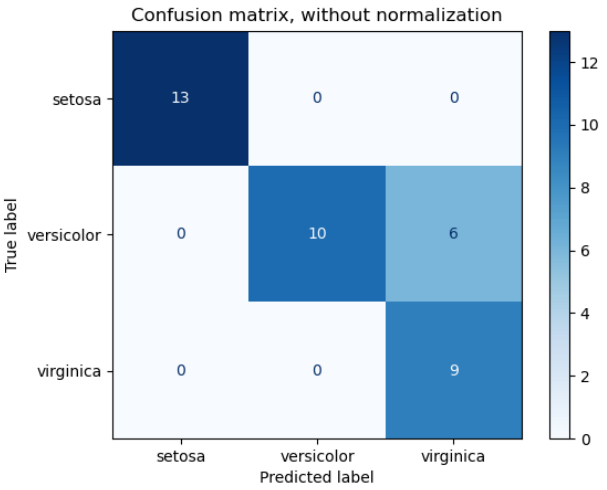
Giả sử chúng ta đang xây dựng mô hình dự đoán tấn công mạng (giả sử các request tấn công chiếm khoảng 1/100000 số lượng request). Nếu mô hình dự đoán tất cả các request đều là bình thường thì độ chính xác cũng lên đến **99.9999%** và con số này thường không thể tin tưởng được trong mô hình phân lớp. Cách tính

accuracy ở trên thường chỉ cho chúng ta biết được có bao nhiêu phần trăm dữ liệu được dự đoán đúng chứ chưa chỉ ra được mỗi class được phân loại cụ thể như thế nào.

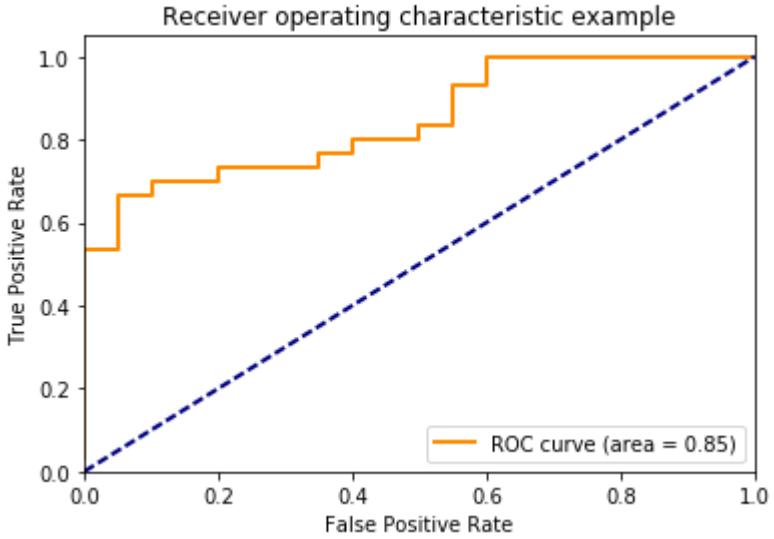
Thay vào đó chúng ta có thể sử dụng **Confusion matrix**.

Về cơ bản, confusion matrix thể hiện có bao nhiêu điểm dữ liệu thực sự thuộc vào một class, và được dự đoán là rơi vào một class.

Nó có dạng mẫu như hình bên phải



Ngoài ra để biểu diễn sự thay đổi của các chỉ số True Possitive và False Positive tương ứng với từng threshold quy định việc phân lớp thì chúng ta có được một đồ thị gọi là **Receiver Operating Characteristic - ROC**. Dựa trên ROC biết được mô hình có hiệu quả hay không.



Một đường ROC lý tưởng là đường **màu cam** càng gần góc trên cùng bên trái (tức True Postitive cao và False Positive thấp) thì càng tốt.

Một đại lượng khác để thể hiện độ tốt của mô hình đó là **Area Under the Curve** chỉ phần diện tích nằm phía dưới của **ROC Curve**. Diện tích này càng cao chứng tỏ mô hình càng có hiệu quả.

5. Bạn hiểu như thế nào về Backpropagation? Giải thích cơ chế hoạt động?

Câu hỏi này nhằm kiểm tra kiến thức về cách mà một mạng nơ ron hoạt động. Bạn cần phải làm rõ lên được các điểm

- Quá trình forward (tính toán tiến) là quá trình giúp mô hình tính toán các trọng số của từng layers và kết quả sau khi tính toán sẽ cho ra một kết quả *yp*. Lúc này sẽ tính toán được giá trị của hàm loss, giá trị của hàm loss sẽ thể hiện được độ tôt của mô hình như thế nào. Nếu thấy hàm loss chưa đủ tốt thì ta cần phải tìm cách làm giảm giá trị hàm loss. Training một mạng nơ ron bản chất là cực tiểu hoá một hàm loss. Hàm loss $L(yp,yt)$ thể hiện mức độ sai khác giữa giá trị đầu ra của mô hình *yp* và giá trí thực của nhãn dữ liệu *yt*.
- Muốn giảm giá trị của hàm loss ta cần sử dụng đạo hàm. Back-propagation chính là giúp chúng ta tính toán được đạo hàm cho từng layers của mạng. Căn cứ vào giá trị của đạo hàm trên từng layers

thì các optimizer (Adam, SGD, AdaDelta...) áp dụng gradient descent sẽ cập nhật lại trọng số của mạng.

- **Backpropagation** sử dụng cơ chế chain-rule hay còn gọi là đạo hàm hàm hợp để tính toán giá trị gradient của từng layers kể từ layers cuối cùng đến layer đầu tiên

6. Ý nghĩa của hàm activation function là gì? Thế nào là điểm bão hòa của các activation functions?

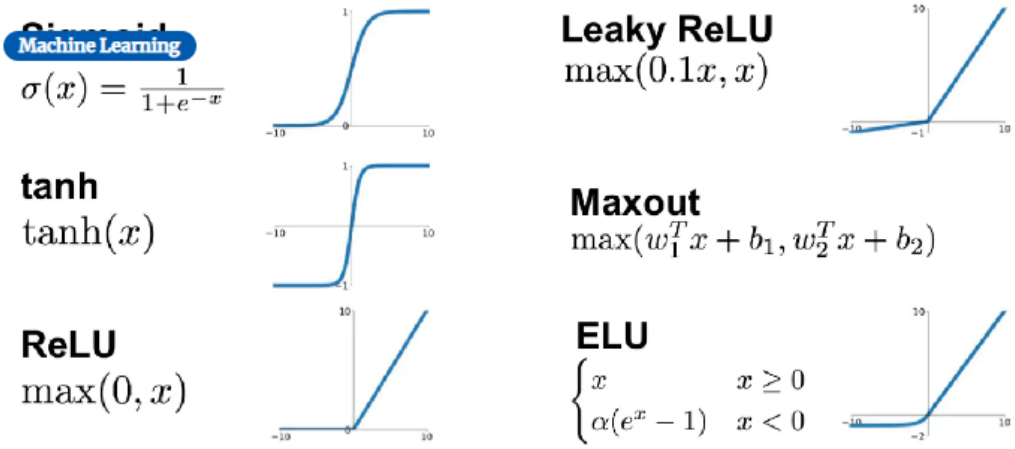
Ý nghĩa của activation function

Hàm kích hoạt hay **activation functions** được sinh ra với mục đích **bẻ gãy sự tuyến tính** của mạng nơ ron. Các hàm này có thể hiểu đơn giản như một bộ lọc để quyết định xem thông tin có được đi qua nơ ron hay không. Trong quá trình huấn luyện mạng nơ ron, các hàm kích hoạt đóng vai trò quan trọng trong việc điều chỉnh độ dốc của đạo hàm.

Một số hàm kích hoạt giống như **sigmoid**, **tanh** hay **ReLU** sẽ được bàn bạc kỹ hơn trong các phần tiếp theo. Tuy nhiên chúng ta cần hiểu rằng tính chất của các hàm phi tuyến này giúp cho mạng nơ ron có thể học được biểu diễn của các hàm phức tạp hơn là chỉ sử dụng các hàm tuyến tính. **Hầu hết** các activation functions là các **continuous** và khả vi **differentiable** functions Các hàm này là các hàm liên tục (continuous), tức là có sự thay đổi nhỏ ở kết quả đầu ra nếu như đầu vào có sự thay đổi nhỏ và khả vi (differentiable) tức là có đạo hàm tại mọi điểm trong miền xác định của nó. Tất nhiên rồi, như đã đề cập ở phía trên thì việc tính toán được đạo hàm là rất quan trọng và nó là một yếu tố quyết định đến nơ ron của chúng ta có thể training được hay không. Có một số hàm activation function có thể kể đến như Sigmoid, Softmax, ReLU.

Khoảng bão hoà của activation function

Các hàm kích hoạt phi tuyến như hàm Tanh, hàm Sigmoid, hàm ReLU đều có những **khoảng bão hoà**



(from aicurious.io)

Có thể hiểu đơn giản rằng các **khoảng bão hoà** của hàm kích hoạt là các khoảng mà giá trị đầu ra của hàm không thay đổi mặc dù giá trị đầu vào thay đổi.

Có hai vấn đề của **khoảng thay đổi** đó là

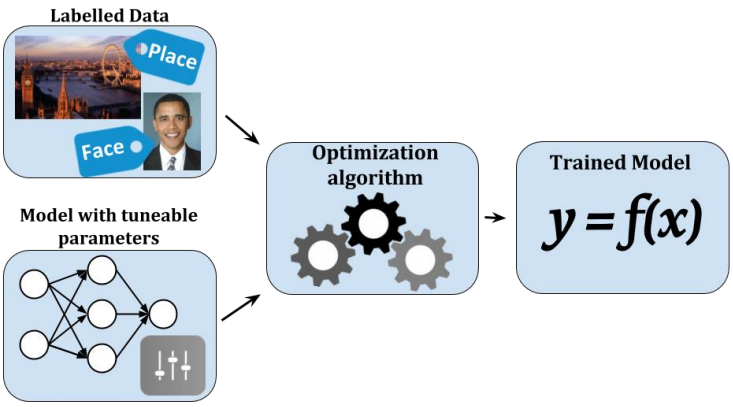
1. Thứ nhất, trong chiều forward của mạng nơ ron thì những giá trị của layer bị rơi vào khoảng bão hoà của hàm kích hoạt sẽ dẫn đến có nhiêu giá trị đầu ra giống nhau. Dẫn đến luồng dữ liệu bị giống nhau trong toàn model. Hiện tượng này là hiện tượng **covariance shifting**.

2. Vấn đề thứ hai là theo chiều backward, đạo hàm sẽ bằng 0 tại vùng bão hoà và do đó mạng gần như sẽ không học được thêm gì. Đó chính là lý do tại sao chúng ta cần đưa khoảng giá trị về **mean zero** giống như đã đề cập trong phần **Batch Normalization**

7. Hyperparameters của mô hình là gì? Khác với parameters như thế nào?

Với câu hỏi này mình xin phép được trích lại một đoạn giải thích rất kỹ về vấn đề này trong một bài viết mà mình đã viết từ trước đó. Một vài hiểu nhầm khi mới học Machine Learning

Model parameter là gì



Quay về bản chất của **Machine Learning** một chút, đầu tiên để làm về **học máy** chúng ta cần phải có một **tập dữ liệu** - muốn nói gì thì nói chứ không có dữ liệu chúng ta sẽ lấy cái gì mà học với chả hành đúng không?
Sau khi có dữ liệu rôi thì việc cần làm của **máy** là tìm ra một mối liên hệ nào đó trong cái đống dữ liệu này. Giả sử dữ liệu của chúng ta là thông tin về thời tiết như độ ẩm lượng mưa nhiệt độ... và yêu cầu cho máy thực hiện là tìm mối liên hệ giữa các yếu tố trên và việc **người yêu có giận ta hay không giận?**.

Nghe thì có vẻ không liên quan lắm nhưng việc cần làm của máy học đôi khi là những thứ khá vớ vẩn như vậy đó. Bây giờ giả sử chúng ta sử dụng biến *y* để biểu diễn việc **người yêu có giận hay không giận?** các biến *x*1, *x*2, *x*3... đại diện cho các yếu tố thời tiết. Chúng ta quy một liên hệ về việc tìm hàm *f*(*x*) như sau:

$$y = f(x) = w1.x1 + w2.x2 + w3.x3$$

Các bạn có thấy các hệ số *w*1,*w*2,*w*3.. không? Đó chính là **mối liên hệ** giữa đống dữ liệu và yếu tố chúng ta đang yêu cầu đó, chính các hệ số này được gọi là **Model Parameter** đó. Như vậy chúng ta có thể định nghĩa **model parameter** như sau:

> **Model Parameter** là các giá trị của model được sinh ra từ dữ liệu huấn luyện giúp thể hiện mối liên hệ giữa các đại lượng trong dữ liệu

Như vậy khi chúng ta nói **tìm được mô hình tốt nhất cho bài toán** thì nên ngầm hiểu rằng chúng ta đã tìm ra được các **Model parameter** (w_i) phù hợp nhất cho bài toán trên tập dữ liệu hiện có. Nó có một số đặc điểm như sau:

- Nó được sử dụng để dự đoán đối với dữ liệu mới
- Nó thể hiện sức mạnh của mô hình chúng ta đang sử dụng. Thường được thể hiện bằng tỷ lệ **accuracy** hay chúng ta gọi là độ chính xác
- Được **học** trực tiếp từ tập dữ liệu huấn luyện
- Thường **không** được đặt thủ công bởi con người

Model paramter có thể bắt gặp trong một số dạng như là

- các ***trọng số*** trong mạng nơ ron,
- các **support vectors** trong SVM,
- các **coefficients** trong các giải thuật linear regression hoặc logistic regression...

Model Hyperparameter là gì?

Có lẽ do thời quen thường dịch **Hyperparameter** là **siêu tham số** nên chúng ta thường ngầm định nó giống **Model Parameter** nhưng có phần **khủng hơn**.

Thực ra hai khái niệm này là hoàn toàn tách biệt.

- Nếu như **Model parameter** được mô hình sinh ra từ chính tập dữ liệu huấn luyện
- thì **Model Hyperparameter** lại hoàn toàn khác. Nó hoàn toàn **nằm ngoài** mô hình và không phụ thuộc và tập dữ liệu huấn luyện.

Như vậy mục đích của nó là gì? Thực ra chúng có một vài nhiệm vụ như sau:

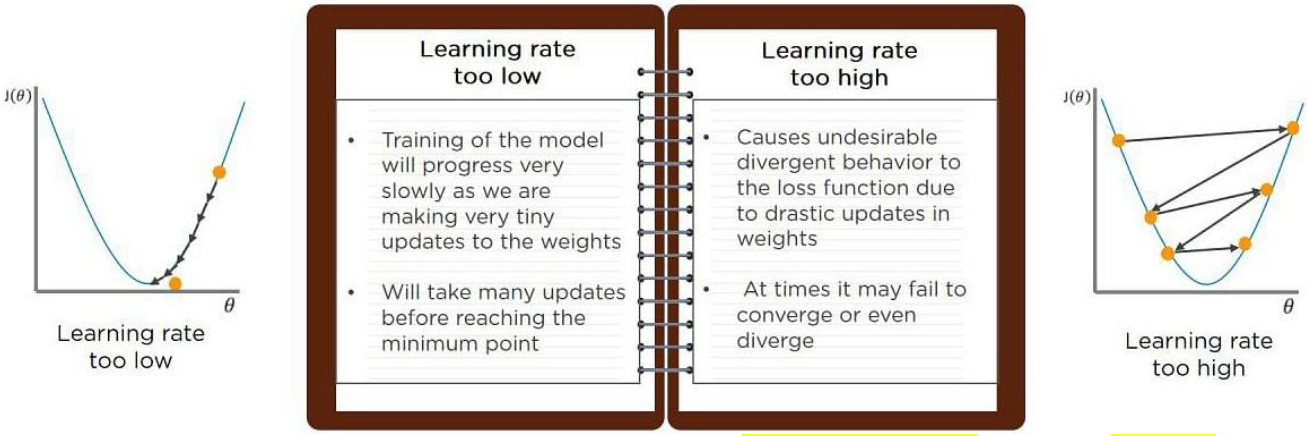
- Được sử dụng trong quá trình huấn luyện, giúp mô hình tìm ra được các **parameters** hợp lý nhất
- Nó thường được lựa chọn thủ công bởi những người tham gia trong việc huấn luyện mô hình
- Nó có thể được định nghĩa dựa trên một vài chiến lược **heuristics**

Chúng ta hoàn toàn không thể biết được đối với một bài toán cụ thể thì đâu là **Model Hyperparameter** tốt nhất. Chính vì thế trong thực tế chúng ta cần sử dụng một số kỹ thuật để ước lượng được một khoảng giá trị tốt nhất (Ví dụ như hệ số *k* trong mô hình **k Nearest Neighbor**) như **Grid Search** chẳng hạn.

Sau đây mình xin đưa một vài ví dụ về **Model Hyperparameter**:

- Chỉ số **learning rate** khi training một ***mạng nơ ron nhân tạo***
- Tham số **C** và **sigma** khi training một **Support Vector Machine**
- Hệ số ***k*** trong mô hình **k Nearest Neighbor**

8. Điều gì xảy ra khi learning rate quá lớn hoặc quá nhỏ?

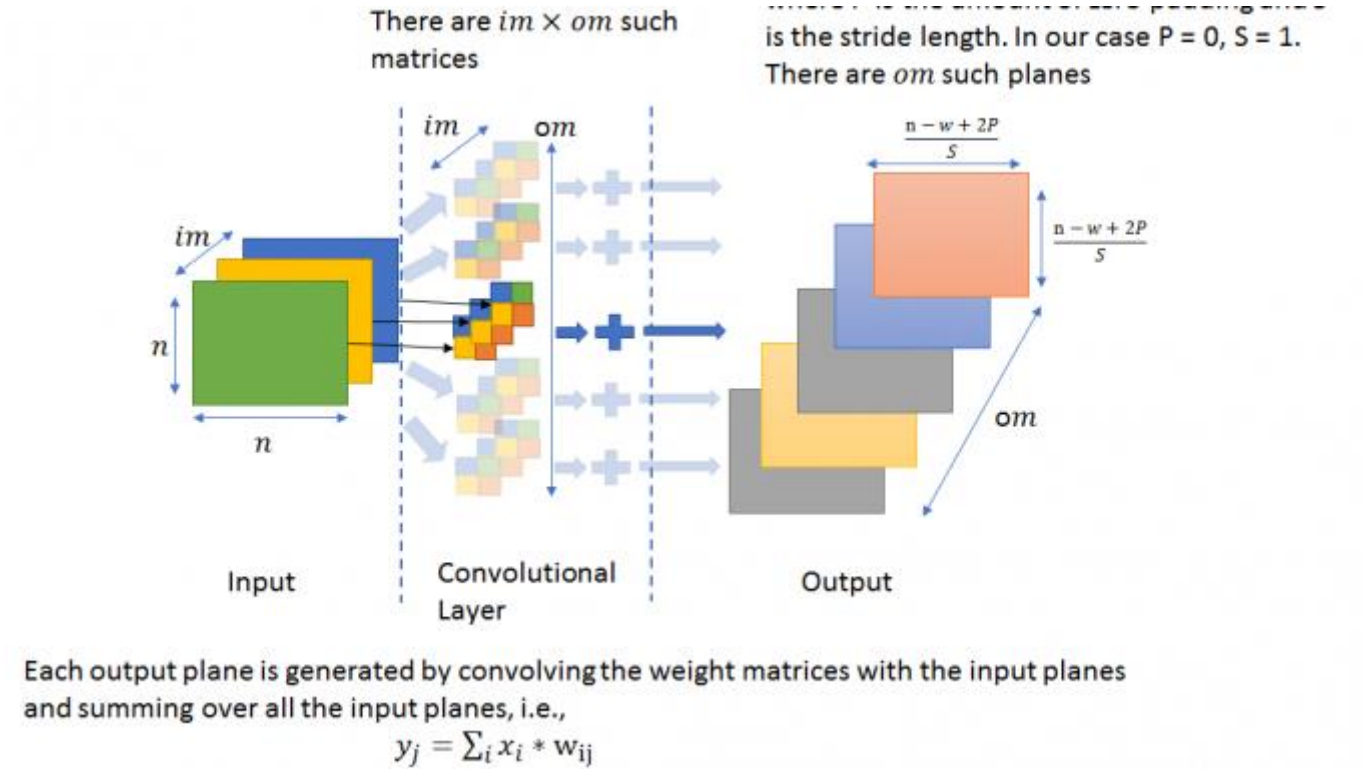


- Khi learning rate của mô hình được đặt quá thấp, việc huấn luyện mô hình sẽ diễn ra rất chậm vì nó phải thực hiện các cập nhật rất nhỏ cho các trọng số. Nó sẽ mất nhiều cập nhật trước khi đạt đến điểm tối ưu cục bộ
- Nếu learning rate được cài đặt quá cao thì khả năng mô hình sẽ khó hội tụ do sự cập nhật quá mạnh của các trọng số mà có thể trong một bước cập nhật trọng số thì mô hình đã vượt qua khỏi tối ưu cục bộ khiến cho các lần cập nhật sau đó mô hình sẽ khó có thể trở lại điểm tối ưu (có thể tưởng tượng mô hình đang chạy qua chạy lại giữa điểm tối ưu cục bộ do đã **nhảy quá xa**)

9. Khi kích thước ảnh đầu vào tăng gấp đôi thì số lượng tham số của CNN tăng lên bao nhiêu lần? Tại sao?

Đây là một câu hỏi rất dễ gây hiểu nhầm cho các ứng viên vì đa phần mọi người sẽ bị suy nghĩ theo hướng của câu hỏi rằng số lượng tham số của CNN sẽ tăng lên bao nhiêu lần.

Tuy nhiên hãy cùng nhìn lại kiến trúc của mạng CNN



Chúng ta có thể thấy số lượng tham số của mô hình CNN phụ thuộc vào số lượng, kích thước của các filter chứ không phụ thuộc vào ảnh đầu vào. Chính vì thế việc kích thước của ảnh tăng gấp đôi cũng không làm cho số lượng tham số của mô hình thay đổi.

10. Với những tập dữ liệu bị imbalance thì có những cách xử lý nào?

Đây là một câu hỏi kiểm tra việc tiếp cận của ứng viên đối với những bài toán có dữ liệu thực tế. Thông thường dữ liệu thực tế sẽ khác rất nhiều cả về tính chất cũng như số lượng dữ liệu so với các tập dữ liệu mẫu (các tập dữ liệu chuẩn không cần chỉnh).

Với tập dữ liệu trên thực tế có thể có một trường hợp là dữ liệu bị imbalanced tức dữ liệu bị mất cân bằng giữa các class. Lúc này chúng ta có thể tính đến các kỹ thuật sau:

Lựa chọn đúng metrics đánh giá mô hình:

Việc đầu tiên là cần phải lựa chọn chính xác metrics để đánh gía mô hình với một tập dữ liệu mất cân bằng thì việc sử dụng **accuracy** để đánh giá là một việc làm rất nguy hiểm như đã trình bày trong các phần trên, nên lựa chọn metrics đánh giá phù hợp như **Precision, Recall, F1 Score, AUC**

Resample lại tập dữ liệu training:

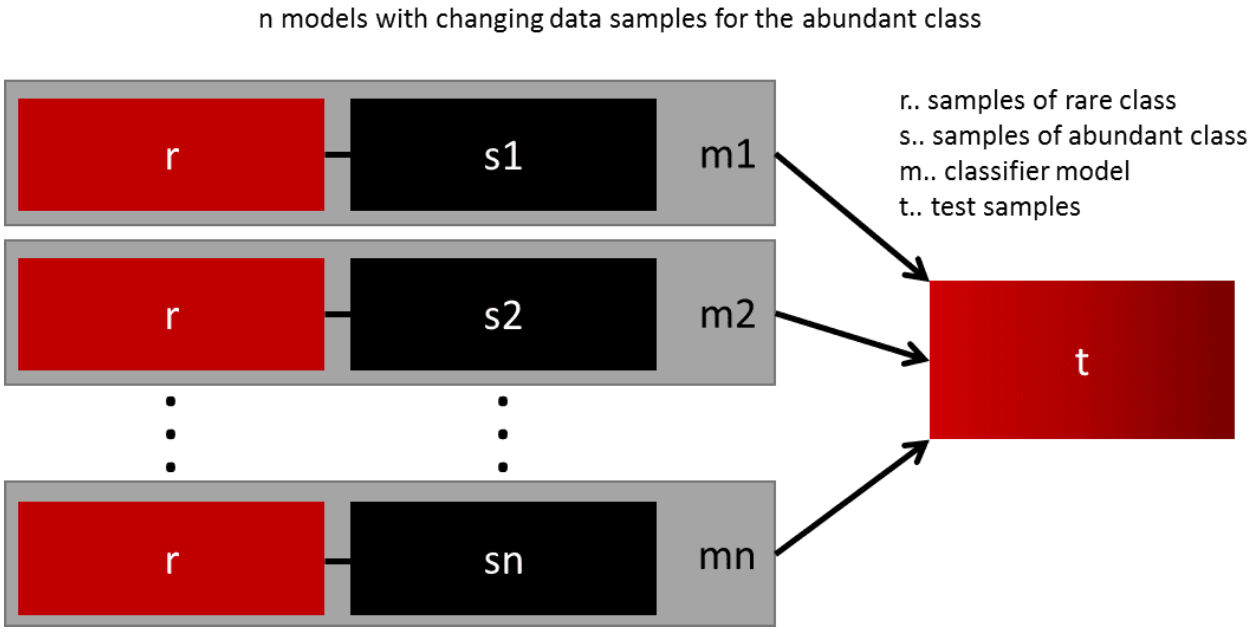
Ngoài việc sử dụng các tiêu chí đánh giá khác nhau, người ta cũng có thể áp dụng các kỹ thuật để có được các tập dữ liệu khác nhau.

Hai cách tiếp cận để tạo ra một bộ dữ liệu cân bằng từ một mất cân bằng đó là **Under-sampling** và **Over-sampling** với các phương pháp như repetition, bootstrapping or SMOTE (Synthetic Minority Over-Sampling Technique)

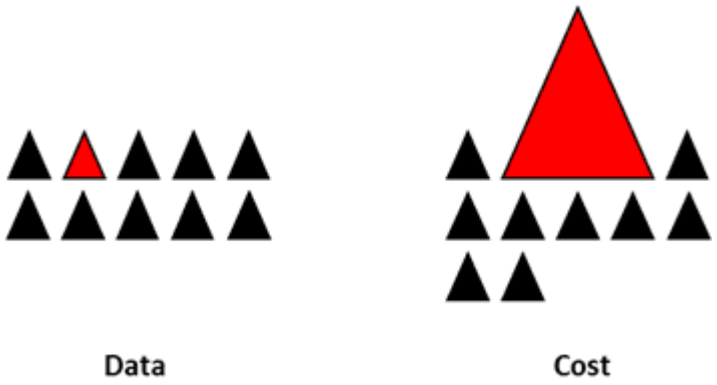
Ensemble nhiều mô hình khác nhau:

Việc thực hiện tổng quát hoá mô hình bằng cách tạo thêm nhiều dữ liệu không phải lúc nào cũng khả thi trong thực tế. Ví dụ bạn có hai lớp một lớp hiếm có 1000 dữ liệu, một lớp đại trà chứa 10000 mẫu dữ liệu. Vậy thay vì cố gắng tìm được 9000 mẫu dữ liệu của lớp hiếm để thực hiện training một mô

hình thì chúng ta có thể nghĩ đến giải pháp training 10 mô hình. Mỗi mô hình được training từ 1000 lớp hiếm và 1000 lớp đại trà. Sau đó sử dụng kỹ thuật ensemble để cho ra kết quả tốt nhất



Thiết kế lại mô hình - cost function:



Sử dụng các kỹ thuật penalty để phạt thật nặng (???) các lớp phong phú trong cost function giúp cho bản thân model có khả năng học tốt hơn các dữ liệu của lớp hiếm. Điều này khiến cho giá trị của hàm loss biểu diễn được tổng quát hơn giữa các lớp.

11. Các khái niệm Epoch, Batch và Iteration có ý nghĩa gì khi training mô hình Deep Learning

Đây là các khái niệm rất cơ bản trong khi training một mạng nơ ron nhưng thực tế là có khá nhiều ứng viên bị lúng túng khi phân biệt các khái niệm này. Cụ thể các bạn nên trả lời như sau:

- **epoch** - Đại diện cho một lần lặp trên toàn bộ tập dữ liệu (mọi thứ được đưa vào mô hình đào tạo).
- **batch** - Đề cập đến khi chúng ta không thể chuyển toàn bộ tập dữ liệu vào mạng nơ ron cùng một lúc, vì vậy chúng ta chia tập dữ liệu thành nhiều batch (một tập dữ liệu nhỏ hơn).
- **iteration**/ˌɪtəˈreɪʃn/ LÀ số batchs cần lặp để đi hết một epoch.

  Giả sử, chúng ta có 10.000 hình ảnh dưới dạng dữ liệu và kích thước của batch (batch_size) là 200. thì một epoch sẽ gồm có 50 Iteration (10.000 = 200 x 50).

12. Khái niệm Data Generator là gì? Cần dùng nó khi nào?

Là một khái niệm cũng rất quan trọng trong lập trình, đó là hàm sinh. Hàm sinh dữ liệu giúp cho chúng ta sinh ra trực tiếp dữ liệu để fit vào mô hình trong từng batch training.

Việc tận dụng hàm sinh giup ích rất nhiều trong quá trình training các dữ liệu lớn. Vì không phải lúc nào tập dữ liệu cũng cần phải load hết vào RAM gây lãng phí bộ nhớ, hơn nữa nếu như tập dữ liệu quá lớn thì có thể dẫn đên tràn bộ nhớ và thời gian tiền xử lý dữ liệu đầu vào sẽ lâu hơn.

Tổng kết

Trên đây là 12 câu hỏi phỏng vấn về Deep Learning mà mình thường hỏi ứng viên nhất trong quá trình phỏng vấn. Tuy nhiên tuỳ thuộc vào mỗi ứng viên mà cách hỏi sẽ khác nhau hoặc cũng có những câu hỏi được hỏi ngẫu hứng từ những bài toán mà ứng viên đã từng làm qua. Mặc dù bài viết là về các vấn đề kỹ thuật nhưng có liên quan đến việc phỏng vấn và quan điểm cá nhân của mình về việc phỏng vấn luôn là **thái độ quyết định đến 50% sự thành công của buổi phỏng vấn**. Vậy nên ngoài việc tích luỹ cho bản thân những kiến thức, kỹ năng cứng thì hãy luôn thể hiện mình với một thái độ chân thành, cầu tiến, khiêm tốn thì chắc chắn các bạn sẽ gặt hái được nhiều thành công trong bất cứ cuộc đối thoại nào. Chúc các bạn sớm đạt được những mong muốn của mình. Xin chào và hẹn gặp lại trong các blog tiếp theo.