**Python programming for Machine Learning**

1. **Định nghĩa**

***Câu 1: Supervised learning là gì?***

Supervised Learning (Học có giám sát) là một nhóm thuật toán sử dụng dữ liệu được gán nhãn nhằm mô hình hóa mối quan hệ giữa biến đầu vào (x) và biến đầu ra (y). Hai nhóm bài toán cơ bản trong học có giám sát là classification (phân loại) và regression (hồi quy), trong đó biến đầu ra của bài toán phân loại có các giá trị rời rạc trong khi biến đầu ra của bài toán hồi quy có các giá trị liên tục. Với Supervised Learning, bên cạnh xây dựng các mô hình mạnh, việc thu thập và gán nhãn dữ liệu tốt và hợp lý cũng đóng vai trò then chốt để giải quyết các bài toán trong thực tế.

***Câu 2: Supervised learning được phân thành mấy loại cơ bản?***

Phân loại (Classification): Một bài toán được phân nhóm classification nếu các nhãn của dữ liệu đầu vào được chia thành một số hữu hạn nhóm, giá trị của output cần dự đoán là các giá trị rời rạc. Ví dụ như một vấn đề thuộc nhóm Classification khi biến đầu ra là một danh mục, chẳng hạn như “màu đỏ” hoặc “màu xanh”, “bệnh” hoặc “không bệnh”.

Hồi quy (Regression): Một vấn đề hồi quy là khi biến đầu ra là một giá trị thực cụ thể và liên tục, chẳng hạn như “đơn vị đô la” hoặc “trọng lượng”.

***Câu 3: Supervised learning được chia nhỏ thành các loại nào?***

* Hồi quy tuyến tính:

Kỹ thuật hồi quy dự đoán một giá trị đầu ra duy nhất sử dụng dữ liệu huấn luyện.

Ví dụ: Sử dụng hồi quy để dự đoán giá nhà từ dữ liệu đào tạo. Các biến đầu vào sẽ là địa phương, kích thước của một ngôi nhà, …

* Hồi quy logistic:

Phương pháp hồi quy logistic được sử dụng để ước tính các giá trị rời rạc dựa trên một tập hợp các biến độc lập. Nó giúp dự đoán xác suất xảy ra sự kiện bằng cách khớp dữ liệu với chức năng logit. Do đó, nó còn được gọi là hồi quy logistic. Vì nó dự đoán xác suất, giá trị đầu ra của nó nằm trong khoảng từ 0 đến 1.

* Phân loại:

Phân loại có nghĩa là nhóm đầu ra bên trong một lớp. Nếu thuật toán cố gắn nhãn đầu vào thành hai lớp riêng biệt, nó được gọi là phân loại nhị phân. Chọn giữa nhiều hơn hai lớp được gọi là phân loại đa lớp.

* Naive Bayes Classifier:

Mô hình Naïve Bayesian (NBN) rất dễ xây dựng và rất hữu ích cho các bộ dữ liệu lớn. Phương pháp này bao gồm các biểu đồ chu kỳ trực tiếp với một phụ huynh và một vài đứa trẻ. Nó giả định sự độc lập giữa các nút con tách khỏi cha mẹ của chúng. Doanh nghiệp có thể áp dụng thuật toán này với công việc có nhiều biến số.

* Cây quyết định:

Thuật toán Cây quyết định sẽ đưa ra các giá trị nhánh dựa trên mỗi giá trị nút dữ liệu biến đầu vào. Mỗi nhánh cây là có dữ liệu biến đầu vào tương ứng với một kết quả biến đầu ra. Nó giúp bạn ước tính giá trị thực (chi phí mua xe, số lượng cuộc gọi, tổng doanh số hàng tháng, v.v.).

* Support Vector Machine:

SVM (Support Vector Machine) là một thuật toán học máy giám sát (supervised learning) được sử dụng để phân loại và dự đoán các dữ liệu phân loại hoặc số.

Ý tưởng chính của SVM là tìm một siêu phẳng (hyperplane) trong không gian n chiều (n là số lượng các đặc trưng của dữ liệu) để phân tách các điểm dữ liệu của các lớp khác nhau. Siêu phẳng này sẽ được tìm bằng cách tối đa hóa khoảng cách (margin) giữa siêu phẳng và các điểm gần nhất của các lớp khác nhau, được gọi là support vector.

Thuật toán SVM có thể được sử dụng cho các bài toán phân loại nhị phân hoặc đa lớp, và có thể được mở rộng để xử lý dữ liệu phi tuyến tính bằng cách sử dụng các hàm kernel. SVM sử dụng trong nhiều ứng dụng khác nhau như nhận dạng chữ viết tay, phân loại ảnh, hay dự đoán giá cổ phiếu.

***Câu 4: Ưu và nhược điểm của Supervised learning:***

Ưu điểm:

* Supervised Learning hay học có giám sát cho phép thu thập dữ liệu đầu vào và tạo ra dữ liệu đầu ra từ những đào tạo trước đó.
* Giúp tối ưu hóa các tiêu chí với sự trợ giúp của kinh nghiệm đã cài đặt.
* Học có giám sát giúp giải quyết nhiều loại vấn đề tính toán khác nhau trong thế giới thực.

Nhược điểm:

* Phân loại với nguồn dữ liệu lớn có thể là một thách thức.
* Mô hình cần quyết định trước các cấu trúc và thuật toán học.
* Mô hình cần nhiều ví dụ hay sử dụng nhiều loại nếu sử dụng thuật toán phân loại.
* Đào tạo cho việc học có giám sát cần nhiều thời gian tính toán.

***Câu 5: Unsupervised learning là gì?***

Unsupervised Learning (Học không giám sát) là một nhóm thuật toán sử dụng dữ liệu không có nhãn. Các thuật toán theo cách tiếp cận này hướng đến việc mô hình hóa được cấu trúc hay thông tin ẩn trong dữ liệu. Hay nói cách khác, sử dụng các phương pháp này thiên về việc mô tả tính chất hay đặc tính của dữ liệu. Thông thường, các thuật toán này dựa trên những thông tin sau:

* Mối quan hệ tương tự (similarity) giữa các ví dụ (được gọi là instance) trong dữ liệu như trong các thuật toán clustering (phân cụm).
* Xác suất đồng xuất hiện của các đối tượng như trong Association mining.
* Các phép biến đổi ma trận để trích xuất các đặc trưng như PCA, SVD.

***Câu 6: Phân loại Unsupervised learning?***

Phân cụm:

Phân cụm là một trong những phương pháp học máy không giám sát hữu ích nhất. Nó được sử dụng để tìm các mẫu mối quan hệ và sự tương đồng giữa các dữ liệu đầu vào. Sau khi tìm thấy các mẫu này, thuật toán không giám sát sẽ nhóm các mẫu dữ liệu có điểm giống nhau thành các nhóm như được minh họa trong sơ đồ bên dưới

Liên kết:

Liên kết được sử dụng để tìm các mẫu liên quan đại diện cho các mối quan hệ giữa nhiều mục dữ liệu trong một tập dữ liệu lớn. Một trong những ví dụ điển hình về sự liên kết là phân tích các mô hình mua sắm của khách hàng.

Giảm kích thước:

Giảm kích thước, như tên gọi, được sử dụng để cắt giảm số lượng các biến đặc trưng. Để làm điều này, nó chọn một tính năng chính cho mọi mẫu dữ liệu. Lý do chính đằng sau việc sử dụng phương pháp này là để loại bỏ vấn đề phức tạp của không gian đặc trưng. Phân tích thành phần chính (PCA) là một trong những phương pháp giảm kích thước phổ biến nhất.

Phát hiện bất thường:

Phát hiện bất thường, như tên gọi, có thể tự động khám phá các điểm dữ liệu bất thường trong tập dữ liệu của bạn. Điều này rất hữu ích trong việc xác định chính xác các giao dịch gian lận, phát hiện ra các phần cứng bị lỗi hoặc xác định lỗi ngoại lệ do lỗi của con người trong quá trình nhập dữ liệu.

***Câu 7: So sánh Supervised learning và Unsupervised learning.***

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Supervised Learning** | **Unsupervised Learning** |
| Mục tiêu | Để đào tạo thuật toán dự đoán. Kết quả chủ yếu xảy ra theo mong đợi của con người | Để đào tạo thuật toán tìm hiểu thông tin chi tiết từ khối lượng lớn dữ liệu chưa được phân loại |
| Dữ liệu để huấn luyện mô hình | Dữ liệu có nhãn | Dữ liệu không có nhãn |
| Kiến thức về lớp | Các lớp dữ liệu đã biết | Số lượng các lớp là không xác định |
| Cách thức học của mô hình | Mô hình hóa mối quan hệ giữa biến đầu vào và biến đầu ra | Học dựa trên các quan hệ tương tự, sự đồng xuất hiện, hay các phép biến đổi ma trận |
| Thuật toán | Support vector machine, Neural network, Hồi quy tuyến tính, Hồi quy logistics, Random forest và Classification trees. | – Các thuật toán clustering như K-mean, DBSCAN, Spectral Clustering, Hierarchical clustering- Apriori (Association Rule Mining)- PCA, SVD |
| Kết quả | Độ chính xác và tin cậy cao | Ít chính xác hơn |

***Câu 8: Evaluation protocol của Supervised learning.***

Evaluation protocol của supervised learning là quá trình đánh giá hiệu suất của mô hình học máy được huấn luyện trên một tập dữ liệu có nhãn (labeled dataset). Mục đích của quá trình đánh giá là xác định khả năng dự đoán của mô hình trên dữ liệu mới chưa được sử dụng trong quá trình huấn luyện.

Các bước trong quá trình đánh giá bao gồm:

1. Chia dữ liệu: Chia tập dữ liệu huấn luyện thành hai tập con: tập huấn luyện (training set) và tập kiểm tra (test set). Tập huấn luyện được sử dụng để huấn luyện mô hình, trong khi tập kiểm tra được sử dụng để đánh giá hiệu suất của mô hình.
2. Huấn luyện mô hình: Sử dụng tập huấn luyện để huấn luyện mô hình học máy. Quá trình này bao gồm tìm kiếm các tham số tối ưu của mô hình.
3. Đánh giá mô hình: Sử dụng tập kiểm tra để đánh giá hiệu suất của mô hình. Các độ đo thường được sử dụng để đánh giá hiệu suất bao gồm độ chính xác (accuracy), độ phủ (recall), độ chính xác dương tính (precision), F1-score và ma trận lỗi (confusion matrix).
4. Điều chỉnh mô hình: Nếu mô hình không cho kết quả tốt trên tập kiểm tra, ta có thể điều chỉnh các tham số của mô hình và huấn luyện lại mô hình với các tham số mới.
5. Đánh giá kết quả cuối cùng: Sau khi tinh chỉnh mô hình, sử dụng lại tập kiểm tra để đánh giá kết quả cuối cùng của mô hình. Nếu kết quả đạt yêu cầu, mô hình có thể được triển khai để sử dụng trong thực tế.

Quá trình đánh giá hiệu suất của mô hình là quan trọng để đảm bảo tính khả thi và độ chính xác của mô hình trên dữ liệu mới.

***Câu 9: Evaluation protocol của Unsupervised learning***

Evaluation protocol của unsupervised learning là quá trình đánh giá hiệu suất của mô hình học máy được huấn luyện trên một tập dữ liệu không có nhãn (unlabeled dataset). Mục đích của quá trình đánh giá là xác định khả năng học và khả năng tạo ra các đặc trưng (features) mới từ dữ liệu cho trước.

Các bước trong quá trình đánh giá bao gồm:

1. Chọn độ đo: Chọn các độ đo phù hợp để đánh giá hiệu suất của mô hình unsupervised learning. Các độ đo thường được sử dụng bao gồm độ tương đồng (similarity), độ đa dạng (diversity) và độ đo mất mát (loss).
2. Huấn luyện mô hình: Sử dụng tập dữ liệu không có nhãn để huấn luyện mô hình unsupervised learning. Quá trình này bao gồm tìm kiếm các tham số tối ưu của mô hình.
3. Đánh giá mô hình: Sử dụng các độ đo đã chọn để đánh giá hiệu suất của mô hình. Ví dụ, đối với bài toán gom cụm (clustering), ta có thể sử dụng các độ đo như độ tương đồng giữa các điểm trong cùng một cụm và độ đa dạng giữa các cụm khác nhau để đánh giá hiệu suất của mô hình.
4. Điều chỉnh mô hình: Nếu mô hình không cho kết quả tốt, ta có thể điều chỉnh các tham số của mô hình và huấn luyện lại mô hình với các tham số mới.
5. Đánh giá kết quả cuối cùng: Sau khi tinh chỉnh mô hình, sử dụng lại các độ đo đã chọn để đánh giá kết quả cuối cùng của mô hình. Nếu kết quả đạt yêu cầu, mô hình có thể được triển khai để sử dụng trong thực tế.

Quá trình đánh giá hiệu suất của mô hình unsupervised learning là quan trọng để đảm bảo tính khả thi và độ chính xác của mô hình trong việc học và tạo ra các đặc trưng mới từ dữ liệu không có nhãn.

***Câu 12: Cách xử lý missing value.***

NaN (Not a Number) là một giá trị đặc biệt được sử dụng để biểu thị các giá trị bị thiếu hoặc không xác định trong dữ liệu. Việc xử lý các giá trị NaN là một bước quan trọng trong tiền xử lý dữ liệu.

Dưới đây là một số phương pháp phổ biến để xử lý các giá trị NaN:

1. Loại bỏ dòng dữ liệu chứa NaN: Nếu số lượng NaN trong dữ liệu không quá nhiều so với tổng số dòng dữ liệu, ta có thể xóa các dòng chứa giá trị NaN để giảm thiểu ảnh hưởng của chúng đến mô hình.
2. Thay thế giá trị NaN bằng giá trị trung bình/median/most frequent: Đối với các trường dữ liệu có kiểu số, ta có thể thay thế giá trị NaN bằng giá trị trung bình của các giá trị còn lại trong cùng trường.
3. Thay thế giá trị NaN bằng giá trị mode: Đối với các trường dữ liệu có kiểu chuỗi hoặc rời rạc, ta có thể thay thế giá trị NaN bằng giá trị mode (giá trị xuất hiện nhiều nhất) của các giá trị còn lại trong cùng trường.
4. Sử dụng mô hình dự báo để dự đoán giá trị NaN: Ta có thể sử dụng các mô hình dự báo, chẳng hạn như hồi quy tuyến tính hoặc mạng nơ-ron, để dự đoán giá trị NaN dựa trên các giá trị còn lại trong cùng trường hoặc các trường khác.
5. Sử dụng các phương pháp khác nhau để thực hiện khôi phục dữ liệu: Có nhiều phương pháp khác nhau để khôi phục dữ liệu bị thiếu hoặc không xác định, chẳng hạn như kĩ thuật tăng cường dữ liệu (data augmentation), kĩ thuật khôi phục dữ liệu bằng cách tìm kiếm giá trị gần nhất (nearest neighbor imputation) hoặc sử dụng các thuật toán phân cụm để xác định giá trị NaN.

Tuy nhiên, việc xử lý các giá trị NaN cần được thực hiện cẩn thận, bởi vì phương pháp xử lý khác nhau có thể dẫn đến kết quả khác nhau và ảnh hưởng đến hiệu suất của mô hình. Do đó, việc lựa chọn phương pháp xử lý NaN cần được căn cứ vào tính chất của dữ liệu và mục đích sử dụng của mô hình. Một số phương pháp xử lý NaN có thể phù hợp với một loại dữ liệu nhưng không phù hợp với loại dữ liệu khác.

Ngoài ra, việc xử lý NaN cũng có thể tạo ra các giá trị nhiễu hoặc đưa ra kết quả sai lệch nếu không được thực hiện đúng cách. Vì vậy, nên kiểm tra và xử lý NaN trước khi tiến hành các bước tiền xử lý dữ liệu khác như chuẩn hóa, mã hóa, hoặc giảm chiều dữ liệu.

Các phương pháp khác nhau có thể kết hợp với nhau để xử lý các giá trị NaN. Ví dụ: nếu số lượng NaN nhỏ và dữ liệu có phân phối chuẩn, ta có thể sử dụng phương pháp điền giá trị trung bình, nếu số lượng NaN lớn ta có thể sử dụng mô hình dự báo để dự đoán giá trị NaN. Nếu các giá trị NaN xuất hiện ở cùng một dòng thì ta có thể xóa dòng đó.

Khi thực hiện xử lý NaN, nên đánh giá hiệu quả của phương pháp bằng cách so sánh kết quả của mô hình trước và sau khi xử lý NaN. Nếu phương pháp xử lý NaN ảnh hưởng đến kết quả của mô hình, ta có thể thay đổi phương pháp hoặc tìm kiếm phương pháp khác để xử lý NaN.

***Câu 13: Ép giá trị về miền 0-1, khi nào dùng minmax, khi nào dùng phân phối chuẩn***

* Thường thì chúng ta sử dụng Min-Max Scaling khi chúng ta quan tâm đến việc giữ lại phạm vi ban đầu của dữ liệu và muốn chuyển đổi giá trị của biến về khoảng [0,1]. Điều này có ý nghĩa khi chúng ta muốn các gias trị trong dữ liệu tương đương với nhau và giữ nguyên sự tương quan giữa các giá trị.
* Tuy nhiên, khi chúng ta muốn chuẩn hóa dữ liệu và tạo ra một phân phối chuẩn, thì chugns ta sử dụng phương pháp phân phối chuẩn (Standardization). Phân phối chuẩn giúp đưa các gias trị về cùng một thang đo, giản thiểu tác động của giá trị ngoại lai và cho pheps áp dụng các giả định phân phối chuẩn trong các mô hình thống kê và học máy.
* Tóm lại:

1. MinMax Scaling:
   * Khi muốn giữ lại phạm vi ban đầu của dữ liệu và đồng thời giữ nguyên sự tương quan giữa các giá trị. Ví dụ: Trong bài toán về ảnh, cacs điểm ảnh có giá trị từ 0 đến 255, và ta muốn giữ lại phạm vi này để giữ nguyên thông tin về độ sáng và màu sắc của ảnh.
2. Standardization:
   * Khi muốn chuẩn hóa dữ liệu và tạo ra một phân phối chuẩn, giảm thiểu tác động của giá trị ngoại lai và áp dụng các giả định phân phối chuẩn trong mô hình. Ví dụ: Trong vài toán dự đoán giá nhà, ta muốn chuẩn hóa các biến đầu vàu như diện tích, số phòng ngủ,..để các biến này có cùng thang đo và giả định rằng chúng tuân theo phân phối chuẩn.

* Tuy nhiên không có một quy tắc cứng và nhanh để xác định phương pháp chuẩn hóa phù hợp trong mọi trường hợp. Chúng ta nên xem xét kỹ càng đặc điểm của dữ lieeujm yêu cầu của bài toán và cân nhắc sử dụng Min-Max Scaling hoặc Standardization dựa trên hiểu biết của mình về dữ liệu và mục tiêu của mô hình.
* Do đó, khi sử dụng MinMax scaling hoặc phân phối chuẩn (Standardization), cần xem xét yêu cầu của thuật toán và mô hình, kiểm tra phân phối của dữ liệu và điều chỉnh phương pháp chuẩn hóa dữ liệu phù hợp để đạt được kết quả tốt nhất.

***Câu 14: Các loại độ đo, độ đo phù hợp với các loại bài toán (phân lớp, hồi quy). Tại sao có nhiều độ đo khác nhau cho mỗi lớp bài toán? Biết được bản chất khác nhau giữa các bài toán là sẽ làm được. Ứng với mỗi stage của một project model thì nêu ít nhất 3pp.***

Các loại độ đo trong machine learning được sử dụng để đánh giá hiệu suất và độ chính xác của mô hình trên dữ liệu test. Tuy mỗi bài toán có loại độ đo phù hợp riêng, nhưng có một số độ đo phổ biến được sử dụng trong các bài toán phân lớp và hồi quy. Dưới đây là một số độ đo phổ biến và phù hợp với từng loại bài toán:

* Phân lớp (classification):
  + Accuracy (độ chính xác): đây là độ đo đơn giản và phổ biến nhất, tính tỉ lệ dự đoán chính xác của mô hình trên tất cả các điểm dữ liệu. Tuy nhiên, độ đo này không phù hợp khi dữ liệu mất cân bằng (unbalanced data) với sự không đối xứng giữa các lớp.

Accuracy\_score(y\_pred, y\_test)

* + Precision và Recall thường được sử dụng cho bài toán phân loại mà tập dữ liệu của các lớp là chênh lệch nhau rất nhiều.
  + Precision: Precision được định nghĩa là tỉ lệ số điểm true positive trong số những điểm được phân loại là positive (TP + FP)

TP/(TP+FP)

Precision\_score(y\_pred, y\_test)

* + Recall: là tỉ lệ số điểm positive được dự đoán đúng (TP+FN)

TP/(TP+FN)

* + F1-score: là sự kết hợp giữa precision và recall để đánh giá hiệu suất tổng thể của mô hình phân lớp

2\*precision\*recall/(precision+recall)

* Hồi quy (Regression):
  + Mean Squared Error (MSE): đo sai số bình phương trung bình giữa các giá trị được dự đoán và giá trị thực tế. MSE là thước đo chất lượng của một công cụ ước tính, nó luôn không âm và các giá trị càng gần 0 càng tốt.

A picture containing font, text, white, typography

Description automatically generated

* + Mean Absolute Error (MAE): đo lường sai số trung bình tuyệt đối giữa các giá trị dự đoán và giá trị thực tế

A picture containing font, text, white, design

Description automatically generated

* + Root Mean Squared Error (RMSE): căn bậc 2 của mse. Đo lường sai số trung bình và có cùng đơn vị với biến phụ thuộc. A picture containing font, white, text, diagram

    Description automatically generated

Giống nhau: cả MAE và RMSE đều thể hiện trung bình lỗi dự đoán của mô hình theo đơn vị được quan tâm. Cả hai giá trị có thể nằm trong khoảng từ 0 đến vô cùng. Chúng là những điểm số theo định hướng tiêu cực, có nghĩa là giá trị càng thấp thì càng tốt.

Khác nhau: Trong RMSE, do các lỗi được bình phương trước khi được tính trung bình, RMSE đưa ra trọng số tương đối cao hơn cho các lỗi lớn so với MAE. Điều này có nghĩa là RMSE sẽ hữu ích hơn khi đánh giá các giá trị dự đoán có chênh lệch lớn so với giá trị thực tế.

* + R2: hệ số R2 có thể xem là thước đo độ chính xác của một mô hình. Trong mô hình hồi quy, nó là thước đo thống kê về mức độ đường hồi quy dự đoán được gần đúng với các dữ liệu thực tế.

A picture containing text, font, line, white

Description automatically generated

* Mô hình phân loại: việc sử dụng nhiều độ đo cho phép ta có cái nhìn đa chiều và phản ánh đúng các khía cạnh quân trọng của mô hình phân loại. tùy thuộc vào yêu cầu và mục tiêu của bài toán, ta có thể lựa chọn và sử dụng độ đo phù hợp để đánh giá và so sánh hiệu suất của các mô hình phân loại khác nhau.
* Mô hình hồi quy: việc có nhiều độ đo trong bài toán hồi quy cho phép t có cái nhìn đa chiều về hiệu suất của mô hình dự đoán và lựa chọn độ đo phù hợp dựa trên yêu cầu và mục tiêu của bài toán

Các độ đo được sử dụng trong các bài toán khác nhau vì mỗi bài toán có yêu cầu và mục tiêu khác nhau. Ví dụ, trong phân lớp, độ chính xác có thể không phù hợp khi dữ liệu mất cân bằng, nên ta sử dụng precision, recall và F1-score để đánh giá hiệu suất phân loại trên từng lớp riêng biệt hoặc AUC-ROC để đánh giá khả năng phân biệt. Trong hồi quy, MSE, MAE và R-squared được sử dụng để đánh giá sự chính xác và khả năng giải thích của mô hỉnh.

Các độ đo khác nhau giữa các bài toán phản ánh bản chất và yêu cầu của từng loại bài toán. Các độ đo được thiết kế để đánh giá và đo lường mục tiêu cụ thể của từng loại bài toán, như dự đoán nhãn lớp trong phân lớp hay dự đoán giá trị số trong hồi quy. Việc sử dụng đúng độ đo phù hợp với bài toán giúp đảm bảo đánh giá và đo lường hiệu suất mô hình một cách chính xác và đáng tin cậy.

***Câu 15: Sơ lược về eda, tại sao eda? Eda sai thì sao? Ví dụ. Vậy nếu không có eda thì sao, không có thì mô hình sẽ ra sao, data bị gì, khác biệt ra sao?***

EDA (Exploratory Data Analysis) là quá trình khám phá và phân tích dữ liệu để hiểu sâu về dữ liệu, khám phá các mối quan hệ, xu hướng và tính chất của dữ liệu. EDA giúp xác định các đặc điểm quan trọng của dữ liệu, tìm hiểu về phân phối, tương quan và biến thiên của các biến, phát hiện giá trị ngoại lệ và thiếu sót dữ liệu, từ đó xây dựng cơ sở để đưa ra quyết định về xử lý dữ liệu, lựa chọn mô hình và định hình chiến lược phân tích.

EDA (Exploratory Data Analysis) là quá trình khám phá dữ liệu để hiểu và khám phá thông tin từ tập dữ liệu mà chúng ta có sẵn. EDA giups chungs ta có cái nhìn tổng quan về dữ liệu, tìm hiểu các mẫu, quan hệ, xu hướng và biểu đồ trong dữ liệu.

Các lợi ích của EDA:

* Hiểu rõ dữ liệu: EDA giúp chúng ta nắm bắt được sự phân bố, tính chất và sự biến thiên của dữ liệu. Điều này giúp hiểu rõ hơn về nguồn gốc và ý nghĩa của dữ liệu
* Phát hiện các mối quan hệ: EDA giúp ta tìm ra các mối quan hệ và tương tác giữa các biến. Bằng cách khám phá mối liên hệ giữa các biến, ta có thể đưa ra các giả định và hy vọng về quan hệ giữa các biến trong mô hình.
* Phát hiện giá trị ngoại lệ và dữ liệu thiếu: EDA giúp ta xác định và xử lý các giá trị ngoại lệ (outliers) và dữ liệu thiếu (mising data). Điều này là cần thiết để đảm bảo tính chính xác và đáng tin cậy của phân tích dữ liệu.
* Hướng dẫn xử lý dữ liệu: EDA cho pháp ta đưa ra quyết định về việc xử lý dữ liệu, bao gồm việc chọn lọc, rời rạc hóa, chuẩn hóa hoặc sửa chữa dữ liệu. Điều này giúp cải thiện chất lượng dữ liệu và đảm bảo phù hợp cho mô hình.

Vì sao cần khám phá dữ liệu trong phân tích dữ liệu?

* Hỗ trợ làm sạch dữ liệu với các kỹ thuật xác định các giá trị bị thiếu, sai sót hoặc các điểm dữ liệu bất thường
* Nắm rõ đặc điểm, cấu trúc và mô hình của tập dữ liệu
* Phát triển và kiểm chứng các giả thuyết và giả định
* Xây dựng data model
* Xác định phạm vi sai lệch của dữ liệu
* Xác định các công cụ thống kê và kỹ thuật phân tích thích hợp nhất
* Phát hiện các pattern, xu hướng thay đổi của các biến
* Hiểu rõ hơn về đặc điểm mô tả của các biến và tập dữ liệu

Nếu không thực hiện EDA hoặc EDA không chính xác, có thể xảy ra những vấn đề sau:

* Không hiểu rõ cấu trúc dữ liệu: nếu không hiểu rõ cấu trúc dữ liệu, có thể gặp khó khăn trong việc xác định loại dữ liệu và biểu diễn đúng cho mỗi biến. Điều này có thể dẫn đến việc sử dụng các phương pháp xử lý dữ liệu không phù hợp.
* Bỏ qua giá trị thiếu và nhiễu: Nếu không phát hiện được giá trị thiếu hoặc nhiễu trong dữ liệu, các phân tích và mô hình hóa có thể bị ảnh hưởng bởi các giá trị không hợp lý hoặc gây sai lệch trong kết quả.
* Thiếu thông tin quan trọng: EDA giúp chúng ta khám phá thông tin quan trọng và tìm hiểu mô hình dữ liệu. Nếu thiếu EDA hoặc không thực hiện đúng, có thể bot qua các thông tin quan trọng và không hiểu rõ quan hệ giữa các biến
* Không chuẩn bị dữ liệu đúng cách: EDA giúp chúng ta hiểu rõ dữ liệu và chuẩn bị nos cho quá trình mô hình hóa. Nếu không thực hiện Eda hoặc không chuẩn bị dữ liệu đúng cách có thể dẫn đến việc sử dụng các phương pháp xử lý dữ liệu không phù hợp hoặc không tối ưu.
* Sai lầm trong mô hình hóa: Nếu khoogn có EDA, cos thể bỏ qua các quy tắc và mẫu xuất hiện trong dữ liệu. Điều này có thể dẫn đến việc xây dựng mô hình không chính xác hoặc không phù hợp với dữ liệu thực tế.

Nếu không thực hiện eda:

* Mô hình: dữ liệu không phù hợp, overfitting hoặc underfitting, thiếu thông tin quan trọng, không xử lý sai sót và nhiễu, không tận dụng thông tin tiềm năng
* Data: thiếu thông tin, sai sót và nhiễu, biased hoặc không cân bằng, không tận dụng thông tin.
* Khác biệt: thiếu thông tin, rủi ro overfitting, sai sót và nhiễu, lựa chọn không tốt về biến.

Nếu không thực hiện eda hoặc eda bị sai, chúng ta có thể bỏ qua các vấn đề quan trọng trong dữ liệu, không thể hiểu rõ mô hình dữ liệu và có thể dẫn đến các dự đoán hoặc quyết định không chính xác

***Câu 16: tại sao phải preprocess data:***

Dữ liệu thô có thể chứa các giá trị bị thiếu, không nhất quán hoặc thừa thông tin. Việc tiền xử lý dữ liệu sẽ làm giảm đáng kể các khuyển điểm này của bộ dữ liệu, cải thiện độ chính xác và độ tin cậy của dữ liệu, làm cho dữ liệu nhất quán và cải thiện khả năng dự đoán của các mô hình máy học.

***Câu 17: Tại sao phải train\_test\_split***

Trong máy học, việc chia tập dữ liệu thành taapjt rain và test thường được thực hiện để tránh overfit. Đó là một trường hợp mà một mô hình học quá kỹ dữ liệu ban đầu và khó học dữ liệu mới. Tập train sẽ được dùng để huấn luyện mô hình, tập test sẽ dùng để đánh giá mô hình qua việc dự đoán các dữ liệu mới

***Câu 18: khi nào dùng k-fold cross validation***

Cross-validation thường được sử dụng với mục đích tận dụng bộ dữ liệu một cách tối đa, đặc biệt là khi bộ dữ liệu quá nhỏ, khó có thể áp dụng các phương pháp phân chia khác như chia train, test. Trong k-fold cross-validation, bộ dữ liệu được chia thành k phần có kích thước bằng nhau để tránh bias.

**Ôn python máy học**

# **Lý thuyết**

**Quá trình thực hiện một bài toán máy học**

- Preprocessing

+ Đọc/ghi dữ liệu

+ Trực quan hóa dữ liệu

+ Lọc nhiễu

+ Chuẩn hóa dữ liệu

- Chọn mô hình : fit, predict

- Đánh giá mô hình

+ Chọn Độ đo

+ Validation: đánh giá trong quá trình train /Testing: đánh giá sau khi train(dùng cho các bài toán không có output sẵn)

- Tune(siêu) tham số mô hình: tham số chỉ được thay đổi khi khởi tạo mô hình và ko đc thay đổi trong suốt quá trình

+ Grid Search

+ Random

- Deploy mô hình

## **1. metric**

- Confusion matrix:

|  |  |  |
| --- | --- | --- |
|  | Predicted as Positive | Predicted as Neg |
| Actually Pos | TP | FN |
| Actually Neg | FP | TN |

-

*-*

*-*

*-*

## **2. Phân nhóm các thuật toán Machine Learning**

**- Supervised Learning (Học có giám sát)**

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). Supervised learning là nhóm phổ biến nhất trong các thuật toán Machine Learning.

**+ Classification (Phân loại)**

Một bài toán được gọi là classification nếu các label của input data được chia thành một số hữu hạn nhóm.

1. K - Nearest – Neighbor

2. Logistic Regression

3. Decision Tree

4. SVM – Support Vector Machines

5. Kernel SVM

6. Naive Bayes

7. Random Forest Classification

8. Neural Networks

9. Deep Learning

**+ Regression (Hồi quy)**

Nếu label không được chia thành các nhóm mà là một giá trị thực cụ thể.

1. Linear Regression.

2. Logistic Regression.

**- Unsupervised Learning (Học không giám sát)**

Trong thuật toán này, chúng ta không biết được outcome hay nhãn mà chỉ có dữ liệu đầu vào. 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.

\_ Clustering

\_ Dimension reduction :PCA

Học phân bổ trên tập dữ liệu.

Giảm chiều của dữ liệu bằng cách phân cụm.

Để đánh giá được kết quả của bài toán unsupervised learning phải xem xét và đánh giá các giá trị lân cận của nó

**+ Clustering (phân nhóm)**

1. K-means

Một bài toán phân nhóm toàn bộ dữ liệu X thành các nhóm nhỏ dựa trên sự liên quan giữa các dữ liệu trong mỗi nhóm.

**+ Association**

Là bài toán khi chúng ta muốn khám phá ra một quy luật dựa trên nhiều dữ liệu cho trước.

## **3. overfitting**

- Overfitting là hiện tượng mô hình tìm được quá khớp với dữ liệu training. Việc quá khớp này có thể dẫn đến việc dự đoán nhầm nhiễu, và chất lượng mô hình không còn tốt trên dữ liệu test nữa.

- Về cơ bản, overfitting xảy ra khi mô hình quá phức tạp để mô phỏng training data. Điều này đặc biệt xảy ra khi lượng dữ liệu training quá nhỏ trong khi độ phức tạp của mô hình quá cao

- Muốn biết chương trình có bị overfitting hay ko thì phải vẽ 2 biểu đồ loss và validation

- Overfitting xảy ra khi kết quả trên tập test tệ hơn kết quả trên tập train rất nhiều khoảng 20%

**- Tránh overfitting:**

+ Dùng tập validation: *trích* từ tập training data ra một tập con nhỏ và thực hiện việc đánh giá mô hình trên tập con nhỏ này. Thông thường, ta bắt đầu từ mô hình đơn giản, sau đó tăng dần độ phức tạp của mô hình. Tới khi nào *validation error* có chiều hướng tăng lên thì chọn mô hình ngay trước đó.

+ Cross-validation: chia tập training ra k tập con không có phần tử chung, có kích thước gần bằng nhau. Tại mỗi lần kiểm thử , được gọi là *run*, một trong số k tập con được lấy ra làm *validate set*. Mô hình sẽ được xây dựng dựa vào hợp của k-1 tập con còn lại. Mô hình cuối được xác định dựa trên trung bình

+ Early Stopping: Early stopping tức dừng thuật toán trước khi hàm mất mát đạt giá trị quá nhỏ, giúp tránh overfitting.

+ weight decay, …

## **4. preprocessing**

Hầu hết các bộ dữ liệu được sử dụng trong các vấn đề liên quan đến Học Máy cần được xử lý, làm sạch và biến đổi trước khi một thuật toán Học Máy có thể được huấn luyện trên những bộ dữ liệu này. Các kỹ thuật tiền xử lý dữ liệu phổ biến hiện nay bao gồm: xử lý dữ liệu bị khuyết (missing data), mã hóa các biến nhóm (encoding categorical variables), chuẩn hóa dữ liệu (standardizing data), co giãn dữ liệu (scaling data),…

## **5. evaluation protocol**

Khi đào tạo các mô hình máy học, một trong những bước quan trọng nhất là quyết định quy trình đánh giá. Cách chọn để đánh giá hiệu suất của mô hình có thể thay đổi kết quả của các thuật toán.

Bao gồm: tiền xử lý dữ liệu ntn? Chia train test bao nhiêu, có dùng tập val không? Dùng cross- validation? Dùng metric gì để đánh giá?

# **Coding**

Các bài tập trên course

* Preprocession:
  + Sử dụng thư viện pandas để đọc dữ liệu sau đó dùng hàm .head() để trực quan hóa dữ liệu
  + Sử dụng hàm .shape để kiểm tra số lượng giá trị và số lượng feature của data
  + Sử dụng hàm .infor() để xem thông tin data (bao gồm các feature, số lượng giá trị có trong từng feature, số lượng giá trị bị thiếu trong từng feature và kiểu dữ liệu của từng feature).
  + Sử dụng hàm .dropna() để xóa các dòng có giá trị bị thiếu.
  + Chuẩn hóa dữ liệu:
    - Sử dụng tất cả các feature ngoại trừ feature address (đối với address thường là các giá trị unique nên sẽ không cần thiết để train mô hình).
    - Sử dụng G3 cho y còn lại sử dụng cho X.
    - Sử dụng hàm OneHotEncoder (One-Hot Encoding) để chuẩn hóa feature school, sex, famsize.
    - Sử dụng hàm LabelEncoder (Label Encoding) để chuẩn hóa feature Mjob và Fjob.
  + Chia tập dữ liệu: sử dụng hàm train\_test\_split để chia tập train và test theo tỉ lệ 8/2 với random\_state = 42.
* Chọn mô hình: đây là một bài toán không quá phức tạp và output của bài toán dạng regression nên em sẽ chọn model LinearRegression để train model.
* Đánh giá mô hình:
  + Sau khi train xong tiến hành predict trên tập x\_test để thực hiện việc đánh giá mô hình.
  + Sử dụng độ đo MSE và MAE để tính độ lỗi của kết quả predict so với kết quả thực tế.

from sklearn.decomposition import PCA

# Khởi tạo PCA với số thành phần chính cần giữ lại

pca = PCA(n\_components=2)

# Fit và transform dữ liệu vào PCA

X\_pca = pca.fit\_transform(X)

import xgboost as xgb

# Tạo và huấn luyện mô hình XGBoost

model = xgb.XGBClassifier() # XGBoost cho bài toán phân lớp

# Hoặc sử dụng: model = xgb.XGBRegressor() # XGBoost cho bài toán hồi quy

model.fit(X\_train, y\_train)

# Dự đoán

y\_pred = model.predict(X\_test)

# Đánh giá mô hình

accuracy = model.score(X\_test, y\_test)

from sklearn.model\_selection import GridSearchCV

from sklearn.svm import SVC

# Tạo một đối tượng mô hình

model = SVC()

# Định nghĩa các giá trị tham số cần tìm kiếm

param\_grid = {'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf'], 'gamma': [0.1, 0.5]}

# Tạo đối tượng GridSearchCV

grid\_search = GridSearchCV(estimator=model, param\_grid=param\_grid, cv=5)

# Huấn luyện mô hình với Grid Search

grid\_search.fit(X\_train, y\_train)

# In ra kết quả tốt nhất

print("Best parameters found: ", grid\_search.best\_params\_)

print("Best accuracy found: ", grid\_search.best\_score\_)

import pandas as pd

from sklearn.preprocessing import OneHotEncoder

# Tạo DataFrame mẫu

data = {'color': ['Red', 'Blue', 'Green', 'Red', 'Yellow']}

df = pd.DataFrame(data)

# Sử dụng One-Hot Encoding

encoder = OneHotEncoder()

encoded\_data = encoder.fit\_transform(df[['color']]).toarray()

# Tạo DataFrame mới từ dữ liệu đã mã hóa

encoded\_df = pd.DataFrame(encoded\_data, columns=encoder.get\_feature\_names(['color']))

# In kết quả

print(encoded\_df)

from sklearn.preprocessing import LabelEncoder

# Tạo một đối tượng LabelEncoder

encoder = LabelEncoder()

# Chuẩn bị dữ liệu đầu vào

data = ['red', 'blue', 'green', 'red', 'blue']

# Áp dụng Label Encoding cho dữ liệu

encoded\_data = encoder.fit\_transform(data)

# In kết quả

print(encoded\_data)

from sklearn.model\_selection import KFold

from sklearn.linear\_model import LinearRegression

import numpy as np

# Tạo một mảng dữ liệu đặc trưng X và một mảng nhãn y (giả sử đã được chuẩn bị trước)

X = np.array([[1, 2], [3, 4], [5, 6], [7, 8], [9, 10]])

y = np.array([3, 5, 7, 9, 11])

# Sử dụng K-Fold Cross Validation với số folds là 3

kfold = KFold(n\_splits=3, shuffle=True, random\_state=42)

# Huấn luyện và đánh giá mô hình trên từng fold

for train\_index, test\_index in kfold.split(X):

X\_train, X\_test = X[train\_index], X[test\_index]

y\_train, y\_test = y[train\_index], y[test\_index]

model = LinearRegression()

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

# Đánh giá hiệu suất mô hình (ví dụ: tính MSE)

mse = np.mean((y\_test - y\_pred) \*\* 2)

print("MSE:", mse)

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error

# Dữ liệu dự đoán

y\_pred = [1.5, 2.0, 3.5, 4.8, 2.1]

# Dữ liệu thực tế

y\_true = [1.2, 2.5, 3.8, 4.0, 2.8]

# Tính MSE

mse = mean\_squared\_error(y\_true, y\_pred)

print("MSE:", mse)

# Tính MAE

mae = mean\_absolute\_error(y\_true, y\_pred)

print("MAE:", mae)

# Tính RMSE

rmse = mean\_squared\_error(y\_true, y\_pred, squared=False)

print("RMSE:", rmse)

Kfold = Kfold(n\_splits=3, shuffle = true, random\_state=42)

For train\_index, test\_index in Kfold.split(X):  
 X\_train,X\_test = X[train\_index], X[test\_index]

Y\_train,y\_test = y[train\_index], y[test\_index]

Pca = PCA(n\_components = 2)

Pca.fit(X\_train)

X\_train = pca.transform(X\_train)

X\_test = pca.transform(X\_test)

XGBClassifier

XGBRegressor

Encoder = LabelEncoder

Encode\_data = encoder.fit\_transform(data)

Df=pd.DataFrame(data)

Encoder = OneHotEncoder()

Encode\_data = encoder.fit\_transform(df[‘color’]).toarray()

Encode\_df = pd.DataFrame(encode\_data, columns=encoder.get\_feature\_names([‘color]’))