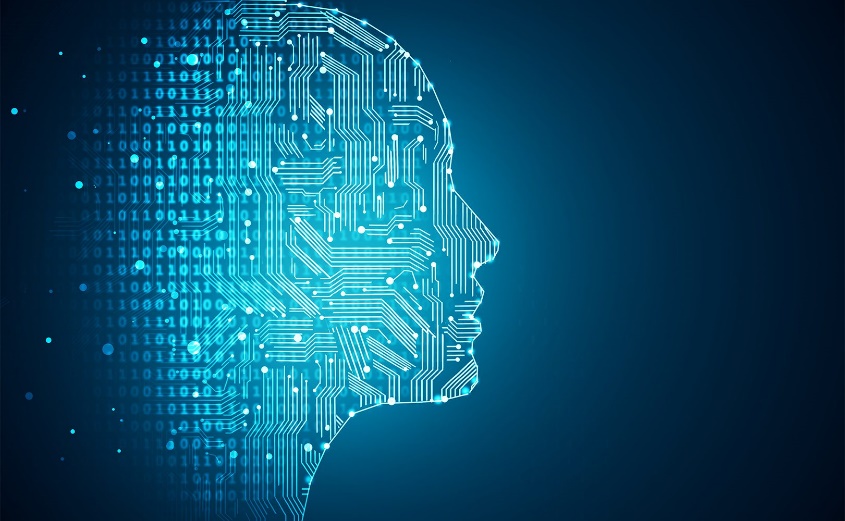
**BDRPC187TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**HOMEWORK**

**MACHINE LEARNING**

**CÁC THUẬT TOÁN MACHINE LEARNING CƠ BẢN**

****

**Giảng viên: TS. Vũ Quang Huy**

**Sinh viên thực hiện:**

**Lê Minh Chương – 15110020**

**Nhận xét của giảng viên hướng dẫn**

1. **Linear Regression**
2. Tìm hiểu thuật toán:

"Hồi quy tuyến tính" là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục. Ví dụ, dự đoán giao thông ở một cửa hàng bán lẻ, dự đoán thời gian người dùng dừng lại một trang nào đó hoặc số trang đã truy cập vào một website nào đó v.v...

* Tương quan (r) - Giải thích mối quan hệ giữa hai biến, giá trị có thể chạy từ -1 đến +1
* Phương sai (σ2) - Đánh giá độ phân tán trong dữ liệu của bạn
* Độ lệch chuẩn (σ) - Đánh giá độ phân tán trong dữ liệu của bạn (căn bậc hai của phương sai)
* Phân phối chuẩn
* Sai số (lỗi) - {giá trị thực tế - giá trị dự đoán}

1. Bài toán:

Chúng ta có 1 bảng dữ liệu về chiều cao và cân nặng của 15 người như dưới đây:

| **Chiều cao (cm)** | **Cân nặng (kg)** | **Chiều cao (cm)** | **Cân nặng (kg)** |
| --- | --- | --- | --- |
| 147 | 49 | 168 | 60 |
| 150 | 50 | 170 | 72 |
| 153 | 51 | 173 | 63 |
| 155 | 52 | 175 | 64 |
| 158 | 54 | 178 | 66 |
| 160 | 56 | 180 | 67 |
| 163 | 58 | 183 | 68 |
| 165 | 59 |  |  |

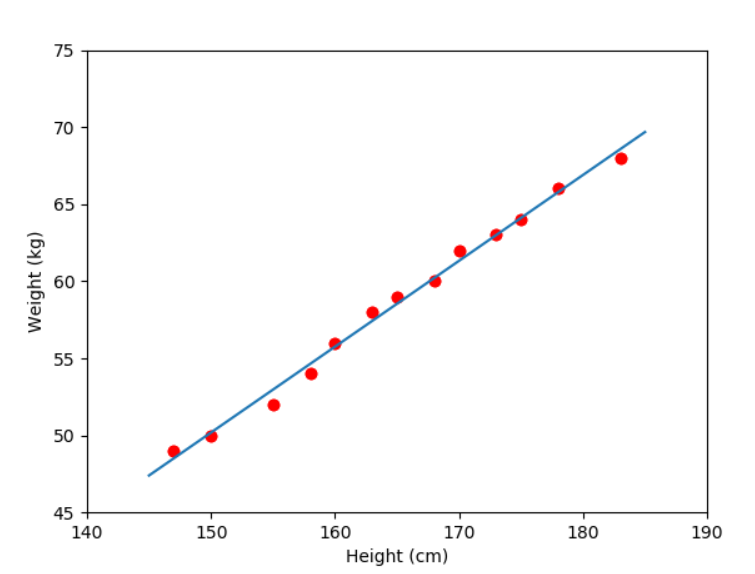
Bài toán đặt ra là: liệu có thể dự đoán cân nặng của một người dựa vào chiều cao của họ không?

Chúng ta có thể thấy là cân nặng sẽ tỉ lệ thuận với chiều cao (càng cao càng nặng), nên có thể sử dụng Linear Regression model cho việc dự đoán này. Để kiểm tra độ chính xác của model tìm được, chúng ta sẽ giữ lại cột 153 và 180 cm để kiểm thử, các cột còn lại được sử dụng để huấn luyện (train) model.

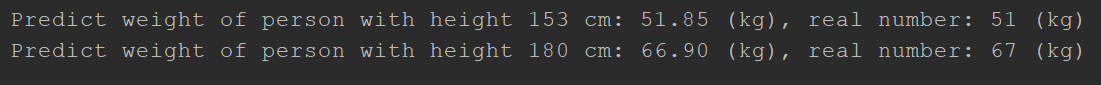
* Input: bảng dữ liệu
* Output: kết quả dự đoán được

1. Thực hiện:

* Đồ thị:



* Kết quả dự đoán:



1. **K-means Clustering**
2. Tìm hiểu thuật toán:

* Ý tưởng của thuật toán k-means:
* Phương pháp k-means là 1 trong số những thuật toán phân cụm (clustering). Đầu vào tập dữ liệu cần phân cụm và số cụm (cluster), đầu ra chúng ta sẽ được kết quả dữ liệu đã được phân về các cluster.
* Mục đích là phân dữ liệu thành các cụm (cluster) khác nhau sao cho dữ liệu trong cùng một cụm có tính chất giống nhau.

1. Bài toán: Tách vật thể

Giả sử chúng ta có bức ảnh dưới đây và muốn một thuật toán tự động nhận ra vùng khuôn mặt và tách nó ra.



* Input: hình ảnh
* Output: hình ảnh đã được phân cụm theo màu sắc chính

1. Thực hiện:

* Kết quả sau khi phân cụm với các số lượng cluster khác nhau:



K = 2 K =5



K = 10 K = 15



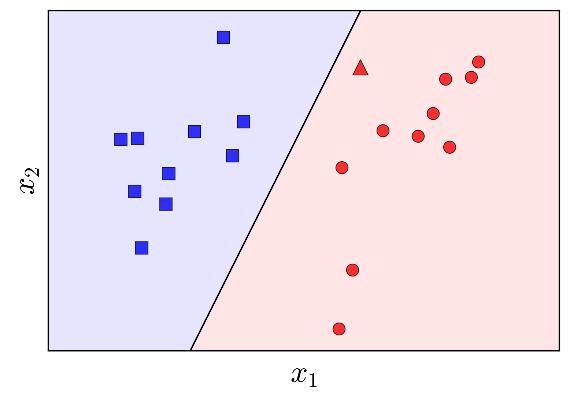
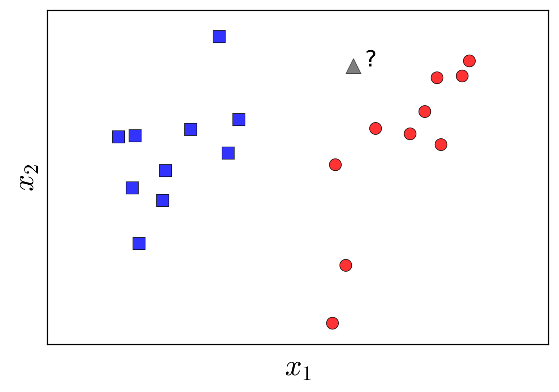
K = 20

1. **Perceptron Learning Algorithm**
2. Tìm hiểu thuật toán
   * + Bài toán Perceptron được phát biểu như sau: Cho hai class được gán nhãn, hãy tìm một đường phẳng sao cho toàn bộ các điểm thuộc class 1 nằm về 1 phía, toàn bộ các điểm thuộc class 2 nằm về phía còn lại của đường phẳng đó. Với giả định rằng tồn tại một đường phẳng như thế.

* Nếu tồn tại một đường phẳng phân chia hai class thì ta gọi hai class đó là linearly separable. Các thuật toán classification tạo ra các boundary là các đường phẳng được gọi chung là Linear Classifier.
  + - Cũng giống như các thuật toán lặp trong K-means Clustering và Gradient Descent, ý tưởng cơ bản của PLA là xuất phát từ một nghiệm dự đoán nào đó, qua mỗi vòng lặp, nghiệm sẽ được cập nhật tới một ví trí tốt hơn. Việc cập nhật này dựa trên việc giảm giá trị của một hàm mất mát nào đó.

1. Bài toán

Từ dữ liệu của hai tập được gán nhãn cho trước, hãy xây dựng một classifier (bộ phân lớp) để khi có một điểm dữ liệu hình tam giác màu xám mới, ta có thể dự đoán được màu (nhãn) của nó.



Hiểu theo một cách khác, chúng ta cần tìm lãnh thổ của mỗi class sao cho, với mỗi một điểm mới, ta chỉ cần xác định xem nó nằm vào lãnh thổ của class nào rồi quyết định nó thuộc class đó. Để tìm lãnh thổcủa mỗi class, chúng ta cần đi tìm biên giới (boundary) giữa hai lãnh thổ này. Vậy bài toán classification có thể coi là bài toán đi tìm boundary giữa các class. Và boundary đơn giản nhât trong không gian hai chiều là một đường thằng, trong không gian ba chiều là một mặt phẳng, trong không gian nhiều chiều là một siêu mặt phẳng (hyperplane) (tôi gọi chung những boundary này là đường phẳng). Những boundary phẳng này được coi là đơn giản vì nó có thể biểu diễn dưới dạng toán học bằng một hàm số đơn giản có dạng tuyến tính, tức linear. Tất nhiên, chúng ta đang giả sử rằng tồn tại một đường phẳng để có thể phân định lãnh thổ của hai class. Hình 1 bên phải minh họa một đường thẳng phân chia hai class trong mặt phẳng. Phần có nền màu xanh được coi là lãnh thổ của lớp xanh, phần có nên màu đỏ được coi là lãnh thổ của lớp đỏ. Trong trường hợp này, điểm dữ liệu mới hình tam giác được phân vào class đỏ.

1. Thực hiện:

* Chọn ngẫu nhiên một vector hệ số w với các phần tử gần 0.
* Duyệt ngẫu nhiên qua từng điểm dữ liệu x\_i :
* Nếu x\_iđược phân lớp đúng, tức sgn(w^T x\_(i))= y\_i, chúng ta không cần làm gì.
* Nếu x\_i bị misclassifed, cập nhật w theo công thức: w=w+y\_i x\_i w
* Kiểm tra xem có bao nhiêu điểm bị misclassifed. Nếu không còn điểm nào, dừng thuật toán. Nếu còn, quay lại bước 2.

1. **Gradient Descent**
2. Tìm hiểu thuật toán
   * + Trong Machine Learning nói riêng và Toán Tối Ưu nói chung, chúng ta thường xuyên phải tìm giá trị nhỏ nhất (hoặc đôi khi là lớn nhất) của một hàm số nào đó. Ví dụ như các hàm mất mát trong hai bài Linear Regression và K-means Clustering. Nhìn chung, việc tìm global minimum của các hàm mất mát trong Machine Learning là rất phức tạp, thậm chí là bất khả thi. Thay vào đó, người ta thường cố gắng tìm các điểm local minimum, và ở một mức độ nào đó, coi đó là nghiệm cần tìm của bài toán.
     + Các điểm local minimum là nghiệm của phương trình đạo hàm bằng 0. Nếu bằng một cách nào đó có thể tìm được toàn bộ (hữu hạn) các điểm cực tiểu, ta chỉ cần thay từng điểm local minimum đó vào hàm số rồi tìm điểm làm cho hàm có giá trị nhỏ nhất. Tuy nhiên, trong hầu hết các trường hợp, việc giải phương trình đạo hàm bằng 0 là bất khả thi. Nguyên nhân có thể đến từ sự phức tạp của dạng của đạo hàm, từ việc các điểm dữ liệu có số chiều lớn, hoặc từ việc có quá nhiều điểm dữ liệu.
     + Hướng tiếp cận phổ biến nhất là xuất phát từ một điểm mà chúng ta coi là gần với nghiệm của bài toán, sau đó dùng một phép toán lặp để tiến dần đến điểm cần tìm, tức đến khi đạo hàm gần với 0. Gradient Descent (viết gọn là GD) và các biến thể của nó là một trong những phương pháp được dùng nhiều nhất.
3. Bài toán:

Xét hàm số f(x)=x2+5sin(x) với đạo hàm f′(x)=2x+5cos(x) . Áp dụng thuật toán Gradient Desent với Learning Rate là 0.1 để tìm nghiệm. Với 2 điểm bắt đầu cho trước có x = -5 và x = 5.

1. Thực hiện:

Kết quả:



1. **Logistic Regression**
2. Tìm hiểu thuật toán
   * + Phương pháp hồi quy logistic là một mô hình hồi quy nhằm dự đoán giá trị đầu ra rời rạc (discrete target variable) y ứng với một véc-tơ đầu vào x. Việc này tương đương với chuyện phân loại các đầu vào x vào các nhóm y tương ứng.
     + Ví dụ, xem một bức ảnh có chứa một con mèo hay không. Thì ở đây ta coi đầu ra y=1 nếu bước ảnh có một con mèo và y=0 nếu bức ảnh không có con mèo nào. Đầu vào x ở đây sẽ là các pixel một bức ảnh đầu vào.
3. Bài toán:

Một nhóm 20 sinh viên dành thời gian trong khoảng từ 0 đến 6 giờ cho việc ôn thi. Thời gian ôn thi này ảnh hưởng đến xác suất sinh viên vượt qua kỳ thi như thế nào?

Kết quả thu được:

| **Hours** | **Pass** | **Hours** | **Pass** |
| --- | --- | --- | --- |
| .5 | 0 | 2.75 | 1 |
| .75 | 0 | 3 | 0 |
| 1 | 0 | 3.25 | 1 |
| 1.25 | 0 | 3.5 | 0 |
| 1.5 | 0 | 4 | 1 |
| 1.75 | 0 | 4.25 | 1 |
| 1.75 | 1 | 4.5 | 1 |
| 2 | 0 | 4.75 | 1 |
| 2.25 | 1 | 5 | 1 |
| 2.5 | 0 | 5.5 | 1 |

Do kết quả chỉ có pass và không pass nên ta sử dụng hàm sigmoid để thể hiện đồ thị.

Dựa theo thuật toán Logistic regression thì chúng ta hãy tìm các hệ số của 1 hàm sigmoid để thể hiện được bản khảo sát trên 1 cách rõ ràng nhất.

1. Thực hiện:

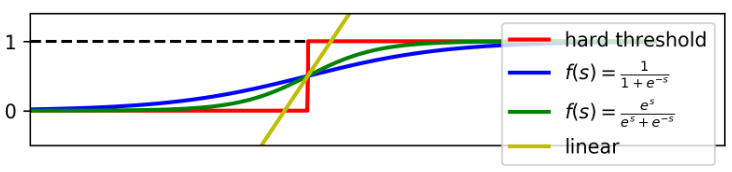
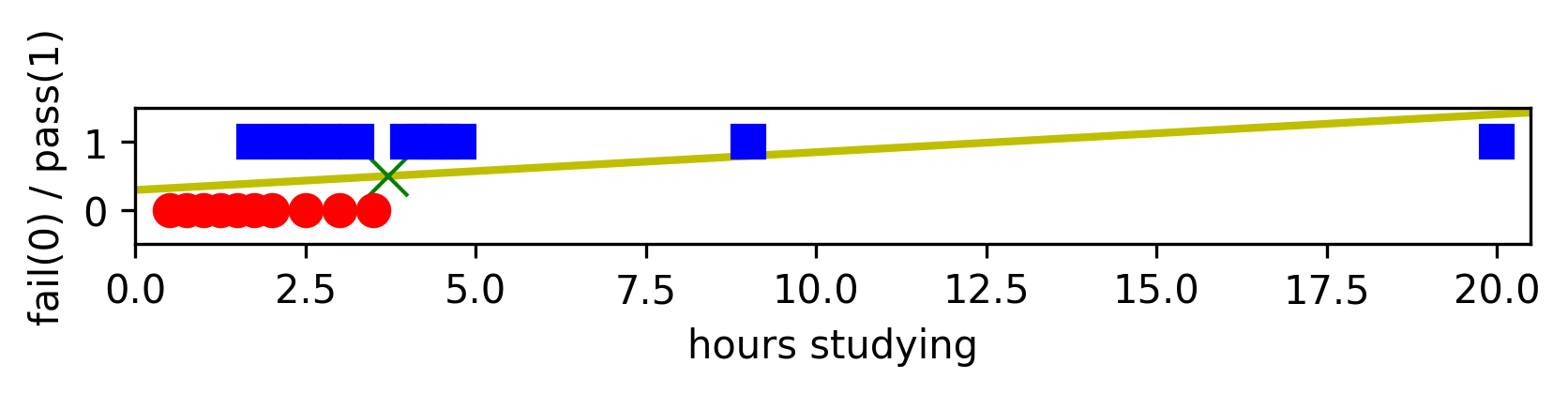
* Đầu ra dự đoán của:

Linear Regression: f(x)= w^T x

PLA: f(x)=sgn (w^T x)

* Đầu ra dự đoán của logistic regression thường được viết chung dưới dạng: f(x)=θ (w^T x)

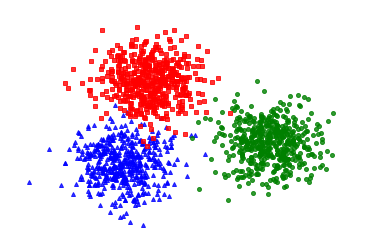
Trong đó θ được gọi là logistic function. Một số activation cho mô hình tuyến tính được cho trong hình dưới đây:



* Đường màu vàng (thuật toán LR) và mà đỏ (PLA) không phù hợp với bài toán.
* Đường màu xanh lục và xanh lam phù hợp với bài toán của chúng ta hơn.

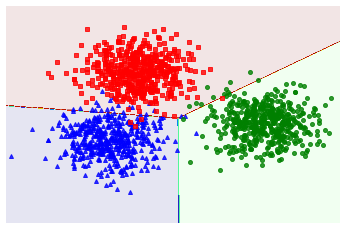
1. **Softmax Regression**
2. Bài toán:

Tạo một bộ dữ liệu với 3 nhóm, mỗi nhóm có 500 điểm dữ liệu. Sử dụng thuật toán Softmax Regression để tìm 3 đường thẳng nhằm chia bộ dữ liệu thành 3 vùng với mỗi vùng là một nhóm. Khi tạo dữ liệu thì có vài điểm của nhóm này sẽ xuất hiện ở trong nhóm khác. Trong ví dụ này, ta tìm các đường thẳng chia các nhóm, sao cho hợp lí nhất và sai số thấp nhất có thể.



1. Thực hiện:

Bộ dữ liệu sau khi chia:



1. **Support Vector Machine**
2. Tìm hiểu thuật toán

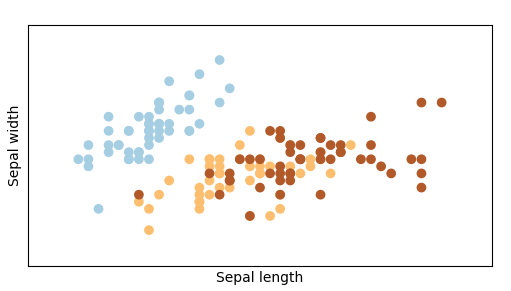
Support vector machines (SVM) là một thuật toán phân loại có giám sát nổi tiếng tạo ra một đường phân chia giữa các loại dữ liệu khác nhau. Cách tính véc tơ này, hiểu theo cách đơn giản nhất, bằng cách tối ưu hóa để điểm gần nhất trong mỗi nhóm sẽ cách nhau xa. Nói cách khác, bài toán đi tìm đường phân chia sao cho margin là lớn nhất. Việc margin rộng hơn sẽ mang lại hiệu ứng phân lớp tốt hơn vì sự phân chia giữa hai classes là rạch ròi hơn.

1. Bài toán:
   * Sử dụng SVM trong bài toán phân loại hoa.
   * Mô tả bài toán: Ta sẽ xây dựng một mô hình (model) sao cho khi input vào là một bông hoa và ouput trả ra kết quả nó là loài hoa gì.
   * Tập dữ liệu Iris Flowers:

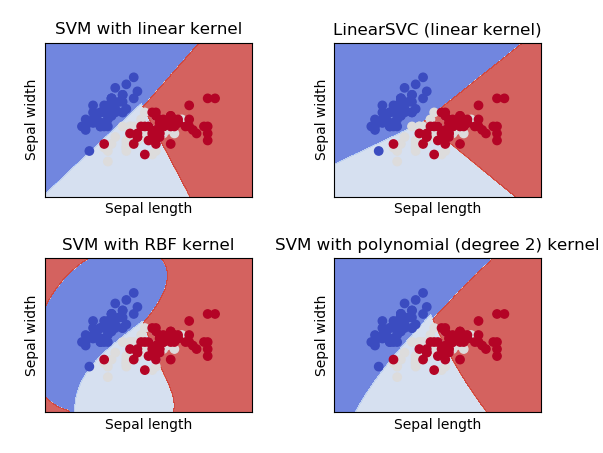
* Tập dữ liệu này gồm 50 mẫu về 3 loài hoa khác nhau của họ Iris là (Iris setosa, Iris virginica và Iris versicolor)
* Với mỗi một mẫu hoa này tập dữ liệu thu thập bốn thuộc tính là chiều dài và chiều rộng của đài hoa và cánh hoa với đơn vị centimet. Để có thể sử dụng tập dữ liệu này chúng ta sẽ sử dụng thư viện datasets trong sklearn.

1. Thực hiện:

* Biểu diễn tập dữ liệu bằng đồ thị 2D: Chúng ta tưởng tượng tập dữ liệu của ta là một tập hợp của 150 điểm dữ liệu tương ứng với 150 bông hoa..
* Lúc này đã có một tập hợp các điểm dữ liệu. Mỗi điểm dữ liệu bao gồm 4 thuộc tính như đã nói ở trên. Tuy nhiên để biểu diễn trong đồ thị hai chiều chúng ta cần phải giảm bớt số thuộc tính biểu diễn. Ở đây giả sử chọn hai thuộc tính đầu tiên là độ rộng và chiều cao của đài hoa.
* Kết quả:



* Phân lớp sử dụng SVM với các Kernel khác nhau:
* Với tập dữ liệu Iris chúng ta cần phân loại các bông hoa thành 3 lớp dữ liệu. Sử dụng SVM với các phương pháp khác nhau sẽ cho hiệu quả phân lớp khác nhau. Cũng tương tự như trên, chúng ta chỉ xem xét đến 2 thuộc tính đầu tiên của tập dữ liệu, tức là phân lớp trong không gian 2 chiều.
* Kết quả:



1. **Decision Tree**
2. Tìm hiểu thuật toán

* Cây quyết định (Decision Tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật (series of rules). Khi cho dữ liệu về các đối tượng gồm các thuộc tính cùng với lớp (classes) của nó, cây quyết định sẽ sinh ra các luật để dự đoán lớp của các đối tượng chưa biết (unseen data).
* Decision Trees gồm 3 phần chính: 1 node gốc (root node), những node lá (leaf nodes) và các nhánh của nó (branches). Node gốc là điểm bắt đầu của cây quyết định và cả hai node gốc và node chứa câu hỏi hoặc tiêu chí để được trả lời. Nhánh biểu diễn các kết quả của kiểm tra trên nút.
* Ví dụ câu hỏi ở node đầu tiên yêu cầu câu trả lời là “yes” hoặc là “no” thì sẽ có 1 node con chịu trách nhiệm cho phản hồi là “yes”, 1 node là “no”.
* Trong tài liệu này, em sẽ tìm hiểu về một thuật toán xây dựng decision tree ra đời từ rất sớm và rất phổ biến: Iterative Dichotomiser 3 (ID3). ID3 là một thuật toán decision tree được áp dụng cho các bài toán classification mà tất cả các thuộc tính đều ở dạng categorical. Trong ID3, chúng ta cần xác định thứ tự của thuộc tính cần được xem xét tại mỗi bước.

1. Bài toán:

Bảng dữ liệu này mô tả mối quan hệ giữa thời tiết trong 14 ngày (bốn cột đầu, không tính cột id) và việc một đội bóng có chơi bóng hay không (cột cuối cùng). Nói cách khác, ta phải dự đoán giá trị ở cột cuối cùng nếu biết giá trị của bốn cột còn lại.

| **id** | **outlook** | **temperature** | **humidity** | **wind** | **play** |
| --- | --- | --- | --- | --- | --- |
| 1 | sunny | hot | high | weak | no |
| 2 | sunny | hot | high | strong | no |
| 3 | overcast | hot | high | weak | yes |
| 4 | rainy | mild | high | weak | yes |
| 5 | rainy | cool | normal | weak | yes |
| 6 | rainy | cool | normal | strong | no |
| 7 | overcast | cool | normal | strong | yes |
| 8 | sunny | mild | high | weak | no |
| 9 | sunny | cool | normal | weak | yes |
| 10 | rainy | mild | normal | weak | yes |
| 11 | sunny | mild | normal | strong | yes |
| 12 | overcast | mild | high | strong | yes |
| 13 | overcast | hot | normal | weak | yes |
| 14 | rainy | mild | high | strong | no |

Có bốn thuộc tính thời tiết:

* *Outlook* nhận một trong ba giá trị: sunny, overcast, rainy.
* *Temperature* nhận một trong ba giá trị: hot, cool, mild.
* *Humidity* nhận một trong hai giá trị: high, normal.
* *Wind* nhận một trong hai giá trị: weak, strong.

1. Thực hiện:

Kết quả:

