BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**A picture containing logo

Description automatically generated**

**BÁO CÁO THÍ NGHIỆM/ THỰC NGHIỆM**

**HỌC PHẦN: TRÍ TUỆ NHÂN TẠO**

**ĐỀ TÀI**

**DỰ BÁO THỜI TIẾT BẰNG THUẬT TOÁN ID3**

Sinh viên thực hiện: Ngô Thế Duy – 2021605852

Đỗ Tiến Đạt – 2021606218

Nguyễn Đức Minh Quí – 2021604830

Lớp, Khóa: 20231IT6043008, K16

Nhóm: 17

Người hướng dẫn: Ths. Nguyễn Lan Anh

**Hà Nội, 12/2023**

# MỤC LỤC

[**MỤC LỤC** 2](#_gjdgxs)

[**LỜI CẢM ƠN** 4](#_30j0zll)

[**MỞ ĐẦU** 5](#_tyjcwt)

[1. Lý do chọn đề tài 5](#_3dy6vkm)

[2. Mục đích 5](#_1t3h5sf)

[3. Ý tưởng sơ bộ 5](#_4d34og8)

[4. Cấu trúc tài liệu báo cáo 5](#_2s8eyo1)

[**Chương 1. Tổng quan về Trí tuệ nhân tạo**](#_17dp8vu) 6

[1.1 Khái niệm 6](#_3rdcrjn)

[1.2 Lịch sử phát triển 7](#_lnxbz9)

[1.3 Ứng dụng 11](#_35nkun2)

[1.4 Các phương pháp và kỹ thuật TTNT cơ bản 12](#_1ksv4uv)

[1.5 Các thành phần của hệ thống TTNT 13](#_44sinio)

[**Chương 2. Một số thuật toán xây dựng cây quyết định**](#_2jxsxqh) 15

[2.1. Thuật toán ID3 15](#_z337ya)

[2.1.1. Khái niệm 15](#_3j2qqm3)

[2.1.2. Hàm số Entropy 15](#_4i7ojhp)

[2.1.3. Ưu/Nhược 16](#_2xcytpi)

[2.2. Thuật toán C4.5 16](#_2bn6wsx)

[2.2.1 Khái niệm 16](#_qsh70q)

[2.2.2 Các bước tiến hành 17](#_3as4poj)

[2.2.3 C4.5 dùng Gain-entropy làm độ đo lựa chọn thuộc tính “tốt nhất”. 18](#_1pxezwc)

[2.3. Thuật toán CART 19](#_49x2ik5)

[2.3.1. Khái niệm 19](#_2p2csry)

[2.3.2.](#_147n2zr) Đặc điểm 20

[2.3.3.](#_3o7alnk) Biểu diễn của mô hình CART: 21

[2.3.4.](#_23ckvvd) Huấn luyện mô hình CART từ dữ liệu 21

[2.3.5.](#_ihv636) Phân chia tham lam (Greedy splitting) 21

[2.4.](#_32hioqz) Thuật toán CHAID 23

[2.4.1.](#_1hmsyys) Khái niệm 23

[2.4.2.](#_41mghml) Các bước thực hiện 23

[2.5.](#_2grqrue) Thuật toán Conditional Inference Trees 24

[2.5.1.](#_vx1227) Khái niệm 24

[2.5.2.](#_3fwokq0) Đặc điểm 25

[2.6.](#_1v1yuxt) Thuật toán Random Forest 26

[2.6.1.](#_4f1mdlm) Khái niệm 26

[2.6.2.](#_2u6wntf) Đặc điểm 26

[2.6.3.](#_19c6y18) Các bước xây dựng thuật toán 28

[**Chương 3. Bài toán, cài đặt chương trình**](#_28h4qwu) 30

[3.1. Bài toán 30](#_nmf14n)

[3.1.1. Đặt vấn đề 30](#_37m2jsg)

[3.1.2. Giải quyết bài toán 31](#_1mrcu09)

[3.1.3. Kết quả 36](#_46r0co2)

[3.2. Cài đặt chương trình. 36](#_2lwamvv)

[3.2.1. Import thư viện 36](#_111kx3o)

[3.2.2. Đọc dữ liệu và tách nhãn 37](#_3l18frh)

[3.2.3. Tạo Node 37](#_206ipza)

[3.2.4. Tính Entropy 38](#_4k668n3)

[3.2.5. Tính gain từng thuộc tính 39](#_2zbgiuw)

[3.2.6. Tạo cây quyết định 39](#_1egqt2p)

[3.2.7. Kiểm tra dữ liệu 40](#_3ygebqi)

[3.2.8. Kết quả 40](#_sqyw64)

[**KẾT LUẬN VÀ KIẾN NGHỊ** 42](#_1rvwp1q)

[**TÀI LIỆU THAM KHẢO** 43](#_2r0uhxc)

# 

# LỜI CẢM ƠN

*Lời đầu tiên chúng em xin gửi lời cảm ơn đến cô giáo Nguyễn Lan Anh vì đã hỗ trợ chứng em trong quá trình hoàn thiện báo cáo thực nghiệm này, cảm ơn cô Nguyễn Lan Anh đã hỗ trợ chúng em về về mặt chuyên môn cũng như kiến thức cần thiết để chứng em có thể hoàn thiện báo cáo cũng như sản phẩm thực nghiệm của mình.*

*Nhóm chúng em thực hiện đề tài “****DỰ BÁO THỜI TIẾT BẰNG THUẬT TOÁN ID3****” nhằm mục đích xây dựng một trương trình để giải quyết quá trình dự báo thời tiết. Trong quá trình thực hiện đề tài sẽ khó tránh khỏi những sai sót trong việc kiến thức chưa được đầy đủ, nhóm chúng em mong rằng sẽ nhận được sự góp ý và đánh giá của cô.*

## *Chúng em xin chân thành cảm ơn!*

# MỞ ĐẦU

## **Lý do chọn đề tài**

Thời tiết ảnh hưởng rất nhiều đến những sinh hoạt đời sống của con người trên thế giới. Nếu có thể dự đoán được thời tiết thì con người sẽ có thể chủ động được trong công việc, cuộc sống. Để có thể dự báo thời tiết, ở bài báo cáo này chúng em đưa ra ý tưởng sử dụng thuật toán ID3 với dữ liệu có sẵn tạo ra một cây quyết định để dự báo thời tiết.

## **Mục đích**

Mục đích đề tài : Bằng vào kiến thức đã học và được hướng dẫn để tạo ra một sản phẩm dự báo thời tiết bằng ngôn ngữ python.

## **Ý tưởng sơ bộ**

Giải thuật ID3 là một thuật toán cây quyết định(decision tree) được áp dụng cho các bài toán phân lớp (classification) mà tất cả các thuộc tính ở dạng categorical. Trong ID3, chúng ta cần xác định thứ tự của thuộc tính cần được xem xét tại mỗi bước. Với các bài toán có nhiều thuộc tính và mỗi thuộc tính có nhiều giá trị khác nhau, việc tìm được nghiệm tối ưu thường là không khả thi. Thay vào đó, một phương pháp đơn giản thường được sử dụng là tại mỗi bước, một thuộc tính tốt nhất sẽ được chọn ra dựa trên một tiêu chuẩn nào đó. Và trong bài báo cáo này chúng em chỉ đề cập tới hàm số entropy để đánh giá chính xác độ hỗn tạp thông tin thu được. Với mỗi thuộc tính được chọn, ta chia dữ liệu vào các child node tương ứng với các giá trị của thuộc tính đó rồi tiếp tục áp dụng phương pháp này cho mỗi child node. Việc chọn ra thuộc tính tốt nhất ở mỗi bước như thế này được gọi là cách chọn greedy (tham lam). Cách chọn này có thể không phải là tối ưu, những cách làm này khiến cho bài toán cần giải quyết trở nên đơn giản hơn.

## **Cấu trúc tài liệu báo cáo**

Chương 1 : Tổng quan về Trí tuệ nhân tạo

Chương 2 : Một số thuật toán xây dựng cây quyết định

Chương 3 : Bài toán, cài đặt chương trình

# Tổng quan về Trí tuệ nhân tạo

## **Khái niệm**

**Trí tuệ nhân tạo là gì ?**



Trí tuệ nhân tạo hay trí thông minh nhân tạo (Artificial intelligence – viết tắt là AI) là một ngành thuộc lĩnh vực khoa học máy tính (Computer science). Là trí tuệ do con người lập trình tạo nên với mục tiêu giúp máy tính có thể tự động hóa các hành vi thông minh như con người.

Trí tuệ nhân tạo khác với việc lập trình logic trong các ngôn ngữ lập trình là ở việc ứng dụng các hệ thống học máy (machine learning) để mô phỏng trí tuệ của con người trong các xử lý mà con người làm tốt hơn máy tính.

Trí tuệ nhân tạo giúp máy tính có được những trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ, tiếng nói, biết học và tự thích nghi,…

Tuy rằng trí thông minh nhân tạo có nghĩa rộng như là trí thông minh trong các tác phẩm khoa học viễn tưởng, nó là một trong những ngành trọng yếu của tin học. Trí thông minh nhân tạo liên quan đến cách cư xử, sự học hỏi và khả năng thích ứng thông minh của máy móc.

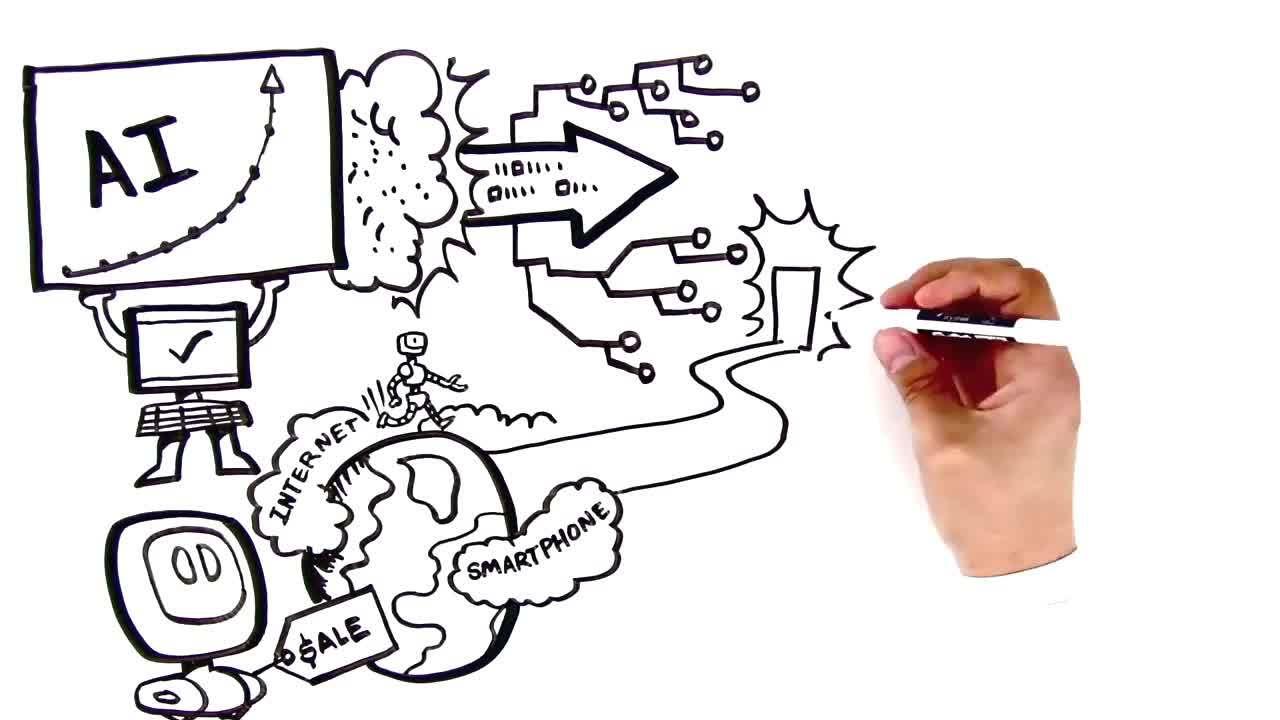
Trí tuệ nhân tạo được phân thành 3 loại hệ thống khác nhau:

* Trí tuệ nhân tạo phân tích
* Trí tuệ nhân tạo lấy cảm hứng từ con người
* Nhân tạo

Có 4 loại công nghệ AI:

* Công nghệ AI phản ứng
* Công nghệ AI và bộ nhớ hạn chế
* Lý thuyết trí tuệ nhân tạo
* Tự nhận thức

## **Lịch sử phát triển**



Năm 1920, nhà viết kịch người Czech có tên Karel Capek cho ra đời một vở kịch có chủ đề khoa học viễn tưởng mang tên Rossum's Universal Robot (RUR), với nội dung xoay quanh một nhà máy sản xuất ra những con người nhân tạo, mang tên là robot.

Năm 1941, máy tính điện tử đầu tiên ra đời.

Năm 1943, Warren McCulloch và Walter Pitts xuất bản cuốn “A Logical Calculus of Ideas Immanent in Nervous Activity”, dịch ra là “ Một tính toán Logic của những ý tưởng tiềm ẩn trong hoạt động thần kinh”. Bài viết đề xuất mô hình toán học đầu tiên để xây dựng một mạng lưới thần kinh.

Năm 1949, máy tính thương mại đầu tiên ra đời.

Năm 1950:

* Alan Turing xuất bản “Computing Machinery and Intelligence” – “Máy tính và trí thông minh”, đề xuất phép thử Turing – một phương pháp để xác định xem một máy tính có thông minh hay không.
* Đại học Harvard Marvin Minsky và Dean Edmonds xây dựng SNAGS máy tính mạng thần kinh đầu tiên.
* Claude Shannon xuất bản bài báo “Lập trình máy tính để chơi cờ”.
* Isaac Asimov xuất bản “Ba định luật về robot”.

Năm 1952, Arthur Samuel phát triển một chương trình tự học để chơi cờ.

Năm 1954, Thí nghiệm dịch máy Georgetown-IBM tự động dịch 60 câu tiếng Nga được chọn cẩn thận sang tiếng Anh.

Năm 1956, Trí tuệ nhân tạo ra đời.

Năm 1958, John McCarthy phát triển ngôn ngữ lập trình AI Lisp và xuất bản bài báo “Programs with Common Sense”. Bài viết đã đề xuất nhà tư vấn giả thuyết, một hệ thống AI hoàn chỉnh với khả năng học hỏi kinh nghiệm hiệu quả như con người.

Năm 1959:

* Allen Newell, Herbert Simon và JC Shaw giải quyết vấn đề chung (GPS), một chương trình được thiết kế để bắt chước giải quyết vấn đề của con người.
* Herbert Gelernter phát triển chương trình Định lý hình học.
* Arthur Samuel đồng xu với thuật ngữ học máy khi con IBM.
* John McCarthy và Marvin Minsky đã tìm thấy Dự án Trí tuệ nhân tạo MIT.

Năm 1963, John McCarthy bắt đầu Phòng thí nghiệm AI ở Stanford.

Năm 1968, SHRDLU – trình máy tính hiểu ngôn ngữ tự nhiên ban đầu được phát triển bởi Terry Winograd tại MIT.

Năm 1969, một chương trình XX và MYCIN được thiết kế để chẩn đoán nhiễm trùng máu, được tạo ra tạo Stanford.

Năm 1970, hệ thống chuyên gia đầu tiên xuất hiện.

Năm 1972, ngôn ngữ lập trình logic PRITAL được tạo ra.

Năm 1973-1980, “Báo cáo Lighthill” cùng với liên tiếp thất vọng trong phát triển của AI dẫn đến việc cắt giảm nghiêm trọng tài trợ trong các dự án trí tuệ nhân tạo và trợ cấp học thuật. Thời kỳ này được gọi là “Mùa đông AI đầu tiên”.

Năm 1980, hệ thống chuyên gia thương mại thành công lần đầu tiên, khởi đầu cho sự bùng nổ đầu tư các hệ thống chuyên gia, kết thúc hiệu quả “ Mùa đông AI đầu tiên”.

Năm 1982, Bộ Thương mại Quốc tế và Công nghiệp Nhật Bản khởi động dự án Hệ thống máy tính thế hệ thứ năm với mục tiêu phát triển hiệu năng giống như siêu máy tính và nền tảng để phát triển trí tuệ nhân tạo AI.

Năm 1983, Chính phủ Hoa Kì khởi động “Sáng kiến điện toán chiến lược” nhằm cung cấp nghiên cứu được tài trợ bởi DARPA trong điện toán tiên tiến và trí tuệ nhân tạo.

Năm 1985, các công ty chi một số tiền rất lớn một năm cho các hệ thống chuyên gia và toàn bộ ngành công nghiệp được gọi là thị trường máy Lisp để hỗ trợ họ.

Năm 1986, tìm ra ứng dụng mạng nơron và giải thuật di truyền.

Năm 1987, Công nghệ điện toán đám mây được cải thiện => xuất hiện nhiều lựa chọn thay thế rẻ hơn => thị trường máy Lisp sụp đổ, mở ra “Mùa đông AI thứ hai”.

Năm 1993, không đạt được kỳ vọng như đã tính toán khi đã chi gần 1 tỷ $, DARPA kết thúc “Sáng kiến Điện toán chiến lược”.

Năm 1995, phát triển sự sống nhân tạo, AI phân tán.

Năm 1997, RoboCup tên đầy đủ là Robot Soccer World Cup, là giải thi đấu bóng đá dành cho robot, được tổ chức lần đầu tiên tại Nhật Bản với 38 đội tham gia từ 11 nước.

Năm 2000, Khai phá dữ liệu, học thống kê, Web ngữ nghĩa, tin sinh học, mạng xã hội. ASIMO – người máy có thể di chuyển bằng hai chân như người do “Trung tâm Nghiên cứu Kỹ thuật Cơ bản Waco” của tập đoàn Honda (Nhật Bản) chế tạo.

Năm 2005, Chiếc xe tự lái STANLEY chiến thắng cuộc thi DARPA Grand Challenge. Quân đội Hoa Kỳ bắt đầu đầu tư vào các robot tự hành như “Big Dog” của Boston Dynamic và “PackBot” của iRobot.

Năm 2008, Google tạo ra những bước đột phá trong nhận dạng giọng nói và giới thiệu tinh năng này trong ứng dụng iPhone.

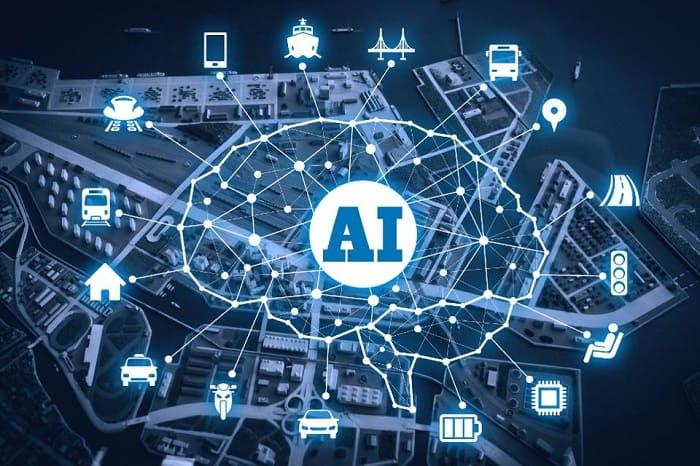
Năm 2011, Watson của IBM tuyên bố cạnh tranh về Jeopardy.

Năm 2012, Andrew Ng người sáng lập dự án Google Brain Deep Learning, cung cấp một mạng lưới thần kinh bằng cách sử dụng thuật toán 10 triệu video YouTube dưới dạng tập huấn luyện. Mạng lưới thần kinh đã học cách nhận ra một con mèo mà không được cho biết con mèo là gì.

Năm 2014, Google tạo ra chiếc xe tự lái đầu tiên để vượt qua bài kiểm tra lái xe của nhà nước.

Năm 2016, AlphaGo của Google DeepMind đánh bại nhà vô địch cờ vây Lee Sedol.

## **Ứng dụng**



Các bệnh viện đang áp dụng máy để chẩn đoán tốt hơn và nhanh hơn con người. Một trong những biết công nghệ tốt nhất chăm sóc sức khỏe là IBM Watson. Nó có thể hiểu ngôn ngữ tự nhiên và có khả năng trả lời các câu hỏi. Hệ thống khai thác dữ liệu bệnh nhân và các nguồn dữ liệu có sẵn khác để tạo thành một giả thuyết, sau đó đưa ra một lược đồ chấm điểm tin cậy.

Chatbot chương trình máy tính được sử dụng trực tuyến để trả lời các câu hỏi và hỗ trợ khách hàng, giúp sắp xếp các cuộc hẹn theo dõi hoặc hỗ trợ bệnh nhân thông qua quy trình thanh toán và trợ lý sức khỏe ảo cung cấp phản hồi y tế cơ bản.

AI trong kinh doanh:

* Tự động hóa quá trình robot đang được áp dụng cho các công việc có tính chất lặp đi lặp lại.
* Các thuật toán được tích hợp vào các nền tảng phân tích và CRM để khám phá thông tin về cách phục vụ khách hàng tốt hơn.
* Chatbots đã được kết hợp vào các trang web để cung cấp dịch vụ ngay lập tức cho khách hàng.

AI trong giáo dục:

* AI có thể tự động hóa việc chấm điểm, giúp các giáo viên có thêm thời gian.
* AI có thể đánh giá năng lực và quản lý sinh viên.
* Gia sư AI có thể cung cấp hỗ trợ cho sinh viên, đảm bảo họ hoàn thành  
  đúng theo giáo trình từ trước.
* AI còn có thể thay thế một số giáo viên.

AI trong tài chính: AI được ứng dụng trong lĩnh vực tài chính, các ứng dụng như thu thập dữ liệu cá nhân và cung cấp tư vấn tài chính.

AI trong pháp luật: Tự động hóa trong quá trình khám phá, sàng lọc thông tin tài liệu trong pháp luật giúp con người sử dụng thời gian hiệu quả hơn.

AI trong sản xuất: Robot công nghiệp được sử dụng để thực hiện các nhiệm vụ đơn lẻ và giải phóng sức lao động của con người.

## **Các phương pháp và kỹ thuật TTNT cơ bản**

Các kỹ thuật Trí tuệ nhân tạo cơ bản bao gồm:

Lý thuyết giải bài toán và suy diễn thông minh**:** cho phép viết các chương trình giải câu đố, các trò chơi thông qua các suy luận mang tính người.

Lý thuyết tìm kiếm may rủi**:** gồm các phương pháp và kỹ thuật tìm kiếm với sự hỗ trợ của thông tin phụ để giải bài toán một cách có hiệu quả.

Các ngôn ngữ về TTNT**:** Các ngôn ngữ chuyên dụng này cho phép lưu trữ, xử lý thông tin ký hiệu, tri thức. VD: LISP, PROLOG, ...

Lý thuyết thể hiện tri thức và hệ chuyên gia**:** TTNT là khoa học về thể hiện và sử dụng tri thức. Mạng ngữ nghĩa, logic vị từ, Frame…

Lý thuyết nhận dạng và xử lý tiếng nói**:** Ứng dụng của phương pháp này trong việc nhận dạng chữ viết, âm thanh…

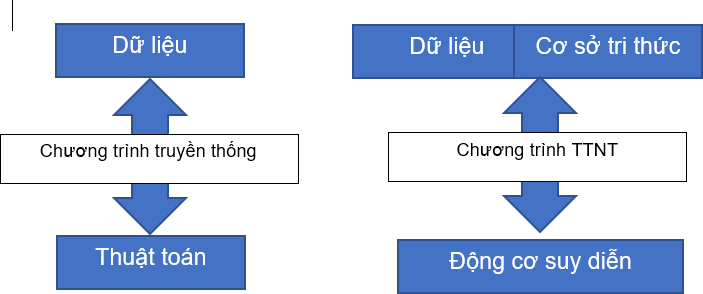
Người máy**:** là sự kết hợp giữa cơ học và TTNT để tạo ra những người máy có khả năng suy nghĩ, hoạt động giống con người.

Tâm lý học xử lý thông tin**:** Các kết quả nghiên cứu của tâm lý học giúp TTNT xây dựng các cơ chế trả lời theo hành vi, có ý thức; nó giúp cho việc thực hiện các suy diễn mang tính người.

## **Các thành phần của hệ thống TTNT**

Hai thành phần cơ bản của hệ thống TTNT:

* Các phương pháp biểu diễn vấn đề, biểu diễn tri thức
* Các phương pháp tìm kiếm trong không gian bài toán, các chiến lược suy diễn



*Hình 1.5.1. So sánh chương trình truyền thống và chương trình AI*

**Tri thức và cơ sở tri thức:**

Tri thức (Knowledge) là sự hiểu biết bằng lý thuyết hay thực tế về một chủ đề hay lĩnh vực. Tri thức là tổng của những cái đang biết hiện nay; tri thức là sức mạnh. Những người có tri thức tốt là những chuyên gia (expert)

Cơ sở tri thức (Knowledge Base) là tập hợp các tri thức liên quan đến vấn đề mà chương trình quan tâm giải quyết. Cơ sở tri thức chứa các kiến thức được sử dụng để giải quyết các vấn đề (bài toán) trong trí tuệ nhân tạo.

**Biểu diễn tri thức:**

Tri thức của bài toán có thể chia làm 3 loại: tri thức mô tả; tri thức thủ tục và tri thức điều khiển.

Một số phương pháp biểu diễn tri thức:

* Phương pháp biểu diễn nhờ luật.
* Phương pháp biểu diễn nhờ mạng ngữ nghĩa.
* Phương pháp biểu diễn nhờ bộ ba liên hợp OAV.
* Phương pháp biểu diễn nhờ Frame.
* Phương pháp biểu diễn nhờ logic vị từ.

# Một số thuật toán xây dựng cây quyết định

## **Thuật toán ID3**

### Khái niệm

ID3 hay Iterative Dichotomiser 3 được phát triển bởi Quinlan trong AI và Breiman, Friedman, Olsen và Stone trong thống kê. ID3 là một giải thuật học khá đơn giản nhưng lại đem lại rất nhiều thành công trong các lĩnh vực khác nhau.

ID3 là thuật toán nổi tiếng để xây dựng Decision Tree, áp dụng cho bài toán Phân loại (Classification) mà tất các các thuộc tính để ở dạng category.

Trong thuật toán ID3, các thuộc tính được đánh giá dựa trên Hàm số Entropy, hàm số phổ biến trong toán học xác suất.

### Hàm số Entropy

Để đánh giá chính xác thông tin thu được, dùng Entropy(S): Độ bất định (độ pha trộn/ độ hỗn tạp) của S liên quan đến sự phân loại đang xét

Công thức tính Entropy:

**Entropy(S)= pi log2 pi**

Trong đó pi là xác suất xuất hiện trạng thái i của hệ thống. Theo lý thuyết thông tin : mã có độ dài tối ưu là mã gán –log2p bits cho thông điệp có xác suất là p. S là một tập huấn luyện.

Quy định 0log0  = 0

***Chú ý:***

Đại lượng Entropy = 0 nếu tất cả thành viên của tập S cùng thuộc một lớp (vì nếu tất cả là dương (P+ = 1), do đó P- = 0, Entropy(S) = - 1log21 – 0log20 = 0).

Đại lượng Entropy(S) = 1 khi tập S chứa tỉ lệ tập mẫu âm và mẫu dương là như nhau. Nếu tập S chứa tập mẫu âm và tập mẫu dương có tỉ lệ P+ khác P- thì Entropy(S) (- (0,1).

Dựa trên sự xác định entropy, ta tính Gain(S, A) = Lượng giảm entropy mong đợi qua việc chia các ví dụ theo thuộc tính A

Công thức tính Gain:

**Gain(S,A) = Entropy(S) – VValues(A)  Entropy(SV)**

### Ưu/Nhược

1. Ưu điểm

* Cây quyết định dễ hiểu
* Không yêu cầu chuẩn hóa dữ liệu
* Không yêu cầu mở rộng quy mô dữ liệu
* Các giá trị bị thiếu trong dữ liệu không ảnh hưởng đến quá trình xây dựng cây quyết định ở bất kỳ mức độ nào

1. Nhược điểm

* Một thay đổi nhỏ trong dữ liệu có thể gây ra thay đổi lớn trong cấu trúc của cây quyết định
* Đôi khi việc tính toán có thể phức tạp hơn nhiều so với thuật toán khác
* Không thỏa đáng để áp dụng hồi quy và dự đoán giá trị liên tục

## **Thuật toán C4.5**

### Khái niệm

Với những đặc điểm C4.5 là thuật toán phân lớp dữ liệu dựa trên cây quyết định hiệu quả và phổ biến trong những ứng dụng khai phá cơ sở dữ liệu có kích thước nhỏ. C4.5 sử dụng cơ chế lưu trữ dữ liệu thường trú trong bộ nhớ, chính đặc điểm này làm C4.5 chỉ thích hợp với những cơ sở dữ liệu nhỏ, và cơ chế sắp xếp lại dữ liệu tại mỗi node trong quá trình phát triển cây quyết định. C4.5 còn chứa một kỹ thuật cho phép biểu diễn lại cây quyết định dưới dạng một danh sách sắp thứ tự các luật **if-then** (một dạng quy tắc phân lớp dễ hiểu). Kỹ thuật này cho phép làm giảm bớt kích thước tập luật và đơn giản hóa các luật mà độ chính xác so với nhánh tương ứng cây quyết định là tương đương.

### Các bước tiến hành

Tư tưởng phát triển cây quyết định của C4.5 là phương pháp Hunt đã nghiên cứu ở trên. Chiến lược phát triển theo độ sâu (depth-first strategy) được áp dụng cho C4.5.

Mã giả của thuật toán C4.5:

(1) ComputerClassFrequency(T);  
(2) if OneClass or FewCases  
 return a leaf;  
Create a decision node N;  
(3) ForEach Attribute A  
 ComputeGain(A);  
(4) N.test=AttributeWithBestGain;  
(5) if (N.test is continuous)  
 find Threshold;  
(6) ForEach T' in the splitting of T  
(7) If ( T' is Empty )  
 Child of N is a leaf  
else  
(8) Child of N=FormTree(T');  
(9) ComputeErrors of N;  
return N;

C4.5 có những đặc điểm khác với các thuật toán khác, đó là: cơ chế chọn thuộc tính để kiểm tra tại mỗi node, cơ chế xử lý với những giá trị thiếu, việc tránh “quá vừa” dữ liệu, ước lượng độ chính xác và cơ chế cắt tỉa cây.

### C4.5 dùng Gain-entropy làm độ đo lựa chọn thuộc tính “tốt nhất”.

Phần lớn các hệ thống đều cố gắng để tạo ra một cây càng nhỏ càng tốt, vì những cây nhỏ hơn thì dễ hiểu hơn và dễ đạt được độ chính xác dự đoán có hơn. Do không thể đảm bảo được sự cực tiểu của cây quyết định, C4.5 dựa vào nghiên cứu tối ưu hóa, và sự lựa chọn cách phân chia mà có độ đo lựa chọn thuộc tính đạt giá trị cực đại.  
 Hai độ đo được sử dụng trong C4.5 là **information gain** và **gain ratio**. RF(Cj,S) biểu diễn tần suất (Relative Frequency) các case trong S thuộc về lớp Cj.

**RF(Cj,S)=**

Với  là kích thước tập các case có giá trị phân lớp là Cj.  là kích thước tập dữ liệu đào tạo.  
 Chỉ số thông tin cần thiết cho sự phân lớp: I(S) với S là tập cần xét sự phân phối lớp được tính bằng:

**I(S) = - log(RF(Cj,S))**

Sau khi S được phân chia thành các tập con S1, S2,…, St bởi test B thì information gain được tính bằng:

**G(S,B) = I(S) - I(Si)**

Test B sẽ được chọn nếu có G(S,B) đạt giá trị lớn nhất.

Tuy nhiên có một vấn đề khi sử dụng G(S,B) ưu tiên test có số lượng lớn kết quả, ví dụ G(S,B) đạt cực đại với test mà từng Si chủi chứa một case đơn. Tiêu chuẩn gain ratio giải quyết được vấn đề này bằng việc đưa vào thông tin tiềm năng (potential information) của bản thân mỗi phân hoạch.

**P(S,B) = - log()**

Test B sẽ được chọn nếu có tỷ số giá trị gain ratio lớn nhất. ( G(S,B) /P(S,B) )Lớn nhất

Trong mô hình phân lớp C4.5 , có thể dùng một trong hai loại chỉ số Information Gain hay Gain ratio để xác định thuộc tính tốt nhất. Trong đó Gain ratio là lựa chọn mặc định.

## **Thuật toán CART**

### Khái niệm

Thuật toán CART (cây phân loại và hồi quy - Classification and Regression Tree), là một thuật toán được Leo Breiman giới thiệu năm 1984, dùng để xây dựng cây quyết định đa năng, với mục đích phân loại và hồi quy dữ liệu.

Thuật toán CART là nền tảng cho những thuật toán khác mạnh mẽ hơn như bagged decision trees (tập hợp nhiều cây quyết định để cùng đoán nhận), random forest (rừng ngẫu nhiên) và boosted decision trees (tập hợp các cây khác nhau, là các model; và một siêu model - metamodel - để tập hợp kết quả đoán nhận của các modem con)

### Đặc điểm

Dưới đây là một số đặc điểm chính của thuật toán C&R (CART):

1. Phân Loại và Dự Đoán:

CART có khả năng thực hiện cả nhiệm vụ phân loại và dự đoán giá trị số. Trong trường hợp phân loại, cây sẽ tạo ra các nhãn phân loại cho các quan sát, trong khi đối với dự đoán giá trị số, cây sẽ tạo ra giá trị dự đoán dựa trên thuộc tính của dữ liệu đầu vào.

1. Chia Nhánh Nhị Phân:

CART sử dụng phương pháp chia nhánh nhị phân, tức là mỗi nút trong cây có thể chia thành hai nhánh con. Quá trình chia nhánh này tiếp tục cho đến khi một điều kiện dừng được đáp ứng, ví dụ như đạt đến độ sâu tối đa, số lượng quan sát tối thiểu tại mỗi nút, hoặc một tiêu chí dừng khác.

1. Độ Đo Chất Lượng Chia Nhánh:

Để quyết định cách chia nhánh, CART sử dụng các độ đo như độ đo Gini cho bài toán phân loại và sự giảm tối thiểu của sai số bình phương cho bài toán dự đoán.

1. Cắt Tỉa Cây (Pruning):

Sau khi cây được xây dựng, quá trình cắt tỉa cây có thể được áp dụng để giảm thiểu overfitting và tối ưu hóa hiệu suất của mô hình.

1. Xử Lý Dữ Liệu Thiếu:

CART có khả năng xử lý dữ liệu thiếu thông tin, tức là nếu có giá trị thiếu tại một nút, thuật toán vẫn có thể quyết định cách chia nhánh dựa trên thông tin có sẵn.

CART là một trong những thuật toán cây quyết định phổ biến và được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm khoa học dữ liệu, học máy, và khai thác dữ liệu.

### Biểu diễn của mô hình CART:

Biểu diễn của mô hình CART chính là một cây nhị phân.

Cây nhị phân này cũng chính là cây nhị phân thường thấy trong môn cấu trúc dữ liệu và giải thuật, với mỗi nút là một biến độc lập riêng lẻ x (trong học máy, ta gọi đây là các đặc trưng), và điểm tách (hay là ngưỡng phân loại) của đặc trưng đó.

Nút lá của cây chứa biến đầu ra y, dùng để thực hiện đoán nhận.

### Huấn luyện mô hình CART từ dữ liệu

Huấn luyện mô hình CART yêu cầu ta phải lựa chọn các biến đầu vào và các ngưỡng phân loại tương ứng cho đến khi ta xây dựng được cây phù hợp. Thường quá trình xây cây sẽ dừng lại khi ta đủ điều kiện thoả mãn tiêu chí dừng. Ví dụ của tiêu chí dừng là "Số lượng bản ghi tối thiểu tương ứng với nút lá của cây".

### Phân chia tham lam (Greedy splitting)

Quá trình ta xây cây quyết định nhị phân là quá trình ta phân chia không gian dữ liệu vào (input space). Một cách tiếp cận vét cạn để chia không gian đó chính là chia đôi đệ quy (recursive binary splitting).

Chia đôi đệ quy là thuật toán heuristics, từng đặc trưng một được lựa chọn, từng giá trị ngưỡng được thử với đặc trưng. Các lựa chọn này được đánh giá thông qua hàm mất mát, và thông số tối ưu hàm mất mát được lựa chọn.

Với bài toán phân loại, ta sử dụng độ thuần khiết Gini. Nhắc lại, độ thuần khiết Gini được tính bằng cách lấy tổng xác suất p; của bản ghi i được chọn nhân với xác suất Epk =1 − Pi, tức xác suất khi phân loại sai bản ghi i. Độ thuần khiết đạt giá trị nhỏ nhất, tức bằng 0 khi tất cả các k+i

trường hợp tại điểm nút cuối đều rơi vào cùng một giá trị.

Độ thuần khiết Gini cũng là độ đo lý thuyết thông tin và tương ứng với entropy dạng Tsallis với hệ số biến dạng q = 2, trong Vật lý, công thức entropy này đại diện cho tính mất thông tin trong các hệ thống lượng tử mất cân bằng, tắt dần.

Khi có q → 1, công thức trở về dạng Boltzmann - Gibbs hoặc công thức entropy của Shannon. Với các điều kiện này, độ thuần khiết Gini có tác dụng như một biến thể của phép đo entropy cho các cây quyết định.

Để tính độ thuần khiết Gini cho một tập các điểm dữ liệu có J lớp, với i ∈ {1,2, ...,J}. và để pi là lượng bản ghi được đánh dấu với lớp i trong tập hợp, thì:

**Ig(G) = k)Pk) = Pi2) = - 2 = 1 - 2**

Hàm mất mát của CART được mô tả như sau:

**J(K,Tk) = Gleft + Gright**

Trong đó :

Gright/left : mô tả tính thuần khiết (hoặc vẩn đục) của tập con bên trái/phải

Mleft/right : là số lượng bản ghi có trong tập con trái/phải

Khi thành công tách bộ dữ liệu làm 2 phần, ta tiếp tục tách các tập con một cách đệ quy cho đến khi đạt được cây quyết định.

Thuật toán CART là một giải thuật tham lam, theo đó, thuật toán liên tục thử chọn để lấy được phương án tối ưu nhất. Những bài toán tìm cây tối ưu là một bài toán NP hoàn chỉnh, với độ phức tạp là O(exp(m)).

## **Thuật toán CHAID**

### Khái niệm

CHAID là một thuật toán khác để xây dựng cây quyết định, nhưng nó tập trung chủ yếu vào việc xây dựng cây để mô hình hóa mối quan hệ giữa biến phụ thuộc và biến độc lập trong trường hợp của biến phụ thuộc rời rạc (categorical). CHAID là viết tắt của "Chi-squared Automatic Interaction Detector."

### Các bước thực hiện

Dưới đây là các bước cơ bản của thuật toán CHAID:

1. Chọn Node Gốc:

Chọn biến độc lập mà cây sẽ bắt đầu.

1. Tạo Phân Nhánh:

Sử dụng các giá trị của biến độc lập để tạo các nhánh (branches) trong cây. Mỗi nhánh tương ứng với một giá trị của biến độc lập.

1. Kiểm Định Tính Chất Sự Tương Tác:

Sử dụng bài kiểm tra chi-square để kiểm định xem có sự tương tác nào đáng kể giữa biến độc lập và biến phụ thuộc hay không. Các giá trị của biến độc lập có thể được kết hợp lại nếu không có sự tương tác đáng kể.

1. Lặp Lại Quy Trình:

Quy trình trên được lặp lại cho mỗi nhánh con cho đến khi cây được xây dựng hoặc một điều kiện dừng được đáp ứng.

1. Cắt tỉa cây (Pruning):

Cây có thể được cắt tỉa để ngăn chặn quá trình mọc của cây và giảm nguy cơ overfitting.

Thuật toán này tập trung vào việc tạo ra các nhóm (hoặc nhánh) của biến độc lập có mối quan hệ tốt nhất với biến phụ thuộc. Mỗi bước xây dựng cây là một quá trình tối ưu hóa thông qua kiểm định chi-square.

Mặc dù CHAID thường được sử dụng cho biến phụ thuộc rời rạc, nhưng có một số phiên bản mở rộng của thuật toán có thể xử lý cả biến phụ thuộc liên tục.

## **Thuật toán Conditional Inference Trees**

### Khái niệm

Conditional Inference Trees (CIT) là một phương pháp xây dựng cây quyết định, đặc biệt được thiết kế để giảm thiểu các vấn đề liên quan đến overfitting và cải thiện tính tổng quát của mô hình. Phương pháp này được thiết kế để đảm bảo tính thống kê chặt chẽ và tránh các vấn đề liên quan đến multiple testing errors.

### Đặc điểm

Dưới đây là một số điểm quan trọng về Conditional Inference Trees:

1. Kiểm Định Điều Kiện (Conditional Inference):

CIT sử dụng các kiểm định thống kê điều kiện để xác định xem việc phân chia nào sẽ được thực hiện tại mỗi bước trong quá trình xây dựng cây. Điều này giúp giảm thiểu overfitting bằng cách chỉ chia dữ liệu khi có đủ bằng chứng thống kê rằng phân chia đó thực sự cần thiết.

1. Điều Chỉnh cho Multiple Testing Errors:

CIT điều chỉnh cho vấn đề lỗi kiểm định nhiều lần (multiple testing errors) bằng cách sử dụng các kiểm định điều kiện và áp dụng một sự điều chỉnh Bonferroni-like để giảm thiểu sai lầm thống kê.

1. Phương pháp Chia Nhánh:

Trong quá trình xây dựng cây, CIT sử dụng một tiêu chí phân chia dựa trên kiểm định điều kiện, thay vì sử dụng các tiêu chí như Gini index hay chi-squared như trong các thuật toán khác.

1. Điều Chỉnh Độ Sâu Cây (Pruning):

CIT thường áp dụng các kỹ thuật cắt tỉa cây (pruning) để giảm thiểu overfitting và tối ưu hóa hiệu suất của mô hình.

1. Loại Dữ Liệu:

CIT có thể được sử dụng với cả biến phụ thuộc rời rạc và biến phụ thuộc liên tục.

Conditional Inference Trees thường được ưa chuộng trong các tình huống khi dữ liệu có tính chất phức tạp và muốn đảm bảo tính chắc chắn thống kê trong quá trình xây dựng cây quyết định.

## **Thuật toán Random Forest**

### Khái niệm

Random Forest là một mô hình máy học kết hợp nhiều cây quyết định để tăng cường hiệu suất và giảm nguy cơ overfitting. Dưới đây là một mô tả chi tiết về thuật toán Random Forest

### Đặc điểm

Thuật toán Random Forest có nhiều đặc điểm đáng chú ý, giúp nó trở thành một trong những mô hình phổ biến và hiệu quả trong máy học. Dưới đây là những đặc điểm quan trọng của Random Forest:

* Kết Hợp Nhiều Cây:

Random Forest là một mô hình kết hợp (ensemble) gồm nhiều cây quyết định. Mỗi cây được xây dựng độc lập từ một phần nhỏ của dữ liệu.

* Ngẫu Nhiên Mẫu:

Mỗi cây trong Random Forest được xây dựng bằng cách sử dụng một tập con ngẫu nhiên từ tập dữ liệu đào tạo. Việc này tạo ra sự đa dạng giữa các cây và giảm nguy cơ overfitting.

* Phương Pháp Chia Nhánh Ngẫu Nhiên:

Khi xây dựng mỗi cây, Random Forest chọn một số lượng ngẫu nhiên các biến để xem xét khi thực hiện quá trình chia nhánh. Điều này giúp tạo ra sự đa dạng giữa các cây và giảm nguy cơ overfitting.

* Dự Đoán Bằng Phiếu Bầu (Voting):

Trong bài toán phân loại, kết quả dự đoán của Random Forest là kết quả của việc phiếu bầu từ tất cả các cây. Trong bài toán dự đoán giá trị số, kết quả là giá trị trung bình của tất cả các cây.

* Quản Lý Overfitting:

Random Forest giảm nguy cơ overfitting bằng cách sử dụng nhiều cây và ngẫu nhiên lựa chọn mẫu và biến. Việc này giúp mô hình tổng hợp thông tin từ nhiều nguồn và làm giảm tương quan giữa các cây.

* Khả Năng Xử Lý Dữ Liệu Rất Lớn:

Random Forest có khả năng xử lý tốt các tập dữ liệu lớn và có nhiều biến. Thuật toán này cũng có thể xử lý dữ liệu có nhiễu và dữ liệu thiếu.

* Tầm Quan Trọng của Biến:

Random Forest cung cấp một phương pháp để đánh giá tầm quan trọng của các biến trong mô hình, giúp hiểu rõ hơn về cách mô hình đưa ra dự đoán.

* Hiệu Suất Cao và Ổn Định:

Random Forest thường mang lại hiệu suất cao và ổn định trên nhiều loại dữ liệu và tác vụ, làm cho nó trở thành một lựa chọn phổ biến trong thực tế.

Random Forest thường được ưa chuộng trong các vấn đề phân loại, dự đoán, và làm việc với dữ liệu phức tạp.

### Các bước xây dựng thuật toán

Bước 1: Thu thập Dữ Liệu:

Thuật toán bắt đầu bằng việc thu thập dữ liệu đào tạo, gồm các biến độc lập và biến phụ thuộc, từ tập dữ liệu.

Bước 2: Lựa Chọn Ngẫu Nhiên Mẫu:

Random Forest thực hiện việc lấy ngẫu nhiên một số lượng mẫu từ tập dữ liệu đào tạo (có thể trùng lặp) để xây dựng mỗi cây quyết định. Việc này giúp tạo ra sự đa dạng giữa các cây và ngăn chặn overfitting.

Bước 3: Xây Dựng Cây Quyết Định:

Đối với mỗi mẫu được chọn ngẫu nhiên, một cây quyết định (Decision Tree) được xây dựng bằng cách sử dụng một phần của dữ liệu. Cây được xây dựng thông qua quy trình chia nhánh nhị phân, giống như trong thuật toán CART.

Bước 4: Tính Toán Dự Đoán:

Khi tất cả các cây đã được xây dựng, Random Forest sử dụng mỗi cây để đưa ra dự đoán độc lập. Đối với bài toán phân loại, dự đoán cuối cùng là kết quả của việc biểu đồ phiếu bầu (voting) từ tất cả các cây; đối với bài toán dự đoán giá trị số, dự đoán cuối cùng là giá trị trung bình của tất cả các cây.

Bước 5: Giảm Thiểu Overfitting:

Random Forest giảm nguy cơ overfitting bằng cách chia dữ liệu thành nhiều tập con và xây dựng các cây trên các tập con này. Việc này giúp mô hình tổng hợp thông tin từ nhiều nguồn và làm giảm tương quan giữa các cây.

Bước 6: Quản lý Đặc Điểm Quan Trọng:

Random Forest cung cấp một công cụ quan trọng để đánh giá tầm quan trọng của các biến. Các biến được sắp xếp theo độ quan trọng giảm dần, giúp hiểu rõ hơn về cách mô hình đưa ra quyết định.

Random Forest thường được sử dụng hiệu quả trong nhiều ứng dụng khác nhau như phân loại, dự đoán giá trị số, và giải quyết vấn đề của dữ liệu không cân bằng. Nó là một trong những mô hình kết hợp phổ biến và mạnh mẽ trong lĩnh vực máy học.

# Bài toán, cài đặt chương trình

## **Bài toán**

### Đặt vấn đề

Xem xét các yếu tố của thời tiết được đưa ra bởi tập mẫu dưới đây.

Thuộc tính mục tiêu ở đây là “ weather” có giá trị là “ sunny”, “ rain”, “ overcast”.

Giá trị của thuộc tính này được dự đoán dựa vào các thuộc tính mô tả

| Day | temp | humidity | wind | weather |
| --- | --- | --- | --- | --- |
| D1 | Hot | Normal | Weak | Sunny |
| D2 | Hot | Normal | Strong | Overcast |
| D3 | Hot | High | Strong | Rain |
| D4 | Hot | High | Weak | Overcast |
| D5 | Hot | Low | Weak | Sunny |
| D6 | Hot | Low | Strong | Sunny |
| D7 | Cool | Normal | Strong | Overcast |
| D8 | Cool | Low | Avg | Sunny |
| D9 | Cool | High | Strong | Rain |
| D10 | Cool | High | Weak | Rain |
| D11 | Cool | Normal | Weak | Overcast |
| D12 | Cool | Low | Weak | Sunny |
| D13 | Cool | Low | Strong | Sunny |
| D14 | Mild | High | Avg | Overcast |
| D15 | Mild | Low | Strong | Overcast |
| D16 | Mild | Normal | Weak | Rain |
| D17 | Mild | High | Strong | Rain |
| D18 | Mild | Normal | Avg | Rain |
| D19 | Mild | Low | Avg | Overcast |
| D20 | Mild | High | Weak | Overcast |

Data gồm 4 thuộc tính:

* Thuộc tính đầu:
* temp (nhiệt độ): Nhận các giá trị: Hot, Cool, Mild
* humidity (độ ẩm): Nhận các giá trị: High, Low, Normal
* wind (sức gió): Nhận các giá trị: Weak, Strong
* Thuộc tính đích: weather(thời tiết): Nhận các giá trị: Sunny, Rain, Overcast

### Giải quyết bài toán

Sunny = S; Overcast = O; Rain = R; temp = T; humidity = H; wind = H;

* Xét thuộc tính temp:
* Xét thuộc tính humidity:
* Xét thuộc tính wind:

Ta có:

Gain(H) lớn nhất Tách tại humidity

* Cây quyết định sau lần phân hoạch đầu tiên:



* Xét thuộc tính Low của humidity:

|  | temp | wind | weather |
| --- | --- | --- | --- |
| 5 | Hot | Weak | Sunny |
| 6 | Hot | Strong | Sunny |
| 8 | Cool | Avg | Sunny |
| 12 | Cool | Weak | Sunny |
| 13 | Cool | Strong | Sunny |
| 15 | Mild | Strong | Overcast |
| 19 | Mild | Avg | Overcast |

Ta có:

Gain(T1) lớn nhất Chọn Node tiếp theo là temp

* Cây quyết định sau lần phân hoạch thứ 2:











* Xét thuộc tính Normal của humidity:

|  | temp | wind | weather |
| --- | --- | --- | --- |
| 1 | Hot | Weak | Sunny |
| 2 | Hot | Strong | Overcast |
| 7 | Cool | Strong | Overcast |
| 11 | Cool | Weak | Overcast |
| 16 | Mild | Weak | Rain |
| 18 | Mild | Avg | Rain |

Ta có:

Gain(T2) lớn nhất Chọn Node tiếp theo là temp

* Cây quyết định sau lần phân hoạch thứ 3:















Ở 2 lần phân hoạch trước đã chọn 2 node là humidity và temp nên ở lần phân hoạch thứ 3 này sau Node temp ta có thể chọn được Node tiếp theo là Node wind.

* Xét thuộc tính High của humidity:

|  | temp | humidity | wind | weather |
| --- | --- | --- | --- | --- |
| 3 | Hot | High | Strong | Rain |
| 4 | Hot | High | Weak | Overcast |
| 9 | Cool | High | Strong | Rain |
| 10 | Cool | High | Weak | Rain |
| 14 | Mild | High | Avg | Overcast |
| 17 | Mild | High | Strong | Rain |
| 20 | Mild | High | Weak | Overcast |

Ta có:

Gain(W3) lớn nhất Chọn Node tiếp theo là wind

* Cây quyết định sau lần phân hoạch thứ 4:

















Ở lần phân hoạch 1 và lần phân hoạch 4 ta đã chọn được 2 Node là humidity và wind nên sau Node wind ở lần phân hoạch 4 ta chọn luôn Node tiếp theo là Node temp.

### Kết quả



*Hình 3.1. Cây quyết định*













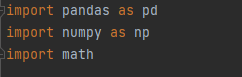




*Hình 3.1.3.1. Cây quyết định*

## **Cài đặt chương trình.**

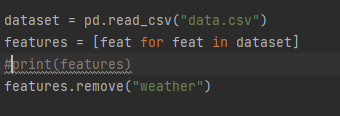
### Import thư viện



*Hình 3.2.1. import thư viện*

* Sử dụng thư viện pandas để xử lý dữ liệu
* Sử dụng thư viện numpy và math để hỗ trợ việc tính toán

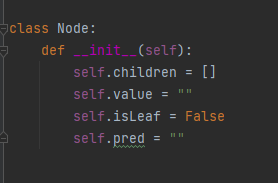
### Đọc dữ liệu và tách nhãn



*Hình 3.2.2. đọc dữ liệu và tách nhãn*

* Đọc dữ liệu từ file vào biến dataset
* Tạo một nhãn features và bỏ thuộc tính đích là “weather”

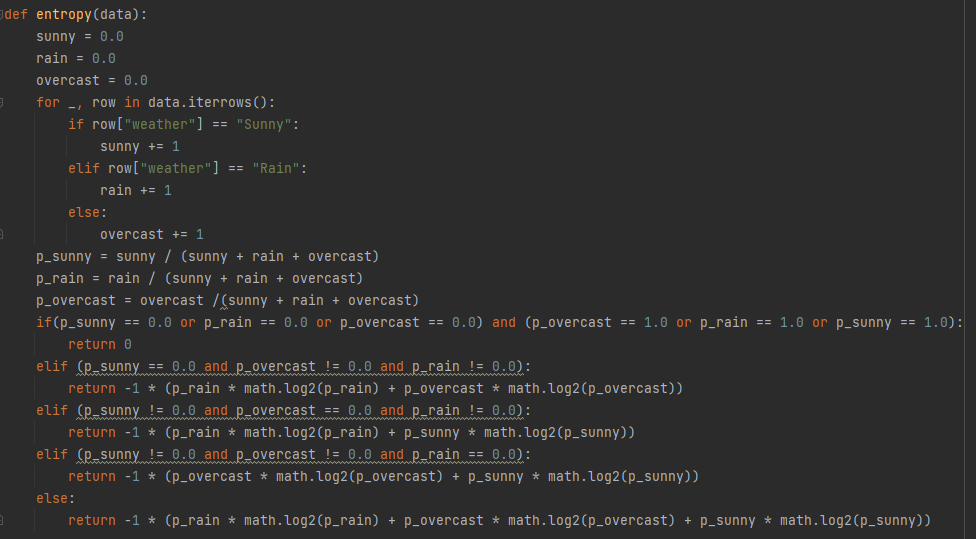
### Tạo Node



*Hình 3.2.3. Tạo Node*

* Biến children dùng để lưu trữ các con của Node
* Biến value dùng để lưu trữ giá trị thuộc tính
* Biến isLeaf dùng để xác định Node có phải là lá không
* Biến pred dùng để lưu quyết định thuộc tính đích

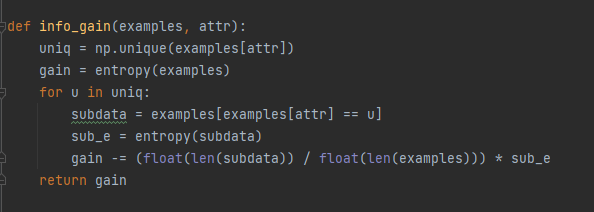
### Tính Entropy



*Hình 3.2.4. Tính Entropy*

* Đếm số lượng weather nhận các giá trị là : ‘Sunny’, ‘Rain’, ‘Overcast’
* Nếu xuất hiện 1 trong 3 giá trị ‘Sunny’, ‘Rain’, ‘Overcast’ nhận giá trị là 0 và tồn tại 1 trong 3 giá trị ‘Sunny’, ‘Rain’, ‘Overcast’ nhận giá trị là 1 thì trả về 0
* Nếu xuất hiện 1 trong 3 giá trị ‘Sunny’, ‘Rain’, ‘Overcast’ nhận giá trị là 0 và tồn tại 1 trong 3 giá trị ‘Sunny’, ‘Rain’, ‘Overcast’ nhận giá trị khác 1 thì bỏ thuộc tính nhận giá trị 0 và trả về theo công thức tính Entropy sau khi loại thuộc tính
* Nếu cả 3 đều khác 0 thì trả về theo công thức tính Entropy

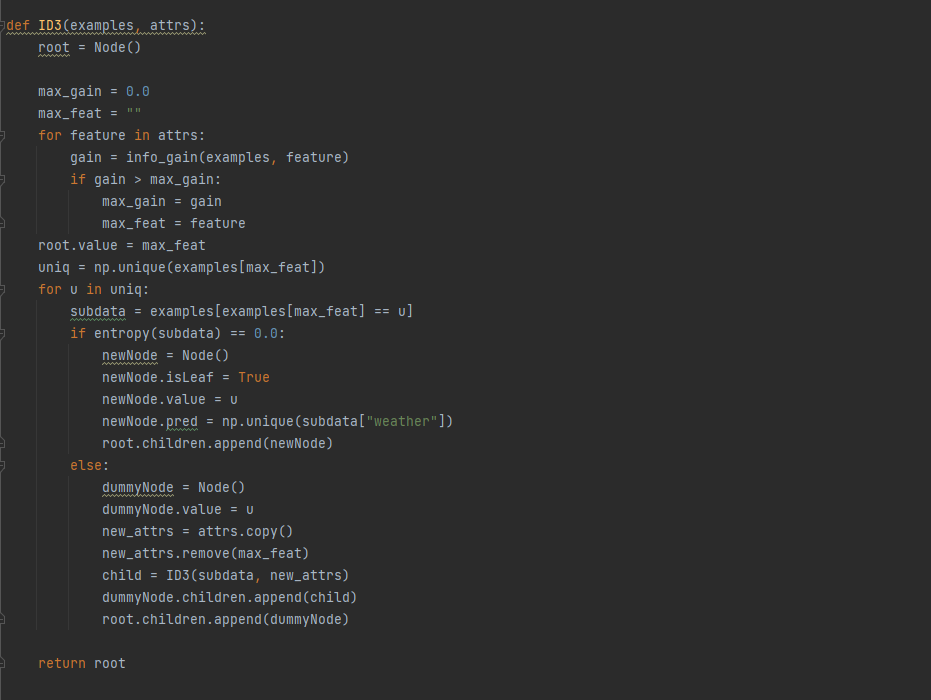
### Tính gain từng thuộc tính



*Hình 3.2.5. Tính gain*

Biến uniq nhận các giá trị không trùng lặp của trong bảng dữ liệu examples với nhãn là attr

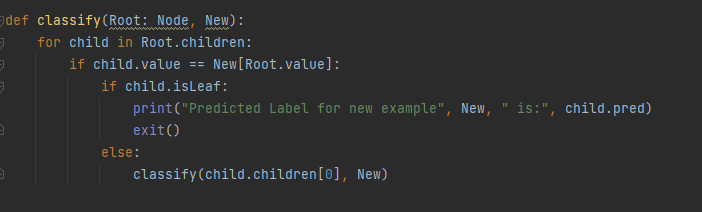
### Tạo cây quyết định



*Hình 3.2.6. Khởi tạo cây*

* Tìm thuộc tính có giá trị Gain lớn nhất gán vào biến max\_gain và nhãn của thuộc tính đó được gán vào biến max\_feat
* Biến uniq nhận các giá trị không trùng lặp của dữ liệu  examples với nhãn là max\_feat
* Biến subdata nhận bảng dữ liệu với examples có nhãn max\_feat  bằng lần lượt giá trị trong biến uniq
* Nếu subdata có Entropy bằng 0 thì nó được khởi tạo là Node lá và có quyết đinh là pred = np.unique(subdata[“weather”])
* Ngược lại thì nó chưa phải là Node lá thì gọi lại function ID3 với bảng dữ liệu mới là subdata với nhãn mới là nhẵn đã bỏ đi max\_feat
* Trả về root

### Kiểm tra dữ liệu

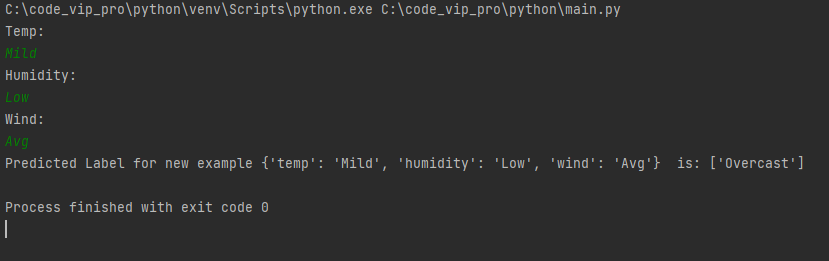


*Hình 3.2.7. kiểm tra dữ liệu*

* Biến child được khởi tạo là các biến con gốc ban đầu của cây
* Nếu giá trị biến child bằng với giá trị của dữ liệu kiểm tra thì đưa ra quyết định và thoát
* Ngược lại thì gọi lại function với con của child và new

### Kết quả





# 

# KẾT LUẬN VÀ KIẾN NGHỊ

Thuật toán ID3 có thể giải quyết bài toán về cây quyết định ứng dụng vào dự báo thời tiết một cách dễ dàng và đạt độ chính xác cao. Tuy nhiên, do thiếu sự hiểu biết về các thuộc tính ảnh hưởng đến thời tiết nhóm chúng em chỉ nêu ra những thuộc tính mà chúng em cảm thấy ảnh hưởng trực tiếp đến thời tiết. Cũng chính vì nhóm chỉ nêu được một số ít thuộc tính nên dữ liệu data sẽ hạn chế đến đến việc huấn luyện dự báo nên kết quả sẽ khó có thể đạt được chính xác.

Nhóm em nhận thức được bài tập lớn này sẽ có sự thiếu sót nên mong rằng cô sẽ đưa ra ý kiến nhận xét để nhóm em hoàn thiện hơn và bổ sung hoặc đưa gợi ý về việc tạo dữ liệu để huấn luyện đưa ra kết quả chính xác hơn.

Chúng em cảm ơn cô!

# TÀI LIỆU THAM KHẢO

1. Nguyễn Phương Nga (2021), Giáo trình trí tuệ nhân tạo, Trường Đại học Công nghiệp Hà Nội.
2. Mitchell, T. (1997). Machine Learning. McGraw-Hill.
3. Quinlan, J. R. (1986). Induction of Decision Trees. Machine Learning, 1(1), 81-106.
4. University of Waikato. (2021). Weka Documentation.
5. Chen, J. (2022). Understanding the ID3 Algorithm: A Comprehensive Guide.
6. Nguyen, H. (2023). An Introduction to Decision Trees and the ID3 Algorithm.