**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**VIỆN KỸ THUẬT CÔNG NGHỆ**



**BÁO CÁO TIỂU LUẬN MÔN**

**HỌC MÁY**

**XÁC ĐỊNH TÌNH TRẠNG ĐỘT QUỴ DỰA TRÊN CÁC YẾU TỐ SỬ DỤNG MÔ HÌNH CÂY QUYẾT ĐỊNH**

**Sinh viên thực hiện : Lê Sỹ Duy - 1724801030022**

**Hồ Phương Chi – 1724801030013**

**Trần Ngọc Hoài – 1724801030045**

**Lớp : D17PM01**

**Ngành : Kỹ thuật phần mềm**

**Giảng viên hướng dẫn : ThS. Bùi Thanh Khiết**

**Bình Dương , Tháng 8/2021**

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**VIỆN KỸ THUẬT CÔNG NGHỆ**



**BÁO CÁO TIỂU LUẬN MÔN**

**HỌC MÁY**

**XÁC ĐỊNH TÌNH TRẠNG ĐỘT QUỴ DỰA TRÊN CÁC YẾU TỐ SỬ DỤNG MÔ HÌNH CÂY QUYẾT ĐỊNH**

**Sinh viên thực hiện : Lê Sỹ Duy - 1724801030022**

**Hồ Phương Chi – 1724801030013**

**Trần Ngọc Hoài – 1724801030045**

**Lớp : D17PM01**

**Ngành : Kỹ thuật phần mềm**

**Giảng viên hướng dẫn : ThS. Bùi Thanh Khiết**

**Bình Dương , Tháng 8/2021**

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**VIỆN KỸ THUẬT – CÔNG NGHỆ**

**NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN**

Họ và tên giảng viên: **ThS.Bùi Thanh Khiết**

Tên đề tài: **XÁC ĐỊNH TÌNH TRẠNG ĐỘT QUỴ DỰA TRÊN CÁC YẾU TỐ SỬ DỤNG MÔ HÌNH CÂY QUYẾT ĐỊNH**

Nội dung nhận xét:

**Điểm:**

Bằng số:

Bằng chữ:

|  |  |
| --- | --- |
|  | **GIẢNG VIÊN CHẤM**  *(Ký, ghi rõ họ tên)* |

**ThS. Bùi Thanh Khiết**

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn chân thành và sâu sắc tới toàn thể thầy cô của Khoa Kỹ Thuật – Công Nghệ của trường Đại học Thủ Dầu Một hỗ trợ chúng em tận tình trong suốt thời gian học tập tại trường để có đủ kiến thức như ngày hôm nay.

Em cũng gửi lời cảm ơn chân thành đến Giảng viên hướng dẫn – Thầy Bùi Thanh Khiết trong suốt thời gian qua đã tận tình chỉ bảo để giúp em có thể hoàn thiện tốt bài tiểu luận này.

Tuy nhiên, do thời gian hạn hẹp, mặc dù đã nỗ lực hết sức mình, nhưng chắc rằng đồ án khó tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự thông cảm và chỉ bảo tận tình của các thầy cô và các bạn.

Em xin chân thành cảm ơn!

LỜI CAM ĐOAN

Em xin cam đoan đây là công trình nghiên cứu của riêng nhóm em và được sự hướng dẫn của TS. Bùi Thanh Khiết. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây.

Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong luận văn còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào em xin hoàn toàn chịu trách nhiệm về nội dung báo cáo của mình.** Trường Đại học Thủ Dầu Một không liên quan đến những vi phạm tác quyền, bản quyền do em gây ra trong quá trình thực hiện (nếu có).

*Bình Dương, ngày tháng 08 năm 2021*

*Người thực hiện  
 (ký tên và ghi rõ họ tên)*

TÓM TẮT

Hiện nay, thế giới đang trên đà phát triển cực mạnh nên việc lo lắng về sức khỏe của mọi người đang là điều vô cùng quan trọng. Đáng nói nhất là việc bị đột quỵ ở mỗi người, trong mọi lứa tuổi bất kì ai cũng có thể dẫn đến đột quỵ. Từ đó thấu hiểu được đột quỵ là mối lo của biết bao người nên chúng em đã xây dụng nên mô hình **“Xác định tình trạng đột quỵ dựa trên các yếu tố sử dụng mô hình cây quyết định”** để kiểm tra sức khỏe cho mọi người.

SUMMARY

Currently, the world is on the verge of strong development, so worrying about people's health is extremely important. Most notably, having a stroke in every person, of any age, can lead to a stroke. Since then, understanding that stroke is a concern for many people, we have built a model to "**Identify stroke conditions based on factors using the decision tree model**" to check people's health.

.

MỤC LỤC

[LỜI CẢM ƠN iii](#_Toc80304396)

[LỜI CAM ĐOAN iv](#_Toc80304397)

[TÓM TẮT v](#_Toc80304398)

[SUMMARY vi](#_Toc80304399)

[MỤC LỤC vii](#_Toc80304400)

[DANH MỤC HÌNH viii](#_Toc80304401)

[DANH MỤC CÁC KÝ TỰ, CHỮ VIẾT TẮT ix](#_Toc80304402)

[CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI 1](#_Toc80304403)

[1.1. Lý do chọn đề tài 1](#_Toc80304404)

[1.2. Bài toán 1](#_Toc80304405)

[1.3. Sự cần thiết của bài toán 2](#_Toc80304406)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 3](#_Toc80304407)

[2.1. Machine Learning 3](#_Toc80304408)

[2.2. Cây quyết định 3](#_Toc80304409)

[2.3. Giải thuật ID3 5](#_Toc80304410)

[CHƯƠNG 3. THỰC HIỆN 7](#_Toc80304411)

[3.1. Chuẩn bị dữ liệu 7](#_Toc80304412)

[3.2. Mô hình 8](#_Toc80304413)

[3.3. Các hàm chính 8](#_Toc80304414)

[3.4. Sản phẩm 15](#_Toc80304415)

[KẾT LUẬN 18](#_Toc80304416)

[1. Kết quả đạt được 18](#_Toc80304417)

[2. Ưu điểm và nhược điểm 18](#_Toc80304418)

[TÀI LIỆU THAM KHẢO 19](#_Toc80304419)

DANH MỤC HÌNH

[Hình 1 Bộ dữ liệu 6](#_Toc80302749)

[Hình 2 Mô hình xử lý 7](#_Toc80302750)

[Hình 3 Giao diện các phần chính của chương trình 14](#_Toc80302751)

[Hình 4 Giao diện khi nhập dữ liệu 15](#_Toc80302752)

[Hình 5 Giao diện khi chạy chương trình 15](#_Toc80302753)

DANH MỤC CÁC KÝ TỰ, CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Tiếng anh** | **Giải thích** |
| AI | AI Artificial Intelligence | Trí tuệ nhân tạo |
| ID3 | Iterative Dichotomiser 3 | Thuật toán cây quyết định |

2. TỔNG QUAN ĐỀ TÀI
   1. Lý do chọn đề tài

Đột quỵ là một trong những nguyên nhân gây tử vong hàng đầu trên thế giới. Trung bình cứ 3 phút lại có 1 ca tử vong do đột quỵ.

Bệnh nhân sau đột quỵ ngày càng được quan tâm ở các nước trên thế giới bởi có rất nhiều biến chứng ảnh hưởng chất lượng cuộc sống như trầm cảm, sa sút trí tuệ và đặc biệt là bệnh nhân còn tiếp tục tử vong.

Bên cạnh đó, trong lĩnh vực [máy học](https://vi.wikipedia.org/wiki/M%C3%A1y_h%E1%BB%8Dc), cây quyết định là một kiểu mô hình dự báo (predictive model), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Kỹ thuật học máy dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

Từ chính những điều nêu trên chúng tôi tiến hành đề tài “Xác định tình trạng đột quỵ dựa trên các yếu tố sử dụng mô hình cây quyết định”.

* 1. Bài 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. Các thuộc tính của đối tượng có thể thuộc các kiểu dữ liệu khác nhau như nhị phân (Binary) , định danh (Nominal), thứ tự (Ordinal), số lượng (Quantitative) trong khi đó thuộc tính phân lớp phải có kiểu dữ liệu là Binary hoặc Ordinal.

Tóm lại, 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 dữ liệu chưa biết.

Cây quyết định là một giải thuật được sử dụng thường xuyên trong đời sống hàng ngày.

ID3 (J. R. Quinlan 1993) sử dụng phương pháp tham lam tìm kiếm từ trên xuống thông qua không gian của các nhánh có thể không có backtracking. ID3 sử dụng Entropy và Information Gain để xây dựng một cây quyết định.

* 1. Sự cần thiết của bài toán

Bài toán “Xác định tình trạng đột quỵ dựa trên các yếu tố sử dụng mô hình cây quyết định” được thực hiện nhằm xác định khả năng bệnh đột quỵ dựa trên một vài yếu tố, từ đó chúng ta có thể xác định khả bị bệnh của bản thân và có phương pháp phòng tránh. Bài toán sử dụng phương pháp đơn giản và quen thuộc dễ nhìn.

1. CƠ SỞ LÝ THUYẾT
   1. Machine Learning

Machine learning gây nên cơn sốt công nghệ trên toàn thế giới trong vài năm nay. Trong giới học thuật, mỗi năm có hàng ngàn bài báo khoa học về đề tài này. Trong giới công nghiệp, từ các công ty lớn như Google, Facebook, Microsoft đến các công ty khởi nghiệp đều đầu tư vào machine learning. Hàng loạt các ứng dụng sử dụng machine learning ra đời trên mọi linh vực của cuộc sống, từ khoa học máy tính đến những ngành ít liên quan hơn như vật lý, hóa học, y học, chính trị. AlphaGo, cỗ máy đánh cờ vây với khả năng tính toán trong một không gian có số lượng phần tử còn nhiều hơn số lượng hạt trong vũ trụ, tối ưu hơn bất kì đại kì thủ nào, là một trong rất nhiều ví dụ hùng hồn cho sự vượt trội của machine learning so với các phương pháp cổ điển.

Machine Learning là một tập con của AI. Theo định nghĩa của Wikipedia, Machine learning is the subfield of computer science that “gives computers the ability to learn without being explicitly programmed”. Nói đơn giản, Machine Learning là một lĩnh vực nhỏ của Khoa Học Máy Tính, nó có khả năng tự học hỏi dựa trên dữ liệu đưa vào mà không cần phải được lập trình cụ thể. [1]

Có rất nhiều cách phân loại machine learning, thông thường thì machine learning sẽ được phân làm hai loại chính sau:

• Supervised learning: học có giám sát.

• Unsupervised learning: học không giám sát.

Ngoài ra, machine learning còn có thể phân làm các loại sau:

• Semi-supervised learning: học bán giám sát.

• Deep learning: học sâu (về một vấn đề nào đó).

• Reinforce learning: học củng cố/tăng cường.

* 1. Cây quyết định

Trong lý thuyết quyết định (chẳng hạn quản lí rủi ro), một cây quyết định (Decision Tree) là một đồ thị của các quyết định và các hậu quả có thể của nó (bao gồm rủi ro và hao phí tài nguyên). Cây quyết định được sử dụng để xây dựng một kế hoạch nhằm đạt được mục tiêu mong muốn. Các cây quyết định được dùng để hỗ trợ quá trình ra quyết định. Cây quyết định là một dạng đặc biệt của cấu trúc cây.

Trong lĩnh vực máy học (Learning Machine), cây quyết định là một kiểu mô hình dự báo (Predictive Model), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Mỗi một nút trong (Internal Node) tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện một giá trị cụ thể cho biến đó. Mỗi nút lá đại diện cho giá trị dự đoán của biến mục tiêu, cho trước các giá trị của các biến được biểu diễn bởi đường đi từ nút gốc tới nút lá đó. Kỹ thuật máy học dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

Học bằng cây quyết định cũng là một phương pháp thông dụng trong khai phá dữ liệu. Khi đó, cây quyết định mô tả một cấu trúc cây, trong đó, các lá đại diện cho các phân loại còn cành đại diện cho các kết hợp của các thuộc tính dẫn tới phân loại đó. Một cây quyết định có thể được học bằng cách chia tập hợp nguồn thành các tập con dựa theo một kiểm tra giá trị thuộc tính. Quá trình này được lặp lại một cách đệ qui cho mỗi tập con dẫn xuất. Quá trình đệ qui hoàn thành khi không thể tiếp tục thực hiện việc chia tách được nữa, hay khi một phân loại đơn có thể áp dụng cho từng phần tử của tập con dẫn xuất. Một bộ phân loại rừng ngẫu nhiên (Random Forest) sử dụng một số cây quyết định để có thể cải thiện tỉ lệ phân loại.

Cây quyết định cũng là một phương tiện có tính mô tả dành cho việc tính toán các xác suất có điều kiện.

Cây quyết định có thể được mô tả như là sự kết hợp của các kỹ thuật toán học và tính toán nhằm hỗ trợ việc mô tả, phân loại và tổng quát hóa một tập dữ liệu cho trước.

Dữ liệu được cho dưới dạng các bản ghi có dạng:

(x, y) = (x1, x2, x3…, xk, y)

Biến phụ thuộc (Dependant Variable) y là biến mà chúng ta cần tìm hiểu, phân loại hay tổng quát hóa. x1, x2, x3 … là các biến sẽ giúp ta thực hiện công việc đó.

So với các phương pháp [khai phá dữ liệu](https://vi.wikipedia.org/wiki/Khai_ph%C3%A1_d%E1%BB%AF_li%E1%BB%87u) khác, cây quyết định là phương pháp có một số ưu điểm:

* Cây quyết định dễ hiểu. Người ta có thể hiểu mô hình cây quyết định sau khi được giải thích ngắn.
* Việc chuẩn bị dữ liệu cho một cây quyết định là cơ bản hoặc không cần thiết. Các kỹ thuật khác thường đòi hỏi [chuẩn hóa dữ liệu](https://vi.wikipedia.org/wiki/Chu%E1%BA%A9n_h%C3%B3a_d%E1%BB%AF_li%E1%BB%87u), cần tạo các biến phụ (dummy variable) và loại bỏ các giá trị rỗng.
* Cây quyết định có thể xử lý cả dữ liệu có giá trị bằng số và dữ liệu có giá trị là tên thể loại. Các kỹ thuật khác thường chuyên để phân tích các bộ dữ liệu chỉ gồm một loại biến. Chẳng hạn, các luật quan hệ chỉ có thể dùng cho các biến tên, trong khi [mạng nơ-ron](https://vi.wikipedia.org/wiki/M%E1%BA%A1ng_n%C6%A1-ron) chỉ có thể dùng cho các biến có giá trị bằng số.
* Cây quyết định là một mô hình hộp trắng. Nếu có thể quan sát một tình huống cho trước trong một mô hình, thì có thể dễ dàng giải thích điều kiện đó bằng logic Boolean. Mạng nơ-ron là một ví dụ về mô hình hộp đen, do lời giải thích cho kết quả quá phức tạp để có thể hiểu được.
* Có thể thẩm định một mô hình bằng các kiểm tra thống kê. Điều này làm cho ta có thể tin tưởng vào mô hình.
* Cây quyết định có thể xử lý tốt một lượng dữ liệu lớn trong thời gian ngắn. Có thể dùng [máy tính cá nhân](https://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh_c%C3%A1_nh%C3%A2n) để phân tích các lượng dữ liệu lớn trong một thời gian đủ ngắn để cho phép các nhà chiến lược đưa ra quyết định dựa trên phân tích của cây quyết định.

Nhược điểm của cây quyết định:

* Khó giải quyết được những vấn đề có dữ liệu phụ thuộc thời gian liên tục - dễ xảy ra lỗi khi có quá nhiều lớp chi phí tính toán để xây dựng mô hình cây quyết định CAO.
  1. Giải thuật ID3

Giải thuật ID3 (gọi tắt là ID3) Được phát triển đồng thời bởi Quinlan trong AI và Breiman, Friedman, Olsen và Stone trong thống kê. ID3 là một giải thuật học đơn giản nhưng tỏ ra thành công trong nhiều lĩnh vực. ID3 là một giải thuật hay vì cách biểu diễn tri thức học được của nó, tiếp cận của nó trong việc quản lý tính phức tạp, heuristic của nó dùng cho việc chọn lựa các khái niệm ứng viên, và tiềm năng của nó đối với việc xử lý dữ liệu nhiễu.

ID3 biểu diễn các khái niệm (concept) ở dạng các cây quyết định (decision tree). Biểu diễn này cho phép chúng ta xác định phân loại của một đối tượng bằng cách kiểm tra các giá trị của nó trên một số thuộc tính nào đó.

Như vậy, nhiệm vụ của giải thuật ID3 là học cây quyết định từ một tập các ví dụ rèn luyện (training example) hay còn gọi là dữ liệu rèn luyện (training data).

Input: Một tập hợp các ví dụ. Mỗi ví dụ bao gồm các thuộc tính mô tả một tình huống, hay một đối tượng nào đó, và một giá trị phân loại của nó.

Output: Cây quyết định có khả năng phân loại đúng đắn các ví dụ trong tập dữ liệu rèn luyện, và hy vọng là phân loại đúng cho cả các ví dụ chưa gặp trong tương lai.

Giải thuật ID3 xây dựng cây quyết định được trình bày như sau:

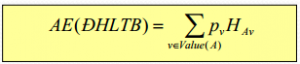
Lặp:

1. Chọn A <= thuộc tính quyết định “tốt nhất” cho nút kế tiếp.
2. Gán A là thuộc tính quyết định cho nút.
3. Với mỗi giá trị của A, tạo nhánh con mới của nút.
4. Phân loại các mẫu huấn luyện cho các nút lá.
5. Nếu các mẫu huấn luyện được phân loại hoàn toàn thì NGƯNG,  
   Ngược lại, lặp với các nút lá mới.

Thuộc tính tốt nhất ở đây là thuộc tính có entropy trung bình thấp nhất theo thuộc tính kết quả với  Entropy được tính như sau:

* Gọi S là tập các mẫu huấn luyện.
* Gọi p là tỷ lệ các mẫu dương trong S.
* Ta có H ≡ – p.log2p – (1 – p).log2(1 – p).

Entropy trung bình của một thuộc tính bằng trung bình theo tỉ lệ của entropy các nhánh:



1. THỰC HIỆN
   1. Chuẩn bị dữ liệu

Dữ liệu được lấy từ trang https://www.kaggle.com/fedesoriano/stroke-prediction-dataset gồm 6 thuộc tính. Sau khi chỉnh sửa lại dữ liệu thì ta có mỗi thuộc tính có 2 kết quả. Việc này giúp cho quá trình tính toán và xác định kết quả được chính xác hơn:

1. gender (giới tính): Female (nữ) và Male (nam).
2. hypertension (tình trạng huyết áp): 1 (huyết áp cao) và 0 (huyết áp thấp).
3. heart\_disease (bệnh tim): 0 ( không có tiền án) 1 ( có bị bệnh).
4. ever\_married (Tình trạng hôn nhân): yes (đã kết hôn) và no (chưa kết hôn).
5. Residence\_type (môi trường sống) : Rural (Nông thôn) và Urban (Đô thị).
6. smoking\_status (Tình trạng hút thuốc): never smoked (chưa từng hút thuốc) và smokes (có hút thuốc).
7. stroke (đột quỵ): 0 (không bị đột quỵ) và 1 (bị đột quỵ).

Dữ liệu gồm 5110 dòng giá trị:

Table

Description automatically generated

**Hình 1 Bộ dữ liệu**

* 1. Mô hình

Dựa theo các thông tin và cơ sở lý thuyết trên chúng em xây dựng mô hình xử lý bài toán “Xác định tình trạng đột quỵ dựa trên các yếu tố sử dụng mô hình cây quyết định” được trình bày như sau:

* Phân tích và thu thập dữ liệu về các yếu tố có khả năng ảnh hưởng đến tình trạng đột quỵ của các bệnh nhân trên toàn thế giới và phân loại xử lý dữ liệu để có thể dễ dàng tính toán.
* Phân tích, tính toán các số liệu có được và từ đó đưa ra được khả năng nhiễm bệnh cao nhất xuất phát chủ yếu từ triệu chứng nào bằng học máy.
* Sau đó tiến hành lưu lựa chọn và phân tách các đặc tính để vẽ cây.
* Cuối cùng xây dựng ứng dụng desktop để mọi người có thể dễ nghiên cứu và xem xét kết quả:

Diagram

Description automatically generated

**Hình 2 Mô hình xử lý**

* 1. Các hàm chính

Dựa theo mô hình trên chương trình có các hàm chính sau:

Giải thuật ID3:

ID3\_algorithm(Training\_Set, Class\_Labels, Attributes)

Tạo nút Root của cây quyết định.

If tất cả các ví dụ của Training\_Set thuộc cùng lớp c

Return Cây quyết định có nút Root được gắn với (có nhãn) lớp c

If Tập thuộc tính Attributes là rỗng

Return Cây quyết định có nút Root được gắn với nhãn lớp ≡

Majority\_Class\_Label(Training Set)

A ← Thuộc tính trong tập Attributes có khả năng phân loại “tốt nhất” đối với Training\_Set

Thuộc tính kiểm tra cho nút Root ← A

For each Giá trị có thể v của thuộc tính A

Bổ sung một nhánh cây mới dưới nút Root, tương ứng với trường hợp: “Giá trị của A là v”

Xác định Training\_Setv = {ví dụ x | x ⊆ Training\_Set, xA=v}

If (Training\_Setv là rỗng) Then

Tạo một nút lá với nhãn lớp ≡ Majority\_Class\_Label(Training\_Set)

Gắn nút lá này vào nhánh cây mới vừa tạo

Else Gắn vào nhánh cây mới vừa tạo một cây con sinh ra bởi

ID3\_algorithm(Training\_Setv, Class\_Labels, {Attributes A})

Return Root:

· Code [C#]:

private TreeNode ID3(List<List<string>> Examples,

List<Attribute> Attribute, string bestat)

{

Solution = Solution + "--------------------------------- Xét " + bestat + " -------------------------------";

if (CheckAllPositive(Examples))

{

Solution += "\n Tất cả các mẫu đều khẳng định => Trả về nút gốc với nhãn 1";

return new TreeNode(new Attribute("1"));

}

if (CheckAllNegative(Examples))

{

Solution += "\n Tất cả các mẫu đều phủ định => Trả về nút gốc với nhãn 0";

return new TreeNode(new Attribute("0"));

}

if (Attribute.Count == 0)

{

Solution += "\n Các thuộc tính rỗng => Trả về nút gốc có giá trị phổ biến nhất ";

return new TreeNode(new

Attribute(GetMostCommonValue(Examples)));

}

Attribute BestAttribute =

GetBestAttribute(Examples, Attribute, bestat);

int LocationBA = Attributes.IndexOf(BestAttribute);

TreeNode Root = new TreeNode(BestAttribute);

for (int i = 0; i < BestAttribute.Value.Count; i++)

**{**

**List<List<string>> Examplesvi = new List<List<string>>();**

**for (int j = 0; j < Examples.Count; j++)**

**{**

**if (Examples[j][LocationBA].ToString() ==**

**BestAttribute.Value[i].ToString())**

**Examplesvi.Add(Examples[j]);**

**}**

**if (Examplesvi.Count == 0)**

**{**

**Solution += "\n Các thuộc tính rỗng => Trả**

**về nút gốc có giá trị phổ biến nhất ";**

**return new TreeNode(new**

**Attribute(GetMostCommonValue(Examplesvi)));**

**}**

**else**

**{**

**Solution += "\n";**

**Attribute.Remove(BestAttribute);**

**Root.AddNode(ID3(Examplesvi, Attribute,**

**BestAttribute.Value[i]));**

**}**

**}**

**return Root;**

**}**

Hàm tính Entropy:

* Công thức:    Entropy (S) = – p+ log2 p+ – p- log2 p-

· Code [C#]:

**private double GetEntropy(int Positives, int Negatives)**

**{**

**if (Positives == 0)**

**return 0;**

**if (Negatives == 0)**

**return 0;**

**double Entropy;**

**int total = Negatives + Positives;**

**double RatePositves = (double)Positives / total;**

**double RateNegatives = (double)Negatives / total;**

**Entropy = -RatePositves \* Math.Log(RatePositves, 2) - RateNegatives \* Math.Log(RateNegatives, 2);**

**return Entropy;**

}

Hàm tính Gain:

* Công thức:

[](http://3.bp.blogspot.com/-dEwCtvagAa0/Uc_qnvhmKJI/AAAAAAAABAk/bH9icsR2aTk/s544/Untitled.png)

· Code [C#]:

**private double Gain(List<List<string>> Examples, Attribute A, string bestat)**

**{**

**double result;**

**int CountPositives = 0;**

**int[] CountPositivesA = new int[A.Value.Count];**

**int[] CountNegativeA = new int[A.Value.Count];**

**int Col = Attributes.IndexOf(A);**

**for (int i = 0; i < A.Value.Count; i++)**

**{**

**CountPositivesA[i] = 0;**

**CountNegativeA[i] = 0;**

**}**

**for (int i = 0; i < Examples.Count; i++)**

**{**

**int j = A.Value.IndexOf(Examples[i][Col].ToString());**

if (Examples[i][Examples[0].Count - 1] == "1")

**{**

**CountPositives++;**

**CountPositivesA[j]++;**

**}**

**else**

**{**

**CountNegativeA[j]++;**

**}**

**}**

**result = GetEntropy(CountPositives, Examples.Count - CountPositives);**

**for (int i = 0; i < A.Value.Count; i++)**

**{**

**double RateValue = (double)(CountPositivesA[i] + CountNegativeA[i]) / Examples.Count;**

**result = result - RateValue \* GetEntropy(CountPositivesA[i], CountNegativeA[i]);**

**}**

**Solution = Solution + "\n \* Gain(" + bestat + "," + A.Name + ") = " + result.ToString();**

**return result;**

}

Hàm chọn đặc tính tốt nhất:

* Phương pháp:

- Dựa vào giá trị gain của các đặc tính, đặc tính nào có Gain lớn nhất.

- Chọn đặc tính đó – đặc tính tốt nhất.

· Code [C#]:

**private Attribute GetBestAttribute(List<List<string>> Examples, List<Attribute> Attributes, string bestat)**

**{**

**double MaxGain = Gain(Examples, Attributes[0], bestat);**

**int Max = 0;**

**for (int i = 1; i < Attributes.Count; i++)**

**{**

**double GainCurrent = Gain(Examples, Attributes[i], bestat);**

**if (MaxGain < GainCurrent)**

**{**

MaxGain = GainCurrent;

**Max = i;**

**}**

**}**

**Solution = Solution + "\n\t=> Ta chọn Gain lớn nhất là : " + Attributes[Max].Name;**

**return Attributes[Max];**

}

**Đánh giá bài toán phân lớp**

Sử dụng phương pháp đánh giá True/False Positive/Negative: Cách đánh giá này thường được áp dụng cho các bài toán phân lớp có hai lớp dữ liệu. Cụ thể hơn, trong hai lớp dữ liệu này có một lớp nghiêm trọng hơn lớp kia và cần được dự đoán chính xác. Trong bài toán xác định có bệnh đột quỵ hay không thì việc không bị sót (miss) quan trọng hơn là việc chẩn đoán nhầm âm tính thành dương tính.

Phương pháp: Trong bài toán này, người ta thường định nghĩa lớp dữ liệu quan trọng hơn cần được xác định đúng là lớp Positive (P-dương tính), lớp còn lại được gọi là Negative (N-âm tính).

* Lấy giá trị phổ biến nhất của tập đích.

· Code [C#]:

**private string GetMostCommonValue(List<List<string>> Examples)**

**{**

**int CountPositive = 0;**

**for (int i = 0; i < Examples.Count; i++)**

**{**

**if (Examples[i][Examples[0].Count - 1] == "1")**

**CountPositive++;**

**}**

**int CountNegative = Examples.Count - CountPositive;**

**string Label;**

**if (CountPositive > CountNegative)**

**Label = "1";**

**else**

**Label = "0";**

**Solution = Solution + " là " + Label;**

**return Label;**

**}**

* Kiểm tra xem tất cả tập có phải là positive không

**.** Code [C#]:

**private bool CheckAllPositive(List<List<string>> Examples)**

**{**

**for (int i = 0; i < Examples.Count; i++)**

**{**

**if (Examples[i][Examples[0].Count - 1].ToString() == "0")**

**return false;**

**}**

**return true;**

**}**

* Kiểm tra xem tất cả tập có phải là Negative không

. Code [C#]:

**private bool CheckAllNegative(List<List<string>> Examples)**

**{**

**for (int i = 0; i < Examples.Count; i++)**

**{**

**if (Examples[i][Examples[0].Count - 1] == "1")**

**return false;**

**}**

**return true;**

**}**

**Vẽ cây**

Mô tả: quá trình vẽ cây gồm 2 phần

* Xây dựng cây:

**.** Code [C#]:

**public void GetTree()**

**{**

**Solution = "";**

**List<Attribute> at = new List<Attribute>();**

**for (int i = 0; i < Attributes.Count; i++)**

**{**

**at.Add(Attributes[i]);**

**}**

**Tree = ID3(Examples, at, "S");**

**Depth = GetDepth(Tree);**

**}**

* Lấy độ sau của cây:

.Code[C#]:

**private int GetDepth(TreeNode tree)**

**{**

**int depth;**

**if (tree.Childs.Length == 0)**

**return 1;**

**else**

**{**

**depth = GetDepth(tree.Childs[0]);**

**for (int i = 1; i < tree.Childs.Length; i++)**

**{**

**int depthchild = GetDepth(tree.Childs[i]);**

**if (depth < depthchild)**

**depth = depthchild;**

**}**

**depth++;**

**}**

**return depth;**

**}**

* 1. Sản phẩm
* Giao diện chính của chương trình Demo gồm 4 phần:

1. Bảng lưu dữ liệu training (Data Training).
2. Ghi ra các bước giải của thuật toán (Solutions).
3. Vẽ cây minh họa cho thuật toán (Decision Tree).
4. Các chức năng của chương trình (Control).

* Có 3 button với các chức năng như sau:

1. Load Data: Đưa dữ liệu training vào chương trình.
2. Run: Chạy giải thuật ID3.
3. Reset: Khởi động, chạy lại chương trình.

* Các bước chạy chương trình:
* Đầu tiên, nạp dữ liệu vào chương trình bằng button Load Data.
* Dữ liệu được đưa lên bảng Data Training.
* Sau đó, nhấn button Run để chạy giải thuật.
* Các bước giải sẽ được hiện ra ở phần bên dưới (Solutions).
* Cây được vẽ ra ở phần Tree.

Giao diện chương trình:

Graphical user interface

Description automatically generated

**Hình 3 Giao diện các phần chính của chương trình**

**Graphical user interface, application

Description automatically generated**

**Hình 4 Giao diện khi nhập dữ liệu**

**Graphical user interface

Description automatically generated with medium confidence**

**Hình 5 Giao diện khi chạy chương trình**

KẾT LUẬN

1. Kết quả đạt được

Sau khi tìm hiểu về các phương pháp học máy trong quá trình học và tiến hành xây dựng ứng dụng “Xác định tình trạng đột quỵ dựa trên các yếu tố sử dụng mô hình cây quyết định”, chúng em đã đạt được một số thứ như sau:

* Xây dựng được ứng dụng hiển thị cây quyết định dựa trên ngôn ngữ C#.
* Tìm hiểu được về các phương pháp xác định và cách tính toán bằng giải thuật ID3.
* Xử lý được các nguồn dữ liệu và làm sạch.
* Tìm hiểu được một số phương pháp máy học như ID3, Random Forest ..

2. Ưu điểm và nhược điểm

* Ưu điểm:
* Mô hình dễ hiểu và dễ giải thích.
* Mô hình dạng white box rõ ràng.
* Xây dựng nhanh.
* Phân lớp nhanh.
* Có thể xử lý cả dữ liệu có giá trị bằng số và dữ liệu có giá trị là tên thể loại.
* **Nhược điểm:**
* Không đảm bảo xây dựng được cây tốt ưu.
* Nguồn dữ liệu chưa thật sự đồng nhất do phải dùng từ nhiều nguồn khác nhau.
* Có thể overfitting (tạo ra những cây quá khớp với dữ liệu huấn luyện hay quá phức tạp).
* Độ chính xác chưa cao vì đối với bài toán xác định tình trạng bệnh rất khó có thể đưa ra quyết định.
* Khó giải quyết được những vấn đề có dữ liệu phụ thuộc thời gian liên tục.
* Dễ xảy ra lỗi khi có quá nhiều lớp chi phí tính toán để xây dựng mô hình cây quyết định CAO.
* Giải thuật chưa tối ưu do còn thiếu kiến thức về học máy.

TÀI LIỆU THAM KHẢO

**Website:**

1. Machine Learning cơ bản, Giới thiệu về Machine Learning: <https://machinelearningcoban.com/>
2. Top Dev, Machine learning là gì?: <https://topdev.vn/blog/machine-learning-la-gi/>
3. Mô phỏng thuật toán ID3: <http://www.cunghocit24.com/2013/06/chuong-trinh-mo-phong-thuat-toan-id3.html>
4. Giả thuật ID3: <https://machinelearningcoban.com/2018/01/14/id3/>

**Youtube:**

1. Tìm hiểu về thuật toán ID3- hỗ trợ ra cây quyết định <https://www.youtube.com/watch?v=_NoEe9MtWrY>
2. Machine Learning | Cây quyết định <https://www.youtube.com/watch?v=dc_YhOmxZ4A>
3. Decision Tree Solved | ID3 Algorithm <https://www.youtube.com/watch?v=UdTKxGQvYdc>

**Sách:**

1. **Machine Learning for Dummies** (IBM Limited Edition – Hurwitz and Kirsch) **,** tác giả **“**[**Luca Massaron**](https://www.google.com/search?q=Luca+Massaron&stick=H4sIAAAAAAAAAOPgE-LVT9c3NEwqLyhOLsoyU4Jx08pTUjJKtGSyk630k_Lzs_XLizJLSlLz4svzi7KtEktLMvKLFrHy-pQmJyr4JhYXJxbl5-1gZQQAfn4AmVEAAAA&sa=X&ved=2ahUKEwjF0ereqb3yAhXIfd4KHYYZDTsQmxMoATAnegQINhAD)**,**[**John Mueller**](https://www.google.com/search?q=machine+learning+for+dummies+john+mueller&stick=H4sIAAAAAAAAAOPgE-LVT9c3NEwqLyhOLsoyU-LSz9U3MDXMKC8w0JLJTrbST8rPz9YvL8osKUnNiy_PL8q2SiwtycgvWsSqmZuYnJGZl6qQk5pYlJeZl66Qll-kkFKam5uZWqyQlZ-Rp5BbmpqTk1q0g5URANpspcVqAAAA&sa=X&ved=2ahUKEwjF0ereqb3yAhXIfd4KHYYZDTsQmxMoAjAnegQINhAE)**”,** xuất bản lần đầu tiên: **10/05/2016.**
2. **Understanding Machine Learning** (Shai Shalev-Shwartz and Shai Ben-David) , tác giả **“Shai Ben-David, Shai Shalev-Shwartz”,** xuất bản lần đầu tiên: **15/05/2014.**