TRƯỜNG ĐẠI HỌC THỦY LỢI

Khoa công nghệ thông tin

-----🙡🕮🙣-----



BÁO CÁO BÀI TẬP LỚN

môn Khai phá dữ liệu

**Đề tài: Khai phá dữ liệu bằng phương pháp phân lớp để dự đoán bệnh tiểu đường**

**Nhóm sinh viên thực hiện :** Nhóm 5

|  |  |
| --- | --- |
| Bùi Văn Cải | 1951060550 |
| Vũ Trung Kiên | 1951064085 |

**Thành viên nhóm :**

**Giảng viên phụ trách môn học : TS. Trần Mạnh Tuấn**

Hà Nội, 8/2023

**MỤC LỤC**

[MỞ ĐẦU 3](#_Toc144069319)

[CHƯƠNG 1 : TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU VÀ PHÁT BIỂU BÀI TOÁN 4](#_Toc144069320)

[1.1 Phát hiện tri thức và khai phá dữ liệu. 4](#_Toc144069321)

[1.2 Quy trình khám phá tri thức trong CSDL. 4](#_Toc144069322)

[1.3 Kỹ thuật khai phá luật kết hợp 5](#_Toc144069323)

[1.4 Kỹ thuật phân lớp 5](#_Toc144069324)

[1.5 Kỹ thuật phân cụm 6](#_Toc144069325)

[CHƯƠNG 2 : CHUẨN BỊ DỮ LIỆU VÀ TIỀN XỬ LÝ DỮ LIỆU 7](#_Toc144069326)

[2.1 Mô tả bài toán và dữ liệu 7](#_Toc144069327)

[2.2 Tiền xử lý dữ liệu 9](#_Toc144069328)

[CHƯƠNG 3 : KHAI PHÁ DỮ LIỆU BẰNG THUẬT TOÁN ID3 11](#_Toc144069329)

[3.1 Các bước thực thi 11](#_Toc144069330)

[3.2 Xây dựng Thuật toán ID3 11](#_Toc144069331)

[3.3 Đánh giá độ chính xác của mô hình. 16](#_Toc144069332)

[CHƯƠNG 4 : KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 19](#_Toc144069333)

[4.1 Kết luận. 19](#_Toc144069334)

[4.2 Hướng phát triển 19](#_Toc144069335)

[TÀI LIỆU THAM KHẢO 20](#_Toc144069336)

# MỞ ĐẦU

Ngày nay ứng dụng công nghệ thông tin vào việc lưu trữ và xử lý thông tin được áp dụng hầu hết trong lĩnh vực, điều này đã tạo ra một lượng lớn dữ liệu được lưu trữ với kích thước tăng lên không ngừng. Vấn đề khai thác các nguồn tri thức đó ngày càng trở nên nóng bỏng và đặt ra thách thức lớn cho nền công nghệ thông tin thế giới.

Nhu cầu về tìm kiếm và xử lý thông tin, cùng với yêu cầu về khả năng kịp thời khai thác chúng để mang lại những năng suất và chất lượng cho công tác quản lý, hoạt động kinh doanh... đã trở nên cấp thiết trong xã hội hiện đại. Để đáp ứng phần nào yêu cầu này, người ta đã xây dựng các công cụ tìm kiếm và xử lý thông tin nhằm giúp cho người dùng tìm kiếm được các thông tin cần thiết.

Các phương pháp khai thác cơ sở dữ liệu truyền thống chưa đáp ứng được các yêu cầu đó. Để giải quyết vấn đề này, một hướng đi mới đó là nghiên cứu và áp dụng kỹ thuật khai phá dữ liệu và khám phá tri thức. Do đó, việc nghiên cứu các mô hình phân lớp dữ liệu và áp dụng quy trình khai phá dữ liệu trong khai phá dữ liệu là một xu thế tất yếu vừa có ý nghĩa khoa học vừa mang ý nghĩa thực tiễn cao.

Từ đó, chúng em đã chọn đề tài: " Khai phá dữ liệu bằng phương pháp phân lớp để dự đoán bệnh tiểu đường " để làm báo cáo kết thúc môn học của mình.

# : TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU VÀ PHÁT BIỂU BÀI TOÁN

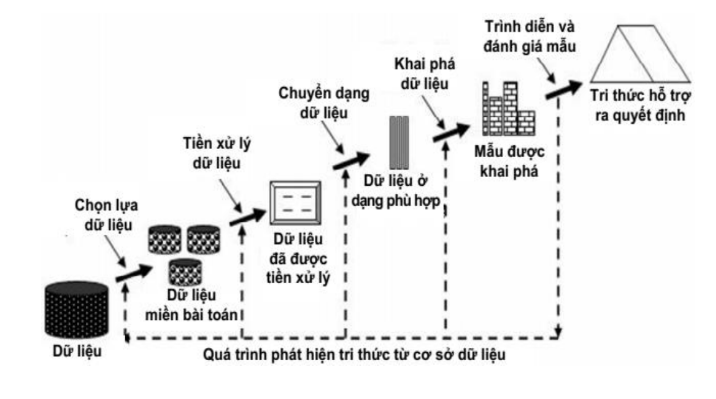
## Phát hiện tri thức và khai phá dữ liệu.

Phát hiện tri thức (Knowledge Discovery) trong các cơ sở dữ liệu là một qui trình nhận biết các mẫu hoặc các mô hình trong dữ liệu với các tính năng: hợp thức, mới, khả ích, và có thể hiểu được.

Khai phá dữ liệu (Data mining) là một quá trình tìm kiếm, phát hiện các tri thức mới, tiềm ẩn, hữu dụng trong CSDL lớn cho các đơn vị, tổ chức, doanh nghiệp, …

Khai phá dữ liệu có thể được sử dụng cho các lĩnh vực y tế, phân tích thị trường, xây dựng ... có thể được xem như là kết quả của sự tiến triển tự nhiên của công nghệ thông tin.

## Quy trình khám phá tri thức trong CSDL.



*Hình 1. 1 Quá trình khám phá tri thức từ cơ sở dữ liệu*

Quá trình phát hiện tri thức bao gồm các bước:

* Chuẩn bị dữ liệu
* Trích chọn dữ liệu: trích chọn những tập dữ liệu cần khai phá từ các tập dữ liệu khác nhau theo một tiêu chí nhất định.
* Làm sạch dữ liệu: Các nhiễu và dữ liệu không nhất quán sẽ được loại bỏ.
* Tích hợp dữ liệu: Dữ liệu từ nhiều nguồn khác có thể được tổ hợp lại.
* Chuyển đổi dữ liệu: Dữ liệu sau khi được chọn lọc sẽ được chuyển đổi hay hợp nhất về dạng thích hợp cho việc khai phá.
* Khai phá dữ liệu: Quá trình cốt lõi, tất yếu trong đó các phương pháp thông minh sẽ được áp dụng nhằm trích rút ra các mẫu dữ liệu.
* Đánh giá mẫu: Các nhà phân tích dữ liệu sẽ dựa trên một số độ đo nào đó để xác định lợi ích thực sự, độ quan trọng của các mẫu biểu diễn tri thức.
* Biểu diễn tri thức: Giai đoạn này các kĩ thuật biểu diễn và hiển thị tri thức sẽ được sử dụng để đưa tri thức đã lấy ra đến người dùng.

## Kỹ thuật khai phá luật kết hợp

Trong khai phá dữ liệu, mục đích của luật kết hợp là tìm ra các mối quan hệ giữa các đối tượng trong khối lượng lớn dữ liệu. Để khai phá luật kết hợp có rất nhiều thuật toán, nhưng dùng phổ biến nhất là thuật toán Apriori. Đây là thuật toán khai phá tập phổ biến trong dữ liệu giao dịch để phát hiện các luật kết hợp dạng khẳng định nhị phân và được sử dụng để xác định, tìm ra các luật kết hợp trong dữ liệu giao dịch. Ngoài ra, còn có các thuật toán FP-growth, thuật toán Partition, …

## Kỹ thuật phân lớp

Trong kỹ thuật phân lớp gồm có các thuật toán:

* Phân lớp bằng cây quyết định (giải thuật ID3, J48): phân lớp dữ liệu dựa trên việc lập nên cây quyết định, nhìn vào cây quyết định có thể ra quyết định dữ liệu thuộc phân lớp nào.
* Phân lớp dựa trên xác suất (Naïve Bayesian): dựa trên việc giả định các thuộc tính độc lập mạnh với nhau qua việc sử dụng định lý Bayes.
* Phân lớp dựa trên khoảng cách (giải thuật K-nearest neighbor): làm như láng giềng làm, dữ liệu sẽ được phân vào lớp của k đối tượng gần với dữ liệu đó nhất.
* Phân lớp bằng SVM: phân lớp dữ liệu dựa trên việc tìm ra một siêu phẳng “tốt nhất” để tách các lớp dữ liệu trên không gian nhiều chiều hơn.

## Kỹ thuật phân cụm

Phân cụm dữ liệu là cách phân bố các đối tượng dữ liệu vào các nhóm/ cụm sao cho các đối tượng trong một cụm thì giống nhau hơn các phần tử khác cụm, gồm có một số phương pháp phân cụm cơ bản như:

* Phân cụm bằng phương pháp K-mean: tìm ra tâm của các cụm mà khoảng cách của tâm đó đến các đối tượng, dữ liệu khác là ngắn.
* Phân cụm trên đồ thị, ngoài ra khai phá dữ liệu có rất nhiều kỹ thuật, nhưng đây là những kỹ thuật cơ bản và đơn giản trong khai phá dữ liệu mà chúng em được tìm hiểu.

# : CHUẨN BỊ DỮ LIỆU VÀ TIỀN XỬ LÝ DỮ LIỆU

## Mô tả bài toán và dữ liệu

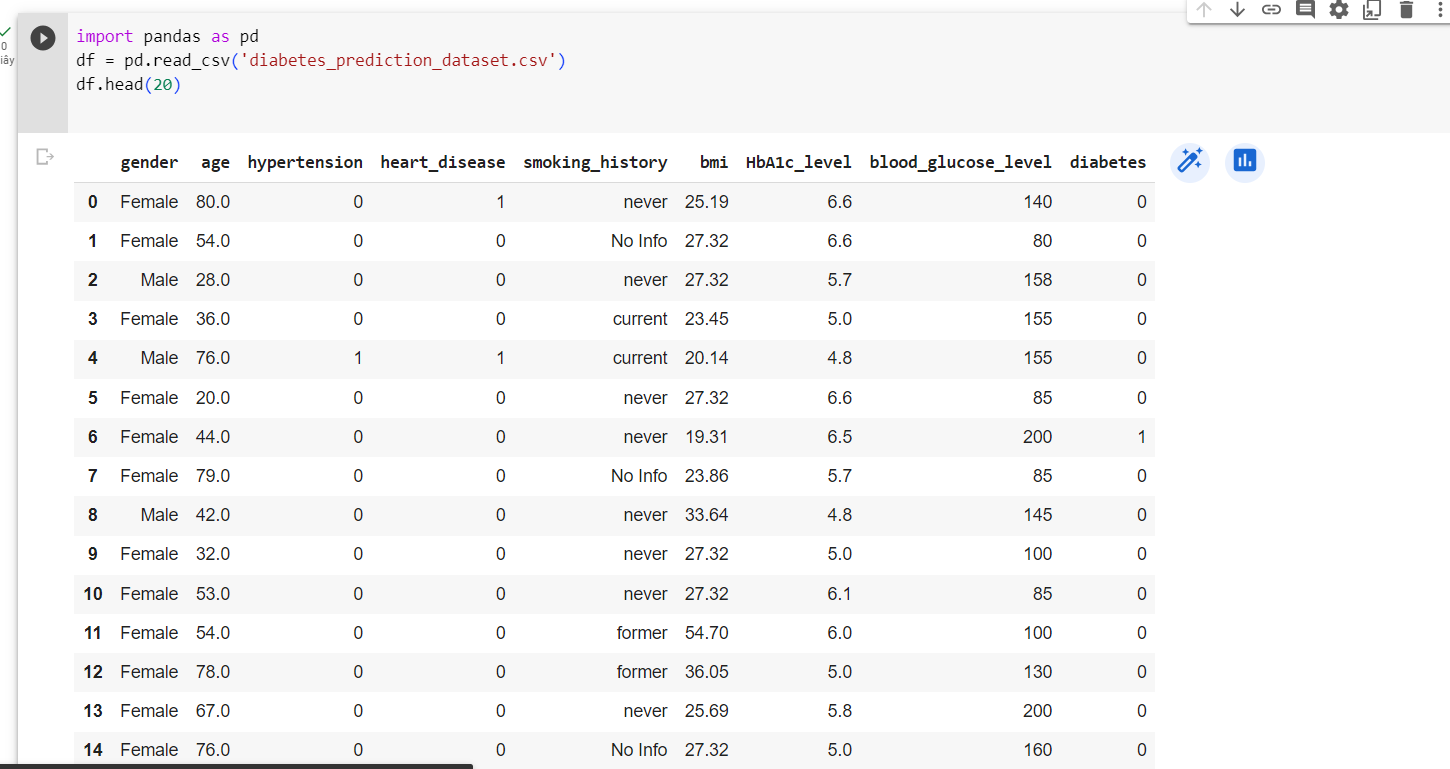
* Bài toán phân lớp bệnh tiểu đường. Từ nguồn dữ liệu thô ta tiến hành chuẩn hóa, sau đó dựa vào những thông tin về các thuộc tính dữ liệu tương ứng, xây dựng thuật toán học máy ID3 và huấn huyện trên dữ liệu có sẵn, khi đưa một input gồm các thông tin của một dữ liệu (x), mô hình sẽ đưa ra dự đoán là xem bệnh nhân đó có bị tiểu đường hay không (y)?
* Bộ dữ liệu dự đoán bệnh tiểu đường là tập hợp dữ liệu y tế và nhân khẩu học từ bệnh nhân, cùng với tình trạng bệnh tiểu đường của họ (tích cực hoặc tiêu cực). Bộ dữ liệu này có thể được sử dụng để xây dựng các mô hình học máy nhằm dự đoán bệnh tiểu đường ở bệnh nhân dựa trên tiền sử bệnh và thông tin nhân khẩu học của họ. Điều này có thể hữu ích cho các chuyên gia chăm sóc sức khỏe trong việc xác định những bệnh nhân có nguy cơ mắc bệnh tiểu đường và phát triển các kế hoạch điều trị cá nhân hóa. Ngoài ra, bộ dữ liệu có thể được các nhà nghiên cứu sử dụng để khám phá mối quan hệ giữa các yếu tố y tế và nhân khẩu học khác nhau và khả năng mắc bệnh tiểu đường.
* Với bộ dữ liệu thô trên có những trường còn thiếu, giá trị các thuộc tính chênh lệch lớn, chất lượng bộ dữ liệu không cao khi sử dụng làm dữ liệu cho các thuật toán học máy dẫn đến kết quả không cao.

Input: Bảng dataset chứa dữ liệu và kết quả phân của 10000 dữ liệu phân loại bệnh tiểu đường.

Output: Dự đoán bệnh nhân có bị tiểu đường hay không.

Bộ dữ liệu bao gồm các đặc điểm như:

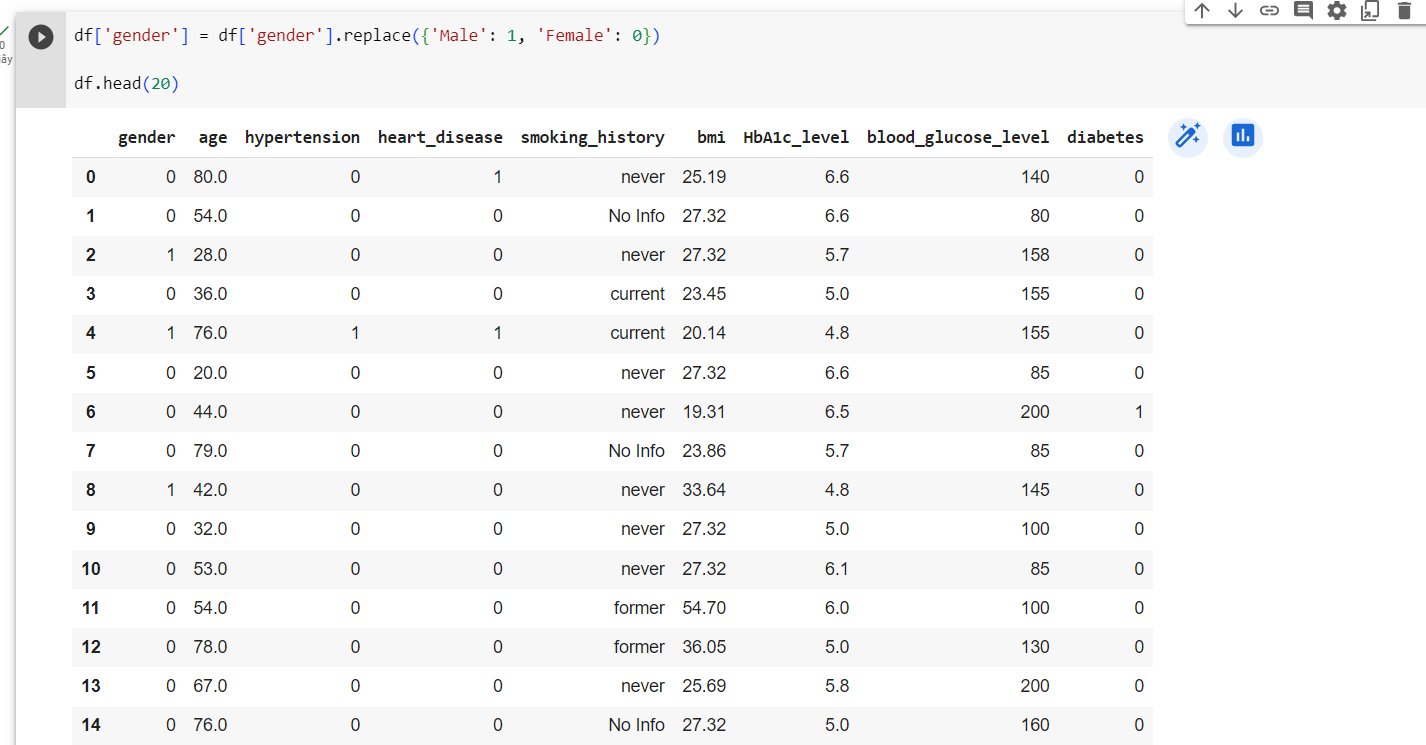
* Giới tính đề cập đến giới tính sinh học của cá nhân, có thể ảnh hưởng đến khả năng mắc bệnh tiểu đường của họ.
* Tuổi tác là một yếu tố quan trọng vì bệnh tiểu đường thường được chẩn đoán ở người lớn tuổi hơn. Độ tuổi dao động từ 0-80 trong bộ dữ liệu.
* Tăng huyết áp là một tình trạng bệnh lý trong đó huyết áp trong động mạch tăng cao liên tục. Nó có các giá trị 0 hoặc 1 trong đó 0 nghĩa là họ không bị tăng huyết áp và 1 nghĩa là họ bị tăng huyết áp.
* Bệnh tim là một tình trạng y tế khác có liên quan đến việc tăng nguy cơ mắc bệnh tiểu đường. Nó có các giá trị 0 hoặc 1 trong đó 0 nghĩa là họ không mắc bệnh tim và 1 nghĩa là họ bị bệnh tim.
* Tiền sử hút thuốc cũng được coi là một yếu tố nguy cơ đối với bệnh tiểu đường và có thể làm trầm trọng thêm các biến chứng liên quan đến bệnh tiểu đường. Trong tập dữ liệu có 5 loại tức là Never, former, current, not current, ever.
* BMI (Body Mass Index) là thước đo lượng mỡ trong cơ thể dựa trên cân nặng và chiều cao. Giá trị BMI cao hơn có liên quan đến nguy cơ mắc bệnh tiểu đường cao hơn. Phạm vi của BMI trong tập dữ liệu là từ 10,16 đến 71,55. BMI dưới 18,5 là thiếu cân, 18,5-24,9 là bình thường, 25-29,9 là thừa cân và từ 30 trở lên là béo phì.
* Mức HbA1c (Hemoglobin A1c) là thước đo mức đường huyết trung bình của một người trong 2-3 tháng qua. Mức độ cao hơn cho thấy nguy cơ phát triển bệnh tiểu đường cao hơn. Hầu hết hơn 6,5% Mức HbA1c cho thấy bệnh tiểu đường.
* Mức glucose trong máu đề cập đến lượng glucose trong máu tại một thời điểm nhất định. Nồng độ glucose trong máu cao là một chỉ số chính của bệnh tiểu đường.
* Bệnh tiểu đường là biến mục tiêu được dự đoán, với giá trị 1 cho biết có bệnh tiểu đường và 0 cho biết không có bệnh tiểu đường.



*Hình 2.1 Dữ liệu thô*

## Tiền xử lý dữ liệu

- Thay đổi giá trị trong cột thuộc tính gender : Female bằng 0, Male bằng 1.



*Hình 2.2 Dữ liệu sau khi thay đổi giá trị thuộc tính gender*

* Chuẩn hóa dữ liệu thuộc tính age, bmi, HbA1c\_level, blood\_glucose\_level bằng phương pháp Min-Max Scaling sử dụng thư viện sk-learn về khoảng [0,1].



*Hình 2.3 Dữ liệu sau khi chuẩn hóa giá trị các thuộc tính*

- Đối với cột thuộc tính smoking\_history ta thực hiện:

Thay đổi các ô dữ liệu trống hay No Info ngẫu nhiên thành 1 trong 5 giá trị: Never, former, current, not current, ever.

Thay giá trị never ngẫu nhiên thành 1, former thành 2, current thành 3, not current thành 4, ever thành 5.



*Hình 2.4 Dữ liệu sau khi thay đổi giá trị thuộc tính smoking\_history*

# : KHAI PHÁ DỮ LIỆU BẰNG THUẬT TOÁN ID3

## Các bước thực thi

- Ngôn ngữ sử dụng: Python

Bước 1: Đọc dữ liệu thô.

Bước 2: Tiền xử lý dữ liệu.

Bước 3: Thiết lập cây quyết định bằng Python

Bước 4: Thực hiện huấn luyện mô hình bằng dữ liệu đã được xử lý.

Bước 4: Tiến hành đánh giá hiệu quả thuật toán đối với tập dữ liệu được dùng theo hai hướng:

- Phương pháp 1: Sử dụng thư viện: so sánh việc huấn luyện dùng dữ liệu thô và dữ liệu đã được xử lý.

- Phương pháp 2: Triển khai thuật toán không sử dụng thư viện: so sánh việc huấn luyện dùng dữ liệu thô và dữ liệu đã được xử lý.

## Xây dựng Thuật toán ID3

class Node():

    def \_\_init\_\_(self, feature\_index=None, threshold=None, left=None, right=None, info\_gain=None, value=None):

        # cho nút quyết định

        self.feature\_index = feature\_index

        self.threshold = threshold

        self.left = left

        self.right = right

        self.info\_gain = info\_gain

        # cho nút lá

        self.value = value

class DecisionTreeClassifier\_handmade():

    def get\_params(self, deep = False):

        return {}

    def \_\_init\_\_(self, min\_samples\_split=2, max\_depth=2):

        ''' constructor '''

        # khởi tạo gốc của cây

        self.root = None

        # điều kiện dừng

        self.min\_samples\_split = min\_samples\_split

        self.max\_depth = max\_depth

    def build\_tree(self, dataset, curr\_depth=0):

        X, Y = dataset[:,:-1], dataset[:,-1]

        num\_samples, num\_features = np.shape(X)

        # chia nhỏ cho đến khi các điều kiện dừng được đáp ứng

        if num\_samples>=self.min\_samples\_split and curr\_depth<=self.max\_depth:

            # chia nhỏ cho đến khi các điều kiện dừng được đáp ứng

            best\_split = self.get\_best\_split(dataset, num\_samples, num\_features)

            # kiểm tra xem thông tin thu được có tích cực không

            if best\_split["info\_gain"]>0:

                # tái diễn trái

                left\_subtree = self.build\_tree(best\_split["dataset\_left"], curr\_depth+1)

                # tái diễn phải

                right\_subtree = self.build\_tree(best\_split["dataset\_right"], curr\_depth+1)

                # trả lại nút quyết định

                return Node(best\_split["feature\_index"], best\_split["threshold"],

                            left\_subtree, right\_subtree, best\_split["info\_gain"])

        # tính toán nút lá

        leaf\_value = self.calculate\_leaf\_value(Y)

        # trả lại nút lá

        return Node(value=leaf\_value)

    def get\_best\_split(self, dataset, num\_samples, num\_features):

        ''' chức năng để tìm sự phân chia tốt nhất '''

        # từ điển để lưu trữ sự phân chia tốt nhất

        best\_split = {}

        max\_info\_gain = -float("inf")

        # lặp lại tất cả các tính năng

        for feature\_index in range(num\_features):

            feature\_values = dataset[:, feature\_index]

            possible\_thresholds = np.unique(feature\_values)

            # lặp lại tất cả các giá trị tính năng có trong dữ liệu

            for threshold in possible\_thresholds:

                # nhận được sự phân chia hiện tại

                dataset\_left, dataset\_right = self.split(dataset, feature\_index, threshold)

                # kiểm tra xem các con không rỗng

                if len(dataset\_left)>0 and len(dataset\_right)>0:

                    y, left\_y, right\_y = dataset[:, -1], dataset\_left[:, -1], dataset\_right[:, -1]

                    # tính toán thu được thông tin

                    curr\_info\_gain = self.information\_gain(y, left\_y, right\_y, "gini")

                    # cập nhật sự phân chia tốt nhất nếu cần

                    if curr\_info\_gain>max\_info\_gain:

                        best\_split["feature\_index"] = feature\_index

                        best\_split["threshold"] = threshold

                        best\_split["dataset\_left"] = dataset\_left

                        best\_split["dataset\_right"] = dataset\_right

                        best\_split["info\_gain"] = curr\_info\_gain

                        max\_info\_gain = curr\_info\_gain

        # trả lại phần chia tốt nhất

        return best\_split

    def split(self, dataset, feature\_index, threshold):

        dataset\_left = np.array([row for row in dataset if row[feature\_index]<=threshold])

        dataset\_right = np.array([row for row in dataset if row[feature\_index]>threshold])

        return dataset\_left, dataset\_right

    def information\_gain(self, parent, l\_child, r\_child, mode="entropy"):

        weight\_l = len(l\_child) / len(parent)

        weight\_r = len(r\_child) / len(parent)

        if mode=="gini":

            gain = self.gini\_index(parent) - (weight\_l\*self.gini\_index(l\_child) + weight\_r\*self.gini\_index(r\_child))

        else:

            gain = self.entropy(parent) - (weight\_l\*self.entropy(l\_child) + weight\_r\*self.entropy(r\_child))

        return gain

    def entropy(self, y):

        class\_labels = np.unique(y)

        entropy = 0

        for cls in class\_labels:

            p\_cls = len(y[y == cls]) / len(y)

            entropy += -p\_cls \* np.log2(p\_cls)

        return entropy

    def gini\_index(self, y):

        class\_labels = np.unique(y)

        gini = 0

        for cls in class\_labels:

            p\_cls = len(y[y == cls]) / len(y)

            gini += p\_cls\*\*2

        return 1 - gini

    def calculate\_leaf\_value(self, Y):

        Y = list(Y)

        return max(Y, key=Y.count)

    def print\_tree(self, tree=None, indent=" "):

        if not tree:

            tree = self.root

        if tree.value is not None:

            print(tree.value)

        else:

            print("X\_"+str(tree.feature\_index), "<=", tree.threshold, "?", tree.info\_gain)

            print("%sleft:" % (indent), end="")

            self.print\_tree(tree.left, indent + indent)

            print("%sright:" % (indent), end="")

            self.print\_tree(tree.right, indent + indent)

    def fit(self, X, Y):

        dataset = np.concatenate((X, Y), axis=1)

        self.root = self.build\_tree(dataset)

    def predict(self, X):

        preditions = [self.make\_prediction(x, self.root) for x in X]

        return preditions

    def make\_prediction(self, x, tree):

        if tree.value!=None: return tree.value

        feature\_val = x[tree.feature\_index]

        if feature\_val<=tree.threshold:

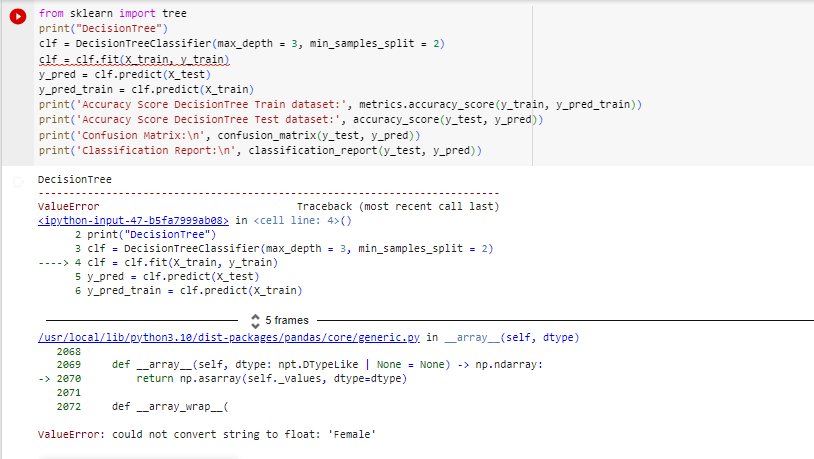
            return self.make\_prediction(x, tree.left)

        else:

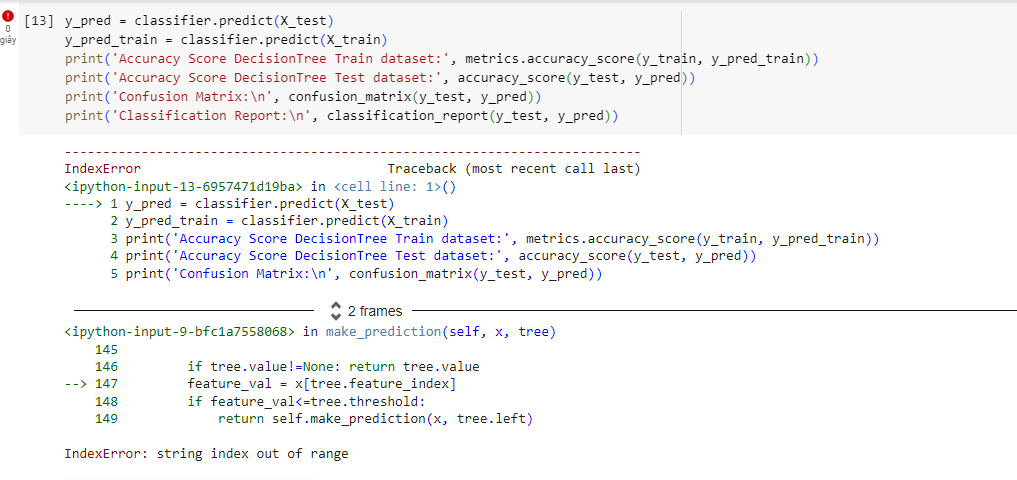
            return self.make\_prediction(x, tree.right)

## Đánh giá độ chính xác của mô hình.

Đối với dữ liệu thô chưa được xử lý, thực hiện với cả hai phương pháp đều không hoạt động được do trong dữ liệu thô có cả dạng dữ liệu string, float, int.

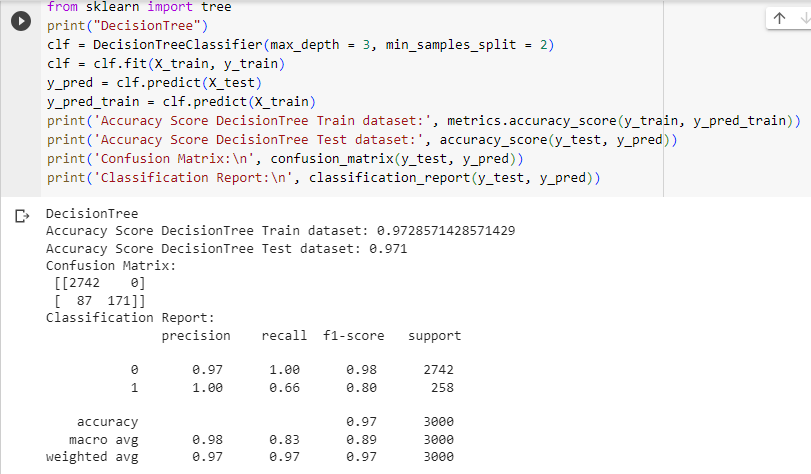


*Hình 3.1 Kết quả chạy phương pháp 2 với dữ liệu thô*



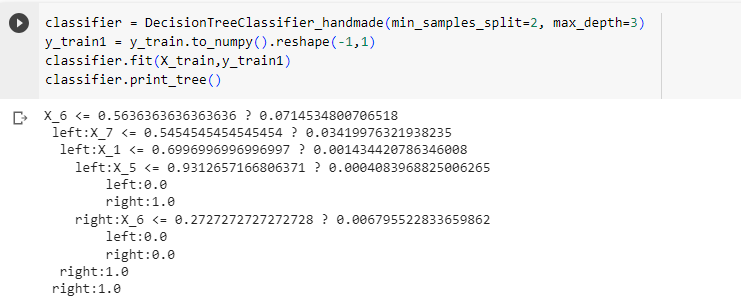
*Hình 3.2 Kết quả chạy phương pháp 2 với dữ liệu thô*

Đối với dữ liệu đã được xử lý khi thực hiện với phương pháp 1 ta được kết quả sau:

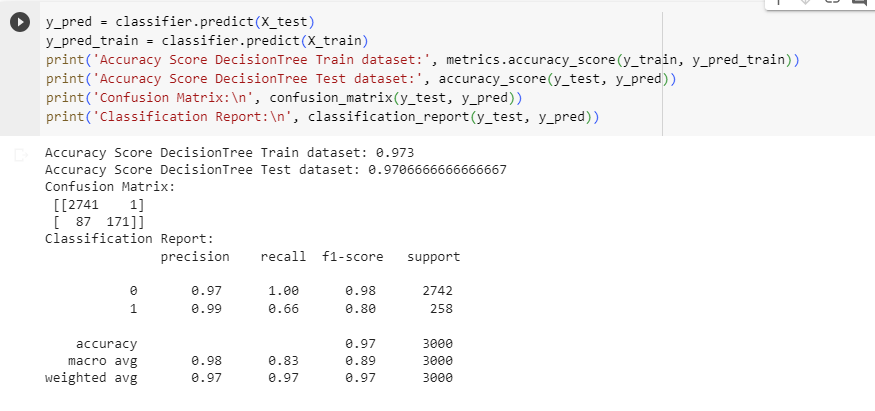


*Hình 3.3 Kết quả các độ đo phương pháp 1 với dữ liệu đã được xử lý*

Đối với dữ liệu đã được xử lý sau khi thực hiện với phương pháp 2 ta thu được kết quả như sau:



*Hình 3.4 Hình ảnh cây quyết định được hình thành với phương pháp 2*



*Hình 3.5 Kết quả các độ đo phương pháp 2 với dữ liệu đã được xử lý*

# : KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận.

Phân lớp dữ liệu là lĩnh vực khá quan trọng trong khai phá dữ liệu nhằm đưa ra những dự đoán, xu hướng trong tương lai, nó được ứng dụng trong nhiều ngành như y tế, thương mại, ngân hàng, giáo dục, .... Trong quá trình tìm hiểu và hoàn thành bài tập lớn với đề tài “**Khai phá dữ liệu bằng phương pháp phân lớp để dự đoán bệnh tiểu đường**”. Nhóm em đã đạt được một số kết quả như sau:

* Tìm hiểu tổng quan về khai phá dữ liệu, bài toán phân lớp, thuật toán ID3 để từ đó xây dựng mô hình hỗ trợ phân lớp dự đoán bệnh tiểu đường.
* Thu thập dữ liệu, tiền xử lý dữ liệu bằng python. Thực hiện thuật toán ID3 bằng thư viện và xây dựng thuật toán ID3 bằng python.
* Đánh giá mô hình phân lớp.

Tuy nhiên bài tập nhóm vẫn còn một số hạn chế:

* Chưa thực hiện trên các thuật toán học máy khác nhau để đánh giá mô hình phù hợp cho tập dữ liệu hiện có.
* Tham số đầu vào chưa tối ưu được độ chính xác mô hình.

## Hướng phát triển

Xây dựng, cải tiến mô hình phân lớp dự đoán bệnh tiểu đường với phương pháp học máy khác như SVM, Mạng Neuron, Random Forest, …

Sau khi hoàn thành dự án bài tập lớn này chúng em đã học được những kiến cơ bản cần nắm vững về khai phá dữ liệu, các mô hình học máy cũng như áp dụng kiến thức đó vào bài toán thực để đưa ra được đề tài bài tập lớn này. Sau cùng nhóm xin gửi lời cảm ơn chân thành đến thầy Trần Mạnh Tuấn đã hướng dẫn và giúp đỡ nhóm trong quá trình thực hiện dự án.

# TÀI LIỆU THAM KHẢO

[1] https://machinelearningcoban.com

[2] Bài giảng Khai Phá dữ liệu