

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

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**🕯✡🕮🕮✡🕯**



**MÔN HỌC: ĐỒ ÁN 3**

**GVHD: Thầy Trần Nhật Quang**

**SVTH:**

**Nguyễn Đức Quy 16110193**

**Võ Huỳnh Yến Nhi 16110171**

**Tp. Hồ Chí Minh, tháng 12 năm 2019**

**MỤC LỤC**

[**A.** **TÌM HIỂU VỀ ĐẠO VĂN** 1](#_Toc26191644)

[**B.** **GIỚI THIỆU THƯ VIỆN SKLEARN** 2](#_Toc26191645)

[**a.** **Giới thiệu** 2](#_Toc26191646)

[**b.** **Lý do dùng Scikit-learn** 2](#_Toc26191647)

[**c.** **Cài đặt Scikit-learn** 2](#_Toc26191648)

[**d.** **Ưu điểm thư viện sklearn** 2](#_Toc26191649)

[**C.** **TÌM HIỂU VỀ DECISION TREE** 3](#_Toc26191650)

[**a.** **Định nghĩa** 3](#_Toc26191651)

[**b.** **Ví dụ thực tế** 4](#_Toc26191652)

[**c.** **Thuật toán** 4](#_Toc26191653)

[**d.** **Ứng dụng** 5](#_Toc26191654)

[**e.** **Ưu điểm và nhược điểm của Decision Tree** 5](#_Toc26191655)

[**D.** **TÌM HIỂU VỀ GRADIENT TREE BOOTING** 5](#_Toc26191656)

[**a.** **Giới thiệu Gradient Booting** 5](#_Toc26191657)

[**b.** **Thuật toán Gradient Booting** 6](#_Toc26191658)

[**c.** **Gradient tree booting** 7](#_Toc26191659)

[**E.** **CODE DEMO DECISION TREE** 7](#_Toc26191660)

[**a.** **Import thư viện** 7](#_Toc26191661)

[**b.** **Mô tả dữ liệu** 8](#_Toc26191662)

[**c.** **Tách dữ liệu tranning và dữ liệu test** 9](#_Toc26191663)

[**d.** **Dùng Decision tree cho tập huấn luyện** 12](#_Toc26191664)

[**e.** **Dự đoán kết quả** 12](#_Toc26191665)

[**f.** **Ước tính bao nhiêu trường hợp sai (tạo ma trận nhầm lẫn)** 12](#_Toc26191666)

[**g.** **Mô hình cây quyết định** 12](#_Toc26191667)

[**h.** **Đánh giá** 13](#_Toc26191668)

[**F.** **TÀI LIỆU THAM KHẢO** 14](#_Toc26191669)

# **TÌM HIỂU VỀ ĐẠO VĂN**

Đạo văn không chỉ đơn thuần là việc copy thành quả của một ai đó hay vay mượn các ý tưởng gốc, mà còn rất nhiều biến hóa khác của đạo văn. Tại các quốc gia phát triển họ coi ý tưởng hay sản phẩm trí thức là một tài sản được Pháp luật bảo vệ nên có thể nói đạo văn là một "tội".

Để hiểu rõ về đạo văn một cách rõ ràng hơn ta sẽ tìm hiểu những hình thức đạo văn khác nhau :

* “The Ghost Writer”: Sử dụng toàn bộ công trình của người khác thành của mình
* “The Photocopy”: Sao chép bố cục của các đoạn văn từ một nguồn duy nhất, không hề sửa đổi lại.
* “The Potluck Paper”: Cố gắng “trá hình” việc đạo văn của mình bằng cách sao chép từ nhiều nguồn khác nhau, biên tập lại các câu sao cho nội dung thật hợp lí.
* “The Poor Disguise”: Giữ lại các nội dung quan trọng của nguồn, nhưng người đó vẫn sửa lại một chút về “diện mạo” của bài viết đó bằng cách thay đổi từ khóa hay câu cú.
* “The Labor of Laziness”: Chú giải các nguồn khác nhau và nối chúng lại với nhau, thay vì dành nỗ lực tương tự cho công việc của mình.
* “The Self-Stealer”: Sao chép đáng kể các thành quả trước đó của chính mình để phục vụ cho bài viết/nghiên cứu mới.

Vậy để tránh việc đạo văn ta có những cách sau :

* Tìm hiểu kỹ về vấn đề mà bạn đang muốn nói tới
* Diễn đạt lại nhiều lần bằng các cách khác nhau.
* Nắm được những gì không cần trích dẫn.
* Nêu tên tác giả của ý tưởng đó.
* Trích dẫn nguồn bất cứ khi nào sử dụng lời trích, chú giải một cách chi tiết và cụ thể.

**Lời cam kết**

“Chúng tôi xin cam đoan đồ án này do chính chúng tôi thực hiện. Chúng tôi không sao chép, sử dụng bất kỳ tài liệu, mã nguồn của người khác mà không ghi rõ nguồn gốc. Chúng tôi xin chịu hoàn toàn trách nhiệm nếu vi phạm.”

# **GIỚI THIỆU THƯ VIỆN SKLEARN**

### **Giới thiệu**

Scikit-learn (sklearn) là thư viện mã nguồn mở về machine learning phổ biến nhất trong cộng đồng Python.

Sklearn chứa hầu hết các thuật toán machine learning từ cơ bản cho đến phức tạp , ta chỉ việc nhét dữ liệu vào, rồi lấy kết quả.

### **Lý do dùng Scikit-learn**

Scikit-learn cho phép ta sử dụng ngay các thuật toán quan trọng một cách đơn giản và hiệu quả.

Scikit-learn còn là một trong những lựa chọn hàng đầu của các researchers và deverlopers.

Trang chủ của thư viện: <http://scikit-learn.org/>

### **Cài đặt Scikit-learn**

Thư viện scikit-learn yêu cầu chúng ta phải cài đặt những module như dưới đây:

* Python
* NumPy
* SciPy

Đầu tiên ta phải cài đặt Python ở trên máy.

Sau khi đã có Python trên máy của mình, ta có thể cài đặt 2 gói NumPy và SciPy bằng lệnh :

**$ pip install numpy scipy matplotlib scikit-learn pandas**

### **Ưu điểm thư viện sklearn**

⬩ Hàm ngắn gọn, dễ hiểu

⬩ Xử lý được lượng dữ liệu lớn

⬩ Độ chính xác và tin cậy cao

⬩ Tốc độ tính toán rất nhanh

⬩ Hỗ trợ rất nhiều thuật toán Machine Learning đơn giản và phức tạp

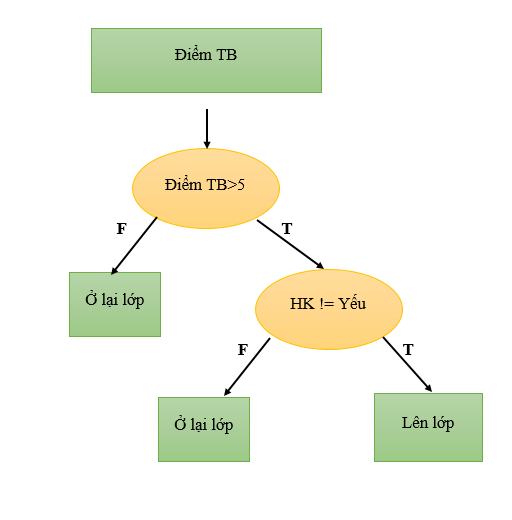
# **TÌM HIỂU VỀ DECISION TREE**

### **Định nghĩa**

* 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.
* Cây quyết định là một phương pháp phổ biến trong việc khai phá dữ liệu.
* Cây quyết định là mô hình hỗ trợ ra quyết định dựa trên đồ thị các điều kiện.
* Tại mỗi nút, ta sẽ đối chiếu các điều kiện thực tế để quyết định rẽ nhánh nào.
* Nút lá là quyết định cuối cùng.
* 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.
* Một cây quyết định bao gồm 4 thành phần chính:
  + Root Node: Nhánh chia đầu tiên của Decision Tree
  + Internal Node: Các nhánh chia tiếp theo của Decision Tree
  + Leaf Node: Các nhánh cuối cùng của một quyết định
  + Dept: số tầng của cây
* Một bảng dữ liệu với rất nhiều biến. Decision Tree sẽ sử dụng một vài biến để đưa ra việc xác định câu hỏi và thứ tự các biến nào chia dữ liệu để tạo ra Decision Tree có khả năng phân loại tốt nhất. Các hệ số này là:
  + Gini
  + Cross-Entropy
* Ta có thể hiểu với một bảng dữ liệu với rất nhiều biến, Decision Tree sẽ tính toán các hệ số Gini hoặc Cross-Entropy của các biến để quyết định xem biến nào sẽ là biến đầu tiên dùng để phân loại và các biến nào là biến lần lượt tiếp theo.
* Cây quyết định có 2 loại:
  + Cây hồi quy (Regression tree) ước lượng các hàm giá có giá trị là số thực thay vì được sử dụng cho các nhiệm vụ phân loại. (ví dụ: ước tính giá một ngôi nhà hoặc khoảng thời gian một bệnh nhân nằm viện)
  + Cây phân loại (Classification tree), phân loại như: giới tính (nam hay nữ), kết quả của một trận đấu (thắng hay thua).

### **Ví dụ thực tế**

* Ta xét ví dụ sau: Dựa vào điểm số và hạnh kiểm của học sinh ta sẽ biết được học sinh đó có được lên lớp không?
* Dữ liệu ban đầu:
  + Điểm TB
  + Hạnh kiểm
* Ta xây dựng được mô hình sau:



* Xét theo mô hình ta thấy:
  + Nếu điểm trung bình bé hơn 5 thì học sinh đó sẽ ở lại lớp
  + Nếu điểm trung bình lớn hơn 5 nhưng hạnh kiểm yếu thì học sinh đó ở lại lớp
  + Các trường hợp còn lại học sinh được lên lớp

### **Thuật toán**

* Để phân tích cây quyết định thì có khá nhiều phương pháp, sự khác nhau như sau:
  + ID3
    - ID3 (Iterative Dichotomiser 3) được phát triển vào nào 1986 bởi Ross Quinlan. Sử dụng lượng thông tin ứng với biến số phân loại sau đó dùng kỹ thuật tham lam ( = lựa chọn tối ưu địa phương ở mỗi bước đi với hy vọng tìm được tối ưu toàn cục )
    - Ví dụ như thuật toán tìm đường đi ngắn nhất của Dijkstra
  + C4.5
    - Được phát triển từ ID3. 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.
    - So với ID3, C4.5 không cần biến số phân loại lượng đặc trưng. Output theo dạng if-then, không hiển thị những phần cành không cần thiết.
  + C5.0
    - Là bản cải tiến của C4.5. Giúp cải thiện vấn đề hiệu năng và sử dụng ít bộ nhớ hơn.
  + CART
    - CART (Classification and Regression Trees) khá giống với C4.5. Được phát triển bởi Breiman năm 1984. Tạo cây phân tích dựa trên biến phân loại, giải thích, mục đính và hồi quy. Và scikit-learn có chứa phiên bản tối ưu hoá của CART.

### **Ứng dụng**

* Ứng dụng trong Banking Finance : có thể tham khảo thêm tại <https://www.kaggle.com/francksylla/titanic-machine-learning-from-disaster>

### **Ưu điểm và nhược điểm của Decision Tree**

* Ưu điểm
  + Mô hình sinh ra các quy tắc dễ hiểu cho người đọc, tạo ra bộ luật với mỗi nhánh lá là một luật của cây.
  + Dữ liệu đầu vào có thể là là dữ liệu missing, không cần chuẩn hóa hoặc tạo biến giả
  + Có thể làm việc với cả dữ liệu số và dữ liệu phân loại
  + Có thể xác thực mô hình bằng cách sử dụng các kiểm tra thống kê
  + Có khả năng là việc với dữ liệu lớn
* Nhược điểm
  + Mô hình cây quyết định phụ thuộc rất lớn vào dữ liệu của bạn. Thậm chí, với một sự thay đổi nhỏ trong bộ dữ liệu, cấu trúc mô hình cây quyết định có thể thay đổi hoàn toàn.
  + Cây quyết định hay gặp vấn đề overfitting

# **TÌM HIỂU VỀ GRADIENT TREE BOOTING**

### **Giới thiệu Gradient Booting**

* Gradient Booting là một kỹ thuật máy học cho các vấn đề hồi quy và phân loại , tạo ra một mô hình dự đoán dưới dạng một nhóm các mô hình dự đoán yếu, điển hình là các cây quyết định . Nó xây dựng mô hình theo kiểu giai đoạn khôn ngoan giống như các phương pháp tăng cường khác, và nó khái quát hóa chúng bằng cách cho phép tối ưu hóa một hàm mất mát khác biệt tùy ý.
* Ý tưởng về việc tăng cường độ dốc bắt nguồn từ quan sát của Leo Breiman rằng việc tăng cường có thể được hiểu là một thuật toán tối ưu hóa trên một hàm J(Ѳ) phù hợp. Các thuật toán tối ưu hóa hàm J(Ѳ) trên không gian hàm bằng cách lặp đi lặp lại chọn một hàm (giả thuyết yếu) chỉ theo hướng gradient âm
* Ta có Hypothesis:



*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

### **Thuật toán Gradient Booting**

* Input:
  + Tập dữ liệu {(xi, yi)}i=1 và i chạy đến n
  + Một số chức năng mất khác biệt: L(y,F(x))
  + Số lần lặp M
* Thuật toán
  + Khởi tạo mô hình với giá trị không đổi



*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

* + Với m=1 đến M
    - Tính toán



*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

* + - Sử dụng bộ dữ liệu tập huấn để học
    - Tính hệ số nhân bằng cách tối ưu hóa một chiều



*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

* + - Cập nhật mô hình

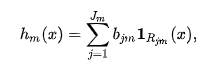


*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

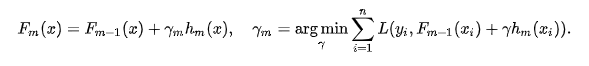
* + Output: Fm(x)

### **Gradient tree booting**

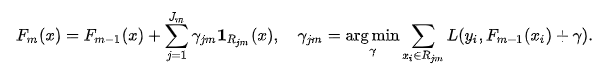
* Gradient Booting thường được sử dụng trong cây quyết định, có kích thước cố định
* Ta có :



*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

* + Sau đó các hệ số bjm được nhân với hệ số Ꝩm, mô hình được cập nhật:

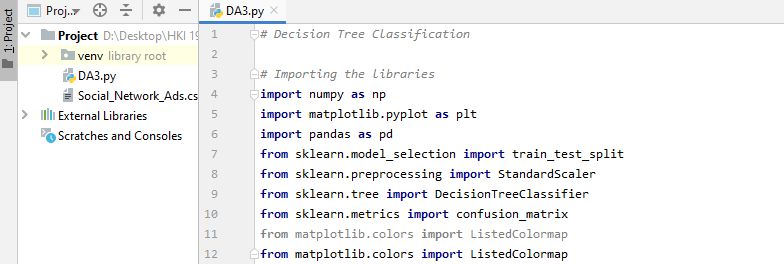
*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

* + Tuy nhiên Ꝩm được dùng cho toàn cây, để tối ưu hơn thì người ta chọn các Ꝩjm cho từng vùng của cây, thay vì cho toàn cây. Khi đó mô hình được cập nhật

*Nguồn:* [*https://en.wikipedia.org/wiki/Gradient\_boosting*](https://en.wikipedia.org/wiki/Gradient_boosting)

# **CODE DEMO DECISION TREE**

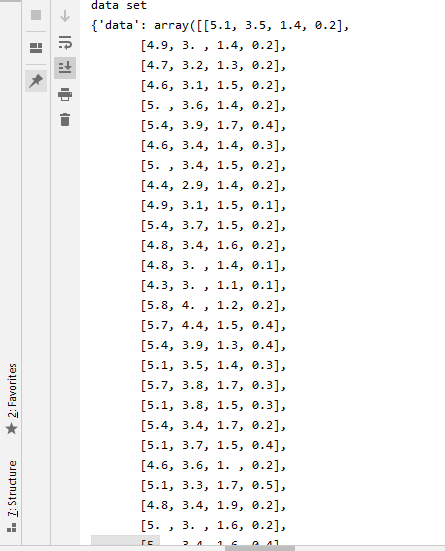
### **Import thư viện**



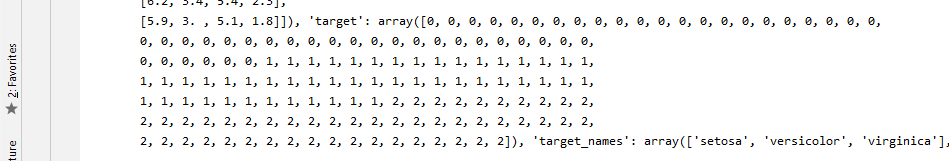
### **Mô tả dữ liệu**

* Sử dụng bộ dữ liệu IRIS từ thư viện sklearn





* Biến giải thích: tương ứng là chiều rộng, chiều dài của đài hoa, cánh hoa

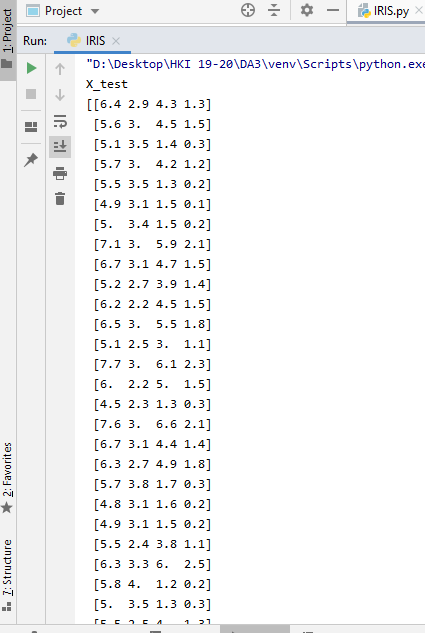


* Biến mục đích: 0, 1, 2 tương ứng với ba loại hóa đó là setosa, versicolor, virginica
* Mô tả dữ liệu: Sử dụng data Iris, có sẵn theo gói thư viện scikit-learn. Tiến hành phân loại hoa Iris, hay còn gọi là hoa cầu vồng, dựa theo độ rộng, dài của cánh hoa, cuống hoa. Hoa Iris rất đa dạng, tầm 150 loại. Chiều rộng, chiều dài của cánh hoa, tràng hoa sẽ là biến số giải thích, biến số mục đích là loại hoa để tiến hành tạo cây quyết định.

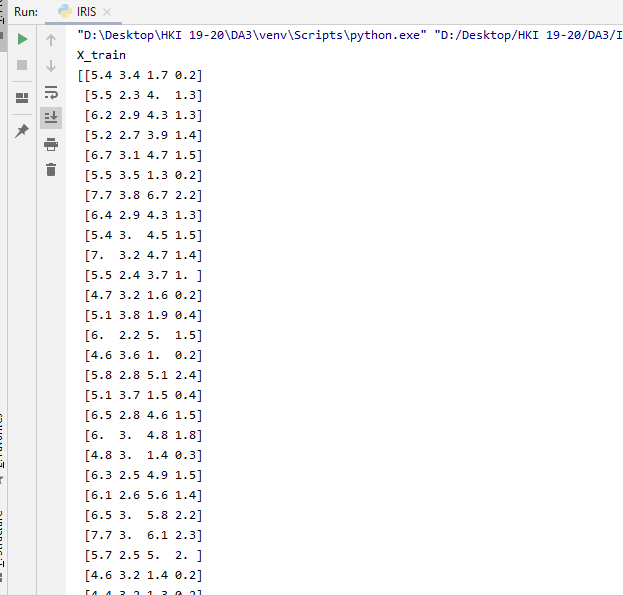
### **Tách dữ liệu tranning và dữ liệu test**



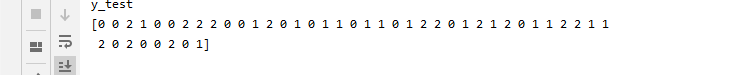
* Lấy 30% dữ liệu làm dữ liệu test và &0% dữ liệu làm dữ liệu train, test\_size=0.3
* Dữ liệu X\_Test



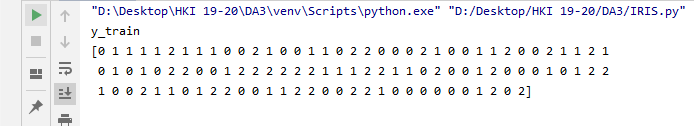
* X\_train



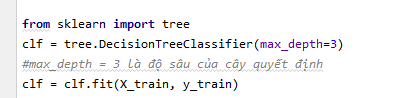
* Y\_test



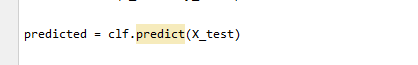
* Y\_train

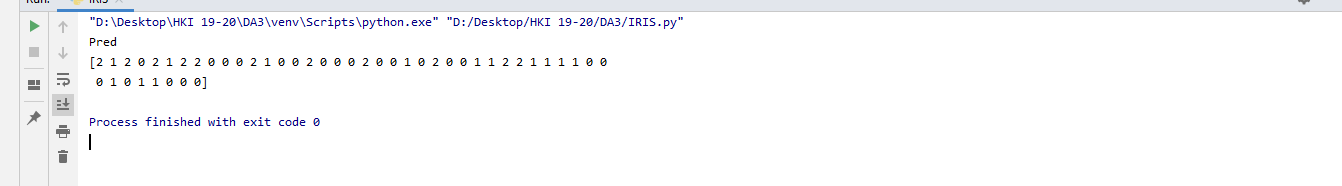


### **Dùng Decision tree cho tập huấn luyện**

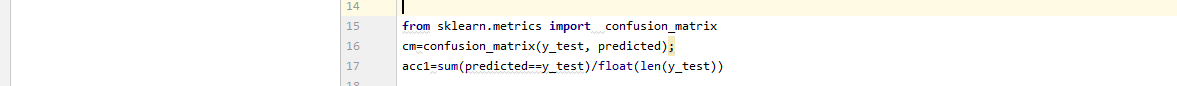


### **Dự đoán kết quả**



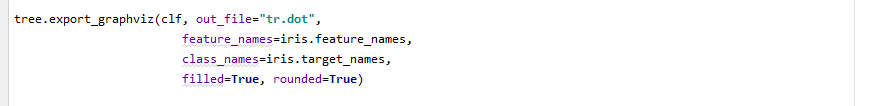


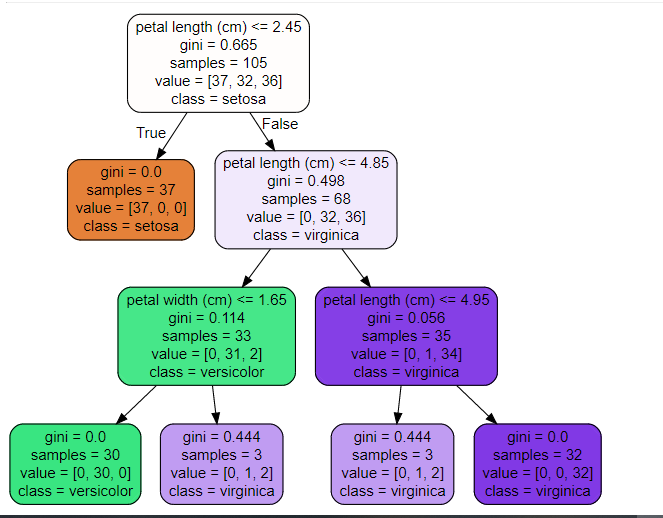
### **Ước tính bao nhiêu trường hợp sai (tạo ma trận nhầm lẫn)**



### **Mô hình cây quyết định**

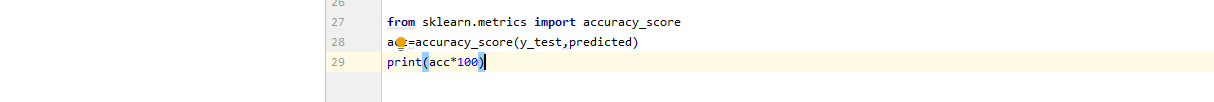
* Tạo ra file .dot sau đó dùng <http://www.webgraphviz.com/> để chuyển thành dạng lưu đồ

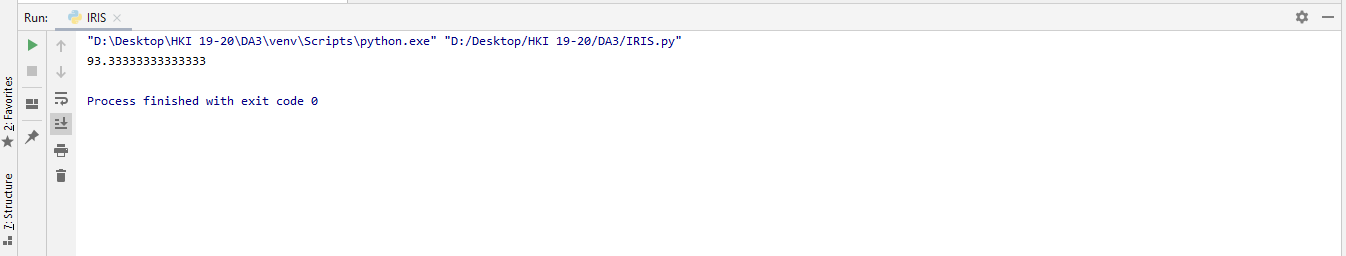




* Khi đó ta sẽ dễ dàng xác định được nhãn của một dữ liệu mới thông qua cây quyết định

### **Đánh giá**





* Độ chính xác: 93.33%

# **TÀI LIỆU THAM KHẢO**

* <https://medium.com/@thanhleo92/gi%E1%BB%9Bi-thi%E1%BB%87u-v%E1%BB%81-machine-learning-92bfcdc4bf4e>
* <https://trituenhantao.io/tin-tuc/decision-tree/>
* <https://en.wikipedia.org/wiki/Gradient_boosting>
* <http://rstudio-pubs-static.s3.amazonaws.com/402681_e26092f7a0b64104a9ecf931ed1f96aa.html>
* <https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/>
* <https://codetudau.com/phan-tich-cay-quyet-dinh-voi-scikit-learn/index.html>