TRƯỜNG ĐẠI HỌC VĂN LANG

KHOA CÔNG NGHỆ THÔNG TIN

A picture containing text, clipart

Description automatically generated

TIỂU LUẬN

HỌC MÁY VÀ ỨNG DỤNG

ĐỀ TÀI :

ÁP DỤNG THUẬT TOÁN RANDOM FOREST TRONG NHẬN DIỆN PHÂN LOẠI RƯỢU VANG ĐỎ

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:**  Sinh viên thực hiện: | TS PHÙNG THẾ BẢO  **1 Nguyễn Thành Đạt...2174802010644**  **2 Đậu Thiện Tường ....2174802010639**  **3 Nguyễn Hữu Vũ.......2174802010641**  4 Nguyễn Huy Hiệu…2174802010048  5 Trương Triều Vinh...2174802010060 |

**MỤC LỤC**

**[LỜI NÓI ĐẦU.](#_Toc150978772)** [1](#_Toc150978772)

**[CHƯƠNG 1: GIỚI THIỆU, MỤC TIÊU VÀ NHIỆM VỤ BÀI TOÁN.](#_Toc150978773)** [2](#_Toc150978773)

**[1.1: Giới thiệu bài toán.](#_Toc150978774)** [2](#_Toc150978774)

**[1.2: Mục tiêu và nhiệm vụ nghiên cứu](#_Toc150978775)** [2](#_Toc150978775)

**[CHƯƠNG 2: MÔ TẢ THUẬT TOÁN RANDOM FOREST](#_Toc150978776)** [3](#_Toc150978776)

**[2.1: Giới thiệu về Thuật toán Random Forest](#_Toc150978777)** [3](#_Toc150978777)

**[2.2: Mô tả thuật toán , ví dụ minh họa](#_Toc150978782)** [6](#_Toc150978782)

**[2.3 Hoạt động của thuật toán Random Forest:](#_Toc150978783)** [7](#_Toc150978783)

**[2.4 Xây dựng thuật toán Random Forest:](#_Toc150978784)**  [10](#_Toc150978784)

**[CHƯƠNG 3: XÂY DỰNG DỰA TRÊN THUẬT TOÁN](#_Toc150978785)** [11](#_Toc150978785)

**[3.1: Cài đặt chương trình](#_Toc150978786)** [11](#_Toc150978786)

**[CHƯƠNG 4: ĐÁNH GIÁ THỰC NGHIỆM CHƯƠNG TRÌNH.](#_Toc150978787)** [11](#_Toc150978787)

**[TÀI LIỆU THAM KHẢO](#_Toc150978788)** [11](#_Toc150978788)

# **LỜI NÓI ĐẦU.**

Lời đầu tiên, nhóm chúng tôi xin gửi lời cảm ơn chân thành đến thầy TS. Phùng Thế Bảo đã dành thời gian hướng dẫn, chỉ dạy cả về chuyên môn cũng như các định hướng, kiến thức về môn học. Vì kiến thức còn hạn hẹp nên trong quá trình thực hiện đề tài không thể tránh khỏi những thiếu sót. Vì vậy, rất mong nhận được sự góp ý từ thầy và các bạn để đề tài ngày càng hoàn thiện hơn và hoạt động tốt hơn.

Trí tuệ nhân tạo (Articifical Intelligence) là một ngành khoa học ngày càng phát triển mạnh mẽ và có ứng dụng rộng rãi trong cuộc sống. Rõ ràng máy tính với độ thông minh nhất định ảnh hưởng không nhỏ đến cuộc sống ngày nay và trong tương lai phát triển của văn minh nhân loại.

Hiện nay, việc dùng rượu vang trong những bữa tiệc lớn đang rất phổ biến. Tuy vậy để có thể biết được những chai rượu vang có chất lượng tốt hay không thì cần phảI qua những qui trình nghiêm ngặt. Vì thế chúng tôi đã tìm hiểu cách nhận diện phân loạI ra rượu vang chất lượng cho bài tiểu luận môn Học máy và Ứng dụng.

Trong quá trình hoàn thành đề tài này, nhóm chúng tôi đã tìm hiểu thêm được thêm nhiều thuật toán đã được trong và đưa ra quyết định chọn thuật toán Random Forest cũng như các kỹ năng lập trình ngôn ngữ Python, cùng với kỹ năng làm việc nhóm để có thể hoàn thành bài tiểu luận này.

# **CHƯƠNG 1: GIỚI THIỆU, MỤC TIÊU VÀ NHIỆM VỤ BÀI TOÁN.**

## **1.1: Giới thiệu bài toán.**

- Random forest là thuật toán supervised learning, có thể giải quyết cả bài toán regression và classification. Là một thuật toán học máy được sử dụng rộng rãi được phát triển bởi Leo Breiman và Adele Cutler, kết hợp đầu ra của nhiều cây quyết định để đạt được một kết quả duy nhất. Tính dễ sử dụng và linh hoạt của nó đã thúc đẩy việc áp dụng nó, vì nó xử lý cả vấn đề phân loại và hồi quy.

## **1.2: Mục tiêu và nhiệm vụ nghiên cứu**

**\* Mục tiêu:**

- Mục tiêu đối với nhóm là xây dựng thành công bài toán cho chương trình đáp ứng được các yêu cầu và mục tiêu đưa ra ban đầu trong việc phân loại rượu vang. Để làm được người này, nhóm xây dựng cần nắm chắc và hiểu rõ được thuật toán rừng ngẫu nhiên và áp dụng nó một cách thành công trong việc tạo ra chương trình giúp nhận diện phân loại rượu vang.

**\* Nhiệm vụ của người xây dựng chương trình:**

- Trong bối cảnh phân loại rượu vang, nhiệm vụ của người xây dựng một chương trình áp dụng thuật toán Random Forest để phân loại rượu vang sẽ là huấn luyện thuật toán trên một tập dữ liệu các mẫu rượu vang với các đặc điểm đã biết và các loại rượu vang tương ứng của chúng. Sau đó, thuật toán sẽ sử dụng dữ liệu huấn luyện này để xây dựng nhiều cây quyết định, mỗi cây sẽ dự đoán loại rượu vang của một mẫu cụ thể dựa trên các đặc điểm của nó. Dự đoán cuối cùng sẽ được thực hiện bằng cách tổng hợp các dự đoán của tất cả các cây quyết định .

# **CHƯƠNG 2: MÔ TẢ THUẬT TOÁN RANDOM FOREST**

## **2.1: Giới thiệu về Thuật toán Random Forest**

**\*Thuật toán cây quyết định (Decision tree)**

- Trước khi vào thuật toán Random Forest chúng ta hãy xem lại thuật toán cây quyết định, bởi vì ở thuật toán Random Forest chúng ta sẽ xây dựng nhiều cây quyết định bằng thuật toán cây quyết địng (Decision Tree).

- Cây quyết định ([Decision Tree](https://trituenhantao.io/tu-dien-thuat-ngu/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.Thuật toán này sử dụng các khái niệm như mức tăng thông tin, entropy và chỉ số Gini để xác định các quy tắc có độ tin cậy cao nhất cho việc phân loại hoặc dự đoán dữ liệu, mỗi nhánh của lá cây đại diện cho một suy luận hổ trọ quyết định của thuật toán. 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 được áp dụng vào cả 2 bài toán: Phân loại (Classification) và Hồi quy (Regression).

**\*Thuật toán Rừng ngẫu nhiên (Random Forest)**

- Random Forest là một thuật toán học máy phổ biến thuộc về kỹ thuật học có giám sát. Nó có thể được sử dụng cho cả vấn đề Phân loại và Hồi quy trong ML. Nó dựa trên khái niệm học tập theo nhóm, là một quá trình kết hợp nhiều bộ phân loại để giải quyết một vấn đề phức tạp và để cải thiện hiệu suất của mô hình.

- Như tên cho thấy, “Rừng ngẫu nhiên là một bộ phân loại chứa một số cây quyết định trên các tập con khác nhau của tập dữ liệu đã cho và lấy giá trị trung bình để cải thiện độ chính xác dự đoán của tập dữ liệu đó.” Thay vì dựa vào một cây quyết định, rừng ngẫu nhiên lấy dự đoán từ mỗi cây và dựa trên đa số phiếu dự đoán, và nó dự đoán kết quả cuối cùng. Số lượng cây lớn hơn trong rừng dẫn đến độ chính xác cao hơn và ngăn ngừa vấn đề trang bị quá mức.

- Sơ đồ dưới đây giải thích hoạt động của thuật toán Rừng ngẫu nhiên:



Kết hợp nhiều cây quyết định ra rừng ngẫu nhiên

**\*Sự khác biệt giữa cây quyết định và rừng ngẫu nhiên**

- Mặc dù một khu rừng ngẫu nhiên là một tập hợp các cây quyết định, có một số khác biệt.

- Nếu nhập một tập dữ liệu đào tạo với các tính năng và nhãn vào cây quyết định, nó sẽ hình thành một số bộ quy tắc, sẽ được sử dụng để đưa ra các dự đoán.

- Ví dụ: để dự đoán liệu một người có nhấp vào quảng cáo trực tuyến hay không, bạn có thể thu thập các quảng cáo mà người đó đã nhấp vào trong quá khứ và một số tính năng mô tả quyết định của họ. Nếu bạn đặt các tính năng và nhãn vào cây quyết định, nó sẽ tạo ra một số quy tắc giúp dự đoán liệu quảng cáo có được nhấp vào hay không. Để so sánh, thuật toán rừng ngẫu nhiên chọn ngẫu nhiên các quan sát và đặc điểm để xây dựng một số cây quyết định và sau đó tính trung bình các kết quả.

- Một sự khác biệt khác là cây quyết định “sâu” có thể bị quá mức. Hầu hết thời gian, rừng ngẫu nhiên ngăn chặn điều này bằng cách tạo các tập hợp con ngẫu nhiên của các đối tượng địa lý và xây dựng các cây nhỏ hơn bằng cách sử dụng các tập hợp con đó. Sau đó, nó kết hợp các cây con. Điều quan trọng cần lưu ý là điều này không phải lúc nào cũng hoạt động và nó cũng làm cho quá trình tính toán chậm hơn, tùy thuộc vào số lượng cây mà khu rừng ngẫu nhiên xây dựng.

## **\*Các ứng dụng của Rừng ngẫu nhiên**

## - Chủ yếu có bốn lĩnh vực mà Rừng ngẫu nhiên chủ yếu được sử dụng:

**+ Ngân hàng:** Lĩnh vực ngân hàng chủ yếu sử dụng thuật toán này để xác định rủi ro cho vay.

**+ Y học:** Với sự trợ giúp của thuật toán này, các xu hướng bệnh tật và nguy cơ của bệnh có thể được xác định.

**+ Sử dụng đất:** Chúng tôi có thể xác định các khu vực sử dụng đất tương tự bằng thuật toán này.

**+ Tiếp thị:** Các xu hướng tiếp thị có thể được xác định bằng cách sử dụng thuật toán này.

## **\*Ưu điểm của Rừng ngẫu nhiên**

- Random Forest có khả năng thực hiện cả hai nhiệm vụ Phân loại và Hồi quy.

- Có khả năng xử lý các tập dữ liệu lớn với kích thước cao.

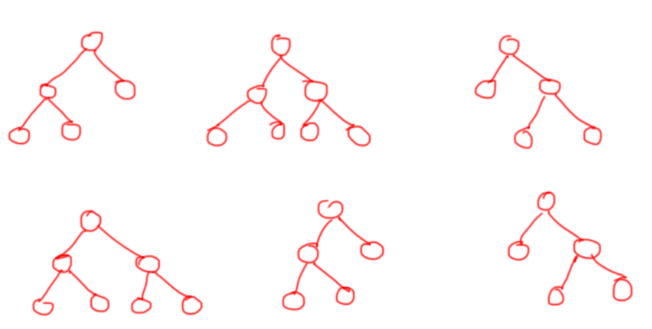
- Nâng cao độ chính xác của mô hình và ngăn chặn vấn đề trang bị quá mức.

## **\*Nhược điểm của Rừng ngẫu nhiên**

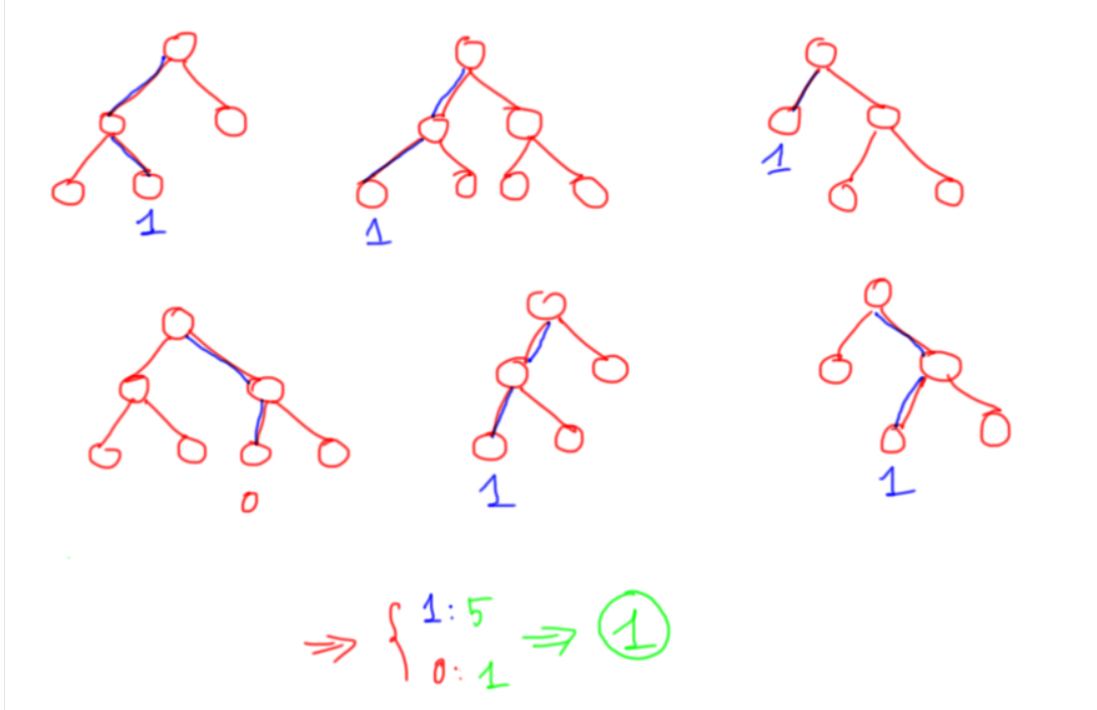
- Mặc dù rừng ngẫu nhiên có thể được sử dụng cho cả nhiệm vụ phân loại và hồi quy, nó không phù hợp hơn cho các nhiệm vụ Hồi quy.

## **2.2: Mô tả thuật toán , ví dụ minh họa**

- Ở bước huấn luyện, chúng ta sẽ xây dựng nhiều cây quyết định, các cây quyết định có thể khác nhau.



Sau đó ở bước dự đoán, với một dữ liệu mới, thì ở mỗi cây quyết định mình sẽ đi từ trên xuống theo các node điều kiện để được các dự đoán, sau đó kết quả cuối cùng được tổng hợp từ kết quả của các cây quyết định.



- Ví dụ như trên, thuật toán Random Forest có 6 cây quyết định, 5 cây dự đoán 1 và 1 cây dự đoán 0, do đó mình sẽ vote là cho ra dự đoán cuối cùng là 1.

# **2.3 Hoạt động của thuật toán Random Forest:**

- Trước khi hiểu hoạt động của thuật toán rừng ngẫu nhiên trong học máy, chúng ta phải xem xét kỹ thuật học tập tổng hợp. Ensemble đơn giản có nghĩa là kết hợp nhiều mô hình. Do đó, một tập hợp các mô hình được sử dụng để đưa ra dự đoán chứ không phải là một mô hình riêng lẻ.

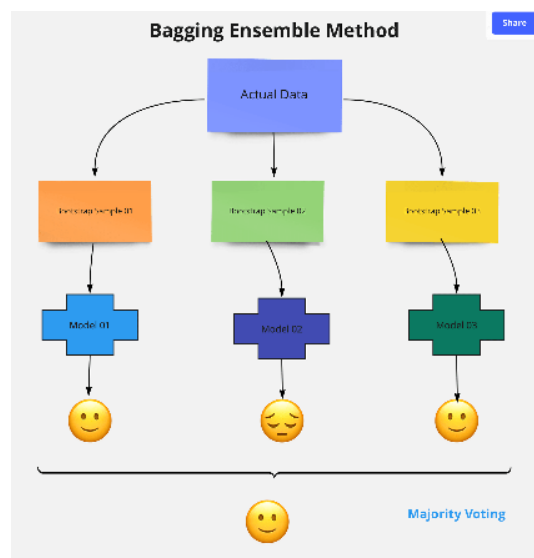
- Ensemble sử dụng hai loại phương pháp:

+ Bagging: Nó tạo ra một tập hợp con đào tạo khác với dữ liệu đào tạo mẫu với sự thay thế &; kết quả cuối cùng dựa trên bỏ phiếu đa số.

+ Boosting: Nó kết hợp những người học yếu thành những người học mạnh bằng cách tạo ra các mô hình tuần tự sao cho mô hình cuối cùng có độ chính xác cao nhất. Ví dụ: ADA BOOST, XG BOOST.

**\*Bagging**

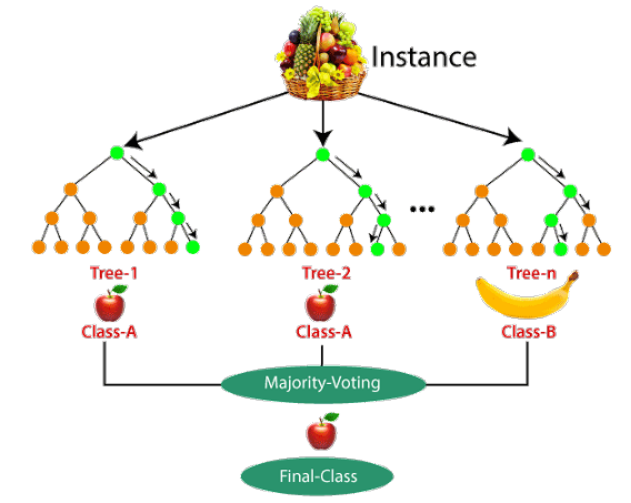
- Bagging:còn được gọi là Bootstrap Aggregation, đóng vai trò là kỹ thuật tổng hợp trong thuật toán Random Forest. Dưới đây là các bước liên quan đến đóng bao:  
 **+** **Selection of Subset**: Việc đóng gói bắt đầu bằng cách chọn một mẫu ngẫu nhiên hoặc tập hợp con từ toàn bộ tập dữ liệu.  
 **+** **Bootstrap Sampling**: Mỗi mô hình sau đó được tạo từ các mẫu này, được gọi là Bootstrap Samples, được lấy từ dữ liệu gốc để thay thế. Quá trình này được gọi là lấy mẫu hàng.  
 **+** **Bootstrapping**: Bước lấy mẫu hàng với sự thay thế được gọi là bootstrapping.  
 **+ Independent Model Training**: Mỗi mô hình được đào tạo độc lập trên Bootstrap Samples tương ứng của nó. Quá trình đào tạo này tạo ra kết quả cho từng mô hình.  
 **+** **Majority Voting**: Kết quả cuối cùng được xác định bằng cách kết hợp kết quả của tất cả các mô hình thông qua bỏ phiếu đa số. Kết quả dự đoán phổ biến nhất trong số các mô hình được chọn.  
 **+** **Aggregation**: Bước này, bao gồm kết hợp tất cả các kết quả và tạo ra kết quả cuối cùng dựa trên bỏ phiếu đa số, được gọi là tổng hợp.



**\*Boosting**

- Boosting: Một trong những kỹ thuật sử dụng khái niệm học tập đồng bộ. Một thuật toán tăng cường kết hợp nhiều mô hình đơn giản (còn được gọi là người học yếu hoặc người ước tính cơ sở) để tạo ra đầu ra cuối cùng. Nó được thực hiện bằng cách xây dựng một mô hình bằng cách sử dụng các mô hình yếu trong chuỗi.

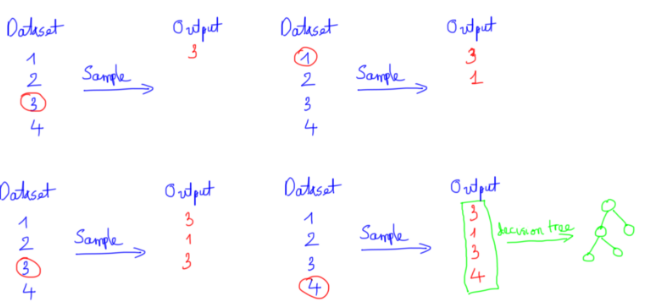
**\*Các bước liên quan đến thuật toán Random Forest:**  
 - Bước 1: Trong mô hình Random Forest, một tập hợp con các điểm dữ liệu và một tập hợp con các tính năng được chọn để xây dựng mỗi cây quyết định. Nói một cách đơn giản, n bản ghi ngẫu nhiên và m tính năng được lấy từ tập dữ liệu có k số bản ghi.  
 - Bước 2: Cây quyết định riêng lẻ được xây dựng cho mỗi mẫu.  
 - Bước 3: Mỗi cây quyết định sẽ tạo ra một đầu ra.  
 - Bước 4: Kết quả cuối cùng được xem xét dựa trên Bỏ phiếu đa số hoặc Trung bình để phân loại và hồi quy, tương ứng.



# **2.4 Xây dựng thuật toán Random Forest:** - Giả sử bộ dữ liệu của mình có n dữ liệu (sample) và mỗi dữ liệu có d thuộc tính (feature). - Để xây dựng mỗi cây quyết định mình sẽ làm như sau: + Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật [Bootstrapping](https://en.wikipedia.org/wiki/Bootstrapping_(statistics)), hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau. + Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính (k < n). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính. + Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

- Do quá trính xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

- Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.



# **CHƯƠNG 3: XÂY DỰNG DỰA TRÊN THUẬT TOÁN**

## **3.1: Cài đặt chương trình**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import os

from pprint import pprint

dataFrame = pd.read\_csv(os.path.join(os.path.dirname(\_\_file\_\_), "winequality-red.csv"))

*#~ In ra 5 hàng đầu tiên của DataFrame bằng dataFrame.head().*

print("     "\*14+"----- In ra 5 hàng đầu tiên -----")

print(dataFrame.head())

print("")

*#~ Tính các thống kê mô tả cơ bản cho từng cột trong dữ liệu*

print("     "\*8+"----- Thống kê mô tả cơ bản -----")

print(dataFrame.describe().T)

print("")

*#~ Kiểm tra và hiển thị số lượng giá trị thiếu (missing values) trong DataFrame `dataFrame`.*

print("     "\*9+"----- Kiểm tra và hiển thị số lượng giá trị thiếu -----")

print(dataFrame.isnull().sum())

print("")

from scipy import stats

*#~ Tính z-score cho mỗi giá trị trong DataFrame `dataFrame`*

print("     "\*11+"----- Tính z-score -----")

z = stats.zscore(dataFrame)

z.to\_csv('Machine\_Learning\_Project/Z-score.csv')

print(z)

print("")

*#~ Loại bỏ các ngoại lai*

print("     "\*11+"----- Loại bỏ các ngoại lai dựa trên z-score -----")

threshold = 3

print(np.where(z > 3))

dataFrame\_o = dataFrame[(z < 3).all(*axis*=1)]

print(dataFrame.shape)

dataFrame\_o.to\_csv('Machine\_Learning\_Project/new\_dataFrame.csv')

print(dataFrame\_o.shape)

print("")

from sklearn.model\_selection import train\_test\_split

*#~ Chuẩn bị dữ liệu để huấn luyện*

print("     "\*11+"----- Chuẩn bị dữ liệu -----")

X = dataFrame\_o.drop(*columns* = 'chất lượng')

y = dataFrame\_o['chất lượng']

print(X.head())

print("")

print(y.head())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,*test\_size*=0.2)

from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(*n\_estimators*=100)

clf.fit(X\_train,y\_train)

y\_pred = clf.predict(X\_test)

from sklearn import metrics

print('Accuracy: ', metrics.accuracy\_score(y\_test,y\_pred))

import pandas as pd

feature\_imp = pd.Series(clf.feature\_importances\_, *index*=dataFrame\_o.columns[:11]).sort\_values(*ascending*=False)

print(feature\_imp)

*#~ Trực quan hóa đặc trưng quan trọng*

import seaborn as sns

sns.barplot(*x*=feature\_imp, *y*=feature\_imp.index, *palette*="hls")

plt.xlabel('Điểm số đặc trưng quan trọng')

plt.ylabel('Đặc trưng')

plt.title("Trực quan hóa đặc trưng quan trọng")

*#// plt.legend()*

plt.show()

*#~ Tinh chỉnh bằng RandomSearchCrossValidation*

from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(*random\_state* = 6)

*# Thông số được sử dụng*

print("     "\*11+'----- Các thông số đang được sử dụng -----')

pprint(rf.get\_params())

print("")

*#~ Thiết lập tham số*

from sklearn.model\_selection import RandomizedSearchCV

*# Số lượng cây trong rừng*

n\_estimators = [int(x) for x in np.linspace(*start* = 200, *stop* = 2000, *num* = 10)]

*# Số lượng đặc trưng mà mô hình sẽ cân nhắc khi xây dựng cây*

max\_features = ['auto', 'sqrt']

*# Độ sâu tối đa của cây quyết định*

max\_depth = [int(x) for x in np.linspace(2, 14, *num* = 7)]

max\_depth.append(None)

*# Số mẫu tối thiểu cần thiết để chia một nút thành hai nút con*

min\_samples\_split = [2, 5, 10]

*# Số mẫu tối thiểu cần thiết để tạo một lá*

min\_samples\_leaf = [1, 2, 4]

*# Xác định mô hình có sử dụng Bootstrap Sampling hay không*

bootstrap = [True, False]

random\_grid = {'n\_estimators': n\_estimators,

               'max\_features': max\_features,

               'max\_depth': max\_depth,

               'min\_samples\_split': min\_samples\_split,

               'min\_samples\_leaf': min\_samples\_leaf,

               'bootstrap': bootstrap}

print("     "\*11+'----- Thiết lập thông số -----')

pprint(random\_grid)

*#~ Chọn ra các tỗ hợp tham số tốt nhất*

*# Use the random grid to search for best hyperparameters*

*# First create the base model to tune*

rf = RandomForestRegressor()

*# Random search of parameters, using 3 fold cross validation,*

*# search across 100 different combinations, and use all available cores*

rf\_random = RandomizedSearchCV(*estimator* = rf, *param\_distributions* = random\_grid, *n\_iter* = 100, *cv* = 3, *verbose*=2, *random\_state*=6, *n\_jobs* = -1)

*# Fit the random search model*

rf\_random.fit(X\_train,y\_train)

print("     "\*11+'----- Tổ hợp tham số tốt nhất -----')

pprint(rf\_random.best\_params\_)

def evaluate(*model*, *X\_test*, *y\_test*):

    predictions = *model*.predict(*X\_test*)

    errors = abs(predictions - *y\_test*)

    mape = 100 \* np.mean(errors / *y\_test*)

    accuracy = 100 - mape

    print('Hiệu suất của model')

    print('Giá trị trung bình của sai số: {:0.4f} degrees.'.format(np.mean(errors)))

    print('Độ chính xác = {:0.2f}%.'.format(accuracy))

    return accuracy

base\_model = RandomForestRegressor(*n\_estimators* = 10, *random\_state* = 6)

base\_model.fit(X\_train, y\_train)

base\_accuracy = evaluate(base\_model, X\_test,y\_test)

best\_random = rf\_random.best\_estimator\_

random\_accuracy = evaluate(best\_random, X\_test, y\_test)

print('Mức độ cải thiện {:0.2f}%.'.format( 100 \* (random\_accuracy - base\_accuracy) / base\_accuracy))

*#~ Thực hiện tìm kiếm siêu tham số*

*#Grid Search with Cross Validation*

from sklearn.model\_selection import GridSearchCV

*# Create the parameter grid based on the results of random search*

param\_grid = {

    'bootstrap': [True],

    'max\_depth': [8, 10, 12, 14],

    'max\_features': [2, 3],

    'min\_samples\_leaf': [3, 4, 5],

    'min\_samples\_split': [8, 10, 12],

    'n\_estimators': [100, 200, 300, 1000]

}

*# Create a based model*

rf = RandomForestRegressor()

*# Instantiate the grid search model*

grid\_search = GridSearchCV(*estimator* = rf, *param\_grid* = param\_grid,

*cv* = 3, *n\_jobs* = -1, *verbose* = 2)

*# Fit the grid search to the data*

grid\_search.fit(X\_train, y\_train)

grid\_search.best\_params\_

best\_grid = grid\_search.best\_estimator\_

pprint(best\_grid)

grid\_accuracy = evaluate(best\_grid, X\_test, y\_test)

*# Đọc dữ liệu bên ngoài*

external\_data = pd.read\_csv(os.path.join(os.path.dirname(\_\_file\_\_), "external\_wine.csv"))  *# Thay đổi đường dẫn tới file CSV của dữ liệu bên ngoài*

*# Hiển thị 5 hàng đầu tiên của dữ liệu bên ngoài*

print("     "\*14+"----- Dữ liệu bên ngoài -----")

print(external\_data.head())

print("")

*# Loại bỏ các cột không cần thiết (nếu có)*

*# Chắc chắn rằng cấu trúc của dữ liệu bên ngoài giống với dữ liệu huấn luyện*

external\_data = external\_data.drop(*columns*='column\_to\_drop', *errors*='ignore')

*# Chuẩn bị dữ liệu để dự đoán*

X\_external = external\_data.drop(*columns*='chất lượng', *errors*='ignore')

*# Dự đoán chất lượng bằng mô hình đã huấn luyện*

external\_predictions = best\_grid.predict(X\_external)

*# Hiển thị kết quả dự đoán cho dữ liệu bên ngoài*

print("     "\*14+"----- Kết quả dự đoán cho dữ liệu bên ngoài -----")

print(external\_predictions)

*# Thêm cột dự đoán vào DataFrame của dữ liệu bên ngoài*

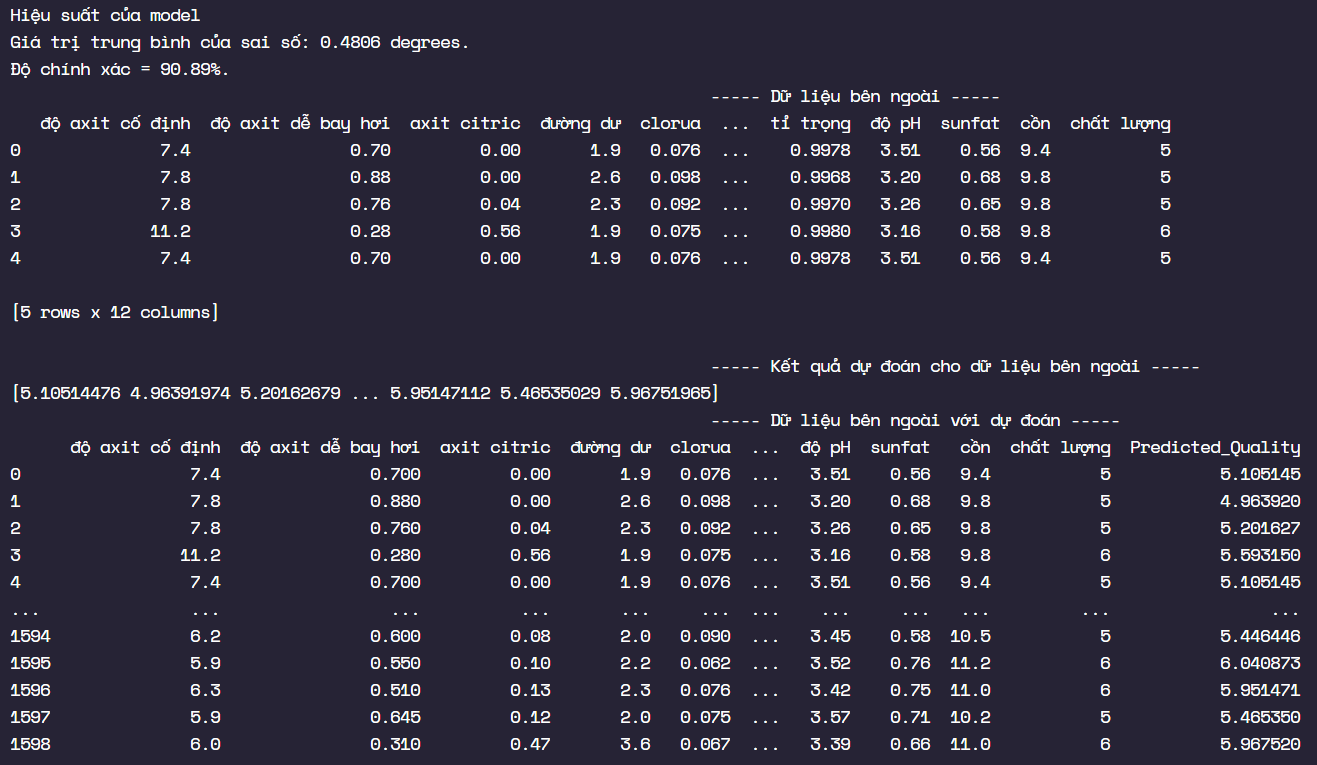
external\_data['Dự đoán chất lượng'] = external\_predictions

*# Hiển thị dữ liệu bên ngoài với cột dự đoán*

print("     "\*14+"----- Dữ liệu bên ngoài với dự đoán -----")

print(external\_data)

# **CHƯƠNG 4: ĐÁNH GIÁ THỰC NGHIỆM CHƯƠNG TRÌNH.**



# **TÀI LIỆU THAM KHẢO**

[Random Forest algorithm — Machine Learning cho dữ liệu dạng bảng (machinelearningcoban.com)](https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html)

[Thuật toán rừng ngẫu nhiên - Hướng dẫn toàn diện với các ví dụ (analyticsvidhya.com)](https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/" \l "h-what-is-random-forest-algorithm)

[Red Wine Quality (kaggle.com)](https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009/code)

**BẢNG PHÂN CÔNG NHIỆM VỤ THỰC HIỆN ĐỐI VỚI ĐỀ TÀI**

|  |  |  |
| --- | --- | --- |
| **TT** | **Họ và tên** | **Nội dung thực hiện** |
| 1 | Nguyễn Thành Đạt | Cài đặt chương trình, giao diện ứng dụng, chỉnh sửa file code, hoàn thiện file word. |
| 2 | Đậu Thiện Tường | Lời mở đầu, giới thiệu bài toán, đánh giá thực nghiệm chương trình, làm file powerpoint. |
| 3 | Nguyễn Hữu Vũ | Mô tả thuật toán Random Forest, phân tích, xây dựng bài toán và mô tả các bước thực hiện, tài liệu tham khảo. |
| 4 | Trương Triều Vinh | Mục tiêu, nhiệm vụ nghiên cứu, đánh giá thực nghiệm chương trình, tổng hợp nội dung powerpoint. |
| 5 | Nguyễn Huy Hiệu | đánh giá thực nghiệm chương trình, tổng hợp nội dung powerpoint. chỉnh sửa |