|  |  |  |
| --- | --- | --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  **VIỆN ĐIỆN TỬ - VIỄN THÔNG**  logo_128  **BÁO CÁO**  **THỰC TẬP TỐT NGHIỆP**  **ĐỀ TÀI :**  **TÌM HIỂU CÁC MÔ HÌNH HỌC MÁY VÀ**  **PHÂN TÍCH BỘ DỮ LIỆU CƠ BẢN**  Giảng viên hướng dẫn: TS. Trần Thị Ngọc Lan  Sinh viên thực hiện:   |  |  | | --- | --- | | Lê Thành Long | 20172669 |   Hà Nội, 8-2022 |

**MỤC LỤC**

[Lời mở đầu 3](#_Toc110279803)

[Phần I: Linear Regression với Numpy và Tensorflow 4](#_Toc110279804)

[1. Linear Regressison 4](#_Toc110279805)

[1.1 Thực hiện thuật toán bằng Numpy 5](#_Toc110279806)

[1.2 Thực hiện bằng Tensorflow 9](#_Toc110279807)

[Phần II: Thực hiện phân tích trên bộ dữ liệu 11](#_Toc110279808)

[Chương 1: TỔNG QUAN VỀ ĐỀ TÀI 11](#_Toc110279809)

[Chương 2: LÝ THUYẾT CHUNG 12](#_Toc110279810)

[2.1 Quy trình cơ bản để phân tích một bộ dữ liệu 12](#_Toc110279811)

[2.2 Giá trị R2 13](#_Toc110279812)

[Chương 3: CÁC BƯỚC THỰC HIỆN 14](#_Toc110279813)

[3.1 Data Collection 14](#_Toc110279814)

[3.2 Data Preprocessing 15](#_Toc110279815)

[3.3 Training Model 16](#_Toc110279816)

[3.4 Tối ưu mô hình 23](#_Toc110279817)

[3.5 Evaluation 27](#_Toc110279818)

[Phần III: KẾT LUẬN 28](#_Toc110279819)

[Tài liệu tham khảo 29](#_Toc110279820)

**Danh mục hình ảnh**

[Hình 1 Mô tả thuật toán Gradient Descent 7](#_Toc110258597)

[Hình 2 Dữ liệu dự đoán so với dữ liệu tập test 8](#_Toc110258598)

[Hình 3 Dữ liệu dự đoán so với tập test khi sử dụng tensorflow 9](#_Toc110258599)

[Hình 4 Quy trình phân tích một bộ data 12](#_Toc110258600)

[Hình 5 So sánh giữa giá trị thực và giá trị dự đoán bằng Decision Tree Regression 20](#_Toc110258601)

[Hình 6 Nguyên lý hoạt động Random Forest Regression 21](#_Toc110258602)

[Hình 7 So sánh dữ liệu dự đoán với dữ liệu thực bằng Random Forest Regression 23](#_Toc110258603)

[Hình 8 So sánh dữ liệu dự đoán đã tối ưu với dữ liệu thực DTR 25](#_Toc110258604)

[Hình 9 So sánh dữ liệu dự đoán đã tối ưu với dữ liệu thực bằng Random Forest Regression 27](#_Toc110258605)

# Lời mở đầu

Trong cuôc sống hiện nay, trí tuệ nhân tạo đang ngày càng phát triển và có vai trò quan trọng đối với xã hội hiện đại. Trí tuệ nhân tạo được ứng dụng trong rất nhiều hoạt động và lĩnh vực khác nhau như nghiên cứu, tính toán và mô phỏng, hỗ trợ chẩn đoán bệnh, phân tích dữ liệu và nhận diện hình ảnh,… Qua đó chúng ta thấy được sự phổ biến và phát triển không ngừng của trí tuệ nhân tạo. Vì vậy trong báo cáo thực tập tốt nghiệp này, em muốn tìm hiểu về cách xử lý một bộ dữ liệu, và cụ thể là bộ dữ liệu về dự đoán công suất sưởi ấm và công suất làm lạnh của căn hộ, từ đó có thể áp dụng vào thực tế công việc của em sau này.

Được tạo điều kiện cũng như nhờ sự giúp đỡ của cô Trần Thị Ngọc Lan đã giúp cho em là sinh viên đang ngồi trên ghế nhà trường có cơ hội thực hiện đề tài về học máy này

Trong bài báo cáo, em sẽ trình bày về những gì đã làm được gồm các phần sau:

* Phần I : Linear Regression với Numpy và Tensorflow
* Phần II : Phân tích bộ dữ liệu năng lượng sử dụng cho căn hộ

# Phần I: Linear Regression với Numpy và Tensorflow

## Linear Regressison

#### Giới thiệu về ‘Hồi quy tuyến tính’

Linear Regression là một thuật toán Supervised learning, cũng là một trong những thuật toán cơ bản nhất của Machine Learning. Thuật toán cũng đôi khi được gọi là Linear Fiting(trong thống kê) hoặc Linear Least Square.

Linear Regression (hồi quy tuyến tính) là một trong những thuật toán cơ bản nhất của Machine Learning. thuật toán Linear Regression (Hồi Quy Tuyến Tính) thuộc nhóm Supervised learning ( Học có giám sát ).

"Hồi quy tuyến tính" là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục. Ví dụ, dự đoán giao thông ở một cửa hàng bán lẻ, dự đoán thời gian người dùng dừng lại một trang nào đó hoặc số trang đã truy cập vào một website nào đó v.v...

Trong khi sử dụng hồi quy tuyến tính, mục tiêu của chúng ta là để làm sao một đường thẳng có thể tạo được sự phân bố gần nhất với hầu hết các điểm. Do đó làm giảm khoảng cách (sai số) của các điểm dữ liệu cho đến đường đó.

Đường hồi quy tuyến tính - Trong khi sử dụng hồi quy tuyến tính, mục tiêu của chúng ta là để làm sao một đường thẳng có thể tạo được sự phân bố gần nhất với hầu hết các điểm. Do đó làm giảm khoảng cách (sai số) của các điểm dữ liệu cho đến đường đó.

Nếu chúng ta có nhiều hơn một biến độc lập, phương pháp phù hợp nhất là "Multiple Regression Linear" - Hồi quy tuyến tính đa biến

***Bộ dữ liệu sử dụng:***

https://www.kaggle.com/c/boston-housing

### 1.1 Thực hiện thuật toán bằng Numpy

#### Viết hàm Cost Function

***Dạng tổng***

Đối với bài toán có 2 features đầu vào, khai triển, ta có kết quả:

*Với m là số các giá trị nonNull trong 1 feature*

*ω1, ω2 là các giá trị trọng số weight*

*x1, x2 là các giá trị đối số của từng feature*

*b là giá trị bias*

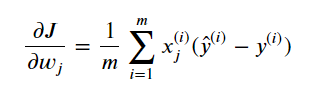
*y là giá trị thực bộ data ứng với từng dự đoán*

***Dạng ma trận***

#### Tính vector gradient của w

***Dạng tổng***

Với i = [1,m] và j = [1,n] ta xét:



|  |
| --- |
| def np\_grad\_fn(w, X, y):  grad\_w = np.zeros(np.array(w).shape)  m = X.shape[0]  y\_hat = np.dot(X, w)  for j in range(grad\_w.shape[0]):  for i in range(m):  grad\_w[j] += (y\_hat[i] - y[i]) \* X[i, j]/m  return grad\_w  # Đưa vào ma trận w=[0,0,0]  # np\_grad\_fn(w,train\_X\_new,train\_y) |

*Định nghĩa ma trận train\_X\_new:*

- Là ma trận có thêm so với ma trận train\_X một cột có tất cả giá trị =1 để làm đối số cho bias

***Dạng ma trận***

Schematic

Description automatically generated with low confidence

|  |
| --- |
| def np\_grad\_fn\_vectorized(w, X, y):    m, n = X.shape  y\_hat = np.dot(X, w)  grad\_w = np.dot(X.T, y\_hat - y)/m    return grad\_w |

Với cùng giá trị train\_X\_new và train\_y đưa vào thì cả 2 dạng gradient trả về một bộ weight + bias giống nhau

#### Tính dựa trên Gradient Descent

Có thể tính giá trị w xấp xỉ như sau

Text

Description automatically generated

Với alpha là hệ số learning rate của gradient descent, alpha lớn thì hàm có thể không đạt được mức tối thiểu, alpha nhỏ thì chương trình chạy chậm, do đó cần chọn một hệ số alpha hợp lý

**Diagram

Description automatically generated**

Hình 1 Mô tả thuật toán Gradient Descent

|  |
| --- |
| **def np\_solve\_via\_gradient\_descent(X,y,print\_every=5000,niter=100000, alpha=0.005):**  **m, n = X.shape**  **w = np.zeros((n,))**  **for k in range(niter):**  **dw = np\_grad\_fn\_vectorized(w, X, y)**  **w = w - alpha\*dw**  **if k % print\_every == 0:**  **print('Weight after %d iteration: %s' % (k, str(w)))**  **return w** |

Kết quả của thuật toán là một bộ số weight bias để từ đó đưa ra được giá trị y\_predict

#### Kết quả

**Table

Description automatically generated**

*Plot giá trị của y\_predict so với giá trị thực tế*

Chart, scatter chart

Description automatically generated

Hình 2 Dữ liệu dự đoán so với dữ liệu tập test

### 1.2 Thực hiện bằng Tensorflow

Thực hiện xử lý dữ liệu như dùng Numpy ở trên, ta được bộ dữ liệu gồm có X\_train,y\_train,X\_test,y\_test

Sử dụng mô hình thuật toán LinearRegression của thư viện sklearn.linear\_model

Text, letter

Description automatically generated

Ta thu được giá trị các bộ giá trị y\_predict. Plot y\_test và y\_predict lên ta được biểu đồ

Chart, scatter chart

Description automatically generated

Hình 3 Dữ liệu dự đoán so với tập test khi sử dụng tensorflow

Sau khi áp dụng bộ dữ liệu vào mô hình Linear Regression ta được giá trị R2 = 0.673 .

Để cải thiện, sử dụng thêm mô hình Ridge Regression

Text

Description automatically generated

Sau quá trình, ta thu được giá trị của hệ số R2 = 0.763.

* Bộ dữ liệu áp dụng với mô hình Ridge Regression mang lại hiệu quả cao hơn so với Linear Regression

# Phần II: Thực hiện phân tích trên bộ dữ liệu

## Chương 1: TỔNG QUAN VỀ ĐỀ TÀI

#### 1.1 Giới thiệu về đề tài

Trong đề tài này, em sẽ phân tích một bộ dữ liệu được cho sẵn trên Kaggle: <https://www.kaggle.com/datasets/ujjwalchowdhury/energy-efficiency-data-set>

Bộ dữ liệu gồm 8 Features đầu vào và có 2 Freatues đầu ra

#### 1.2 Mục tiêu

* Đưa ra được những dự đoán cho 2 giá trị đầu ra là ‘Heating\_Load’ và ’Cooling\_Load’
* Tinh chỉnh lại mô hình

#### 1.3 Phương pháp thực hiện

Thực hiện bằng cách đưa bộ data đã được xử lý qua thuật toán Decision Tree Regression và Random Forest Regression, sau đó điều chỉnh lại giá trị của thuật toán để đưa ra dự đoán tốt nhất cho dataset

## Chương 2: LÝ THUYẾT CHUNG

### 2.1 Quy trình cơ bản để phân tích một bộ dữ liệu

Hình 4 Quy trình phân tích một bộ data

- Data collection – Thu thập dữ liệu: cần có một bộ dữ liệu (dataset), dữ liệu có thể tự thu thập hoặc được lấy từ các nguồn dữ liệu lớn trên internet. Một lưu ý là phải thu thập từ chính hệ thống nguồn, có như vậy mới chính xác dữ liệu và máy có thể học một cách đúng đắng và đạt hiệu quả cao hơn.

- Preprocessing - tiền xử lý: Xử lý dữ liệu null, đánh nhãn dữ liệu, mã hóa một số đặc trưng, ​​trích xuất đặc trưng, ​​rút ​​gọn dữ liệu nhưng vẫn bảo đảm kết quả. Bước 1 và 2 thường sử dụng hơn 70% thời gian thực hiện tổng.

- Model Training – Huấn luyện mô hình: Nhiệm vụ chính của bước này là cần phải đào tạo mô hình, để máy tính có thể học trên bộ dataset đã đưa vào

- Evaluation - Đánh giá mô hình: sau khi huấn luyện xong mô hình, chúng ta cần sử dụng các phương pháp đo lường để đánh giá mô hình, tùy chọn từng độ đo khác nhau mà mô hình cũng được đánh giá tốt hoặc không khác nhau. Độ chính xác của mô hình đạt trên 80% được coi là một mô hình tốt

- Cải thiện mô hình: sau khi đánh giá mô hình, các mô hình đạt độ chính xác không tốt thì cần phải đào tạo lại, chúng ta sẽ lặp lại từ bước 3, cho đến khi đạt được độ chính xác như kỳ vọng.

### 2.2 Giá trị R2

**- R2** là phần trăm mức độ dao động của biến đầu ra có thể được giải thích bằng mô hình tuyến tính. R2 luôn nằm ở trong mức từ [0,1].

*Với:*

*Residual Sum of Squares(****ESS****): tổng các độ lệch bình phương phần dư*

*Total Sum of Squares(****TSS****): tổng các độ lệch bình phương toàn bộ*

- Một cách tổng quan, nếu giá trị R2 càng lớn thì mô hình càng phù hợp với dữ liệu.

#### *Hạn chế của hệ số R bình phương*

Càng đưa thêm nhiều biến vào mô hình, mặc dù chưa xác định biến đưa vào có ý nghĩa hay không thì giá trị R2 sẽ tăng. Lý do là khi càng đưa thêm biến giải thích vào mô hình thì sẽ càng khiến phần dư giảm xuống (vì bản chất những gì không giải thích được đều nằm ở phần dư), do vậy tăng thêm biến sẽ khiến tổng bình phương phần dư (Residual Sum of Squares) giảm, trong khi Total Sum of Squares không đổi, dẫn tới R2 luôn luôn tăng.  
 Giá trị R2 tăng khả năng giải thích của mô hình, nhưng bản chất thì lại không làm rõ được tầm quan trọng của biến đưa vào, do đó nếu dựa vào giá trị R2 để đánh giá tính hiệu quả của mô hình sẽ dẫn đến tình huống không chính xác vì sẽ đưa quá nhiều biến không cần thiết, làm phức tạp mô hình.

## Chương 3: CÁC BƯỚC THỰC HIỆN

### 3.1 Data Collection

Tập dữ liệu này được thu thập từ kho lưu trữ Máy học UCI. Mô tả tập dữ liệu như sau:

“Nghiên cứu này xem xét các yêu cầu về tải sưởi ấm và tải làm mát của các toà nhà để mang lại hiệu quả tính toán công suất dùng năng lượng cho các hộ gia đình

Thực hiện phân tích bằng cách sử dụng 12 hình dạng toà nhà khác nhau được mô phỏng trong Ecotect. Các toà nhà khác nhau về diện tích lắp kính, sự phân bố diện tích lắp kính và hướng, và một số thông số khác để tạo ra 768 hình dạng của toà nhà. Tập dữ liệu bao gồm 768 mẫu và có 8 feature, nhằm dự đoán được hai phản hồi có giá trị thực. Nó cũng có thể được sử dụng như một bài toán phân loại nhiều lớp nếu phản hồi được làm tròn đến số nguyên gần nhất”

Graphical user interface, application

Description automatically generated

Text

Description automatically generated

Nhận thấy không có dữ liệu thiếu trong bộ data

A picture containing graphical user interface

Description automatically generated

Ma trận Correlation giữa các thuộc tính trong bộ data, có giá trị =[-1,1], giá trị càng gần 1 thì sự liên quan giữa các thuộc tính càng lớn

### 3.2 Data Preprocessing

#### Data Selection

Mục đích của bài tập này là làm cho giá trị R2 càng lớn càng tốt, giá trị R2 sẽ tăng lên khi thêm các feature. Để có R2 cao, chúng ta chọn tất cả 8 feature

A screenshot of a computer

Description automatically generated with low confidence

#### Data Normalization

Normalization là phương pháp scale dữ liệu từ miền giá trị bất kì sang miền giá trị nằm trong khoảng 0 đến 1.

Phương pháp này yêu cầu chúng ta cần xác định được giá trị lớn nhất (max) và giá trị nhỏ nhất (min) của dữ liệu.

Giá trị được normalize theo công thức sau:

Với y là biến sau khi normalize, x là biến trước khi normalize

Để normalize dữ liệu, ta cần normalize từng thuộc tính (feature) của dữ liệu. Công thức trên áp dụng đối với từng feature.

Trong đó x là giá trị cần được normalize, maximum và minium là giá trị lớn nhất và nhỏ nhất của trong tất cả các quan sát của feature trong tập dữ liệu.

Trong nội dung bài tập này, em chia bộ dữ liệu ra thành 2 phần biến X và kết quả y, đồng thời sử dụng hàm Normalizer của sklearn.preprocessing

|  |
| --- |
| from sklearn.preprocessing import Normalizer  nr = Normalizer(copy=False)  X = data.drop(['Heating\_Load','Cooling\_Load'], axis=1)  X = nr.fit\_transform(X)  y = data[['Heating\_Load','Cooling\_Load']] |

### 3.3 Training Model

Tập dữ liệu được chia ra làm 2 phần train và test với hệ số 0.8 và 0.2, sử dụng train\_test\_split của thư viện sklearn.model\_selection



Xây dựng hàm đánh giá R2 với từng mô hình

Text, letter

Description automatically generated

#### Model Selection

Sử dụng các thuật toán dựa trên cây để đưa ra mô hình tốt nhất với bộ dữ liệu. Cụ thể sử dụng 2 mô hình cơ bản và sau đó tối ưu hoá mô hình. Hai mô hình được chọn là

* Decision Tree Regression
* Random Forest Regression

##### Decision Tree Regression

Cây quyết định xây dựng mô hình hồi quy hoặc phân loại dưới dạng cấu trúc cây. Nó chia nhỏ tập dữ liệu thành các tập con nhỏ hơn và nhỏ hơn trong khi đồng thời cây quyết định liên quan được phát triển từng bước. Kết quả cuối cùng là một cây với các nút quyết định và các nút lá. Một nút quyết định có hai hoặc nhiều, mỗi nhánh đại diện cho các giá trị cho thuộc tính được kiểm tra. Nút lá thể hiện quyết định về mục tiêu số. Nút quyết định cao nhất trong cây tương ứng với công cụ dự đoán tốt nhất được gọi là nút gốc . Cây quyết định có thể xử lý cả dữ liệu phân loại và dữ liệu số.

*Thuật toán cây quyết định*

Thuật toán cốt lõi để xây dựng cây quyết định được JR Quinlan gọi là ID3, sử dụng một tìm kiếm tham lam từ trên xuống trong không gian của các nhánh có thể có mà không có dấu vết quay lại. Thuật toán ID3 có thể được sử dụng để xây dựng cây quyết định cho hồi quy bằng cách thay thế Mức tăng thông tin bằng Giảm độ lệch chuẩn.

*Độ lệch chuẩn*

Cây quyết định được xây dựng từ trên xuống từ một nút gốc và liên quan đến việc phân chia dữ liệu thành các tập con chứa các cá thể có giá trị tương tự (đồng nhất). Sử dụng độ lệch chuẩn để tính độ đồng nhất của một mẫu số. Nếu mẫu số hoàn toàn đồng nhất thì độ lệch chuẩn của nó bằng không.

Độ lệch chuẩn cho 1 thuộc tính;

Graphical user interface, text, application

Description automatically generated

* Độ lệch chuẩn ( **S** ) là để xây dựng cây (phân nhánh).
* Hệ số sai lệch ( **CV** ) được sử dụng để quyết định thời điểm ngừng phân nhánh. Chúng ta cũng có thể sử dụng Count ( **n** ).
* Trung bình ( **Avg** ) là giá trị trong các nút lá.

Độ lệch chuẩn cho 2 thuộc tính:

Table

Description automatically generated

*Giảm độ lệch chuẩn*

Việc giảm độ lệch chuẩn dựa trên sự giảm độ lệch chuẩn sau khi tập dữ liệu được tách trên một thuộc tính. Xây dựng cây quyết định là tất cả về việc tìm kiếm thuộc tính trả về mức giảm độ lệch chuẩn cao nhất (tức là các nhánh đồng nhất nhất).

*Thực hiện*

Áp dụng mô hình thuật toán Decision Tree Regression lên bộ dữ liệu training:

A screenshot of a computer

Description automatically generated with medium confidence

Sau đó plot giá trị y\_predict và y\_test ta được biểu đồ sau :

Chart, scatter chart

Description automatically generated

Hình 5 So sánh giữa giá trị thực và giá trị dự đoán bằng Decision Tree Regression

Và có giá trị R2 trả về = 0.955

#### Random Forest Regression

##### Giới thiệu thuật toán

Random là ngẫu nhiên, Forest là rừng, nên ở thuật toán Random Forest mình sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random). Sau đó kết quả dự đoán được tổng hợp từ các cây quyết định.

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.

Khi dùng thuật toán Random Forest, cần quan tâm các thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách.

Trong thuật toán Decision Tree, khi xây dựng cây quyết định nếu để độ sâu tùy ý thì cây sẽ phân loại đúng hết các dữ liệu trong tập training dẫn đến mô hình có thể dự đoán tệ trên tập validation/test, khi đó mô hình bị overfitting, hay nói cách khác là mô hình có high variance.

##### Thuật toán hoạt động như thế nào

Diagram

Description automatically generated

Hình 6 Nguyên lý hoạt động Random Forest Regression

* Chọn các mẫu ngẫu nhiên từ tập dữ liệu đã cho.
* Thiết lập cây quyết định cho từng mẫu và nhận kết quả dự đoán từ mỗi quyết định cây.
* Bỏ phiếu cho mỗi kết quả dự đoán.
* Chọn kết quả được dự đoán nhiều nhất là dự đoán cuối cùng.

*Ưu điểm*: Random forests được coi là một phương pháp chính xác và mạnh mẽ vì số cây quyết định tham gia vào quá trình này. Nó không bị vấn đề overfitting. Lý do chính là nó mất trung bình của tất cả các dự đoán, trong đó hủy bỏ những thành kiến. Thuật toán có thể được sử dụng trong cả hai vấn đề phân loại và hồi quy. Random forests cũng có thể xử lý các giá trị còn thiếu. Có hai cách để xử lý các giá trị này: sử dụng các giá trị trung bình để thay thế các biến liên tục và tính toán mức trung bình gần kề của các giá trị bị thiếu.

*Nhược điểm*: Random forests chậm tạo dự đoán bởi vì nó có nhiều cây quyết định. Bất cứ khi nào nó đưa ra dự đoán, tất cả các Tree trong Forest phải đưa ra dự đoán cho cùng một đầu vào cho trước và sau đó thực hiện bỏ phiếu trên đó. Toàn bộ quá trình này tốn thời gian. Mô hình khó hiểu hơn so với cây quyết định, nơi mà có thể dễ dàng đưa ra quyết định bằng cách đi theo đường dẫn trong cây.

#### Thực hiện thuật toán

Áp dụng mô hình thuật toán Decision Tree Regression lên bộ dữ liệu training:

Text, letter

Description automatically generated

Sau đó plot giá trị y\_predict và y\_test ta được biểu đồ sau :

Chart, scatter chart

Description automatically generated

Hình 7 So sánh dữ liệu dự đoán với dữ liệu thực bằng Random Forest Regression

Ta thu được giá rị R2 score trước khi tối ưu = 0.972

### 3.4 Tối ưu mô hình

#### Decision Tree Regression

##### Tối ưu mô hình

Qua các vòng lặp của mô hình thuật toán Decision Tree Regression Chọn ra các giá trị của 3 parameter “Max\_depth”, ”Min\_samples\_split ”, ”Min\_samples\_leaf” để đạt được Accuracy lớn nhất, sau đó ta có biểu đồ

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Từ đó, chọn các giá trị tương ứng với mỗi parameter:

'max\_depth' : [7,8,9],

'min\_samples\_split': [16,17,18],

'min\_samples\_leaf': [6,7,8]

Sau đó sử dụng GridSearch để tìm ra bộ giá trị trả về độ chính xác lớn nhất cho mô hình cây quyết định, ta được bộ giá trị của 3 parameter như sau:

{'max\_depth': 7, 'min\_samples\_leaf': 8, 'min\_samples\_split': 18}

Áp dụng vào bộ dữ liệu, ta được mô hình được tối ưu hoá như sau:

Chart, scatter chart

Description automatically generated

Hình 8 So sánh dữ liệu dự đoán đã tối ưu với dữ liệu thực DTR

Với hệ số R2 score sau khi tối ưu hoá = 0.972

#### Random Forest Regression

##### Tối ưu mô hình

Qua các vòng lặp của mô hình thuật toán Random Forest Regression, chọn ra các giá trị của 4 parameter “Max\_depth”, ”Min\_samples\_split ”, ”Min\_samples\_leaf”,”Estimator” để đạt được Accuracy lớn nhất, sau đó ta có biểu đồ

Chart, line chart

Description automatically generated

Từ đó, chọn các giá trị tương ứng với mỗi parameter:

'max\_depth' : [6,7,8],

'min\_samples\_split': [14,15,16],

'min\_samples\_leaf' : [6,7,8],

'n\_estimators': [53,54,55]

Sau đó sử dụng GridSearch để tìm ra bộ giá trị trả về độ chính xác lớn nhất cho mô hình cây quyết định, ta được bộ giá trị của 4 parameter như sau:

**Text

Description automatically generated**

Áp dụng vào bộ dữ liệu, ta được mô hình được tối ưu hoá như sau:

Chart, scatter chart

Description automatically generated

Hình 9 So sánh dữ liệu dự đoán đã tối ưu với dữ liệu thực bằng Random Forest Regression

Với hệ số R2 score sau khi tối ưu hoá = 0.974

### 3.5 Evaluation

* R2\_score của `Decision Tree Regression` là 0.955
* R2\_score của `Random Forest Regression` là 0.972

Sau khi điều chỉnh các tham số bằng GridSearch ta có:

* R2\_score của `Decision Tree Regression` là 0.971
* R2\_score của `Random Forest Regression` là 0.974

Từ đó ta thấy được mô hình Random Forest Regression mang lại dự đoán tốt hơn cho bộ dữ liệu

# Phần III: KẾT LUẬN

Trong quá trình làm thực tập tốt nghiệp và thực hiện đề tài cùng cô Trần Thị Ngọc Lan, em đã hiểu hơn các kiến thức được học trên lớp và từ đó ứng dụng chúng vào một bài toán cụ thể. Sau khi thực hiện, em đã thu được nhiều kiến thức cũng như kỹ năng về trí tuệ nhân tạo nói chung và phân tích dữ liệu nói riêng. Trong khi thực hiện đề tài cũng có một số hạn chế trong quá trình thực hiện và một số vấn đề phát sinh trong thực tiễn. Nhưng nhờ sự giúp đỡ tận tình của cô đã giúp em giải quyết vấn đề hạn chế đó. Em xin chân thành cảm ơn cô đã tạo điều kiện cho em được tham gia thực hiện đề tài này !

## Tài liệu tham khảo

* <https://www.w3schools.com/python/python_ml_linear_regression.asp#:~:text=Linear%20regression%20uses%20the%20relationship,the%20future%20is%20very%20important>.
* <https://www.kaggle.com/code/busthon/multiple-regression-on-energy-efficiency>
* <https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html>
* <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>

***Code tại: https://github.com/ThanhLongTSM/TTap***