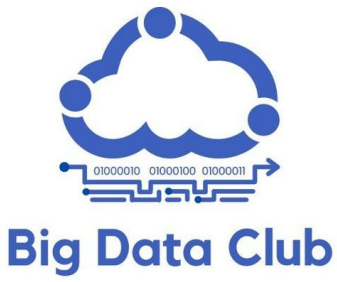
PHÒNG THÍ NGHIỆM TÍNH TOÁN HIỆU NĂNG CAO

CÂU LẠC BỘ BIG DATA

NHÓM 9



ASSIGNMENT 3

PHÂN TÍCH, ĐÁNH GIÁ VÀ TRIỂN KHAI MÔ HÌNH TỐI ƯU CHO DỊCH VỤ KINH DOANH COFFEE

Họ tên sinh viên	*Email*
Lê Quốc Huy	huy.lelequochuy@hcmut.edu.vn
Đoàn Thảo Nhi	nhi.doankhmt20@hcmut.edu.vn
Lưu Quốc Hưng Thịnh	thinh.luukhmt@hcmut.edu.vn
Trương Hoàng Nguyên Vũ	vu.truongcompsci@hcmut.edu.vn

TP.HỒ CHÍ MINH, THÁNG 8/2022

Mục lục

Mục lục **1**

1 Cơ sở lý thuyết **2**
1.1 Feature Selection: sklearn.feature_selection . . . 2
1.2 Model Selection: train_test_split . . . 2
1.3 Tree: DecisionTreeRegressor . . . 2
1.4 Ensemble: Random Forest . . . 3
1.5 Linear model: LinearRegression . . . 3
1.6 Một vài chỉ số đánh giá . . . 3
1.6.1 MSE - Mean Squared Error . . . 3
1.6.2 RMSE - Root Mean Squared Error . . . 4
1.6.3 R-squared . . . 4

2 Huấn luyện mô hình **4**
2.1 Thu thập dữ liệu . . . 4
2.2 Lựa chọn đặc trưng . . . 4
2.3 Huấn luyện . . . 5

3 Đánh giá đề tài **5**
3.1 Ý nghĩa . . . 5
3.2 Hướng phát triển tương lai . . . 5

4 Sản phẩm **5**

TÀI LIỆU THAM KHẢO **6**

1 Cơ sở lý thuyết

Scikit-learning hay **sklearn** là thư viện học máy phổ biến nhất trong cộng đồng khoa học dữ liệu. Được viết bằng Python, **scikit-learning** cung cấp các công cụ hiệu quả và dễ sử dụng để xử lý dữ liệu đến triển khai các mô hình học máy.

Scikit-learn phổ biến chính vì sự dễ sử dụng của thư viện này, ta có thể dễ dàng xây dựng một mô hình học máy phổ biến thông qua một số một số câu lệnh **sklearn**.

Và trong phần một này, chúng ta sẽ cùng tìm hiểu những module chính trong **sklearn** xuất hiện trong phần thực hành project ở phần sau.

1.1 Feature Selection: sklearn.feature_selection

Feature Selection rất quan trọng trong sự thành công của các mô hình học máy. **Scikit-learning** cung cấp một số thuật toán lựa chọn tính năng trong module này.

Mutual_info_regression: Hàm đo mức độ phụ thuộc lẫn nhau giữa hai biến ngẫu nhiên và giá trị trả về là một số không âm. Kết quả trả về bằng 0 chứng tỏ hai biến độc lập, và giá trị càng lớn nghĩa là chúng càng phụ thuộc nhau. Hàm này dùng cho biến mục tiêu là biến liên tục.

```
sklearn.feature_selection.mutual_info_regression(X, y)
```

```python
from sklearn.feature_selection import mutual_info_regression
import matplotlib.pyplot as plt
%matplotlib inline

importances= mutual_info_regression(x_train,y_train)
featureimp=pd.Series(importances)
featureimp.plot(kind='barh',color="teal")
plt.show()
```

Hình 1: *Sử dụng Mutual_info_regression trong project*

1.2 Model Selection: train_test_split

Hàm **train_test_split** thuộc model **model_selection** trong thư viện **sklearn** được sử dụng để chia data thành bộ dữ liệu để train và bộ dữ liệu để test.

```
sklearn.model_selection.train_test_split(*arrays, test_size=None, train_size=None)
```

Trong đó:

- array: dữ liệu đầu vào có thể là list, numpy arrays, pandas dataframes.
- test_size: float hoặc int. Nếu là kiểu float, giá trị này biểu diễn tỉ lệ phần trăm dùng để chia cho bộ dữ liệu test, nằm trong khoảng từ 0.0 đến 1.0. Nếu là kiểu int, giá trị biểu diễn số lượng mẫu trong bộ dữ liệu test.
- train_size: float hoặc int. Nếu là kiểu float, giá trị này biểu diễn tỉ lệ phần trăm dùng để chia cho bộ dữ liệu train, nằm trong khoảng từ 0.0 đến 1.0. Nếu là kiểu int, giá trị biểu diễn số lượng mẫu trong bộ dữ liệu train.

1.3 Tree: DecisionTreeRegressor

Decision Tree là một công cụ ra quyết định bằng cách sử dụng sơ đồ cây như flowchart hoặc là một mô hình cây quyết định với các kết quả có thể xảy ra. Thuật toán **Decision Tree** thuộc loại thuật toán học có giám sát. Nó có thể được dùng cho cả biến liên tục hay phân loại.

Decision Tree Regression: xem xét các đặc trưng của đối tượng và đào tạo trong mô hình ở dạng cấu trúc cây để dự đoán dữ liệu trong tương lai nhằm tạo ra đầu ra là giá trị liên tục có ý nghĩa.

```
sklearn.tree.DecisionTreeRegressor()
```

```
from sklearn.tree import DecisionTreeRegressor
decision = DecisionTreeRegressor().fit(x_train,y_train)
```

Hình 2: *Sử dụng thuật toán Decision Tree trong project*

1.4 Ensemble: Random Forest

Ensemble learning giúp cải thiện kết quả học máy bằng cách kết hợp một số mô hình. Cách tiếp cận này cho phép tạo ra hiệu suất dự đoán tốt hơn so với một mô hình duy nhất.

- **Ưu điểm:** Cải thiện độ chính xác của kết quả dự đoán.
- **Nhược điểm:** Khó để hiểu cách phân loại của **Ensemble**.

Random Forest là một kỹ thuật tổng hợp có khả năng thực hiện hồi quy và phân loại bằng việc sử dụng nhiều cây quyết định và một kỹ thuật đươc gọi là **Bagging**. Ý tưởng cơ bản đằng sau điều này là kết hợp nhiều cây quyết định để xác định đầu ra cuối cùng thay vì dựa vào các cây quyết định riêng lẻ.

`sklearn.ensemble.RandomForestRegressor()`

```
from sklearn.ensemble import RandomForestRegressor
random    = RandomForestRegressor().fit(x_train,y_train)
```

Hình 3: *Sử dụng Random Forest trong project*

1.5 Linear model: LinearRegression

Linear Regression - Hồi quy tuyến tính là một thuật toán học máy dựa trên việc học có giám sát. Hồi quy tuyến tính thực hiện dự đoán một giá trị biến phụ thuộc (y) dựa trên biến độc lập cho trước (x). Kĩ thuật này nhằm tìm ra mối quan hệ tuyến tính giữa đầu vào x và đầu ra y.

`sklearn.linear_model.LinearRegression()`

```
from sklearn.linear_model import LinearRegression
model_linear = LinearRegression()
model_linear.fit(x_train,y_train)
y_pred = model_linear.predict(x_test)
```

Hình 4: *Sử dụng thuật toán Linear Regression trong project*

1.6 Một vài chỉ số đánh giá

1.6.1 MSE - Mean Squared Error

MSE, hay còn gọi là phương sai, là giá trị trung bình của bình phương độ chênh lệch giữa giá trị dự đoán và giá trị thực tế.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Hình 5: *Công thức tính Mean Squared Error - MSE*

1.6.2 RMSE - Root Mean Squared Error

RMSE, hay còn gọi là độ lệch chuẩn, được tính bằng căn bậc hai phương sai.

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\hat{y}_i - y_i)^2}{n}}$$

Hình 6: *Công thức tính Root Mean Squared Error - RMSE*

1.6.3 R-squared

Đây là một thước đo được sử dụng trong thống kê và nó cho chúng ta biết mức độ phù hợp của mô hình nghiên cứu với ý nghĩa là các biến.

Hệ số tương quan R bình phương cho biết mô hình đó hợp với dữ liệu ở mức bao nhiêu phần trăm. Giá trị này càng cao thì mối quan hệ giữa biến độc lập và biến phụ thuộc càng chặt chẽ.

Để kiểm tra hệ số R bình phương:

```
r2_score(y_test, y_predict)
```

2 Huấn luyện mô hình

2.1 Thu thập dữ liệu

Dữ liệu được dùng để huấn luyện cho mô hình là một tập dữ liệu chứa sản lượng bán ra các mặt hàng thực phẩm của một của hàng có nhiều chi nhánh. Chi tiết của bộ dữ liệu như sau:

	id	week	center_id	meal_id	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders
0	1379560	1	55	1885	136.83	152.29	0	0	177
1	1466964	1	55	1993	136.83	135.83	0	0	270
2	1346989	1	55	2539	134.86	135.86	0	0	189
3	1338232	1	55	2139	339.50	437.53	0	0	54
4	1448490	1	55	2631	243.50	242.50	0	0	40

Hình 7: *Dataset cho việc huấn luyện*

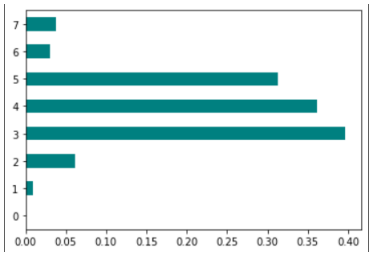
2.2 Lựa chọn đặc trưng

Sau khi đã có dữ liệu, ta sẽ thực hiện lựa chọn ra những đặc trưng phù hợp để có thể thực hiện huấn luyện mô hình. Cụ thể, ta sẽ sử dụng hàm *mutual_ info_ regression* để có thể thấy được mức độ liên quan giữa các nhân tố với biến mục tiêu và thực hiện thể hiện bằng đồ thị.

```
from sklearn.feature_selection import mutual_info_regression
import matplotlib.pyplot as plt
%matplotlib inline

importances= mutual_info_regression(x_train,y_train)
featureimp=pd.Series(importances)
featureimp.plot(kind='barh',color="teal")
plt.show()
```

Hình 8: *Đánh giá mức độ liên quan của các nhân tố với biến mục tiêu*



Hình 9: *Đồ thị thể hiện mức độ liên quan*

Khi đó, ta được kết quả như hình bên trên.

Dựa vào biểu đồ, ta thấy nhân tố 0, 1, 6, 7 có chỉ số khá thấp nên ta sẽ loại khỏi mô hình và chỉ lấy những nhân tố còn lại để thực hiện huấn luyện.

2.3 Huấn luyện

Đối với việc huấn luyện, nhóm sẽ thực hiện huấn luyện trên 3 mô hình khác nhau và đánh giá dựa trên các chỉ số đánh giá như **MSE**.

Sau đây sẽ là một số chỉ số đánh giá khi thực hiện huấn luyện mô hình trên ba loại mô hình khác nhau.

	Linear Regression	Decision Tree	Random Forest
r2_score	0.1113	0.1649	0.1746
MSE	128151.8484	120422.5630	119015.7227
RMSE	357.983	347.0195	344.986

Ta có thể thấy, các chỉ số đánh giá tăng dần khi ta thay đổi mô hình từ **Linear Regression** đến **Decision Tree** và cao nhất là mô hình **Random Forest**.

Ta có thể thấy một số nguyên nhân do mô hình có chứa các biến phân loại nên khi sử dụng mô hình **Linear Regression** sẽ ảnh hưởng khá nhiều do sự phân bố của các nhãn trong biến phân loại, thông thường sẽ sử dụng **OneHotEncoding** cho các biến phân loại nhưng do các nhãn trong các biến phân loại quá nhiều nên khi thực hiện **OneHotEncoding** sẽ khiến cho bộ dữ liệu trở nên rất lớn.

Đối với mô hình **Decision Tree** thì sẽ tránh được câu chuyện của mô hình **Linear Regression**, nhưng vẫn còn một nhược điểm là có thể gây ra **Overfitting** nếu độ sâu của cây quá lớn và cũng có thể không học hết được đặc trưng của bộ dữ liệu vì có thể mô hình sẽ thiên vị cho một vài thuộc tính.

Mô hình **Random Forest** so với **Decision Tree** trong bộ dữ liệu này thì cũng không mang lại kết quả tốt hơn nhiều.

3 Đánh giá đề tài

3.1 Ý nghĩa

Việc đưa ra dự đoán được nhu cầu của các mặt hàng nói chung và của các thực phẩm nói riêng tại các khu vực khác nhau sẽ đem lại lợi ích kinh tế cho các doanh nghiệp, khi đó sẽ cung ứng vừa đủ các mặt hàng đến từng khu vực, hạn chế việc di chuyển nhiều lần giữa các khu vực để cung cấp các mặt hàng, tiết kiệm được thời gian cũng như chi phí di chuyển.

3.2 Hướng phát triển tương lai

Trong tương lai, có thể sẽ sử dụng thêm nhiều mô hình học kết hợp khác để xây dựng mô hình, đồng thời thu thập thêm nhiều dữ liệu có giá trị liên quan đến đề tài.

4 Sản phẩm

Nhấn vào đây để đến trang web trình bày sản phẩm.

Tài liệu

[1] https://scikit-learn.org/stable/index.html

[2] https://www.geeksforgeeks.org/machine-learning/?ref=shm

[3] https://www.datatechnotes.com/2019/10/accuracy-check-in-python-mae-mse-rmse-r.html