ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN THỰC HÀNH

MÔN: TOÁN ỨNG DỤNG VÀ THỐNG KÊ (MTH00057 – 21CLC01)

LAB03: LINEAR REGRESSION

GIẢNG VIÊN LÝ THUYẾT NGUYỄN ĐÌNH THÚC

GIẢNG VIÊN THỰC HÀNH NGUYỄN VĂN QUANG HUY
NGÔ ĐÌNH HY

SINH VIÊN THỰC HIỆN 21127621 – ÂU DƯƠNG KHANG

MỤC LỤC

1) Tổng quan

Sinh viên thực hiện đồ án:

- Họ và tên: Âu Dương Khang
- MSSV: 21127621
- Lớp: 21CLC1

Tổng quan đồ án:

- Chương trình được thử nghiệm trên môi trường text editor Visual Studio Code với phiên bản Python 3.11.0.
- Đánh giá mức độ hoàn thành:

Task	Mức độ hoàn thành
Task 1a	100%
Task 1b	100%
Task 1c	100%
Task 1d	100%

2) Cài đặt

- Chương trình có sử dụng các thư viện dưới đây:

```
import pandas as pd
import numpy as np
# Import thêm dữ thư viện nếu cần
from sklearn.metrics import mean_absolute_error
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import KFold, cross_val_score
import matplotlib.pyplot as plt
import seaborn as sns
```

- import pandas as pd: dùng để đọc file train.csv và test.csv.
- import numpy as np: dùng để tạo ma trận và các phép toán trên ma trận.
- from sklearn.metrics import mean_absolute_error: tính toán MAE
- from sklearn.linear_model import LinearRegression: đưa vào class LinearRegression. Trong đồ án sử dụng class LinearRegressgion với các phương thức fit() cho việc tính nghiệm θ và predict() sau khi tìm được θ.

- from sklearn.model_selection import KFold, cross_val_score:
    - Import hàm KFold có chức năng chia cắt dữ liệu thành 2 phần train và test, số lần cắt dữ liệu là số k cho trước. Khi sử dụng hàm sẽ xáo trộn 1 lần với số random_state = 42 nhằm giữ nguyên thứ tự dữ liệu sau khi xáo trộn. Đầu vào gồm có:
        - n_splits = k: số lần cắt dữ liệu.
        - shuffle = True: có xáo trộn dữ liệu.
        - random_state = 42: giữ nguyên thứ tự dữ liệu sau khi xáo trộn.
    - Hàm cross_val_score có nhiệm vụ tính toán MAE từ k lần dữ liệu bị cắt ra ở trên. Từ đó lấy trung bình để có giá trị MAE của thuộc tính đó. Đầu vào gồm có:
        - clf: thuật toán muốn đưa vào, ở đây là Linear Regression.
        - X_train_x: dữ liệu để train ứng với các thuộc tính yêu cầu đề bài.
        - y_train_x: dữ liệu để train ứng với giá trị mục tiêu.
        - cv = kfolds: cross validation bằng KFold ở trên.
        - scoring = ‘neg_mean_absolute_error’: số điểm được đưa ra, ở đây sử dụng MAE để tính toán.
- import matplotlib.pyplot as plt và import seaborn as sb: dùng để biểu diễn độ tương quan giữa các thuộc tính với giá trị mục tiêu.

3) Danh sách task

a) Task 1a: Xây dựng mô hình sử dụng 11 đặc trưng đầu tiên

- Huấn luyện 1 lần duy nhất cho 11 đặc trưng đầu tiên cho toàn bộ tập huấn luyện (train.csv). Sử dụng phương thức fit() của lớp LinearRegression() để train model tương ứng với dữ liệu X_train_a và y_train_a. Sau khi huấn luyện ta có được model như sau:

```
array([[-23183.33 ,    702.767,   1259.019, -99570.608,  18369.962,
         1297.532,  -8836.727,    141.76 ,    145.742,    114.643,
        34955.75 ]])
```

- Từ đó, ta có công thức hồi quy:

$$\text{Salary} = -23183.33(Gender) + 702.767(10percentage) + 1259.019(12percentage) - 99570.608(CollegeTier) + 18369.962(Degree)$$
$$+1297.532(collegeGPA) - 8836.727(CollegeCityTier) + 141.76(English) + 145.742(Logical) + 114.643(Quant) + 34955.75(Domain)$$

- Sử dụng phương thức predict() cho model để dự đoán kết quả dữ liệu X_test_a, sau đó đánh giá giá trị nhận được bằng hàm mean_absolute_error() (MAE) giữa giá trị dự báo y_new và giá trị thực y_test_a.

```
105052.52978823167
```

- Kết quả ước lượng sai số theo MAE: 105052.53

b) Task 1b: Xây dựng mô hình sử dụng mở 1 đặc trưng tính cách duy nhất, cho biết mô hình nào cho ra kết quả tốt nhất.

- Đầu tiên ta lấy dữ liệu từng đặc trưng tính cách, sử dụng hàm KFold() để chia nhóm tập dữ liệu, sau đó sử dụng hàm cross_val_score() để tính sai số MAE giữa giá trị dự báo và giá trị thực của từng nhóm dữ liệu. Sau đó ta lấy giá trị trung bình của các sai số trên, kết quả được lưu vào mảng chứa các giá trị MAE của từng đặc trưng trên.
- Tham số đầu vào: k nhóm dữ liệu được chia, tập dữ liệu dùng để train X_train_b, y_train_b.
- Tham số đầu ra: mảng giá trị MAE trung bình mỗi đặc trưng.

- Sau khi tìm được đặc trưng có giá trị MAE trung bình thấp nhất, ta sẽ huấn luyện lại mô hình với các bước như câu 1a để tìm ra mô hình cuối cùng.

```
CVData = []
kfolds = 5
clf = LinearRegression()
y_train_b = train[['Salary']]

X_train_b = X_train[['conscientiousness']]
k_folds = KFold(n_splits = kfolds, shuffle=True,random_state=42)

scores = cross_val_score(clf, X_train_b, y_train_b, cv = k_folds,scoring='neg_mean_absolute_error')
CVData.append(-scores.mean())
```

- Bước 1: tạo mảng chứa giá trị MAE trung bình, khởi tạo số k nhóm dữ liệu là 5, chọn thuật toán LinearRegression. Sau đó chọn dữ liệu từng đặc trưng để tính MAE cho từng nhóm dữ liệu. Từ đó, thêm dữ liệu trung bình của MAE vào mảng.

```
X_train_b = X_train[['conscientiousness','agreeableness','extraversion','nueroticism', 'openess_to_experience']]
CVData = np.asarray([X_train_b.columns,CVData])

CVdf = pd.DataFrame(CVData.T, columns=["Mô hình với 1 đặc trưng", "MAE"])

SortedFeatures = CVdf.sort_values(by=['MAE'], ignore_index=True)

# In ra các kết quả cross-validation như yêu cầu
print("Best Feature: ", SortedFeatures.iloc[0, 0])
print("MAE Value: ", SortedFeatures.iloc[0, 1])
display(SortedFeatures)
```

Python

```
Best Feature:  nueroticism
MAE Value:  123473.39978671989
```

	Mô hình với 1 đặc trưng	MAE
0	nueroticism	123473.399787
1	agreeableness	123706.05473
2	extraversion	123809.9262
3	openess_to_experience	123818.333575
4	conscientiousness	124182.563823

- Bước 2: Sắp xếp các giá trị trong mảng theo thứ tự tăng dần, từ đó ta có thuộc tính có giá trị MAE trung bình thấp nhất là **nueroticism.**

- Bước 3: Huấn luyện lại mô hình chỉ 1 thuộc tính nueroticism. Sử dụng phương thức fit() của lớp LinearRegression() để train model tương ứng với dữ liệu X_best_train_b và y_train_b. Sau khi huấn luyện ta có được model như sau:

```
array([[-16021.494]])
```

- Từ đó, ta có công thức hồi quy:

$$\text{Salary} = -16021.494(nueroticism)$$

- Sử dụng phương thức predict() cho model để dự đoán kết quả dữ liệu X_test_b, sau đó đánh giá giá trị nhận được bằng hàm mean_absolute_error() (MAE) giữa giá trị dự báo y_new và giá trị thực y_test_b.

```
119361.91739987816
```

- Kết quả ước lượng sai số theo MAE: 119361.917

c) Task 1c: Xây dựng mô hình sử dụng mở 1 đặc trưng ngoại ngữ, logic, định lượng duy nhất, cho biết mô hình nào cho ra kết quả tốt nhất.

- Đầu tiên ta lấy dữ liệu từng đặc trưng, sử dụng hàm KFold() để chia nhóm tập dữ liệu, sau đó sử dụng hàm cross_val_score() để tính sai số MAE giữa giá trị dự báo và giá trị thực của từng nhóm dữ liệu. Sau đó ta lấy giá trị trung bình của các sai số trên, kết quả được lưu vào mảng chứa các giá trị MAE của từng đặc trưng trên.
- Tham số đầu vào: k nhóm dữ liệu được chia, tập dữ liệu dùng để train X_train_c, y_train_c.
- Tham số đầu ra: mảng giá trị MAE trung bình mỗi đặc trưng.
- Sau khi tìm được đặc trưng có giá trị MAE trung bình thấp nhất, ta sẽ huấn luyện lại mô hình với các bước như câu 1a để tìm ra mô hình cuối cùng.

```
CVData = []
y_train_c = train[['Salary']]


X_train_c = X_train[['English']]
k_folds = KFold(n_splits = kfolds, shuffle=True,random_state=42)
scores = cross_val_score(clf, X_train_c, y_train_c, cv = k_folds,scoring='neg_mean_absolute_error')
CVData.append(-scores.mean())
```

- Bước 1: tạo mảng chứa giá trị MAE trung bình, khởi tạo số k nhóm dữ liệu là 5, chọn thuật toán LinearRegression. Sau đó chọn dữ liệu từng đặc trưng để tính MAE cho từng nhóm dữ liệu. Từ đó, thêm dữ liệu trung bình của MAE vào mảng.

```
X_train_c = X_train[['English','Logical','Quant']]
CVData = np.asarray([X_train_c.columns,CVData])
CVdf = pd.DataFrame(CVData.T, columns=["Mô hình với 1 đặc trưng", "MAE"])
SortedFeatures = CVdf.sort_values(by=['MAE'], ignore_index=True)

# In ra các kết quả cross-validation như yêu cầu
print("Best Feature: ", SortedFeatures.iloc[0, 0])
print("MAE Value: ", SortedFeatures.iloc[0, 1])
display(SortedFeatures)
```

Python

```
Best Feature:  Quant
MAE Value:  117353.83803114605
```

	Mô hình với 1 đặc trưng	MAE
0	Quant	117353.838031
1	Logical	119932.503599
2	English	120728.603666

- Bước 2: Sắp xếp các giá trị trong mảng theo thứ tự tăng dần, từ đó ta có thuộc tính có giá trị MAE trung bình thấp nhất là **Quant.**

- Bước 3: Huấn luyện lại mô hình chỉ 1 thuộc tính Quant. Sử dụng phương thức fit() của lớp LinearRegression() để train model tương ứng với dữ liệu X_best_train_c và y_train_c. Sau khi huấn luyện ta có được model như sau:

```
array([[368.852]])
```

- Từ đó, ta có công thức hồi quy:

$$\text{Salary} = 368.852(Quant)$$

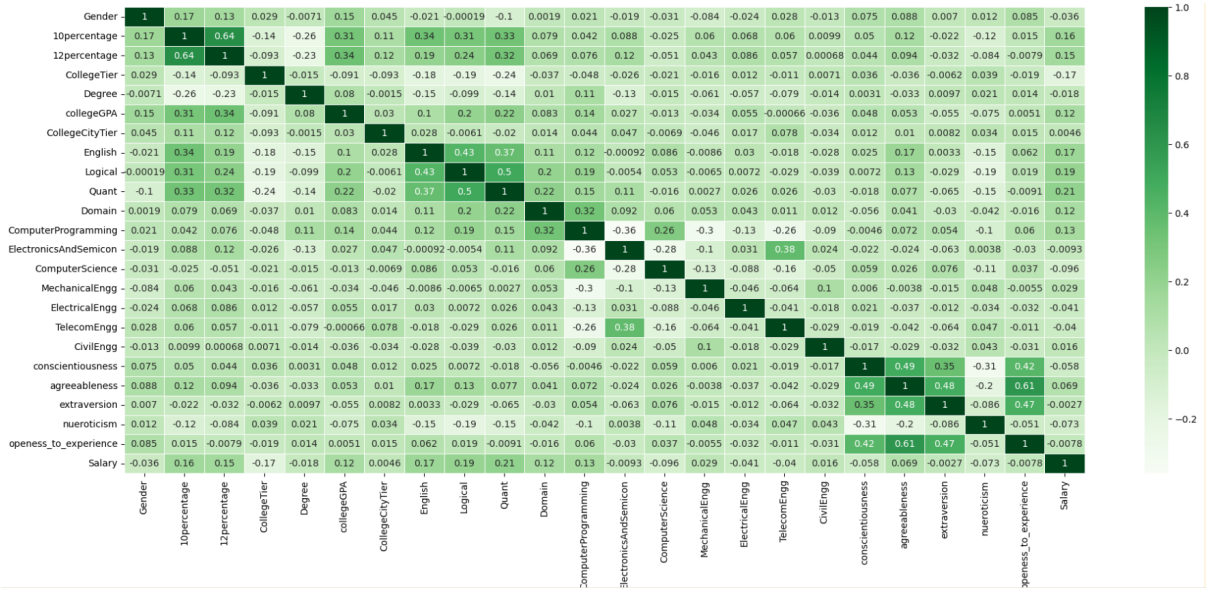
- Sử dụng phương thức predict() cho model để dự đoán kết quả dữ liệu X_test_c, sau đó đánh giá giá trị nhận được bằng hàm mean_absolute_error() (MAE) giữa giá trị dự báo y_new và giá trị thực y_test_c.

```
108814.05968837194
```

- Kết quả ước lượng sai số theo MAE: 108814.06

d) Task 1d: Tự xây dựng model, tìm model cho kết quả tốt nhất.

- Để tìm được mô hình cho ra kết quả tốt nhất, ta sẽ tìm độ tương quan giữa giá trị mục tiêu với các thuộc tính, biểu diễn bằng plt.subplots() và sns.heatmap(). Ma trận tương quan như sau:



- Ta sẽ sử dụng các thuộc tính có độ tương quan lớn hơn 0 với giá trị mục tiêu để tạo nên mô hình. Từ đó chúng ta có 6 model:

```python
#Model 1: Select feature with correlation positive
p1 = X_train[['10percentage','12percentage','collegeGPA','CollegeCityTier','English','Logical','Quant','Domain','ComputerProgramming','MechanicalEngg','CivilEngg','agreeableness']]
#Model 2:  Select feature with correlation positive and < 0.1
p2 = X_train[['CollegeCityTier','MechanicalEngg','CivilEngg','agreeableness']]
#Model 3:  Select feature with correlation > 0.1
p3 = X_train[['10percentage','12percentage','collegeGPA', 'English','Logical','Quant','Domain','ComputerProgramming']]
#Model 4: Select feature with correlation > 0.15
p4 = X_train[['10percentage','English','Logical','Quant']]
#Model 5: Select feature with correlation > 0.1 and select first 6 features
p5 = X_train[['10percentage','12percentage','collegeGPA', 'English','Logical','Quant']]
```

Python

+ Model 1: Chỉ có những thuộc tính có độ tương quan > 0
+ Model 2: Chỉ những thuộc tính có độ tương quan nằm trong khoảng (0, 0.1)
+ Model 3: Chỉ những thuộc tính có độ tương quan > 0.1
+ Model 4: Chỉ những thuộc tính có độ tương quan > 0.15
+ Model 5: Chỉ những thuộc tính có độ tương quan > 0.15 và chọn 6 thuộc tính đầu. (Do các thuộc tính đầu có giá trị toàn dương)

+ Model 6: Sử dụng lại model 5 nhưng các giá trị của thuộc tính sẽ lấy căn.

```
y_train_d = train[['Salary']]
models = []
CVData = []

X_train_d = p1
k_folds = KFold(n_splits = kfolds, shuffle=True,random_state=42)
scores = cross_val_score(clf, X_train_d, y_train_d, cv = k_folds,scoring='neg_mean_absolute_error')
CVData.append(-scores.mean())
```

```
Best Feature:  Model 3
MAE Value:  113048.29783047087
```

	Feature	MAE
0	Model 3	113048.29783
1	Model 1	113080.764182
2	Model 5	114036.037204
3	Model 6 (Sqrt Model 5)	114154.607376
4	Model 4	114689.548963
5	Model 2	123878.349492

- Thử nghiệm từng mô hình bằng KFold Cross Validation với độ sai số bằng MAE đã dùng như câu 1b và 1c, lưu lại giá trị trung bình MAE của từng mô hình vào mảng. Sắp xếp các giá trị trong mảng theo thứ tự tăng dần, từ đó ta có mô hình có giá trị MAE trung bình thấp nhất là mô hình 3.

- Huấn luyện lại mô hình 3. Sử dụng phương thức fit() của lớp LinearRegression() để train mô hình tương ứng với dữ liệu X_best_train_d và y_train_d. Sau khi huấn luyện ta có được mô hình như sau:

```
array([[  671.584,   985.312,  1193.068,   155.09 ,   144.618,   167.789,
        24416.461,    70.416]])
```

- Từ đó, ta có công thức hồi quy:

$$\text{Salary} = 671.584(10percentage) + 985.312(12percentage) + 1193.068(collegeGPA) + 155.09(English) + 144.618(Logical) + 167.789(Quant) + 24416.461(Domain) + 70.416(ComputerProgramming)$$

- Sử dụng phương thức predict() cho mô hình để dự đoán kết quả dữ liệu X_test_d, sau đó đánh giá giá trị nhận được bằng hàm mean_absolute_error() (MAE) giữa giá trị dự báo y_new và giá trị thực y_test_d.

```
104446.3670545784
```

- Kết quả ước lượng sai số theo MAE: 104446.367

4) Nguồn tham khảo

GitHub - Applied Mathematics and Statistics
GitHub - Linear Regression Project
Machine Learning Mastery - K Fold Cross Validation
GitHub - K Fold Cross Validation