KHOA HỌC WEB

PROJECT 3 – PHÂN TÍCH DỮ LIỆU WEB

Giảng Viên: **thầy Lê Ngọc Thành**

Thành viên nhóm:

20424027 – Phạm Thi Minh Hậu

20424056 – Nguyễn Thế Ngọc

20424058 – Nguyễn Văn Nhật

Mục lục

1. Môi trường, thư viện

- Link drive 3 đồ án của nhóm: https://drive.google.com/drive/folders/1HXb2XEt5mfVIRtzOZarSnEXAKTaTR6_y?usp=sharing
- Ngôn ngữ: Python 3.8.5
- Định dạng file: . ipynb
- Phần mềm sử dụng: JupyterLab 2.2.6 trong Anaconda
- Hướng dẫn chạy: cài đầy đủ các thư viện bên dưới (bản mới nhất càng tốt), sau đó mở notebook, chọn Reset kernal and run all cells
- Định dạng Dữ liệu nhóm tổ chức lưu: JSON
- Các thư viện nhóm sử dụng:
  - matplotlib
  - numpy
  - sklearn
  - json
- Dữ liệu nhóm sử dụng: file dữ liệu JSON từ project 2

2. Công việc mỗi thành viên

MSSV – Họ Tên	Công việc
20424027 – Phạm Thị Minh Hậu	Sử dụng Phương pháp Knn cho bài toán số 5 Sử dụng linear cho bài toán số 1
20424056 – Nguyễn Thế Ngọc	Sử dụng Neural network cho bài toán số 4 Sử dụng linear cho bài toán số 2 và 3
20424058 – Nguyễn Văn Nhật	Sử dụng phương pháp Knn cho bài toán số 5 Sử dụng linear cho bài toán số 1

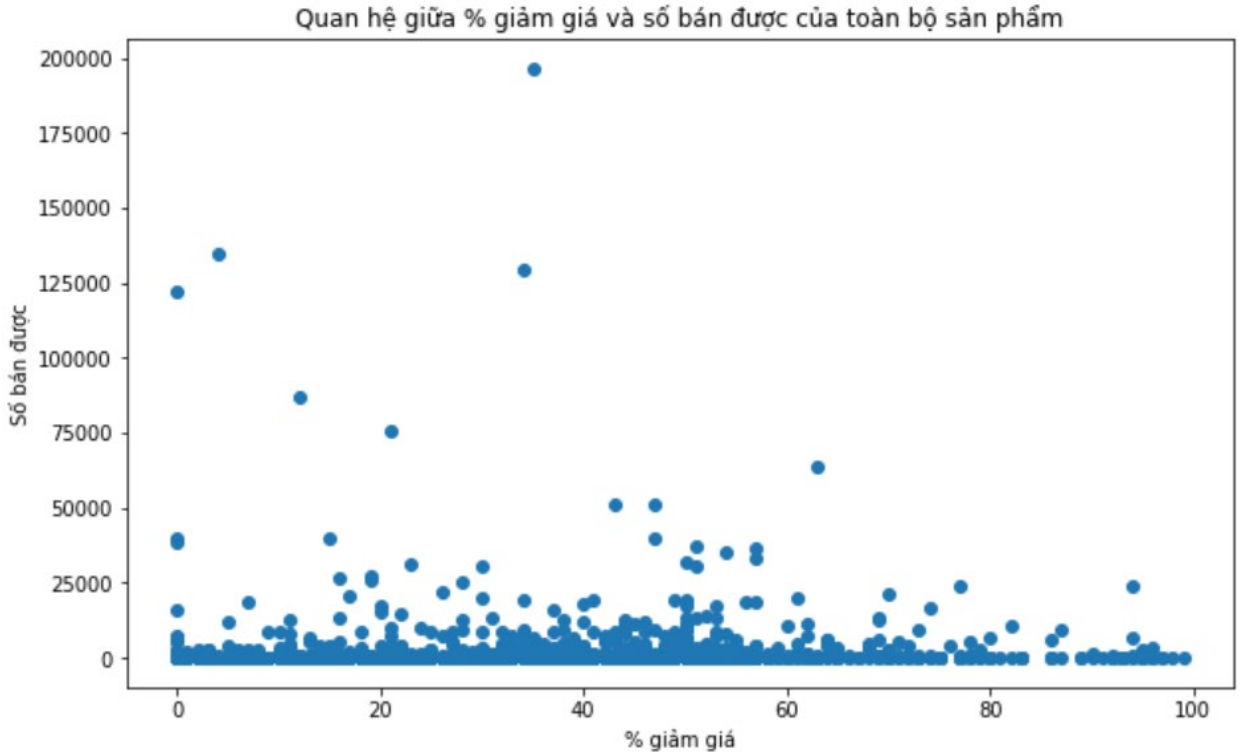
3. Mức độ hoàn thành tổng thể và đánh giá của nhóm

Các tiêu chí đánh giá	Điểm tối đa (%)	Nhóm đánh giá (%)
Đặt ra các vấn đề cần giải quyết	10%	10%
Mô tả dữ liệu liên quan	5%	5%
Chọn lựa, giải thích tính phù hợp của các mô hình máy học trên dữ liệu và bài toán nêu ra	10%	10%
Thực hiện huấn luyện và mô tả chi tiết từng thuật toán để triển khai huấn luyện	25%	25%
Mô tả cách phân chia dữ liệu huấn luyện và kiểm thử, lý giải phân chia và chứng tỏ kết quả không quá phụ thuộc vào cách phân chia đó	10%	10%
Giải thích các độ đo để đánh giá mô hình	5%	5%
Phân tích và trực quan hóa kể quả thu được, lý giải các điểm quan trọng	25%	20%
Kết luận về vấn đề nêu ra ban đầu	10%	5%
Tổng	100%	90%

4. Bài toán số 1

Đặt vấn đề và mô tả dữ liệu liên quan

Kết quả từ project 2, tương quan giữa % giảm giá và số bán được



Nhóm Nhận thấy, nếu ta loại bỏ một vài giá trị mặc định của các sản phẩm không bán được (bán được = 0) thì có lẽ chúng sẽ có tương quan nghịch, nhóm muốn giả sử rằng, liệu ta có thể học và dự đoán doanh số bán tốt dựa vào %giảm giá hay không. Kết quả có lẽ sai số khá vì số bán được sẽ là một con số không theo định luật.

Chọn giải thích tính phù hợp của các mô hình máy học trên dữ liệu bài toán

Nhóm sẽ sử dụng mô hình linear regression để áp dụng cho bài toán này, nhóm muốn tìm ra đáp án rằng liệu chúng có đúng với mong muốn của ta, liệu số bán có cao không.

Thực hiện huấn luyện, mô tả chi tiết

Nhóm sử dụng thư viện Sklearn để sử dụng mo hình linearRegression

tập x sẽ là mảng các % giảm giá, tập y là mảng các số bán được, X[i] sẽ ứng với y[i]

```python
# lấy data cần train, phỏng đoán
# y là cái ta cần phỏng đoán
# bài 1: x là % giảm giá, y là số bán được
# luôn lấy 20% đuôi làm test
x, y = discount_data.copy(), sold_data.copy()
tail = len(x)*20//100
x = x.reshape(-1,1)
print("số phần tử để test:", tail)

# tách data ra thành phần train và phần kiểm tra
# lấy tail phần tử cuối cho test, còn lại là train
x_train = x[:-tail]
x_test = x[-tail:]

y_train = y[:-tail]
y_test = y[-tail:]

# sử dụng modal linear regression từ thư viện sklearn
regr = linear_model.LinearRegression()

# cho học dữ liệu train đã định sẵn
regr.fit(x_train, y_train)
```

Mô tả cách phân chia dữ liệu huấn luyện

Nhóm sử dụng 20% dữ liệu phần cuối data từ project để làm dữ liệu test.

Đây là cách chia dữ liệu cơ bản ở mức độ học tập, Nhóm không sử dụng phân chia validation-test-train, nhóm chỉ sử dụng test 20% và train 80%

```python
x, y = discount_data.copy(), sold_data.copy()
tail = len(x)*20//100
x = x.reshape(-1,1)
print("số phần tử để test:", tail)
```

Giải thích các độ đo

Nhóm sử dụng độ đo 2r, mean squared error, intercept, Coefficients, Score

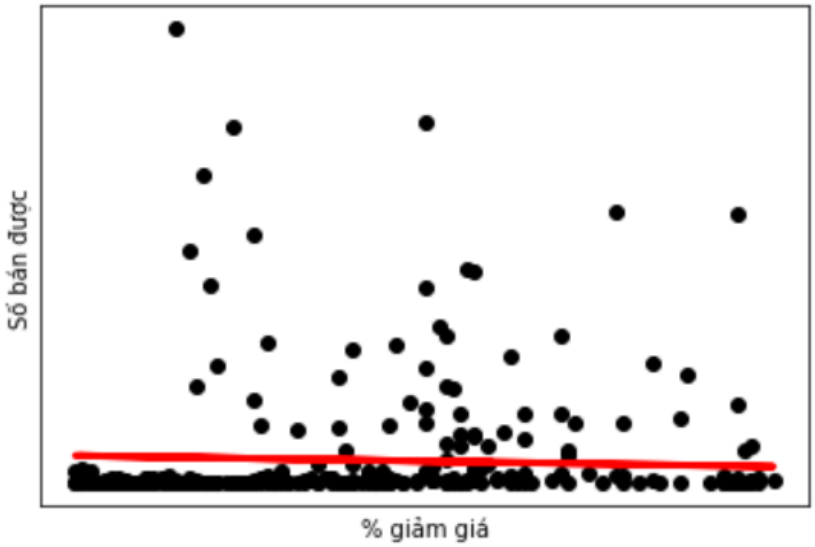
Sử dụng chính là độ đo sai lệch 2r độ đo score. nó giúp đính giá mô hình có chất lượng hay không, với 2R và score, càng gần 0 càng tốt

```
số phần tử để test: 327
Coefficients:  [-9.89280712]
intercept: 2378.570057905246
Mean squared error: 27898493.53
Coefficient of determination: -0.02
score: 0.00039085226634727466
```

Phân tích kết quả thu được

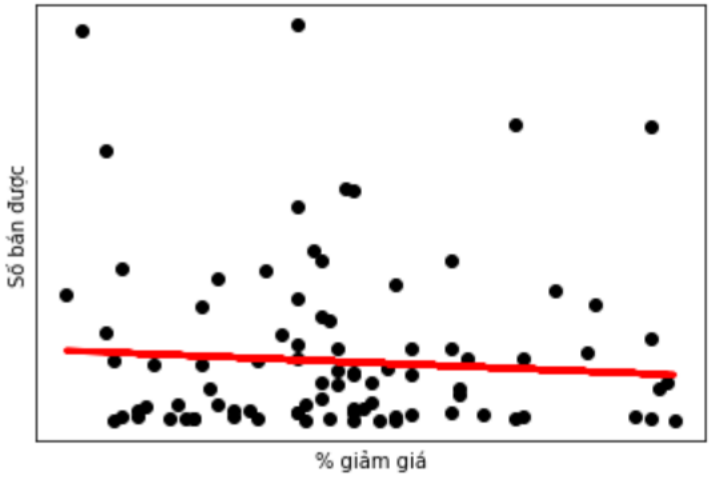
Chạy lần đầu


```
số phần tử để test: 327
Coefficients:  [-9.89280712]
intercept: 2378.570057905246
Mean squared error: 27898493.53
Coefficient of determination: -0.02
score: 0.00039085226634727466
```

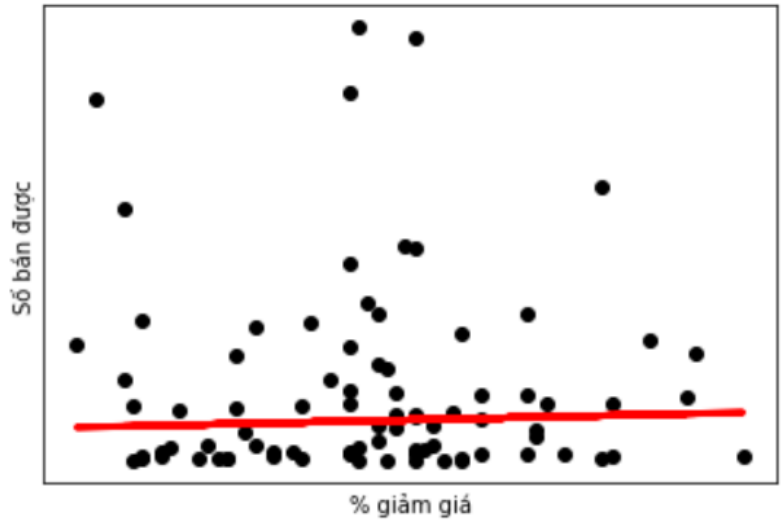
Sau khi loại bỏ các sản phẩm chiếm quá nhiều với số bán thấp và % giảm thấp

```
số phần tử để test: 87
Coefficients:  [-25.34058483]
Mean squared error: 48091013.91
Coefficient of determination: -0.01
score: 0.0005816299996145302
```



Thử tiếp tục loại bỏ các outlier bằng Zscore rồi chạy lại mô hình

```
số phần tử để test: 84
Coefficients:  [17.98332986]
Mean squared error: 75407150.77
Coefficient of determination: -0.11
score: 0.0018726465008461757
```



Nhận xét: khi ta loại bỏ các outlier, các data ta cho rằng nó ngoại lai, ta đã làm Score gần 1 hơn thay vì xa 1 gần 0 hơn. Ta có thể kết luận rằng, Phương pháp outlier sử

dụng cho bài toán này chưa được tốt lắm nhưng đủ để chứng minh rằng ta cố sửa dữ liệu cho tương quan thuận, kết quả dự đoán sẽ đi sai hơn.

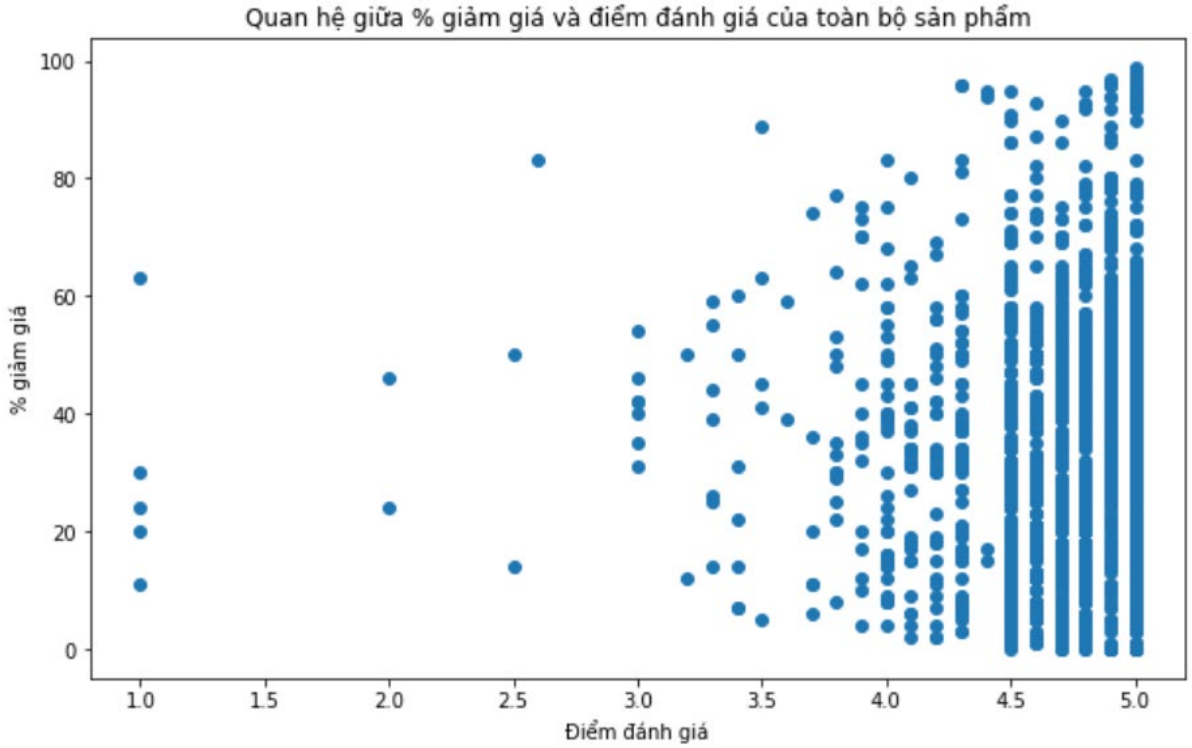
Kết luận về vấn đề đặt ra ban đầu

Mô hình không giải quyết hoàn toàn nhưng giải quyết một phần bài toán, có thể tạm kết luận rằng số bán được cao sẽ tương quan thuận với % giảm giá nhưng rất ít, khi ta cố ép đổi data thành tương quan nghịch, dự đoán sẽ có tỷ lệ sai cao hơn.

5. Bài toán số 2

Đặt vấn đề và mô tả dữ liệu liên quan

Sau khi trực quan hóa dữ liệu ở project 2



Nhóm nhận thấy Nếu dữ liệu trên ta loại các outlier, %giảm giá và điểm đánh giá có thể có tương quan thuận, Nhóm đặt vấn đề rằng ta muốn dự đoán điểm đánh giá sản phẩm dựa vào %giảm giá.

Chọn giải thích tính phù hợp của các mô hình máy học trên dữ liệu bài toán

Nhóm chọn mô hình linear regression để dự đoán, so với bài 1, bài 2 này con số % giảm giá sẽ dao động 0~100, điểm đánh giá sẽ từ 0.0~5.0, nhóm tin rằng như vậy khi ta áp dụng sẽ cho ra kết quả tốt hơn bài 1

Thực hiện huấn luyện, mô tả chi tiết

Tương tự bài 1, bài 2 sử dụng thư viện Sklearn để sử dụng mo hình linearRegression

tập x sẽ là mảng các % giảm giá, tập y là mảng các điểm đánh giá, X[i] sẽ ứng với y[i]

```
# lấy data cần train, phỏng đoán
# y là cái ta cần phỏng đoán
# bài 2: x là % giảm giá, y là điểm đánh giá
x, y = discount_data.copy(), point_data.copy()

tail = len(x)*20//100
print("số phần tử để test:", tail)
x = x.reshape(-1,1)

# tách data ra thành phần train và phần kiểm tra
# lấy tail phần tử cuối cho test, còn lại là train
x_train = x[:-tail]
x_test = x[-tail:]

y_train = y[:-tail]
y_test = y[-tail:]


# sử dụng modal linear regression từ thư viện sklearn
regr = linear_model.LinearRegression()

# cho học dữ liệu train đã định sẵn
regr.fit(x_train, y_train)
```

Mô tả cách phân chia dữ liệu huấn luyện

Nhóm sử dụng 20% dữ liệu phần cuối data từ project để làm dữ liệu test.

Đây là cách chia dữ liệu cơ bản ở mức độ học tập, Nhóm không sử dụng phân chia validation-test-train, nhóm chỉ sử dụng test 20% và train 80%

```
x, y = discount_data.copy(), sold_data.copy()
tail = len(x)*20//100
x = x.reshape(-1,1)
print("số phần tử để test:", tail)
```

Giải thích các độ đo

Nhóm sử dụng độ đo 2r, mean squared error, intercept, Coefficients, Score

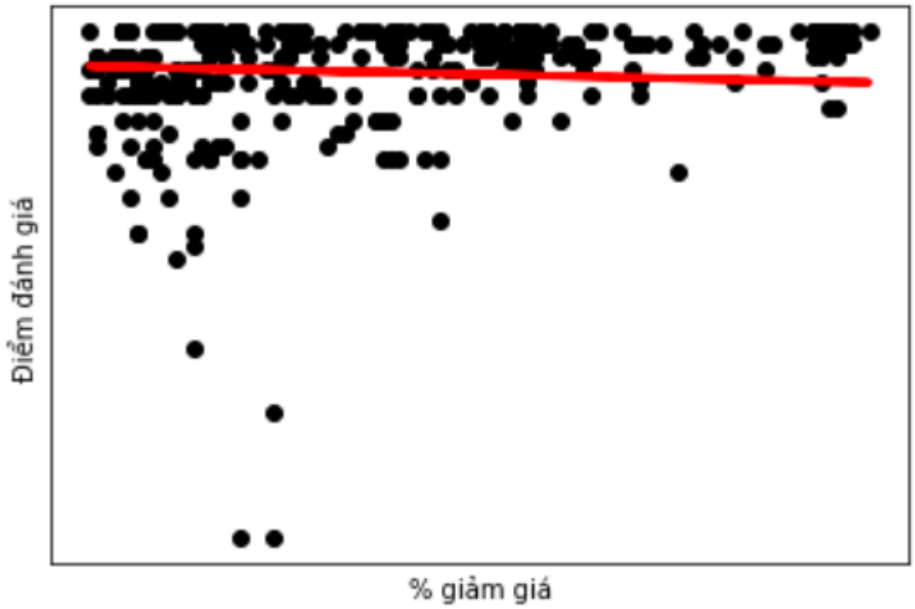
Sử dụng chính là độ đo sai lệch 2r độ đo score. nó giúp đính giá mô hình có chất lượng hay không, với 2R và score, càng gần 0 càng tốt

```
số phần tử để test: 327
Coefficients:  [-0.00130921]
Mean squared error: 0.25
Coefficient of determination: -0.05
score: 0.004256688789116891
```

Phân tích kết quả thu được

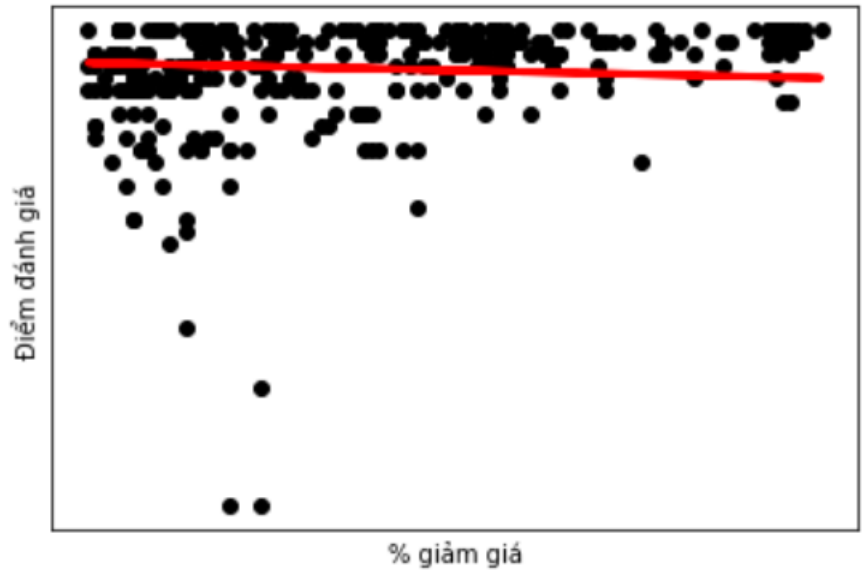
Loại bỏ outlier bằng Zscore

```
số phần tử để test: 327
Coefficients:  [-0.00130921]
Mean squared error: 0.25
Coefficient of determination: -0.05
score: 0.004256688789116891
```



Không loại bỏ outlier

```
số phần tử để test: 327
Coefficients:  [-0.00130921]
Mean squared error: 0.25
Coefficient of determination: -0.05
score: 0.0042566887891170024
```



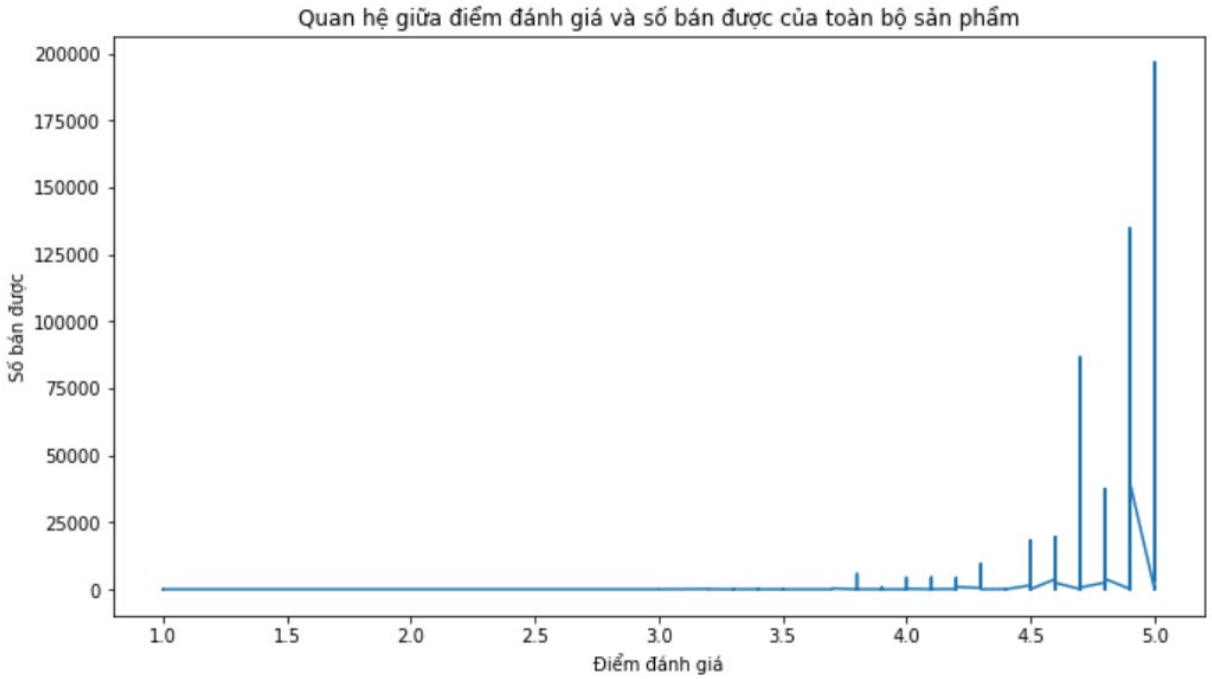
Nhận xét: Sử dụng Zscore để loại bỏ outlier trên dữ liệu này không hiệu quả, không có dữ liệu nào được loại bỏ, ta thấy rằng score của bài toán 2 cho ra gần 1 hơn so với dữ liệu gốc của bài 1, độ sai nhiều hơn.

Kết luận về vấn đề đặt ra ban đầu

Với kỳ vọng rằng điểm đánh giá cao thì %giảm giá cao sẽ sai, sau khi cho học kiểm tra trên data của nhóm, kết quả vẫn khẳng định rằng chúng có độ tương quan nghịch tuy không nhiều. Sau khi học và kiểm thử, có thể nói rằng bài toán tạm thời giải quyết được câu hỏi của nhóm, tuy nhiên sẽ cần thêm rất rất nhiều data để cho kết quả chính xác hơn.

6. Bài toán số 3

Đặt vấn đề và mô tả dữ liệu liên quan



kết quả trên là từ project 2 của nhóm, nhìn qua hình ta có thể thấy, điểm đánh giá có tương quan thuận với số sản phẩm bán ra. Nhóm đặt vấn đề rằng liệu ta có thể dựa vào số điểm đánh giá mà dự đoán được số lượng bán ra hay không.

Chọn giải thích tính phù hợp của các mô hình máy học trên dữ liệu bài toán

Với bài toán này nhóm sẽ chọn mô hình Linear regression, vì trong thực tế, thường các sản phẩm có đánh giá tốt, số bán ra cũng sẽ tương đối tốt, khi ta sử dụng mô hình linear sẽ đánh giá được câu hỏi này.

Thực hiện huấn luyện, mô tả chi tiết

Tập x sẽ là mảng các điểm đánh giá, tập y là mảng các số bán ra, X[i] sẽ ứng với y[i]

```
# lấy data cần train, phỏng đoán
# y là cái ta cần phỏng đoán
# bài 3: x là điểm đánh giá, y là số bán ra
x, y = point_data.copy(), sold_data.copy()

tail = len(x)*20//100
print("số phần tử để test:", tail)
x = x.reshape(-1,1)

# tách data ra thành phần train và phần kiểm tra
# lấy tail phần tử cuối cho test, còn lại là train
x_train = x[:-tail]
x_test = x[-tail:]

y_train = y[:-tail]
y_test = y[-tail:]


# sử dụng modal linear regression từ thư viện sklearn
regr = linear_model.LinearRegression()

# cho học dữ liệu train đã định sẵn
regr.fit(x_train, y_train)

# cho phỏng đoán từ từ tập data đã tách ra từ đầu dựa vào X, ta sẽ phỏng đoán Y
y_predict = regr.predict(x_test)
```

Mô tả cách phân chia dữ liệu huấn luyện

Nhóm sử dụng 20% dữ liệu phần cuối data từ project để làm dữ liệu test.

Đây là cách chia dữ liệu cơ bản ở mức độ học tập, Nhóm không sử dụng phân chia validation-test-train, nhóm chỉ sử dụng test 20% và train 80%

```
# tách lấy 20% dưới làm test
x, y = discount_data.copy(), sold_data.copy()
tail = len(x)*20//100
x = x.reshape(-1,1)
print("số phần tử để test:", tail)
```

Giải thích các độ đo

Nhóm sử dụng độ đo 2r, mean squared error, intercept, Coefficients, Score

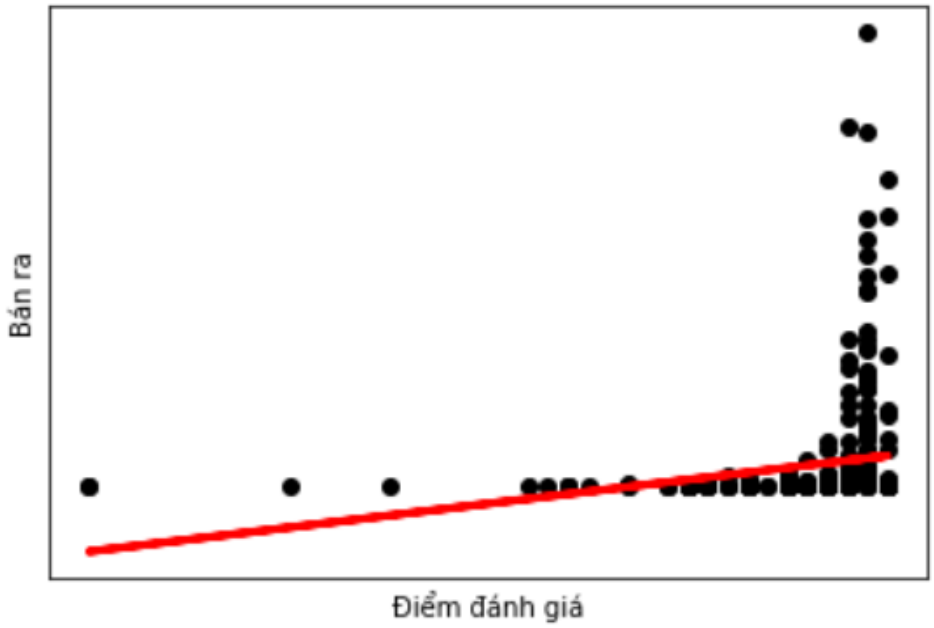
Sử dụng chính là độ đo sai lệch 2r độ đo score. nó giúp đính giá mô hình có chất lượng hay không, với 2R và score, càng gần 0 càng tốt

```
số phần tử để test: 327
Coefficients:  [2114.15369377]
Mean squared error: 26536265.76
Coefficient of determination: 0.03
score: 0.007187753276994724
```

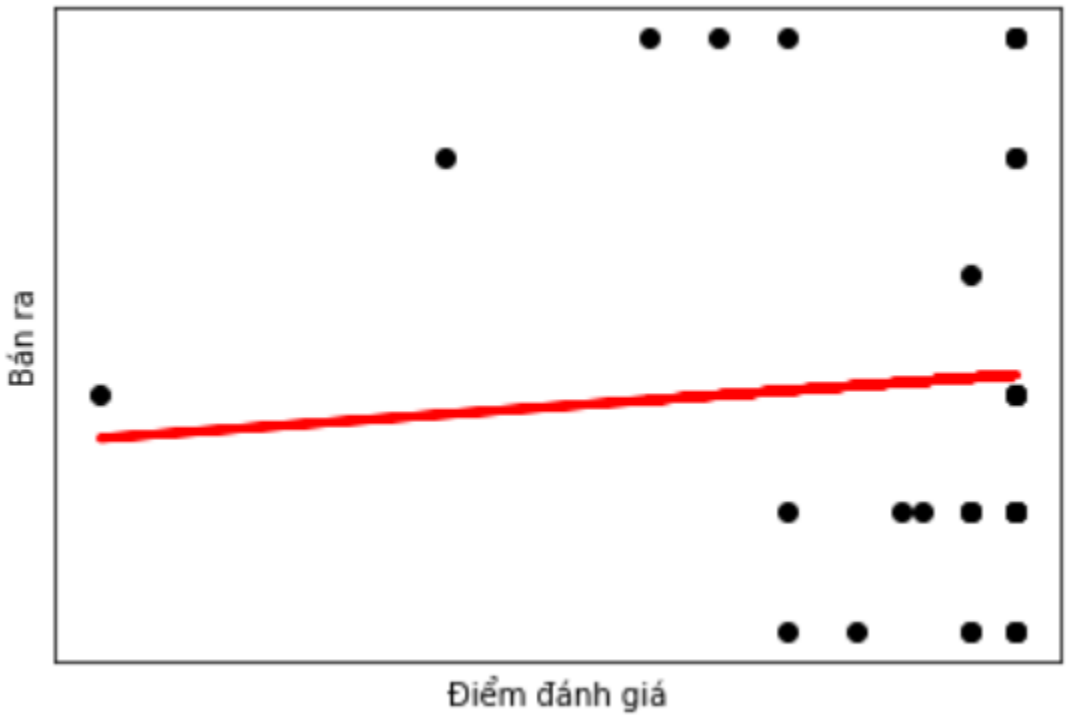
Phân tích kết quả thu được

Khi không loại bỏ các outlier bằng Zscore:

```
số phần tử để test: 327
Coefficients:  [2114.15369377]
Mean squared error: 26536265.76
Coefficient of determination: 0.03
score: 0.007187753276994724
```



Khi ta loại bỏ Outlier bằng Score:



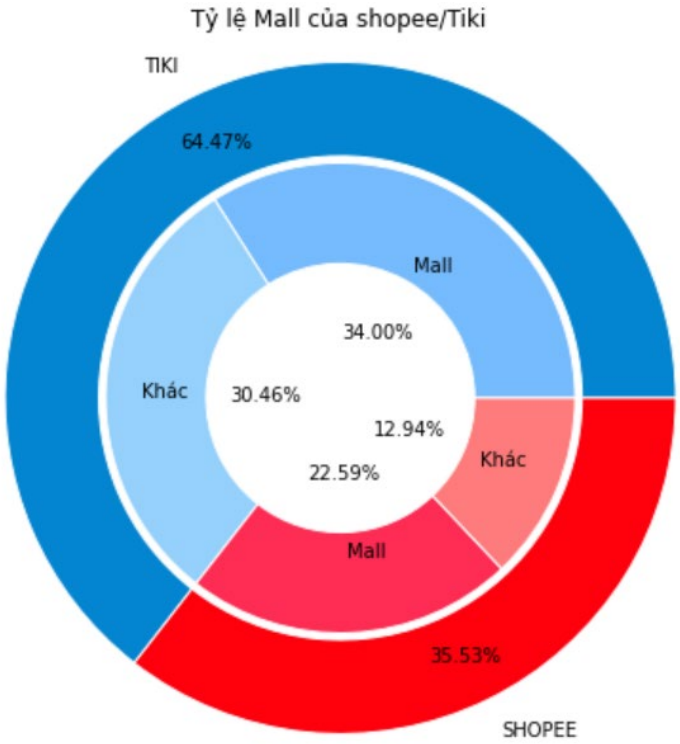
Nhận xét: khá bất ngờ khi với bài toán số 3, các outlier bị loại bỏ gần như 10 lần, loại rất nhiều so với 2 bài toán đầu tiên của nhóm. Về kết quả ta có thể thấy rằng sau khi loại bỏ outlier, kết quả bài toán cho ra tốt hơn, Score sẽ càng xa 1 và gần 0 hơn.

Kết luận về vấn đề đặt ra ban đầu

Qua bài toán số 3, kết quả trực quan và score thu thập được tốt hơn 2 bài toán đầu, có thể kết luận rằng bài toán số 3 giải quyết được câu hỏi nhóm đặt ra rằng điểm đánh giá có ảnh hưởng tới doanh số bán ra, và nó đạt kỳ vọng nhóm đặt ra trong thực tế là chúng thường tương quan thuận.

7. Bài toán số 4

Đặt vấn đề và mô tả dữ liệu liên quan



Từ vấn đề của bài toán số 3, nhóm muốn nâng cấp, nhóm muốn đặt ra vấn đề rằng: liệu từ số bán ra và đán giá, ta có thể dự đoán sản phẩm đó được bán từ 1 shop mall hay không.

Chọn giải thích tính phù hợp của các mô hình máy học trên dữ liệu bài toán

Với bài toán này, dữ liệu không còn 2 chiều nữa, Nhóm sẽ sử dụng mô hình Mạng Nơ ron (neural network) để giải, nó có phù hợp vì tính tương quan 2 trường số bán + điểm đánh giá đã được chứng minh ở bài toán số 3, bài neural này nhóm sẽ thử nghiệm nhiều tầng khác nhau, nhiều nơ ron khác nhau.

Thực hiện huấn luyện, mô tả chi tiết

```
x = []
for i in range(len(point_data)):
    x.append([point_data[i], sold_data[i]])
y = mall_data.copy()
```

```
train_x, test_x, train_y, test_y = train_test_split(x,y, test_size=0.2, random_state=0, stratify=y)

# 2 tầng ,mỗi tầng ảnh 3 noron, hàm giải là tanh, bộ tối ưu sử dụng lbfgs, số vòng lập tối đa là 100000
neural_net_model = MLPClassifier(hidden_layer_sizes=(3,3,), activation="tanh", solver="lbfgs", max_iter=100000, random_state=0)

#train
neural_net_model.fit(train_x, train_y)

```

Nhóm sẽ thử nghiệm X tầng nơ ron với Y số nơ ron mỗi tầng, để học tập cơ bản nhóm sẽ không dùng các kiến trúc Cao cấp, chỉ sử dụng và thử nghiệm ở mức nào đó.

Hàm kích hoạt Nhóm sử dụng xuyên suốt là hàm tanh, số lần lập tối đa nếu nó không hội tụ được là 100 nghìn lần.

Nhóm sử dụng data Y sẽ là mảng 0 hoặc 1, với 0 là cửa hàng thường, 1 là cửa hàng mall.

với data X được sử dụng là mảng kết hợp của Số bán được và điểm đánh giá.

Ví dụ sản phẩm 1 có điểm là 4, số bán được 100, bán từ cửa hàng mall thì:

x[1] = [4,100]

y[1] = 1

Mô tả cách phân chia dữ liệu huấn luyện

Từ dữ liệu gốc, nhóm xài hàm Train_test_split để phân ra làm 80-20, 20 sẽ dành cho test và 80 dành cho train

```
x = []
for i in range(len(point_data)):
    x.append([point_data[i], sold_data[i]])
y = mall_data.copy()
```

```
train_x, test_x, train_y, test_y = train_test_split(x,y, test_size=0.2, random_state=0, stratify=y)
```

Giải thích các độ đo

Khác với 3 bài đầu, bài 4 nhóm sử dụng độ đo lỗi trung bình của tập train và tập test để xem kết quả từng X tầng Y nơ ron sẽ cho kết quả nào tốt nhất.

```
# độ lỗi trên train
print("độ lỗi tập train, 2 lớp 3 noron")
print(np.mean(train_y != neural_net_model.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi tập test, 2 lớp 3 noron")
np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi tập train, 2 lớp 3 noron
0.4251908396946565
độ lỗi tập test, 2 lớp 3 noron
0.43597560975609756
```

Phân tích kết quả thu được

Đầu tiên là thử 2 layer ẩn, mỗi layer 3 nơ ron xem thế nào (chưa tính layer in và output)

```
#train
neural_net_model.fit(train_x, train_y)

# độ lỗi trên train
print("độ lỗi tập train, 2 lớp 3 noron")
print(np.mean(train_y != neural_net_model.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi tập test, 2 lớp 3 noron")
np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi tập train, 2 lớp 3 noron
0.4251908396946565
độ lỗi tập test, 2 lớp 3 noron
]: 0.43597560975609756
```

Ta thử tăng số nơ ron lên 9

```
5
6  #train
7  neural_net_model.fit(train_x, train_y)
8
9  # độ lỗi trên train
10 print("độ lỗi train, 2 lớp 9 noron")
11 print(np.mean(train_y != neural_net_model.predict(train_x)))
12
13 # độ lỗi ngoài tập huấn luyện
14 print("độ lỗi test, 2 lớp 9 noron")
15 np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi train, 2 lớp 9 noron
0.4068702290076336
độ lỗi test, 2 lớp 9 noron
0.42378048780487804
```

Kết quả tập test có vẻ độ lỗi giảm, ta thử tăng số lớp lên 3, giảm số nơ ron mỗi lớp là 6

```
6  #train
7  neural_net_model.fit(train_x, train_y)
8
9  # độ lỗi trên train
10 print("độ lỗi train, 3 lớp 6 noron")
11 print(np.mean(train_y != neural_net_model.predict(train_x)))
12
13 # độ lỗi ngoài tập huấn luyện
14 print("độ lỗi test, 3 lớp 6 noron")
15 np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi train, 3 lớp 6 noron
0.41908396946564885
độ lỗi test, 3 lớp 6 noron
0.4176829268292683
```

Kết quả đo lỗi trên tập test tiếp tục giảm, ta thử tăng số nơ ron lên 9:

```
# độ lỗi trên train
print("độ lỗi train, 3 lớp 9 noron")
print(np.mean(train_y != neural_net_model.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi test, 3 lớp 9 noron")
np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi train, 3 lớp 9 noron
0.4198473282442748
độ lỗi test, 3 lớp 9 noron
0.4146341463414634
```

Kết quả đo lỗi trên train vẫn là 0.42 nhưng trên tập test độ lỗi lại giảm thêm 0.003, ta tiếp tục tăng số lớp nơ ron lên 4, giảm số nơ ron mỗi lớp còn 6.

```
# độ lỗi ngoài tập huấn luyện
print("độ lỗi test, 4 lớp 6 noron")
np.mean(test_y != neural_net_model.predict(test_x))
```

```
độ lỗi train, 4 lớp 6 noron
0.38625954198473283
độ lỗi test, 4 lớp 6 noron
0.4268292682926829
```

Lần này có vẻ như kết quả thu được trên tập test không còn giảm nữa mà lại tăng.

Nhận xét: độ lỗi của tập train chuyển biến không đều, giảm rồi tăng rồi giảm khi ta tăng số lớp và tăng số nơ ron mỗi lớp. Tuy nhiên độ lỗi tập test đáng kỳ vọng, khi ta tăng tới mức 3 lớp và 9 nơron, so với ban đầu 2 lớp 3 nơ ron, độ lỗi giảm dần theo thời gian, ta nên dừng ở 3 lớp vì khi thử 4 lớp, độ lỗi lại tăng.

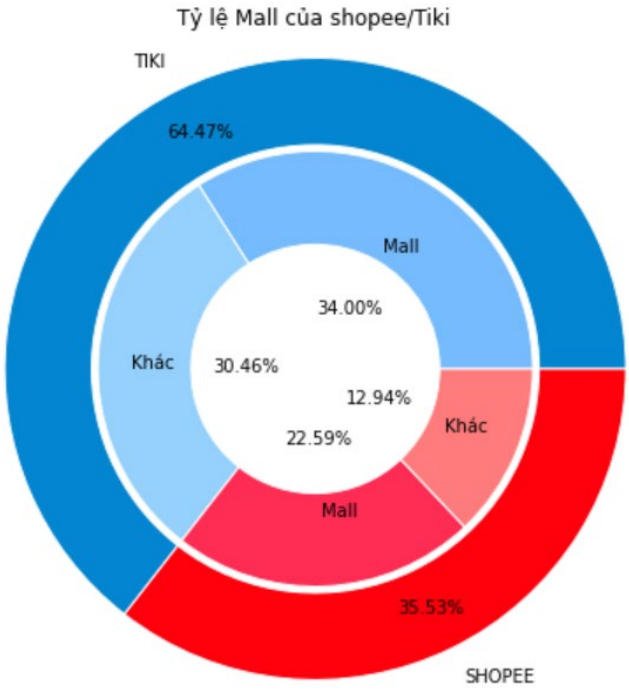
Kết luận về vấn đề đặt ra ban đầu

Sau khi thử nghiệm với mô hình mạng nơron, độ lỗi trên tập test bé nhất là 0.414 và né nhất trên tập train là 0.386, nhóm nhận xét rằng mô hình trả lời được 60% đúng là khá ổn với dữ liệu hạn chế và chưa được đa dạng của nhóm, Nhóm kết luận là vấn đề đặt ra ban đầu được giải quyết.

8. Bài toán số 5

Đặt vấn đề và mô tả dữ liệu liên quan

Tương tự bài 4, tuy nhiên, bài 5 sẽ dự đoán là nền tảng shopee hoặc tiki. Bài 5 này sẽ là bài nâng cấp từ bài toán số 3: dựa vào điểm đánh giá và số bán được để dự đoán đó là nền tảng nào.



Chọn giải thích tính phù hợp của các mô hình máy học trên dữ liệu bài toán

Nhóm sử dụng mô hình Knn để tìm câu trả lời cho bài toán số 5 này, với số K lẻ nhóm sẽ thay đổi liên tục để thử nghiệm, nhóm chọn mô hình này vì 2 dữ liệu đầu vào là dữ liệu số và nhóm cần xem dữ liệu cần dự đoán "gần" shopee hơn hay "gần" tiki hơn.

Thực hiện huấn luyện, mô tả chi tiết

```
# x sẽ dùng data được tạo ở đầu bài neural
y = platform_data.copy()
train_x, test_x, train_y, test_y = train_test_split(x,y, test_size=0.2, random_state=0, stratify=y)

from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(train_x, train_y)
```

Nhóm sử dụng data Y sẽ là mảng tên các nền tảng, chỉ có 2 nền tảng được sử dụng là shopee hoặc tiki.

với data X được sử dụng là mảng kết hợp của Số bán được và điểm đánh giá.

Ví dụ sản phẩm 1 có điểm là 4, số bán được 100, nền tảng shopee thì:

x[1] = [4,100]

y[1] = ‘shopee’

Mô tả cách phân chia dữ liệu huấn luyện

Tương tự 4 bài trên, bài 5 sẽ chia theo tỷ lệ 80-20 , với 20 để test và 80 để train, nhóm không dùng validation nên sẽ không chia làm 3.

```
# x sẽ dùng data được tạo ở đầu bài neural
y = platform_data.copy()
train_x, test_x, train_y, test_y = train_test_split(x,y, test_size=0.2, random_state=0, stratify=y)
```

Nhóm sử dụng hàm từ thư viện là hàm train_test_split để chia, test_size là 20%.

Giải thích các độ đo

Với bài toán KNN nhóm sẽ sử dụng độ đo lỗi mean của tập train và tập test.

```
13
14  # độ lỗi ngoài tập huấn luyện
15  print("độ lỗi test")
16  np.mean(test_y != neigh.predict(test_x))
```

```
độ lỗi train
0.15725190839694655
độ lỗi test
0.3079268292682927
```

Phân tích kết quả thu được

Với K = 3

```
print("KNN với k = 3")
# độ lỗi trên train
print("độ lỗi train")
print(np.mean(train_y != neigh.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi test")
np.mean(test_y != neigh.predict(test_x))
```

```
độ lỗi train
0.15725190839694655
độ lỗi test
0.3079268292682927
```

Ta thử tăng K lên 7

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=7)
neigh.fit(train_x, train_y)

print("KNN với k = 7")
# độ lỗi trên train
print("độ lỗi train")
print(np.mean(train_y != neigh.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi test")
np.mean(test_y != neigh.predict(test_x))
```

```
KNN với k = 7
độ lỗi train
0.22595419847328244
độ lỗi test
0.2682926829268293
```

Ta thấy rằng, độ lỗi trên train tăng tuy nhiên độ lỗi trên tập test lại giảm, ta tiếp tục tăng K lên 15 xem thế nào

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=15)
neigh.fit(train_x, train_y)

print("KNN với k = 15")
# độ lỗi trên train
print("độ lỗi train")
print(np.mean(train_y != neigh.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi test")
np.mean(test_y != neigh.predict(test_x))
```

```
KNN với k = 15
độ lỗi train
0.25190839694656486
độ lỗi test
0.24085365853658536
```

Nhận thấy độ lỗi test và độ lỗi train đang khá tương đồng, ta sẽ note lại k = 15, ta tiếp tục tăng K lên 21 xem kết quả liệu có tương quan như nãy giờ hay không

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=21)
neigh.fit(train_x, train_y)

print("KNN với k = 21")
# độ lỗi trên train
print("độ lỗi train")
print(np.mean(train_y != neigh.predict(train_x)))

# độ lỗi ngoài tập huấn luyện
print("độ lỗi test")
np.mean(test_y != neigh.predict(test_x))
```

```
KNN với k = 21
độ lỗi train
0.2587786259541985
độ lỗi test
0.24390243902439024
```

Nhận xét, khi tăng K lên 21, độ lỗi test không giảm mà lại tăng, độ lỗi train cũng tăng, ta không nên giữ K = 21, K = 15 vẫn là tốt hơn.

Kết luận về vấn đề đặt ra ban đầu

Với vấn đề ban đầu nhóm đặt ra, Phương pháp Knn cho kết quả rất tốt, tốt hơn 4 bài toán trên, Sử dụng Knn với K =15 cho độ lỗi ổn định giữa train và test là khoảng 25% độ sai, dự đoán chính xác chiếm tới khoảng tiệm cận 75%, nhóm kết luận rằng sử dụng Knn giải quyết được câu hỏi của nhóm đặt ra.