TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN CUỐI KỲ

HỌC PHẦN: KHOA HỌC DỮ LIỆU

ĐỀ TÀI: DỰ ĐOÁN DOANH THU CỦA GAME

HỌ VÀ TÊN SINH VIÊN	LỚP HỌC PHẦN	ĐIỂM BẢO VỆ
NGUYỄN HỮU NGUYÊN	19Nh13	
TRẦN THỊ THANH NGA	19Nh13	
TRẦN HỒNG SƠN	19Nh13	

ĐÀ NẴNG, 06/2022

TÓM TẮT

Dựa vào dữ liệu rất lớn của các Game trên nhà phân phối Steam, nhóm đã lựa chọn đề tài Dự đoán doanh thu của Game. Sau khi nghiên cứu, nhóm đã quyết định dùng thư viện request và beautifulsoup để cào data; Label Encoder, Robust Scaling và Outliers Handling để trích xuất đặc trưng; hai mô hình là Linear Regression và SVR để dự đoán doanh thu Game; sau cùng là dùng độ đo RMSE, MAE để đánh giá các mô hình. Kết quả là nhóm đã cào được Data từ web, trích xuất đặc trưng cũng như dùng hai mô hình để đánh giá doanh thu. Tuy cả hai mô hình, đặc biệt là mô hình Linear Regression có độ chính xác còn thấp, chưa đủ tin cậy. Nhóm sẽ tiếp tục phát triển và hoàn thiện trong tương lai.

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tự đánh giá theo 3 mức (Đã hoàn thành/Chưa hoàn thành/Không triển khai)
Nguyễn Hữu Nguyên	Mô tả dữ liệu, EDA Xử lý ngoại lệ Xây dựng và đánh giá mô hình SVR So sánh hai mô hình (phụ)	Đã hoàn thành Đã hoàn thành Đã hoàn thành Đã hoàn thành
Trần Thị Thanh Nga	Xây dựng và đánh giá mô hình Linear regression So sánh hai mô hình (chính)	Đã hoàn thành Đã hoàn thành
Trần Hồng Sơn	Cào dữ liệu từ web Làm sạch dữ liệu Labelencode chuyển từ string sang float cho các đặt trưng cần thiết	Đã hoàn thành Đã hoàn thành Đã hoàn thành

MỤC LỤC

MỤC LỤC ẢNH

1. Giới thiệu

1.1 Các vấn đề cần giải quyết

- Làm sao để cào được dữ liệu từ web.
- Cách để trích xuất dữ liệu.
- Phương pháp để dự đoán doanh thu của game và độ chính xác quả phương pháp đó.

1.2 Giải pháp

- Sử dụng thư viện request, beautifulsoup phục vụ cho việc cào dữ liệu.
- Quan sát bằng mắt và chọn lọc, sau đó cào các đặt trưng đã chọn tương ứng với các thẻ HTML.
- Sử dụng hai mô hình học máy đó là SVR và Linear regression.

2. Thu thập và mô tả dữ liệu

2.1. Thu thập dữ liệu

Vấn đề : Trang Web được sử dụng để cào dữ liệu thì dữ liệu được binding vào trong html và không thể sử dụng API để cào.

Giải pháp : Sử dụng thư viện request để get trang web sau đó Sử dụng thư viện beautifulsoup để cào HTML của trang web về. Sơ qua về beautifulsoup là một gói Python để phân tích cú pháp các tài liệu HTML và XML. Nó tạo một cây phân tích cú pháp cho các trang được phân tích cú pháp có thể được sử dụng để trích xuất dữ liệu từ HTML.

Nguồn dữ liệu : https://games-stats.com/steam/

Công cụ thu thập : Visual Studio Code, python , thư viện beautifulsoup , thư viện request.

Cách thức thu thập :

B1 : Trong terminal sử dụng pip để install thư viện beautifulsoup và request với cú pháp (pip install beautifulsoup4) và (pip install requests).

B2: Lên trang web sử dụng Dev Tool của Chrome để xác định HTML cần lấy trong web.

B3 : Sử dụng thư viện request để get trang web.

```
r = requests.get(link_base)
```

B4 : Đọc HTML của trang web thông qua hàm của thư viện beautifulsoup.

```
soup = BeautifulSoup(r.text, 'html.parser')
```

B5 : Sau đó chúng ta có thể lấy bất cứ text của bất cứ thẻ nào mà chúng ta muốn.

Ví dụ: Để có thể lấy hết thẻ body in ra màn hình ta dùng hàm.

```
data = soup.find_all('tbody')
```

Hình 1:*Kết quả trong terminal*

B6 : Từ đây mình có thể lấy được text có trong body.

```
print(data[0].text)
```

```
Next




Rust
Steam

                        Publisher:
                        Facepunch Studios

                        Developer:
                        Facepunch Studios



Feb 8, 2018
4 years, 4 months ago
```

Hình 2: *Kết quả trong terminal*

B7 : Chuyển list dữ liệu sang file csv

Input : Link web.

Output : 1 file raw csv.

2.2. Mô tả dữ liệu

Số mẫu : 1427 (raw)

Chiều dữ liệu : 1427 x 6

Số đặc trưng của mẫu: 6 đặc trưng

Bảng 1 : Mô tả dữ liệu.

	Name	Pulisher	Price	Net Revenue	Platform	Genres
kiểu dữ liệu	string	string	string	string	string	string
số dữ liệu trống	0	0	0	0	30	2

Dữ liệu raw thu thập được:

```
data = pd.read_csv('./raw.csv')
data.head()
```
✓ 0.8s

	Name	Pulisher	Price	Net Revenue	Platform	Genres
0	Grand Theft Auto V	Rockstar Games	$29.99	~$700 million	Windows	Action,Adventure
1	Cyberpunk 2077	CD PROJEKT RED	$59.99	~$490 million	Windows	RPG
2	Rust	Facepunch Studios	$39.99	~$490 million	Windows Mac	Action,Adventure,Indie,MassivelyMultiplayer,RPG
3	The Witcher® 3: Wild Hunt	CD PROJEKT RED	$39.99	~$430 million	Windows	RPG
4	Tom Clancy's Rainbow Six® Siege	Ubisoft	$19.99	~$340 million	Windows	Action

***Hình 3**:Dữ liệu raw thu thập được*

Sau khi qua bước lọc bỏ ký tự non numberic ra khỏi đặc trưng Price và net Revenue:

```
data.head()
```

	Name	Pulisher	Price	Net Revenue	Platform	Genres
0	Grand Theft Auto V	Rockstar Games	29.99	700	Windows	Action,Adventure
1	Cyberpunk 2077	CD PROJEKT RED	59.99	490	Windows	RPG
2	Rust	Facepunch Studios	39.99	490	Windows Mac	Action,Adventure,Indie,MassivelyMultiplayer,RPG
3	The Witcher® 3: Wild Hunt	CD PROJEKT RED	39.99	430	Windows	RPG
4	Tom Clancy's Rainbow Six® Siege	Ubisoft	19.99	340	Windows	Action

***Hình 4:**Sau khi qua bước lọc bỏ ký tự non numberic ra khỏi đặc trưng Price và net Revenue*

Sau khi qua bước LabelEncode:

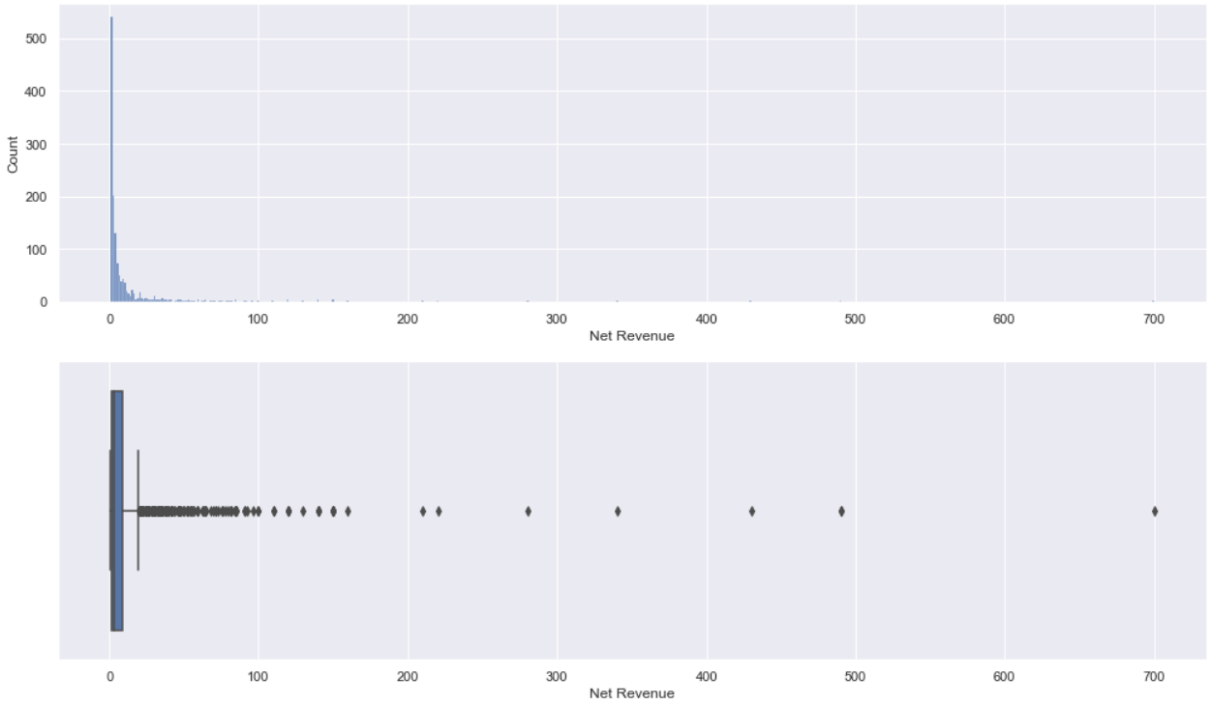
```
print(clean_df.head())
```

	Name	Pulisher	Price	Net Revenue	Platform	Genres
0	Grand Theft Auto V	422	29.99	700	0	1
1	Cyberpunk 2077	91	59.99	490	0	207
2	Rust	188	39.99	490	3	36
3	The Witcher® 3: Wild Hunt	91	39.99	430	0	207
4	Tom Clancy's Rainbow Six® Siege	549	19.99	340	0	0

***Hình 5:**Sau khi qua bước LabelEncode*

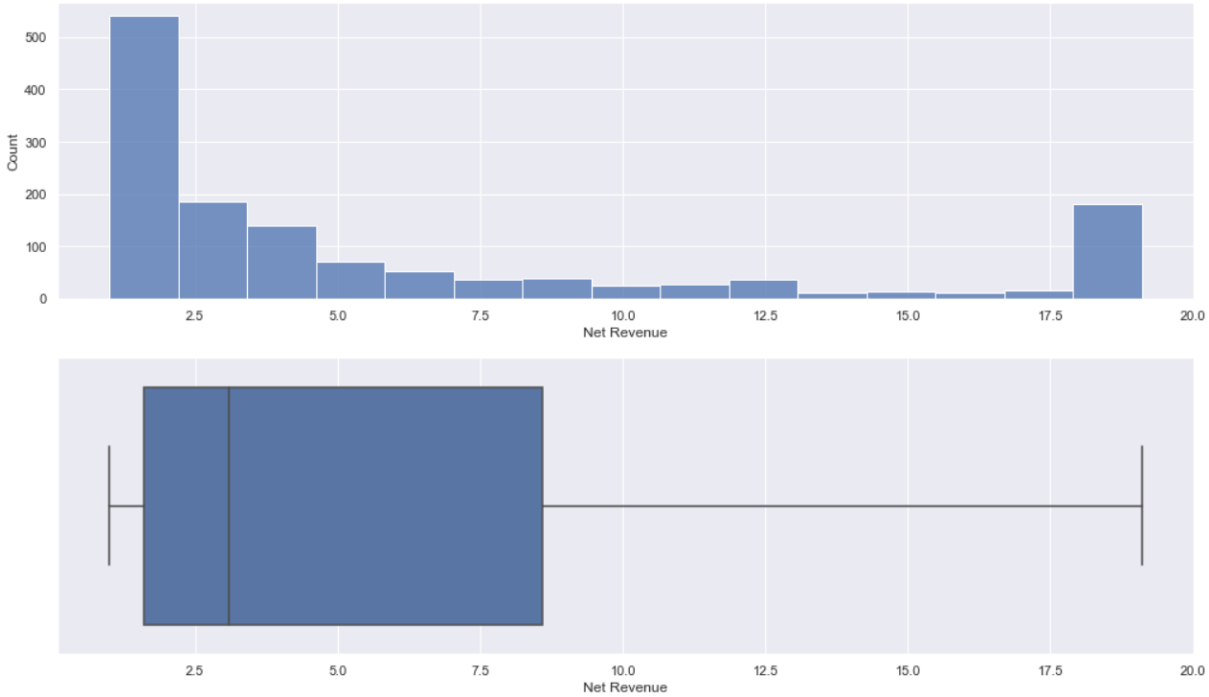
Đặc trưng Net Revenue trước khi xử lý ngoại lệ:

Có rất nhiều ngoại lệ, có phân bố lệch trái, tập trung chủ yếu từ 0 đến 20 triệu đô.



Hình 6:*Đặc trưng Net Revenue trước khi xử lý ngoại lệ*

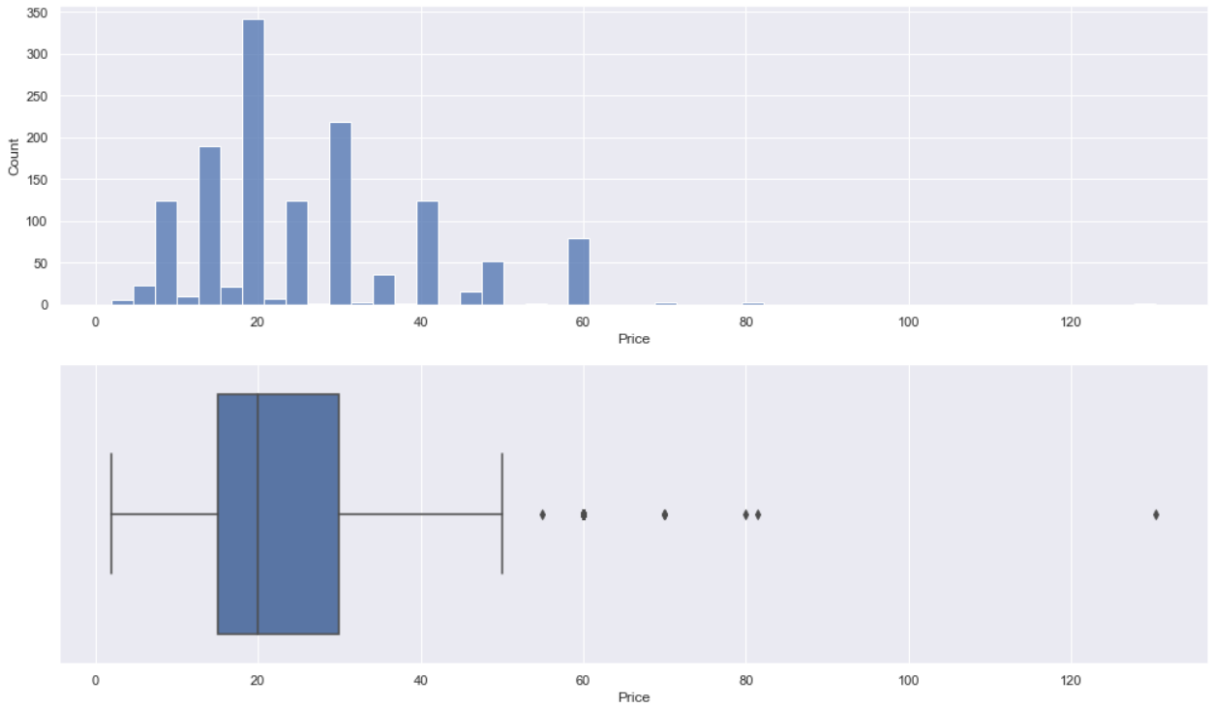
Đặc trưng Net Revenue sau khi xử lý ngoại lệ:



Hình 7: *Đặc trưng Net Revenue sau khi xử lý ngoại lệ*

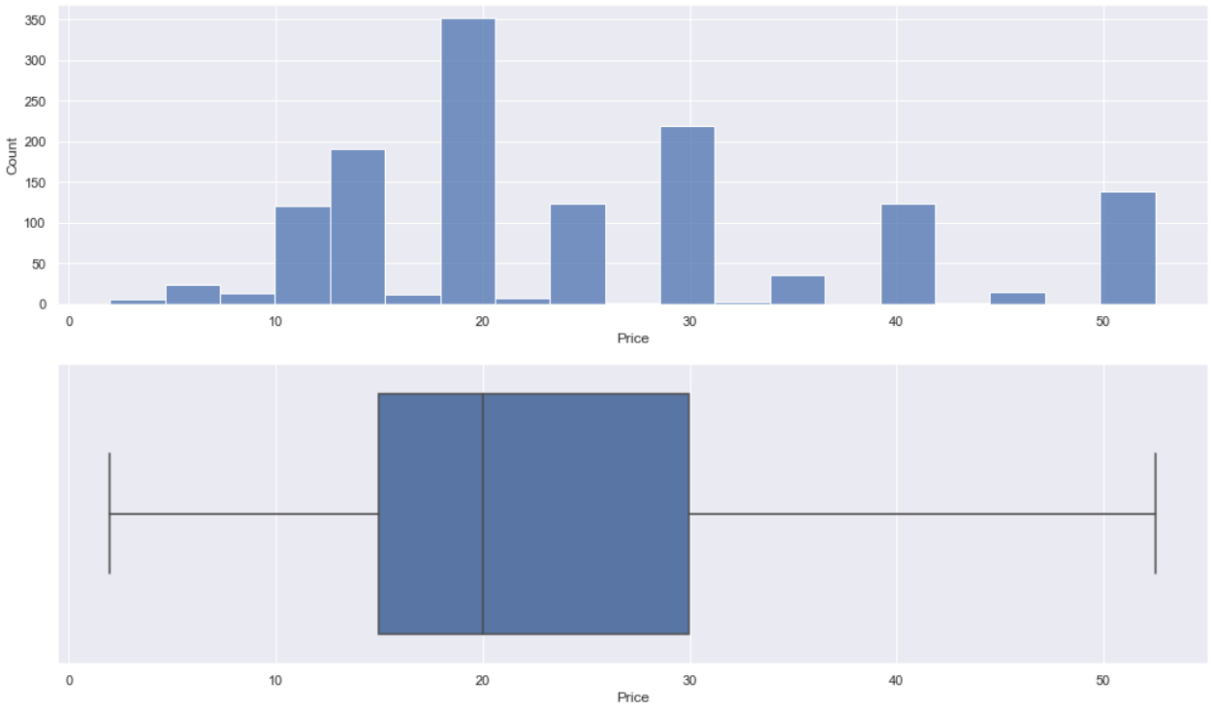
Đặc trưng Price trước khi xử lý ngoại lệ:

Chủ yếu tập trung từ 0 đến 60 đôla.



Hình 8:*Đặc trưng Price trước khi xử lý ngoại lệ*

Đặc trưng Price sau khi xử lý ngoại lệ:

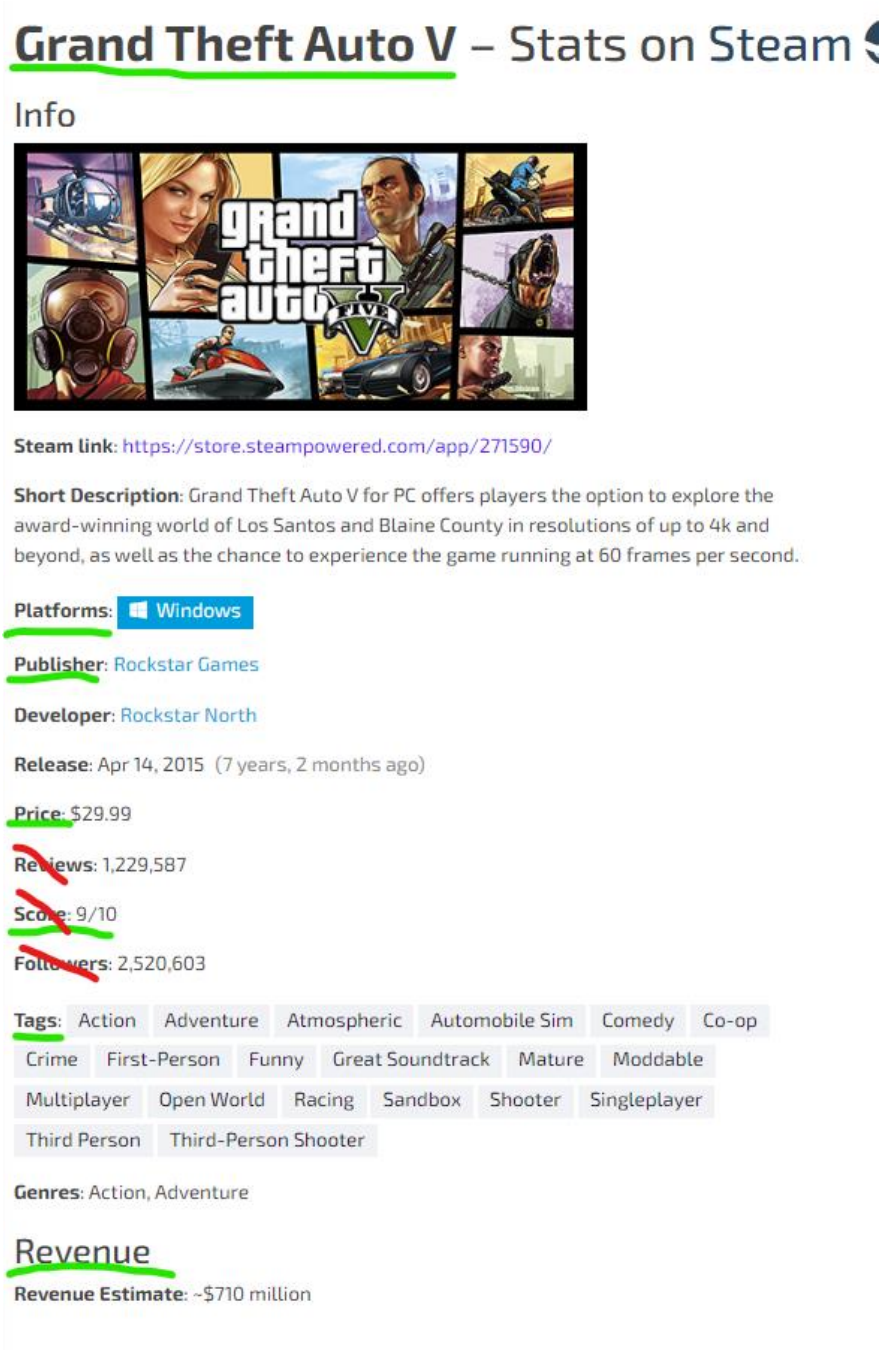


Hình 9*:Đặc trưng Price sau khi xử lý ngoại lệ*

Không có sự tương quan giữa hai đặc trưng Price và Net Revenue. Giá bán không ảnh hưởng nhiều đến doanh thu của game.

Hình 10: *Sự tương quan giữa hai đặc trưng Price và Net Revenue*

3. Trích xuất đặc trưng



Hình 11:*Trích xuất đặc trưng trong web*

Các đặc trưng được lựa chọn: Name, Platforms, Publisher, Price, Tags, Net Revenue.

Các đặc trưng không được lựa chọn: Reviews, Score, Followers. Vì mục tiêu của nhóm là dự đoán doanh thu khi game còn trong giai đoạn ý tưởng, các đặt trưng này chưa tồn tại ở thời điểm đó.

Sau khi đã lựa chọn các đặc trưng thì cào dữ liệu theo các thẻ HTML tương ứng.

Loại bỏ các ô trống bằng hàm dropna().

Loại bỏ các kí tự đặc biệt và chữ trong cột Price và Net Revenue.

Chuẩn hoá dữ liệu: dùng RobertScaller.

Xử lý ngoại lệ với phân bố lệch.

+ Biên trên = quantile(0.75) + 1.5 * iqr

+ Biên dưới = quantile(0.25) - 3 * iqr

4. Mô hình hóa dữ liệu

4.1 Mô hình Linear Regression:

a) Cơ sở lý thuyết:

"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…

Có hai loại: Simple regression và Multivariable regression

- simple regression:
  Y = B0 + B1*X
- Multivariable regression

- Y = B0 + B1*X1 + B2*X2+...+Bn*Xn

Y = Biến phụ thuộc

X = biến độc lập

B0 = Hằng số

B1 = Hệ số mối quan hệ giữa X và Y

⇨ Cần tìm B0, B1,…,Bn

Chia dữ liệu thành tập train và test:

- Bước đầu ta chia dữ liệu thành hai tập là input và target với target là đặc trưng Net Revenue cần dự đoán còn input là 4 đặc trưng phụ thuộc Pulisher, Price, Platform, Genres
- Sau đó ta dùng hàm train_test_split để tách input, target thành 4 tập train, test với tỉ lệ là 70% train, 30% test, random = 0

b) Chia dữ liệu và train mô hình

```
input = df[['Pulisher', 'Price', 'Platform', 'Genres']]
# vì tập dữ liệu nhỏ nên xử lý ngoại lệ trên cả tập train lẫn test
rs = RobustScaler()
input = XuLyNgoaiLe_PhanBoLech(input,"Price")
target = XuLyNgoaiLe_PhanBoLech(df, "Net Revenue")["Net Revenue"]
input_train,input_test,target_train,target_test = train_test_split(input, target, test_size=0.3, random_state=0)
input_train = pd.DataFrame(rs.fit_transform(input_train), columns=input_train.columns)
input_test = pd.DataFrame(rs.transform(input_test), columns=input_test.columns)
```

Hình 12:*Chia dữ liệu và train mô hình*

```
# Xây dựng mô hình hồi quy tuyến tính sử dụng thư viện scikit-learn
l_regr = linear_model.LinearRegression()
```

Hình 13*:Chia dữ liệu và train mô hình*

```
# huấn luyện mô hình
l_regr.fit(input_train, target_train)
print("[m1,m2,m3,m4] = ", l_regr.coef_)
print("m0 = ", l_regr.intercept_)
```

```
[m1,m2,m3,m4] =  [ 0.32601963  2.30330001  0.79490784 -1.21524662]
m0 =  4.908085492592432
```

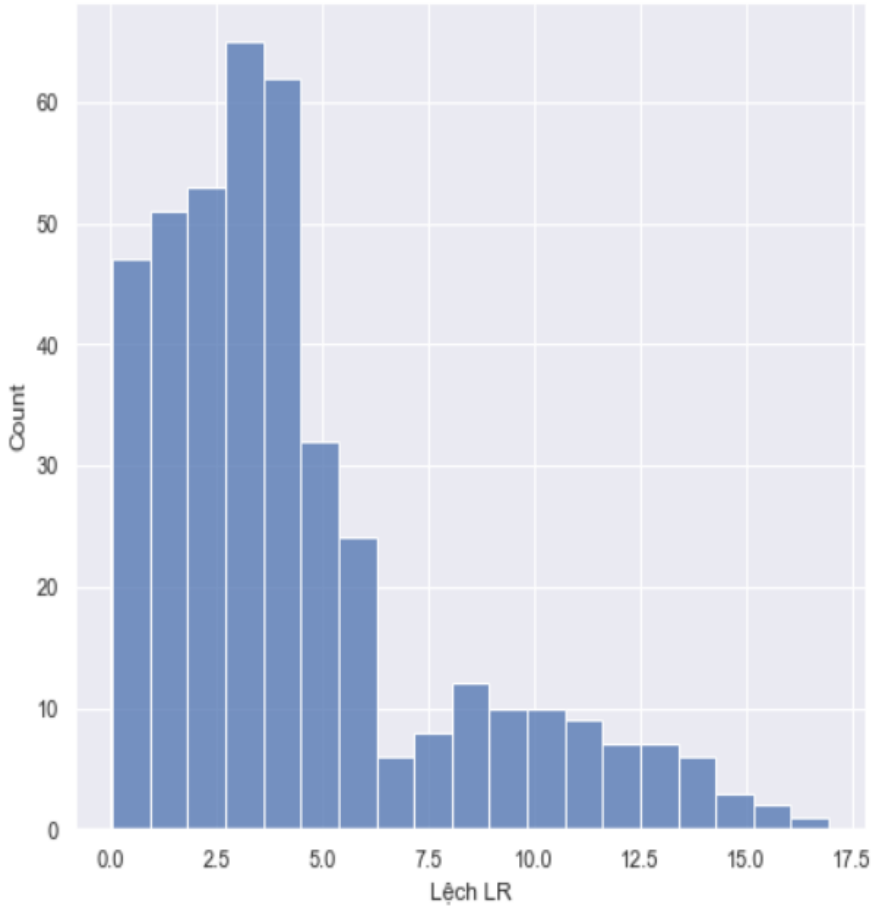
Hình 14:*Chia dữ liệu và train mô hình*

c) Độ lệch dự đoán so với thực tế

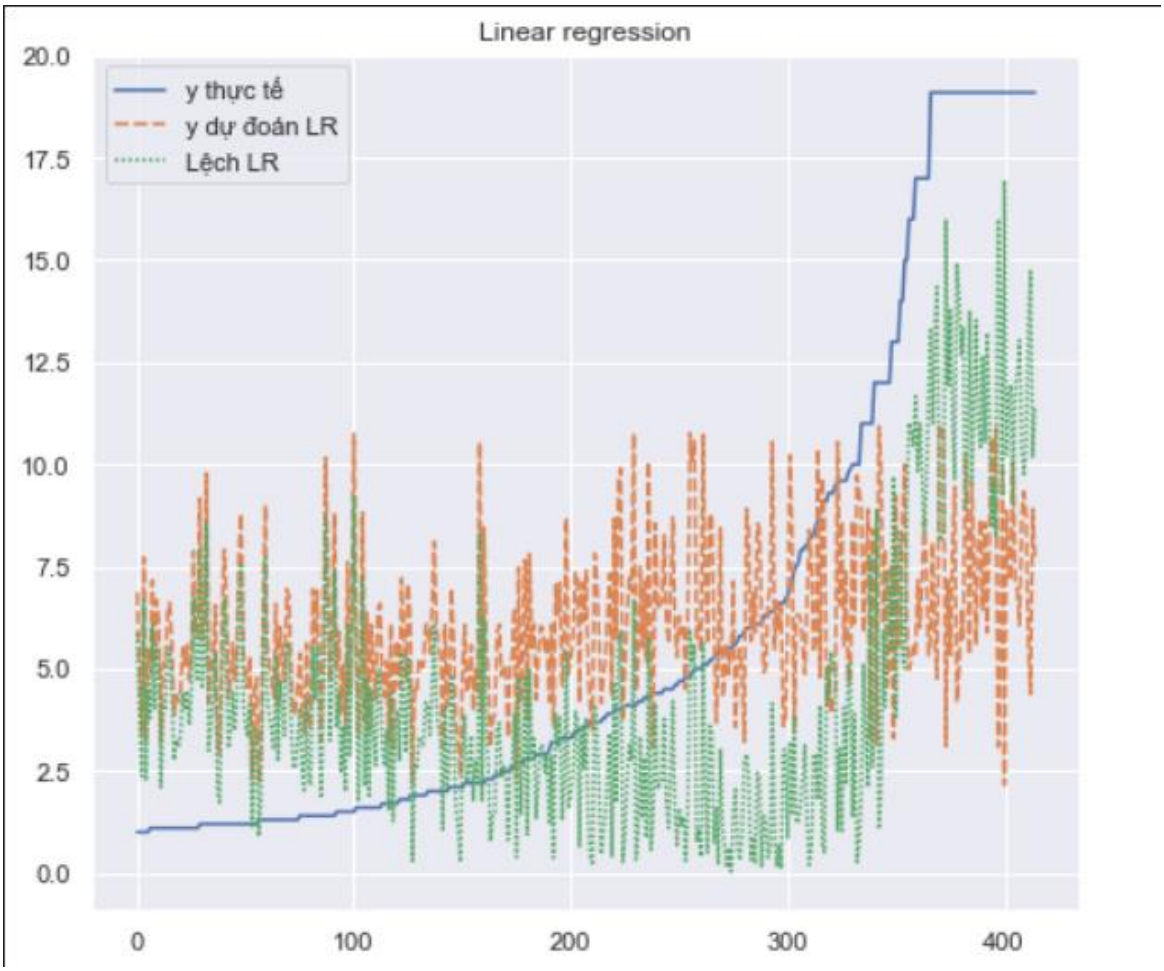
```
result2 = pd.DataFrame(data=np.array([target_test, target_test_pred,
abs(target_test - target_test_pred)]).T, columns=["y thực tế", "y dự đoán LR", "Lệch LR"])
print(result2.head(10))
```

	y thực tế	y dự đoán LR	Lệch LR
0	3.5	4.154337	0.654337
1	1.8	4.737512	2.937512
2	5.8	5.493835	0.306165
3	3.7	3.457367	0.242633
4	5.6	3.565295	2.034705
5	19.1	4.174826	14.925174
6	14.0	8.714472	5.285528
7	16.0	5.177229	10.822771
8	4.4	6.449122	2.049122
9	7.5	6.284536	1.215464

d) Một vài đồ thị thể hiện hiệu suất



***Hình 15**:Histogram thể hiện độ lệch của dự đoán so với thực tế(triệu đô)*



Hình 16:*Lineplot thể hiện tương quan của y thực tế, y dự đoán và độ lệch.*

e) Đánh giá theo hai metrics là RMSE và MAE

Sử dụng độ đo RMSE (căn bậc 2 của trung bình bình phương lỗi)

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

```
rmse_LR = math.sqrt(mean_squared_error(target_test, target_test_pred))
print(f'RMSE = {rmse_LR}')
```

```
RMSE = 5.614153027476399
```

Hình 17:*Đánh giá theo metrics RMSE*

Sử dụng MAE (độ lệch trung bình tuyệt đối)

Mean Absolute Error = (1/n) * ∑|yi – xi|

```
mae_LR = mean_absolute_error(target_test, target_test_pred)
print(f'MAE = {mae_LR}')
```

```
MAE = 4.3848351838120605
```

Hình 18:*Đánh giá theo metrics MAE*

4.2 Mô hình SVR

a) Cơ sở lý thuyết

Giả sử có tập dữ liệu huấn luyên {(x1,y1),...,(xn,yn)}, với x là đầu vào, y là kết quả đầu ra. Mục đích của hồi quy vector hỗ trợ SVR đó là tìm ra một hàm f(x) có sai số nhỏ nhất so với mục tiêu thực sự thu được đó là y.

Chia dữ liệu thành tập train và test:

- Bước đầu ta chia dữ liệu thành hai tập là x và y với y là đặc trưng cần dự đoán còn x là 4 đặc trưng phụ thuộc.
- Sau đó ta dùng hàm train_test_split để tách input, target thành 4 tập train, test với tỉ lệ là 70% train, 30% test, random = 0.

b) Chia dữ liệu và train mô hình

```
x = df[["Pulisher", "Price", "Platform", "Genres"]]
y = pd.DataFrame(df, columns=["Net Revenue"])
# vì tập dữ liệu nhỏ nên xử lý ngoại lệ trên cả tập train lẫn tập test
x = XuLyNgoaiLe_PhanBoLech(x, "Price")
y = XuLyNgoaiLe_PhanBoLech(y, "Net Revenue")["Net Revenue"]
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=0)
pipe = Pipeline([("transformer", RobustScaler()), ("estimator", SVR())])
pipe.fit(X_train, y_train)
```

Hình 19:*Chia dữ liệu và train mô hình*

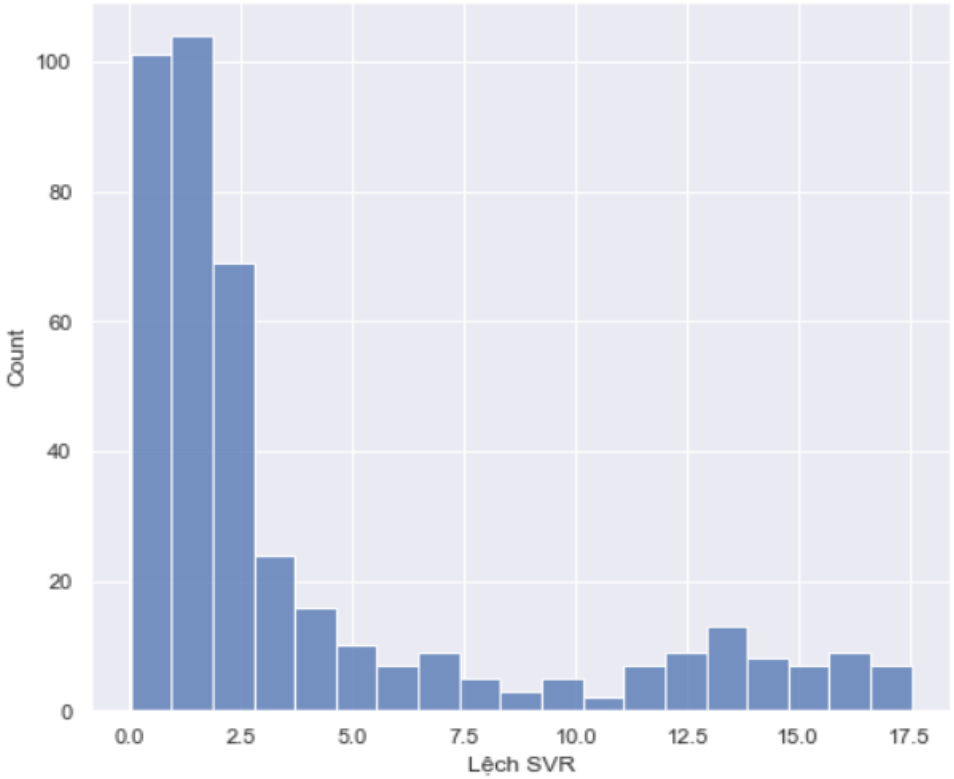
c) Độ lệch dự đoán so với thực tế

```
result1 = pd.DataFrame(data=np.array([y_test, y_test_pred, abs(y_test - y_test_pred)]).T
, columns=["y thực tế", "y dự đoán SVR", "Lệch SVR"])
result1.head(10)
```

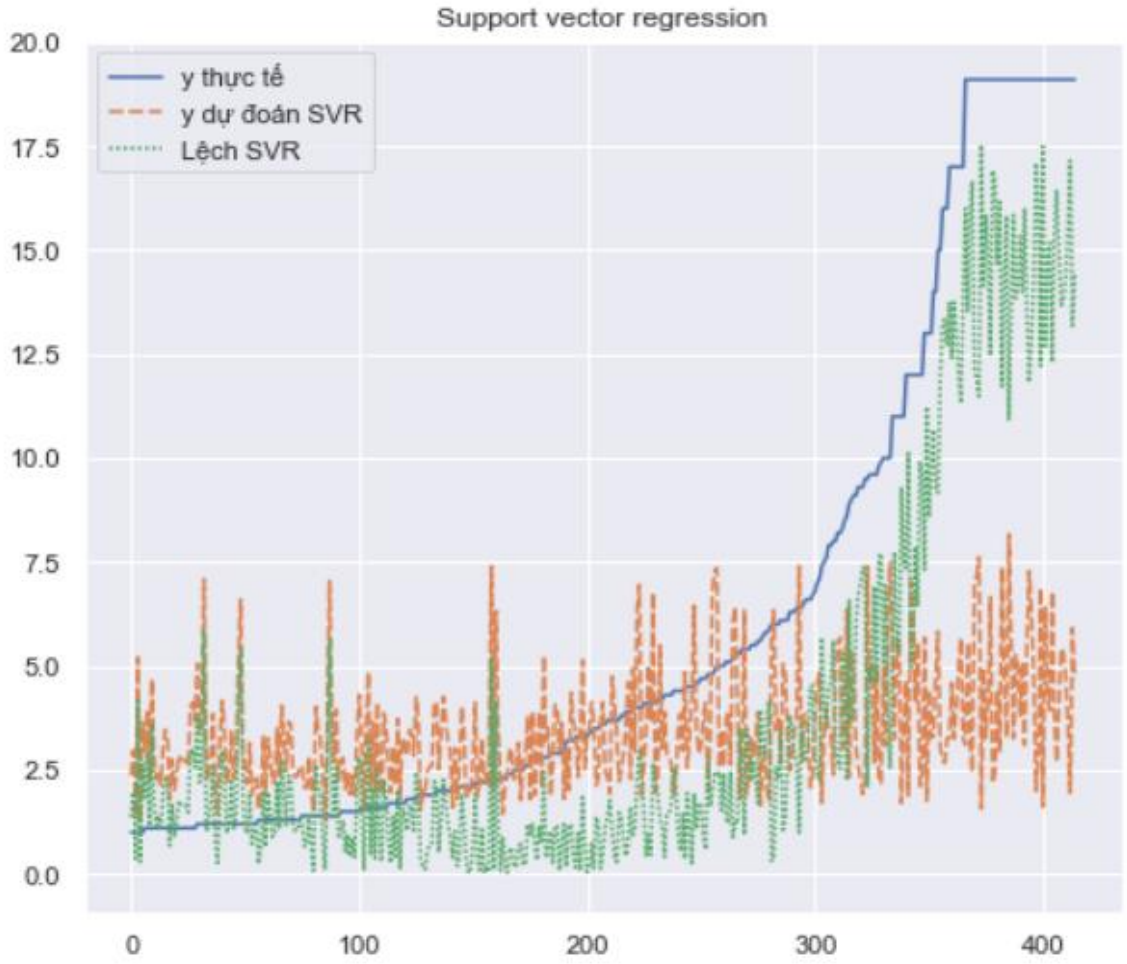
	y thực tế	y dự đoán SVR	Lệch SVR
0	3.5	2.126923	1.373077
1	1.8	2.845639	1.045639
2	5.8	2.985528	2.814472
3	3.7	1.930127	1.769873
4	5.6	1.643812	3.956188
5	19.1	2.180940	16.919060
6	14.0	4.786522	9.213478
7	16.0	2.671805	13.328195
8	4.4	3.497342	0.902658
9	7.5	3.344061	4.155939

Hình 20:*Độ lệch dự đoán so với thực tế*

d) Một vài đồ thị thể hiện hiệu suất



Hình 21: Histogram thể hiện độ lệch của dự đoán so với thực tế(triệu đô)



Hình 22:*Lineplot thể hiện tương quan của y thực tế, y dự đoán và độ lệch.*

e) Đánh giá theo hai metrics là RMSE và MAE

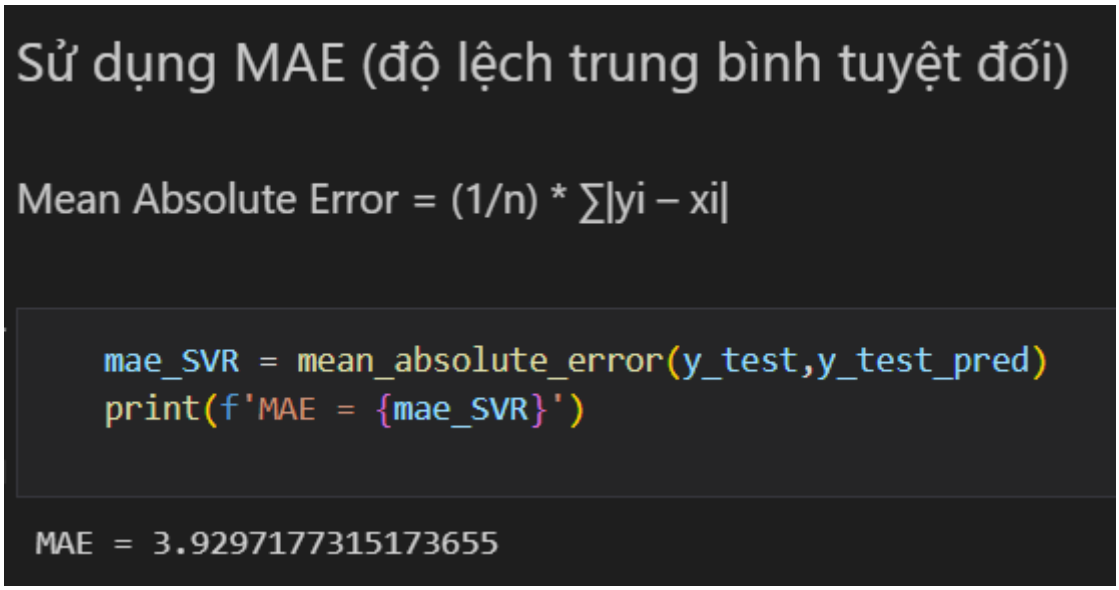
Sử dụng độ đo RMSE (căn bậc 2 của trung bình bình phương lỗi)

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

```
rmse_SVR = math.sqrt(mean_squared_error(y_test, y_test_pred))
print(f'RMSE = {rmse_SVR}')
```

```
RMSE = 6.137316679022097
```

Hình 23:*Đánh giá theo metrics RMSE*



Hình 24:*Đánh giá theo metrics MAE*

5. Kết luận

Những việc đã làm và kết quả:

- Đã cào được dữ liệu từ web.
- Trích xuất và chuẩn hoá dữ liệu.
- Xây dựng và kiểm thử hai mô hình.
- Đánh giá doanh thu dự đoán được từ hai mô hình Linear regression và SVR:
  - Mô hình SVR cho ra giá trị chính xác hơn so với Linear regression.
  - Cả hai đều dự đoán không tốt với ngoại lệ, cho ra giá trị sai lệch rất lớn.

Hướng phát triển:

- Nhóm sử dụng mục Net Revenue là doanh thu toàn thời gian, dẫn đến các tựa game ra đời sớm thường sẽ có doanh thu lớn hơn nhiều so với các game mới ra mắt. Để khắc phục thì nhóm phải cào thêm đặt trưng Release đó là ngày ra mắt, sau đó tính số năm đã ra mắt của game và tính trung bình doanh thu hằng năm của toàn bộ game đã cào được.
- Nếu cào thêm đặt trưng Score thì sẽ tăng mạnh độ chính xác của mô hình, vì các game điểm cao thì sẽ có chất lượng tốt và được mua nhiều hơn. Tuy nhiên nếu áp dụng đặt trưng này vào mô hình thì sau này x_predict cũng sẽ cần đặt trưng

Score, đồng nghĩa với việc game đã phải hoàn thành để có đánh giá điểm số (đi ngược lại với mục đích ban đầu của nhóm, đó là dự đoán khi game còn trong giai đoạn lên ý tưởng).

6. Tài liệu tham khảo

[1] Linear Regression - Hồi quy tuyến tính trong Machine Learning,

https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRlY3

[2] Học máy và Khai phá dữ liệu https://users.soict.hust.edu.vn/khoattq/ml-dm-course/?fbclid=IwAR0nfxCDeSWCBjA-FfR4hFUqxBv7oVuRkI3ZamhEg7ds4iejT3Y1KOQCVk0

[3] Các thông số và cách sử dụng của SVR, thư viện sklearn

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

[4] Hướng dẫn crawl thu thập dữ liệu từ trang web bằng python

https://www.youtube.com/watch?v=NUF_Av4mJgM&t=3475s