ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN – TIN

MÔN: NHẬP MÔN KHOA HỌC DỮ LIỆU

---------- ☙🕮❧ ----------

BÁO CÁO THỰC NGHIỆM

Đề tài: HOUSE PRICE PREDICTION

NHÓM 12:

22280034 – Trương Minh Hoàng
22280037 – Nguyễn Thị Xuân Hương
22280052 – Phan Thị Ngọc Linh
22280088 – Hồ Trần Anh Thư

THÀNH PHỐ HỒ CHÍ MINH, 2024

Mục lục

1. Introduction 3

2. Data Collection 3

3. Data Preprocessing 4

a) Handling outliers 4

b) Handling missing values 5

4. Method 7

5. Conlusion 7

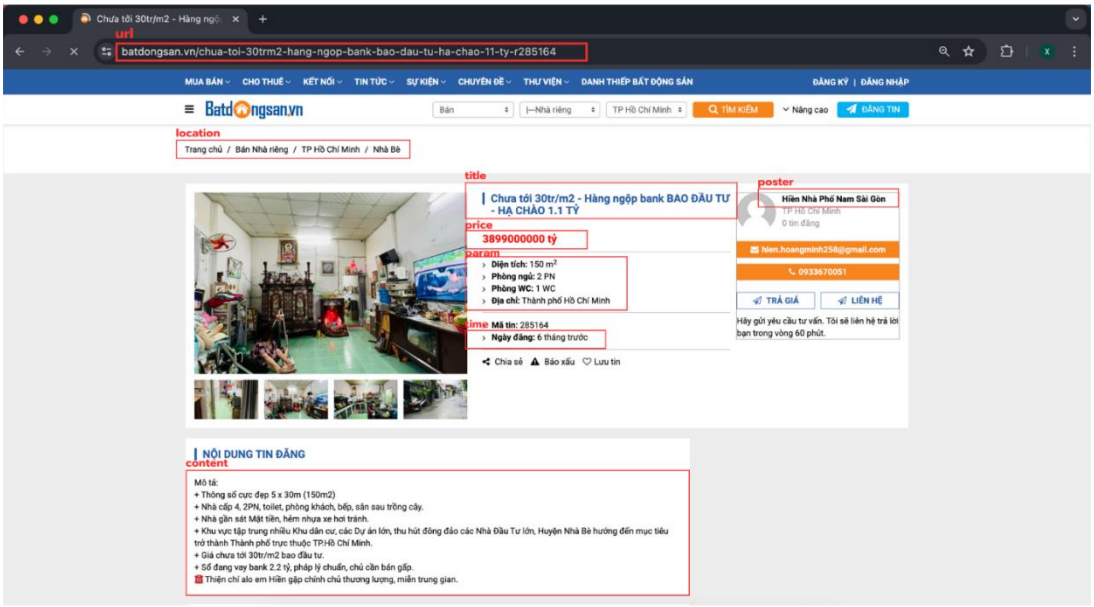
1. Introduction

Trong quá trình làm bài, tụi em nhận thấy mình sẽ có nhiều phương pháp xử lý trong cùng một vấn đề, từ đó có nhiều kinh nghiệm trong quá trình làm đồ án. Nên tụi em viết báo cáo thực nghiệm này nhằm so sánh các phương pháp và lý do lựa chọn phương pháp vận dụng vào bài lab chính.

Báo cáo này bổ sung cho báo cáo Jupyter NoteBook - ghi lại những phương pháp nhóm đã thử nghiệm trong quá trình thực hiện lab và đã bị loại bỏ khỏi Jupyter NoteBook.

2. Data Collection

✓ Ở phần cào dữ liệu, ban đầu chúng em đã chia ra 2 bạn cào và trích xuất dữ liệu bằng RegEx, 2 bạn trích xuất bằng LLM. Sau đó vì trích xuất bằng LLM cho ra kết quả không ổn định nên tụi em đã chọn dùng RegEx để cào.



1. Những mục cần cào trong file raw_data

✓ Chúng em đã cào 1 file raw data lấy kết tất cả các thông tin như trên *hình 1*. Sau đó viết hàm transform từ file raw data chứa các dữ liệu trên để lấy các features có ích.

- ✓ Sau đó đến bước xử lý dữ liệu chúng em đã nhận thấy các features *bedroom*, *wc* và *frontage*, chúng em thấy chúng có các sai sót mà RegEx khó có thể xử lý được. Do đó chúng em đã kết hợp LLM để trích xuất các features này. Các vấn đề có thể gặp ở các features này như:
    - Các biến *bedroom* và *wc* có lúc được người đăng liệt kê theo tầng, do đó ta cần cộng chúng vào với nhau. Có lúc người đăng nhắc tới tổng số phòng trước, sau đó lại tiếp tục liệt kê theo tầng.
        - ⇨ Nếu cộng tất cả số *bedroom* và *wc* lại với nhau có những trường hợp *bedroom* và *wc* trở nên lớn hơn bình thường (**tạo ra outlier**).
    - Biến *frontage* có những lúc người dùng chỉ bảo nhà **gần mặt tiền** nhưng căn nhà đó vẫn là nhà trong hẻm, nếu dùng RegEx sẽ không lấy được chính xác những biến này. Ban đầu tụi em chỉ dùng RegEx đã thu được correlation coefficient giữa target và *frontage* là **0.15**, sau khi kết hợp với LLM (check trên các biến có *frontage* là *True* và kiểm tra lại bằng LLM) cho biến này là **0.2**
- ⇨ Sự tương quan với target cải thiện khá tốt khi kết hợp **RegEx và LLM**.

3. Data Preprocessing

- ✓ Sau khi thu thập dữ liệu xong thì nhận thấy được trong dữ liệu còn nhiều missing value và các outlier.
- ✓ Missing value xuất hiện nhiều nhất tại các cột ‘area, bedroom, wc, floor’ như hình bên:

	column_name	total_missing	%_missing
0	area	414	4.14
1	bedroom	567	5.67
2	wc	2305	23.05
3	floor	1748	17.48
4	frontage	0	0.00
5	house_type	0	0.00
6	province	0	0.00
7	district	0	0.00
8	datetime	0	0.00
9	description	0	0.00
10	title	0	0.00
11	longitude	0	0.00
12	latitude	0	0.00
13	price	227	2.27

a) Handling outliers

- Trước tiên tụi em xử lý các giá trị bất thường của *price*. Sau khi tìm kiếm các nguồn tin thì tụi em xác định ngưỡng giá trị cho *price* là từ **0.1 tỷ (100**

triệu) đến 1000 tỷ. Nếu nằm ngoài ngưỡng này thì được xem là giá trị bất thường.

- Tiếp theo nhóm em đã thử 2 phương pháp xử lý outliers cho các biến numerical:
  - Phương pháp capping ouliers: với upper limit và lower limit được chọn là tại percentile 0.95 và 0.05 của dữ liệu, nhóm em đã sử dụng 2 cách là thay các điểm dữ liệu nằm ngoài khoảng đấy thành None và thay chúng bởi các percentile ấy. Cả 2 đều có kết quả không tốt bằng phương pháp IQR.

	area	bedroom	wc	floor	price
lower_limit	30.0	2.0	1.0	2.0	2.0
upper_limit	196.0	7.0	6.0	6.0	85.0

  - Phương pháp IQR: phương pháp này có loại bỏ tất cả các điểm được cho là outlier (được trực quan hóa trên biểu đồ boxplot). Loại bỏ khá tốt các điểm ngoại lai nhưng vẫn tồn tại khả năng mất một phần dữ liệu.

	Q1	median	Q3	IQR	lower_bound	upper_bound	outliers_num	outliers_percentage
area	42.0	60.0	85.0	43.0	-22.50	149.50	743.0	7.813650
bedroom	2.0	3.0	4.0	2.0	-1.00	7.00	352.0	3.701756
wc	2.0	3.0	4.0	2.0	-1.00	7.00	198.0	2.082238
floor	2.0	3.0	5.0	3.0	-2.50	9.50	20.0	0.210327
price	4.3	6.8	18.0	13.7	-16.25	38.55	1415.0	14.880639

b) Handling missing values

✓ Trước hết, đối với missing values của biến target `price`, nhóm đưa ra 2 lựa chọn:

- Một là điền missing values bằng học máy bán giám sát (semi - supervised learning).
- Hai là drop missing values.

✓ Mục đích của việc thực hiện học giám sát là để tận dụng dữ liệu có biến target NaN, tăng thêm dữ liệu cho tập train khi nguồn dữ liệu khan hiếm. Tuy nhiên, với quy mô dự án nhỏ, missing value của price là không đáng kể so với bộ dữ liệu 10k nhóm thu thập được. Do đó, nhóm lựa chọn drop các missing values của `price`.

✓ Với missing values của các biến còn lại, nhóm em thực hiện thử nghiệm trên 2 phương pháp điền missing values:

- **Phương pháp 1:** Đầu tiên nhóm sử dụng phương pháp phổ biến nhất điền các missing values của từng cột bằng mean của cột tương ứng.
- **Phương pháp 2:** Vì khi kiểm tra mối tương quan của các biến có missing values, nhóm nhận thấy chúng có quan hệ mật thiết với nhau. Từ đó, nhóm đưa ra phương pháp điền missing values bổ trợ cho nhau được trình bày như trong Jupyter NoteBook.

✓ Kết quả so sánh hai phương pháp:

Model	**Phương pháp 1**			**Phương pháp 2**		
	MSE	**MAE**	**R2**	**MSE**	**MAE**	**R2**
Linear	9.074	2.173	0.355	49.116	4.225	0.208
Ridge	9.075	2.173	0.355	49.116	4.225	0.208
Lasso	9.1	2.177	0.353	49.156	4.227	0.208
Decision Tree	13.093	2.279	0.07	76.854	4.858	-0.239
Random Forest	7.075	1.786	0.497	47.044	4.074	0.242
XGBoost	6.979	1.769	0.594	47.483	4.088	0.235
CatBoost	6.915	1.739	0.509	47.136	4.073	0.24

➢ Mặc dù, phương pháp 1 cho kết quả model cao hơn hẳn phương pháp 2, nhóm vẫn quyết định chọn phương pháp 2 vì:

- Missing values tổng các cột đạt đến hơn 3000 dữ liệu, chiếm gần một nửa dữ liệu gốc. Do đó, nếu điền tất cả các missing values bằng phương pháp 1 sẽ làm cho phần lớn dữ liệu bị giống nhau, dẫn đến việc model training và testing tốt trên tập dữ liệu hiện có và gặp khó khăn trong việc dự đoán một observation mới chưa gặp bao giờ.

4. Method

Nhóm em đã có thử nghiệm build Neural Network model bằng PyTorch, hiểu được cơ bản cách thức hoạt động. Thế nhưng chúng em cảm thấy như vậy là chưa đủ để build được một model Deep Learning tối ưu, hiệu suất của model chưa quá khác biệt so với các model Machine Learning. Vì vậy chúng em quyết định không đưa mô hình Neural Network của mình vào bài làm.

5. Conlusion

- ✓ Sau khi thử nghiệm xong thì nhóm đã nhận thấy được những điều sau
  - Phân phối của dữ liệu rất quan trọng trong quá trình xử lí dữ liệu.
  - Có thể kết hợp các feature rời rạc với nhau để tạo thành 1 feature có hữu ích hơn.
  - Data Preprocessing và Feature Engineering phụ thuộc rất nhiều vào đặc trưng của tập dữ liệu. Không có một phương pháp nào là tốt nhất.
  - Giữa các mô hình, không phải mô hình nào cho kết quả tốt trên train đều sẽ cho kết quả tốt trên test, việc đánh giá mô hình phải thông qua thực nghiệm, mô hình chỉ tốt khi nó tốt trên thực nghiệm.
  - Không phải lúc nào việc xử lý outlier cũng đạt được kết quả tối ưu. Nên cần cân nhắc và phân tích kỹ lưỡng trước khi quyết định loại bỏ outlier, vì một số outlier có thể chứa thông tin quan trọng về xu hướng và đặc trưng của dữ liệu.