Data_Preprocessing

August 4, 2023

1 Các nội dung chính

1. Mục tiêu:

- Nắm được các bước cơ bản trong khâu tiền xử lí dữ liệu.

2. Dữ liệu:

- Dữ liệu bất động sản - *Bengaluru_House_Data* > Gồm các trường dữ liệu: location, size, total_sqft, price, ...

  Link Kaggle: https://www.kaggle.com/amitabhajoy/bengaluru-house-price-data

3. Yêu cầu:

- Sử dụng các công cụ (Pandas, Seaborn, ...) để thực hiện xem xét, đánh giá đặc điểm của dữ liệu, từ đó đưa ra phương án tiền xử lí dữ liệu (làm sạch, trích xuất thông tin ban đầu, ...)

2 Nội dung thực hành

[ ]:
```
#Nếu chạy trên Google Colab thì cần kết nối với máy chủ trước
from google.colab import drive
drive.mount('/content/drive')
```

2.0.1 Chuẩn bị các thư viện cần thiết

[3]:
```
import numpy as np #Làm việc với các dữ liệu mảng nhiều chiều
import pandas as pd #Giúp làm việc với các dữ liệu dạng bảng
import matplotlib.pyplot as plt #Thư viện hỗ trợ trực quan hóa dữ liệu
import seaborn as sns #Thư viện giúp trực quan hóa dữ liệu, được xây trên␣
 ↪matplotlib
```

2.0.2 Load dữ liệu từ file đã tải về

1. Đọc dữ liệu bằng pandas, dạng dataframe

[ ]:
```
%cd /content/drive/MyDrive/ML_course/Preprocessing_practice/Practice/
 ↪Bangalore_House_Price_data
# Nếu chạy trên colab thì cũng cần trỏ tới thư mục phù hợp để lấy data
# cd DIR_PATH
```

[ ]:
```
path = "Bengaluru_House_Data.csv"
df_raw = pd.read_csv(path)
df_raw.shape
```

2. Review 5 sample đầu tiên

[ ]:
```
df_raw.head() # return DataFrame
```

3. Review 5 sample cuối cùng

[ ]:
```
df_raw.tail()
```

2.0.3 Exploratory Data Analysis (EDA)

[10]:
```
df = df_raw.copy() #Tạo bản sao để thực hiện EDA
```

1. Thông tin cơ bản về dữ liệu, tên trường, số giá trị non-null của từng trường, kiểu dữ liệu của từng trường

[ ]:
```
df.info()
```

2. Thống kê 1 số thuộc tính cơ bản của dữ liệu, bao gồm count, mean, std, min, max, quartile

[ ]:
```
df.describe()
```

3. Thống kê các giá trị duy nhất của từng trường và số lần xuất hiện của chúng

[ ]:
```
def value_count(df):
  for var in df.columns:
    print(df[var].value_counts())
    print("---------------------------------")

value_count(df)
```

4. Xem xét tương quan về giá trị của các cặp trường số

[ ]:
```
sns.pairplot(df)
```

[ ]:
```
num_vars = ["bath", "balcony", "price"]
sns.heatmap(df[num_vars].corr(),cmap="coolwarm", annot=True)
```

[ ]:
```
sns.heatmap(df.corr(),cmap="coolwarm", annot=False)
```

2.0.4 Chuẩn bị dữ liệu để huấn luyện mô hình

Xử lí các giá trị Null/ Nan

1. Thống kê tỉ lệ giá trị null của từng thuộc tính

[ ]:
```
df.isnull().mean()*100  # Tỷ lệ giá trị null của từng thuộc tính
```

2. Loại đi trường society vì tỉ lệ null cao (41%)

[ ]:
```
df2 = df.drop('society', axis='columns')
df2.shape
```

3. Thay thế giá trị null trong trường balcony bằng giá trị trung bình của các giá trị not null

[ ]:
```
df2['balcony'] = df2['balcony'].fillna(df2['balcony'].mean())
df2.isnull().sum()
```

4. Xóa đi các điểm dữ liệu (hàng) có giá trị nan (không có giá trị)

[ ]:
```
df3 = df2.dropna()
df3.shape
```

[ ]:
```
df3.isnull().sum() #Thống kê lại xem đã xử lí hết các dữ liệu null hay chưa?
```

Xử lí các trường thuộc tính

[22]:
```
#Cho phép in ra toàn bộ các giá trị output có thể của câu lệnh
pd.set_option("display.max_columns", None)
pd.set_option("display.max_rows", None)
```

1. Converting 'total_sqft' cat feature in numeric (float)

[ ]:
```
# Xem xét trường total_sqft
df3['total_sqft'].value_counts()
```

[24]:
```
#Xử lí các giá trị của trường total_sqft và lưu vào một biến tạm

total_sqft_float = [] #Biến tạm để lưu giá trị được xử lí
for str_val in df3['total_sqft']:
  try:
    total_sqft_float.append(float(str_val))
  except:
    try:
      temp = []
      temp = str_val.split('-')
      total_sqft_float.append((float(temp[0])+float(temp[-1]))/2)
    except:
      total_sqft_float.append(np.nan) #Gia tri ngoai le se duoc dat thanh null

df4 = df3.reset_index(drop=True)
```

[ ]:
```
# Thêm trường total_sqft_float:
```

```
df5 = df4.join(pd.DataFrame({'total_sqft_float':total_sqft_float}))
df5.head() #Quan sát kết quả sau khi xử lí
```

[ ]:
```
# Xem xét lại thông tin về các giá trị null
df5.isnull().sum()
```

[ ]:
```
# Loại bỏ các hàng có giá trị null
df6 = df5.dropna()
df6.shape
```

[ ]:
```
# Xem lại thông tin của dataframe
df6.info()
```

2. Làm việc với feature: *size*

[ ]:
```
# Quan sát sự phân bố giá trị của trường 'size' với value_counts
df6['size'].value_counts()
```

[30]:
```
#Chuyển thuộc tính số phòng từ dạng category về dạng numeric
size_int = []
for str_val in df6['size']:
  temp=[]
  temp = str_val.split(" ")
  try:
    size_int.append(int(temp[0]))
  except:
    size_int.append(np.nan)
    print("Noise = ",str_val)
```

[31]:
```
#Đánh lại index cho các hàng theo dãy số tự nhiên liên tiếp
df6 = df6.reset_index(drop=True)
```

[ ]:
```
# Thêm trường dữ liệu số phòng (bhk)
df7 = df6.join(pd.DataFrame({'bhk':size_int}))
df7.shape
```

[ ]:
```
#In ra kết quả thực hiện các thao tác kể trên?
df7.tail()
```

3. Phát hiện ngoại lệ (outlier) và loại bỏ

- Dựa trên biểu đồ boxplot/ hoặc công cụ khác để phát hiện và loại bỏ các điểm ngoại lai:

[ ]:
```
# Xem xét trường diện tích:
sns.boxplot(x = df7['total_sqft_float'])
```

[ ]:
```
# Chọn ngưỡng diện tích là 3500 để xem xét
df_temp7 = df7[df7['total_sqft_float'] < 3500]
```

```
sns.boxplot(x = df_temp7['total_sqft_float'])
```

[ ]:
```
# Loại bỏ đi các điểm dữ liệu có diện tích >= 2200 hoặc <= 300
# (Q1 - 1.5 * IQR or Q3 + 1.5 * IQR)
df8 = df7[(df7['total_sqft_float'] > 500) & (df7['total_sqft_float'] <2000)]
sns.boxplot(x = df8['total_sqft_float'])
df8.shape
```

[ ]:
```
# Tạo thêm trường dữ liệu price_per_sqft (giá/ diện tích feet vuông)
df8 = df8.reset_index(drop=True)
df8['price_per_sqft'] = df8['price']*100000 / df8['total_sqft_float']
df8.head()
```

[ ]:
```
df8.price_per_sqft.describe()
```

2.1 Bài tập bổ sung (tự làm)

Phần bài tập này là các câu hỏi mở rộng, làm tiếp theo bài toán ở trên. Học viên cần viết mã để thực hiện các yêu cầu dưới đây:

Bài tập 0: Sử dụng sns.boxplot() để quan sát đặc điểm phân bố dữ liệu của các trường số, mỗi trường này có outlier ko?

[ ]:
```
# Sử dụng boxplot để quan sát phân bố của dữ liệu, phát hiện ngoại lai (xử lí
 ↪nếu cần) của từng trường dữ liệu trong vars
# Gợi ý: sns.boxplot(data_field)

vars = ['price', 'total_sqft_float', 'price_per_sqft', 'balcony', 'bath', 'bhk']
plt.figure(figsize=(16,12))

#Code ở đây
```

Bài tập 1: Viết hàm bỏ đi các điểm dữ liệu có price per sqft dựa trên mean, std của các ngôi nhà dựa trên từng vị trí

Gợi ý: Xét trên từng vị trí (location), ngôi nhà thỏa mãn phải có $price\_per\_sqft \in [mean - std, mean + std]$

[ ]:
```
def remove_pps_outliers(df):
  #Code ở đây
#--------
df9 = remove_pps_outliers(df8)
df9.shape
```

Bài tập 2: Loại bỏ outlier xét theo trường bkh (số phòng)

Xét theo từng khu vực địa lí và theo từng loại nhà với số lượng phòng khác nhau, có một số ngôi nhà có giá không hợp lí (outliers), hãy tìm cách loại bỏ các outlier này. Cần ghi rõ quy tắc ghi nhận outlier

[ ]:
```
#location_df.groupby('bhk')
```

[ ]:
```
def remove_bhk_outliers(df):
  # Code ở đây

df10 = remove_bhk_outliers(df9)
df10.shape
```

Bài tập 3: Loại bỏ outlier khi xét trường 'bathroom'

[ ]:
```
df10.bath.unique() #Có thể quan sát thấy một số căn nhà có số phòng tắm quá lớn
↪(VD: 10!!!)
```

[ ]:
```
df10[df10.bath > df10.bhk+2]
```

[ ]:
```
df11 = #Code ở đây, sao cho: df10[df10.bath < df10.bhk+2]
df11.shape
```

[ ]:
```
df11.head()
```

[ ]:
```
# Quan sát lại kết quả sau khi xử lí với boxplot

# (Dùng lại hàm đã code bên trên)
```

Bài tập 4: Xem xét bỏ đi các trường không cần thiết

Gợi ý: bỏ đi ['area_type', 'availability',"location","size","total_sqft"]

[49]:
```
df12 = #Code ở đây
df12.head()
```

[49]:

	bath	balcony	price	total_sqft_float	bhk	price_per_sqft
0	3.0	2.0	150.0	1672.0	3	8971.291866
1	3.0	3.0	149.0	1750.0	3	8514.285714
2	3.0	2.0	150.0	1750.0	3	8571.428571
3	2.0	3.0	44.0	1250.0	3	3520.000000
5	2.0	2.0	83.0	1200.0	2	6916.666667

[ ]:
```
#Lưu kết quả xử lí cuối cùng:

df12.to_csv("clean_data.csv", index=False)
```

Bài tập 5*: Viết hàm trực quan hóa thể hiện mối tương quan giữa tổng diện tích (total_sqft) và giá nhà (price) theo từng vị trí địa lí (location) (tùy chọn minh họa theo 2 vị trí nào đó), của những căn nhà có 2 hoặc 3 phòng. Và cần phân biệt rõ điểm dữ liệu nào tương ứng với nhà có 2 phòng, điểm nào tương ứng với nhà có 3 phòng?

Gợi ý: Kết quả tương tự như hình dưới/ hoặc biểu đồ khác có ý nghĩa tương đương

[ ]:
```
#Gợi ý: Sử dụng plt.scatter() .... hoặc câu lệnh khác tương đương. Làm với df9

def plot_scatter_chart(df,location):
  #Viết code ở đây
```

[ ]:
```
plot_scatter_chart(df9, "Hebbal")
```

Bài tập 6*: Thực hiện các câu lệnh để trả lời các câu hỏi dưới đây:

- Thống kê giá nhà theo từng loại khu vực (area_type). Làm với df9:
- xem xét theo từng khu vực, thì giá nhà trung bình (price_per_sqft) là bao nhiêu, tương quan về giá nhà trung bình giữa các khu vực

Gợi ý: Phần này có thể đưa ra kết quả dạng bảng hoặc biểu đồ (cột, histogram, …). - Sử dụng các lệnh: df.groupby(), df.sortvalues(), … để trích xuất giá trị - Sử dụng matplotlib: plt.bar(), …

[ ]:
```
# Code ở đây
```