Phòng thí nghiệm cho Dự án Cuối kỳ - Phân tích Dữ liệu cho Bộ Dữ liệu Giá Nhà

1. Hướng dẫn

Trong bài tập này, bạn là Chuyên viên Phân tích Dữ liệu làm việc tại một Quỹ Đầu tư Bất động sản. Quỹ này muốn bắt đầu đầu tư vào bất động sản nhà ở. Bạn được giao nhiệm vụ xác định giá thị trường của một ngôi nhà dựa trên một tập hợp các đặc điểm. Bạn sẽ phân tích và dự đoán giá nhà bằng cách sử dụng các thuộc tính hoặc đặc điểm như diện tích, số phòng ngủ, số tầng, v.v. Đây là một sổ tay mẫu; nhiệm vụ của bạn là hoàn thành mười câu hỏi. Một số gợi ý cho các câu hỏi được đưa ra.

Trong khi hoàn thành sổ tay này, hãy chụp và lưu ảnh chụp màn hình kết quả cuối cùng của các giải pháp (ví dụ: biểu đồ, bảng, kết quả tính toán, v.v.). Chúng sẽ cần được chia sẻ trong phần Đánh giá ngang hàng sau đây của mô-đun Dự án Cuối kỳ.

2. Giới thiệu về Bộ dữ liệu

Bộ dữ liệu này chứa giá bán nhà tại Quận King, bao gồm cả Seattle. Bộ dữ liệu bao gồm các ngôi nhà được bán từ tháng 5 năm 2014 đến tháng 5 năm 2015. Bộ dữ liệu được lấy từ đây. Bộ dữ liệu cũng đã được sửa đổi một chút cho mục đích của khóa học này.

Variable	Description
id	A notation for a house
date	Date house was sold
price	Price is prediction target
bedrooms	Number of bedrooms
bathrooms	Number of bathrooms
sqft_living	Square footage of the home
sqft_lot	Square footage of the lot
floors	Total floors (levels) in house
waterfront	House which has a view to a waterfront
view	Has been viewed
condition	How good the condition is overall
grade	overall grade given to the housing unit, based on King County grading system
sqft_above	Square footage of house apart from basement
sqft_basement	Square footage of the basement
yr_built	Built Year
yr_renovated	Year when house was renovated
zipcode	Zip code
lat	Latitude coordinate
long	Longitude coordinate
sqft_living15	Living room area in 2015(implies-- some renovations) This might or might not have affected the lotsize area
sqft_lot15	LotSize area in 2015(implies-- some renovations)

3. Nhập các thư viện cần thiết

```
# All Libraries required for this lab are listed below. The libraries pre-installed on Skills Network Labs are commented.
# !mamba install -qy pandas==1.3.4 numpy==1.21.4 seaborn==0.9.0 matplotlib==3.5.0 scikit-learn==0.20.1
# Note: If your environment doesn't support "!mamba install", use "!pip install"
```

```
# Surpress warnings:
def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
```

```
#!pip install -U scikit-learn
```

```
import piplite
await piplite.install('seaborn')

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
%matplotlib inline
```

4. Mô-đun 1: Nhập Bộ Dữ liệu

Tải xuống tập dữ liệu bằng cách chạy ô bên dưới.

```
from pyodide.http import pyfetch

async def download(url, filename):
    response = await pyfetch(url)
```

```
    if response.status == 200:
        with open(filename, "wb") as f:
            f.write(await response.bytes())
```

```
filepath='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DA0101EN-SkillsNetwork/labs/FinalModule_Coursera/data/kc_house_data_NaN.csv'
```

```
await download(filepath, "housing.csv")
file_name = "housing.csv"
```

Tải tệp csv:

```
df = pd.read_csv(file_name)
```

Lưu ý: Phiên bản thực hành này sử dụng JupyterLite, yêu cầu phải tải tập dữ liệu xuống giao diện. Trong khi thực hành trên phiên bản đã tải xuống của sổ ghi chép này trên máy cục bộ (Jupyter Anaconda), người học có thể **bỏ qua các bước trên** và sử dụng trực tiếp URL trong hàm pandas.read_csv(). Bạn có thể bỏ chú thích và chạy các câu lệnh trong ô bên dưới.

```
#filepath='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DA0101EN-SkillsNetwork/labs/FinalModule_Coursera/data/kc_house_data_NaN.csv'
#df = pd.read_csv(filepath, header=None)
```

Chúng ta sử dụng phương thức head để hiển thị 5 cột đầu tiên của khung dữ liệu.

```
df.head()
```

	Unnamed: 0	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated
0	0	7129300520	20141013T000000	221900.0	3.0	1.00	1180	5650	1.0	0	...	7	1180	0	1955	0
1	1	6414100192	20141209T000000	538000.0	3.0	2.25	2570	7242	2.0	0	...	7	2170	400	1951	1991
2	2	5631500400	20150225T000000	180000.0	2.0	1.00	770	10000	1.0	0	...	6	770	0	1933	0
3	3	2487200875	20141209T000000	604000.0	4.0	3.00	1960	5000	1.0	0	...	7	1050	910	1965	0
4	4	1954400510	20150218T000000	510000.0	3.0	2.00	1680	8080	1.0	0	...	8	1680	0	1987	0

5 rows × 22 columns

Câu hỏi 1

Hiển thị kiểu dữ liệu của từng cột bằng hàm dtypes. Chụp ảnh màn hình mã của bạn và xuất ra. Bạn sẽ cần nộp ảnh chụp màn hình cho dự án cuối cùng.

```
1  #Enter Your Code, Execute and take the Screenshot
2  df.dtypes
```

```
Unnamed: 0         int64
id                 int64
date              object
price            float64
bedrooms         float64
bathrooms        float64
sqft_living        int64
sqft_lot           int64
floors           float64
waterfront         int64
view               int64
condition          int64
grade              int64
sqft_above         int64
sqft_basement      int64
yr_built           int64
yr_renovated       int64
zipcode            int64
lat              float64
long             float64
sqft_living15      int64
sqft_lot15         int64
dtype: object
```

Chúng ta sử dụng phương thức describe() để có được bản tóm tắt thống kê của khung dữ liệu.

```
1  df.describe()
```

	Unnamed: 0	id	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above
count	21613.00000	2.161300e+04	2.161300e+04	21600.000000	21603.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	...	21613.000000	21613.000000
mean	10806.00000	4.580302e+09	5.400881e+05	3.372870	2.115736	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	...	7.656873	1788.390691
std	6239.28002	2.876566e+09	3.671272e+05	0.926657	0.768996	918.440897	4.142051e+04	0.539989	0.086517	0.766318	...	1.175459	828.090978
min	0.00000	1.000102e+06	7.500000e+04	1.000000	0.500000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	...	1.000000	290.000000
25%	5403.00000	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	...	7.000000	1190.000000
50%	10806.00000	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	...	7.000000	1560.000000
75%	16209.00000	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	...	8.000000	2210.000000
max	21612.00000	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	...	13.000000	9410.000000

8 rows × 21 columns

5. Mô-đun 2: Xử lý Dữ liệu

Câu hỏi 2

Xóa các cột **"id"** và **"Unnamed: 0"** khỏi trục 1 bằng phương thức `drop()`, sau đó sử dụng phương thức `describe()` để lấy tóm tắt thống kê dữ liệu. Đảm bảo tham số **inplace** được đặt thành **True**. Chụp ảnh màn hình mã của bạn và xuất ra. Bạn sẽ cần nộp ảnh chụp màn hình cho dự án cuối cùng.

```
1  #Enter Your Code, Execute and take the Screenshot
2  df.drop(['id','Unnamed: 0'], axis=1, inplace=True)
3  df.describe()
```

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	2
count	2.161300e+04	21600.000000	21603.000000	21613.000000	2.161300e+04	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	21613.000000	
mean	5.400881e+05	3.372870	2.115736	2079.899736	1.510697e+04	1.494309	0.007542	0.234303	3.409430	7.656873	1788.390691	291.509045	
std	3.671272e+05	0.926657	0.768996	918.440897	4.142051e+04	0.539989	0.086517	0.766318	0.650743	1.175459	828.090978	442.575043	
min	7.500000e+04	1.000000	0.500000	290.000000	5.200000e+02	1.000000	0.000000	0.000000	1.000000	1.000000	290.000000	0.000000	
25%	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03	1.000000	0.000000	0.000000	3.000000	7.000000	1190.000000	0.000000	
50%	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03	1.500000	0.000000	0.000000	3.000000	7.000000	1560.000000	0.000000	
75%	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04	2.000000	0.000000	0.000000	4.000000	8.000000	2210.000000	560.000000	
max	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06	3.500000	1.000000	4.000000	5.000000	13.000000	9410.000000	4820.000000	

Chúng ta có thể thấy chúng ta thiếu các giá trị cho các cột phòng ngủ và phòng tắm

```
1  print("number of NaN values for the column bedrooms :",
   df['bedrooms'].isnull().sum())
2  print("number of NaN values for the column bathrooms :",
   df['bathrooms'].isnull().sum())
```

```
number of NaN values for the column bedrooms : 13
number of NaN values for the column bathrooms : 10
```

Chúng ta có thể thay thế các giá trị bị thiếu của cột **'bedrooms'** bằng giá trị trung bình của cột **'bedrooms'** bằng phương thức `replace()`. Đừng quên đặt tham số **inplace** thành **True**.

```
1  mean=df['bedrooms'].mean()
2  df['bedrooms'].replace(np.nan,mean, inplace=True)
```

Chúng ta cũng thay thế các giá trị bị thiếu của cột 'bathrooms' bằng giá trị trung bình của cột ' bathrooms ' bằng phương thức replace(). Đừng quên đặt tham số **inplace** bằng **True.**

```
1  mean=df['bathrooms'].mean()
2  df['bathrooms'].replace(np.nan,mean, inplace=True)
```

```
1  print("number of NaN values for the column bedrooms :",
   df['bedrooms'].isnull().sum())
2  print("number of NaN values for the column bathrooms :",
   df['bathrooms'].isnull().sum())
```

```
number of NaN values for the column bedrooms : 0
number of NaN values for the column bathrooms : 0
```

6. Mô-đun 3: Phân tích Dữ liệu Khám phá

Câu hỏi 3

Sử dụng phương thức value_counts() để đếm số lượng ngôi nhà với các giá trị tầng (floor) khác nhau. Dùng tiếp phương thức .to_frame() để chuyển kết quả đó thành một DataFrame. Hãy chụp ảnh màn hình đoạn mã của bạn cùng với kết quả đầu ra. Bạn sẽ cần nộp ảnh chụp màn hình đó cho dự án cuối khóa.

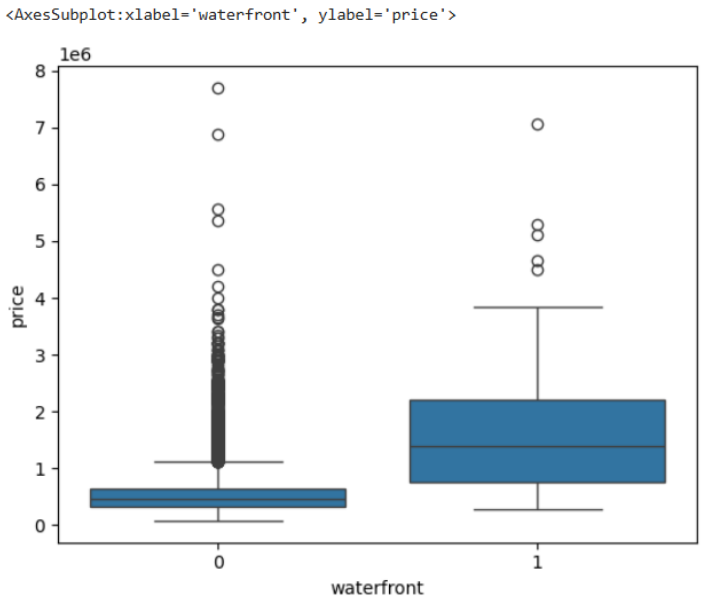
```
1  #Enter Your Code, Execute and take the Screenshot
2  df['floors'].value_counts().to_frame()
```

floors	count
1.0	10680
2.0	8241
1.5	1910
3.0	613
2.5	161
3.5	8

Câu hỏi 4

Sử dụng hàm `boxplot()` trong thư viện **seaborn** để xác định xem nhà có tầm nhìn ra bờ sông hay không có tầm nhìn ra bờ sông có nhiều giá trị ngoại lệ hơn. Chụp ảnh màn hình mã nguồn và boxplot của bạn. Bạn sẽ cần nộp ảnh chụp màn hình cho dự án cuối kỳ.

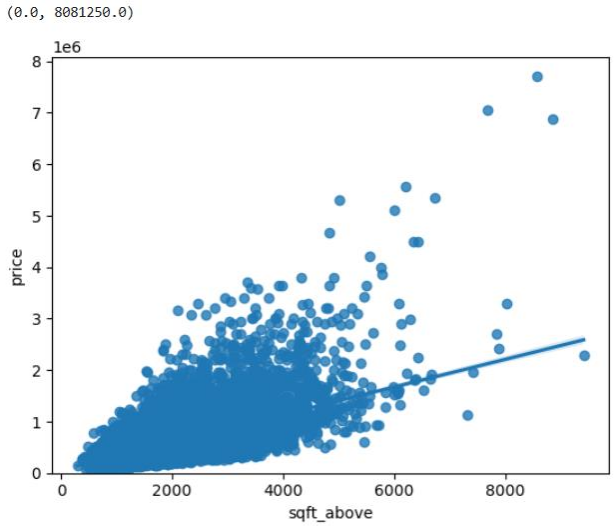
```
1  sns.boxplot(x='waterfront', y='price', data=df)
```



Câu hỏi 5

Sử dụng hàm `regplot()` trong thư viện **seaborn** để xác định xem đặc trưng sqft_above có tương quan nghịch hay thuận với giá. Chụp ảnh màn hình mã nguồn và biểu đồ phân tán của bạn. Bạn sẽ cần nộp ảnh chụp màn hình cho dự án cuối kỳ.

```
1  #Enter Your Code, Execute and take the Screenshot
2  sns.regplot(x='sqft_above', y='price', data=df)
3  plt.ylim(0,)
```



Chúng ta có thể sử dụng phương thức `corr()` của Pandas để tìm ra đặc trưng (feature) – ngoài giá (price) – có mối tương quan cao nhất với giá.

```
1  df_numeric = df.select_dtypes(include=[np.number])
2  df_numeric.corr()['price'].sort_values()
```

```
zipcode         -0.053203
long             0.021626
condition        0.036362
yr_built         0.054012
sqft_lot15       0.082447
sqft_lot         0.089661
yr_renovated     0.126434
floors           0.256794
waterfront       0.266369
lat              0.307003
bedrooms         0.308797
sqft_basement    0.323816
view             0.397293
bathrooms        0.525738
sqft_living15    0.585379
sqft_above       0.605567
grade            0.667434
sqft_living      0.702035
price            1.000000
Name: price, dtype: float64
```

Giải thích:

- **Dòng 1**: `df.select_dtypes(include=[np.number])`. Lọc ra **các cột kiểu số** trong DataFrame df, tức là chỉ giữ lại những cột có kiểu dữ liệu như int, float… Mục đích là để chuẩn bị cho việc tính toán hệ số tương quan (corr()) – vì chỉ tính được giữa các cột số thôi.
- **Dòng 2**: `df_numeric.corr()['price'].sort_values()`. Cách ghi như vậy là: `df_numeric.corr()`: Tính **ma trận tương quan** giữa tất cả các cột số (các hệ số Pearson). `['price']` → Chọn ra **cột tương quan với 'price'**, tức là mức độ tương quan giữa price với từng cột khác. `.sort_values()` → Sắp xếp các giá trị tương quan theo thứ tự tăng dần.

Trong **Pandas**, bạn **hoàn toàn có thể gọi phương thức xong rồi truy cập trực tiếp vào cột**, nếu kết quả của phương thức là một DataFrame. Đây là một **pattern rất phổ biến**!

Tổng quát cú pháp: `df.phuong_thuc()['ten_cot']`

Một số ví dụ tương tự:

1. **corr() + ['cột']**
2. **describe() + ['cột']**
3. **groupby() + ['cột']**
4. **pivot_table() + ['cột']**

Có thể kết hợp thêm .sort_values(), .plot(), .reset_index()... sau nữa, ví dụ: df.corr()['price'].sort_values().plot(kind='barh')

7. Mô-đun 4: Phát triển mô hình

Chúng ta có thể huấn luyện một mô hình hồi quy tuyến tính sử dụng đặc trưng kinh độ **'long'** và tính giá trị **R^2**.

```
1  X = df[['long']]
2  Y = df['price']
3  lm = LinearRegression()
4  lm.fit(X,Y)
5  lm.score(X, Y)
```

```
0.00046769430149007363
```

Câu hỏi 6

Sử dụng mô hình hồi quy tuyến tính để dự đoán **'price'** bằng cách sử dụng thuộc tính **'sqft_living'**, sau đó tính **R^2**. Chụp ảnh màn hình mã của bạn và giá trị **R^2**. Bạn sẽ cần nộp mã này cho bài tập cuối kỳ.

```
1  #Enter Your Code, Execute and take the Screenshot
2  E = df[['sqft_living']]
3  lm.fit(E, Y)
4  lm.score(E, Y)
```

```
0.4928532179037931
```

Câu hỏi 7

Sử dụng mô hình hồi quy tuyến tính để dự đoán **'price'** bằng cách sử dụng danh sách các đặc điểm sau:

```
1  features = ["floors", "waterfront", "lat", "bedrooms", "sqft_basement",
   "view", "bathrooms", "sqft_living15", "sqft_above", "grade",
   "sqft_living"]
```

Sau đó, hãy tính $\mathbf{R^2}$. Chụp ảnh màn hình mã của bạn và giá trị $\mathbf{R^2}$. Bạn sẽ cần nộp mã này cho dự án cuối kỳ.

```
1  #Enter Your Code, Execute and take the Screenshot
2  features = df[["floors", "waterfront", "lat", "bedrooms", "sqft_basement",
   "view", "bathrooms", "sqft_living15", "sqft_above", "grade",
   "sqft_living"]]
3  lm.fit(features, Y)
4  lm.score(features, Y)
```

```
0.6576890354915759
```

Điều này sẽ giúp ích cho Câu hỏi 8

Tạo một danh sách các bộ, phần tử đầu tiên trong bộ chứa tên của ước lượng:

- 'scale'
- 'polynomial'
- 'model'

Phần tử thứ hai trong bộ chứa hàm tạo mô hình

- StandardScaler()
- PolynomialFeatures(include_bias=False)
- LinearRegression()

```
1  Input=[('scale',StandardScaler()),('polynomial',
   PolynomialFeatures(include_bias=False)),('model',LinearRegression())]
```

Câu hỏi 8

Sử dụng danh sách để tạo một đối tượng đường ống nhằm dự đoán **'price'**, điều chỉnh đối tượng bằng các đặc trưng trong danh sách và tính toán $\mathbf{R^2}$. Chụp ảnh màn hình mã của bạn và giá trị $\mathbf{R^2}$. Bạn sẽ cần nộp mã này cho dự án cuối kỳ.

```
#Enter Your Code, Execute and take the Screenshot
pipe = Pipeline(Input)
features = features.astype('float')
pipe.fit(features, Y)
yhat = pipe.predict(features)
r2_score(Y, yhat)
```

```
0.7512051345272872
```

8. Mô-đun 5: Đánh giá và Tinh chỉnh Mô hình

Nhập các mô-đun cần thiết:

```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
print("done")
```

Chúng tôi sẽ chia dữ liệu thành tập huấn luyện và tập kiểm tra:

```
features = ["floors", "waterfront", "lat", "bedrooms", "sqft_basement", "view", "bathrooms", "sqft_living15", "sqft_above", "grade", "sqft_living"]
X = df[features]
Y = df['price']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.15, random_state=1)

print("number of test samples:", x_test.shape[0])
print("number of training samples:",x_train.shape[0])
```

```
number of test samples: 3242
number of training samples: 18371
```

Câu hỏi 9

Tạo và huấn luyện một đối tượng **hồi quy Ridge** bằng dữ liệu huấn luyện, đặt **tham số chính quy hóa thành 0,1** và tính toán $\mathbf{R^2}$ bằng dữ liệu kiểm tra. Chụp ảnh màn hình mã của bạn và giá trị R^2. Bạn sẽ cần nộp mã này cho dự án cuối kỳ.

```
1  from sklearn.linear_model import Ridge
```

```
1  #Enter Your Code, Execute and take the Screenshot
2  RidgeModel = Ridge(alpha=0.1)
3  RidgeModel.fit(x_train, y_train)
4  yhat1 = RidgeModel.predict(x_test)
5  RidgeModel.score(x_test, y_test)
```

```
0.647875916393907
```

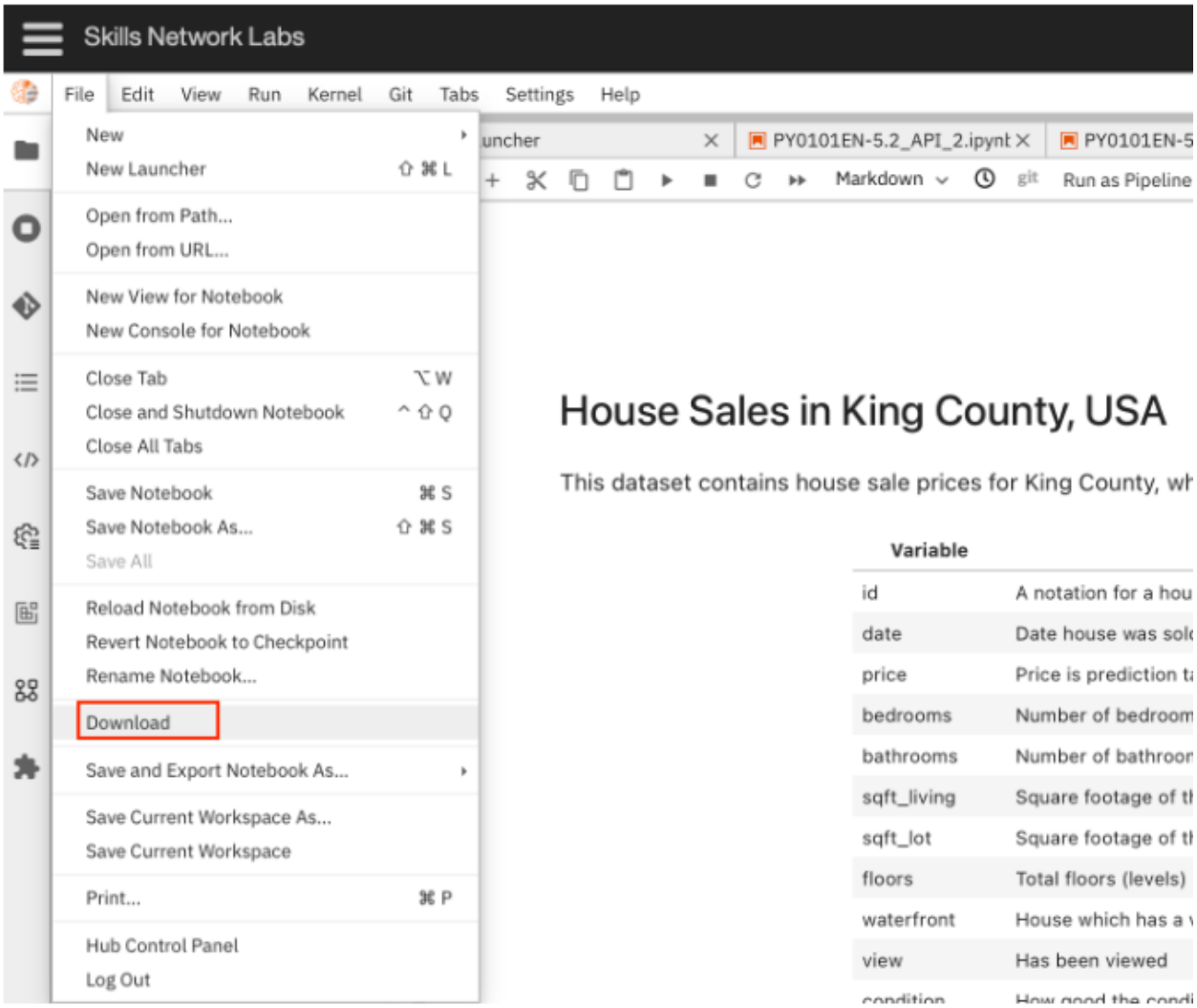
Câu hỏi 10

Thực hiện phép biến đổi đa thức **bậc hai** trên cả dữ liệu huấn luyện và dữ liệu kiểm tra. Tạo và điều chỉnh một đối tượng hồi quy Ridge bằng dữ liệu huấn luyện, đặt tham số chính quy hóa thành 0,1 và tính toán $\mathbf{R^2}$ bằng dữ liệu kiểm tra được cung cấp. Chụp ảnh màn hình mã của bạn và $\mathbf{R^2}$. Bạn sẽ cần nộp ảnh này cho dự án cuối kỳ.

```
1  #Enter Your Code, Execute and take the Screenshot
2  Pr = PolynomialFeatures(degree=2)
3  x_train_pr = Pr.fit_transform(x_train)
4  x_test_pr = Pr.fit_transform(x_test)
5  RidgeModel.fit(x_train_pr, y_train)
6  yhat_pr = RidgeModel.predict(x_test_pr)
7  r2_score(y_test, yhat_pr)
```

```
0.7002744263583341
```

Sau khi hoàn thành sổ ghi chép, bạn sẽ phải chia sẻ nó. Bạn có thể tải sổ ghi chép xuống bằng cách vào mục "File" và nhấp vào nút "Download".



Thao tác này sẽ lưu tệp (.ipynb) vào máy tính của bạn. Sau khi lưu, bạn có thể tải tệp này lên tab "My Submission" trong mục "Peer-graded Assignment".