Python pricing engine script

1. Price Prediction Model of AIRBNB Dataset

Import the Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

%matplotlib inline

import seaborn as sns

import warnings

warnings.filterwarnings('ignore')

import scipy as spy

Import the dataset

airbnb=pd.read\_csv('D:\intership\Airbnb\_Data.csv')

Shape of the dataset

airbnb.shape

1. Descriptive analysis of the dataset

all\_columns=airbnb.columns

print('All columns of the Airbnb dataset')

for column in all\_columns:

print(column)

Checking the variables types

These are divided into two types

1. Numeric data -Continuous data,Discrete data
2. Categorical data-ordinal data,Nominal data

Categorical data

print('This is categorical data ')

print('------------------------------')

uni\_nhood=airbnb['neighbourhood'].unique()

uni\_room\_type=airbnb['room\_type'].unique()

uni\_instant\_bookable=airbnb['instant\_bookable'].unique()

print('1. Neighbourhood :')

for neighbourhood in uni\_nhood:

print(neighbourhood)

print('-----------------------------------------')

print('2. Room\_types : ')

for room\_type in uni\_room\_type:

print(room\_type)

print('-------------------------------------------')

print('3. Instant\_bookable:')

for instant\_bookable in uni\_instant\_bookable:

print(instant\_bookable)

Numeric data

numercial\_cols=airbnb.select\_dtypes(include='number').columns

discrete\_numercial\_cols=[ "log\_price","accommodates", "bedrooms","bathrooms"

, "number\_of\_reviews","latitude" , "longitude"]

continuous\_numercial\_cols=[col for col in numercial\_cols if col not in discrete\_numercial\_cols]

print('This is the Numercial Data columns')

print('-------------------------------------------')

print('Discrete Numercial Data:')

for column in discrete\_numercial\_cols:

uni\_values=airbnb[column].unique()

uni\_values\_str=','.join(map(str,uni\_values))

print(f"{column}={uni\_values\_str}")

print('-----------------------------------------------------------------')

print('Continuous Numercial Data :')

for column in discrete\_numercial\_cols:

if col=='log\_price':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

elif col=='accommodates':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

elif col=='bedrooms':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

elif col=='bathrooms':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

elif col=='number\_of\_reviews':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

elif col=='longitude':

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

else:

min\_value=airbnb[column].min()

max\_value=airbnb[column].max()

print(f"{column}:Min={min\_value},Max={max\_value}")

Frequency Distribution

freq\_area=airbnb['neighbourhood'].value\_counts().reset\_index()

freq\_area.columns=['neighbourhood','Frequency']

freq\_area['Percent']=(freq\_area['Frequency']/len(airbnb))\*100

freq\_area=freq\_area.sort\_values(by='Frequency')

freq\_area=freq\_area.reset\_index(drop=True)

print(freq\_area)

room\_type=airbnb['room\_type'].value\_counts().reset\_index()

room\_type.columns=['room\_type','Frequency']

room\_type['Precent']=(room\_type['Frequency']/len(airbnb))\*100

room\_type=room\_type.sort\_values(by='Frequency')

room\_type=room\_type.reset\_index(drop=True)

print(room\_type)

Conclusion form the Frequency Distribution

airbnb.head()

airbnb.describe()

airbnb.info()

Checking for null values

airbnb.isnull().sum()

def nulls\_summary\_table(airbnb):

null\_values=pd.DataFrame(airbnb.isnull().sum())

null\_values[1]=null\_values[0]/len(airbnb)

null\_values.columns=['null\_count','null\_percent']

return null\_values

nulls\_summary\_table(airbnb)

Drop the unnecessary columns

airbnb\_copy=airbnb

airbnb\_copy.drop(['id','last\_review'],axis=1,inplace=True)

Check the changes

airbnb\_copy.head()

airbnb\_copy.fillna({'number\_of\_reviews':0},inplace=True)

airbnb\_copy.isnull().sum().sum()

Remove the NAN values from the dataset

airbnb\_copy.isnull().sum()

airbnb\_copy.dropna(how='any',inplace=True)

Lets finally check the Results

airbnb.info()

Removing where the price equal to 0

airbnb\_copy=airbnb\_copy[airbnb\_copy['log\_price']!=0]

airbnb\_copy.head()

zero\_price\_count=airbnb\_copy[airbnb\_copy['log\_price']==0].count()['log\_price']

print(zero\_price\_count)

* 1. Data Visualizations

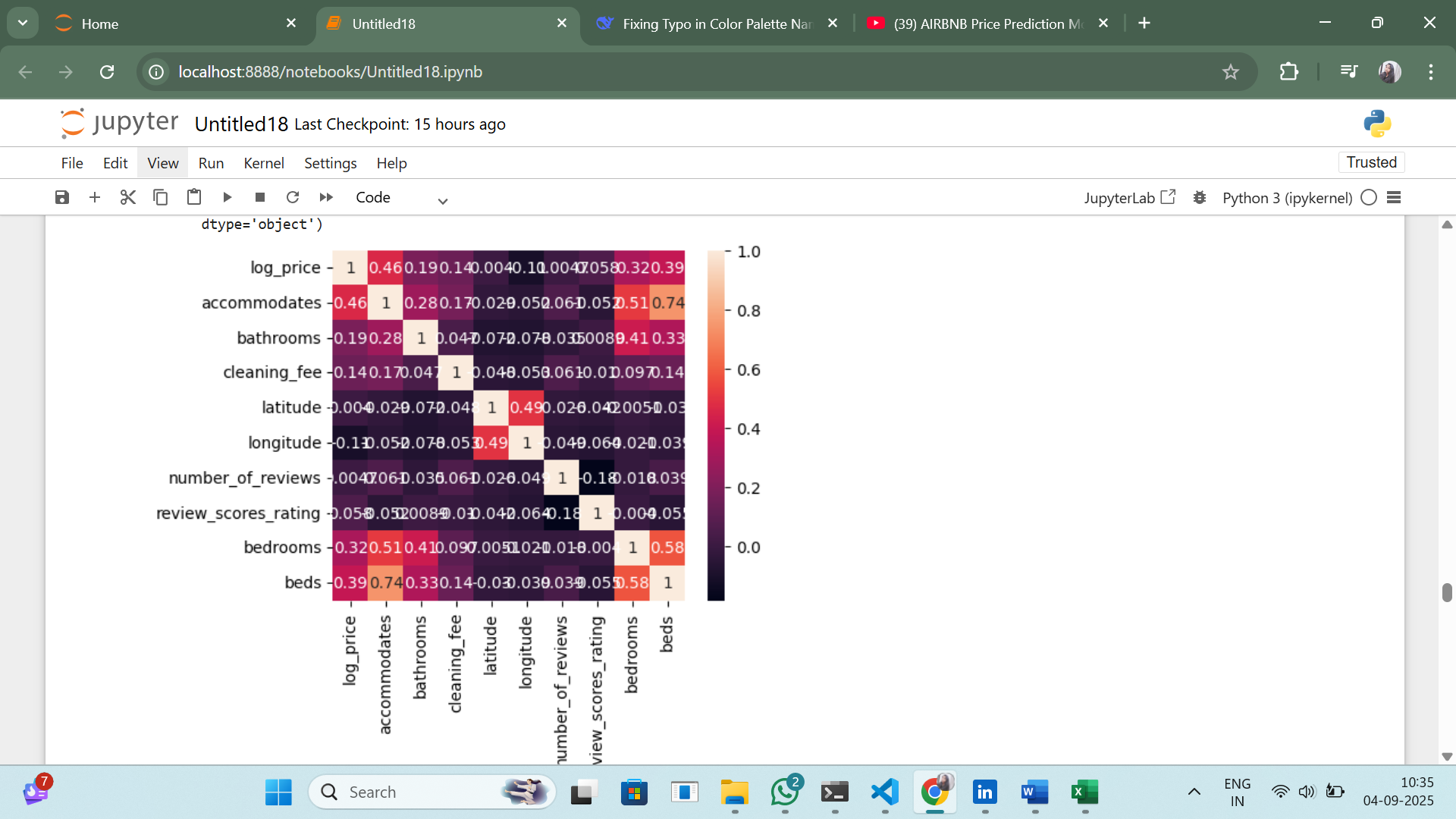
Heatmap-get correlation between different variables

corr=airbnb\_copy.corr(method='kendall',numeric\_only=True)

plt.figure(figsize=(15, 8))

sns.heatmap(corr, annot=True, )

airbnb.columns



Checking the effect of variables on the price(compare the effect on the model by the other variables and checking which is important parameter to describe the model)

Room\_Type

sns.countplot(data=airbnb\_copy,x='room\_type')

plt.title('room\_type')

fig=plt.gcf()

fig.set\_size\_inches(6,5)

plt.show()

print("After this we check the relation of the price w.r.t to room\_type")

median\_color="red"

plt.figure(figsize=(12,6))

ax=sns.boxplot(data=airbnb,x='room\_type',y='log\_price',

boxprops={'edgecolor':median\_color},

flierprops={'marker':'o','markerfacecolor':'purple','markeredgecolor':'purple'},

medianprops={'color':median\_color,'linewidth':2})

plt.title('Price Distribution by Room Type')

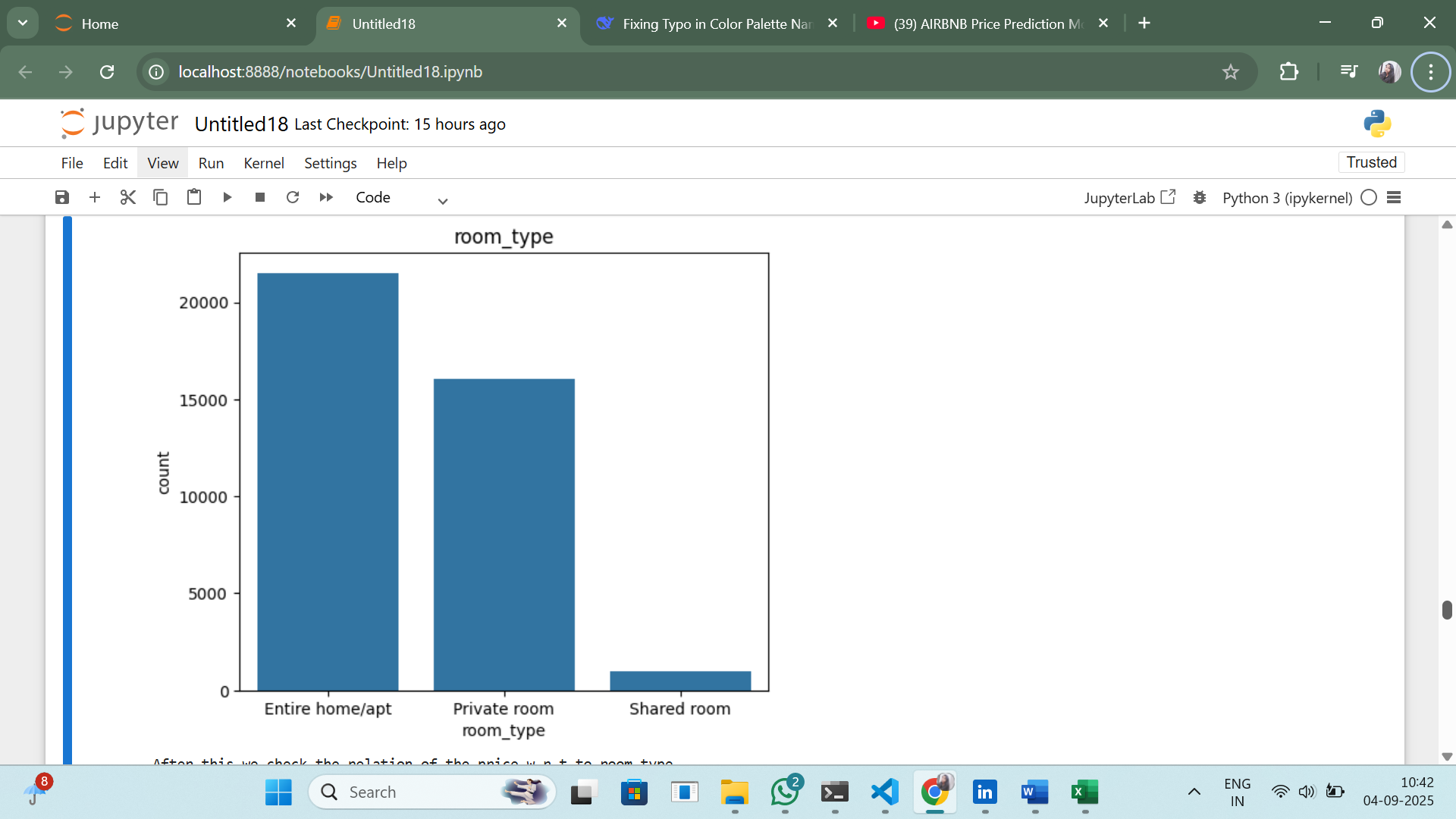
plt.xticks(rotation=45)

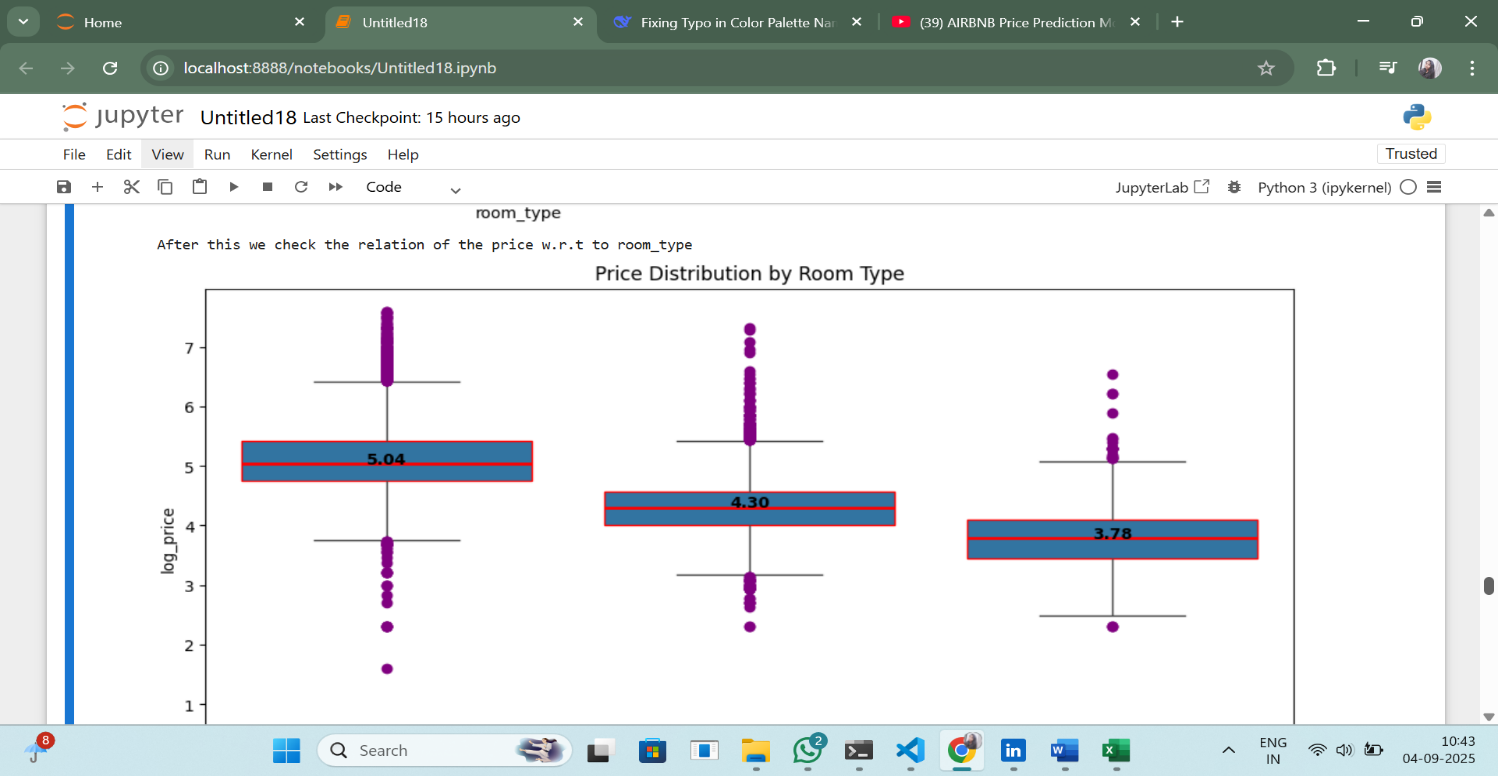
medians=airbnb.groupby(['room\_type'])['log\_price'].median()

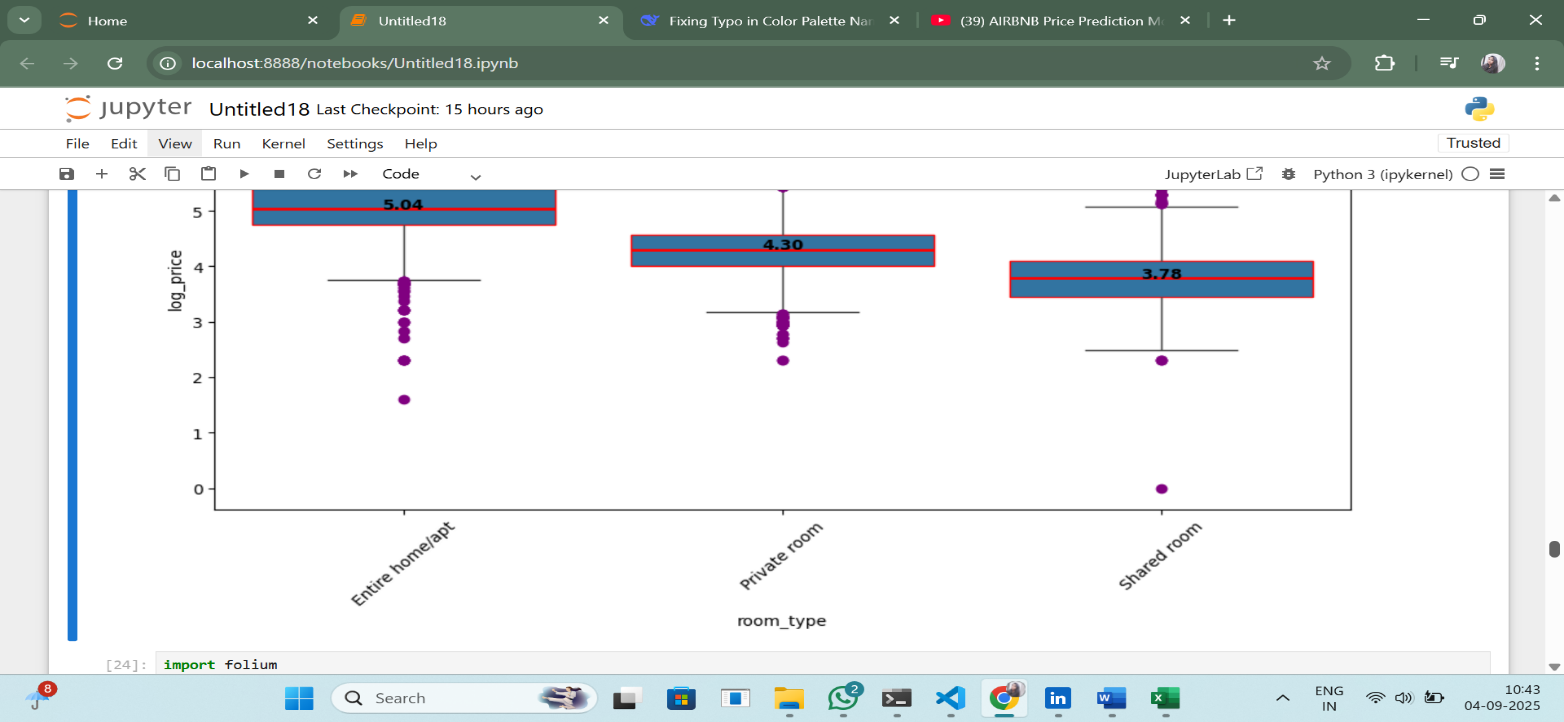
for xtick in ax.get\_xticks():

ax.text(xtick,medians[xtick],f'{medians[xtick]:.2f}',horizontalalignment='center',color='black',weight='bold')

plt.show()







import folium

import pandas as pd

import branca.colormap as cm

# Create the map

m = folium.Map(location=[40.7128, -74.0060], zoom\_start=11)

# Filter your DataFrame correctly (maybe you meant <=1000 & >=10000? but I'll keep your logic).

filtered\_price = airbnb[(airbnb['log\_price'] <= 1000) & (airbnb['log\_price'] <= 10000)]

# Define your colormap properly using scalar vmax

price\_color\_scale = cm.LinearColormap(

['green', 'yellow', 'red'],

vmin=filtered\_price['log\_price'].min(),

vmax=filtered\_price['log\_price'].max()

)

# Add circle markers with appropriate colors

for index, row in filtered\_price.iterrows():

folium.CircleMarker(

location=[row['latitude'], row['longitude']],

radius=5,

color=price\_color\_scale(row['log\_price']),

fill=True,

fill\_color=price\_color\_scale(row['log\_price']),

fill\_opacity=0.6,

popup=f"log\_price: ${row['log\_price']:.2f}<br>Room\_type: {row['room\_type']}",

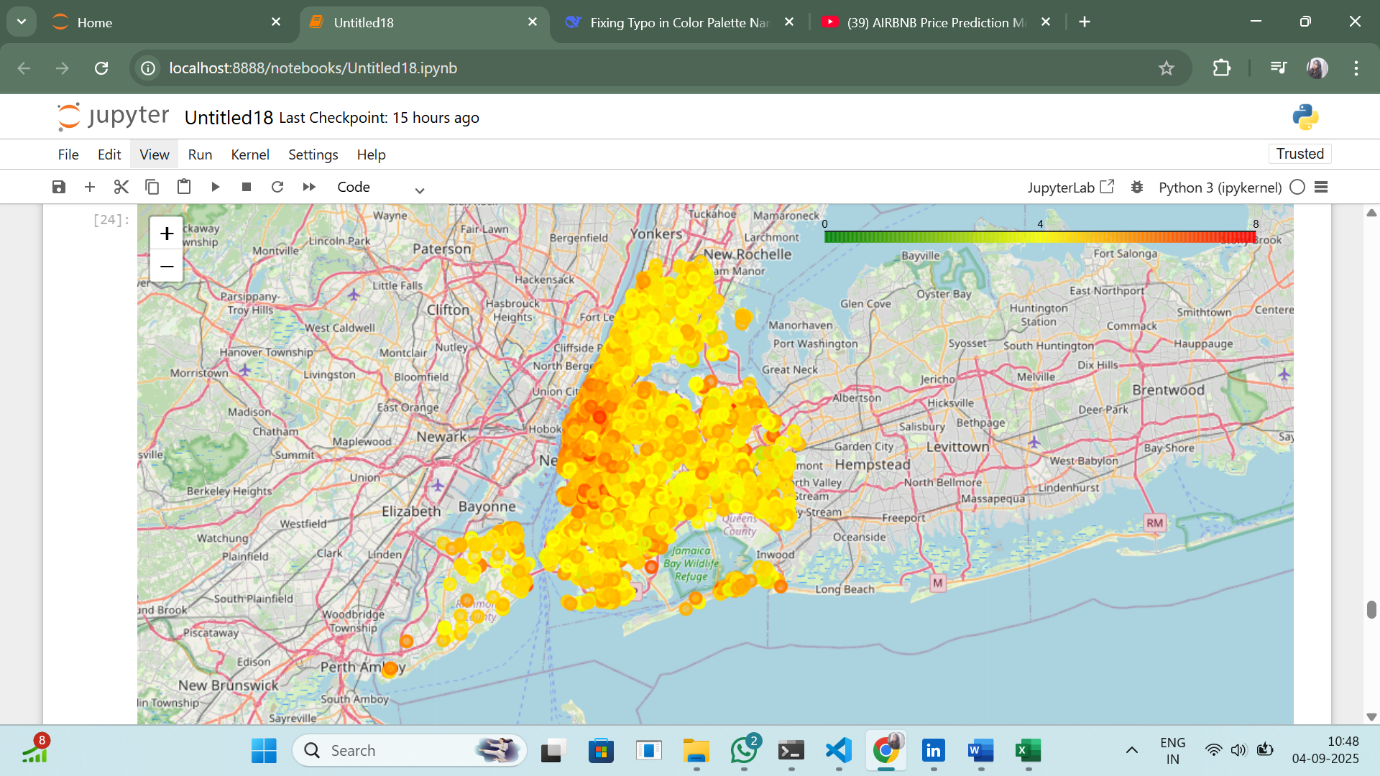
).add\_to(m)

# Add color scale legend to map

price\_color\_scale.add\_to(m)

# Display

m



Room\_Type Vs Log\_Price

sns.countplot(airbnb\_copy['room\_type'],palette="plasma")

fig=plt.gcf()

fig.set\_size\_inches(5,5)

plt.show()

print("Lets compare the room type effect on the log\_price")

plt.figure(figsize=(15,12))

sns.scatterplot(x='room\_type',y='log\_price',data=airbnb)

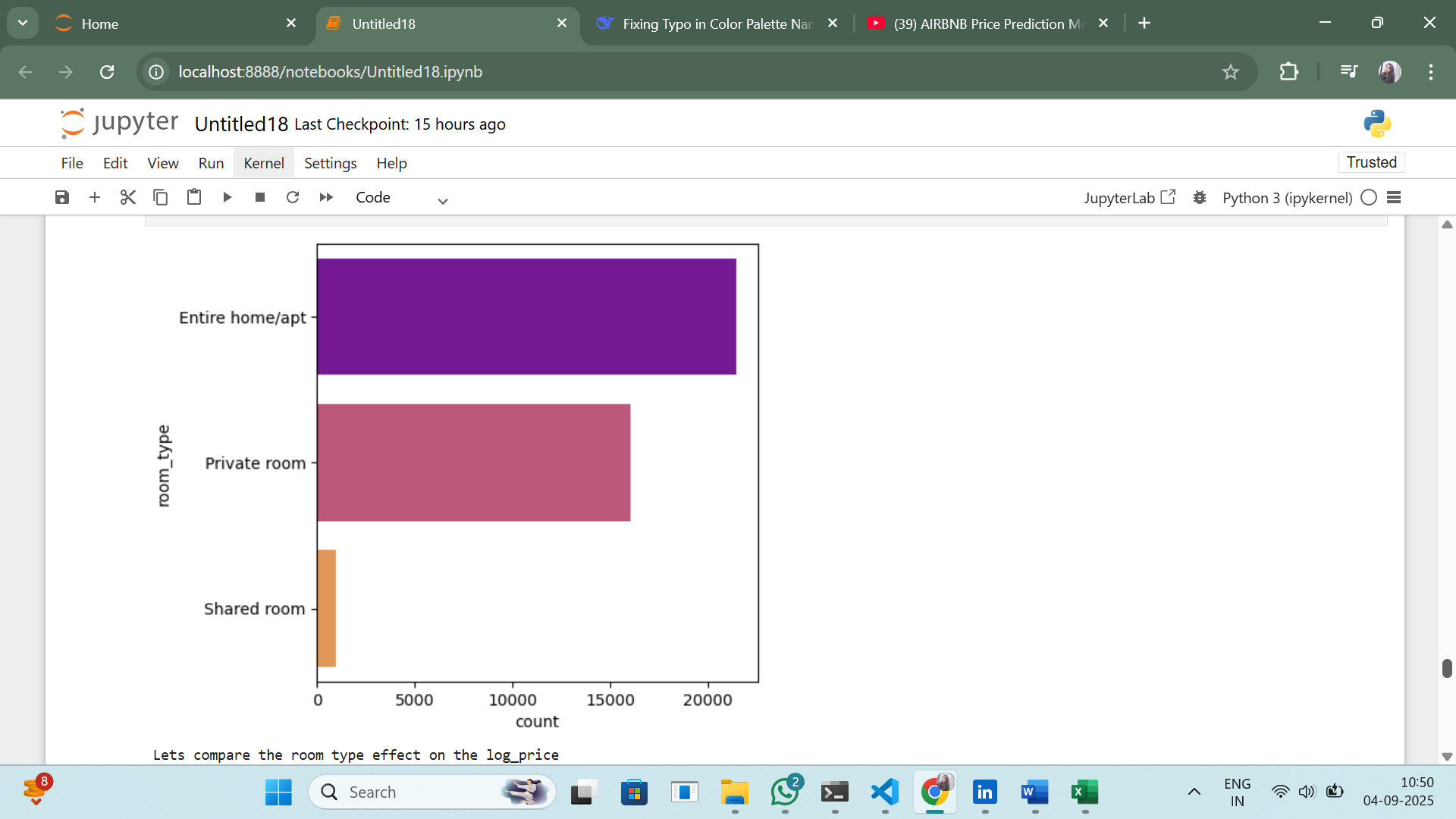
plt.xlabel("Room Type",size=13)

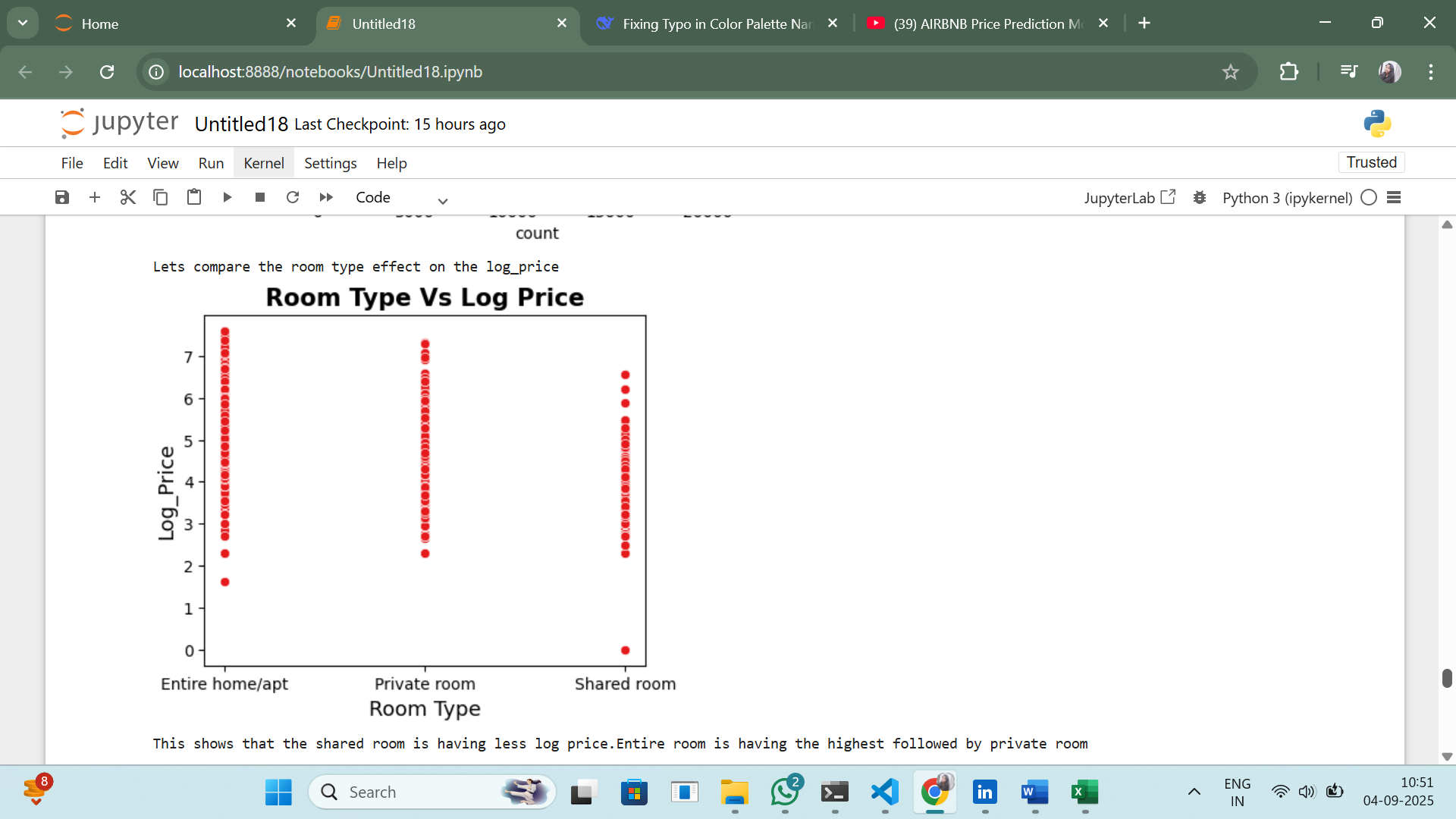
plt.ylabel("Log\_Price",size=13)

plt.title("Room Type Vs Log Price",size=15,weight='bold')

plt.show()

print("This shows that the shared room is having less log price.Entire room is having the highest followed by private room")





Room\_Type VS Log\_Price based on Property\_Type

import matplotlib.pyplot as plt

import seaborn as sns

fig,axes=plt.subplots(2,2,figsize=(25,15))

fig.suptitle("Room\_type Vs Log\_price Vs property\_type", fontsize=16,fontweight='bold')

room\_types=airbnb\_copy['room\_type'].unique()

for i, room\_type in enumerate(room\_types):

row=i//2

col=i%2

ax=axes[row,col]

data\_group = airbnb\_copy[airbnb\_copy['room\_type']==room\_type]

sns.scatterplot(x='property\_type',y='log\_price',hue='property\_type',size='property\_type',sizes=(50,200),palette='Dark2',data=data\_group,ax=ax)

ax.set\_title(f"Room Type : {room\_type}")

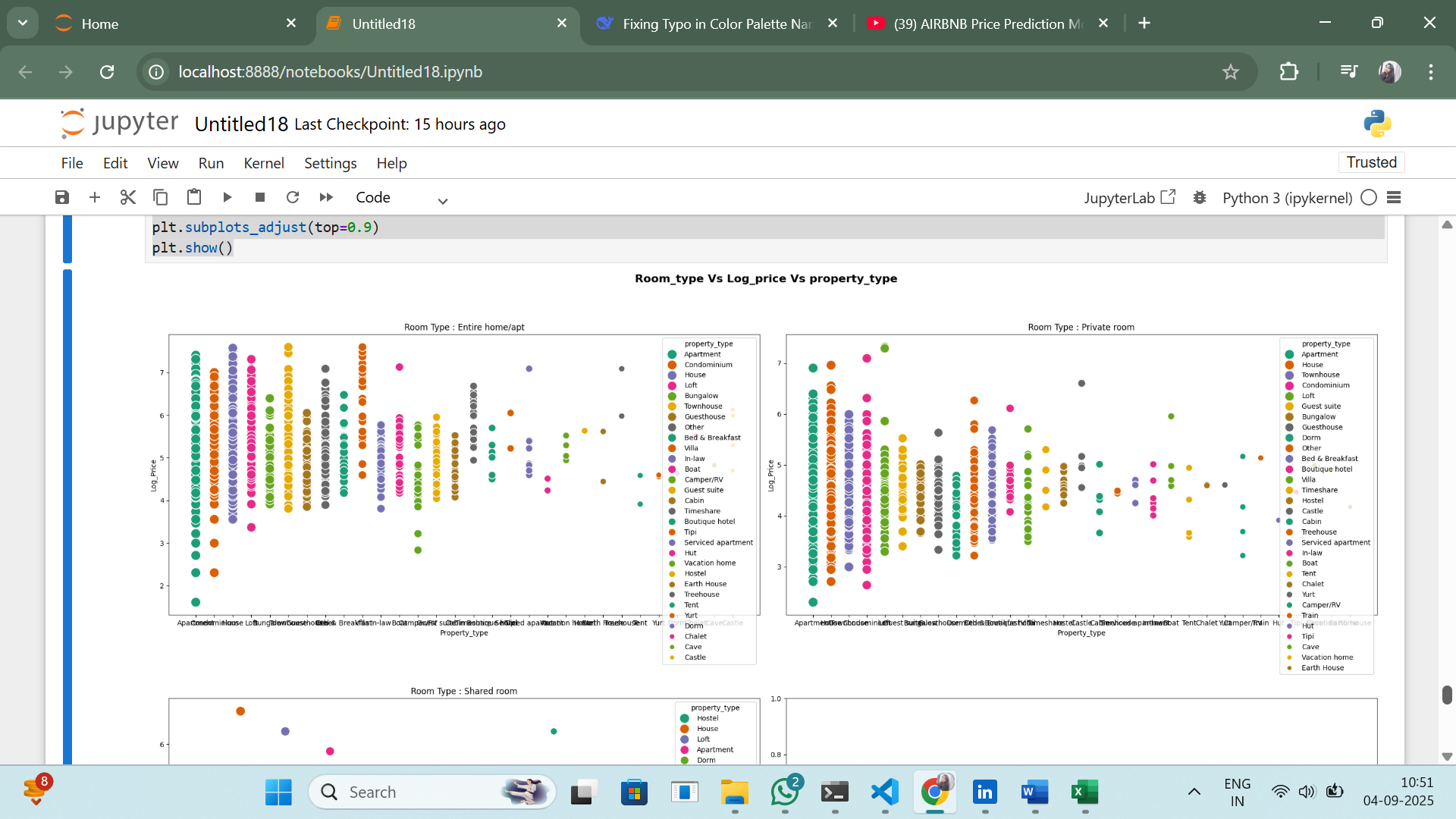
ax.set\_xlabel("Property\_type")

ax.set\_ylabel("Log\_Price")

plt.tight\_layout()

plt.subplots\_adjust(top=0.9)

plt.show()





Log\_price Vs Number\_of\_reviews Vs Property\_type

plt.figure(figsize=(5,6))

sns.set\_palette("Set1")

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Apartment'], label='Apartment'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Condominium'],label='Condominium'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='House'],label='House'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Townhouse'], label='Townhouse'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Loft'],label='Loft'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Hostel'],label='Hostel'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Guest Suite'], label='Guest Suite'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Bungalow'],label='Bungalow'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Dorm'],label='Dorm'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Other'], label='Apartment'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='In-law'],label='In-law'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Boat'],label='Boat'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Camper/Rv'], label='Camper/Rv'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Castle'],label='Castle'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Cabin'],label='Cabin'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Treehouse'], label='Treehouse'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Serviced apartment'],label='Serviced apartment'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Tipi'],label='Tipi'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Hut'], label='Hut'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Vacation home'],label='Vacation home'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Tent'],label='Tent'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Chalet'], label='Chalet'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Yurt'],label='Yurt'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Earth House'],label='Earth House'),

sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Train'],label='Train'),

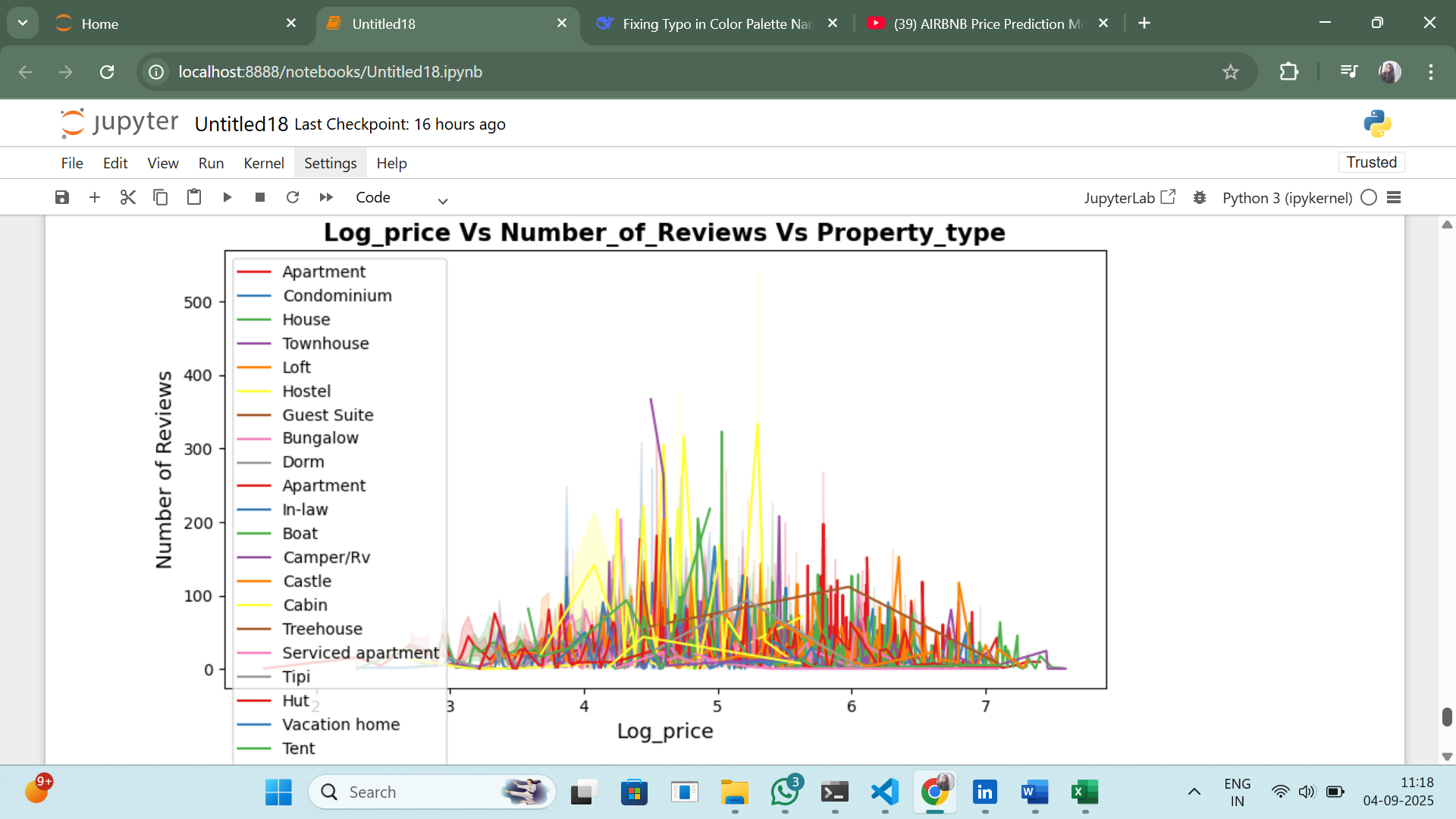
sns.lineplot(x='log\_price',y='number\_of\_reviews',data=airbnb\_copy[airbnb\_copy['property\_type']=='Cave'],label='Cave')

plt.xlabel("Log\_price",size=13)

plt.ylabel("Number of Reviews",size=13)

plt.title("Log\_price Vs Number\_of\_Reviews Vs Property\_type",size=15,weight='bold')

plt.show()



Log\_Price Vs Bed\_type

import folium

import pandas as pd

import branca.colormap as cm

m=folium.Map(location=[40.7128, -74.0060], zoom\_start=11)

colors=['green','blue','purple','red','orange']

price\_min=airbnb\_copy['log\_price'].min()

price\_max=airbnb\_copy['log\_price'].max()

colormap=cm.LinearColormap(colors=colors,vmin=price\_min, vmax=price\_max)

for index, row in airbnb.iterrows():

folium.CircleMarker(

location=[row['latitude'], row['longitude']],

radius=5,

color=colormap(row['log\_price']),

fill=True,

fill\_color=colormap(row['log\_price']),

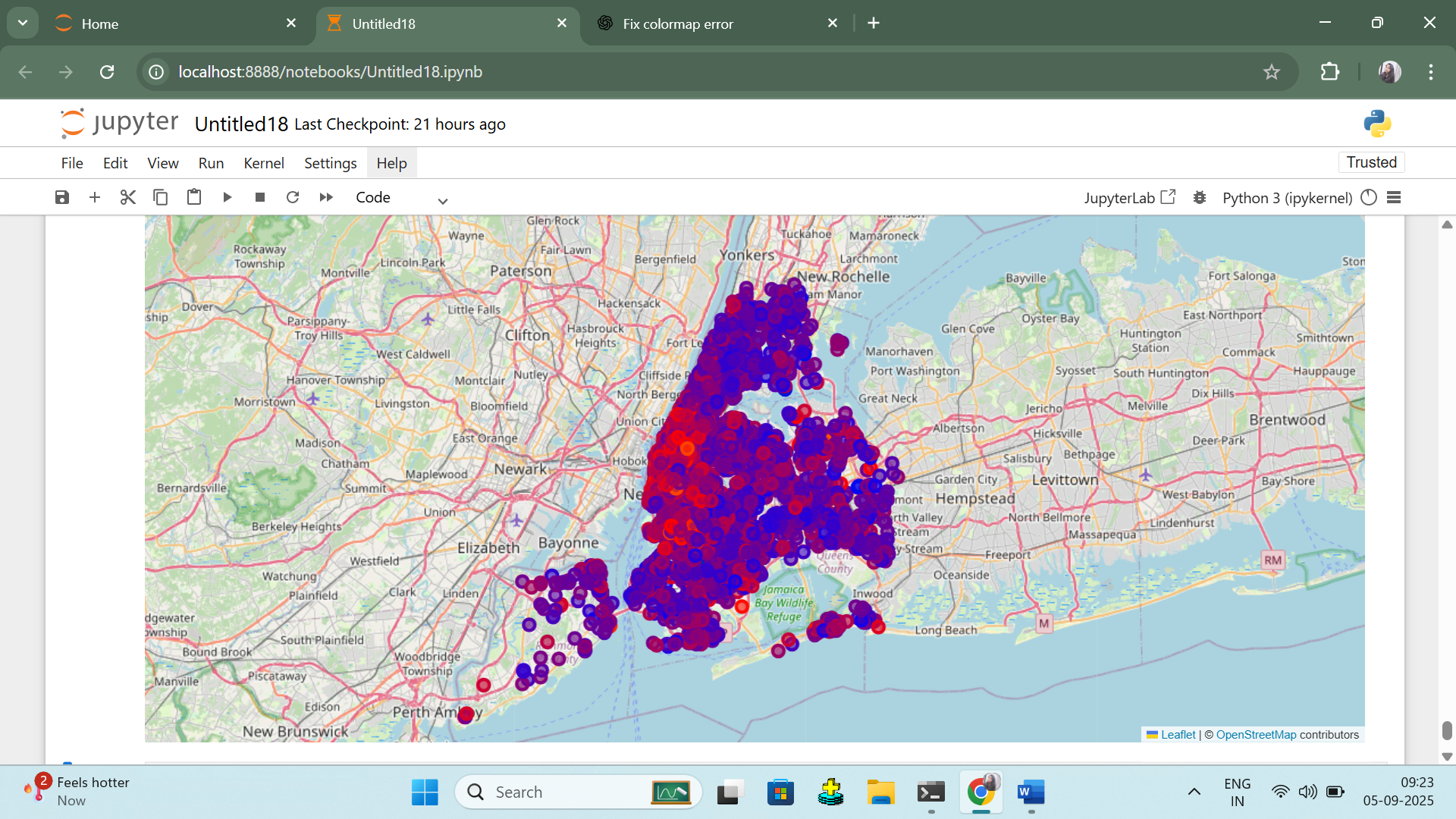
fill\_opacity=0.6,

popup=f"log\_price: ${row['log\_price']:.2f},bed\_type: {row['bed\_type']}",

).add\_to(m)

colormap.add\_to(m)

m



Data Modeling

airbnb\_copy.head()

import matplotlib.pyplot as plt

plt.figure(figsize=(12,4))

plt.scatter(airbnb['host\_response\_rate'],airbnb['log\_price'],alpha=0.5)

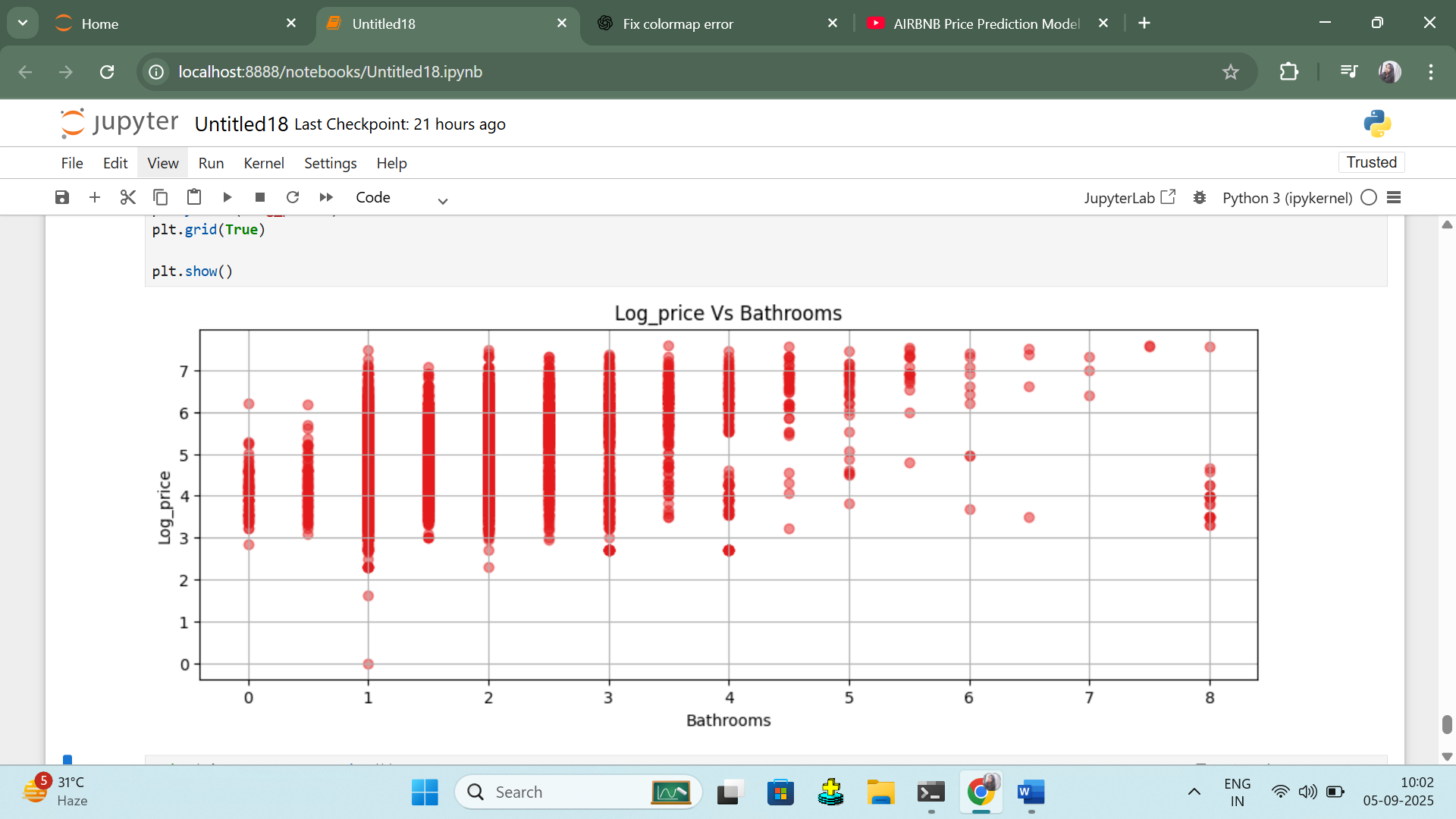
plt.title('Log\_price Vs host\_response\_rate')

plt.xlabel('host\_response\_rate')

plt.ylabel('Log\_price')

plt.grid(True)

plt.show()



import matplotlib.pyplot as plt

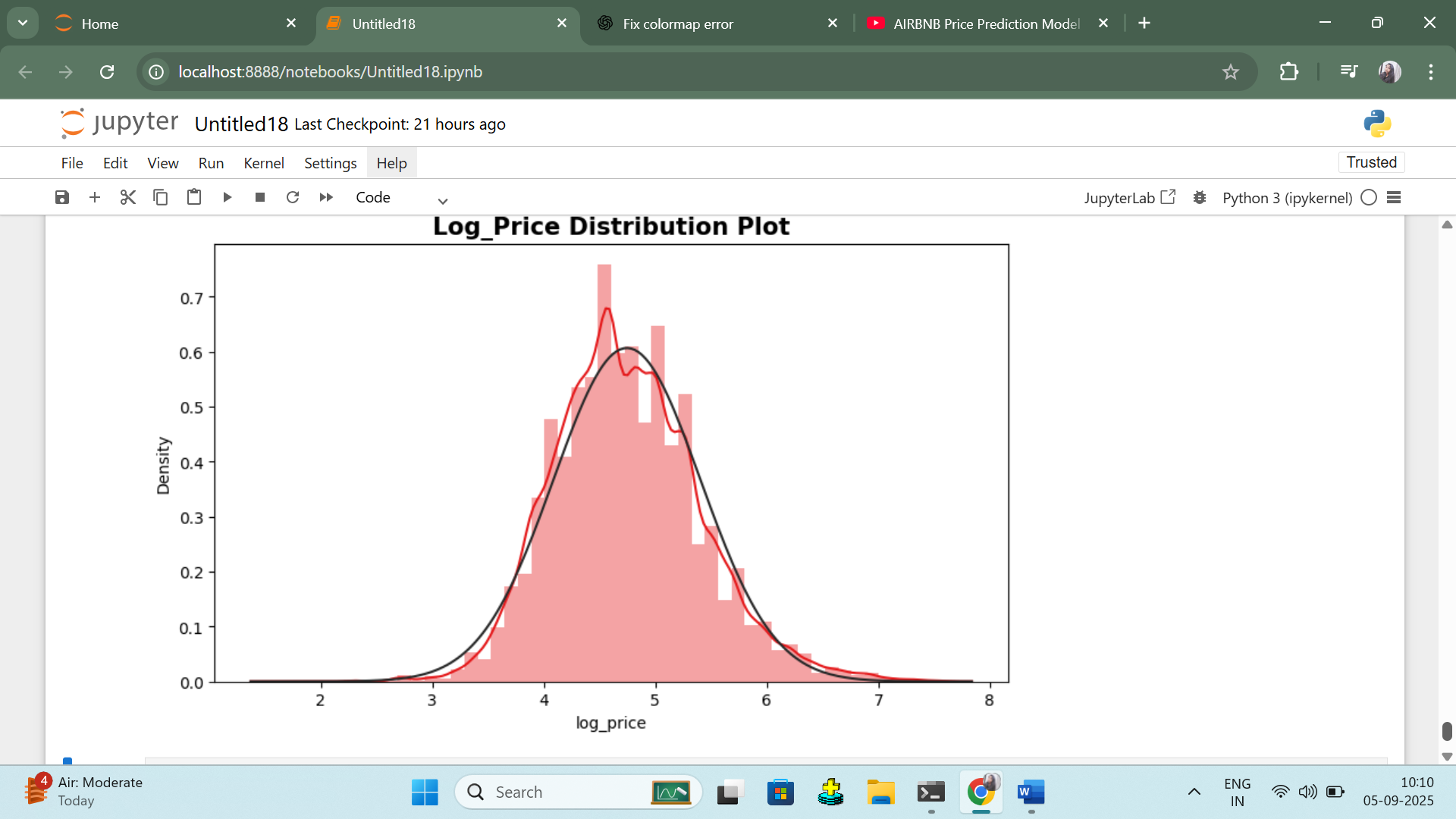
from scipy.stats import norm

import seaborn as sns

plt.figure(figsize=(9,5))

sns.distplot(airbnb\_copy['log\_price'],fit=norm)

plt.title("Log\_Price Distribution Plot",size=15,weight='bold')



airbnb.head()

airbnb\_copy.info()

airbnb\_copy.drop(['latitude','longitude','neighbourhood','number\_of\_reviews'],axis=1,inplace=True)

airbnb\_copy.isnull().sum()

Checking Multicollinearity

multicollinearity, v=np.linalg.eig(corr)

multicollinearity

Encoding the data

def Encode(airbnb\_copy):

for column in airbnb\_copy.columns[airbnb\_copy.columns.isin(['neighbourhood','room\_type'])]:

airbnb\_copy[column]=airbnb\_copy[column].factorize()[0]

return airbnb\_copy

airbnb\_copy\_en=Encode(airbnb\_copy.copy())

airbnb\_copy\_en.head()

#Import necessary libraries

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression,Lasso,Ridge

from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import r2\_score

#Define independent and dependent variables

x=airbnb\_copy\_en.iloc[:,[0,1,3,4,5]]

y=airbnb\_copy\_en['log\_price']

#Split the data into training and testing sets

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.1,random\_state=353)

#Display a sample of the training data

print("Sample of x\_train:")

print(x\_train.head())

print("\nSample of y\_train:")

print(y\_train.head())

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error

import numpy as np

airbnb = airbnb.replace(r'\r+|\n+|\t+', '', regex=True)

#Create and train a linear Regression model

reg=LinearRegression()

reg.fit(x\_train,y\_train)

#Make predictions using the linear Regression model

y\_pred\_linear=reg.predict(x\_test)

#Calculated and print the R-squared score for Linear Regression

linear\_regression\_r2=r2\_score(y\_test,y\_pred\_linear)

print("\nR-squared score for Linear Regression :",linear\_regression\_r2)

#calculate RMSE for linear Regression

linear\_rmse=np.sqrt(mean\_squared\_error(y\_test,y\_pred\_linear))

print("RMSE for Linear Regression :",linear\_rmse)

#calculate MAE for linear Regression

linear\_mae=mean\_absolute\_error(y\_test,y\_pred\_linear)

print("MAE for Linear Regression :",linear\_mae)

#Create a dataframe to display actual vs predicted values

comparison\_airbnb=pd.DataFrame({'Actual':y\_test,'Predicted':y\_pred\_linear})

#Print the DataFrame

print(comparison\_airbnb)

