Hands-on lab: Exploratory Data Analysis - Laptops Pricing dataset

Estimated time needed: **45** minutes

In this lab, you will use the skills acquired throughout the module, to explore the effect of different features on the price of laptops.

Objectives

After completing this lab you will be able to:

- Visualize individual feature patterns
- Run descriptive statistical analysis on the dataset
- Use groups and pivot tables to find the effect of categorical variables on price
- Use Pearson Correlation to measure the interdependence between variables

Setup

For this lab, we will be using the following libraries:

- `skillsnetwork` for downloading the data
- `pandas` for managing the data.
- `numpy` for mathematical operations.
- `scipy` for statistical operations.
- `seaborn` for visualizing the data.
- `matplotlib` for additional plotting tools.

Install Required Libraries

You can install the required libraries by simply running the `pip install` command with a `%` sign before it. For this environment, `seaborn` library requires installation.

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

Importing Required Libraries

We recommend you import all required libraries in one place (here):

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
%matplotlib inline
```

Import the dataset

You should download the modified version of the data set from the last module. Run the following code block to download the CSV file to this environment.

The functions below will download the dataset into your browser:

```
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-Coursera/laptop_pricing_dataset_mod2.csv"

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

Import the file to a pandas dataframe.

```
df = pd.read_csv(file_name, header=0)
```

> Note: This version of the lab is working on JupyterLite, which requires the dataset to be downloaded to the interface. While working on the downloaded version of this notebook on their local machines, the learners can simply **skip the steps above**, and simply use the URL directly in the `pandas.read_csv()` function. You can uncomment and run the statements in the cell below.

```
#filepath="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DA0101EN-Coursera/laptop_pricing_dataset_mod2.csv"
#df = pd.read_csv(filepath, header=None)
```

Print the first 5 entries of the dataset to confirm loading.

```
df.head(5)
```

```
   Unnamed: 0.1  Unnamed: 0 Manufacturer  Category  GPU  OS  CPU_core
\
0             0           0         Acer         4    2   1         5

1             1           1         Dell         3    1   1         3

2             2           2         Dell         3    1   1         7

3             3           3         Dell         4    2   1         5

4             4           4           HP         4    2   1         7


   Screen_Size_inch  CPU_frequency  RAM_GB  Storage_GB_SSD
Weight_pounds  \
0              14.0       0.551724       8             256
3.52800
1              15.6       0.689655       4             256
4.85100
2              15.6       0.931034       8             256
4.85100
3              13.3       0.551724       8             128
2.69010
4              15.6       0.620690       8             256
4.21155

   Price Price-binned  Screen-Full_HD  Screen-IPS_panel
0    978          Low               0                 1
1    634          Low               1                 0
2    946          Low               1                 0
3   1244          Low               0                 1
4    837          Low               1                 0
```

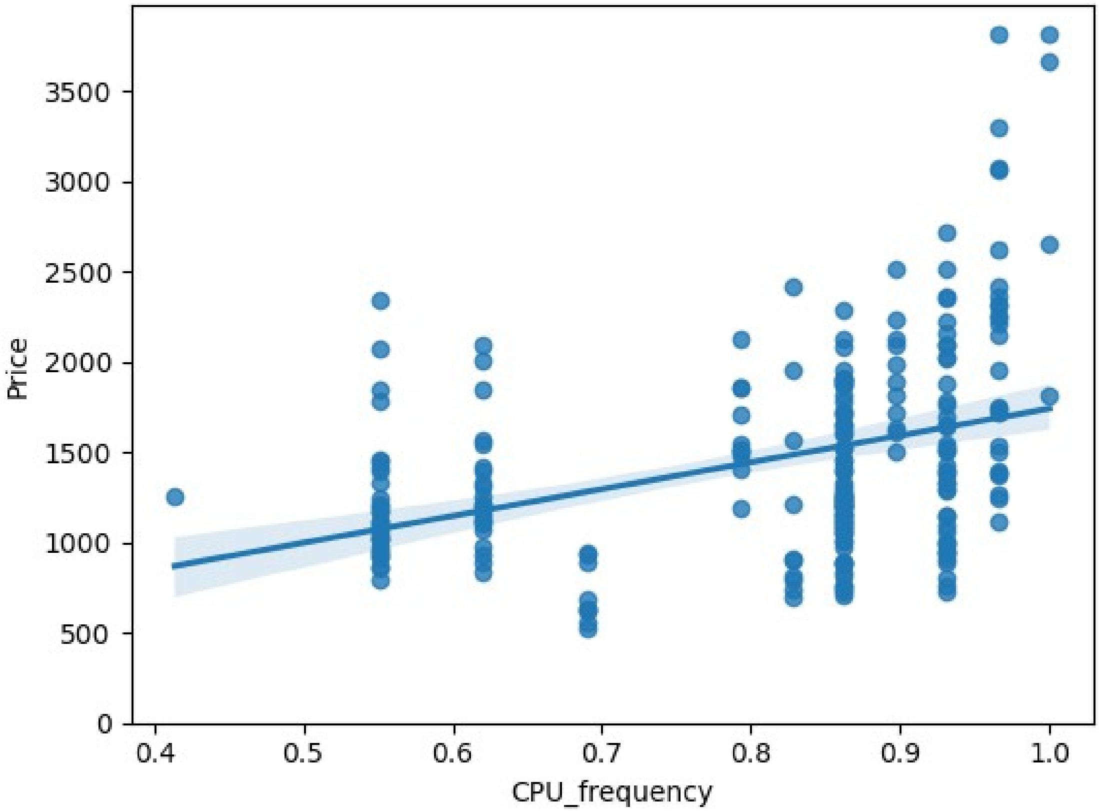
Task 1 - Visualize individual feature patterns

Continuous valued features

Generate regression plots for each of the parameters "CPU_frequency", "Screen_Size_inch" and "Weight_pounds" against "Price". Also, print the value of correlation of each feature with "Price".

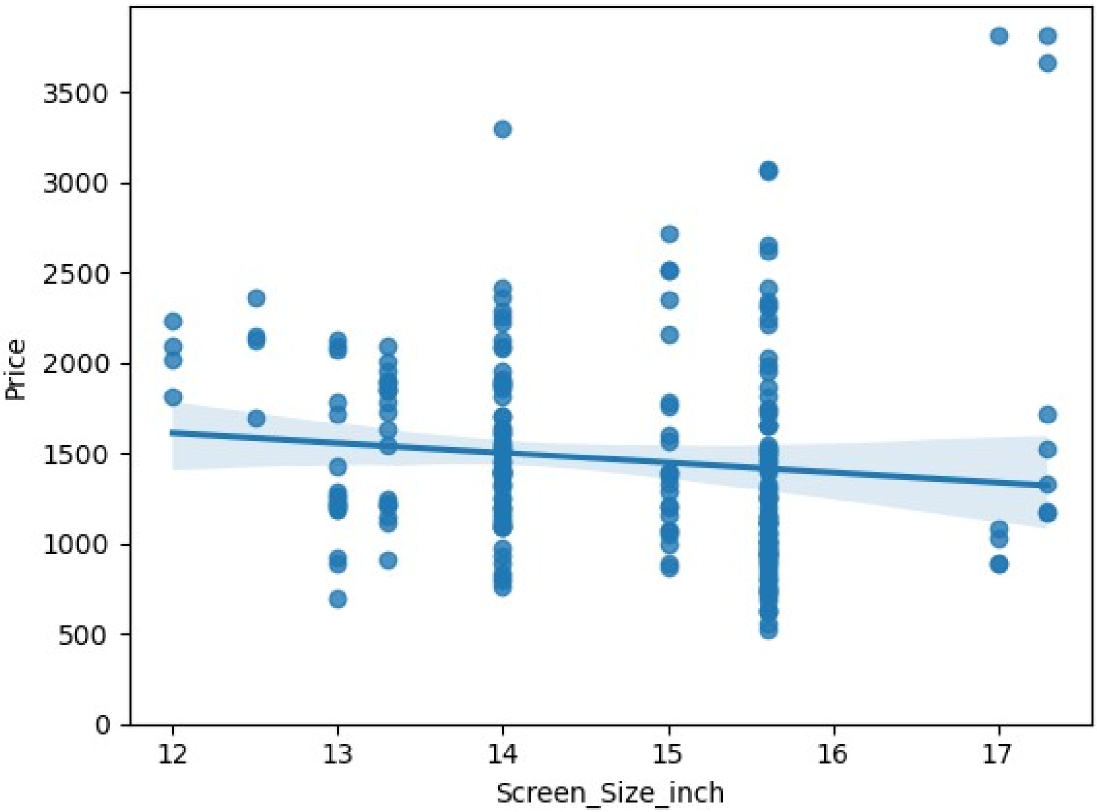
```
# Write your code below and press Shift+Enter to execute
# CPU_frequency plot
sns.regplot(x="CPU_frequency", y="Price", data=df)
plt.ylim(0,)
```

```
(0.0, 3974.15)
```



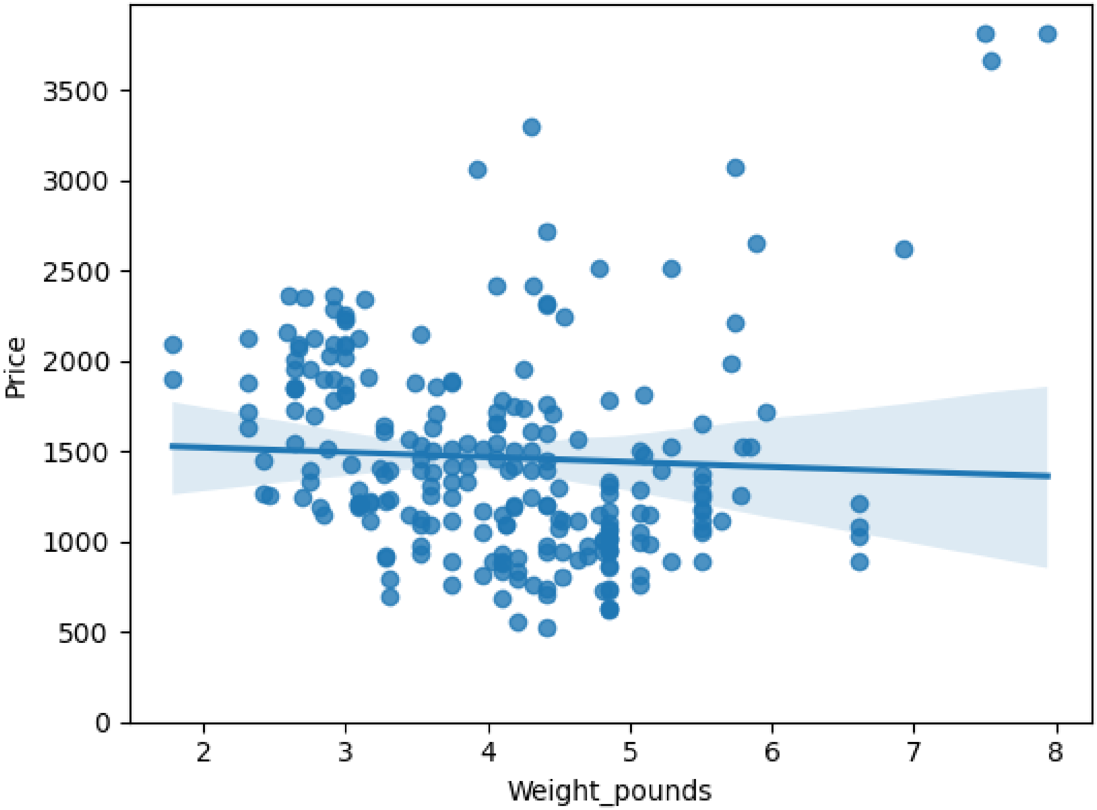
```
# Write your code below and press Shift+Enter to execute
# Screen_Size_inch plot
sns.regplot(x="Screen_Size_inch", y="Price", data=df)
plt.ylim(0,)
```

```
(0.0, 3974.15)
```



```
# Write your code below and press Shift+Enter to execute
# Weight_pounds plot
sns.regplot(x="Weight_pounds", y="Price", data=df)
plt.ylim(0,)
```

```
(0.0, 3974.15)
```



```
# Correlation values of the three attributes with Price
for param in ["CPU_frequency", "Screen_Size_inch","Weight_pounds"]:
    print(f"Correlation of Price and {param} is ",
df[[param,"Price"]].corr())

Correlation of Price and CPU_frequency is
CPU_frequency     Price
CPU_frequency       1.000000  0.366666
Price               0.366666  1.000000
Correlation of Price and Screen_Size_inch is
Screen_Size_inch     Price
Screen_Size_inch          1.000000 -0.110644
Price                    -0.110644  1.000000
Correlation of Price and Weight_pounds is
Weight_pounds     Price
Weight_pounds       1.000000 -0.050312
Price              -0.050312  1.000000
```

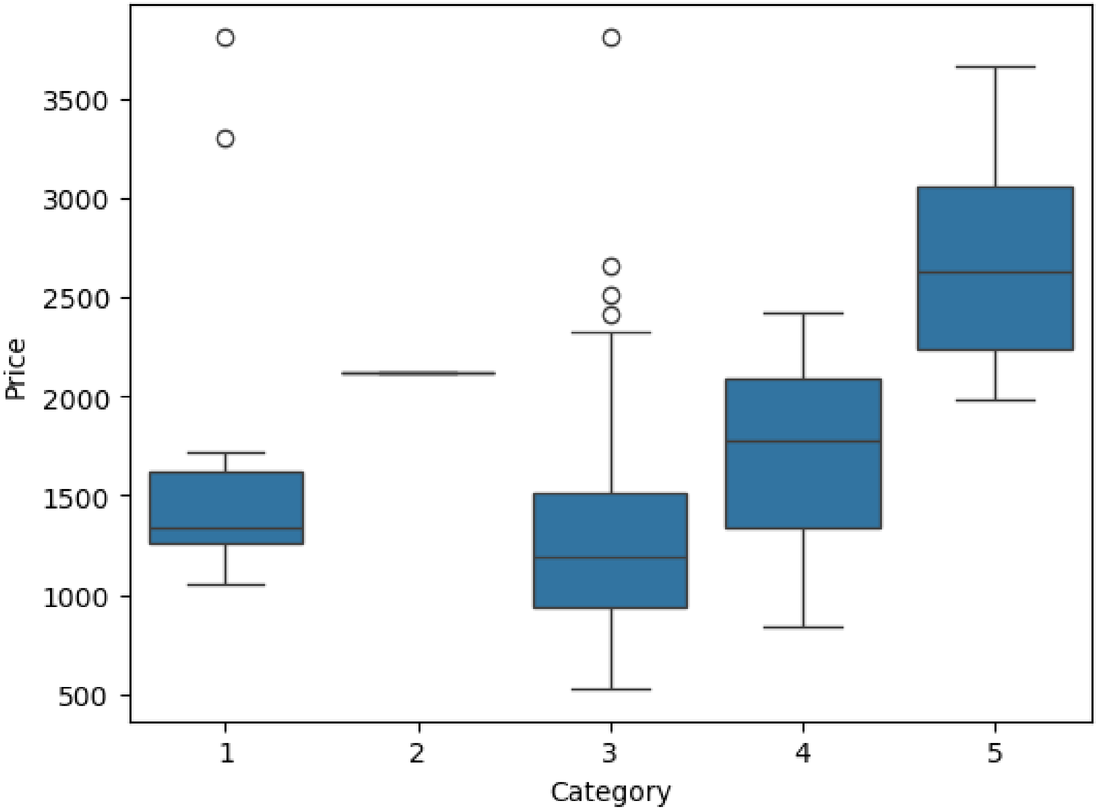
Interpretation: "CPU_frequency" has a 36% positive correlation with the price of the laptops. The other two parameters have weak correlation with price.

Categorical features

Generate Box plots for the different feature that hold categorical values. These features would be "Category", "GPU", "OS", "CPU_core", "RAM_GB", "Storage_GB_SSD"

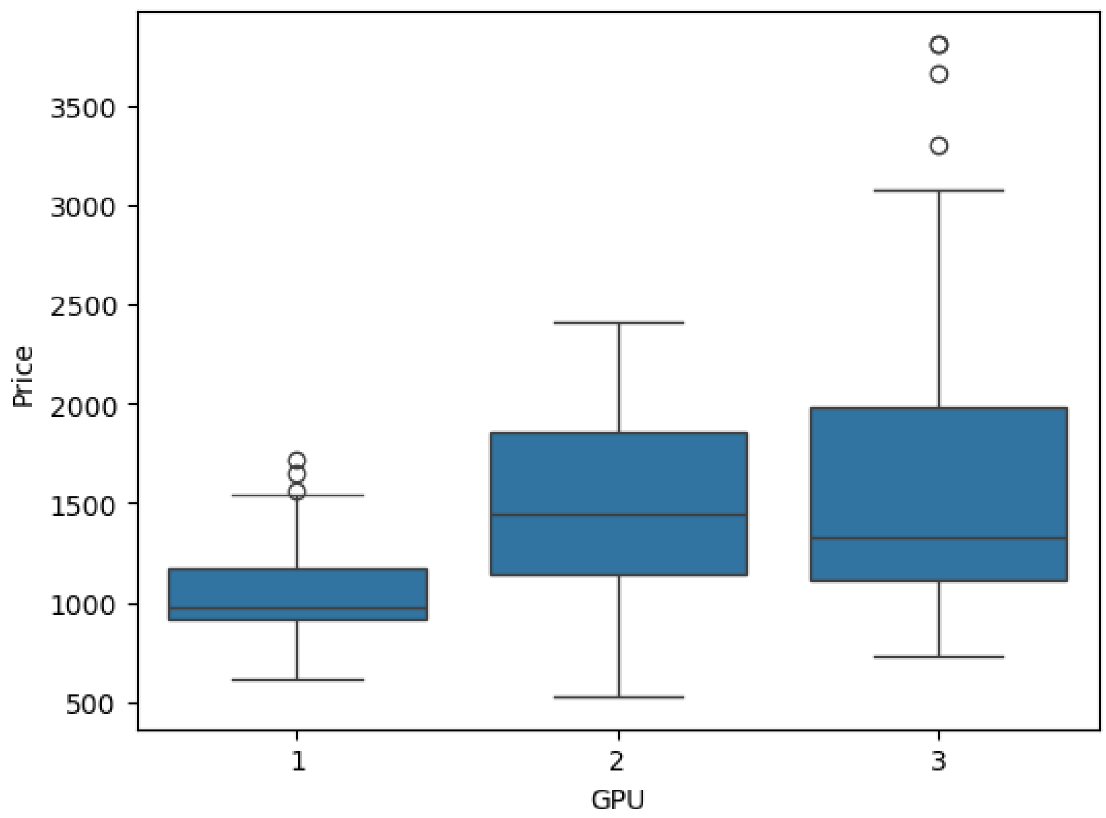
```
# Write your code below and press Shift+Enter to execute
# Category Box plot
sns.boxplot(x="Category", y="Price", data=df)
```

```
<AxesSubplot:xlabel='Category', ylabel='Price'>
```

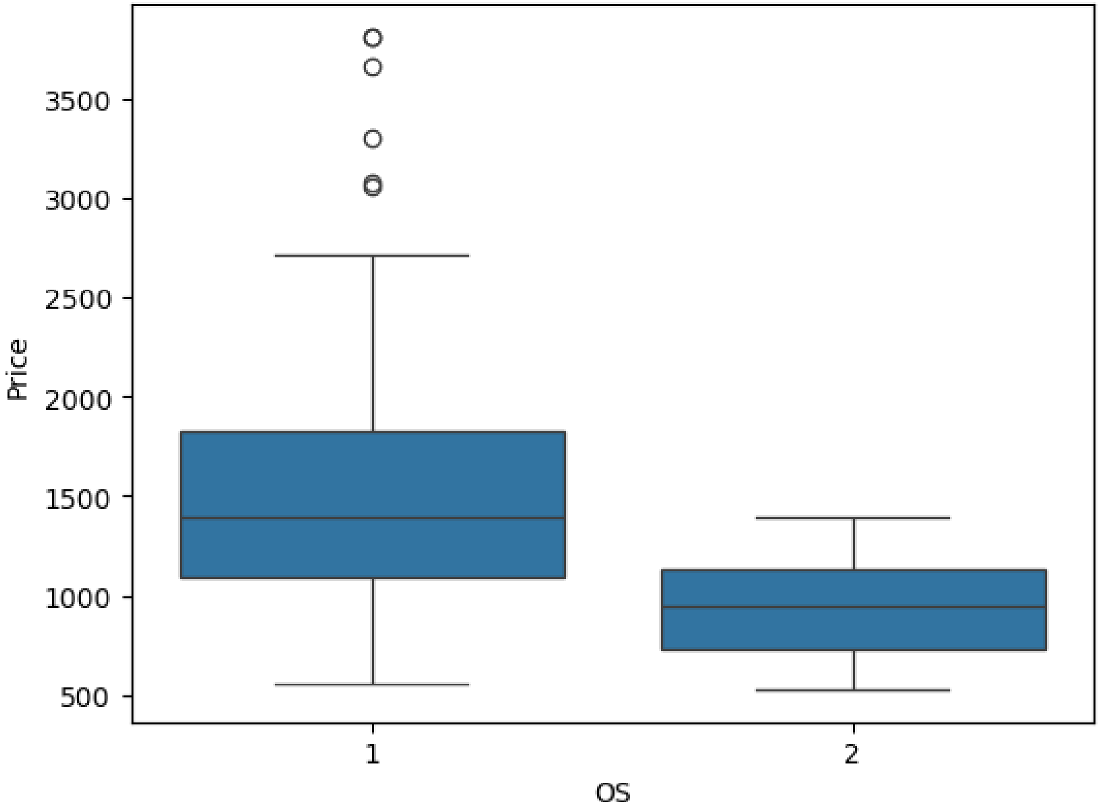


```
# Write your code below and press Shift+Enter to execute
# GPU Box plot
sns.boxplot(x="GPU", y="Price", data=df)
```

```
<AxesSubplot:xlabel='GPU', ylabel='Price'>
```

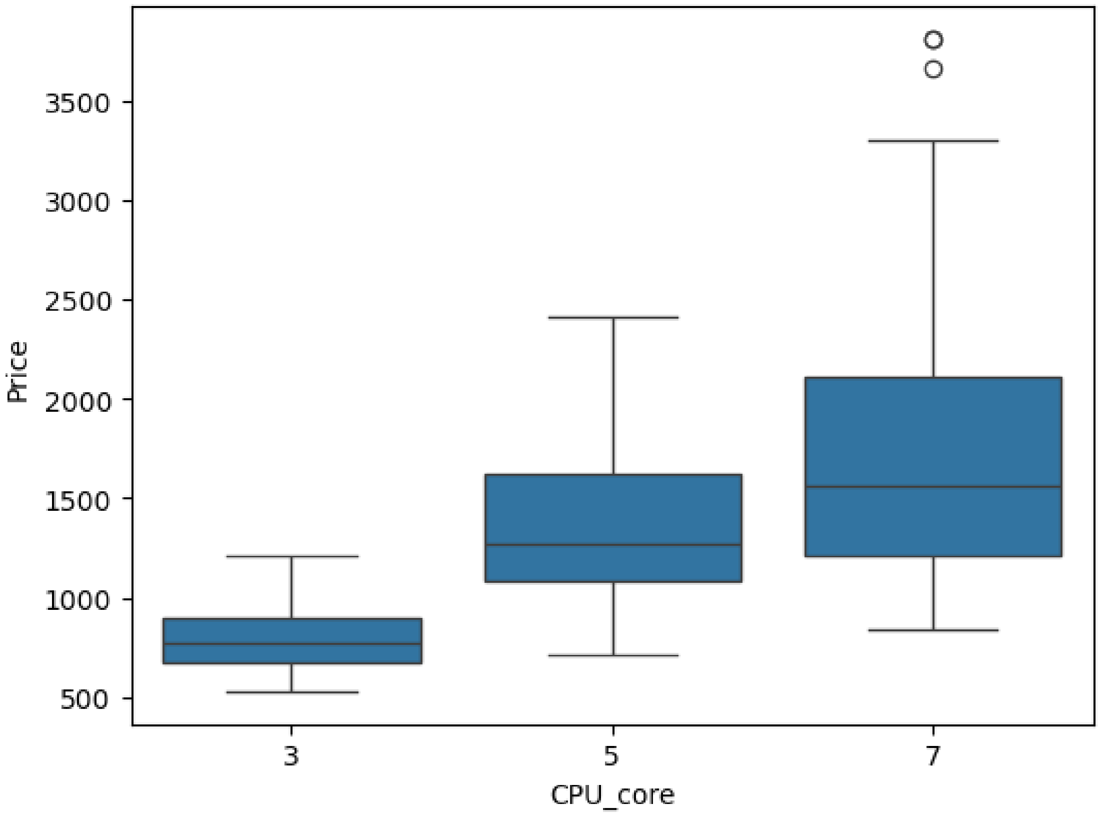


```
# Write your code below and press Shift+Enter to execute
# OS Box plot
sns.boxplot(x="OS", y="Price", data=df)

<AxesSubplot:xlabel='OS', ylabel='Price'>
```



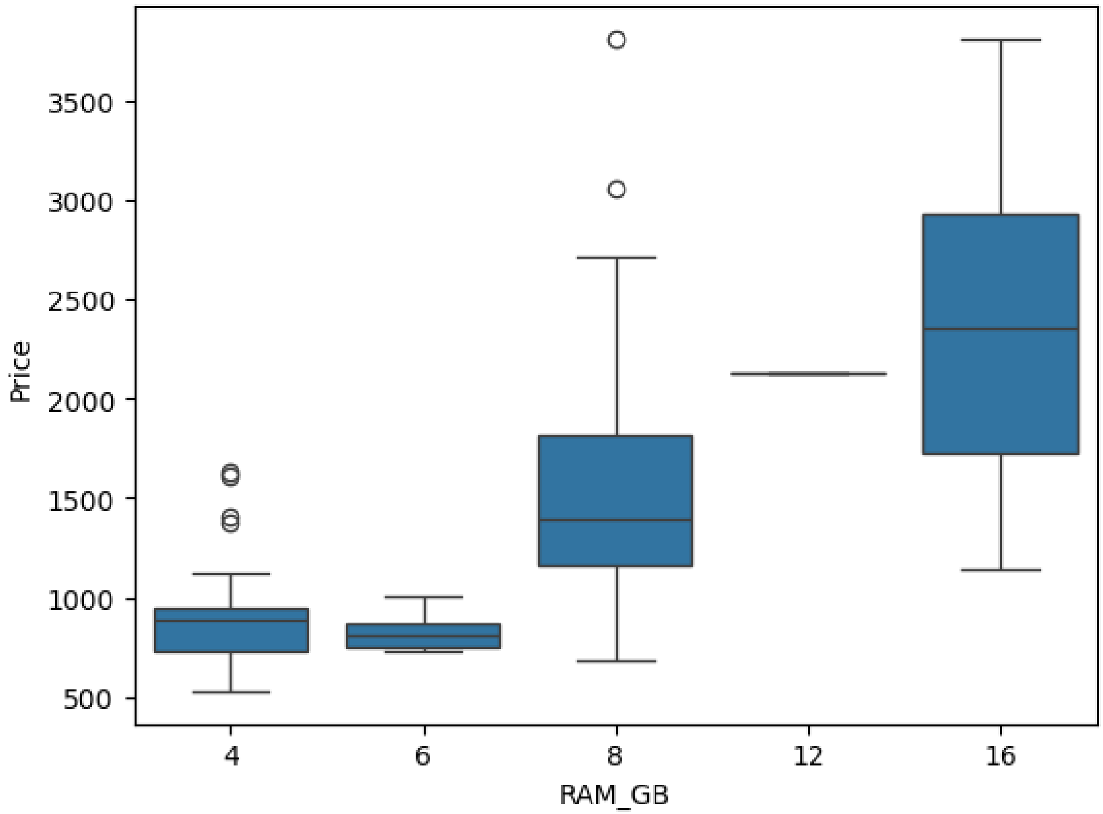
```
# Write your code below and press Shift+Enter to execute
# CPU_core Box plot
sns.boxplot(x="CPU_core", y="Price", data=df)
```

```
<AxesSubplot:xlabel='CPU_core', ylabel='Price'>
```

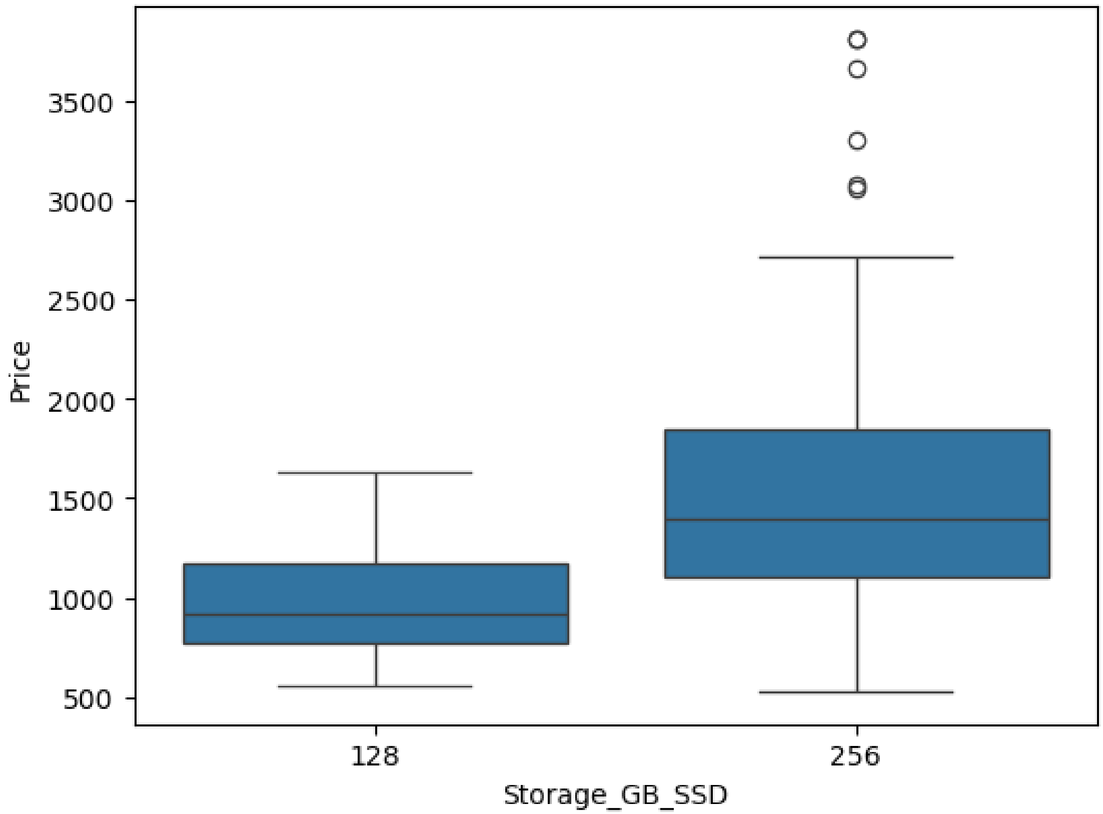


```
# Write your code below and press Shift+Enter to execute
# RAM_GB Box plot
sns.boxplot(x="RAM_GB", y="Price", data=df)
```

```
<AxesSubplot:xlabel='RAM_GB', ylabel='Price'>
```



```
# Write your code below and press Shift+Enter to execute
# Storage_GB_SSD Box plot
sns.boxplot(x="Storage_GB_SSD", y="Price", data=df)

<AxesSubplot:xlabel='Storage_GB_SSD', ylabel='Price'>
```



Task 2 - Descriptive Statistical Analysis

Generate the statistical description of all the features being used in the data set. Include "object" data types as well.

```
# Write your code below and press Shift+Enter to execute
print(df.describe())
print(df.describe(include=['object']))

       Unnamed: 0.1  Unnamed: 0    Category         GPU          OS  \
count    238.000000  238.000000  238.000000  238.000000  238.000000   
mean     118.500000  118.500000    3.205882    2.151261    1.058824   
std       68.848868   68.848868    0.776533    0.638282    0.235790   
min        0.000000    0.000000    1.000000    1.000000    1.000000   
25%       59.250000   59.250000    3.000000    2.000000    1.000000   
50%      118.500000  118.500000    3.000000    2.000000    1.000000   
75%      177.750000  177.750000    4.000000    3.000000    1.000000   
max      237.000000  237.000000    5.000000    3.000000    2.000000   

         CPU_core  Screen_Size_inch  CPU_frequency      RAM_GB  \
count  238.000000        238.000000     238.000000  238.000000   
mean     5.630252         14.688655       0.813822    7.882353   
std      1.241787          1.166045       0.141860    2.482603   
```

```
min      3.000000         12.000000       0.413793   4.000000
25%      5.000000         14.000000       0.689655   8.000000
50%      5.000000         15.000000       0.862069   8.000000
75%      7.000000         15.600000       0.931034   8.000000
max      7.000000         17.300000       1.000000  16.000000

       Storage_GB_SSD  Weight_pounds        Price  Screen-Full_HD  \
count      238.000000     238.000000   238.000000      238.000000
mean       245.781513       4.106221  1462.344538        0.676471
std         34.765316       1.078442   574.607699        0.468809
min        128.000000       1.786050   527.000000        0.000000
25%        256.000000       3.246863  1066.500000        0.000000
50%        256.000000       4.106221  1333.000000        1.000000
75%        256.000000       4.851000  1777.000000        1.000000
max        256.000000       7.938000  3810.000000        1.000000

       Screen-IPS_panel
count        238.000000
mean           0.323529
std            0.468809
min            0.000000
25%            0.000000
50%            0.000000
75%            1.000000
max            1.000000
       Manufacturer Price-binned
count           238          238
unique           11            3
top            Dell          Low
freq             71          160
```

Task 3 - GroupBy and Pivot Tables

Group the parameters "GPU", "CPU_core" and "Price" to make a pivot table and visualize this connection using the pcolor plot.

```
# Write your code below and press Shift+Enter to execute
# Create the group
df_gptest = df[['GPU','CPU_core','Price']]
grouped_test1 =
df_gptest.groupby(['GPU','CPU_core'],as_index=False).mean()
print(grouped_test1)

   GPU  CPU_core        Price
0    1         3   769.250000
1    1         5   998.500000
2    1         7  1167.941176
```

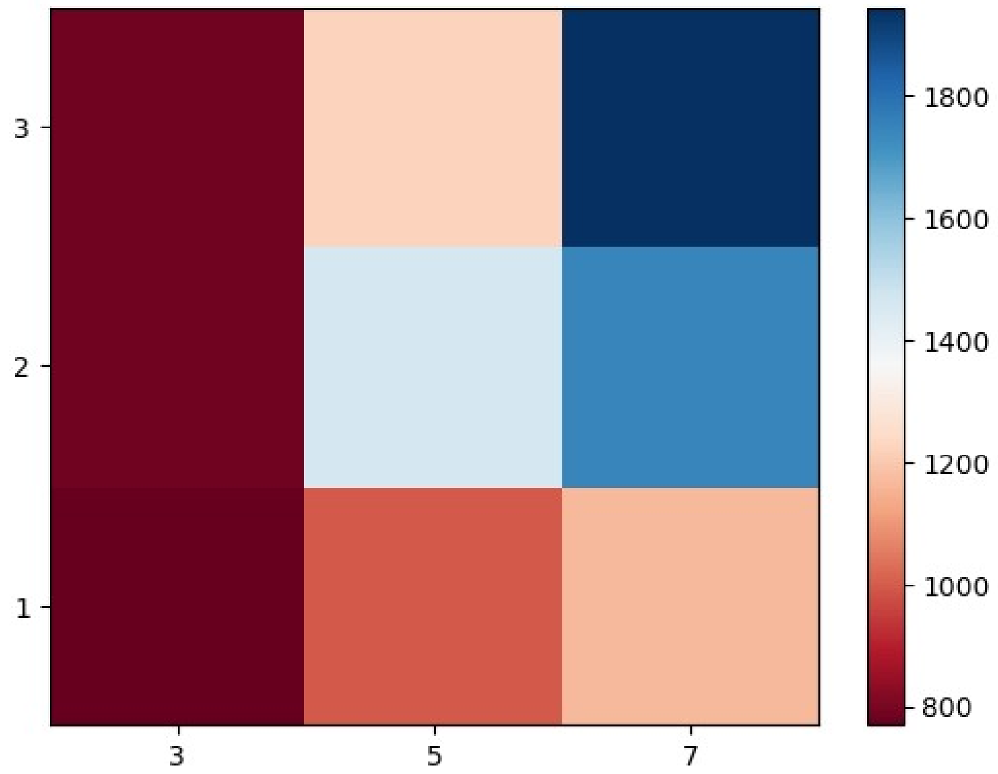
```
3    2          3   785.076923
4    2          5  1462.197674
5    2          7  1744.621622
6    3          3   784.000000
7    3          5  1220.680000
8    3          7  1945.097561
```

```
# Write your code below and press Shift+Enter to execute
# Create the Pivot table
grouped_pivot = grouped_test1.pivot(index='GPU',columns='CPU_core')
print(grouped_pivot)
```

```
               Price
CPU_core           3            5            7
GPU
1         769.250000   998.500000  1167.941176
2         785.076923  1462.197674  1744.621622
3         784.000000  1220.680000  1945.097561
```

```
# Write your code below and press Shift+Enter to execute
# Create the Plot
fig, ax = plt.subplots()
im = ax.pcolor(grouped_pivot, cmap='RdBu')

#label names
row_labels = grouped_pivot.columns.levels[1]
col_labels = grouped_pivot.index

#move ticks and labels to the center
ax.set_xticks(np.arange(grouped_pivot.shape[1]) + 0.5, minor=False)
ax.set_yticks(np.arange(grouped_pivot.shape[0]) + 0.5, minor=False)

#insert labels
ax.set_xticklabels(row_labels, minor=False)
ax.set_yticklabels(col_labels, minor=False)

fig.colorbar(im)
```

```
<matplotlib.colorbar.Colorbar at 0xa2147e0>
```



Task 4 - Pearson Correlation and p-values

Use the `scipy.stats.pearsonr()` function to evaluate the Pearson Coefficient and the p-values for each parameter tested above. This will help you determine the parameters most likely to have a strong effect on the price of the laptops.

```
# Write your code below and press Shift+Enter to execute
for param in
['RAM_GB','CPU_frequency','Storage_GB_SSD','Screen_Size_inch','Weight_
pounds','CPU_core','OS','GPU','Category']:
    pearson_coef, p_value = stats.pearsonr(df[param], df['Price'])
    print(param)
    print("The Pearson Correlation Coefficient for ",param," is",
pearson_coef, " with a P-value of P =", p_value)
```

```
RAM_GB
The Pearson Correlation Coefficient for  RAM_GB  is 0.5492972971857849
with a P-value of P = 3.6815606288424503e-20
CPU_frequency
The Pearson Correlation Coefficient for  CPU_frequency  is
0.3666655589258861  with a P-value of P = 5.50246335071342e-09
Storage_GB_SSD
The Pearson Correlation Coefficient for  Storage_GB_SSD  is
0.24342075521810297  with a P-value of P = 0.00014898923191724168
```

```
Screen_Size_inch
The Pearson Correlation Coefficient for  Screen_Size_inch  is -
0.11064420817118291  with a P-value of P = 0.08853397846830661
Weight_pounds
The Pearson Correlation Coefficient for  Weight_pounds  is -
0.050312258377515455  with a P-value of P = 0.4397693853433894
CPU_core
The Pearson Correlation Coefficient for  CPU_core  is
0.45939777733551174  with a P-value of P = 7.912950127008979e-14
OS
The Pearson Correlation Coefficient for  OS  is -0.22172980114827356
with a P-value of P = 0.0005696642559246817
GPU
The Pearson Correlation Coefficient for  GPU  is 0.2882981988881427
with a P-value of P = 6.166949698364507e-06
Category
The Pearson Correlation Coefficient for  Category  is
0.286242755812641  with a P-value of P = 7.225696235806858e-06
```

Congratulations! You have completed the lab

Authors

Abhishek Gagneja

Vicky Kuo

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2023-09-15	0.1	Abhishek Gagneja	Initial Version Created
2023-09-18	0.2	Vicky Kuo	Reviewed and Revised