LAPTOP PRICE PREDICTOR

Project By : PRASAD JADHAV

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
from sklearn.model_selection import train_test_split

from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error

from sklearn.linear_model import LinearRegression,Ridge,Lasso
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor

from sklearn.ensemble import VotingRegressor
from sklearn.ensemble import StackingRegressor

from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
```

In [3]:

```
import pickle
import joblib
```

In [4]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [5]:
```
dataset = pd.read_csv('laptop_data.csv')
dataset.drop(columns=['Unnamed: 0'],inplace=True)
dataset.shape
```

Out[5]:
```
(1303, 11)
```

In [6]:
```
dataset.head()
```

Out[6]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg

In [7]:
```
dataset.tail()
```

Out[7]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	W
1298	Lenovo	2 in 1 Convertible	14.0	IPS Panel Full HD / Touchscreen 1920x1080	Intel Core i7 6500U 2.5GHz	4GB	128GB SSD	Intel HD Graphics 520	Windows 10	
1299	Lenovo	2 in 1 Convertible	13.3	IPS Panel Quad HD+ / Touchscreen 3200x1800	Intel Core i7 6500U 2.5GHz	16GB	512GB SSD	Intel HD Graphics 520	Windows 10	
1300	Lenovo	Notebook	14.0	1366x768	Intel Celeron Dual Core N3050 1.6GHz	2GB	64GB Flash Storage	Intel HD Graphics	Windows 10	
1301	HP	Notebook	15.6	1366x768	Intel Core i7 6500U 2.5GHz	6GB	1TB HDD	AMD Radeon R5 M330	Windows 10	2.
1302	Asus	Notebook	15.6	1366x768	Intel Celeron Dual Core N3050 1.6GHz	4GB	500GB HDD	Intel HD Graphics	Windows 10	2

In [8]:

```
print('Numbar of Rows:',dataset.shape[0])
print('Numbar of Columns:',dataset.shape[1])
```

```
Numbar of Rows: 1303
Numbar of Columns: 11
```

In [9]:

```
dataset.isnull().sum()
```

Out[9]:

```
Company             0
TypeName            0
Inches              0
ScreenResolution    0
Cpu                 0
Ram                 0
Memory              0
Gpu                 0
OpSys               0
Weight              0
Price               0
dtype: int64
```

In [10]:

```
dataset.isna().sum()
```

Out[10]:

```
Company             0
TypeName            0
Inches              0
ScreenResolution    0
Cpu                 0
Ram                 0
Memory              0
Gpu                 0
OpSys               0
Weight              0
Price               0
dtype: int64
```

In [11]:

```
dataset.duplicated().sum()
```

Out[11]:

```
29
```

In [12]:

```
dataset = dataset.drop_duplicates()
```

In [13]:

```
dataset.describe()
```

Out[13]:

	Inches	Price
count	1274.000000	1274.000000
mean	15.022449	60503.185074
std	1.429940	37333.222977
min	10.100000	9270.720000
25%	14.000000	32495.605200
50%	15.600000	52693.920000
75%	15.600000	79773.480000
max	18.400000	324954.720000

In [14]:

```
dataset.corr()
```

Out[14]:

	Inches	Price
Inches	1.00000	0.06699
Price	0.06699	1.00000

In [15]:

```
dataset.shape
```

Out[15]:

```
(1274, 11)
```

In [16]:

```
dataset['Company'].unique()
```

Out[16]:

```
array(['Apple', 'HP', 'Acer', 'Asus', 'Dell', 'Lenovo', 'Chuwi', 'MSI',
       'Microsoft', 'Toshiba', 'Huawei', 'Xiaomi', 'Vero', 'Razer',
       'Mediacom', 'Samsung', 'Google', 'Fujitsu', 'LG'], dtype=object)
```

In [17]:

```
dataset['TypeName'].unique()
```

Out[17]:

```
array(['Ultrabook', 'Notebook', 'Netbook', 'Gaming', '2 in 1 Convertible',
       'Workstation'], dtype=object)
```

In [18]:

```
dataset['Inches'].unique()
```

Out[18]:

```
array([13.3, 15.6, 15.4, 14. , 12. , 11.6, 17.3, 10.1, 13.5, 12.5, 13. ,
       18.4, 13.9, 12.3, 17. , 15. , 14.1, 11.3])
```

In [19]:

```
dataset['ScreenResolution'].unique()
```

Out[19]:

```
array(['IPS Panel Retina Display 2560x1600', '1440x900',
       'Full HD 1920x1080', 'IPS Panel Retina Display 2880x1800',
       '1366x768', 'IPS Panel Full HD 1920x1080',
       'IPS Panel Retina Display 2304x1440',
       'IPS Panel Full HD / Touchscreen 1920x1080',
       'Full HD / Touchscreen 1920x1080',
       'Touchscreen / Quad HD+ 3200x1800',
       'IPS Panel Touchscreen 1920x1200', 'Touchscreen 2256x1504',
       'Quad HD+ / Touchscreen 3200x1800', 'IPS Panel 1366x768',
       'IPS Panel 4K Ultra HD / Touchscreen 3840x2160',
       'IPS Panel Full HD 2160x1440',
       '4K Ultra HD / Touchscreen 3840x2160', 'Touchscreen 2560x1440',
       '1600x900', 'IPS Panel 4K Ultra HD 3840x2160',
       '4K Ultra HD 3840x2160', 'Touchscreen 1366x768',
       'IPS Panel Full HD 1366x768', 'IPS Panel 2560x1440',
       'IPS Panel Full HD 2560x1440',
       'IPS Panel Retina Display 2736x1824', 'Touchscreen 2400x1600',
       '2560x1440', 'IPS Panel Quad HD+ 2560x1440',
       'IPS Panel Quad HD+ 3200x1800',
       'IPS Panel Quad HD+ / Touchscreen 3200x1800',
       'IPS Panel Touchscreen 1366x768', '1920x1080',
       'IPS Panel Full HD 1920x1200',
       'IPS Panel Touchscreen / 4K Ultra HD 3840x2160',
       'IPS Panel Touchscreen 2560x1440',
       'Touchscreen / Full HD 1920x1080', 'Quad HD+ 3200x1800',
       'Touchscreen / 4K Ultra HD 3840x2160',
       'IPS Panel Touchscreen 2400x1600'], dtype=object)
```

In [20]:

```
dataset['ScreenResolution'].value_counts()
```

Out[20]:

```
Full HD 1920x1080                                505
1366x768                                         262
IPS Panel Full HD 1920x1080                      226
IPS Panel Full HD / Touchscreen 1920x1080         51
Full HD / Touchscreen 1920x1080                   47
1600x900                                          23
Touchscreen 1366x768                              16
Quad HD+ / Touchscreen 3200x1800                  15
IPS Panel 4K Ultra HD 3840x2160                   12
IPS Panel 4K Ultra HD / Touchscreen 3840x2160     11
4K Ultra HD / Touchscreen 3840x2160               10
4K Ultra HD 3840x2160                              7
Touchscreen 2560x1440                              7
IPS Panel 1366x768                                 7
IPS Panel Retina Display 2560x1600                 6
IPS Panel Retina Display 2304x1440                 6
Touchscreen 2256x1504                              6
IPS Panel Touchscreen 2560x1440                    5
IPS Panel Quad HD+ / Touchscreen 3200x1800         4
IPS Panel Touchscreen 1920x1200                    4
1440x900                                           4
IPS Panel Retina Display 2880x1800                 4
IPS Panel 2560x1440                                4
2560x1440                                          3
Quad HD+ 3200x1800                                 3
1920x1080                                          3
Touchscreen 2400x1600                              3
IPS Panel Quad HD+ 2560x1440                       3
IPS Panel Touchscreen 1366x768                     3
IPS Panel Touchscreen / 4K Ultra HD 3840x2160      2
IPS Panel Full HD 2160x1440                        2
IPS Panel Quad HD+ 3200x1800                       2
IPS Panel Retina Display 2736x1824                 1
IPS Panel Full HD 1920x1200                        1
IPS Panel Full HD 2560x1440                        1
IPS Panel Full HD 1366x768                         1
Touchscreen / Full HD 1920x1080                    1
Touchscreen / Quad HD+ 3200x1800                   1
Touchscreen / 4K Ultra HD 3840x2160                1
IPS Panel Touchscreen 2400x1600                    1
Name: ScreenResolution, dtype: int64
```

In [21]:

```
dataset['Cpu'].unique()
```

```
Out[21]: array(['Intel Core i5 2.3GHz', 'Intel Core i5 1.8GHz',
                'Intel Core i5 7200U 2.5GHz', 'Intel Core i7 2.7GHz',
                'Intel Core i5 3.1GHz', 'AMD A9-Series 9420 3GHz',
                'Intel Core i7 2.2GHz', 'Intel Core i7 8550U 1.8GHz',
                'Intel Core i5 8250U 1.6GHz', 'Intel Core i3 6006U 2GHz',
                'Intel Core i7 2.8GHz', 'Intel Core M m3 1.2GHz',
                'Intel Core i7 7500U 2.7GHz', 'Intel Core i7 2.9GHz',
                'Intel Core i3 7100U 2.4GHz', 'Intel Atom x5-Z8350 1.44GHz',
                'Intel Core i5 7300HQ 2.5GHz', 'AMD E-Series E2-9000e 1.5GHz',
                'Intel Core i5 1.6GHz', 'Intel Core i7 8650U 1.9GHz',
                'Intel Atom x5-Z8300 1.44GHz', 'AMD E-Series E2-6110 1.5GHz',
                'AMD A6-Series 9220 2.5GHz',
                'Intel Celeron Dual Core N3350 1.1GHz',
                'Intel Core i3 7130U 2.7GHz', 'Intel Core i7 7700HQ 2.8GHz',
                'Intel Core i5 2.0GHz', 'AMD Ryzen 1700 3GHz',
                'Intel Pentium Quad Core N4200 1.1GHz',
                'Intel Atom x5-Z8550 1.44GHz',
                'Intel Celeron Dual Core N3060 1.6GHz', 'Intel Core i5 1.3GHz',
                'AMD FX 9830P 3GHz', 'Intel Core i7 7560U 2.4GHz',
                'AMD E-Series 6110 1.5GHz', 'Intel Core i5 6200U 2.3GHz',
                'Intel Core M 6Y75 1.2GHz', 'Intel Core i5 7500U 2.7GHz',
                'Intel Core i3 6006U 2.2GHz', 'AMD A6-Series 9220 2.9GHz',
                'Intel Core i7 6920HQ 2.9GHz', 'Intel Core i5 7Y54 1.2GHz',
                'Intel Core i7 7820HK 2.9GHz', 'Intel Xeon E3-1505M V6 3GHz',
                'Intel Core i7 6500U 2.5GHz', 'AMD E-Series 9000e 1.5GHz',
                'AMD A10-Series A10-9620P 2.5GHz', 'AMD A6-Series A6-9220 2.5GHz',
                'Intel Core i5 2.9GHz', 'Intel Core i7 6600U 2.6GHz',
                'Intel Core i3 6006U 2.0GHz',
                'Intel Celeron Dual Core 3205U 1.5GHz',
                'Intel Core i7 7820HQ 2.9GHz', 'AMD A10-Series 9600P 2.4GHz',
                'Intel Core i7 7600U 2.8GHz', 'AMD A8-Series 7410 2.2GHz',
                'Intel Celeron Dual Core 3855U 1.6GHz',
                'Intel Pentium Quad Core N3710 1.6GHz',
                'AMD A12-Series 9720P 2.7GHz', 'Intel Core i5 7300U 2.6GHz',
                'AMD A12-Series 9720P 3.6GHz',
                'Intel Celeron Quad Core N3450 1.1GHz',
                'Intel Celeron Dual Core N3060 1.60GHz',
                'Intel Core i5 6440HQ 2.6GHz', 'Intel Core i7 6820HQ 2.7GHz',
                'AMD Ryzen 1600 3.2GHz', 'Intel Core i7 7Y75 1.3GHz',
                'Intel Core i5 7440HQ 2.8GHz', 'Intel Core i7 7660U 2.5GHz',
                'Intel Core i7 7700HQ 2.7GHz', 'Intel Core M m3-7Y30 2.2GHz',
                'Intel Core i5 7Y57 1.2GHz', 'Intel Core i7 6700HQ 2.6GHz',
                'Intel Core i3 6100U 2.3GHz', 'AMD A10-Series 9620P 2.5GHz',
                'AMD E-Series 7110 1.8GHz', 'Intel Celeron Dual Core N3350 2.0GHz',
                'AMD A9-Series A9-9420 3GHz', 'Intel Core i7 6820HK 2.7GHz',
                'Intel Core M 7Y30 1.0GHz', 'Intel Xeon E3-1535M v6 3.1GHz',
                'Intel Celeron Quad Core N3160 1.6GHz',
                'Intel Core i5 6300U 2.4GHz', 'Intel Core i3 6100U 2.1GHz',
                'AMD E-Series E2-9000 2.2GHz',
                'Intel Celeron Dual Core N3050 1.6GHz',
                'Intel Core M M3-6Y30 0.9GHz', 'AMD A9-Series 9420 2.9GHz',
                'Intel Core i5 6300HQ 2.3GHz', 'AMD A6-Series 7310 2GHz',
                'Intel Atom Z8350 1.92GHz', 'Intel Xeon E3-1535M v5 2.9GHz',
                'Intel Core i5 6260U 1.8GHz',
                'Intel Pentium Dual Core N4200 1.1GHz',
                'Intel Celeron Quad Core N3710 1.6GHz', 'Intel Core M 1.2GHz',
                'AMD A12-Series 9700P 2.5GHz', 'Intel Core i7 7500U 2.5GHz',
                'Intel Pentium Dual Core 4405U 2.1GHz',
                'AMD A4-Series 7210 2.2GHz', 'Intel Core i7 6560U 2.2GHz',
                'Intel Core M m7-6Y75 1.2GHz', 'AMD FX 8800P 2.1GHz',
                'Intel Core M M7-6Y75 1.2GHz', 'Intel Core i5 7200U 2.50GHz',
                'Intel Core i5 7200U 2.70GHz', 'Intel Atom X5-Z8350 1.44GHz',
                'Intel Core i5 7200U 2.7GHz', 'Intel Core M 1.1GHz',
                'Intel Pentium Dual Core 4405Y 1.5GHz',
                'Intel Pentium Quad Core N3700 1.6GHz', 'Intel Core M 6Y54 1.1GHz',
                'Intel Core i7 6500U 2.50GHz',
                'Intel Celeron Dual Core N3350 2GHz',
                'Samsung Cortex A72&A53 2.0GHz', 'AMD E-Series 9000 2.2GHz',
                'Intel Core M 6Y30 0.9GHz', 'AMD A9-Series 9410 2.9GHz'],
               dtype=object)
```

```
In [22]: dataset['Ram'].unique()
```

```
Out[22]: array(['8GB', '16GB', '4GB', '2GB', '12GB', '6GB', '32GB', '24GB', '64GB'],
               dtype=object)
```

```
In [23]: dataset['Memory'].unique()
```

```
Out[23]: array(['128GB SSD', '128GB Flash Storage', '256GB SSD', '512GB SSD',
                '500GB HDD', '256GB Flash Storage', '1TB HDD',
                '32GB Flash Storage', '128GB SSD +  1TB HDD',
                '256GB SSD +  256GB SSD', '64GB Flash Storage',
                '256GB SSD +  1TB HDD', '256GB SSD +  2TB HDD', '32GB SSD',
                '2TB HDD', '64GB SSD', '1.0TB Hybrid', '512GB SSD +  1TB HDD',
                '1TB SSD', '256GB SSD +  500GB HDD', '128GB SSD +  2TB HDD',
                '512GB SSD +  512GB SSD', '16GB SSD', '16GB Flash Storage',
                '512GB SSD +  256GB SSD', '512GB SSD +  2TB HDD',
                '64GB Flash Storage +  1TB HDD', '180GB SSD', '1TB HDD +  1TB HDD',
                '32GB HDD', '1TB SSD +  1TB HDD', '512GB Flash Storage',
                '128GB HDD', '240GB SSD', '8GB SSD', '508GB Hybrid', '1.0TB HDD',
                '512GB SSD +  1.0TB Hybrid', '256GB SSD +  1.0TB Hybrid'],
               dtype=object)
```

```
In [24]: dataset['Gpu'].unique()
```

Out[24]:
```
array(['Intel Iris Plus Graphics 640', 'Intel HD Graphics 6000',
       'Intel HD Graphics 620', 'AMD Radeon Pro 455',
       'Intel Iris Plus Graphics 650', 'AMD Radeon R5',
       'Intel Iris Pro Graphics', 'Nvidia GeForce MX150',
       'Intel UHD Graphics 620', 'Intel HD Graphics 520',
       'AMD Radeon Pro 555', 'AMD Radeon R5 M430',
       'Intel HD Graphics 615', 'AMD Radeon Pro 560',
       'Nvidia GeForce 940MX', 'Intel HD Graphics 400',
       'Nvidia GeForce GTX 1050', 'AMD Radeon R2', 'AMD Radeon 530',
       'Nvidia GeForce 930MX', 'Intel HD Graphics',
       'Intel HD Graphics 500', 'Nvidia GeForce 930MX ',
       'Nvidia GeForce GTX 1060', 'Nvidia GeForce 150MX',
       'Intel Iris Graphics 540', 'AMD Radeon RX 580',
       'Nvidia GeForce 920MX', 'AMD Radeon R4 Graphics', 'AMD Radeon 520',
       'Nvidia GeForce GTX 1070', 'Nvidia GeForce GTX 1050 Ti',
       'Nvidia GeForce MX130', 'AMD R4 Graphics',
       'Nvidia GeForce GTX 940MX', 'AMD Radeon RX 560',
       'Nvidia GeForce 920M', 'AMD Radeon R7 M445', 'AMD Radeon RX 550',
       'Nvidia GeForce GTX 1050M', 'Intel HD Graphics 515',
       'AMD Radeon R5 M420', 'Intel HD Graphics 505',
       'Nvidia GTX 980 SLI', 'AMD R17M-M1-70', 'Nvidia GeForce GTX 1080',
       'Nvidia Quadro M1200', 'Nvidia GeForce 920MX ',
       'Nvidia GeForce GTX 950M', 'AMD FirePro W4190M ',
       'Nvidia GeForce GTX 980M', 'Intel Iris Graphics 550',
       'Nvidia GeForce 930M', 'Intel HD Graphics 630',
       'AMD Radeon R5 430', 'Nvidia GeForce GTX 940M',
       'Intel HD Graphics 510', 'Intel HD Graphics 405',
       'AMD Radeon RX 540', 'Nvidia GeForce GT 940MX',
       'AMD FirePro W5130M', 'Nvidia Quadro M2200M', 'AMD Radeon R4',
       'Nvidia Quadro M620', 'AMD Radeon R7 M460',
       'Intel HD Graphics 530', 'Nvidia GeForce GTX 965M',
       'Nvidia GeForce GTX1080', 'Nvidia GeForce GTX1050 Ti',
       'Nvidia GeForce GTX 960M', 'AMD Radeon R2 Graphics',
       'Nvidia Quadro M620M', 'Nvidia GeForce GTX 970M',
       'Nvidia GeForce GTX 960<U+039C>', 'Intel Graphics 620',
       'Nvidia GeForce GTX 960', 'AMD Radeon R5 520',
       'AMD Radeon R7 M440', 'AMD Radeon R7', 'Nvidia Quadro M520M',
       'Nvidia Quadro M2200', 'Nvidia Quadro M2000M',
       'Intel HD Graphics 540', 'Nvidia Quadro M1000M', 'AMD Radeon 540',
       'Nvidia GeForce GTX 1070M', 'Nvidia GeForce GTX1060',
       'Intel HD Graphics 5300', 'AMD Radeon R5 M420X',
       'AMD Radeon R7 Graphics', 'Nvidia GeForce 920',
       'Nvidia GeForce 940M', 'Nvidia GeForce GTX 930MX',
       'AMD Radeon R7 M465', 'AMD Radeon R3', 'Nvidia GeForce GTX 1050Ti',
       'AMD Radeon R7 M365X', 'AMD Radeon R9 M385',
       'Intel HD Graphics 620 ', 'Nvidia Quadro 3000M',
       'Nvidia GeForce GTX 980 ', 'AMD Radeon R5 M330',
       'AMD FirePro W4190M', 'AMD FirePro W6150M', 'AMD Radeon R5 M315',
       'Nvidia Quadro M500M', 'AMD Radeon R7 M360',
       'Nvidia Quadro M3000M', 'Nvidia GeForce 960M', 'ARM Mali T860 MP4'],
      dtype=object)
```

In [25]:
```
dataset['OpSys'].unique()
```

Out[25]:
```
array(['macOS', 'No OS', 'Windows 10', 'Mac OS X', 'Linux', 'Android',
       'Windows 10 S', 'Chrome OS', 'Windows 7'], dtype=object)
```

In [26]:
```
dataset['Weight'].unique()
```

Out[26]:

```
array(['1.37kg', '1.34kg', '1.86kg', '1.83kg', '2.1kg', '2.04kg', '1.3kg',
       '1.6kg', '2.2kg', '0.92kg', '1.22kg', '0.98kg', '2.5kg', '1.62kg',
       '1.91kg', '2.3kg', '1.35kg', '1.88kg', '1.89kg', '1.65kg',
       '2.71kg', '1.2kg', '1.44kg', '2.8kg', '2kg', '2.65kg', '2.77kg',
       '3.2kg', '0.69kg', '1.49kg', '2.4kg', '2.13kg', '2.43kg', '1.7kg',
       '1.4kg', '1.8kg', '1.9kg', '3kg', '1.252kg', '2.7kg', '2.02kg',
       '1.63kg', '1.96kg', '1.21kg', '2.45kg', '1.25kg', '1.5kg',
       '2.62kg', '1.38kg', '1.58kg', '1.85kg', '1.23kg', '1.26kg',
       '2.16kg', '2.36kg', '2.05kg', '1.32kg', '1.75kg', '0.97kg',
       '2.9kg', '2.56kg', '1.48kg', '1.74kg', '1.1kg', '1.56kg', '2.03kg',
       '1.05kg', '4.4kg', '1.90kg', '1.29kg', '2.0kg', '1.95kg', '2.06kg',
       '1.12kg', '1.42kg', '3.49kg', '3.35kg', '2.23kg', '4.42kg',
       '2.69kg', '2.37kg', '4.7kg', '3.6kg', '2.08kg', '4.3kg', '1.68kg',
       '1.41kg', '4.14kg', '2.18kg', '2.24kg', '2.67kg', '2.14kg',
       '1.36kg', '2.25kg', '2.15kg', '2.19kg', '2.54kg', '3.42kg',
       '1.28kg', '2.33kg', '1.45kg', '2.79kg', '1.84kg', '2.6kg',
       '2.26kg', '3.25kg', '1.59kg', '1.13kg', '1.78kg', '1.10kg',
       '1.15kg', '1.27kg', '1.43kg', '2.31kg', '1.16kg', '1.64kg',
       '2.17kg', '1.47kg', '3.78kg', '1.79kg', '0.91kg', '1.99kg',
       '4.33kg', '1.93kg', '1.87kg', '2.63kg', '3.4kg', '3.14kg',
       '1.94kg', '1.24kg', '4.6kg', '4.5kg', '2.73kg', '1.39kg', '2.29kg',
       '2.59kg', '2.94kg', '1.14kg', '3.8kg', '3.31kg', '1.09kg',
       '3.21kg', '1.19kg', '1.98kg', '1.17kg', '4.36kg', '1.71kg',
       '2.32kg', '4.2kg', '1.55kg', '0.81kg', '1.18kg', '2.72kg',
       '1.31kg', '0.920kg', '3.74kg', '1.76kg', '1.54kg', '2.83kg',
       '2.07kg', '2.38kg', '3.58kg', '1.08kg', '2.20kg', '2.75kg',
       '1.70kg', '2.99kg', '1.11kg', '2.09kg', '4kg', '3.0kg', '0.99kg',
       '3.52kg', '2.591kg', '2.21kg', '3.3kg', '2.191kg', '2.34kg',
       '4.0kg'], dtype=object)
```

In [27]:

```
dataset['Ram'] = dataset['Ram'].str.replace('GB','')
dataset['Weight'] = dataset['Weight'].str.replace('kg','')
```

In [28]:

```
dataset['Ram'] = dataset['Ram'].astype('int32')
dataset['Weight'] = dataset['Weight'].astype('float32')
```

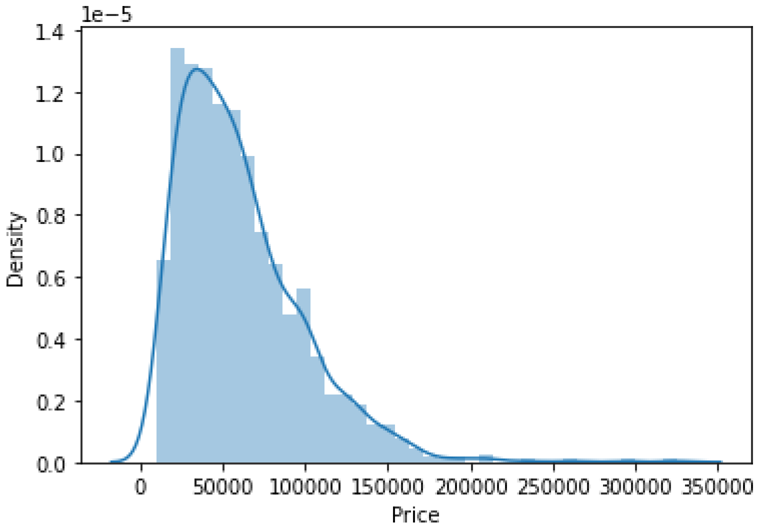
In [29]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1274 entries, 0 to 1273
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1274 non-null   object 
 1   TypeName          1274 non-null   object 
 2   Inches            1274 non-null   float64
 3   ScreenResolution  1274 non-null   object 
 4   Cpu               1274 non-null   object 
 5   Ram               1274 non-null   int32  
 6   Memory            1274 non-null   object 
 7   Gpu               1274 non-null   object 
 8   OpSys             1274 non-null   object 
 9   Weight            1274 non-null   float32
 10  Price             1274 non-null   float64
dtypes: float32(1), float64(2), int32(1), object(7)
memory usage: 109.5+ KB
```

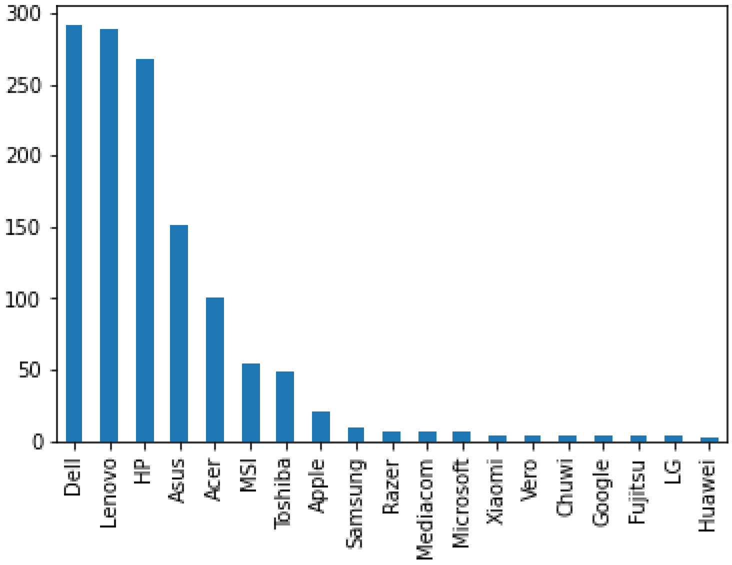
In [30]:

```
sns.distplot(dataset['Price'])
plt.show()
```



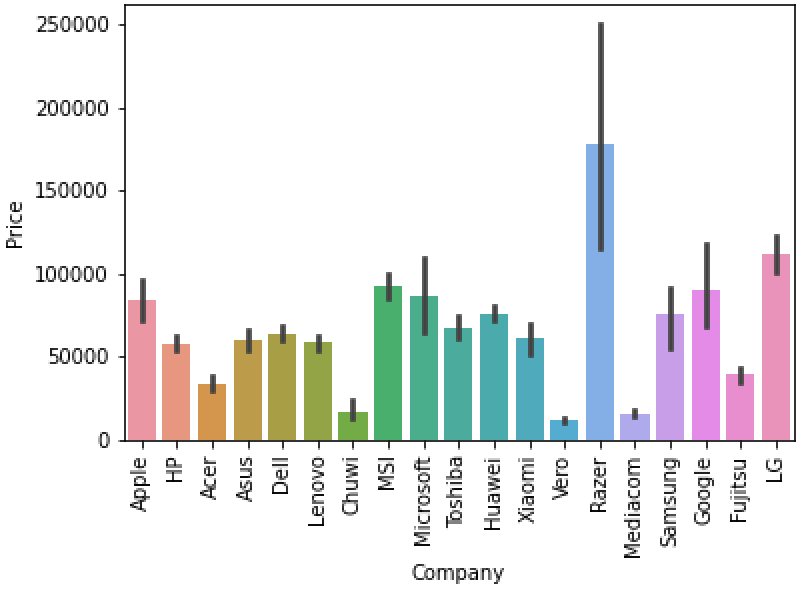
In [31]:

```
dataset['Company'].value_counts().plot(kind='bar')
plt.show()
```



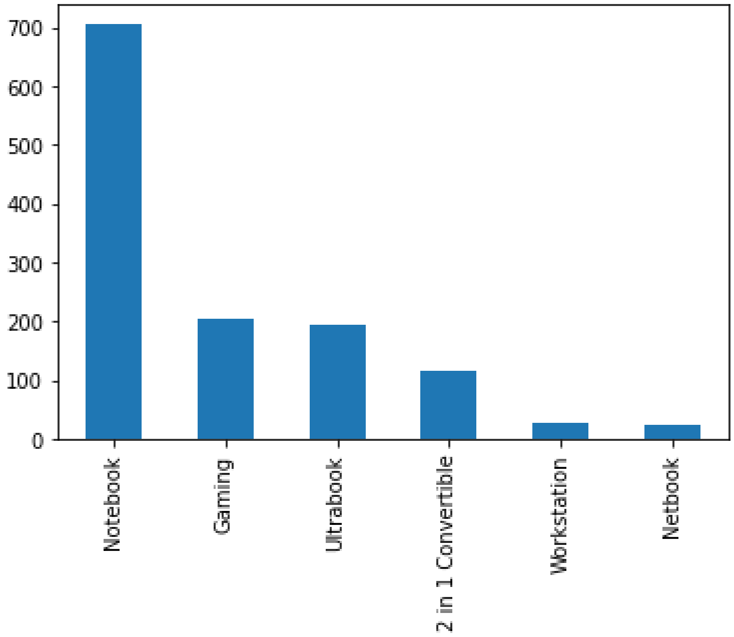
In [32]:

```
sns.barplot(dataset['Company'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



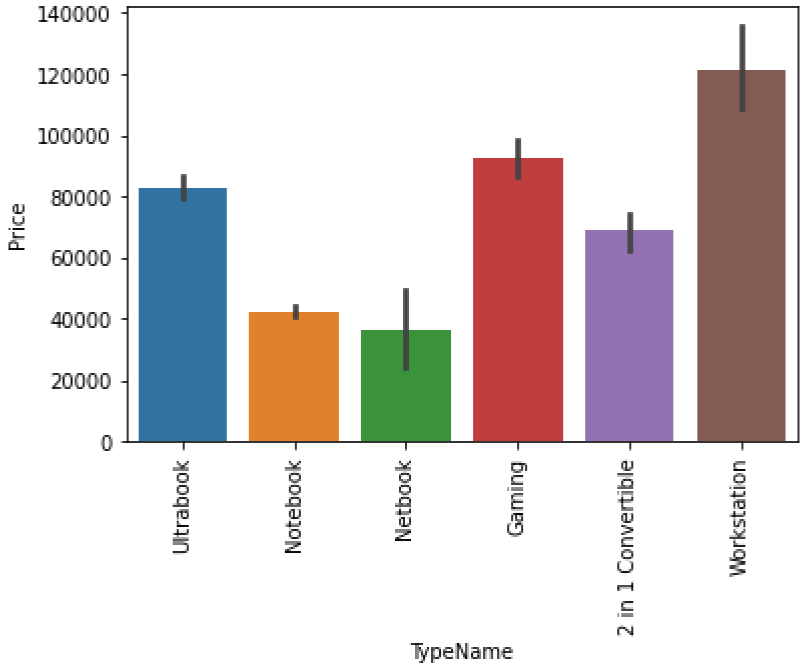
In [33]:

```
dataset['TypeName'].value_counts().plot(kind='bar')
plt.show()
```

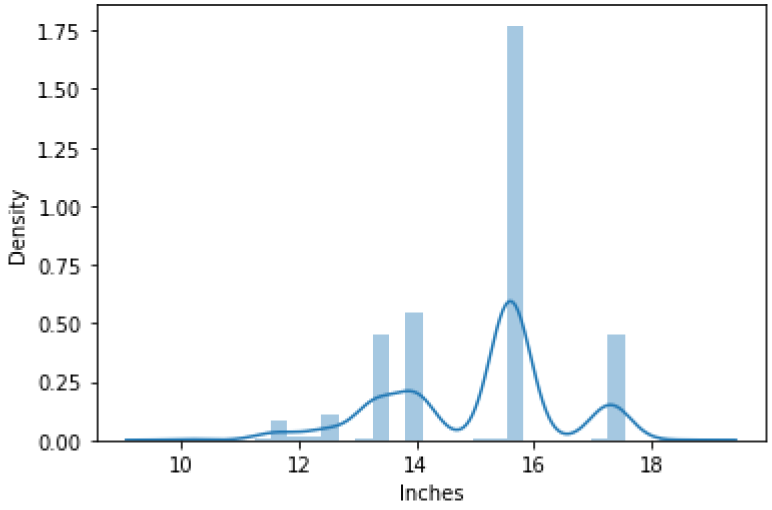


In [34]:

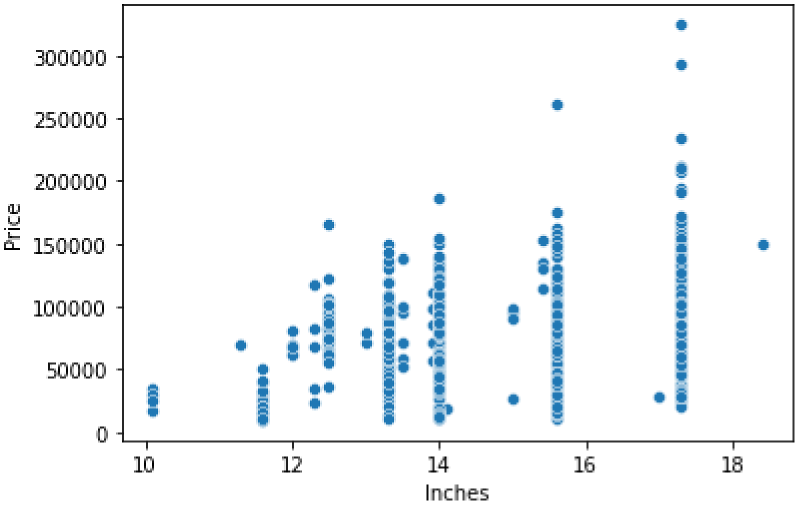
```
sns.barplot(dataset['TypeName'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



In [35]:
```
sns.distplot(dataset['Inches'])
plt.show()
```



In [36]:
```
sns.scatterplot(dataset['Inches'],dataset['Price'])
plt.show()
```



```
In [37]: dataset['Touchscreen'] = dataset['ScreenResolution'].apply(lambda x:1 if 'Touc
```

```
In [38]: dataset.sample(11)
```

Out[38]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	We
71	Dell	Ultrabook	13.3	IPS Panel Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	8	256GB SSD	AMD Radeon 530	Windows 10	
335	HP	Notebook	14.0	Full HD 1920x1080	Intel Core i5 7300U 2.6GHz	8	256GB SSD	Intel HD Graphics 620	Windows 10	
513	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	16	256GB SSD + 2TB HDD	AMD Radeon 530	Windows 10	
1209	Asus	Gaming	15.6	Full HD 1920x1080	Intel Core i7 7700HQ 2.8GHz	16	256GB SSD + 1TB HDD	Nvidia GeForce GTX 1070	Windows 10	
356	Lenovo	Notebook	15.6	1366x768	Intel Celeron Dual Core N3350 1.1GHz	4	1TB HDD	Intel HD Graphics 500	No OS	
329	Dell	Notebook	15.6	4K Ultra HD / Touchscreen 3840x2160	Intel Core i7 7700HQ 2.8GHz	32	1TB SSD	Nvidia GeForce GTX 1050	Windows 10	
37	Dell	Notebook	17.3	IPS Panel Full HD 1920x1080	Intel Core i5 8250U 1.6GHz	8	128GB SSD + 1TB HDD	AMD Radeon 530	Windows 10	
1256	Asus	Gaming	17.3	IPS Panel Full HD 1920x1080	Intel Core i7 6700HQ 2.6GHz	16	128GB SSD + 1TB HDD	Nvidia GeForce GTX 970M	Windows 10	
24	HP	Ultrabook	15.6	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	8	256GB SSD	Intel HD Graphics 620	Windows 10	
1117	Razer	Ultrabook	12.5	Touchscreen / 4K Ultra HD 3840x2160	Intel Core i7 6500U 2.5GHz	8	256GB SSD	Intel HD Graphics 520	Windows 10	
1250	Dell	Notebook	15.6	1366x768	Intel Pentium Quad Core N3710 1.6GHz	4	500GB HDD	Intel HD Graphics	Linux	

In [39]:

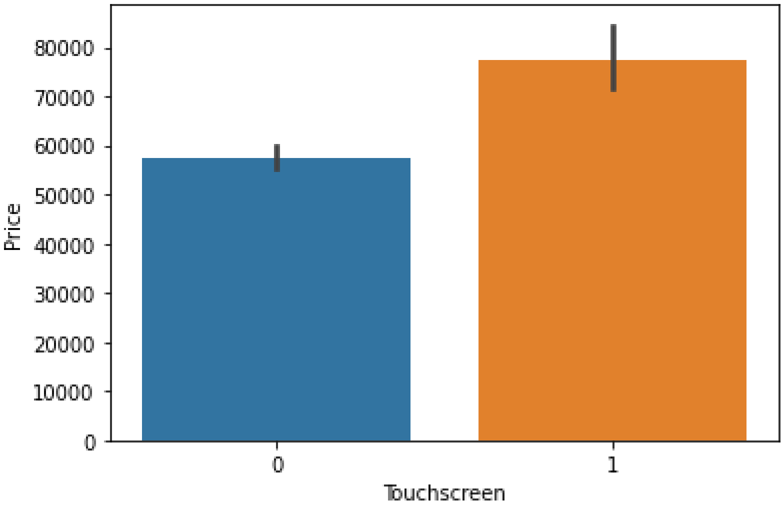
```
dataset['Touchscreen'].value_counts()
```

Out[39]:

```
0    1086
1     188
Name: Touchscreen, dtype: int64
```

In [40]:

```
sns.barplot(dataset['Touchscreen'],dataset['Price'])
plt.show()
```



In [41]:

```
dataset['Ips'] = dataset['ScreenResolution'].apply(lambda x:1 if 'IPS' in x el
```

In [42]:

```
dataset.sample(11)
```

Out[42]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	We
87	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	Windows 10	
197	HP	Notebook	13.3	Full HD 1920x1080	Intel Core i5 8250U 1.6GHz	8	512GB SSD	Intel UHD Graphics 620	Windows 10	
696	Lenovo	2 in 1 Convertible	14.0	Full HD / Touchscreen 1920x1080	Intel Core i7 7500U 2.7GHz	16	512GB SSD	Intel HD Graphics 620	Windows 10	
88	Asus	Gaming	15.6	IPS Panel Full HD 1920x1080	Intel Core i7 7700HQ 2.8GHz	16	128GB SSD + 1TB HDD	Nvidia GeForce GTX 1060	Windows 10	
61	Dell	Ultrabook	14.0	Full HD 1920x1080	Intel Core i5 8250U 1.6GHz	8	256GB SSD	Intel UHD Graphics 620	Windows 10	
1208	Acer	Notebook	17.3	1600x900	Intel Core i3 6006U 2.0GHz	8	1TB HDD	Nvidia GeForce 940MX	Windows 10	
1187	Acer	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	4	256GB SSD	Nvidia GeForce 940MX	Windows 10	
865	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 6200U 2.3GHz	4	500GB HDD	Intel HD Graphics 520	Windows 10	
440	Lenovo	Notebook	17.3	1600x900	AMD A6-Series 9220 2.5GHz	8	1TB HDD	AMD Radeon R4	Windows 10	
992	Lenovo	Notebook	15.6	1366x768	Intel Celeron Dual Core N3350 1.1GHz	4	128GB SSD	Intel HD Graphics 500	No OS	
271	Asus	Gaming	17.3	Full HD 1920x1080	AMD Ryzen 1700 3GHz	16	256GB SSD + 1TB HDD	AMD Radeon RX 580	Windows 10	

In [43]:

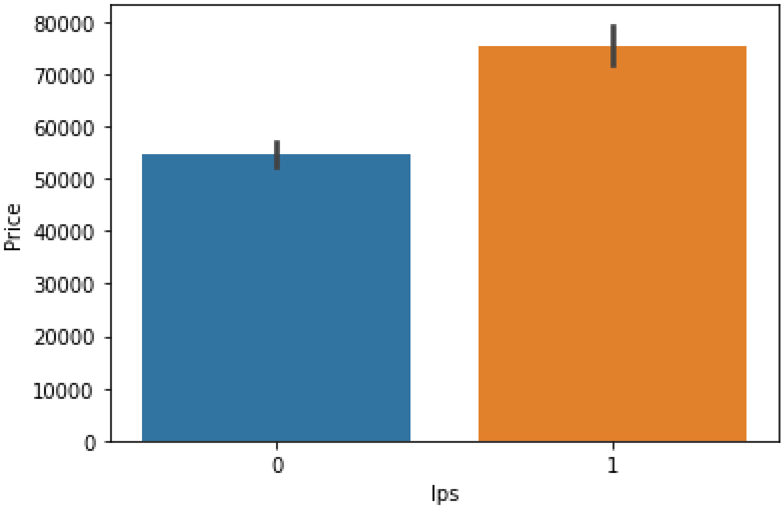
```
dataset['Ips'].value_counts()
```

Out[43]:

```
0    917
1    357
Name: Ips, dtype: int64
```

In [44]:

```
sns.barplot(dataset['Ips'],dataset['Price'])
plt.show()
```



In [45]:

```
dataset_ = dataset['ScreenResolution'].str.split('x',n=1,expand=True)
```

In [46]:

```
dataset['X_Res'] = dataset_[0]
dataset['Y_Res'] = dataset_[1]
```

In [47]:

```
dataset.head(11)
```

Out[47]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weigh
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37
5	Acer	Notebook	15.6	1366x768	AMD A9-Series 9420 3GHz	4	500GB HDD	AMD Radeon R5	Windows 10	2.10
6	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.2GHz	16	256GB Flash Storage	Intel Iris Pro Graphics	Mac OS X	2.04
7	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	256GB Flash Storage	Intel HD Graphics 6000	macOS	1.34
8	Asus	Ultrabook	14.0	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	16	512GB SSD	Nvidia GeForce MX150	Windows 10	1.30
9	Acer	Ultrabook	14.0	IPS Panel Full HD 1920x1080	Intel Core i5 8250U 1.6GHz	8	256GB SSD	Intel UHD Graphics 620	Windows 10	1.60
10	HP	Notebook	15.6	1366x768	Intel Core i5 7200U 2.5GHz	4	500GB HDD	Intel HD Graphics 620	No OS	1.86

In [48]:

```
dataset['X_Res'] = dataset['X_Res'].str.replace(',','').str.findall(r'(\d+\.?\
```

In [49]:

```
dataset.sample(5)
```

Out[49]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	We
94	Asus	Ultrabook	14.0	Full HD 1920x1080	Intel Core i7 7500U 2.7GHz	8	256GB SSD	Intel HD Graphics 620	Windows 10	
130	Dell	Notebook	15.6	1366x768	Intel Core i5 7200U 2.5GHz	8	1TB HDD	AMD Radeon R7 M445	Windows 10	
936	Dell	Notebook	15.6	1366x768	Intel Core i3 6006U 2.0GHz	4	1TB HDD	Intel HD Graphics 520	Windows 10	
801	Asus	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	Windows 10	
451	Dell	Workstation	15.6	IPS Panel Full HD 1920x1080	Intel Core i7 6820HQ 2.7GHz	16	256GB SSD	Nvidia Quadro M620	Windows 10	

In [50]:

```
dataset['X_Res'] = dataset['X_Res'].astype('int')
dataset['Y_Res'] = dataset['Y_Res'].astype('int')
```

In [51]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1274 entries, 0 to 1273
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1274 non-null   object 
 1   TypeName          1274 non-null   object 
 2   Inches            1274 non-null   float64
 3   ScreenResolution  1274 non-null   object 
 4   Cpu               1274 non-null   object 
 5   Ram               1274 non-null   int32  
 6   Memory            1274 non-null   object 
 7   Gpu               1274 non-null   object 
 8   OpSys             1274 non-null   object 
 9   Weight            1274 non-null   float32
 10  Price             1274 non-null   float64
 11  Touchscreen       1274 non-null   int64  
 12  Ips               1274 non-null   int64  
 13  X_Res             1274 non-null   int32  
 14  Y_Res             1274 non-null   int32  
dtypes: float32(1), float64(2), int32(3), int64(2), object(7)
memory usage: 139.3+ KB
```

In [52]:

```
dataset.corr()['Price']
```

Out[52]:

```
Inches         0.066990
Ram            0.740106
Weight         0.212192
Price          1.000000
Touchscreen    0.188631
Ips            0.250358
X_Res          0.552074
Y_Res          0.548111
Name: Price, dtype: float64
```

In [53]:

```
dataset['Ppi'] = (((dataset['X_Res']**2) + (dataset['Y_Res']**2))**0.5/dataset
```

In [54]:

```
dataset.corr()['Price']
```

Out[54]:

```
Inches         0.066990
Ram            0.740106
Weight         0.212192
Price          1.000000
Touchscreen    0.188631
Ips            0.250358
X_Res          0.552074
Y_Res          0.548111
Ppi            0.469539
Name: Price, dtype: float64
```

In [55]:

```
dataset.drop(columns=['ScreenResolution'],inplace=True)
```

In [56]:

```
dataset.drop(columns=['Inches','X_Res','Y_Res'],inplace=True)
```

In [57]:

```
dataset.head()
```

Out[57]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0
3	Apple	Ultrabook	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360	0
4	Apple	Ultrabook	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0

In [58]:

```
dataset['Cpu'].value_counts()
```

Out[58]:

```
Intel Core i5 7200U 2.5GHz     190
Intel Core i7 7700HQ 2.8GHz    146
Intel Core i7 7500U 2.7GHz     132
Intel Core i7 8550U 1.8GHz      73
Intel Core i5 8250U 1.6GHz      72
                              ...
Intel Core M M3-6Y30 0.9GHz      1
AMD A9-Series 9420 2.9GHz        1
Intel Core i5 2.9GHz             1
AMD A6-Series 7310 2GHz          1
AMD A9-Series 9410 2.9GHz        1
Name: Cpu, Length: 118, dtype: int64
```

In [59]:

```
dataset['CpuName'] = dataset['Cpu'].apply(lambda x:' '.join(x.split()[0:3]))
```

In [60]:

```
dataset.head(1)
```

Out[60]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0

In [61]:

```
def Fetch_Processor(text):

    if text == 'Intel Core i7' or text == 'Intel Core i5' or text == 'Intel Co
        return text
    else:
        if text.split()[0] == 'Intel':
            return 'Other Intel Processor'
        else:
            return 'AMD Processor'
```

In [62]:

```
dataset['CpuBrand'] = dataset['CpuName'].apply(Fetch_Processor)
```

In [63]:

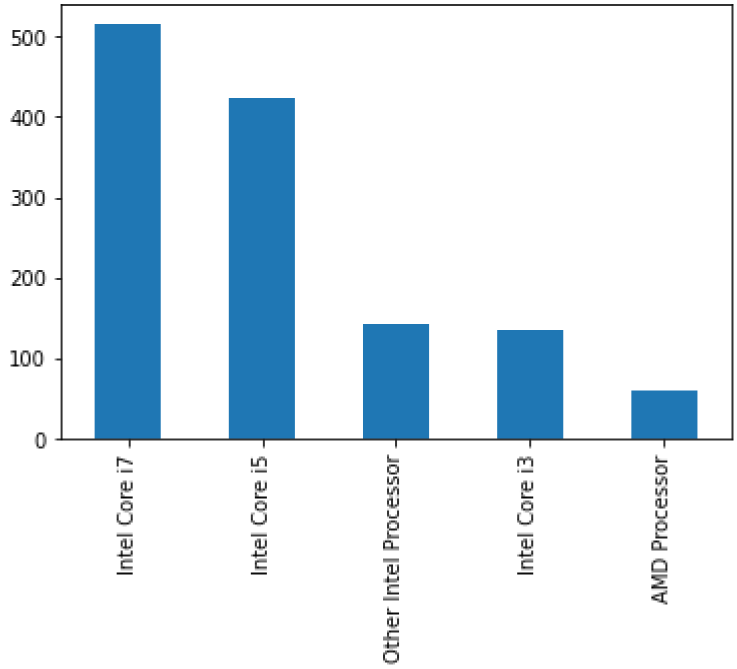
```
dataset.head()
```

Out[63]:

	Company	TypeName	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen
0	Apple	Ultrabook	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0
1	Apple	Ultrabook	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0
2	HP	Notebook	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0
3	Apple	Ultrabook	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360	0
4	Apple	Ultrabook	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0

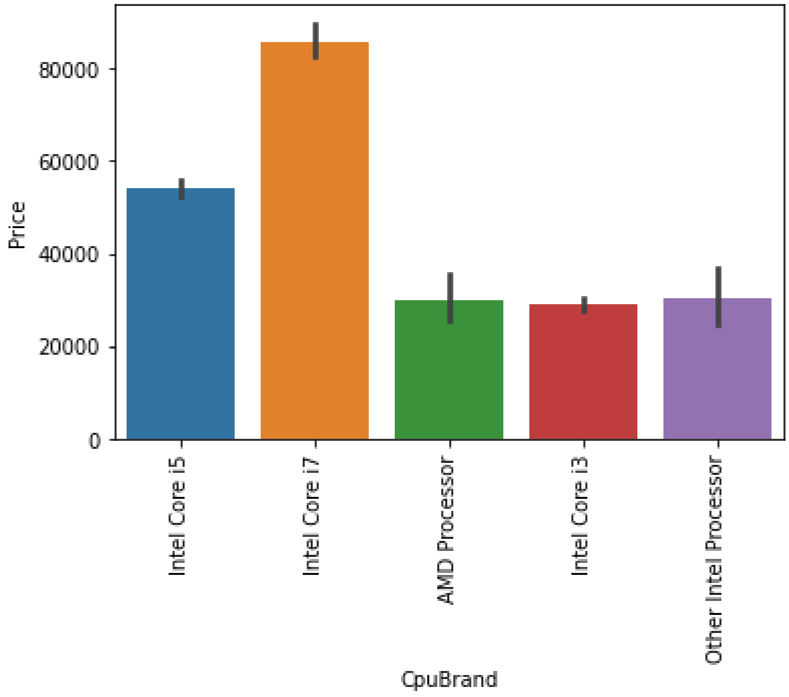
In [64]:

```
dataset['CpuBrand'].value_counts().plot(kind='bar')
plt.show()
```



In [65]:

```
sns.barplot(dataset['CpuBrand'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



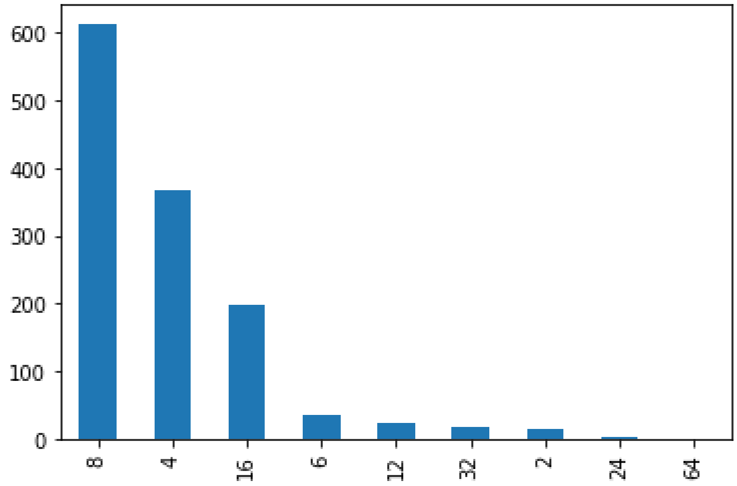
```
In [66]: dataset.drop(columns=['Cpu','CpuName'],inplace=True)
```

```
In [67]: dataset.head()
```

Out[67]:

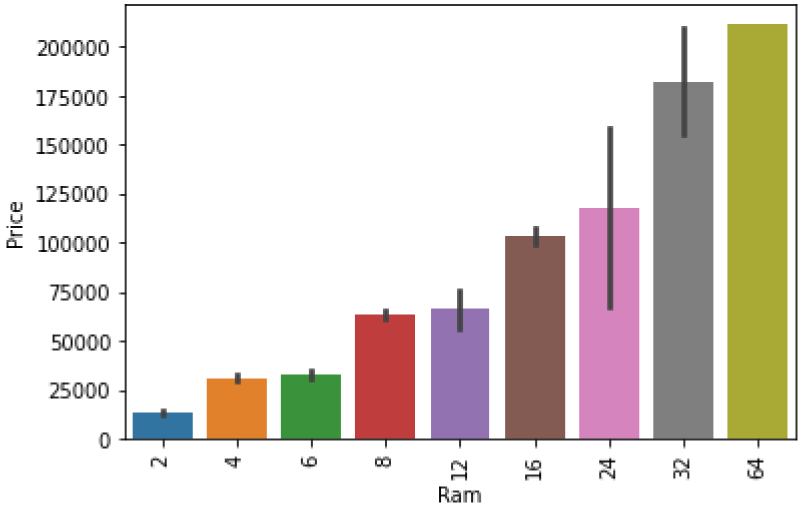
	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price	Touchscreen	Ips	
0	Apple	Ultrabook	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	22
1	Apple	Ultrabook	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	12
2	HP	Notebook	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	14
3	Apple	Ultrabook	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	1	22
4	Apple	Ultrabook	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0	1	22

```
In [68]: dataset['Ram'].value_counts().plot(kind='bar')
         plt.show()
```



In [69]:

```
sns.barplot(dataset['Ram'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



In [70]:

```
dataset['Memory'].value_counts()
```

Out[70]:

```
256GB SSD                        412
1TB HDD                          215
500GB HDD                        123
512GB SSD                        114
128GB SSD +  1TB HDD              94
128GB SSD                         74
256GB SSD +  1TB HDD              73
32GB Flash Storage                36
2TB HDD                           16
512GB SSD +  1TB HDD              14
1TB SSD                           14
64GB Flash Storage                13
256GB SSD +  2TB HDD              10
256GB Flash Storage                8
16GB Flash Storage                 7
1.0TB Hybrid                       7
32GB SSD                           6
180GB SSD                          5
128GB Flash Storage                4
512GB SSD +  2TB HDD               3
16GB SSD                           3
512GB Flash Storage                2
1TB SSD +  1TB HDD                 2
256GB SSD +  500GB HDD             2
128GB SSD +  2TB HDD               2
256GB SSD +  256GB SSD             2
512GB SSD +  256GB SSD             1
512GB SSD +  512GB SSD             1
64GB Flash Storage +  1TB HDD      1
1TB HDD +  1TB HDD                 1
32GB HDD                           1
64GB SSD                           1
128GB HDD                          1
240GB SSD                          1
8GB SSD                            1
508GB Hybrid                       1
1.0TB HDD                          1
512GB SSD +  1.0TB Hybrid          1
256GB SSD +  1.0TB Hybrid          1
Name: Memory, dtype: int64
```

In [71]:

```
dataset['Memory'] = dataset['Memory'].astype(str).replace('\.0', '', regex=Tru
dataset['Memory'] = dataset['Memory'].str.replace('GB', '')
dataset['Memory'] = dataset['Memory'].str.replace('TB', '000')
df_new = dataset['Memory'].str.split('+',n=1,expand=True)

dataset['first'] = df_new[0]
dataset['first'] = dataset['first'].str.strip()

dataset['second'] = df_new[1]

dataset['Layer1HDD'] = dataset['first'].apply(lambda x: 1 if 'HDD' in x else 0
dataset['Layer1SSD'] = dataset['first'].apply(lambda x: 1 if 'SSD' in x else 0
dataset['Layer1Hybrid'] = dataset['first'].apply(lambda x: 1 if 'Hybrid' in x
dataset['Layer1Flash_Storage'] = dataset['first'].apply(lambda x: 1 if 'Flash

dataset['first'] = dataset['first'].str.replace(r'\D', '')

dataset['second'].fillna("0", inplace = True)
```

```
dataset['Layer2HDD'] = dataset['second'].apply(lambda x: 1 if 'HDD' in x else (
dataset['Layer2SSD'] = dataset['second'].apply(lambda x: 1 if 'SSD' in x else (
dataset['Layer2Hybrid'] = dataset['second'].apply(lambda x: 1 if 'Hybrid' in x
dataset['Layer2Flash_Storage'] = dataset['second'].apply(lambda x: 1 if 'Flash

dataset['second'] = dataset['second'].str.replace(r'\D', '')

dataset['first'] = dataset['first'].astype(int)
dataset['second'] = dataset['second'].astype(int)

dataset['HDD'] = (dataset['first']*dataset['Layer1HDD']+dataset['second']*datas
dataset['SSD'] = (dataset['first']*dataset['Layer1SSD']+dataset['second']*datas
dataset['Hybrid'] = (dataset['first']*dataset['Layer1Hybrid']+dataset['second']
dataset['Flash_Storage'] = (dataset['first']*dataset['Layer1Flash_Storage']+da

dataset.drop(columns=['first','second','Layer1HDD','Layer1SSD','Layer1Hybrid',
       'Layer1Flash_Storage','Layer2HDD','Layer2SSD','Layer2Hybrid',
       'Layer2Flash_Storage'],inplace=True)
```

In [72]:
```
dataset.drop(columns=['Memory'],inplace=True)
```

In [73]:
```
dataset.sample()
```

Out[73]:

	Company	TypeName	Ram	Gpu	OpSys	Weight	Price	Touchscreen	Ips	Pp
448	MSI	Gaming	8	Nvidia GeForce GTX 1050	Windows 10	2.2	54757.9872	0	0	141.21199

In [74]:
```
dataset.corr()['Price']
```

Out[74]:
```
Ram              0.740106
Weight           0.212192
Price            1.000000
Touchscreen      0.188631
Ips              0.250358
Ppi              0.469539
HDD             -0.098011
SSD              0.669957
Hybrid           0.022533
Flash_Storage   -0.037176
Name: Price, dtype: float64
```

In [75]:
```
dataset.drop(columns=['Hybrid','Flash_Storage'],inplace=True)
```

In [76]:
```
dataset['Gpu'].value_counts()
```

Out[76]:

```
Intel HD Graphics 620      279
Intel HD Graphics 520      181
Intel UHD Graphics 620      68
Nvidia GeForce GTX 1050     66
Nvidia GeForce GTX 1060     48
                          ... 
AMD Radeon R5 520            1
AMD Radeon R7                1
Intel HD Graphics 540        1
AMD Radeon 540               1
ARM Mali T860 MP4            1
Name: Gpu, Length: 110, dtype: int64
```

In [77]:

```
dataset['GpuBrand'] = dataset['Gpu'].apply(lambda x:x.split()[0])
```

In [78]:

```
dataset['GpuBrand'].value_counts()
```

Out[78]:

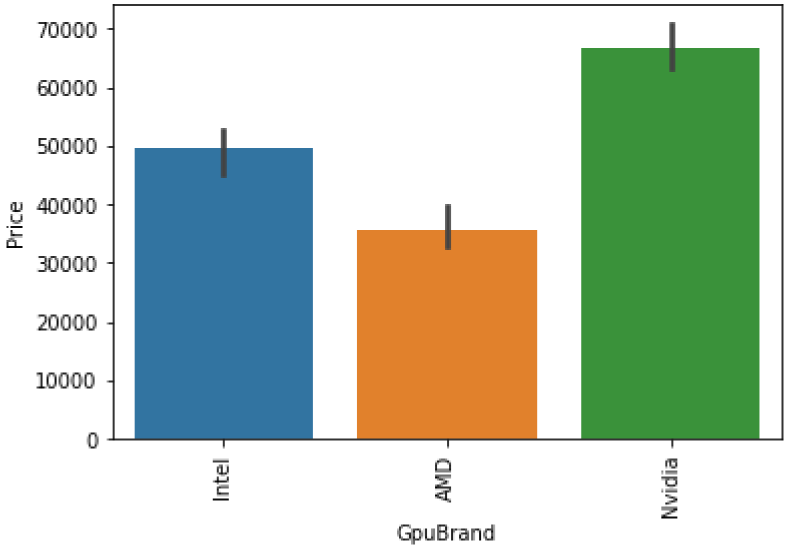
```
Intel     703
Nvidia    396
AMD       174
ARM         1
Name: GpuBrand, dtype: int64
```

In [79]:

```
dataset = dataset[dataset['GpuBrand'] != 'ARM']
```

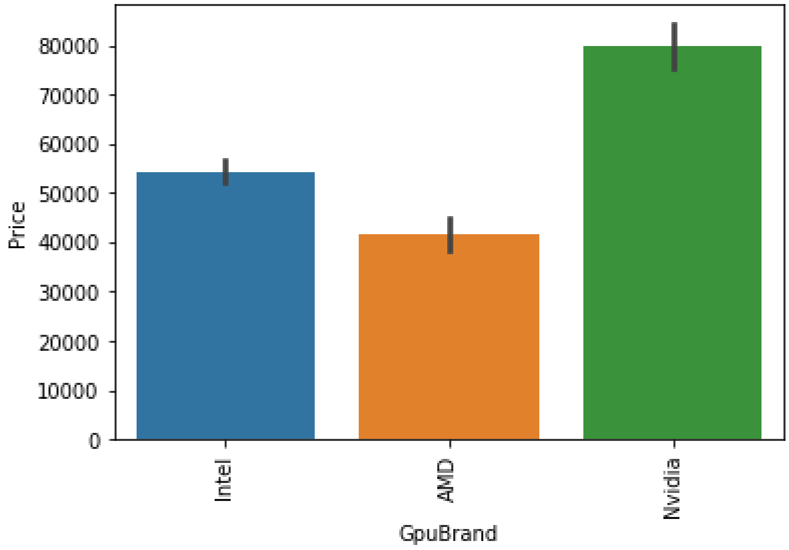
In [80]:

```
sns.barplot(dataset['GpuBrand'],dataset['Price'],estimator=np.median)
plt.xticks(rotation='vertical')
plt.show()
```



In [81]:

```
sns.barplot(dataset['GpuBrand'],dataset['Price'],estimator=np.mean)
plt.xticks(rotation='vertical')
plt.show()
```



In [82]:

```
dataset.drop(columns=['Gpu'],inplace=True)
```

In [83]:

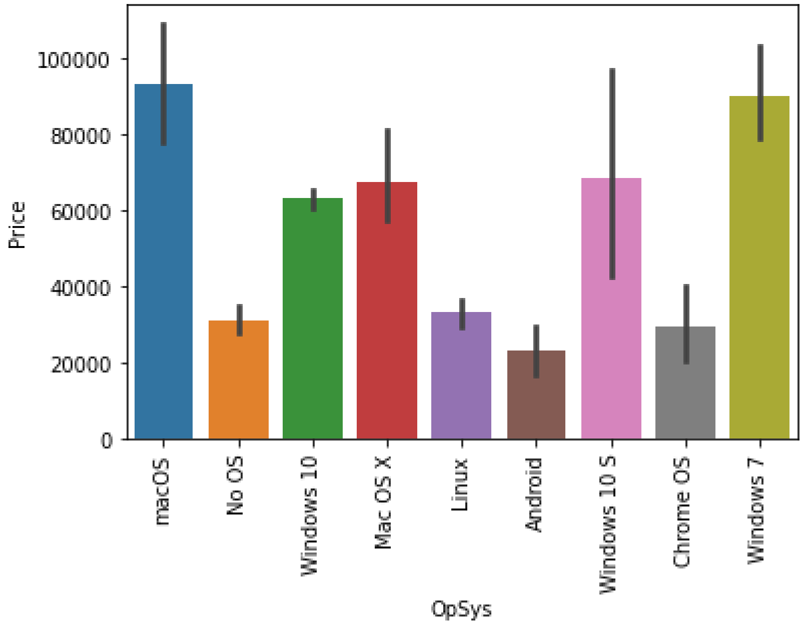
```
dataset['OpSys'].value_counts()
```

Out[83]:

```
Windows 10      1047
No OS             66
Linux             58
Windows 7         45
Chrome OS         26
macOS             13
Mac OS X           8
Windows 10 S       8
Android            2
Name: OpSys, dtype: int64
```

In [84]:

```
sns.barplot(dataset['OpSys'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



In [85]:
```python
def Operating_System(inp):

    if inp == 'Windows 10' or inp == 'Windows 7' or inp == 'Windows 10 S':
        return 'Windows'
    elif inp == 'macOS' or inp == 'Mac OS X':
        return 'Mac'
    else:
        return 'Others/No OS/Linux'
```

In [86]:
```python
dataset['Operating_System'] = dataset['OpSys'].apply(Operating_System)
```

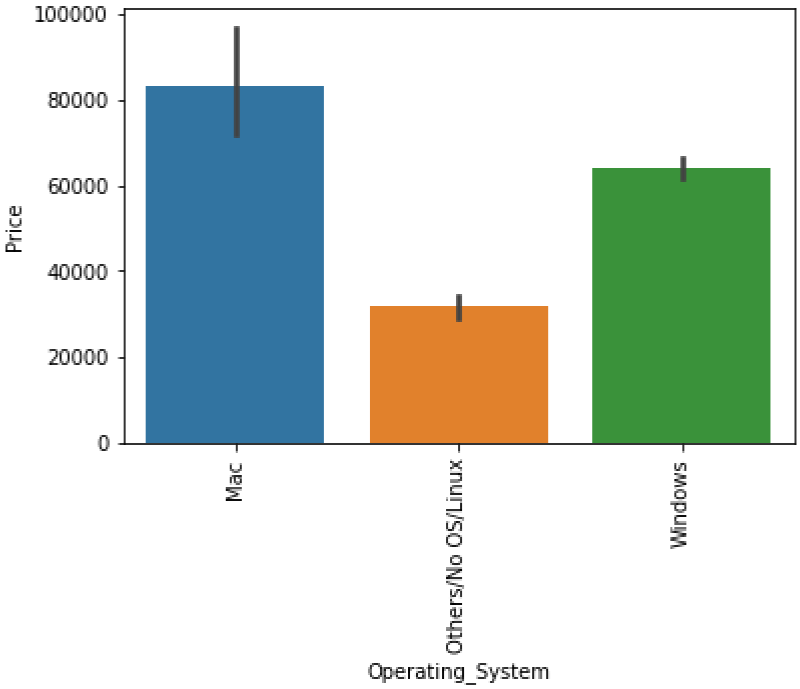
In [87]:
```python
dataset.drop(columns=['OpSys'],inplace=True)
```

In [88]:
```python
dataset.head()
```

Out[88]:

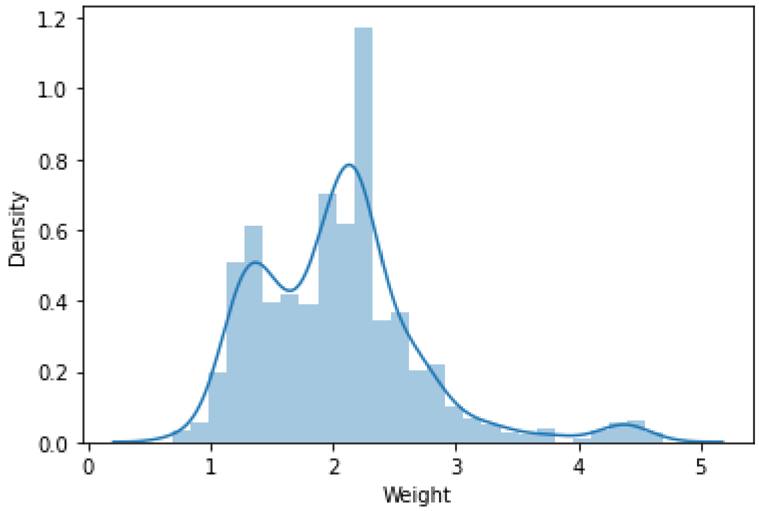
	Company	TypeName	Ram	Weight	Price	Touchscreen	Ips	Ppi	CpuBrand	HDD	S
0	Apple	Ultrabook	8	1.37	71378.6832	0	1	226.983005	Intel Core i5	0	1
1	Apple	Ultrabook	8	1.34	47895.5232	0	0	127.677940	Intel Core i5	0	
2	HP	Notebook	8	1.86	30636.0000	0	0	141.211998	Intel Core i5	0	2
3	Apple	Ultrabook	16	1.83	135195.3360	0	1	220.534624	Intel Core i7	0	5
4	Apple	Ultrabook	8	1.37	96095.8080	0	1	226.983005	Intel Core i5	0	2

In [89]:
```python
sns.barplot(dataset['Operating_System'],dataset['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



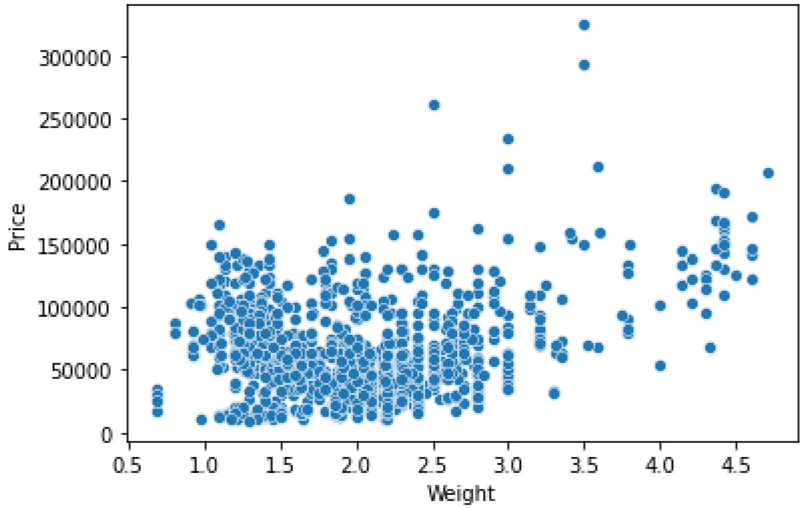
In [90]:

```
sns.distplot(dataset['Weight'])
plt.show()
```



In [91]:

```
sns.scatterplot(dataset['Weight'],dataset['Price'])
plt.show()
```



In [92]:

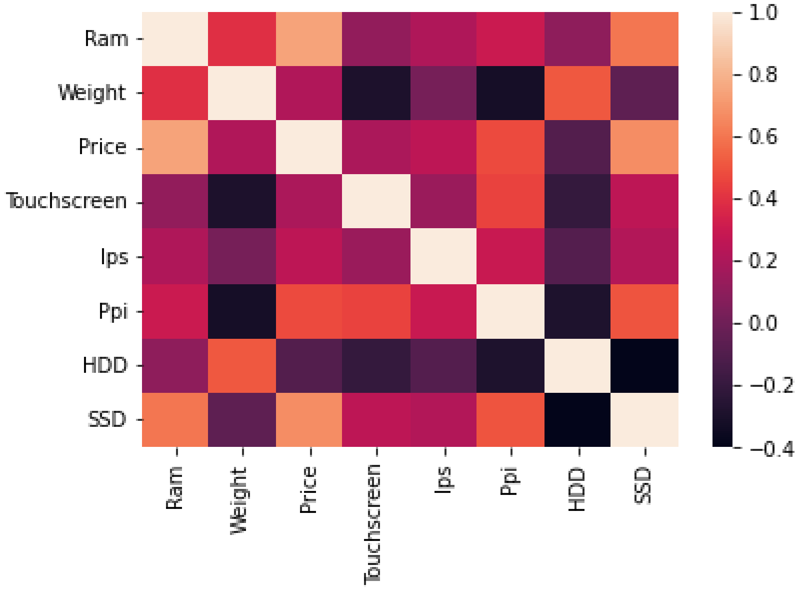
```
dataset.corr()['Price']
```

Out[92]:

```
Ram            0.739996
Weight         0.211667
Price          1.000000
Touchscreen    0.190382
Ips            0.251514
Ppi            0.471481
HDD           -0.098481
SSD            0.669808
Name: Price, dtype: float64
```

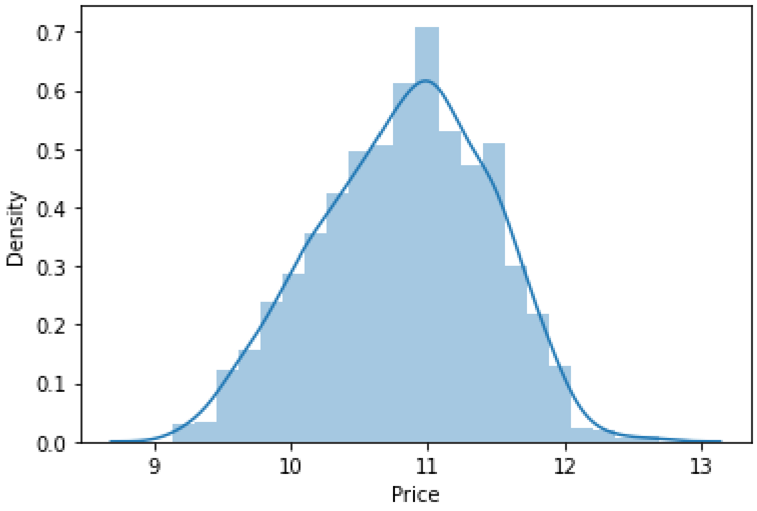
In [93]:

```
sns.heatmap(dataset.corr())
plt.show()
```



In [94]:

```
sns.distplot(np.log(dataset['Price']))
plt.show()
```



In [95]:

```
X = dataset.drop(columns=['Price'])
y = np.log(dataset['Price'])
```

In [96]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.15,random_sta
```

Linear Regression

In [97]:

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = LinearRegression()

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])
```

```
pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.7831567115450297
MAE 0.22167696491054562
```

Ridge Regression

In [98]:

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = Ridge(alpha=10)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.7954034359736732
MAE 0.21818884825637255
```

Lasso Regression

In [99]:

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = Lasso(alpha=0.001)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.7930860253542004
MAE 0.2187422326636071
```

KNN

In [100…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = KNeighborsRegressor(n_neighbors=3)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.7474369731031494
MAE 0.21441712657056713
```

Decision Tree

In [101…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = DecisionTreeRegressor(max_depth=8)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8106669247495699
MAE 0.20340316501737926
```

SVM

In [102…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = SVR(kernel='rbf',C=10000,epsilon=0.1)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)
```

```
y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8239727305710124
MAE 0.20196514668624901
```

Random Forest

In [103…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = RandomForestRegressor(n_estimators=100,
                              random_state=3,
                              max_samples=0.5,
                              max_features=0.75,
                              max_depth=15)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8497764509753524
MAE 0.18072585460593074
```

Extra Trees

In [104…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = ExtraTreesRegressor(n_estimators=100,
                              random_state=3,
                              max_samples=0.5,
                              max_features=0.75,
                              max_depth=15,bootstrap=True)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8556644099374368
MAE 0.17894313934894918
```

AdaBoost

In [105…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = AdaBoostRegressor(n_estimators=15,learning_rate=1.0)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.7768270112727226
MAE 0.23617321287648987
```

Gradient Boost

In [106…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = GradientBoostingRegressor(n_estimators=500)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.845750862468905
MAE 0.17453030722998336
```

XG Boost

In [107…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = XGBRegressor(n_estimators=45,max_depth=5,learning_rate=0.5)

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8728680414721783
MAE 0.16774693663306275
```

Voting Regressor

In [108…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')


rf = RandomForestRegressor(n_estimators=350,random_state=3,max_samples=0.5,max_
gbdt = GradientBoostingRegressor(n_estimators=100,max_features=0.5)
xgb = XGBRegressor(n_estimators=25,learning_rate=0.3,max_depth=5)
et = ExtraTreesRegressor(n_estimators=100,random_state=3,max_samples=0.5,max_f

step_2 = VotingRegressor([('rf', rf), ('gbdt', gbdt), ('xgb',xgb), ('et',et)],

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8613865152835886
MAE 0.17606426914012774
```

Stacking Regressor

In [109…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')


estimators = [
    ('rf', RandomForestRegressor(n_estimators=350,random_state=3,max_samples=0
    ('gbdt',GradientBoostingRegressor(n_estimators=100,max_features=0.5)),
    ('xgb', XGBRegressor(n_estimators=25,learning_rate=0.3,max_depth=5))
]

step_2 = StackingRegressor(estimators=estimators,final_estimator=Ridge(alpha=1

pipe = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe.fit(X_train,y_train)

y_pred = pipe.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8559921722484545
MAE 0.18218416201484888
```

In [130…

```
step_1 = ColumnTransformer(transformers=[
    ('col_tnf',OneHotEncoder(sparse=False,drop='first'),[0,1,7,10,11])
],remainder='passthrough')

step_2 = XGBRegressor(n_estimators=45,max_depth=5,learning_rate=0.5)

pipe_1 = Pipeline([
    ('step_1',step_1),
    ('step_2',step_2)
])

pipe_1.fit(X_train,y_train)

y_pred = pipe_1.predict(X_test)

print('R2_Score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
```

```
R2_Score 0.8728680414721783
MAE 0.16774693663306275
```

In [131…

```
pickle.dump(dataset,open('df.pkl','wb'))
pickle.dump(pipe_1,open('model_1.pkl','wb'))
```

In [133…

```
model = pickle.load(open('model_1.pkl','rb'))
```

In [141…

```
new_df = pd.DataFrame({
    'Company':'HP',
    'TypeName':'Notebook',
    'Ram':8,
    'Weight':1.86,
    'Touchscreen':0,
    'Ips':0,
    'Ppi':141.211998,
    'CpuBrand':'Intel Core i5',
    'HDD':0,
    'SSD':256,
    'GpuBrand':'Intel',
    'Operating_System':'Others/No OS/Linux'
},index=[7])
```

In [ ]:

```
model.predict(new_df)
```

CONNECT WITH ME :

LinkedIn GitHub kaggle Medium

PRASADMJADHAV2

```
In [ ]:
```