Import Libraries

In [265]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Reading the Dataset

In [266]:
```
df = pd.read_csv('laptop_data.csv')
df.head()
```

Out[266]:

	Unnamed: 0	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu
0	0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640
1	1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000
2	2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620
3	3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455
4	4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650

In [267]:
```
df1 = df.copy()
```

In [268]:
```
df.shape  # Rows - 1303,Columns - 12
```

Out[268]: (1303, 12)

In [269]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Unnamed: 0        1303 non-null   int64  
 1   Company           1303 non-null   object 
 2   TypeName          1303 non-null   object 
 3   Inches            1303 non-null   float64
 4   ScreenResolution  1303 non-null   object 
 5   Cpu               1303 non-null   object 
 6   Ram               1303 non-null   object 
 7   Memory            1303 non-null   object 
 8   Gpu               1303 non-null   object 
 9   OpSys             1303 non-null   object 
 10  Weight            1303 non-null   object 
 11  Price             1303 non-null   float64
dtypes: float64(2), int64(1), object(9)
memory usage: 122.3+ KB
```

In [270]:
```
df.isnull().sum()
```

Out[270]:
```
Unnamed: 0          0
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 [271]:
```
df.duplicated().sum()
```

Out[271]:
```
0
```

In [272]:
```
df.drop(columns=['Unnamed: 0'],inplace=True)
```

In [273]:
```
df['Ram'] = df['Ram'].str.replace('GB','')
```

In [274]:
```
df.head()
```

Out[274]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	We
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.3
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.3
2	HP	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.8
3	Apple	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.8
4	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.3

In [275]:
```
df['Weight'] = df['Weight'].str.replace('kg','')
```

In [276]:
```
df.head()
```

Out[276]:

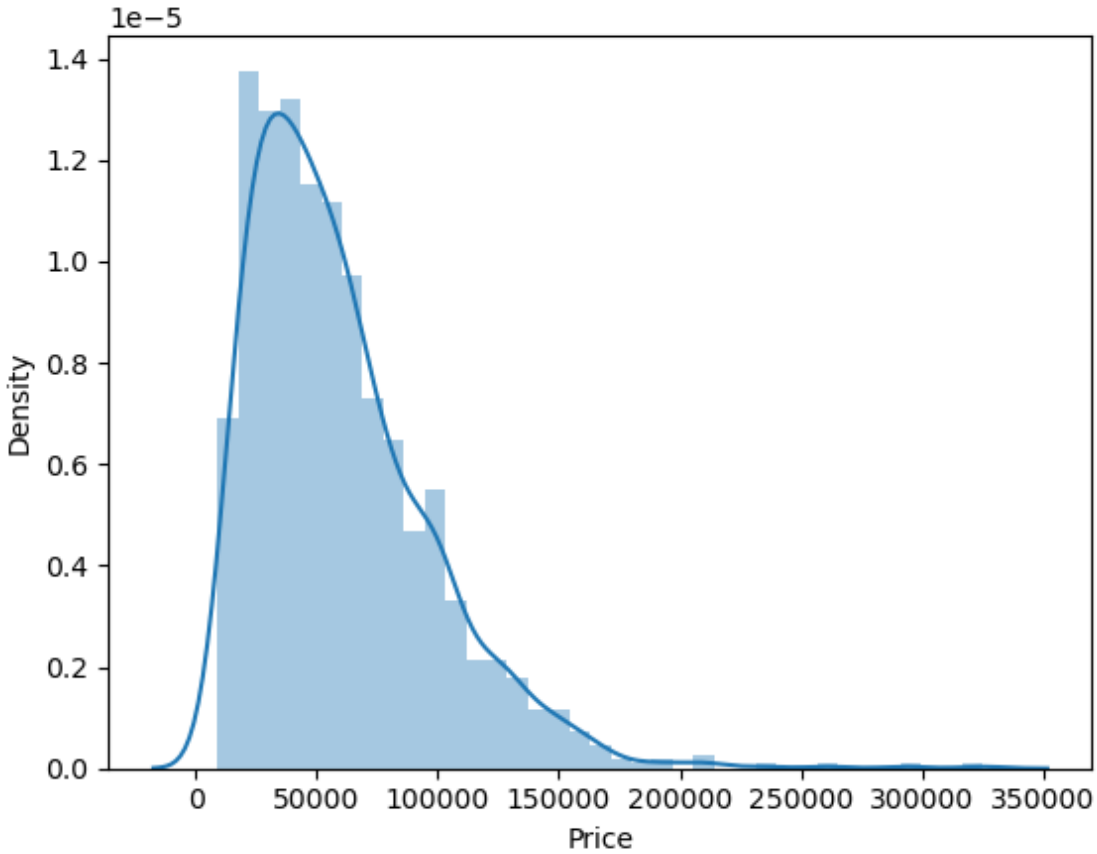
	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	We
0	Apple	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	
1	Apple	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	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

In [277]:
```
df['Ram'] = df['Ram'].astype('int32')
df['Weight'] = df['Weight'].astype('float32')
```

In [278]:
```
df.info()
```

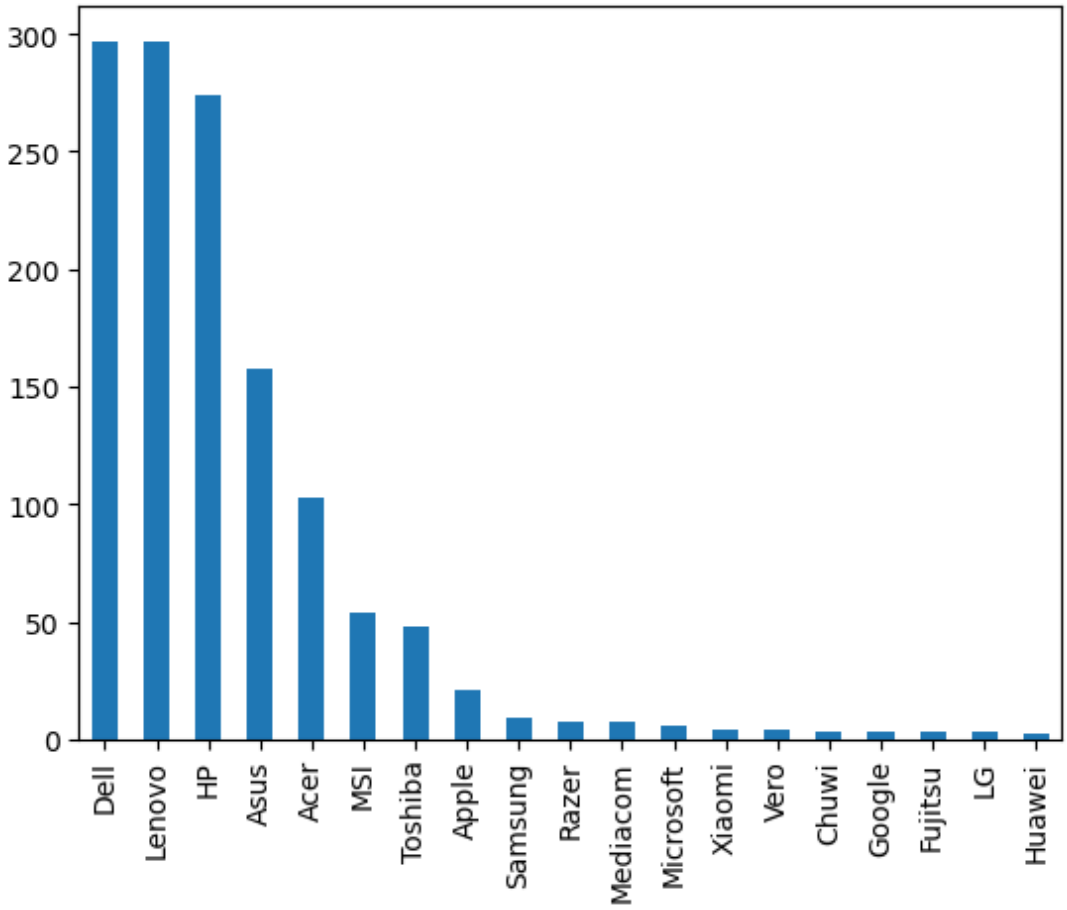
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Company           1303 non-null   object
 1   TypeName          1303 non-null   object
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object
 4   Cpu               1303 non-null   object
 5   Ram               1303 non-null   int32
 6   Memory            1303 non-null   object
 7   Gpu               1303 non-null   object
 8   OpSys             1303 non-null   object
 9   Weight            1303 non-null   float32
 10  Price             1303 non-null   float64
dtypes: float32(1), float64(2), int32(1), object(7)
memory usage: 101.9+ KB
```

In [279]:
```
sns.distplot(df['Price'])
plt.show()   # It is Rightly Skewed
```



In [280]:

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

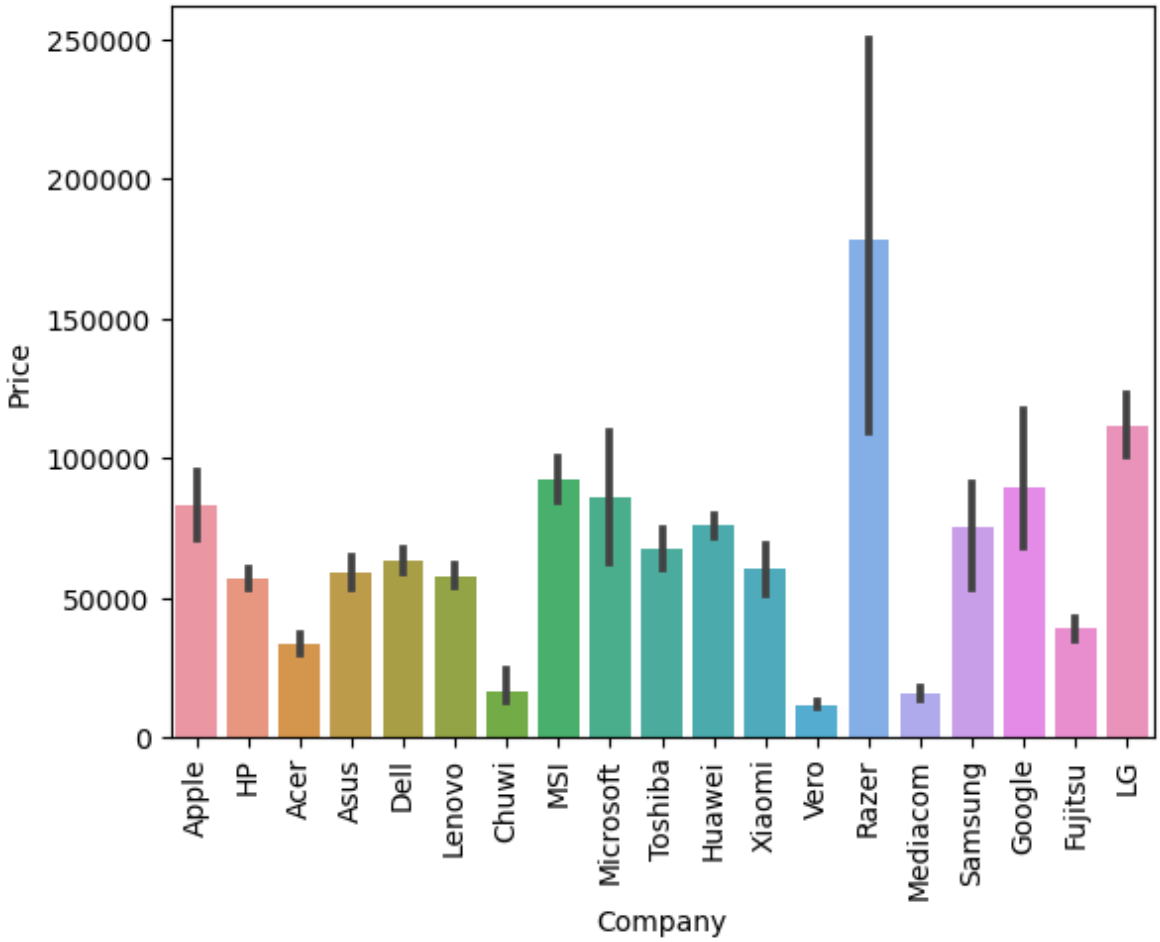


Inference

According to the data people buy more laptops of Dell

In [281]:

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

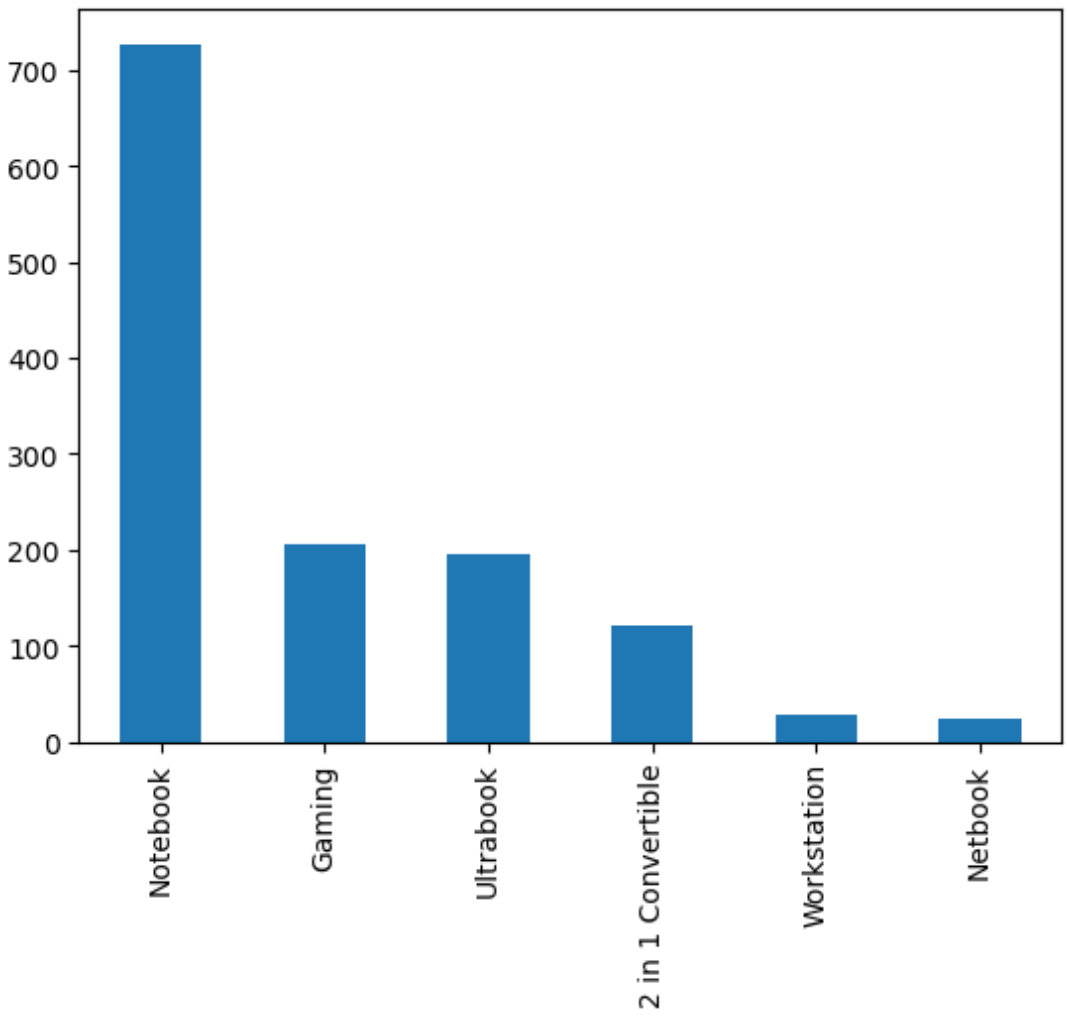


Inference

According to the Data Razer's laptop are more costly

In [282]:

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

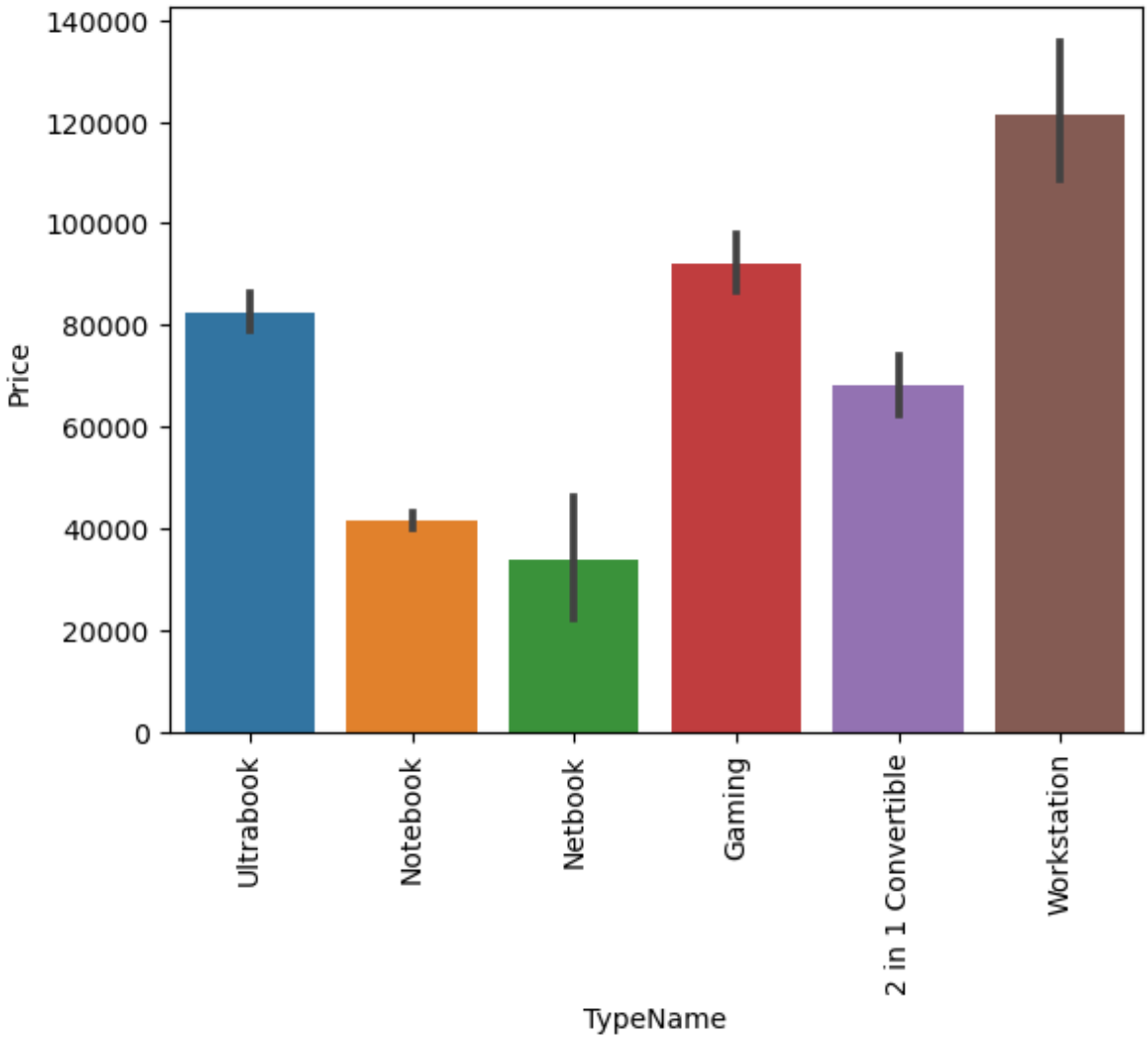


Inference

According to the data People buy more notebook types laptop

In [283]:

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

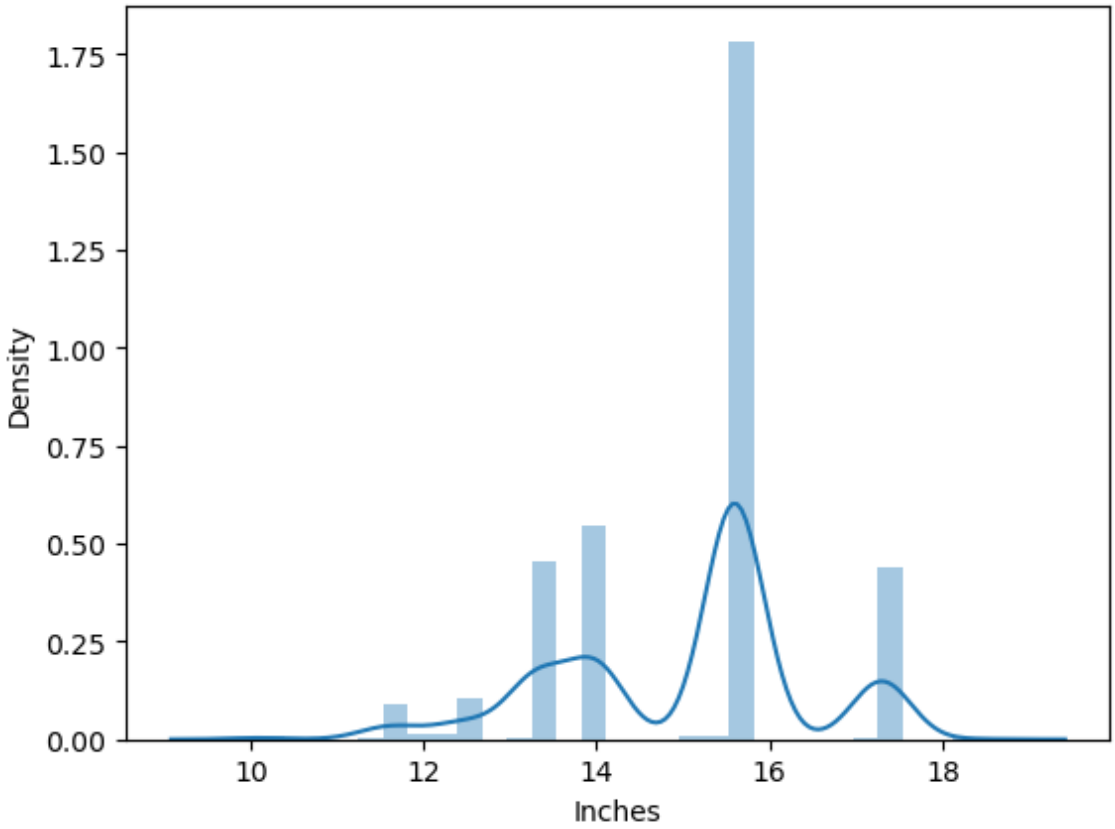


Inference

Accoring to the data Workstation is more costly compare to other laptop's type

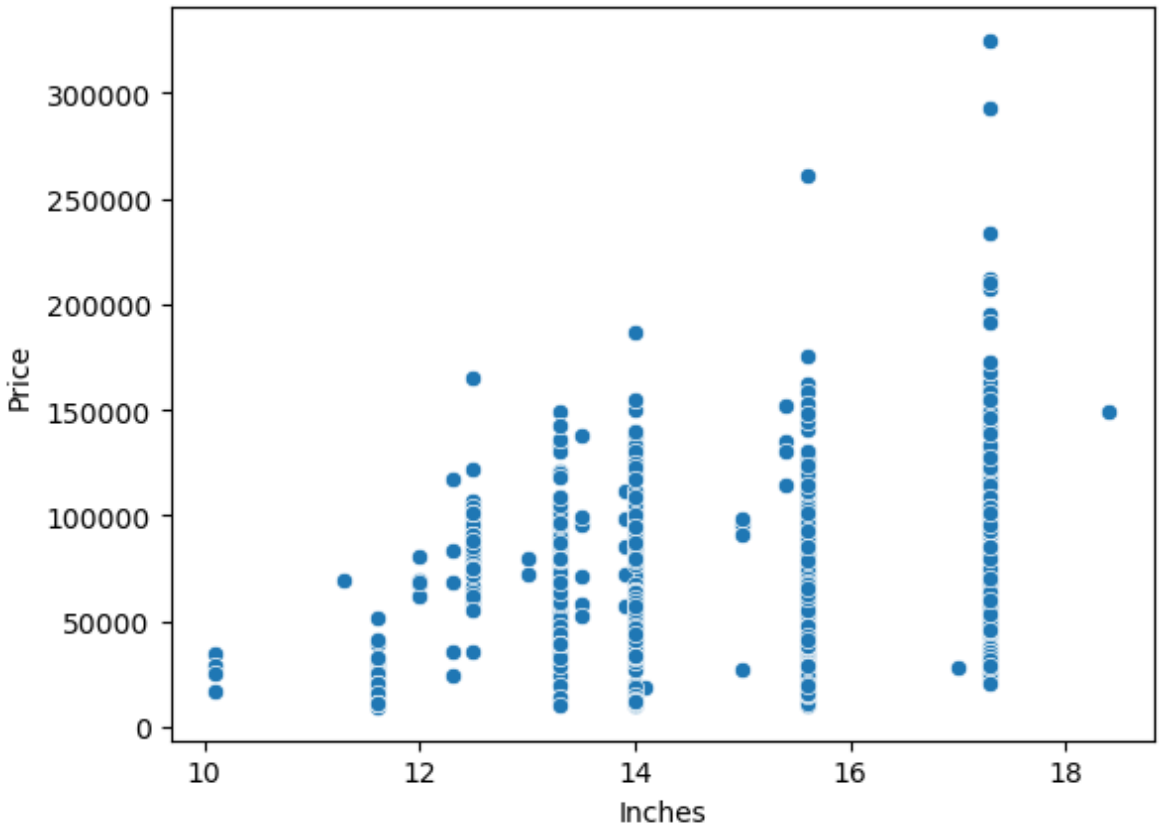
In [284]:

```
sns.distplot(df['Inches'])
plt.show()
```



In [285]:

```
sns.scatterplot(x=df['Inches'],y=df['Price'])
plt.show()
```



Inference

As we see as a inches increase price also increase

Feature Engineering

In [286]:
```
df['ScreenResolution'].value_counts()
```

Out[286]:
```
Full HD 1920x1080                                   507
1366x768                                            281
IPS Panel Full HD 1920x1080                         230
IPS Panel Full HD / Touchscreen 1920x1080            53
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 Quad HD+ / Touchscreen 3200x1800            6
IPS Panel Retina Display 2560x1600                    6
IPS Panel Retina Display 2304x1440                    6
Touchscreen 2256x1504                                 6
IPS Panel Touchscreen 2560x1440                       5
```

Creating New Feature Touch Screen

In [287]:
```
df['TouchScreen'] = df['ScreenResolution'].apply(lambda x :1 if 'Touchscre
```

In [288]:
```
df.sample(10)
```

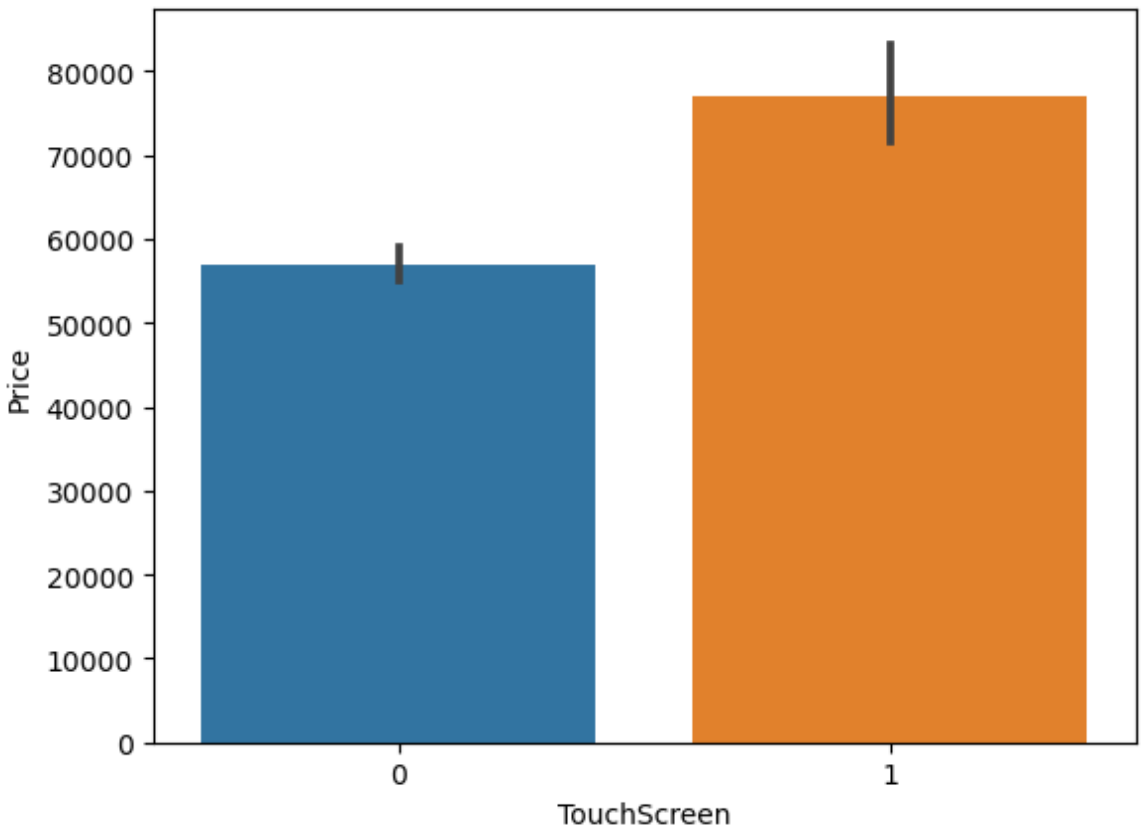
Out[288]:

	Company	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys
231	HP	Notebook	15.6	1366x768	AMD E-Series 9000e 1.5GHz	4	500GB HDD	AMD Radeon R2	Windows 1(
276	Dell	Notebook	17.3	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	8	128GB SSD + 1TB HDD	AMD Radeon 530	Linux
72	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	8	256GB SSD	AMD Radeon 530	Windows 1(
1203	Dell	Ultrabook	13.3	Quad HD+ / Touchscreen 3200x1800	Intel Core i7 7500U 2.7GHz	16	512GB SSD	Intel HD Graphics 620	Windows 1(
136	Lenovo	Notebook	15.6	1366x768	Intel Celeron Dual Core N3350 1.1GHz	4	1TB HDD	Intel HD Graphics 500	No OS
285	Acer	Notebook	15.6	IPS Panel Full HD / Touchscreen 1920x1080	Intel Core i7 7500U 2.7GHz	12	1TB HDD	Intel HD Graphics 620	Windows 1(
77	Dell	Notebook	15.6	Full HD 1920x1080	Intel Core i7 8550U 1.8GHz	8	128GB SSD + 1TB HDD	Intel UHD Graphics 620	Windows 1(
1085	HP	Notebook	14.0	Full HD 1920x1080	Intel Core i5 6200U 2.3GHz	4	500GB HDD	Intel HD Graphics 520	Windows 7
252	Asus	Notebook	15.6	1366x768	AMD A9-Series 9420 3GHz	4	1TB HDD	AMD Radeon R5 M420	Windows 1(
1209	Asus	Gaming	15.6	Full HD 1920x1080	Intel Core i7 7700HQ 2.8GHz	16	256GB SSD + 1TB HDD	Nvidia GeForce GTX 1070	Windows 1(

In [289]:
```
df['TouchScreen'].value_counts()  # 0 - not Touch Screen, 1 - Touch Screen
```

Out[289]:
```
0    1111
1     192
Name: TouchScreen, dtype: int64
```

In [290]:
```
sns.barplot(x=df['TouchScreen'],y=df['Price'])
plt.show()
```



Inference

As Price has positive realtion with Touch Screen If laptop is touch screen it is more costly

Creating New Column IPS Display

In [291]:
```
df['IPS'] = df['ScreenResolution'].apply(lambda x:1 if 'IPS' in x else 0)
# 1 - For IPS Display, 0 - Not IPS Display
```

In [292]:
```
df.head()
```

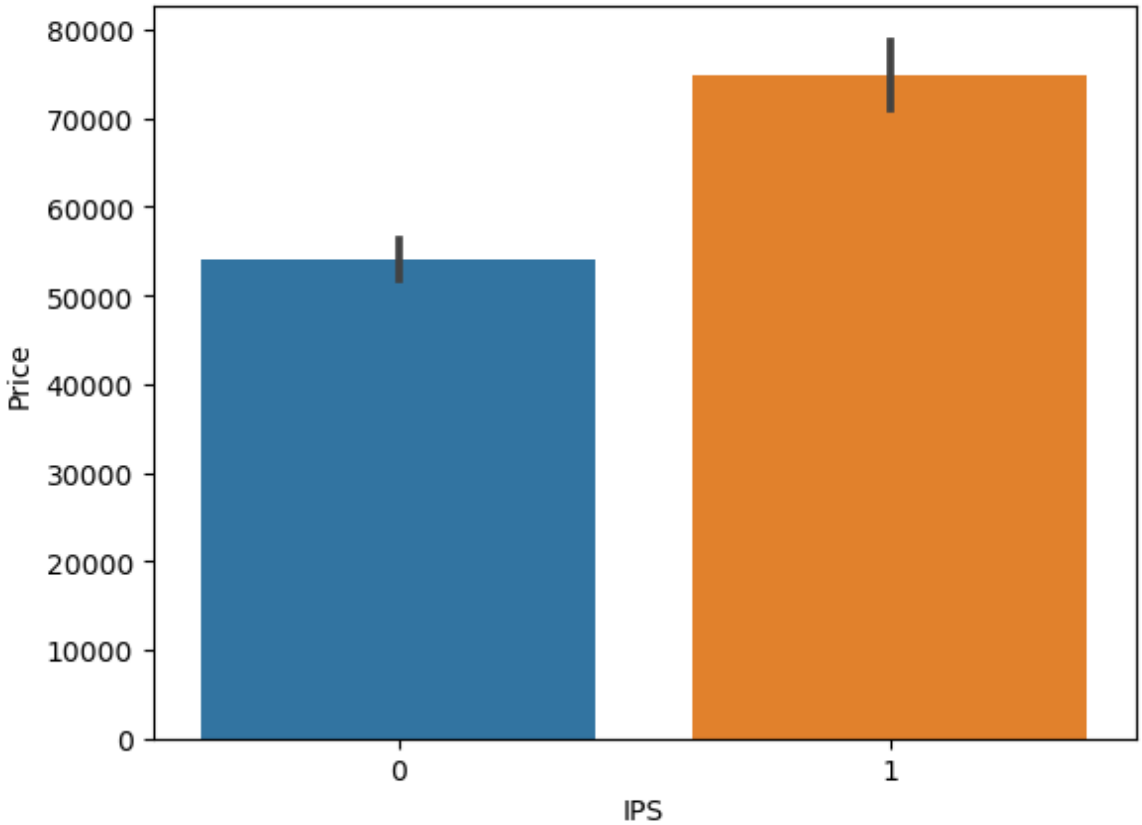
Out[292]:

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

In [293]:
```
df['IPS'].value_counts()
```

Out[293]:
```
0    938
1    365
Name: IPS, dtype: int64
```

In [294]:
```
sns.barplot(x=df['IPS'],y=df['Price'])
plt.show()
```



Inference

As Price has positive realtion with IPS Display If laptop has IPS Display it is more costly

In [295]:
```
new = df['ScreenResolution'].str.split('x',n=1,expand=True)
```

In [296]:
```
df['X_res'] = new[0]
df['Y_res'] = new[1]
```

In [297]:
```
df.head()
```

Out[297]:

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

In [298]:
```
df['X_res'] = df['X_res'].str.replace(',','').str.findall(r'(\d+\.?\d+)').
```

In [299]:

```
df.head()
```

Out[299]:

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

In [300]:

```
df['X_res'] = df['X_res'].astype('int')
df['Y_res'] = df['Y_res'].astype('int')
```

In [301]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Company           1303 non-null   object
 1   TypeName          1303 non-null   object
 2   Inches            1303 non-null   float64
 3   ScreenResolution  1303 non-null   object
 4   Cpu               1303 non-null   object
 5   Ram               1303 non-null   int32
 6   Memory            1303 non-null   object
 7   Gpu               1303 non-null   object
 8   OpSys             1303 non-null   object
 9   Weight            1303 non-null   float32
 10  Price             1303 non-null   float64
 11  TouchScreen       1303 non-null   int64
 12  IPS               1303 non-null   int64
 13  X_res             1303 non-null   int32
 14  Y_res             1303 non-null   int32
dtypes: float32(1), float64(2), int32(3), int64(2), object(7)
memory usage: 132.5+ KB
```

In [302]:
```
df.corr()['Price']
```

Out[302]:
```
Inches         0.068197
Ram            0.743007
Weight         0.210370
Price          1.000000
TouchScreen    0.191226
IPS            0.252208
X_res          0.556529
Y_res          0.552809
Name: Price, dtype: float64
```

In [303]:
```
df['ppi'] = (((df['X_res']**2) + (df['Y_res']**2))**0.5/df['Inches']).asty
```

In [304]:
```
df.corr()['Price']
```

Out[304]:
```
Inches         0.068197
Ram            0.743007
Weight         0.210370
Price          1.000000
TouchScreen    0.191226
IPS            0.252208
X_res          0.556529
Y_res          0.552809
ppi            0.473487
Name: Price, dtype: float64
```

In [305]:
```
df.drop(columns=['ScreenResolution'],inplace=True)
```

In [306]:
```
df.head()
```

Out[306]:

	Company	TypeName	Inches	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price
0	Apple	Ultrabook	13.3	Intel Core i5 2.3GHz	8	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832
1	Apple	Ultrabook	13.3	Intel Core i5 1.8GHz	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232
2	HP	Notebook	15.6	Intel Core i5 7200U 2.5GHz	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000
3	Apple	Ultrabook	15.4	Intel Core i7 2.7GHz	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360
4	Apple	Ultrabook	13.3	Intel Core i5 3.1GHz	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080

In [307]:
```
df.drop(columns=['Inches','X_res','Y_res'],inplace=True)
```

In [308]:
```
df.head()
```

Out[308]:

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

In [309]:
```
df['Cpu'].value_counts()
```

Out[309]:
```
Intel Core i5 7200U 2.5GHz       190
Intel Core i7 7700HQ 2.8GHz      146
Intel Core i7 7500U 2.7GHz       134
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 i3 6006U 2.2GHz          1
AMD A6-Series 7310 2GHz             1
Intel Xeon E3-1535M v6 3.1GHz       1
Name: Cpu, Length: 118, dtype: int64
```

In [310]:
```
df['Cpu Name'] = df['Cpu'].apply(lambda x:" ".join(x.split()[0:3]))
```

In [311]:

```
df.head()
```

Out[311]:

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

In [312]:

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

In [313]:

```
df['Cpu brand'] = df['Cpu Name'].apply(fetch_processor)
```

In [314]:

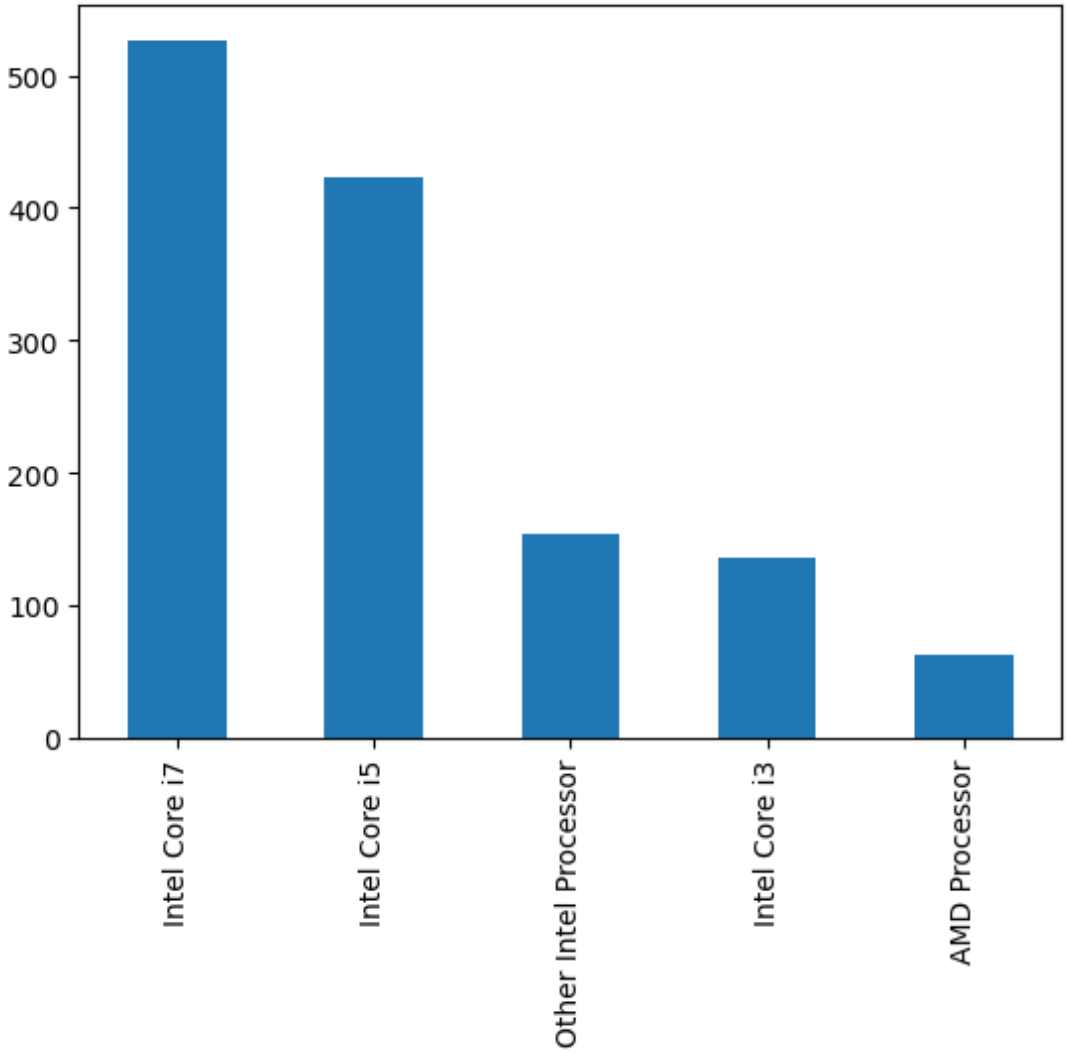
```
df.head()
```

Out[314]:

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

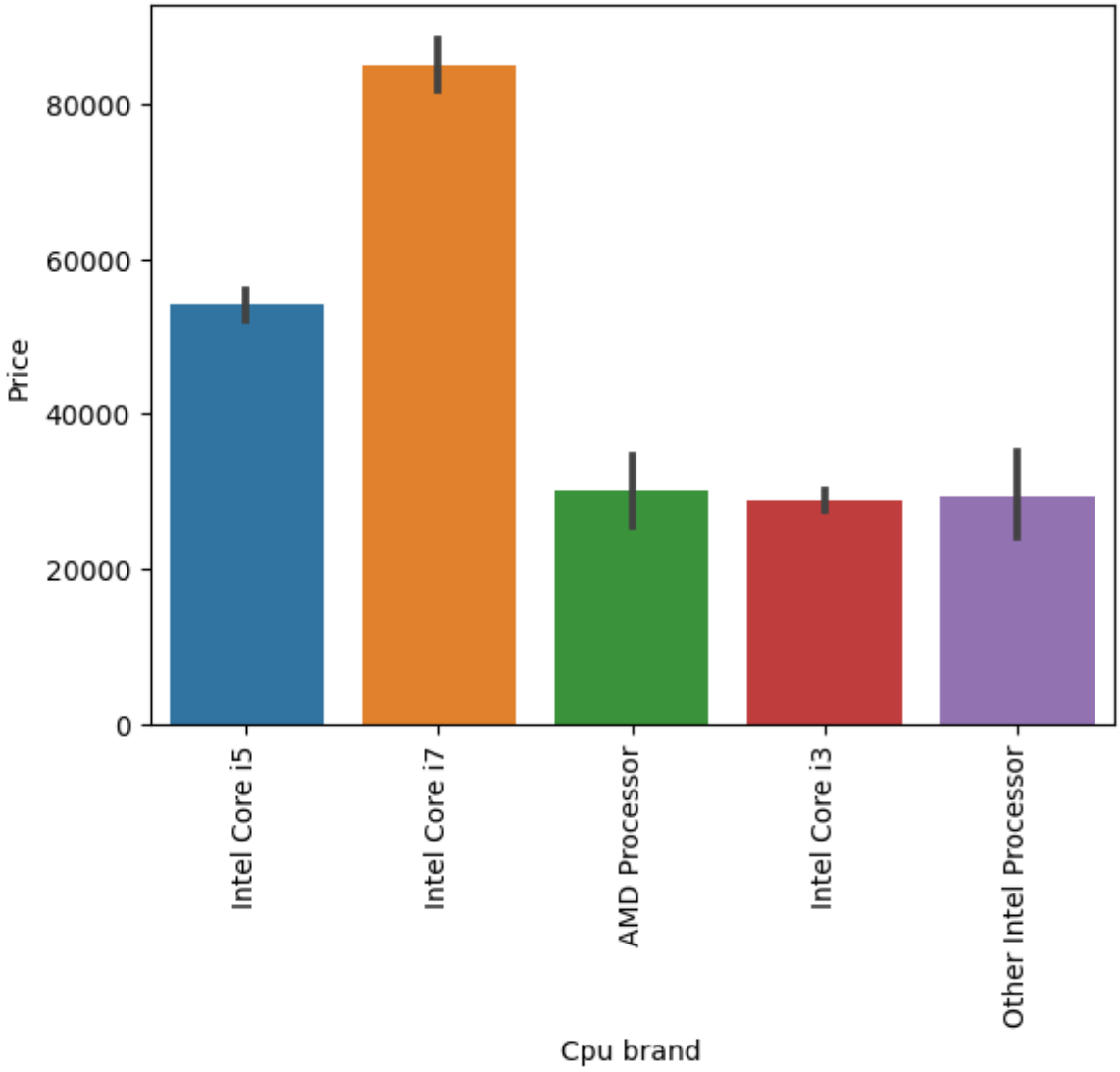
In [315]:
```
df['Cpu brand'].value_counts().plot(kind='bar')
```

Out[315]: <AxesSubplot:>



In [316]:

```
sns.barplot(x=df['Cpu brand'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



In [317]:

```
df.drop(columns=['Cpu','Cpu Name'],inplace=True)
```

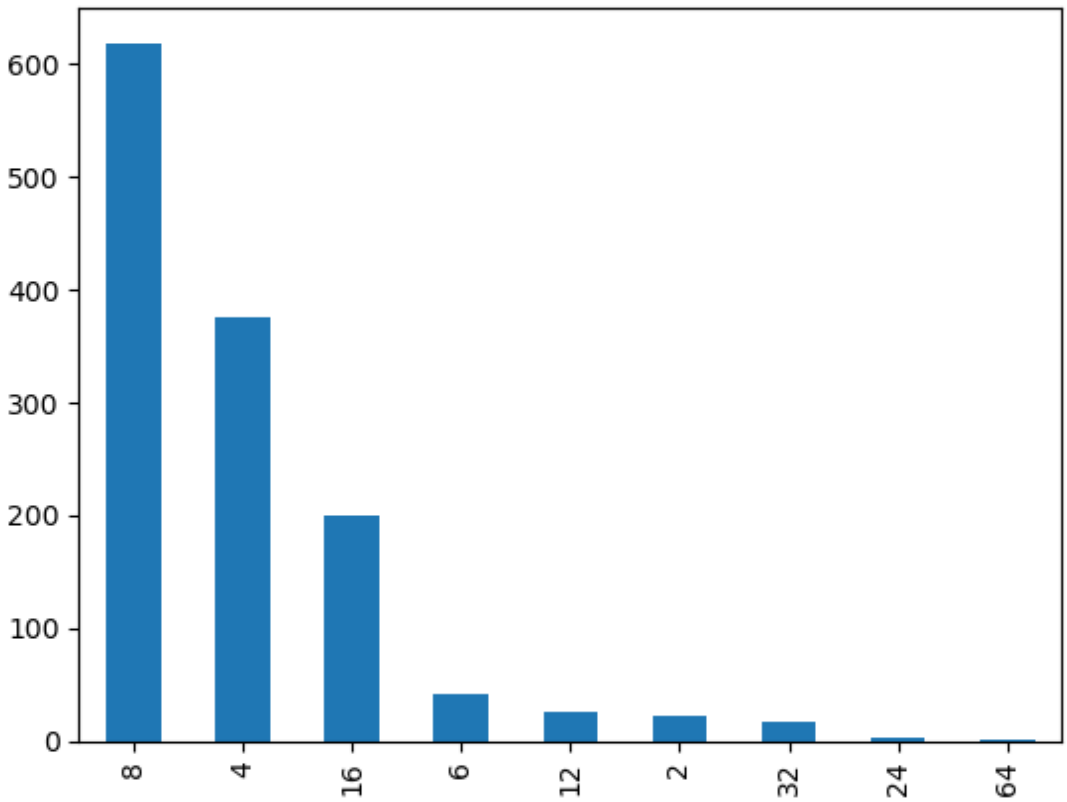
In [318]:
```
df.head()
```

Out[318]:

	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	Apple	Ultrabook	8	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	
2	HP	Notebook	8	256GB SSD	Intel HD Graphics 620	No OS	1.86	30636.0000	0	
3	Apple	Ultrabook	16	512GB SSD	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	
4	Apple	Ultrabook	8	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0	

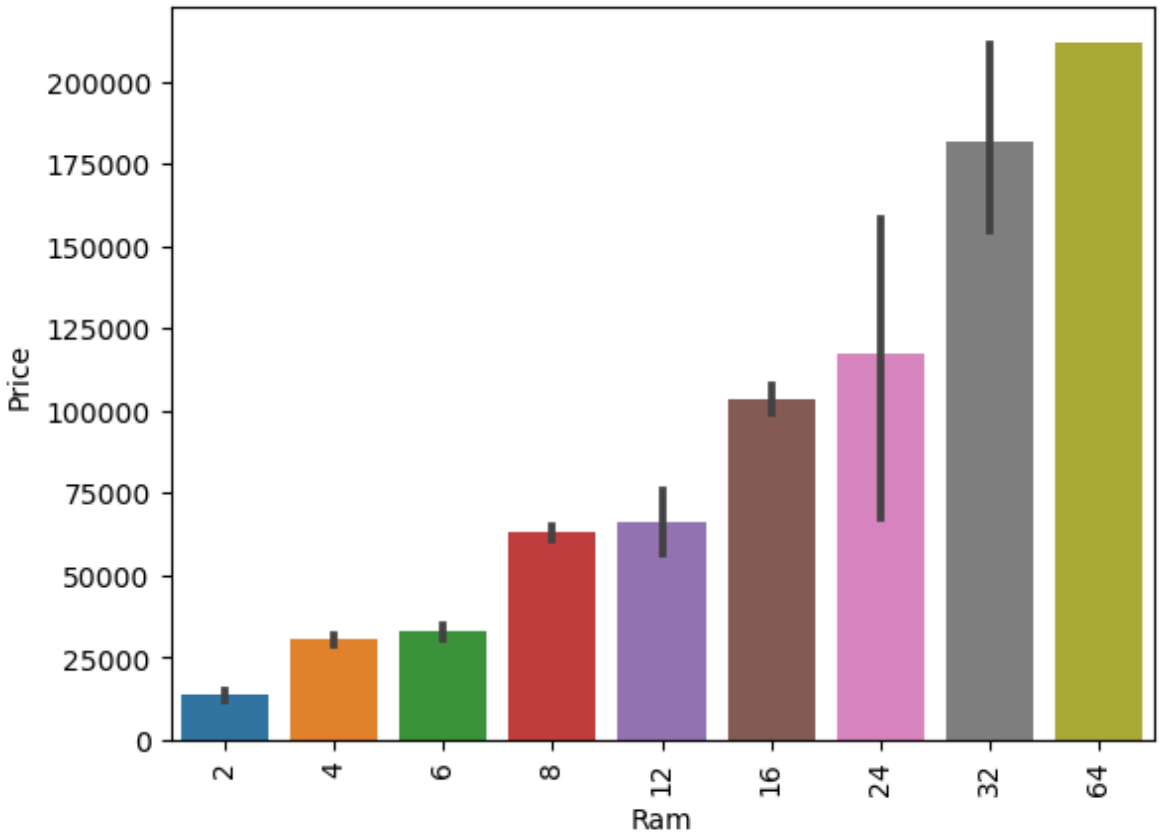
In [319]:
```
df['Ram'].value_counts().plot(kind='bar')
```

Out[319]: <AxesSubplot:>



In [320]:

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



In [321]:
```
df['Memory'].value_counts()
```

Out[321]:
```
256GB SSD                        412
1TB HDD                          223
500GB HDD                        132
512GB SSD                        118
128GB SSD +  1TB HDD              94
128GB SSD                         76
256GB SSD +  1TB HDD              73
32GB Flash Storage                38
2TB HDD                           16
64GB Flash Storage                15
512GB SSD +  1TB HDD              14
1TB SSD                           14
256GB SSD +  2TB HDD              10
1.0TB Hybrid                       9
256GB Flash Storage                8
16GB Flash Storage                 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 [322]:

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

df["first"]= new[0]
df["first"]=df["first"].str.strip()

df["second"]= new[1]

df["Layer1HDD"] = df["first"].apply(lambda x: 1 if "HDD" in x else 0)
df["Layer1SSD"] = df["first"].apply(lambda x: 1 if "SSD" in x else 0)
df["Layer1Hybrid"] = df["first"].apply(lambda x: 1 if "Hybrid" in x else 0
df["Layer1Flash_Storage"] = df["first"].apply(lambda x: 1 if "Flash Storag

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

df["second"].fillna("0", inplace = True)

df["Layer2HDD"] = df["second"].apply(lambda x: 1 if "HDD" in x else 0)
df["Layer2SSD"] = df["second"].apply(lambda x: 1 if "SSD" in x else 0)
df["Layer2Hybrid"] = df["second"].apply(lambda x: 1 if "Hybrid" in x else
df["Layer2Flash_Storage"] = df["second"].apply(lambda x: 1 if "Flash Stora

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

df["first"] = df["first"].astype(int)
df["second"] = df["second"].astype(int)

df["HDD"]=(df["first"]*df["Layer1HDD"]+df["second"]*df["Layer2HDD"])
df["SSD"]=(df["first"]*df["Layer1SSD"]+df["second"]*df["Layer2SSD"])
df["Hybrid"]=(df["first"]*df["Layer1Hybrid"]+df["second"]*df["Layer2Hybrid
df["Flash_Storage"]=(df["first"]*df["Layer1Flash_Storage"]+df["second"]*df

df.drop(columns=['first', 'second', 'Layer1HDD', 'Layer1SSD', 'Layer1Hybri
        'Layer1Flash_Storage', 'Layer2HDD', 'Layer2SSD', 'Layer2Hybrid',
        'Layer2Flash_Storage'],inplace=True)
```

In [323]:
```
df.sample(5)
```

Out[323]:

	Company	TypeName	Ram	Memory	Gpu	OpSys	Weight	Price	TouchScreen
1051	HP	Notebook	8	1000 HDD	Nvidia GeForce 940MX	Windows 10	1.91	52161.1200	0
751	HP	Notebook	8	256 SSD	Intel HD Graphics 520	Windows 10	1.84	101232.0000	0
205	Lenovo	Gaming	16	512 SSD	Nvidia GeForce GTX 1060	No OS	2.40	74485.4400	0
720	Lenovo	Ultrabook	8	512 SSD	Intel HD Graphics 520	Windows 10	1.17	89864.1792	0
1130	HP	Notebook	8	2000 HDD	Intel HD Graphics 620	Windows 10	2.04	33513.1200	0

In [324]:
```
df.drop(columns=['Memory'],inplace=True)
```

In [325]:
```
df.head()
```

Out[325]:

	Company	TypeName	Ram	Gpu	OpSys	Weight	Price	TouchScreen	IPS	
0	Apple	Ultrabook	8	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983
1	Apple	Ultrabook	8	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677
2	HP	Notebook	8	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211
3	Apple	Ultrabook	16	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	1	220.534
4	Apple	Ultrabook	8	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0	1	226.983

In [326]:
```
df.corr()['Price']
```

Out[326]:
```
Ram              0.743007
Weight           0.210370
Price            1.000000
TouchScreen      0.191226
IPS              0.252208
ppi              0.473487
HDD             -0.096441
SSD              0.670799
Hybrid           0.007989
Flash_Storage   -0.040511
Name: Price, dtype: float64
```

In [327]:
```
df.drop(columns=['Hybrid','Flash_Storage'],inplace=True)
```

In [328]:
```
df.head()
```

Out[328]:

	Company	TypeName	Ram	Gpu	OpSys	Weight	Price	TouchScreen	IPS	
0	Apple	Ultrabook	8	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983
1	Apple	Ultrabook	8	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677
2	HP	Notebook	8	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211
3	Apple	Ultrabook	16	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	1	220.534
4	Apple	Ultrabook	8	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0	1	226.983

In [329]:
```
df['Gpu'].value_counts()
```

Out[329]:
```
Intel HD Graphics 620      281
Intel HD Graphics 520      185
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 [330]:
```
df['Gpu brand'] = df['Gpu'].apply(lambda x:x.split()[0])
```

In [331]:
```
df.head()
```

Out[331]:

	Company	TypeName	Ram	Gpu	OpSys	Weight	Price	TouchScreen	IPS	
0	Apple	Ultrabook	8	Intel Iris Plus Graphics 640	macOS	1.37	71378.6832	0	1	226.983
1	Apple	Ultrabook	8	Intel HD Graphics 6000	macOS	1.34	47895.5232	0	0	127.677
2	HP	Notebook	8	Intel HD Graphics 620	No OS	1.86	30636.0000	0	0	141.211
3	Apple	Ultrabook	16	AMD Radeon Pro 455	macOS	1.83	135195.3360	0	1	220.534
4	Apple	Ultrabook	8	Intel Iris Plus Graphics 650	macOS	1.37	96095.8080	0	1	226.983

In [332]:
```
df['Gpu brand'].value_counts()
```

Out[332]:
```
Intel     722
Nvidia    400
AMD       180
ARM         1
Name: Gpu brand, dtype: int64
```

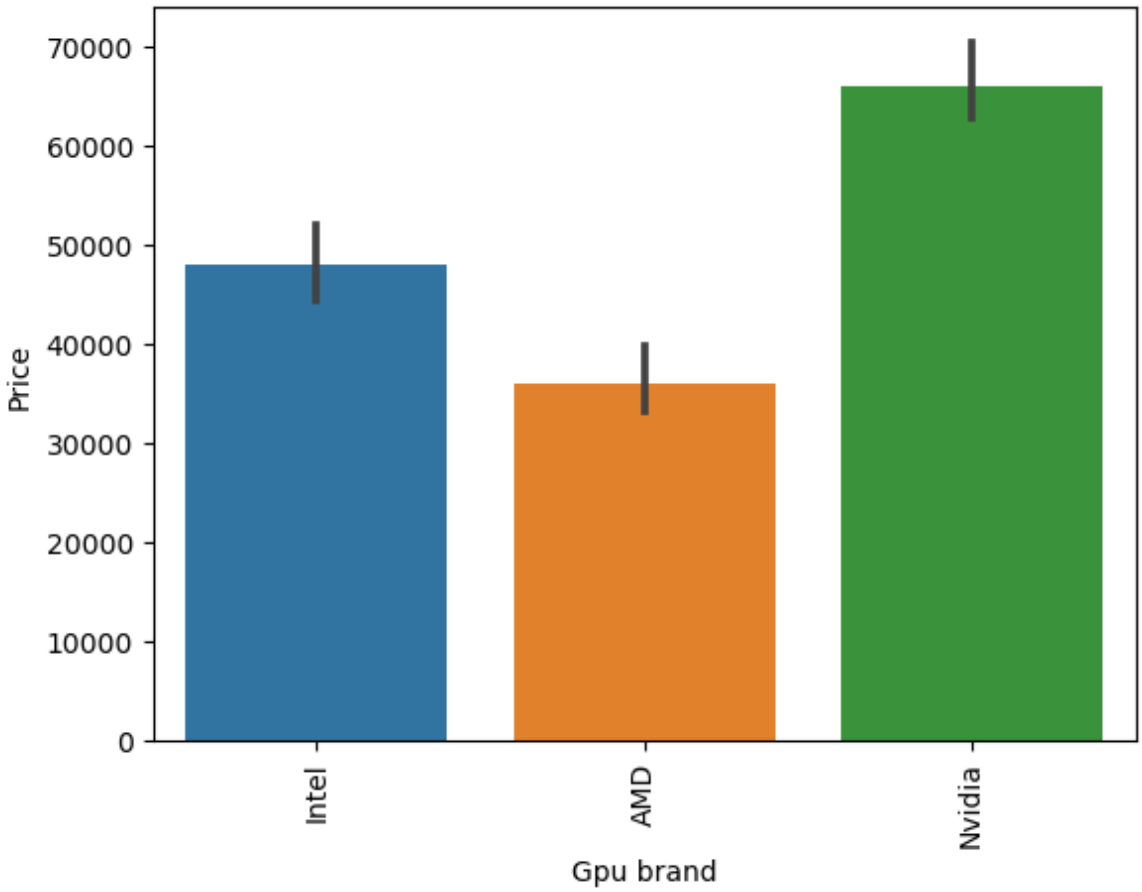
In [333]:
```
df = df[df['Gpu brand'] != 'ARM']
```

In [334]:
```
df['Gpu brand'].value_counts()
```

Out[334]:
```
Intel     722
Nvidia    400
AMD       180
Name: Gpu brand, dtype: int64
```

In [335]:

```
sns.barplot(x=df['Gpu brand'],y=df['Price'],estimator=np.median)
plt.xticks(rotation='vertical')
plt.show()
```



In [336]:

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

In [337]:

```
df.head()
```

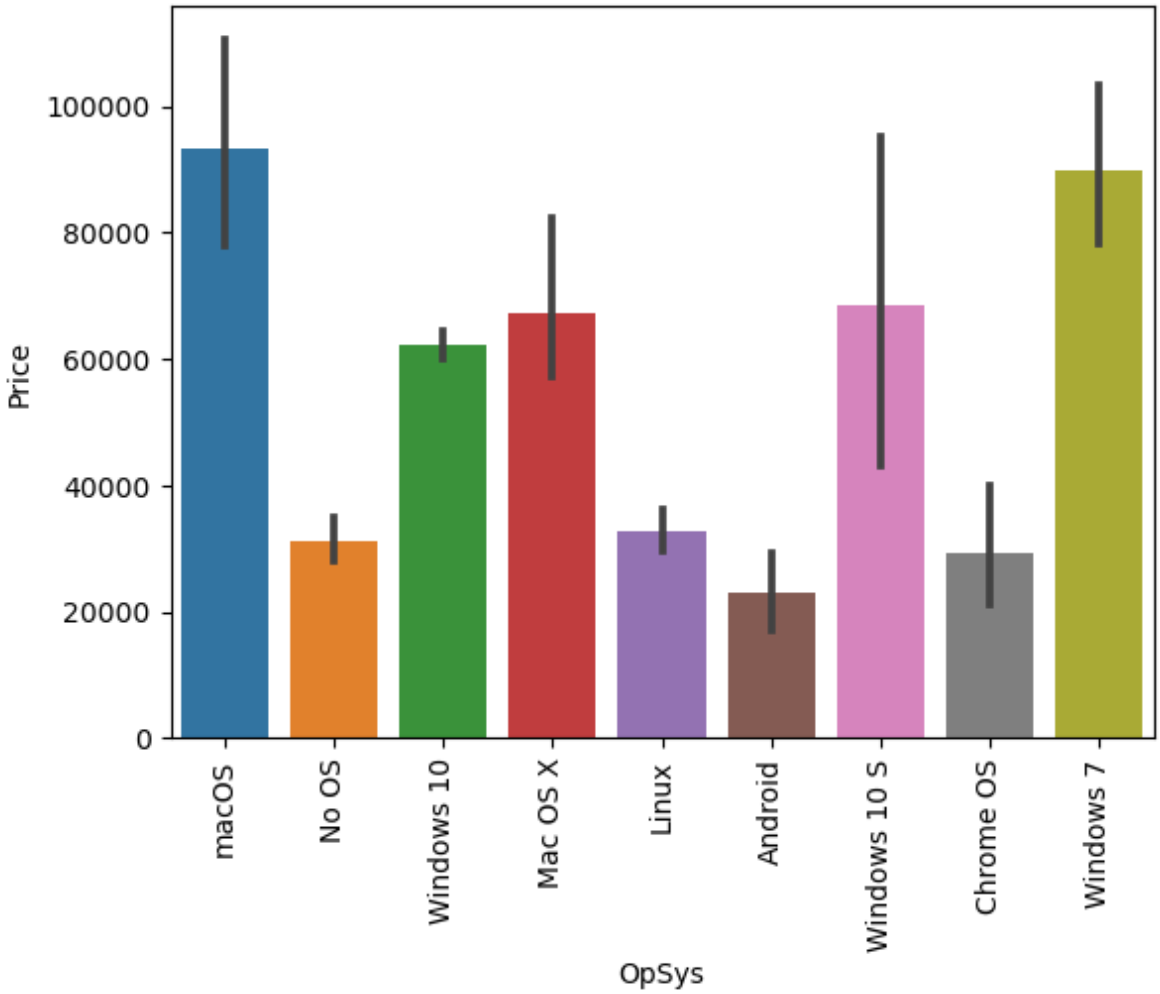
Out[337]:

	Company	TypeName	Ram	OpSys	Weight	Price	TouchScreen	IPS	ppi	Cp bran
0	Apple	Ultrabook	8	macOS	1.37	71378.6832	0	1	226.983005	Int Cor
1	Apple	Ultrabook	8	macOS	1.34	47895.5232	0	0	127.677940	Int Cor
2	HP	Notebook	8	No OS	1.86	30636.0000	0	0	141.211998	Int Cor
3	Apple	Ultrabook	16	macOS	1.83	135195.3360	0	1	220.534624	Int Cor
4	Apple	Ultrabook	8	macOS	1.37	96095.8080	0	1	226.983005	Int Cor

In [338]:
```
df['OpSys'].value_counts()
```

Out[338]:
```
Windows 10      1072
No OS             66
Linux             62
Windows 7         45
Chrome OS         26
macOS             13
Mac OS X           8
Windows 10 S       8
Android            2
Name: OpSys, dtype: int64
```

In [339]:
```
sns.barplot(x=df['OpSys'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



In [340]:
```
def cat_os(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 [341]:
```
df['os'] = df['OpSys'].apply(cat_os)
```

In [342]:
```
df.head()
```

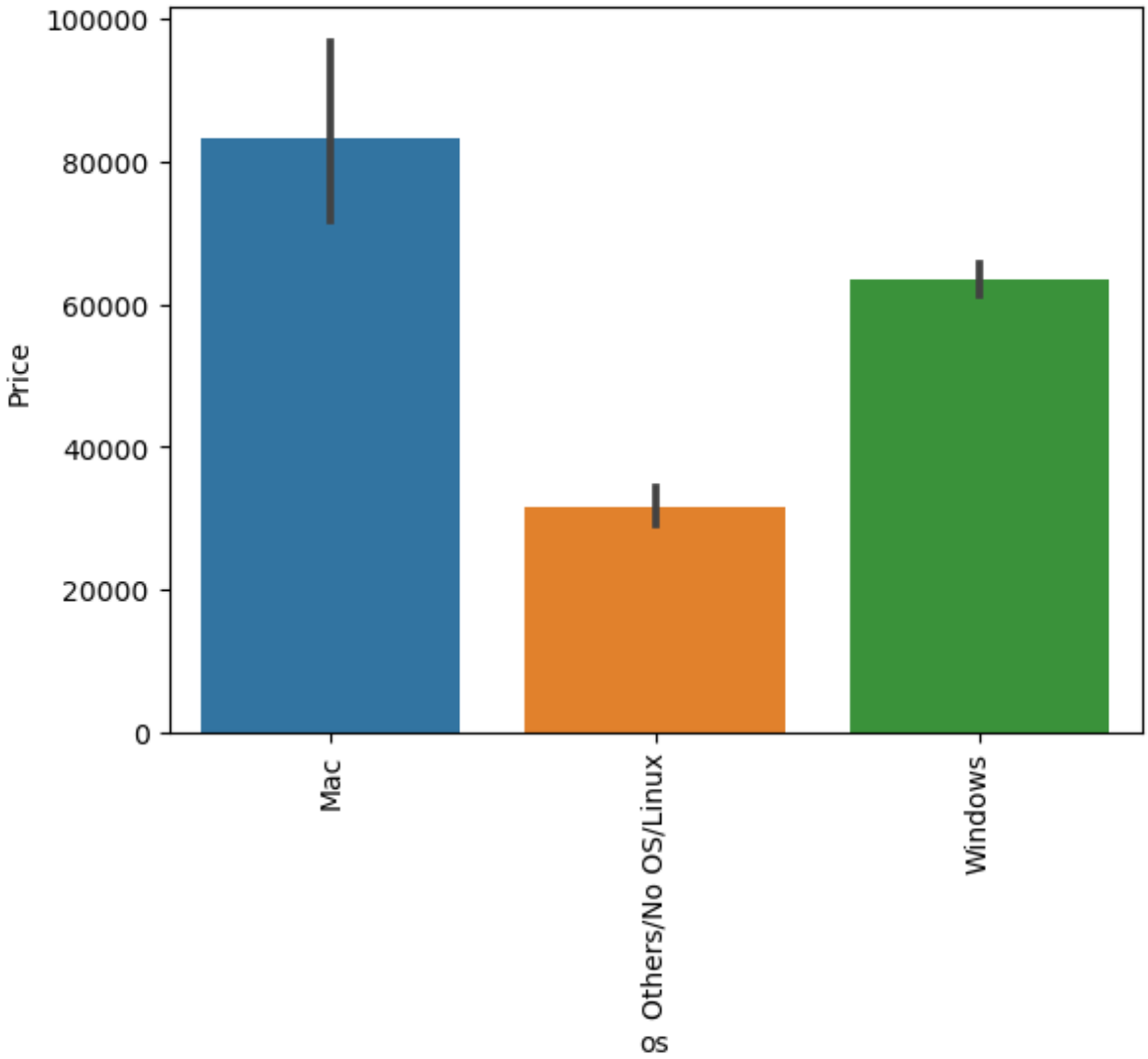
Out[342]:

	Company	TypeName	Ram	OpSys	Weight	Price	TouchScreen	IPS	ppi	Cp bran
0	Apple	Ultrabook	8	macOS	1.37	71378.6832	0	1	226.983005	Int Cor
1	Apple	Ultrabook	8	macOS	1.34	47895.5232	0	0	127.677940	Int Cor
2	HP	Notebook	8	No OS	1.86	30636.0000	0	0	141.211998	Int Cor
3	Apple	Ultrabook	16	macOS	1.83	135195.3360	0	1	220.534624	Int Cor
4	Apple	Ultrabook	8	macOS	1.37	96095.8080	0	1	226.983005	Int Cor

In [343]:
```
df.drop(columns=['OpSys'],inplace=True)
```

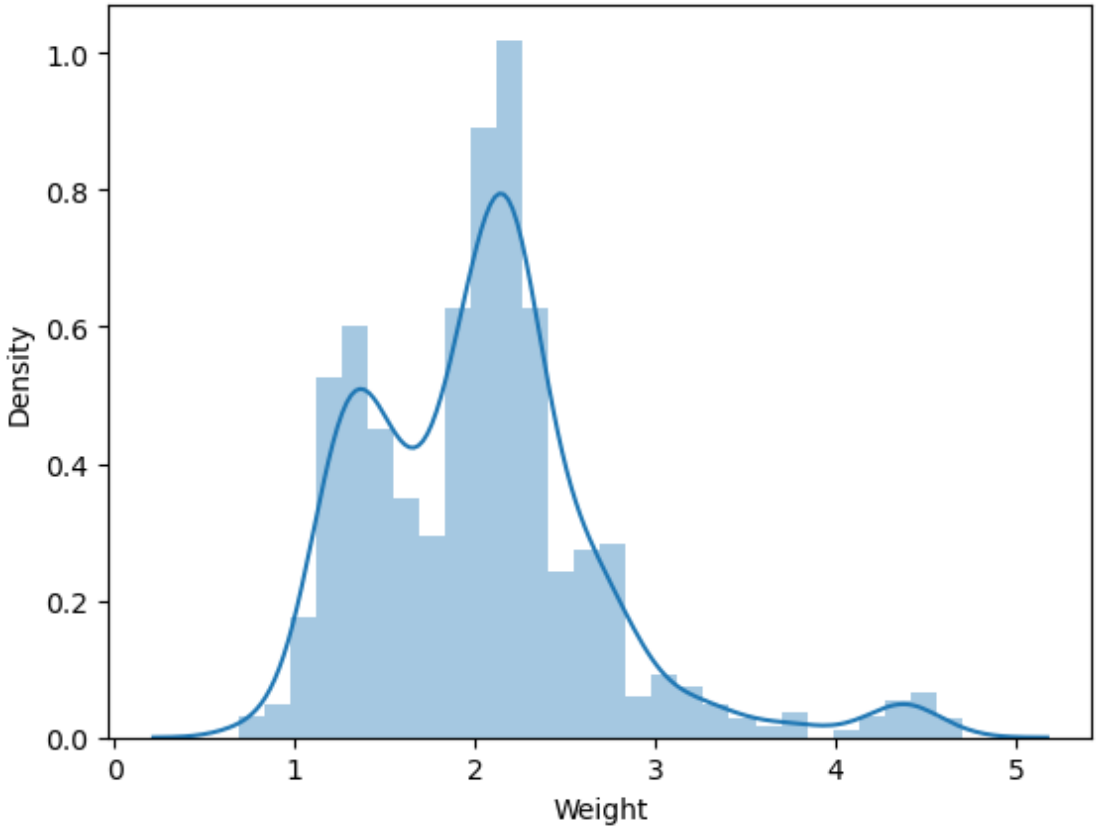
In [344]:

```
sns.barplot(x=df['os'],y=df['Price'])
plt.xticks(rotation='vertical')
plt.show()
```



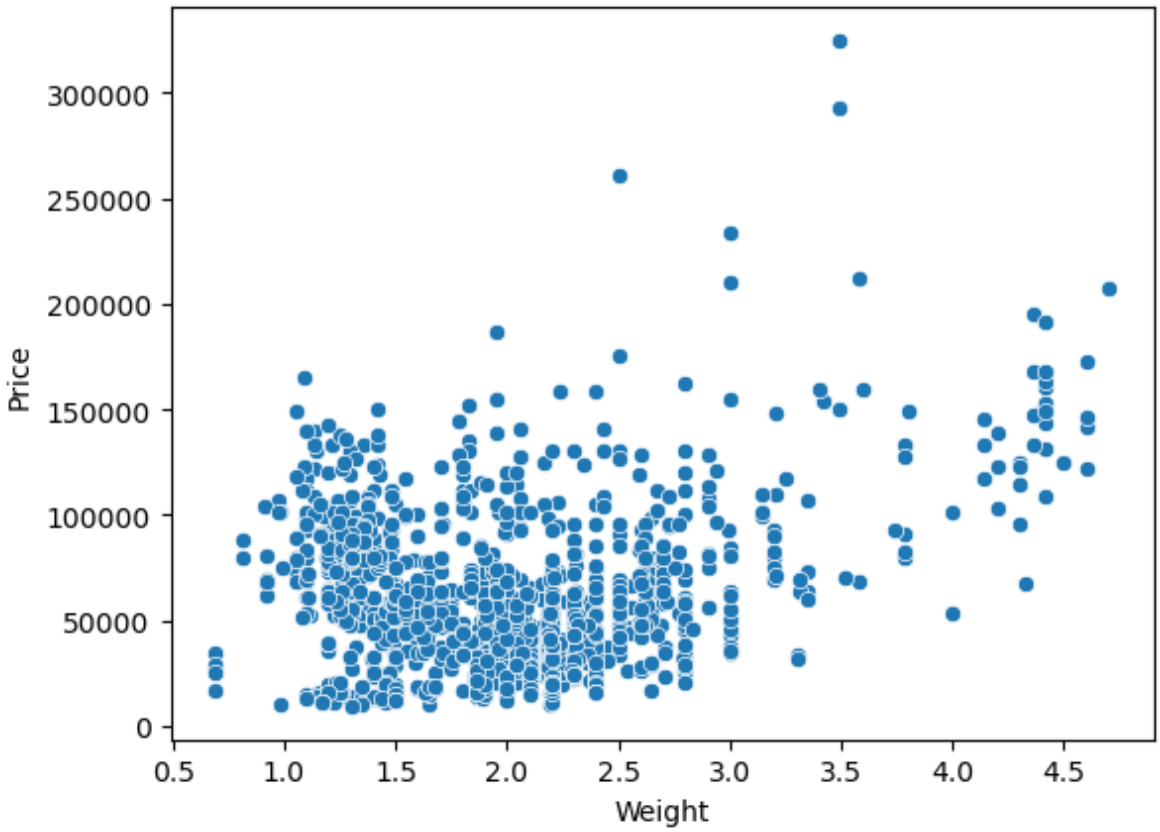
In [345]:
```
sns.distplot(df['Weight'])
```

Out[345]: <AxesSubplot:xlabel='Weight', ylabel='Density'>



In [346]:
```
sns.scatterplot(x=df['Weight'],y=df['Price'])
```

Out[346]: <AxesSubplot:xlabel='Weight', ylabel='Price'>

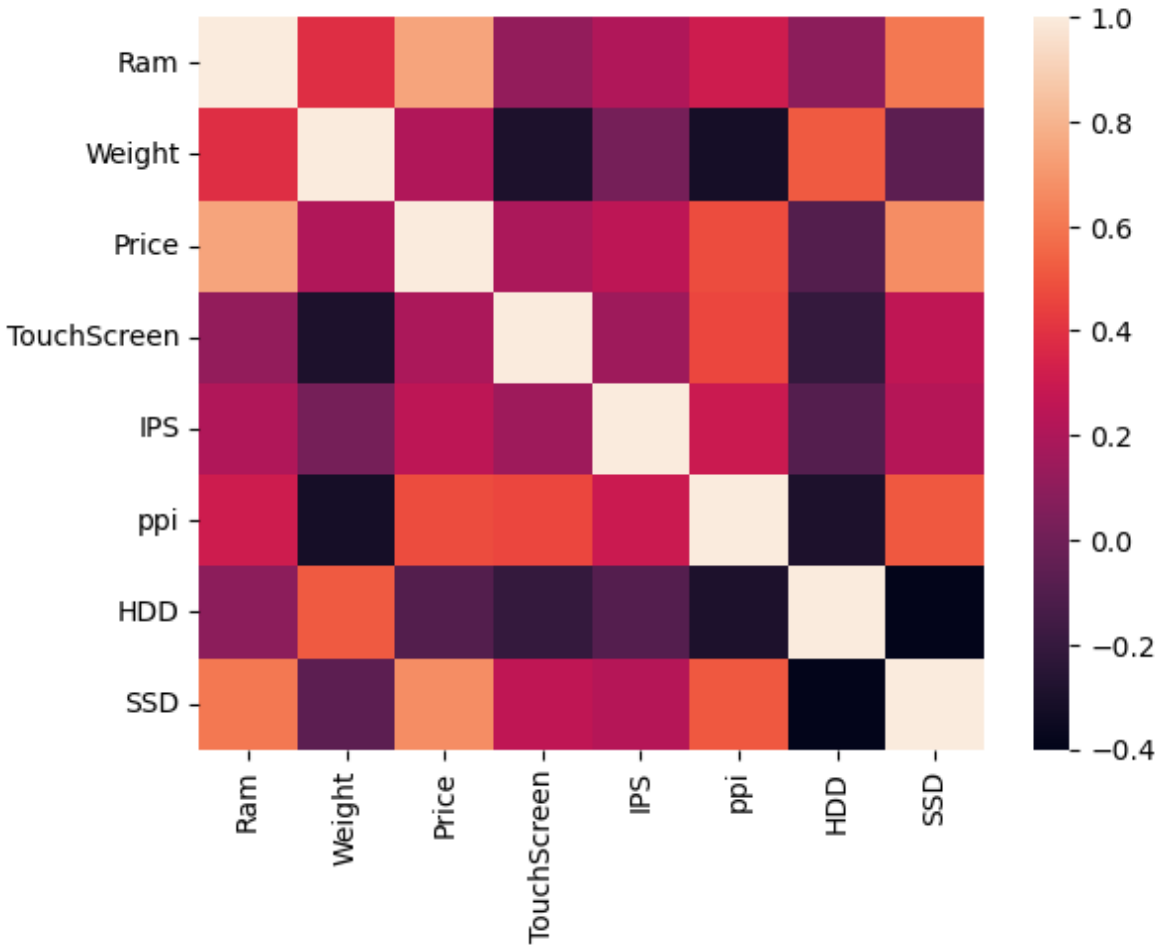


In [347]:
```
df.corr()['Price']
```

Out[347]:
```
Ram            0.742905
Weight         0.209867
Price          1.000000
TouchScreen    0.192917
IPS            0.253320
ppi            0.475368
HDD           -0.096891
SSD            0.670660
Name: Price, dtype: float64
```

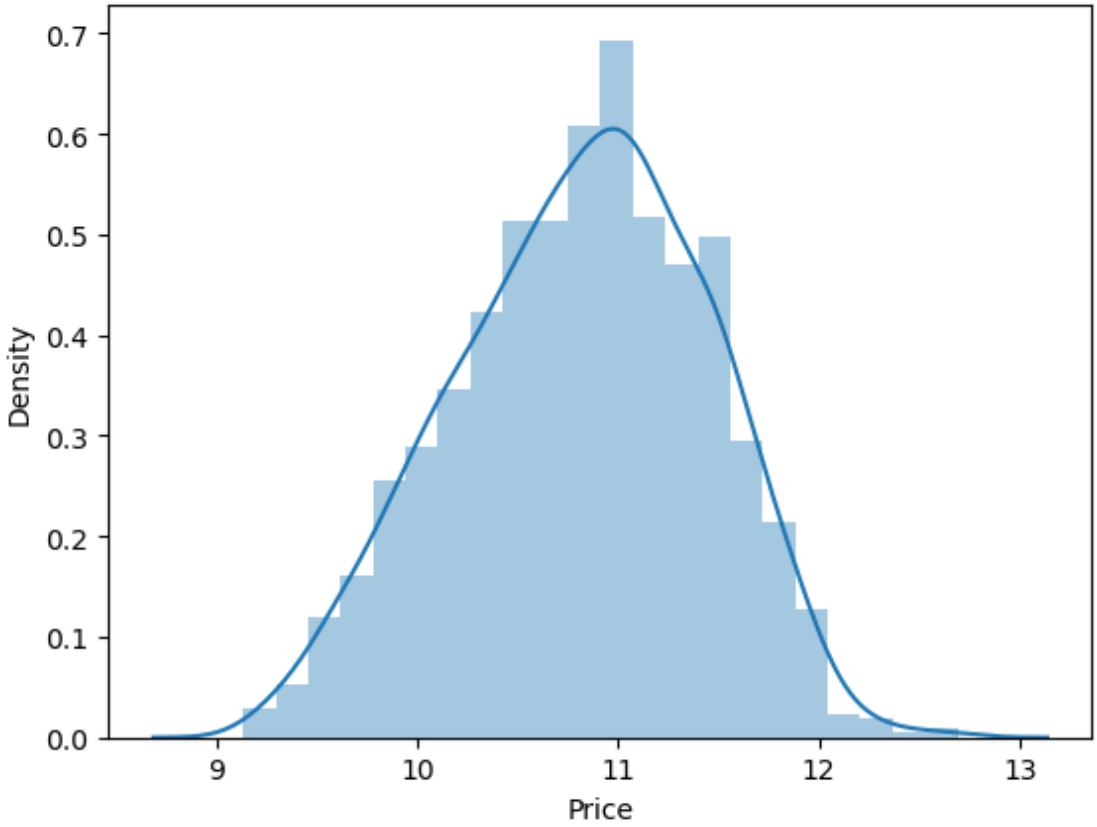
In [348]:
```
sns.heatmap(df.corr())
```

Out[348]:
```
<AxesSubplot:>
```



In [349]:
```
sns.distplot(np.log(df['Price']))
```

Out[349]: <AxesSubplot:xlabel='Price', ylabel='Density'>



Oultier Treatment

In [350]:
```
df.describe(percentiles=[0.01,0.02,0.03,0.05,0.95,0.96,0.97,0.98,0.99]).T
```

Out[350]:

	count	mean	std	min	1%	2%	
Ram	1302.0	8.385561	5.085166	2.000000	2.00000	4.00000	4.0
Weight	1302.0	2.039415	0.665274	0.690000	0.97000	1.08020	1.1
Price	1302.0	59889.058673	37251.183866	9270.720000	12201.12000	13747.30560	14811.3
TouchScreen	1302.0	0.146697	0.353940	0.000000	0.00000	0.00000	0.0
IPS	1302.0	0.279570	0.448960	0.000000	0.00000	0.00000	0.0
ppi	1302.0	146.568497	43.069016	90.583402	100.45467	100.45467	100.4
HDD	1302.0	414.101382	515.889348	0.000000	0.00000	0.00000	0.0
SSD	1302.0	183.874040	186.969314	0.000000	0.00000	0.00000	0.0

As we see there is minimul number of outliers so we are not clipping it

In [351]:
```
df.head()
```

Out[351]:

	Company	TypeName	Ram	Weight	Price	TouchScreen	IPS	ppi	Cpu brand	HDD
0	Apple	Ultrabook	8	1.37	71378.6832	0	1	226.983005	Intel Core i5	0
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
3	Apple	Ultrabook	16	1.83	135195.3360	0	1	220.534624	Intel Core i7	0
4	Apple	Ultrabook	8	1.37	96095.8080	0	1	226.983005	Intel Core i5	0

Encoding Catagorical Columns

In [352]:
```
df['Company'].value_counts()
```

Out[352]:
```
Dell        297
Lenovo      297
HP          274
Asus        158
Acer        103
MSI          54
Toshiba      48
Apple        21
Samsung       8
Razer         7
Mediacom      7
Microsoft     6
Xiaomi        4
Vero          4
Chuwi         3
Google        3
Fujitsu       3
LG            3
Huawei        2
Name: Company, dtype: int64
```

In [353]:
```
cat_cols = df.dtypes[df.dtypes=='object'].index
num_cols = df.dtypes[df.dtypes!='object'].index
print(cat_cols)
print(num_cols)
```

```
Index(['Company', 'TypeName', 'Cpu brand', 'Gpu brand', 'os'], dtype='objec
t')
Index(['Ram', 'Weight', 'Price', 'TouchScreen', 'IPS', 'ppi', 'HDD', 'SSD'],
dtype='object')
```

In [354]:

```
df_dum = pd.get_dummies(df,columns=cat_cols,drop_first=True)
print(df_dum.shape)
print(df_dum.columns)
```

```
(1302, 39)
Index(['Ram', 'Weight', 'Price', 'TouchScreen', 'IPS', 'ppi', 'HDD', 'SSD',
       'Company_Apple', 'Company_Asus', 'Company_Chuwi', 'Company_Dell',
       'Company_Fujitsu', 'Company_Google', 'Company_HP', 'Company_Huawei',
       'Company_LG', 'Company_Lenovo', 'Company_MSI', 'Company_Mediacom',
       'Company_Microsoft', 'Company_Razer', 'Company_Samsung',
       'Company_Toshiba', 'Company_Vero', 'Company_Xiaomi', 'TypeName_Gamin
g',
       'TypeName_Netbook', 'TypeName_Notebook', 'TypeName_Ultrabook',
       'TypeName_Workstation', 'Cpu brand_Intel Core i3',
       'Cpu brand_Intel Core i5', 'Cpu brand_Intel Core i7',
       'Cpu brand_Other Intel Processor', 'Gpu brand_Intel',
       'Gpu brand_Nvidia', 'os_Others/No OS/Linux', 'os_Windows'],
      dtype='object')
```

In [355]:

```
x = df_dum.drop('Price',axis=1)
y = np.log(df_dum['Price'])
print(type(x))
print(type(y))
print(x.shape)
print(y.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.series.Series'>
(1302, 38)
(1302,)
```

In [356]:

```
from sklearn.model_selection import train_test_split
```

In [357]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1041, 38)
(261, 38)
(1041,)
(261,)
```

Linear Regression

In [358]:

```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_scor
```

In [359]:
```
def eval_model(ytest,ypred):
    mae = mean_absolute_error(ytest,ypred)
    mse = mean_squared_error(ytest,ypred)
    rmse = mean_squared_error(ytest,ypred,squared=False)
    r2s = r2_score(ytest,ypred)
    return {'MAE':mae,'MSE':mse,'RMSE':rmse}

def model_res(model,x_train,x_test,y_train,y_test,ypred,mname):
    train_r2 = model.score(x_train,y_train)
    test_r2 = model.score(x_test,y_test)
    w = eval_model(y_test,ypred)
    res_metrics = {'Train_R2':train_r2,'Test_R2':test_r2,'Test_MSE':w['MSE
                   'Test_RMSE':w['RMSE'],'Test_MAE':w['MAE']}
    res = pd.DataFrame(res_metrics, index=[mname])
    return res, res_metrics
```

In [360]:
```
from sklearn.linear_model import LinearRegression
```

In [361]:
```
lr1 = LinearRegression()
lr1.fit(x_train,y_train)
```

Out[361]:
```
LinearRegression()
```

In [362]:
```
ypred_lr1 =  lr1.predict(x_test)
```

In [363]:
```
lr1_df,lr_res = model_res(lr1,x_train,x_test,y_train,y_test,ypred_lr1,'Lin
lr1_df
```

Out[363]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
LinReg	0.832685	0.81543	0.075368	0.274531	0.213859

Decision Tree Regressor

In [364]:
```
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, AdaBoostRegressor
```

In [365]:
```
dt = DecisionTreeRegressor(max_depth=8,min_samples_split=10,min_samples_le
dt.fit(x_train,y_train)
```

Out[365]:
```
DecisionTreeRegressor(max_depth=8, min_samples_leaf=10, min_samples_split=10)
```

In [366]:
```
ypred_dt = dt.predict(x_test)
```

In [367]:
```
dt_df,dt_res = model_res(dt,x_train,x_test,y_train,y_test,ypred_dt,'DTree_
dt_df
```

Out[367]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
DTree_Reg	0.868534	0.826653	0.070785	0.266054	0.203464

Random Forest Regressor

In [368]:
```
rf = RandomForestRegressor(n_estimators=300,max_depth=10,min_samples_split
rf.fit(x_train,y_train)
```

Out[368]:
```
RandomForestRegressor(max_depth=10, min_samples_split=12, n_estimators=300)
```

In [369]:
```
ypred_rf =  rf.predict(x_test)
```

In [370]:
```
rf_df,rf_res = model_res(rf,x_train,x_test,y_train,y_test,ypred_rf,'RF_Reg
rf_df
```

Out[370]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
RF_Reg	0.927071	0.874928	0.051072	0.225991	0.173465

In [371]:
```
rf2 = RandomForestRegressor(n_estimators=300,max_depth=9,min_samples_split
rf2.fit(x_train,y_train)
```

Out[371]:
```
RandomForestRegressor(max_depth=9, min_samples_split=4, n_estimators=300)
```

In [372]:
```
ypred_rf1 =  rf.predict(x_test)
```

In [373]:
```
rf_df1,rf_re1s = model_res(rf2,x_train,x_test,y_train,y_test,ypred_rf1,'RF_
rf_df1
```

Out[373]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
RF_Reg1	0.939111	0.877364	0.051072	0.225991	0.173465

AdaBoost Regressor

In [374]:
```
ada = AdaBoostRegressor(n_estimators=200,random_state=8)
ada.fit(x_train,y_train)
```

Out[374]:
```
AdaBoostRegressor(n_estimators=200, random_state=8)
```

In [375]:
```
ypred_ada = ada.predict(x_test)
```

In [376]:
```
ada_df,ada_res = model_res(ada,x_train,x_test,y_train,y_test,ypred_ada,'Ad
ada_df
```

Out[376]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
AdaBoost_Reg	0.819083	0.804794	0.079711	0.282331	0.233961

XGBoost Regressor

In [377]:
```
from xgboost import XGBRegressor
```

In [378]:
```
xgb1 = XGBRegressor()
xgb1.fit(x_train,y_train)
```

Out[378]:
```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

In [379]:
```
ypred_xgb1 = xgb1.predict(x_test)
```

In [380]:
```
xgb_df,xgb_res = model_res(xgb1,x_train,x_test,y_train,y_test,ypred_xgb1,'
xgb_df
```

Out[380]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
XGBoost_Reg	0.988499	0.896291	0.042348	0.205787	0.158527

Concating the result

In [381]:
```
all_res = pd.concat([lr1_df,dt_df,rf_df,ada_df,xgb_df,rf_df1])
all_res
```

Out[381]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
LinReg	0.832685	0.815430	0.075368	0.274531	0.213859
DTree_Reg	0.868534	0.826653	0.070785	0.266054	0.203464
RF_Reg	0.927071	0.874928	0.051072	0.225991	0.173465
AdaBoost_Reg	0.819083	0.804794	0.079711	0.282331	0.233961
XGBoost_Reg	0.988499	0.896291	0.042348	0.205787	0.158527
RF_Reg1	0.939111	0.877364	0.051072	0.225991	0.173465

Applying hyperparameter Tuning For Random Forest

In [382]:
```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
```

In [383]:
```
params_rf = {'n_estimators':[200,220,240,260,280,300,320,350,400,450,500],
             'max_depth':[9,10,11,12],
             'min_samples_split':[2,3,4,5]}
```

In [384]:
```
rf_base = RandomForestRegressor(random_state=42)
rs_rf1 = GridSearchCV(estimator=rf_base,param_grid= params_rf,scoring='r2'
rs_rf1.fit(x_train,y_train)
```

Out[384]:
```
GridSearchCV(cv=5, estimator=RandomForestRegressor(random_state=42),
             param_grid={'max_depth': [9, 10, 11, 12],
                         'min_samples_split': [2, 3, 4, 5],
                         'n_estimators': [200, 220, 240, 260, 280, 300, 320,
                                          350, 400, 450, 500]},
             scoring='r2')
```

In [385]:
```
print(rs_rf1.best_estimator_)
print(rs_rf1.best_params_)
print(rs_rf1.best_score_)
```

```
RandomForestRegressor(max_depth=12, n_estimators=300, random_state=42)
{'max_depth': 12, 'min_samples_split': 2, 'n_estimators': 300}
0.8710015012042819
```

In [386]:
```
rf2 = RandomForestRegressor(**rs_rf1.best_params_)
rf2.fit(x_train,y_train)
```

Out[386]:
```
RandomForestRegressor(max_depth=12, n_estimators=300)
```

In [387]:
```
ypred_rf2 =  rf.predict(x_test)
```

In [388]:
```
rf_df2,rf_res2 = model_res(rf2,x_train,x_test,y_train,y_test,ypred_rf1,'RF_
rf_df2
```

Out[388]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
RF_Reg2	0.966966	0.887251	0.051072	0.225991	0.173465

In [389]:
```
all_res = pd.concat([lr1_df,dt_df,rf_df,ada_df,xgb_df,rf_df1,rf_df2])
all_res
```

Out[389]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
LinReg	0.832685	0.815430	0.075368	0.274531	0.213859
DTree_Reg	0.868534	0.826653	0.070785	0.266054	0.203464
RF_Reg	0.927071	0.874928	0.051072	0.225991	0.173465
AdaBoost_Reg	0.819083	0.804794	0.079711	0.282331	0.233961
XGBoost_Reg	0.988499	0.896291	0.042348	0.205787	0.158527
RF_Reg1	0.939111	0.877364	0.051072	0.225991	0.173465
RF_Reg2	0.966966	0.887251	0.051072	0.225991	0.173465

In [390]:

```
actual_ypred_rf1 = ypred_rf1

res_df = pd.DataFrame({'Actual_y_test':y_test,'Pred':actual_ypred_rf1})
res_df.sample(20)
```

Out[390]:

	Actual_y_test	Pred
453	11.179710	10.991067
1056	10.188167	10.420069
455	10.147262	10.017596
230	9.964497	10.153023
385	11.708369	11.451403
56	10.060060	10.276091
1012	11.347050	10.853445
1065	11.260382	11.304921
285	10.466285	10.777419
1224	10.208009	10.227241
236	9.941708	9.963336

This is how we use Machine Learning Algorithm for doing laptop price prediction