In [1]:
```
import pandas as pd
import pickle
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
a=pd.read_csv("C:\\Users\\reshma_koduri\\OneDrive\\Documents\\archive.zip")
a
```

Out[2]:

	Unnamed: 0	Unnamed: 0.1	brand	name	price	spec_rating	processor	CPU	Ram	Ra
0	0	0	HP	Victus 15-fb0157AX Gaming Laptop	49900	73.000000	5th Gen AMD Ryzen 5 5600H	Hexa Core, 12 Threads	8GB	
1	1	1	HP	15s-fq5007TU Laptop	39900	60.000000	12th Gen Intel Core i3 1215U	Hexa Core (2P + 4E), 8 Threads	8GB	
2	2	2	Acer	One 14 Z8-415 Laptop	26990	69.323529	11th Gen Intel Core i3 1115G4	Dual Core, 4 Threads	8GB	
3	3	3	Lenovo	Yoga Slim 6 14IAP8 82WU0095IN Laptop	59729	66.000000	12th Gen Intel Core i5 1240P	12 Cores (4P + 8E), 16 Threads	16GB	I
4	4	4	Apple	MacBook Air 2020 MGND3HN Laptop	69990	69.323529	Apple M1	Octa Core (4P + 4E)	8GB	
...	...	...	...	...	...	...	...	...	...	...
888	926	1015	Asus	Vivobook 15X 2023 K3504VAB-NJ321WS Laptop	44990	69.323529	13th Gen Intel Core i3 1315U	Hexa Core (2P + 4E), 8 Threads	8GB	
889	927	1016	Asus	TUF A15 FA577RM-HQ032WS Laptop	110000	71.000000	6th Gen AMD Ryzen 7 6800H	Octa Core, 16 Threads	16GB	
890	928	1017	Asus	ROG Zephyrus G14 2023 GA402XV-N2034WS Gaming L...	189990	89.000000	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	32GB	
891	929	1018	Asus	TUF Gaming F15 2023 FX507VU-LP083WS Gaming Laptop	129990	73.000000	13th Gen Intel Core i7 13700H	14 Cores (6P + 8E), 20 Threads	16GB	

	Unnamed: 0	Unnamed: 0.1	brand	name	price	spec_rating	processor	CPU	Ram	Ra
892	930	1019	Asus	TUF Gaming A15 2023 FA577XU-LP041WS Gaming Laptop	131990	84.000000	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	16GB	

893 rows × 18 columns

In [3]:
```
a.head(5)
```

Out[3]:

	Unnamed: 0	Unnamed: 0.1	brand	name	price	spec_rating	processor	CPU	Ram	Ram_t
0	0	0	HP	Victus 15-fb0157AX Gaming Laptop	49900	73.000000	5th Gen AMD Ryzen 5 5600H	Hexa Core, 12 Threads	8GB	DI
1	1	1	HP	15s-fq5007TU Laptop	39900	60.000000	12th Gen Intel Core i3 1215U	Hexa Core (2P + 4E), 8 Threads	8GB	DI
2	2	2	Acer	One 14 Z8-415 Laptop	26990	69.323529	11th Gen Intel Core i3 1115G4	Dual Core, 4 Threads	8GB	DI
3	3	3	Lenovo	Yoga Slim 6 14IAP8 82WU0095IN Laptop	59729	66.000000	12th Gen Intel Core i5 1240P	12 Cores (4P + 8E), 16 Threads	16GB	LPDI
4	4	4	Apple	MacBook Air 2020 MGND3HN Laptop	69990	69.323529	Apple M1	Octa Core (4P + 4E)	8GB	DI

In [4]:
```
a.tail(5)
```

Out[4]:

	Unnamed: 0	Unnamed: 0.1	brand	name	price	spec_rating	processor	CPU	Ram	Ram_
888	926	1015	Asus	Vivobook 15X 2023 K3504VAB-NJ321WS Laptop	44990	69.323529	13th Gen Intel Core i3 1315U	Hexa Core (2P + 4E), 8 Threads	8GB	D
889	927	1016	Asus	TUF A15 FA577RM-HQ032WS Laptop	110000	71.000000	6th Gen AMD Ryzen 7 6800H	Octa Core, 16 Threads	16GB	

	Unnamed: 0	Unnamed: 0.1	brand	name	price	spec_rating	processor	CPU	Ram	Ram_
890	928	1017	Asus	ROG Zephyrus G14 2023 GA402XV-N2034WS Gaming L...	189990	89.000000	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	32GB	D
891	929	1018	Asus	TUF Gaming F15 2023 FX507VU-LP083WS Gaming Laptop	129990	73.000000	13th Gen Intel Core i7 13700H	14 Cores (6P + 8E), 20 Threads	16GB	D
892	930	1019	Asus	TUF Gaming A15 2023 FA577XU-LP041WS Gaming Laptop	131990	84.000000	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	16GB	D

In [5]:
```
list(a)
```

Out[5]:
```
['Unnamed: 0',
 'Unnamed: 0.1',
 'brand',
 'name',
 'price',
 'spec_rating',
 'processor',
 'CPU',
 'Ram',
 'Ram_type',
 'ROM',
 'ROM_type',
 'GPU',
 'display_size',
 'resolution_width',
 'resolution_height',
 'OS',
 'warranty']
```

In [6]:
```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 893 entries, 0 to 892
Data columns (total 18 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Unnamed: 0         893 non-null    int64
 1   Unnamed: 0.1       893 non-null    int64
 2   brand              893 non-null    object
 3   name               893 non-null    object
 4   price              893 non-null    int64
 5   spec_rating        893 non-null    float64
 6   processor          893 non-null    object
```

```
 7   CPU                893 non-null    object
 8   Ram                893 non-null    object
 9   Ram_type           893 non-null    object
 10  ROM                893 non-null    object
 11  ROM_type           893 non-null    object
 12  GPU                893 non-null    object
 13  display_size       893 non-null    float64
 14  resolution_width   893 non-null    float64
 15  resolution_height  893 non-null    float64
 16  OS                 893 non-null    object
 17  warranty           893 non-null    int64
dtypes: float64(4), int64(4), object(10)
memory usage: 125.7+ KB
```

In [7]:
```
a.describe()
```

Out[7]:

	Unnamed: 0	Unnamed: 0.1	price	spec_rating	display_size	resolution_width	resolution
count	893.000000	893.000000	893.000000	893.000000	893.000000	893.000000	893.
mean	467.135498	521.382979	79907.409854	69.379026	15.173751	2035.393057	1218.
std	270.209769	299.916605	60880.043823	5.541555	0.939095	426.076009	326.
min	0.000000	0.000000	9999.000000	60.000000	11.600000	1080.000000	768.
25%	235.000000	265.000000	44500.000000	66.000000	14.000000	1920.000000	1080.
50%	467.000000	531.000000	61990.000000	69.323529	15.600000	1920.000000	1080.
75%	702.000000	784.000000	90990.000000	71.000000	15.600000	1920.000000	1200.
max	930.000000	1019.000000	450039.000000	89.000000	18.000000	3840.000000	3456.

In [8]:
```
a.groupby("price").count()
```

Out[8]:

price	Unnamed: 0	Unnamed: 0.1	brand	name	spec_rating	processor	CPU	Ram	Ram_type	ROM	I
9999	1	1	1	1	1	1	1	1	1	1	
10990	3	3	3	3	3	3	3	3	3	3	
12990	1	1	1	1	1	1	1	1	1	1	
13990	1	1	1	1	1	1	1	1	1	1	
14490	1	1	1	1	1	1	1	1	1	1	
...	...	...	...	...	...	...	...	...	...	...	
415000	1	1	1	1	1	1	1	1	1	1	
419990	1	1	1	1	1	1	1	1	1	1	
420000	1	1	1	1	1	1	1	1	1	1	
429990	1	1	1	1	1	1	1	1	1	1	
450039	1	1	1	1	1	1	1	1	1	1	

464 rows × 17 columns

In [9]:
```
b=a.drop(['display_size','resolution_width','resolution_height','spec_rating'],axis=
b
```

Out[9]:

	Unnamed: 0	Unnamed: 0.1	brand	name	price	processor	CPU	Ram	Ram_type	RO
0	0	0	HP	Victus 15-fb0157AX Gaming Laptop	49900	5th Gen AMD Ryzen 5 5600H	Hexa Core, 12 Threads	8GB	DDR4	512(
1	1	1	HP	15s-fq5007TU Laptop	39900	12th Gen Intel Core i3 1215U	Hexa Core (2P + 4E), 8 Threads	8GB	DDR4	512(
2	2	2	Acer	One 14 Z8-415 Laptop	26990	11th Gen Intel Core i3 1115G4	Dual Core, 4 Threads	8GB	DDR4	512(
3	3	3	Lenovo	Yoga Slim 6 14IAP8 82WU0095IN Laptop	59729	12th Gen Intel Core i5 1240P	12 Cores (4P + 8E), 16 Threads	16GB	LPDDR5	512(
4	4	4	Apple	MacBook Air 2020 MGND3HN Laptop	69990	Apple M1	Octa Core (4P + 4E)	8GB	DDR4	256(
...	...	...	...	...	...	...	...	...	...	
888	926	1015	Asus	Vivobook 15X 2023 K3504VAB-NJ321WS Laptop	44990	13th Gen Intel Core i3 1315U	Hexa Core (2P + 4E), 8 Threads	8GB	DDR4	512(
889	927	1016	Asus	TUF A15 FA577RM-HQ032WS Laptop	110000	6th Gen AMD Ryzen 7 6800H	Octa Core, 16 Threads	16GB	DDR	1
890	928	1017	Asus	ROG Zephyrus G14 2023 GA402XV-N2034WS Gaming L...	189990	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	32GB	DDR5	1
891	929	1018	Asus	TUF Gaming F15 2023 FX507VU-LP083WS Gaming Laptop	129990	13th Gen Intel Core i7 13700H	14 Cores (6P + 8E), 20 Threads	16GB	DDR4	512(
892	930	1019	Asus	TUF Gaming A15 2023 FA577XU-LP041WS	131990	7th Gen AMD Ryzen 9 7940HS	Octa Core, 16 Threads	16GB	DDR4	1

	Unnamed: 0	Unnamed: 0.1	brand	name	price	processor	CPU	Ram	Ram_type	RO
				Gaming Laptop						

893 rows × 14 columns

In [10]:
```
c=pd.get_dummies(b,dtype=int)
c
```

Out[10]:

	Unnamed: 0	Unnamed: 0.1	price	warranty	brand_AXL	brand_Acer	brand_Apple	brand_Asus	br
0	0	0	49900	1	0	0	0	0	
1	1	1	39900	1	0	0	0	0	
2	2	2	26990	1	0	1	0	0	
3	3	3	59729	1	0	0	0	0	
4	4	4	69990	1	0	0	1	0	
...	...	...	...	...	...	...	...	...	
888	926	1015	44990	1	0	0	0	1	
889	927	1016	110000	1	0	0	0	1	
890	928	1017	189990	1	0	0	0	1	
891	929	1018	129990	1	0	0	0	1	
892	930	1019	131990	1	0	0	0	1	

893 rows × 1238 columns

In [11]:
```
y=c['price']
```

In [12]:
```
x=c.drop(['price'],axis=1)
```

In [13]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=55)
```

In [14]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(x_train,y_train)
```

Out[14]:
```
LinearRegression()
```

In [15]:
```
ypred=reg.predict(x_test)
ypred
```

Out[15]:
```
array([ 28132.00598808, 110292.78535286,  90834.24715621, 208929.24340991,
        51787.19210377,  24132.79400868, 202241.51972906, 111266.82518015,
        54178.63096428,  46323.44665505,  56288.58898272,  66202.04196873,
         3066.86482515, 165197.25671488,  61198.62900546,  98019.73227747,
        39901.8696003 ,  53861.33546167,  69179.43656734, 142312.57967178,
        -1082.79170485,  70848.37251907, 112110.02074327,  52925.32345885,
        50031.95703246,  53517.84272618,  84125.94614661,  59112.47734245,
        43712.67050279, 137886.02607223,  17404.92515154,  49715.68782386,
        70121.37514786,  81073.91712058,  52394.60606303,  65535.74852672,
        54054.44951777,  67280.91694053, 124997.02576278,  34150.44361406,
        87077.37961557,  64070.84821082, 123042.50382713,  86974.58241405,
        52763.34309068, 260459.02857622,  68459.41267198,  59173.47948846,
        41218.82383275,  34937.16821978,  48932.97664152,  87883.13825814,
        43853.55441921,  53374.6768221 ,  28668.35142316,  55516.90306811,
        75847.37878028, 124562.8800278 ,  59080.62536777,  66939.98645773,
        75353.54012359,  47096.41583122,  28554.36314068,  77415.74714013,
       172422.1143808 ,  14303.91855783,  94794.11351621,  85959.68928383,
        52893.31858749,  44985.61624093,  10273.45819188,  40618.75252649,
        83114.52113453,  19600.71588911,  79497.16095512,  72987.04143631,
       237781.70582374,  43873.25578464, 343934.87297662,  65960.27309083,
        60736.21436804,  85694.96524254,  31103.13893843,  44540.42726717,
        48336.45750524,  56314.55596484,  28472.12587733,  36457.8048238 ,
        63108.40654248, 293861.26880784, 121459.38166542,  45962.36483497,
        40090.16334734,  85689.4534275 ,  73764.84292612,  67081.54850995,
        40864.8773914 , 143953.98296551,  50067.1489263 ,  41894.68776821,
        49797.73663128,  42247.15958147, 196317.1831246 ,  39424.94801009,
       101347.00334553,  45999.1334568 , 260910.62597445,  76302.95539177,
       117602.58927196,  66345.54983699,  73001.18799567,  58164.22972429,
        41164.20831017,  74600.60848368,  87432.91061002, 152444.90422117,
        80946.10555016,  92962.37494628, 108984.68672901,  81307.56288625,
        66412.92320009,  56352.46147466,  49976.62761513,  56141.42460189,
        99492.64424418,  36665.55565245, 111656.47653427, 101126.32391898,
        97928.77268503,  76209.29606024,  40293.22160235,  28549.07364347,
       159430.1810741 ,  38297.60320317,  41480.80150595,  88490.34467643,
       103189.67929249,  44721.28385474,  80700.54129271,  60483.54038935,
        47357.09865763, 130856.36017646,  32914.6595472 , 150020.70713139,
        80635.45891963, 283234.117992  ,  77287.9426144 ,  58237.62921917,
        40468.80744528,  30807.62666749, 189287.33477421, 137373.69291986,
        55913.93698884,  59264.50675703,  75672.79175782,  64540.55556984,
        37687.63503832,  40496.30114554,  61581.13650806, 128609.98418775,
        39842.6867437 ,  39007.48875492,  43012.23863178,  26374.28333539,
        56513.88643577,  46657.96725083,  65053.25955189,  78738.71066227,
       101643.53886729,  97327.23118364,  58562.51252863,  92138.37936862,
        61146.32062739,  25569.91904842,  79949.92836007,  43222.52012606,
       134248.53866079,  86185.71747835, 222456.57540857,  56443.30335792,
        64645.12023864,  55935.13321761,  32355.07666906, 197402.46416748,
       147219.72113207,  16486.71951524,  70370.99255915, 110543.44889706,
        39769.32546465,  74269.11769288,  67782.35668783,  74701.54406413,
        90414.07458287,  31837.96957666,  90146.77061391, 133250.24933139,
        10904.14732691,  62045.6173818 ,  41069.34160105,  37059.91178059,
        48808.50465076,  74557.67940014,  45044.11769986, 202289.77838706,
       136481.19361016,  68903.90375904,  45958.35413492,  53691.43832757,
       276132.54482197, 180009.29806122,  35168.23885576, 101645.04396193,
        50954.17770129,  49478.16616432, 130813.98093707,  22753.94903588,
        47024.2475479 , 114800.66959605,  70960.52873277, 116726.40075589,
        73179.96356473, 116034.39584361,  78674.85136918,  91767.13252594,
        59788.68267626, 101311.58675821,  57753.70702151, 115774.72018021,
       103181.06678031,  33038.25865801,  39480.39081545,  84749.34390235,
        74737.67069511,  35075.65923565,  55873.31554727,   -780.02395034,
        41688.3692299 ,  67857.66785261,  34582.21588873,  73464.16314608,
        61182.74605727,  44685.56956333,  50933.99629889,  45051.37065528,
        92891.2294081 ,  80451.85953639, 191677.68565659,  72454.55235197,
        96521.99703476,  45554.73312118,  37980.05960795,  80360.93258818,
        54288.0960399 ,  61777.84671081, 147789.10947701,  41084.78247088,
```

```
148103.34231419,  83561.41671168,  32261.37133033,  65085.25749226,
122074.78611998,  39713.41827835, 118237.28964595,  62745.5709075 ,
 40724.50460297,  53786.45821639,  23623.62784374,  38511.53268761,
 43850.23146258,  57689.33939898,  55388.53317275,  11037.05012185,
107736.22698251,  39967.96418589,  59027.516862  , 128519.87292693,
 56487.27170534,  96833.04764575,  41884.64600975,  41451.85362119,
 41570.61485584,  98464.77572239,  65263.20120784,  74321.09461157,
 54931.77814071,  28556.24893098,  41749.69263359,  65863.93243478,
102843.13685173,  40964.56862779,  65444.65286807, 115124.36959219,
159776.01415013,  59122.66721916, 112805.14497998])
```

In [16]:
```
from sklearn.metrics import r2_score
r2_score(ypred,y_test)
```

Out[16]:
```
0.6745230741872765
```

In [17]:
```
from sklearn.metrics import mean_squared_error
mean_squared_error(y_test,ypred)
```

Out[17]:
```
838379094.2015575
```

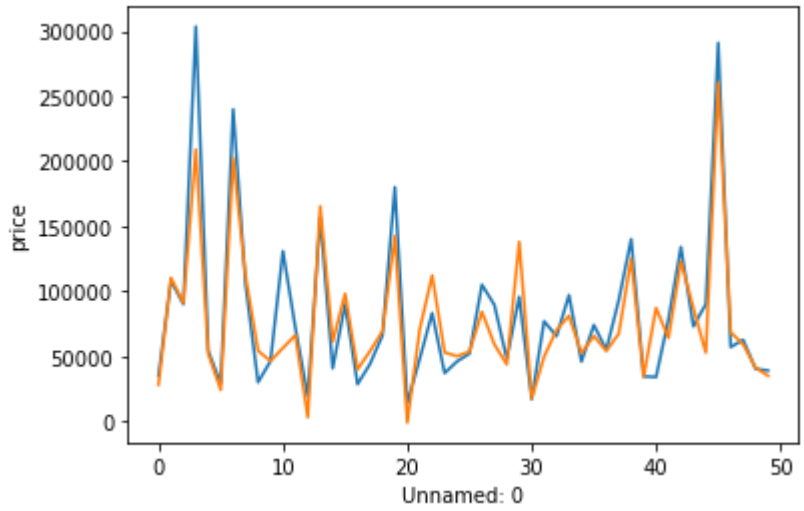
In [18]:
```
results=pd.DataFrame(columns=['price','Predicted'])
results['price']=y_test
results["Predicted"]=ypred
results=results.reset_index()
results['Unnamed: 0']=results.index
results.head(5)
```

Out[18]:

	index	price	Predicted	Unnamed: 0
0	611	34999	28132.005988	0
1	668	107990	110292.785353	1
2	245	89990	90834.247156	2
3	821	303490	208929.243410	3
4	604	53980	51787.192104	4

In [19]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.lineplot(x='Unnamed: 0',y='price',data=results.head(50))
sns.lineplot(x='Unnamed: 0',y='Predicted',data=results.head(50))
plt.plot()
```

Out[19]:
```
[]
```



In [20]:
```
cor=c.corr()
cor
```

Out[20]:

	Unnamed: 0	Unnamed: 0.1	price	warranty	brand_AXL	brand_Acer	brand_Apple	bra
Unnamed: 0	1.000000	0.999665	0.162473	0.157482	0.071466	-0.049608	-0.066338	-0
Unnamed: 0.1	0.999665	1.000000	0.162619	0.158614	0.069722	-0.049247	-0.065033	-0
price	0.162473	0.162619	1.000000	0.117101	-0.050938	-0.112569	0.209386	-0
warranty	0.157482	0.158614	0.117101	1.000000	-0.011528	-0.078402	-0.084532	-0
brand_AXL	0.071466	0.069722	-0.050938	-0.011528	1.000000	-0.015267	-0.006399	-0
...	...	...	...	...	...	...	...	
OS_Windows 10 OS	0.012044	0.013284	-0.029673	-0.025893	-0.005042	-0.034291	-0.014374	-0
OS_Windows 10 OS	0.012544	0.016962	-0.034228	0.054543	-0.008524	-0.035963	-0.024301	-0
OS_Windows 11 OS	0.094991	0.095620	0.041161	0.128195	-0.006193	-0.042118	-0.017655	0
OS_Windows 11 OS	-0.058015	-0.062330	0.016752	-0.012201	0.017850	0.051639	-0.358511	0
OS_Windows OS	0.021454	0.023397	-0.026647	-0.024550	-0.004780	0.044297	-0.013629	0

1238 rows × 1238 columns

In [ ]:

In [ ]: