In [1]:
```python
import numpy as np
import pandas as pd
import math

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import mean_absolute_error, r2_score, mean_squared_error
from sklearn.metrics import accuracy_score
```

In [2]:
```python
df = pd.read_csv('https://raw.githubusercontent.com/amankharwal/Website-data/master/CarPrice.csv')
df.head()
```

Out[2]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio	stroke	compressionratio	hor
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	

5 rows × 26 columns

In [3]:
```python
df.dtypes
```

Out[3]:
```
car_ID                int64
symboling             int64
CarName              object
fueltype             object
aspiration           object
doornumber           object
carbody              object
drivewheel           object
enginelocation       object
wheelbase           float64
carlength           float64
carwidth            float64
carheight           float64
curbweight            int64
enginetype           object
cylindernumber       object
enginesize            int64
fuelsystem           object
boreratio           float64
stroke              float64
compressionratio    float64
horsepower            int64
peakrpm               int64
citympg               int64
highwaympg            int64
price               float64
dtype: object
```

In [4]:
```python
df.shape
```

Out[4]:
```
(205, 26)
```

In [5]:
```python
df.isnull().sum()
```

Out[5]:
```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
dtype: int64
```

In [6]:
```python
df.duplicated().sum()
```

Out[6]:
```
0
```

In [7]:
```python
df.drop(columns=['car_ID'],axis=1,inplace=True)
```

In [8]:
```python
df
```

Out[8]:

	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	...	enginesize	fuelsystem	boreratio	stroke	compressionratio
0	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	168.8	...	130	mpfi	3.47	2.68	9.0
1	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	168.8	...	130	mpfi	3.47	2.68	9.0
2	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	171.2	...	152	mpfi	2.68	3.47	9.0
3	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	176.6	...	109	mpfi	3.19	3.40	10.0
4	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	176.6	...	136	mpfi	3.19	3.40	8.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
200	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	front	109.1	188.8	...	141	mpfi	3.78	3.15	9.5
201	-1	volvo 144ea	gas	turbo	four	sedan	rwd	front	109.1	188.8	...	141	mpfi	3.78	3.15	8.7
202	-1	volvo 244dl	gas	std	four	sedan	rwd	front	109.1	188.8	...	173	mpfi	3.58	2.87	8.8
203	-1	volvo 246	diesel	turbo	four	sedan	rwd	front	109.1	188.8	...	145	idi	3.01	3.40	23.0
204	-1	volvo 264gl	gas	turbo	four	sedan	rwd	front	109.1	188.8	...	141	mpfi	3.78	3.15	9.5

205 rows × 25 columns

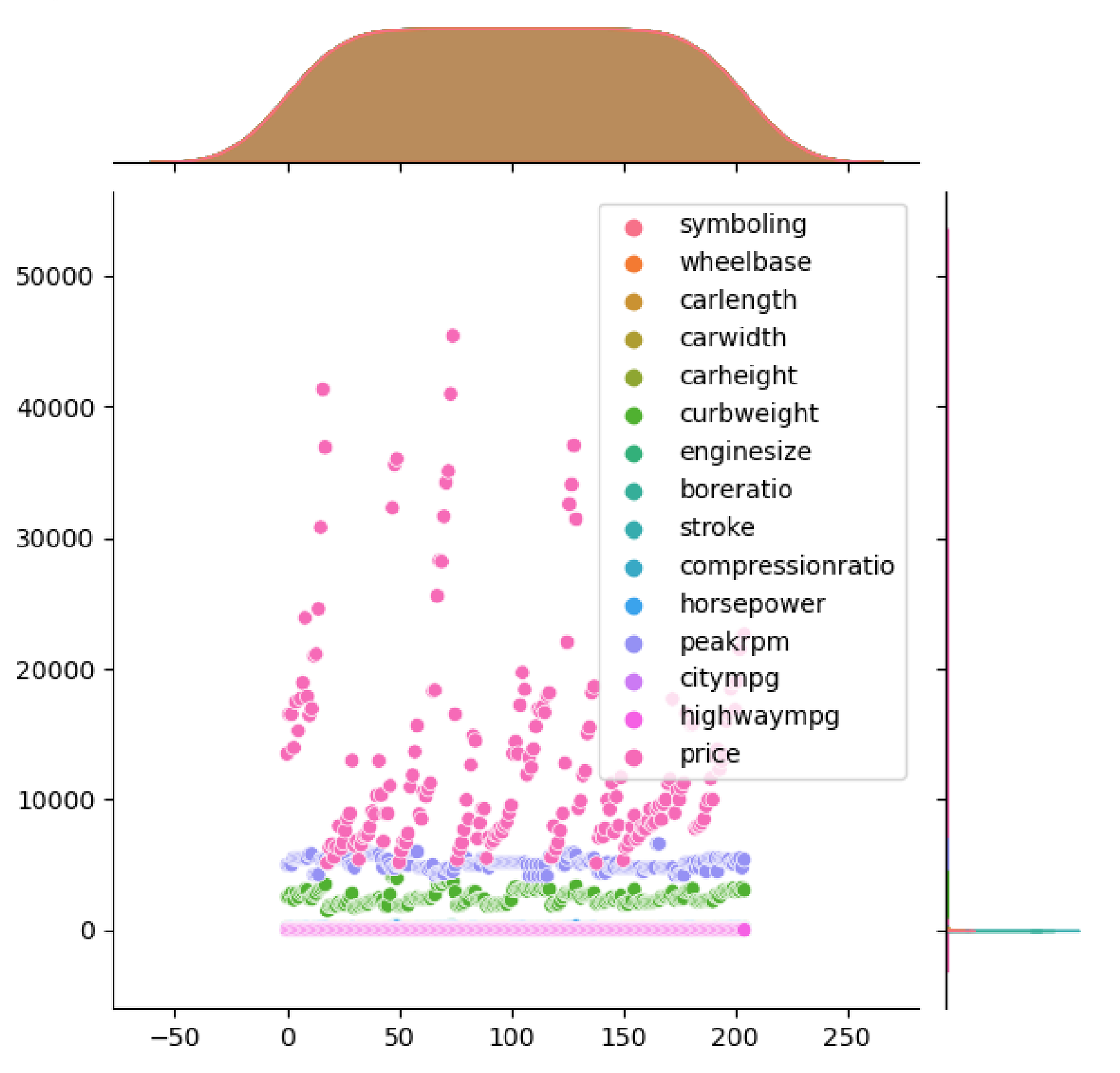
In [9]:
```python
df.describe()
```

Out[9]:

	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	highwaympg
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854	126.907317	3.329756	3.255415	10.142537	104.117073	5125.121951	25.219512	30.751220
std	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	0.270844	0.313597	3.972040	39.544167	476.985643	6.542142	6.886443
min	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000	7.000000	48.000000	4150.000000	13.000000	16.000000
25%	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	3.150000	3.110000	8.600000	70.000000	4800.000000	19.000000	25.000000
50%	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000	9.000000	95.000000	5200.000000	24.000000	30.000000
75%	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	3.580000	3.410000	9.400000	116.000000	5500.000000	30.000000	34.000000
max	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	3.940000	4.170000	23.000000	288.000000	6600.000000	49.000000	54.000000

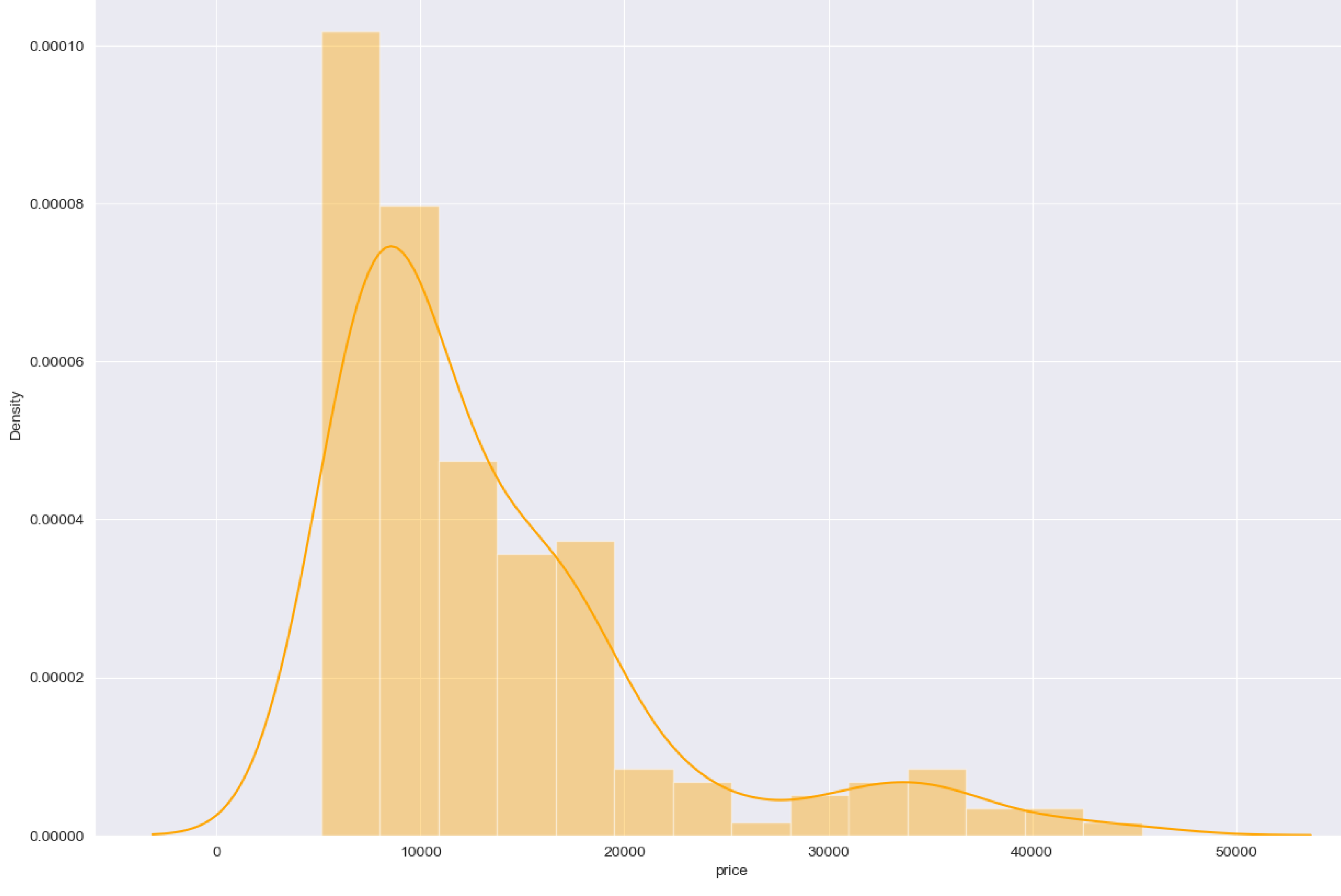
In [10]:
```python
plt.figure(figsize = (10, 10))
sns.jointplot(data = df)
plt.show()
```

```
<Figure size 1000x1000 with 0 Axes>
```



In [13]:
```python
sns.set_style('darkgrid')
plt.figure(figsize=(15, 10))
sns.distplot(df.price , color ="Orange")
plt.show()
```

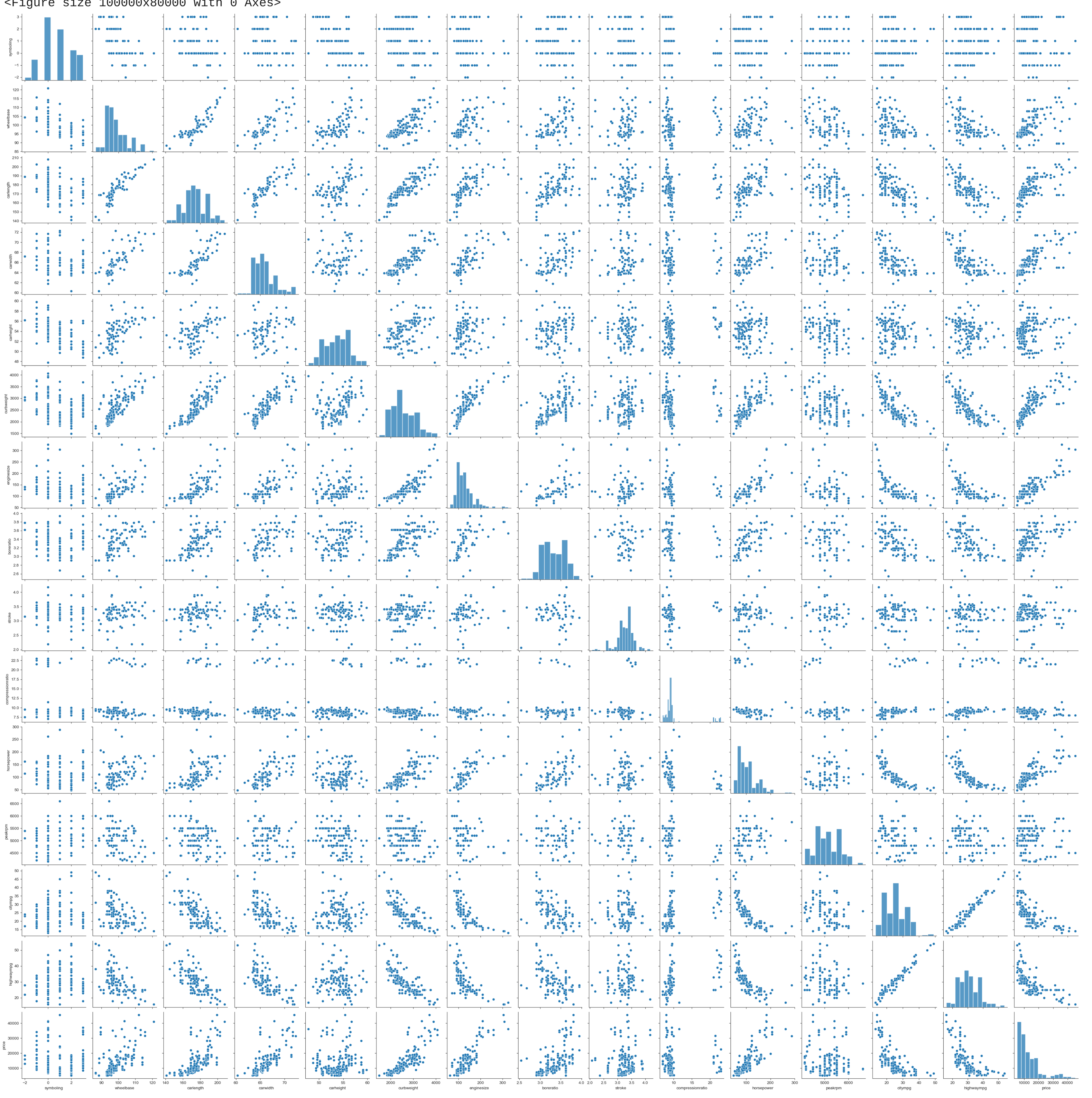
```
C:\Users\KIIT\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level fu
nction for histograms).
  warnings.warn(msg, FutureWarning)
```



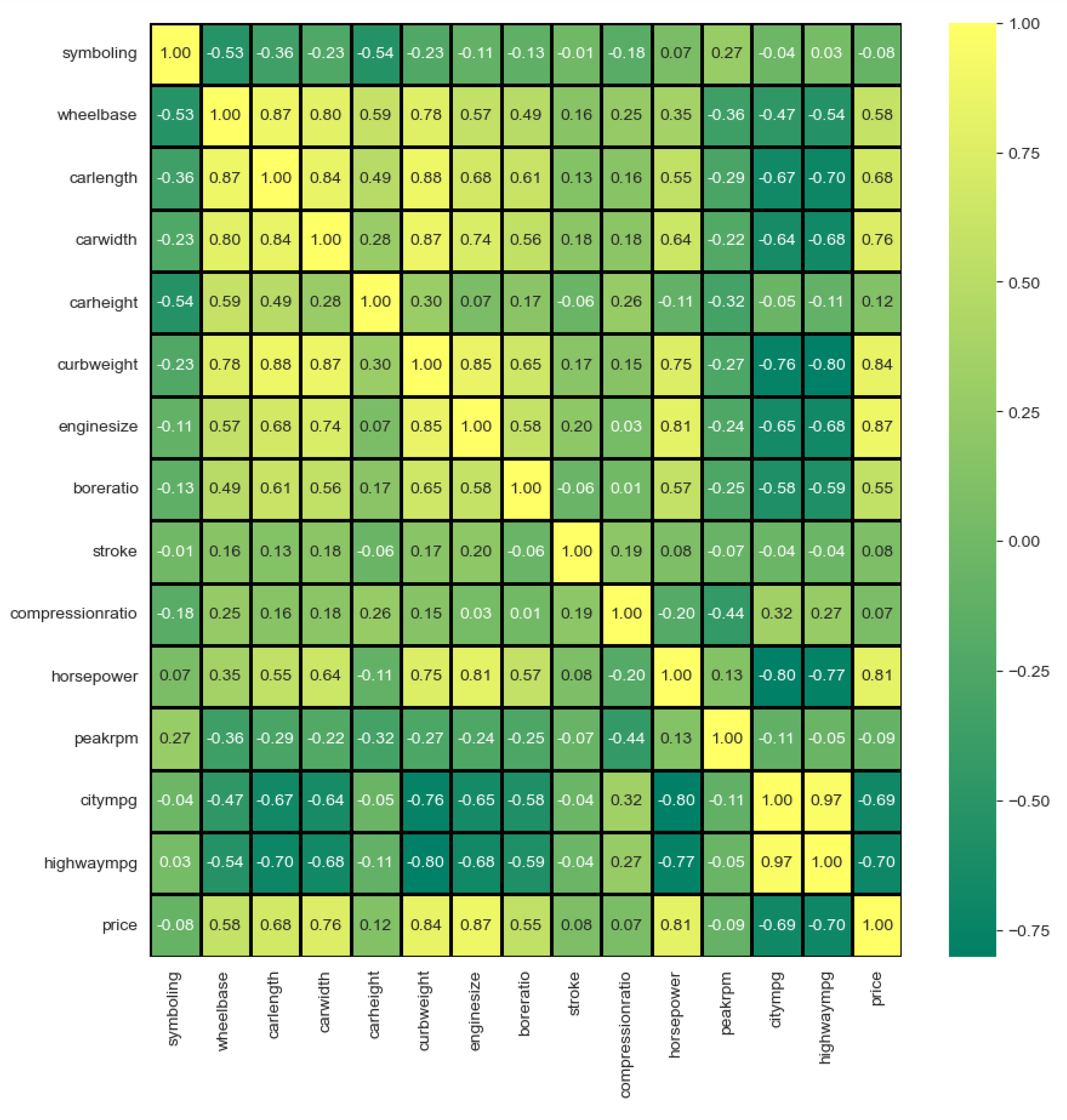
In [18]:
```python
plt.figure(figsize=(1000,800))
sns.set_style('ticks')
sns.pairplot(df)
```

Out[18]:
```
<seaborn.axisgrid.PairGrid at 0x1c153b27b20>

<Figure size 100000x80000 with 0 Axes>
```



In [28]:
```python
plt.figure(figsize = (10,10))
sns.heatmap(df.corr(), annot = True, cmap = 'summer',linecolor='black',linewidth=1.0,fmt='0.2f')
plt.show()
```



In [32]:
```python
data = df[["symboling", "wheelbase", "carlength", "carwidth", "carheight", "curbweight", "enginesize", "boreratio", "stroke", "compressionratio", "h
data
```

Out[32]:

	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	highwaympg	price
0	3	88.6	168.8	64.1	48.8	2548	130	3.47	2.68	9.0	111	5000	21	27	13495.0
1	3	88.6	168.8	64.1	48.8	2548	130	3.47	2.68	9.0	111	5000	21	27	16500.0
2	1	94.5	171.2	65.5	52.4	2823	152	2.68	3.47	9.0	154	5000	19	26	16500.0
3	2	99.8	176.6	66.2	54.3	2337	109	3.19	3.40	10.0	102	5500	24	30	13950.0
4	2	99.4	176.6	66.4	54.3	2824	136	3.19	3.40	8.0	115	5500	18	22	17450.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
200	-1	109.1	188.8	68.9	55.5	2952	141	3.78	3.15	9.5	114	5400	23	28	16845.0
201	-1	109.1	188.8	68.8	55.5	3049	141	3.78	3.15	8.7	160	5300	19	25	19045.0
202	-1	109.1	188.8	68.9	55.5	3012	173	3.58	2.87	8.8	134	5500	18	23	21485.0
203	-1	109.1	188.8	68.9	55.5	3217	145	3.01	3.40	23.0	106	4800	26	27	22470.0
204	-1	109.1	188.8	68.9	55.5	3062	141	3.78	3.15	9.5	114	5400	19	25	22625.0

205 rows × 15 columns

In [33]:
```python
x = data.drop(["price"], 1)
y = data["price"]
xtrain, xtest, ytrain, ytest = train_test_split(x, y, random_state = 16, test_size = 0.25, shuffle=True)
```

```
C:\Users\KIIT\AppData\Local\Temp\ipykernel_12584\313162150.py:1: FutureWarning: In a future version of pandas all arguments of DataFrame.drop excep
t for the argument 'labels' will be keyword-only.
  x = data.drop(["price"], 1)
```

In [34]:
```python
model = DecisionTreeRegressor()
```

In [35]:
```python
model.fit(xtrain, ytrain)
model.score(xtrain, ytrain)
```

Out[35]:
```
0.9997917694852244
```

In [36]:
```python
Prediction = model.predict(xtest)
print("Decision Tree Regression Model:")
print("******************************")
print('R2_score:', r2_score(ytest,Prediction))
print('Mean Absolute Error:', mean_absolute_error(ytest,Prediction))
print('Mean Squared Error:', mean_squared_error(ytest,Prediction))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(ytest,Prediction)))
```

```
Decision Tree Regression Model:
******************************
R2_score: 0.9263969588782768
Mean Absolute Error: 1511.8782115384615
Mean Squared Error: 5158535.641382481
Root Mean Squared Error: 2271.2409914807545
```

In [37]:
```python
print("Accuracy of Decision Tree Regression Model: ", model.score(xtrain, ytrain))
```

```
Accuracy of Decision Tree Regression Model:  0.9997917694852244
```

In [ ]: