In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\admin\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarn
ing: A NumPy version >=1.16.5 and <1.23.0 is required for this version of
SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [3]:
```
cars = pd.read_csv('car.csv')
cars.head()
```

Out[3]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	e
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	

5 rows × 26 columns

In [4]:
```
1 cars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   car_ID            205 non-null    int64  
 1   symboling         205 non-null    int64  
 2   CarName           205 non-null    object 
 3   fueltype          205 non-null    object 
 4   aspiration        205 non-null    object 
 5   doornumber        205 non-null    object 
 6   carbody           205 non-null    object 
 7   drivewheel        205 non-null    object 
 8   enginelocation    205 non-null    object 
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64  
 14  enginetype        205 non-null    object 
 15  cylindernumber    205 non-null    object 
 16  enginesize        205 non-null    int64  
 17  fuelsystem        205 non-null    object 
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64  
 22  peakrpm           205 non-null    int64  
 23  citympg           205 non-null    int64  
 24  highwaympg        205 non-null    int64  
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

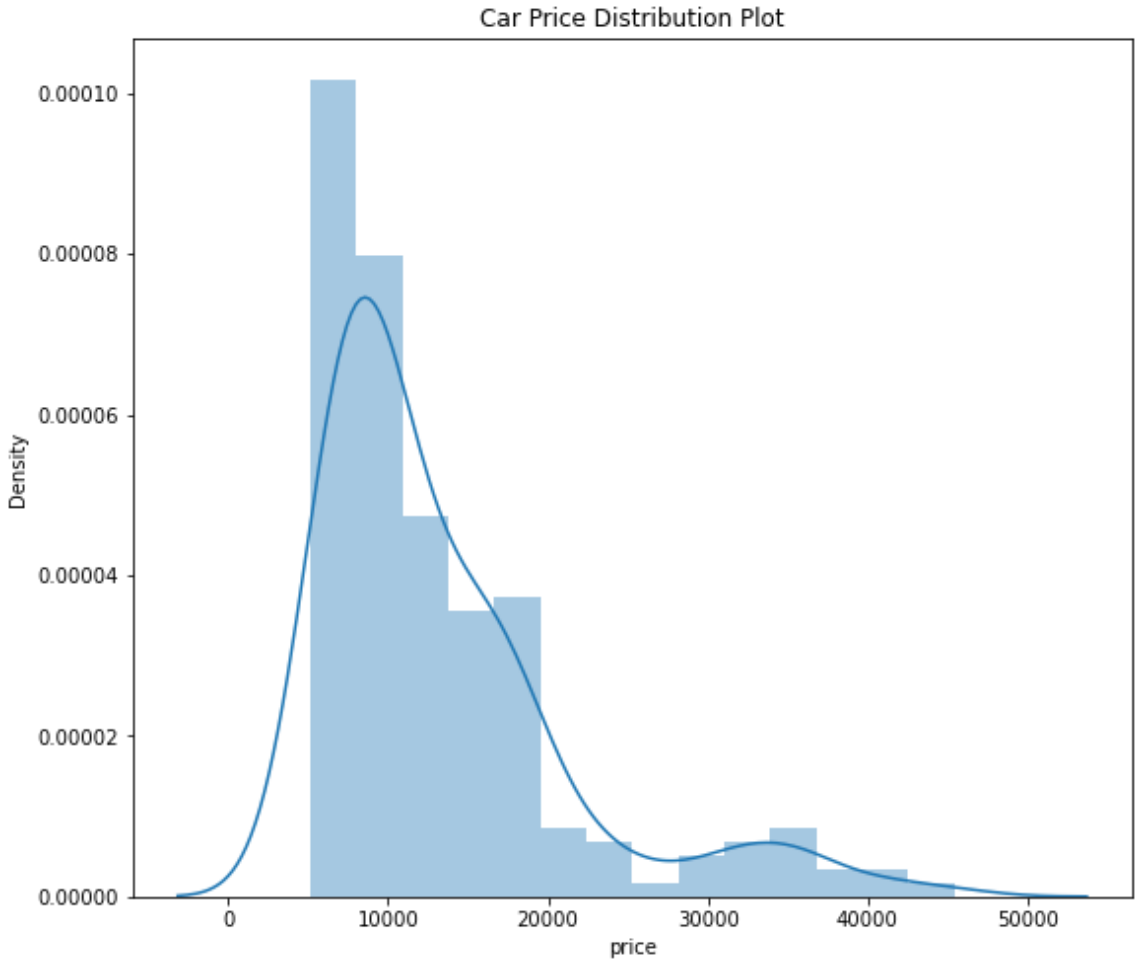
In [5]:
```
1 cars.isnull().sum()
```

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]:
```
1 plt.figure(figsize=(20,8))
2 plt.subplot(1,2,1)
3 plt.title("Car Price Distribution Plot")
4 sns.distplot(cars.price)
```

```
C:\Users\admin\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 fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[6]:
```
<AxesSubplot:title={'center':'Car Price Distribution Plot'}, xlabel='pric
e', ylabel='Density'>
```



In [7]:
```
1 cars.describe()
```

Out[7]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	103.000000	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854
std	59.322565	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204
min	1.000000	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000
25%	52.000000	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000
50%	103.000000	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000
75%	154.000000	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000
max	205.000000	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000

In [8]:
```
1 cars.price.mean()
```

Out[8]:
```
13276.710570731706
```

In [9]:
```
1 cars.price.std()
```

Out[9]:
```
7988.85233174315
```

In [10]:
```
1 L = cars.price.mean() - 3*cars.price.std()
2 L
```

Out[10]:
```
-10689.846424497744
```

In [11]:
```
1 U = cars.price.mean() + 3*cars.price.std()
2 U
```

Out[11]:
```
37243.267565961156
```

In [12]:
```
1 cars[(cars.price > U) | (cars.price < L)]
```

Out[12]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	eng
16	17	0	bmw x5	gas	std	two	sedan	rwd	
73	74	0	buick century special	gas	std	four	sedan	rwd	
74	75	1	buick regal sport coupe (turbo)	gas	std	two	hardtop	rwd	

3 rows × 26 columns

In [13]:
```
1 mcars = cars[(cars.price <= U) & (cars.price >= L)]
```

In [14]:
```
1 mcars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 202 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   car_ID            202 non-null    int64  
 1   symboling         202 non-null    int64  
 2   CarName           202 non-null    object 
 3   fueltype          202 non-null    object 
 4   aspiration        202 non-null    object 
 5   doornumber        202 non-null    object 
 6   carbody           202 non-null    object 
 7   drivewheel        202 non-null    object 
 8   enginelocation    202 non-null    object 
 9   wheelbase         202 non-null    float64
 10  carlength         202 non-null    float64
 11  carwidth          202 non-null    float64
 12  carheight         202 non-null    float64
 13  curbweight        202 non-null    int64  
 14  enginetype        202 non-null    object 
 15  cylindernumber    202 non-null    object 
 16  enginesize        202 non-null    int64  
 17  fuelsystem        202 non-null    object 
 18  boreratio         202 non-null    float64
 19  stroke            202 non-null    float64
 20  compressionratio  202 non-null    float64
 21  horsepower        202 non-null    int64  
 22  peakrpm           202 non-null    int64  
 23  citympg           202 non-null    int64  
 24  highwaympg        202 non-null    int64  
 25  price             202 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 42.6+ KB
```

In [15]:
```
1 x = mcars.iloc[:, 20:24].values
2 y = mcars.iloc[:, 25].values
```

In [16]:
```
1 from sklearn.model_selection import train_test_split
2 x_train,x_test,y_train,y_test = train_test_split(x, y, test_size = 0.2,
```

In [17]:
```
1 from sklearn.linear_model import LinearRegression
2 model = LinearRegression()
```

In [18]:
```
1 model.fit(x_train,y_train)
```

Out[18]:
```
LinearRegression()
```

In [19]:
```
ypred = model.predict(x_test)
ypred
```

Out[19]:
```
array([ 4175.45309753, 15843.56580946, 19611.01376363, 20380.85458482,
       20274.44466859,  9415.74662413, 14287.6930766 , 13548.51569155,
       16840.88783595,  7439.39262542, 10020.66612176, 25905.82412337,
       11009.80004235,  7332.25613048, 18042.99225923, 14287.6930766 ,
        6491.09930247, 14416.19472681,  8780.56313391,  6049.88783137,
       20558.82018667, 11033.06393096, 11440.65035548,  8003.82352924,
       10020.66612176, 25905.82412337,  9415.74662413, 10358.23807168,
       18576.32921943,  4175.45309753, 23581.08118075, 13784.50821037,
        6096.2578279 , 19611.01376363, 11190.83416232, 10085.02617381,
       14421.94723586, 14898.48545067, 10477.13737206, 13447.05590649,
       20201.65979893])
```

In [20]:
```
x_train.shape
```

Out[20]:
```
(161, 4)
```

In [21]:
```
from sklearn.metrics import r2_score
r2_score(y_test,ypred)
```

Out[21]:
```
0.7038832041380029
```

In [22]:
```
print(model.coef_)
print(model.intercept_)
```

```
[ 575.25090463  112.5268254    -1.3594733  -335.78369938]
10877.103037647543
```

In [23]:
```
def scatter(x,fig):
    plt.subplot(5,2,fig)
    plt.scatter(cars[x], cars['price'])
    plt.title(x + ' vs Price')
    plt.ylabel('Price')
    plt.xlabel(x)
```

In [24]:
```
plt.figure(figsize = (10,20))
scatter('carlength', 1)
scatter('carwidth', 2)
scatter('carheight', 3)
scatter('curbweight', 4)
plt.tight_layout()
```