In [1]:

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

Exploratory Data Analysis

In [2]:

```
df=pd.read_csv("cars.csv")
df
```

Out[2]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	en
0	3	?	alfa-romero	gas	convertible	rwd	front	64.1	48.8	
1	3	?	alfa-romero	gas	convertible	rwd	front	64.1	48.8	
2	1	?	alfa-romero	gas	hatchback	rwd	front	65.5	52.4	
3	2	164	audi	gas	sedan	fwd	front	66.2	54.3	
4	2	164	audi	gas	sedan	4wd	front	66.4	54.3	
...	...	...	...	...	...	...	...	...	...	
200	-1	95	volvo	gas	sedan	rwd	front	68.9	55.5	
201	-1	95	volvo	gas	sedan	rwd	front	68.8	55.5	
202	-1	95	volvo	gas	sedan	rwd	front	68.9	55.5	
203	-1	95	volvo	diesel	sedan	rwd	front	68.9	55.5	
204	-1	95	volvo	gas	sedan	rwd	front	68.9	55.5	

205 rows × 15 columns

In [3]:

```
df.shape
```

Out[3]:

```
(205, 15)
```

In [4]:

```
df.describe()
```

Out[4]:

	symboling	width	height	engine-size	city-mpg	highway-mpg	pri
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.0000
mean	0.834146	65.907805	53.724878	126.907317	25.219512	30.751220	13227.4780
std	1.245307	2.145204	2.443522	41.642693	6.542142	6.886443	7902.6516
min	-2.000000	60.300000	47.800000	61.000000	13.000000	16.000000	5118.0000
25%	0.000000	64.100000	52.000000	97.000000	19.000000	25.000000	7788.0000
50%	1.000000	65.500000	54.100000	120.000000	24.000000	30.000000	10345.0000
75%	2.000000	66.900000	55.500000	141.000000	30.000000	34.000000	16500.0000
max	3.000000	72.300000	59.800000	326.000000	49.000000	54.000000	45400.0000

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   symboling          205 non-null    int64  
 1   normalized-losses  205 non-null    object 
 2   make               205 non-null    object 
 3   fuel-type          205 non-null    object 
 4   body-style         205 non-null    object 
 5   drive-wheels       205 non-null    object 
 6   engine-location    205 non-null    object 
 7   width              205 non-null    float64
 8   height             205 non-null    float64
 9   engine-type        205 non-null    object 
 10  engine-size        205 non-null    int64  
 11  horsepower         205 non-null    object 
 12  city-mpg           205 non-null    int64  
 13  highway-mpg        205 non-null    int64  
 14  price              205 non-null    int64  
dtypes: float64(2), int64(5), object(8)
memory usage: 24.1+ KB
```

Handling Missing Values

In [6]:
```
df.isna()
```

Out[6]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine-type
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
200	False	False	False	False	False	False	False	False	False	False
201	False	False	False	False	False	False	False	False	False	False
202	False	False	False	False	False	False	False	False	False	False
203	False	False	False	False	False	False	False	False	False	False
204	False	False	False	False	False	False	False	False	False	False

205 rows × 15 columns

In [7]:
```
df.isna().sum()
```

Out[7]:
```
symboling            0
normalized-losses    0
make                 0
fuel-type            0
body-style           0
drive-wheels         0
engine-location      0
width                0
height               0
engine-type          0
engine-size          0
horsepower           0
city-mpg             0
highway-mpg          0
price                0
dtype: int64
```

Changing Dtype from Object to Float64

In [8]:

```
'''df["normalized-losses"].replace("?",np.nan,inplace=True)
df["horsepower"].replace("?",np.nan,inplace=True)

df["normalized-losses"]=df["normalized-losses"].astype("float64")
df["horsepower"]=df["horsepower"].astype("float64")

nmean=df["normalized-losses"].mean()
hmean=df["horsepower"].mean()

df["normalized-losses"].fillna(nmean,inplace=True)
df["horsepower"].fillna(hmean,inplace=True)'''
```

Out[8]:

```
'df["normalized-losses"].replace("?",np.nan,inplace=True)\ndf["horsepowe
r"].replace("?",np.nan,inplace=True)\n\ndf["normalized-losses"]=df["norma
lized-losses"].astype("float64")\ndf["horsepower"]=df["horsepower"].astyp
e("float64")\n\nnmean=df["normalized-losses"].mean()\nhmean=df["horsepowe
r"].mean()\n\ndf["normalized-losses"].fillna(nmean,inplace=True)\ndf["hor
sepower"].fillna(hmean,inplace=True)'
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   symboling          205 non-null    int64  
 1   normalized-losses  205 non-null    object 
 2   make               205 non-null    object 
 3   fuel-type          205 non-null    object 
 4   body-style         205 non-null    object 
 5   drive-wheels       205 non-null    object 
 6   engine-location    205 non-null    object 
 7   width              205 non-null    float64
 8   height             205 non-null    float64
 9   engine-type        205 non-null    object 
 10  engine-size        205 non-null    int64  
 11  horsepower         205 non-null    object 
 12  city-mpg           205 non-null    int64  
 13  highway-mpg        205 non-null    int64  
 14  price              205 non-null    int64  
dtypes: float64(2), int64(5), object(8)
memory usage: 24.1+ KB
```

In [10]:

```
#df["normalized-losses"]
```

In [11]:

```
#df["horsepower"]
```

In [12]:

```
df["normalized-losses"].replace("?",np.nan,inplace=True)
df["horsepower"].replace("?",np.nan,inplace=True)

df["normalized-losses"]=df["normalized-losses"].astype("float64")
df["horsepower"]=df["horsepower"].astype("float64")
```

In [13]:

```
from sklearn.impute import SimpleImputer
```

In [14]:

```
si=SimpleImputer(missing_values=np.nan,strategy="mean")
```

In [15]:

```
df[["normalized-losses","horsepower"]]=si.fit_transform(df[["normalized-lo
```

In [16]:

```
df["normalized-losses"]
```

Out[16]:

```
0      122.0
1      122.0
2      122.0
3      164.0
4      164.0
       ...
200     95.0
201     95.0
202     95.0
203     95.0
204     95.0
Name: normalized-losses, Length: 205, dtype: float64
```

In [17]:

```
df["horsepower"]
```

Out[17]:

```
0      111.0
1      111.0
2      154.0
3      102.0
4      115.0
       ...
200    114.0
201    160.0
202    134.0
203    106.0
204    114.0
Name: horsepower, Length: 205, dtype: float64
```

In [18]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   symboling          205 non-null    int64  
 1   normalized-losses  205 non-null    float64
 2   make               205 non-null    object 
 3   fuel-type          205 non-null    object 
 4   body-style         205 non-null    object 
 5   drive-wheels       205 non-null    object 
 6   engine-location    205 non-null    object 
 7   width              205 non-null    float64
 8   height             205 non-null    float64
 9   engine-type        205 non-null    object 
 10  engine-size        205 non-null    int64  
 11  horsepower         205 non-null    float64
 12  city-mpg           205 non-null    int64  
 13  highway-mpg        205 non-null    int64  
 14  price              205 non-null    int64  
dtypes: float64(4), int64(5), object(6)
memory usage: 24.1+ KB
```

In [19]:

```
df.head()
```

Out[19]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engin typ
0	3	122.0	alfa-romero	gas	convertible	rwd	front	64.1	48.8	doh
1	3	122.0	alfa-romero	gas	convertible	rwd	front	64.1	48.8	doh
2	1	122.0	alfa-romero	gas	hatchback	rwd	front	65.5	52.4	ohc
3	2	164.0	audi	gas	sedan	fwd	front	66.2	54.3	oh
4	2	164.0	audi	gas	sedan	4wd	front	66.4	54.3	oh

In [20]:

```
df.describe()
```

Out[20]:

	symboling	normalized-losses	width	height	engine-size	horsepower	city-mp
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.00000
mean	0.834146	122.000000	65.907805	53.724878	126.907317	104.256158	25.21951
std	1.245307	31.681008	2.145204	2.443522	41.642693	39.519211	6.54214
min	-2.000000	65.000000	60.300000	47.800000	61.000000	48.000000	13.00000
25%	0.000000	101.000000	64.100000	52.000000	97.000000	70.000000	19.00000
50%	1.000000	122.000000	65.500000	54.100000	120.000000	95.000000	24.00000
75%	2.000000	137.000000	66.900000	55.500000	141.000000	116.000000	30.00000
max	3.000000	256.000000	72.300000	59.800000	326.000000	288.000000	49.00000

In [21]:

```
feature=df.iloc[:,:-1]
target=df["price"]
```

In [22]:

```
feature
```

Out[22]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	en
0	3	122.0	alfa-romero	gas	convertible	rwd	front	64.1	48.8	
1	3	122.0	alfa-romero	gas	convertible	rwd	front	64.1	48.8	
2	1	122.0	alfa-romero	gas	hatchback	rwd	front	65.5	52.4	
3	2	164.0	audi	gas	sedan	fwd	front	66.2	54.3	
4	2	164.0	audi	gas	sedan	4wd	front	66.4	54.3	
...	...	...	...	...	...	...	...	...	...	
200	-1	95.0	volvo	gas	sedan	rwd	front	68.9	55.5	
201	-1	95.0	volvo	gas	sedan	rwd	front	68.8	55.5	
202	-1	95.0	volvo	gas	sedan	rwd	front	68.9	55.5	
203	-1	95.0	volvo	diesel	sedan	rwd	front	68.9	55.5	
204	-1	95.0	volvo	gas	sedan	rwd	front	68.9	55.5	

205 rows × 14 columns

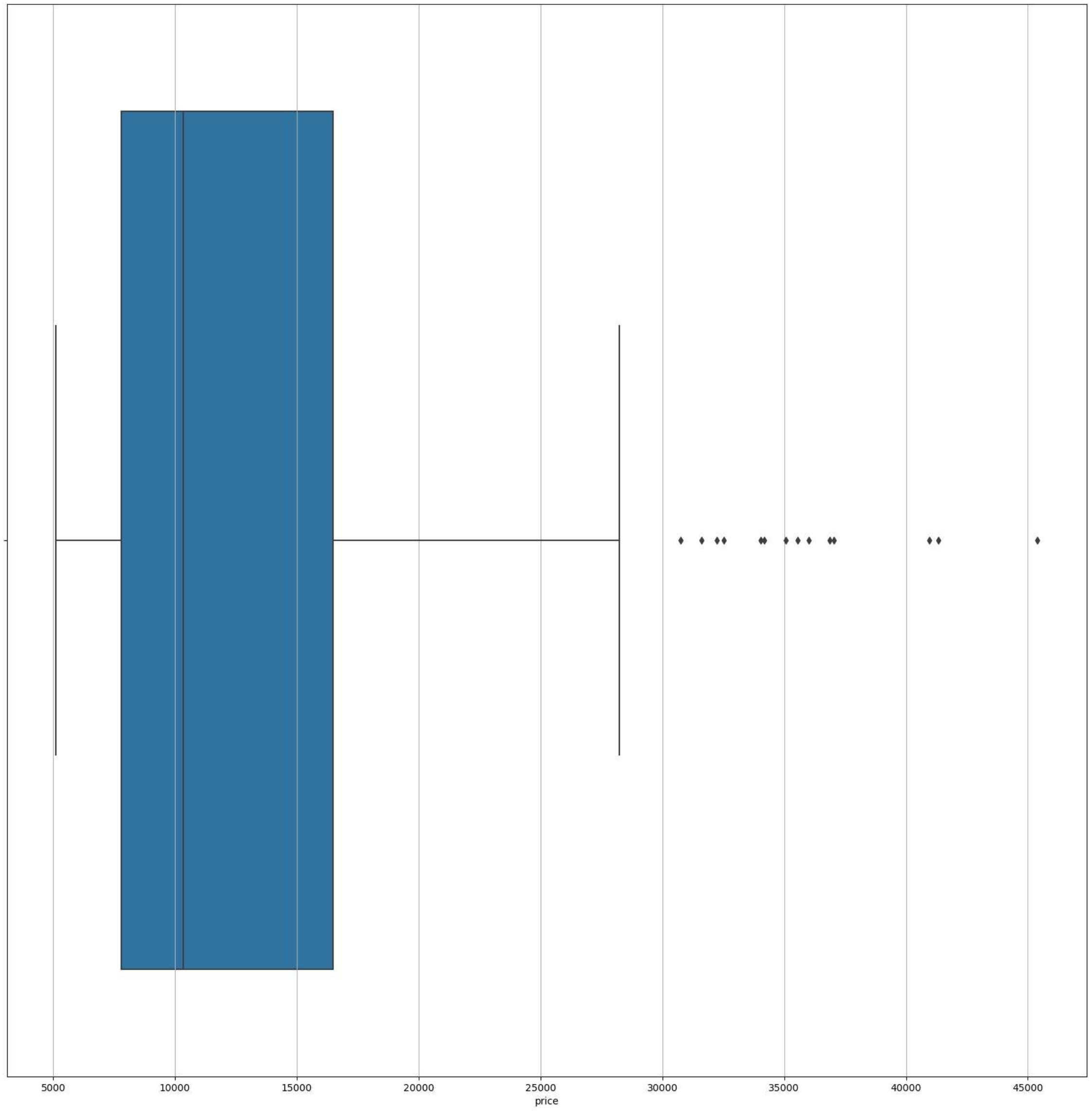
```
In [23]: target
```

```
Out[23]: 0      13495
         1      16500
         2      16500
         3      13950
         4      17450
                ...
         200    16845
         201    19045
         202    21485
         203    22470
         204    22625
         Name: price, Length: 205, dtype: int64
```

Handling outliers

In [24]:

```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target);
```



In [25]:

```
df[(df["make"]=="dodge")&(df["price"]>9000)]
```

Out[25]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine typ
29	3	145.0	dodge	gas	hatchback	fwd	front	66.3	50.2	oh

In [26]:

```
df.loc[29,"price"]
```

Out[26]:

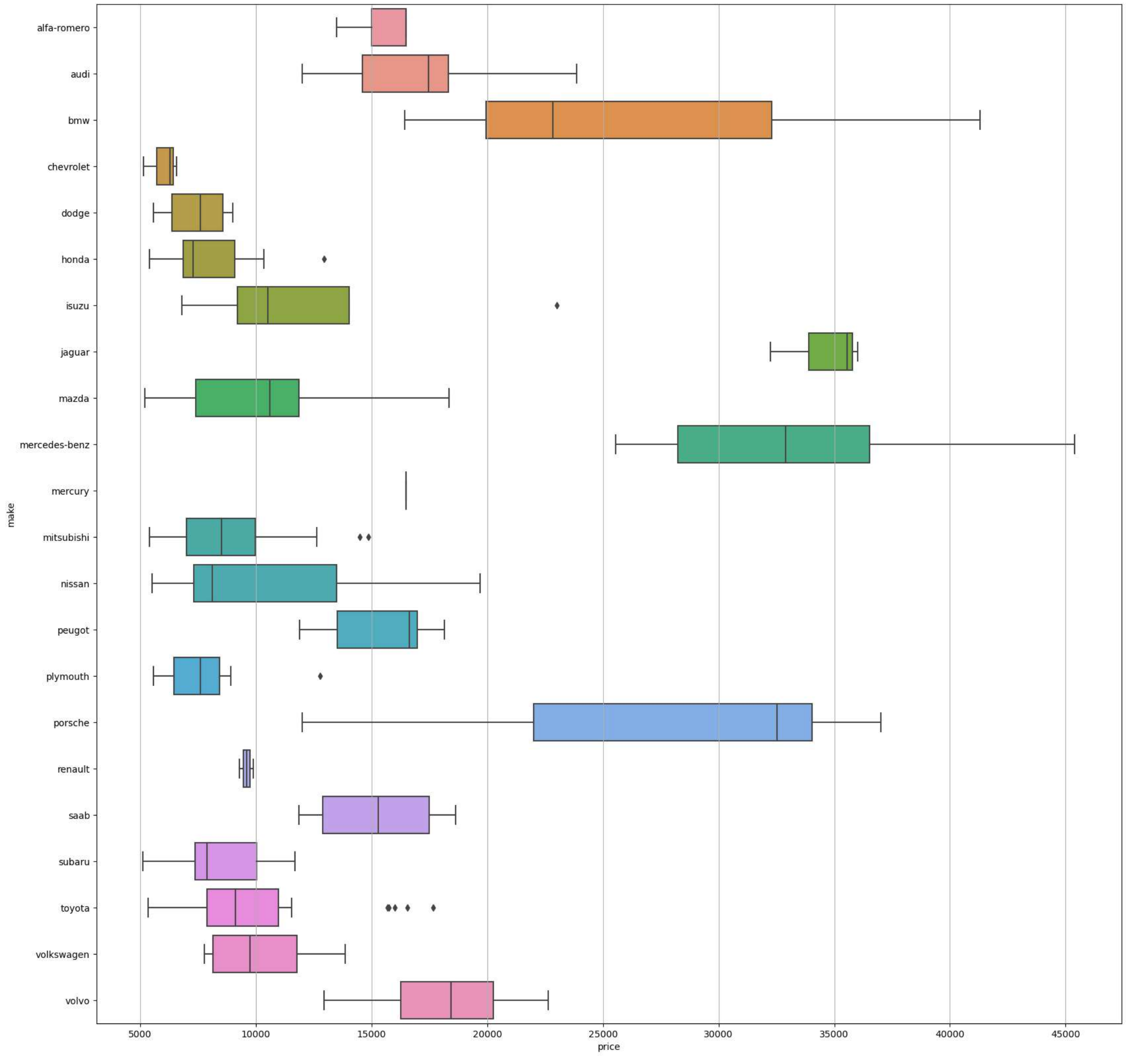
```
12964
```

In [27]:

```
df.loc[29,"price"]=9000
```

In [28]:

```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target,y="make");
```



In [29]:

```
df[(df["make"]=="honda")&(df["price"]>12500)]
```

Out[29]:

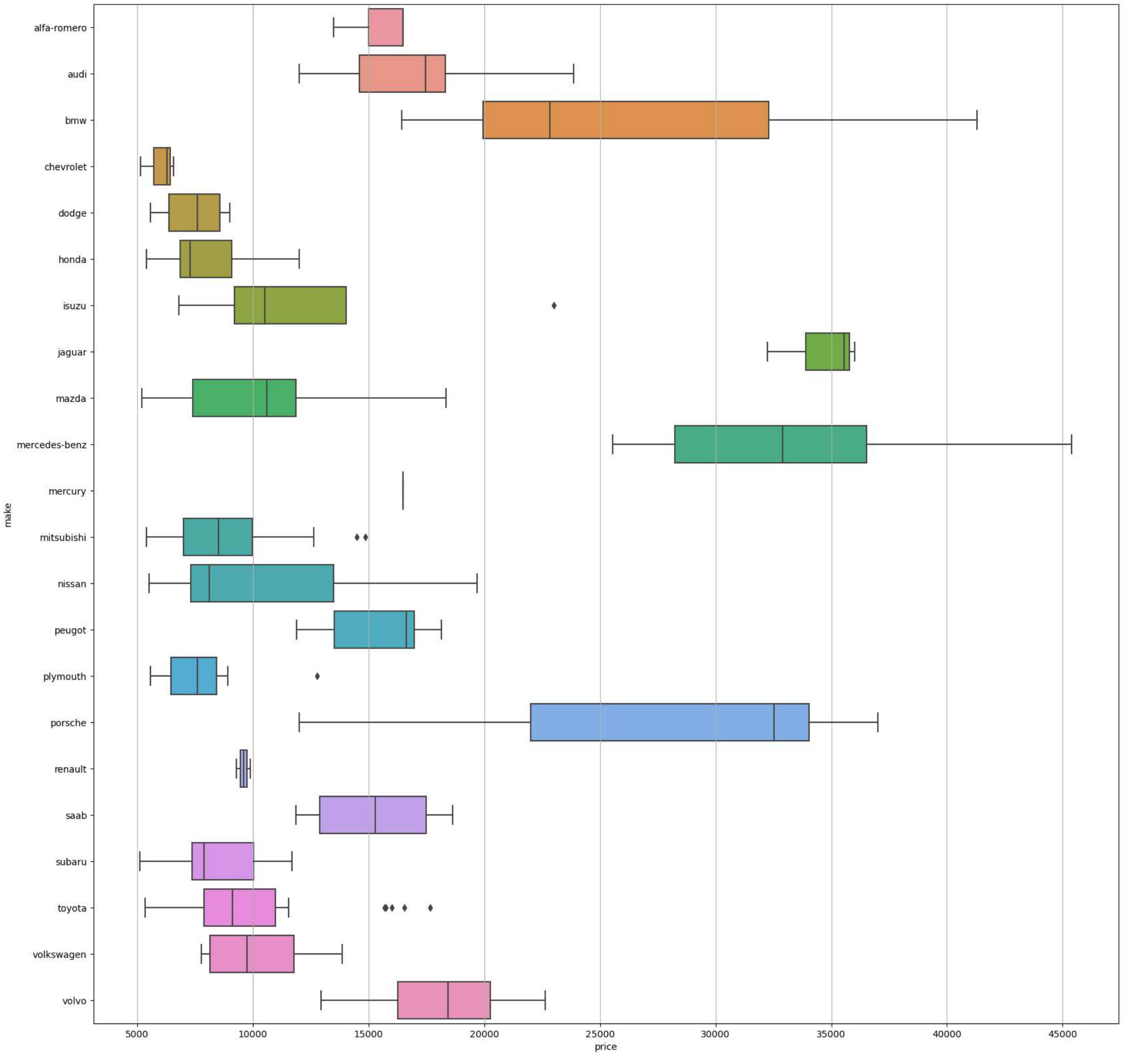
	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine-type
41	0	85.0	honda	gas	sedan	fwd	front	65.2	54.1	ohc

In [30]:

```
df.loc[41,"price"]=12000
```

In [31]:

```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target,y="make");
```



In [32]:

```
df[(df["make"]=="toyota")&(df["price"]>13500)]
```

Out[32]:

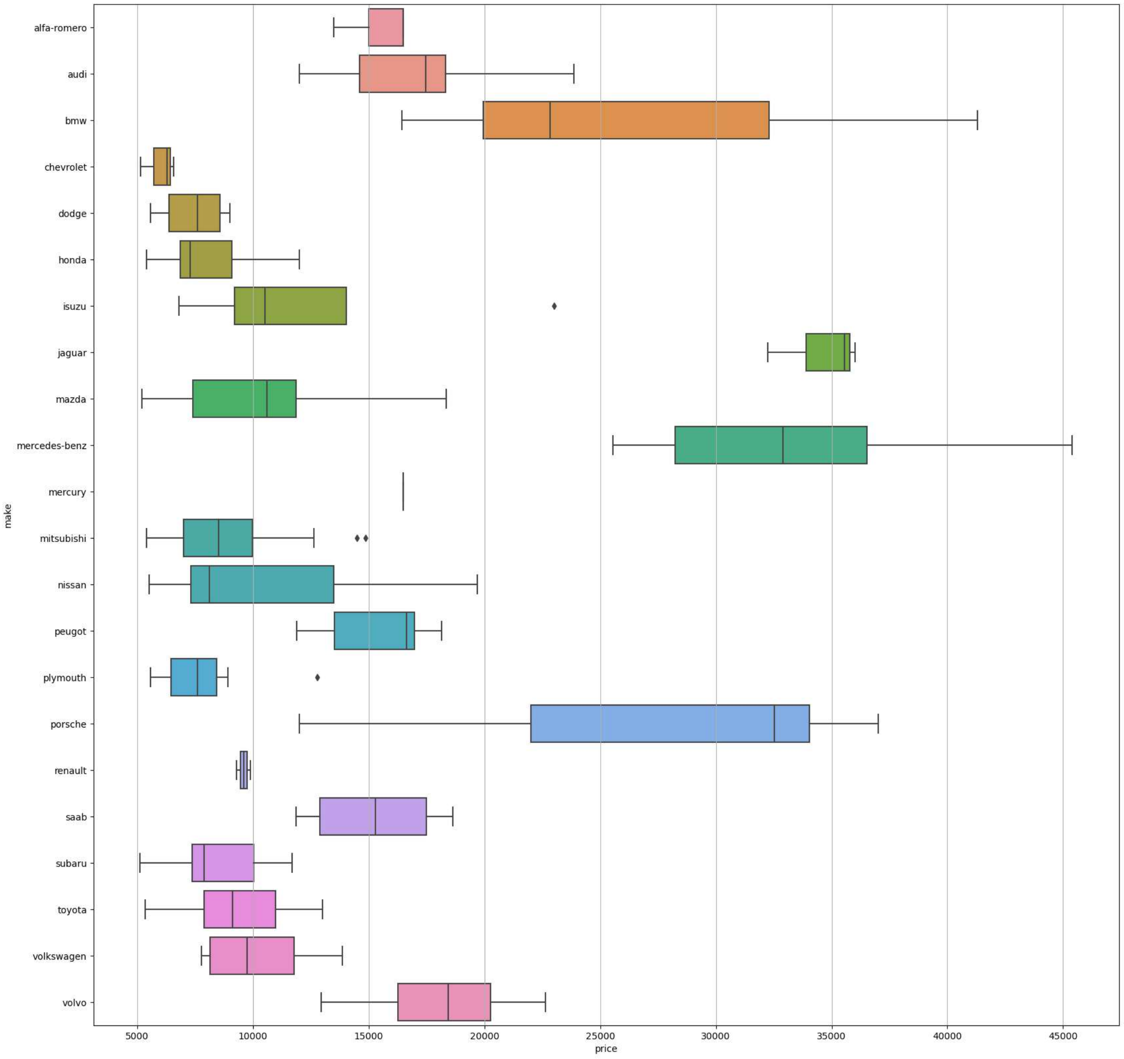
	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engi ty
172	2	134.0	toyota	gas	convertible	rwd	front	65.6	53.0	c
178	3	197.0	toyota	gas	hatchback	rwd	front	67.7	52.0	dc
179	3	197.0	toyota	gas	hatchback	rwd	front	67.7	52.0	dc
180	-1	90.0	toyota	gas	sedan	rwd	front	66.5	54.1	dc
181	-1	122.0	toyota	gas	wagon	rwd	front	66.5	54.1	dc

```
In [33]:  df.loc[[172,178,179,180,181],"price"]
```

```
Out[33]: 172    17669
         178    16558
         179    15998
         180    15690
         181    15750
         Name: price, dtype: int64
```

```
In [34]:  df.loc[[172,178,179,180,181],"price"]=13000
```

```
In [35]:  plt.figure(figsize=(20,20))
          plt.grid()
          sns.boxplot(data=feature,x=target,y="make");
```



In [36]:

```
df[(df["make"]=="mitsubishi")&(df["price"]>13000)]
```

Out[36]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	en
83	3	122.0	mitsubishi	gas	hatchback	fwd	front	66.3	50.2	
84	3	122.0	mitsubishi	gas	hatchback	fwd	front	66.3	50.2	

In [37]:

```
df.loc[[83,84],"price"]
```

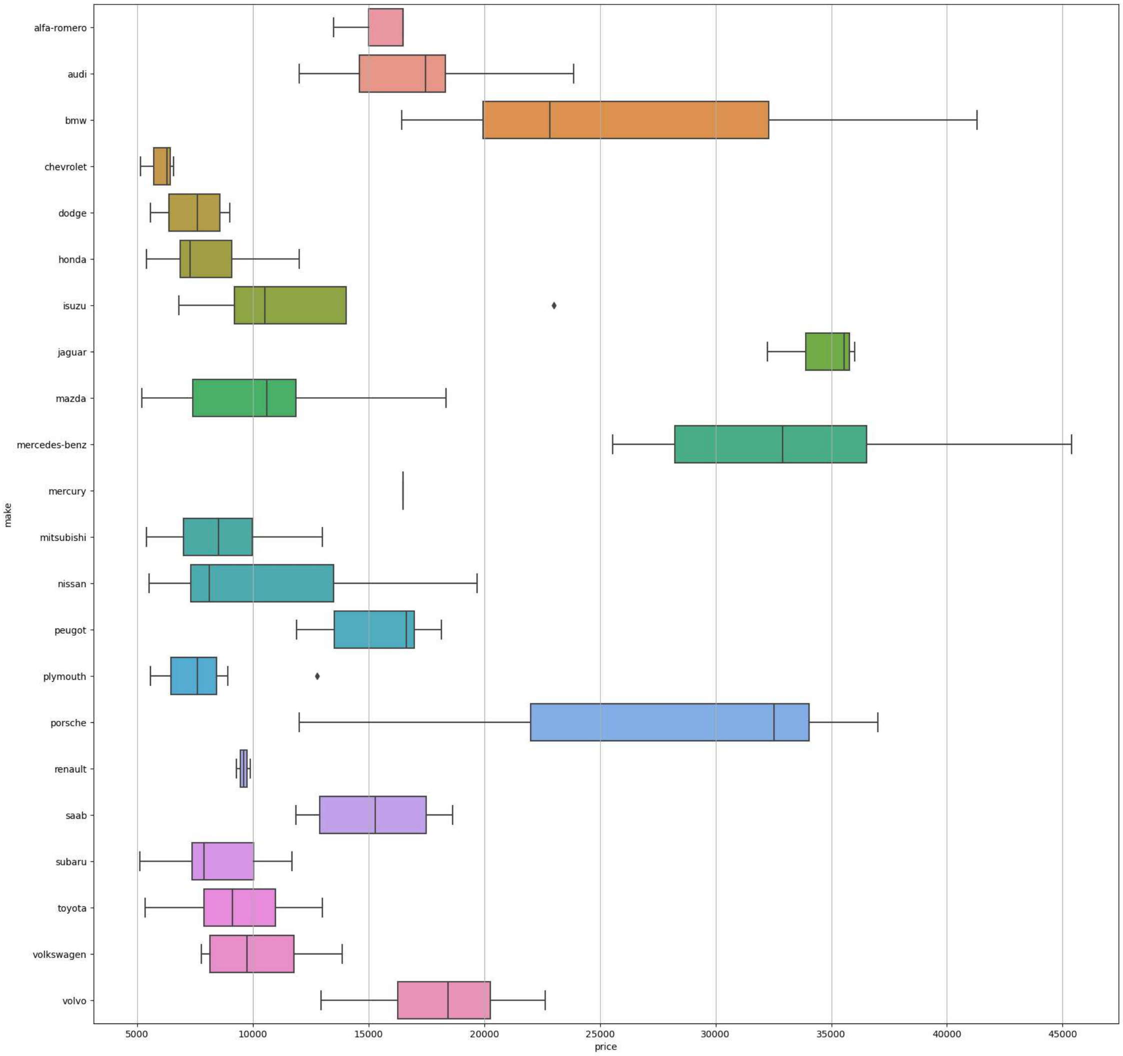
Out[37]:

```
83    14869
84    14489
Name: price, dtype: int64
```

In [38]:

```
df.loc[[83,84],"price"]=13000
```

In [39]:
```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target,y="make");
```



In [40]:
```
df[(df["make"]=="plymouth")&(df["price"]>10000)]
```

Out[40]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	en
124	3	122.0	plymouth	gas	hatchback	rwd	front	66.3	50.2	

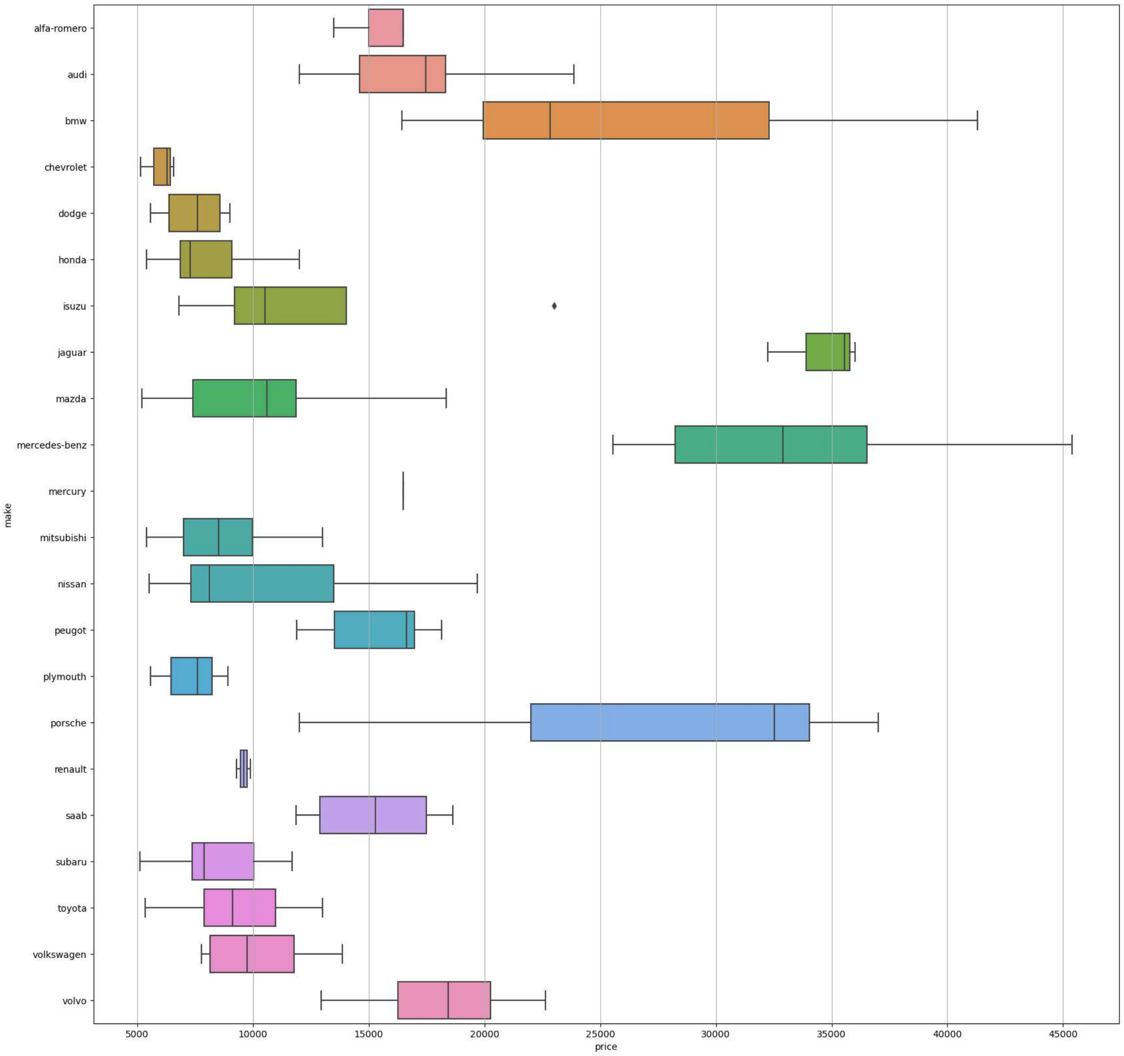
In [41]:
```
df.loc[124,"price"]
```

Out[41]:
```
12764
```

In [42]:
```
df.loc[124,"price"]=8500
```

In [43]:

```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target,y="make");
```



In [44]:

```
df[(df["make"]=="isuzu")&(df["price"]>22500)]
```

Out[44]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine-type	e
45	0	122.0	isuzu	gas	sedan	fwd	front	63.6	52.0	ohc	

In [45]:

```
df.loc[45,"price"]
```

Out[45]:

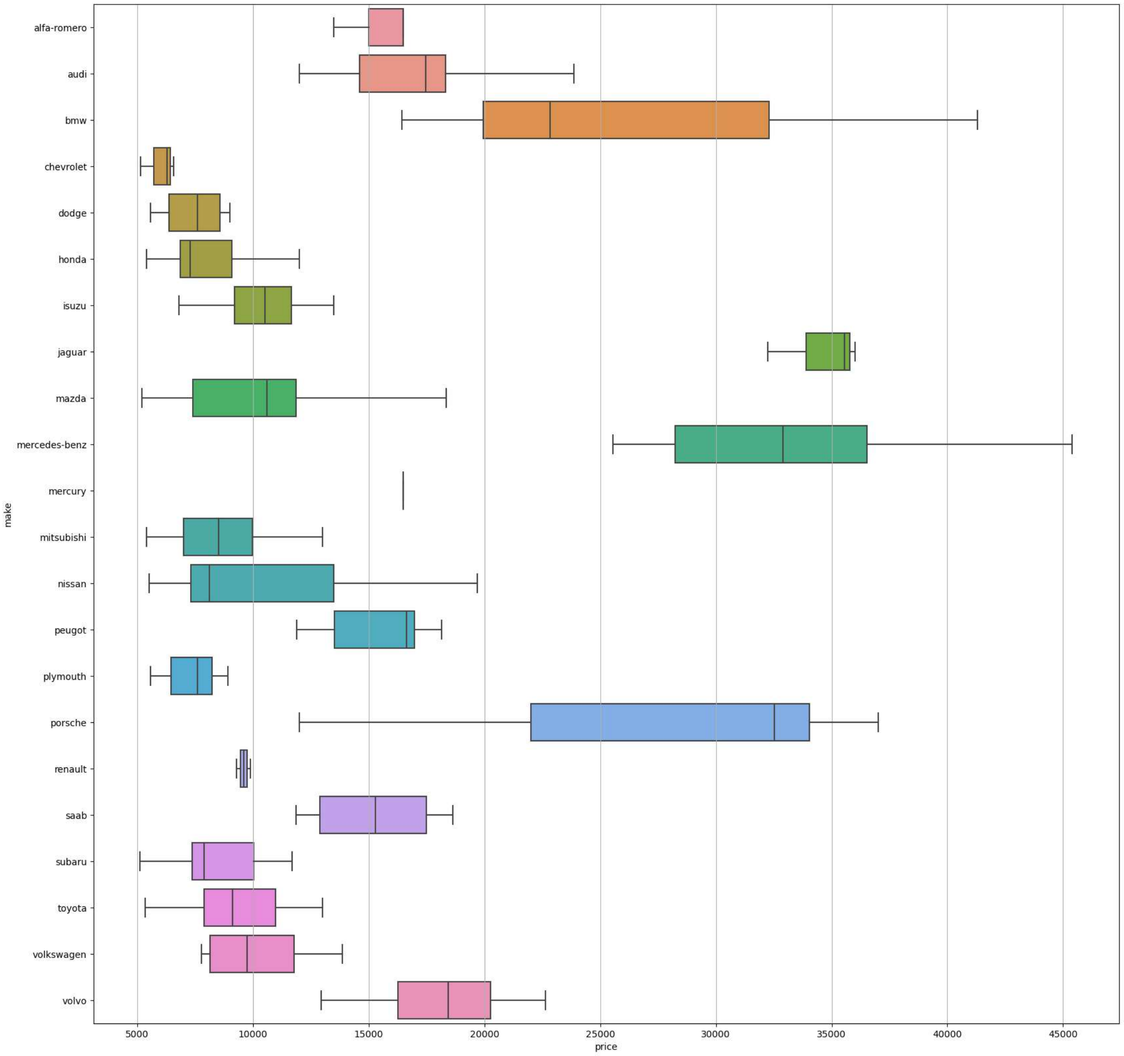
```
23000
```

In [46]:

```
df.loc[45,"price"]=13500
```

In [47]:

```
plt.figure(figsize=(20,20))
plt.grid()
sns.boxplot(data=feature,x=target,y="make");
```



To identify skew from numeric datatype and how to apply log transformation

In [48]:
```
df.describe()
```

Out[48]:

	symboling	normalized-losses	width	height	engine-size	horsepower	city-mp
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.00000
mean	0.834146	122.000000	65.907805	53.724878	126.907317	104.256158	25.21951
std	1.245307	31.681008	2.145204	2.443522	41.642693	39.519211	6.54214
min	-2.000000	65.000000	60.300000	47.800000	61.000000	48.000000	13.00000
25%	0.000000	101.000000	64.100000	52.000000	97.000000	70.000000	19.00000
50%	1.000000	122.000000	65.500000	54.100000	120.000000	95.000000	24.00000
75%	2.000000	137.000000	66.900000	55.500000	141.000000	116.000000	30.00000
max	3.000000	256.000000	72.300000	59.800000	326.000000	288.000000	49.00000

In [49]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   symboling          205 non-null    int64  
 1   normalized-losses  205 non-null    float64
 2   make               205 non-null    object 
 3   fuel-type          205 non-null    object 
 4   body-style         205 non-null    object 
 5   drive-wheels       205 non-null    object 
 6   engine-location    205 non-null    object 
 7   width              205 non-null    float64
 8   height             205 non-null    float64
 9   engine-type        205 non-null    object 
 10  engine-size        205 non-null    int64  
 11  horsepower         205 non-null    float64
 12  city-mpg           205 non-null    int64  
 13  highway-mpg        205 non-null    int64  
 14  price              205 non-null    int64  
dtypes: float64(4), int64(5), object(6)
memory usage: 24.1+ KB
```

In [50]:
```
feature.select_dtypes(["int64","float64"]).columns
```

Out[50]:
```
Index(['symboling', 'normalized-losses', 'width', 'height', 'engine-siz
e',
       'horsepower', 'city-mpg', 'highway-mpg'],
      dtype='object')
```

In [51]:

```
colname=feature.select_dtypes(["int64","float64"]).columns
```

In [52]:

```
colname
```

Out[52]:

```
Index(['symboling', 'normalized-losses', 'width', 'height', 'engine-siz
e',
       'horsepower', 'city-mpg', 'highway-mpg'],
      dtype='object')
```

In [53]:

```
from scipy.stats import skew
```

In [54]:

```
skew(feature["normalized-losses"])
```

Out[54]:

```
0.8485348696008058
```

In [55]:

```
feature[colname]
```

	symboling	normalized losses	width	height	engine size	horsepower	city mpg	highway mpg
0	3	122.0	64.1	48.8	130	111.0	21	27
1	3	122.0	64.1	48.8	130	111.0	21	27
2	1	122.0	65.5	52.4	152	154.0	19	26
3	2	164.0	66.2	54.3	109	102.0	24	30
4	2	164.0	66.4	54.3	136	115.0	18	22
...	...	...	...	...	...	...	...	...
200	-1	95.0	68.9	55.5	141	114.0	23	28
201	-1	95.0	68.8	55.5	141	160.0	19	25
202	-1	95.0	68.9	55.5	173	134.0	18	23
203	-1	95.0	68.9	55.5	145	106.0	26	27
204	-1	95.0	68.9	55.5	141	114.0	19	25

205 rows × 8 columns

In [56]:

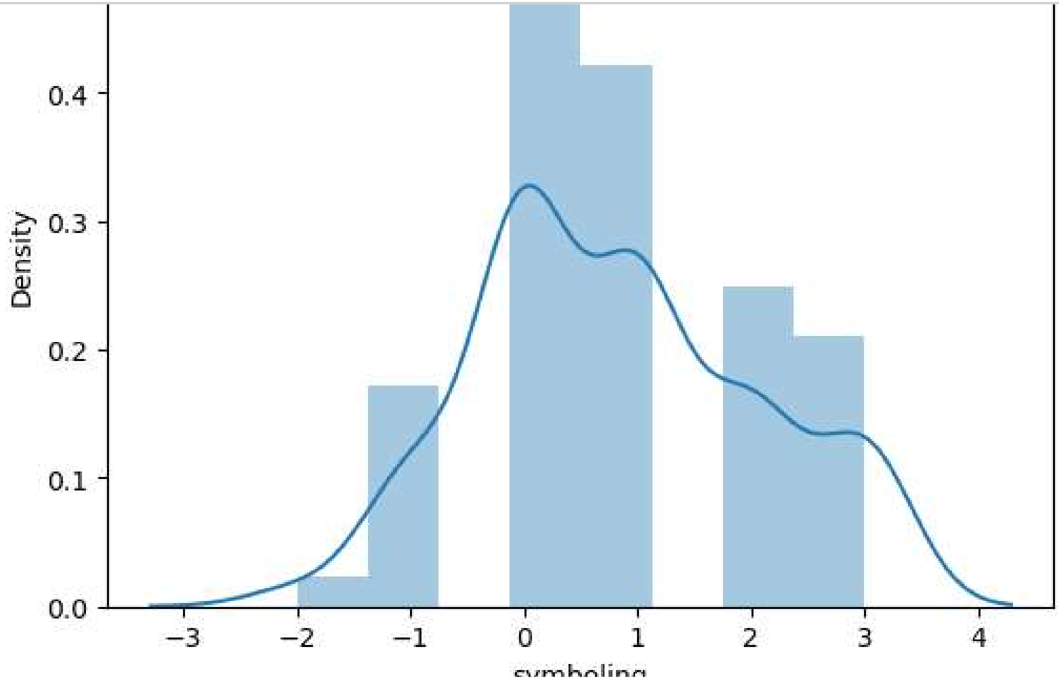
```
skew(feature[colname])
```

Out[56]:

```
array([0.20952469, 0.84853487, 0.89737535, 0.06265992, 1.93337485,
       1.38751473, 0.65883775, 0.53603793])
```

In [57]:

```
for i in feature[colname]:
    print(i)
    print(skew(feature[i]))
    plt.figure()
    sns.distplot(feature[i])
    plt.show()
```



In [58]:

```
df.corr()# to check the corelation
```

```
C:\Users\Reshmi\AppData\Local\Temp\ipykernel_21608\3597234383.py:1: Futur
eWarning: The default value of numeric_only in DataFrame.corr is deprecat
ed. In a future version, it will default to False. Select only valid colu
mns or specify the value of numeric_only to silence this warning.
  df.corr()# to check the corelation
```

Out[58]:

	symboling	normalized-losses	width	height	engine-size	horsepower	city-mpg
symboling	1.000000	0.465190	-0.232919	-0.541038	-0.105790	0.071389	-0.035823
normalized-losses	0.465190	1.000000	0.084195	-0.370706	0.110997	0.203434	-0.218749
width	-0.232919	0.084195	1.000000	0.279210	0.735433	0.642195	-0.642704
height	-0.541038	-0.370706	0.279210	1.000000	0.067149	-0.110137	-0.048640
engine-size	-0.105790	0.110997	0.735433	0.067149	1.000000	0.810713	-0.653658
horsepower	0.071389	0.203434	0.642195	-0.110137	0.810713	1.000000	-0.803162
city-mpg	-0.035823	-0.218749	-0.642704	-0.048640	-0.653658	-0.803162	1.000000
highway-mpg	0.034606	-0.178221	-0.677218	-0.107358	-0.677470	-0.770903	0.971337
price	-0.099208	0.123851	0.722863	0.150782	0.848517	0.736585	-0.655155

In [59]:

```
pd.concat([feature,target],axis=1).corr().style.background_gradient()
```

```
C:\Users\Reshmi\AppData\Local\Temp\ipykernel_21608\4285486589.py:1: Futur
eWarning: The default value of numeric_only in DataFrame.corr is deprecat
ed. In a future version, it will default to False. Select only valid colu
mns or specify the value of numeric_only to silence this warning.
  pd.concat([feature,target],axis=1).corr().style.background_gradient()
```

Out[59]:

	symboling	normalized-losses	width	height	engine-size	horsepower	city-mpg
symboling	1.000000	0.465190	-0.232919	-0.541038	-0.105790	0.071389	-0.035823
normalized-losses	0.465190	1.000000	0.084195	-0.370706	0.110997	0.203434	-0.218749
width	-0.232919	0.084195	1.000000	0.279210	0.735433	0.642195	-0.642704
height	-0.541038	-0.370706	0.279210	1.000000	0.067149	-0.110137	-0.048640
engine-size	-0.105790	0.110997	0.735433	0.067149	1.000000	0.810713	-0.653658
horsepower	0.071389	0.203434	0.642195	-0.110137	0.810713	1.000000	-0.803162
city-mpg	-0.035823	-0.218749	-0.642704	-0.048640	-0.653658	-0.803162	1.000000
highway-mpg	0.034606	-0.178221	-0.677218	-0.107358	-0.677470	-0.770903	0.971337
price	-0.099208	0.123851	0.722863	0.150782	0.848517	0.736585	-0.655155

In [60]:

```
df["normalized-losses"].unique()
```

Out[60]:

```
array([122., 164., 158., 192., 188., 121.,  98.,  81., 118., 148., 110.,
       145., 137., 101.,  78., 106.,  85., 107., 104., 113., 150., 129.,
       115.,  93., 142., 161., 153., 125., 128., 103., 168., 108., 194.,
       231., 119., 154.,  74., 186.,  83., 102.,  89.,  87.,  77.,  91.,
       134.,  65., 197.,  90.,  94., 256.,  95.])
```

In [61]:

```
feature["normalized-losses"]=np.log(feature["normalized-losses"])
```

In [62]:

```
skew(feature["normalized-losses"])
```

Out[62]:

```
0.03137735337911685
```

Encoding

1.OneHotEncoding 2.LabelEncoding

In [63]:

```
from sklearn.preprocessing import OneHotEncoder
```

In [64]:

```
one=OneHotEncoder()
one.fit_transform(feature[["make"]]).toarray()
```

Out[64]:

```
array([[1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.]])
```

In [65]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
le.fit_transform(target)
```

Out[65]:

```
array([119, 134, 134, 125, 144, 128, 145, 153, 163, 109, 133, 141, 157,
       158, 164, 168, 180, 177,   1,  11,  17,   7,  13,  49,  10,  20,
        41,  64,  67,  70,  14,  25,   5,  16,  32,  34,  34,  47,  71,
        66,  91, 109,  92,  22,  88, 121,  98, 170, 175, 176,   2,   8,
        23,  21,  37,  97, 106, 122, 131,  66,  61,  93,  90,  95, 100,
       148, 149, 165, 167, 166, 169, 173, 174, 179, 181, 135,   4,   9,
        19,  42,  82,  62, 113, 115, 115,  28,  55,  74,  74,   6,  30,
        18,  24,  36,  35,  46,  39,  52,  58,  69,  79, 120, 126, 120,
       143, 156, 150, 108, 116, 112, 124, 130, 140, 138, 142, 137, 146,
       147,   7,  49,  10,  20,  41,  67,  63, 160, 171, 172, 178, 109,
        75,  81, 107, 110, 127, 129, 147, 152,   0,  29,  40,  31,  44,
        83,  72, 102,  38,  89,  53, 105,   3,  12,  15,  26,  48,  65,
        27,  33,  48,  45,  43,  59,  73,  54,  57,  76,  78,  60,  80,
        86,  99, 103, 115,  68,  94,  85,  96, 101, 115, 115, 115, 115,
        44,  50,  51,  56,  61,  77,  87, 104,  84, 117, 123, 111, 114,
       118, 132, 136, 151, 154, 139, 155, 159, 161, 162], dtype=int64)
```

In [66]:

```
catcol=feature.select_dtypes("object").columns
```

In [67]:

```
catcol
```

Out[67]:

```
Index(['make', 'fuel-type', 'body-style', 'drive-wheels', 'engine-locatio
n',
       'engine-type'],
      dtype='object')
```

Using OrdinalEncoder

input

In [68]:

```
from sklearn.preprocessing import OrdinalEncoder
oe=OrdinalEncoder()
feature[catcol]=oe.fit_transform(feature[catcol])
```

In [69]:

```
feature[catcol]
```

Out[69]:

	make	fuel-type	body-style	drive-wheels	engine-location	engine-type
0	0.0	1.0	0.0	2.0	0.0	0.0
1	0.0	1.0	0.0	2.0	0.0	0.0
2	0.0	1.0	2.0	2.0	0.0	5.0
3	1.0	1.0	3.0	1.0	0.0	3.0
4	1.0	1.0	3.0	0.0	0.0	3.0
...	...	...	...	...	...	...
200	21.0	1.0	3.0	2.0	0.0	3.0
201	21.0	1.0	3.0	2.0	0.0	3.0
202	21.0	1.0	3.0	2.0	0.0	5.0
203	21.0	0.0	3.0	2.0	0.0	3.0
204	21.0	1.0	3.0	2.0	0.0	3.0

205 rows × 6 columns

In [70]:

```
feature.head()
```

Out[70]:

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine-type	en
0	3	4.804021	0.0	1.0	0.0	2.0	0.0	64.1	48.8	0.0	
1	3	4.804021	0.0	1.0	0.0	2.0	0.0	64.1	48.8	0.0	
2	1	4.804021	0.0	1.0	2.0	2.0	0.0	65.5	52.4	5.0	
3	2	5.099866	1.0	1.0	3.0	1.0	0.0	66.2	54.3	3.0	
4	2	5.099866	1.0	1.0	3.0	0.0	0.0	66.4	54.3	3.0	

In [ ]: