# fgmbh2c29

March 30, 2023

Data cleaning on Automobile 1985 dataset and perform descriptive analytics

[3]:
```
import numpy as np
import pandas as pd
```

[4]:
```
data = pd.read_csv('auto_clean.csv')
```

[7]:
```
data.head(2)
```

[7]:
```
   symboling  normalized-losses         make aspiration num-of-doors  \
0          3                122  alfa-romero        std          two
1          3                122  alfa-romero        std          two

    body-style drive-wheels engine-location  wheel-base    length  …  \
0  convertible          rwd           front        88.6  0.811148  …
1  convertible          rwd           front        88.6  0.811148  …

   compression-ratio  horsepower  peak-rpm city-mpg highway-mpg  price  \
0                9.0       111.0    5000.0       21          27  13495
1                9.0       111.0    5000.0       21          27  16500

  city-L/100km  horsepower-binned  diesel  gas
0    11.190476             Medium       0    1
1    11.190476             Medium       0    1

[2 rows x 29 columns]
```

[8]:
```
data.columns
```

[8]:
```
Index(['symboling', 'normalized-losses', 'make', 'aspiration', 'num-of-doors',
       'body-style', 'drive-wheels', 'engine-location', 'wheel-base', 'length',
       'width', 'height', 'curb-weight', 'engine-type', 'num-of-cylinders',
       'engine-size', 'fuel-system', 'bore', 'stroke', 'compression-ratio',
       'horsepower', 'peak-rpm', 'city-mpg', 'highway-mpg', 'price',
       'city-L/100km', 'horsepower-binned', 'diesel', 'gas'],
      dtype='object')
```

[9]:
```
data.describe()
```

```
[9]:          symboling  normalized-losses  wheel-base      length       width  \
     count  201.000000          201.00000  201.000000  201.000000  201.000000
     mean     0.840796          122.00000   98.797015    0.837102    0.915126
     std      1.254802           31.99625    6.066366    0.059213    0.029187
     min     -2.000000           65.00000   86.600000    0.678039    0.837500
     25%      0.000000          101.00000   94.500000    0.801538    0.890278
     50%      1.000000          122.00000   97.000000    0.832292    0.909722
     75%      2.000000          137.00000  102.400000    0.881788    0.925000
     max      3.000000          256.00000  120.900000    1.000000    1.000000

                height  curb-weight  engine-size        bore      stroke  \
     count  201.000000   201.000000   201.000000  201.000000  197.000000
     mean    53.766667  2555.666667   126.875622    3.330692    3.256904
     std      2.447822   517.296727    41.546834    0.268072    0.319256
     min     47.800000  1488.000000    61.000000    2.540000    2.070000
     25%     52.000000  2169.000000    98.000000    3.150000    3.110000
     50%     54.100000  2414.000000   120.000000    3.310000    3.290000
     75%     55.500000  2926.000000   141.000000    3.580000    3.410000
     max     59.800000  4066.000000   326.000000    3.940000    4.170000

            compression-ratio  horsepower     peak-rpm    city-mpg highway-mpg  \
     count         201.000000  201.000000   201.000000  201.000000  201.000000
     mean           10.164279  103.405534  5117.665368   25.179104   30.686567
     std             4.004965   37.365700   478.113805    6.423220    6.815150
     min             7.000000   48.000000  4150.000000   13.000000   16.000000
     25%             8.600000   70.000000  4800.000000   19.000000   25.000000
     50%             9.000000   95.000000  5125.369458   24.000000   30.000000
     75%             9.400000  116.000000  5500.000000   30.000000   34.000000
     max            23.000000  262.000000  6600.000000   49.000000   54.000000

                   price  city-L/100km      diesel         gas
     count    201.000000    201.000000  201.000000  201.000000
     mean   13207.129353      9.944145    0.099502    0.900498
     std     7947.066342      2.534599    0.300083    0.300083
     min     5118.000000      4.795918    0.000000    0.000000
     25%     7775.000000      7.833333    0.000000    1.000000
     50%    10295.000000      9.791667    0.000000    1.000000
     75%    16500.000000     12.368421    0.000000    1.000000
     max    45400.000000     18.076923    1.000000    1.000000
```

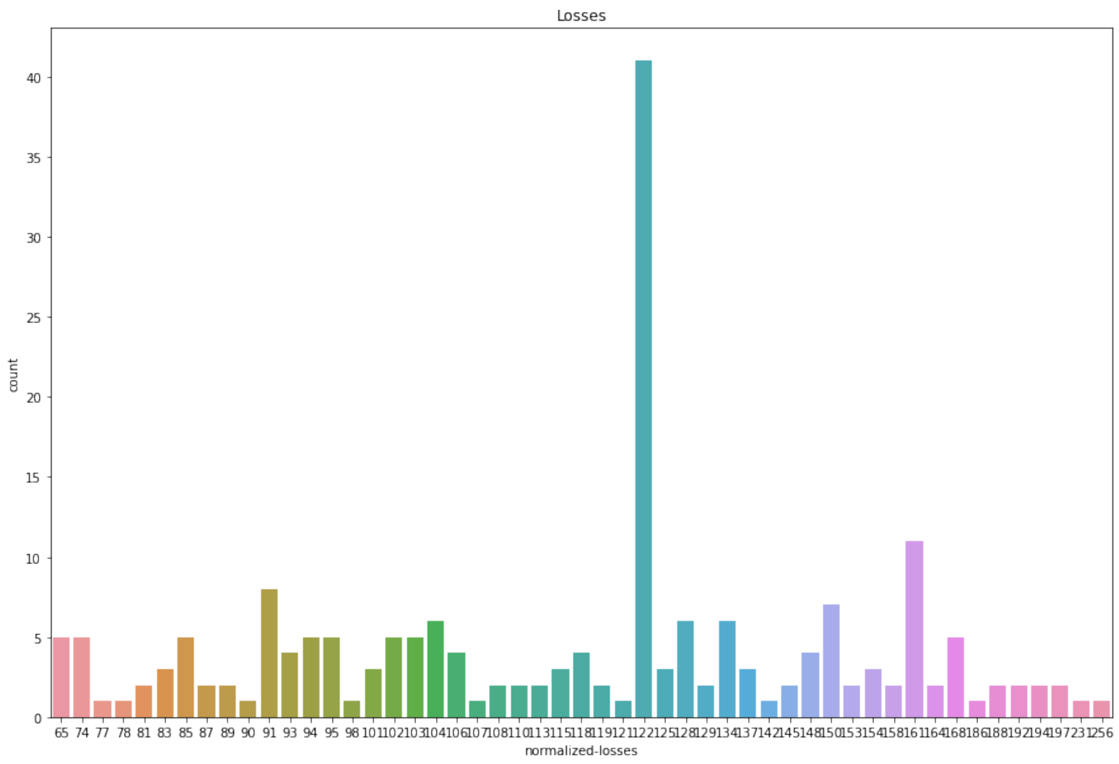
```
[10]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 201 entries, 0 to 200
Data columns (total 29 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
```

```
 0   symboling          201 non-null    int64
 1   normalized-losses  201 non-null    int64
 2   make               201 non-null    object
 3   aspiration         201 non-null    object
 4   num-of-doors       201 non-null    object
 5   body-style         201 non-null    object
 6   drive-wheels       201 non-null    object
 7   engine-location    201 non-null    object
 8   wheel-base         201 non-null    float64
 9   length             201 non-null    float64
 10  width              201 non-null    float64
 11  height             201 non-null    float64
 12  curb-weight        201 non-null    int64
 13  engine-type        201 non-null    object
 14  num-of-cylinders   201 non-null    object
 15  engine-size        201 non-null    int64
 16  fuel-system        201 non-null    object
 17  bore               201 non-null    float64
 18  stroke             197 non-null    float64
 19  compression-ratio  201 non-null    float64
 20  horsepower         201 non-null    float64
 21  peak-rpm           201 non-null    float64
 22  city-mpg           201 non-null    int64
 23  highway-mpg        201 non-null    int64
 24  price              201 non-null    int64
 25  city-L/100km       201 non-null    float64
 26  horsepower-binned  200 non-null    object
 27  diesel             201 non-null    int64
 28  gas                201 non-null    int64
dtypes: float64(10), int64(9), object(10)
memory usage: 45.7+ KB
```

[11]:
```python
# Loss
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(15,10))
sns.countplot(data['normalized-losses'])
plt.title('Losses')
plt.show()
```

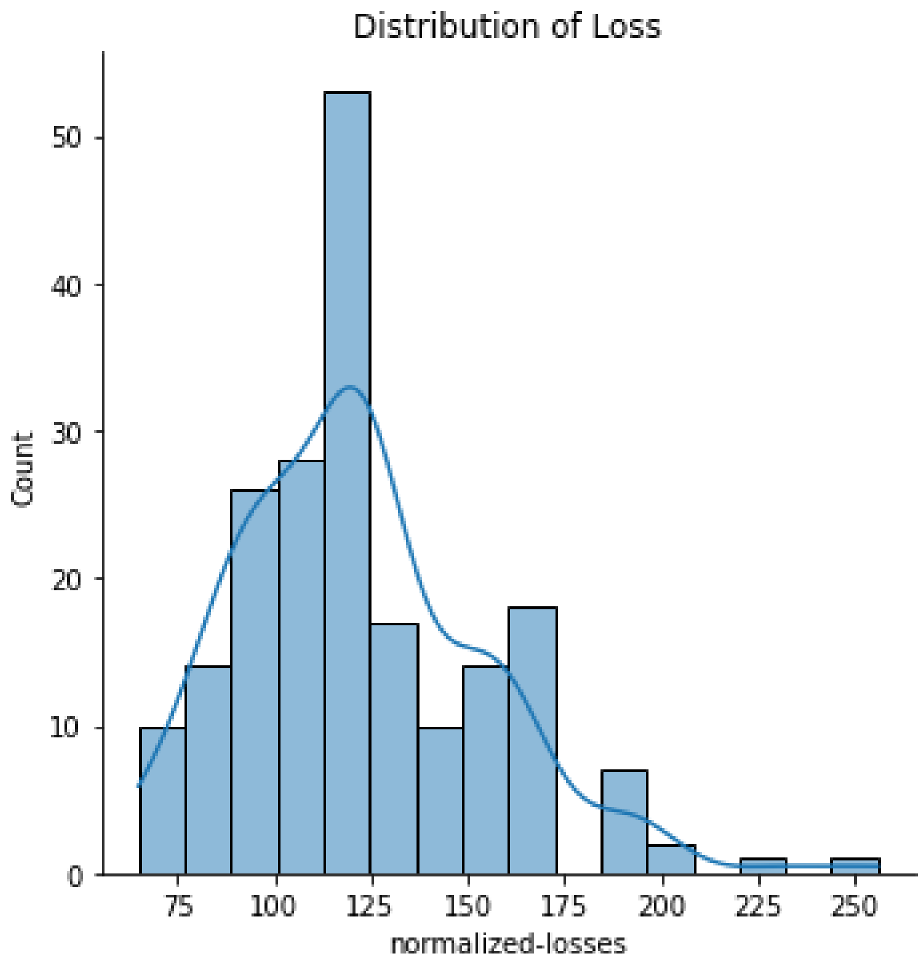
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



[13]:
```
data['normalized-losses'].describe()
```

[13]:
```
count    201.00000
mean     122.00000
std       31.99625
min       65.00000
25%      101.00000
50%      122.00000
75%      137.00000
max      256.00000
Name: normalized-losses, dtype: float64
```

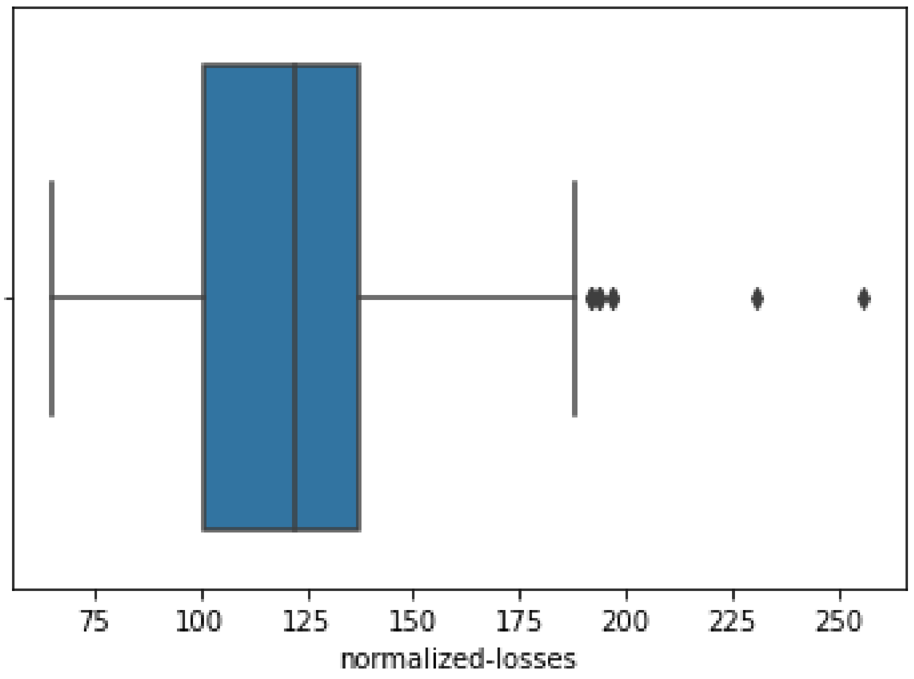
[14]:
```
sns.displot(data['normalized-losses'],kde=True)
plt.title('Distribution of Loss')
plt.show()
```



```
[15]: sns.boxplot(data['normalized-losses'])
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

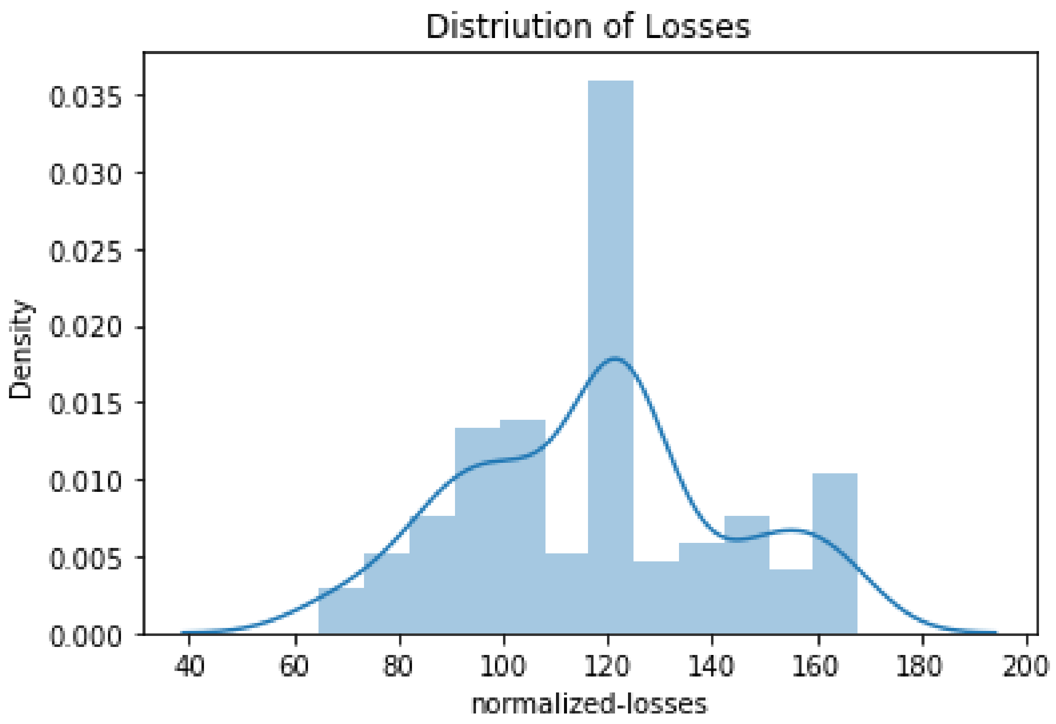
```
[15]: <AxesSubplot:xlabel='normalized-losses'>
```



```
[17]: # Outliers handling
      data['normalized-losses']=data['normalized-losses'].apply(lambda x :␣
       ↪data['normalized-losses'].mean() if (x>175) else x)
```

```
[18]: sns.distplot(data['normalized-losses'],kde=True)
      plt.title('Distriution of Losses')
      plt.show()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
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 function for
histograms).
  warnings.warn(msg, FutureWarning)
```

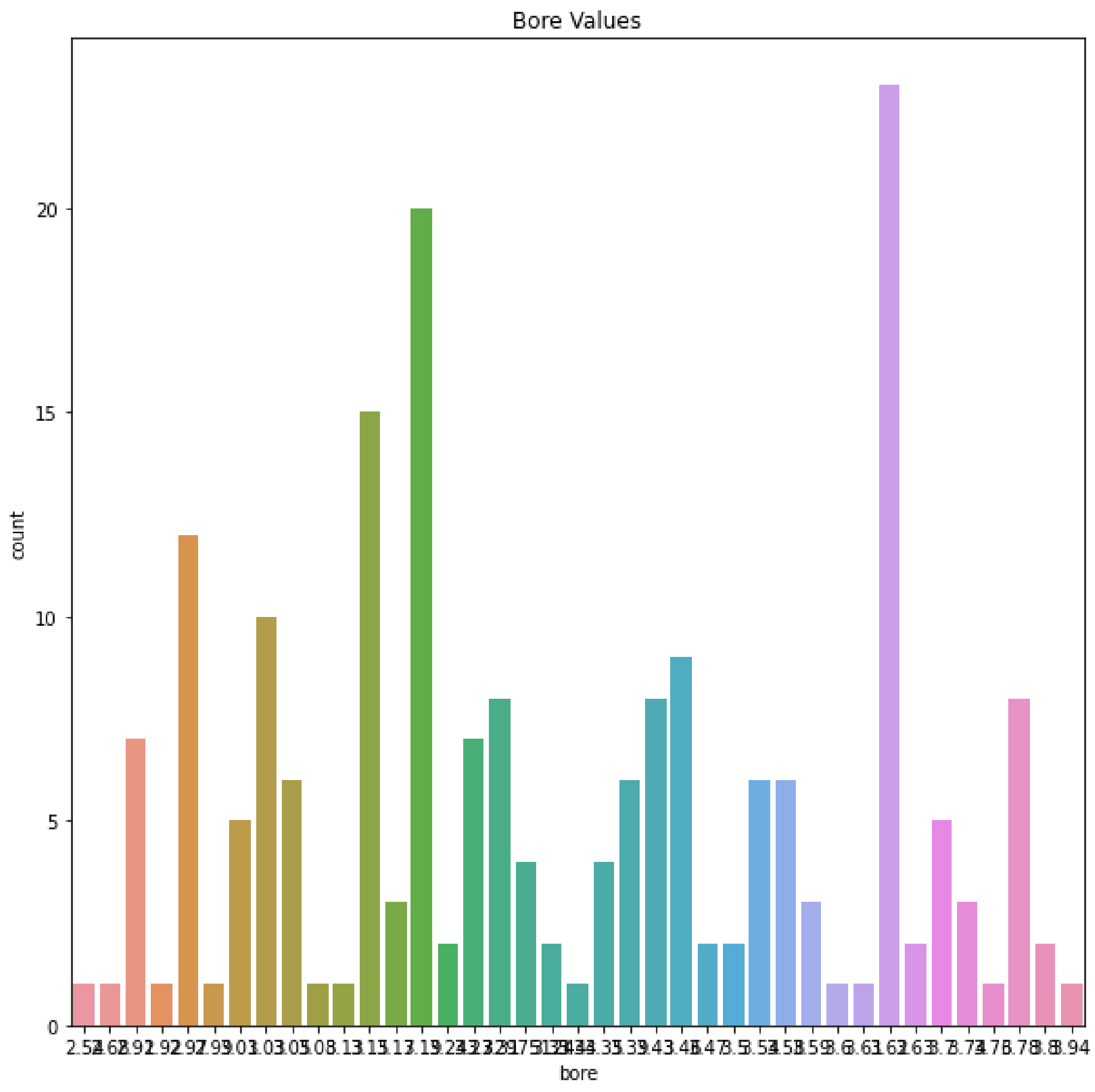


[19]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 201 entries, 0 to 200
Data columns (total 29 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          201 non-null    int64
 1   normalized-losses  201 non-null    float64
 2   make               201 non-null    object
 3   aspiration         201 non-null    object
 4   num-of-doors       201 non-null    object
 5   body-style         201 non-null    object
 6   drive-wheels       201 non-null    object
 7   engine-location    201 non-null    object
 8   wheel-base         201 non-null    float64
 9   length             201 non-null    float64
 10  width              201 non-null    float64
 11  height             201 non-null    float64
 12  curb-weight        201 non-null    int64
 13  engine-type        201 non-null    object
 14  num-of-cylinders   201 non-null    object
 15  engine-size        201 non-null    int64
 16  fuel-system        201 non-null    object
 17  bore               201 non-null    float64
 18  stroke             197 non-null    float64
 19  compression-ratio  201 non-null    float64
 20  horsepower         201 non-null    float64
 21  peak-rpm           201 non-null    float64
 22  city-mpg           201 non-null    int64
 23  highway-mpg        201 non-null    int64
 24  price              201 non-null    int64
 25  city-L/100km       201 non-null    float64
 26  horsepower-binned  200 non-null    object
 27  diesel             201 non-null    int64
 28  gas                201 non-null    int64
dtypes: float64(11), int64(8), object(10)
memory usage: 45.7+ KB
```

[20]:
```
#  Bore

plt.figure(figsize=(10,10))
sns.countplot(data['bore'])
plt.title('Bore Values')
plt.show()
```

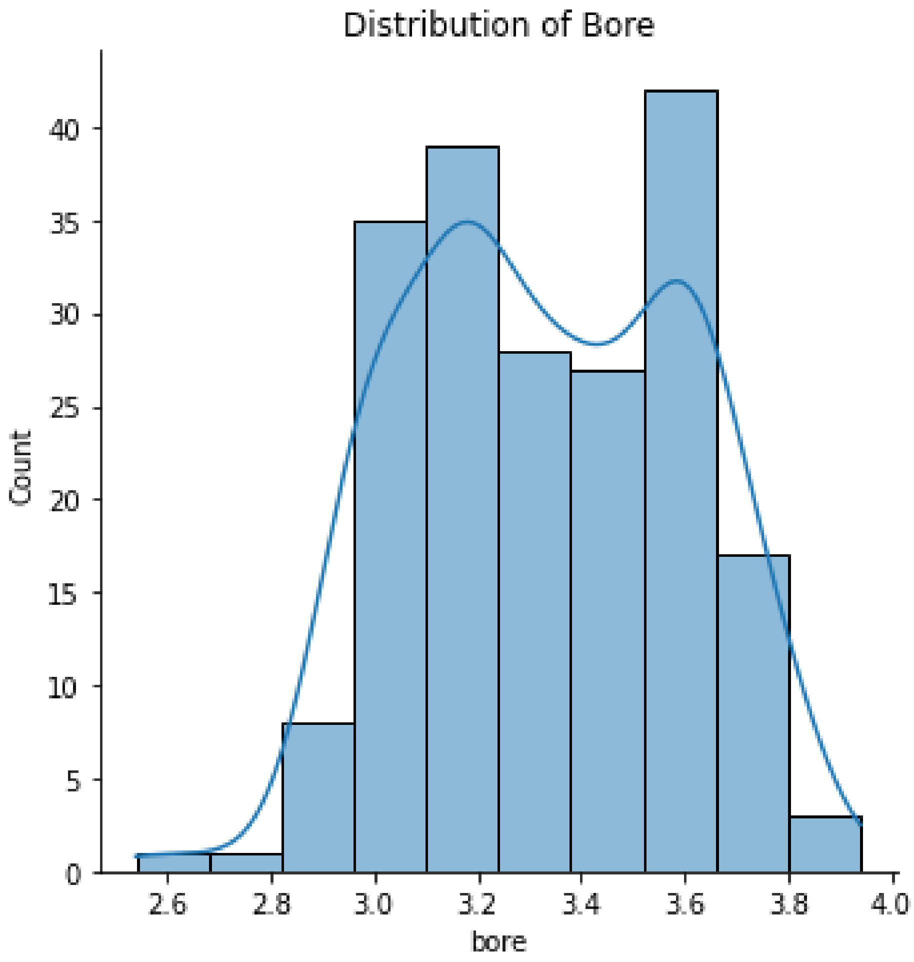
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



[21]: 
```
data['bore'].describe()
```

[21]: 
```
count    201.000000
mean       3.330692
std        0.268072
min        2.540000
25%        3.150000
50%        3.310000
75%        3.580000
max        3.940000
Name: bore, dtype: float64
```

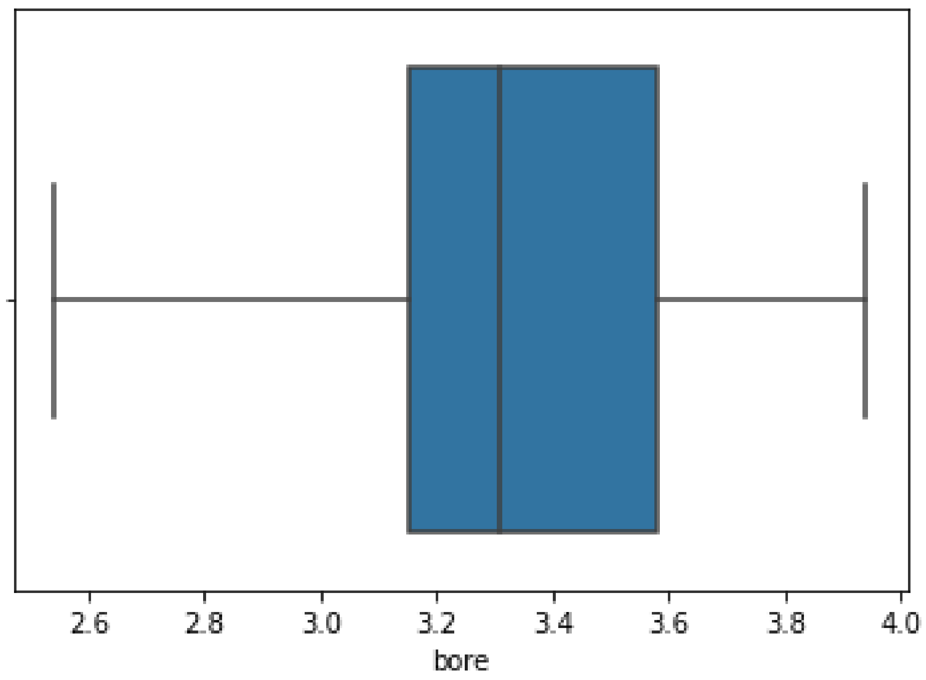
[22]:
```
sns.displot(data['bore'],kde=True)
plt.title('Distribution of Bore')
plt.show()
```



[23]:
```
# Handling outliers
sns.boxplot(data['bore'])
```

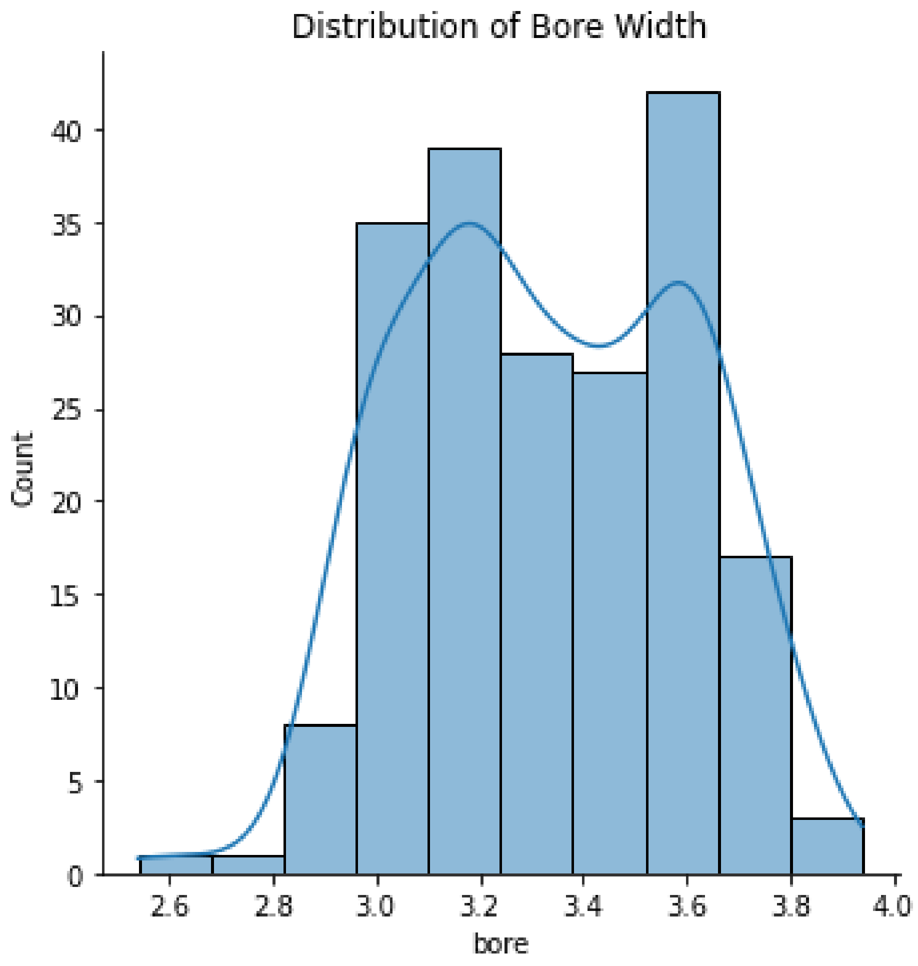
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[23]:
```
<AxesSubplot:xlabel='bore'>
```



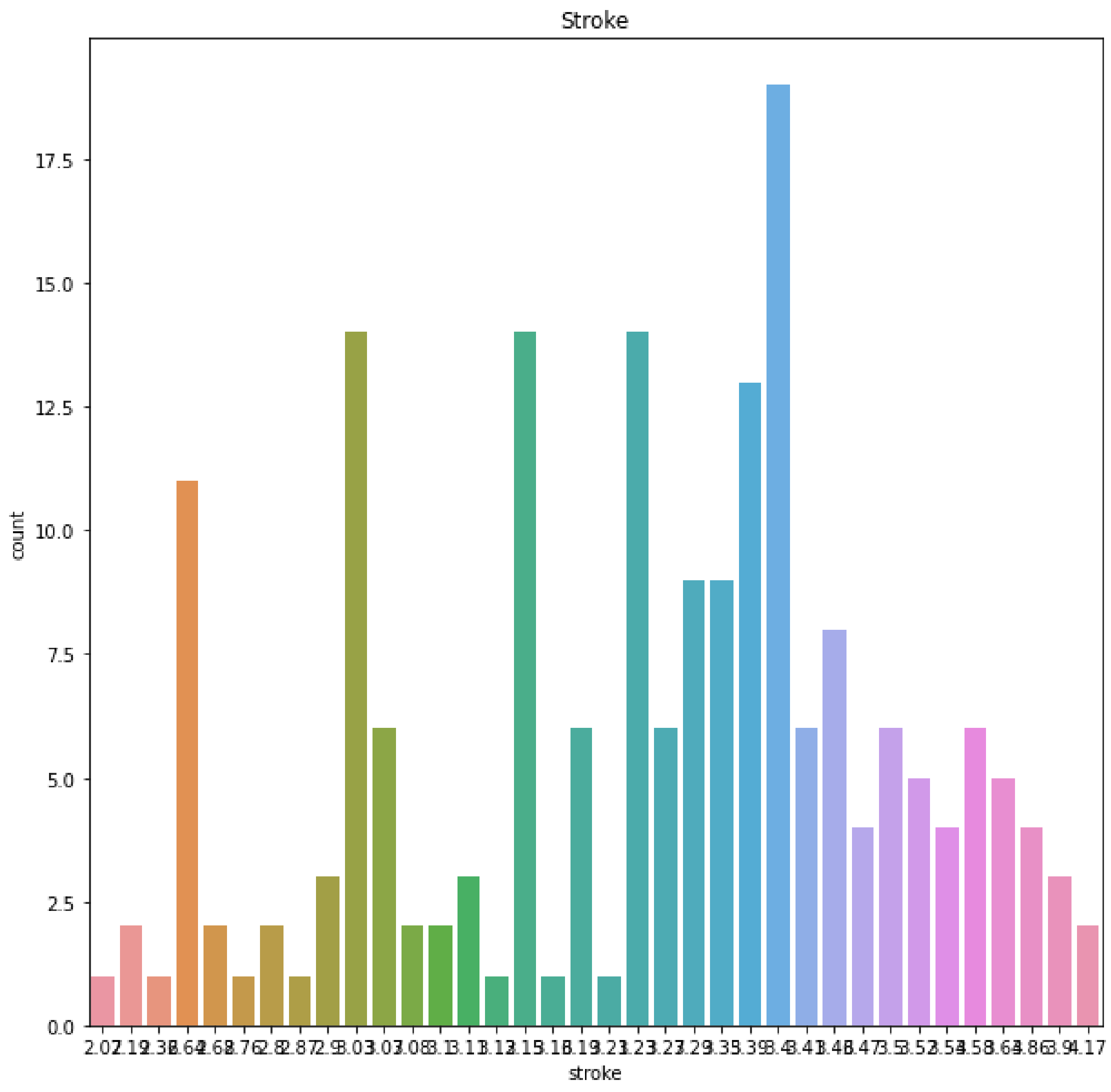
[24]:
```
data['bore']=data['bore'].apply(lambda x: data['bore'].mean() if (x==0) else x)
```

[25]:
```
sns.displot(data['bore'],kde=True)
plt.title('Distribution of Bore Width')
plt.show()
```



```
[26]: # Stroke
      plt.figure(figsize=(10,10))
      sns.countplot(data['stroke'])
      plt.title('Stroke')
      plt.show()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



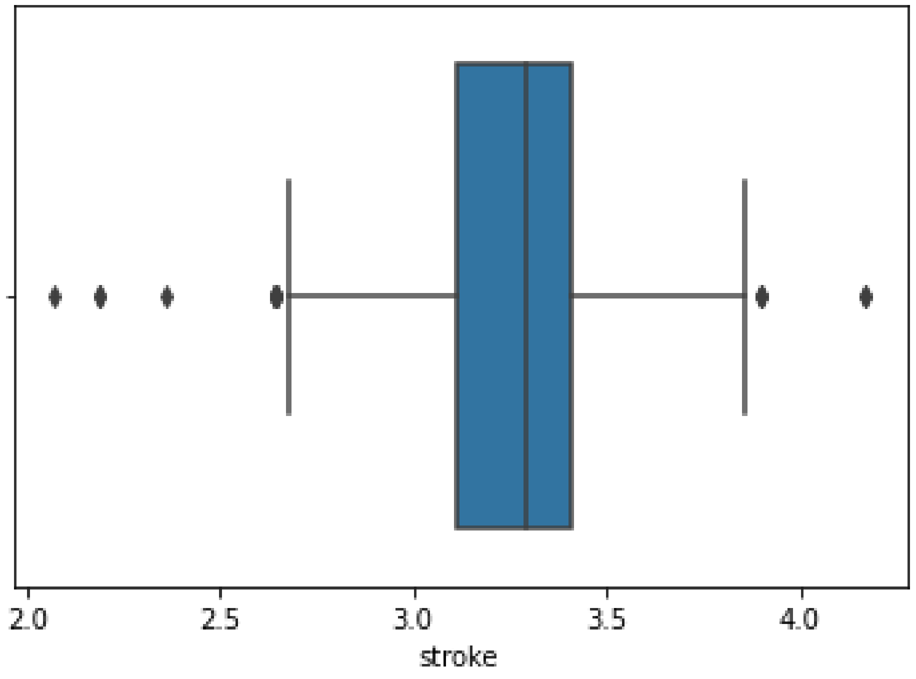
[27]:
```
data['stroke'].describe()
```

[27]:
```
count    197.000000
mean       3.256904
std        0.319256
min        2.070000
25%        3.110000
50%        3.290000
75%        3.410000
max        4.170000
Name: stroke, dtype: float64
```

[28]:
```
# Handling outliers
sns.boxplot(data['stroke'])
```

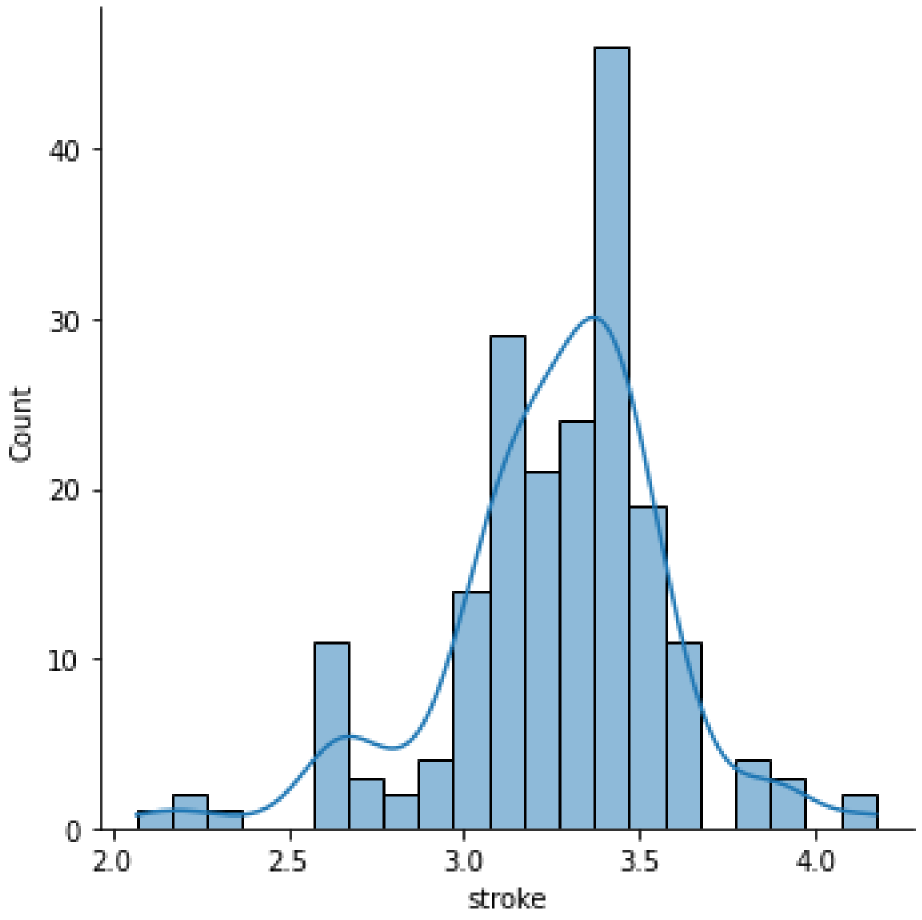
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[28]: `<AxesSubplot:xlabel='stroke'>`



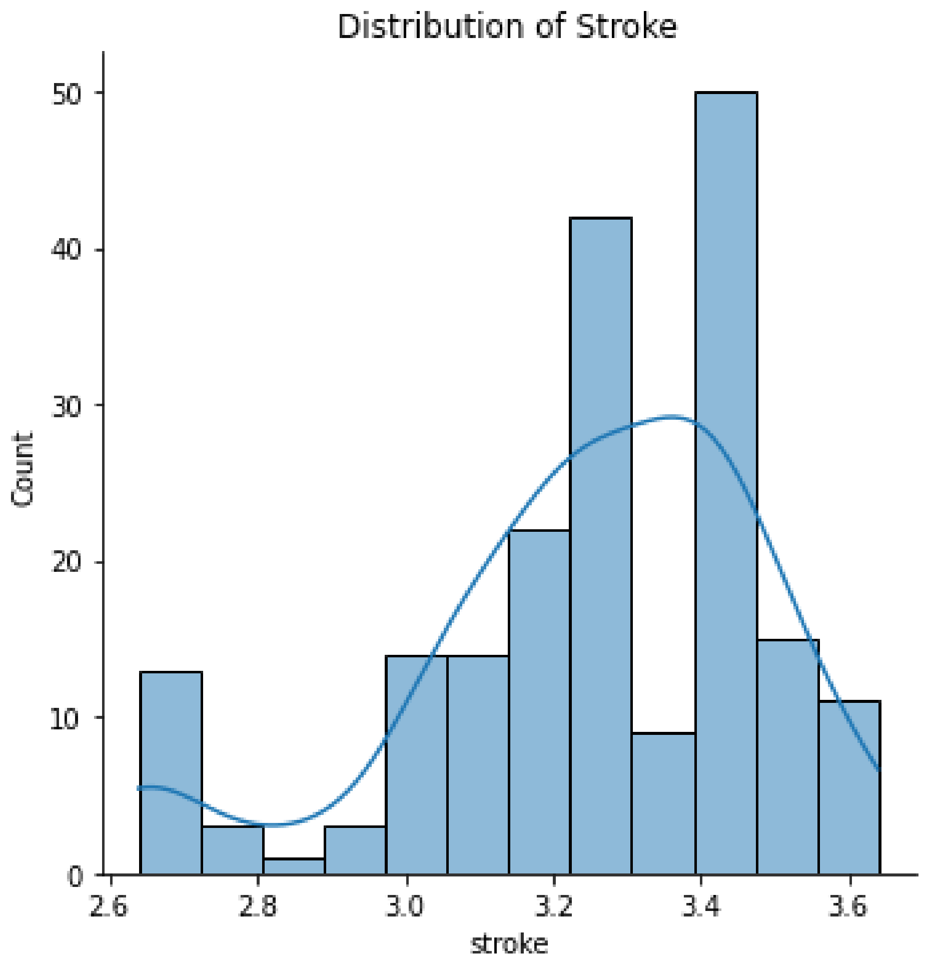
[29]:
```
sns.displot(data['stroke'],kde=True)
```

[29]: `<seaborn.axisgrid.FacetGrid at 0x2500f407a60>`



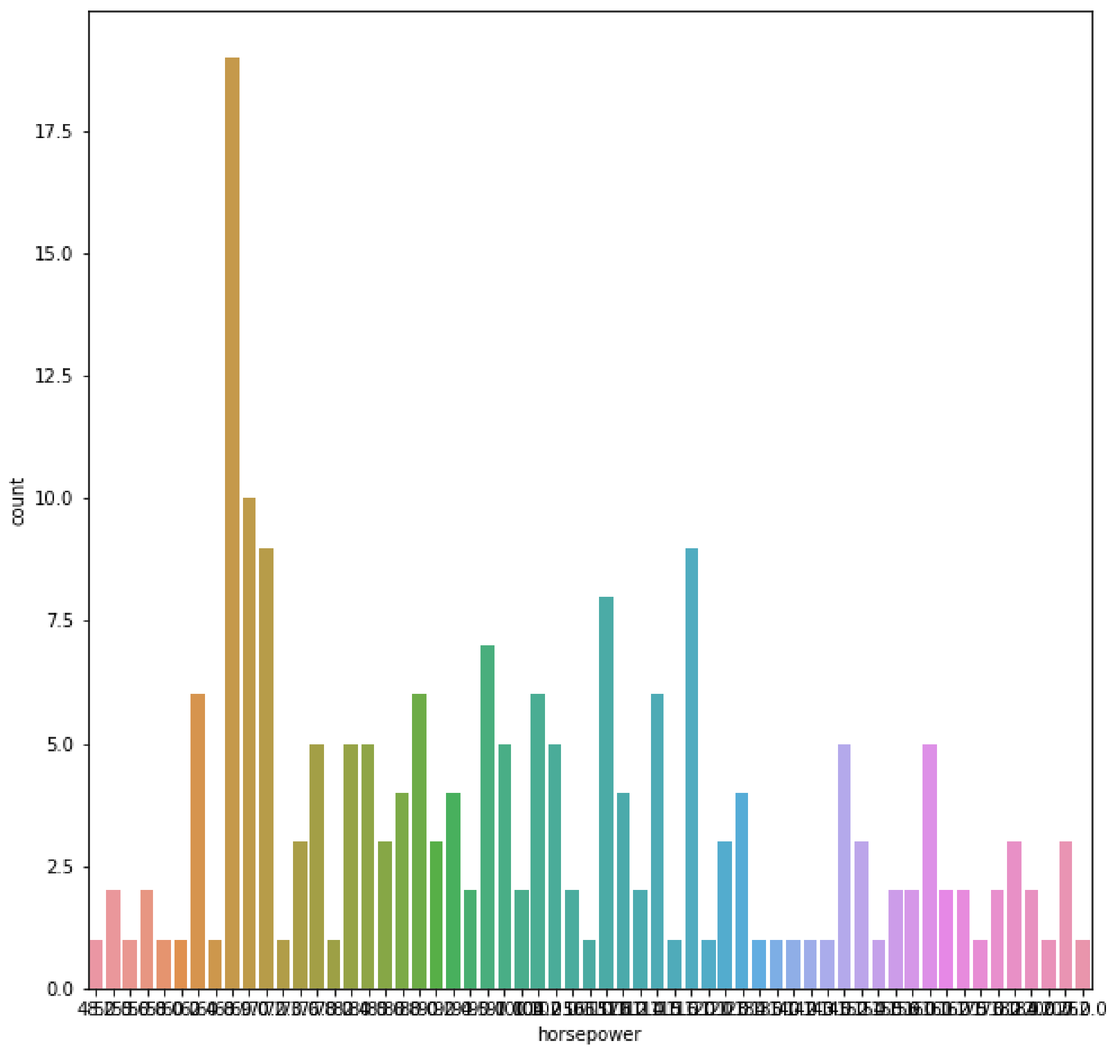
```
[30]: data['stroke']=data['stroke'].apply(lambda x: data['stroke'].mean() if (x<2.
       ↪50)or(x>3.80) else x)
```

```
[31]: sns.displot(data['stroke'],kde=True)
      plt.title('Distribution of Stroke')
      plt.show()
```



[32]:
```
# hp
plt.figure(figsize=(10,10))
sns.countplot(data['horsepower'])
plt.show()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



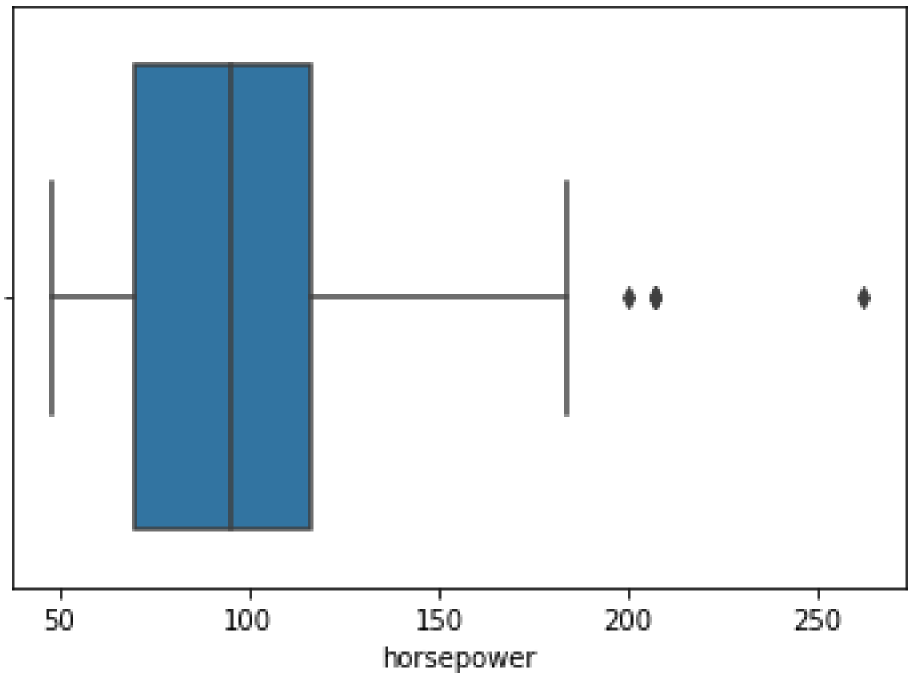
[33]:
```
data['horsepower'].describe()
```

[33]:
```
count    201.000000
mean     103.405534
std       37.365700
min       48.000000
25%       70.000000
50%       95.000000
75%      116.000000
max      262.000000
Name: horsepower, dtype: float64
```

[34]:
```
# Handling Outliers
sns.boxplot(data['horsepower'])
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

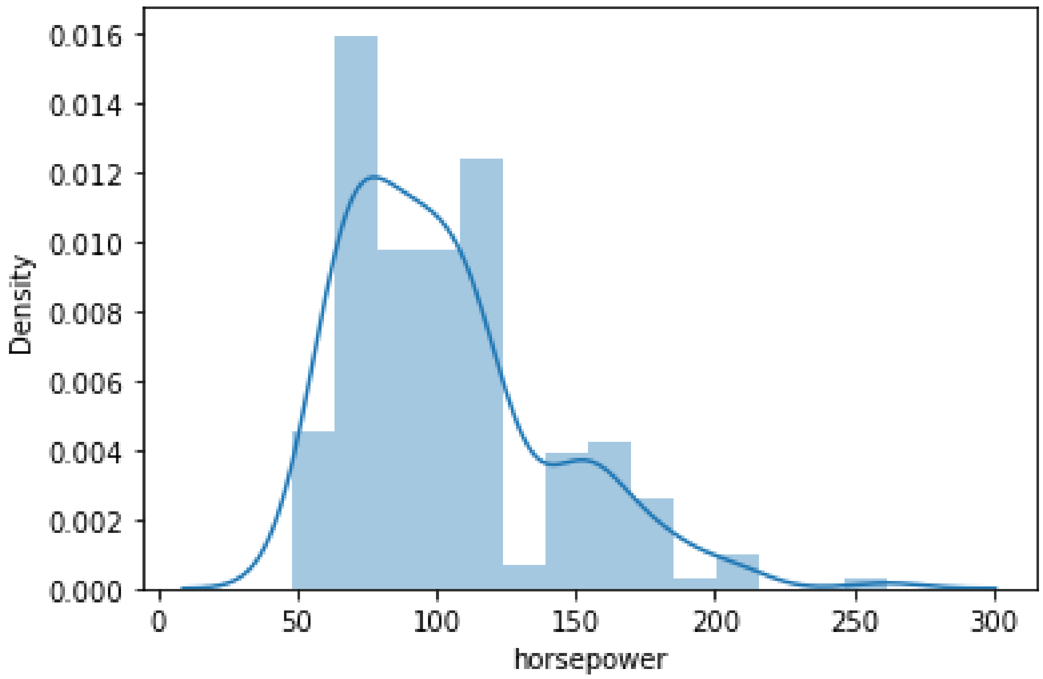
[34]:
```
<AxesSubplot:xlabel='horsepower'>
```



[35]:
```
sns.distplot(data['horsepower'],kde=True)
```

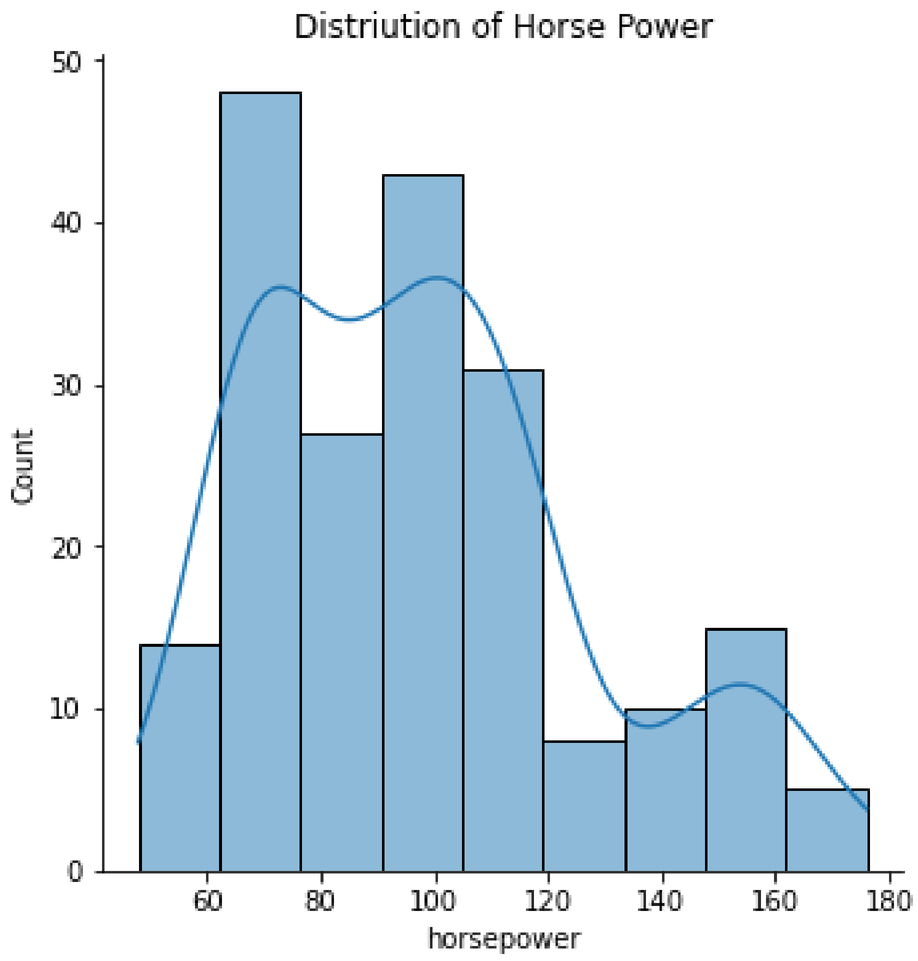
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[35]:
```
<AxesSubplot:xlabel='horsepower', ylabel='Density'>
```



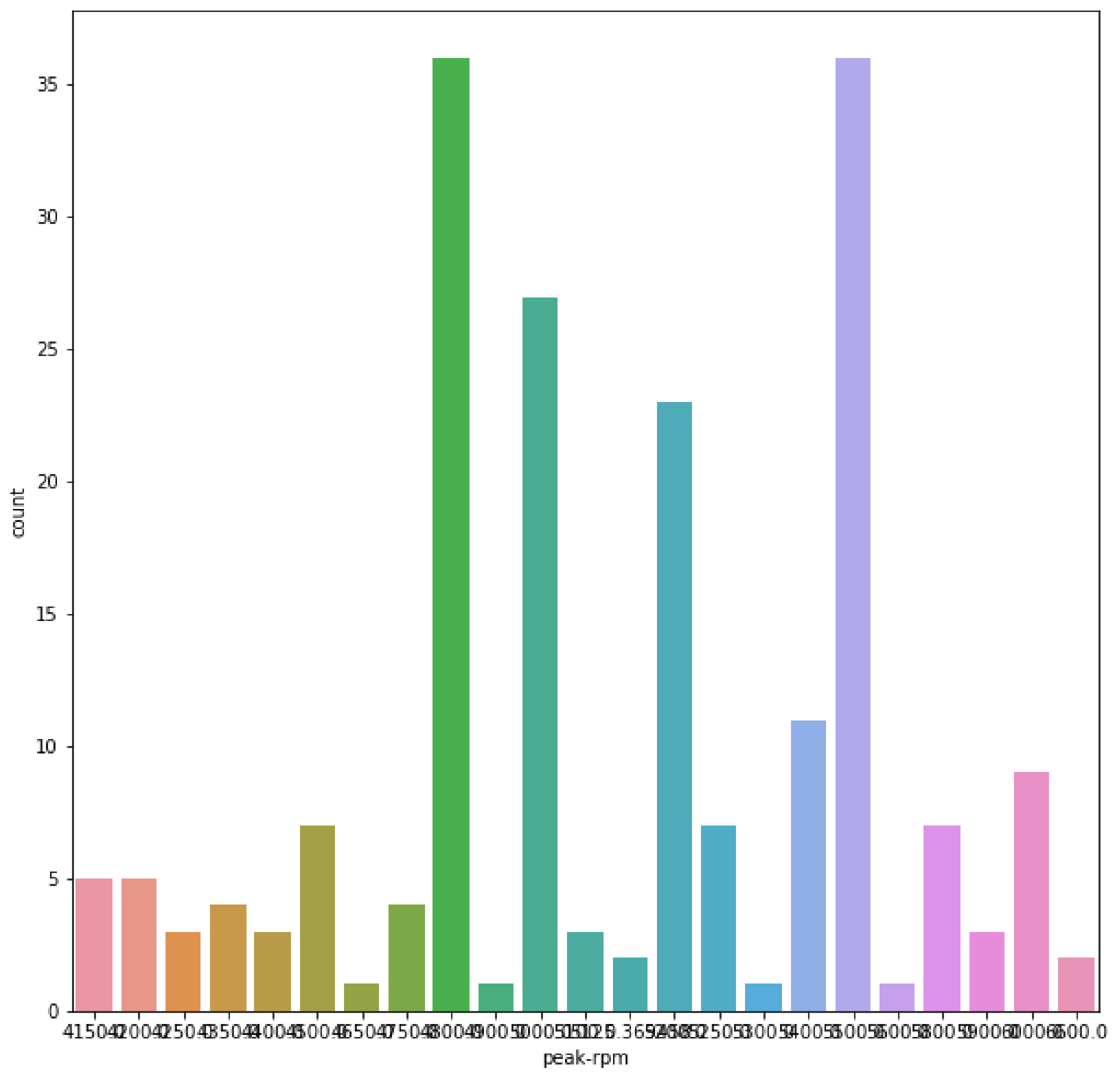
```
[36]: data['horsepower']=data['horsepower'].apply(lambda x : data['horsepower'].
       ↪mean() if (x>180) else x)
```

```
[37]: sns.displot(data['horsepower'],kde=True)
      plt.title('Distriution of Horse Power')
      plt.show()
```



```
[38]: # Peak RPM
      plt.figure(figsize=(10,10))
      sns.countplot(data['peak-rpm'])
      plt.show()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

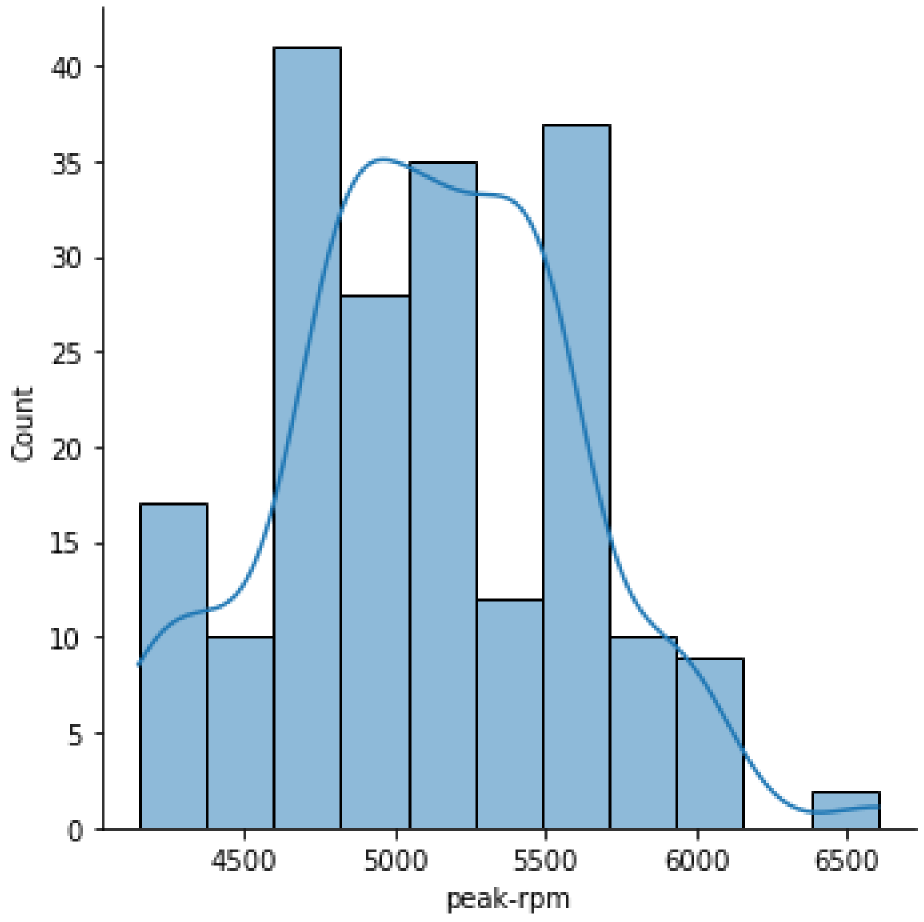


```
[39]: data['peak-rpm'].describe()
```

```
[39]: count     201.000000
      mean     5117.665368
      std       478.113805
      min      4150.000000
      25%      4800.000000
      50%      5125.369458
      75%      5500.000000
      max      6600.000000
      Name: peak-rpm, dtype: float64
```

```
[40]: # Handling Outliers
      sns.displot(data['peak-rpm'],kde=True)
```

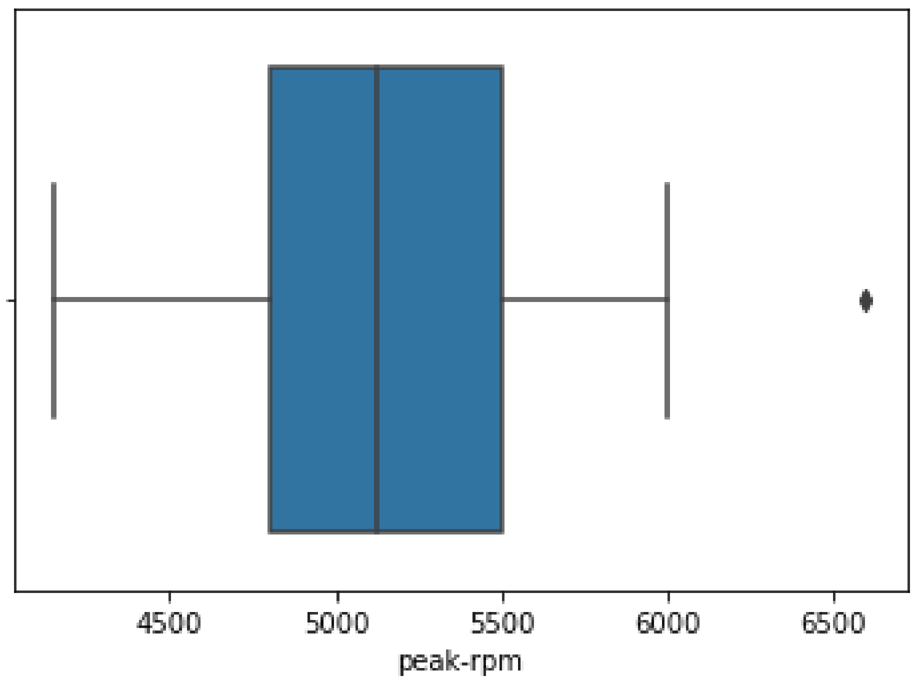
[40]: <seaborn.axisgrid.FacetGrid at 0x2500ee112b0>



[41]:
```
sns.boxplot(data['peak-rpm'])
```

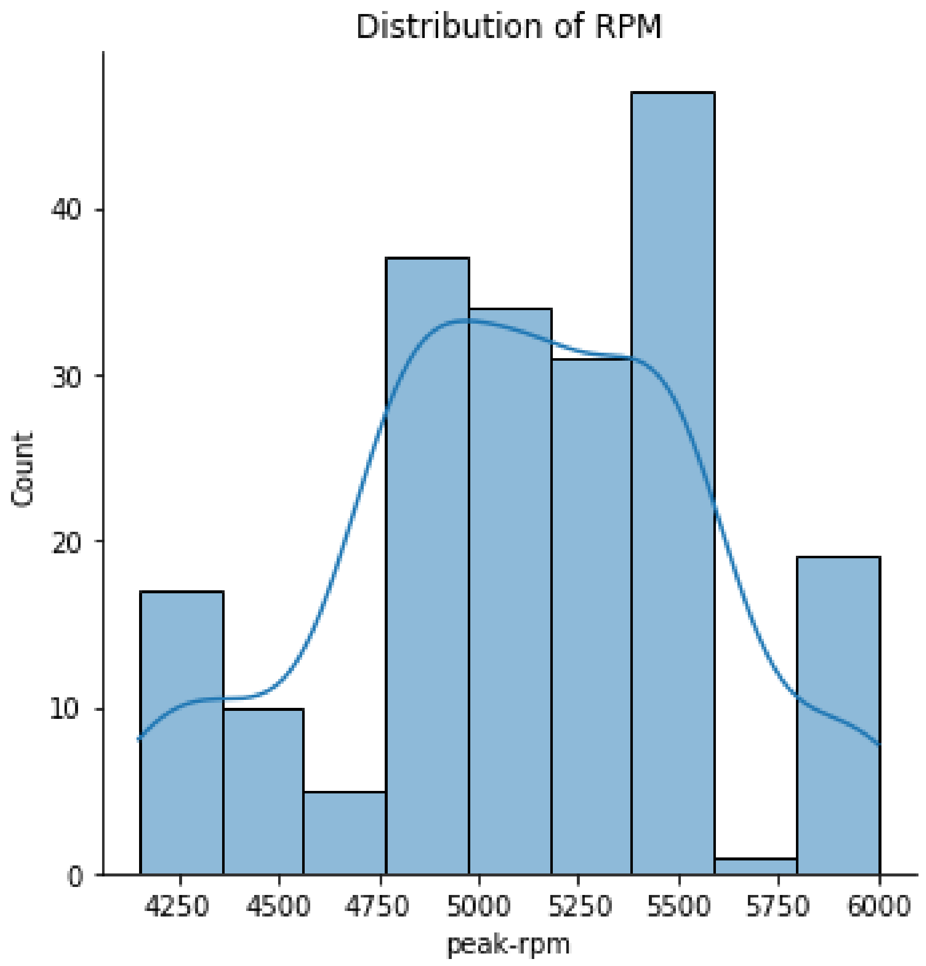
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[41]: <AxesSubplot:xlabel='peak-rpm'>



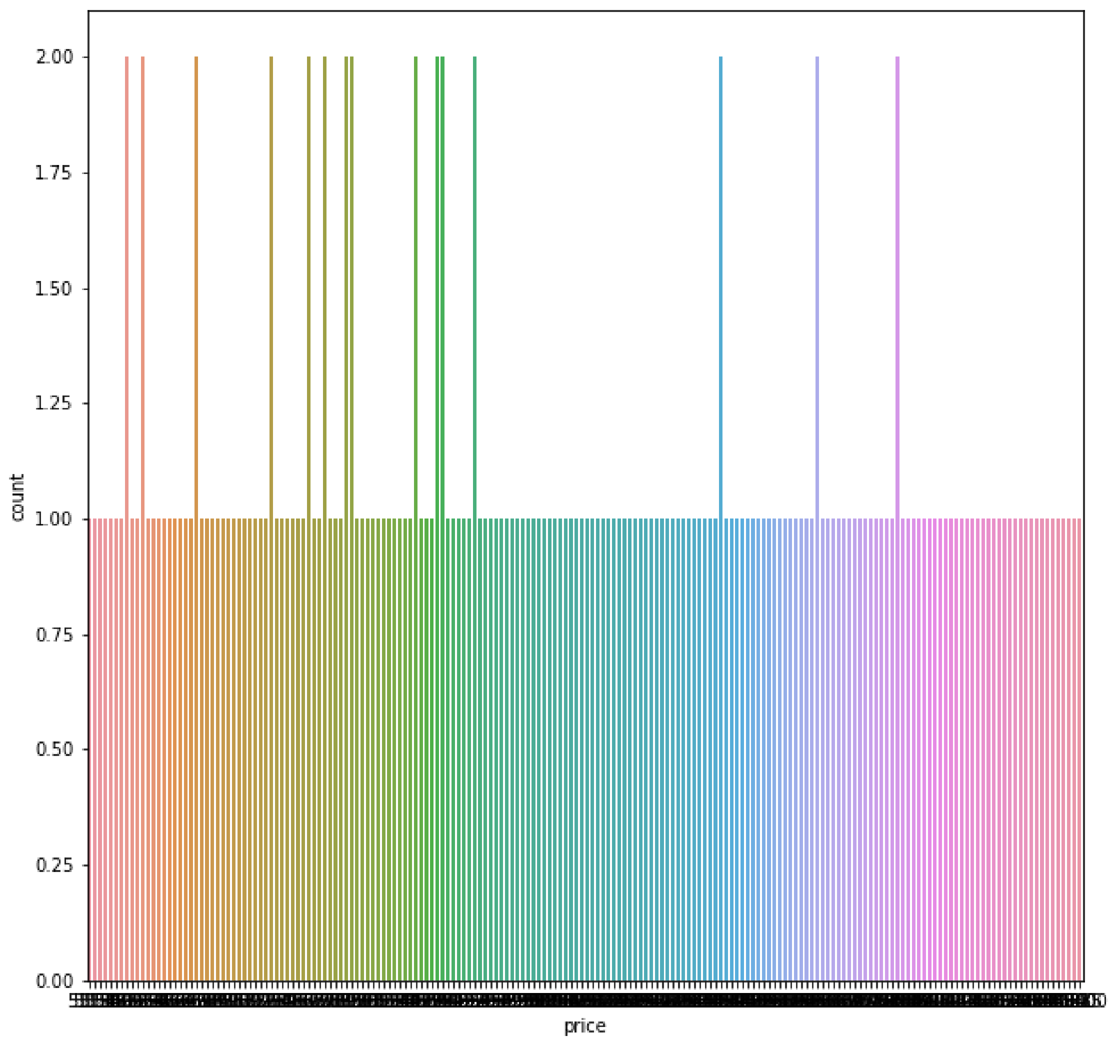
[42]:
```
data['peak-rpm']=data['peak-rpm'].apply(lambda x:data['peak-rpm'].mean() if␣
 ↪(x>6000) else x)
```

[43]:
```
sns.displot(data['peak-rpm'],kde=True)
plt.title('Distribution of RPM')
plt.show()
```



[49]:
```
# Price
plt.figure(figsize=(10,10))
sns.countplot(data['price'])
plt.show()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```



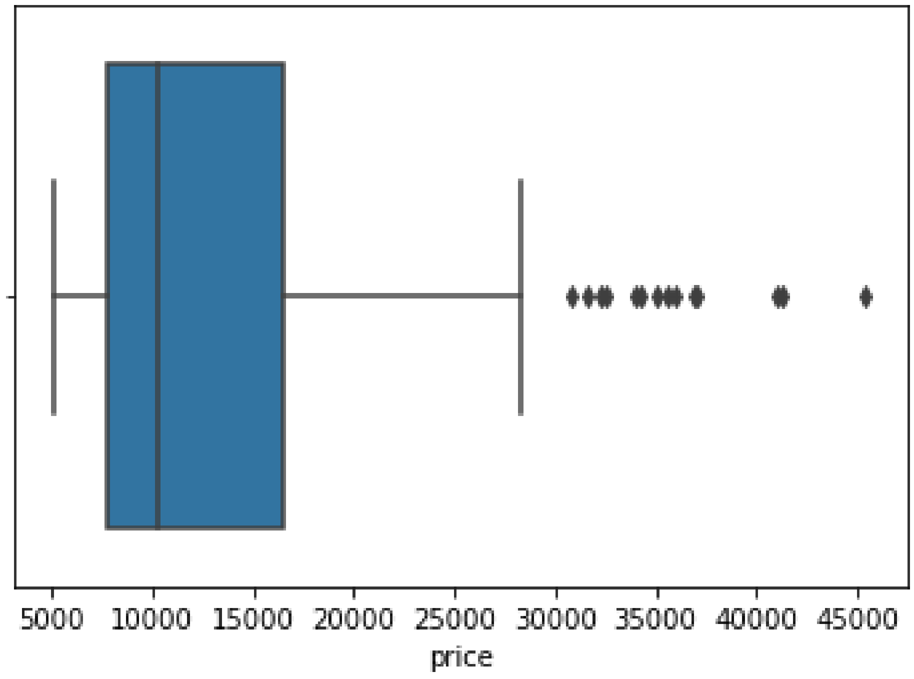
```
[45]: data['price'].describe()
```

```
[45]: count      201.000000
      mean     13207.129353
      std       7947.066342
      min       5118.000000
      25%       7775.000000
      50%      10295.000000
      75%      16500.000000
      max      45400.000000
      Name: price, dtype: float64
```

```
[50]: # Handling Outliers
      sns.boxplot(data['price'])
```

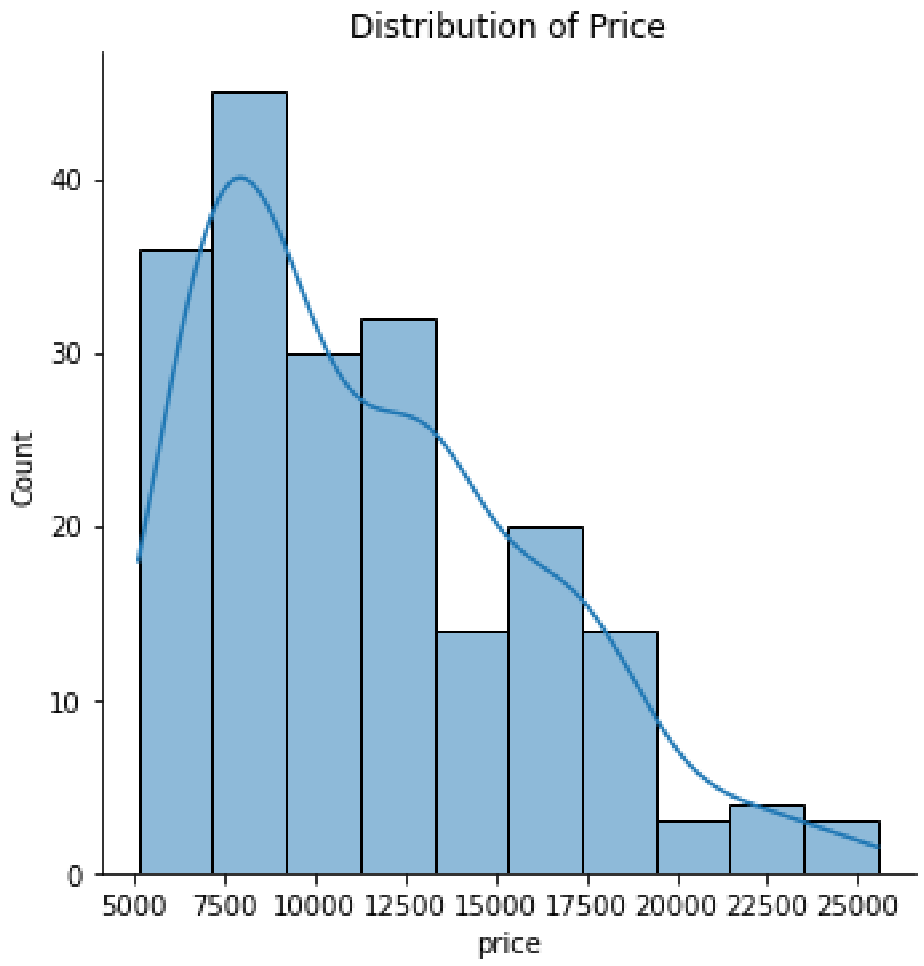
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[50]:
```
<AxesSubplot:xlabel='price'>
```



[51]:
```
data['price']=data['price'].apply(lambda x:data['price'].mean() if (x>28000)␣
 ↪else x)
```

[52]:
```
sns.displot(data['price'],kde=True)
plt.title('Distribution of Price')
plt.show()
```

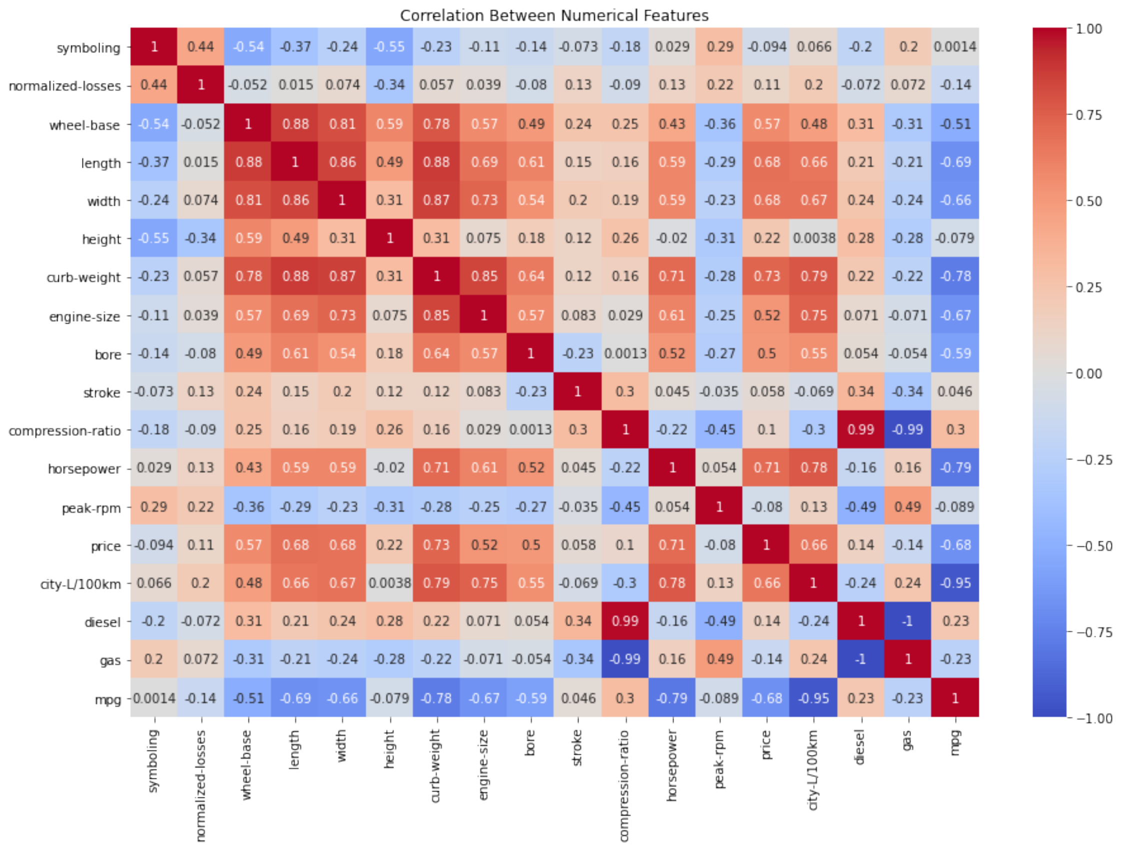


[58]:
```
data.select_dtypes(include='number').head(2)
```

[58]:
```
   symboling  normalized-losses  wheel-base    length     width  height  \
0          3              122.0        88.6  0.811148  0.890278    48.8
1          3              122.0        88.6  0.811148  0.890278    48.8

   curb-weight  engine-size  bore  stroke  compression-ratio  horsepower  \
0         2548          130  3.47    2.68                9.0       111.0
1         2548          130  3.47    2.68                9.0       111.0

   peak-rpm    price  city-L/100km  diesel  gas   mpg
0    5000.0  13495.0     11.190476       0    1  24.0
1    5000.0  16500.0     11.190476       0    1  24.0
```

[59]:
```
plt.figure(figsize=(15,10))
sns.heatmap(data.select_dtypes(include='number').
 ↪corr(),annot=True,cmap='coolwarm')
```

```
plt.title('Correlation Between Numerical Features')
plt.show()
```



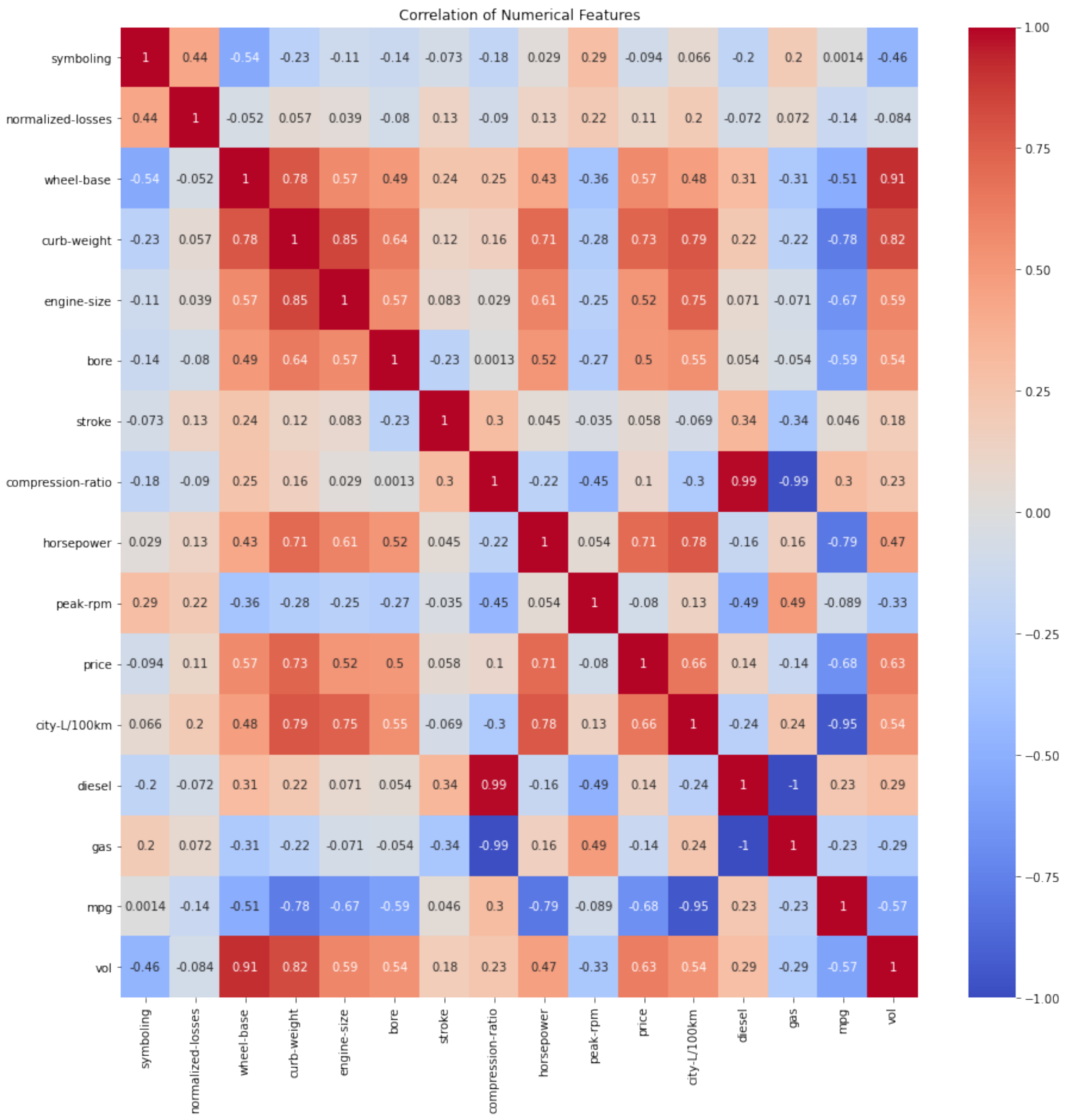
Combing height,weight,volumes

```
[61]: data['vol'] = (data['width']*data['length']*data['height'])/(12.54**3)
      data.drop(['width','length','height'],axis=1,inplace=True)
```

```
[63]: data.select_dtypes(include='number').head(2)
```

```
[63]:    symboling  normalized-losses  wheel-base  curb-weight  engine-size  bore  \
      0          3              122.0        88.6         2548          130  3.47
      1          3              122.0        88.6         2548          130  3.47

         stroke  compression-ratio  horsepower  peak-rpm    price  city-L/100km  \
      0    2.68                9.0       111.0    5000.0  13495.0     11.190476
      1    2.68                9.0       111.0    5000.0  16500.0     11.190476

         diesel  gas   mpg       vol
      0       0    1  24.0  0.017871
      1       0    1  24.0  0.017871
```

[65]:
```python
plt.figure(figsize=(15,15))
sns.heatmap(data.select_dtypes(include='number').
 ↪corr(),annot=True,cmap='coolwarm')
plt.title('Correlation of Numerical Features')
plt.show()
```

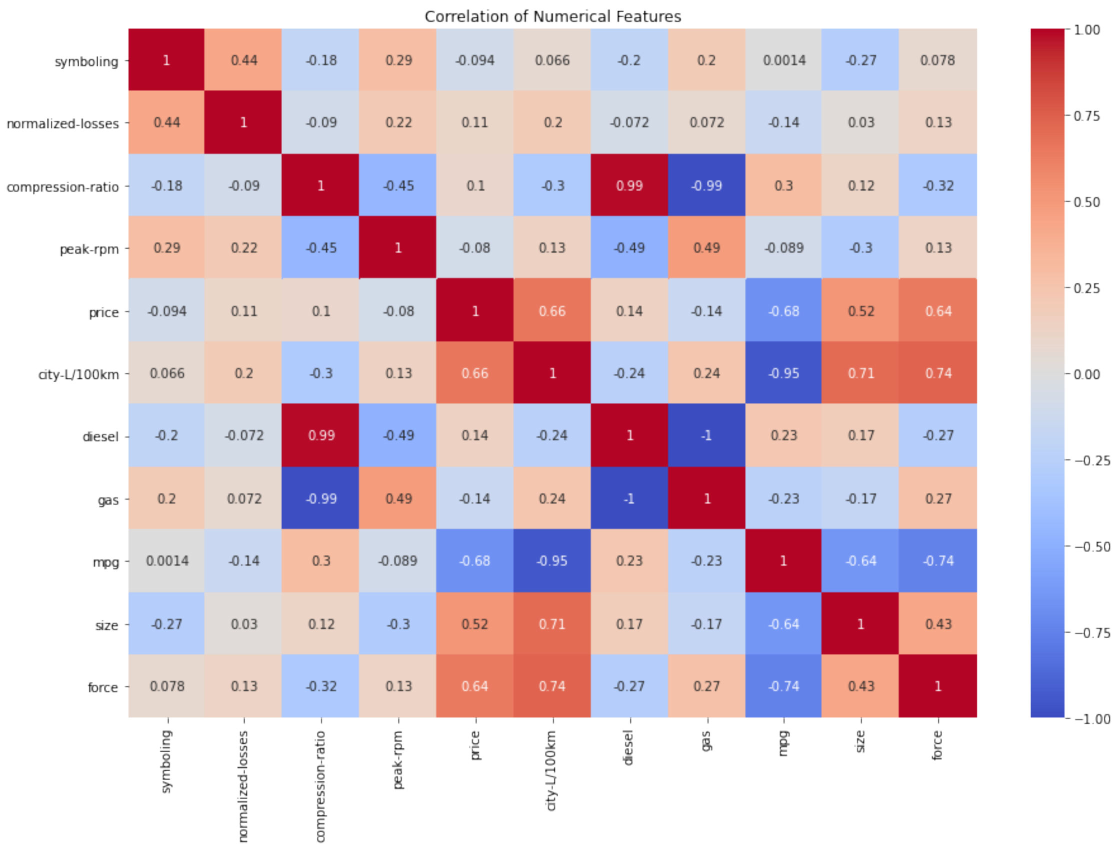


[66]:
```python
# Feature Engineering with other features

data['weights']=(data['wheel-base']*data['curb-weight']*data['engine-size'])/3
data.drop(['wheel-base','curb-weight','engine-size'],axis=1,inplace=True)
```

[67]:
```
data['size'] = (data['vol']*data['weights'])/9.81
data.drop(['vol','weights'],axis=1,inplace=True)
```

[68]:
```
data['force'] = (data['horsepower']/(data['bore']*data['stroke']))
data.drop(['horsepower','bore','stroke'],axis=1,inplace=True)
```

[69]:
```
plt.figure(figsize=(15,10))
sns.heatmap(data.select_dtypes(include='number').
  ↪corr(),annot=True,cmap='coolwarm')
plt.title('Correlation of Numerical Features')
plt.show()
```

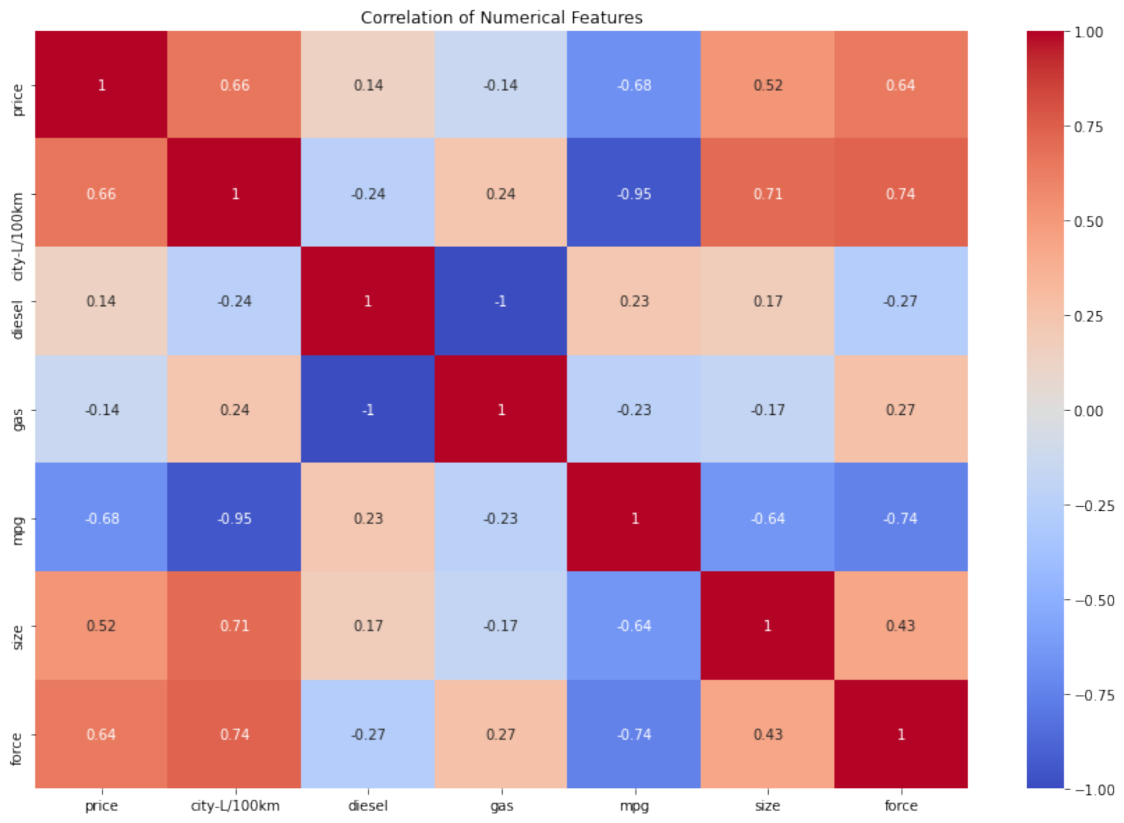


[70]:
```
data.columns
```

[70]:
```
Index(['symboling', 'normalized-losses', 'make', 'aspiration', 'num-of-doors',
       'body-style', 'drive-wheels', 'engine-location', 'engine-type',
       'num-of-cylinders', 'fuel-system', 'compression-ratio', 'peak-rpm',
       'price', 'city-L/100km', 'horsepower-binned', 'diesel', 'gas', 'mpg',
       'size', 'force'],
      dtype='object')
```

[71]:
```
data.
  ↪drop(['symboling','normalized-losses','compression-ratio','peak-rpm'],axis=1,inplace=True)
```

[72]:
```
plt.figure(figsize=(15,10))
sns.heatmap(data.select_dtypes(include='number').
  ↪corr(),annot=True,cmap='coolwarm')
plt.title('Correlation of Numerical Features')
plt.show()
```



|  | price | city-L/100km | diesel | gas | mpg | size | force |
|---|---|---|---|---|---|---|---|
| price | 1 | 0.66 | 0.14 | -0.14 | -0.68 | 0.52 | 0.64 |
| city-L/100km | 0.66 | 1 | -0.24 | 0.24 | -0.95 | 0.71 | 0.74 |
| diesel | 0.14 | -0.24 | 1 | -1 | 0.23 | 0.17 | -0.27 |
| gas | -0.14 | 0.24 | -1 | 1 | -0.23 | -0.17 | 0.27 |
| mpg | -0.68 | -0.95 | 0.23 | -0.23 | 1 | -0.64 | -0.74 |
| size | 0.52 | 0.71 | 0.17 | -0.17 | -0.64 | 1 | 0.43 |
| force | 0.64 | 0.74 | -0.27 | 0.27 | -0.74 | 0.43 | 1 |

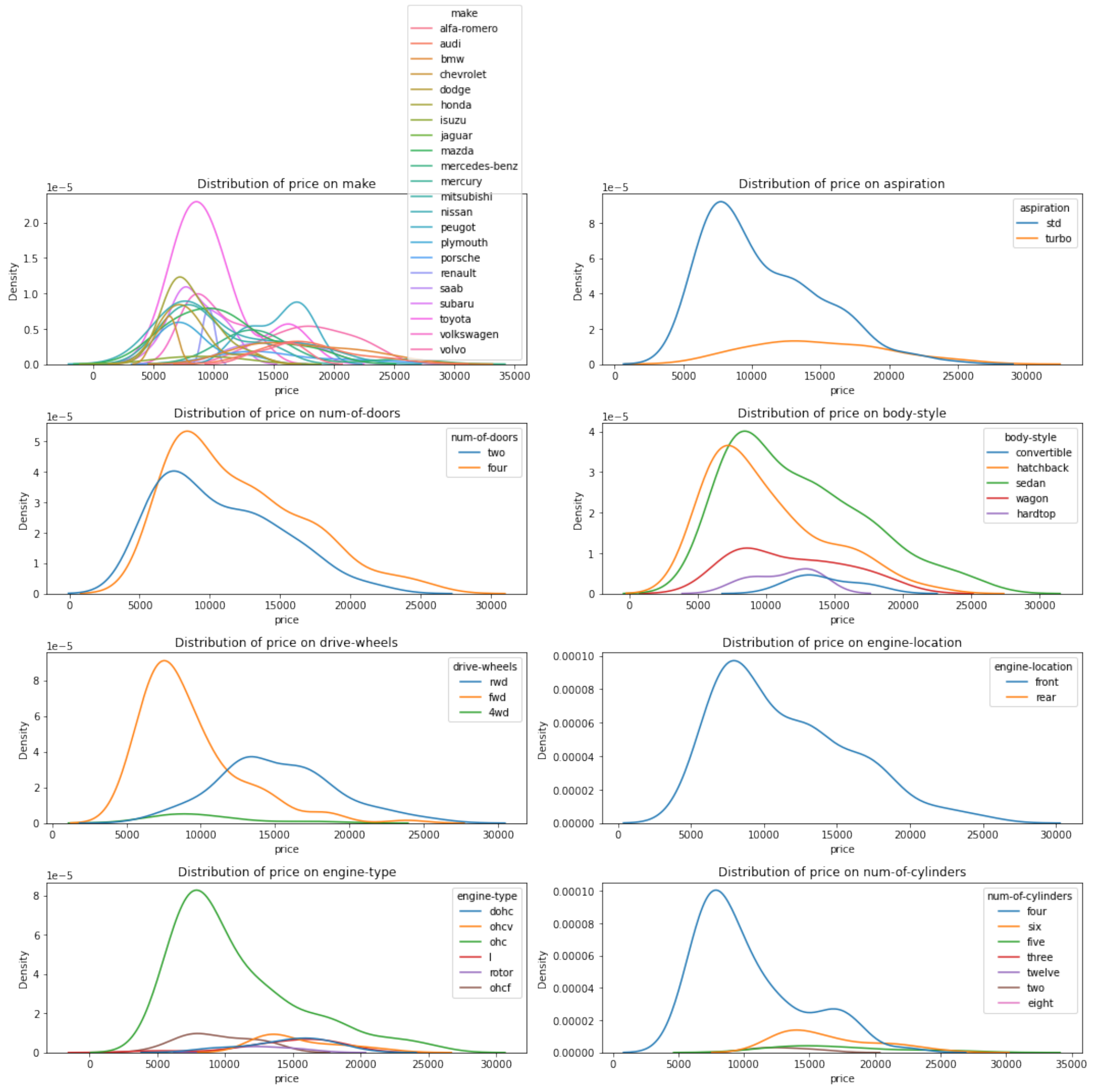
[73]:
```
# Analysing categorical features

data.select_dtypes(exclude='number').head(2)
```

[73]:
```
          make aspiration num-of-doors   body-style drive-wheels  \
0  alfa-romero        std          two  convertible          rwd
1  alfa-romero        std          two  convertible          rwd

  engine-location engine-type num-of-cylinders fuel-system horsepower-binned
0           front        dohc             four        mpfi            Medium
1           front        dohc             four        mpfi            Medium
```

```
[74]: cat_col = data.select_dtypes(exclude='number').columns
```

```
[82]: fig,axes = plt.subplots(4,2)
      fig.set_figheight(15)
      fig.set_figwidth(15)

      for ax,col in zip(axes.flatten(),cat_col):
          sns.kdeplot(ax=ax,data=data,x='price',hue=col)
          ax.set_title('Distribution of price on '+str(col))
          plt.tight_layout()
```

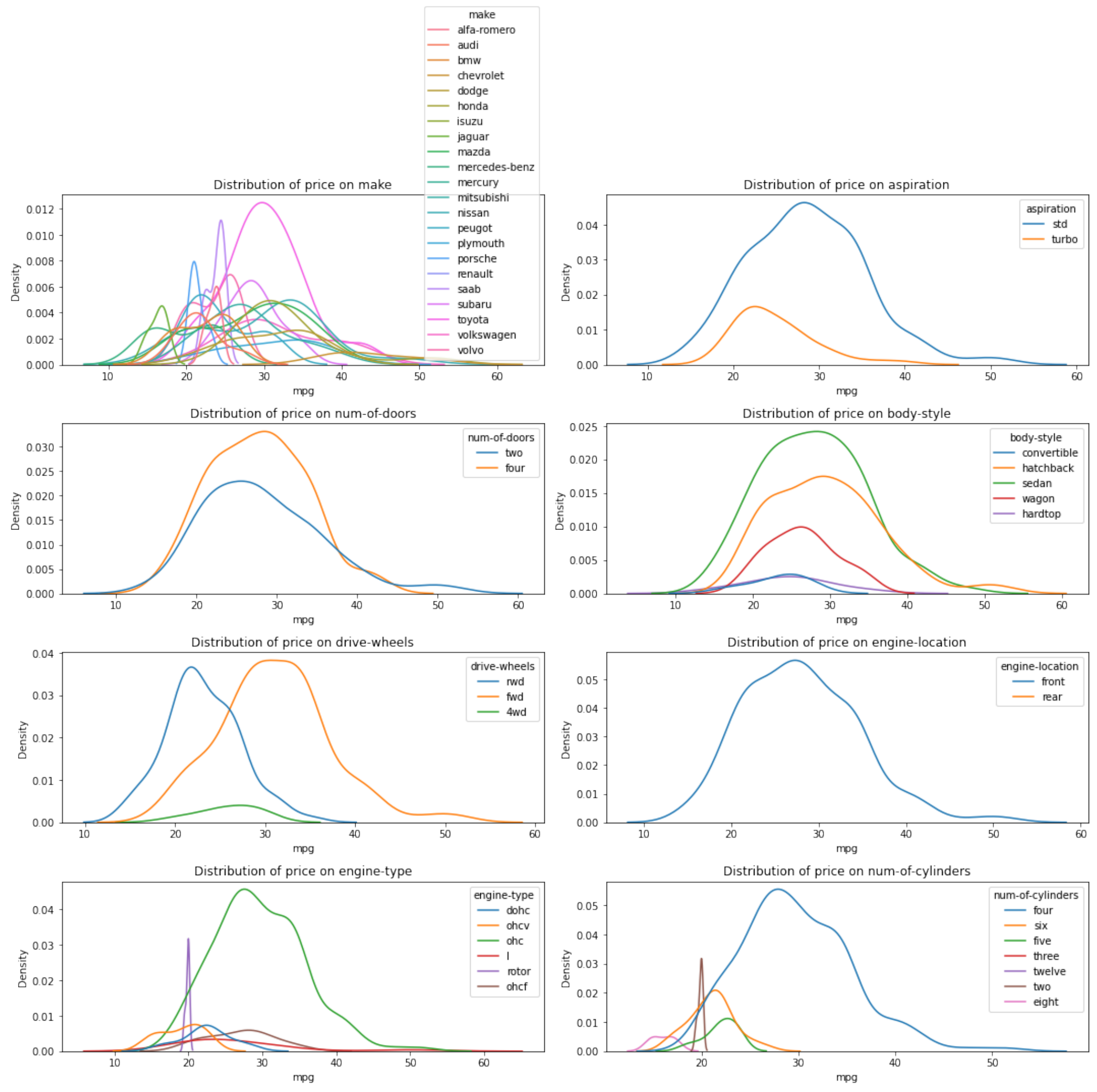
```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```



```
[76]: fig,axes = plt.subplots(4,2)
      fig.set_figheight(15)
      fig.set_figwidth(15)

      for ax,col in zip(axes.flatten(),cat_col):
          sns.kdeplot(ax=ax,data=data,x='mpg',hue=col)
          ax.set_title('Distribution of price on '+str(col))
          plt.tight_layout()
```

```
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
```

```
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\87548\anaconda3\lib\site-packages\seaborn\distributions.py:306:
UserWarning: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```



```
[ ]:
```