In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
energy_data = pd.read_csv('energy_dataset.csv')
```

In [3]:

```
energy_data.head()
```

Out[3]:

	time	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil	ge f
0	2015-01-01 00:00:00+01:00	447.0	329.0	0.0	4844.0	4821.0	162.0	
1	2015-01-01 01:00:00+01:00	449.0	328.0	0.0	5196.0	4755.0	158.0	
2	2015-01-01 02:00:00+01:00	448.0	323.0	0.0	4857.0	4581.0	157.0	
3	2015-01-01 03:00:00+01:00	438.0	254.0	0.0	4314.0	4131.0	160.0	
4	2015-01-01 04:00:00+01:00	428.0	187.0	0.0	4130.0	3840.0	156.0	

5 rows × 29 columns

In [4]:

```
energy_data.tail()
```

Out[4]:

	time	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil
35059	2018-12-31 19:00:00+01:00	297.0	0.0	0.0	7634.0	2628.0	178.0
35060	2018-12-31 20:00:00+01:00	296.0	0.0	0.0	7241.0	2566.0	174.0
35061	2018-12-31 21:00:00+01:00	292.0	0.0	0.0	7025.0	2422.0	168.0
35062	2018-12-31 22:00:00+01:00	293.0	0.0	0.0	6562.0	2293.0	163.0
35063	2018-12-31 23:00:00+01:00	290.0	0.0	0.0	6926.0	2166.0	163.0

5 rows × 29 columns

In [5]:

```
energy_data.shape
```

Out[5]:

```
(35064, 29)
```

In [6]:

```
energy_data.columns
```

Out[6]:

```
Index(['time', 'generation biomass', 'generation fossil brown coal/lignit
e',
       'generation fossil coal-derived gas', 'generation fossil gas',
       'generation fossil hard coal', 'generation fossil oil',
       'generation fossil oil shale', 'generation fossil peat',
       'generation geothermal', 'generation hydro pumped storage aggregate
d',
       'generation hydro pumped storage consumption',
       'generation hydro run-of-river and poundage',
       'generation hydro water reservoir', 'generation marine',
       'generation nuclear', 'generation other', 'generation other renewab
le',
       'generation solar', 'generation waste', 'generation wind offshore',
       'generation wind onshore', 'forecast solar day ahead',
       'forecast wind offshore eday ahead', 'forecast wind onshore day ahe
ad',
       'total load forecast', 'total load actual', 'price day ahead',
       'price actual'],
      dtype='object')
```

In [7]:

```
energy_data.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

```
energy_data.isnull().sum()
```

Out[8]:

```
time                                               0
generation biomass                                19
generation fossil brown coal/lignite              18
generation fossil coal-derived gas                18
generation fossil gas                             18
generation fossil hard coal                       18
generation fossil oil                             19
generation fossil oil shale                       18
generation fossil peat                            18
generation geothermal                             18
generation hydro pumped storage aggregated     35064
generation hydro pumped storage consumption       19
generation hydro run-of-river and poundage        19
generation hydro water reservoir                  18
generation marine                                 19
generation nuclear                                17
generation other                                  18
generation other renewable                        18
generation solar                                  18
generation waste                                  19
generation wind offshore                          18
generation wind onshore                           18
forecast solar day ahead                           0
forecast wind offshore eday ahead              35064
forecast wind onshore day ahead                    0
total load forecast                                0
total load actual                                 36
price day ahead                                    0
price actual                                       0
dtype: int64
```

In [9]:

```
energy_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35064 entries, 0 to 35063
Data columns (total 29 columns):
 #   Column                                       Non-Null Count  Dtype  
---  ------                                       --------------  -----  
 0   time                                         35064 non-null  object 
 1   generation biomass                           35045 non-null  float64
 2   generation fossil brown coal/lignite         35046 non-null  float64
 3   generation fossil coal-derived gas           35046 non-null  float64
 4   generation fossil gas                        35046 non-null  float64
 5   generation fossil hard coal                  35046 non-null  float64
 6   generation fossil oil                        35045 non-null  float64
 7   generation fossil oil shale                  35046 non-null  float64
 8   generation fossil peat                       35046 non-null  float64
 9   generation geothermal                        35046 non-null  float64
 10  generation hydro pumped storage aggregated   0 non-null      float64
 11  generation hydro pumped storage consumption  35045 non-null  float64
 12  generation hydro run-of-river and poundage   35045 non-null  float64
 13  generation hydro water reservoir             35046 non-null  float64
 14  generation marine                            35045 non-null  float64
 15  generation nuclear                           35047 non-null  float64
 16  generation other                             35046 non-null  float64
 17  generation other renewable                   35046 non-null  float64
 18  generation solar                             35046 non-null  float64
 19  generation waste                             35045 non-null  float64
 20  generation wind offshore                     35046 non-null  float64
 21  generation wind onshore                      35046 non-null  float64
 22  forecast solar day ahead                     35064 non-null  int64  
 23  forecast wind offshore eday ahead            0 non-null      float64
 24  forecast wind onshore day ahead              35064 non-null  int64  
 25  total load forecast                          35064 non-null  int64  
 26  total load actual                            35028 non-null  float64
 27  price day ahead                              35064 non-null  float64
 28  price actual                                 35064 non-null  float64
dtypes: float64(25), int64(3), object(1)
memory usage: 7.8+ MB
```

In [10]:

```
energy_data.describe()
```

Out[10]:

	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil	g
count	35045.000000	35046.000000	35046.0	35046.000000	35046.000000	35045.000000	
mean	383.513540	448.059208	0.0	5622.737488	4256.065742	298.319789	
std	85.353943	354.568590	0.0	2201.830478	1961.601013	52.520673	
min	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	
25%	333.000000	0.000000	0.0	4126.000000	2527.000000	263.000000	
50%	367.000000	509.000000	0.0	4969.000000	4474.000000	300.000000	
75%	433.000000	757.000000	0.0	6429.000000	5838.750000	330.000000	
max	592.000000	999.000000	0.0	20034.000000	8359.000000	449.000000	

8 rows × 28 columns

In [11]:

```
energy_data = energy_data.drop(['generation hydro pumped storage aggregated',
                                'forecast wind offshore eday ahead'], axis = 1)
```

In [12]:

```
energy_data = energy_data.dropna()
```

In [13]:

```
energy_data.isnull().sum()
```

Out[13]:

```
time                                           0
generation biomass                             0
generation fossil brown coal/lignite           0
generation fossil coal-derived gas             0
generation fossil gas                          0
generation fossil hard coal                    0
generation fossil oil                          0
generation fossil oil shale                    0
generation fossil peat                         0
generation geothermal                          0
generation hydro pumped storage consumption    0
generation hydro run-of-river and poundage     0
generation hydro water reservoir               0
generation marine                              0
generation nuclear                             0
generation other                               0
generation other renewable                     0
generation solar                               0
generation waste                               0
generation wind offshore                       0
generation wind onshore                        0
forecast solar day ahead                       0
forecast wind onshore day ahead                0
total load forecast                            0
total load actual                              0
price day ahead                                0
price actual                                   0
dtype: int64
```

In [14]:

```
energy_data.nunique()
```

Out[14]:

```
time                                           35017
generation biomass                               423
generation fossil brown coal/lignite             956
generation fossil coal-derived gas                 1
generation fossil gas                           8293
generation fossil hard coal                     7265
generation fossil oil                            321
generation fossil oil shale                        1
generation fossil peat                             1
generation geothermal                              1
generation hydro pumped storage consumption     3311
generation hydro run-of-river and poundage      1684
generation hydro water reservoir                7029
generation marine                                  1
generation nuclear                              2388
generation other                                 103
generation other renewable                        78
generation solar                                5331
generation waste                                 262
generation wind offshore                           1
generation wind onshore                        11462
forecast solar day ahead                        5356
forecast wind onshore day ahead                11329
total load forecast                            14786
total load actual                              15123
price day ahead                                 5747
price actual                                    6641
dtype: int64
```

In [15]:

```
energy_data = energy_data.drop(['time'], axis = 1)
```

In [16]:

```
round((energy_data.isnull().sum()/len(energy_data)*100),2)
```

Out[16]:

```
generation biomass                             0.0
generation fossil brown coal/lignite           0.0
generation fossil coal-derived gas             0.0
generation fossil gas                          0.0
generation fossil hard coal                    0.0
generation fossil oil                          0.0
generation fossil oil shale                    0.0
generation fossil peat                         0.0
generation geothermal                          0.0
generation hydro pumped storage consumption    0.0
generation hydro run-of-river and poundage     0.0
generation hydro water reservoir               0.0
generation marine                              0.0
generation nuclear                             0.0
generation other                               0.0
generation other renewable                     0.0
generation solar                               0.0
generation waste                               0.0
generation wind offshore                       0.0
generation wind onshore                        0.0
forecast solar day ahead                       0.0
forecast wind onshore day ahead                0.0
total load forecast                            0.0
total load actual                              0.0
price day ahead                                0.0
price actual                                   0.0
dtype: float64
```

In [17]:

```
energy_data.corr()
```

Out[17]:

	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil	generat fossil sh
generation biomass	1.000000	0.229608	NaN	-0.021187	0.433113	0.458499	N
generation fossil brown coal/lignite	0.229608	1.000000	NaN	0.500119	0.768905	0.314732	N
generation fossil coal-derived gas	NaN	NaN	NaN	NaN	NaN	NaN	N
generation fossil gas	-0.021187	0.500119	NaN	1.000000	0.542141	0.310711	N
generation fossil hard coal	0.433113	0.768905	NaN	0.542141	1.000000	0.440374	N
generation fossil oil	0.458499	0.314732	NaN	0.310711	0.440374	1.000000	N
generation fossil oil shale	NaN	NaN	NaN	NaN	NaN	NaN	N
generation fossil peat	NaN	NaN	NaN	NaN	NaN	NaN	N
generation geothermal	NaN	NaN	NaN	NaN	NaN	NaN	N
generation hydro pumped storage consumption	-0.044836	-0.323907	NaN	-0.420602	-0.406085	-0.331405	N
generation hydro run-of-river and poundage	-0.285804	-0.525184	NaN	-0.271238	-0.498581	-0.107619	N
generation hydro water reservoir	-0.034102	-0.229371	NaN	0.060461	-0.158107	0.160220	N
generation marine	NaN	NaN	NaN	NaN	NaN	NaN	N
generation nuclear	-0.023269	-0.008795	NaN	-0.112049	-0.025069	0.013163	N
generation other	0.658608	0.097381	NaN	-0.065878	0.264419	0.374703	N
generation other renewable	-0.563450	0.104013	NaN	0.336101	-0.020198	-0.117448	N
generation solar	-0.005010	0.040535	NaN	0.074938	0.046091	0.099879	N
generation waste	-0.348220	0.282625	NaN	0.276167	0.170160	-0.177810	N

	generation biomass	generation fossil brown coal/lignite	generation fossil coal-derived gas	generation fossil gas	generation fossil hard coal	generation fossil oil	generat fossi sh
generation wind offshore	NaN	NaN	NaN	NaN	NaN	NaN	N
generation wind onshore	-0.069010	-0.434509	NaN	-0.397280	-0.442063	-0.052254	N
forecast solar day ahead	-0.008692	0.042471	NaN	0.080235	0.047454	0.096547	N
forecast wind onshore day ahead	-0.072183	-0.436250	NaN	-0.397565	-0.444425	-0.058051	N
total load forecast	0.085351	0.278777	NaN	0.543711	0.394443	0.499435	N
total load actual	0.083211	0.280531	NaN	0.548947	0.396637	0.497069	N
price day ahead	0.108867	0.568146	NaN	0.640889	0.671667	0.293068	N
price actual	0.142799	0.364206	NaN	0.461918	0.466703	0.285351	N

26 rows × 26 columns

In [18]:

```
correlations = energy_data.corr(method='pearson')
```

In [19]:

```
print(correlations['price actual'].sort_values(ascending=False).to_string())
```

```
price actual                                   1.000000
price day ahead                                0.733508
generation fossil hard coal                    0.466703
generation fossil gas                          0.461918
total load forecast                            0.436235
total load actual                              0.435873
generation fossil brown coal/lignite           0.364206
generation fossil oil                          0.285351
generation other renewable                     0.256398
generation waste                               0.169290
generation biomass                             0.142799
forecast solar day ahead                       0.101463
generation other                               0.099759
generation solar                               0.098774
generation hydro water reservoir               0.072210
generation nuclear                            -0.051817
generation hydro run-of-river and poundage    -0.136752
generation wind onshore                       -0.221761
forecast wind onshore day ahead               -0.223099
generation hydro pumped storage consumption   -0.427032
generation fossil coal-derived gas                  NaN
generation fossil oil shale                         NaN
generation fossil peat                              NaN
generation geothermal                               NaN
generation marine                                   NaN
generation wind offshore                            NaN
```

In [20]:

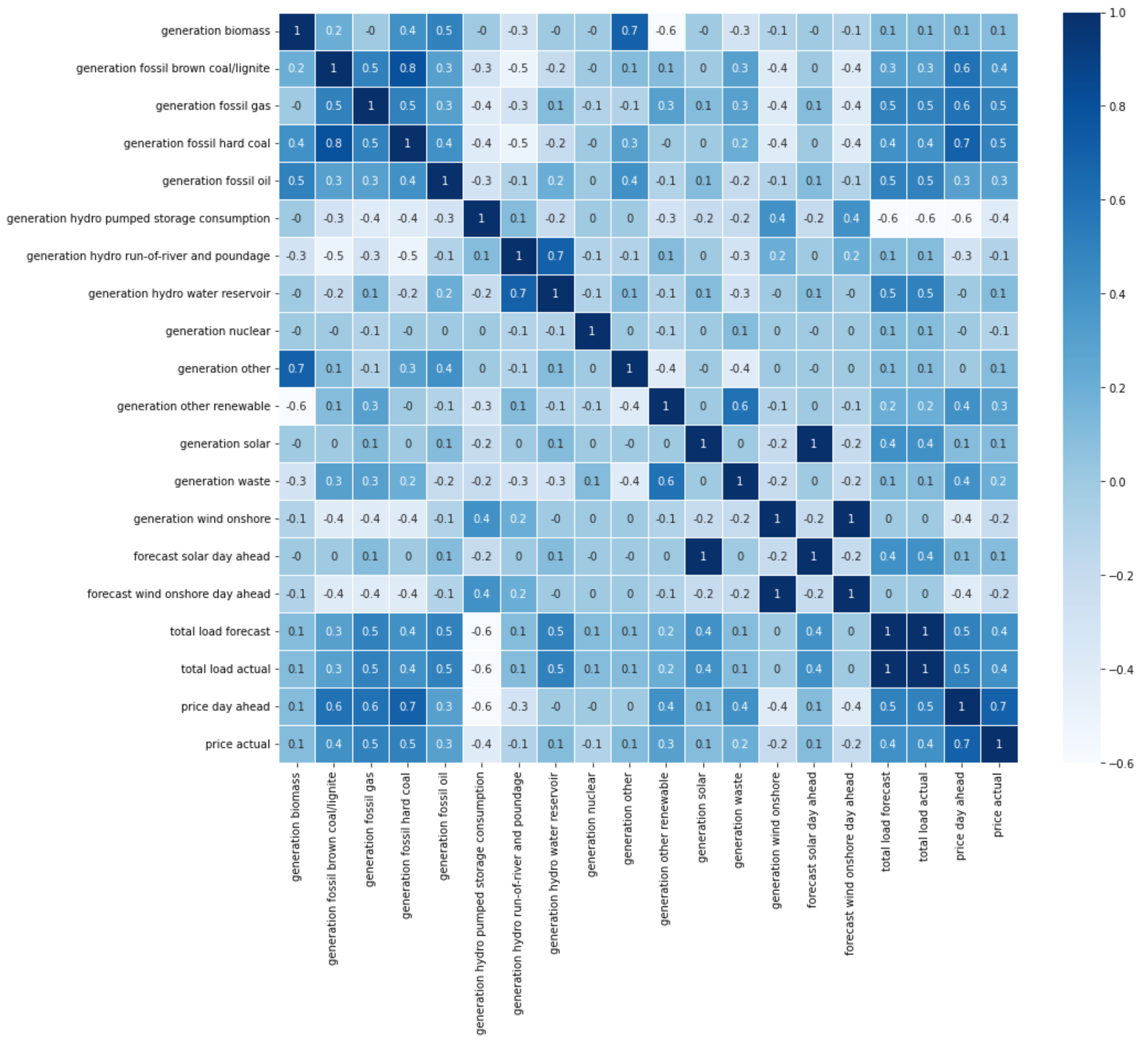
```
null_val_cols = ['generation marine',
                 'generation geothermal',
                 'generation fossil peat',
                 'generation wind offshore',
                 'generation fossil oil shale',
                 'generation fossil coal-derived gas']
```

In [21]:

```
heat_map_features = energy_data.drop(columns=null_val_cols,axis=1)
```

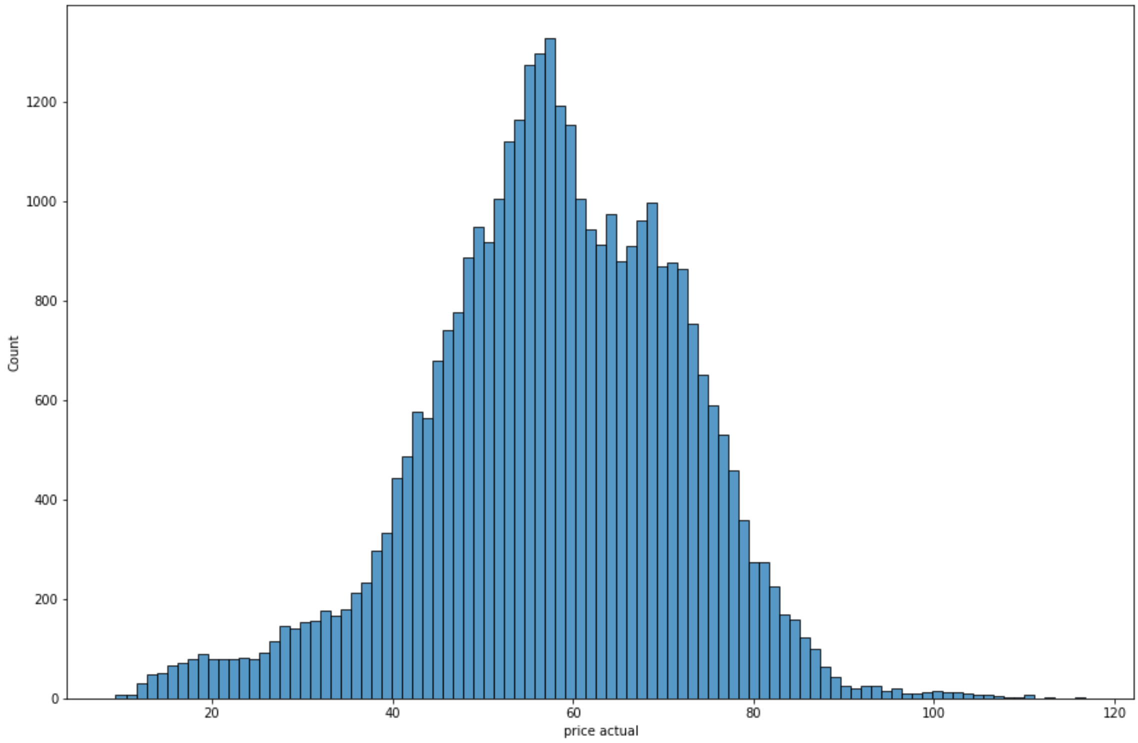
In [22]:

```
plt.figure(figsize=(15,12.5))
sns.heatmap(round(heat_map_features.corr(),1),annot=True,
            cmap='Blues',linewidth=0.9)
plt.show();
```



In [23]:

```
plt.figure(figsize=(15,10))
sns.histplot(energy_data,x='price actual');
plt.show();
```

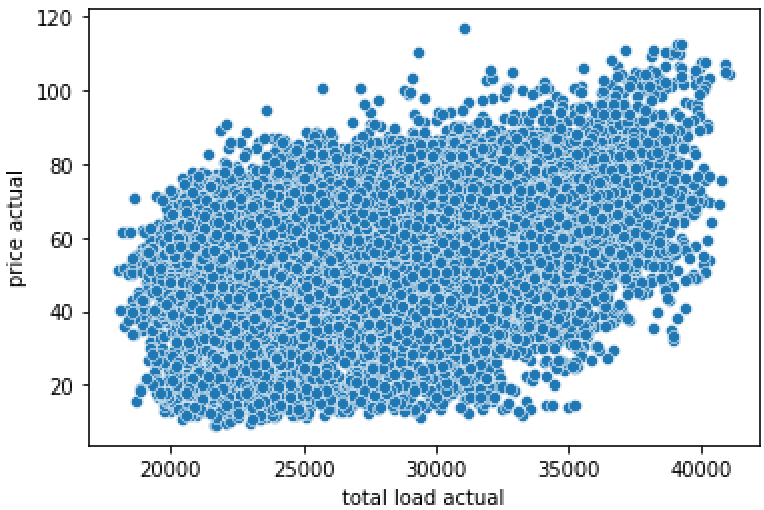


In [24]:

```
sns.scatterplot(x='total load actual',y='price actual',
                data = energy_data)
```

Out[24]:

```
<AxesSubplot: xlabel='total load actual', ylabel='price actual'>
```



In [25]:

```
x = energy_data.drop(['price actual'], axis = 1)
y = energy_data['price actual']
```

In [26]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
```

In [27]:

```
x = scaler.fit_transform(x)
```

In [28]:

```
from sklearn.model_selection import train_test_split
```

In [29]:

```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2, random_state=42)
```

In [30]:

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

In [31]:

```
from sklearn.linear_model import Ridge, LinearRegression
```

In [32]:

```
model = LinearRegression()
model.fit(xtrain, ytrain)
```

Out[32]:

```
▾ LinearRegression
LinearRegression()
```

In [33]:

```
y_pred = model.predict(xtest)
```

In [34]:

```
print("Training Accuracy :", model.score(xtrain, ytrain))
print("Testing Accuracy :", model.score(xtest, ytest))
```

```
Training Accuracy : 0.573367749107316
Testing Accuracy : 0.5722372770459524
```

In [35]:

```
mse = mean_squared_error(ytest, y_pred)
r2 = r2_score(ytest, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared score: {r2}")
```

```
Mean Squared Error: 83.09513080741849
R-squared score: 0.5722372770459524
```

In [36]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [37]:

```
regressor = RandomForestRegressor()
regressor.fit(xtrain, ytrain)
```

Out[37]:

```
▾ RandomForestRegressor
RandomForestRegressor()
```

In [38]:

```
y_pred = regressor.predict(xtest)
```

In [39]:

```
print("Training Accuracy :", regressor.score(xtrain, ytrain))
print("Testing Accuracy :", regressor.score(xtest, ytest))
```

```
Training Accuracy : 0.9780138020126133
Testing Accuracy : 0.8447256918471826
```

In [40]:

```
mse = mean_squared_error(ytest, y_pred)
r2 = r2_score(ytest, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared score: {r2}")
```

```
Mean Squared Error: 30.162840880306963
R-squared score: 0.8447256918471826
```

In [41]:

```
import xgboost as xgb
```

In [42]:

```
xgb_reg = xgb.XGBRegressor()
```

In [43]:

```
xgb_reg.fit(xtrain, ytrain)
```

Out[43]:

```
▾ XGBRegressor
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree
=1,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthw
ise',
             importance_type=None, interaction_constraints='',
             learning_rate=0.300000012, max_bin=256, max_cat_to_onehot
=4,
             max_delta_step=0, max_depth=6, max_leaves=0, min_child_we
```

In [44]:

```
y_pred = xgb_reg.predict(xtest)
```

In [45]:

```
print("Training Accuracy :", xgb_reg.score(xtrain, ytrain))
print("Testing Accuracy :", xgb_reg.score(xtest, ytest))
```

```
Training Accuracy : 0.9044437032799999
Testing Accuracy : 0.8201084417705423
```

In [46]:

```
mse = mean_squared_error(ytest, y_pred)
r2 = r2_score(ytest, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared score: {r2}")
```

```
Mean Squared Error: 34.94486957395053
R-squared score: 0.8201084417705423
```