```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder, StandardScaler,
MinMaxScaler
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf


from sklearn.preprocessing import LabelEncoder

from plotly.subplots import make_subplots
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import acf, pacf
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler

df_weather_features = pd.read_csv( '/content/weatherdata_file.csv')


df_energy_features = pd.read_csv( '/content/energy_dataset.csv',)

df_weather_features.head()
```

{"type":"dataframe","variable_name":"df_weather_features"}

```
df_weather_features.shape

(178396, 19)

df_weather_features.columns

Index(['Unnamed: 0', 'dt_iso', 'city_name', 'temp', 'temp_min',
'temp_max',
       'pressure', 'humidity', 'wind_speed', 'wind_deg', 'rain_1h',
'rain_3h',
       'snow_3h', 'clouds_all', 'weather_id', 'weather_main',
       'weather_description', 'weather_icon', 'holiday'],
      dtype='object')

df_weather_features = df_weather_features.drop(['Unnamed: 0'], axis=1,
errors='ignore')
print(df_weather_features)

                           dt_iso city_name     temp  temp_min
temp_max  \
0       2014-12-31 23:00:00+00:00  Valencia  270.475   270.475
270.475
```

```
1       2015-01-01 00:00:00+00:00  Valencia  270.475   270.475
270.475
2       2015-01-01 01:00:00+00:00  Valencia  269.686   269.686
269.686
3       2015-01-01 02:00:00+00:00  Valencia  269.686   269.686
269.686
4       2015-01-01 03:00:00+00:00  Valencia  269.686   269.686
269.686
...                           ...       ...      ...       ...        .
..
178391  2018-12-31 18:00:00+00:00   Seville  287.760   287.150
288.150
178392  2018-12-31 19:00:00+00:00   Seville  285.760   285.150
286.150
178393  2018-12-31 20:00:00+00:00   Seville  285.150   285.150
285.150
178394  2018-12-31 21:00:00+00:00   Seville  284.150   284.150
284.150
178395  2018-12-31 22:00:00+00:00   Seville  283.970   282.150
285.150

        pressure  humidity  wind_speed  wind_deg  rain_1h  rain_3h
snow_3h  \
0           1001        77           1        62      0.0      0.0
0.0
1           1001        77           1        62      0.0      0.0
0.0
2           1002        78           0        23      0.0      0.0
0.0
3           1002        78           0        23      0.0      0.0
0.0
4           1002        78           0        23      0.0      0.0
0.0
...          ...       ...         ...       ...      ...      ...
...
178391      1028        54           3        30      0.0      0.0
0.0
178392      1029        62           3        30      0.0      0.0
0.0
178393      1028        58           4        50      0.0      0.0
0.0
178394      1029        57           4        60      0.0      0.0
0.0
178395      1029        70           3        50      0.0      0.0
0.0

        clouds_all  weather_id weather_main weather_description
weather_icon  \
0                0         800        clear        sky is clear
```

```
01n
1                 0          800        clear         sky is clear
01n
2                 0          800        clear         sky is clear
01n
3                 0          800        clear         sky is clear
01n
4                 0          800        clear         sky is clear
01n
...             ...          ...          ...                  ...
...
178391            0          800        clear         sky is clear
01n
178392            0          800        clear         sky is clear
01n
178393            0          800        clear         sky is clear
01n
178394            0          800        clear         sky is clear
01n
178395            0          800        clear         sky is clear
01n

        holiday
0             1
1             1
2             1
3             1
4             1
...         ...
178391        1
178392        1
178393        1
178394        1
178395        1

[178396 rows x 18 columns]
```

```
df_weather_features.duplicated().sum()
```

```
21
```

```
df_weather_features=df_weather_features.drop_duplicates()
```

```
df_weather_features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 178375 entries, 0 to 178395
Data columns (total 18 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   dt_iso               178375 non-null  object
```

```
 1   city_name             178375 non-null  object
 2   temp                  178375 non-null  float64
 3   temp_min              178375 non-null  float64
 4   temp_max              178375 non-null  float64
 5   pressure              178375 non-null  int64
 6   humidity              178375 non-null  int64
 7   wind_speed            178375 non-null  int64
 8   wind_deg              178375 non-null  int64
 9   rain_1h               178375 non-null  float64
 10  rain_3h               178375 non-null  float64
 11  snow_3h               178375 non-null  float64
 12  clouds_all            178375 non-null  int64
 13  weather_id            178375 non-null  int64
 14  weather_main          178375 non-null  object
 15  weather_description   178375 non-null  object
 16  weather_icon          178375 non-null  object
 17  holiday               178375 non-null  int64
dtypes: float64(6), int64(7), object(5)
memory usage: 25.9+ MB
```

```
df_weather_features.nunique()
```

```
dt_iso                 35064
city_name                  5
temp                   20743
temp_min               18553
temp_max               18591
pressure                 190
humidity                 100
wind_speed                36
wind_deg                 361
rain_1h                    7
rain_3h                   89
snow_3h                   66
clouds_all                97
weather_id                38
weather_main              12
weather_description       43
weather_icon              24
holiday                    2
dtype: int64
```

```
def df_convert_dtypes(df, convert_from, convert_to):
  values = df.select_dtypes(include=[convert_from]).columns
  for val in values:
         df[val] = df[val].astype(convert_to)

  return df
```

```
df_weather_features = df_convert_dtypes(df_weather_features, np.int64, np.float64)

df_weather_features = (df_weather_features.assign(time=pd.to_datetime(df_weather_features['dt_iso'], utc=True, infer_datetime_format=True)).drop(columns=['dt_iso']).set_index('time'))
```

```
<ipython-input-253-8c78dc69c458>:1: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.
```

```
missingvalues = df_weather_features.isnull().sum().sum()
print(f'There are {missingvalues} missing values or NaNs in df_weather_features.')
```

```
There are 0 missing values or NaNs in df_weather_features.
```

```
duplicaterows = df_weather_features.duplicated().sum()


print(f'There are {duplicaterows} duplicate rows in df_weather_features.')
```

```
There are 8442 duplicate rows in df_weather_features.
```

```
weather_description_unique = df_weather_features['weather_description'].unique()
weather_description_unique
```

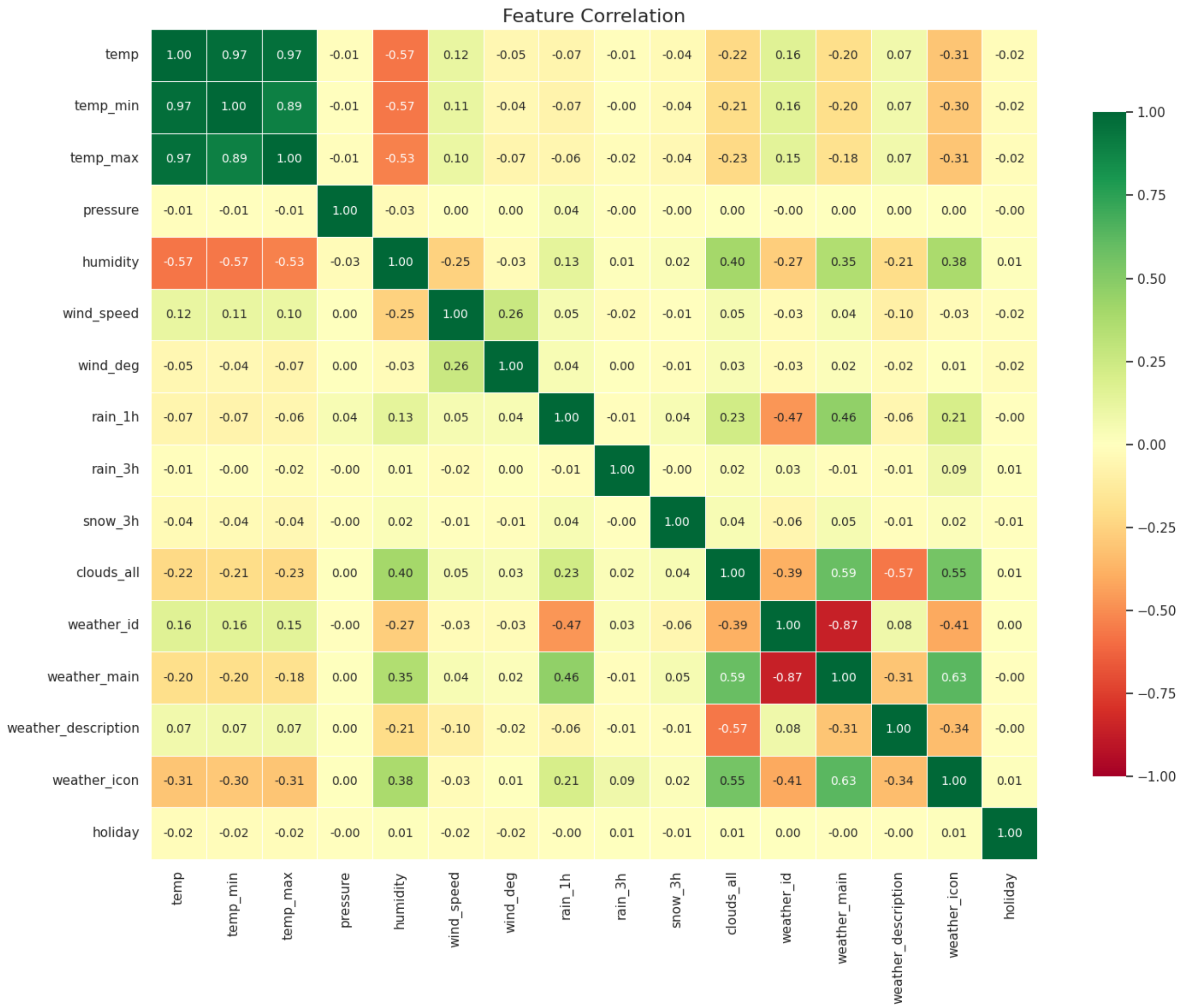
```
array(['sky is clear', 'few clouds', 'scattered clouds', 'broken clouds',
       'overcast clouds', 'light rain', 'moderate rain',
       'heavy intensity rain', 'mist', 'heavy intensity shower rain',
       'shower rain', 'very heavy rain', 'thunderstorm with heavy rain',
       'thunderstorm with light rain', 'thunderstorm with rain',
       'proximity thunderstorm', 'thunderstorm',
       'light intensity shower rain', 'light intensity drizzle', 'fog',
       'drizzle', 'smoke', 'heavy intensity drizzle', 'haze',
       'proximity shower rain', 'light intensity drizzle rain',
       'light snow', 'rain and snow', 'light rain and snow', 'snow',
       'light thunderstorm', 'heavy snow', 'sleet', 'rain and drizzle',
       'shower sleet', 'light shower sleet', 'light shower snow',
```

```
        'proximity moderate rain', 'ragged shower rain',
        'sand dust whirls', 'proximity drizzle', 'dust', 'squalls'],
      dtype=object)
```

```
weather_uniqueid = df_weather_features['weather_id'].unique()
weather_uniqueid
```

```
array([800., 801., 802., 803., 804., 500., 501., 502., 701., 522.,
521.,
       503., 202., 200., 201., 211., 520., 300., 741., 301., 711.,
302.,
       721., 310., 600., 616., 615., 601., 210., 602., 611., 311.,
612.,
       620., 531., 731., 761., 771.])
```

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

def feat_corr(input_df):

    numeric_df = input_df.select_dtypes(include=np.number)


    corr = numeric_df.corr()


    plt.figure(figsize=(15, 12))
    sns.heatmap(corr, annot=True, fmt='.2f', cmap='RdYlGn', vmin=-1,
vmax=1,
                linewidths=0.5, cbar_kws={"shrink": 0.8},
annot_kws={"size": 10})

    plt.title('Feature Correlation', fontsize=16)
    plt.tight_layout()

    plt.show()

df_temp = df_weather_features.copy(deep = True)
labels = ['weather_id',
'weather_main','weather_description','weather_icon']
for col in labels:
    df_temp[col] =
LabelEncoder().fit_transform(df_weather_features[col])

feat_corr(df_temp)
```




```
col_drop_name = ['weather_id',
'weather_main','weather_description','weather_icon', 'temp_min',
'temp_max']
# col_drop_name = ['weather_id',
'weather_main','weather_description','weather_icon']
df_weather_features.drop(col_drop_name, axis = 1 , inplace = True)

missingvalues = df_weather_features.isnull().sum().sum()
print(f'There are {missingvalues} missing values or NaNs in
df_weather_features.')
```

```
There are 0 missing values or NaNs in df_weather_features.
```

```
duplicaterows = df_weather_features.duplicated().sum()


print(f'There are {duplicaterows} duplicate rows in
df_weather_features.')
```

```
There are 12907 duplicate rows in df_weather_features.

df_weather_features =
df_weather_features.reset_index().drop_duplicates()

df_weather_features
```

{"type":"dataframe","variable_name":"df_weather_features"}

```
df_energy_features.shape

(35064, 29)

df_energy_features.head()
```

{"type":"dataframe","variable_name":"df_energy_features"}

```
df_energy_features.columns

Index(['time', 'generation biomass', 'generation fossil brown
coal/lignite',
       '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
aggregated',
       'generation hydro pumped storage consumption',
       'generation hydro run-of-river and poundage',
       'generation hydro water reservoir', 'generation marine',
       'generation nuclear', 'generation other', 'generation other
renewable',
       'generation solar', 'generation waste', 'generation wind
offshore',
       'generation wind onshore', 'forecast solar day ahead',
       'forecast wind offshore eday ahead', 'forecast wind onshore day
ahead',
       'total load forecast', 'total load actual', 'price day ahead',
       'price actual'],
      dtype='object')

df_energy_features.duplicated().sum()

0

df_energy_features.isnull().sum()

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
```

```
df_energy_features.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
float64
 23  forecast wind offshore eday ahead            0 non-null
float64
 24  forecast wind onshore day ahead              35064 non-null
float64
 25  total load forecast                          35064 non-null
float64
 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(28), object(1)
memory usage: 7.8+ MB


df_energy_features['time'] =
pd.to_datetime(df_energy_features['time'], utc=True,
infer_datetime_format=True)
df_energy_features= df_energy_features.set_index('time')
```

```
<ipython-input-272-5d91c7a4741e>:1: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed
in a future version. A strict version of it is now the default, see
https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-
parsing.html. You can safely remove this argument.
```

```
df_energy_features.describe()
```

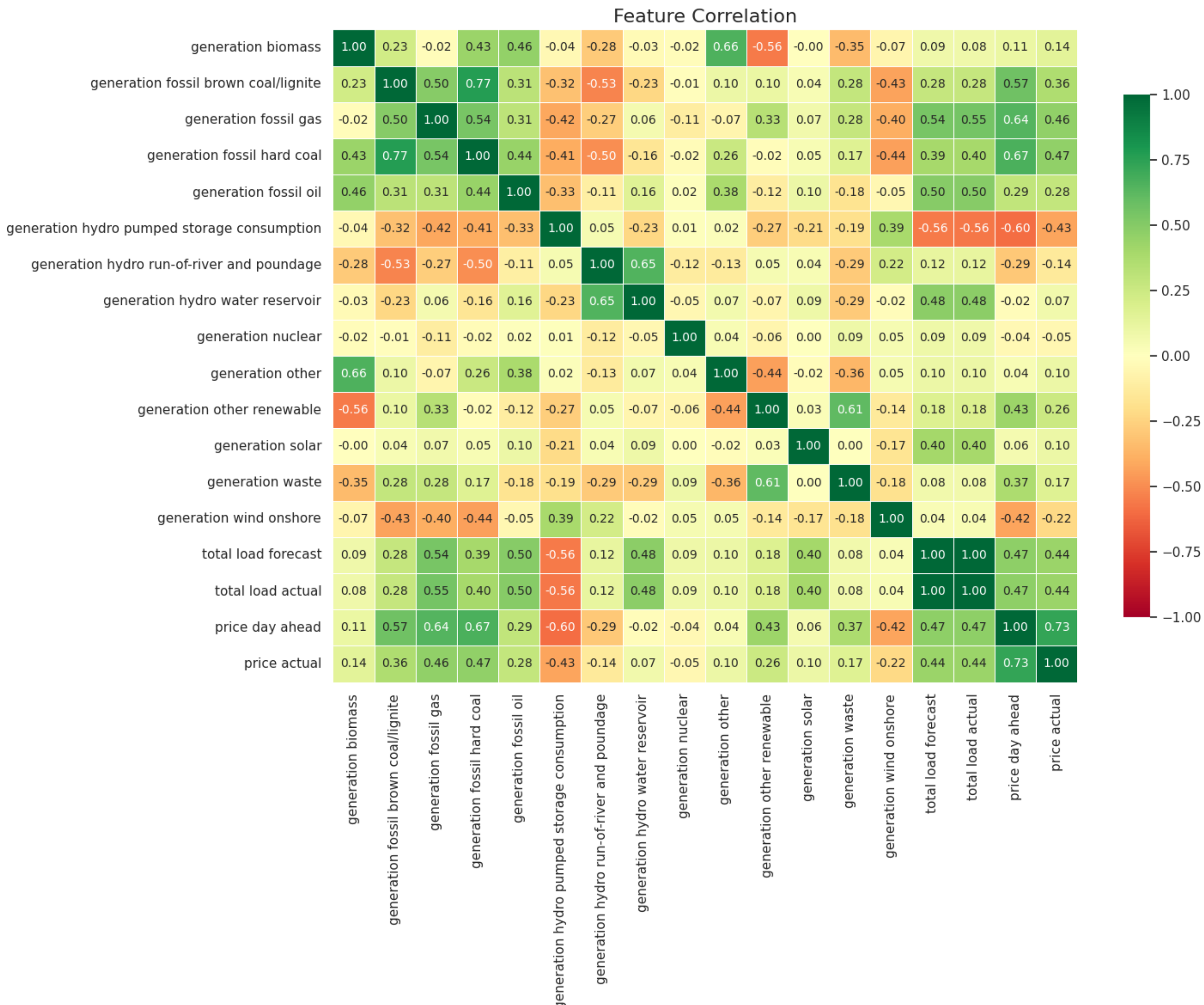
{"type":"dataframe"}

```
df_energy_features.nunique()
```

```
generation biomass                                423
generation fossil brown coal/lignite              956
generation fossil coal-derived gas                  1
generation fossil gas                            8297
generation fossil hard coal                      7266
generation fossil oil                             321
generation fossil oil shale                         1
generation fossil peat                              1
generation geothermal                               1
generation hydro pumped storage aggregated          0
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                         11465
forecast solar day ahead                         5356
forecast wind offshore eday ahead                   0
forecast wind onshore day ahead                 11332
total load forecast                             14790
total load actual                               15127
price day ahead                                  5747
price actual                                     6653
dtype: int64
```

```
# columns to be removed due to all 0 or Nan values
columns = ['generation fossil coal-derived gas', 'generation fossil
oil shale','generation fossil peat',
           'generation geothermal', 'generation hydro pumped storage
aggregated', 'generation marine',
           'generation wind offshore', 'forecast wind offshore eday
```

```
ahead', 'forecast solar day ahead',
           'forecast wind onshore day ahead']

df_energy_features = df_energy_features.drop(columns, axis = 1)

feat_corr(df_energy_features)
```




```
df_energy_features.duplicated().sum()

0

df_energy_features.isnull().sum(axis=0)

generation biomass                                 19
generation fossil brown coal/lignite               18
generation fossil gas                              18
generation fossil hard coal                        18
generation fossil oil                              19
```

```
generation hydro pumped storage consumption    19
generation hydro run-of-river and poundage     19
generation hydro water reservoir               18
generation nuclear                             17
generation other                               18
generation other renewable                     18
generation solar                               18
generation waste                               19
generation wind onshore                        18
total load forecast                             0
total load actual                              36
price day ahead                                 0
price actual                                    0
dtype: int64
```

```
df_energy_features
```

{"summary":"{\n \"name\": \"df_energy_features\",\n \"rows\": 35064,\n \"fields\": [\n   {\n   \"column\": \"time\",\n   \"properties\": {\n   \"dtype\": \"date\",\n   \"min\": \"2014-12-31 23:00:00+00:00\",\n   \"max\": \"2018-12-31 22:00:00+00:00\",\n   \"num_unique_values\": 35064,\n   \"samples\": [\n   \"2015-09-10 21:00:00+00:00\",\n   \"2018-09-20 07:00:00+00:00\",\n   \"2016-01-04 13:00:00+00:00\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation biomass\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 85.3539430538196,\n   \"min\": 0.0,\n   \"max\": 592.0,\n   \"num_unique_values\": 423,\n   \"samples\": [\n   577.0,\n   227.0,\n   405.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation fossil brown coal/lignite\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 354.5685904349092,\n   \"min\": 0.0,\n   \"max\": 999.0,\n   \"num_unique_values\": 956,\n   \"samples\": [\n   883.0,\n   431.0,\n   601.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation fossil gas\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 2201.830477836224,\n   \"min\": 0.0,\n   \"max\": 20034.0,\n   \"num_unique_values\": 8297,\n   \"samples\": [\n   4880.0,\n   7674.0,\n   11327.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation fossil hard coal\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 1961.6010133302004,\n   \"min\": 0.0,\n   \"max\": 8359.0,\n   \"num_unique_values\": 7266,\n   \"samples\": [\n   5620.0,\n   4557.0,\n   1804.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation fossil oil\",\n

```
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
52.52067255592221,\n        \"min\": 0.0,\n        \"max\": 449.0,\n
\"num_unique_values\": 321,\n        \"samples\": [\n          290.0,\
n          210.0,\n          329.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"generation hydro pumped storage
consumption\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 792.4066137134466,\n        \"min\":
0.0,\n        \"max\": 4523.0,\n        \"num_unique_values\": 3311,\n
\"samples\": [\n          325.0,\n          2819.0,\n          689.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation hydro run-of-river
and poundage\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 400.7775358935912,\n        \"min\":
0.0,\n        \"max\": 2000.0,\n        \"num_unique_values\": 1684,\n
\"samples\": [\n          435.0,\n          458.0,\n          879.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation hydro water
reservoir\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 1835.1997450877438,\n        \"min\": 0.0,\n
\"max\": 9728.0,\n        \"num_unique_values\": 7029,\n
\"samples\": [\n          3051.0,\n          5160.0,\n
2557.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation nuclear\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 839.6679576780471,\n        \"min\":
0.0,\n        \"max\": 7117.0,\n        \"num_unique_values\": 2388,\n
\"samples\": [\n          6496.0,\n          6163.0,\n
6440.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation other\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 20.238380906782545,\n        \"min\":
0.0,\n        \"max\": 106.0,\n        \"num_unique_values\": 103,\n
\"samples\": [\n          95.0,\n          64.0,\n          99.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation other renewable\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
14.077554099310847,\n        \"min\": 0.0,\n        \"max\": 119.0,\n
\"num_unique_values\": 78,\n        \"samples\": [\n          83.0,\n
73.0,\n          84.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation solar\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1680.119887007093,\n        \"min\":
0.0,\n        \"max\": 5792.0,\n        \"num_unique_values\": 5331,\n
\"samples\": [\n          4158.0,\n          2093.0,\n
1187.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation waste\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 50.195535633578885,\n        \"min\":
```

0.0,\n   \"max\": 357.0,\n   \"num_unique_values\": 262,\n   \"samples\": [\n   344.0,\n   323.0,\n   308.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation wind onshore\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 3213.6915870178127,\n   \"min\": 0.0,\n   \"max\": 17436.0,\n   \"num_unique_values\": 11465,\n   \"samples\": [\n   5887.0,\n   8053.0,\n   6943.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"total load forecast\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 4594.100854174032,\n   \"min\": 18105.0,\n   \"max\": 41390.0,\n   \"num_unique_values\": 14790,\n   \"samples\": [\n   21970.0,\n   30045.0,\n   23844.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"total load actual\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 4574.987950049498,\n   \"min\": 18041.0,\n   \"max\": 41015.0,\n   \"num_unique_values\": 15127,\n   \"samples\": [\n   26099.0,\n   38645.0,\n   23443.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"price day ahead\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 14.618899685404326,\n   \"min\": 2.06,\n   \"max\": 101.99,\n   \"num_unique_values\": 5747,\n   \"samples\": [\n   33.63,\n   45.73,\n   32.64\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"price actual\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 14.204083293241405,\n   \"min\": 9.33,\n   \"max\": 116.8,\n   \"num_unique_values\": 6653,\n   \"samples\": [\n   83.51,\n   57.22,\n   54.1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df_energy_features"}

```
df_energy_features[df_energy_features.isna().any(axis=1)]
```

{"summary":"{\n \"name\": \"df_energy_features[df_energy_features\",\n \"rows\": 46,\n \"fields\": [\n   {\n   \"column\": \"time\",\n   \"properties\": {\n   \"dtype\": \"date\",\n   \"min\": \"2015-01-05 02:00:00+00:00\",\n   \"max\": \"2018-07-11 07:00:00+00:00\",\n   \"num_unique_values\": 46,\n   \"samples\": [\n   \"2016-09-28 07:00:00+00:00\",\n   \"2015-04-23 19:00:00+00:00\",\n   \"2015-05-02 08:00:00+00:00\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"generation biomass\",\n   \"properties\": {\n   \"dtype\":

```
\"number\",\n        \"std\": 136.92515683941565,\n        \"min\":
0.0,\n        \"max\": 569.0,\n        \"num_unique_values\": 26,\n
\"samples\": [\n          465.0,\n          220.0,\n          449.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation fossil brown
coal/lignite\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 297.6701305367346,\n        \"min\":
0.0,\n        \"max\": 961.0,\n        \"num_unique_values\": 16,\n
\"samples\": [\n          312.0,\n          302.0,\n          326.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation fossil gas\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1927.7806440672164,\n        \"min\": 2969.0,\n        \"max\":
12336.0,\n        \"num_unique_values\": 28,\n        \"samples\": [\n
7386.0,\n          10064.0,\n          6127.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"generation fossil hard coal\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
2141.2732042374196,\n        \"min\": 0.0,\n        \"max\": 8250.0,\n
\"num_unique_values\": 27,\n        \"samples\": [\n          5912.0,\
n          5677.0,\n          6002.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"generation fossil oil\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
98.60468138421822,\n        \"min\": 0.0,\n        \"max\": 385.0,\n
\"num_unique_values\": 25,\n        \"samples\": [\n          340.0,\n
151.0,\n          222.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"generation hydro pumped storage consumption\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
567.4673629180871,\n        \"min\": 0.0,\n        \"max\": 2270.0,\n
\"num_unique_values\": 11,\n        \"samples\": [\n          1340.0,\
n          480.0,\n          1413.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"generation hydro run-of-river and
poundage\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 439.1490151985514,\n        \"min\": 0.0,\n        \"max\":
1648.0,\n        \"num_unique_values\": 26,\n        \"samples\": [\n
1214.0,\n          1622.0,\n          980.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"generation hydro water
reservoir\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 2026.4761645532978,\n        \"min\": 0.0,\n
\"max\": 6895.0,\n        \"num_unique_values\": 27,\n
\"samples\": [\n          4684.0,\n          5289.0,\n
6187.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation nuclear\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1906.493960566064,\n        \"min\":
```

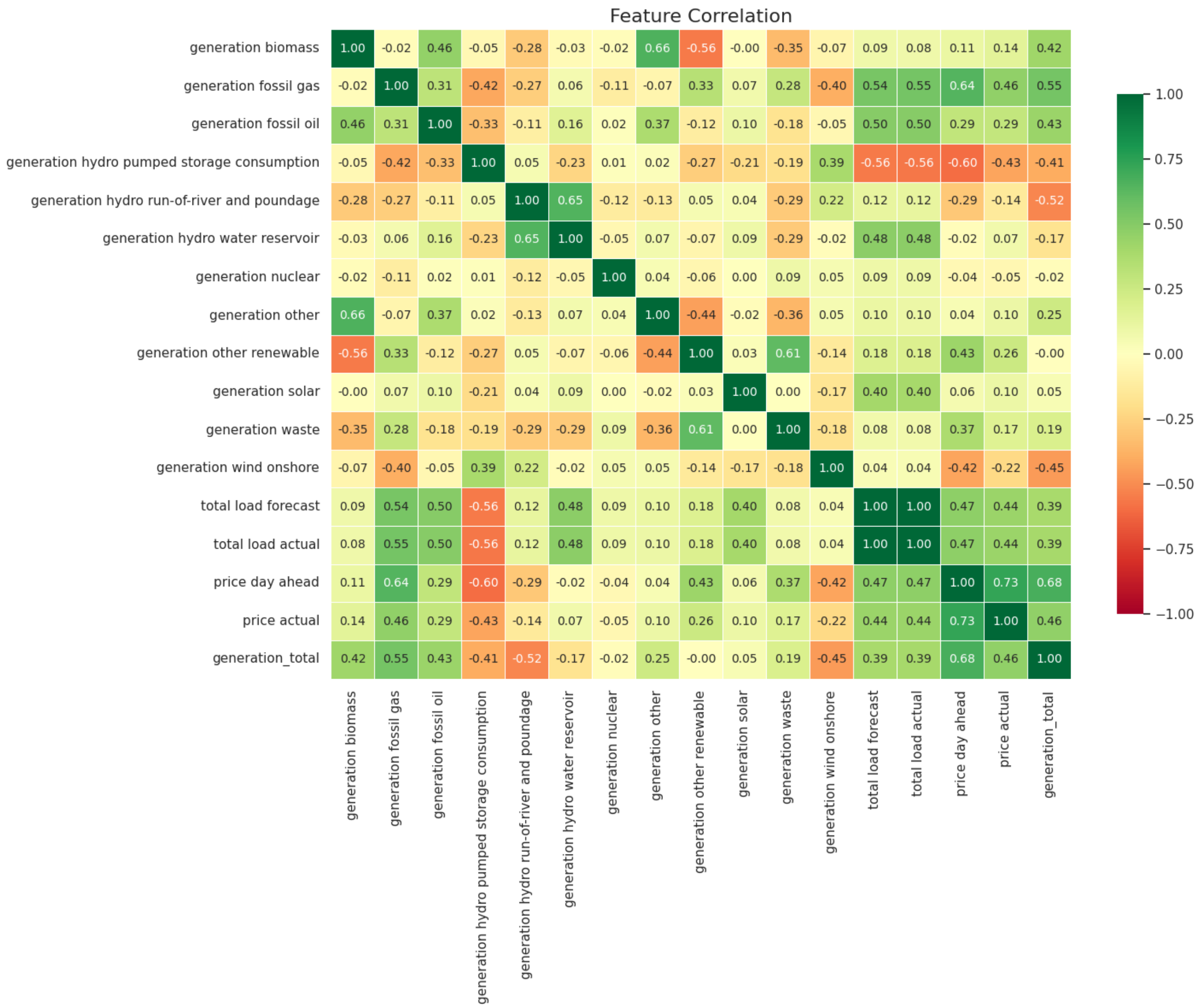
```
0.0,\n        \"max\": 7101.0,\n        \"num_unique_values\": 21,\n
\"samples\": [\n          7101.0,\n          6967.0,\n
5058.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation other\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 20.685347419939507,\n        \"min\":
0.0,\n        \"max\": 93.0,\n        \"num_unique_values\": 19,\n
\"samples\": [\n          44.0,\n          81.0,\n          51.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation other renewable\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
21.98565549668801,\n        \"min\": 0.0,\n        \"max\": 96.0,\n
\"num_unique_values\": 18,\n        \"samples\": [\n          75.0,\n
71.0,\n          62.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"generation solar\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1340.2586795250863,\n        \"min\":
0.0,\n        \"max\": 3836.0,\n        \"num_unique_values\": 26,\n
\"samples\": [\n          1281.0,\n          150.0,\n          48.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"generation waste\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
72.87885588651646,\n        \"min\": 0.0,\n        \"max\": 309.0,\n
\"num_unique_values\": 21,\n        \"samples\": [\n          208.0,\n
299.0,\n          309.0\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"generation wind onshore\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 2756.0547705276376,\n
\"min\": 0.0,\n        \"max\": 10903.0,\n
\"num_unique_values\": 27,\n        \"samples\": [\n          1857.0,\
n          10903.0,\n          1864.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"total load forecast\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
5778.164024515305,\n        \"min\": 20016.0,\n        \"max\":
39644.0,\n        \"num_unique_values\": 46,\n        \"samples\": [\n
31072.0,\n          31421.0,\n          39644.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"total load actual\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7218.587889762251,\n        \"min\": 21182.0,\n        \"max\":
39304.0,\n        \"num_unique_values\": 10,\n        \"samples\": [\n
26583.0,\n          39304.0,\n          23614.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"price day ahead\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
14.878207673469248,\n        \"min\": 15.0,\n        \"max\": 75.71,\n
\"num_unique_values\": 44,\n        \"samples\": [\n          49.72,\n
58.49,\n          45.93\n        ],\n        \"semantic_type\": \"\",\
```

n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"price actual\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 22.569020505330933,\n
\"min\": 16.98,\n   \"max\": 88.95,\n
\"num_unique_values\": 46,\n   \"samples\": [\n   56.4,\n
82.57,\n   59.09\n   ],\n   \"semantic_type\": \"\",\
n   \"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe"}

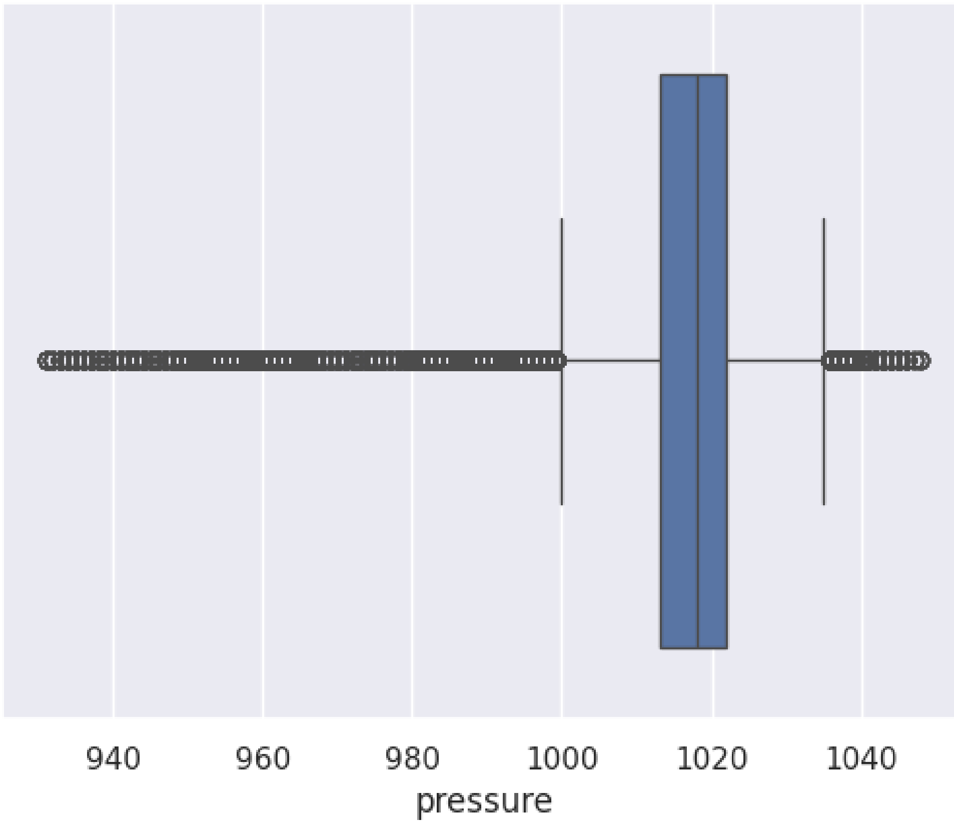
```
df_energy_features.interpolate(method='linear',
limit_direction='forward', inplace=True, axis=0)

df_energy_features.isnull().sum()
```

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

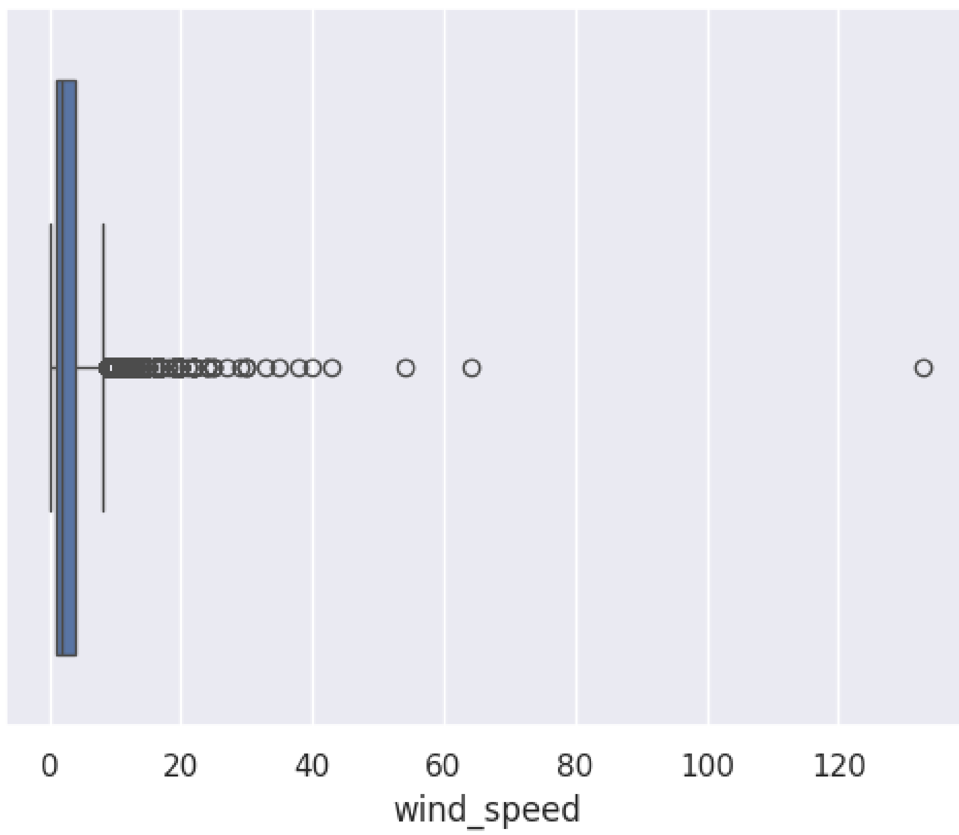
```
df_energy_features = (df_energy_features
              .assign(generation_total=df_energy_features['generation
fossil hard coal'] + df_energy_features['generation fossil brown
coal/lignite'])
              .drop(['generation fossil hard coal', 'generation fossil
brown coal/lignite'], axis=1))


feat_corr(df_energy_features)
```



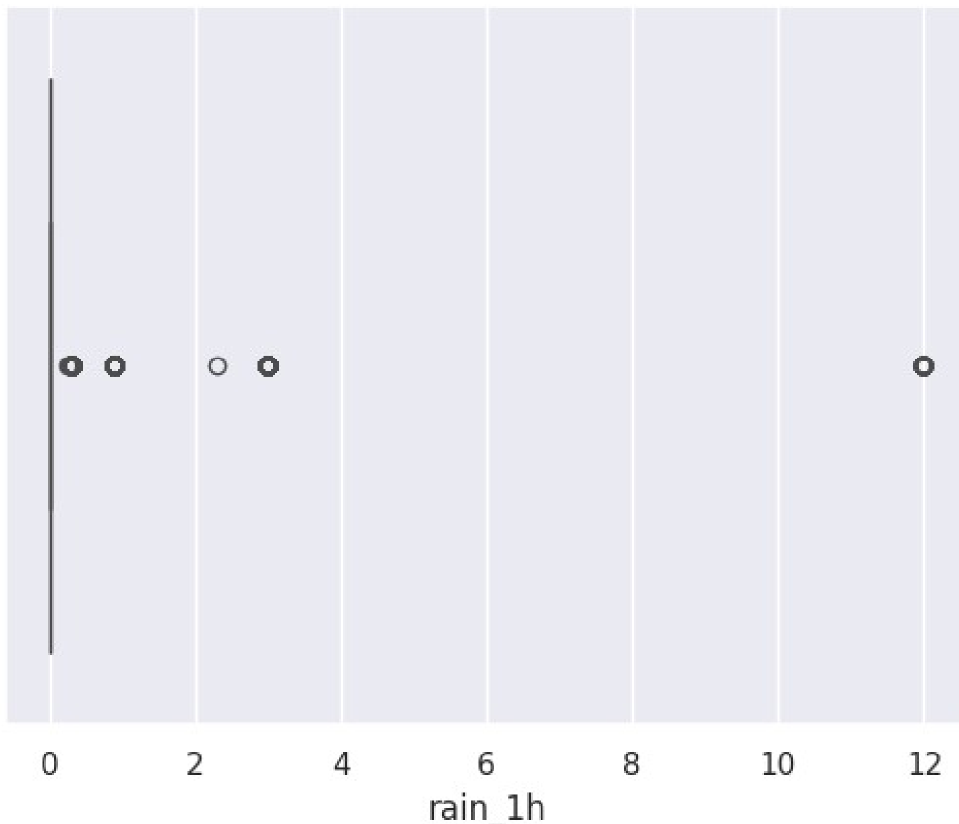

```
import statsmodels.api as sm

df_weather_features.loc[df_weather_features.pressure > 1050,
'pressure'] = np.nan
df_weather_features.loc[df_weather_features.pressure < 930,
'pressure'] = np.nan

df_weather_features['pressure'] =
df_weather_features['pressure'].where((df_weather_features['pressure']
<= 1051) & (df_weather_features['pressure'] >= 931), other=pd.NA)


if df_weather_features.index.duplicated().any():
    print("Duplicate index values found.")

sns.boxplot(x=df_weather_features['pressure'])
plt.show()
```



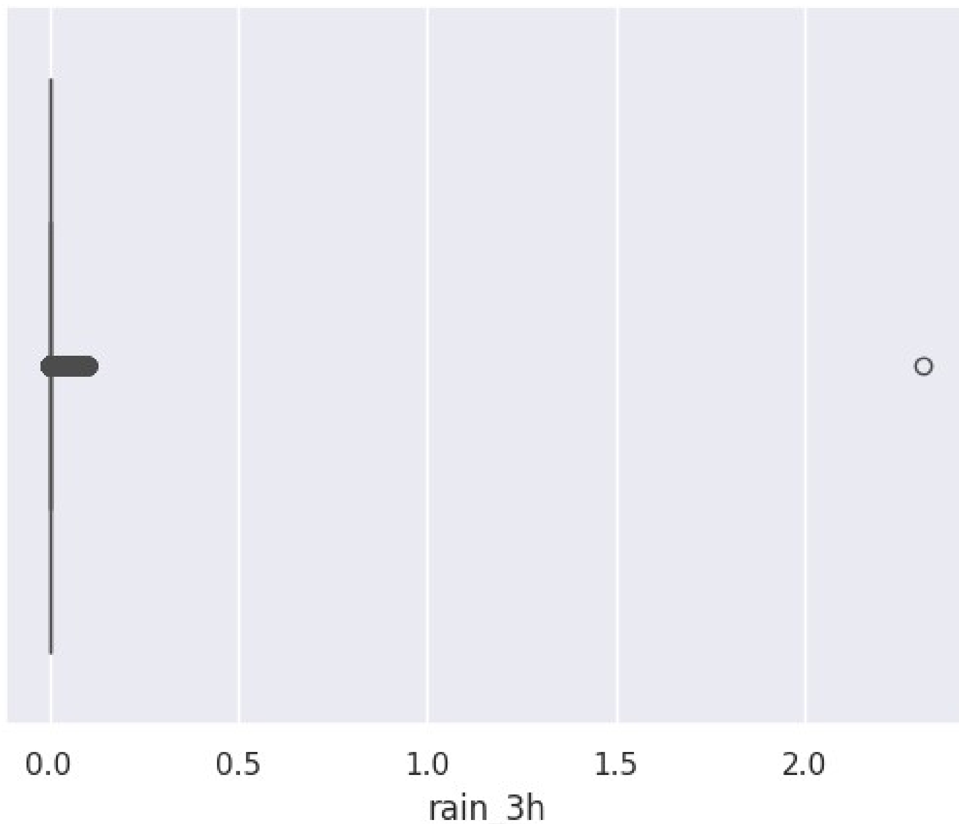

```
sns.boxplot(x=df_weather_features['wind_speed'])
plt.show()
```




```
sns.boxplot(x=df_weather_features['rain_1h'])
plt.show()
```




```
sns.boxplot(x=df_weather_features['rain_3h'])
plt.show()
```




```
df_weather_features.drop(['rain_3h'], axis = 1 , inplace = True)

print(f'Number of samples in df_energy is
{df_energy_features.shape[0]}')

city_list = df_weather_features['city_name'].unique()
grouped_weather = df_weather_features.groupby('city_name')

for city in city_list:
    print(f'Number of samples in df_weather in {city} is
{grouped_weather.get_group(city).shape[0]}')

Number of samples in df_energy is 35064
Number of samples in df_weather in Valencia is 35064
Number of samples in df_weather in Madrid is 35064
Number of samples in df_weather in Bilbao is 35064
Number of samples in df_weather in  Barcelona is 35064
Number of samples in df_weather in Seville is 35064

df_weather_cleaned =
df_weather_features.reset_index().drop_duplicates(subset=['time',
'city_name'], keep='first').set_index('time')

df_weather_cleaned
```

{"type":"dataframe","variable_name":"df_weather_cleaned"}

```
print(f'Number of samples in df_energy is
{df_energy_features.shape[0]}')

city_list = df_weather_features['city_name'].unique()
grouped_weather = df_weather_cleaned.groupby('city_name')

for city in city_list:
    print(f'Number of samples in df_weather in {city} is
{grouped_weather.get_group(city).shape[0]}')
```

```
Number of samples in df_energy is 35064
Number of samples in df_weather in Valencia is 35064
Number of samples in df_weather in Madrid is 35064
Number of samples in df_weather in Bilbao is 35064
Number of samples in df_weather in  Barcelona is 35064
Number of samples in df_weather in Seville is 35064
```

```
df_weather_all_cities = [grouped_weather.get_group(x) for x in
grouped_weather.groups]

df_weather_all_cities[0]
```

{"summary":"{\n \"name\": \"df_weather_all_cities[0]\",\n \"rows\":
35064,\n \"fields\": [\n   {\n   \"column\": \"time\",\n
\"properties\": {\n   \"dtype\": \"date\",\n   \"min\":
\"2014-12-31 23:00:00+00:00\",\n   \"max\": \"2018-12-31
22:00:00+00:00\",\n   \"num_unique_values\": 35064,\n
\"samples\": [\n   \"2015-09-10 21:00:00+00:00\",\n
\"2018-09-20 07:00:00+00:00\",\n   \"2016-01-04
13:00:00+00:00\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"index\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 10236,\n   \"min\": 107350,\n   \"max\": 142821,\n
\"num_unique_values\": 35064,\n   \"samples\": [\n
113461,\n   140294,\n   116253\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"city_name\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 1,\n   \"samples\": [\n   \"
Barcelona\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"temp\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 6.723623493439147,\n   \"min\": 262.24,\n
\"max\": 309.15,\n   \"num_unique_values\": 5296,\n
\"samples\": [\n   280.96\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"pressure\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\":

```
7.526447844007588,\n        \"min\": 932.0,\n        \"max\": 1039.0,\
n        \"num_unique_values\": 62,\n        \"samples\": [\n
984.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"humidity\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 17.72080890337698,\n        \"min\":
0.0,\n        \"max\": 100.0,\n        \"num_unique_values\": 99,\n
\"samples\": [\n          31.0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"wind_speed\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1.9960811671624321,\n
\"min\": 0.0,\n        \"max\": 15.0,\n        \"num_unique_values\":
16,\n        \"samples\": [\n          7.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"wind_deg\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
108.56450488979964,\n        \"min\": 0.0,\n        \"max\": 360.0,\n
\"num_unique_values\": 361,\n        \"samples\": [\n          192.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"rain_1h\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.6677707337523273,\n        \"min\": 0.0,\n        \"max\": 12.0,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          0.3\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"snow_3h\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.0,\n        \"min\": 0.0,\n        \"max\": 0.0,\n
\"num_unique_values\": 1,\n        \"samples\": [\n          0.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"clouds_all\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
27.32844286577024,\n        \"min\": 0.0,\n        \"max\": 100.0,\n
\"num_unique_values\": 86,\n        \"samples\": [\n          49.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"holiday\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.2525137547258822,\n        \"min\": 0.0,\n        \"max\": 1.0,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          0.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe"}

# Loop through each DataFrame in the list and drop the 'index' column
for i in range(len(df_weather_all_cities)):
    df_weather_all_cities[i] =
df_weather_all_cities[i].drop(['index'], axis=1, errors='ignore')

# Print the first DataFrame in the list (or any specific one)
print(df_weather_all_cities[0])
```

```
                          city_name     temp  pressure  humidity  \
time
2014-12-31 23:00:00+00:00   Barcelona  281.625    1035.0     100.0
2015-01-01 00:00:00+00:00   Barcelona  281.625    1035.0     100.0
2015-01-01 01:00:00+00:00   Barcelona  281.286    1036.0     100.0
2015-01-01 02:00:00+00:00   Barcelona  281.286    1036.0     100.0
2015-01-01 03:00:00+00:00   Barcelona  281.286    1036.0     100.0
...                               ...      ...       ...       ...
2018-12-31 18:00:00+00:00   Barcelona  284.130    1027.0      71.0
2018-12-31 19:00:00+00:00   Barcelona  282.640    1027.0      62.0
2018-12-31 20:00:00+00:00   Barcelona  282.140    1028.0      53.0
2018-12-31 21:00:00+00:00   Barcelona  281.130    1028.0      50.0
2018-12-31 22:00:00+00:00   Barcelona  280.130    1028.0     100.0

                           wind_speed  wind_deg  rain_1h  snow_3h
clouds_all  \
time

2014-12-31 23:00:00+00:00         7.0      58.0      0.0      0.0
0.0
2015-01-01 00:00:00+00:00         7.0      58.0      0.0      0.0
0.0
2015-01-01 01:00:00+00:00         7.0      48.0      0.0      0.0
0.0
2015-01-01 02:00:00+00:00         7.0      48.0      0.0      0.0
0.0
2015-01-01 03:00:00+00:00         7.0      48.0      0.0      0.0
0.0
...                               ...       ...      ...      ...
...
2018-12-31 18:00:00+00:00         1.0     250.0      0.0      0.0
0.0
2018-12-31 19:00:00+00:00         3.0     270.0      0.0      0.0
0.0
2018-12-31 20:00:00+00:00         4.0     300.0      0.0      0.0
0.0
2018-12-31 21:00:00+00:00         5.0     320.0      0.0      0.0
0.0
2018-12-31 22:00:00+00:00         5.0     310.0      0.0      0.0
0.0

                           holiday
time
2014-12-31 23:00:00+00:00      1.0
2015-01-01 00:00:00+00:00      1.0
2015-01-01 01:00:00+00:00      1.0
2015-01-01 02:00:00+00:00      1.0
2015-01-01 03:00:00+00:00      1.0
...                            ...
2018-12-31 18:00:00+00:00      1.0
```

```
2018-12-31 19:00:00+00:00      1.0
2018-12-31 20:00:00+00:00      1.0
2018-12-31 21:00:00+00:00      1.0
2018-12-31 22:00:00+00:00      1.0

[35064 rows x 10 columns]

df_weather_energy = df_energy_features

for df_city in df_weather_all_cities:
    city_name = df_city.iloc[0]['city_name'].replace(' ', '')
    df_temp_city = df_city.add_suffix(f'_{city_name}')
    df_weather_energy = pd.concat([df_weather_energy, df_temp_city],
axis=1)
    df_weather_energy =
df_weather_energy.drop(f'city_name_{city_name}' , axis=1)

df_weather_energy.isnull().sum()

generation biomass                                0
generation fossil gas                             0
generation fossil oil                             0
generation hydro pumped storage consumption       0
generation hydro run-of-river and poundage        0
                                                 ..
wind_deg_Valencia                                 0
rain_1h_Valencia                                  0
snow_3h_Valencia                                  0
clouds_all_Valencia                               0
holiday_Valencia                                  0
Length: 62, dtype: int64

df_weather_energy.duplicated().sum()

0

df_weather_energy
```

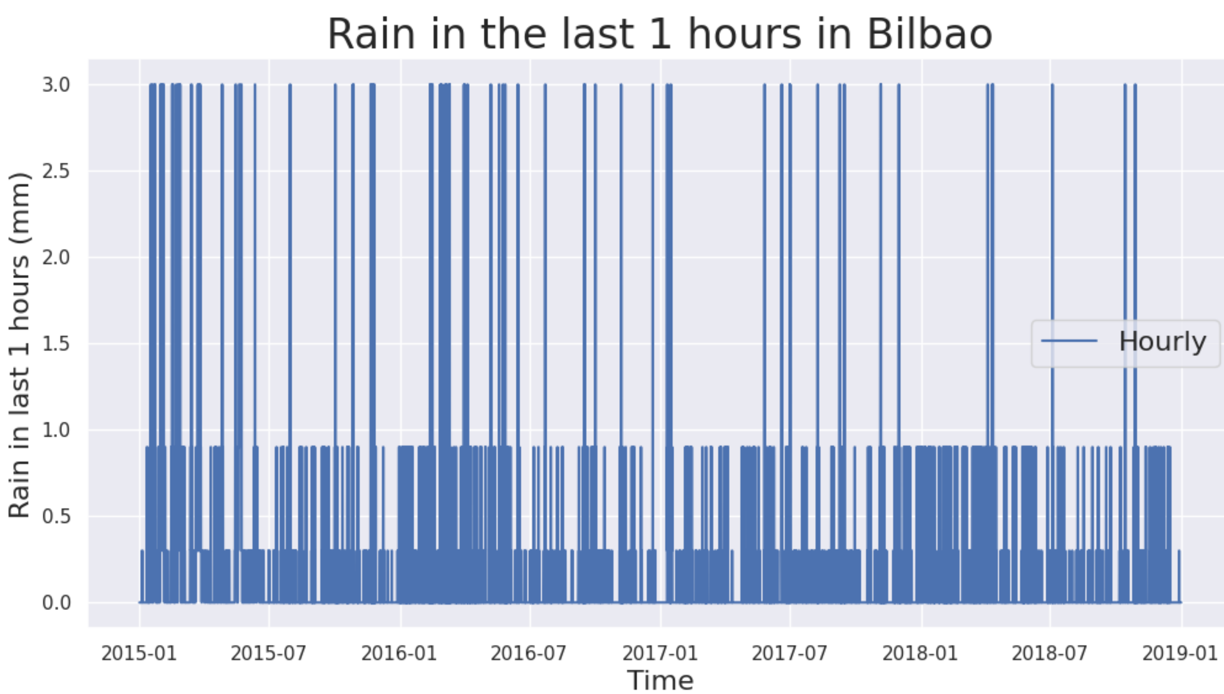
{"type":"dataframe","variable_name":"df_weather_energy"}

```
df_weather_energy.columns

Index(['generation biomass', 'generation fossil gas', 'generation
fossil oil',
       'generation hydro pumped storage consumption',
       'generation hydro run-of-river and poundage',
       'generation hydro water reservoir', 'generation nuclear',
       'generation other', 'generation other renewable', 'generation
solar',
       'generation waste', 'generation wind onshore', 'total load
forecast',
```

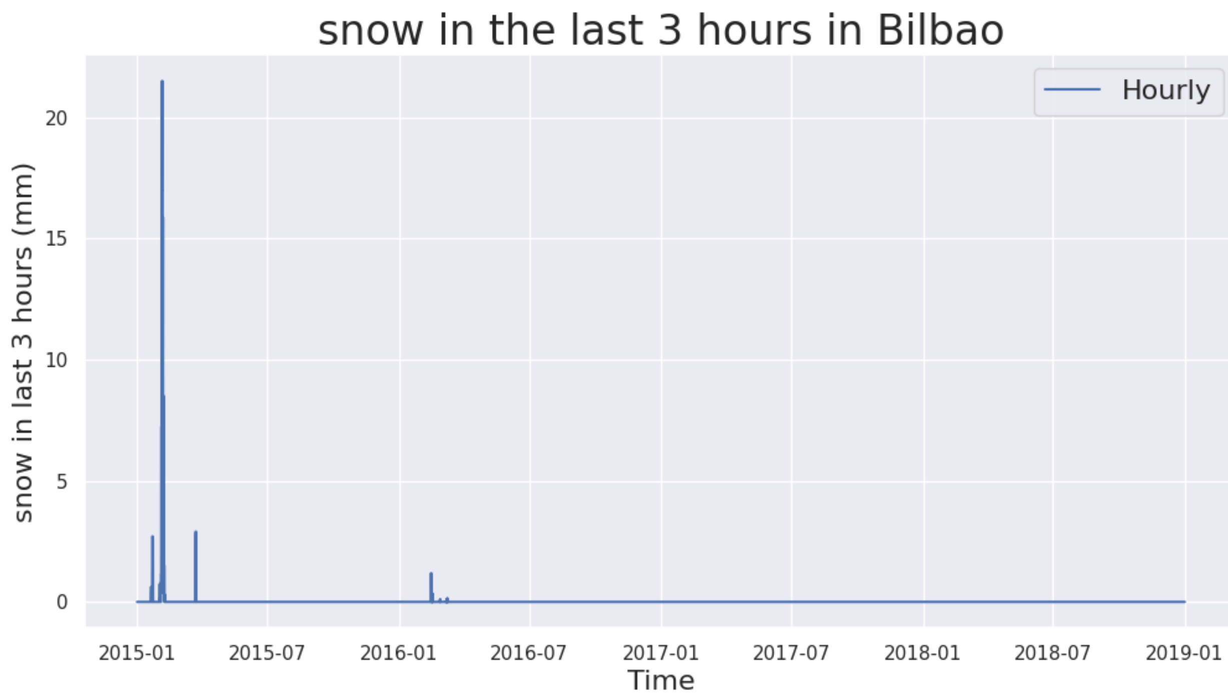
```
       'total load actual', 'price day ahead', 'price actual',
       'generation_total', 'temp_Barcelona', 'pressure_Barcelona',
       'humidity_Barcelona', 'wind_speed_Barcelona',
'wind_deg_Barcelona',
       'rain_1h_Barcelona', 'snow_3h_Barcelona',
'clouds_all_Barcelona',
       'holiday_Barcelona', 'temp_Bilbao', 'pressure_Bilbao',
       'humidity_Bilbao', 'wind_speed_Bilbao', 'wind_deg_Bilbao',
       'rain_1h_Bilbao', 'snow_3h_Bilbao', 'clouds_all_Bilbao',
       'holiday_Bilbao', 'temp_Madrid', 'pressure_Madrid',
'humidity_Madrid',
       'wind_speed_Madrid', 'wind_deg_Madrid', 'rain_1h_Madrid',
       'snow_3h_Madrid', 'clouds_all_Madrid', 'holiday_Madrid',
'temp_Seville',
       'pressure_Seville', 'humidity_Seville', 'wind_speed_Seville',
       'wind_deg_Seville', 'rain_1h_Seville', 'snow_3h_Seville',
       'clouds_all_Seville', 'holiday_Seville', 'temp_Valencia',
       'pressure_Valencia', 'humidity_Valencia',
'wind_speed_Valencia',
       'wind_deg_Valencia', 'rain_1h_Valencia', 'snow_3h_Valencia',
       'clouds_all_Valencia', 'holiday_Valencia'],
      dtype='object')
```

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

def plot_series(df=None, column=None, series=None,
                label=None, ylabel=None, title=None, start=0,
end=None):

    sns.set()

    # Create the figure and axis
    fig, px = plt.subplots(figsize=(12, 6))  # More reasonable default
size

    # Set the x-axis label
    px.set_xlabel('Time', fontsize=16)

    # Plot from DataFrame if 'column' is provided
    if df is not None and column is not None:
        # Slice the DataFrame for start:end
        px.plot(df[column].iloc[start:end], label=label)

    # Plot from Series if 'series' is provided
    elif series is not None and not series.empty:
        px.plot(series.iloc[start:end], label=label)

    # Set the y-axis label if provided
```

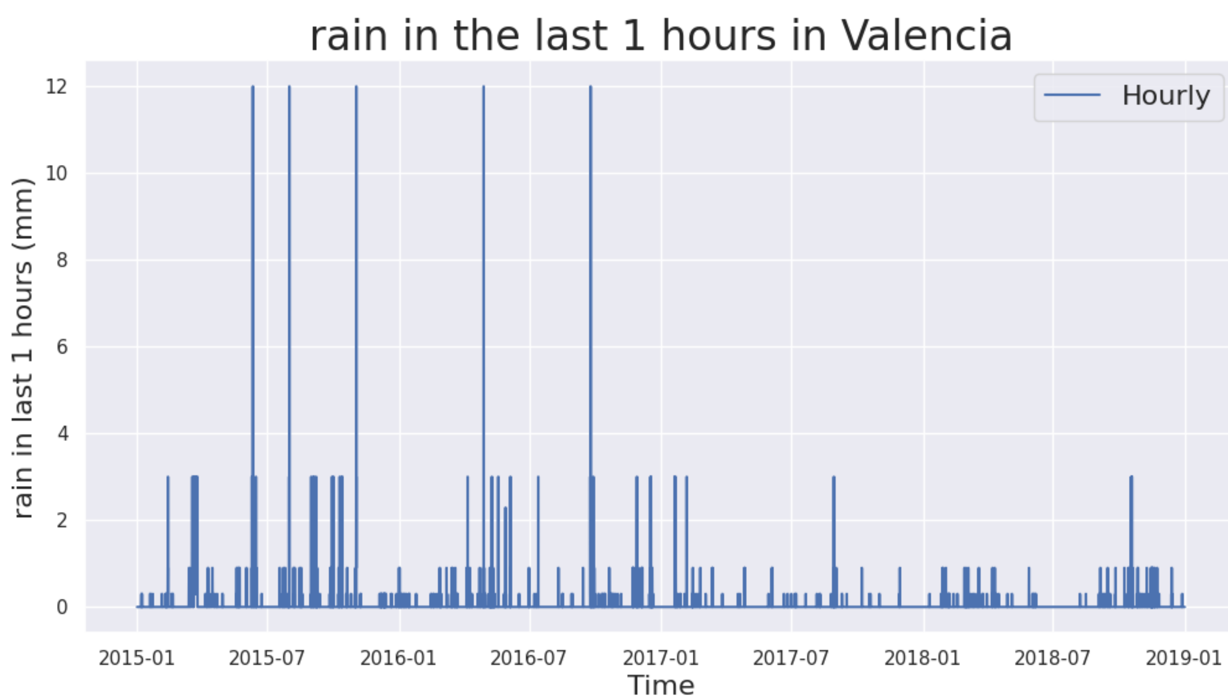
```
    if ylabel:
        px.set_ylabel(ylabel, fontsize=16)

    # Set the title if provided
    if title:
        px.set_title(title, fontsize=24)

    # Add legend if label is provided
    if label:
        px.legend(fontsize=16)

    # Enable grid
    px.grid(True)

    # Return the axis object
    return px

px = plot_series(df_weather_energy, 'rain_1h_Bilbao',
                 label='Hourly', ylabel='Rain in last 1 hours (mm)',
                 title='Rain in the last 1 hours in Bilbao')
plt.show()
```



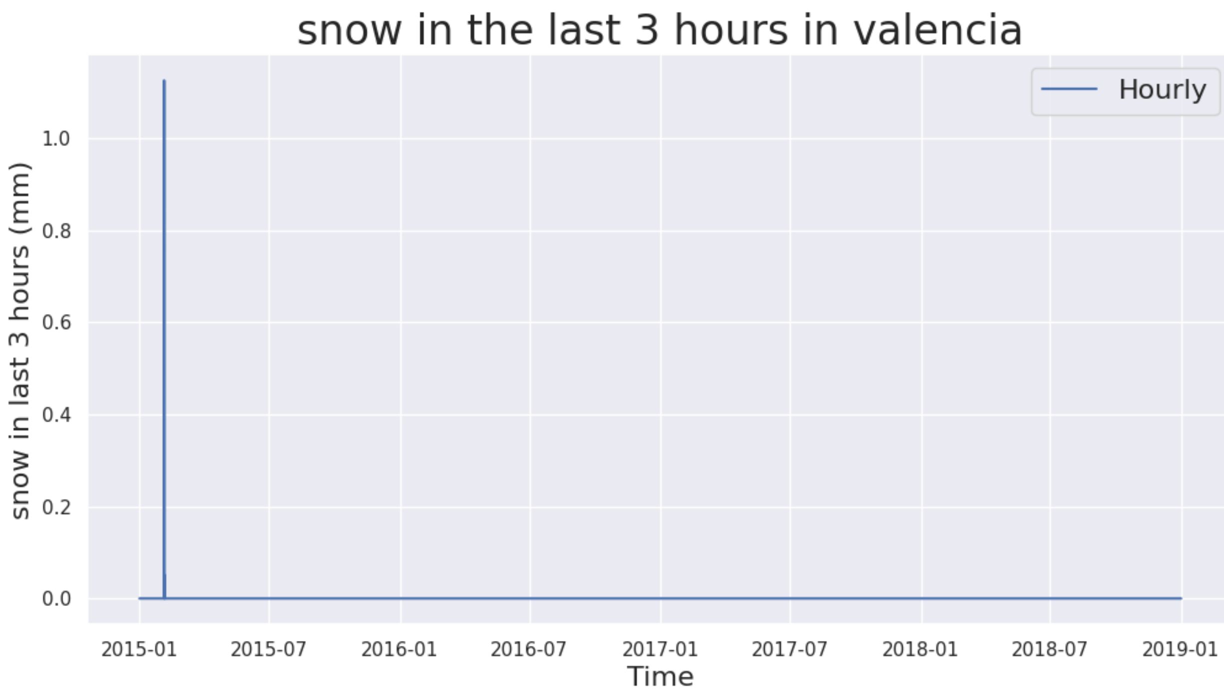

```
px = plot_series(df_weather_energy, 'snow_3h_Bilbao',
                 label='Hourly', ylabel='snow in last 3 hours (mm)',
                 title='snow in the last 3 hours in Bilbao')
plt.show()
```




```
px = plot_series(df_weather_energy, 'rain_1h_Valencia',
                 label='Hourly', ylabel='rain in last 1 hours (mm)',
                 title='rain in the last 1 hours in Valencia')
plt.show()
```

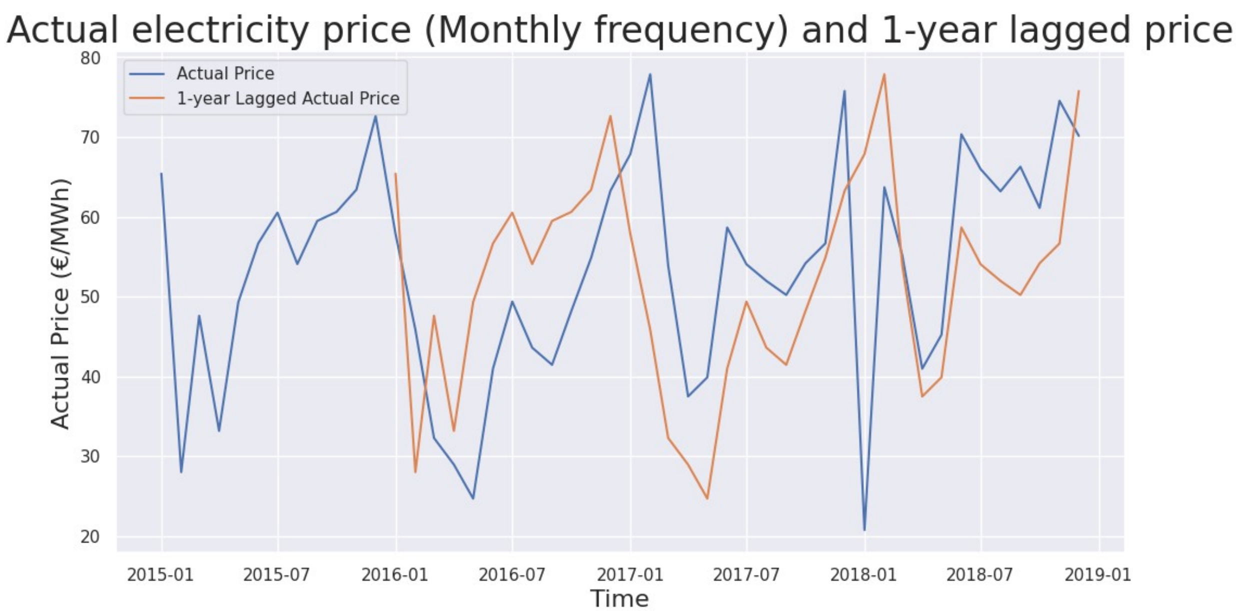


```
px = plot_series(df_weather_energy, 'snow_3h_Valencia',
                 label='Hourly', ylabel='snow in last 3 hours (mm)',
                 title='snow in the last 3 hours in valencia')
plt.show()
```




```
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Prepare data
time_index = df_weather_energy.index
price_actual = df_weather_energy["price actual"]
daily_rolling = price_actual.rolling(window=24).mean()
weekly_rolling = price_actual.rolling(window=24*7).mean()

# Create subplot
fig = make_subplots()

# Add traces for actual price, daily average, and weekly average with
custom colors
fig.add_trace(go.Scatter(x=time_index, y=price_actual, mode='lines',
name="price actual", line=dict(color='blue')))
fig.add_trace(go.Scatter(x=time_index, y=daily_rolling, mode='lines',
name="rolling window = daily ave", line=dict(color='green')))
fig.add_trace(go.Scatter(x=time_index, y=weekly_rolling, mode='lines',
name="rolling window = weekly ave", line=dict(color='orange')))

# Optionally, add range slider (uncomment if needed)
# fig.update_xaxes(rangeslider_visible=True)
```

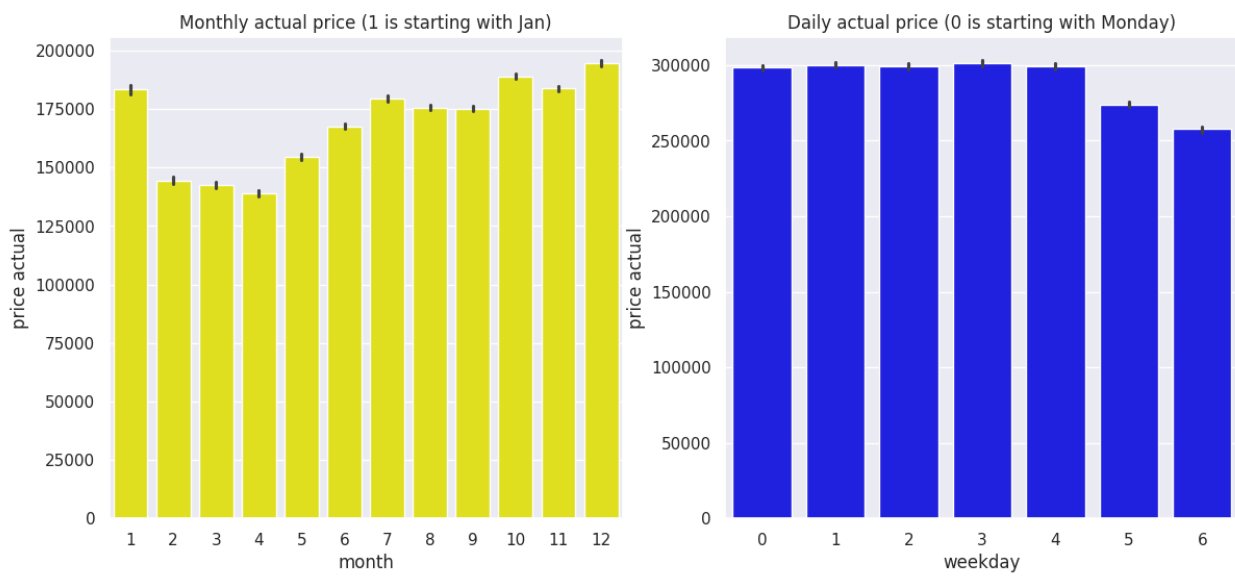
```
# Show the plot
fig.show()

monthly_price = df_weather_energy['price actual'].asfreq('ME')
ax = plot_series(series=monthly_price, ylabel='Actual Price (€/MWh)',
                 title='Actual electricity price (Monthly frequency)
and 1-year lagged price')
shifted = df_weather_energy['price actual'].asfreq('ME').shift(12)
ax.plot(shifted, label='Hourly')
ax.legend(['Actual Price', '1-year Lagged Actual Price'])
plt.show()
```




```
df_weather_energy['hour'] =
df_weather_energy.index.to_series().apply(lambda x: x.hour)
df_weather_energy['weekday'] =
df_weather_energy.index.to_series().apply(lambda x: x.weekday())
df_weather_energy['month'] =
df_weather_energy.index.to_series().apply(lambda x: x.month)
df_weather_energy['year'] =
df_weather_energy.index.to_series().apply(lambda x: x.year)

fig, axes = plt.subplots(ncols=2, figsize=(14, 6))
sns.set(style="darkgrid")

sns.barplot(
    x="month",
    y="price actual",
    data=df_weather_energy,
    estimator=sum,
    color='yellow',
```

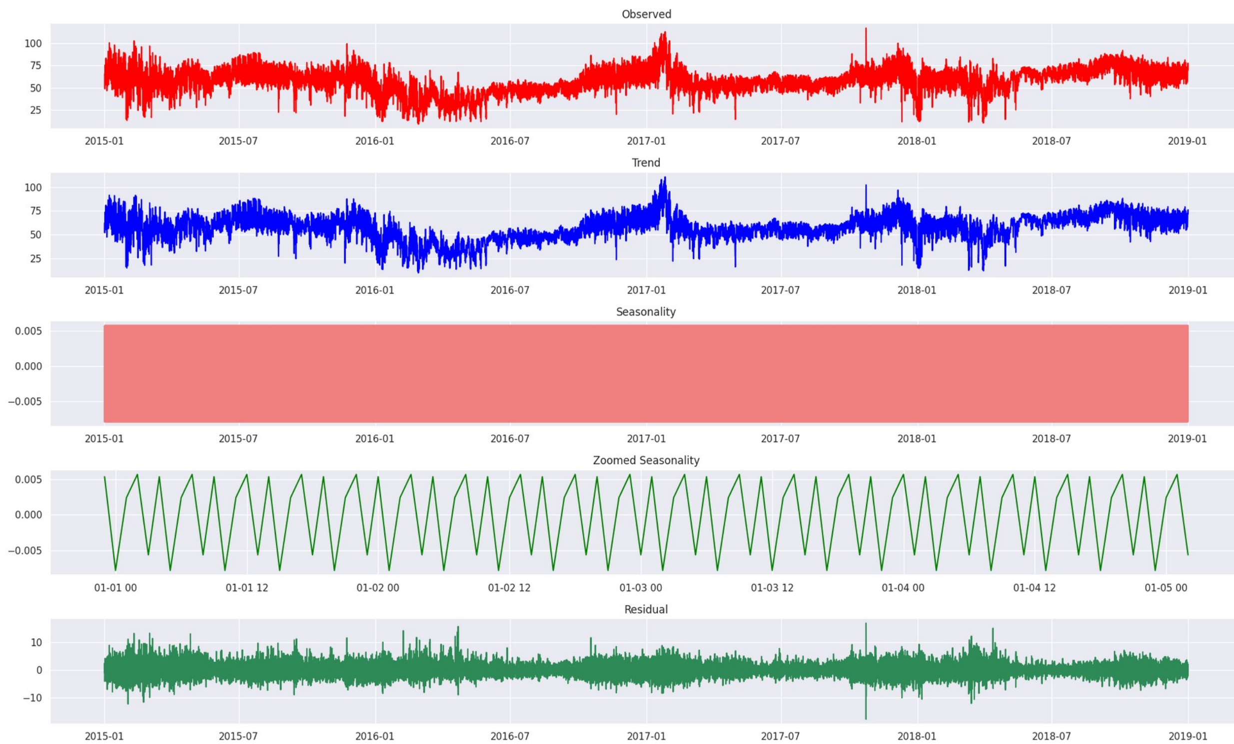
```
    ax=axes[0]);
axes[0].set_title('Monthly actual price (1 is starting with Jan)')

sns.barplot(
    x="weekday",
    y="price actual",
    data=df_weather_energy,
    estimator=sum,
    color='blue',
    ax=axes[1]);
axes[1].set_title('Daily actual price (0 is starting with Monday)')
```

```
Text(0.5, 1.0, 'Daily actual price (0 is starting with Monday)')
```




```
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

# Perform decomposition
decompose_result = seasonal_decompose(df_weather_energy['price
actual'], period=5, model='additive')

# Components to plot
components = {
    'Observed': decompose_result.observed,
    'Trend': decompose_result.trend,
    'Seasonality': decompose_result.seasonal,
    'Zoomed Seasonality': decompose_result.seasonal[:100],
    'Residual': decompose_result.resid
}
colors = ['red', 'blue', 'lightcoral', 'green', 'seagreen']

# Plot components in a loop
```

```python
fig, axes = plt.subplots(5, 1, figsize=(20, 12))
for ax, (title, data), color in zip(axes, components.items(), colors):
    ax.plot(data, color=color)
    ax.set_title(title)

fig.tight_layout()
plt.show()
```




```python
result = adfuller(df_weather_energy[['price actual']])
print('ADF Statistic:', result[0])
print('p-value:', result[1])
print('Critical Values:', result[4])
```

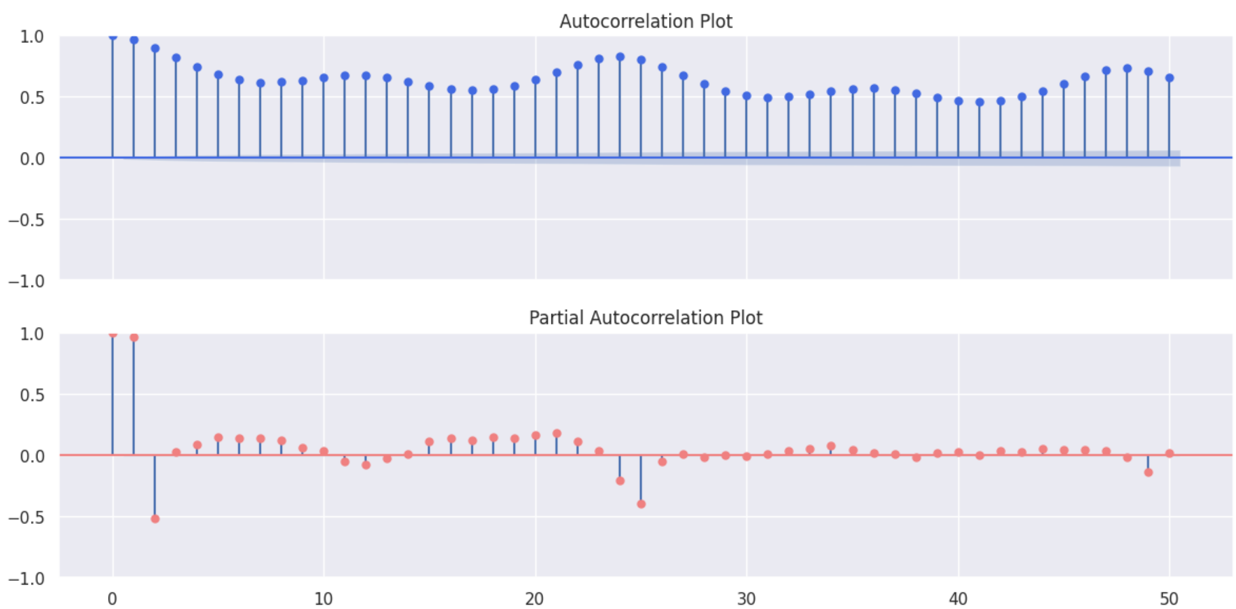
```
ADF Statistic: -9.147016232851248
p-value: 2.750493484933306e-15
Critical Values: {'1%': -3.4305367814665044, '5%': -
2.8616225527935106, '10%': -2.566813940257257}
```

```python
result = adfuller(df_weather_energy[['total load actual']])
print('ADF Statistic:', result[0])
print('p-value:', result[1])
print('Critical Values:', result[4])
```

```
ADF Statistic: -21.420315756960584
p-value: 0.0
Critical Values: {'1%': -3.43053679213716, '5%': -2.8616225575095284,
'10%': -2.566813942767471}
```

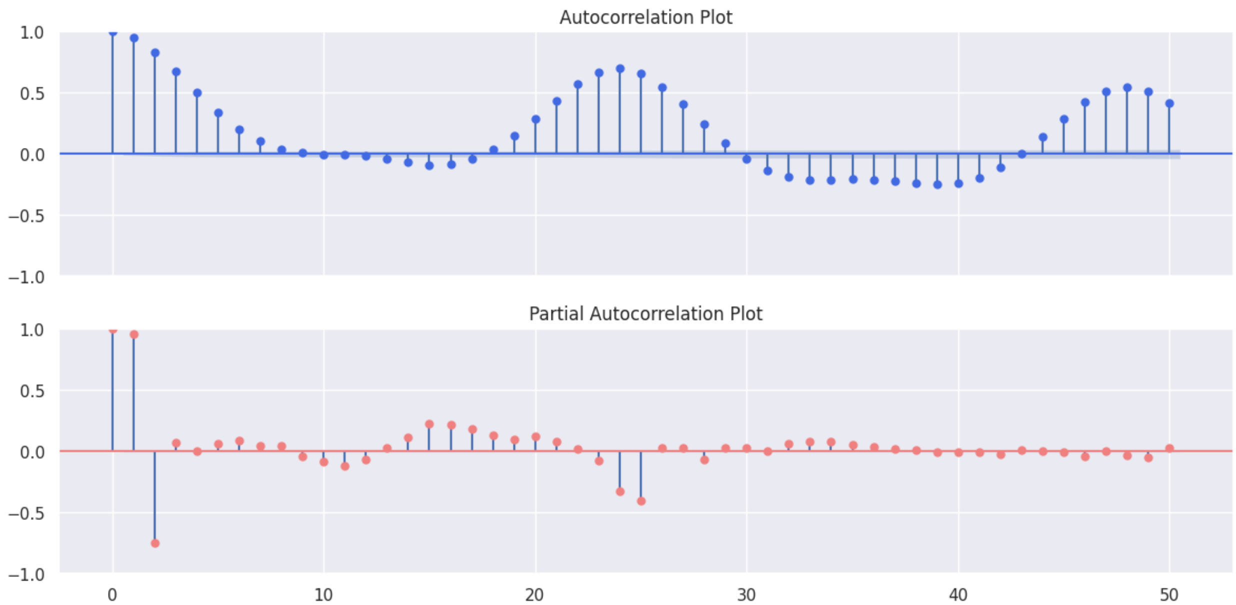
Conclusion: The time series is stationary, so no additional differencing is needed.

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt

# Define plot settings for autocorrelation and partial autocorrelation
fig, axes = plt.subplots(2, 1, figsize=(12, 6), sharex=True)
titles = ['Autocorrelation Plot', 'Partial Autocorrelation Plot']
colors = ['royalblue', 'lightcoral']
plots = [plot_acf, plot_pacf]

# Loop through each plot type
for i, plot_func in enumerate(plots):
    plot_func(df_weather_energy['price actual'], lags=50, ax=axes[i],
color=colors[i])
    axes[i].set_title(titles[i])

plt.tight_layout()
plt.show()
```

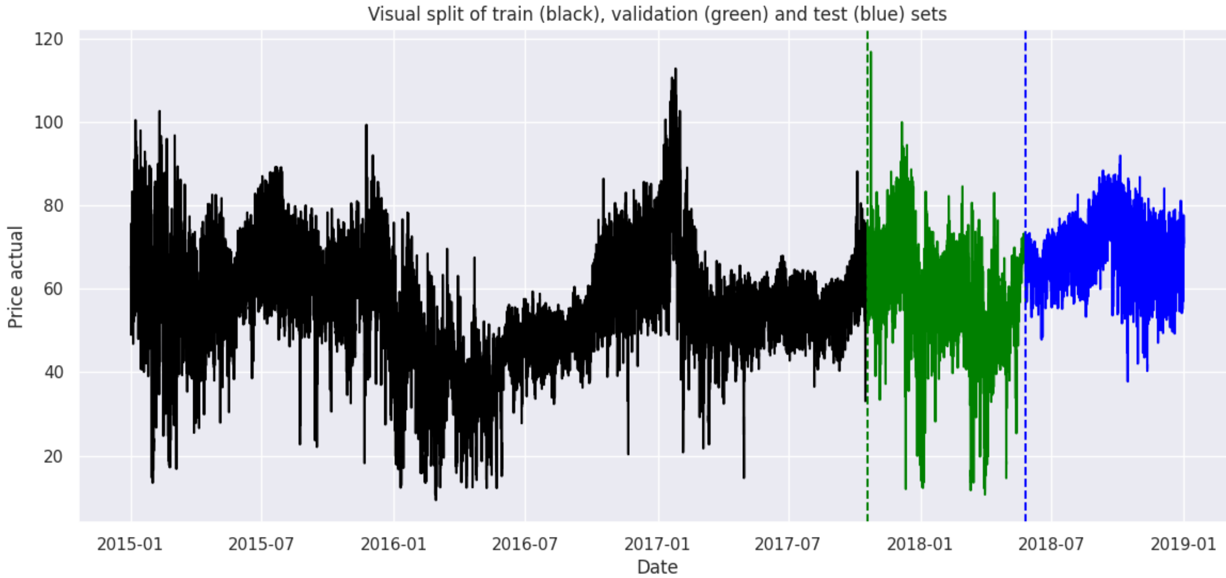



```
fig, axes = plt.subplots(2, 1, figsize=(12, 6), sharex=True)
titles = ['Autocorrelation Plot', 'Partial Autocorrelation Plot']
colors = ['royalblue', 'lightcoral']
plots = [plot_acf, plot_pacf]

# Loop through each plot type
for i, plot_func in enumerate(plots):
    plot_func(df_weather_energy['total load actual'], lags=50,
ax=axes[i], color=colors[i])
    axes[i].set_title(titles[i])
```

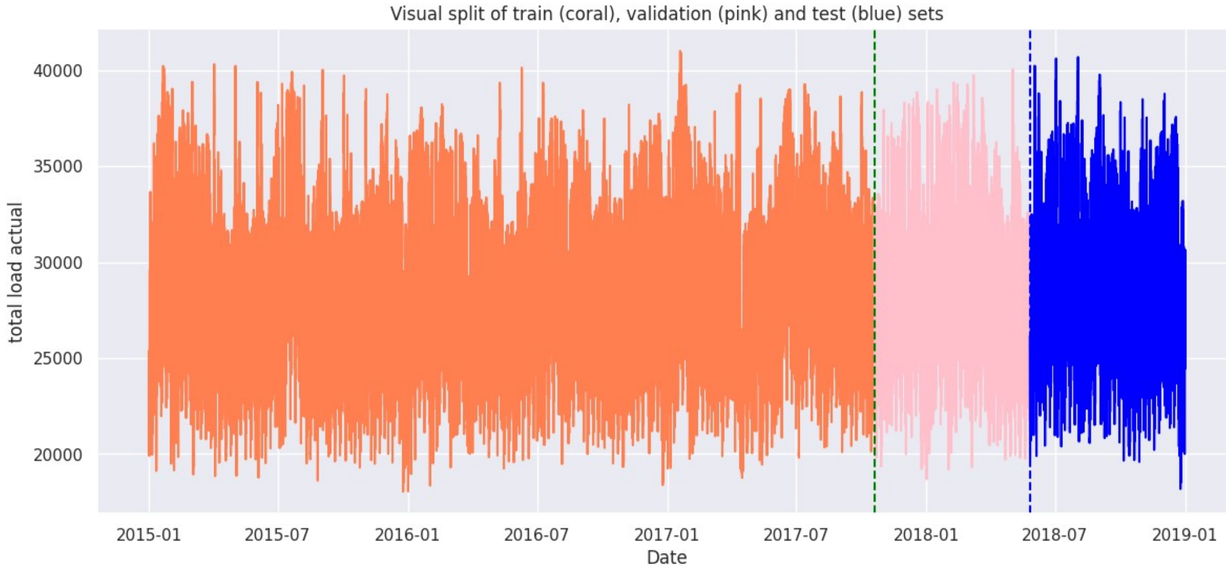
```
plt.tight_layout()
plt.show()
```




```
import matplotlib.pyplot as plt

train_cutoff = int(len(df_weather_energy) * 0.7)  # Example: 70% for
training
val_cutoff = int(len(df_weather_energy) * 0.85)  # Example: 15% for
validation (85% - 70%)

fig, axes = plt.subplots(figsize=(14, 6))
axes.plot(df_weather_energy['price actual'].iloc[:train_cutoff],
color='black')
axes.plot(df_weather_energy['price actual'].iloc[train_cutoff +
1:val_cutoff], color='green')
axes.plot(df_weather_energy['price actual'].iloc[val_cutoff + 1:],
color='blue')
axes.axvline(x=df_weather_energy.index[train_cutoff], color='green',
linestyle='--')
axes.axvline(x=df_weather_energy.index[val_cutoff], color='blue',
linestyle='--')
axes.set_title('Visual split of train (black), validation (green) and
test (blue) sets')
axes.set_xlabel('Date')
axes.set_ylabel('Price actual')
plt.show()
```




```
train_cutoff = int(len(df_weather_energy) * 0.7)  # Example: 70% for
training
val_cutoff = int(len(df_weather_energy) * 0.85)  # Example: 15% for
validation (85% - 70%)

fig, axes = plt.subplots(figsize=(14, 6))
axes.plot(df_weather_energy['total load actual'].iloc[:train_cutoff],
color='coral')
axes.plot(df_weather_energy['total load actual'].iloc[train_cutoff +
1:val_cutoff], color='pink')
axes.plot(df_weather_energy['total load actual'].iloc[val_cutoff +
1:], color='blue')
axes.axvline(x=df_weather_energy.index[train_cutoff], color='green',
linestyle='--')
axes.axvline(x=df_weather_energy.index[val_cutoff], color='blue',
linestyle='--')
axes.set_title('Visual split of train (coral), validation (pink) and
test (blue) sets')
axes.set_xlabel('Date')
axes.set_ylabel('total load actual')
plt.show()
```



XG BOOST MODEL

```
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
import xgboost as xgb
import pandas as pd
import numpy as np

# Assuming 'df_weather_energy' is already loaded as a DataFrame
# Step 1: Prepare features (X) and target (y)
X = df_weather_energy.drop(['price actual'], axis=1)
y = df_weather_energy['price actual']

# Step 2: Train-Test Split
X_train, X_temp, y_train, y_temp = train_test_split(X, y,
test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp,
test_size=0.5, random_state=42)

# Step 3: Convert Data to DMatrix
dtrain = xgb.DMatrix(X_train, label=y_train)
dval = xgb.DMatrix(X_val, label=y_val)
dtest = xgb.DMatrix(X_test, label=y_test)

# Step 4: Define Parameters for XGBoost
params = {
    'objective': 'reg:squarederror',  # Regression objective
    'random_state': 42,
    'eval_metric': 'rmse',  # Metric to evaluate during training
}
```

```python
# Step 5: Train XGBoost Model with Early Stopping
evals = [(dtrain, 'train'), (dval, 'eval')]
evals_result = {}  # To store the evaluation results

xgboost_model = xgb.train(
    params=params,
    dtrain=dtrain,
    num_boost_round=500,  # Maximum number of boosting rounds
    early_stopping_rounds=10,  # Stop training if no improvement after
10 rounds
    evals=evals,
    evals_result=evals_result,  # Store evaluation results here
    verbose_eval=True
)

# Step 6: Make Predictions
y_pred = xgboost_model.predict(dtest)

# Step 7: Calculate Metrics
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mape = np.mean(np.abs((y_test - y_pred) / y_test)) * 100  # Mean
Absolute Percentage Error

# Calculate Accuracy as 1 - (Normalized MAE)
mean_actual = np.mean(y_test)
accuracy = (1 - (mae / mean_actual)) * 100

# Print Results
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Percentage Error (MAPE): {mape:.2f}%")
print(f"Accuracy: {accuracy:.2f}%")

# Plot Results using the plot_results function
def plot_results(y_pred, y_test, evals_result, model_name):
    fig, ax = plt.subplots(2, 1, figsize=(12, 6))

    # Plotting prediction vs actual for the first 1000 observations
    ax[0].plot(y_pred[:1000], label='Prediction')
    ax[0].plot(y_test[:1000].values, label='Actual')
    ax[0].legend(loc='upper left')
    ax[0].set_title(f'Prediction vs Actual for 1000 observations in
Test Set ({model_name})')
    ax[0].set_xlabel('Observation')
    ax[0].set_ylabel('price actual')

    # Extract training and validation RMSE from evals_result
    train_rmse = evals_result['train']['rmse']
    val_rmse = evals_result['eval']['rmse']
```

```python
    # Plot RMSE for training and validation
    ax[1].plot(train_rmse, label='Training RMSE')
    ax[1].plot(val_rmse, label='Validation RMSE')

    ax[1].legend()
    ax[1].set_title(f'Training and Validation RMSE ({model_name})')
    ax[1].set_xlabel('Iteration')
    ax[1].set_ylabel('RMSE')

    fig.tight_layout()
    plt.show()

# Call the function to plot results
plot_results(y_pred, y_test, evals_result, 'XGBoost')
```

```
[0]	train-rmse:11.38927	eval-rmse:11.33749
[1]	train-rmse:9.49666	eval-rmse:9.54363
[2]	train-rmse:8.24840	eval-rmse:8.37787
[3]	train-rmse:7.38621	eval-rmse:7.60264
[4]	train-rmse:6.77502	eval-rmse:7.03955
[5]	train-rmse:6.33795	eval-rmse:6.64540
[6]	train-rmse:5.95068	eval-rmse:6.31109
[7]	train-rmse:5.65098	eval-rmse:6.05233
[8]	train-rmse:5.34229	eval-rmse:5.78414
[9]	train-rmse:4.98549	eval-rmse:5.45353
[10]	train-rmse:4.83535	eval-rmse:5.31613
[11]	train-rmse:4.70675	eval-rmse:5.21496
[12]	train-rmse:4.52258	eval-rmse:5.04659
[13]	train-rmse:4.40622	eval-rmse:4.93890
[14]	train-rmse:4.31609	eval-rmse:4.86779
[15]	train-rmse:4.23425	eval-rmse:4.79228
[16]	train-rmse:4.16175	eval-rmse:4.72743
[17]	train-rmse:4.04820	eval-rmse:4.62982
[18]	train-rmse:3.95808	eval-rmse:4.56161
[19]	train-rmse:3.89846	eval-rmse:4.50902
[20]	train-rmse:3.84802	eval-rmse:4.45727
[21]	train-rmse:3.81566	eval-rmse:4.43067
[22]	train-rmse:3.77556	eval-rmse:4.38767
[23]	train-rmse:3.72363	eval-rmse:4.34813
[24]	train-rmse:3.66229	eval-rmse:4.30377
[25]	train-rmse:3.62402	eval-rmse:4.27363
[26]	train-rmse:3.56306	eval-rmse:4.23637
[27]	train-rmse:3.52139	eval-rmse:4.21152
[28]	train-rmse:3.47709	eval-rmse:4.18282
[29]	train-rmse:3.41014	eval-rmse:4.12289
[30]	train-rmse:3.37630	eval-rmse:4.09194
[31]	train-rmse:3.34253	eval-rmse:4.06087
[32]	train-rmse:3.31469	eval-rmse:4.03975
[33]	train-rmse:3.29060	eval-rmse:4.01448
```

```
[34]	train-rmse:3.26279	eval-rmse:3.99697
[35]	train-rmse:3.23069	eval-rmse:3.96667
[36]	train-rmse:3.19139	eval-rmse:3.94352
[37]	train-rmse:3.15114	eval-rmse:3.92578
[38]	train-rmse:3.13272	eval-rmse:3.91372
[39]	train-rmse:3.09996	eval-rmse:3.89237
[40]	train-rmse:3.08979	eval-rmse:3.88515
[41]	train-rmse:3.05480	eval-rmse:3.86289
[42]	train-rmse:3.03801	eval-rmse:3.85157
[43]	train-rmse:3.02373	eval-rmse:3.83911
[44]	train-rmse:2.99663	eval-rmse:3.82250
[45]	train-rmse:2.97765	eval-rmse:3.80438
[46]	train-rmse:2.95221	eval-rmse:3.78888
[47]	train-rmse:2.93800	eval-rmse:3.78084
[48]	train-rmse:2.90555	eval-rmse:3.76316
[49]	train-rmse:2.88388	eval-rmse:3.75126
[50]	train-rmse:2.87536	eval-rmse:3.74451
[51]	train-rmse:2.86244	eval-rmse:3.73523
[52]	train-rmse:2.84467	eval-rmse:3.72077
[53]	train-rmse:2.83743	eval-rmse:3.71625
[54]	train-rmse:2.82206	eval-rmse:3.71254
[55]	train-rmse:2.79395	eval-rmse:3.68943
[56]	train-rmse:2.78174	eval-rmse:3.68493
[57]	train-rmse:2.76171	eval-rmse:3.67586
[58]	train-rmse:2.73933	eval-rmse:3.66392
[59]	train-rmse:2.72369	eval-rmse:3.65446
[60]	train-rmse:2.72036	eval-rmse:3.65288
[61]	train-rmse:2.70256	eval-rmse:3.64514
[62]	train-rmse:2.67449	eval-rmse:3.63448
[63]	train-rmse:2.66749	eval-rmse:3.63247
[64]	train-rmse:2.64816	eval-rmse:3.62332
[65]	train-rmse:2.63071	eval-rmse:3.61405
[66]	train-rmse:2.61008	eval-rmse:3.60517
[67]	train-rmse:2.59011	eval-rmse:3.59473
[68]	train-rmse:2.56816	eval-rmse:3.57831
[69]	train-rmse:2.55043	eval-rmse:3.56364
[70]	train-rmse:2.52969	eval-rmse:3.54701
[71]	train-rmse:2.51953	eval-rmse:3.54579
[72]	train-rmse:2.51578	eval-rmse:3.54461
[73]	train-rmse:2.49932	eval-rmse:3.53537
[74]	train-rmse:2.47578	eval-rmse:3.52025
[75]	train-rmse:2.45465	eval-rmse:3.50653
[76]	train-rmse:2.43914	eval-rmse:3.49404
[77]	train-rmse:2.42670	eval-rmse:3.48207
[78]	train-rmse:2.40762	eval-rmse:3.47160
[79]	train-rmse:2.39953	eval-rmse:3.46805
[80]	train-rmse:2.38679	eval-rmse:3.45782
[81]	train-rmse:2.37203	eval-rmse:3.44599
[82]	train-rmse:2.36705	eval-rmse:3.44041
```

```
[83]	train-rmse:2.35102	eval-rmse:3.42902
[84]	train-rmse:2.33621	eval-rmse:3.42338
[85]	train-rmse:2.32452	eval-rmse:3.41649
[86]	train-rmse:2.31234	eval-rmse:3.41457
[87]	train-rmse:2.29208	eval-rmse:3.40001
[88]	train-rmse:2.27821	eval-rmse:3.38936
[89]	train-rmse:2.26578	eval-rmse:3.37989
[90]	train-rmse:2.25651	eval-rmse:3.37559
[91]	train-rmse:2.24594	eval-rmse:3.37229
[92]	train-rmse:2.23849	eval-rmse:3.37163
[93]	train-rmse:2.22981	eval-rmse:3.36851
[94]	train-rmse:2.22486	eval-rmse:3.36415
[95]	train-rmse:2.22152	eval-rmse:3.36535
[96]	train-rmse:2.21627	eval-rmse:3.36330
[97]	train-rmse:2.20489	eval-rmse:3.35841
[98]	train-rmse:2.19337	eval-rmse:3.35264
[99]	train-rmse:2.18834	eval-rmse:3.35093
[100]	train-rmse:2.17949	eval-rmse:3.34724
[101]	train-rmse:2.17547	eval-rmse:3.34608
[102]	train-rmse:2.16203	eval-rmse:3.34089
[103]	train-rmse:2.15011	eval-rmse:3.33898
[104]	train-rmse:2.14122	eval-rmse:3.33685
[105]	train-rmse:2.12653	eval-rmse:3.33045
[106]	train-rmse:2.11127	eval-rmse:3.32029
[107]	train-rmse:2.09908	eval-rmse:3.31432
[108]	train-rmse:2.09173	eval-rmse:3.30788
[109]	train-rmse:2.08359	eval-rmse:3.30627
[110]	train-rmse:2.07529	eval-rmse:3.30032
[111]	train-rmse:2.06937	eval-rmse:3.29747
[112]	train-rmse:2.06586	eval-rmse:3.29693
[113]	train-rmse:2.06459	eval-rmse:3.29615
[114]	train-rmse:2.05352	eval-rmse:3.29141
[115]	train-rmse:2.04504	eval-rmse:3.28847
[116]	train-rmse:2.03047	eval-rmse:3.27831
[117]	train-rmse:2.02030	eval-rmse:3.27451
[118]	train-rmse:2.00847	eval-rmse:3.26891
[119]	train-rmse:1.99320	eval-rmse:3.26135
[120]	train-rmse:1.98775	eval-rmse:3.26056
[121]	train-rmse:1.98139	eval-rmse:3.25730
[122]	train-rmse:1.97810	eval-rmse:3.25605
[123]	train-rmse:1.96683	eval-rmse:3.25465
[124]	train-rmse:1.96267	eval-rmse:3.25288
[125]	train-rmse:1.95196	eval-rmse:3.24776
[126]	train-rmse:1.94270	eval-rmse:3.24519
[127]	train-rmse:1.93933	eval-rmse:3.24340
[128]	train-rmse:1.93466	eval-rmse:3.23946
[129]	train-rmse:1.92750	eval-rmse:3.23685
[130]	train-rmse:1.91811	eval-rmse:3.23458
[131]	train-rmse:1.90872	eval-rmse:3.23249
```

```
[132]	train-rmse:1.90305	eval-rmse:3.23004
[133]	train-rmse:1.89249	eval-rmse:3.22443
[134]	train-rmse:1.88239	eval-rmse:3.21933
[135]	train-rmse:1.87964	eval-rmse:3.21853
[136]	train-rmse:1.87493	eval-rmse:3.21697
[137]	train-rmse:1.87132	eval-rmse:3.21428
[138]	train-rmse:1.86287	eval-rmse:3.20956
[139]	train-rmse:1.85292	eval-rmse:3.20437
[140]	train-rmse:1.84314	eval-rmse:3.20314
[141]	train-rmse:1.83248	eval-rmse:3.19823
[142]	train-rmse:1.82669	eval-rmse:3.19809
[143]	train-rmse:1.82017	eval-rmse:3.19687
[144]	train-rmse:1.81060	eval-rmse:3.19051
[145]	train-rmse:1.79934	eval-rmse:3.18992
[146]	train-rmse:1.79580	eval-rmse:3.18894
[147]	train-rmse:1.78525	eval-rmse:3.18150
[148]	train-rmse:1.77812	eval-rmse:3.17876
[149]	train-rmse:1.77164	eval-rmse:3.18041
[150]	train-rmse:1.76555	eval-rmse:3.17758
[151]	train-rmse:1.75844	eval-rmse:3.17286
[152]	train-rmse:1.74979	eval-rmse:3.17133
[153]	train-rmse:1.74774	eval-rmse:3.17185
[154]	train-rmse:1.74515	eval-rmse:3.17044
[155]	train-rmse:1.73586	eval-rmse:3.16630
[156]	train-rmse:1.72665	eval-rmse:3.16126
[157]	train-rmse:1.72138	eval-rmse:3.16047
[158]	train-rmse:1.71491	eval-rmse:3.15961
[159]	train-rmse:1.70776	eval-rmse:3.15381
[160]	train-rmse:1.70104	eval-rmse:3.15138
[161]	train-rmse:1.69809	eval-rmse:3.15128
[162]	train-rmse:1.69615	eval-rmse:3.15186
[163]	train-rmse:1.68774	eval-rmse:3.14884
[164]	train-rmse:1.68618	eval-rmse:3.14794
[165]	train-rmse:1.68158	eval-rmse:3.14767
[166]	train-rmse:1.67714	eval-rmse:3.14632
[167]	train-rmse:1.67457	eval-rmse:3.14559
[168]	train-rmse:1.67081	eval-rmse:3.14491
[169]	train-rmse:1.66421	eval-rmse:3.14185
[170]	train-rmse:1.65703	eval-rmse:3.13860
[171]	train-rmse:1.65055	eval-rmse:3.13468
[172]	train-rmse:1.64164	eval-rmse:3.13227
[173]	train-rmse:1.63289	eval-rmse:3.13122
[174]	train-rmse:1.62635	eval-rmse:3.12917
[175]	train-rmse:1.62223	eval-rmse:3.12822
[176]	train-rmse:1.61698	eval-rmse:3.12378
[177]	train-rmse:1.61029	eval-rmse:3.12152
[178]	train-rmse:1.60486	eval-rmse:3.12019
[179]	train-rmse:1.59921	eval-rmse:3.11963
[180]	train-rmse:1.59410	eval-rmse:3.11844
```

```
[181]	train-rmse:1.59147	eval-rmse:3.11798
[182]	train-rmse:1.58560	eval-rmse:3.11463
[183]	train-rmse:1.57958	eval-rmse:3.11033
[184]	train-rmse:1.57206	eval-rmse:3.10738
[185]	train-rmse:1.56508	eval-rmse:3.10419
[186]	train-rmse:1.55930	eval-rmse:3.10275
[187]	train-rmse:1.55568	eval-rmse:3.10211
[188]	train-rmse:1.54946	eval-rmse:3.09985
[189]	train-rmse:1.54446	eval-rmse:3.10162
[190]	train-rmse:1.53350	eval-rmse:3.09570
[191]	train-rmse:1.52759	eval-rmse:3.09285
[192]	train-rmse:1.52176	eval-rmse:3.09100
[193]	train-rmse:1.51580	eval-rmse:3.08756
[194]	train-rmse:1.51362	eval-rmse:3.08696
[195]	train-rmse:1.50785	eval-rmse:3.08530
[196]	train-rmse:1.50049	eval-rmse:3.08245
[197]	train-rmse:1.49420	eval-rmse:3.08102
[198]	train-rmse:1.49069	eval-rmse:3.07937
[199]	train-rmse:1.48706	eval-rmse:3.07804
[200]	train-rmse:1.48426	eval-rmse:3.07860
[201]	train-rmse:1.47736	eval-rmse:3.07735
[202]	train-rmse:1.47261	eval-rmse:3.07645
[203]	train-rmse:1.46913	eval-rmse:3.07483
[204]	train-rmse:1.46318	eval-rmse:3.07306
[205]	train-rmse:1.45803	eval-rmse:3.07073
[206]	train-rmse:1.45381	eval-rmse:3.06877
[207]	train-rmse:1.44376	eval-rmse:3.06743
[208]	train-rmse:1.44043	eval-rmse:3.06711
[209]	train-rmse:1.43455	eval-rmse:3.06328
[210]	train-rmse:1.42985	eval-rmse:3.06289
[211]	train-rmse:1.42543	eval-rmse:3.06059
[212]	train-rmse:1.42334	eval-rmse:3.06065
[213]	train-rmse:1.41817	eval-rmse:3.05900
[214]	train-rmse:1.41596	eval-rmse:3.05799
[215]	train-rmse:1.41360	eval-rmse:3.05797
[216]	train-rmse:1.40930	eval-rmse:3.05722
[217]	train-rmse:1.40445	eval-rmse:3.05697
[218]	train-rmse:1.40115	eval-rmse:3.05636
[219]	train-rmse:1.39421	eval-rmse:3.05551
[220]	train-rmse:1.38875	eval-rmse:3.05327
[221]	train-rmse:1.38384	eval-rmse:3.04960
[222]	train-rmse:1.38080	eval-rmse:3.04903
[223]	train-rmse:1.37686	eval-rmse:3.04862
[224]	train-rmse:1.37096	eval-rmse:3.04503
[225]	train-rmse:1.36286	eval-rmse:3.04279
[226]	train-rmse:1.35949	eval-rmse:3.04185
[227]	train-rmse:1.35558	eval-rmse:3.04145
[228]	train-rmse:1.35131	eval-rmse:3.04203
[229]	train-rmse:1.34516	eval-rmse:3.04092
```

```
[230]	train-rmse:1.34274	eval-rmse:3.03997
[231]	train-rmse:1.33953	eval-rmse:3.03850
[232]	train-rmse:1.33793	eval-rmse:3.03786
[233]	train-rmse:1.33436	eval-rmse:3.03646
[234]	train-rmse:1.33240	eval-rmse:3.03610
[235]	train-rmse:1.33022	eval-rmse:3.03407
[236]	train-rmse:1.32469	eval-rmse:3.03322
[237]	train-rmse:1.32311	eval-rmse:3.03296
[238]	train-rmse:1.31864	eval-rmse:3.03245
[239]	train-rmse:1.31302	eval-rmse:3.02920
[240]	train-rmse:1.30969	eval-rmse:3.02844
[241]	train-rmse:1.30593	eval-rmse:3.02779
[242]	train-rmse:1.30245	eval-rmse:3.02605
[243]	train-rmse:1.29775	eval-rmse:3.02442
[244]	train-rmse:1.29587	eval-rmse:3.02456
[245]	train-rmse:1.29428	eval-rmse:3.02297
[246]	train-rmse:1.28837	eval-rmse:3.02094
[247]	train-rmse:1.28690	eval-rmse:3.02009
[248]	train-rmse:1.28315	eval-rmse:3.01808
[249]	train-rmse:1.28239	eval-rmse:3.01750
[250]	train-rmse:1.28078	eval-rmse:3.01650
[251]	train-rmse:1.27567	eval-rmse:3.01570
[252]	train-rmse:1.27140	eval-rmse:3.01243
[253]	train-rmse:1.26505	eval-rmse:3.01414
[254]	train-rmse:1.26195	eval-rmse:3.01413
[255]	train-rmse:1.25620	eval-rmse:3.01366
[256]	train-rmse:1.25187	eval-rmse:3.01262
[257]	train-rmse:1.24852	eval-rmse:3.01145
[258]	train-rmse:1.24424	eval-rmse:3.01031
[259]	train-rmse:1.24166	eval-rmse:3.01089
[260]	train-rmse:1.23715	eval-rmse:3.01023
[261]	train-rmse:1.23299	eval-rmse:3.01015
[262]	train-rmse:1.22911	eval-rmse:3.00797
[263]	train-rmse:1.22513	eval-rmse:3.00764
[264]	train-rmse:1.21975	eval-rmse:3.00533
[265]	train-rmse:1.21681	eval-rmse:3.00587
[266]	train-rmse:1.21412	eval-rmse:3.00397
[267]	train-rmse:1.20859	eval-rmse:3.00246
[268]	train-rmse:1.20394	eval-rmse:3.00028
[269]	train-rmse:1.19868	eval-rmse:2.99932
[270]	train-rmse:1.19376	eval-rmse:2.99996
[271]	train-rmse:1.19032	eval-rmse:2.99969
[272]	train-rmse:1.18534	eval-rmse:2.99991
[273]	train-rmse:1.17927	eval-rmse:2.99793
[274]	train-rmse:1.17378	eval-rmse:2.99829
[275]	train-rmse:1.17019	eval-rmse:2.99817
[276]	train-rmse:1.16767	eval-rmse:2.99725
[277]	train-rmse:1.16560	eval-rmse:2.99764
[278]	train-rmse:1.16171	eval-rmse:2.99646
```

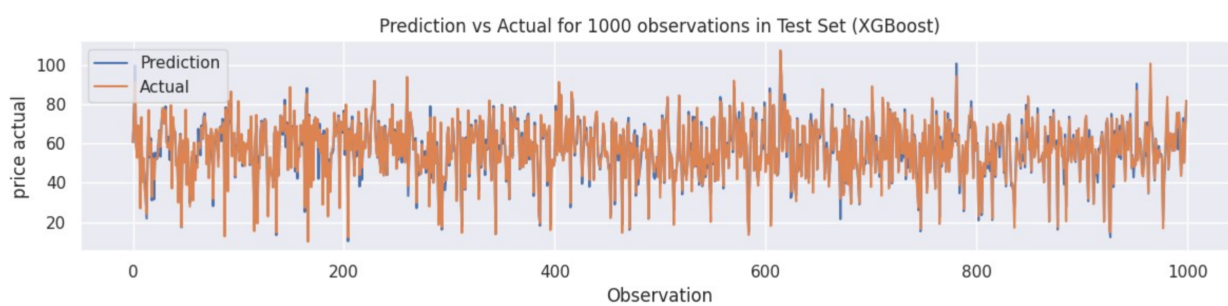
```
[279]	train-rmse:1.15553	eval-rmse:2.99629
[280]	train-rmse:1.15040	eval-rmse:2.99716
[281]	train-rmse:1.14905	eval-rmse:2.99756
[282]	train-rmse:1.14439	eval-rmse:2.99494
[283]	train-rmse:1.14407	eval-rmse:2.99467
[284]	train-rmse:1.14076	eval-rmse:2.99483
[285]	train-rmse:1.13596	eval-rmse:2.99381
[286]	train-rmse:1.13394	eval-rmse:2.99378
[287]	train-rmse:1.12980	eval-rmse:2.99223
[288]	train-rmse:1.12492	eval-rmse:2.99041
[289]	train-rmse:1.12161	eval-rmse:2.99068
[290]	train-rmse:1.11770	eval-rmse:2.98894
[291]	train-rmse:1.11560	eval-rmse:2.98799
[292]	train-rmse:1.11225	eval-rmse:2.98809
[293]	train-rmse:1.11000	eval-rmse:2.98895
[294]	train-rmse:1.10534	eval-rmse:2.98842
[295]	train-rmse:1.10242	eval-rmse:2.98723
[296]	train-rmse:1.09868	eval-rmse:2.98574
[297]	train-rmse:1.09623	eval-rmse:2.98567
[298]	train-rmse:1.09368	eval-rmse:2.98499
[299]	train-rmse:1.09059	eval-rmse:2.98518
[300]	train-rmse:1.08543	eval-rmse:2.98399
[301]	train-rmse:1.08215	eval-rmse:2.98375
[302]	train-rmse:1.07736	eval-rmse:2.98423
[303]	train-rmse:1.07340	eval-rmse:2.98393
[304]	train-rmse:1.06878	eval-rmse:2.98347
[305]	train-rmse:1.06582	eval-rmse:2.98255
[306]	train-rmse:1.06267	eval-rmse:2.98245
[307]	train-rmse:1.05928	eval-rmse:2.98142
[308]	train-rmse:1.05696	eval-rmse:2.98129
[309]	train-rmse:1.05234	eval-rmse:2.98152
[310]	train-rmse:1.04813	eval-rmse:2.98071
[311]	train-rmse:1.04370	eval-rmse:2.97983
[312]	train-rmse:1.04045	eval-rmse:2.97819
[313]	train-rmse:1.03860	eval-rmse:2.97803
[314]	train-rmse:1.03672	eval-rmse:2.97753
[315]	train-rmse:1.03427	eval-rmse:2.97703
[316]	train-rmse:1.03156	eval-rmse:2.97660
[317]	train-rmse:1.02754	eval-rmse:2.97634
[318]	train-rmse:1.02507	eval-rmse:2.97537
[319]	train-rmse:1.02202	eval-rmse:2.97560
[320]	train-rmse:1.01804	eval-rmse:2.97500
[321]	train-rmse:1.01414	eval-rmse:2.97551
[322]	train-rmse:1.00884	eval-rmse:2.97316
[323]	train-rmse:1.00431	eval-rmse:2.97164
[324]	train-rmse:1.00166	eval-rmse:2.97114
[325]	train-rmse:0.99869	eval-rmse:2.97103
[326]	train-rmse:0.99674	eval-rmse:2.97087
[327]	train-rmse:0.99588	eval-rmse:2.97046
```

```
[328]	train-rmse:0.99381	eval-rmse:2.97082
[329]	train-rmse:0.98923	eval-rmse:2.96862
[330]	train-rmse:0.98646	eval-rmse:2.96710
[331]	train-rmse:0.98414	eval-rmse:2.96715
[332]	train-rmse:0.98017	eval-rmse:2.96755
[333]	train-rmse:0.97621	eval-rmse:2.96755
[334]	train-rmse:0.97345	eval-rmse:2.96759
[335]	train-rmse:0.97178	eval-rmse:2.96635
[336]	train-rmse:0.96859	eval-rmse:2.96613
[337]	train-rmse:0.96649	eval-rmse:2.96553
[338]	train-rmse:0.96578	eval-rmse:2.96570
[339]	train-rmse:0.96439	eval-rmse:2.96576
[340]	train-rmse:0.96110	eval-rmse:2.96424
[341]	train-rmse:0.95753	eval-rmse:2.96374
[342]	train-rmse:0.95434	eval-rmse:2.96304
[343]	train-rmse:0.95214	eval-rmse:2.96333
[344]	train-rmse:0.94971	eval-rmse:2.96312
[345]	train-rmse:0.94822	eval-rmse:2.96337
[346]	train-rmse:0.94428	eval-rmse:2.96128
[347]	train-rmse:0.94220	eval-rmse:2.96075
[348]	train-rmse:0.93848	eval-rmse:2.95989
[349]	train-rmse:0.93561	eval-rmse:2.95873
[350]	train-rmse:0.93266	eval-rmse:2.95773
[351]	train-rmse:0.93160	eval-rmse:2.95756
[352]	train-rmse:0.92877	eval-rmse:2.95745
[353]	train-rmse:0.92538	eval-rmse:2.95774
[354]	train-rmse:0.92304	eval-rmse:2.95727
[355]	train-rmse:0.91922	eval-rmse:2.95653
[356]	train-rmse:0.91778	eval-rmse:2.95606
[357]	train-rmse:0.91368	eval-rmse:2.95289
[358]	train-rmse:0.90942	eval-rmse:2.95249
[359]	train-rmse:0.90683	eval-rmse:2.95241
[360]	train-rmse:0.90359	eval-rmse:2.95222
[361]	train-rmse:0.90042	eval-rmse:2.95248
[362]	train-rmse:0.89813	eval-rmse:2.95128
[363]	train-rmse:0.89400	eval-rmse:2.95030
[364]	train-rmse:0.89261	eval-rmse:2.94973
[365]	train-rmse:0.88955	eval-rmse:2.94948
[366]	train-rmse:0.88713	eval-rmse:2.94937
[367]	train-rmse:0.88376	eval-rmse:2.94896
[368]	train-rmse:0.88114	eval-rmse:2.94839
[369]	train-rmse:0.87906	eval-rmse:2.94771
[370]	train-rmse:0.87752	eval-rmse:2.94779
[371]	train-rmse:0.87605	eval-rmse:2.94719
[372]	train-rmse:0.87285	eval-rmse:2.94744
[373]	train-rmse:0.87096	eval-rmse:2.94633
[374]	train-rmse:0.86938	eval-rmse:2.94602
[375]	train-rmse:0.86667	eval-rmse:2.94577
[376]	train-rmse:0.86465	eval-rmse:2.94573
```

```
[377]	train-rmse:0.86299	eval-rmse:2.94571
[378]	train-rmse:0.85971	eval-rmse:2.94609
[379]	train-rmse:0.85748	eval-rmse:2.94536
[380]	train-rmse:0.85591	eval-rmse:2.94545
[381]	train-rmse:0.85506	eval-rmse:2.94501
[382]	train-rmse:0.85233	eval-rmse:2.94471
[383]	train-rmse:0.84964	eval-rmse:2.94618
[384]	train-rmse:0.84592	eval-rmse:2.94541
[385]	train-rmse:0.84397	eval-rmse:2.94472
[386]	train-rmse:0.84195	eval-rmse:2.94464
[387]	train-rmse:0.83851	eval-rmse:2.94295
[388]	train-rmse:0.83649	eval-rmse:2.94336
[389]	train-rmse:0.83281	eval-rmse:2.94332
[390]	train-rmse:0.82917	eval-rmse:2.94236
[391]	train-rmse:0.82718	eval-rmse:2.94233
[392]	train-rmse:0.82435	eval-rmse:2.94131
[393]	train-rmse:0.82213	eval-rmse:2.94039
[394]	train-rmse:0.81926	eval-rmse:2.93896
[395]	train-rmse:0.81788	eval-rmse:2.93918
[396]	train-rmse:0.81618	eval-rmse:2.93876
[397]	train-rmse:0.81417	eval-rmse:2.93791
[398]	train-rmse:0.81315	eval-rmse:2.93761
[399]	train-rmse:0.81045	eval-rmse:2.93706
[400]	train-rmse:0.80779	eval-rmse:2.93640
[401]	train-rmse:0.80502	eval-rmse:2.93576
[402]	train-rmse:0.80294	eval-rmse:2.93516
[403]	train-rmse:0.80073	eval-rmse:2.93543
[404]	train-rmse:0.79780	eval-rmse:2.93478
[405]	train-rmse:0.79553	eval-rmse:2.93525
[406]	train-rmse:0.79358	eval-rmse:2.93462
[407]	train-rmse:0.79091	eval-rmse:2.93424
[408]	train-rmse:0.79043	eval-rmse:2.93400
[409]	train-rmse:0.78876	eval-rmse:2.93424
[410]	train-rmse:0.78755	eval-rmse:2.93412
[411]	train-rmse:0.78599	eval-rmse:2.93433
[412]	train-rmse:0.78363	eval-rmse:2.93374
[413]	train-rmse:0.78037	eval-rmse:2.93306
[414]	train-rmse:0.77928	eval-rmse:2.93328
[415]	train-rmse:0.77820	eval-rmse:2.93321
[416]	train-rmse:0.77631	eval-rmse:2.93345
[417]	train-rmse:0.77452	eval-rmse:2.93364
[418]	train-rmse:0.77224	eval-rmse:2.93382
[419]	train-rmse:0.77204	eval-rmse:2.93380
[420]	train-rmse:0.76965	eval-rmse:2.93279
[421]	train-rmse:0.76804	eval-rmse:2.93273
[422]	train-rmse:0.76715	eval-rmse:2.93259
[423]	train-rmse:0.76586	eval-rmse:2.93213
[424]	train-rmse:0.76447	eval-rmse:2.93169
[425]	train-rmse:0.76357	eval-rmse:2.93139
```

```
[426]	train-rmse:0.76090	eval-rmse:2.93079
[427]	train-rmse:0.76029	eval-rmse:2.93090
[428]	train-rmse:0.75820	eval-rmse:2.92996
[429]	train-rmse:0.75611	eval-rmse:2.93000
[430]	train-rmse:0.75346	eval-rmse:2.92931
[431]	train-rmse:0.75124	eval-rmse:2.92871
[432]	train-rmse:0.74926	eval-rmse:2.92811
[433]	train-rmse:0.74761	eval-rmse:2.92799
[434]	train-rmse:0.74478	eval-rmse:2.92806
[435]	train-rmse:0.74240	eval-rmse:2.92699
[436]	train-rmse:0.74051	eval-rmse:2.92704
[437]	train-rmse:0.73803	eval-rmse:2.92641
[438]	train-rmse:0.73575	eval-rmse:2.92537
[439]	train-rmse:0.73413	eval-rmse:2.92538
[440]	train-rmse:0.73254	eval-rmse:2.92488
[441]	train-rmse:0.72935	eval-rmse:2.92459
[442]	train-rmse:0.72792	eval-rmse:2.92422
[443]	train-rmse:0.72593	eval-rmse:2.92380
[444]	train-rmse:0.72477	eval-rmse:2.92271
[445]	train-rmse:0.72389	eval-rmse:2.92252
[446]	train-rmse:0.72141	eval-rmse:2.92246
[447]	train-rmse:0.72010	eval-rmse:2.92260
[448]	train-rmse:0.71782	eval-rmse:2.92224
[449]	train-rmse:0.71558	eval-rmse:2.92209
[450]	train-rmse:0.71367	eval-rmse:2.92167
[451]	train-rmse:0.71170	eval-rmse:2.92117
[452]	train-rmse:0.70975	eval-rmse:2.92157
[453]	train-rmse:0.70847	eval-rmse:2.92147
[454]	train-rmse:0.70664	eval-rmse:2.92121
[455]	train-rmse:0.70481	eval-rmse:2.92148
[456]	train-rmse:0.70230	eval-rmse:2.92123
[457]	train-rmse:0.70144	eval-rmse:2.92071
[458]	train-rmse:0.70035	eval-rmse:2.91999
[459]	train-rmse:0.69920	eval-rmse:2.91920
[460]	train-rmse:0.69709	eval-rmse:2.91818
[461]	train-rmse:0.69553	eval-rmse:2.91805
[462]	train-rmse:0.69284	eval-rmse:2.91794
[463]	train-rmse:0.69120	eval-rmse:2.91807
[464]	train-rmse:0.68898	eval-rmse:2.91789
[465]	train-rmse:0.68807	eval-rmse:2.91790
[466]	train-rmse:0.68583	eval-rmse:2.91790
[467]	train-rmse:0.68363	eval-rmse:2.91774
[468]	train-rmse:0.68101	eval-rmse:2.91686
[469]	train-rmse:0.67943	eval-rmse:2.91640
[470]	train-rmse:0.67705	eval-rmse:2.91635
[471]	train-rmse:0.67508	eval-rmse:2.91629
[472]	train-rmse:0.67393	eval-rmse:2.91601
[473]	train-rmse:0.67168	eval-rmse:2.91551
[474]	train-rmse:0.66940	eval-rmse:2.91495
```

```
[475] train-rmse:0.66754    eval-rmse:2.91464
[476] train-rmse:0.66614    eval-rmse:2.91376
[477] train-rmse:0.66401    eval-rmse:2.91313
[478] train-rmse:0.66176    eval-rmse:2.91296
[479] train-rmse:0.66110    eval-rmse:2.91318
[480] train-rmse:0.65895    eval-rmse:2.91257
[481] train-rmse:0.65762    eval-rmse:2.91236
[482] train-rmse:0.65611    eval-rmse:2.91235
[483] train-rmse:0.65415    eval-rmse:2.91219
[484] train-rmse:0.65238    eval-rmse:2.91182
[485] train-rmse:0.65021    eval-rmse:2.91169
[486] train-rmse:0.64942    eval-rmse:2.91172
[487] train-rmse:0.64727    eval-rmse:2.91106
[488] train-rmse:0.64477    eval-rmse:2.91041
[489] train-rmse:0.64360    eval-rmse:2.91016
[490] train-rmse:0.64232    eval-rmse:2.90971
[491] train-rmse:0.64166    eval-rmse:2.90959
[492] train-rmse:0.63998    eval-rmse:2.90933
[493] train-rmse:0.63953    eval-rmse:2.90914
[494] train-rmse:0.63821    eval-rmse:2.90917
[495] train-rmse:0.63626    eval-rmse:2.90963
[496] train-rmse:0.63600    eval-rmse:2.90956
[497] train-rmse:0.63452    eval-rmse:2.91008
[498] train-rmse:0.63331    eval-rmse:2.90986
[499] train-rmse:0.63169    eval-rmse:2.90935
Mean Absolute Error (MAE): 2.124053182436049
Root Mean Squared Error (RMSE): 3.067974857555092
Mean Absolute Percentage Error (MAPE): 4.25%
Accuracy: 96.30%
```



```
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
import xgboost as xgb
import pandas as pd
import numpy as np

# Assuming 'df_weather_energy' is already loaded as a DataFrame
# Step 1: Prepare features (X) and target (y)
X = df_weather_energy.drop(['total load actual'], axis=1)
y = df_weather_energy['total load actual']

# Step 2: Train-Test Split
X_train, X_temp, y_train, y_temp = train_test_split(X, y,
test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp,
test_size=0.5, random_state=42)

# Step 3: Convert Data to DMatrix
dtrain = xgb.DMatrix(X_train, label=y_train)
dval = xgb.DMatrix(X_val, label=y_val)
dtest = xgb.DMatrix(X_test, label=y_test)

# Step 4: Define Parameters for XGBoost
params = {
    'objective': 'reg:squarederror',  # Regression objective
    'random_state': 42,
    'eval_metric': 'rmse',  # Metric to evaluate during training
}

# Step 5: Train XGBoost Model with Early Stopping
evals = [(dtrain, 'train'), (dval, 'eval')]
evals_result = {}  # To store the evaluation results

xgboost_model = xgb.train(
    params=params,
    dtrain=dtrain,
    num_boost_round=500,  # Maximum number of boosting rounds
    early_stopping_rounds=10,  # Stop training if no improvement after
10 rounds
    evals=evals,
    evals_result=evals_result,  # Store evaluation results here
    verbose_eval=True
)

# Step 6: Make Predictions
y_pred = xgboost_model.predict(dtest)

# Step 7: Calculate Metrics
mae = mean_absolute_error(y_test, y_pred)
```

```python
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mape = np.mean(np.abs((y_test - y_pred) / y_test)) * 100  # Mean
Absolute Percentage Error

# Calculate Accuracy as 1 - (Normalized MAE)
mean_actual = np.mean(y_test)
accuracy = (1 - (mae / mean_actual)) * 100

# Print Results
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Percentage Error (MAPE): {mape:.2f}%")
print(f"Accuracy: {accuracy:.2f}%")

# Plot Results using the plot_results function
def plot_results(y_pred, y_test, evals_result, model_name):
    fig, ax = plt.subplots(2, 1, figsize=(12, 6))

    # Plotting prediction vs actual for the first 1000 observations
    ax[0].plot(y_pred[:1000], label='Prediction')
    ax[0].plot(y_test[:1000].values, label='Actual')
    ax[0].legend(loc='upper left')
    ax[0].set_title(f'Prediction vs Actual for 1000 observations in
Test Set ({model_name})')
    ax[0].set_xlabel('Observation')
    ax[0].set_ylabel('Total Load')

    # Extract training and validation RMSE from evals_result
    train_rmse = evals_result['train']['rmse']
    val_rmse = evals_result['eval']['rmse']

    # Plot RMSE for training and validation
    ax[1].plot(train_rmse, label='Training RMSE')
    ax[1].plot(val_rmse, label='Validation RMSE')

    ax[1].legend()
    ax[1].set_title(f'Training and Validation RMSE ({model_name})')
    ax[1].set_xlabel('Iteration')
    ax[1].set_ylabel('RMSE')

    fig.tight_layout()
    plt.show()

# Call the function to plot results
plot_results(y_pred, y_test, evals_result, 'XGBoost')
```

```
[0]	train-rmse:3223.45802	eval-rmse:3247.51617
[1]	train-rmse:2280.16447	eval-rmse:2302.82134
[2]	train-rmse:1627.23376	eval-rmse:1649.82824
[3]	train-rmse:1180.38659	eval-rmse:1204.05018
```

```
[4]	train-rmse:879.86241	eval-rmse:906.13044
[5]	train-rmse:683.60161	eval-rmse:712.63274
[6]	train-rmse:560.72581	eval-rmse:592.44403
[7]	train-rmse:486.45202	eval-rmse:520.86192
[8]	train-rmse:442.20571	eval-rmse:479.34657
[9]	train-rmse:415.27903	eval-rmse:454.56851
[10]	train-rmse:398.29575	eval-rmse:437.67499
[11]	train-rmse:386.37849	eval-rmse:426.42089
[12]	train-rmse:376.95510	eval-rmse:418.64664
[13]	train-rmse:368.77415	eval-rmse:411.59055
[14]	train-rmse:361.27884	eval-rmse:405.95663
[15]	train-rmse:355.99736	eval-rmse:402.29231
[16]	train-rmse:350.14027	eval-rmse:396.42569
[17]	train-rmse:346.25532	eval-rmse:394.40096
[18]	train-rmse:341.48924	eval-rmse:390.65301
[19]	train-rmse:337.67796	eval-rmse:387.01968
[20]	train-rmse:334.32903	eval-rmse:384.12726
[21]	train-rmse:330.43209	eval-rmse:379.58460
[22]	train-rmse:326.44989	eval-rmse:377.31836
[23]	train-rmse:322.63927	eval-rmse:375.08381
[24]	train-rmse:320.34407	eval-rmse:373.92342
[25]	train-rmse:316.55530	eval-rmse:371.25287
[26]	train-rmse:315.04865	eval-rmse:370.53741
[27]	train-rmse:310.86390	eval-rmse:368.04975
[28]	train-rmse:307.83893	eval-rmse:366.03314
[29]	train-rmse:305.05817	eval-rmse:363.49304
[30]	train-rmse:302.49913	eval-rmse:361.57088
[31]	train-rmse:299.96419	eval-rmse:360.22626
[32]	train-rmse:298.00610	eval-rmse:358.63855
[33]	train-rmse:295.38414	eval-rmse:355.98190
[34]	train-rmse:293.16310	eval-rmse:354.27693
[35]	train-rmse:290.69570	eval-rmse:352.15991
[36]	train-rmse:287.88470	eval-rmse:349.92237
[37]	train-rmse:285.62469	eval-rmse:348.22948
[38]	train-rmse:283.76141	eval-rmse:346.94857
[39]	train-rmse:281.69572	eval-rmse:345.91711
[40]	train-rmse:279.34233	eval-rmse:344.97726
[41]	train-rmse:277.09168	eval-rmse:344.06711
[42]	train-rmse:274.45140	eval-rmse:342.67339
[43]	train-rmse:271.99143	eval-rmse:341.78846
[44]	train-rmse:270.62794	eval-rmse:341.04953
[45]	train-rmse:268.38909	eval-rmse:339.82841
[46]	train-rmse:267.49667	eval-rmse:339.24613
[47]	train-rmse:265.73578	eval-rmse:338.20682
[48]	train-rmse:263.92930	eval-rmse:337.05187
[49]	train-rmse:262.09739	eval-rmse:336.59458
[50]	train-rmse:260.34705	eval-rmse:335.61718
[51]	train-rmse:259.09789	eval-rmse:334.83762
[52]	train-rmse:256.70379	eval-rmse:333.82348
```

```
[53]	train-rmse:254.66352	eval-rmse:333.18993
[54]	train-rmse:253.02151	eval-rmse:332.15605
[55]	train-rmse:251.53814	eval-rmse:331.00930
[56]	train-rmse:249.20678	eval-rmse:329.18847
[57]	train-rmse:247.24644	eval-rmse:328.36958
[58]	train-rmse:246.07892	eval-rmse:327.56316
[59]	train-rmse:243.85230	eval-rmse:326.68492
[60]	train-rmse:241.51642	eval-rmse:325.39876
[61]	train-rmse:240.30438	eval-rmse:325.06884
[62]	train-rmse:239.97924	eval-rmse:325.01255
[63]	train-rmse:238.76363	eval-rmse:324.05233
[64]	train-rmse:237.27469	eval-rmse:323.22162
[65]	train-rmse:236.04262	eval-rmse:322.73509
[66]	train-rmse:234.91258	eval-rmse:322.02842
[67]	train-rmse:233.31813	eval-rmse:321.10199
[68]	train-rmse:231.51256	eval-rmse:320.32465
[69]	train-rmse:230.24194	eval-rmse:319.52275
[70]	train-rmse:228.89590	eval-rmse:318.84657
[71]	train-rmse:227.89987	eval-rmse:318.46617
[72]	train-rmse:226.42091	eval-rmse:317.60857
[73]	train-rmse:224.99314	eval-rmse:316.97440
[74]	train-rmse:224.23121	eval-rmse:316.90428
[75]	train-rmse:222.64241	eval-rmse:315.91318
[76]	train-rmse:221.74452	eval-rmse:315.58945
[77]	train-rmse:220.50470	eval-rmse:314.76526
[78]	train-rmse:219.50465	eval-rmse:314.19790
[79]	train-rmse:218.24037	eval-rmse:313.53473
[80]	train-rmse:217.48076	eval-rmse:313.32533
[81]	train-rmse:215.85348	eval-rmse:312.29923
[82]	train-rmse:214.87251	eval-rmse:312.05040
[83]	train-rmse:214.20848	eval-rmse:311.83342
[84]	train-rmse:213.75122	eval-rmse:311.68815
[85]	train-rmse:212.26016	eval-rmse:311.20887
[86]	train-rmse:211.03939	eval-rmse:310.92337
[87]	train-rmse:210.17760	eval-rmse:310.55841
[88]	train-rmse:209.56694	eval-rmse:310.40097
[89]	train-rmse:209.03689	eval-rmse:310.21633
[90]	train-rmse:208.33951	eval-rmse:309.95642
[91]	train-rmse:207.48732	eval-rmse:309.68739
[92]	train-rmse:205.94881	eval-rmse:308.86746
[93]	train-rmse:205.09422	eval-rmse:308.40228
[94]	train-rmse:203.82815	eval-rmse:307.84083
[95]	train-rmse:202.86984	eval-rmse:307.48715
[96]	train-rmse:202.09819	eval-rmse:307.19847
[97]	train-rmse:200.72338	eval-rmse:306.48979
[98]	train-rmse:199.70640	eval-rmse:306.04835
[99]	train-rmse:198.50090	eval-rmse:305.34014
[100]	train-rmse:197.58081	eval-rmse:305.22192
[101]	train-rmse:196.55189	eval-rmse:304.48591
```

```
[102]	train-rmse:195.56340	eval-rmse:304.05271
[103]	train-rmse:195.02162	eval-rmse:303.75581
[104]	train-rmse:194.29210	eval-rmse:303.44303
[105]	train-rmse:193.69860	eval-rmse:303.20166
[106]	train-rmse:192.84628	eval-rmse:302.87531
[107]	train-rmse:192.01813	eval-rmse:302.72371
[108]	train-rmse:190.95316	eval-rmse:302.08483
[109]	train-rmse:189.85129	eval-rmse:301.46702
[110]	train-rmse:189.37432	eval-rmse:301.19305
[111]	train-rmse:188.67012	eval-rmse:301.10548
[112]	train-rmse:187.73792	eval-rmse:300.52070
[113]	train-rmse:187.05957	eval-rmse:300.25409
[114]	train-rmse:186.05944	eval-rmse:299.96178
[115]	train-rmse:184.90429	eval-rmse:299.65009
[116]	train-rmse:184.24610	eval-rmse:299.68154
[117]	train-rmse:183.42392	eval-rmse:299.26088
[118]	train-rmse:182.80316	eval-rmse:298.73278
[119]	train-rmse:181.72619	eval-rmse:298.47628
[120]	train-rmse:181.18014	eval-rmse:298.43631
[121]	train-rmse:179.98692	eval-rmse:297.84965
[122]	train-rmse:179.27006	eval-rmse:297.48281
[123]	train-rmse:178.21927	eval-rmse:297.02209
[124]	train-rmse:177.60691	eval-rmse:296.86864
[125]	train-rmse:176.89527	eval-rmse:296.79588
[126]	train-rmse:176.26115	eval-rmse:296.61736
[127]	train-rmse:175.16103	eval-rmse:296.17871
[128]	train-rmse:174.55099	eval-rmse:295.97999
[129]	train-rmse:173.82134	eval-rmse:295.75394
[130]	train-rmse:173.18966	eval-rmse:295.57321
[131]	train-rmse:172.22835	eval-rmse:295.12982
[132]	train-rmse:171.56627	eval-rmse:294.99578
[133]	train-rmse:170.76321	eval-rmse:294.43152
[134]	train-rmse:170.00900	eval-rmse:294.16723
[135]	train-rmse:169.56228	eval-rmse:294.01125
[136]	train-rmse:168.65604	eval-rmse:293.59036
[137]	train-rmse:168.12747	eval-rmse:293.11453
[138]	train-rmse:167.12981	eval-rmse:292.65202
[139]	train-rmse:166.63437	eval-rmse:292.39134
[140]	train-rmse:166.17592	eval-rmse:292.31037
[141]	train-rmse:165.64342	eval-rmse:291.89953
[142]	train-rmse:164.78046	eval-rmse:291.56721
[143]	train-rmse:164.13178	eval-rmse:291.13305
[144]	train-rmse:163.57274	eval-rmse:290.74291
[145]	train-rmse:162.90890	eval-rmse:290.49952
[146]	train-rmse:162.12088	eval-rmse:290.15305
[147]	train-rmse:161.56819	eval-rmse:289.69499
[148]	train-rmse:160.81289	eval-rmse:289.23995
[149]	train-rmse:160.55283	eval-rmse:289.16233
[150]	train-rmse:159.57585	eval-rmse:288.61040
```

```
[151] train-rmse:158.90931  eval-rmse:288.40035
[152] train-rmse:158.23858  eval-rmse:288.00974
[153] train-rmse:157.41004  eval-rmse:287.81162
[154] train-rmse:156.95968  eval-rmse:287.44138
[155] train-rmse:156.67902  eval-rmse:287.22790
[156] train-rmse:155.97504  eval-rmse:287.08471
[157] train-rmse:155.36836  eval-rmse:286.79603
[158] train-rmse:154.77151  eval-rmse:286.20203
[159] train-rmse:154.02272  eval-rmse:285.99222
[160] train-rmse:153.73455  eval-rmse:285.83035
[161] train-rmse:153.11894  eval-rmse:285.67774
[162] train-rmse:152.65837  eval-rmse:285.36191
[163] train-rmse:151.90450  eval-rmse:285.03264
[164] train-rmse:151.11442  eval-rmse:284.91459
[165] train-rmse:150.23118  eval-rmse:284.39505
[166] train-rmse:149.87124  eval-rmse:284.47405
[167] train-rmse:149.66708  eval-rmse:284.42387
[168] train-rmse:149.21613  eval-rmse:284.38542
[169] train-rmse:148.65983  eval-rmse:284.08019
[170] train-rmse:148.09285  eval-rmse:283.75480
[171] train-rmse:147.58153  eval-rmse:283.62943
[172] train-rmse:146.83234  eval-rmse:283.61864
[173] train-rmse:146.24001  eval-rmse:283.30970
[174] train-rmse:145.94281  eval-rmse:283.23784
[175] train-rmse:145.31354  eval-rmse:282.99425
[176] train-rmse:145.05061  eval-rmse:282.89862
[177] train-rmse:144.63694  eval-rmse:282.66819
[178] train-rmse:144.08135  eval-rmse:282.58122
[179] train-rmse:143.72104  eval-rmse:282.53932
[180] train-rmse:143.22858  eval-rmse:282.32482
[181] train-rmse:142.71879  eval-rmse:282.32607
[182] train-rmse:142.14785  eval-rmse:282.31705
[183] train-rmse:141.61597  eval-rmse:282.30576
[184] train-rmse:141.19133  eval-rmse:282.16159
[185] train-rmse:140.66257  eval-rmse:281.98923
[186] train-rmse:140.14672  eval-rmse:281.97518
[187] train-rmse:139.77340  eval-rmse:281.93502
[188] train-rmse:139.12836  eval-rmse:281.66346
[189] train-rmse:138.68388  eval-rmse:281.60527
[190] train-rmse:138.00507  eval-rmse:281.34714
[191] train-rmse:137.80337  eval-rmse:281.39928
[192] train-rmse:137.34728  eval-rmse:281.26261
[193] train-rmse:136.77428  eval-rmse:281.34427
[194] train-rmse:136.17411  eval-rmse:281.24353
[195] train-rmse:135.78164  eval-rmse:281.08512
[196] train-rmse:135.33230  eval-rmse:280.86888
[197] train-rmse:134.93047  eval-rmse:280.77227
[198] train-rmse:134.53445  eval-rmse:280.61850
[199] train-rmse:134.37156  eval-rmse:280.61041
```

```
[200] train-rmse:134.14675  eval-rmse:280.54170
[201] train-rmse:133.71261  eval-rmse:280.34768
[202] train-rmse:133.21737  eval-rmse:280.25667
[203] train-rmse:132.71899  eval-rmse:280.09901
[204] train-rmse:132.09755  eval-rmse:279.98000
[205] train-rmse:131.41099  eval-rmse:279.68653
[206] train-rmse:130.94094  eval-rmse:279.64623
[207] train-rmse:130.42684  eval-rmse:279.56832
[208] train-rmse:130.10825  eval-rmse:279.56881
[209] train-rmse:129.88064  eval-rmse:279.43022
[210] train-rmse:129.35487  eval-rmse:279.12969
[211] train-rmse:129.07443  eval-rmse:279.08102
[212] train-rmse:128.67728  eval-rmse:278.81270
[213] train-rmse:128.36705  eval-rmse:278.79908
[214] train-rmse:127.70575  eval-rmse:278.53614
[215] train-rmse:127.42405  eval-rmse:278.49167
[216] train-rmse:127.04798  eval-rmse:278.31131
[217] train-rmse:126.60925  eval-rmse:278.27445
[218] train-rmse:126.15341  eval-rmse:278.14244
[219] train-rmse:125.75951  eval-rmse:278.09651
[220] train-rmse:125.27735  eval-rmse:277.81915
[221] train-rmse:125.02284  eval-rmse:277.58675
[222] train-rmse:124.62354  eval-rmse:277.38067
[223] train-rmse:124.30036  eval-rmse:277.32634
[224] train-rmse:124.04165  eval-rmse:277.25914
[225] train-rmse:123.63271  eval-rmse:277.14802
[226] train-rmse:123.24930  eval-rmse:276.89723
[227] train-rmse:123.06062  eval-rmse:276.92778
[228] train-rmse:122.44173  eval-rmse:276.72425
[229] train-rmse:122.09310  eval-rmse:276.55676
[230] train-rmse:121.56630  eval-rmse:276.42925
[231] train-rmse:120.98734  eval-rmse:276.24173
[232] train-rmse:120.52485  eval-rmse:276.13723
[233] train-rmse:120.15163  eval-rmse:275.99325
[234] train-rmse:119.82695  eval-rmse:275.83519
[235] train-rmse:119.39846  eval-rmse:275.72256
[236] train-rmse:119.05542  eval-rmse:275.55808
[237] train-rmse:118.51027  eval-rmse:275.55939
[238] train-rmse:118.28119  eval-rmse:275.42950
[239] train-rmse:117.77911  eval-rmse:275.46751
[240] train-rmse:117.42709  eval-rmse:275.45248
[241] train-rmse:116.85284  eval-rmse:275.40472
[242] train-rmse:116.63223  eval-rmse:275.36526
[243] train-rmse:116.47122  eval-rmse:275.33363
[244] train-rmse:115.86108  eval-rmse:274.90667
[245] train-rmse:115.28402  eval-rmse:274.73099
[246] train-rmse:114.92690  eval-rmse:274.59979
[247] train-rmse:114.54724  eval-rmse:274.61286
[248] train-rmse:114.21582  eval-rmse:274.63445
```

```
[249]	train-rmse:113.93855	eval-rmse:274.56356
[250]	train-rmse:113.50096	eval-rmse:274.50925
[251]	train-rmse:113.18914	eval-rmse:274.17633
[252]	train-rmse:112.88660	eval-rmse:274.10413
[253]	train-rmse:112.64103	eval-rmse:274.00689
[254]	train-rmse:112.42942	eval-rmse:274.04078
[255]	train-rmse:112.00166	eval-rmse:274.08548
[256]	train-rmse:111.64843	eval-rmse:274.04097
[257]	train-rmse:111.11925	eval-rmse:274.04619
[258]	train-rmse:110.92135	eval-rmse:274.00196
[259]	train-rmse:110.41612	eval-rmse:273.83771
[260]	train-rmse:110.02162	eval-rmse:274.05587
[261]	train-rmse:109.67058	eval-rmse:274.01387
[262]	train-rmse:109.32374	eval-rmse:273.96974
[263]	train-rmse:109.02020	eval-rmse:273.82191
[264]	train-rmse:108.63990	eval-rmse:273.72882
[265]	train-rmse:108.24203	eval-rmse:273.65339
[266]	train-rmse:107.68474	eval-rmse:273.57554
[267]	train-rmse:107.34146	eval-rmse:273.50153
[268]	train-rmse:107.01698	eval-rmse:273.55570
[269]	train-rmse:106.58979	eval-rmse:273.55133
[270]	train-rmse:106.01960	eval-rmse:273.34600
[271]	train-rmse:105.74767	eval-rmse:273.27316
[272]	train-rmse:105.34368	eval-rmse:273.17813
[273]	train-rmse:104.95248	eval-rmse:273.03321
[274]	train-rmse:104.69556	eval-rmse:272.96279
[275]	train-rmse:104.41670	eval-rmse:272.88394
[276]	train-rmse:103.98793	eval-rmse:273.01192
[277]	train-rmse:103.48732	eval-rmse:272.88686
[278]	train-rmse:103.11562	eval-rmse:272.85811
[279]	train-rmse:103.00523	eval-rmse:272.85095
[280]	train-rmse:102.72313	eval-rmse:272.74940
[281]	train-rmse:102.29170	eval-rmse:272.93333
[282]	train-rmse:101.95560	eval-rmse:272.92593
[283]	train-rmse:101.76007	eval-rmse:272.91576
[284]	train-rmse:101.49626	eval-rmse:272.92764
[285]	train-rmse:101.18681	eval-rmse:272.92718
[286]	train-rmse:100.98566	eval-rmse:272.82626
[287]	train-rmse:100.53203	eval-rmse:272.58966
[288]	train-rmse:100.20615	eval-rmse:272.50649
[289]	train-rmse:99.96444	eval-rmse:272.46757
[290]	train-rmse:99.73293	eval-rmse:272.45412
[291]	train-rmse:99.27752	eval-rmse:271.99420
[292]	train-rmse:98.96367	eval-rmse:271.93557
[293]	train-rmse:98.64551	eval-rmse:271.78518
[294]	train-rmse:98.30396	eval-rmse:271.70232
[295]	train-rmse:97.92731	eval-rmse:271.66397
[296]	train-rmse:97.78525	eval-rmse:271.62875
[297]	train-rmse:97.56421	eval-rmse:271.53420
```

```
[298] train-rmse:97.25975   eval-rmse:271.48884
[299] train-rmse:97.02469   eval-rmse:271.46180
[300] train-rmse:96.66167   eval-rmse:271.23872
[301] train-rmse:96.54024   eval-rmse:271.31153
[302] train-rmse:96.30093   eval-rmse:271.20162
[303] train-rmse:95.97068   eval-rmse:271.25612
[304] train-rmse:95.65882   eval-rmse:271.24513
[305] train-rmse:95.33903   eval-rmse:271.15776
[306] train-rmse:94.95253   eval-rmse:271.03660
[307] train-rmse:94.61056   eval-rmse:271.01193
[308] train-rmse:94.39063   eval-rmse:270.98323
[309] train-rmse:94.06300   eval-rmse:270.96116
[310] train-rmse:93.82309   eval-rmse:270.90274
[311] train-rmse:93.49638   eval-rmse:270.88595
[312] train-rmse:93.30756   eval-rmse:270.90722
[313] train-rmse:93.17677   eval-rmse:270.87443
[314] train-rmse:93.01330   eval-rmse:270.87900
[315] train-rmse:92.78063   eval-rmse:270.82332
[316] train-rmse:92.71783   eval-rmse:270.79849
[317] train-rmse:92.63714   eval-rmse:270.76858
[318] train-rmse:92.29723   eval-rmse:270.75531
[319] train-rmse:92.08893   eval-rmse:270.75382
[320] train-rmse:91.78171   eval-rmse:270.63386
[321] train-rmse:91.44788   eval-rmse:270.62731
[322] train-rmse:91.18219   eval-rmse:270.53708
[323] train-rmse:90.93612   eval-rmse:270.52245
[324] train-rmse:90.59876   eval-rmse:270.38990
[325] train-rmse:90.38054   eval-rmse:270.28336
[326] train-rmse:90.18083   eval-rmse:270.28917
[327] train-rmse:89.84885   eval-rmse:270.25022
[328] train-rmse:89.59772   eval-rmse:270.19590
[329] train-rmse:89.21754   eval-rmse:270.13697
[330] train-rmse:88.90410   eval-rmse:270.04006
[331] train-rmse:88.71664   eval-rmse:269.98791
[332] train-rmse:88.59335   eval-rmse:269.91521
[333] train-rmse:88.20721   eval-rmse:269.71869
[334] train-rmse:87.95186   eval-rmse:269.63900
[335] train-rmse:87.73593   eval-rmse:269.57069
[336] train-rmse:87.42289   eval-rmse:269.54730
[337] train-rmse:87.13537   eval-rmse:269.51371
[338] train-rmse:86.98558   eval-rmse:269.46179
[339] train-rmse:86.65877   eval-rmse:269.44436
[340] train-rmse:86.52663   eval-rmse:269.44633
[341] train-rmse:86.42036   eval-rmse:269.42996
[342] train-rmse:86.17272   eval-rmse:269.42990
[343] train-rmse:86.04014   eval-rmse:269.35061
[344] train-rmse:85.82803   eval-rmse:269.34096
[345] train-rmse:85.60559   eval-rmse:269.39220
[346] train-rmse:85.38596   eval-rmse:269.33726
```

```
[347] train-rmse:85.35807   eval-rmse:269.32453
[348] train-rmse:85.14694   eval-rmse:269.26986
[349] train-rmse:84.87761   eval-rmse:269.15960
[350] train-rmse:84.70549   eval-rmse:269.06939
[351] train-rmse:84.54648   eval-rmse:269.03725
[352] train-rmse:84.32665   eval-rmse:269.00862
[353] train-rmse:84.27271   eval-rmse:268.98587
[354] train-rmse:84.16533   eval-rmse:268.92549
[355] train-rmse:83.96028   eval-rmse:268.82821
[356] train-rmse:83.72791   eval-rmse:268.73082
[357] train-rmse:83.56270   eval-rmse:268.63956
[358] train-rmse:83.27796   eval-rmse:268.65964
[359] train-rmse:83.06485   eval-rmse:268.55970
[360] train-rmse:82.78765   eval-rmse:268.57382
[361] train-rmse:82.65904   eval-rmse:268.53044
[362] train-rmse:82.33577   eval-rmse:268.52266
[363] train-rmse:81.97809   eval-rmse:268.39288
[364] train-rmse:81.81615   eval-rmse:268.40002
[365] train-rmse:81.56212   eval-rmse:268.41549
[366] train-rmse:81.34642   eval-rmse:268.41277
[367] train-rmse:81.04957   eval-rmse:268.36058
[368] train-rmse:80.83640   eval-rmse:268.32243
[369] train-rmse:80.62708   eval-rmse:268.22560
[370] train-rmse:80.35539   eval-rmse:268.18134
[371] train-rmse:80.12992   eval-rmse:267.82607
[372] train-rmse:79.96685   eval-rmse:267.79015
[373] train-rmse:79.82643   eval-rmse:267.82458
[374] train-rmse:79.75109   eval-rmse:267.78881
[375] train-rmse:79.60880   eval-rmse:267.69531
[376] train-rmse:79.46848   eval-rmse:267.68478
[377] train-rmse:79.29181   eval-rmse:267.65623
[378] train-rmse:79.05145   eval-rmse:267.65125
[379] train-rmse:78.88979   eval-rmse:267.65973
[380] train-rmse:78.69805   eval-rmse:267.63107
[381] train-rmse:78.40282   eval-rmse:267.44418
[382] train-rmse:78.18713   eval-rmse:267.41811
[383] train-rmse:78.07928   eval-rmse:267.40473
[384] train-rmse:78.00095   eval-rmse:267.39144
[385] train-rmse:77.87441   eval-rmse:267.33408
[386] train-rmse:77.62428   eval-rmse:267.33187
[387] train-rmse:77.39703   eval-rmse:267.27846
[388] train-rmse:77.26087   eval-rmse:267.30074
[389] train-rmse:77.09913   eval-rmse:267.31907
[390] train-rmse:76.99366   eval-rmse:267.34721
[391] train-rmse:76.80716   eval-rmse:267.37328
[392] train-rmse:76.62940   eval-rmse:267.32575
[393] train-rmse:76.53286   eval-rmse:267.28996
[394] train-rmse:76.31556   eval-rmse:267.28428
[395] train-rmse:76.13838   eval-rmse:267.26129
```

```
[396]	train-rmse:75.89475	eval-rmse:267.16884
[397]	train-rmse:75.78825	eval-rmse:267.15689
[398]	train-rmse:75.64503	eval-rmse:267.12978
[399]	train-rmse:75.57839	eval-rmse:267.12004
[400]	train-rmse:75.40708	eval-rmse:267.10901
[401]	train-rmse:75.24946	eval-rmse:267.02442
[402]	train-rmse:75.06949	eval-rmse:266.98539
[403]	train-rmse:74.87017	eval-rmse:266.95516
[404]	train-rmse:74.61665	eval-rmse:266.91032
[405]	train-rmse:74.41124	eval-rmse:266.92212
[406]	train-rmse:74.05648	eval-rmse:266.85039
[407]	train-rmse:73.86946	eval-rmse:266.80150
[408]	train-rmse:73.66996	eval-rmse:266.77515
[409]	train-rmse:73.55119	eval-rmse:266.73803
[410]	train-rmse:73.41140	eval-rmse:266.72553
[411]	train-rmse:73.20871	eval-rmse:266.62864
[412]	train-rmse:73.03716	eval-rmse:266.59352
[413]	train-rmse:72.86692	eval-rmse:266.59815
[414]	train-rmse:72.77534	eval-rmse:266.55080
[415]	train-rmse:72.57807	eval-rmse:266.56102
[416]	train-rmse:72.31173	eval-rmse:266.48492
[417]	train-rmse:72.18858	eval-rmse:266.50452
[418]	train-rmse:72.04718	eval-rmse:266.56171
[419]	train-rmse:71.83397	eval-rmse:266.51630
[420]	train-rmse:71.68657	eval-rmse:266.46853
[421]	train-rmse:71.42888	eval-rmse:266.47477
[422]	train-rmse:71.23655	eval-rmse:266.46694
[423]	train-rmse:71.12199	eval-rmse:266.40550
[424]	train-rmse:71.07516	eval-rmse:266.40688
[425]	train-rmse:71.03750	eval-rmse:266.39360
[426]	train-rmse:70.82502	eval-rmse:266.27174
[427]	train-rmse:70.46102	eval-rmse:266.10800
[428]	train-rmse:70.26154	eval-rmse:266.05881
[429]	train-rmse:70.02052	eval-rmse:266.01311
[430]	train-rmse:69.82462	eval-rmse:265.99893
[431]	train-rmse:69.66632	eval-rmse:265.98559
[432]	train-rmse:69.45305	eval-rmse:266.03097
[433]	train-rmse:69.28060	eval-rmse:266.08198
[434]	train-rmse:69.11388	eval-rmse:266.07910
[435]	train-rmse:68.98393	eval-rmse:266.05599
[436]	train-rmse:68.78900	eval-rmse:265.96349
[437]	train-rmse:68.56558	eval-rmse:265.97006
[438]	train-rmse:68.37930	eval-rmse:265.96665
[439]	train-rmse:68.23525	eval-rmse:265.99509
[440]	train-rmse:68.14637	eval-rmse:265.95764
[441]	train-rmse:67.85197	eval-rmse:265.96834
[442]	train-rmse:67.66454	eval-rmse:265.95461
[443]	train-rmse:67.47074	eval-rmse:265.94336
[444]	train-rmse:67.35038	eval-rmse:265.86693
```

```
[445]	train-rmse:67.23000	eval-rmse:265.93804
[446]	train-rmse:67.13408	eval-rmse:265.93546
[447]	train-rmse:66.91337	eval-rmse:265.88270
[448]	train-rmse:66.72273	eval-rmse:265.84704
[449]	train-rmse:66.47791	eval-rmse:265.80407
[450]	train-rmse:66.32427	eval-rmse:265.81179
[451]	train-rmse:66.11146	eval-rmse:265.82755
[452]	train-rmse:65.94222	eval-rmse:265.76749
[453]	train-rmse:65.71962	eval-rmse:265.76564
[454]	train-rmse:65.60640	eval-rmse:265.76827
[455]	train-rmse:65.42633	eval-rmse:265.71514
[456]	train-rmse:65.20490	eval-rmse:265.65879
[457]	train-rmse:64.96860	eval-rmse:265.56031
[458]	train-rmse:64.75971	eval-rmse:265.55782
[459]	train-rmse:64.64198	eval-rmse:265.53722
[460]	train-rmse:64.45175	eval-rmse:265.53067
[461]	train-rmse:64.27173	eval-rmse:265.49459
[462]	train-rmse:64.01486	eval-rmse:265.52136
[463]	train-rmse:63.79736	eval-rmse:265.50128
[464]	train-rmse:63.64662	eval-rmse:265.42097
[465]	train-rmse:63.39911	eval-rmse:265.35223
[466]	train-rmse:63.15483	eval-rmse:265.35533
[467]	train-rmse:62.97005	eval-rmse:265.31432
[468]	train-rmse:62.88753	eval-rmse:265.27540
[469]	train-rmse:62.70699	eval-rmse:265.29063
[470]	train-rmse:62.54174	eval-rmse:265.29804
[471]	train-rmse:62.37633	eval-rmse:265.25098
[472]	train-rmse:62.20717	eval-rmse:265.24135
[473]	train-rmse:61.90404	eval-rmse:265.16059
[474]	train-rmse:61.75259	eval-rmse:265.13646
[475]	train-rmse:61.62221	eval-rmse:265.14879
[476]	train-rmse:61.49460	eval-rmse:265.11827
[477]	train-rmse:61.33096	eval-rmse:265.05617
[478]	train-rmse:61.15564	eval-rmse:264.98569
[479]	train-rmse:61.03238	eval-rmse:264.95820
[480]	train-rmse:60.89288	eval-rmse:264.91944
[481]	train-rmse:60.82753	eval-rmse:264.86935
[482]	train-rmse:60.69062	eval-rmse:264.85353
[483]	train-rmse:60.48674	eval-rmse:264.75778
[484]	train-rmse:60.28108	eval-rmse:264.69830
[485]	train-rmse:60.10813	eval-rmse:264.65616
[486]	train-rmse:59.92712	eval-rmse:264.60070
[487]	train-rmse:59.75729	eval-rmse:264.58997
[488]	train-rmse:59.67365	eval-rmse:264.52826
[489]	train-rmse:59.57111	eval-rmse:264.50025
[490]	train-rmse:59.39568	eval-rmse:264.50430
[491]	train-rmse:59.32805	eval-rmse:264.49053
[492]	train-rmse:59.28239	eval-rmse:264.49473
[493]	train-rmse:59.15059	eval-rmse:264.45674
[494]	train-rmse:59.05337	eval-rmse:264.46220
```

```
[495] train-rmse:58.95331   eval-rmse:264.46948
[496] train-rmse:58.81351   eval-rmse:264.44453
[497] train-rmse:58.60417   eval-rmse:264.49280
[498] train-rmse:58.47491   eval-rmse:264.45806
[499] train-rmse:58.29402   eval-rmse:264.38921
Mean Absolute Error (MAE): 194.73541898098645
Root Mean Squared Error (RMSE): 334.0486547952177
Mean Absolute Percentage Error (MAPE): 0.69%
Accuracy: 99.32%
```