**Building an electricity price prediction model involves several steps, including loading and preprocessing the dataset. Here's a step-by-step guide**

Import Necessary Libraries:

First, you'll need to import the necessary Python libraries for data manipulation, visualization, and modeling. Common libraries include pandas, numpy, matplotlib, and scikit-learn.

PYTHON CODE:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

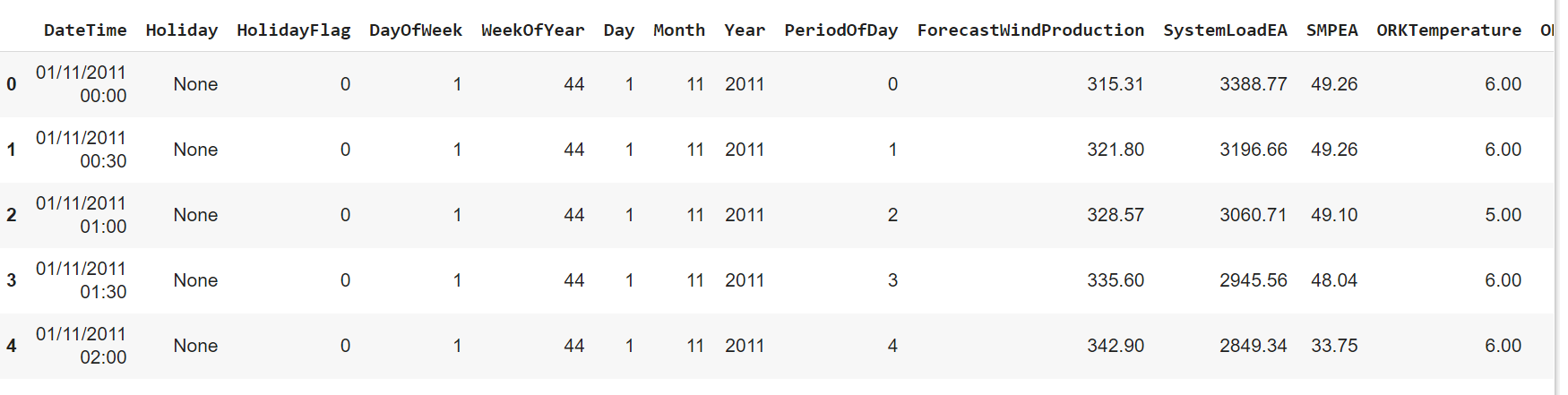
1. **Load the Dataset:**

Assuming that your dataset is stored in a CSV file (let's call it your\_dataset.csv'), you can load it using the `read\_csv` function in Pandas:

#data loading

df=pd.read\_csv("/content/sample\_data/electricity.gui (1).zip")

df.head()



**2.Explore the Dataset:**

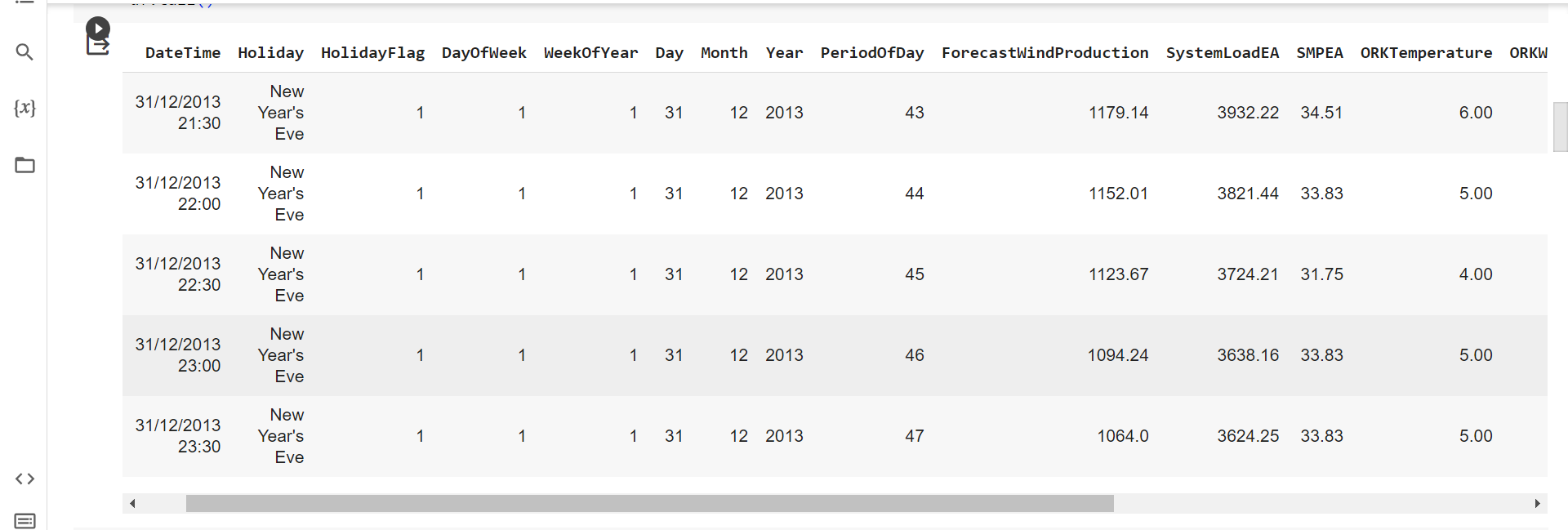
Now that you've loaded the dataset, you should explore it to understand its structure and contents. You can start by looking at the first few rows of the dataset:

Python Code:

df . shape

(38014, 18)

df . tail()



**3.Preprocess the Data:**

**Preprocessing involves tasks like handling missing values, encoding categorical variables, and scaling numerical features.**

**Handle Missing Values:**

# missing value query

df.isna().sum()

WeekOfYear 0

Day 0

Month 0

Year 0

PeriodOfDay 0

ForecastWindProduction 0

SystemLoadEA 0

SMPEA 0

ORKTemperature 0

ORKWindspeed 0

CO2Intensity 0

ActualWindProduction 0

SystemLoadEP2 0

SMPEP2 0

dtype: int64

Encode Categorical Variables:

Encoding categorical variables is an essential step in preparing data for machine learning models, as most algorithms require numerical inputs.

#create a list for numeric and categorical values

cat\_list=[]

num\_list=[]

for i in df.columns:

unique\_val=len(df[i].unique())

if unique\_val<40:

cat\_list.append(i)

else:

num\_list.append(i)

cat\_list.append("WeekOfYear")

cat\_list

**OUTPUT:**

['Holiday',

'HolidayFlag',

'DayOfWeek',

'Day',

'Month',

'Year',

'ORKTemperature',

'WeekOfYear']

# distributions of numeric attributes

# distributions of numeric attributes

num\_list.remove("DateTime")

num\_list

**OUTPUT:**

['WeekOfYear',

'PeriodOfDay',

'ForecastWindProduction',

'SystemLoadEA',

'SMPEA',

'ORKWindspeed',

'CO2Intensity',

'ActualWindProduction',

'SystemLoadEP2',

'SMPEP2']

**Scale Numerical Features:**

Scaling numerical features is an important preprocessing step in many machine learning algorithms. It helps ensure that all features contribute equally to the model's training process, preventing features with larger scales from dominating the learning process.

num\_list.append("ORKTemperature")

k=1

plt.figure(figsize=(12,12))

plt.suptitle("distribution of numerical values")

for i in df.loc[:,num\_list]:

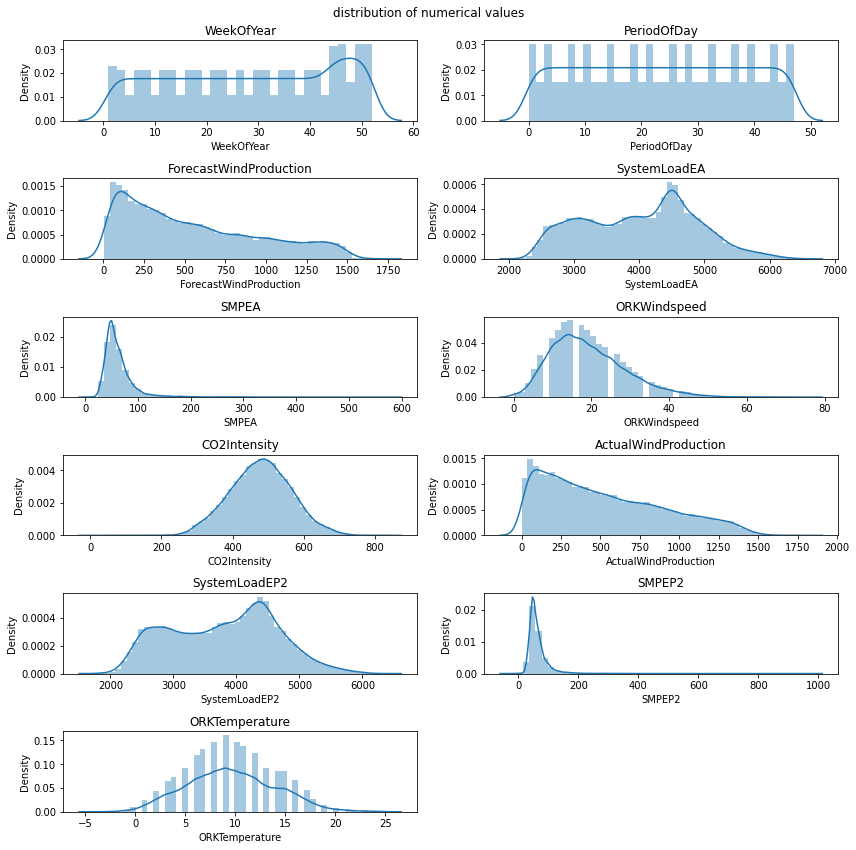
plt.subplot(6,2,k)

sns.distplot(df[i])

plt.title(i)

k+=1

plt.tight\_layout()

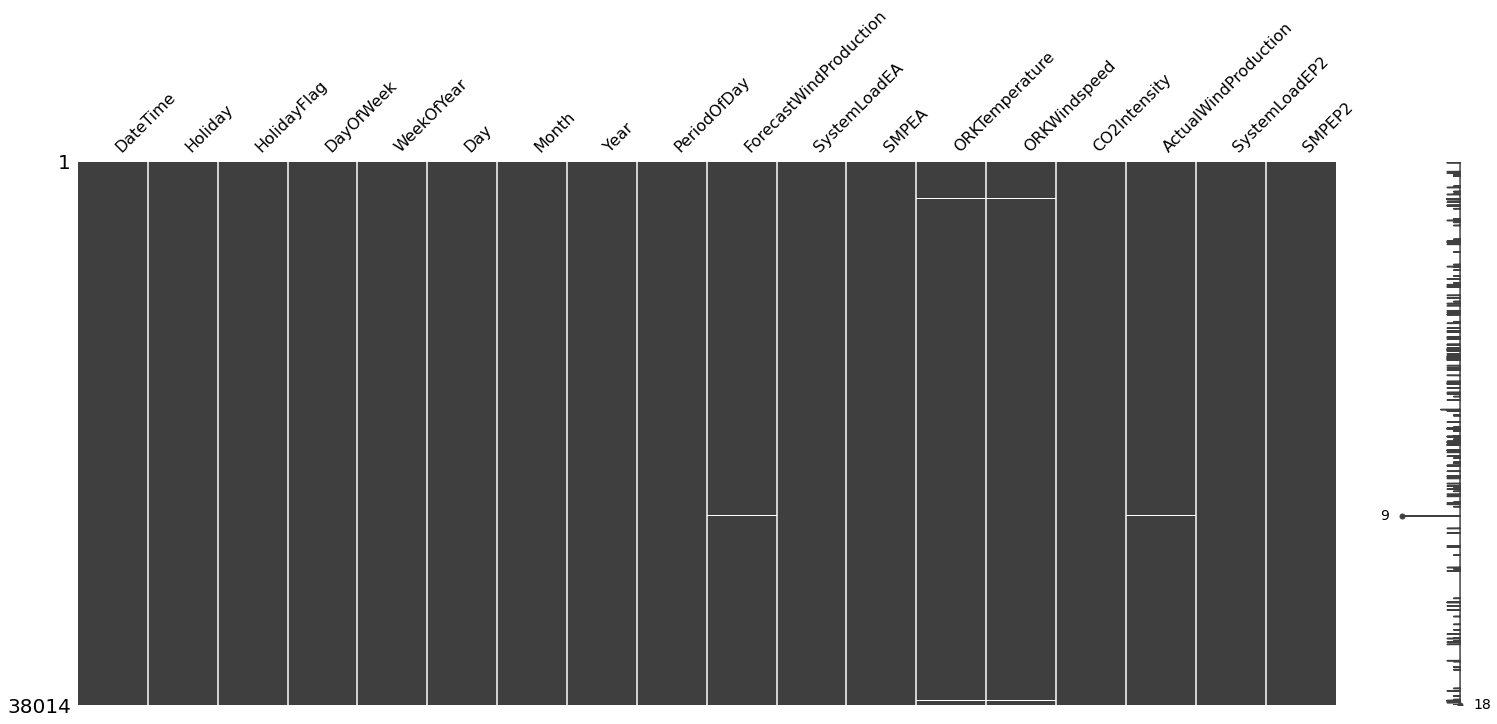


Visualization of missing values:

**Visualizing missing data is crucial for understanding the extent and patterns of missingness in a dataset. It helps in making informed decisions about how to handle missing values.**

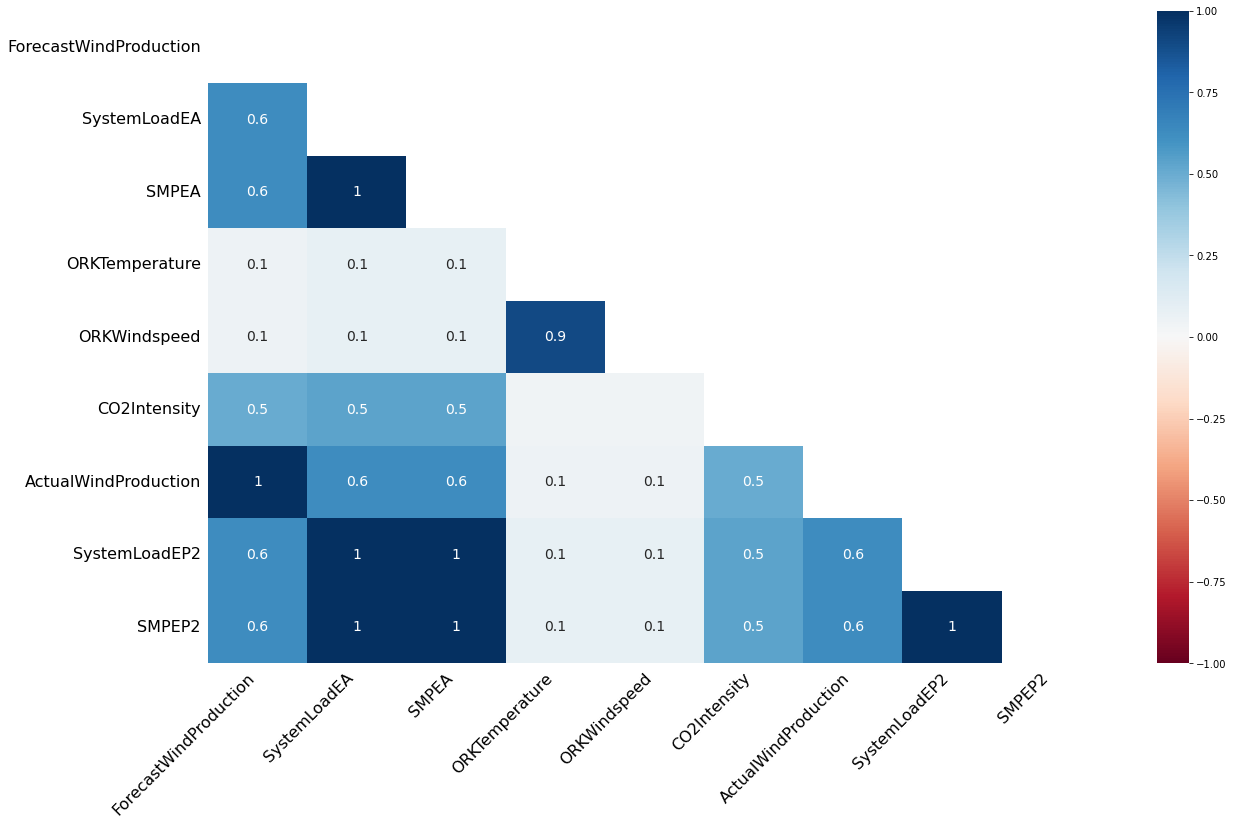
import missingno as msno

msno.matrix(df);



# let's visualize whether there is a relationship between the missing values

msno.heatmap(df);



Different Analysis:

Time Series Analysis:

Time series analysis is a statistical technique used to analyze and extract meaningful information from time-ordered data points.

from datetime import datetime

df["DateTime"] = pd.to\_datetime(df.DateTime)

df['year'] = df['DateTime'].dt.year

df['month'] = df['DateTime'].dt.month

df["day"]=df["DateTime"].dt.day

# We have created 3 new columns

# we can start our time series analysis

# change of real price of consumed electricity with time

custgroup=df.groupby('DateTime').mean()

plt.figure(figsize=(12,5))

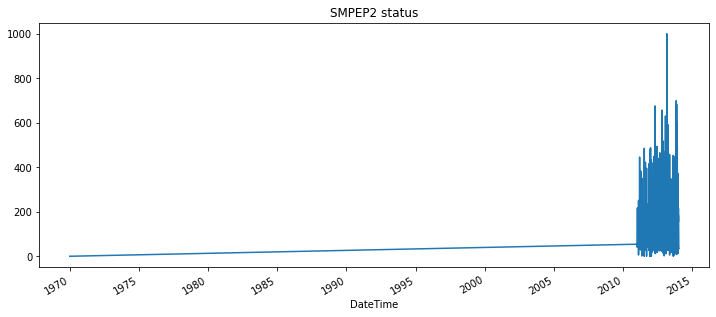
custgroup['SMPEP2'].plot(x=df.DateTime)

plt.figure(figsize=(12,5))

custgroup['SMPEP2'].plot(x=df.DateTime)

plt.title("SMPEP2 status")

plt.show()



Data Visualize:

**Data visualization is a critical part of the data analysis process. It helps in understanding the underlying patterns, trends, and relationships in the data.**

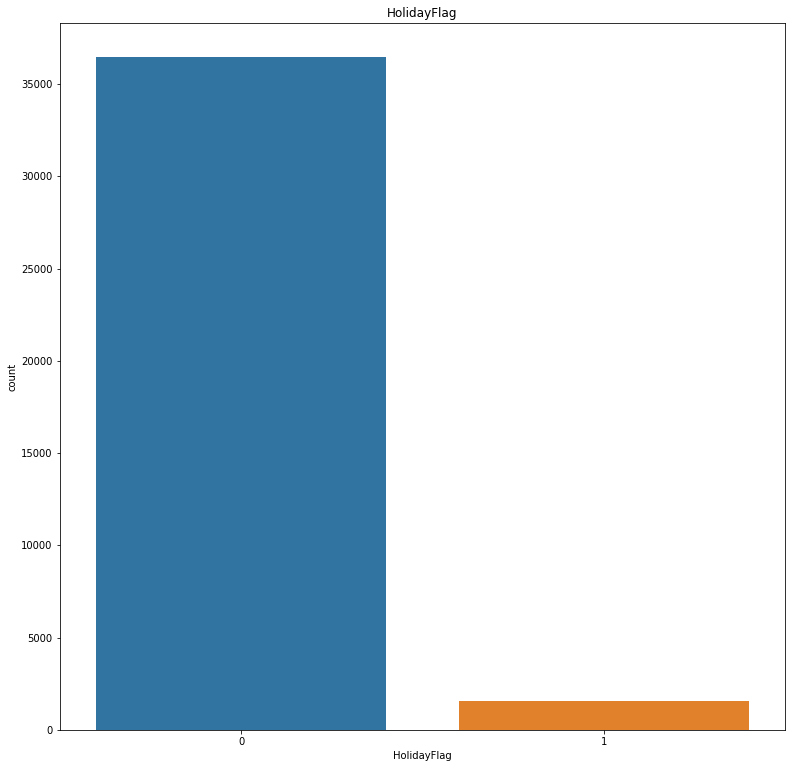
# Categorical Analysis

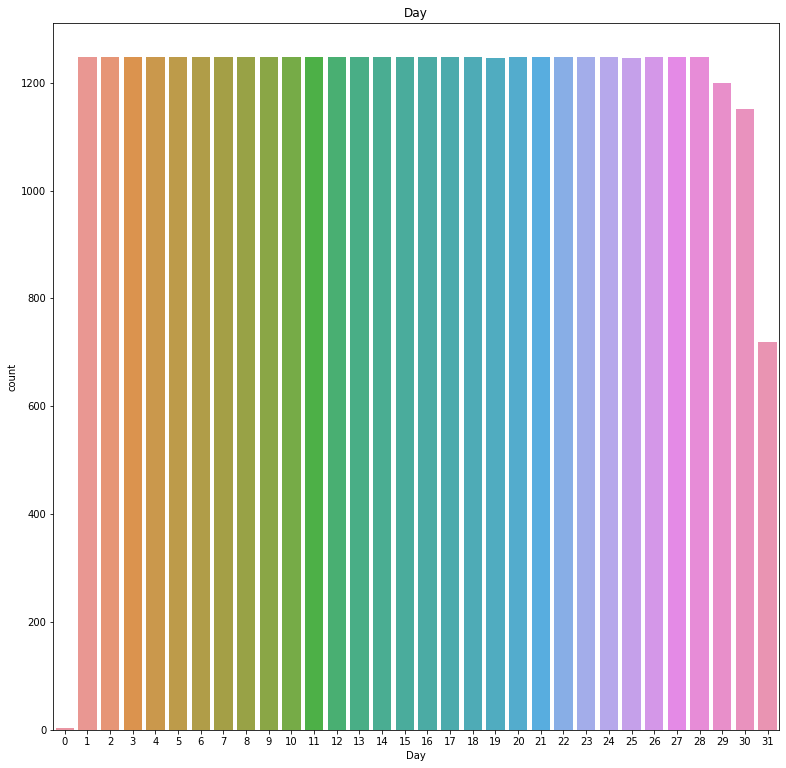
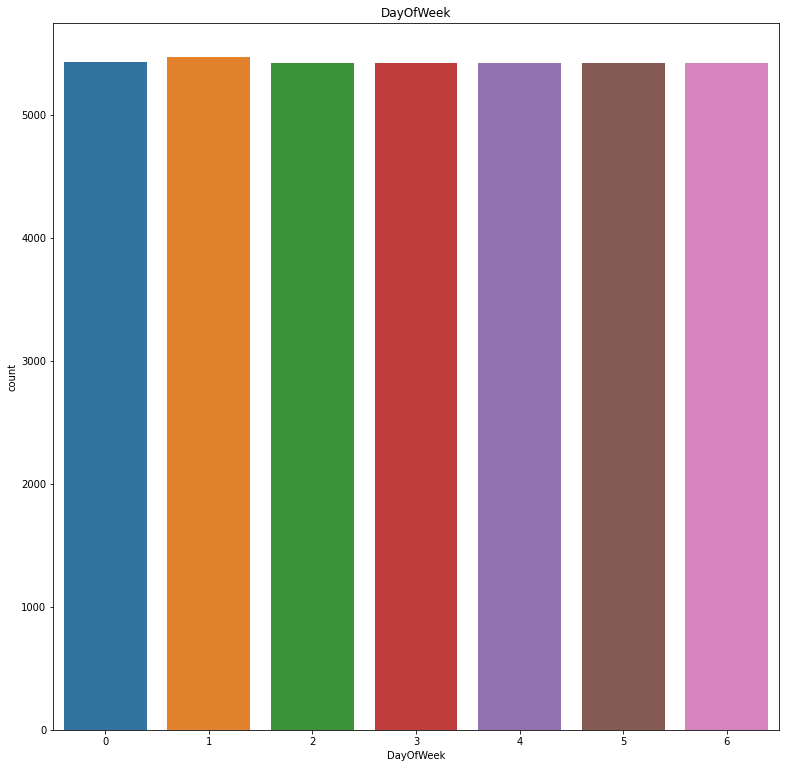
for i in cat\_list:

plt.figure(figsize=(13,13))

sns.countplot(x=i,data=df.loc[:,cat\_list])

plt.title(i)

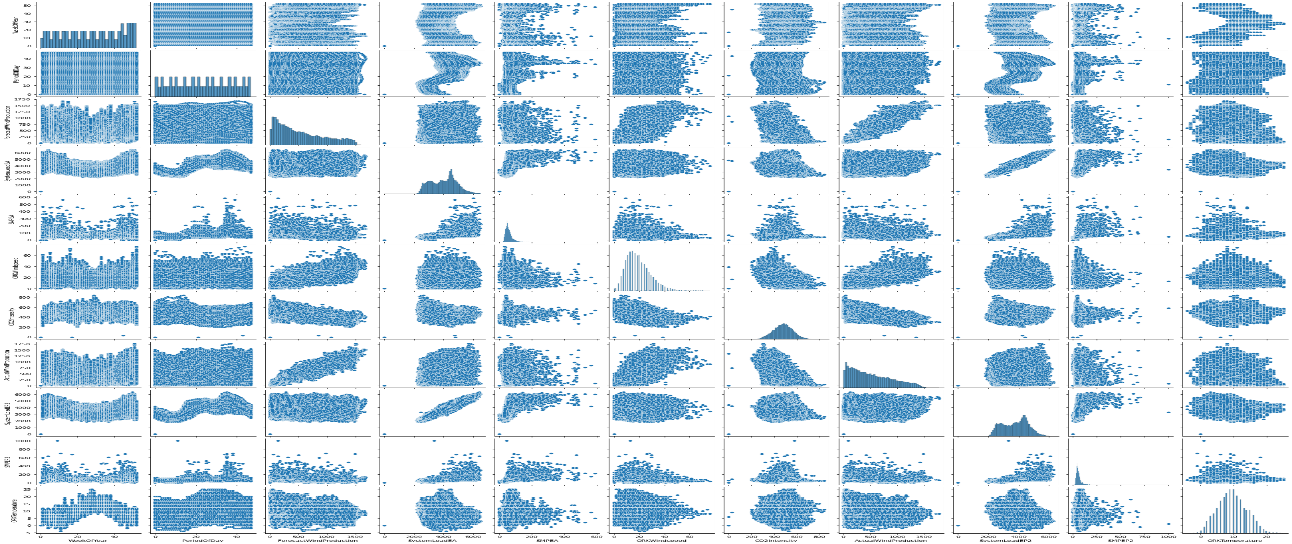




# numerical analysis

Numerical analysis is a branch of mathematics and computer science that deals with the development and application of computational algorithms to solve mathematical problems. It involves techniques for approximating solutions to mathematical problems that may be too complex to solve analytically.

sns.pairplot(df.loc[:,num\_list]);



# histogram

A histogram is a graphical representation of the distribution of a dataset. It provides a visual summary of the underlying frequency distribution of a set of continuous or discrete data.

df.hist(figsize=(9,9));

