Topic: development part 2

Electricity price prediction

Electricity is a basic human need and definitely one of the most important factors of societal progress. In recent decades however, electricity has entered the market as a tradeable commodity and the power industry of many countries has been **deregulated**.

In Spain, the Electric Power Act 54/1997 exposed all of the stakeholders to high amounts of uncertainty as the price of electricity is determined by countless factors and also, due to the fact that electricity cannot be stored in large quantities efficiently [1]. With the emergence of this new market, the need for reliable forecasting methods at all scales (hourly, daily, long-term, etc.) has also emerged and has become a large area of research.

Import os

Import pandas as pd

Import numpy as np

Import matplotlib.pyplot as plt

Import seaborn as sns

Import statsmodels.api as sm

Import tensorflow as tf

Import xgboost as xgb

Import os

Import warnings

From tensorflow.keras.layers import Dense, LSTM, Conv1D, MaxPooling1D, TimeDistributed, Flatten, Dropout, RepeatVector

From statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

From statsmodels.tsa.stattools import adfuller, kpss, ccf

From sklearn.metrics import mean\_squared\_error, r2\_score

From sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler

From sklearn.decomposition import PCA

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

From math import sqrt

%matplotlib inline

For dirname, \_, filenames in os.walk(‘/kaggle/input’):

For filename in filenames:

Print(os.path.join(dirname, filename))

Warnings.simplefilter(action=’ignore’, category=(FutureWarning, UserWarning

Exploration and Cleaning¶

Weather\_features.csv’: Contains hourly information about the weather conditions (e.g. temperature, wind speed, humidity, rainfall, qualitative desctiption) of 5 major cities in Spain (Madrid, Barcelona, Valencia, Seville and Bilbao).

Energy\_dataset.csv’: Contains hourly information about the generation of energy in Spain. In particular, there is info (in MW) about the amount of electricty generated by the various energy sources (fossil gas, fossil hard coal and wind energy dominate the energy grid), as well as about the total load (energy demand) of the national grid and the price of energy (€/MWh).

Df\_weather = pd.read\_csv(

‘/kaggle/input/energy-consumption-generation-prices-and-weather/weather\_features.csv’,

Parse\_dates=[‘dt\_iso’]

)

Df\_energy = pd.read\_csv(

‘/kaggle/input/energy-consumption-generation-prices-and-weather/energy\_dataset.csv’,

Parse\_dates=[‘time’]

)

Energy dataset¶

Df\_energy.head()

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 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

0 2015-01-01 00:00:00+01:00 447.0 329.0 0.0 4844.0 4821.0 162.0 0.0 0.0 0.0 … 196.0 0.0 6378.0 17.0 NaN 6436.0 26118.0 25385.0 50.10 65.41

1 2015-01-01 01:00:00+01:00 449.0 328.0 0.0 5196.0 4755.0 158.0 0.0 0.0 0.0 … 195.0 0.0 5890.0 16.0 NaN 5856.0 24934.0 24382.0 48.10 64.92

2 2015-01-01 02:00:00+01:00 448.0 323.0 0.0 4857.0 4581.0 157.0 0.0 0.0 0.0 … 196.0 0.0 5461.0 8.0 NaN 5454.0 23515.0 22734.0 47.33 64.48

3 2015-01-01 03:00:00+01:00 438.0 254.0 0.0 4314.0 4131.0 160.0 0.0 0.0 0.0 … 191.0 0.0 5238.0 2.0 NaN 5151.0 22642.0 21286.0 42.27 59.32

4 2015-01-01 04:00:00+01:00 428.0 187.0 0.0 4130.0 3840.0 156.0 0.0 0.0 0.0 … 189.0 0.0 4935.0 9.0 NaN 4861.0 21785.0 20264.0 38.41 56.04

5 rows × 29 columns

We will drop all the columns that are constituted by zeroes and NaNs, as they are unusable. We will also remove the columns which will not be used at all in our analysis and which contain day-ahead forecasts for the total load, the solar energy and the wind energy.

# Drop unusable columns

Df\_energy = df\_energy.drop([‘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’,

‘total load forecast’, ‘forecast solar day ahead’,

‘forecast wind onshore day ahead’],

Df\_energy.info()

<class ‘pandas.core.frame.DataFrame’>

RangeIndex: 35064 entries, 0 to 35063

Data columns (total 18 columns):

Time 35064 non-null object

Generation biomass 35045 non-null float64

Generation fossil brown coal/lignite 35046 non-null float64

Generation fossil gas 35046 non-null float64

Generation fossil hard coal 35046 non-null float64

Generation fossil oil 35045 non-null float64

Generation hydro pumped storage consumption 35045 non-null float64

Generation hydro run-of-river and poundage 35045 non-null float64

Generation hydro water reservoir 35046 non-null float64

Generation nuclear 35047 non-null float64

Generation other 35046 non-null float64

Generation other renewable 35046 non-null float64

Generation solar 35046 non-null float64

Generation waste 35045 non-null float64

Generation wind onshore 35046 non-null float64

Total load actual 35028 non-null float64

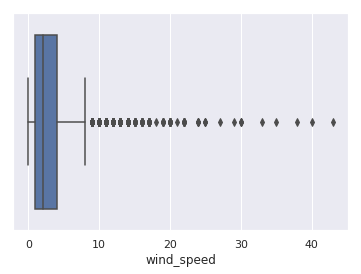
Price day ahead 35064 non-null float64

Price actual 35064 non-null float64

Dtypes: float64(17), object(1)

Memory usage: 4.8+ MBAxis=1)

The ‘time’ column, which we also want to function as the index of the observations in a time-series, has not been parsed correctly and is recognized as an object.

Df\_energy[‘time’] = pd.to\_datetime(df\_energy[‘time’], utc=True, infer\_datetime\_format=True)

Df\_weather.interpolate(method=’linear’, limit\_direction=’forward’, inplace=True, axis=0)

Merging the two datasets

Df\_1, df\_2, df\_3, df\_4, df\_5 = [x for \_, x in df\_weather.groupby(‘city\_name’)]

Dfs = [df\_1, df\_2, df\_3, df\_4, df\_5]

Df\_final = df\_energy

For df in dfs:

City = df[‘city\_name’].unique()

City\_str = str(city).replace(“’”, “”).replace(‘[‘, ‘’).replace(‘]’, ‘’).replace(‘ ‘, ‘’)

Df = df.add\_suffix(‘\_{}’.format(city\_str))

Df\_final = df\_final.merge(df, on=[‘time’], how=’outer’)

Df\_final = df\_final.drop(‘city\_name\_{}’.format(city\_str), axis=1)

Plot\_model\_rmse\_and\_loss(history)

