Time Series Analysis of Household Electricity Consumption

Ankush Dey (MDS202108) Aniket Santra (MDS202106)

Under The Guidance Of Prof.Srinivasan Mamandur

April 19, 2023

CHENNAI MATHEMATICAL INSTITUTE

Outline

Dataset

Data source:- https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

Abstract:- Measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years.

Columns Description:-

- date: Date in format dd/mm/yyyy
- time: time in format hh:mm:ss
- global_active_power: household global minute-averaged active power(in kilowatt)
- global_reactive_power: household global minute-averaged reactive power (in kilowatt)
- voltage: minute-averaged voltage (in volt)
- global_intensity: household global minute-averaged current intensity (in ampere)

Dataset

- sub_metering_1: energy sub-metering No. 1 (in watt-hour of active energy). It corresponds to the kitchen, containing mainly a dishwasher, an oven, and a microwave (hot plates are not electric but gas-powered)
- sub_metering_2: energy sub-metering No. 2 (in watt-hour of active energy). It corresponds to the laundry room, containing a washing machine, a tumble drier, a refrigerator, and a light.
- sub_metering_3: energy sub-metering No. 3 (in watt-hour of active energy). It corresponds to an electric water heater and an air conditioner.

Dataset

```
1 data.isnull().sum()
```

```
Global_active_power      25979
Global_reactive_power    25979
Voltage                  25979
Global_intensity         25979
Sub_metering_1           25979
Sub_metering_2           25979
Sub_metering_3           25979
```

We have used the method of 'forward-fill' to handle the missing values.

Dataset

we have created an additional column power consumption.
The formula is:-

$$\text{power\_consumption} = \left(\frac{Global\_active\_power * 1000}{60}\right) - (\text{sub\_metering\_1} + \text{sub\_metering\_2} + \text{sub\_metering\_3})$$

Goals

- Visualize and understand the data

Goals

- Visualize and understand the data
- analyzing the components of time series like trend, and seasonality

Goals

- Visualize and understand the data
- analyzing the components of time series like trend, and seasonality
- Checking the stationarity and the pattern of the data

Goals

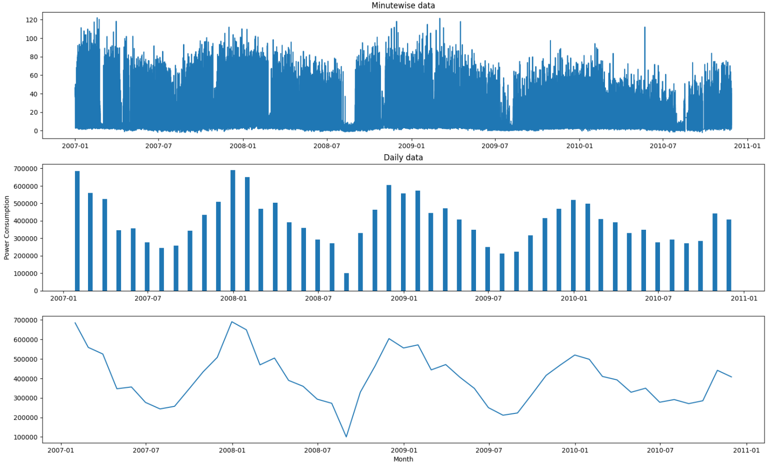
- Visualize and understand the data
- analyzing the components of time series like trend, and seasonality
- Checking the stationarity and the pattern of the data
- Building several time series and machine learning models to forecast future power consumption

Goals

- Visualize and understand the data
- analyzing the components of time series like trend, and seasonality
- Checking the stationarity and the pattern of the data
- Building several time series and machine learning models to forecast future power consumption
- Checking the model accuracies using several metrics like RMSE, MAE, MAPE.

Exploratory Data Analysis

Visualizing the Power Consumption data



Exploratory Data Analysis

Summary Statistics for minute-wise data

```
count     2.075259e+06
mean      9.287283e+00
std       9.545391e+00
min      -2.400000e+00
25%       3.800000e+00
50%       5.500000e+00
75%       1.040000e+01
max       1.248333e+02
Name: power_consumption, dtype: float64
```

Summary statistics for monthly data

```
count         47.000000
mean      400333.752482
std       131087.349658
min        99918.100000
25%       292499.150000
50%       392487.866667
75%       484446.700000
max       690648.100000
Name: power_consumption, dtype: float64
```

Trend, Stationarity, Seasonality

Trend

The trend is a pattern in data that shows the movement of a series to relatively higher or lower values over a long period of time

Stationarity

Stationarity means that the statistical properties of a process generating a time series do not change over time.

Seasonality

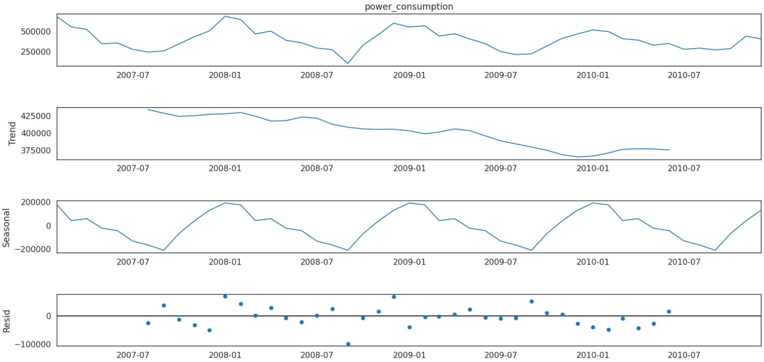
It is a characteristic of a time series in which the data experiences regular and predictable changes that recur every calendar year

Trend, Stationarity, Seasonality

So it is clear that there is decreasing trend in the data

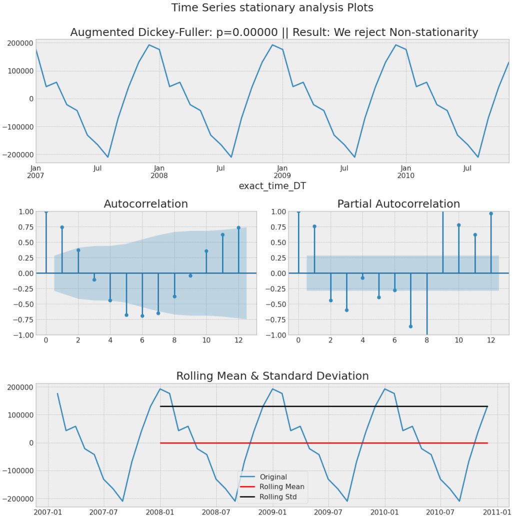
Trend, Stationarity, Seasonality

Seasonal component



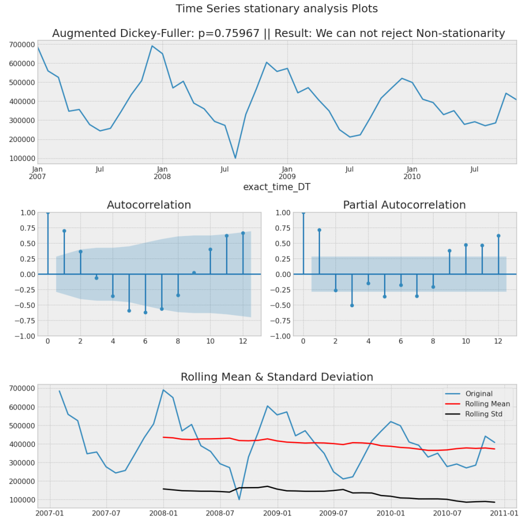
Trend, Stationarity, Seasonality

ACF ,PACF of seasonality component



Trend, Stationarity, Seasonality

ADF test of original data



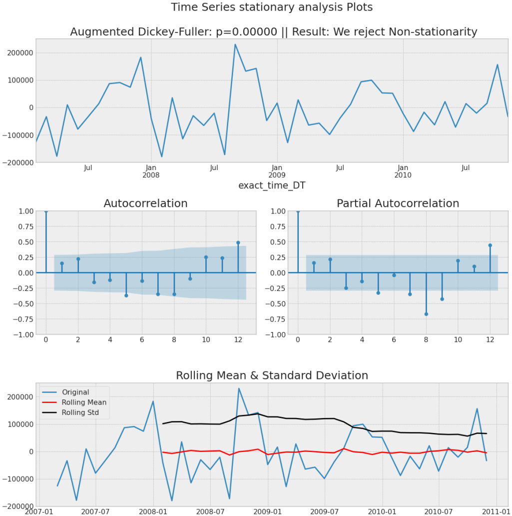
Trend, Stationarity, Seasonality

So from the above plot, it is clear that the data is not stationary, to make it stationary we will use the successive difference method.

```
data_power_consumption['diff'] = data_power_consumption['power_consumption'].diff().dropna()

sts.adfuller(data_power_consumption['diff'].dropna())

(-5.709250064396282,
 7.36469100176175e-07,
 10,
 35,
 {'1%': -3.6327426647230316,
  '5%': -2.9485102040816327,
  '10%': -2.6130173469387756},
 865.9928440411717)
```

Trend, Stationarity, Seasonality

ADF test of differenced data



Trend, Stationarity, Seasonality

Grid Search

	q=0	q=1
p=0	866.372926	866.929248
p=1	867.496562	867.879178

SARIMA model

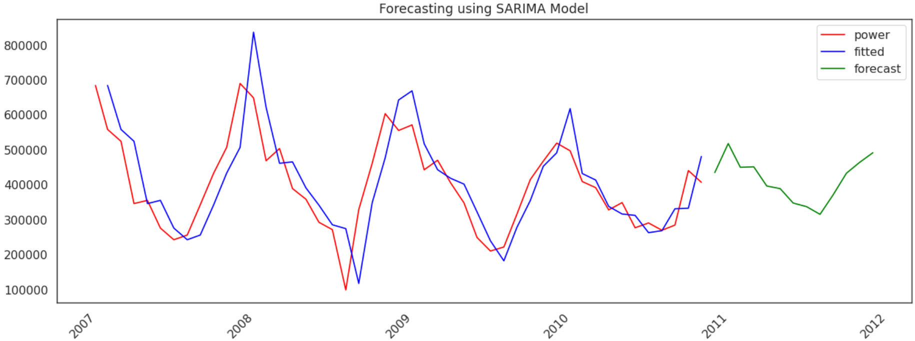
Autoregressive Integrated Moving Average, or ARIMA, is one of the most widely used forecasting methods for univariate time series data forecasting.
Although the method can handle data with a trend, it does not support time series with a seasonal component.
An extension to ARIMA that supports the direct modeling of the seasonal component of the series is called SARIMA.

the formula of our model is SARIMA(0,1,0)(1,1,1,12) is given as:-

$$\Delta Y_t = c + \Phi_1(\Delta Y_{t-12}) + \Theta_1(\epsilon_{t-12}) + \epsilon_t$$

SARIMA model

Prediction and Forecasting



SARIMA model

Evaluation Metric for SARIMA

	mae	mape	rmse	R2_score
SARIMA	70089.006759	0.200388	88879.063104	0.627672

XGboost model

Definition

It's a machine learning algorithm used for supervised learning problems, especially for regression and classification tasks. It is an extension of the classic Gradient Boosting algorithm.

It works by combining several weak decision tree models into a single strong model. It is based on the principle of ensemble learning, where multiple weak learners are combined to create a single powerful model.

It is a powerful and efficient tool for time series analysis, with several features that make it suitable for complex datasets. It can handle missing values, determine feature importance, prevent overfitting, generate predictions quickly, and combine weak models into strong ensemble models.

XGboost

```
import xgboost as xgb
from xgboost import plot_importance
reg = xgb.XGBRegressor(n_estimators=1000)
reg.fit(X_train, y_train,
eval_set=[(X_train, y_train), (X_test, y_test)],
        early_stopping_rounds=50, verbose=True)
```

```
1 X_train.head(5)
```

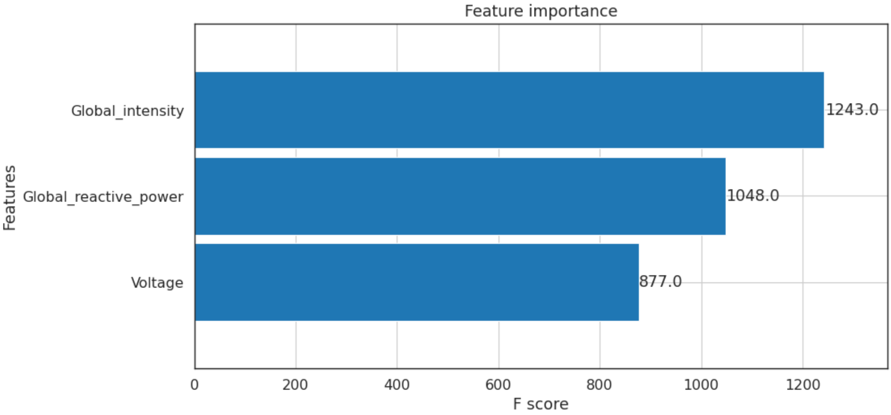
	Global_reactive_power	Voltage	Global_intensity
dt			
2006-12-16 17:24:00	0.418	234.84	18.4
2006-12-16 17:25:00	0.436	233.63	23.0
2006-12-16 17:26:00	0.498	233.29	23.0
2006-12-16 17:27:00	0.502	233.74	23.0
2006-12-16 17:28:00	0.528	235.68	15.8

```
1 y_train.head(5)
```

	power_consumption
dt	
2006-12-16 17:24:00	52.266667
2006-12-16 17:25:00	72.333333
2006-12-16 17:26:00	70.566667
2006-12-16 17:27:00	71.800000
2006-12-16 17:28:00	43.100000

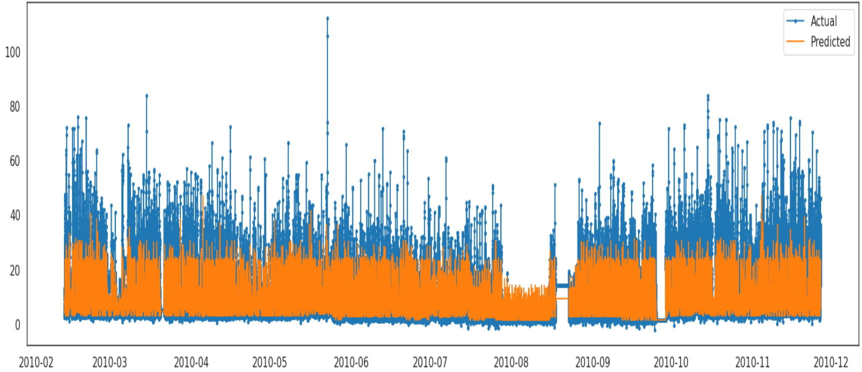
XGboost

Feature Importance



XGboost

Results



XGboost

Evaluation Metrics

```
root mean squared error for train data:  6.724696815162945

Mean Absolute Error for train data: 3.5185407788694625

root mean squared error for test data:  4.975816228263861

Mean Absolute Error for test data: 2.772107264830035

R2 score : 0.4739645444118866
```

LSTM model

Definition

It is a type of recurrent neural network (RNN) that is designed to handle the issue of vanishing gradients in traditional RNNs.

LSTM model will learn a function that maps a sequence of past observations as input to an output observation. As such, the sequence of observations must be transformed into multiple examples from which the LSTM can learn.

LSTM model

LSTM Architecture

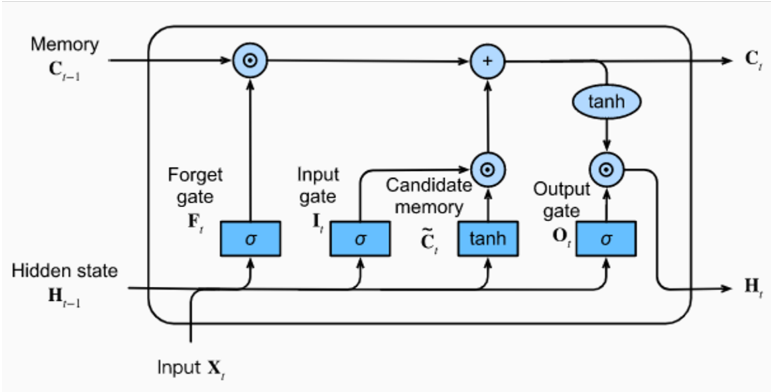


Figure: LSTM

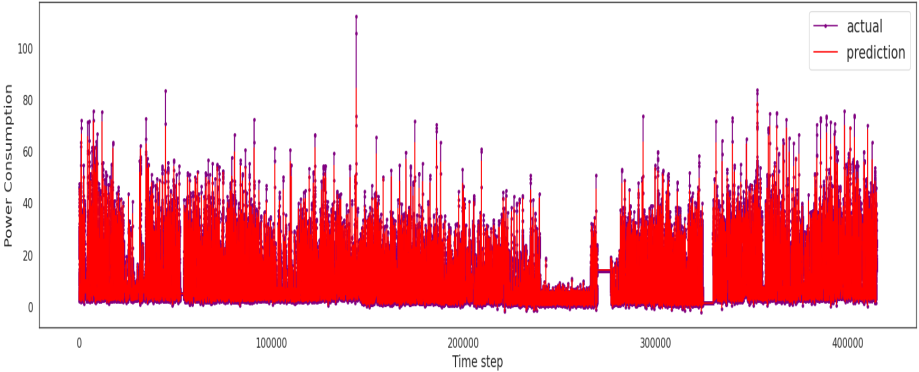
LSTM model

Model Structure

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100)               52400

 dropout (Dropout)           (None, 100)               0

 dense (Dense)               (None, 1)                 101

=================================================================
Total params: 52,501
Trainable params: 52,501
Non-trainable params: 0
_________________________________________________________________
```

LSTM

LSTM Results



LSTM Model

Evalution Metrics

Train Mean Absolute Error: 1.2241271541889627

Train Root Mean Squared Error: 2.9263701267319977

Test Mean Absolute Error: 1.111916316360433

Test Root Mean Squared Error: 2.157108086066503

R2 score: 0.9011431091034725

Model Comparison

	XGboost	LSTM
RMSE	4.9758	2.1500
MAE	2.7720	1.1119