Power consumption prediction

Jasurbek Usmonaliev

Contents

1. About

Electric energy consumption prediction using synthetic data

Pipeline:

Data Cleaning – organizing – visualizing – prep.for modelling –feature selection – sequence creation - modeling – getting results

2.Data

(https://github.com/Jasurbek701/Jasurbek701.git)

Dataset contains below columns:

1.time (object)
2.temperature (float)
3.weather (object)
4.condition (object)
5.powr_mhw (float)

```
df.head()
```

	time	temperature	weather	condition	power_mhw
0	2020-01-01 00:00:00	-11.06	cold	night	59.99
1	2020-01-01 01:00:00	-10.20	cold	night	55.10
2	2020-01-01 02:00:00	-8.83	cold	night	56.30
3	2020-01-01 03:00:00	-11.23	cold	night	61.16
4	2020-01-01 04:00:00	-9.97	cold	night	57.43

Next steps: Generate code with df | View recommended plots | New interactive sheet

Description:

```
Descriptive statistics:
         temperature     power_mhw
count  52585.000000  52585.000000
mean      15.844243     66.211626
std       18.341340     18.123292
min      -18.270000     16.810000
25%       -1.120000     53.380000
50%        9.990000     61.700000
75%       34.040000     83.890000
max       51.630000    105.930000
```

-Minimum temperature overtime is –18C, and max is 51.2C,

-Min power_mhw is 17C and 106C max

-dataset contains 52585 rows and 5 columns

-0 NA vor missing values among all dataset

Example and types of values:

Time: 2020-01-01 03:00:00

Temperature : from −18.27 to 51.63C

Weather: cold, cool, unknown, warm, hot

weather	count
cold	22523
hot	20364
cool	5210
warm	4283
unknown	205

dtype: int64

condition	count
day	28483
night	24102

dtype: int64

Condition: day and night

3.Cleaning

Actually, the data is clean enough. But there is 205 ‘unknown’ values in weather column. Se we do not need it. 205 rows with this value will be deleted
Finally there are reliable values:

```
Unique values in 'weather' after deletion:
 ['cold' 'cool' 'warm' 'hot']

Shape of the DataFrame after deletion:
 (52380, 5)
```

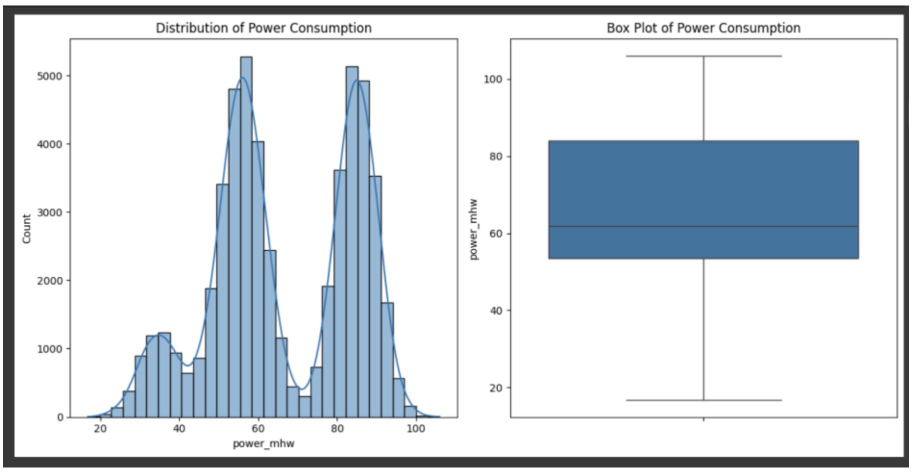
For better understanding there has been done recategorizing values in weather column:

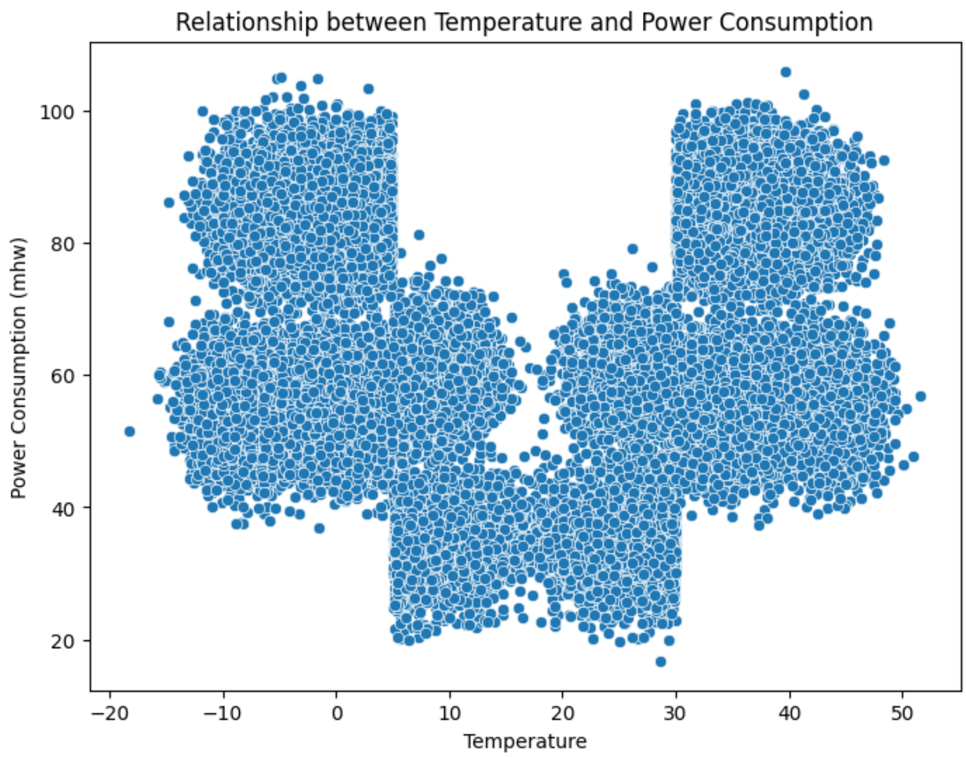
```
def categorize_temperature(temp):
if temp <= 5:
return 'cold'
elif 5.1 <= temp <= 15:
return 'cool'
elif 15.1 <= temp <= 28:
return 'warm'
elif 28.1 <= temp <= 60:
return 'hot'
else:
return 'unknown'

df['weather'] = df['temperature'].apply(categorize_temperature)
```

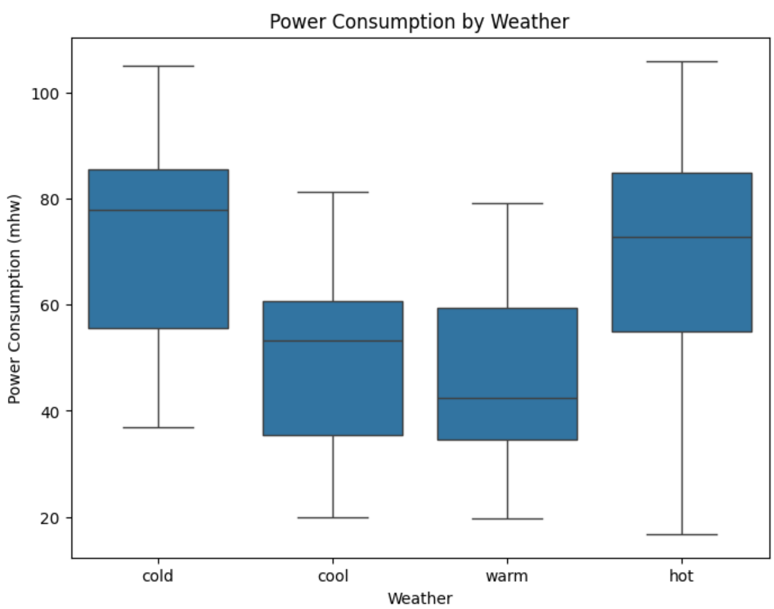
Before that, all column with weather values was not suitable for temperature column, because even +10 showed as cold in the dataset, to make it more suitable I decided reshape and put new rules for the weather column, for example if temperature is more than 28.1C it is ‘hot’

4. VISUALIZATION





From the scatterplot I can say that in cold and hot weather energy consumption increases significantly even 2x or 3x.



The bar plot shows:

In cold weather the average consumption is around 80mhw, in cool and warm weather between 40and 60, but in hot the average consumption of energy is 75mhw. The scatter plot shows a positive correlation between temperature and power consumption, with higher temperatures generally associated with higher power consumption.

5.Preparation for Modelling

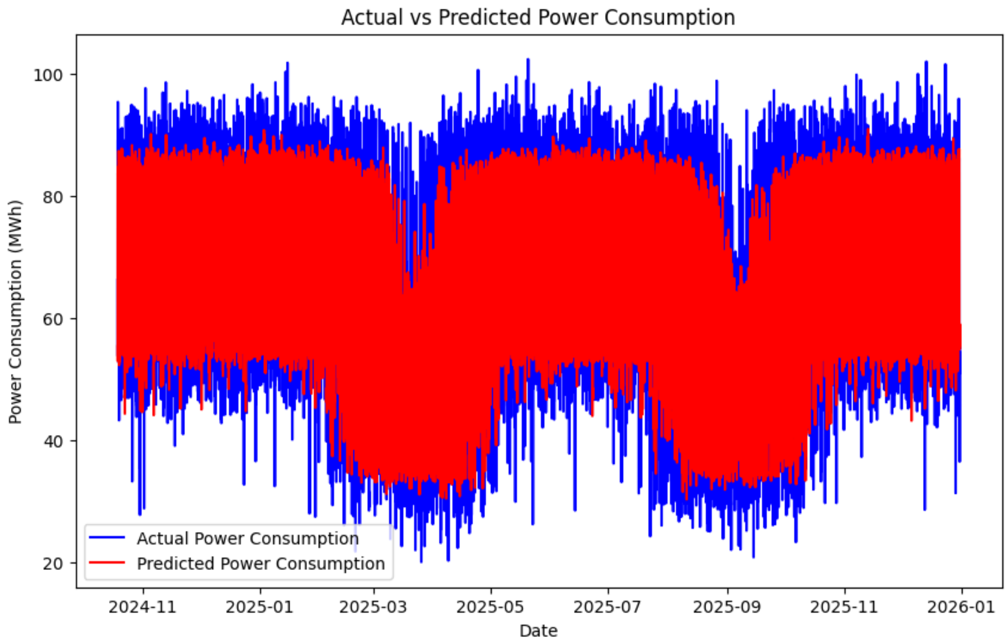
Converting ‘time’ column to DataTime format -> Feature Engineering -> Dropping any rows if any -> Splitting data to training and testing ->

6.Modelling

6.1 RANDOM FOREST

Trained -> Testing -> Evaluation -> Prediction

Results: Mean Absolute Error (MAE): 5.338492026009669
Root Mean Squared Error (RMSE): 7.179241937226792
R-squared: 0.9224911321536201



RANDOM FOREST

6.2 XGBOOST

Feature Engineering -> Encoding Cat variables -> Preparing X and Y -> Splitting dataset -> model defining (n_estimators=50, learning_rate=0.1, max_depth=3) -> Training -> Evaluating -> Predicting

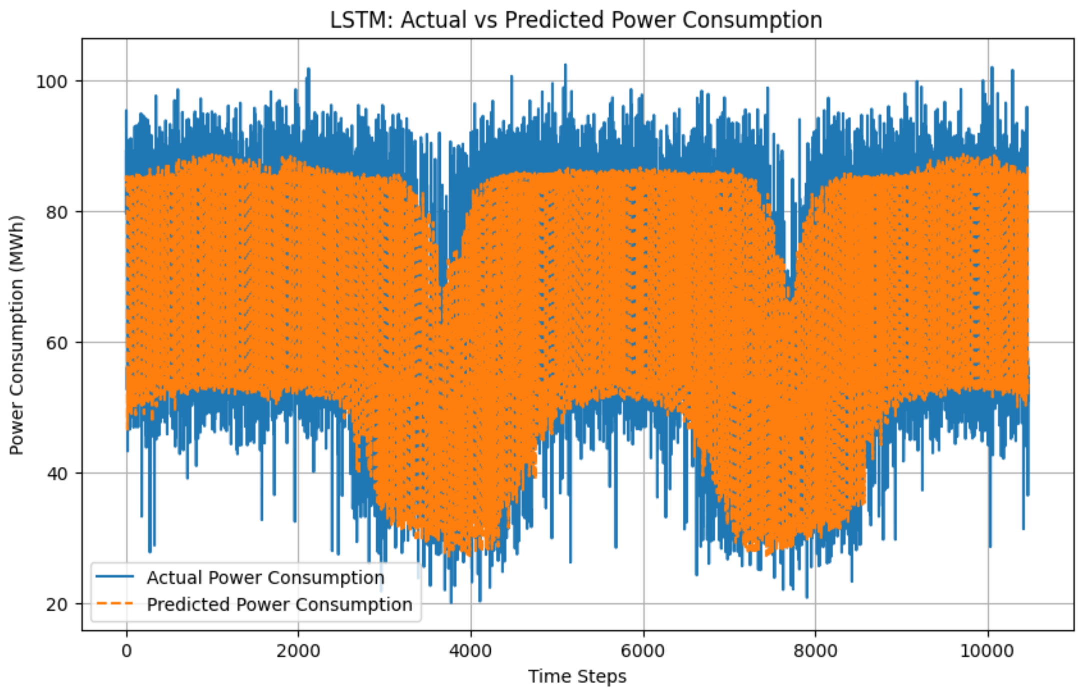
Results: R-squared: 0.9224911321536201

Mean Absolute Error (MAE): 4.020165487568358

Root Mean Squared Error (RMSE): 5.052119978188211

6.3 LSTMFeature selection -> Scaling -> Sequencing -> Splitting -> Defining model -> Training the model -> predicting

RESULTS



LSTM

Mean Absolute Error (MAE): 5.338492026009669

Root Mean Squared Error (RMSE): 7.179241937226792

R-squared (R^2): 0.8400872837153099

7. CONCLUSION

The XGBoostmodel performs best, as it has the lowest MAE (4.020) and lowest RMSE (5.052), indicating more accurate predictions with smaller errors. The R^2 value is 0.922 for the Random Forest and Boosting models, meaning both explain around 92% of the variance. However, since the second model has better error metrics (MAE and RMSE), it is the best-performing model overall.