NAZARBAYEV UNIVERSITY

NAZARBAYEV UNIVERSITY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
ELCE 455 Machine Learning with Python

Solar panel's hourly power prediction using the XGBoost algorithm

Student 1: Ulan Sharipov
Student 2: Ibragim Gimalay
Instructor: Amin Zollanvari

School of Engineering and Digital Sciences
Academic Year 2023-2024

NAZARBAYEV UNIVERSITY

Abstract

Forecasting the generation of electrical energy is an important part of consumer trust in the manufacturer. Solar energy as one of the sources of electricity is highly dependent on the amount of irradiation and other geographical parameters, which makes it difficult to control. To successfully implement solar power plants into the grid, it is necessary to develop data-based forecasting models. This project used the xgboost algorithm with accompanying procedures for deploying a machine-learning model. The results showed the forecasting accuracy on the test date with an r2 indicator of 99%.

Introduction

Renewable energy types such as PV solar panels are promising technologies that could potentially be cheap in usage and produce significantly less carbon emissions. More and more countries are concerned about climate change due to fossil fuel pollution. It is estimated that ⅔ of all carbon pollution is from this type of energy generation that primarily uses coal [1]. There is no doubt that continued reliance on coal will lead to significant temperature rises around the globe.

Despite having advantages, PV solar panels have one weak point, which is their predictability. It is known that PV panels will work only during sun time. However, time under the sun could be affected by cloud opacity, and efficiency in conversion to electricity might be decreased by hot temperatures. Without reasonable predictability, it is unlikely that PV panels will be inserted into the country's main grid infrastructure.

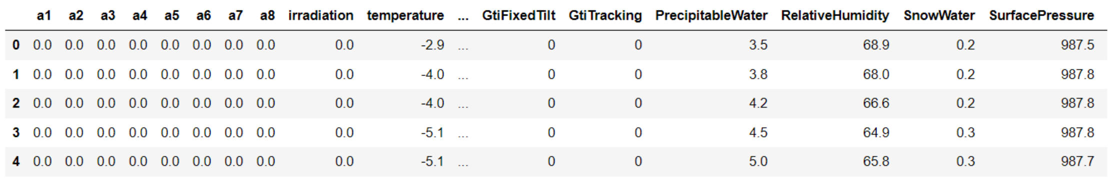
This project will try to answer this problem using a dataset consisting of more than 2100 rows.

Methodology

The methodology consists of 3 main parts: data preparation, feature selection, and estimation.

Data preparation

First, we uploaded a CSV file with features and target values into the pandas data frame

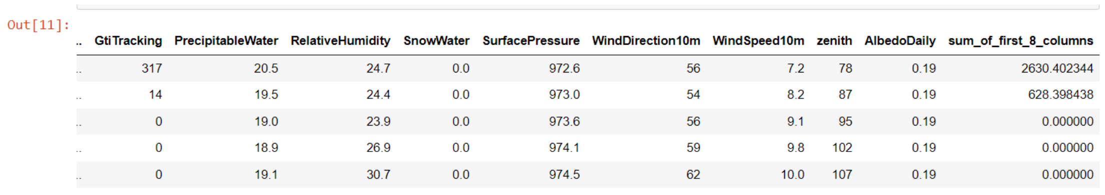


	a1	a2	a3	a4	a5	a6	a7	a8	irradiation	temperature	...	GtiFixedTilt	GtiTracking	PrecipitableWater	RelativeHumidity	SnowWater	SurfacePressure
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-2.9	...	0	0	3.5	68.9	0.2	987.5
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-4.0	...	0	0	3.8	68.0	0.2	987.8
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-4.0	...	0	0	4.2	66.6	0.2	987.8
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-5.1	...	0	0	4.5	64.9	0.3	987.8
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-5.1	...	0	0	5.0	65.8	0.3	987.7

Figure 1. DataFrame visualization

The shape of our data is (2184, 32). Using the df.isnull().sum() method we checked whether any of the rows contained missing values to avoid problems in the final steps.

Then we performed data manipulation to combine the first 8 columns in Figure 1. Into one column that will be the target value:

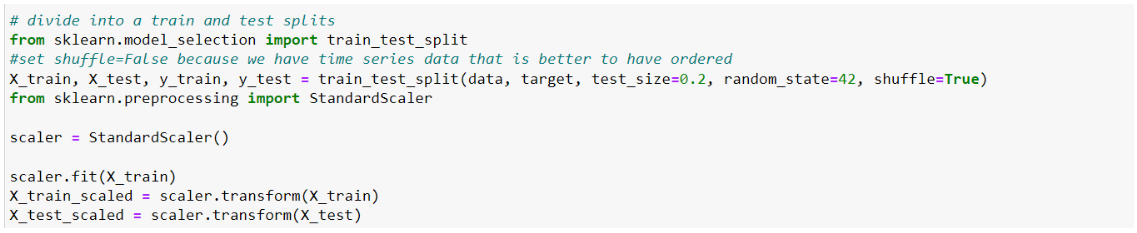


Out[11]:

..	GtiTracking	PrecipitableWater	RelativeHumidity	SnowWater	SurfacePressure	WindDirection10m	WindSpeed10m	zenith	AlbedoDaily	sum_of_first_8_columns
..	317	20.5	24.7	0.0	972.6	56	7.2	78	0.19	2630.402344
..	14	19.5	24.4	0.0	973.0	54	8.2	87	0.19	628.398438
..	0	19.0	23.9	0.0	973.6	56	9.1	95	0.19	0.000000
..	0	18.9	26.9	0.0	974.1	59	9.8	102	0.19	0.000000
..	0	19.1	30.7	0.0	974.5	62	10.0	107	0.19	0.000000

Figure 2. The last column is the target value

In Figure 3. We applied normalization to avoid feature bias.

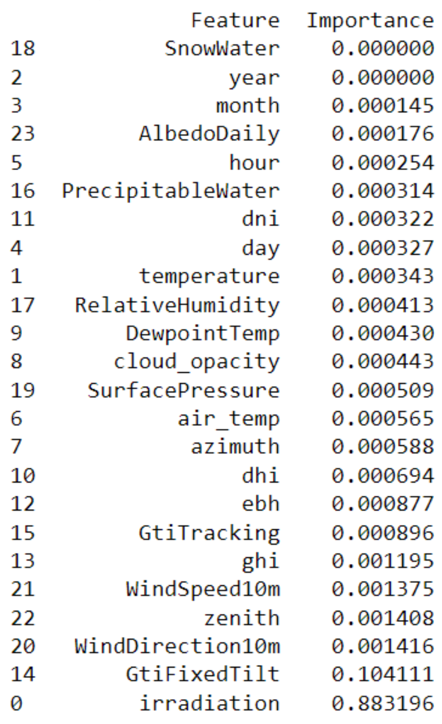


```
# divide into a train and test splits
from sklearn.model_selection import train_test_split
#set shuffle=False because we have time series data that is better to have ordered
X_train, X_test, y_train, y_test = train_test_split(data, target, test_size=0.2, random_state=42, shuffle=True)
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(X_train)
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Figure 3. Using StandardScaler() method

Feature selection

In this section, we tried to cut a number of features using the RandomForestRegressor estimator. The results in terms of the importance of the particular feature are shown in Figure 4.



```
              Feature  Importance
18          SnowWater    0.000000
2                year    0.000000
3               month    0.000145
23        AlbedoDaily    0.000176
5                hour    0.000254
16  PrecipitableWater    0.000314
11                dni    0.000322
4                 day    0.000327
1         temperature    0.000343
17   RelativeHumidity    0.000413
9        DewpointTemp    0.000430
8       cloud_opacity    0.000443
19    SurfacePressure    0.000509
6            air_temp    0.000565
7             azimuth    0.000588
10                dhi    0.000694
12                ebh    0.000877
15        GtiTracking    0.000896
13                ghi    0.001195
21        WindSpeed10m   0.001375
22             zenith    0.001408
20   WindDirection10m    0.001416
14        GtiFixedTilt   0.104111
0          irradiation   0.883196
```

Figure 4. Feature importance method in Random Forest Regressor

Here it is shown that the hour feature is marked by low importance. From our domain of knowledge, the hour is important for the estimation of power generation, and that is why we introduced little bias in feature selection. Some other low-ranked features such as SnowWater and year are dropped from the DataFrame.

Estimation

Before applying the estimator, hyperparameter tuning is made with the GridSearch cross-validation. Max_depth, gamma, and learning_rate parameters are introduced into a cross-validation that implements a simple for loop to find the best parameters. Scoring of cross-validation was done using MAE, MSE, and r^2. 5-fold CV hyperparameters are then inserted into the main XGBRegressor estimator. This estimator is accessed using the validation dataset

Numerical Experiments (including results/discussion)

Performing XGBoost Regression on the normalized dataset with tuned hyperparameters showed the following statistical metrics:

MAE: 902.17
MSE: 2249730.02
RMSE: 1499.91
R^2: 0.99

Here is a visualization of the prediction:

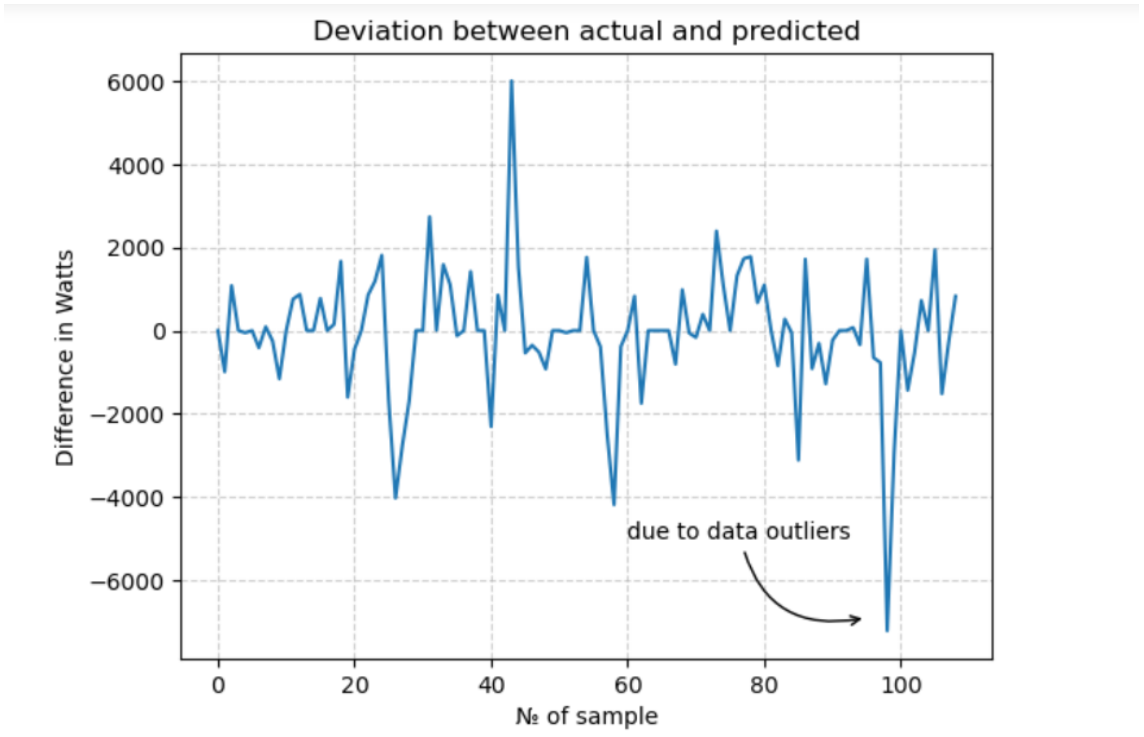


Figure 5. The deviation between actual and predicted for each test sample

From the graph, we can observe some large outliers and error concentrations in the [-2000:2000] range. It is also supported by the Gaussian distribution:

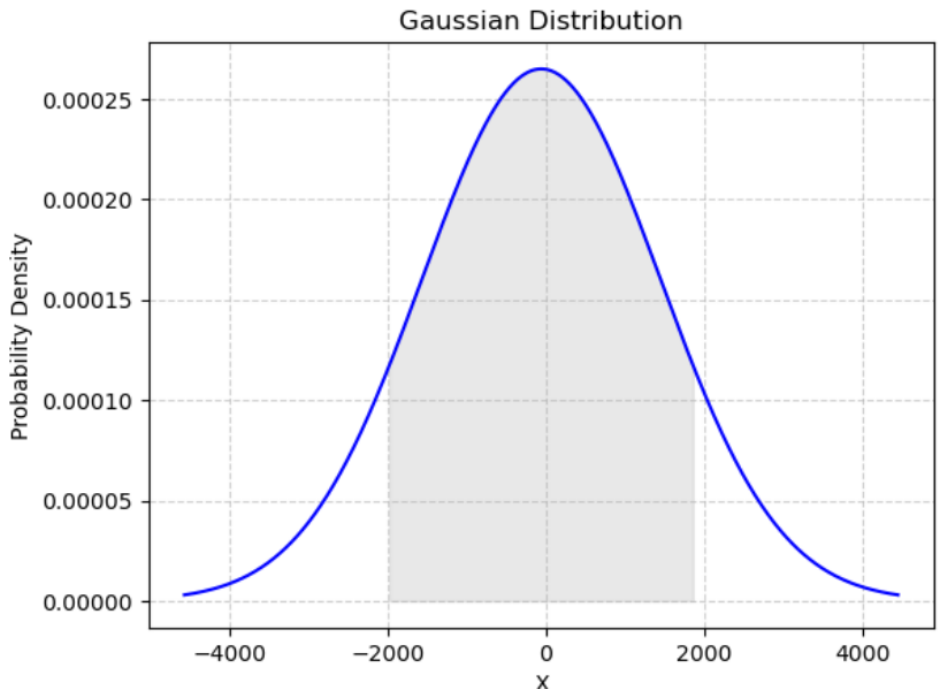


Figure 6. Gaussian distribution. Gray represents 80% of the error

It is worth noting that the larger the deviation between actual and predicted the larger the target value itself. For example for the actual value of 2000, the predicted was 2200 with a deviation of 200. But for the actual value of 42000, the predicted was 36000 with a deviation of 6000.

From the results we can conclude 3 main things:

1. The model shows a strong correlation between features and target values explaining most of the dependence
2. Model rarely but periodically shows high prediction error due to lack of data over the entire yearly time frame` (we have only 6 month period).
3. A more consistent and accurate dataset, model will demonstrate less of this deviation
4. Because of the validation dataset, the model overfeeding parameter was mitigated

Conclusion

PV solar panels can produce the necessary amount of energy with much less air pollution. However, predictability was the main disadvantage of this energy generation type. Using simple machine learning algorithms it is possible to pretty decently predict power generation for a particular hour within 10% which is in most cases considered fairly accurate. This model is trained on insufficient data for a period of 6 months. Hence, results should not be extrapolated for other times and dates. For further studies, the larger dataset should be tested on this model.

References

[1] R. Ahmed, V. Sreeram, Y. Mishra, and M. D. Arif, "A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization," Renewable and Sustainable Energy Reviews, vol. 124, p. 109792, 2020.