**A Machine Learning approach on the solar plant power yield predictions.**

Pol Abadia Conejos

**Abstract—** Climate change has become a number one problem in today’s society. To solve this huge environmental problem, it has become important to come up with solutions of diverse nature.

Energy is a resource fundamental to every human activity, from the electricity used to power up whole cities and countries, to the manufacturing of the materials and goods that we use on our daily. As such, it should be crucial to tackle down the way we produce it. One of the many current promising solutions is the technology of solar powered energy.

Solar plants are becoming more and more widespread. However, as these grow in size and number, it becomes more difficult to manage their performance, making it also more intricate to incorporate them into the power grid, where they must meet some standards in order to be compatible with the whole electric infrastructure. The main reason for this is the fact that, unlike fossil fueled power plants, solar plants are subject to the often-chaotic behaviour of weather.

─────── ♦ ───────

# 1 Introduction

The study drew its data from Kaggle. The dataset consisted of two pairs of dataframes from two different solar plants located in India. For the sake of simplicity and lack of time, the study is only focused on one of the power plants, so from now on this article will only cover the first pair of dataframes.

Each dataframe is a time series recorded over a period of 34 days at an interval of 15 minutes from each sample.

The first dataframe contains data about the power output of the solar plant at each one of these 15 minute intervals, as well as the daily accumulated and total power yield until a given moment.

As for the remaining dataframe, it contains the weather conditions at the plant: solar irradiation, ambient and modular temperature (the last one being the temperature inside the solar panels). For more information about the dataset look into the link provided in the references above.

The aim of this study is to be able to predict the power output of the solar plant two days ahead. This output can be predicted by means of DC or AC current, both of which appear logged into the dataset. Since AC current is considered the standard worldwide, we’re choosing the latter as the main target.

# 2 Work method

## 2.1 Exploratory data analysis

To start the research, we dive into some exploratory data analysis to gain an insight about the data.

The solar plant can be dissected into 22 distinct solar panel arrays, with each array’s generated power converted to AC through local inverters. So the first thing to do was to separate the dataframe containing all the power data into a list of 22 dataframes so that each dataframe is associated to an inverter.

Plotting the power values into a graphic, a sinusoidal cycilic pattern emerges.

Imagen que contiene Gráfico

Descripción generada automáticamente

Fig. 1. Plot of the power output of three different inverters, each color corresponding to a distinct inverter.

However, even though the global tendency is sinusoidal, locally one can observe white noise present in the data, which could be associated to clouds or similar natural phenomena.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Fig. 2. White noise present in the power generation of an inverter

## 2.2 Feature engineering

We move on to an essential part of this study, that is molding our available data into a dataset with which later on our models can catch better the trend of the data.

Three main techniques are employed in that process.

First of all we merged the weather dataframe to the power yield dataset, that could give the models more information about future values since higher solar irradiance and temperature translates into more power output.

After this, we split the datetime feature into different columns for months, days, hours and minutes. That is to make a dataset that only consists of numerical values and to provide a seasonal/ cyclic trend to the data.

Last, we want to apply a technique common in the study of time series, which is called lagging features. The way this works is to make N new columns where we insert the n-th previous values relative to a sample T.

In this process we also removed unnecessary and unmeaningful features from the dataset.

# 3 Model and metric selection

Several models were implemented to predict the power output.

To test their performance, we split each of the inverters’ datasets into a train set and test set.

In the usual machine learning procedure, one would shuffle the data at the moment of splitting it into the train and test sets. Nonetheless, this can’t be done when working with time series, given that our goal is to predict values of the events that only happen after the event of our samples. Doing so, one could still obtain a reasonably high score on the test, since after all, the test set is data with known values, yet treated as we didn’t know them. But when predicting actual unknown future values, that would only confuse the models, leading to nefast predictions.

## 3.1 Evaluation metrics

The Mean Absolute Error (MAE) (1) was chosen as an evaluation metric as it is not altered by outliers and can offer a more intuitive depiction of the results of the models.

Another useful metric is the MAPE (2) as it shows the relative error with respect to the real value. However, this metric is only applied for daylight samples, since otherwise would equal 0 and the MAPE metric would yield infinite values.

## 3.2 Employed models

As said earlier, 4 models were implemented, even though one of them was left behind for lack of time and resources, as its execution time was very high and results were poor.

So the three remaining proposed models are XGBoost and sklearn’s AdaBoost and neural network based Multi Layer Perceptron. Also sklearn’s GridSearchCV function was used to find the best parameters for the chosen algorithms.

|  |  |  |  |
| --- | --- | --- | --- |
|  | MAE | MAPE | GridSearch score |
| AdaBoost | 65.63 | 0.45 | 0.79 |
| XGBoost | 92.67 | 0.61 | 0.58 |
| MLPreg | aprox. 240 | 1.003 | 0.011 |

The scores in the benchmark clearly puts AdaBoost with an advantage over the other two. Nonetheless, making some changes in the lagging features setting we were able to observe a similar performance with XGBoost.

# Forma, Flecha Descripción generada automáticamente

Fig. 3. XGboost’s yielded power predictions (orange) overlapped with the real power values (blue).

Forma, Flecha

Descripción generada automáticamente

Fig. 4. AdaBoost’s yielded power predictions (orange) overlapped with the real power values (blue).

# 4 Conclusions

We were able to catch the subtleties and seasonality dynamic of the data fairly well with fast an easy to use machine learning algorithms such as XGBoost and AdaBoost. These two provide a good balance between resource/time consumption and accurate enough results. Clearly, they are an excellent first choice when diving into complex datasets.

However, much of the noise goes unnoticed, and for more accurate predictions one should seek the usage of far more complex models like neural networks.

**references**

[1] https://www.kaggle.com/datasets/anikannal/solar-power-generation-data/data