Multi-Plant Photovoltaic Energy Forecasting Challenge: Second place solution

Clément Gautrais[1], Yann Dauxais[1], and Maël Guilleme[2]

[1] University of Rennes 1/Inria Rennes clement.gautrais@irisa.fr
[2] Energiency/University of Rennes 1

Abstract. This paper presents the approach we took to solve the Multi-Plant Photovoltaic Energy Forecasting Challenge for ECML/PKDD 2017. The approach we took granted us the second place of that challenge. In the paper, we will present how we moved from standard regression techniques to simple function optimization to tackle the challenge.

1 Introduction

The Multi-Plant Photovoltaic Energy Forecasting Challenge for ECML/PKDD 2017 focuses on predicting power output of photovoltaic (PV) power plants. This challenge is of prime interest, as tackling the main challenges raised by this new energy market (such as grid integration) requires a reliable monitoring of production. The task proposed in this challenge is power forecasting for multiple photovoltaic (PV) plants closely located in Italy.

Forecasting power production of solar panels has been widely studied, as achieving reliable energy production predictions is essential for the development of smarter energy networks. Most of the existing models are able to make reliable predictions for few hours horizon [1–3].

These models usually aim to predict the solar irradiance instead of the solar power production. This is because the solar power production can be derived from the solar irradiance, but may depend on the panel characteristics. Predicting the solar irradiance is therefore more appealing.

Different strategies have been used to predict the irradiance. For short horizons, model based on previous irradiance values perform well, whereas model based on weather features are more reliable for larger horizons [3].

While the irradiance is strongly linked to the power production, the weather conditions, especially the cloud cover, plays a significant role in predicting the power output of a solar plant. Different methods based on standard machine learning algorithms have been developed to take into account weather conditions in the power output prediction [1, 4].

Finally, the dataset provided for the challenge has been previously studied [5]. However, the main focus of [5] is to study the effect of spatial and temporal information on the predicted power output. As we do not know the location of each plant in the challenge dataset, and only have one year of data for each plant (versus more than 2 years in the original dataset), the conclusions drawn in [5] could not really be transposed to the challenge dataset.

2 Method

In this section, we present the dataset that was used for the challenge, as well as the different approaches that we developed to tackle the challenge.

2.1 Dataset and challenge

The challenge dataset contains power production, weather conditions and plant sensors hourly values of three solar plant closely located in Italy for the year 2012. The goal of the challenge is to predict the hourly power production of each plant during the first 3 months of 2013, given the hourly values of weather conditions and plant sensors on these 3 months. The dataset of year 2012 is called the *train set* thereafter and the dataset for the 3 first months of 2013 is called the *test set*.

There are 7 weather conditions variables: cloud cover, dew point, temperature, humidity, pressure, wind bearing and wind speed. We also have values for 2 plant sensors: irradiance and temperature. For the year 2012, we also have the power output sensor values. The plants are active between 2 a.m and 8 p.m, which yields a total of 19 daily measures for each feature and plant.

The performance of the power output forecasting is evaluated with the Root Mean Square Error (RMSE) of the normalized power output.

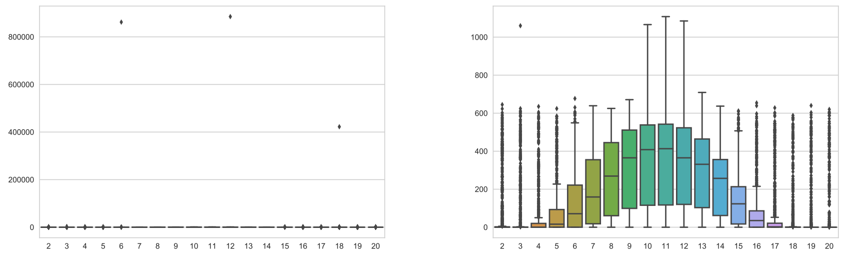
The challenge format allowed 5 submissions on a validation set. This validation set contains 10% of the test set and provides the RMSE of the submission on this set. This allowed us to test different approaches and get some feedback from the test set. In the end, only one model is sent for the final submission.

2.2 Preprocessing

The first step that we performed on the dataset was to normalize the power output values. Before computing the maximum and minimum values for the power output, we filtered possible outliers. Figure 1 allows us to detect graphically the most aberrant outliers. There were 3 outliers with aberrant power output values equivalent to a production of 885.12, 862 and 422.26 kW per hour. Once we removed these values, we found that the maximum power output was 1108 W.

Figure 2 shows the correlation between the power production and three features in the training set. Those features are weather pressure, plant temperature and plant irradiance and the plots show the correlation for March and September. We can see on those plots that the power is more correlated with the plant features and that it is highly correlated with the irradiance. For September, the Pearson correlation coefficient between the irradiance and the power is equal to 0.95, that indicates an almost perfect correlation. On the other hand, it seems to be difficult to use the weather pressure for the power prediction, as the Pearson coefficient is lower than 0.1 on March and September. With this knowledge, it seems to be easy to predict the power from the date and the irradiance. A second information that we can see on those graphs is the occurrence of numerous irradiance values equals to 0, while the power output is not null. As those 0 are

Fig. 1. The power boxplot distributions per hour. The left plot corresponds to the raw data before preprocessing. The right plot corresponds to the data without the three outliers easily observable on the left plot.



uniformly distributed on the power values, it could indicate an anomaly in the irradiance sensors. Since the irradiance is the best feature to predict the power production of a plant, we can suppose that the others features will be more useful to predict the power production when we do not have information on the irradiance *ie.* when the irradiance equals to 0.

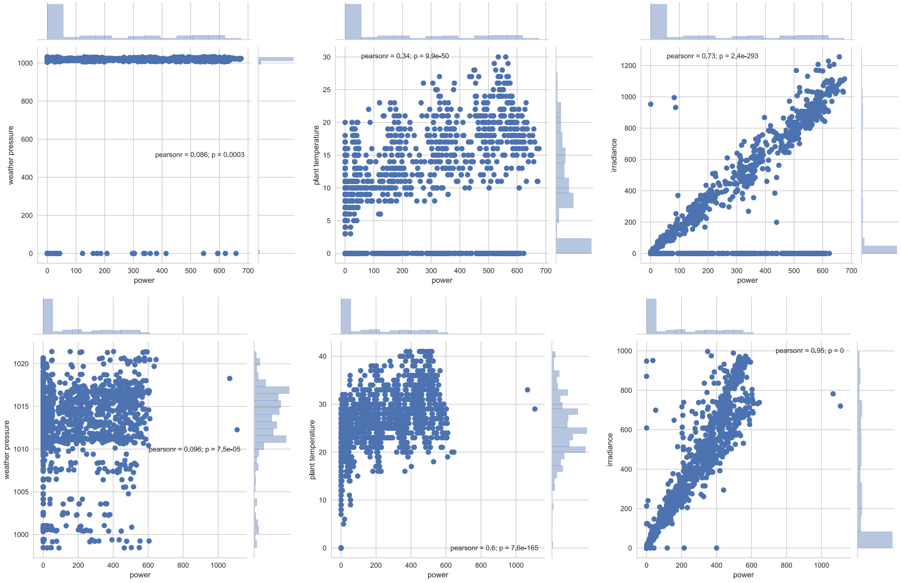
2.3 Initial model

From the first analysis of the features and from the known relation between irradiance, temperature and power output [1], we decided to develop a model that predicts the power output from the irradiance and temperature plant sensors. We developed a model for each hour of the day, that uses the irradiance et temperature features of the whole day. While this setting might appear impractical as we use future values of irradiance and temperature, the challenge allowed for the development of such method, as the irradiance and plant temperature are known for the first three months of 2013.

Linear regression The first type of model that we tried was linear regression. We used the Python library *scikit-learn* [6]. As we have many potential features (38 in total) and that not all of them are likely to be relevant, we used a regularized version of the linear regression called Lasso regression [7]. This regularization influences the learning of the linear regression parameters, such that only few features are used to make a prediction in the learned model.

To validate the learned models, we used a 10 fold cross validation on the data from 2012. We did not take into account the date in the split, as we solely rely on daily features, and do not rely on data from other days. The results are presented in Table 1. While there is no model that is clearly dominating the other on both RMSE and adjusted R^2, we decided to choose a model that had a good compromise between both measures. We therefore used a Lasso regularization with $\alpha = 0.1$.

Fig. 2. The correlation between the power production and three features: weather pressure, plant temperature and plant irradiance. The first line of plot corresponds to the correlation of those features in March. The second corresponds to September.



Neural networks The second type of model that we tried was neural networks. We used the Python library *keras* [8]. At the opposite of the first approach, we used neural networks to learn from all the features. There are numerous ways to construct a neural network, we limited our research on sequential neural network using the optimizer adam. The activation function used by each node is the rectified linear unit (ReLU). Obviously, the loss function to optimize is the mean squared error. The number of epoch for the training is set up to 2000 and the batch size to 12.

As the last 3 months of 2012 were similar to the first 3 months 2013 on many features, we decided to build our model on the end of 2012 *eg.* after October 14. As for the previous method, a 10 fold cross validation was used.

Our first model contained 3 neuron layers and 2 dropout layers. The first layer was a classic input layer with as many neurons as features. The last layer was a classic output layer with as many neurons as power production hours. The other neuron layer was an hidden layer with a number of neurons equals to 60% of the number of inputs. The two dropouts were between the input an the hidden layer and between the hidden and the output layer. Table 2, shows the result of this model as attempt 1.

The second attempt was done to test the usefulness of the hidden layer. We then removed the hidden layer and a dropout layer and it seemed to improve the efficiency of the model. The four other attempts tried to optimize the parameter of the dropout layer. The results lead to a model similar to the linear regression

Table 1. Root Mean Squared Error (RMSE) and adjusted R^2 for different regularized linear regression. Standard deviations are computed on a 10 fold cross-validation. Lower RMSE is better, higher adjusted R^2 is better.

Linear model	α value	Mean RMSE	Std RMSE	Mean Adj. R^2	Std Adj. R^2
Lasso	0.1	0.0647	0.0052	0.7595	0.0360
Lasso	1	0.0665	0.0051	**0.7667**	0.0402
Lasso	10	0.0843	0.0041	0.647	0.0384
Ridge	0.01	**0.0646**	0.0042	0.693	0.047
Ridge	0.1	0.0659	0.0046	0.674	0.044

but using an automatic selection of the best features. The fact that the hidden layer performed worse than a linear model could be explained by the fact that the power output is strongly correlated with the irradiance, and by the small dataset size.

Table 2. Mean RMSE and standard deviation of the RMSE for each attempt of neural network model. Each neural network model is described by its number of hidden layer and the parameter of the dropout layers.

Attempt	Hidden layer	Dropout	Mean RMSE	Std RMSE
1	1	0.1	0.0969	0.0100
2	0	0.1	0.0722	**0.0096**
3	0	0	0.0802	0.0154
4	0	0.2	0.0824	**0.0096**
5	0	0.05	0.0717	0.0109
6	0	0.07	**0.0685**	0.0111

We then sent the optimal model found by each techniques, to get a first feedback from the validation set. This yielded an RMSE of nearly 0.21 for the linear model and nearly 0.22 for the neural network which was larger than what we expected.

2.4 First hypothesis: the normalization

Given that our first submission had a larger RMSE than what we expected, we tried to understand what could cause such a difference between the error on the validation set and the one we obtained during the cross-validation. A first hypothesis is that the normalization in the test and train sets is different. Indeed, if the maximum power value is different in both sets then the model learned on the train set predicts a normalized output power that is based on the train set maximum power value, whereas it is compared with a power value that is normalized by the test set maximum power value.

To test this hypothesis, we sent a model where all power predictions are equal to 0. By doing so, the RMSE we get from the validation is equal to the mean of all squared power values. Then, we can compare this value with the mean of all squared power values in the train set. If both values are similar, then we can assume that the normalization does not explain the large difference in RMSE for the first model.

The mean of the squared power values of the three first months in the train set is equal to 0.22244, this of the three last months is equal to 0.19388 and the value we obtained from the submission on the validation set was 0.20676. We can see that those values are quite similar, which means that the normalization is not the main cause of the difference of RMSE in the first model. However, we noticed that sending this model predicting an absence of production everyday was performing quite well compared to the other approaches (it was ranked second in the partial leader board). We kept this observation in mind but we decided to look at another hypothesis that could explain the difference: the fact that the irradiance is not linked to the power output in the test set, like it is in the train set.

2.5 Second hypothesis: train and validation sets have different irradiance properties

The first models that we submitted were mainly relying on irradiance to predict the power output. One possible explanation of the difference between the RMSE in the train and validation sets is that the relation between power output and irradiance is different in both sets. In that case, this means that the properties of the train and test sets are different, which could explain why the model learned on the train set performs poorly on the test set.

To test this hypothesis, we developed a simple model to predict the power output: we simply normalized the irradiance and used it as our predicted normalized power output. By doing this, we are testing the correlation between the irradiance and the power. In the train set, this simple method yields an RMSE equal to 0.08, which indicates a strong correlation. The submission of this model on the validation yielded a value of 0.22 for the RMSE. As one can see, both values are quite different, which suggests that the irradiance and the power output do not share the same properties in the train and test sets.

From that observation, we then saw 2 choices: developing a model based on all features except irradiance, or develop a simple model that tries to optimize the RMSE and guarantee a good ranking. We ended up choosing the last solution, as we only had one submission remaining and betting on a new model solely based on weather features was hazardous.

2.6 First solution model

To optimize the RMSE we first decided to predict a single power output for all days and hours. While this choice may appear over simplistic, it allows us to find the value that minimizes the RMSE using only one submission. Let us first define $s_{i,j} = \langle s_{i,j,2} \dots s_{i,j,h} \dots s_{i,j,20} \rangle$, with $s_{i,j,h}$ the normalized power output of plant i

for day j at hour h . The RMSE becomes $\sqrt{\frac{1}{N} * \sum_{i,j,h}(s_{i,j,h} - \widehat{s_{i,j,h}})^2}$. As we are predicting a single value for all power outputs, we have $\widehat{s_{i,j,h}} = a$. In that case, minimizing the RMSE is equivalent to minimizing $\sqrt{\frac{1}{N} * \sum_{i,j,h}(s_{i,j,h} - a)^2}$. The minimum of that function is reached for $a = \frac{1}{N} * \sum_{i,j,h} s_{i,j,h}$, that is when a is the mean power output.

The mean of the power output can be obtained by sending two submissions predicting a single power output. As we had one submission remaining and one submission result from the model that predicted no power output, we were able to compute the optimal value a. It should be noted that for the value to be optimal on the test set, we make the assumption that the validation set (10% of the test set) is representative of the whole test set. If this is not the case, the value a will overfit the validation set and possibly perform poorly on the whole test set.

Given a value a_1, its associated RMSE b_1 on the validation set and b_0 the RMSE for the model that always predicts 0, we have $a = \frac{b_0^2 + a_1^2 - b_1^2}{2 * a_1}$.

From the value of a, we are also able to compute the associated RMSE. The RMSE we were expecting was granting us forth place in the partial leader board. From this observation, we decided to refine our solution to hope for a higher ranking, even though we did not have any submission left.

2.7 Final solution

From Figure 1, one can see that a single power output for the whole day is not appropriate. From this observation, we decided to refine our basic method by predicting a different value for each hour, instead of having a single value for all hours. However, the main challenge lies in learning what value to output for each hour.

A simple idea would be to compute the average power output for each hour on the first three months of 2012. Indeed, we can expect the average power output to be similar in early 2012 and early 2013. However, the power output for the first three months of 2012 are heavily polluted by missing values. We decided not to rely on these polluted values and decided to instead use the last three months of 2012 to learn our model. While analyzing the characteristics of the power output for the last three months of 2012, we noticed that its power output had similarities with the first three months of 2013. For example, the mean squared power output are similar in both quarters.

When computing the mean power output value for each hour on the last three months of 2012, we obtained a prediction similar to the one depicted in Figure 1. We also normalized our predicted power output, so that its mean squared power is equal to the mean of the squared power output in the validation set. The normalization simply consists in dividing all hours power output prediction by the same constant. This solution scored 0.22345 on the final leader board, which granted us second place.

3 Conclusion

In this paper, we have presented how we tackled the Multi-Plant Photovoltaic Energy Forecasting Challenge. We first started by studying previous work predicting solar power output. Then, we did a basic analysis of the dataset, noticing that the power output was highly correlated with one feature: the irradiance. We developed different models based on the irradiance and on other features to predict the power output. After a few submissions on a validation set, we concluded that the correlation between the irradiance and the power output was different in the train and test sets. We therefore decided to directly optimize the error function, by using constant values for the power output prediction. The final iteration of that method granted us the second place in the challenge.

Even though, our method scored well in the challenge, the limited number of submissions made us choose a simple solution over potentially better, but more complex ones. For example, it would be interesting to see if the power output can be reliably predicted using all features, apart from the irradiance. Model based on time series analysis could also have been interesting, as there is some periodicity in the power output of a solar plant.

References

1. Shi, J., Lee, W.J., Liu, Y., Yang, Y., Wang, P.: Forecasting power output of photovoltaic systems based on weather classification and support vector machines. IEEE Transactions on Industry Applications **48**(3) (2012) 1064–1069
2. Pedro, H.T., Coimbra, C.F.: Assessment of forecasting techniques for solar power production with no exogenous inputs. Solar Energy **86**(7) (2012) 2017–2028
3. Bacher, P., Madsen, H., Nielsen, H.A.: Online short-term solar power forecasting. Solar Energy **83**(10) (2009) 1772–1783
4. Sharma, N., Sharma, P., Irwin, D., Shenoy, P.: Predicting solar generation from weather forecasts using machine learning. In: Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on, IEEE (2011) 528–533
5. Ceci, M., Corizzo, R., Fumarola, F., Malerba, D., Rashkovska, A.: Predictive modeling of pv energy production: How to set up the learning task for a better prediction? IEEE Transactions on Industrial Informatics **13**(3) (2017) 956–966
6. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830
7. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) (1996) 267–288
8. Chollet, F., et al.: Keras. https://github.com/fchollet/keras (2015)