Solar Power Generation Forecast Model for 24 Hours Ahead Prediction

Juan Caridad CS458

I. Abstract

Abstract–This paper will go over the creation process of a Solar Power Generation Forecast Model to predict over twenty-four hours. Using data gathered from three solar power plants from Australia, the aim is to create an accurate prediction for the power that will be outputted the next day. With the data gathered, the aim is to train and test the data using various models and regressors to attain the best accuracy for the prediction.

Using the programming language Python, various built in functions and modules help gain the best possible prediction. Using these modules the data is thoroughly split between the train and test dataset, to further fit it towards regression models and classifiers to see which ones attain the best score. These scores are then evaluated, giving the accuracy of the twenty-four hour predictions, indicating if the model created was accurate. In this paper the full step by step process of this program model will be gone over.

II. Introduction

With the large amount of data that was given, the first step was to split between the three different power plants. The problem was to solve how to utilize this data to build a Solar Power Generation Forecast Model to predict the power output for the next twenty-four hours. Solving this problem was to see if it was possible to get a prediction for the power generated while keeping the scores accurate.

The first thing was to solve how to split this data so that it would be easier to work with. Given the fact there were three different power plants, figuring out the best way to split the data to normalize, train and test the data was the first step.

After that problem was tackled, the next step was to figure out which type of classifier or regressor would give the best output of data. Based on the training and testing dataset that is created, fitting that data into a classifier or regressor, keeping in mind what would give the least bias, but also keeping in mind underfitting and overfitting. After that the full problem should be solved.

III. Background and Related Work

One paper that I came across was about a one day ahead hourly forecasting for solar generation. This paper created this prediction model by using cluster analysis as well as ensemble models. Like me they tested with Random Forest Regression, but they also added more hyperparameters to gain better accuracy of their data. Prior to using the regressor, they decided to cluster the data based on the weather regimes to improve their forecasting performance. Also, by using ridge regression to calculate the weights of each weather regime, they are able to remove the manual weights. By using cluster analysis to attain better forecasting performance, as well as the use of ridge regression to get weights for each weather regime, and lastly Random Forest Regression which is one of the classifiers that get the best accuracy, they are able to attain an accurate forecast model(Pan).

IV. Methodology

To begin, the first problem needed to be solved was to figure out how to train the data that was given to us. Given parameters of how to split the training and testing dataset, using Python's various modules, the data was read into a dataframe, split and sent back out into two different datasets. The parameters for the training dataset was from “20120401 01:00 to 20130701 00:00” and for the test dataset it was “20130701 01:00 to 20140701 00:00”.

After the datasets were split into a training and testing dataset, the next step was how to make sure the data was usable. By normalization and standardization to process the data and check missing values, it made sure that the data was ready to be used. After, the data was then split into the three zones to gain a better accuracy of the power that was being predicted.

Then came looking for the best classifier or regressor to fit and test the data. Starting off with a linear regression model, by taking the variables of the data from twenty-four hours prior to the desired data, and the power from my time frame, the data was trained. Then, by using the actual power generation and predicted power generation at time t to evaluate the model. After looking at the predictions of this data, and the accuracy scores, I came to a conclusion that it was not the best model to use for this project. This is because, based on the way I normalized the data in order for it to be trained, it didn’t give the best output. After that many different regressors and classifiers were tried like Nearest Neighbors Regression, Ridge and lastly the one that was chosen, Random Forest regressor.

The Random Forest Regressor uses an ensemble learning method, which means multiple machine learning algorithms are used to combine predictions. Moreover, the reason Random Forest is useful in this case is because due to the large amount of data given, by increasing the size of the “forest” it would give better accuracy. As shown below in Figure 1, the parameters of the Random Forest Regressor are given. When looking at the parameters, n_estimators is what sets the amount of trees to be created in the forest. Having a higher amount of trees should lead to lower bias, meaning that the lower the bias the less errors in our data we should have. For the next parameter, max_depth I decided to have a higher max depth even though the higher depth could lead to overfitting it also leads to lower bias. The random state parameter is set to zero to

make sure the outcome of data is consistent across each call when fitting the data. Lastly, for min_samples_split I decided to go with a higher split which makes things more clustered and could lead to underfitting, but since I had a higher depth which could lead to overfitting, I felt

```
RanForReg1 = RandomForestRegressor(n_estimators = 300, max_depth = 15, random_state = 0,  min_samples_split = 8)
RanForReg1.fit(Xtrain1, yTrain1)
```

that the two together would offset.

Figure 1: Line of code showing the model chosen and the parameters

V. Evaluation Results

Once the data was trained, I grabbed the predictions from the test data that I created and I put the test data alongside the prediction data. Placing these into a dataframe, I was able to show the actual versus the predicted value by printing a section of each dataframe. In the figures below, the output of these dataframes over a twenty-four hour period is shown.

```
ZONE 1
December 31, 2013 21:00 PM, Actual Power:  0.159935897 Predicted Power:  0.2547665703216161
December 31, 2013 22:00 PM - January 01, 2014 21:00 PM, Forecast power:
             A         P
4413  0.289487  0.482962
4414  0.643846  0.593774
4415  0.532500  0.642309
4416  0.706218  0.568053
4417  0.409615  0.460485
4418  0.173462  0.425319
4419  0.151923  0.336971
4420  0.106410  0.275665
4421  0.095769  0.231845
4422  0.066218  0.123139
4423  0.024808  0.062172
4424  0.013141  0.006542
4425  0.000385  0.001739
4426  0.000000  0.001273
4427  0.000000  0.006012
4428  0.000000  0.005940
4429  0.000000  0.006318
4430  0.000000  0.008367
4431  0.000000  0.047765
4432  0.000000  0.065816
4433  0.000000  0.069070
```

Figure 2: Zone 1 dataframe showing actual vs predicted data from 12-31-13 to 01-01-14

```
ZONE 2
December 31, 2013 21:00 PM, Actual Power:  0.099676113 Predicted Power:  0.2547665703216161
December 31, 2013 22:00 PM - January 01, 2014 21:00 PM, Forecast power
             A         P
4413  0.239271  0.299168
4414  0.434960  0.349161
4415  0.470769  0.438256
4416  0.714231  0.600062
4417  0.868077  0.392809
4418  0.880769  0.330425
4419  0.125223  0.226992
4420  0.091862  0.219794
4421  0.673421  0.156554
4422  0.496761  0.145900
4423  0.028421  0.095932
4424  0.017976  0.043550
4425  0.000000  0.026880
4426  0.000000  0.052898
4427  0.000000  0.086934
4428  0.000000  0.094082
4429  0.000000  0.104676
4430  0.000000  0.091308
4431  0.000000  0.108523
4432  0.000000  0.187751
4433  0.000000  0.209176
```

Figure 3: Zone 2 dataframe showing actual vs predicted data from 12-31-13 to 01-01-14

```
ZONE 3
December 31, 2013 21:00 PM, Actual Power:  0.1746 Predicted Power:  0.2547665703216161
December 31, 2013 22:00 PM - January 01, 2014 21:00 PM, Forecast power:
             A         P
4413  0.352550  0.326240
4414  0.676025  0.519177
4415  0.463825  0.647037
4416  0.739125  0.579771
4417  0.504350  0.540614
4418  0.239600  0.526963
4419  0.177375  0.346182
4420  0.119625  0.257227
4421  0.073975  0.212698
4422  0.072050  0.110476
4423  0.031550  0.049496
4424  0.017425  0.002873
4425  0.000700  0.000399
4426  0.000000  0.000392
4427  0.000000  0.000396
4428  0.000000  0.000406
4429  0.000000  0.001243
4430  0.000000  0.022342
4431  0.000000  0.229010
4432  0.000000  0.265399
4433  0.000000  0.245936
```

Figure 4: Zone 3 dataframe showing actual vs predicted data from 12-31-13 to 01-01-14

After, by using RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error), I was able to get the error of my model. Using built in functions from Python, all that was necessary for me to do was to grab the errors of each zone and append them to a list. Afterwards, I got the sum of the list and divided it by the length giving me the overall value. This allowed me to get the scores shown in Figure 5 below.

ZONEID	1	2	3	Overall
MAE	.0584329	.09588842	.08117063	.07849732
RMSE	.1122332	.16037859	.13267738	.13509639

Figure 5: Table of MAE and RMSE based on zone as well as the overall error.

VI. Conclusion

From the results and scores, I am able to see that the solar power generation forecast model for twenty-four hours was successful. First looking at the accuracy of the data we get after training the data to the Random Forest Regressor, we see in Figure 6, that the accuracy of the models are less than three percent off from being one hundred percent accurate. This shows that by separating the data into each zone was very accurate for us, and with the use of Random Forest Regressor, we were able to get a model that has almost one hundred percent accuracy. Moving on, looking at our errors from MAE and RMSE, we see that the overall error is less than 8% for MAE and less than 14% for RMSE. Given these error results and accuracy results I can come to a conclusion that the forecast model produces close and almost accurate predictions.

```
Zone1 accuracy:  0.974563865857889
Zone2 accuracy:  0.9752844203870384
Zone3 accuracy:  0.9751745182450151
```

Figure 6: Accuracy score for each zone/model created.

VII. Citation

Pan, Cheng, and Jie Tan. "Day-ahead hourly forecasting of solar generation based on cluster analysis and ensemble model." *IEEE Access* 7 (2019): 112921-112930.