ENGR 301
Engineering Applications of Data Science

Laboratory 3
Topic: Environmental Engineering; Subtopic: Solar Energy

Introduction: In the first lab, we explored wind power using exploratory data analysis. In this assignment we will focus on another type of renewable energy, solar power. In solar panel devices, incoming sunlight is captured and used to produce electricity. In places that can support this type of renewable energy system, modules consisting of various numbers of solar panels are used to capture the sunlight. Variable direct current (DC) is output from the module, and a solar inverter is used to convert this into alternating current (AC) which can be fed into the electrical grid. A schematic and picture of such a solar power plant is shown below.

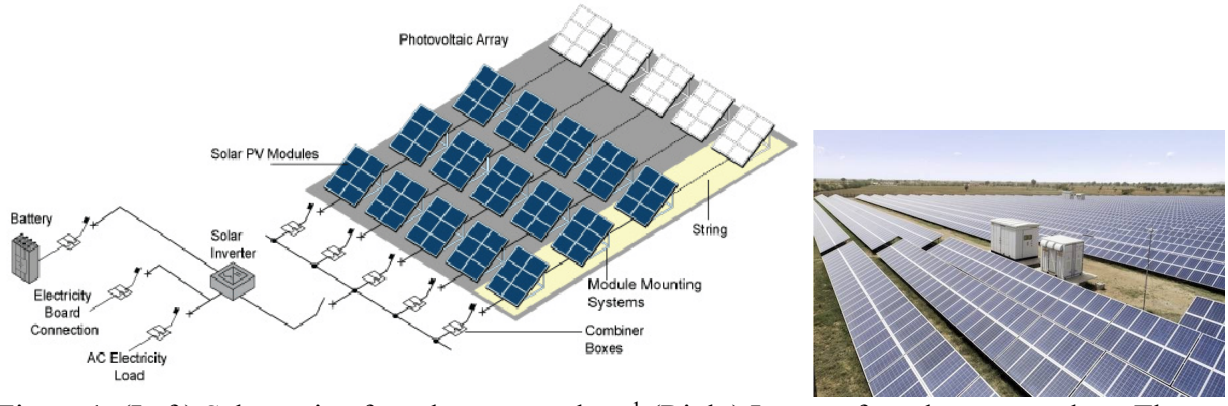

Figure 1. (Left) Schematic of a solar power plant.[1] (Right) Image of a solar power plant. The inverter can be seen in the center.[2]

The goal of this lab is to find which weather features are predictive of power output at a solar power plant. This data you will use was taken from two solar plants in India over the course about a month. You will be provided with four CSV files: two containing information about each plant's power output and two containing information about the weather at each plant.

Discipline Specific Information: The data set contains the following information:

Plant Information

Column Name	Description
DATE_TIME	Date and time of each measurement.
PLANT_ID	Identification number of the plant. Note that there is only information about one plant in each CSV file.
SOURCE_KEY	Identification number of each inverter.
DC_POWER	Quantity of DC power measured by a given inverter at a given time in units of kilowatts (kW).
AC_POWER	Quantity of AC power generated by a given inverter at a given time in units of kilowatts (kW).
DAILY_YIELD	Cumulative sum of the power generation on a given day up to that point in time (kW).
TOTAL_YIELD	Total yield for a given inverter up to that point in time (kW).

Weather Information

Column Name	Description
DATE_TIME	Date and time of each measurement.
PLANT_ID	Identification number of the plant. Note that there is only information about one plant in each CSV file.
AMBIENT_TEMPERATURE	Overall ambient/outside temperature at the solar plant in Celsius.
MODULE_TEMPERATURE	Temperature of a module attached to each sensor panel in Celsius.
IRRADIATION	Amount of solar irradiation (amount of sunlight) during a given time interval in units of Watts per square meter (W/m^2).

Tasks

Note: For full credit, all plots must have axis labels and units on the x- and y-axes, and the discussion must be in complete and full sentences.

Note: When model_name is specified, this means you should replace it with the name of your model.

Part 1 – Exploratory Data Analysis and Data Cleaning

1. Import the appropriate packages (numpy, pandas, matplotlib, seaborn) into a Jupyter notebook. The data here is provided in four CSV files called:

plant1_power.csv
plant1_weather.csv
plant2_power.csv
plant2_weather.csv

Use *pandas* to import each data set into its own data frame in a Jupyter notebook.

2. Which plant (1 or 2) has the higher average daily power yield over the 34 day period? Which plant was on average hotter? Which inverter at each plant produced the highest average daily power yield?

3. It is easier to analyze the data if the power and weather data for each plant is in one data frame.

3a. Note that the plant data and weather data have two different date formats (the plant data has the day first, while the weather data has the year first). We will use the pandas datetime feature to first standardize the format between the two.

Overwrite the "DATE_TIME" column in each data frame by calling "pd.to_datetime()" with two arguments. The first is the column you want to change (i.e., "DATE_TIME"), and the second is letting pandas know what format the date information is in; for the plant data you do this by setting dayfirst = True while for the weather data it is yearfirst = True.

This will standardize the DATE_TIME data across each data set.

3b. We did this because we want to merge the power output and weather data for each plant based on the time each measurement was taken. The data must match exactly for this to occur.

Now create a new data frame for each plant based on the two separate data frames. We will do this using the "merge" command in pandas. For example, for the first plant,

merged_plant1 = plant1.merge(right = weather1, how = 'left', on = 'DATE_TIME')

This will create a new data frame in which the weather columns will be placed to the right of the power columns for each measurement which have the exact same DATE_TIME values.

4. Do your new data frames have any NA values? If so, how many? Drop each row that contains an NA value. Does this still provide you with enough data to do an analysis?

5. What features seem the most strongly correlated in this dataset? You can use a pair plot or correlation table to determine this.

Part 2 – Linear Regression Model Training and Analysis

We will now train a variety of linear regression models to predict the DC power output based on weather information.

1. Let's first investigate the predictive power of solar irradiation on DC power output. Begin with the data from plant 1. First, we must divide our data into a training set and a test set. To do this, import the "test_train_split" from the "sklearn.linear_model" module.

1a. Create two arrays, an x one containing the IRRADIATION data and a y one containing the DC_POWER data. You can do this using the array functionality of NumPy on the appropriate column. (Don't forget to reshape the array).

1b. Create four arrays (x_train, x_test, y_train, and y_test) using the "train_test_split" function on the two arrays from 1a. Put 30% of the data in the test set. Why did we do this?

1c. Now we will create and fit a linear regression model between these two variables. Import "LinearRegression" from the "sklearn.linear_model" module.

Create a new instance of a model (you can choose the model name) by assigning it "LinearRegression()".

Next, fit your model by calling "model_name.fit()" with the x and y training data as the input.

Finally, create a new array of predicted y values by calling "model_name.predict()" with your x test data as the input.

1d. We will now compare these predicted y values to the y values in the test set. Import "r2_score" from the "sklearn.metrics" module. Call the "r2_score()" command y test and the y prediction values as inputs.

What is the R^2 value of your model?

1e. Calculate the residuals by subtracting the predicted y values from the y values in the test set. Create a scatterplot of residuals versus the data in the y test set. How does their spread look? What does this tell you about the validity of your model?

1f. Let's check for overfitting. One quick and easy way to do this is check how well your model performs in predicting the training data. If the two R^2 scores are similar (one from the training data and one from the test data), then it is likely that you avoided overfitting with your model.

You can do this by calling "model_name.score()" with the x training data and y training data as input.

Did you avoid overfitting in this case? Why does this approach let you know if you avoided overfitting?

2. Create a new model and perform the same steps but with the ambient temperature instead of irradiation. What is the R^2 value of this model?

Given these results, is ambient temperature or irradiation a better predictor for the DC power output? Does this make sense to you? Why or why not?

What does this tell you about the model?

Part 3 – Improving the Model with Multiple Linear Regression and Cross Validation

In this part, as in the previous part, the y array (*i.e.*, the desired output) will always be DC_POWER.

1. Create a new x array consisting of both AMBIENT_TEMPERATURE and IRRADIATION. (Note: in this case you don't need to reshape the data). Create new tests and training sets containing 30% of the data. Fit a new linear regression model on this data. What is the R^2 value of this model on the test data?

2. Create a new x array consisting of AMBIENT_TEMPERATURE, IRRADIATION, MODULE_TEMPERATURE. (Note: in this case you don't need to reshape the data). Create new tests and training sets containing 30% of the data. Fit a new linear regression model on this data. What is the R^2 value of this model on the test data?

What did you learn from these two models? How could you use these models for predicting solar power output? Can you think of an application besides solar power where such a model would be applicable?

3. Finally, let's use cross validation to test the robustness of the model with three features (Part 3, Number 2). Import "cross_val_score" from the "sklearn.model_selection" module.

Create a new array by calling and assigning the "cross_val_score()" command with the model name, x training data, y training data, scoring='r2', and cv=5. This will perform a 5-fold cross validation test on your training set, and the array will contain the R^2 values of each cross validation test. What do you notice about the R^2 values?

Figure References

[1]https://www.researchgate.net/publication/323650831_Intelligent_monitoring_and_maintenance_of_solar_plants_using_real-time_data_analysis

[2]https://kenbrooksolar.com/solar-power-plants/mw-solar-power-grid