**Short-Term Forecasting of Solar Radiation Using Machine Learning Techniques**

**Background**

Solar PV has been becoming a more significant part of the electric grid in the United States over the past 20 years and its role is only expected to increase. The Solar Energy Industries Association (SEIA) estimates there were close to 90,000 MWdc of installations in the US in 2020.

![SEIA Cumulative US PV Installations](Images/Cumulative\_PV\_Installtions\_US.png)

Source: [SEIA](https://www.seia.org/solar-industry-research-data)

With more and more solar PV contributing to electricity production in the United States, better forecasting models will be needed to maintain the stability of the electric grid and optimize the utilization of grid resources such as energy storage devices as well as other types of generating facilities.

For this project I have downloaded 30 minute interval weather data from various locations around Pittsburgh, Pennsylvania from the National Renewable Energy Lab's [National Soalar Radiation Database (NSRDB) viewer](https://maps.nrel.gov/nsrdb-viewer/). I will use this data to forecast the Global Horizontal Irradiance(GHI) in Pittsburgh 4 hours ahead of the most recent measurements. GHI is the combination of Direct Normal Irradiance (DNI) and Diffuse Horizontal Irradiance (DHI) and is the most important value to know when calculating the production of a PV array. [Calculating the Energy Yield of Si-Based Solar Cells for Belgium and Vietnam Regions at Arbitrary Tilt and orientation under Actual Weather Conditions]

![GHI vs DNI and DHI](Images/GHI\_DNI\_DHI.png)

Source: [Homer Energy](https://www.homerenergy.com/products/pro/docs/latest/global\_horizontal\_irradiance\_ghi.html)

The locations from which data was gathered can be seen below.

![City Data Sources](Images/City\_Data\_Sources.png)

In this notebook, a few machine learning algorithms will be explored to determine which may be the most useful for the task of forecasting GHI.

**Data**

As stated before, data was acquired from NREL's NSRDB for various locations around Pittsburgh, PA. Data was collected and reported in 30 minute increments. Data was collected for the years 1998-2019. The features of the dataset include the following:

*Hour, Minute, Day, Month, Year, DHI, DNI, GHI, Clearsky DHI, Clearsky DNI, Clearsky GHI, Dew Point, Surface Albedo, Wind Speed, Wind Direction, Relative Humidity, Temperature, Pressure, Global Horizontal UV Irradiance (280-400nm), Global Horizontal Irradiance (295-385nm)*

These features were collected for each of the cities shown above. To prepare the data, the following steps were taken:

1. For each city, append each year’s data to the end of the previous year’s data.

|  |
| --- |
| **City A Data, 1998** |
| **City A Data, 1999** |
| **…** |
| **City A Data, 2019** |

1. Remove the *Year* attribute from each dataframe
2. Concatenate each city’s data along axis 1

|  |  |  |
| --- | --- | --- |
| **City A Data** | **City B Data** | **…** |

1. Remove all but one of the city’s *Hour, Minute, Day,* and *Month* attributes.
2. Convert *Hour* and *Minute* to *Time\_X* and *Time\_Y* where the time is mapped to the unit circle where 0:00 is represented by the coordinates (0, 1), 12:00 by (0, -1) and time progressing in a clockwise fashion.

The purpose of this is to explicitly tell the model that 23:30 and 0:00 are very close to one another, just as 4:00 and 4:30 are.

1. Similarly, convert *Day* and *Month* to *Day\_X* and *Day\_Y* where the day is mapped to the unit circle where January 1st is at the top of the unit circle and the day proceeds in a clockwise fashion. Again, this is to remove discontinuities and to emphasize that March 31st and April 1st are adjacent dates, just as December 31st and January 1st are.
2. Convert *Wind Speed* and *Wind Direction* to *Wind\_X* and *Wind\_Y* where the latter are cartesian coordinates rather than polar coordinates. Again, this is to remove the discontinuity between wind with a direction of 359º and 0º.

**Creation of Features and Targets**

The objective of this project was to predict the GHI in Pittsburgh four hours ahead of time. As for features, however, three different sets of features were used to experiment. One set of features was all current weather, time, and day measurement matched up with a target of the GHI in Pittsburgh 4 hours (8 time steps) ahead. Another set used the same target, but used both the current measurement and the measurements of the previous time step (30 minutes prior). Finally the last set of features used the current measurements as well as the measurements made in the previous 3 time steps (1.5 hours, 1 hour, and 30 minutes prior).

Feature and target set 1:

|  |  |
| --- | --- |
| **Features** | **Target** |
| All city data at timestep 0 | Pittsburgh GHI at timestep 8 |
| All city data at timestep 1 | Pittsburgh GHI at timestep 9 |
| All city data at timestep 2 | Pittsburgh GHI at timestep 10 |
| … | … |

Feature and target set 2:

|  |  |  |
| --- | --- | --- |
| **Features** | | **Target** |
| All city data at timestep 0 | All city data at timestep 1 | Pittsburgh GHI at timestep 9 |
| All city data at timestep 1 | All city data at timestep 2 | Pittsburgh GHI at timestep 10 |
| All city data at timestep 2 | All city data at timestep 3 | Pittsburgh GHI at timestep 11 |
| … | … | … |

Feature and target set 3:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Features** | | | | **Target** |
| All city data at timestep 0 | All city data at timestep 1 | All city data at timestep 2 | All city data at timestep 3 | Pittsburgh GHI at timestep 11 |
| All city data at timestep 1 | All city data at timestep 2 | All city data at timestep 3 | All city data at timestep 4 | Pittsburgh GHI at timestep 12 |
| All city data at timestep 2 | All city data at timestep 3 | All city data at timestep 4 | All city data at timestep 5 | Pittsburgh GHI at timestep 13 |
| … | … | … | … | … |

**Training, Validation, and Testing Dataset Split**

Two splits of each dataset were made. The first was to split the training\_validation set from the testing set. This was done using a test size of 0.2 without shuffling (in order to be able to visualize sequential data when testing and validation were performed). Next, another split was made between the training and validation sets, again with validation size of 0.2 and without shuffling. This splitting process was performed for each of the three datasets.

**Scaling**

Almost all machine learning algorithms require that the data be normalized or scaled in order to perform well. A simple min\_max scaler was utilized in this project. Importantly, so as not to allow any data leakage, one min\_max scaler was fit only on the training data, and then the training and validation sets were transformed using this scaler. These would be the datasets that would be using the select hyperparameters. A second scaler was fit on the combination of the training and validation set and the testing set and train\_val sets were transformed using this scaler. These datasets would be used for the final model training and subsequent testing.

**Model Selection**

**Linear Model with Ridge Regression**

A linear model with ridge regression is a linear model:

Or,

where,

And is the predicted output for a given input, ***x****,* and is selected to minimize the cost function

Or,

Where ***y*** is the vector of actual outputs, *y(i)* for given actual inputs, ***x****(i)*. And

Where is the 1st instance of the 1st attribute, is the nth instance of the 1st attribute, and so on.

The first term on the right-hand side of (EQUATION ##) is the sum of the squared difference between the actual output and the predicted output. While the second term is meant to restrict the magnitude of , which can help in avoiding overfitting the data to the training set.

For a given scalar . is a *n* x *(m+1)* matrix where *m* is the number of features of the data and *n* is the number of instances. The solution to this is shown below:

Note that since,

Then,

which implies,

Therefore,

To find that minimizes this function, set its derivative with respect to equal to zero.

For = 0, this reduces to a basic linear regression. The purpose of the second term in **(EQUATION #)** is to provide regularization, which helps avoid overfitting the parameters to the given data and potentially generalizes better to yet-unseen data. Including this term in the objection function drives high magnitude parameters of towards 0. To see how this happens, we can look at (**EQUATION ##)** for the case where = 0.

Note that is positive semidefinite since for *any* non-zero vector ***a***,

And,

If and only if

Which can only hold if is *not* full rank (since we restrict to be a non-zero vector). So, if is full rank, we can see that is positive definite. Which implies that is invertible. Furthermore, since is real-symmetric, there exist a set of m+1 eigenvectors which span . Therefore, is diagonalizable, which means

s.t.

Where D is a diagonal matrix and P is invertible. The diagonal elements of D are the eigenvalues of and the columns of P are the corresponding eigenvectors which can be selected such that all eigenvectors are normalized and the eigenvalues are ordered such that .

Returning to (**EQUATION ###)**,

Where

(EQUATION ###) could also be rewritten as

Where

And

This implies that is found by transforming ***z*** via a matrix which has eigenvectors equal to those of but has corresponding eigenvalues .

Now, returning to the general case, where ,

Where

Which implies

(EQUATION ###) can also be rewritten as

Where

Which implies that is found by transforming ***z*** by a matrix with eigenvectors equal to those of as well as A, but has corresponding eigenvalues .

Comparing these two methods, we can see that for both cases is obtained by transforming ***z*** along the eigenvectors of . However, during ridge regression, the eigenvalues are all smaller than those during linear regression. The amount each eigenvalue decreases is correlated to the magnitude of the original eigenvalue (). If is small, then will have a large impact and . However, if is large, then will not have much impact and .

Interpretation:

If the data is first scaled by a standard scaler where

Where is the new scaled value of the ith instance of the jth attribute, is the unscaled value of the ith instance of the jth attribute, is the mean of the instances of the jth attribute and is the standard deviation of the jth attribute. Then

And

Where is the Pearson correlation matrix for a population.

The optimal choice of can be made by fitting the training data to ridge regression models with a range of values, calculating the RMSE error for this model on the validation data, and comparing the models. Finally, the optimal choice of can be made, and then a new model can be fit on the combination of the training and validation data. Ultimately the RMSE can be calculated on the testing data.

The plots of RMSE vs alpha parameters are below:

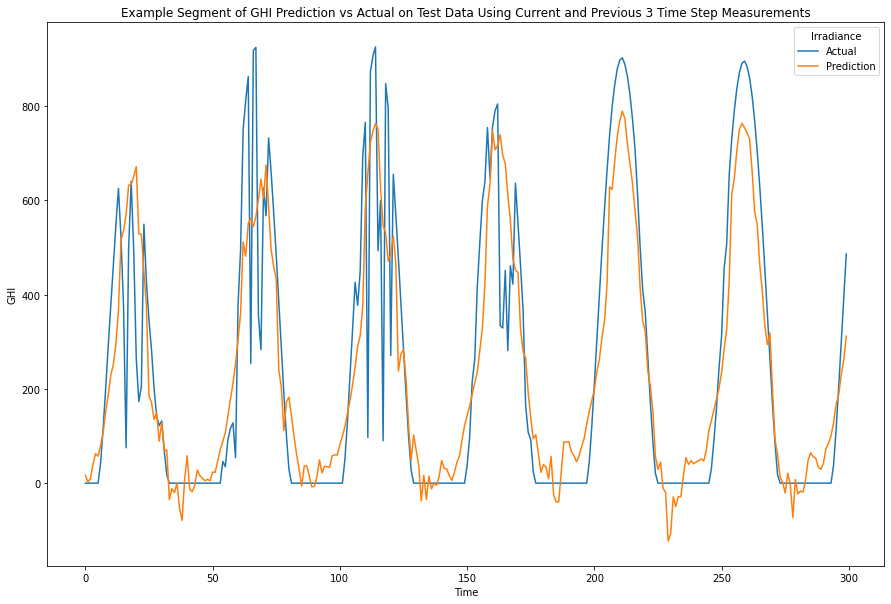
|  |  |  |
| --- | --- | --- |
|  |  |  |

**Fig. ####**

As can be seen, for all of these datasets, the model performed best as a simple linear model without regularization. This is likely due to the large number of examples in the dataset. Therefore, all three models were fit to the combined training-validation set with alpha set to 0, and evaluated on the test set.

|  |  |  |  |
| --- | --- | --- | --- |
| Linear model fit with: | Current measurements only | Current and previous measurements | Current and previous 3 measurements |
| Score | 121.84 | 114.04 | 112.46 |

An example of the predictions on the testing set is shown below. This segment of testing data was selected due to the presence of a few days of highly volatile GHI and a few days of very smooth GHI curves, so that the impacts of both can be seen.



Some observations of this are that the predictor sometimes estimates a negative GHI, which is physically impossible. Additionally, the predictor does not predict a steady value of zero during the night. Other GHI curves are followed decently, but rapidly changing conditions are not necessarily captured.

**Decision Tree**

A decision tree is an algorithm which, through a series of comparisons can be used to perform regression. An initial node is selected from which two branches emerge. The decision on which branch to take depends on the selected threshold value, *v*, of a single feature, *k*. If, for a new instance, *xk v*, then the branch on the left is taken. Otherwise, the branch on the right is selected. After each split there could be another node or a leaf. Once a leaf is reached, the prediction value is given by that leaf and all instances that reach that leaf predict the same value. Once the decision tree is formed, it is quite simple to evaluate new instances.

Decision trees are generally considered to be greedy algorithms, in that they are not globally optimized, but optimized at each step for the most immediate improvement. When performing regression, the cost function of each node is the CART function

Where

And

(**FROM Geron pg 184)**