Solar Energy Prediction

Using Machine Learning

General Assembly - Data Science Immersive

Capstone Project

Jeff Patra

6 June 2017

Solar Energy Prediction Using Machine Learning

Capstone Project

Executive Summary

Solar energy is rapidly becoming a viable renewable energy source. It offers many environmental advantages; however fluctuations with changing weather patterns have always been problematic. While solar may not entirely replace fossil fuels it has its place in the power generation portfolio. Many electricity companies are looking to add solar energy into their mix of power generation and in order to do so they require accurate solar production forecasts. Errors in forecasting could lead to large expenses like excess fuel consumption or emergency purchases of electricity from neighbouring utilities. Power forecasts are typically derived from numerical weather prediction models, but statistical and machine learning techniques are increasingly being used in conjunction with the numerical models to produce more accurate forecasts.

For my capstone I will predict solar energy across Oklahoma State using weather forecast data. The prediction is trained and tested against the solar

energy produced at 98 weather stations across the state over a fourteen year time period from 1994-2007.

The machine learning techniques applied in the prediction model are: multi-linear regression, decision tree, random forest and gradient boosting. After running all the models the best predictive model was gradient boosting method which yielded an average r-squared of 78%.

Gradient boosting was the best model for this excise because it allowed for tuning of the hyper parameters yielding more accurate prediction. Tuning of the hyper parameters was completed using a grid search. Principal component analysis (PCA) was also applied and reduced the total dimensions from fifteen features down to three principal components saving 25% in computing time. The coding for my capstone is accompanied in the adjacent Jupiter notebook (Capstone_solar_prediction.ipynb). Reference to the original contest can be found here: https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest

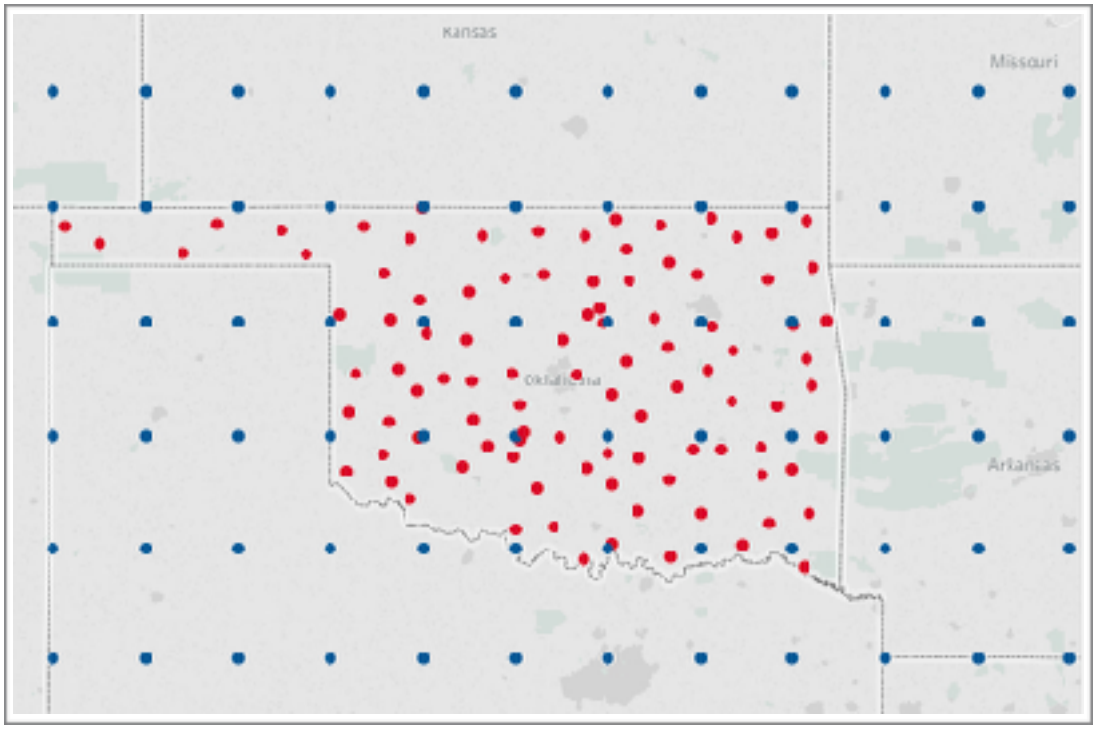
Data Description

This study was initially completed as a kaggle contest in 2013-14. I will use the data provided from this contest to create my own results.

From the data provided we can describe them in two categories:

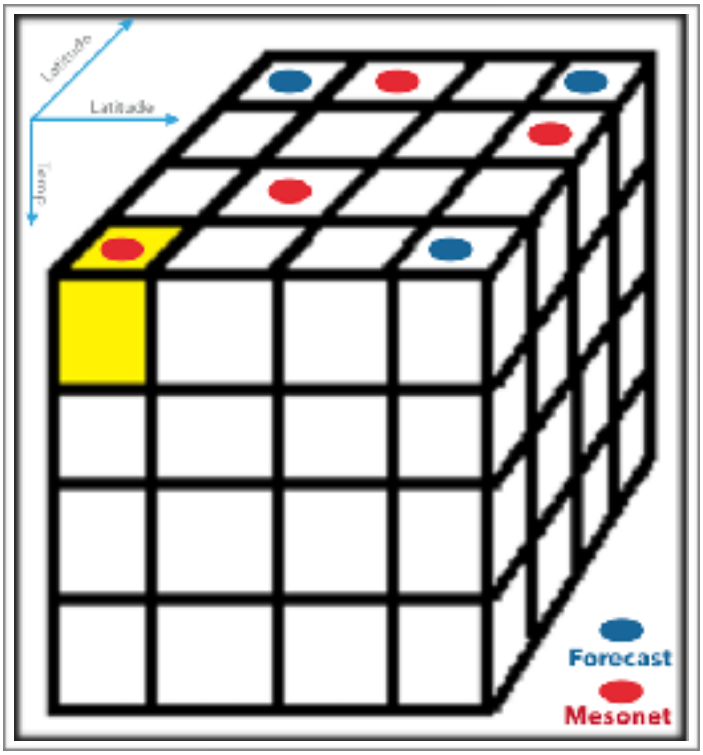
1. Global forecast information consisting of 15 netCDF4 files. Each file contains a specific forecast element array (i.e temperature, pressure, precipitation, etc) at a given latitude and longitude over a fourteen year period from 1994 - 2007 where the step rate is in days.

2. The second data set consists of calculated solar energy acquired from 98 weather stations (mesonets) during the same fourteen year period.



In simple terms the goal is to use the **blue** dots (weather forecast data) to predict the **red** dots (solar energy).

At each specific latitude and longitude of the forecast data (blue dot) exists a multi-dimensional array of multiple models in time for each of the fifteen features. To simplify I took the average of all forecast models to create a one dimensional array for each feature.



The table below is a description of each feature:

Features	Description	Units
1. apcp_sfc	3-Hour accumulated precipitation at the surface	kg m-2
2. dlwrf_sfc	Downward long-wave radiative flux average at the surface	W m-2
3. dswrf_sfc	Downward short-wave radiative flux average at the surface	W m-2
4. pres_msl	Air pressure at mean sea level	Pa
5. pwat_eatm	Precipitable Water over the entire depth of the atmosphere	kg m-2
6. spfh_2m	Specific Humidity at 2 m above ground	kg kg-1
7. tcdc_eatm	Total cloud cover over the entire depth of the atmosphere	%
8. tcolc_eatm	Total column-integrated condensate over the entire atmos.	kg m-2
9. tmax_2m	Maximum Temperature over the past 3 hours at 2 m above the ground	K
10. tmin_2m	Mininmum Temperature over the past 3 hours at 2 m above the ground	K
11. tmp_2m	Current temperature at 2 m above the ground	K
12. tmp_sfc	Temperature of the surface	K
13. ulwrf_sfc	Upward long-wave radiation at the surface	W m-2
14. ulwrf_tatm	Upward long-wave radiation at the top of the atmosphere	W m-2
15. uswrf_sfc	Upward short-wave radiation at the surface	W m-2

Data Description from Kaggle
Each netCDF4 file contains the total data for one of the model variables and are stored in a multidimensional array. The first dimension is the date of the model run and will correspond directly with a row in either the train.csv or sampleSubmission.csv files. The second dimension is the ensemble member that the forecast comes from. The GEFS has 11 ensemble members with perturbed initial conditions. The third dimension is the forecast hour, which runs from 12 to 24 hours in 3 hour increments. All model runs start at 00 UTC, so they will always correspond to the same universal time although local solar time will vary over each year. The fourth and fifth dimensions are the latitude and longitude uniform spatial grid. The longitudes in the file are in positive degrees from the Prime Meridian, so subtracting 360 from them will translate them to a similar range of values as in station_info.csv.

To extract the data I need from the netCDF4 files I am going to create a dictionary for each feature where the key is the latitude/longitude and the value is the one dimensional array. Once I have the features loaded in a dictionary format we can then pair to the meson location. Since the the forecast data is more sparse than the mesonets I rounded the lat/lon of the station data to nearest integer and used the closes forecast location to predict solar energy. For detailed coding of data extraction, loading and conversion to dictionaries refer to jupyter notebook (Capstone_Part4_Jeff).

Exploratory Data Analysis (EDA)

Before we do any solar prediction we need to understand the data that we have. This section will be split into two parts where we examine the:

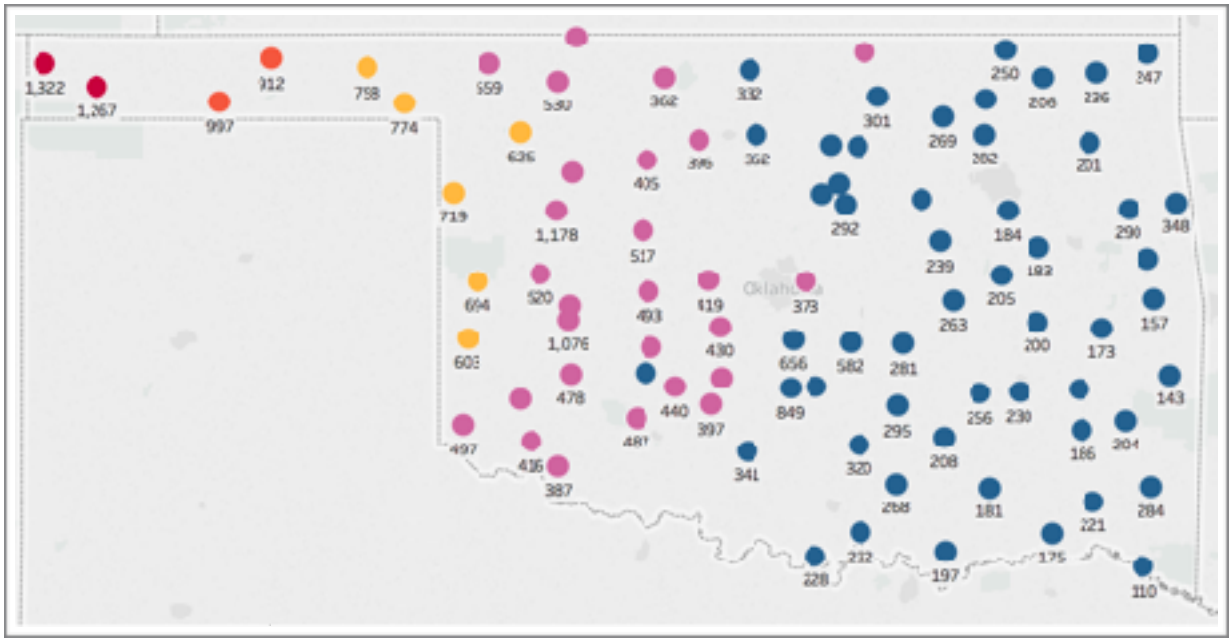
1. Station data (98 mesonets)
2. Feature data (15 features)

Station Data EDA

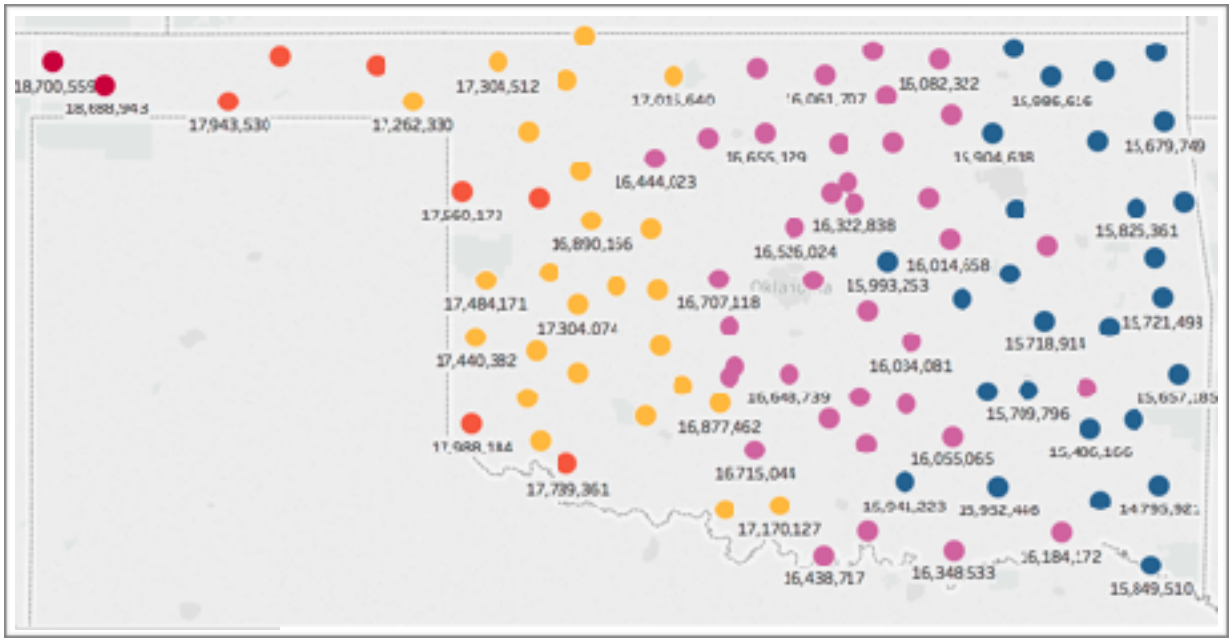
With regards to the station data there were two files supplied:

1. Station information (latitude, longitude and elevation)
2. Calculated daily solar energy for 98 mesonet over a fourteen year period in days

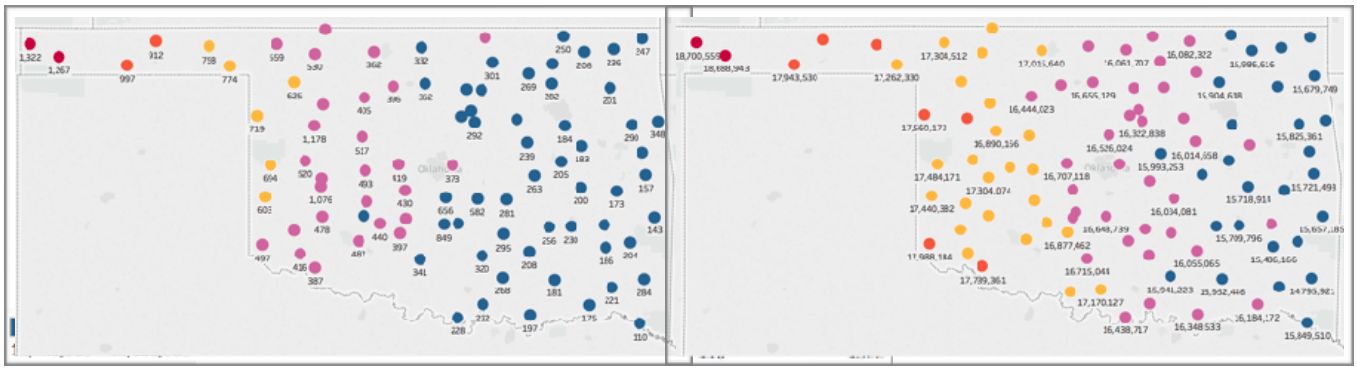
The first step was to ensure that the location and elevation information provided for each station was correct. The plot below displays the location of each mesonet with the associated elevation value displayed in feet. Based on the location information provided everything appears accurate and correct. One noticeable observation is that elevation is lowest in the southeast and gradually increases towards the northwest.



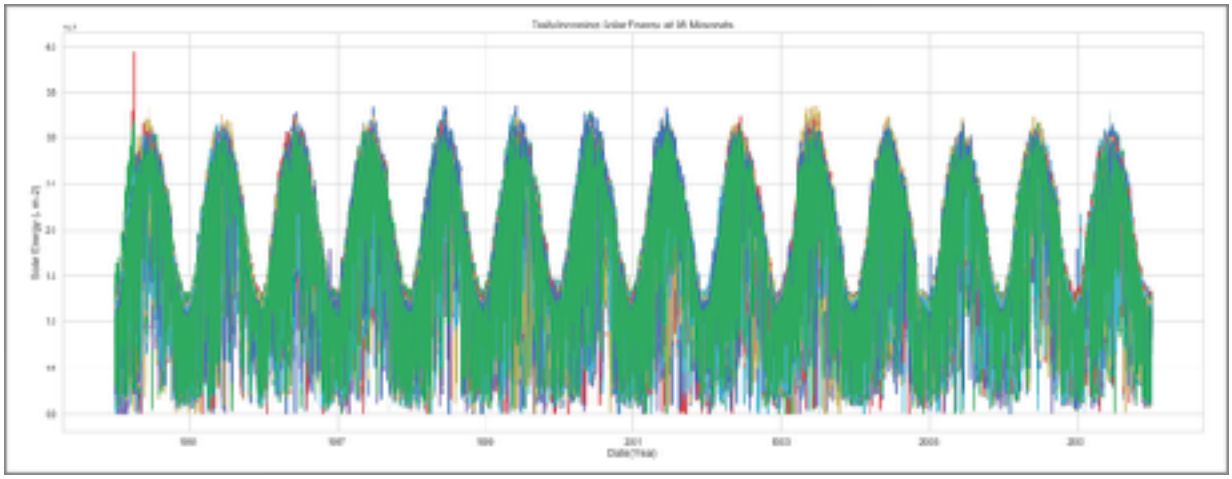
Now we can load the calculated solar energy for each station. The plot below displays the average solar energy generated for station over the fourteen year time period.



Based on the this quick look of averages we can see there is a relationship between elevation and solar energy generation.

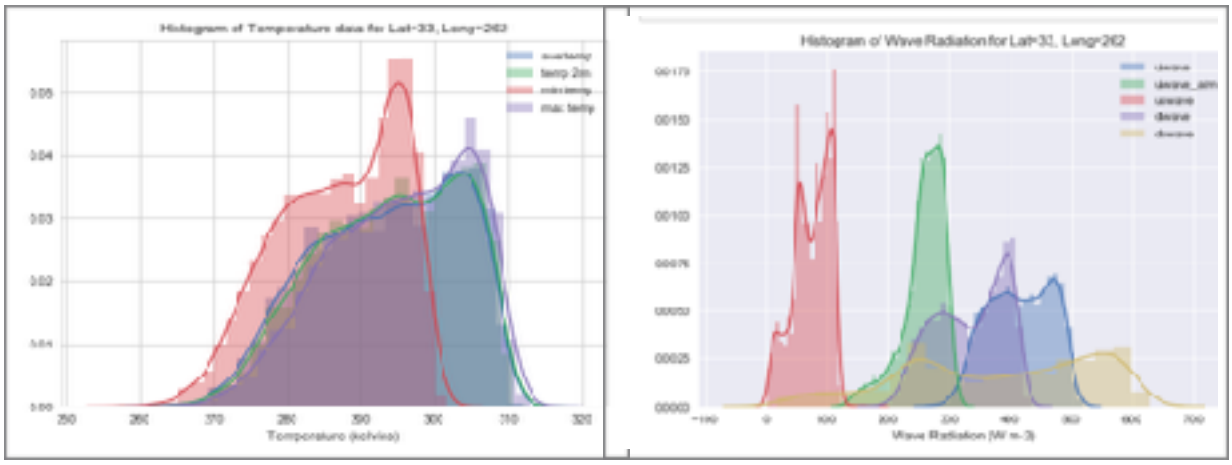


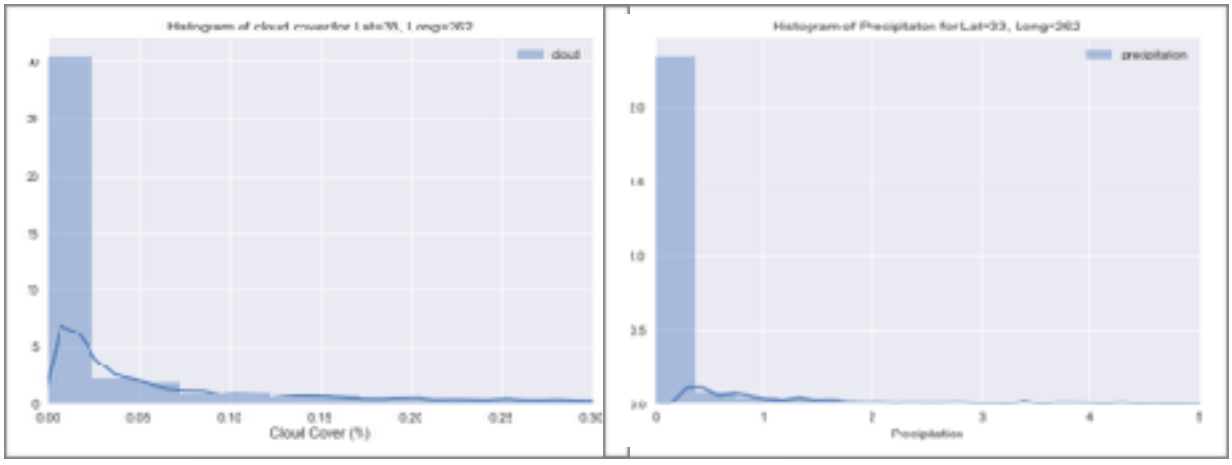
It is also important that we have a closer look at how the solar energy generation changes with time. The plot below displays the change in solar energy generation over time. One thing that stands out is that solar generation is cyclical. We also do not see any null values or 'broken' mesonets.

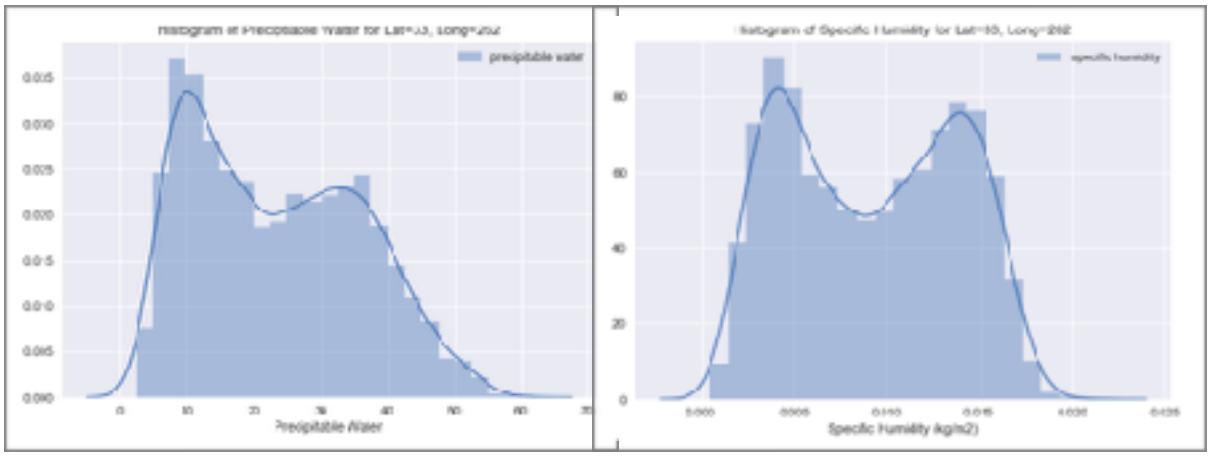


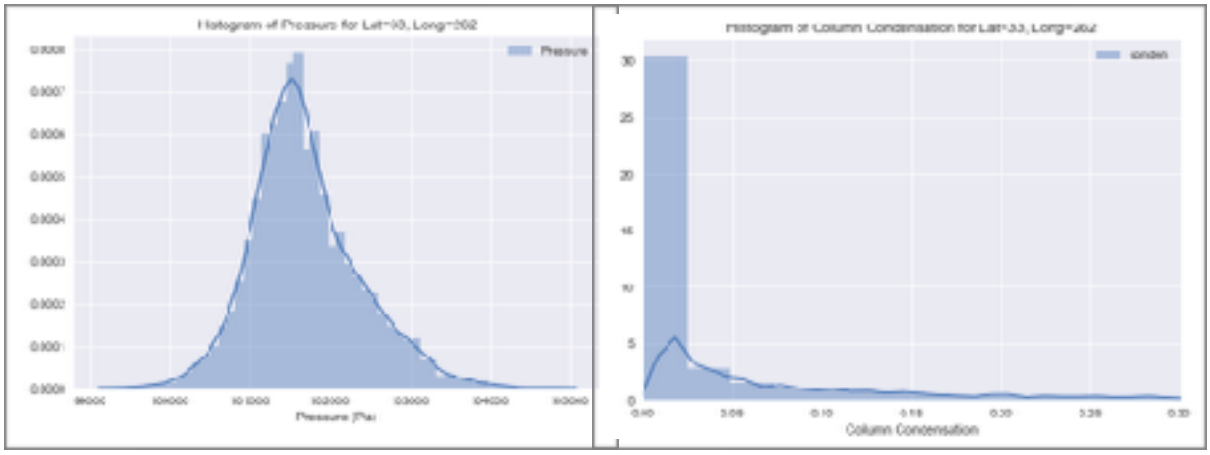
Feature Data EDA

Using the dictionary function we created earlier we can now start to view the feature data. Below are histogram plots of all the features for one station:





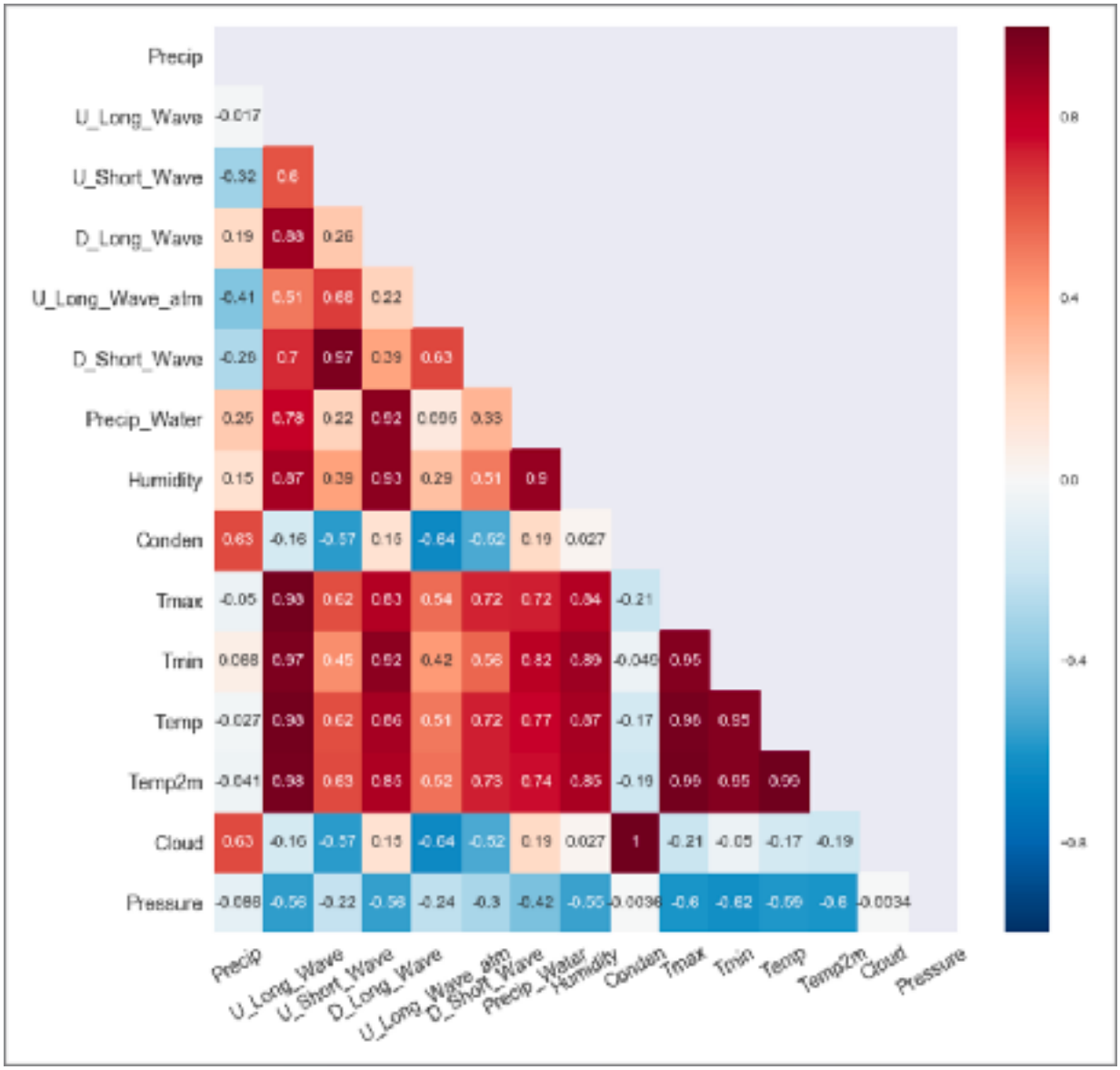




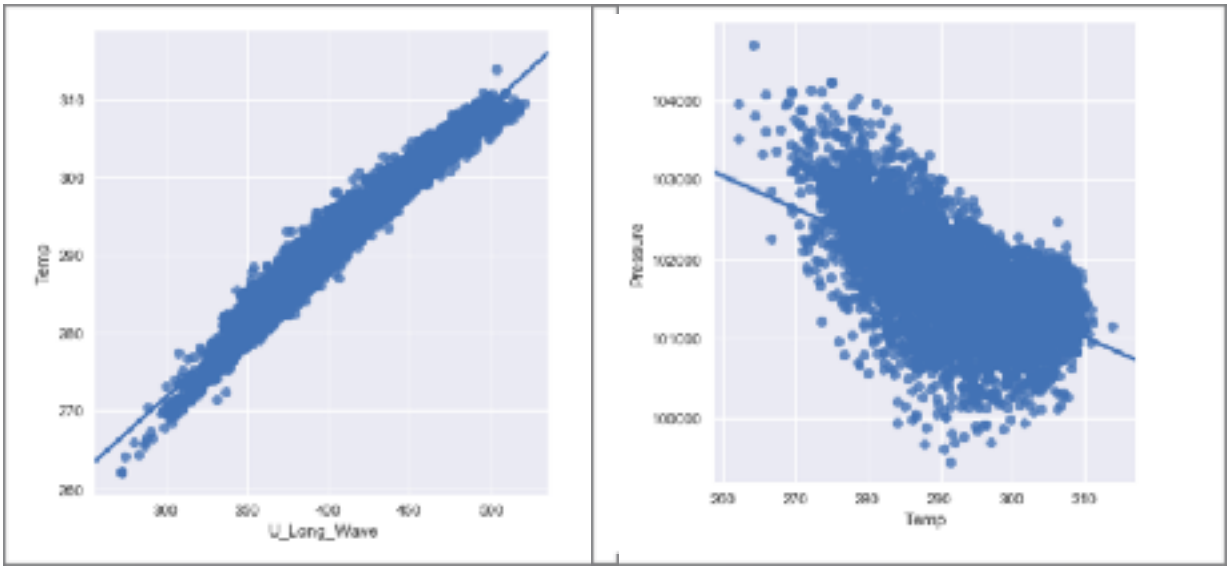
After viewing all the histograms there are a few observations:

1. Not all the data is normally distributed
2. Condensation, precipitation and cloud cover have very low distributions. Most values are very close to zero.
4. The frequency distributions between the long and short wave radiation are quite variable. Was expecting them to be more similar in distribution
5. Temperature distributions look to mimic each other
6. Pressure had a normal distribution
7. Specific humidity and precipitable water appear to have binomial distributions suggesting the weather is either really wet or really dry.

The next step is to determine if any of our features are correlated with each other. To do so we will view the correlation heat map below:



Based on the heat correlation map there are some pretty strong correlations between certain features (temperature and long wave radiation). There are also some anti-correlations like temperature and pressure (which is expected). Let's have a closer look at some of these correlations plotted versus each other:



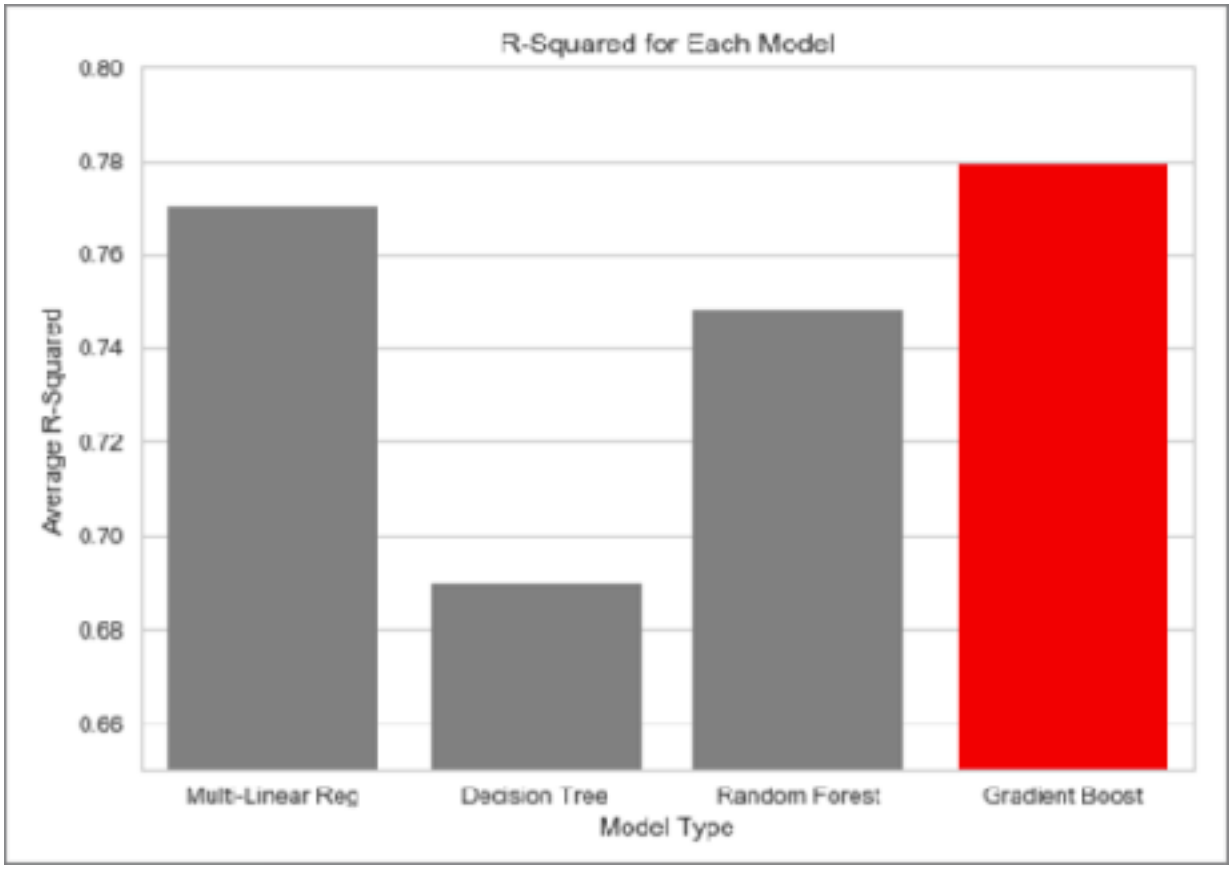
With all the features loaded we are now ready to perform our model prediction. To view the detailed coding refer to the Jupiter notebook (Capstone_Part4_Jeff)

Modelling, Machine Learning and Evaluation

Before we try and predict solar energy across the entire data set the approach we will use is to predict solar energy at one station for one year. Once we get all the models running we can expand to the rest of the period and stations. In this experimental phase we will use the ACME weather station to test and train.

Prior to running the model we will standardise our data. Standardising takes all our features and centres their mean to zero. We will then split the data through a train/test split. In this data set I used a 90% train and 10% test.

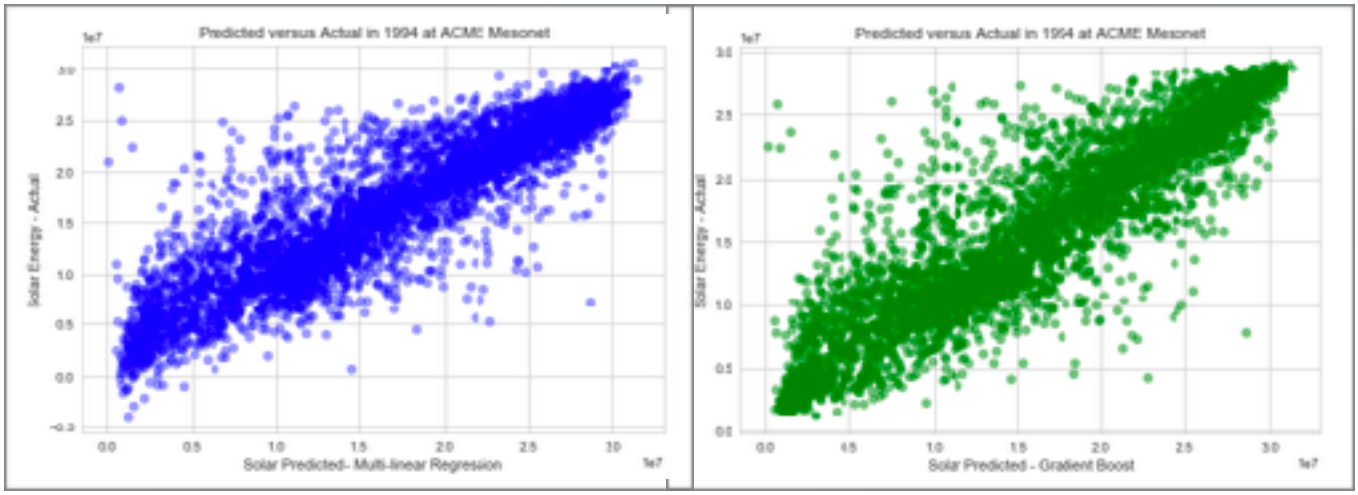
The first predictive model we will try on our data will be a simple multi-linear regression using all fifteen features as inputs. The multi-linear regression produced an r-squared of 77%. Based on just a straight multi-linear regression this is a pretty high baseline score. When we run other machine learning models using the sci-kit learn module in python we get the following cross validated results:



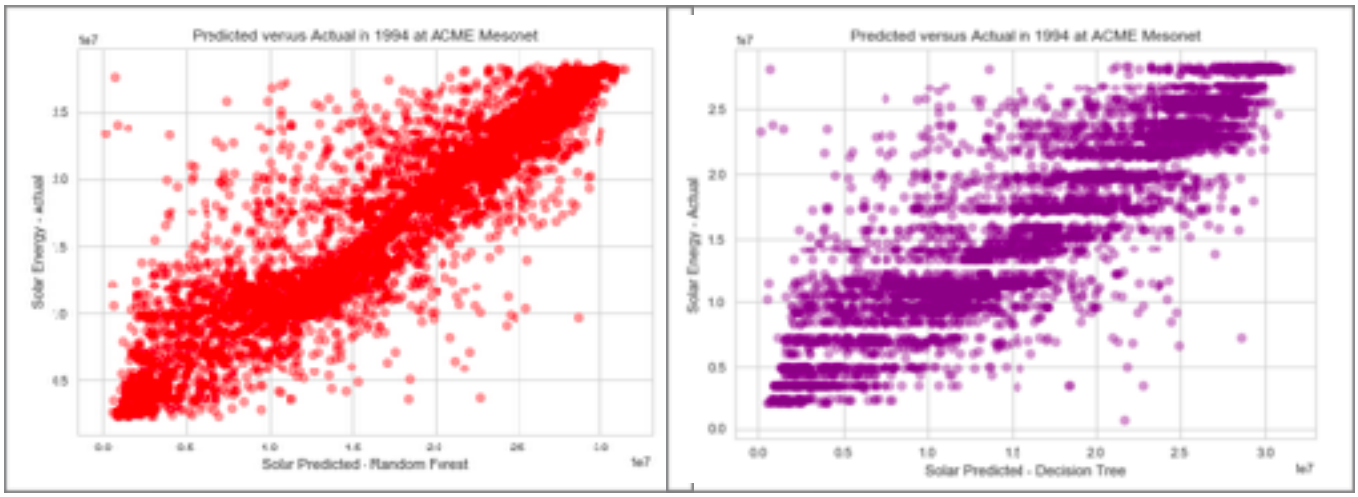
From the r-squared scores gradient boosting yields the best results at 78%. I was surprised to see that multi-linear regression maintained its high prediction rate and was better than the random forest model.

Even though we are evaluating the models based on their r-squared score we want to see how they are performing versus the actual solar energy. Below are the four models plotted versus actual (mlr-blue, gradient boost-green, random forest-red and decision tree-purple):

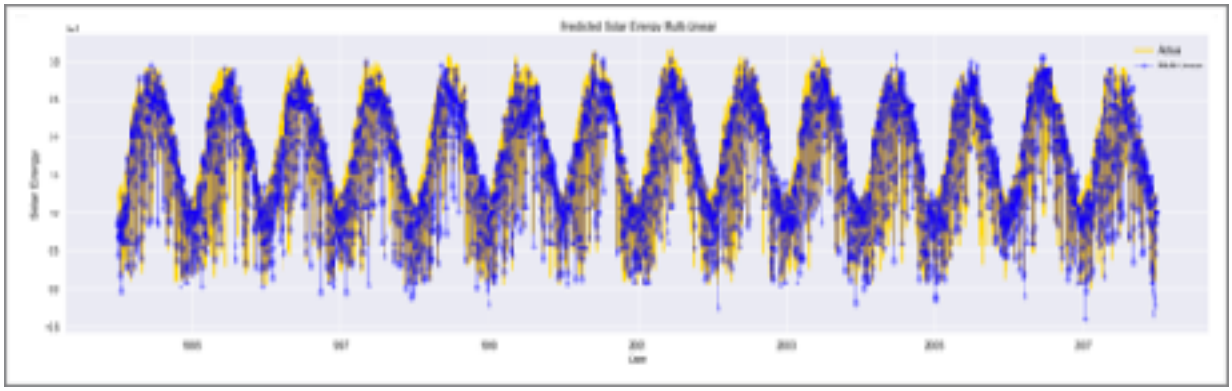
When viewing the predicted versus actual plots we can see that gradient boosting is performing the best because most the points are concentrated on the centre line with only a few points scattered . With the multi-linear regression we see more outliers and a much 'thicker' band down the centre line suggesting less accuracy. Whilst random forest and decision tree produce satisfactory results they both performed well below gradient boosting method. Notice that in all

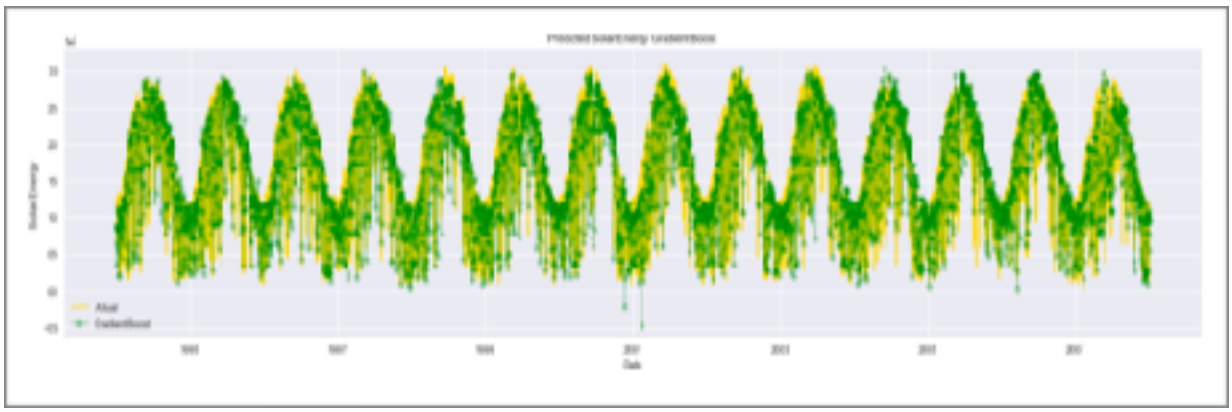


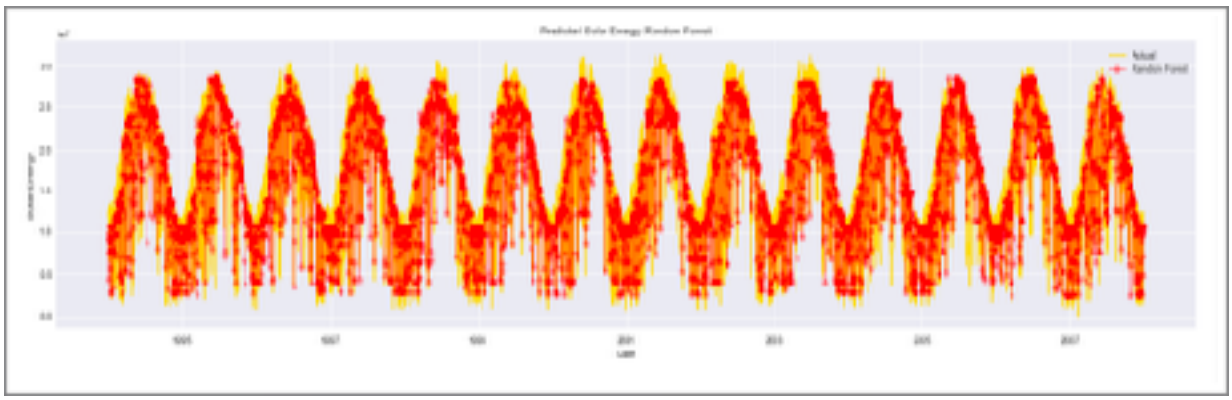
models they each struggle in the 5,000,000 - 10,000,000 J/m2 which suggests that there is an underlying feature in the data.

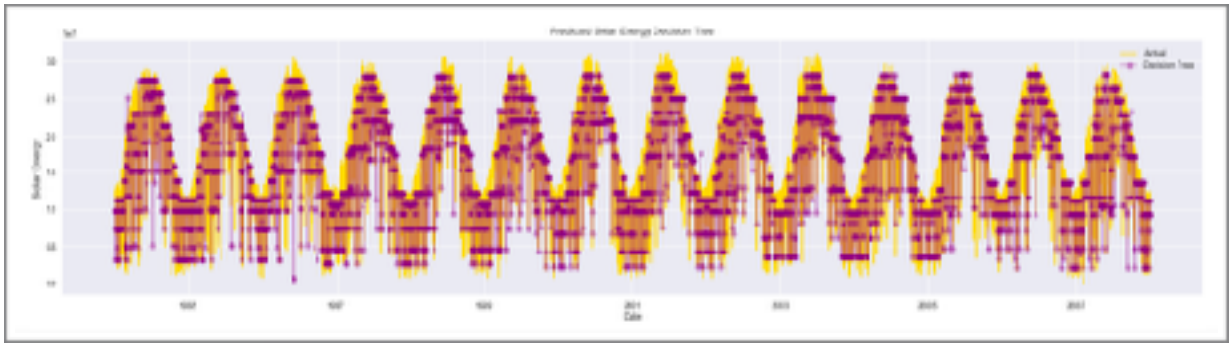


Now that we have established a method to predict solar energy we can modify the code to propagate each model through to all stations over the fourteen year time period.









When we view the each prediction model over time we start to see the strengths of the gradient boost method. It really comes into its own at the high end of the solar energy prediction where the other models tend to under predict, especially random forest and decision tree.

Model Tuning

So now that we have established a way to predict we can now focus on tuning the model and possibly improving the efficiency of the running time. One of the benefits of gradient boosting is that it allows flexibility in tuning of the hyper parameters. When we first ran the model we used the default input parameters. Now let's a grid search to evaluate if we can tune the parameters to output a better result. Based on some online research and discussion forums most users suggest that the following hyper parameters have the greatest impact on the gradient boost method:

1. learning_rate: range [0.05,0.08,0.11,0.14,0.17,0.2]
2. min_samples_split range [25,35,45,55]
3. min_samples_leaf range [25,50,75]
4. n_estimators range [50,100,150]
5. max_depth range [4,6,8,10]

To run the grid search fitting five folds with the above input parameters for each of 864 candidates, totalling 4320 fits. The total time taken to run the grid search was 277 seconds (4.6 minutes). Based on the input parameters the best parameters to use were:

Gradient Boost Hyper Parameters

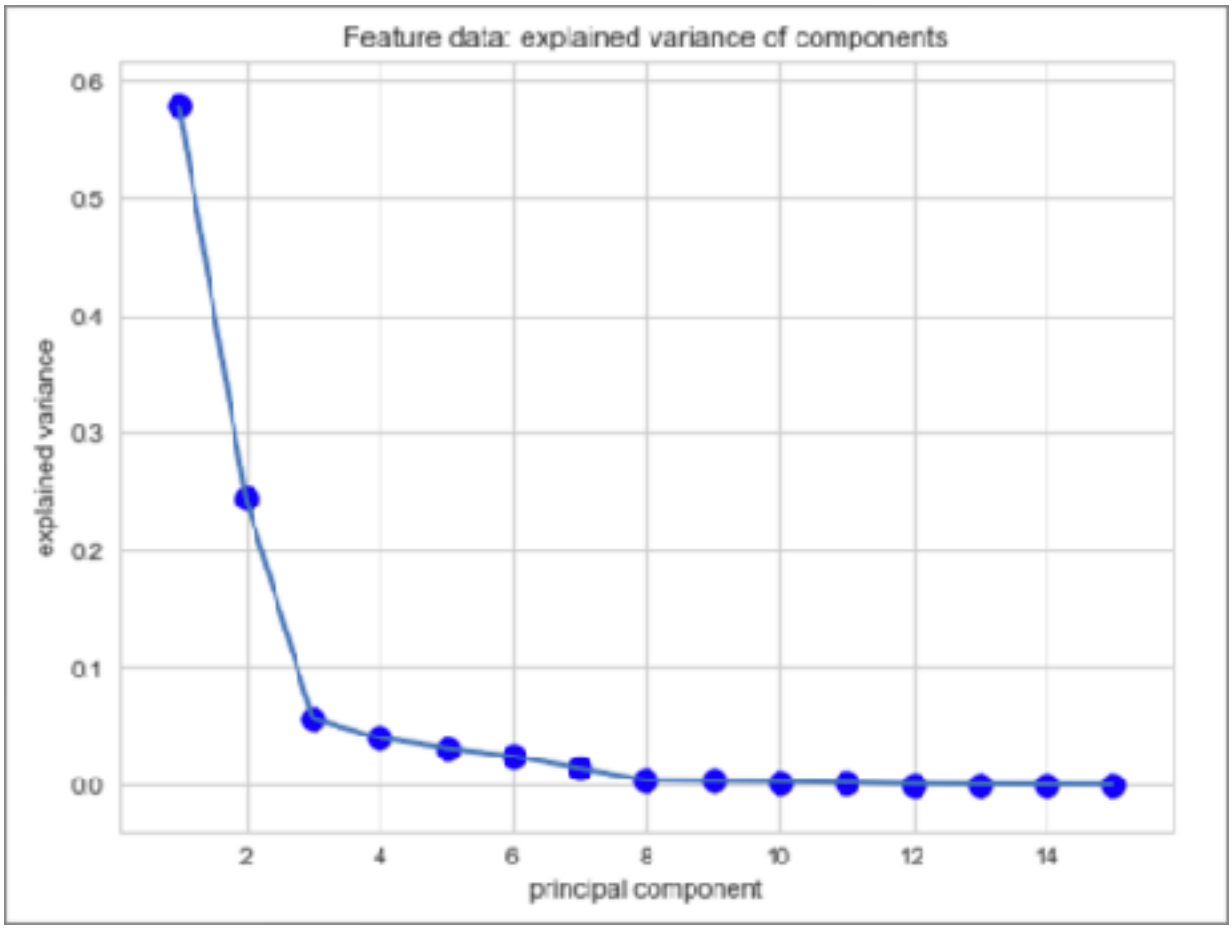
Hyper Parameters	Default input	Post Gridsearch
Learning Rate	0.15	0.14
min_sample_split	2	2
min_samples_leaf	1	50
n_estimators	100	50
max_depth	5	4
R-Squared	78.5%	79.7%

Based on the grid search tuning we improved the r-squared by 1.2%.
Now that we have tuned our hyper parameters we can perform principal component analysis (PCA). Through PCA we can evaluate if there is any opportunity at reducing dimensionality. Doing so could lead to better prediction results and reduced computing time. Using the principal component module in sci-kit learn we convert our fifteen features into principal component space. Once all our features were in the PC space we plotted the explained variance ratio versus each component to identify which ones were the major contributors. Below is the elbow plot of the each component:

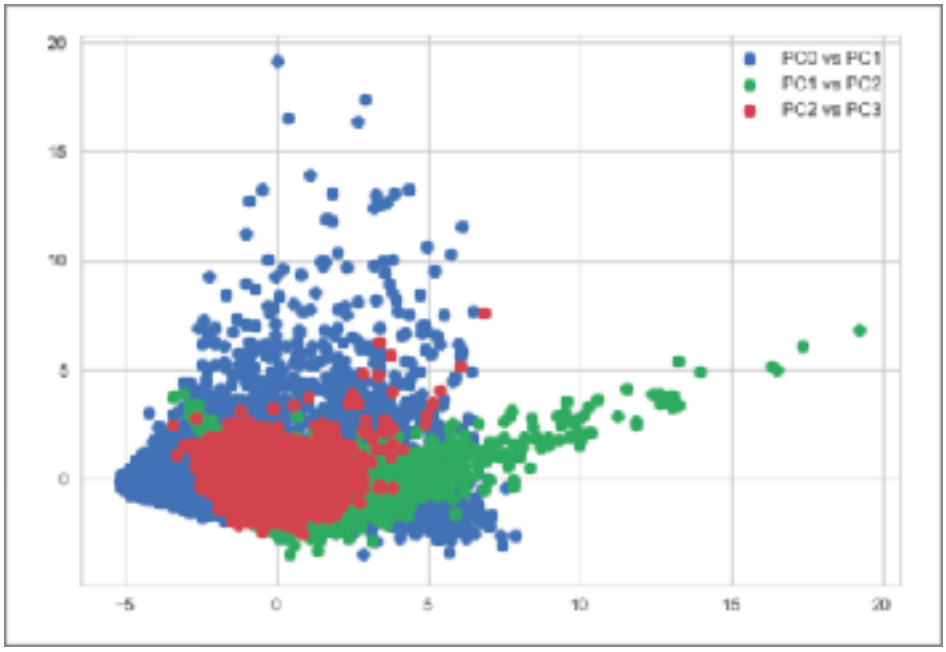
The plot above indicates that the major components are PC1, PC2 and PC3 where PC4 - PC15 explain less than one percent of all the components. Now lets see if we can see if there any major features in each component that are driving the result:

Feature	PC 1	PC 2	PC 3
Cloud	0.0677	0.475	0.219
Condensation	0.673	0.475	0.221
Down wave long radiation	-0.297	0.211	-0.083
Down wave short radiation	-0.252	-0.249	0.392
Humidity	-0.316	0.115	0.043
Precipitation	0.009	0.388	0.298
Pressure	0.191	-0.138	0.609
Precipitable water	-0.274	0.219	0.066
Temperature	-0.335	0.031	0.015

Feature	PC 1	PC 2	PC 3
Temperature at 2m	-0.335	0.020	-0.004
Max temperature	-0.333	0.012	-0.027
Minimum temperature	-0.325	0.102	-0.101
Upward long wave radiation	-0.335	0.038	0.002
Upward short wave radiation	-0.198	-0.316	-0.079
Upward long wave radiation atmosphere	-0.189	-0.313	0.509

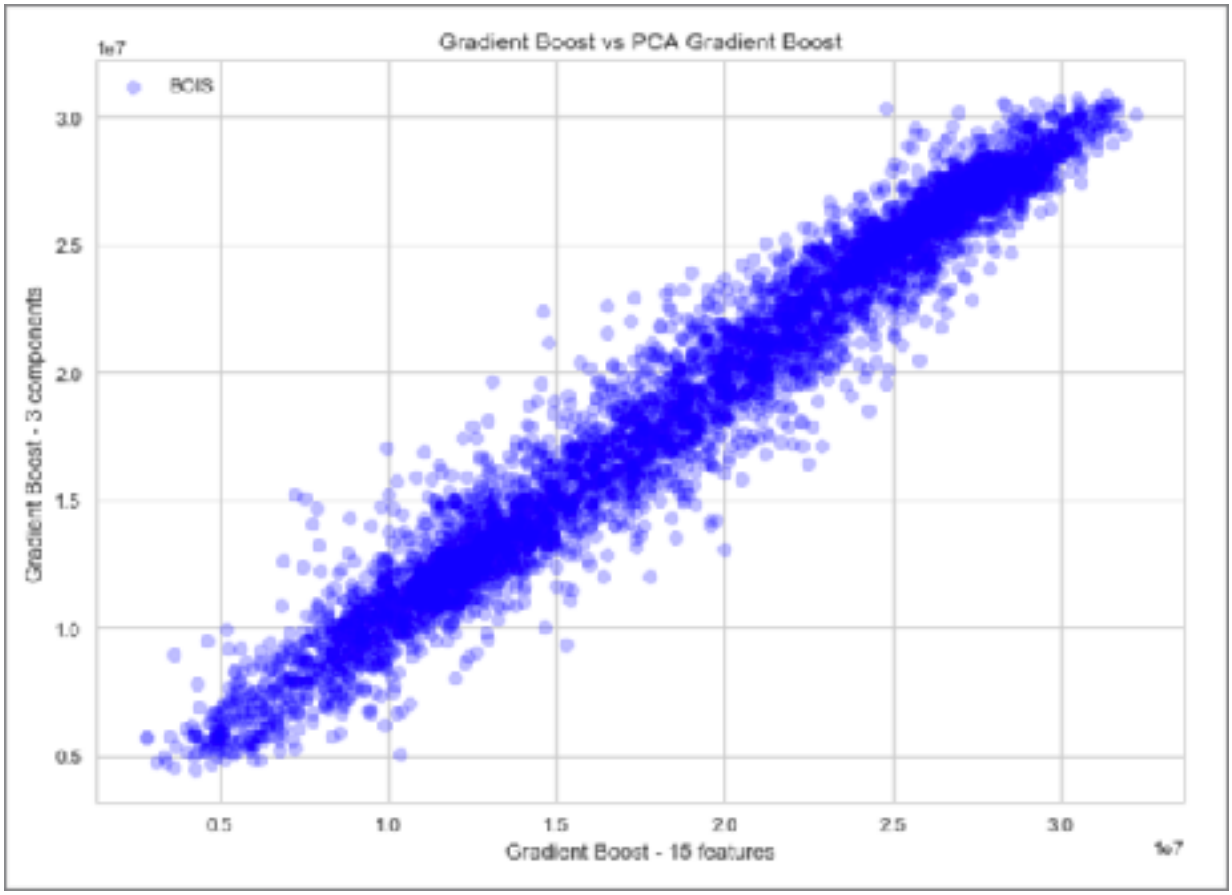


Comparing the values within each component to the features does not indicate any dominant features. This is also displayed on a scatter plot to see if there are any dominant trends in each component:

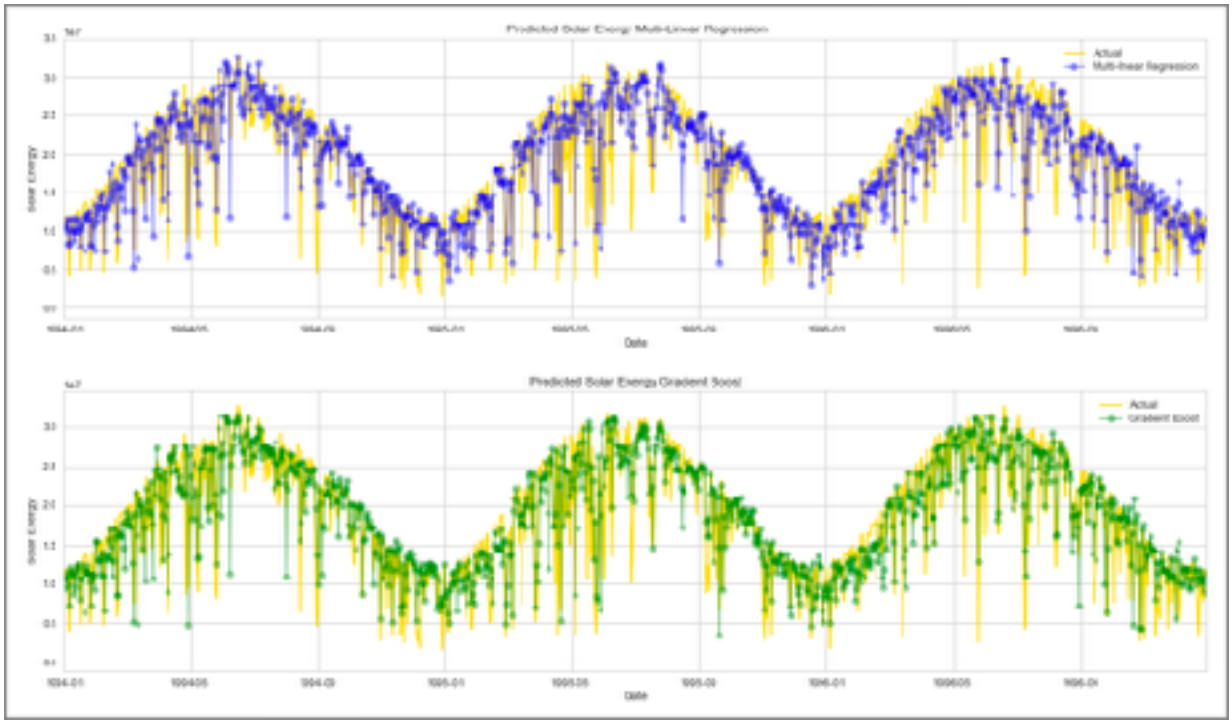


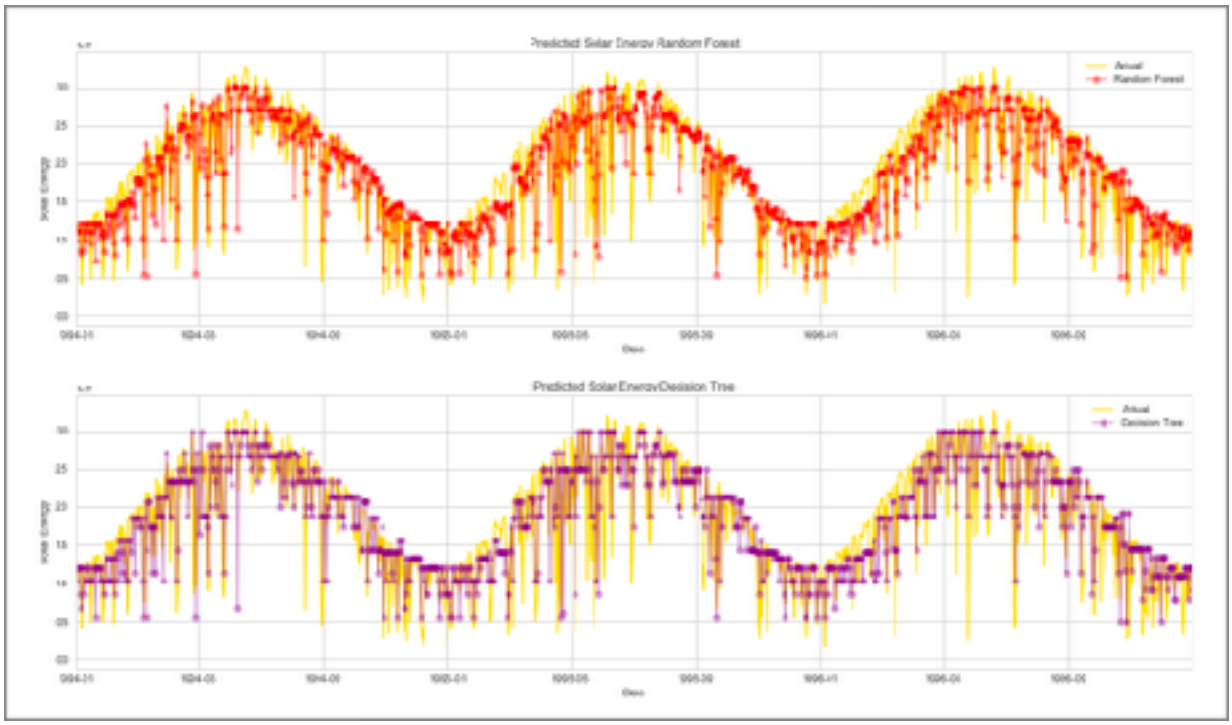
Full Scale Model Run and Processing

In the interest of science we are going to run all the models so we can evaluate and compare the results. If this were deployment I would choose the best two and compare. We performed two full scale runs. The first run looped through each station over the entire fourteen year period using a 90% train, 10% test split. This model did not utilise any tuning or PCA. The total run time of this model was 1475 seconds (24.6 minutes). The second run utilised the grid search input parameters and the reduced dimensionality from the principal component analysis. The total computing time for the second model run was 1087s (18.1 minutes) which is equivalent to 26% savings in time (6.5 minutes). Whilst this is not a large absolute time the PCA has the potential to save a lot if we were evaluating a larger region like the entire USA. As a check to ensure that we are not losing accuracy with the reduced dimensionality we plotted the gradient boost with 15 features versus PC reduced gradient boost. The plot below demonstrates no loss.



A closer view of the model predictions indicate that each model struggles to predict at the high end. One possible cause is the effect of severe weather patterns like tornadoes.


Predicted Solar Energy Multi-Linear Regression
Actual
Multi-linear Regression
Solar Energy
Date
Predicted Solar Energy Gradient Boost
Actual
Gradient Boost
Solar Energy
Date


Predicted Solar Energy Random Forest
Actual
Random Forest
Solar Energy
Predicted Solar Energy Decision Tree
Actual
Decision Tree
Solar Energy

Another way to visualise the weather prediction is to average the prediction for each month and then stack them into a gif. Below we can see the changes in solar energy production by month. In the winter months we can observe that solar production is low (blue) and as the season transitions to summer solar production increases (red) .
(If viewing in pages click on image to play gif).

Discussion & Recommendations

Gradient boosting method resulted in the best model prediction. Using PCA we were able to reduce the dimensionality from fifteen to three resulting in a 26% savings in computing time. The time savings could be significant in the commercial space if we were evaluating a larger area like the entire USA. Unfortunately when we looked at each principal component we were unable to identify the major contributing features so as an input we still require all fifteen features.

Gridsearching was used on the gradient boost method to tune the hyper parameters which resulted in a 1% improvement. If we had unlimited computing we could incorporate gridsearching for each station and use the best parameters to yield the best score however this would result in excess computing time. For example, under current computing specs the estimated run time incorporating grid searching all 98 stations would be approximately 8 hours.

Whilst multi-linear regression yields a similar r-squared it is largely affected by severe weather which reduces the accuracy of the regression.

Future recommendations:
Since forecast data is in time we could try time series prediction, however it is likely that time series prediction would only provide accurate forecasting unto one week in advance.

To further fine tune this model we could potentially look at dividing the data set up into seasons. This would potentially reduce the range of the prediction and

improve the accuracy. We could also look at a severe weather identifier. Oklahoma is prone to tornadoes which are severe low pressure systems. Using the pressure data as a cutoff we could identify tornadoes and train two seperate models based on 'normal' weather and 'severe' weather. By removing the severe weather data we may be better able to predict the peak solar production because the current model struggles at the high end of the solar energy prediction. I suspect that its the tornadoes that are bringing the prediction down.

Deployment

Recall that during the EDA we identified that there was a correlation between elevation and solar production. So the best location for a solar farm based purely on solar production would be the area with the highest elevation which is the northwest corner of Oklahoma. Of course this does not take into account surface infrastructure constraints and requirements.

If we were to take into account surface requirements then this project turns into an optimisation project where solar production would be another input.

As a test it would be interesting to apply this model to another geographic area to see how it performs (for example, California or even Australia).

If deploying this model on a commercial scale I would choose PCA gradient boosting and multi-linear regression. Eliminating the other two models would save further in computational time. Gridsearching each station is not really a commercial solution for the incremental gain.

Possible scenarios for further tuning and re-training involve splitting the time frame into seasons and build gradient boosting model for each season. The weather patterns would possibly be more consistent and easier to train in each season. We could also incorporate a 'severe weather' identifier.

From an energy company perspective I assume the goal would be to have consistent production. The priority would be to choose a solar farm location least affected by severe weather and closest to surface infrastructure. Avoiding severe weather patterns will cause fewer power interruptions.

Conclusion

- Machine learning for solar prediction works.
- Gradient boosting had best result yielding an average r-squared of 78%.
- Tuning in GB resulted in an improvement of 1.2% on the r-squared score.
- PCA reduced dimensionality and computing time by 25%
- All models struggled to predict at the top end. Possibly due to severe weather bring the forecast down.
- Elevation and solar energy are correlated. Higher elevation equals higher solar energy. Based on Oklahoma map northwest corner is best. This does not account for any surface restrictions or infrastructure and towns.
- For commercial use my recommendation would be to choose an area that has the most consistent solar production with fewest severe weather events. This would yield lower service interruptions.
- Future re-training and improvement might involve splitting the data by season and incorporating a 'severe weather' identifier.
- To review the coding and model in depth refer to Jupyter notebook (Capstone_Part4_Jeff).
- Blog…?