SOLAR ENERGY PREDICTION

ISDS 577

ABSTRACT

Phase 02 of the analytics project presents a description of the dataset and data cleaning and presents findings from the analytical methods used to estimate the number of solar panels required to aid in solar energy adoption within Orange County.

Group 6

Azmery Laizo, Mansi Borkar, Reeva Patel, Risheng Pan, Swati Murmu

Introduction

As the topic of Climate Change gets wide public attention, generating clean energy has become one of the top policy agendas. Governments across America are incentivizing the public to utilize clean energies, especially in the State of California. Solar panels have been installed in many California homes since solar is the most accessible source of clean energy. According to Evan Halper's research at Los Angeles Times (2014), California installed 234,600 solar panels in 2014, leading the whole nation in terms of the total number of panels installed.

One of the questions that California homeowners ask most is that how many solar panels do they need? Given the wide geographical variation in California, it is hard to generalize one standard for all California homeowners.

As data analysts who live in Orange County, we are using this research to help homeowners in Orange County accurately predict the number of panels needed with our analytic efforts.

Dataset

To answer our research questions, we collected a large amount of data.

First of all, solar radiation information of Orange County is needed since it is the source of solar power. Additionally, factors, such as cloud type, solar zenith angle, temperature, and wind speed, that influence the solar radiation level also need to be considered. Fortunately, we can take good advantage of the National Solar Radiation Database. We extracted solar data on all 69 Orange County zip code zones along with the information on cloud type, solar zenith angle, temperature, and wind speed through the API (Application Programming Interface) service of

the National Solar Radiation Database (https://nsrdb.nrel.gov/). We integrated the API into our Python code and collected the information on the hourly format between the year 2010 to the year 2019.

Nevertheless, one disadvantage of the National Solar Radiation Database data is that it does not specify the zip code zones. The only provided location information is latitude and longitude. Hence we extracted the latitude and longitude information of each Orange County zip code from Opendatasoft.com.

Data Processing

The data we extracted from the National Solar Radiation Database contains 6,044,400 rows and 25 columns. As you can see from Figure 1, all the columns with the dataset are numeric. Some columns are integers, and others are float.

Postal Code	int64
Latitude	float64
Longitude	float64
Year	int64
Month	int64
Day	int64
Hour	int64
Minute	int64
DHI	int64
DNI	int64
Wind Speed	float64
Temperature	float64
Clearsky DHI	int64
Clearsky DNI	int64
Clearsky GHI	int64
Cloud Type	int64
Dew Point	float64
Fill Flag	int64
GHI	int64
Relative Humidity	float64
Solar Zenith Angle	float64
Surface Albedo	float64
Pressure	int64
Precipitable Water	float64
Wind Direction	float64

Figure 1: Column Data Type

We also checked if any missing value exists in the dataset using the “missingno” library in Python. As Figure 2 shows, every column within the dataset has 100% valid values. Hence, filling missing values is not needed for our project.

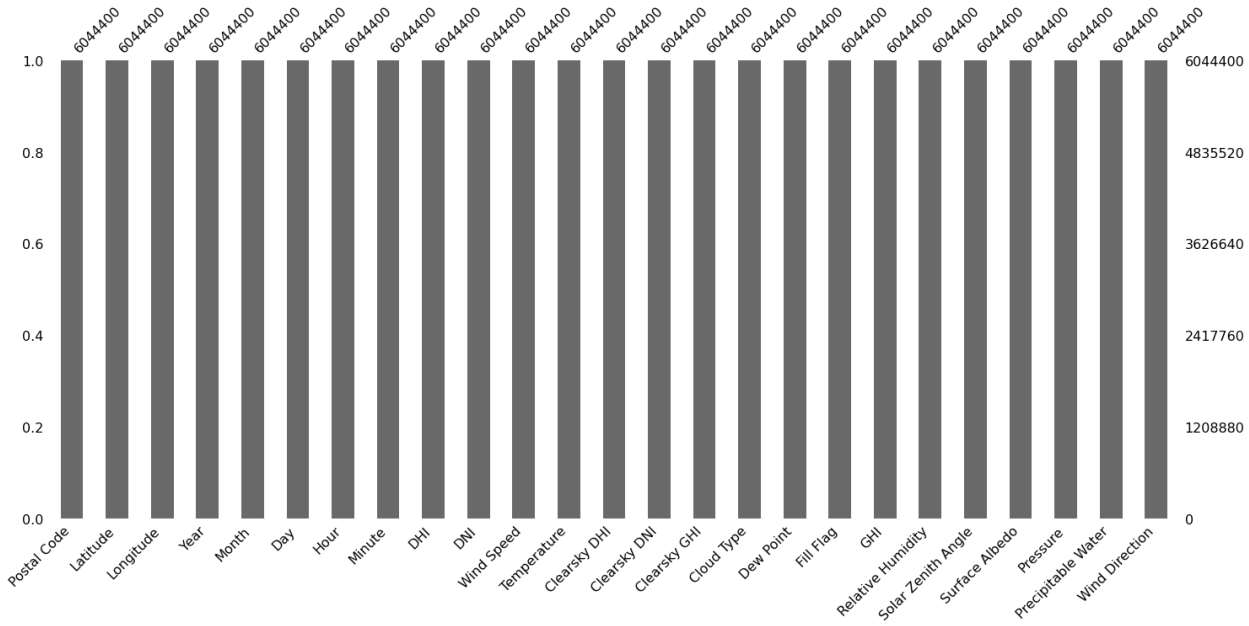


Figure 2: Missing Values Chart

Given the cleanness of our dataset, the only process that we did is using information from Opendatasoft to match zip code zones with the corresponding latitude and longitude in our dataset. The dataset from Opendatasoft includes three columns: Zip, Latitude, and Longitude.

```
Zip          int64
Latitude   float64
Longitude  float64
```

Figure 3: Column Data Type

The approach for matching is to loop through every record of the Solar Radiation dataset and see its latitude and longitude match any zip code's latitude and longitude. When the records have matched zip code zones, we append the matched zip code into our original dataset. However, the first attempt did work out well because we did not have exact matches. We investigated the problem and found slight longitude or latitude differences between the two

pairs of longitude and latitude in situations where there should be a match. For example, one of the records in the solar dataset has a latitude of 33.45 and a longitude of 117.66. It should belong to the zip code 92624. However, the latitude and longitude of zip code 92624 in the Opendatasoft dataset appear to be 33.4 and 117.66, respectively. Hence, the reason behind this unmatched case is the 0.05 difference in latitude.

To avoid this type of matching error, we changed the mechanism of looking for zip code matches. Instead of finding the exact match, we looked for zip codes with the minimum distance from our solar radiation records. We calculated the distance based on the Euclidean distance, which is $\sqrt{\ }(x_1-x_2)^2+(y_1-y_2)^2$

After matching zip codes with our solar dataset, another missing-value check is conducted to ensure that we have clean data for our model building process.

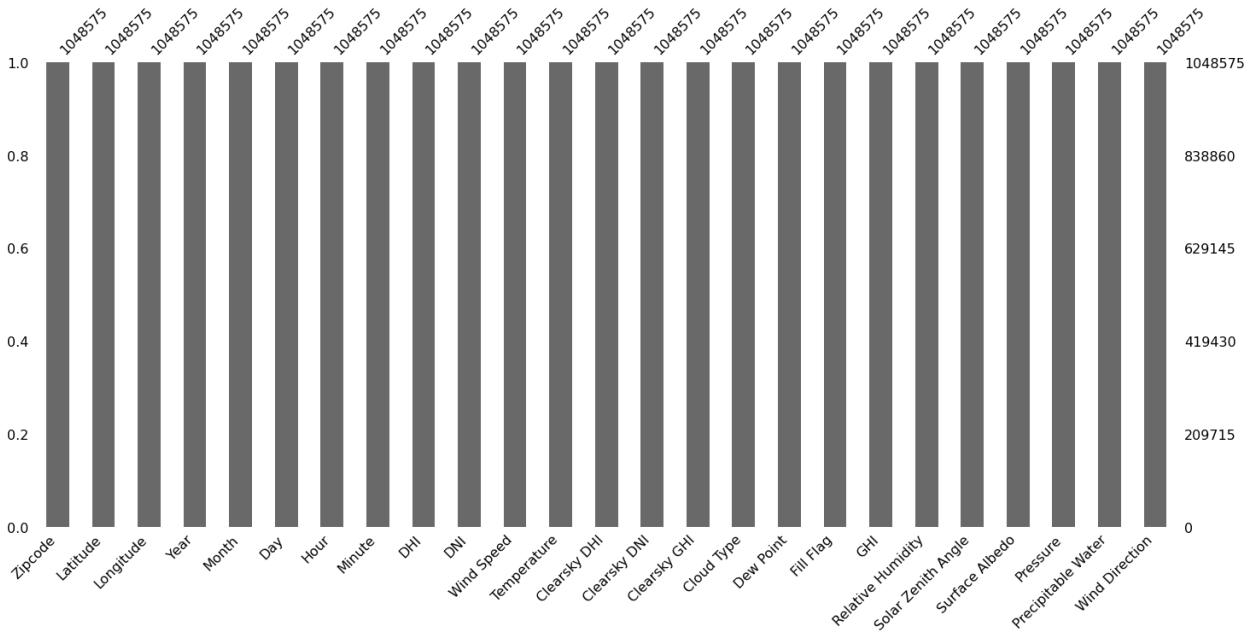


Figure 4: Missing Values Chart

Research Questions

The primary goal of our project is to calculate the ideal number of solar panels based on the zipcodes in Orange County and the power consumption of the user's household. We have approached this by answering three research questions that helped us predict the number of solar panels. The following were the research questions we answered:

Can we predict the weather based on several variables?

The first task is to determine if we can predict the weather based on several variables. We have the following two approaches to dig deeper into this question:

- Predicting the wind speed using time series analysis to get an understanding of the wind speed patterns for future periods
- To predict temperatures of Orange County broken down into 69 Zipcodes for the next 25 years using Random Forest Algorithm. These temperature estimates will be used to calculate the number of solar panels required for a single household unit.

How has solar irradiation changed over the past ten years?

In this question, we will analyze if there is a pattern in the Solar Irradiation(GHI)over the past ten years. We will do so by visualizing a pattern in the following two ways:

- Calculating the average yearly GHI over the past ten years for each zip code.
- Calculating the average monthly GHI over the past ten years for each zip code.

How many solar panels would be needed for generating the power required to support a household within Orange County?

This question will finally predict the ideal number of solar panels needed to generate the power required for a household based on the user's zip code and yearly power consumption.

Research Question 1:

To predict the temperature of Orange County (69 Zip codes) for the next 25 years based on different atmospheric conditions?

The question aims to study the temperatures of Orange county to make estimates for the future. The temperature predictions will be used to calculate the number of solar panels needed for a single household unit of Orange county.

To predict changes in the temperature patterns, we considered different atmospheric conditions from the past. Below are other variables (factors) from our dataset that we will use to train the machine learning model and estimate future weather.

- Wind Speed

- Wind Direction
- Precipitable Water
- Temperature
- Year
- Month
- Day

Machine Learning Model: Random Forest Algorithm

Random Forest is based on ensemble learning; the methods include bagging and boosting. Ensemble learning is a model that makes predictions based on numerous different models. It tends to be more adaptable with minor variation.

- **Bootstrap and Aggregation- Bagging:** We consider a subset of data to train each model simultaneously.
- **Boosting:** We consider a subset of data in a sequence, and each model is derived from the previous one.

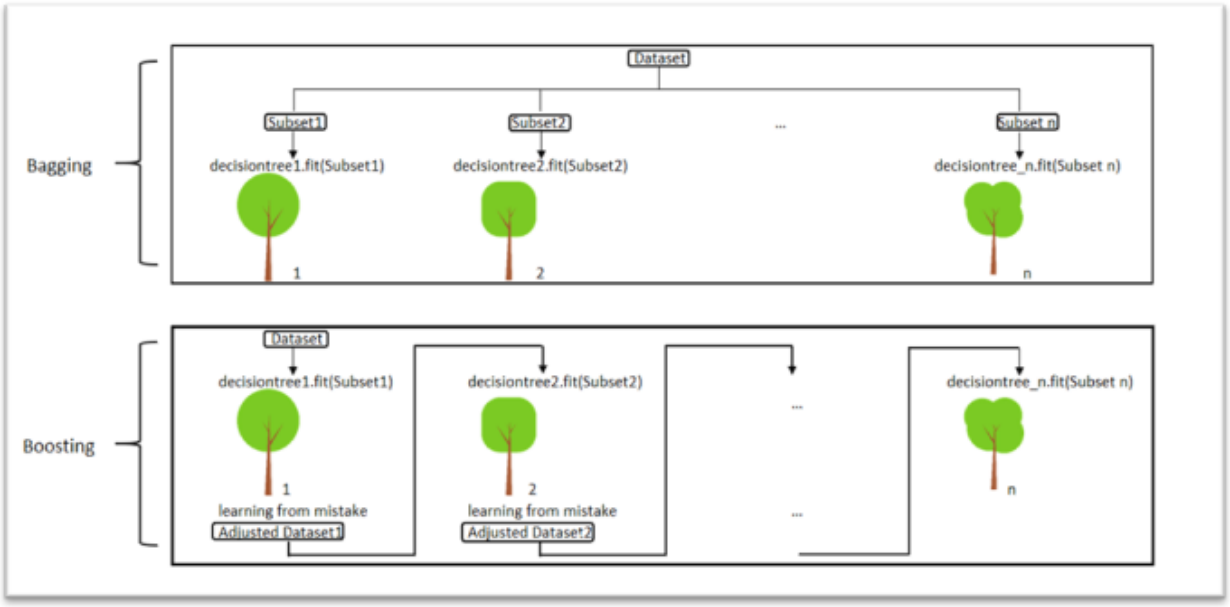


Figure 5: Ensemble Techniques – Bagging and Boosting (Source)

Structure of Random Forest **:** The structure includes multiple decision trees presented with distinct instances, and we consider the majority as the preferred prediction. In Random Forest Regressor, the output obtained finally is the average of all the results, and so it is also called as Aggregation technique.

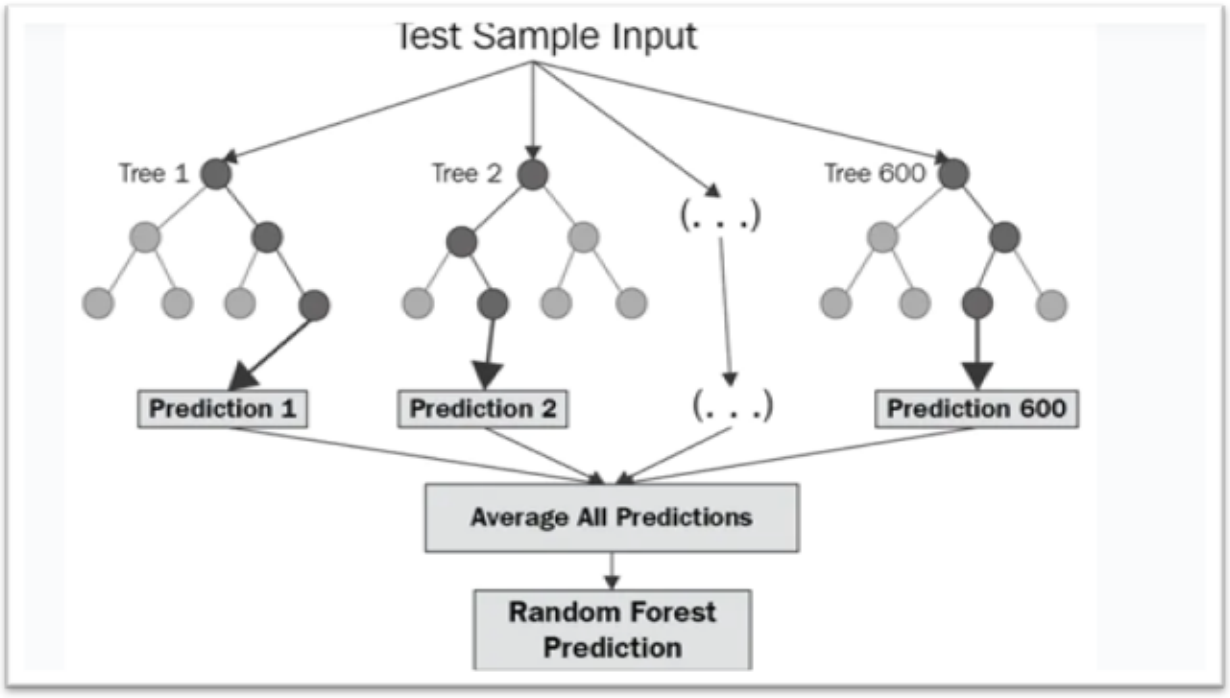


Figure 6: Structure of Random Forest

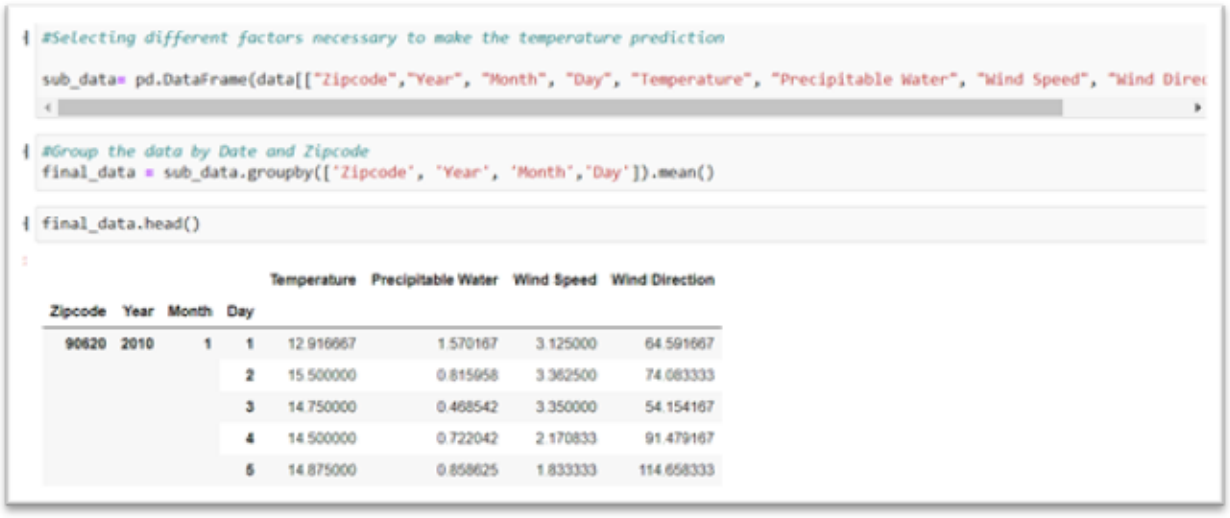
Advantages

- Random forests handle numerous variables and perform complex tasks at a fast pace.
- Works well with the missing data, and among presently available classification methods, random forests are highly accurate.
- It is helpful in both classification and regression tasks, and we can check the importance it allocates to the different input variables.

Data set:

We selected Wind Speed, Wind Direction, Precipitable Water, Temperature, Year, Month, Day from our dataset. Later, we grouped by Zipcode, Year, Month, and Day column to get the average temperature, average wind speed, average wind direction, and average precipitation.

The figure below is a snapshot of the data file selected to train the Random forest model.



```
#Selecting different factors necessary to make the temperature prediction

sub_data= pd.DataFrame(data[["Zipcode","Year", "Month", "Day", "Temperature", "Precipitable Water", "Wind Speed", "Wind Direc
```

```
#Group the data by Date and Zipcode
final_data = sub_data.groupby(['Zipcode', 'Year', 'Month','Day']).mean()
```

```
final_data.head()
```

				Temperature	Precipitable Water	Wind Speed	Wind Direction
Zipcode	Year	Month	Day				
90620	2010	1	1	12.916667	1.570167	3.125000	64.591667
			2	15.500000	0.815958	3.362500	74.083333
			3	14.750000	0.468542	3.350000	54.154167
			4	14.500000	0.722042	2.170833	91.479167
			5	14.875000	0.858625	1.833333	114.658333

Figure 7: Data Snapshot

Model Development

Further, we split the data file into train-test data with 60 percent training data and 40 percent testing data. We have considered 1000 random subsets of data and generated 1000 random decision trees with the optimal split of 3 for each decision tree to make the appropriate temperature predictions.

```
train_data, test_data, train_labels, test_labels= train_test_split(final_dataset,labels, test_size=0.40,random_state=4)

print('Training data shape:', train_data.shape)
print('Training labels shape:', train_labels.shape)
print('Testing data shape:', test_data.shape)
print('Testing label shape:', test_labels.shape)

Training data shape: (151110, 5)
Training labels shape: (151110,)
Testing data shape: (100740, 5)
Testing label shape: (100740,)

#Random Forest Regressor
rf=RandomForestRegressor(n_estimators=1000, random_state=3)
rf.fit(train_data,train_labels)

: RandomForestRegressor(n_estimators=1000, random_state=3)
```

Figure 8: Model Training

```
# actual y_test= test_labels
predictions=rf.predict(test_data)
print(predictions)

[12.27441667 13.31166667 12.173625   ... 19.43103333 16.98179167
 21.63741667]
```

Figure 9: Model Fitting

Model Accuracy

We got an accuracy of 99.35% with the Random forest model, which is the highest accuracy of all models. The mean absolute error rate is 0.31 degree Celsius, the Mean Squared error is 0.39 degree Celsius, and Root Mean squared error is 0.55 degree Celsius.

```
errors=abs(predictions - test_labels)
mape=100* (errors/test_labels)
accuracy=100-np.mean(mape/3)
print('Accuracy of the model:', round(accuracy,2),'%')

Accuracy of the model: 99.35 %

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(test_labels, predictions))
print('Mean Squared Error:', metrics.mean_squared_error(test_labels, predictions))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(test_labels, predictions)))

Mean Absolute Error: 0.312572262093839
Mean Squared Error: 0.30985164222616
Root Mean Squared Error: 0.5566431911253025
```

Figure 10: Performance Metrics

Below are graphical representations of four zip codes out of 69 Zip codes for predicted temperature versus Actual temperature. Due to the space constraint, only four graphs are presented.

Predicted Temperature

Actual Temperature

Predicting Future Temperature Values

With the help of predicted temperature values for their respective Date, we derived the relation between the two by finding the equation slope. We obtained the slope of the equation as 0.00074, and the p-value is 0.00, which is statistically significant.

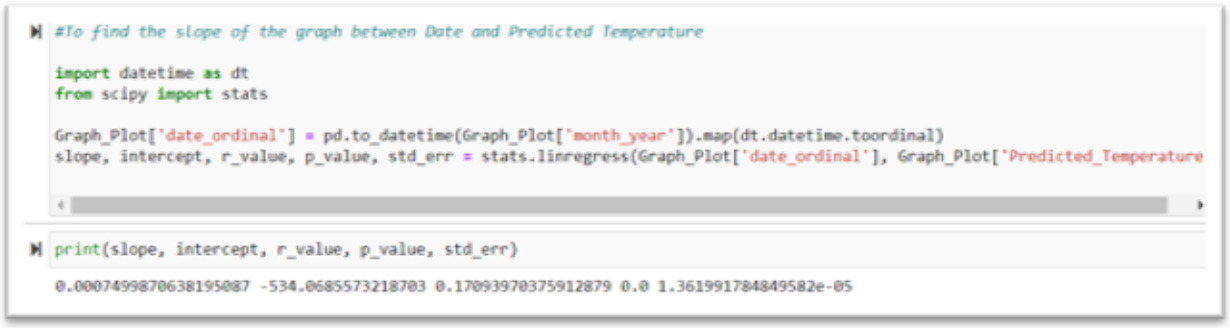


Figure 11: Significance

Using the below formula,

Predicted temperature = Avg. temperature of 2019(month) + slope * month

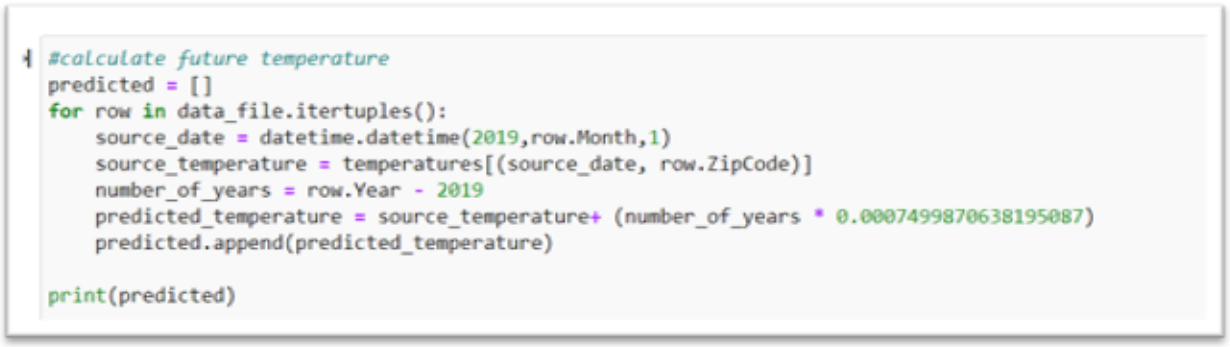


Figure 12: Model Predict

```
data_file.head(15)
```

	ZipCode	Year	Month	Day	Date	Predicted_Temperature
0	92624	2021	1	1	2021-01-01	14.668167
1	92624	2021	2	1	2021-02-01	12.274227
2	92624	2021	3	1	2021-03-01	14.668167
3	92624	2021	4	1	2021-04-01	17.101500
4	92624	2021	5	1	2021-05-01	16.701500
5	92624	2021	6	1	2021-06-01	19.589735
6	92624	2021	7	1	2021-07-01	22.215786
7	92624	2021	8	1	2021-08-01	23.801500
8	92624	2021	9	1	2021-09-01	24.092409
9	92624	2021	10	1	2021-10-01	20.924577
10	92624	2021	11	1	2021-11-01	18.287214
11	92624	2021	12	1	2021-12-01	13.751500
12	92624	2022	1	1	2022-01-01	14.668917
13	92624	2022	2	1	2022-02-01	12.274977
14	92624	2022	3	1	2022-03-01	14.668917

```
#Save the data_file in csv format
data_file.to_csv(r'C:\Users\CSUFTitan\Documents\ISDS_577\Future_Predicted_Temperatures.csv', index = False)
```

Figure 13: Prediction Output

We predicted the future temperature values for the next 25 years. We also generated the data file used to calculate the number of solar panels needed for each household unit of Orange County.

Machine Learning Model: Time Series Analysis using SARIMA

Wind Speed prediction using Time Series Analysis:

We used time-series analysis to predict the average monthly wind speed years after 2019 in Orange County. Using the data as far as 2010 is not a good indicator for today's wind speed as other factors might have influenced the wind speed.

Thus, for analysis and forecasting, we only use a subset of data from 2014 to 2019 as it accounts for more recent factors and gives a better understanding of the trend over the year.

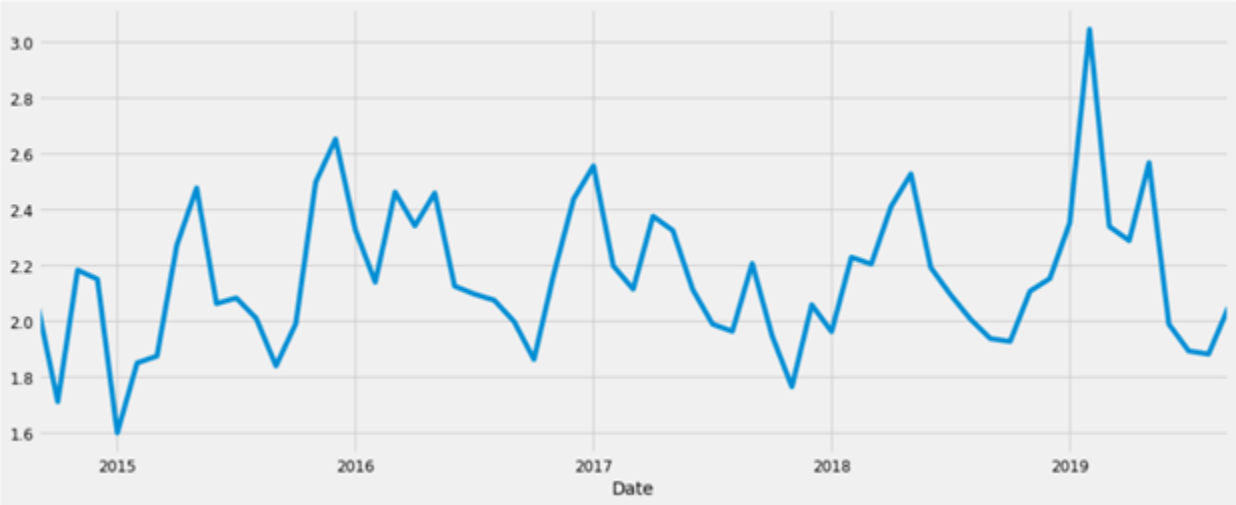


Figure 14: Wind Speed vs. Time trend

We observed a slightly increasing trend which looks to be about the same magnitude over time, so we use additive decomposition. We decompose the time series into trend, seasonal, and noise components.

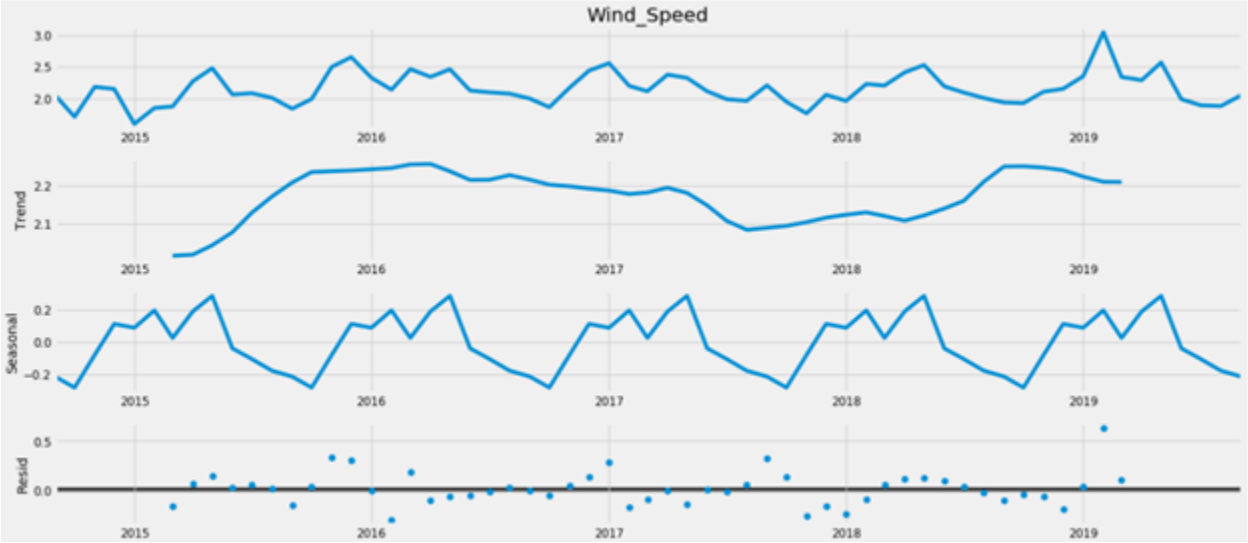


Figure 15: Decomposition of the time series

Before we can build a model, we must ensure that the time series is stationary. We used the rolling statistics method to determine the stationarity of the series.

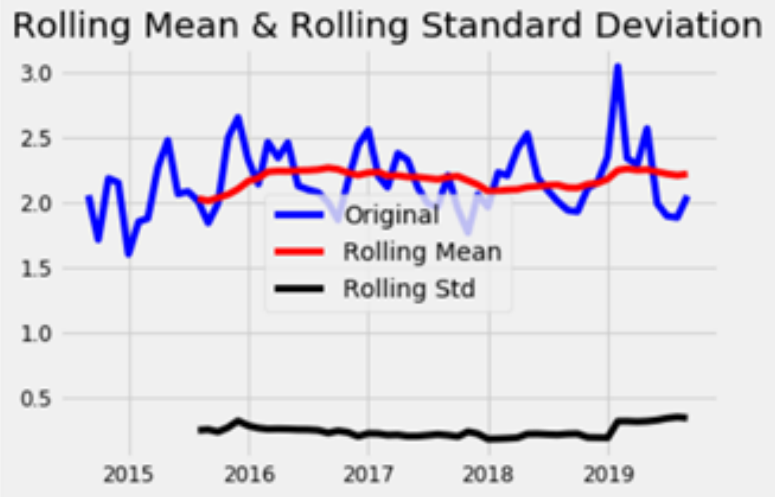


Figure 16: Rolling mean and rolling standard deviation graph

The time series is stationary as the rolling mean, and the rolling standard deviation is parallel to the X-axis and remains constant with time.

Seasonal ARIMA model

We used the SARIMA model to forecast future values after deciding that our time series is stationary. SARIMA is the model's abbreviation (p, d, q). The letters (P, D, Q) lag. These three parameters take seasonality, pattern, and noise into account in data. By employing grid search, we could automate the process of evaluating a large number of hyper-parameters. We created a function to compute the AIC (Akaike knowledge criterion), a mathematical method for evaluating how well our time series fits the data. We then used the p, d, and q values based on the lowest AIC score.

```
# Using Grid Search find the optimal set of parameters that yields the best performance
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(y, order = param, seasonal_order = param_seasonal,
                                            enforce_stationary = False,enforce_invertibility=False)
            result = mod.fit()
            print('ARIMA{}x{}12 - AIC:{}'.format(param, param_seasonal, result.aic))
        except:
            continue
```

Figure 17: Model Detail

Model Validation

We ran a diagnostics test to check for the behavior of the residuals. The residuals have constant variance and are normally distributed, indicating that the model has adequately captured the information in the data.

The red KDE line closely resembles the N(0,1) line in the top right plot (where N(0,1) is the regular notation for a normal distribution with mean 0 and standard deviation of 1), implying that the residuals are normally distributed.

The Q-Q plot on the bottom left shows that the ordered residual distribution (blue dots) follows the linear pattern of samples drawn from a regular normal distribution with N. (0, 1). Again, this strongly suggests that the residuals are usually dispersed.

The residuals over time (top left plot) do not display seasonality. The autocorrelation (i.e., correlogram) plot on the bottom right confirms this argument. It also demonstrates that the time series residuals have a low association with lagged versions of themselves.

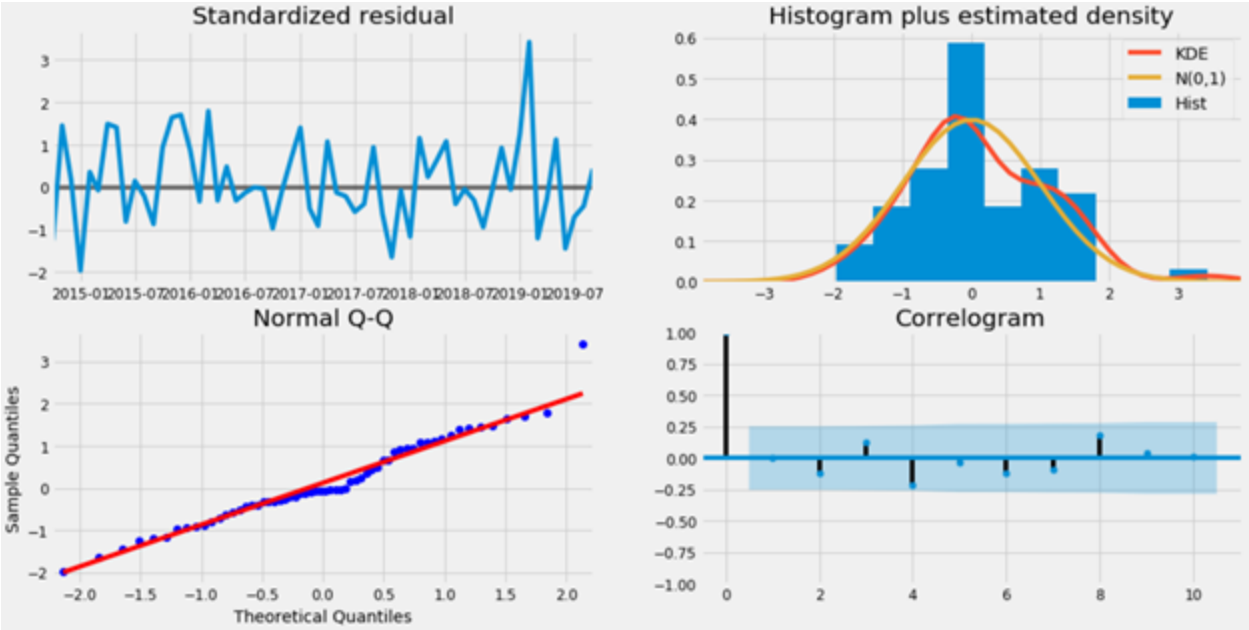


Figure 18: Residual plots

In-sample prediction

We compare the predicted wind speed to the actual wind speed of the time series and set forecasts to start from "2016-09-01" till "2019-09-13".

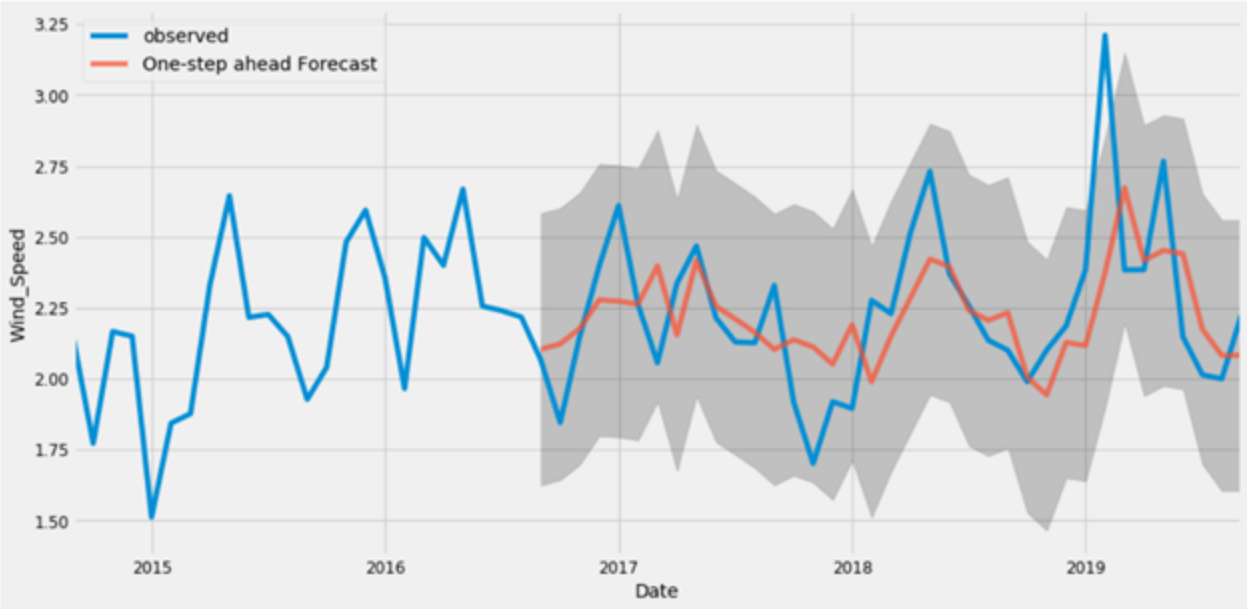


Figure 19: In-sample forecast

Model Accuracy

We checked for errors and found that the MSE and RMSE values, 0.05 and 0.22, respectively, are low, indicating that the predicted values are closer to the actual forecast.

Forecasting for out of sample data

We then predicted the wind speed until 2025 using the get_forecast() function in statsmodels with a confidence interval of 95%.

```
# forcasting for out of sample data
pred_uc = result.get_forecast(steps = 75)
pred_ci = pred_uc.conf_int()

ax = y.plot(label = 'observed', figsize = (14, 7))
pred_uc.predicted_mean.plot(ax = ax, label = 'forecast')
ax.fill_between(pred_ci.index, pred_ci.iloc[:, 0], pred_ci.iloc[:, 1], color = 'k', alpha = 0.25)
ax.set_xlabel('Date')
ax.set_ylabel('Wind Speed')

plt.legend()
plt.show()
```

Figure 20: Forecast Code

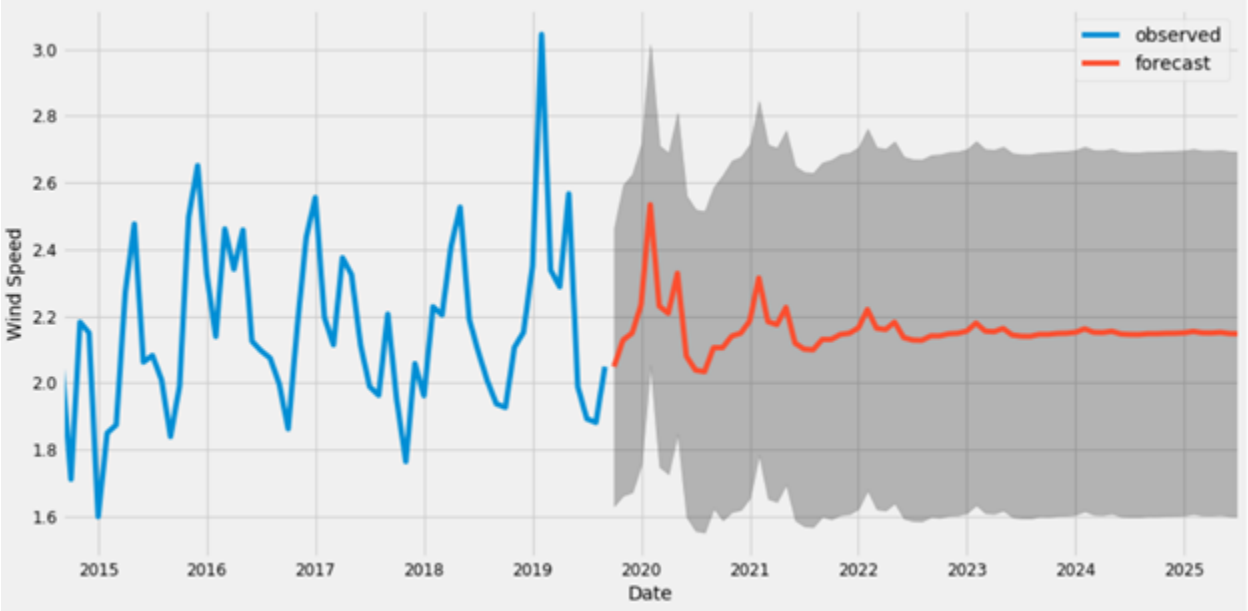


Figure 21: Out of Sample forecast (till 2025)

	lower Wind_Speed	upper Wind_Speed
2019-10-01	1.631032	2.465607
2019-11-01	1.662786	2.593508
2019-12-01	1.671789	2.625492
2020-01-01	1.754697	2.714641
2020-02-01	2.052823	3.014684
...	...	...
2025-08-01	1.599844	2.693385
2025-09-01	1.600908	2.694450
2025-10-01	1.600820	2.694540
2025-11-01	1.601286	2.695090
2025-12-01	1.601397	2.695240

75 rows × 2 columns

Figure 22: Forecast Output

As we forecast further out into the future, we become less confident in our values. It is reflected by the confidence intervals generated by our model, which grows larger as we move further out into the future.

Research question 2: How has solar irradiation changed over the past ten years?

The purpose of this research question is to analyze the trend in solar irradiation observed between the years 2010 - 2019 and search for any significant changes over time. It will provide valuable input into our model to predict the number of solar panels required for a particular household located within a zip code. We combined the variables “Year,” “Month,” and "Day" to create an index column titled “Date”. Variable ”GHI” and variable “Zipcode” were selected to find the average monthly GHI for each zip code. The results were displayed using line charts for

each zip code. A total of 69 different charts were created by running a “for” loop. Due to space constraints, only four charts have been shown in Figure 23 below.

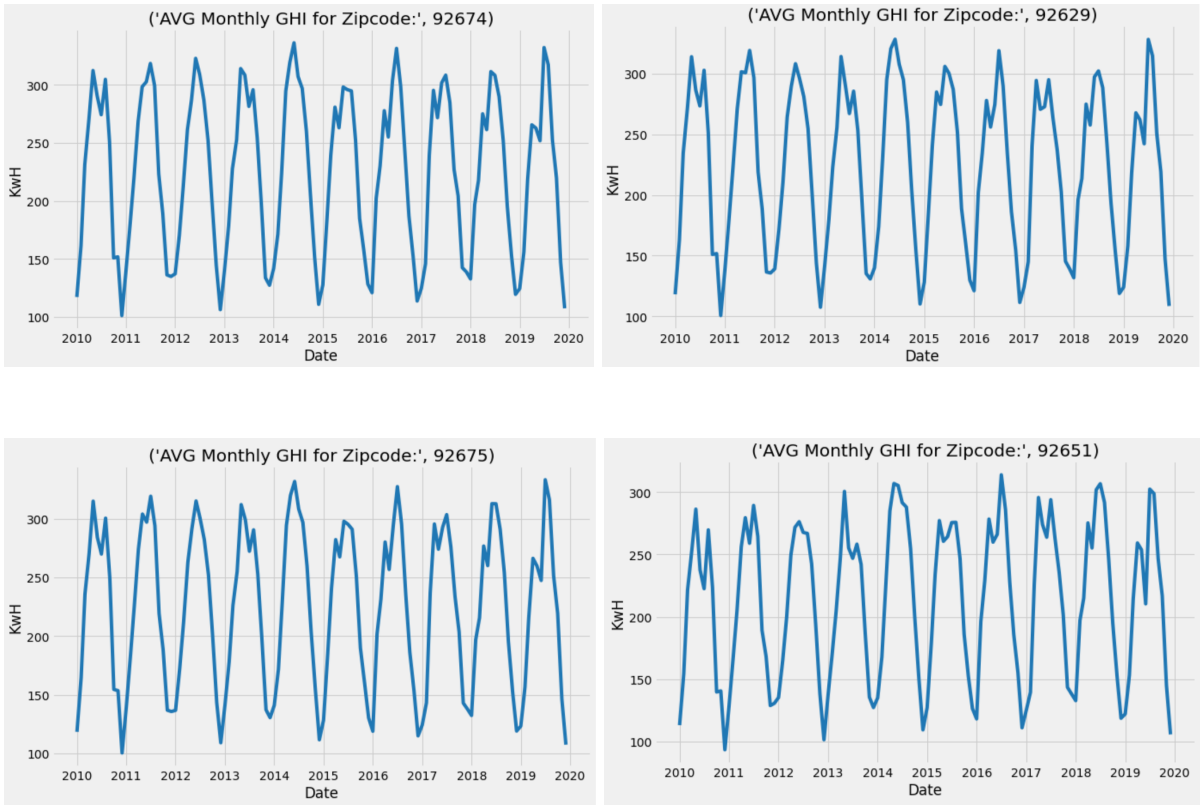


Figure 23: AVG Monthly GHI for four Zipcodes

From the charts, we see that the general trend observed for average monthly GHI from year to year seems consistent. There are seasonal increases and decreases throughout the year, but no significant increases are seen over time. Next, we look at the total GHI for each year. Again, this has been done for each zip code totaling 79 different line charts. Four of the charts are shown in Figure 24 below.

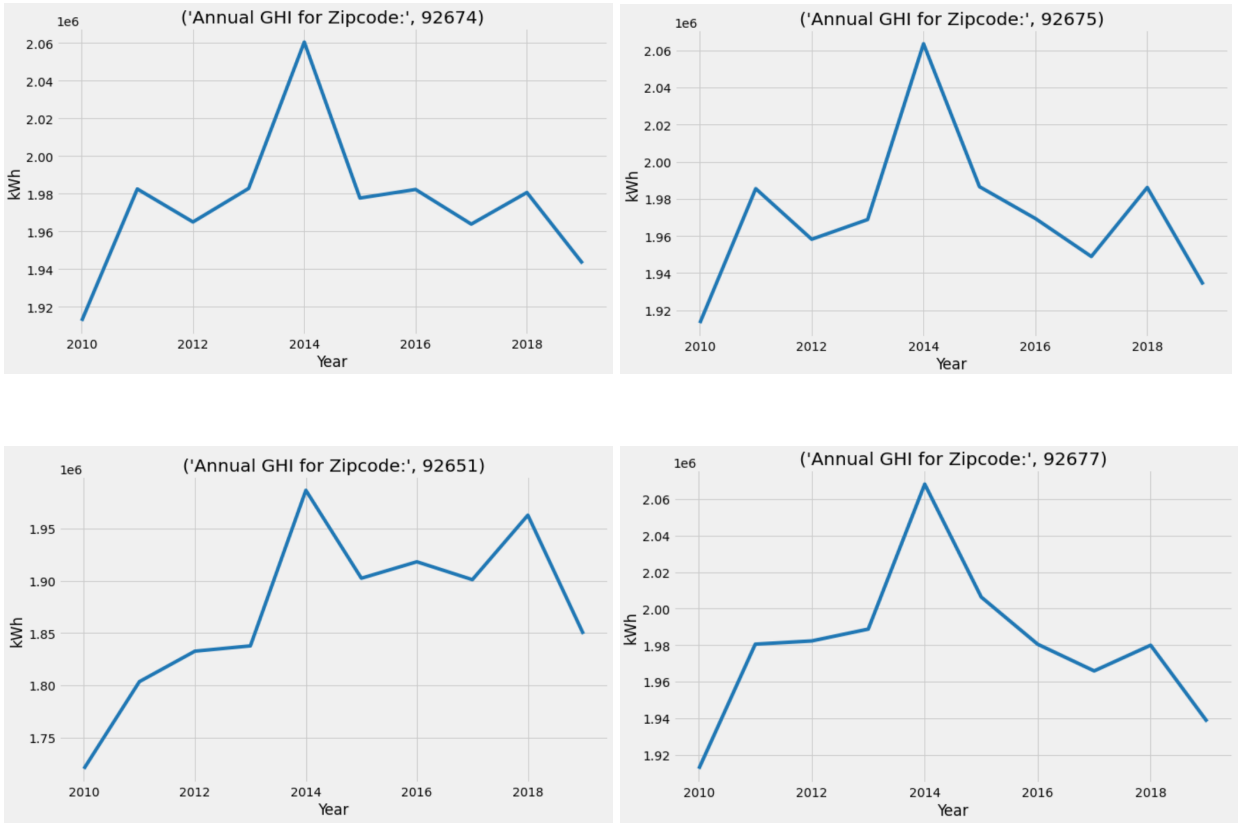


Figure 24: Total GHI for four zip codes

We see differences in total GHI year over year vary according to zip code from the charts above. We also observe a higher total GHI for the year 2014. However, there doesn't seem to be a linear increasing or decreasing pattern observed in total GHI over the years. We also look at the percentage change in GHI for each zip code over the years to determine whether there has been a significant change from one year to another. Results for four zip codes are shown in figure 25 below.

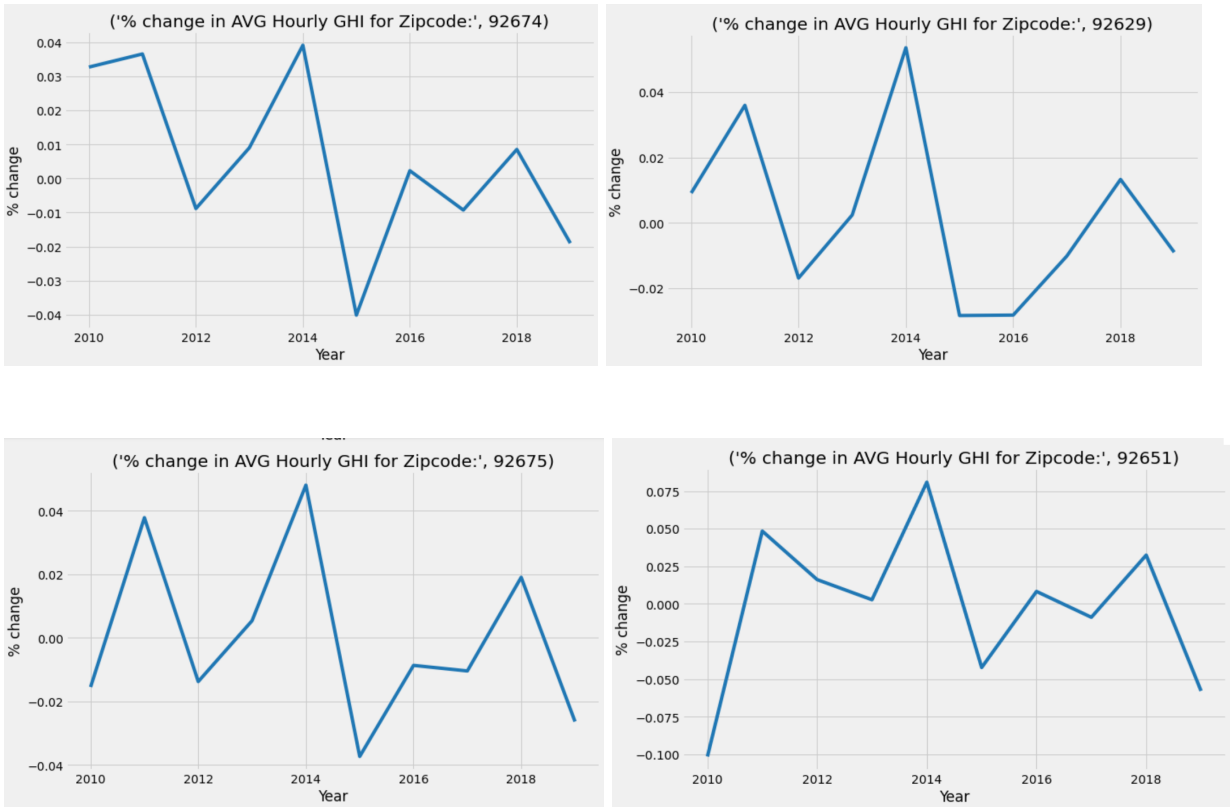


Figure 25: Percentage change in AVG hourly GHI for four zip codes

From the charts above, we observe that the percentage change in hourly average GHI for each year varies by a maximum of 0.1% for the zip code 92651. At the same time, the percentage change for the other three locations remains between 0.04% – 0.05%. From our analysis of GHI data for each zip code, we can conclude no noticeable increase or decrease pattern in GHI between the years 2010 to 2019. There have been changes in hourly and total GHI over the years, and they vary based on the zip code. However, the percentage change does not exceed a value of +- 0.1%.

Research question 3: How many solar panels needed for generating the power required to support a household within Orange County?

This section uses the result from the previous sections to predict the number of solar panels required for a household with our model. The model receives user inputs such as annual power usage and the zip code where the home is located within Orange County. In research question 2, we observed average monthly GHI, percentage change in hourly average GHI and total annual GHI data for each zip code. We also plot the hourly GHI for each zip code and find summary statistics to determine the significance of the change in GHI over the years.

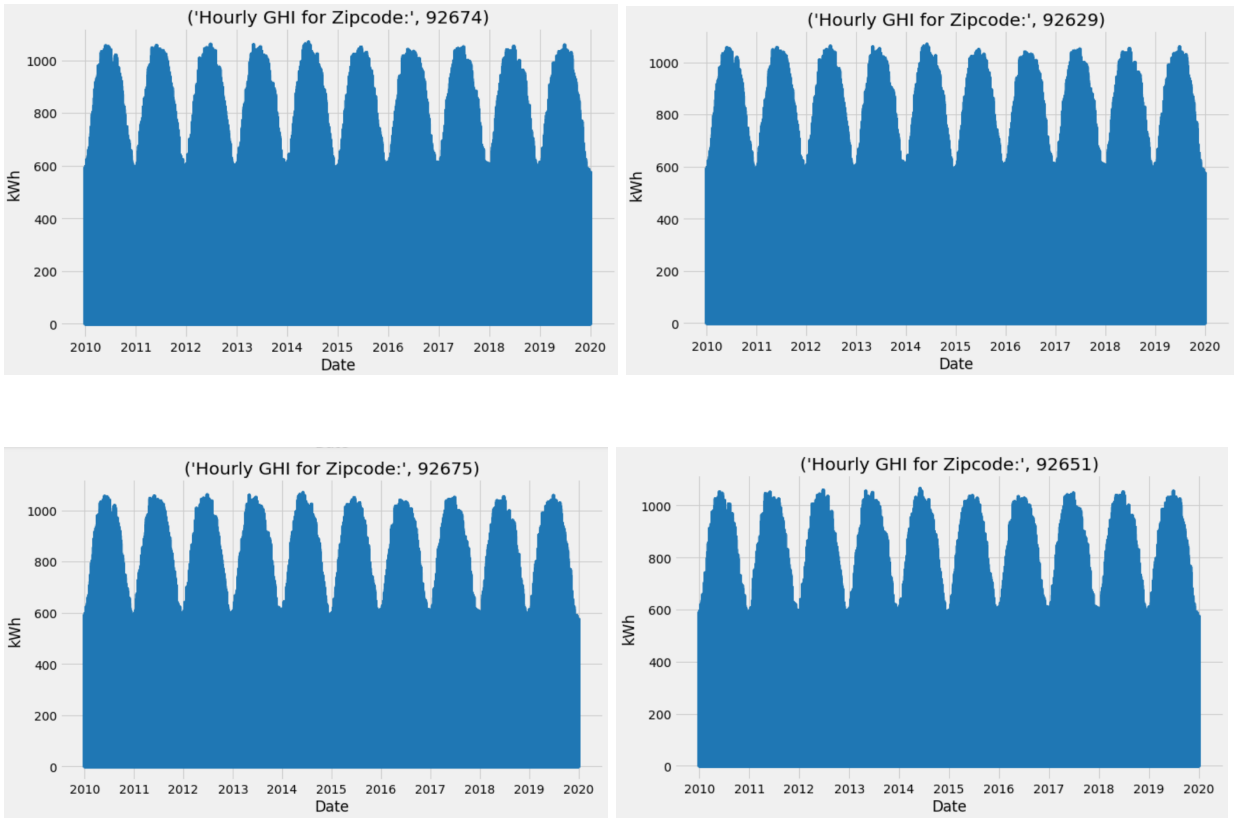


Figure26: Hourly GHI for four zip codes

In the figure above, we see the hourly GHI data of 10 years plotted for four zip codes selected from a total of 69 charts. The charts do not display a noticeable trend within the GHI data. And no increase or decrease patterns were observed.

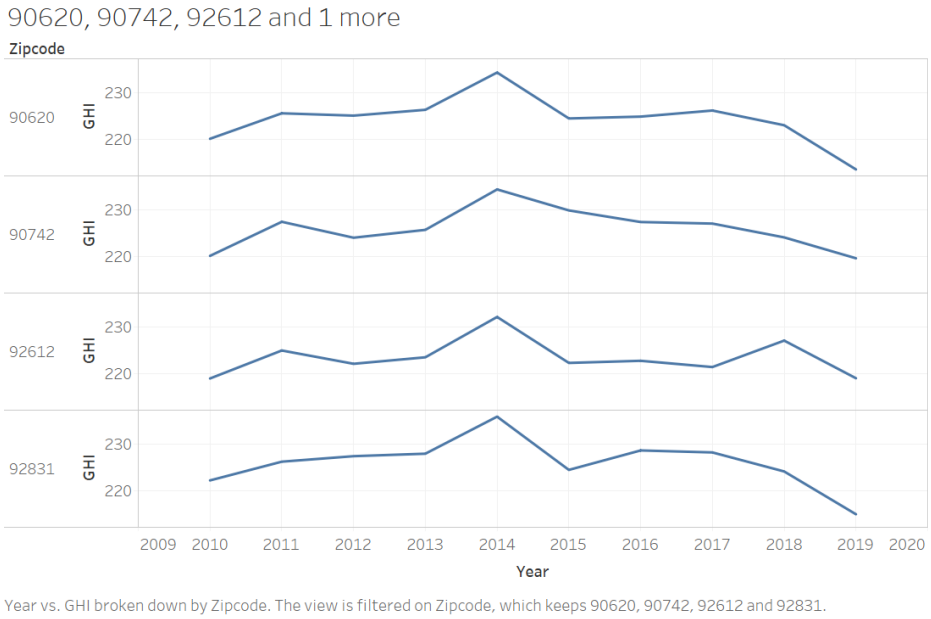


Figure 27: Year vs. GHI broken down by zip code

The figure above shows us a combined graph to better understand the GHI pattern over the past ten years. The chart clearly shows us an identical pattern in GHI.

We see the summary statistics for the hourly GHI data in the figure below.

```
Slope: -0.0013138405803907569
Intercept: 1194.2749512440957
R-square: -0.004396634083956001
p-value: 0.193165170973288
Std_error: 0.0010096504808804673
```

Figure28: Summary statistics for hourly GHI data

The summary statistics show a negative slope value for the hourly GHI data and a positive intercept. The p-value is much higher than 0.05, and it indicates that the significance of the slope of the graph of hourly GHI data and Date is relatively low. Thus, we conclude that using a daily average of GHI based on the ten years of data is a better alternative than predicting future values for GHI.

Predicting Number of Solar Panels

To calculate the number of necessary solar panels for a household, we create a new data frame containing relevant variables such as “Date” “Zipcode” and “GHI”. We then find the total daily GHI for each zip code using the “groupby” function. Next, we use the “groupby” function again to get the daily average GHI for each zip code. It is used in the calculation to determine the number of solar panels required.

In addition to finding the average daily GHI for each zip code, we also require estimates for temperature, temperature coefficient, solar panel output efficiency, and solar panel annual degradation rate.

The temperature has a significant effect on solar panel output efficiency. Research shows that solar panels work best in lower temperatures and are tested at 25 degrees Celsius (Does Solar

Panel Temperature Coefficient Matter, 2018). A one degree Celsius increase in temperature would mean an average of 0.28 decrease in efficiency of solar panel output. -0.28 is the temperature coefficient that will be used for our model.

To find the decrease in efficiency caused by temperature, we require temperature estimates for the next 25 years. These estimates are predicted using the slope from the random forest model created in research question 1 for 2021 to 2045. Since we only care about the values higher than 25 degrees Celsius, we use an IF function in Microsoft Excel to find the decrease in efficiency caused by values greater than 25. We then read the data using pandas in Python and find the average decrease in efficiency for each zip code for 25 years.

Once we have the values for average daily GHI and decrease in efficiency caused by an increase in temperature, we create a dictionary with “Zipcodes” as keys and tuples of “GHI”, ”Temperature_efficiency” as values for efficient retrieval of data. Furthermore, we use average industry values of 19.10% and 0.8% for solar panel output efficiency and solar panel annual degradation rates, respectively, in the calculations.

```
final_df = {}
for row in df5.itertuples():
    key = row.Zipcode
    val = (row.GHI, row.Temperature_efficiency)
    final_df[key] = val

print(final_df)
```

```
{90620: (5385.757534246575, 100.0), 90621: (5423.654246575343, 100.0),
```

Figure29: Dictionary of “Zipcodes”, “GHI”, and "Temperature_efficiency"

The last step in predicting the number of solar panels required is to create functions to receive user input for annual power needs in kWh of a household and the zip code for the home's location and perform calculations using different estimates. The following code is used to receive user input and convert the data types to float and NumPy int variables.

```
annual_power = input("Please enter your annual power (kWh) needs in numeric value: ")
postal_code = input("Please enter your zipcode in numeric value: ")

try:
  annual_power = float(annual_power)
except:
  print("Error, your input for annual power is not a numeric value!")
  annual_power = input("Please try again: ")

try:
  postal_code = np.int(postal_code)
except:
  print("Error, your input for zipcode is not a numeric value!")
  postal_code = input("Please try again: ")
```

Figure30: User input and data type conversion code

Once the model received user input of annual power needs and postal code, we calculate the daily power needs of the household. We also run a loop to match the postal code with the zip codes in the dictionary and extract values for average daily GHI and efficiency decrease caused by temperature for that particular zip code.

```
try:
  for i in final_df:
    if i == postal_code:
      panel_output = (panel_efficiency*degradation_rate*final_df[i][0]*(final_df[i][1]/100))
      num_panels = math.ceil(daily_power/panel_output)
      print("You will need", num_panels, "solar panels to fulfill your power needs for the next 25 years.")
    else:
      continue
except:
  print("Error! Try again")
```

```
Please enter your annual power (kWh) needs in numeric value: 10800
Please enter your zipcode in numeric value: 92831
You will need 36 solar panels to fulfill your power needs for the next 25 years.
```

Figure31: for loop to calculate the number of solar panels required

The for loop above takes the values for average daily GHI and multiplies it with solar panel output efficiency, solar panel annual degradation rate, and efficiency decrease caused by temperature to find output from one solar panel. The number of solar panels is calculated using the panel output and the daily power needs. The figure above shows the output from the algorithm.

Conclusion:

We perform a time series analysis on the hourly solar irradiance and meteorological data for ten years to find the wind speed to predict temperature. We created a random forest model to predict temperature values in each zip code for 25 years.

We analyze the solar radiation values from 2010 to 2019 to check for any trend in the data to help predict the number of solar panels. Line charts for the 69 Zipcodes show no significant changes for 2010 to 2019. We summarize the statistics for hourly GHI values and find that the change in GHI is not statistically significant and hence take the average values for GHI based on the past ten years of data.

We take postal code and the annual power needed in kWh for a household as users input. The power required is calculated using average industry values of 19.10% and 0.8% for solar panel output efficiency and solar panel annual degradation rates. Solar panels are most efficient at lower temperatures. The efficiency decreases with every 1 degree Celsius increase. We take the temperature coefficient of -0.28 to account for the temperature change. Hence when calculating the number of solar panels, we also consider the temperature change.

We get the panel output value for one solar panel using panel output efficiency, solar panel annual degradation rate, and efficiency decrease caused by temperature. Finally, we get the number of solar panels for 25 years by dividing the daily power needs for the household by the power generated from one solar panel.

References:

Cheng, L. (2019, January 2). *Basic Ensemble Learning (Random Forest, AdaBoost, Gradient Boosting)- Step by Step Explained*. Medium. https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Corporate Finance Institute. (2021, April 12). *Random Forest*. https://corporatefinanceinstitute.com/resources/knowledge/other/random-forest/

Halper, E. (2014, August 30). Rules prevent solar panels in many states with abundant sunlight. *Los Angeles Times*. https://www.latimes.com/nation/la-na-no-solar-20140810-story.html

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice, 2nd edition*. OTexts. https://otexts.com/fpp2/seasonal-arima.htm

Jordan, D.C. & Kurtz, S. R. (2012). *Photovoltaic Degradation Rates — An Analytical Review.* Progress in Photovoltaics: Research and Applications. Retrieved on 05/14/2021 from https://www.nrel.gov/docs/fy12osti/51664.pdf

Matasci, S. (2021, May 7). *What are the best solar panels available? Top brands and products compared*. Solar News. https://news.energysage.com/best-solar-panels-complete-ranking/

Ost, I. (2020, April 20). *Does Solar Panel Temperature Coefficient Matter?* Solar.Com. https://www.solar.com/learn/does-solar-panel-temperature-coefficient-matter/

Richardson, L. (2021, May 3). *How long do solar panels last? Solar panel lifespan explained*. Solar News. https://news.energysage.com/how-long-do-solar-panels-last/