# Solar Power Generation


Jainil Patel, Harsh Gandhi

Authors


## Courses Covered

- Exploratory Data Analysis
- Python For Data Science
- Data Preprocessing

# Topic Background

The world’s energy consumption is outpacing supply due to population growth and technological advancements.
For future energy demands, it is critical to progress toward a dependable, cost-effective, and sustainable renewable energy source.
Solar energy, along with all other alternative energy sources, is a potential renewable resource to manage these enduring challenges in the energy crisis.
Solar power generation is expanding globally as a result of growing energy demands and depleting fossil fuel reserves, which are presently the primary sources of power generation.
In the realm of solar power generation, photovoltaic (PV) panels are used to convert solar radiation into energy.
They are subjected to the constantly changing state of the environment, resulting in a wide range of defects.

In this presentation we focus on certain key things:
Solar power prediction.

- Using Machine learning model for timely fault detection.
- Combining weather and sensor based data to draw anomaly detection insights.
- Consolidating various Exploratory Data Analysis Plots for better understanding of the data.
- Using NULL values to identify missing values in daily yield production.

# Dataset Description

This data has been gathered at two solar power plants over a 34 day period.

The power generation datasets are gathered at the inverter level - each inverter has multiple lines of solar panels attached to it. It has two pairs of files - each pair has one power generation dataset and one sensor readings dataset. The sensor data is gathered at a plant level - single array of sensors optimally placed at the plant.

It a Date-time series dataset with interval of 15 minutes. Areas of concern:

- Grid power management and power prediction.
- Panel Maintenance.
- Faults and Defects Detection.

# Dataset Description

Plant_1_Generation_Data

Plant_2_Generation_Data

## Attributes:

- **DATE_TIME**: Date and time for each observation. Observations recorded at 15 minute intervals.
- **PLANT_ID**: this will be common for the entire file
- **SOURCE_KEY**: Source key in this file stands for the inverter id.
- **DC_POWER**: Amount of DC power generated by the inverter (source_key) in this 15 minute interval. Units - kW.
- **AC_POWER**: Amount of AC power generated by the inverter (source_key) in this 15 minute interval. Units - kW.
- **TOTAL_YIELD**: This is the total yield for the inverter till that point in time.

# Dataset Description

Plant_1_Weather_Sensor_Data

Plant_2_Weather_Sensor_Data

## Attributes:

- SOURCE_KEY: Stands for the sensor panel id. This will be common for the entire file because there's only one sensor panel for the plant.
- AMBIENT_TEMPERATURE: This is the ambient temperature at the plant. Note: After comparing this data with weather data in Gandikotta (Andhra), I assume the correct unit for this data is °C
- MODULE_TEMPERATURE: There's a module (solar panel) attached to the sensor panel. This is the temperature reading for that module. Note: After comparing this data with other publications, I assume the correct unit for this data is °C
- IRRADIATION: Amount of irradiation for the 15 minute interval. Note: After comparing this data with other publications, I assume the correct unit for this data is kW/m2

# Dataset Description

Plant_1_Generation_Data

Plant_1_Weather_Sensor_Data

Plant_2_Generation_Data

Plant_2_Weather_Sensor_Data

## Attributes:

| | DC_POWER | AC_POWER | DAILY_YIELD | TOTAL_YIELD |
|---|---|---|---|---|
| count | 3158.000000 | 3158.000000 | 3158.000000 | 3158.000000 |
| mean | 68547.713729 | 6703.628149 | 71782.817545 | 151989182.227813 |
| std | 88044.612181 | 8603.120476 | 65974.417997 | 10616697.925153 |
| min | 0.000000 | 0.000000 | 0.000000 | 26540040.000000 |
| 25% | 0.000000 | 0.000000 | 90.750000 | 152097608.231750 |
| 50% | 8515.285714 | 823.033036 | 66068.000000 | 153531986.983000 |
| 75% | 140386.504463 | 13750.606696 | 129398.500000 | 154994999.624000 |
| max | 298937.785710 | 29150.212499 | 193770.000000 | 156142755.000000 |

| | AMBIENT_TEMPERATURE | MODULE_TEMPERATURE | IRRADIATION |
|---|---|---|---|
| count | 3182.000000 | 3182.000000 | 3182.000000 |
| mean | 25.531606 | 31.091015 | 0.228313 |
| std | 3.354856 | 12.261222 | 0.300836 |
| min | 20.398505 | 18.140415 | 0.000000 |
| 25% | 22.705182 | 21.090553 | 0.000000 |
| 50% | 24.613814 | 24.618060 | 0.024653 |
| 75% | 27.920532 | 41.307840 | 0.449588 |
| max | 35.252486 | 65.545714 | 1.221652 |

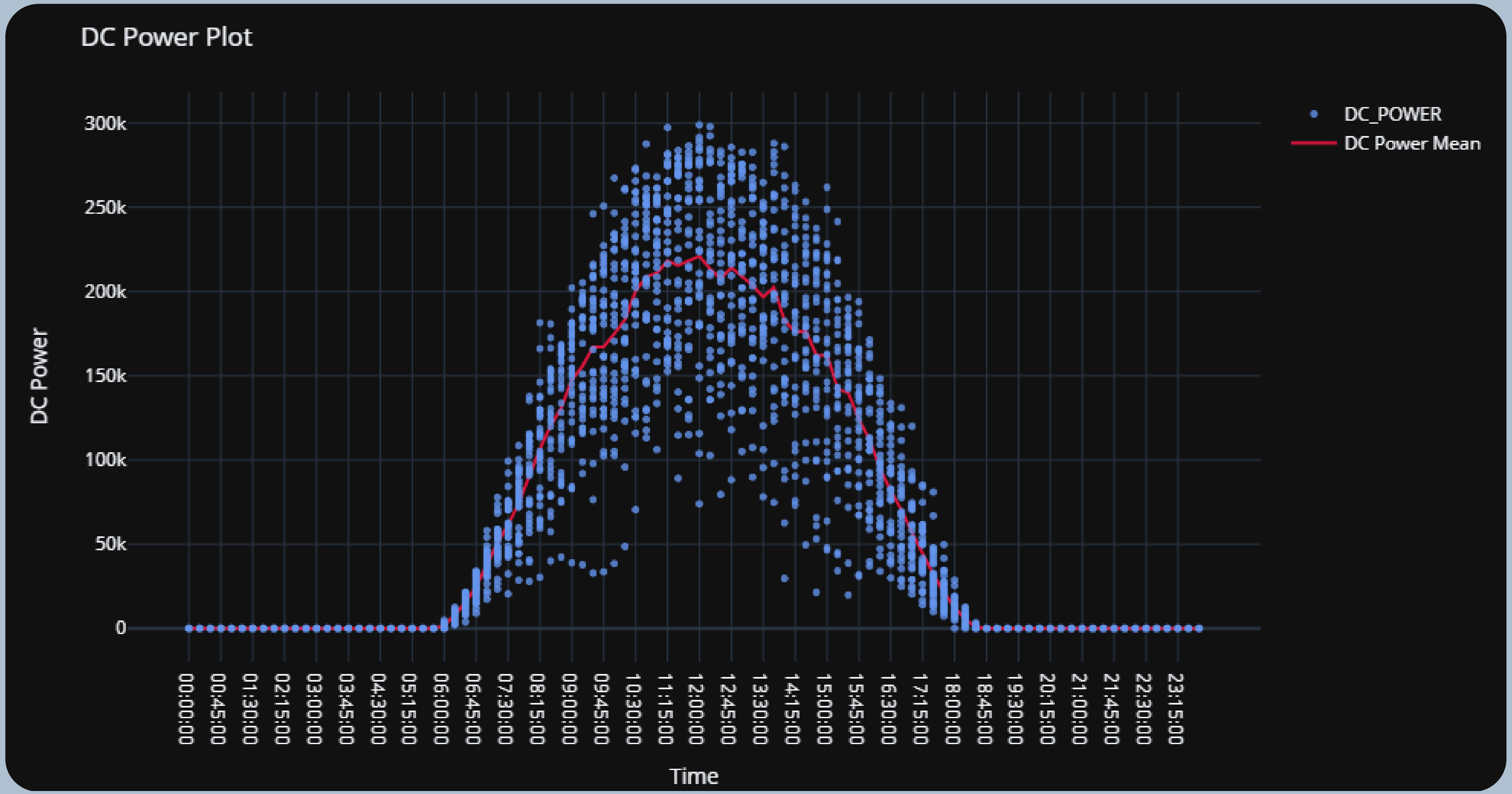
# Data Preprocessing

After applying groupby function on 'DATE_TIME' column for the DC_POWER,  AC_POWER, DAILY_YIELD, TOTAL_YIELD columns we eliminate the SOURCE_KEY column so by following this snippet we narrowed down the data of individual inverters by summing up their per 15 minute data. Thus, we obtained the day wise data for the plant.

```
plant1_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68778 entries, 0 to 68777
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   DATE_TIME    68778 non-null  object 
 1   PLANT_ID     68778 non-null  int64  
 2   SOURCE_KEY   68778 non-null  object 
 3   DC_POWER     68778 non-null  float64
 4   AC_POWER     68778 non-null  float64
 5   DAILY_YIELD  68778 non-null  float64
 6   TOTAL_YIELD  68778 non-null  float64
dtypes: float64(4), int64(1), object(2)
memory usage: 3.7+ MB
```

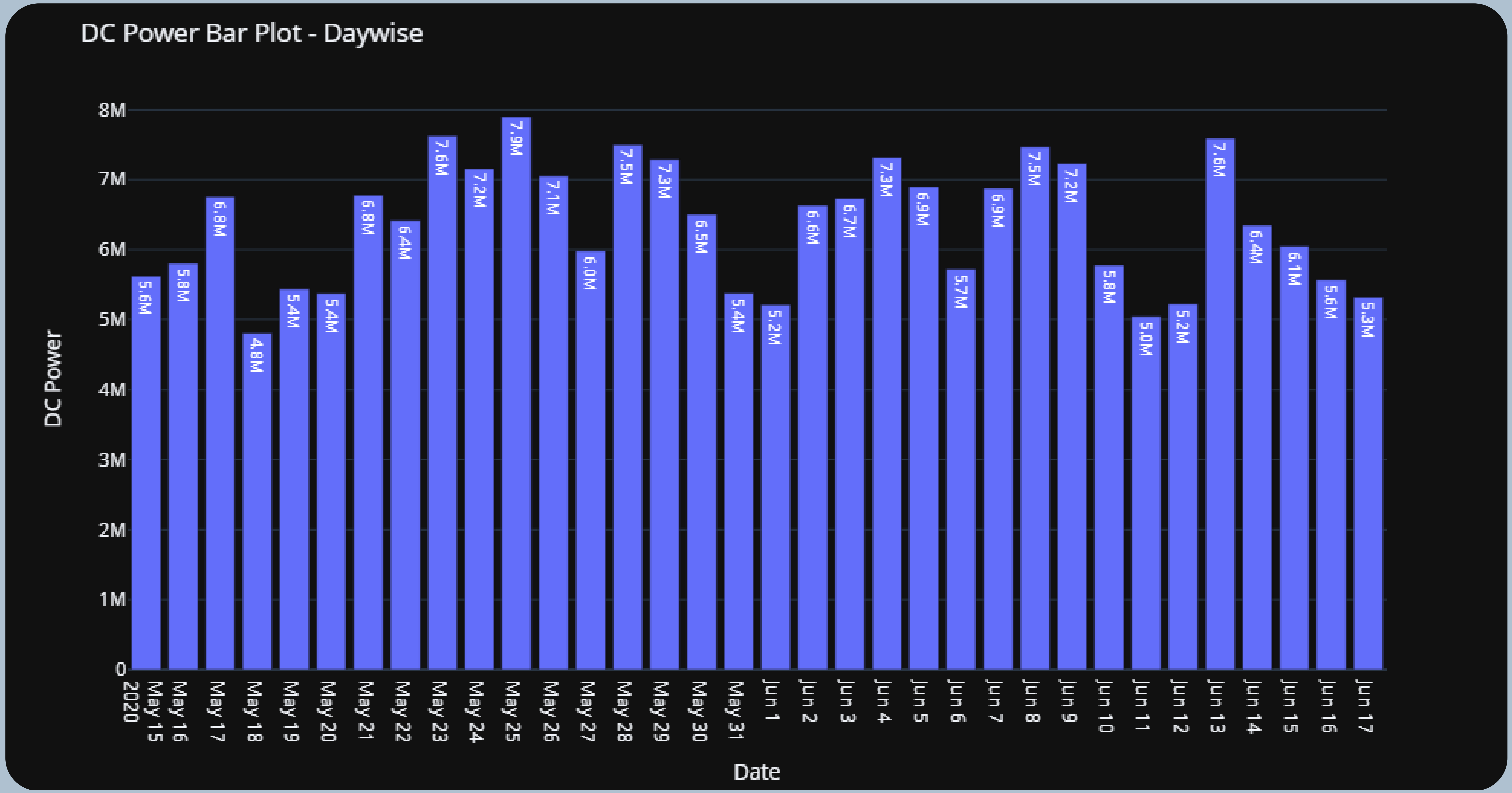
```
plant1_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3158 entries, 0 to 3157
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   DATE_TIME    3158 non-null   datetime64[ns]
 1   DC_POWER     3158 non-null   float64       
 2   AC_POWER     3158 non-null   float64       
 3   DAILY_YIELD  3158 non-null   float64       
 4   TOTAL_YIELD  3158 non-null   float64       
 5   time         3158 non-null   object        
 6   date         3158 non-null   datetime64[ns]
dtypes: datetime64[ns](2), float64(4), object(1)
memory usage: 172.8+ KB
```

# Exploratory Data Analysis



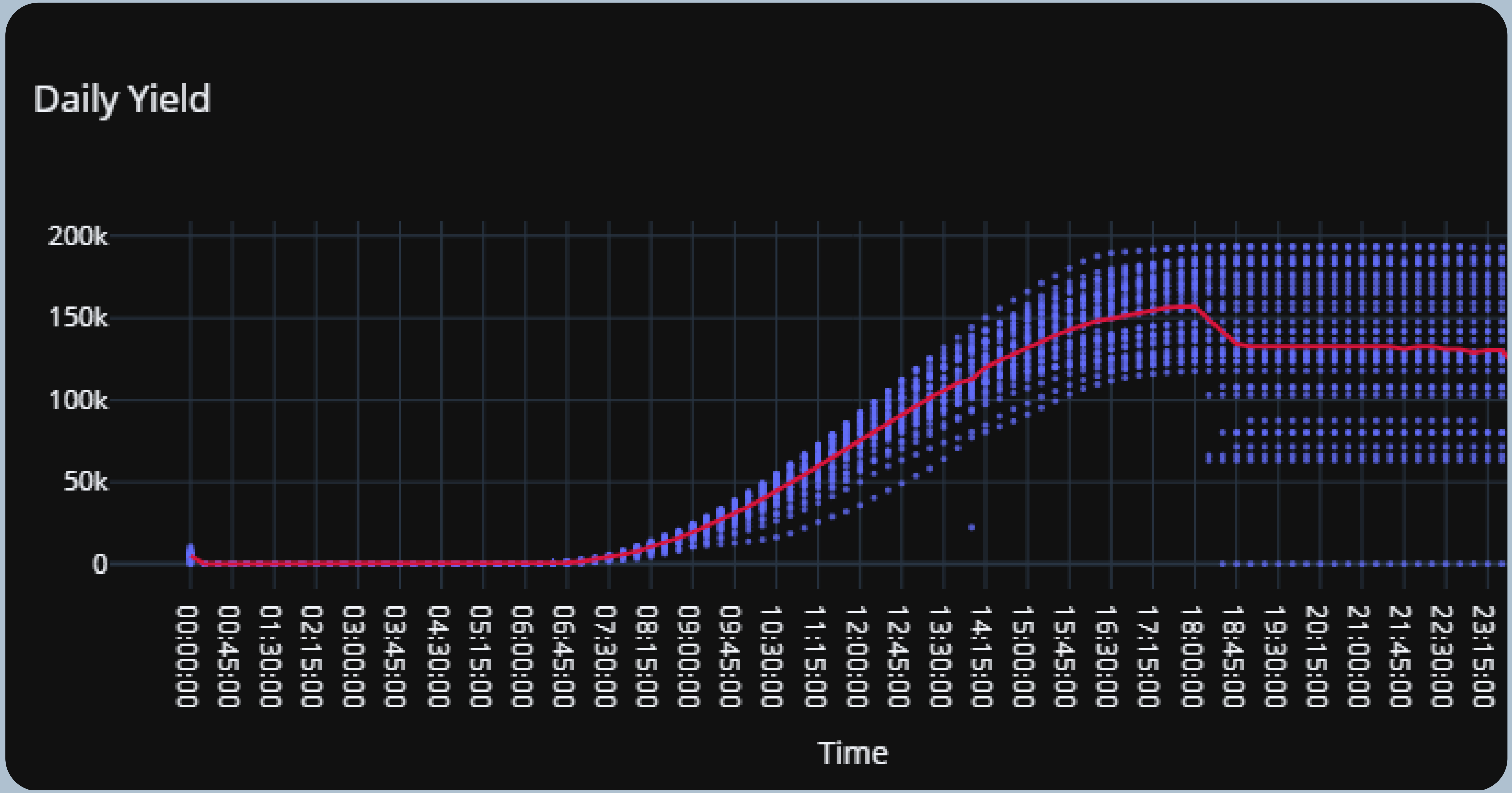

Between 05:33:20 and 18:00:00, the Plant generates a DC Power but where as during remaining hours of the day there was no power generation. The reason is sunlight.

# Exploratory Data Analysis



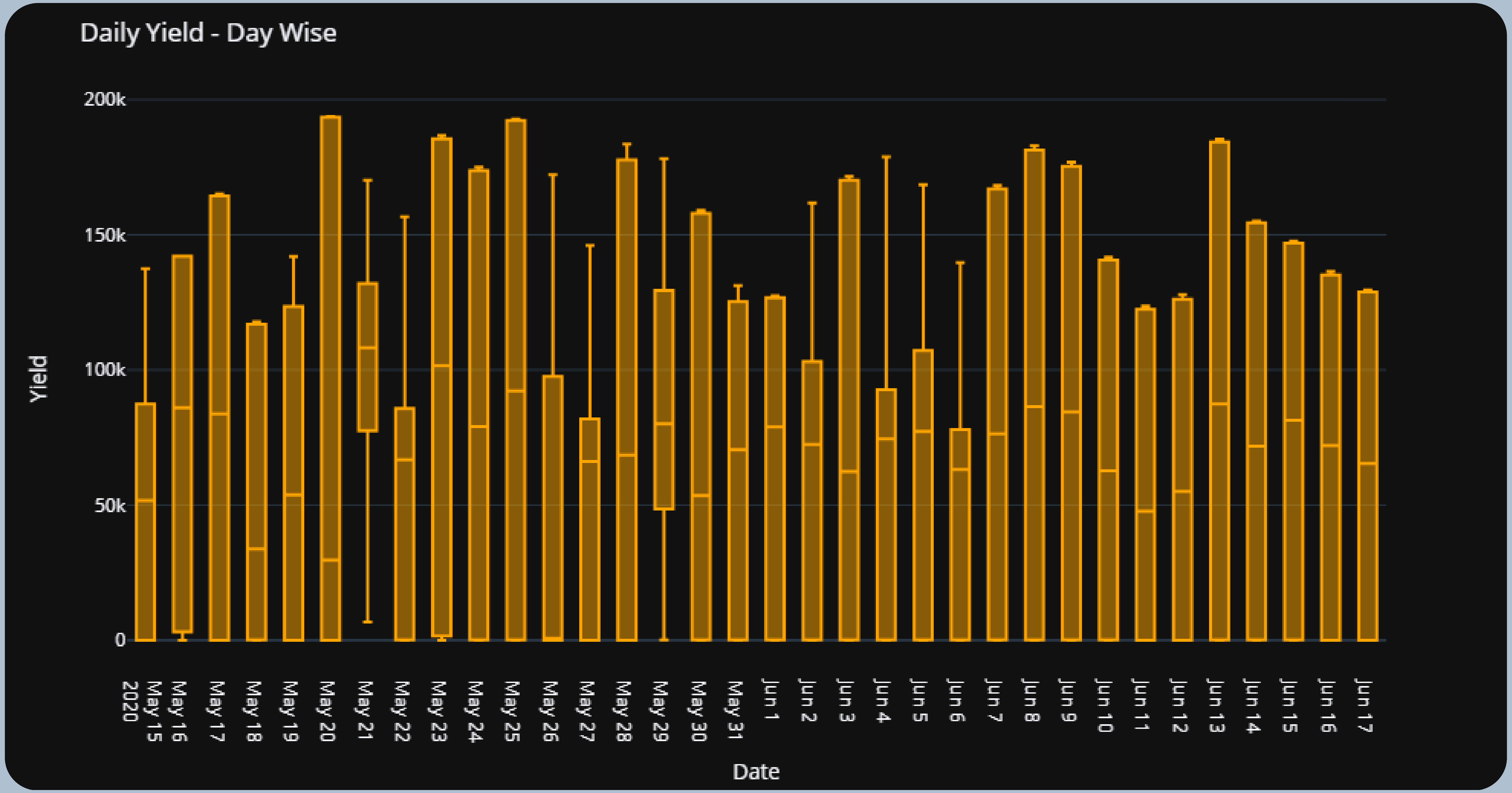

Dc Power is maximun on May 25 .

# Exploratory Data Analysis




The above logistics function like graph indicates that after 18:00 PM the power production decreases slowly, and usually at 00:00 AM the graph shows sudden breakdown indicating the production according to various time slots of the day.

# Exploratory Data Analysis




- For each day, the daily yield change.
- some day is high.
- The observation of all boxes is good, outliers does not exist.

# Machine Learning Models

### LinearRegressor

LinearRegression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

### RandomForestRegressor

In random forests each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.

### DecisionTreeRegressor

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

# Machine Learning Models

```
Mean absolute error = 25.9
Mean squared error = 3219.04
Median absolute error = 8.41
Explain variance score = 0.98
R2 score = 97.92 %
```

Linear Regressor

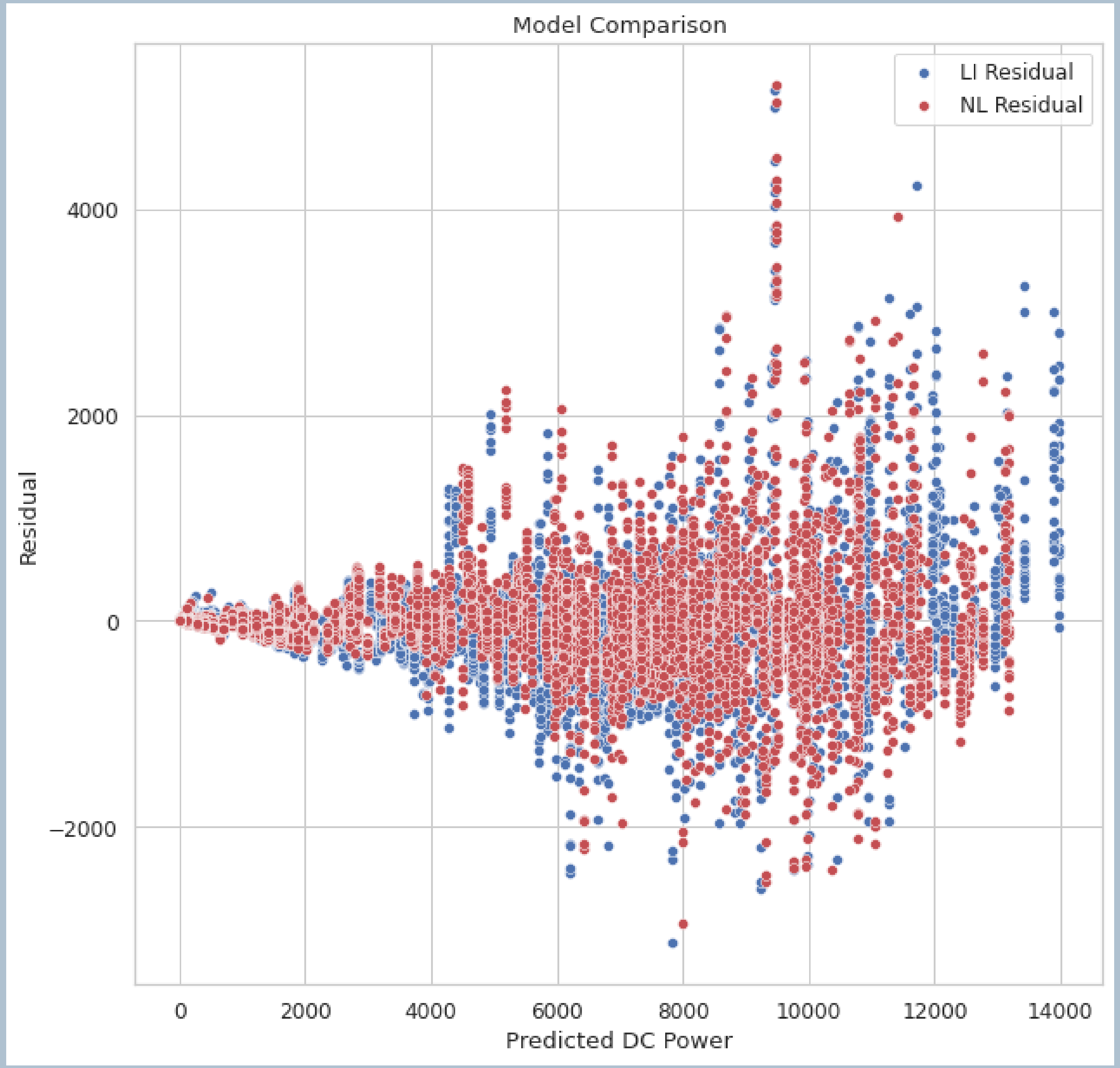
```
Mean absolute error = 16.01
Mean squared error = 2183.27
Median absolute error = 0.47
Explain variance score = 0.99
R2 score = 98.59 %
```

RandomForestRegressor

```
Mean absolute error = 15.99
Mean squared error = 2177.17
Median absolute error = 0.47
Explain variance score = 0.99
R2 score = 98.59 %
```

DecisionTreeRegressor

# Machine Learning Models




To compare the two models we can take a look at their respective residuals. The nonlinear model seems to perform slightly better than the linear model, especially at times of high irradiance.

# Future Scope

1.

In solar panels, equipment-grounding conductors are one of the crucial elements that would start as grounding wires on the array and continue with the power wires through the rest of the system. WEEB is one of the alternatives for conventional grounding. However, in the washer–electrical equipment bond, diagnosing the grounding fault is highly challenging.

2.

Since the instruments have a slow sample rate, it might not be able to capture the transient state of current if one wishes to monitor fault current readings and transient state readings. The sampling rate required to catch the fault current occurrence is much higher than the 30 s that the IoT can update the data at. An oscilloscope for digital storage can be used to capture the fault event in real time.

3.

The IoT can be used to update the status of the panel and data rate less than the 30 s sampling rate. If a signal varies much faster than this rate, it may not be updated in the cloud since it depends on the network's speed and web updating speed. A single-shot external signal out is required for an analog oscilloscope, as well as a particular camera to record the image.

# END OF PRESENTATION

Jainil Patel
21070126039

Harsh Gandhi
21070126031