Optimizing Solar Panel Plant Efficiency: Energy Forecasting, Performance Classification and Anomaly Detection

Christian Braga - Loris Salsi

July 2024

Contents

1 Introduction

1.1 Description of the Problem

The aim of this project was to exploit data from two solar panel plants to address three crucial tasks: predicting the energy production of individual panels, classifying their performance and developing an anomaly detection tool to monitor the health and efficiency of the solar panels.

The global transition to renewable energy has gained unprecedented momentum, driven by the need to reduce carbon emissions and combat climate change. Solar energy, in particular, is one of the fastest-growing sources of renewable energy, with massive investments being made in solar farms and rooftop installations. By 2024, photovoltaic (PV) technology has advanced significantly, resulting in more efficient, cost-effective, and scalable solutions for energy production. These challenges underscore the importance of the tasks addressed in this project.

Accurate energy prediction at the panel level enables better grid integration, allowing energy companies to optimize the balance between supply and demand. With reliable predictions, operators can plan for fluctuations in solar energy generation due to weather conditions or seasonal changes, thus enhancing the stability of the energy supply.

Classifying the performance of individual panels is essential for identifying underperforming units. Panels can degrade over time, accumulate dust, or experience issues like shading, all of which reduce their efficiency. By classifying the panels based on their performance, plant operators can prioritize maintenance efforts, focus on repairing or replacing faulty panels, and ensure the overall health of the system. This proactive approach minimizes downtime and ensures that the plant operates at maximum efficiency, ultimately reducing operational costs.

The anomaly detection tool plays a critical role in maintaining the long-term viability of solar plants. Anomalies in the data could indicate potential issues such as electrical faults, inverter failures, or unexpected drops in panel output. Detecting these anomalies early helps prevent costly equipment failures and unplanned outages. It also contributes to the overall reliability of solar energy as a consistent power source.

Addressing these tasks not only improves the operational efficiency of photovoltaic plants but also aligns with the broader goals of the renewable energy sector. In a world striving for sustainability, every optimization in solar plant management contributes to reducing dependency on fossil fuels and mitigating environmental impacts.

1.2 Methodologies and Results

To address the problem, we first performed an Exploratory Data Analysis (EDA) to detect null values, outliers, and understand the behavior and contributions of each feature to our analysis. After gaining a deep understanding of the sensor data, aided by data visualization and statistical methods, we identified and handled several anomalies related to the data collection process of the sensors. Once the data was properly cleaned and prepared, we applied various statistical methods, which were compared and analyzed to select the most suitable model for each task.

For the task of *energy forecasting*, we evaluated several methods:

- Multiple Linear Regression
- Multiple Linear Regression with Forward Stepwise Feature Selection
- Lasso Regression
- Decision Tree Regressor

- Random Forest Regressor

To *classify the performance* of the solar panels, we compared the following models:

- Multinomial Logistic Regression
- Linear Discriminant Analysis (LDA)
- K-Nearest Neighbors (KNN)
- Random Forest Classifier

For the anomaly detection system, we proposed a supervised approach based on the forecasting algorithm that achieved the highest accuracy. This simple approach involves setting a threshold between the predicted energy output (based on inputs) and the actual energy produced by the panels. Values that deviate beyond this threshold are flagged as anomalies, allowing for improved management of the solar plant.
Our analysis revealed that the best model for energy forecasting was the Random Forest Regressor, with a mean accuracy of $R^2 = 96.5\%$ across both plants.
The best model for classifying the performance of the solar panels is: Random Forest Classifier. With an accuracy of 80%
As a result, the Anomaly Detection tool was built using the Random Forest Regressor.
Possible improvements include using time series algorithms to enhance energy forecasting or comparing our model's performance against neural networks.

1.3 Code Structure

In the GitHub repository, you will find the following files:

- Full_project.ipynb = The main notebook where the entire analysis is conducted.
-Anomaly_detection_tool.ipynb = Code for the anomaly detection tool.
-second_plant_analysis.ipynb = The same analysis as in the main notebook, but applied to the second plant's data.

DATA

-Plant_1 (2)_Generation_Data.csv = Dataset related to the energy production of the first (second) plant.
-Plant_1 (2)_Weather_Sensor_Data.csv = Data concerning the atmospheric and operational conditions of the first (second) plant.
-first_plant.csv = Cleaned and merged dataset for the first plant.
-second_plant.csv = Cleaned and merged dataset for the second plant.

2 The Data

2.1 Solar Panel Plant Operation

A solar power plant works by harnessing sunlight and converting it into usable electricity through the photovoltaic effect. This process begins when photons from the sun hit the surface of a photovoltaic cell, typically made of silicon. These photons transfer their energy to the electrons within the silicon, causing the electrons to become excited. As a result, the electrons break free from their atoms and start moving, creating an electric current. The current generated by the photovoltaic cells is direct current (DC).
However, most homes, businesses, and power grids use alternating current (AC), so the electricity produced by the solar panels must be converted. This is the role of the inverter, a key component in the solar power system. The inverter converts the direct current (DC) from the solar panels into alternating current (AC), which is suitable for distribution and use in electrical grids and devices.
In every solar power plant, there are two main components: the solar modules and the inverters. The modules, or panels, capture sunlight and convert it into DC electricity, while the inverters handle the conversion of that energy into AC electricity.
Several factors affect the performance of a solar power plant. One of the key factors is temperature, as photovoltaic cells tend to be less efficient at higher temperatures. Another factor is dirtiness; if the panels become covered in dust or debris, their ability to capture sunlight and generate electricity is reduced. The efficiency of the inverters also plays a critical role, as energy is lost during the conversion process if the inverter is not functioning at optimal levels. Finally, the age of the panels and inverters can impact performance, as older equipment typically operates less efficiently over time. Provided this introduction to how solar panel plants work, let us proceed with a description of the dataset.

2.2 Dataset Description

The data used represent the energy produced by two solar power plants in India over a period of 34 days. for each plant we have a pair of files, the first one called: power generation, contains various performance information collected at the inverter level, each inverter has multiple lines of solar panels attached to it. while the second file: sensor data, is collected at a plant level and contains information about the operating conditions of the plant (atmospheric/environmental parameters).

Brief Description of the Available Variables

- Generation Data:

- 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 = stands for the inverter id.
- DC_POWER = Amount of DC power generated by the solar panel and sent to the inverter (source_key) in this 15 minute interval. Units - kW.
- AC_POWER = Amount of AC power converted by the inverter from the DC_power produced by the panel in the 15 minute interval. Units - kW.
- DAILY_YIELD = Daily yield is a cumulative sum of power generated (AC_power) on that day from the specific inverter, till that point in time.
- TOTAL_YIELD = This is the total yield for the inverter till that point in time (from the moment in which it has been installed).

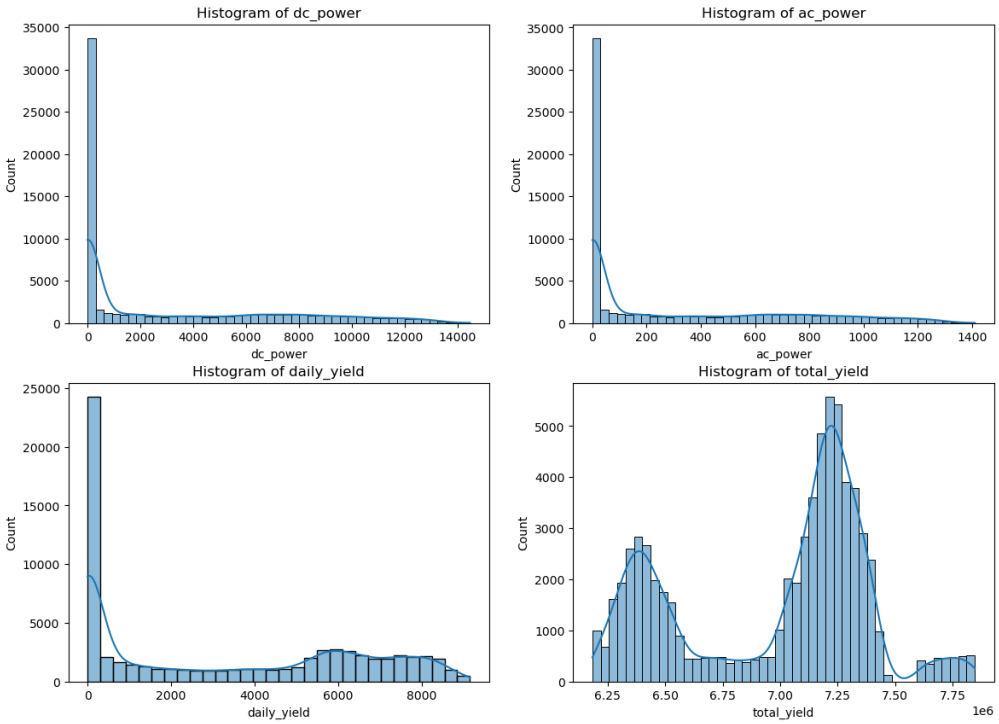
- Weather Sensor Data:

- 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 = Stands for the sensor panel id. This will be common for the entire file because there's only one sensor panel for the entire plant
- AMBIENT_TEMPERATURE = This is the ambient temperature at the plant.
- MODULE_TEMPERATURE = There's a module (solar panel) attached to the sensor panel. This is the temperature reading for that module.
- IRRADIATION = Amount of irradiation for the 15 minute interval.

2.3 Exploratory Data Analysis (EDA)

To gain a comprehensive understanding of the data available and extract valuable insights for our analysis, we began by performing an Exploratory Data Analysis (EDA). We started with the Generation Dataset from Plant 1. First, we checked the size of the dataset, which was (68,778, 7), and verified the presence of null values, of which there were none. Next, we examined each feature to understand its characteristics. For example, regarding the "date_time" column, we confirmed that the dataset contains values every 15 minutes, from May 15, 2020, 00:00, to June 17, 2020, 23:45. To facilitate working with this feature, we converted it into a proper date-time format. We then reviewed the other features, such as "source_key," to identify all the inverters in the plant, discovering that there were 22 in total.
Next, we conducted the same preliminary assessments on the sensor data file for Plant 1 and found that there were no null values in this case either. To better understand our data and identify potential anomalies, we proceeded to perform data exploration / visualization.

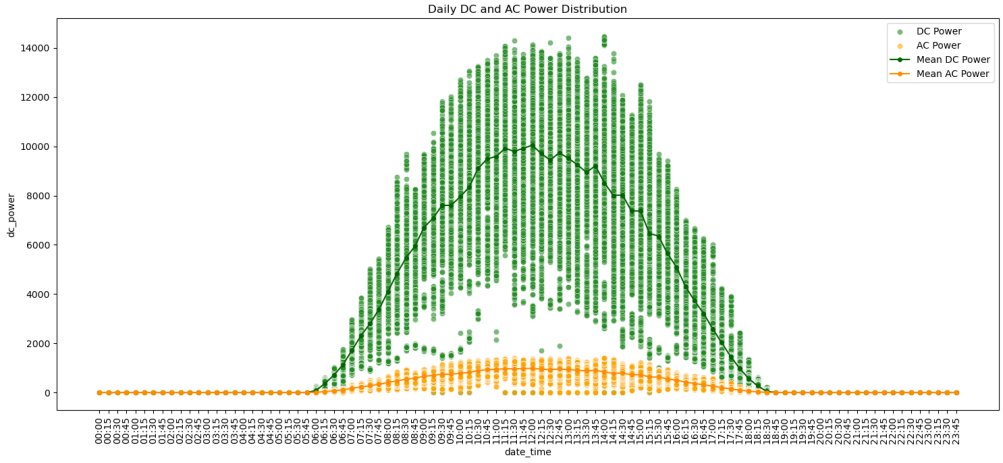
First of all we plotted the distribution of all the features with histograms:



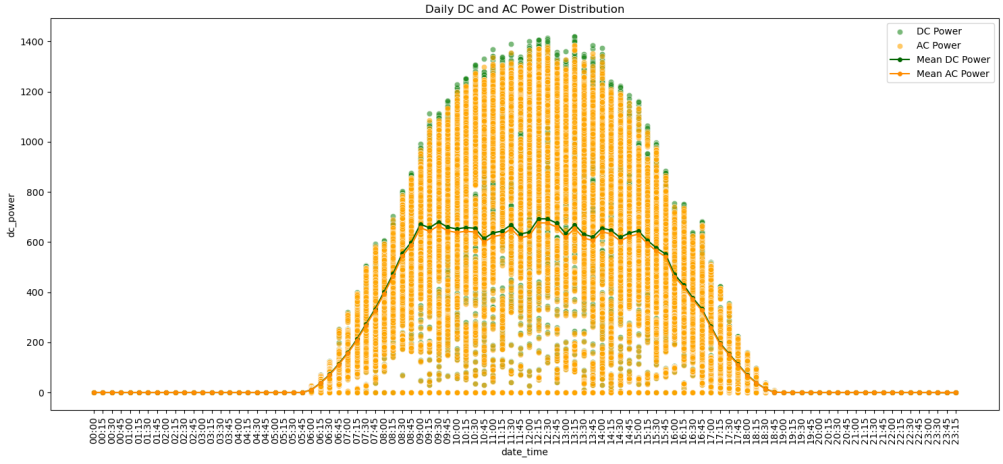
The first thing we noticed is that the distributions for AC_power and DC_power contain numerous values of 0. However, this is not a concern, as it is due to the fact that during nighttime, the panels are unable to

produce energy, making it normal to see many 0 values in electricity production. In our subsequent analyses, we will investigate whether these values are always accurate or if they could sometimes indicate a malfunction. Indeed, some of these 0.0 output values could suggest that the inverters which converts the energy are malfunctioning. Additionally, from the distribution of the daily yield, we observed the same peak around 0, which can be attributed to the same phenomenon. Since daily yield is the cumulative sum of the AC_power produced, if production starts only when the sun rises, from midnight until that point, the daily yield will mirror the AC_power values, resulting in 0. Regarding the distribution of total_yield, it is less relevant for our analysis, as it represents the energy converted by the inverter since its installation, which falls outside the time period of our study.

Then we create a box plot to identify the presence of outliers, but we noticed that they were not. Then we investigate the distribution of AC_power and DC_power produced daily:

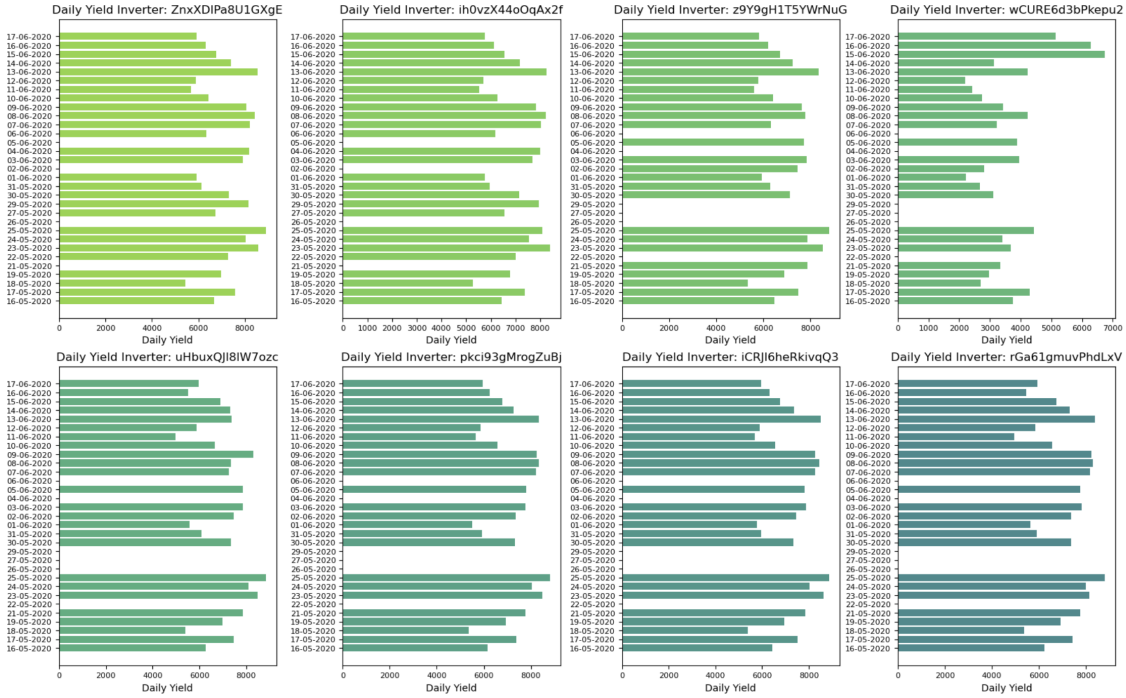


This graph helped us better understand the behavior of the solar plant. Recall that DC_power represents the amount of solar energy collected by the panels and converted to direct current, while AC_power represents the direct current that has been converted to alternating current by the inverter, allowing it to be used and transported. As shown by the daily distribution, current is produced from approximately 5:45 a.m. to 6:45 p.m., during the hours when the panels are exposed to sunlight.
Upon reviewing this distribution, we noticed an unusual relationship, particularly because the graph suggests that the conversion from DC_power to AC_power performed by the inverter is extremely inefficient. We found that the efficiency (AC/DC) of our inverters in this conversion was only about 9%, which is highly unusual, as many studies indicate that the typical efficiency for an inverter in such systems is around 97%. To verify this, we checked the efficiency in a second plant and found that the expected distribution and efficiency were correct.

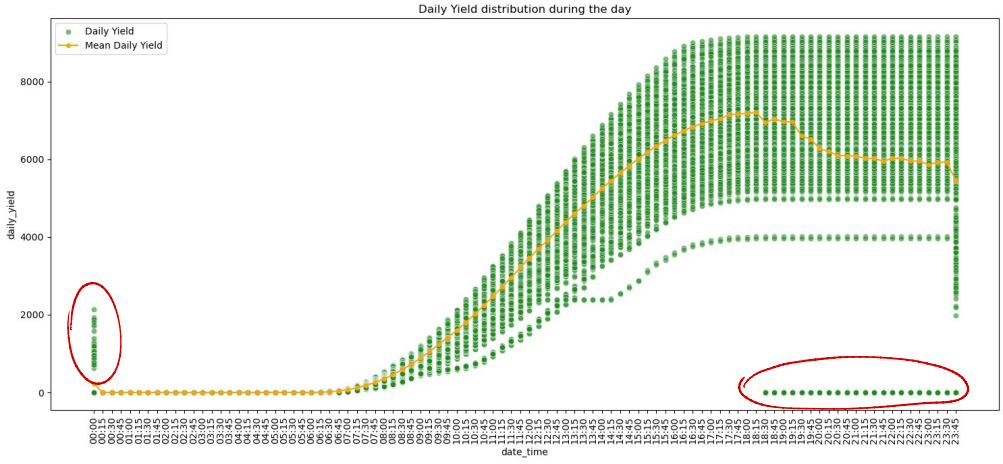


After carefully analyzing the data from Plant 1 and comparing it with that of the second plant, we concluded that the issue was not due to poor inverter efficiency but rather a data recording error. We observed that the scale of the values had been shifted by a factor of ten, causing the efficiency to appear as 9% instead of 97% for all inverters. The error became evident when we compared each value with the second plant, noticing unjustified differences in the value scales. We have since corrected this anomaly.

We then focused on the overall performance of the first plant. Specifically, we investigated the daily yield produced by each inverter during the last time interval (11:45 p.m.). Since the daily yield is the cumulative sum of the AC_power, we were particularly interested in the total amount of alternating current generated by the end of each day.

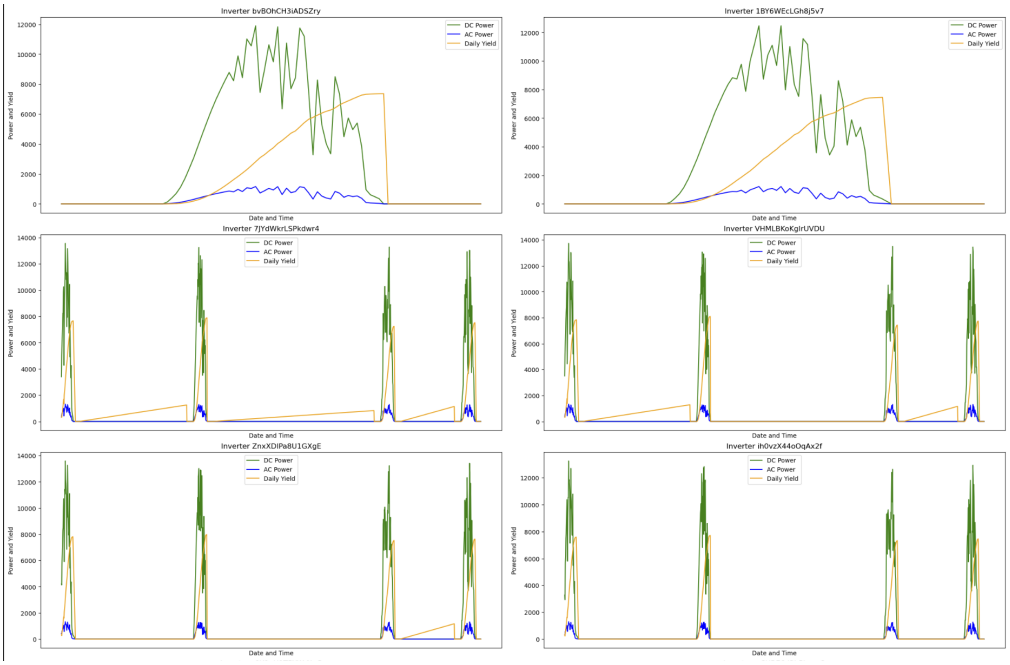


In this section of the complete graph (which can be fully viewed in the project's Python file), we noticed something interesting. Specifically, there are certain days when some inverters do not produce any energy. Additionally, the performance of the inverters varies, with some having more non-production days than others. To investigate the reasons for this and any potential suboptimal performance, we closely examined the distribution of the daily yield. To gain more insight, we plotted the daily yield against the hours of the day for each inverter on each day.



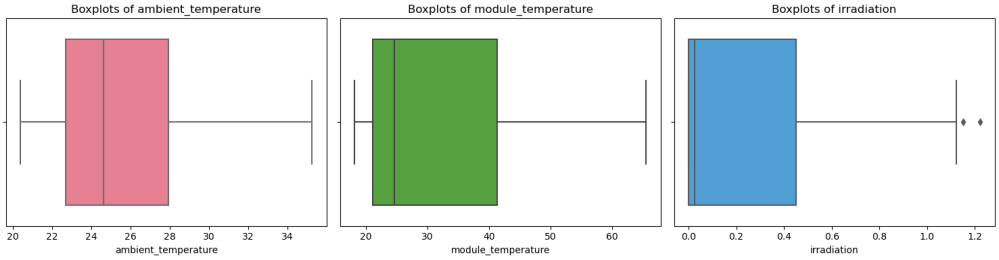
Thanks to this plot, we were able to identify two types of anomalies. First, for some inverters, there is a daily yield measurement recorded at 12:00 a.m., which is an error since energy production, as seen earlier, only begins around 5:30 a.m. We will correct this anomaly by setting these daily yield values to 0.00.
We also identified another issue: the daily yield values that drop to 0 after 6:30 p.m. are anomalies. Due to this error, some days appear as if there was no production, when in fact energy was produced until 6:30 p.m., and only stopped after that. This does not mean the daily yield for those days should be 0.00. We will adjust these values to reflect the last available yield recorded for each specific day. This adjustment will not affect the distribution of energy produced on those days.

An additional check to this anomalies are done by plotting the AC, DC power and daily yield of each inverters in the days that appear with a 0.00 energy production at 23.45 pm. In the following graph, the presence of such peaks, confirm our suspiciuos and shows indeed that some days were not improductive but some energy were created (if the measurements were correct these graphs should be flat and show no peaks). Instead as we can see these peaks are due to the fact that at some point in the day a measurement error brings the daily yield to zero representing that day as non-productive. We will then go on to correct these daily yield values with the latest measurement available for that day.



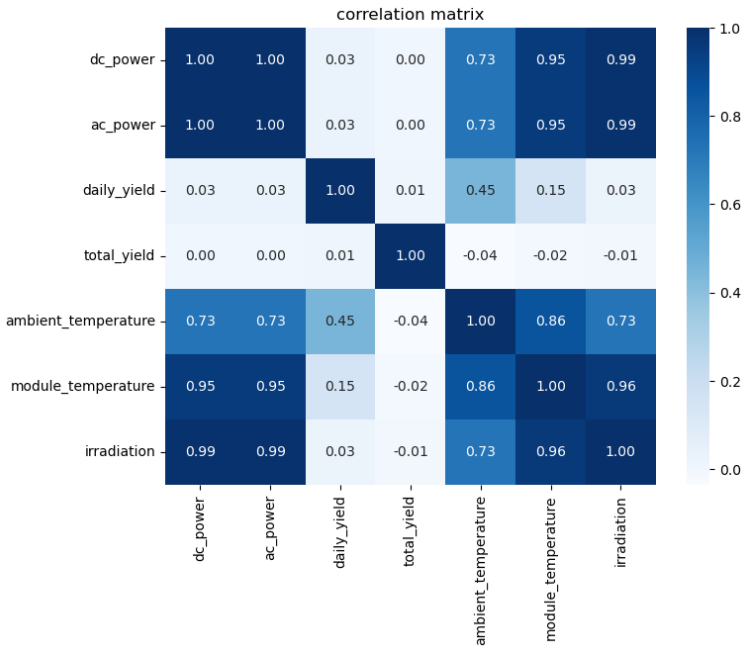
After correcting these anomalies, the previous graph no longer showed any peaks, indicating that we successfully addressed the errors.

After we analyzed the sensor dataset of Plant 1 and performed the same exploratory data analysis, this dataset contains important information related to the solar panel's operational environmental conditions. Upon closer examination, we identified the presence of outliers in the irradiation feature.



We detected them by the creation of the box plot and using the Interquartile Range (IQR) methodology, a widely adopted statistical technique to detect anomalies in a dataset. The process begins by calculating the first quartile (Q1), which represents the 25th percentile, meaning that 25% of the data points lie below this value. Similarly, the third quartile (Q3) is determined, corresponding to the 75th percentile, where 75% of the data points fall below this value. The IQR is then computed as the difference between Q3 and Q1, capturing the spread of the central 50% of the data. Once the IQR is established, lower and upper bounds are defined to identify potential outliers. The lower bound is calculated by subtracting 1.5 times the IQR from Q1, while the upper bound is obtained by adding 1.5 times the IQR to Q3. Any data point that falls below the lower bound or above the upper bound is considered an outlier. In this case, after identifying the outliers, it was noted that the number of anomalous data points was very small so the chosen approach was to remove them from the dataset instead of replacing them with other values.

Finally before to apply our statistical models we merge the two datasets related to the first plant and creating a new dataframe composed by 9 features, on that we will develop our solutions to the problem pointed out in order to ensure a good managment of the plant. The following is the correlation matrix of the merged dataset:



As we can see, some variables are strongly correlated, such as DC_power and AC_power, since the latter represents a conversion with 97% efficiency of the former. Additionally, ambient temperature and module temperature are, unsurprisingly, highly correlated. On the other hand, the total_yield feature is weakly correlated with all the others, which can be attributed to the nature of the data. These relationships between variables will be important to consider when applying the following models to avoid issues like multicollinearity.

3 Statistical Models

3.1 Objectives

To recall the main objectives of this analysis, we aim to optimize the management of the solar panel plant. To achieve this, we intend to:

- predict the energy production of the plant to facilitate better grid integration
- classify the performance of the panels in order to identify underperforming units
- develop an anomaly detection tool that helps prevent costly equipment failures and unplanned outage

3.2 Feature Engineering

In order to apply the following prediction models we needed to perform some feature engeenering. Fist of all we handled the categorical variable source_key, in particular we encoded this variable which represent the id's of the different inverters. To do that we create some dummy varibales, this technique involves converting a categorical variable into multiple binary numeric columns, where each category is represented by a new column with values of 0 or 1. This allows machine learning models, which require numeric data, to incorporate categorical variables into their calculations.

Then we needed to handle the multicollinearity problem. Multicollinearity occurs when two or more predictor variables in a dataset are highly correlated, meaning they provide redundant information. This makes it difficult for a machine learning model or regression to accurately estimate the relationship between each predictor and the target variable, leading to unstable or unreliable coefficient estimates, reduced interpretability, and potentially overfitting. In severe cases, it can affect the model's ability to make accurate predictions. Thanks to the correlation matrix that we have seen before some features appear to be very strongly correlated with each others: we will drop the variable 'DC_power' which has a 0.99 positive correlation with 'AC_power'. While for the variables 'module_temperature' and 'ambient_temperature' despite the high correlation since they provide different information that could be important to our model we try to keep them. In particular because the first represent the temperature of the solar panel which of course will be strongly affected by the ambient temperature, but we should consider that there could be some anomalies, for example if the panel goes in overheating, for that reason could be important to keep both this variables which provide essentially different information.

Finally to conclude this process of feature engenieering, we managed the value 'date_time' in a way in which the models could read that information. To allow that we converts the values from datetime objects to Unix timestamps (number of seconds elapsed since January 1, 1970). This step is necessary in order to allow our machine learning models to manage data in a date time format.

Setting the Variables

In developing the following statistical models, our dependent variable (y) will be: 'daily_yield' so the amount of energy that has been produced from the solar panel and converted by the inverter in the specific time interval. While the independet varibles will be all the others feature previously analyzed (comprensive of all the inverters encoded)

Model Evaluation

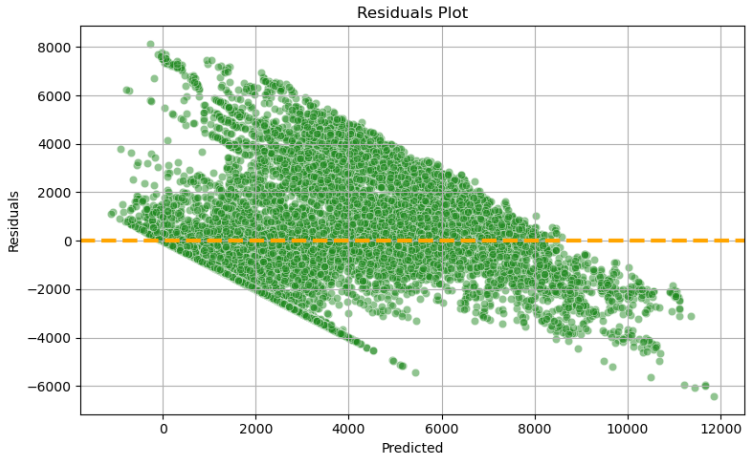
In order to select the best model for each task we splitted the entire dataset using the train_test_split method of the library sklearn. In specific we used the 80% of the dataset as training set and the remaining 20% as test set, on which we evaluated the performance of the statistical learning models. The selected evaluation metric was the R^2 score or coefficient of determination. It measures how well a regression model fits the data. It represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s). In simpler terms, R^2 tells us how well the model's predictions align with the actual observed

data. A score of 1 means the model perfectly explains the variation in the data, while a score of 0 indicates that the model does not explain any of the variability, and the predictions are no better than simply using the mean value of the dependent variable.

3.3 Energy Production Forecast

3.3.1 Multiple Linear Regression

The first model we attempted to build was Multiple Linear Regression, which yielded an R^2 score of 0.4628, indicating poor performance. To gain a better understanding of this model, we plotted the residuals (the differences between observed and predicted values) against the predicted values, and we noticed an important insight:



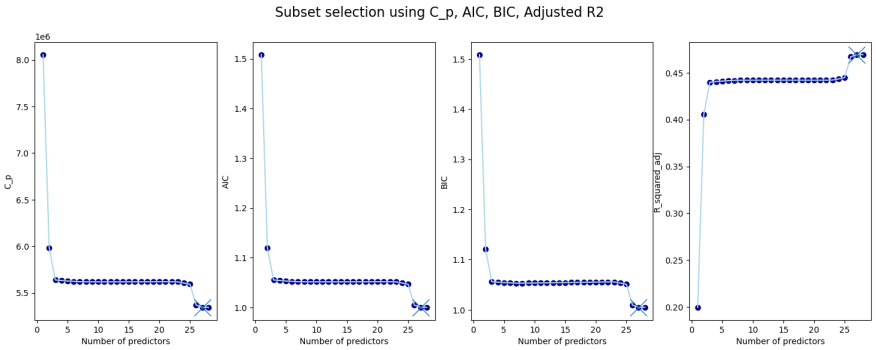
As can be seen from this plot, the residuals exhibit a funnel shape, indicating a potential issue of heteroscedasticity. This dispersion suggests that the variance of the residuals is not constant across the range of predicted values, with the variance decreasing as the predicted values increase. Such a condition may affect the reliability of statistical inferences drawn from the regression model.
In ordinary least squares (OLS) regression, the assumption of constant variance (homoscedasticity) is essential for the model to provide unbiased, efficient, and reliable estimates of the coefficients. When the variance of the errors is not constant (heteroscedasticity), several issues can arise, including inefficiency of the OLS estimators, misleading statistical significance, and overstated or understated predictive accuracy.

In order to address this problem we tried different solutions: first of all we take the logarithm of the y variable, this is a common solution that can help stabilize the variance of the residuals and make the data more normally distributed, which improves the performance and reliability of the regression model. But the R^2 score of this new model was 0.4181, which is worse.

3.3.2 Forward Stepwise Selection

To address the issues identified in the model, we implemented an alternative fitting procedure by applying Forward Stepwise Selection. This method is a subset selection technique designed to identify the most significant predictors, focusing on those that contribute the most to predicting the dependent variable. The core concept behind Forward Stepwise Selection is to start with no predictors and gradually add variables that yield the greatest improvement in the model's performance at each step. This approach offers several advantages over standard multiple linear regression. First, it helps produce more robust models by selecting variables based on their ability to enhance performance, thereby reducing the risk of overfitting and potentially improving the handling of residuals. After developing this model, we needed to compare the best model for each subset of predictors with a specific number of features. To accomplish this, we selected several

parameters that allow us to compare models with different numbers of predictors: AIC, BIC, Mallows' CP, and adjusted R^2.



After having found the best subset that better explain our dependent varibale y we found that the R2 of this model was 0.4695, so we had a little improvement with respect to the multiple linear regression.

3.3.3 Lasso Regression

Since we have noticed that a more robust model has perfomed better we followed this path by developing a even less flexible linear model: The Lasso Regression. Lasso regression (Least Absolute Shrinkage and Selection Operator) is a type of linear regression that applies L1 regularization to the model. This technique adds a penalty equal to the absolute value of the magnitude of the coefficients to the loss function, which forces some of the coefficients to be exactly zero. As a result, Lasso helps in feature selection by shrinking less important feature weights to zero, making it useful for models with many features. This model had a R^2 score of 0.4375 so we decreased our performances. This help us to understand the the path of the linear model was not right and that maybe the relationship between our x and y is not linear.

3.3.4 Decision Tree Regression

Since we observed that linear models were not performing optimally, we decided to try a nonlinear model that does not assume constant error variance. We began with Decision Tree Regression, setting the maximum depth of the tree to 5, which, in our tests, provided the best performance while avoiding overfitting. Then we evaluate it on the test set and got $R^2 = 0.6381$ which is better than all the linear models tested before.

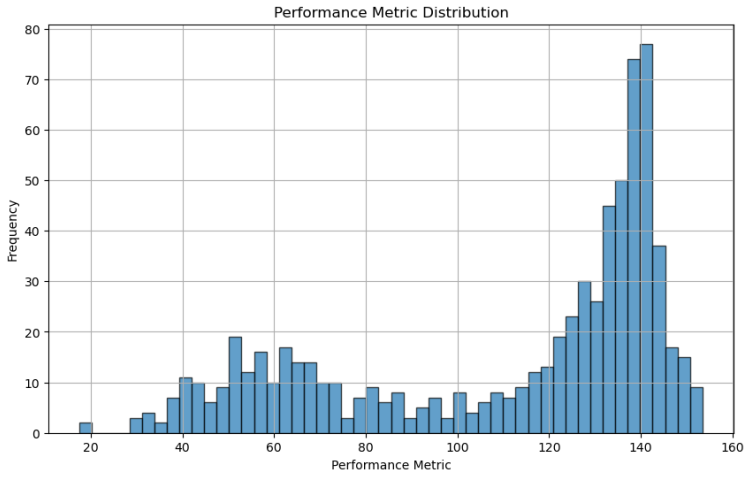
3.3.5 Random Forest Regression

After implementing the Decision Tree, since it is well known that, despite being an interpretable model, it can suffer from certain limitations such as overfitting and high variance, as a small variation in the data can drastically change the tree structure. To address this issue, we decided to use Random Forest, an ensemble of decision trees that builds multiple trees on different random subsets of the dataset and features. This method reduces overfitting by combining the predictions of multiple trees, resulting in a more robust model that is less sensitive to variations in the data, thereby improving overall performance. Moreover, Random Forest can capture complex and nonlinear relationships between the predictor variables and the response variable so it can better fit complex models and nonlinear data. Indeed, this model performed exceptionally well, surpassing the performance of the Decision Tree and demonstrating that nonlinear models were a better choice for this type of problem. The overall performance on the test set for Random Forest was remarkably high, with an R^2 score of 0.9971. To further validate its effectiveness, we applied the model to data from a second plant, where it again performed well, achieving an R^2 of 0.9472 on the test set.

3.4 Solar Panel Performance Classification

In order to classify the performance of the panels and identify underperforming units we built different classification models. But before that we performed additional features engeneering to take in consideration the

structure and the nature of our data. First of all we created a new feature called: performance_metric which has been obtained by dividing the total DC power by the total irradiation . This will create a feature that is able to measure the efficency of the solar panels in our plant. Irradiation is used because it is a highly influential variable on the DC power production of the panels, as seen from its high correlation and the coefficients obtained in the regression analysis. So this metric will be our target variable.
Then The data has been aggregated at a daily level to avoid nighttime irradiation values (which are always 0), which would otherwise cause the dc_power/irradiation ratio to be zero. Classifying the solar panel performance using these zero values would not make sense. Aggregating the data allows for a better assessment of the daily performance of the panel line. Then we visualized the distribution of this new metric in order to understand how to split it into classes.



Based on the observed distribution of the data, we decided to use quantiles to categorize performance into distinct classes. Specifically, we used the 33rd percentile to define the threshold for the first class, which we labeled as 'low performance.' This means that approximately 33% of the data falls into this category. To further divide the remaining data, we used the 67th percentile as the cutoff point between the second and third classes. The observations between the 33rd and 67th percentiles were categorized as 'medium performance,' while those above the 67th percentile were classified as 'high performance.' This quantile-based approach allows us to establish meaningful thresholds for classification by determining clear cutoff points that separate the different performance categories. As a result, we obtained 236 observations in the low-performance category, 244 in the medium-performance category, and 236 in the high-performance category, indicating a well-balanced dataset.

3.4.1 Multinomial Logistic Regression

The first classifier we built was a Multinomial Logistic Regression model, which is designed for multi-class classification problems where the dependent variable has three or more categories. This method estimates the probability of each class by employing multiple logistic functions. We trained this model on the training set and evaluated its performance on the test set using a confusion matrix. A confusion matrix is a tool used to evaluate the performance of a classification model. It provides a summary of the prediction results by displaying the counts of true positive, true negative, false positive, and false negative predictions. In a multi-class setting, the matrix shows how many instances of each class were correctly classified versus misclassified. This allows for a more understanding of the model's accuracy, revealing not only overall performance but also specific areas where the model may be struggling, such as particular classes that are often confused with one another. The confusion matrix for this algorithms shows: an overall precision of 42% which is poor. Indeed rom the data, it appears that the model has great difficulty in predicting the "low performance" class, with 0 correct predictions (precision and recall both 0.00). The "high performance" class performs better with an accuracy of 51% and recall of 68%, showing that the model can identify this class better. The "medium" class has a lower accuracy (34%), but a recall of 57%.

3.4.2 Linear Discriminant Analysis (LDA)

This is a classification method used to find a linear combination of features that best separates two or more classes of objects or events. It is particularly useful for classification problems with more than two classes and works by modeling the differences between the classes. the performance obtained were The confusion matrix associated to this algorithm show a general improvement over the previous model. The "high" class is predicted with 61% accuracy and 82% recall, resulting in an F1-score of 0.70, indicating a good ability of the model to identify this class. The "low" class also performs well with 63% accuracy and 62% recall, which is a marked improvement over the first matrix. The "medium" class, on the other hand, performs lower with an accuracy of 51 percent and recall of 34 percent, indicating that the model continues to confound more on this class. The overall accuracy of the model increased to 60%, showing a substantial improvement.

3.4.3 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a non-parametric classification algorithm used to assign a class to a sample based on the majority class of its k nearest neighbors in the feature space. It is effective for problems where the decision boundary is not linear and can be easily implemented and understood. Choosing the optimal number of neighbors (k) is crucial for the performance of the K-Nearest Neighbors algorithm. So we evaluate the model for different values of k to identify the one that minimizes the error on the test set. After identifying the value of k for which the test error is reduced i.e., k = 11 the following confusion matrix shows: an overall accuracy of 48 percent, which represents a deterioration from the previous matrix (which had an accuracy of 60%). Therefore, we will try another method to increase the previous result.

3.4.4 Random Forest Classifier

Finally, for the classification problem, we utilized an ensemble method known as Random Forest, which is highly effective for classification tasks. This method constructs multiple decision trees and combines their predictions to achieve a more accurate and stable outcome. Each individual tree in the Random Forest provides a class prediction, and the class with the most votes becomes the model's final prediction. This approach yielded the best performance among the models tested, making it the classifier of choice for identifying the performance of our panels in future analyses. In this confusion matrix, the Random Forest Classifier achieved an overall accuracy of 80%, which is significantly better than the previous models such as multinomial logistic regression, LDA, and KNN. The model performed exceptionally well for the "high" class, with a precision of 90%, recall of 91%, and an F1-score of 0.90. For the "low" class, the model achieved a precision of 80%, recall of 74%, and an F1-score of 0.77, indicating strong performance in this category as well. The "medium" class, which was problematic for other models, showed improved results with a precision of 69%, recall of 73%, and an F1-score of 0.71. Random Forest performed better than multinomial logistic regression, LDA, and KNN because it handles non-linear interactions more effectively; while logistic regression and LDA are based on linear relationships between variables, Random Forest is a non-linear model that captures complex variable interactions, leading to improved performance, especially on complex datasets. Additionally, Random Forest is more resistant to overfitting due to its ensemble nature, which reduces the risk of overfitting compared to simpler models like KNN or single decision trees. It also excels at handling mixed data types without relying on strong distributional assumptions, unlike LDA, which assumes Gaussian distributions for each class.

3.5 Anomaly Detection Tool

Lastly, we aimed to develop an anomaly detection tool to prevent costly equipment failures and unplanned outages. To achieve this, we adopted a supervised approach. Given the exceptionally high prediction accuracy of the Random Forest Regressor model—99% for the first plant and 94% for the second we decided to utilize these predictions for anomaly detection.
The method we employed is straightforward and follows a threshold approach. After creating and training the Random Forest Regression Model, we generated predictions for each observation in the test set. We established a threshold of 30%, meaning that if the predicted value deviated from the actual value by at least 30%, the observation would be classified as an anomaly.
The output of this tool is a DataFrame that displays all detected observations, enabling us to draw various insights. For instance, it allows us to easily identify which inverter is responsible for a specific anomaly,

the time it occurred, the expected energy output of the inverter, and the actual value produced. With this information, we can conduct a thorough investigation into the underlying causes of these issues.
Furthermore, by monitoring the frequency of anomalies detected over time, we can identify patterns that indicate recurring issues with specific equipment. For example, if a particular inverter consistently generates anomalies, it may signal underlying mechanical or operational problems that require attention. Based on this assessment, we can establish criteria for replacement.
Finally, this model could be implemented for real-time operation by being integrated with sensors that continuously provide data from the solar plant. This would enable timely detection of potential failures, significantly reducing repair costs, optimizing the plant's performance, and preventing the escalation of issues that could lead to serious damage in the solar panel plant.

4 Conclusion

In conclusion, this project focused on optimizing the management of two solar panel plants through predictive analytics, performance classification, and anomaly detection. We began by performing an extensive exploratory data analysis (EDA), which helped us clean the dataset and identify key features. We applied various statistical models to predict energy production, classify panel performance, and detect anomalies.
For energy production forecasting, we found that the Random Forest Regressor was the most effective model, achieving an impressive R^2 score of 99.7% for the first plant and 94.7% for the second. This high level of accuracy highlights the model's ability to handle non-linear relationships and complex datasets, making it an ideal choice for energy prediction tasks.
In classifying panel performance, the Random Forest Classifier also outperformed other models, achieving an overall accuracy of 80%. This model demonstrated its superiority over multinomial logistic regression, LDA, and KNN by efficiently capturing non-linear interactions and resisting overfitting, particularly when dealing with diverse data types.
Finally, we developed an anomaly detection tool using the predictions from the Random Forest Regressor. This tool allows plant operators to identify potential failures in real time, helping to prevent costly repairs and unplanned downtime. By flagging anomalies early, the tool ensures that underperforming inverters or panels can be addressed before they cause significant damage or loss of productivity.
Overall, this project demonstrates the value of advanced machine learning techniques in enhancing the operational efficiency of solar panel plants. By improving energy prediction, identifying underperforming units, and preventing failures through anomaly detection, this analysis contributes to the broader goal of making renewable energy systems more reliable, efficient, and cost-effective. Future improvements could focus on incorporating time-series forecasting models or neural networks to further enhance prediction accuracy and extend the capabilities of the anomaly detection tool.