



**Disclaimer: The content is curated from online/offline resources and used for educational purpose only**

**LAB MANUAL**

**Unit III – Machine Learning**

**Unit III – Machine Learning**

**Lab 2. Predicting Solar Power Output Using Linear Regression**

**Objective**

To implement solar power output prediction using Linear Regression, a general approach to how you can tackle the solar power output prediction problem using linear regression, along with a sample implementation using Python and common libraries like Pandas, Scikit-learn, and Matplotlib. The model will be trained on historical data and will predict the solar power output for given values of the input features. The performance of the model will be evaluated using standard metrics such as Mean Squared Error (MSE) and R-squared (R²).

**Problem**

The global shift towards sustainable energy has led to an increasing reliance on renewable energy sources such as solar power. Accurate predictions of solar power output are crucial for effective grid management, energy storage, and optimizing the use of solar energy. we aim to predict the solar power output based on certain environmental factors, which include temperature, humidity, solar\_irradiance and wind\_speed. The goal is to develop a predictive model using Linear Regression, a simple yet powerful algorithm, to estimate the solar power output from these features.

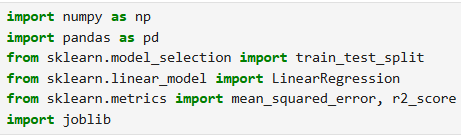
**Solution**

To predict solar power output using linear regression and save the model, we'll go through the following steps:

1. Import required libraries
2. Prepare the dataset
   1. Load dataset
   2. Select Features and target variable
3. Train a linear regression model
   1. Split the dataset into training and testing sets (80% train, 20% test)
   2. Initialize and train the linear regression model
4. Evaluate the model
   1. Predicting the solar power output on the test set
   2. Evaluate the model performance
5. Visualization
6. Save the model for later use
7. Load and use the Model

**Procedures**

**1. Import required libraries**



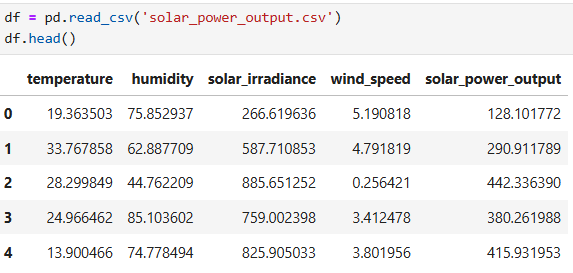
**2. Prepare the dataset**

**Features:**

* temperature: in the range of 10-35°C to represent different climates.
* humidity: In between 20% and 100% as humidity varies across regions.
* solar\_irradiance: In between 100 W/m² and 1000 W/m² (typical range for solar radiation).
* wind\_speed: In between 0 m/s and 10 m/s, representing wind conditions.

**Target Variable (solar\_power\_output)**: This is calculated as a function of the other features, with random noise added to make the data more realistic. The relationship assumes that higher irradiance and temperature increase power output, while high humidity reduces efficiency. Wind speed has a small positive impact.

**2.1 Load Dataset**



**2.2 Select Features and target variable**

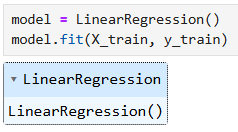
****

**3. Train a linear regression model**

Now, we’ll split the dataset into training and testing sets, train the linear regression model, and evaluate its performance.

**3.1 Split the dataset into training and testing sets (80% train, 20% test)**

** 3.2 Initialize and train the linear regression model**

****

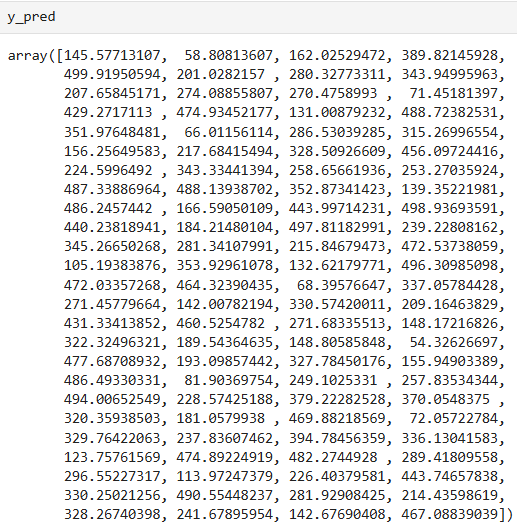
**4. Evaluate the model**

We will now predict the solar power output using the test data and evaluate the performance using metrics like Mean Squared Error (MSE) and R-squared (R²).

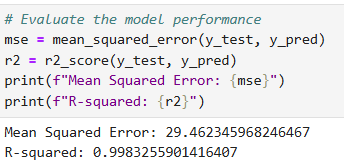
**4.1 Predicting the solar power output on the test set**

****

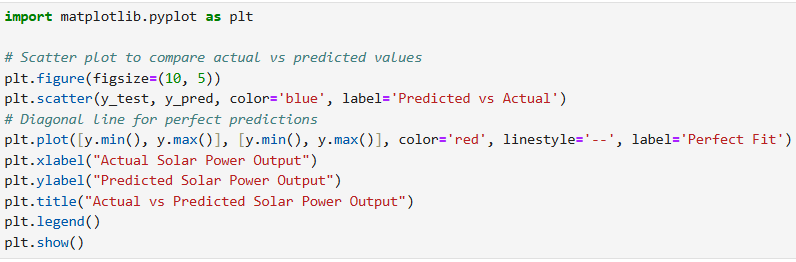
Predicted outputs of test dataset,

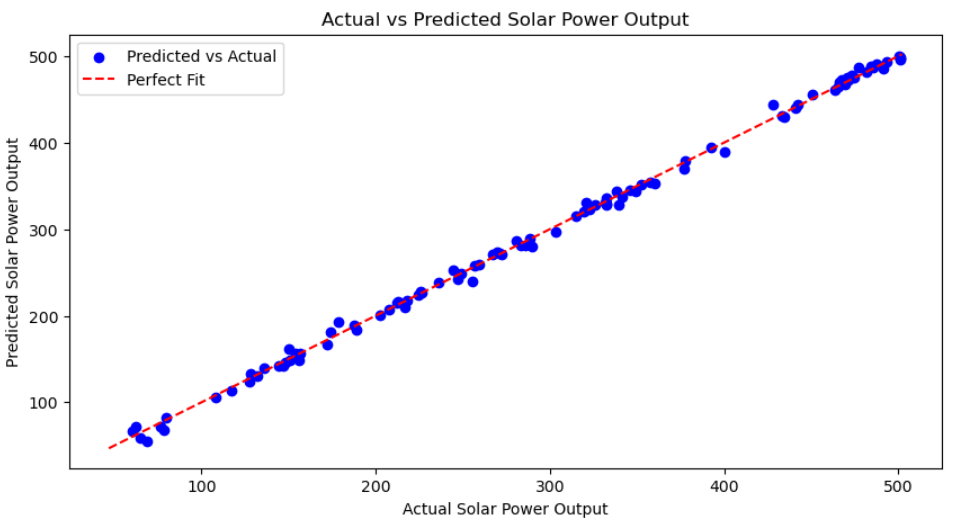
****

**4.2 Evaluate the model performance**

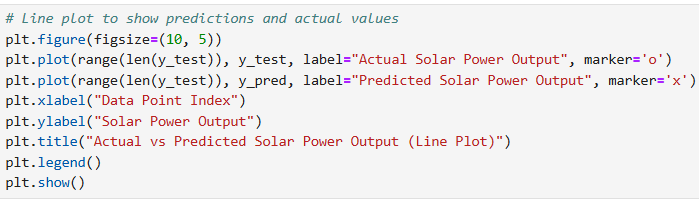
****

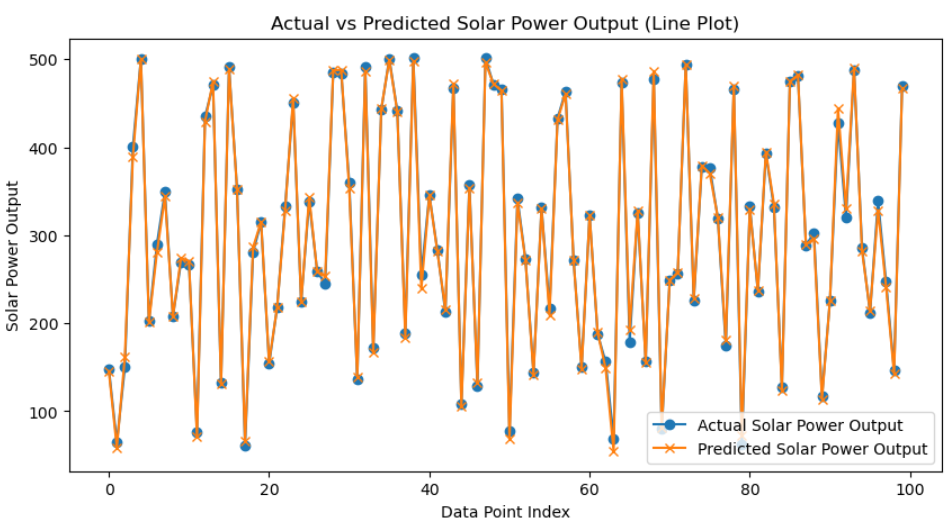
**5. Visualization**





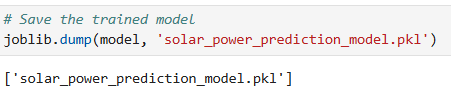
* Each blue dot represents a comparison between an actual and predicted value.
* The red dashed line represents a perfect fit line, where predicted values would exactly match actual values if the model were perfect.
* The closer the points are to this line, the better the model's accuracy.





* Plots both actual and predicted solar power outputs for each test data point index.
* Helps visualize how predictions follow (or deviate from) the actual trend across the dataset.

**6. Save the model for later use**



**7. Load and use the Model**

