Internship Report

in partial fulfillment for the award of the degree
of

Bachelor of Technology

in

Computer Science and Engineering

Submitted By

Stuti 22bcs093

Under the Guidance of
Dr. Dipankar Deb IIT Mandi

School of Computer Science & Engineering
Shri Mata Vaishno Devi University
Kakryal, Katra, J&K-182320

2025

Abstract

Accurate forecasting plays a crucial role in supporting efficient energy storage, grid stability, and backup planning. This report investigates the prediction of solar energy output using weather data to address the variability and intermittency of solar power generation. Various machine learning regression algorithms are implemented to forecast energy output based on environmental factors, including temperature, ground radiation intensity, and atmospheric radiation intensity. These models are evaluated by comparing their prediction errors to understand their effectiveness under different weather conditions. The aim is to contribute to more reliable and efficient management of solar energy systems through data-driven predictive modeling.

Stuti June, 2025

List of Figures

List of Tables

Contents

Chapter 1

1.1 Introduction

Solar energy is one of the most readily available and renewable sources of energy, offering a promising solution to the growing demand for sustainable power. Solar panels are widely used to harness this energy; however, the electricity they generate is not consistent throughout the day or year. Various atmospheric and environmental factors including cloud cover, wind speed, and precipitation, significantly influence the amount of energy produced.

Moreover, during certain periods, such as rainy or cloudy days and at night, solar panels generate little to no power. This variability presents a significant challenge for energy providers and users who rely on a consistent power supply. To maintain system stability, it is essential to predict how much power can be generated at a given time in the future. Such predictions help in two key ways: ensuring that sufficient backup power is available when generation is low, and enabling the storage of excess energy when production exceeds demand.

To address this challenge, this project investigates the use of machine learning regression techniques to predict future solar power generation based on historical weather and energy data. The primary aim is to evaluate and compare different machine learning models to determine their effectiveness in forecasting power output, thereby supporting more reliable and efficient solar energy systems

1.2 Motivation

The increasing global reliance on renewable energy underscores the need for accurate solar power forecasting. The primary objective of this project is to systematically evaluate various machine learning methods for predicting solar power generation, utilizing key environmental variables such as **solar irradiance, ground irradiance, and temperature**. Accurate predictions are essential for:

- **Grid Stability**: Mitigating supply-demand imbalances by anticipating fluctuations in solar generation.
- **Energy Storage Optimization**: Identifying surplus generation periods to store excess energy efficiently.

By identifying the most robust predictive models, this work aims to enhance the reliability and scalability of integrating solar energy into power systems. Ultimately, improved forecasting supports the transition to sustainable energy infrastructures while addressing intermittency challenges inherent to renewable sources.

Chapter 2

2.1 Workflow

In this project, a structured and systematic workflow is implemented to analyze and forecast solar power generation using time-series data. The dataset, obtained from a publicly available repository, comprises approximately 5,800 hourly records collected from a single solar power plant over the period of January to August 2019. It includes essential features such as timestamps, generated power, ambient temperature, and solar irradiance. During the preprocessing phase, timestamps are converted into multiple time-based features—such as hour of the day, day of the week, week number, and month—to better capture temporal patterns that influence power generation.

Following preprocessing, exploratory data analysis (EDA) is carried out using various visualization techniques like line plots, histograms, and heatmaps to identify daily, weekly, and seasonal trends in the data. Based on the insights gained, a range of machine learning models—including Linear Regression, Support Vector Regressor, and Random Forest—as well as deep learning models such as LSTM and GRU are applied to predict solar power output. These models are rigorously evaluated using standard regression metrics: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the R^2 score. Additionally, visual comparisons of actual versus predicted power values are generated to offer a more intuitive understanding of each model's performance. These combined evaluations help determine the most accurate and robust model for practical solar power forecasting applications.

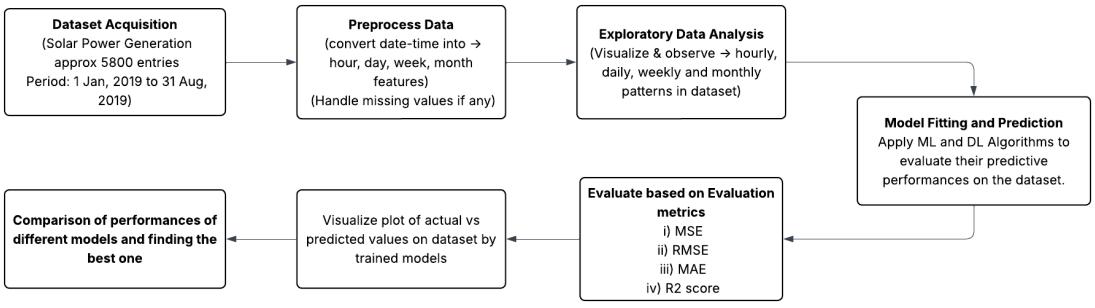

Figure 2.1: Work Flow

2.2 About Dataset

For this comparison-based study of solar power generation prediction, I have used a time-series dataset that has records of hourly measurements of environmental and power variables over a course of 8 months, starting from June 1st, 2019, till August 31st, 2019. The main features used are:

- Date_time
- temperature
- humidity
- atmosphere_radiation_intensity
- ground_radiation_intensity
- photovoltaic_power_generation (class attribute)

The whole data set has temperature in negative degrees Celsius, which indicates that the environment in which the measurements were made has a cold climate. Humidity values are very close to zero, which indicates a dry environment. Photovoltaic power generation is the actual power output from the photovoltaic system at each hour, and its unit is kilowatt-hours. It is the target variable for prediction.

2.3 Libraries Used

2.3.1 Scikit-learn

Scikit-learn is a widely used open-source Python library for machine learning that provides efficient tools for building and evaluating models. It supports a wide range of algorithms for classification, regression, and clustering, such as SVMs, random forests, k-means, and gradient boosting. I have used various Regression algorithms from this library to evaluate on the mentioned dataset, given below:

- Linear Regression
- Support Vector Regression
- Random Forest
- K Neighbors Regressor
- XGBoost(Extreme Gradient Boosting)
- LightGBM
- LSTM(Long Short-Term Memory)
- MLP(Multi Layer Perceptron)
- GRU (Gated Recurrent Unit)
- ANN (Artificial Neural Network)
- GAM (Generalized Additive Model)

2.3.2 GridSearchCV

GridSearchCV is a tool in scikit-learn that automates the process of hyperparameter tuning for machine learning models. It works by defining a grid of possible hyperparameter values and then systematically training and evaluating the model for each combination using cross-validation. Rather than relying on a manual trial-and-error method, it ensures a thorough search for the part of hyperparameter tuning.

2.4 Exploratory Data Analysis(EDA)

In this section, an in-depth exploration of the solar power generation dataset is conducted to extract meaningful insights and guide the subsequent modeling process. EDA helps in understanding the temporal behavior, relationships between variables, and overall structure of the data.

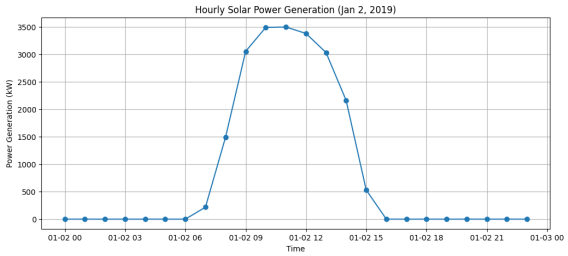

Figure 2.2: Hourly analysis of Jan 2, 2019

The **fig 2.2** presents hourly solar power generation on **January 2, 2019**, revealing a typical daily generation pattern. Power generation begins around 7 AM, peaks between 10 AM and 1 PM (reaching approximately 3,500 kW), and then declines to zero by around 6 PM. This curve reflects natural sunlight availability and highlights the predictable daily solar generation cycle, which is crucial for time-series modeling.

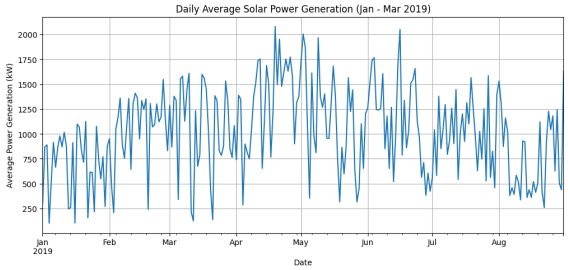

Figure 2.3: Daily analysis of Dataset

The **fig 2.3** shows the daily average solar power generation from January to August 2019. A gradual increase in average power generation is observed from January to May, followed by fluctuations in June and a notable decline in July and August. This suggests a seasonal trend, likely influenced by changes in weather conditions such as cloud cover or monsoon patterns.

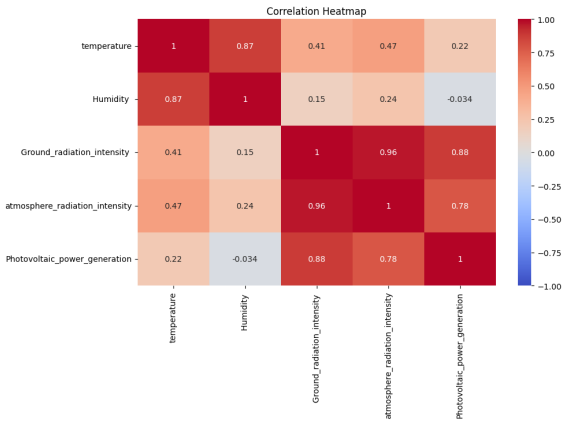

Figure 2.4: Correlation Heatmap

The **correlation heatmap** further enhances our understanding of inter-feature relationships. It is observed that:

- **Ground radiation intensity** and **atmospheric radiation intensity** have strong positive correlations with **photovoltaic power generation** (0.88 and 0.78, respectively).
- **Temperature** also shows a mild positive correlation (0.22), whereas **humidity** has a slightly negative correlation (-0.034), indicating minimal impact.

These insights indicate that irradiance-related features play a dominant role in determining power output, whereas temperature and humidity have less pronounced effects.

2.5 Algorithms Implemented

2.5.1 Linear Regression

Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables. It predicts the value of a dependent variable based on the independent variable(s) by fitting a straight line to the data. The line is determined using the method of least squares, minimizing the sum of squared differences between observed and predicted values. The regression equation is typically written as $y=a+bx$, where a is the intercept and b is the slope. Linear regression assumes a linear relationship, constant variance of errors, normality of residuals, no multicollinearity among predictors, and no autocorrelation of errors.

2.5.2 SVR (Support Vector Regression)

Support Vector Regression (SVR) is the regression counterpart of SVM(Support Vector Machine), which is designed to predict continuous values rather than classes. Instead of fitting a line through the data, SVR fits a tube (called the epsilon-insensitive tube) around the data points, aiming to keep as many points as possible within this tube. Only points outside the tube, known as support vectors, influence the position and orientation of the tube. The goal is to minimize the error for points outside the tube while ignoring errors within the tube. SVR can handle both linear and nonlinear relationships using kernel functions.

2.5.3 K Neighbors Regressor

KNeighborsRegressor is a non-parametric regression algorithm based on the k-nearest neighbors method. For a given input, it finds the k closest data points in the training set (neighbors) and predicts the output by finding the mean of the target values of these neighbors. The choice of k and the weighting scheme can significantly affect performance. KNeighborsRegressor is easy to understand and implement, works well for small to medium-sized datasets, and does not require explicit model training. It is sensitive to the choice of distance metric and feature scaling.

2.5.4 Random Forest

Random Forest is a robust and adaptable supervised machine learning algorithm that creates and integrates several decision trees to form a "forest". Each decision tree is trained using a random subset of both the data and its features, reducing correlation among the trees and minimizing the likelihood of overfitting. The prediction is derived from the average results generated by all the trees. Random Forests are effective at managing noise and outliers, capable of handling large datasets efficiently, and adept at capturing complex patterns. Although they offer increased accuracy and reliability, their interpretability is lower compared to individual decision trees.

2.5.5 XGBoost

XGBoost, or Extreme Gradient Boosting, is a distributed and scalable machine learning library tailored for high-performance applications. It implements optimized gradient-boosted decision trees (GBDT), facilitating parallel tree construction and efficient management of extensive datasets. XGBoost employs a level-wise method to enhance split decision-making, thereby boosting both accuracy and computational effectiveness. It is extensively used for tasks such as regression, classification, and ranking. XGBoost provides support for regularization, handling of missing values, and custom objective functions.

2.5.6 LightGBM

LightGBM is a gradient boosting framework developed by Microsoft that employs tree-based learning algorithms. It is designed to be highly efficient, scalable, and speed, as it supports both parallel and GPU learning. To enhance training speed and reduce memory consumption, LightGBM employs a histogram-based technique for split finding. Unlike level-wise growth, it expands trees leaf-wise, potentially attaining higher accuracy but requiring careful tuning to avoid overfitting. It adeptly manages large and high-dimensional datasets and inherently supports categorical features.

2.5.7 LSTM (Long Short-Term Memory)

LSTM is a type of recurrent neural network (RNN) architecture designed to model sequential data and capture long-term dependencies. It addresses the vanishing gradient problem of traditional RNNs by introducing memory cells and gating mechanisms (input, output, and forget gates) that regulate the flow of information. They can remember information for long periods, making them effective for tasks where context and order matter. LSTMs require

significant computational resources and careful tuning but are highly effective for complex sequence modeling problems.

2.5.8 MLP (Multi-Layer Perceptron)

A Multi-Layer Perceptron (MLP) is a type of feedforward artificial neural network that includes a minimum of three layers: an input layer, one or multiple hidden layers, and an output layer. Neurons in one layer are fully connected to neurons in the subsequent layer, employing nonlinear activation functions to capture intricate patterns. MLPs can learn non-linear relationships and are utilized for both regression and classification tasks. They are trained through backpropagation and gradient descent. MLPs are essential in deep learning, functioning as fundamental components for more advanced architectures.

2.5.9 GRU (Gated Recurrent Unit)

GRU is a recurrent neural network(RNN) design that is more straightforward than LSTM. To manage information flow and identify relationships in sequential data, it makes use of gating methods, such as reset and update gates. Because GRUs need fewer parameters than LSTMs, training is quicker, and there is less chance of overfitting. They work well for modeling language, time series, and other sequential data where long-term dependencies are crucial.

2.5.10 ANN (Artificial Neural Network)

An artificial neural network (ANN) is a computational model made up of layers of interconnected nodes (neurons) that is modeled after the structure of the human brain. By utilizing algorithms like backpropagation to modify the weights of connections during training, ANNs are able to extract intricate patterns from data. They are employed in many different tasks, including image recognition, regression, and classification. From basic single-layer networks to deep networks with numerous hidden layers, artificial neural networks (ANNs) can have a variety of designs.

2.5.11 GAM (Generalized Additive Model)

A Generalized Additive Model (GAM) is a flexible regression technique that models the relationship between the dependent variable and independent variables as a sum of smooth functions. Each predictor's effect is estimated using non-parametric functions, allowing for non-linear relationships while maintaining interpretability. GAMs are useful when the relationship between variables is not strictly linear but needs to be understood and visualized. GAMs can be extended to handle various types of response variables and incorporate regularization to prevent overfitting.

Chapter 3

Evaluation and Results

All the mentioned machine learning and deep learning algorithms were implemented on the solar power generation dataset, with models trained using the prepared input features. Initially, each model was trained using its default hyperparameters to establish a baseline performance, providing insight into how each algorithm performs without optimization. Following this, hyperparameter tuning was performed using GridSearchCV. To ensure robustness and prevent overfitting, k-fold cross-validation was employed during both baseline evaluation and tuning phases. The final performance of each model was assessed using error metrics such as **MSE, RMSE, MAE, and R^2**, derived from the cross-validated and tuned models, allowing for a fair and accurate comparison of their predictive capabilities.

3.1 Error Metrics Comparison

Table 3.1: Comparison of MSE, RMSE, MAE, and R^2 Scores Across Models (Best Values Highlighted)

Model	MSE	RMSE	MAE	R^2 Score
Linear Regression	404 421.37	635.94	455.63	0.7920
SVR	208 397.21	456.51	269.55	0.8400
KNN	218 701.82	467.66	281.62	0.8300
Random Forest	193 514.82	439.90	253.91	0.8500
XGBoost	200 662.78	447.95	274.40	0.8400
LightGBM	195 915.38	442.62	267.98	0.8460
LSTM	237 117.56	486.95	340.92	0.8136
ANN	229 073.23	478.62	279.88	0.8199
MLP	233 180.11	482.67	278.74	**0.8724**
GRU	251 380.02	501.38	310.54	0.8024
PoissonGAM	**191505.25**	**437.61**	**251.55**	0.8494

Mean Squared Error (MSE):

MSE measures the average of the squared differences between predicted and actual values. It's calculated as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of samples. Fig 3.1, given below, shows a bar graph, showing the comparison of performances between the various models in terms of MSE used on the dataset:

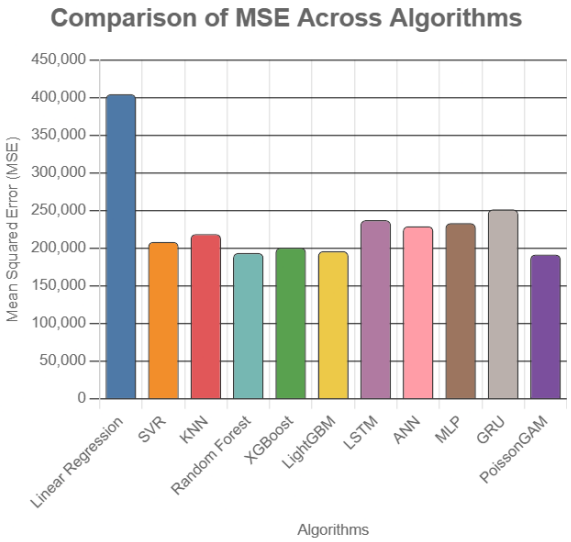

Figure 3.1: Comparison of MSE between the models

It shows PoissonGAM with the lowest MSE (191,505), followed by Random Forest (193,515) and LightGBM (195,915). Linear Regression has the highest MSE (404,421).

Root Mean Squared Error (RMSE):

RMSE is the square root of MSE, providing an error metric in the same units as the target variable:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

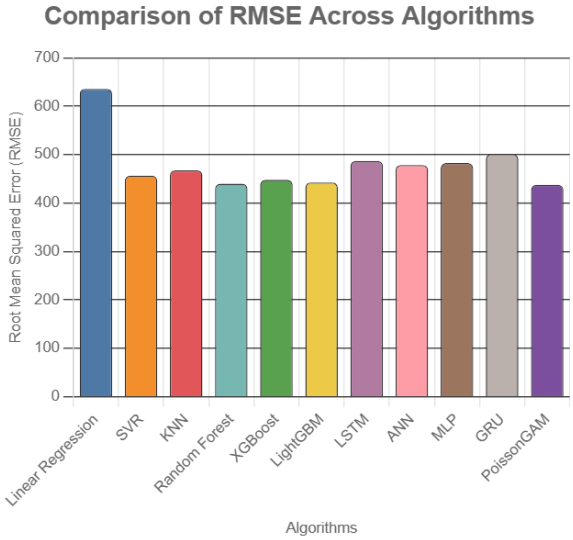

Figure 3.2: Comparison of RMSE between the models

Fig 3.2 shows PoissonGAM with the lowest RMSE (437.61), followed by Random Forest (439.90) and LightGBM (442.62). Linear Regression has the highest RMSE (635.94).

Mean Absolute Error (MAE):

MAE measures the average of the absolute differences between predicted and actual values:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

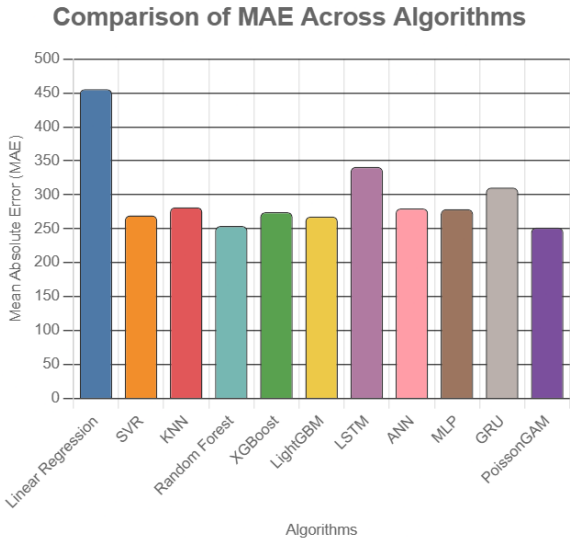

Figure 3.3: Comparison of MAE between the models

Fig 3.3 shows PoissonGAM with the lowest MAE (251.55), followed by Random Forest (253.91) and LightGBM (267.98). Linear Regression has the highest MAE (455.63).

R^2 Score (Coefficient of Determination)

R^2 measures the proportion of variance in the dependent variable explained by the model. It's calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the actual values.

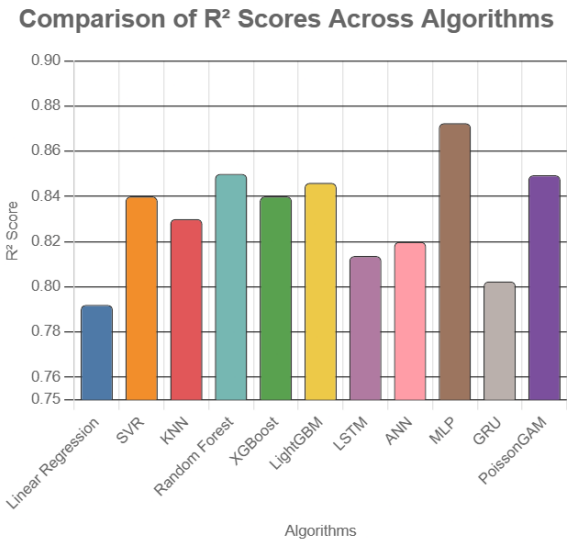

Figure 3.4: Comparison of R^2 between the models

Fig 3.4 shows that MLP has the highest R^2 score (0.872), followed closely by PoissonGAM (0.849) and Random Forest (0.850). GRU has the lowest R^2 score (0.802).

3.2 Plots of Actual versus Predicted Values

The plots for all models showing the Actual versus Predicted Values are shown in the graphs below. Two lines are plotted: blue for actual values and orange for predicted values. The x-axis typically represents a sequential time variable for the first 500 samples, allowing for easier plotting. The y-axis represents the target variable being predicted, "Power (kW)". Gaps between the actual and predicted lines highlight prediction errors. Large deviations indicate where the model struggles, such as failing to capture sudden spikes or dips.

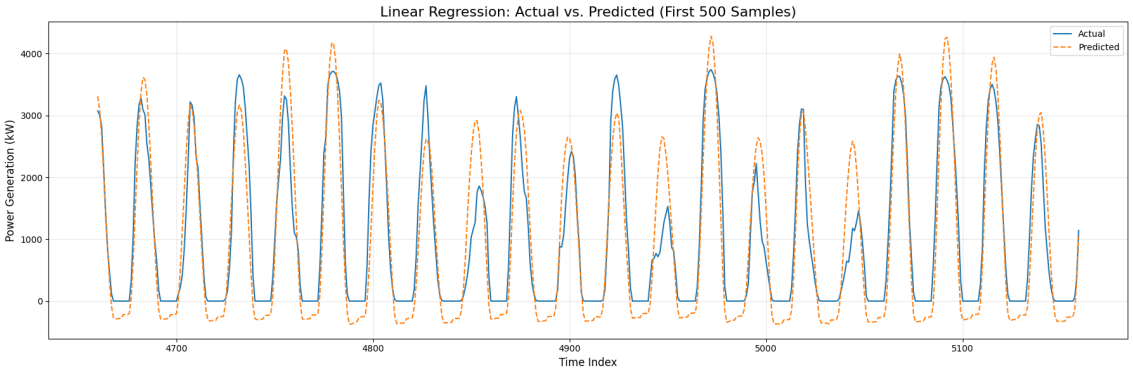

Figure 3.5: Linear Regression

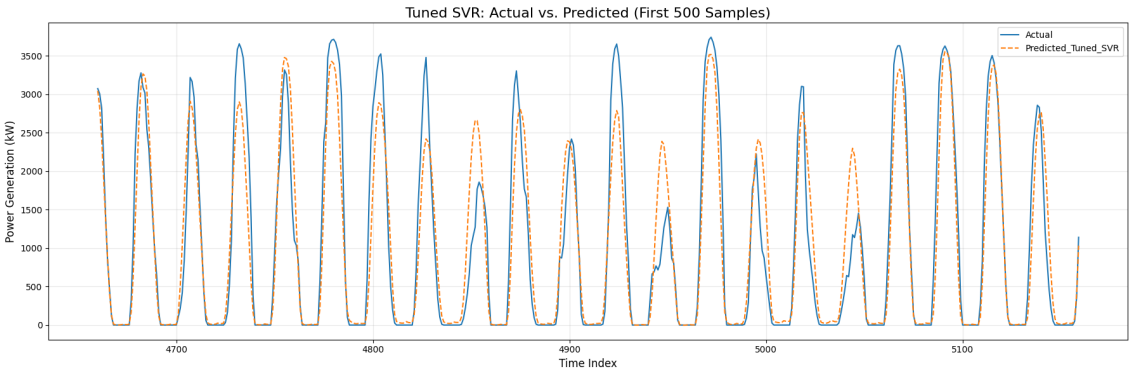

Figure 3.6: SVR

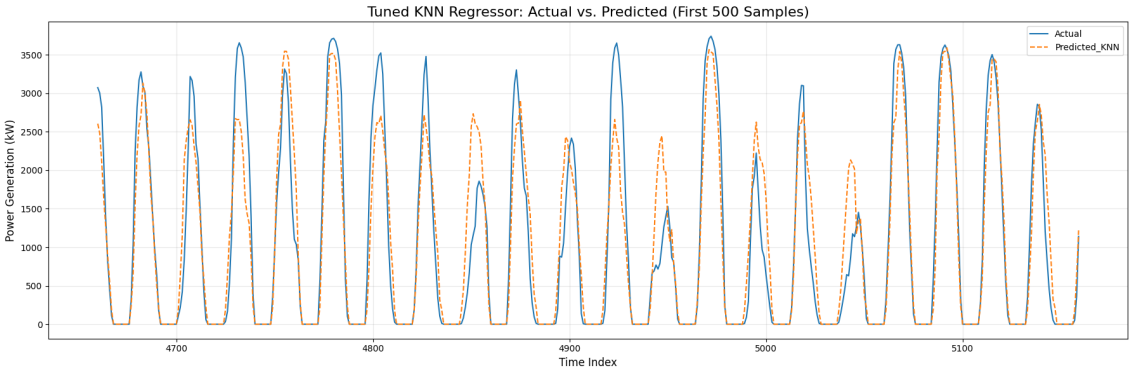

Figure 3.7: KNR

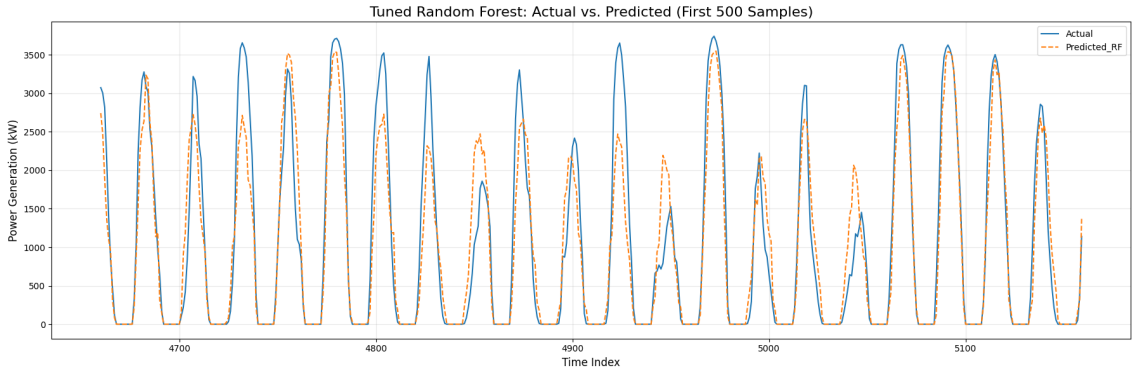

Figure 3.8: Random Forest

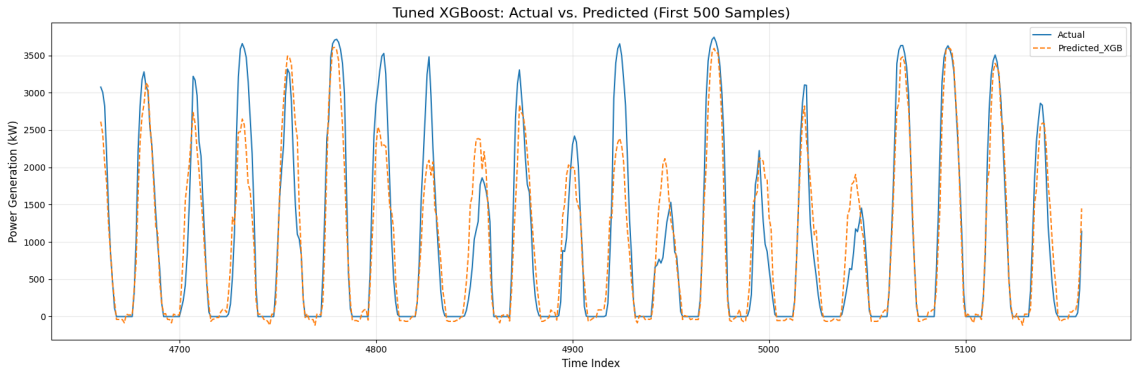

Figure 3.9: XGBoost

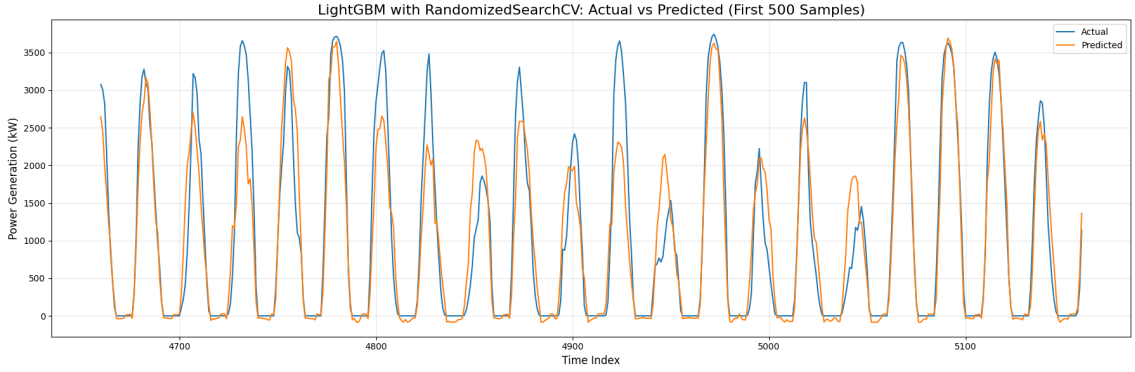

Figure 3.10: LightGBM

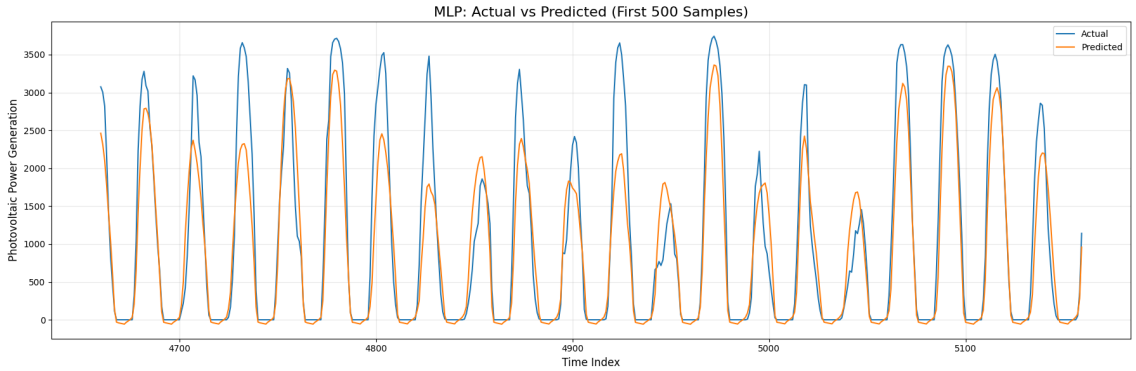

Figure 3.13: MLP

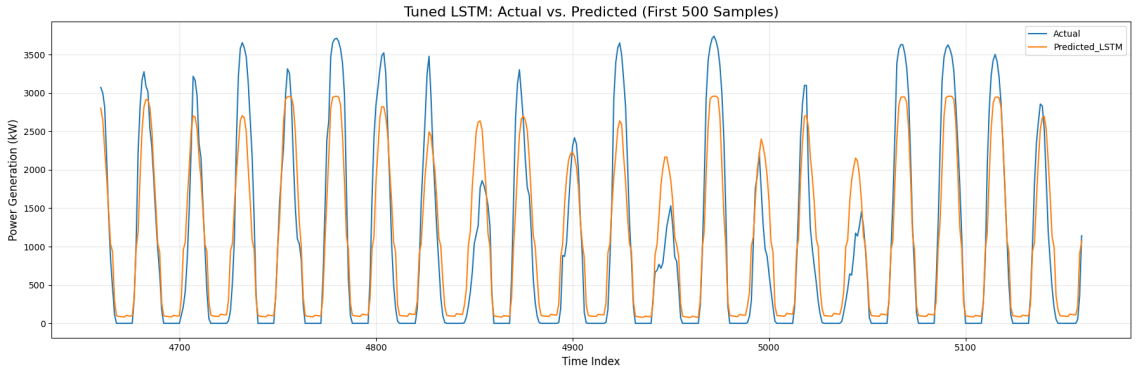

Figure 3.11: LSTM

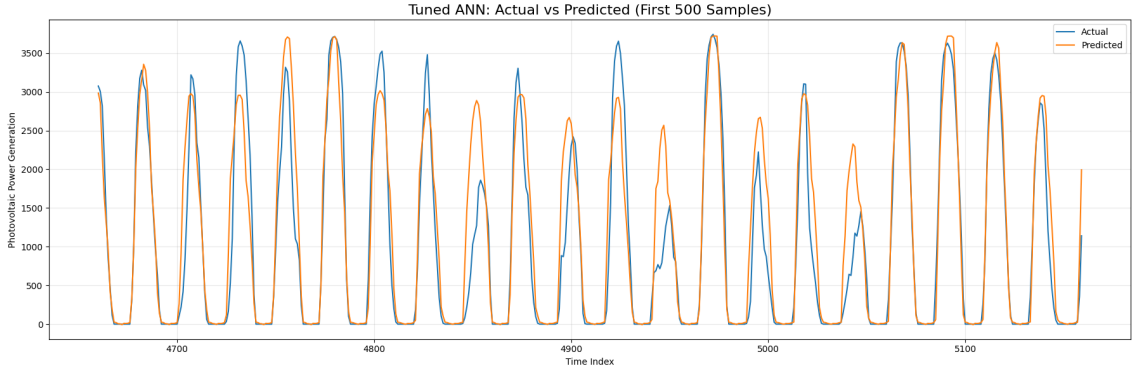

Figure 3.12: ANN

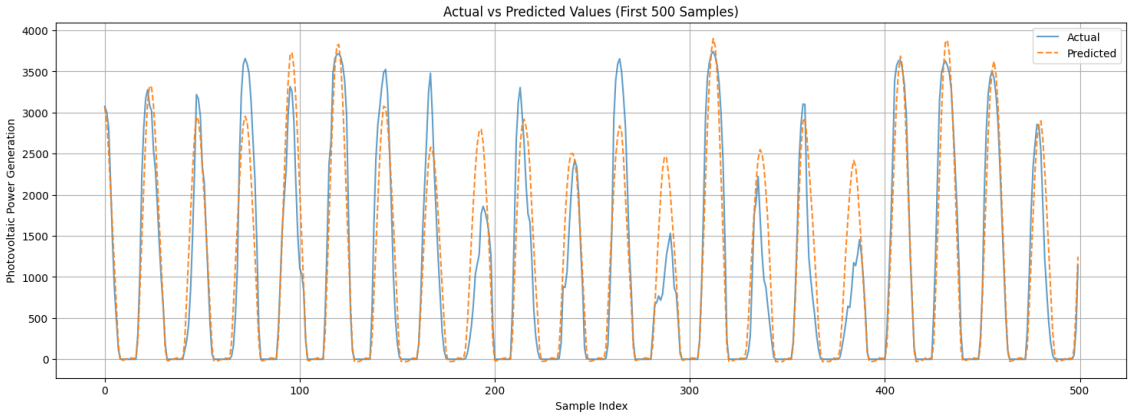

Figure 3.14: GRU

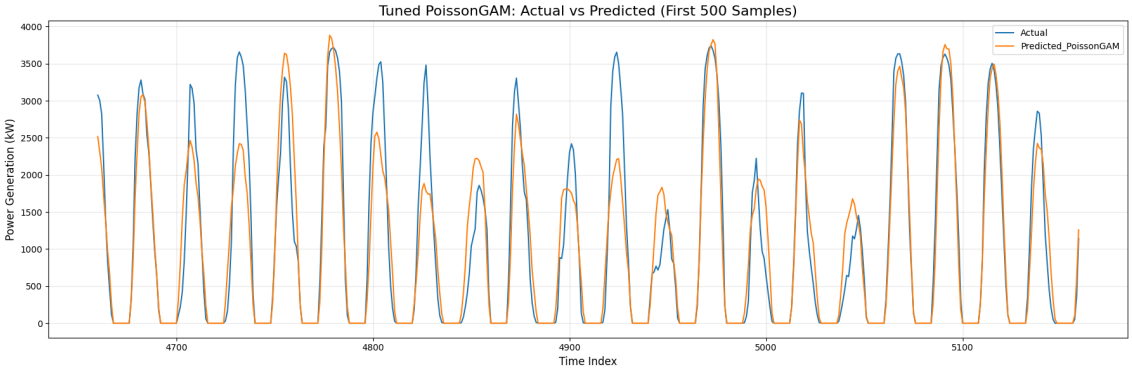

Figure 3.15: PoissonGAM

Chapter 4

4.1 Conclusion

- Linear Regression: Linear Regression fails on cyclic patterns and underpredicts peaks. It's unfit for this non-linear data.
- Tuned SVR: SVR follows trends but misses peak amplitudes. It has variability limitations.
- KNN Regressor: KNN Regressor aligns with trends but underpredicts peaks. It struggles with capturing variability.
- Random Forest: Random Forest tracks patterns but misses peak heights. It struggles with extremes.
- XGBoost: XGBoost captures trends but underpredicts peaks. It also faces challenges with extremes.
- LightGBM: LightGBM matches trends but underestimates peaks.
- LSTM: LSTM handles sequences well but lags on rapid changes.
- ANN: ANN follows trends but underpredicts sharp peaks. It has issues with rapid changes.
- MLP: MLP tracks trends but underpredicts peaks. It struggles with non-linear fluctuations.
- PoissonGAM: PoissonGAM tracks trends but underpredicts peaks.

All the models, except for linear regression, are performing well in finding patterns. Overall, while most models capture general trends, they commonly struggle with extreme values and rapid fluctuations, highlighting areas for improvement in handling non-linear and highly variable data.