A Deep Machine Hybrid Learning model for the Local Forecasting of renewable sources: Case study on the combination of the real world AI models.

CHELBI GROUP LABs

CHELBI ZAHIA ZINEB

Abstract

This project explores various machine and deep learning models to predict power generated based on meteorological and astronomical data. The dataset, containing information such as temperature, wind speed, sky cover, and time-related features, is preprocessed to handle missing values and prepare it for model training. A range of regression models, including traditional methods like Linear Regression, ensemble methods like Random Forest, Gradient Boosting, AdaBoost, CatBoost, Voting Regressor, and Stacking Regressor, as well as deep learning models like ANN, CNN, RNN, LSTM, CNN-LSTM, and Bidirectional LSTM, are trained and evaluated. The performance of each model is assessed using metrics such as Root Mean Squared Error (RMSE), R2 Score, and Mean Absolute Error (MAE). The project aims to identify the most effective model for accurate power generation prediction, which can be crucial for optimizing energy management and resource allocation.

Introduction

Accurate forecasting of power generation is a critical task for effective energy management, grid stability, and the integration of renewable energy sources. Fluctuations in power output from sources like solar and wind are heavily influenced by environmental factors and time-based patterns. This project aims to develop and compare various machine learning models to predict power generation based on a comprehensive dataset containing meteorological and astronomical information. By evaluating a diverse set of regression techniques, from traditional methods to deep learning architectures, we seek to identify the most robust and accurate model for predicting power output. The insights gained from this analysis can support better planning, scheduling, and utilization of generated power.

Dataset Description

This dataset contains information related to power generation, likely from a renewable energy source like solar or wind, recorded over two years (2008 and 2009). The data is collected at different times of the day, indicated by the 'First Hour of Period'. Key features include meteorological data such as 'Average Temperature (Day)', 'Average Wind Direction (Day)', 'Average Wind Speed (Day)', 'Sky Cover', 'Visibility', and 'Relative Humidity', as well as 'Average Barometric Pressure (Period)'. It also includes astronomical information like 'Day of Year' and 'Distance to Solar Noon', and a boolean indicating 'Is Daylight'. The target variable for prediction is 'Power Generated'. There are a total of 2920 entries and 16 columns. One column, 'Average Wind Speed (Period)', has one missing value, which was handled by dropping the row. The 'Is Daylight' column was dropped during preprocessing. The dataset is suitable for building a regression model to predict power generation based on the provided features.

We have done the data general information and statistical description for the purpose of the clarity data analysis and visualization step, the following figure can perform a better understanding for the

columns data row combination and data types in order to start data processing and mining steps after the organization of the project pipeline.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2920 entries, 0 to 2919
Data columns (total 16 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Day of Year                           2920 non-null   int64
 1   Year                                  2920 non-null   int64
 2   Month                                 2920 non-null   int64
 3   Day                                   2920 non-null   int64
 4   First Hour of Period                  2920 non-null   int64
 5   Is Daylight                           2920 non-null   bool
 6   Distance to Solar Noon                2920 non-null   float64
 7   Average Temperature (Day)             2920 non-null   int64
 8   Average Wind Direction (Day)          2920 non-null   int64
 9   Average Wind Speed (Day)              2920 non-null   float64
 10  Sky Cover                             2920 non-null   int64
 11  Visibility                            2920 non-null   float64
 12  Relative Humidity                     2920 non-null   int64
 13  Average Wind Speed (Period)           2919 non-null   float64
 14  Average Barometric Pressure (Period)  2920 non-null   float64
 15  Power Generated                       2920 non-null   int64
dtypes: bool(1), float64(5), int64(10)
memory usage: 345.2 KB
```

Figure 1: Dataset General Information.

We can observe the generated power of the year over the days affection distribution and here we start making the inspection of the predictive pipeline we need to follow.

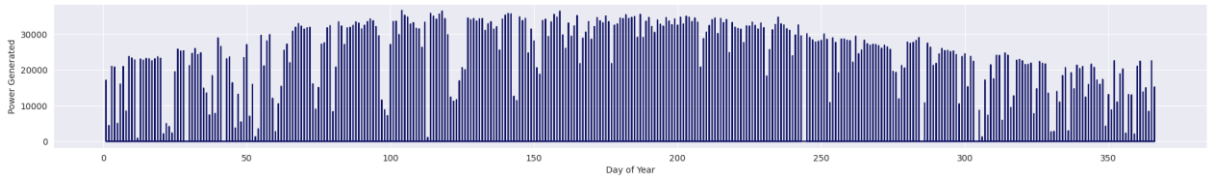

Figure 2: The power generated over Days of the year.

The correlation coefficients are one of the most powerful techniques to discover the feature importance step using linear relationships on the parameters, generating the heatmap can make the feature selection and input optimization phase less complicated before going to the mutual information or any other techniques.

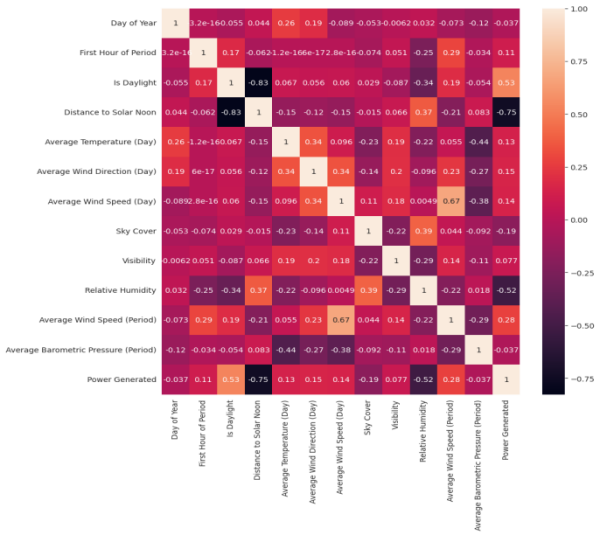

Figure 3: The correlation heatmap matrix of the dataset.

Data Processing and Mining

The data processing and mining pipeline involved several key steps to prepare the raw data for model training and analysis:

Data Loading: The dataset was loaded into a pandas DataFrame from the provided CSV file ('/content/BigML_Dataset_5f50a4cc0d052e40e6000034 (1).csv').

Initial Exploration: The dataset was explored to understand its structure, identify missing values, and examine the data types of each column using methods like df.head(), df.info(), and df.describe().

Handling Missing Values: The Average Wind Speed (Period) column was identified to have one missing value. The row containing this missing value was dropped from the dataset using df.dropna().

Sometimes we have to visualize the outlier values before going to the next steps so that we do not get errors in the scaling or the feature dependencies, we can solve this problem using zscore technique, IQR or Isolation forest.

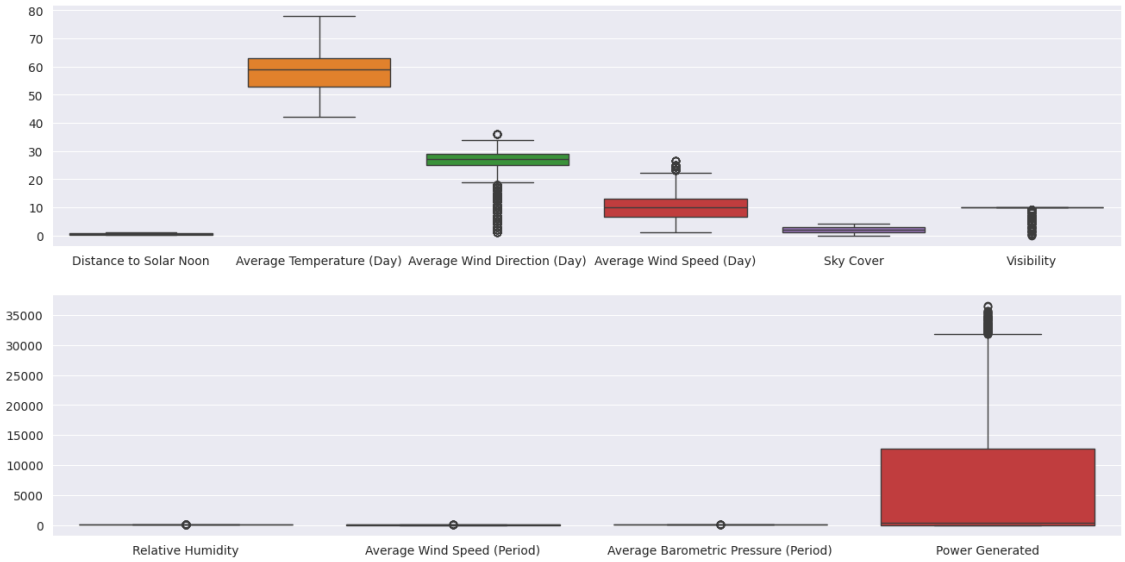

Figure 4: Outlier values visualization.

Feature Selection: Based on initial exploration and potential redundancy, the 'Is Daylight' column was dropped. Additionally, the 'Day', 'Year', and 'Month' columns were dropped before training the models, likely because 'Day of Year' was considered a sufficient representation of the time component or to avoid multicollinearity.

Data Splitting: The dataset was split into training, validation, and testing sets to allow for proper model training, hyperparameter tuning (using the validation set), and final evaluation on unseen data. The split ratios were 70% for training, 15% for validation, and 15% for testing.

Feature Scaling: The features were scaled using StandardScaler from sklearn.preprocessing. This is an important step for many machine learning models, especially those that are sensitive to the magnitude of features, such as neural networks and SVMs. Scaling ensures that all features contribute equally to the model training process.

Data Reshaping for Time Series Models: For the time series models (LSTM, CNN-LSTM, Bidirectional LSTM, and RNN), the input data was reshaped into a 3D format (number of samples, number of time steps, number of features), as required by these network architectures. In this case, each sample was treated as a single time step with multiple features.

Data Modeling and Predictions

In this project, a variety of regression models were implemented and evaluated to predict power generation. The models explored include traditional machine learning techniques, ensemble methods, and deep learning architectures. The data, after undergoing the preprocessing steps, was used to train and test each model.

The following models were trained and evaluated:

AdaBoost Regressor: An ensemble boosting method that combines multiple weak learners (typically decision trees) to create a strong learner. The default hyperparameters were used.

Gradient Boosting Regressor: Another ensemble boosting method that builds trees sequentially, where each new tree corrects the errors of the previous one. The default hyperparameters were used.

KNN Regressor: A non-parametric lazy learning algorithm that predicts the value of a new data point based on the values of its nearest neighbors in the training data. The default hyperparameters were used.

SVR (Support Vector Regression): A powerful model that uses support vector machines to perform regression. It finds a hyperplane that best fits the data points while minimizing the error. The default hyperparameters were used.

Decision Tree Regressor: A tree-like structure where each internal node represents a test on a feature, each branch represents an outcome of the test, and each leaf node represents the predicted value. The default hyperparameters were used.

ANN (Artificial Neural Network): A deep learning model consisting of interconnected layers of neurons. The architecture included multiple dense layers with ReLU activation and dropout layers for regularization. The model was compiled with the Adam optimizer and trained for 200 epochs with a batch size of 128.

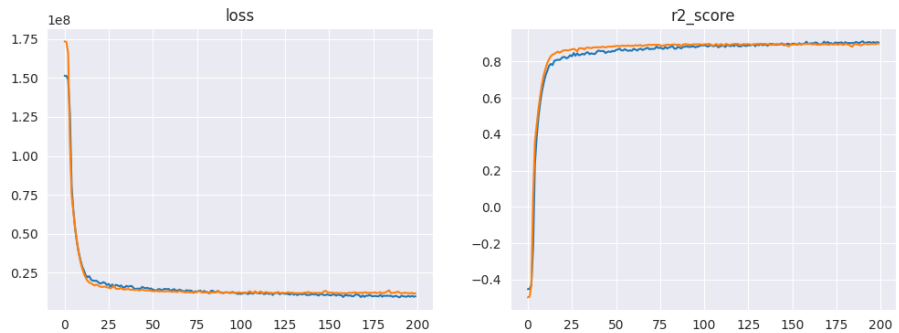

Figure 5: ANN model MSE,R2 score.

CNN (Convolutional Neural Network): Primarily used for grid-like data such as images, a 1D CNN was adapted here to extract features from the sequential input data. The model included a Conv1D layer, MaxPooling1D, Flatten, and dense layers. It was trained with the Adam optimizer for 200 epochs with a batch size of 128.

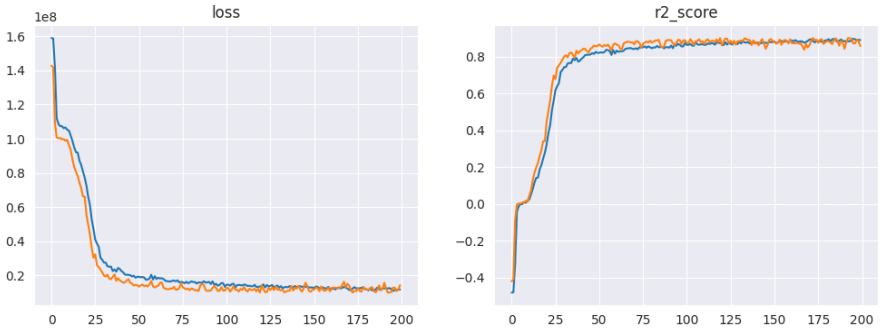

Figure 6: CNN model MSE,R2 score.

RNN (Simple Recurrent Neural Network): A type of neural network designed to handle sequential data by maintaining a hidden state that captures information from previous steps. The model included SimpleRNN layers with tanh activation and dense layers. It was trained with the Adam optimizer for 75 epochs with a batch size of 64.

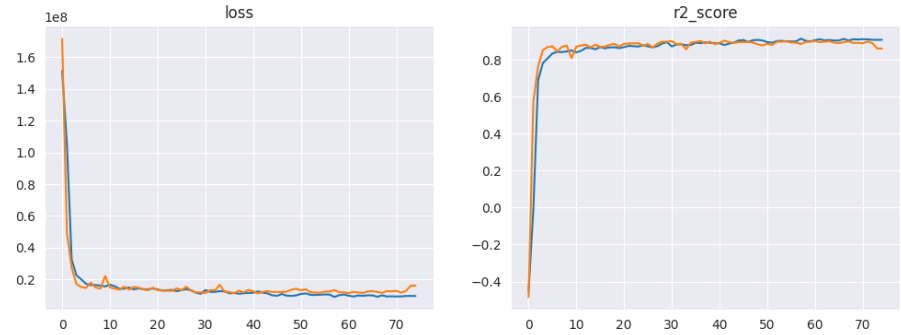

Figure 7: RNN model MSE,R2 score.

LSTM (Long Short-Term Memory): A type of RNN that is particularly effective at learning long-term dependencies in sequential data. The model incorporated LSTM layers, Conv1D, MaxPooling1D, Flatten, and dense layers. It was trained with the Adam optimizer for 200 epochs with a batch size of 128.

CatBoost Regressor: A gradient boosting algorithm that uses a symmetric decision tree as the base predictor and is known for its high performance and handling of categorical features (although not explicitly used for categorical features in this case). The default hyperparameters were used.

Voting Regressor: An ensemble method that combines the predictions of multiple individual regressors by averaging them. In this case, a Linear Regression, Decision Tree, and KNN model were combined.

Stacking Regressor: An ensemble method where the predictions of multiple base regressors are used as input to a final meta-regressor. Here, Linear Regression and Decision Tree were used as base estimators, and SVR was used as the final estimator.

CNN-LSTM: A hybrid deep learning model that combines the feature extraction capabilities of CNNs with the sequence modeling capabilities of LSTMs. The model architecture included LSTM layers, Conv1D, MaxPooling1D, Flatten, and dense layers. It was trained with the Adam optimizer for 200 epochs with a batch size of 128.

Bidirectional LSTM: A type of LSTM that processes the sequence data in both forward and backward directions, allowing it to capture dependencies from both past and future time steps. The model used Bidirectional LSTM layers and dense layers. It was trained with the Adam optimizer for 200 epochs with a batch size of 128.

The performance of each model was evaluated using the following metrics on the test set:

	Model	RMSE	R2	MAE	hyperparameters tuning
0	AdaBoost Regressor	3489.921147	88.584346	208723.267966	default
1	Gradient Boosting Regressor	2536.891742	93.967832	137792.280665	default
2	KNN Regressor	4037.068083	84.724279	243055.159817	default
3	SVR	12163.852138	-38.679358	680731.867447	default
4	Decision Tree Regressor	4145.464016	83.892956	172520.091324	default
5	ANN	2917.872314	92.020011	158982.653809	epochs:200, batch_size:128, optimizer:adam
6	CNN	3335.353516	89.573145	195637.695312	epochs:200, batch_size:128, optimizer:adam
7	RNN	3009.397217	91.511548	159530.969238	epochs:75, batch_size:64, optimizer:adam
8	LSTM	2793.333984	92.686665	150224.267578	epochs:200, batch_size:128, optimizer:adam
9	CatBoost Regressor	2509.650848	94.096682	123291.341606	default
10	Voting Regressor	3789.268870	86.542004	267807.635612	Linear Regression, Decision Tree, KNN
11	Stacking Regressor	11877.839553	-32.234406	652835.587938	Linear Regression, Decision Tree (estimators),...
12	CNN-LSTM	2748.373047	92.920202	146704.406738	epochs:200, batch_size:128, optimizer:adam
13	Bidirectional LSTM	3094.090576	91.027045	156083.959961	epochs:200, batch_size:128, optimizer:adam

Figure 7: Performance table.

Root Mean Squared Error (RMSE): Measures the square root of the average of the squared differences between the predicted and actual values. A lower RMSE indicates better performance.

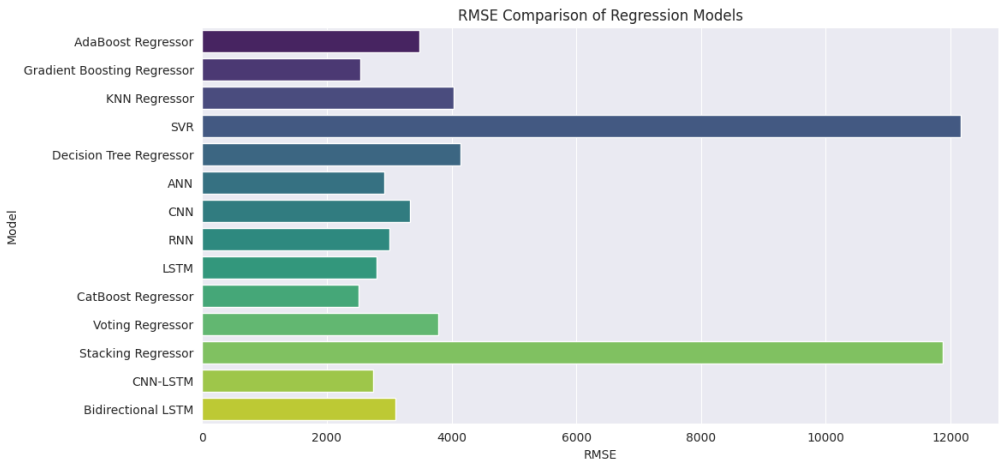

Figure 8: Models RMSE performances.

R2 Score: Represents the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R2 score indicates a better fit.

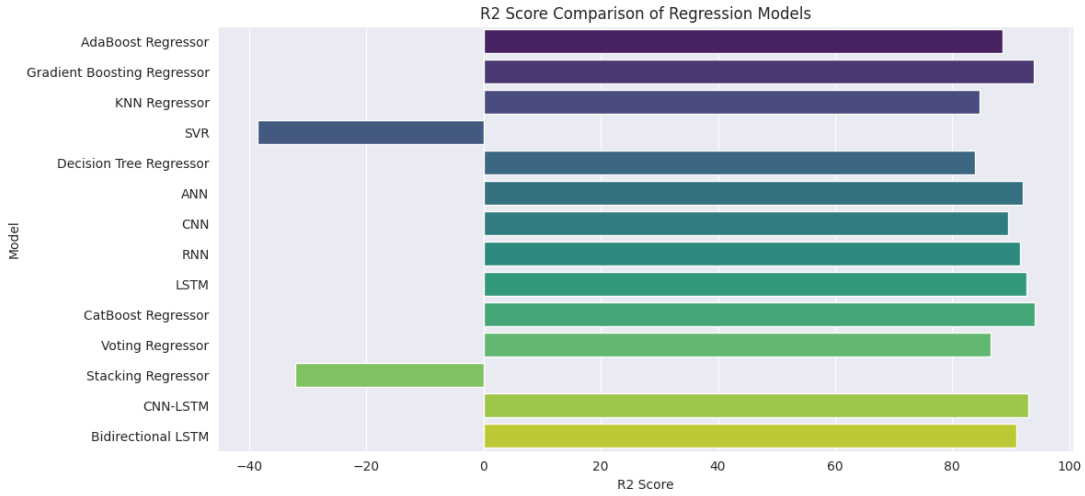

Figure 9: Models R2 Score performances.

Mean Absolute Error (MAE): Measures the average of the absolute differences between the predicted and actual values. A lower MAE indicates better performance.

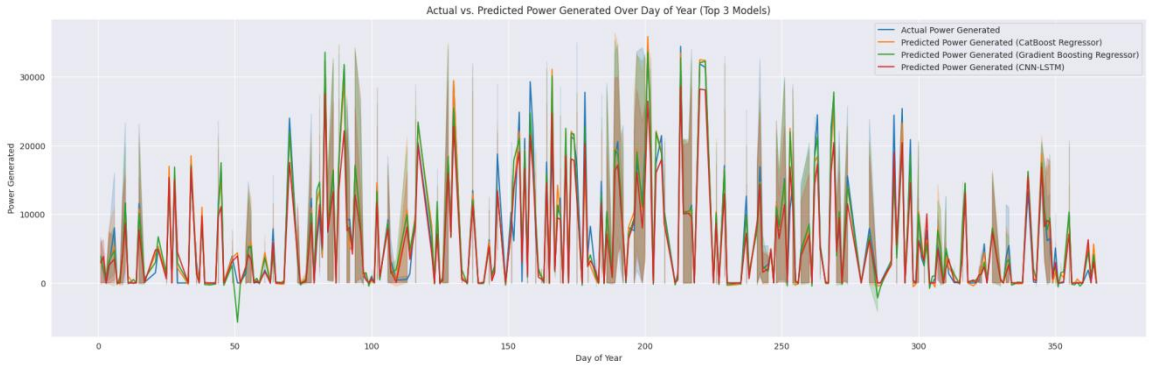

Figure 10: Generated actual and predicted power.

Cheking for the best three performance models we can go to the next phase of simulations and laboratory experiemental computing.

index	Model	RMSE	R2	MAE	hyperparameters tuning
9	CatBoost Regressor	2509.650847722899	94.09668213240994	123291.34160590936	default
1	Gradient Boosting Regressor	2536.8917423651014	93.96783200032125	137792.28066515175	default
12	CNN-LSTM	2748.373046875	92.92020201683044	146704.40673828125	epochs:200, batch_size:128, optimizer:adam

Figure 11: Best three models.

We can make the difference in regression plots of the best models for the purpose of choosing the model to use in the software application board.

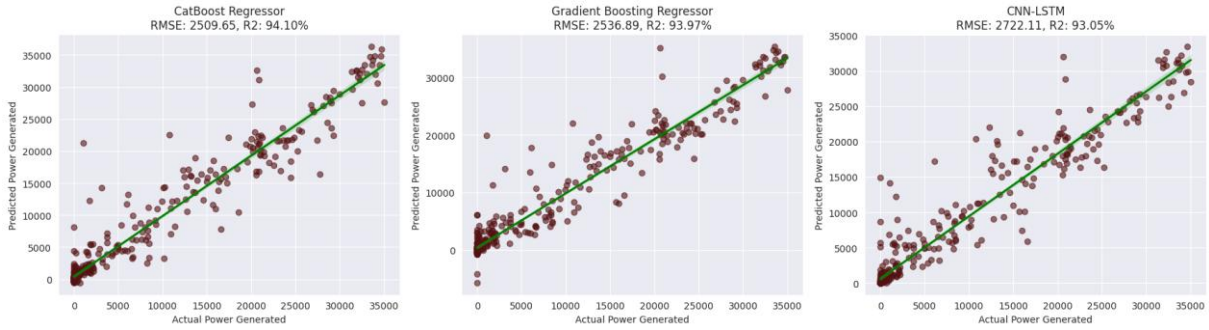

Figure 11: Regression plots of the best three models.

Conclusion

Based on the evaluation of various regression models for power generation prediction, the CatBoost Regressor and Gradient Boosting Regressor demonstrated the best performance, achieving the lowest RMSE and highest R2 scores. The CNN-LSTM model also showed promising results, indicating the potential of deep learning approaches for this time-series regression problem. The traditional models like Linear Regression and SVR did not perform as well, highlighting the complexity of the relationship between the input features and power generation.

The key contributions of this project include:

A comprehensive comparison of a wide range of machine learning and deep learning models for power generation prediction using a real-world dataset.

Identification of high-performing models (CatBoost and Gradient Boosting) that can be used for accurate power forecasting.

Demonstration of the applicability of deep learning architectures (CNN-LSTM) for this specific regression task, even with a relatively small dataset size.

A clear data processing and modeling pipeline that can be adapted for similar time-series regression problems in the energy domain.

These findings can inform the selection of appropriate models for power generation forecasting, leading to improved energy planning, grid stability, and efficient integration of renewable energy sources. Future work could involve hyperparameter tuning for the deep learning models, exploring more advanced time series architectures, and incorporating additional relevant features to further enhance prediction accuracy.