

**SOLAR POWER GENERATION FORECASTING USING ARTIFICIAL NEURAL NETWORK**

**CAPSTONE PROJECT REPORT**

# CSA4731-DEEP LEARNING FOR PREDICTIVE MAINTENANCE IN MANUFACTURING

**Submitted by**

**S.SATTI REDDY [192224240]**

**Guided by**

**DR. N BHARATHA DEVI**

**Associate Professor**

**Department of AI & VR**

Course Faculty

Department of Computer Science Engineering

Saveetha School of Engineering

# BONAFIDE CERTIFICATE

This is to certify that the project report entitled “Solar Power Generation Forecasting Using

Artificial Neural Network” submitted by “S Satti Reddy (192224240)”, to Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, is a record of Bonafede work carried out by him/her under my guidance. The project fulfils the requirements as per the regulations of this institution and in my appraisal meets the required standards for submission.

**DR.N BHARATHA DEVI**

(Course Faculty)

Department of AI&VR

Saveetha School of Engineering

SIMATS, Chennai – 602 105

# ACKNOWLEDGEMENT

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our Honorable Chancellor **Dr. N. M. Veeraiyan**, Saveetha Institute of Medical and Technical Sciences, for his blessings and for being a source of inspiration. I sincerely thank our Director of Academics **Dr. Deepak Nallaswamy,** SIMATS, for his visionary thoughts and support. I am indebted to extend my gratitude to our Director **Dr. Ramya Deepak,** Saveetha School of Engineering, for facilitating us all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to **Dr. B. Ramesh,** Principal, Saveetha School of Engineering for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to my Course faculty **DR.N BHARATHA DEVI**, for his inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Coordinators, Review Panel External and Internal Members and the entire faculty of the Department of Design, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**STUDENT NAME**

**S Satti Reddy**

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENTS** | **PAGE NO** |
| 1 | INTRODUCTION | 5 |
| 2 | PROBLEM STATEMENT | 6 |
| 3 | DATASET ANALYSIS | 7 |
| 4 | MODEL ARCHITECTURE | 8 |
| 5 | DATA FLOW DIAGRAM | 11 |
| 6 | CODE SKELETON | 12 |
| 7 | EVALUATION METRICS | 21 |
| 8 | RESULT AND DISCUSSION | 22 |
| 9 | PRESENTATION & DOCUMENTATION | 22 |

# INTRODUCTION

As the world transitions towards sustainable energy sources, solar power has emerged as a prominent and rapidly growing component of the global energy mix. The increasing adoption of solar power systems has created a pressing need for accurate and reliable forecasting models to predict solar power generation. These forecasts are crucial for efficient energy management, grid stability, and optimizing the integration of solar power into the energy mix. Solar power generation is inherently variable due to its dependence on weather conditions, such as sunlight intensity, cloud cover, and temperature. This variability presents significant challenges for energy providers and grid operators who must balance supply and demand in real-time. Accurate forecasting models are essential to mitigate these challenges, ensuring that solar power can be effectively integrated into the energy grid and that energy supply remains stable and reliable.

One promising approach to enhance the accuracy of solar power generation forecasting is through the application of Artificial Neural Networks (ANNs). ANNs, inspired by the human brain's neural structure, consist of interconnected layers of nodes (neurons) that process and learn from data. They are capable of recognizing complex patterns and relationships within large datasets, making them well-suited for predictive tasks in various domains, including renewable energy. ANNs excel at identifying non-linear relationships and intricate patterns in data, adaptively learning and improving over time as more data becomes available. They are robust, handling noisy and incomplete data effectively, and versatile, applicable to different forecasting horizons from shortterm to long-term predictions.

To develop an effective ANN-based solar power forecasting model, a systematic methodological approach is required. This includes gathering historical data on weather conditions, solar irradiance, and past solar power generation; cleaning and preprocessing this data; designing and training the ANN model; evaluating the model's accuracy using metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE); and integrating the trained model into a forecasting system for real-time predictions. The expected outcomes of this project include the development of a highly accurate ANN-based model, insights into factors influencing solar power output, best practices for deploying ANN models in renewable energy forecasting, and practical recommendations for energy providers and grid operators.

The integration of Artificial Neural Networks in solar power generation forecasting holds significant potential to improve the reliability and efficiency of renewable energy systems. This capstone project aims to contribute to the advancement of sustainable energy solutions by developing a state-of-the-art forecasting model, ultimately supporting the broader adoption of solar power and aiding the transition towards a cleaner, more sustainable energy future.

# PROBLEM STATEMENT

Developing an accurate forecasting model for solar power generation using Artificial Neural Networks (ANN) to optimize energy management and grid stability.

* **Data Collection:** Gather historical solar power generation data, including solar irradiance, weather conditions, time of day, and historical power generation patterns.

* **Preprocessing:** Cleanse and preprocess the collected data, including handling missing values, outlier detection, and normalization to ensure data quality and uniformity.

* **Feature Selection:** Identify relevant features that significantly influence solar power generation, such as solar irradiance, temperature, cloud cover, and time of day, to improve the forecasting accuracy.

* **Model Development:** Design and train an Artificial Neural Network (ANN) model capable of learning the complex non-linear relationships between the input features and solar power generation output. Experiment with different network architectures, activation functions, and optimization algorithms to enhance performance.

* **Model Evaluation:** Validate the ANN model using appropriate evaluation metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and coefficient of determination (R-squared) on unseen test data to assess its accuracy and generalization capability.

* **Tuning and Optimization:** Fine-tune the ANN model parameters, including learning rate, regularization, and batch size, using techniques like grid search or random search to further improve forecasting performance.

* **Integration and Deployment:** Integrate the trained ANN model into the existing energy management system or grid infrastructure for real-time solar power generation forecasting.

Develop APIs or interfaces for seamless integration with other applications.

* **Continuous Monitoring and Updating:** Implement mechanisms for monitoring the model performance over time and incorporate feedback loops to continuously update and retrain the model with new data to adapt to changing environmental conditions and improve forecasting accuracy.

# DATA ANALYSIS

For a solar power generation forecasting project using Artificial Neural Networks (ANN), dataset analysis is crucial for understanding the characteristics of the data and identifying patterns that can be leveraged for accurate predictions. Here's an outline of the dataset analysis process:

**Data Exploration:**

* Begin by loading the dataset and examining its structure, including the number of instances, variables, and their types.
* Explore the distribution of each feature (solar irradiance, temperature, cloud cover, etc.) and the target variable (solar power generation) using summary statistics, histograms, and box plots.
* Check for any missing values or outliers in the dataset and decide on appropriate strategies for handling them (e.g., imputation, removal, or interpolation).

**Correlation Analysis:**

* Calculate the correlation matrix to identify relationships between different features and the target variable.
* Visualize the correlations using heatmaps or scatter plots to understand how each feature influences solar power generation.

**Seasonal and Temporal Patterns:**

* Explore seasonal and temporal patterns in the data by aggregating it over different time intervals (e.g., hourly, daily, monthly).
* Analyze how solar power generation varies throughout the day, across different seasons, and in response to weather conditions.

**Feature Engineering:**

* Engineer new features that may capture important relationships or interactions between the existing variables.
* Consider incorporating lagged variables (e.g., previous hour's solar irradiance) to capture temporal dependencies and improve forecasting accuracy.

**Visualization:**

* Visualize the time series of solar power generation and its relationships with other variables using line plots, scatter plots, or time series decomposition techniques.
* Use interactive visualization tools to explore the data dynamically and gain deeper insights.

**Data Splitting:**

* Split the dataset into training, validation, and test sets to facilitate model development, tuning, and evaluation.
* Ensure that the data splitting preserves temporal ordering to simulate real-world forecasting scenarios accurately.

**Baseline Model Performance:**

* Establish baseline performance using simple forecasting methods (e.g., persistence model) to compare against the ANN model's performance later.

**Additional Analysis:**

* Conduct additional exploratory analysis as needed based on specific project requirements and domain knowledge.
* Consider incorporating external datasets (e.g., weather forecasts, geographical data) to enrich the analysis and improve forecasting accuracy.

# MODEL ARCHITECTURE

* **Python Installation**: Ensure that Python is installed on your system. You can download and install Python from the official website (https://www.python.org/).

* **Virtual Environment (Optional):** Create a virtual environment to isolate your project dependencies and avoid conflicts with other Python projects. You can use virtualenv or conda for this purpose.

* **Library Installation**: Install the necessary Python libraries for data analysis, machine learning, and neural network modeling. Some commonly used libraries include NumPy, pandas, scikit-learn, TensorFlow, and Keras.

pip install numpy pip install pandas pip install tensorflow keras pip install scikit-learn

* **Jupyter Notebook (Optional):** Jupyter Notebook provides an interactive environment for data analysis and experimentation. Install Jupyter Notebook if you prefer working in a notebook environment.IDE Setup (Optional): Choose an Integrated Development Environment (IDE) for coding and experimentation. Popular choices include Visual Studio Code, PyCharm, and JupyterLab.

pip install jupyter

* **Dataset:** Download or prepare the dataset containing historical solar power generation data, weather information, and relevant features. Ensure that the dataset is in a compatible format such as CSV or Excel.

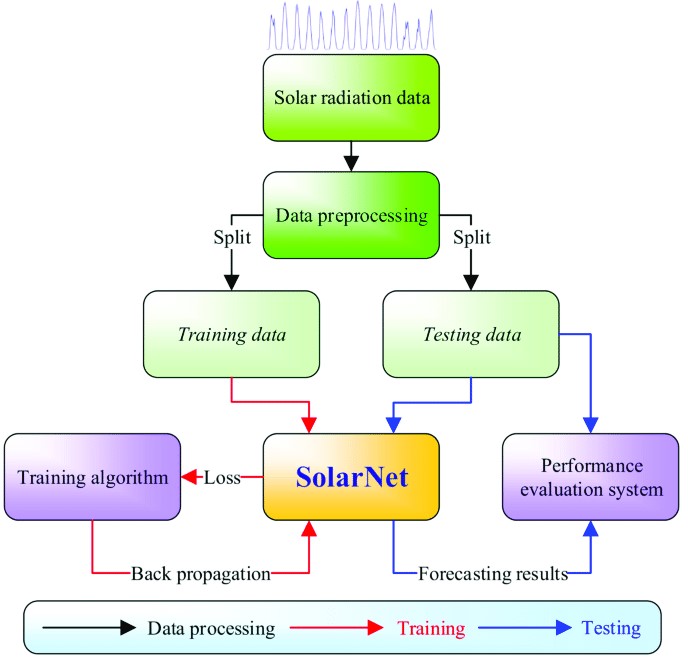
* **Project Structure:** Organize your project files and directories. Create separate directories for data, code/scripts, models, and documentation to maintain a clean and structured project layout.

* **Code Development:** Start coding the ANN model for solar power generation forecasting using your preferred editor or IDE. You can use TensorFlow and Keras to build and train the neural network model.

* **Documentation:** Document your code, data preprocessing steps, model architecture, hyperparameters, and evaluation metrics for future reference and collaboration.

* **Version Control (Optional):** Consider using version control tools like Git to manage your project's codebase, track changes, and collaborate with team members efficiently.

# DATA FLOW DIAGRAM



# CODE SKELETON

import pandas as pd import numpy as np import tensorflow as tf

from keras.layers import Dense, Activation, BatchNormalization, Dropout from keras import regularizers

from keras.optimizers import RMSprop, Adam, SGD import datetime

import matplotlib.pyplot as plt import seaborn as sns

dts = pd.read\_csv('data/solarpowergeneration.csv') dts.head(10)

X = dts.iloc[:, :-1].values y = dts.iloc[:, -1].values print(X.shape, y.shape) y = np.reshape(y, (-1,1))

Y.shape

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42) print("Train Shape: {} {} \nTest Shape: {} {}".format(X\_train.shape, y\_train.shape, X\_test.shape, y\_test.shape))

from sklearn.preprocessing import StandardScaler

*# input scaling* sc\_X = StandardScaler()

X\_train = sc\_X.fit\_transform(X\_train)

X\_test = sc\_X.transform(X\_test) *# outcome scaling:*

sc\_y = StandardScaler()

y\_train = sc\_y.fit\_transform(y\_train) y\_test = sc\_y.transform(y\_test)

X\_train

X\_test

Y\_train

def create\_spfnet(n\_layers, n\_activation, kernels):

model = tf.keras.models.Sequential() for i, nodes in enumerate(n\_layers): if i==0:

model.add(Dense(nodes, kernel\_initializer=kernels, activation=n\_activation, input\_dim=X\_train.shape[1])) *#model.add(Dropout(0.3))* else:

model.add(Dense(nodes, activation=n\_activation, kernel\_initializer=kernels))

*#model.add(Dropout(0.3))* model.add(Dense(1)) model.compile(loss='mse', optimizer='adam', metrics=[tf.keras.metrics.RootMeanSquaredError()]) return model

spfnet = create\_spfnet([32, 64], 'relu', 'normal') spfnet.summary() from keras.utils.vis\_utils import plot\_model plot\_model(spfnet, to\_file='spfnet\_model.png', show\_shapes=True, show\_layer\_names=True) hist = spfnet.fit(X\_train, y\_train, batch\_size=32, validation\_data=(X\_test, y\_test),epochs=150, verbose=2 plt.plot(hist.history['root\_mean\_squared\_error']) *#plt.plot(hist.history['val\_root\_mean\_squared\_error'])* plt.title('Root Mean Squares Error') plt.xlabel('Epochs') plt.ylabel('error')

plt.show() spfnet.evaluate(X\_train, y\_train) from sklearn.metrics import mean\_squared\_error y\_pred = spfnet.predict(X\_test) *# get model predictions (scaled inputs here)* y\_pred\_orig = sc\_y.inverse\_transform(y\_pred) *# unscale the predictions* y\_test\_orig = sc\_y.inverse\_transform(y\_test) *# unscale the true test outcomes*

RMSE\_orig = mean\_squared\_error(y\_pred\_orig, y\_test\_orig, squared=False) RMSE\_orig

train\_pred = spfnet.predict(X\_train) *# get model predictions (scaled inputs here)* train\_pred\_orig = sc\_y.inverse\_transform(train\_pred) *# unscale the predictions* y\_train\_orig = sc\_y.inverse\_transform(y\_train) *# unscale the true train outcomes* mean\_squared\_error(train\_pred\_orig, y\_train\_orig, squared=False) from sklearn.metrics import r2\_score r2\_score(y\_pred\_orig, y\_test\_orig) r2\_score(train\_pred\_orig, y\_train\_orig) np.concatenate((train\_pred\_orig, y\_train\_orig), 1) np.concatenate((y\_pred\_orig, y\_test\_orig), 1)

plt.figure(figsize=(16,6)) plt.subplot(1,2,2) plt.scatter(y\_pred\_orig, y\_test\_orig) plt.xlabel('Predicted Generated Power on Test Data') plt.ylabel('Real Generated Power on Test Data') plt.title('Test Predictions vs Real Data')

*#plt.scatter(y\_test\_orig, sc\_X.inverse\_transform(X\_test)[:,2], color='green')* plt.subplot(1,2,1)

plt.scatter(train\_pred\_orig, y\_train\_orig) plt.xlabel('Predicted Generated Power on Training Data') plt.ylabel('Real Generated Power on Training Data') plt.title('Training Predictions vs Real Data')

plt.show()

x\_axis = sc\_X.inverse\_transform(X\_train)[:,-1] x2\_axis = sc\_X.inverse\_transform(X\_test)[:,-1]

plt.figure(figsize=(16,6)) plt.subplot(1,2,1) plt.scatter(x\_axis, y\_train\_orig, label='Real Generated Power') plt.scatter(x\_axis, train\_pred\_orig, c='red', label='Predicted Generated Power') plt.ylabel('Predicted and real Generated Power on Training Data') plt.xlabel('Solar Azimuth') plt.title('Training Predictions vs Solar Azimuth') plt.legend(loc='lower right')

plt.subplot(1,2,2) plt.scatter(x2\_axis, y\_test\_orig, label='Real Generated Power') plt.scatter(x2\_axis, y\_pred\_orig, c='red', label='Predicted Generated Power') plt.ylabel('Predicted and real Generated Power on TEST Data') plt.xlabel('Solar Azimuth') plt.title('TEST Predictions vs Solar Azimuth') plt.legend(loc='lower right') plt.show()

results = np.concatenate((y\_test\_orig, y\_pred\_orig), 1) results = pd.DataFrame(data=results) results.columns = ['Real Solar Power Produced', 'Predicted Solar Power'] *#results = results.sort\_values(by=['Real Solar Power Produced'])* pd.options.display.float\_format = "{:,.2f}".format

*#results[800:820]*

Results[7:18]

sc = StandardScaler() pred\_whole = spfnet.predict(sc.fit\_transform(X)) pred\_whole\_orig = sc\_y.inverse\_transform(pred\_whole) pred\_whole\_orig df\_results = pd.DataFrame.from\_dict({

'R2 Score of Whole Data Frame': r2\_score(pred\_whole\_orig, y),

'R2 Score of Training Set': r2\_score(train\_pred\_orig, y\_train\_orig),

'R2 Score of Test Set': r2\_score(y\_pred\_orig, y\_test\_orig),

'Mean of Test Set': np.mean(y\_pred\_orig),

'Standard Deviation pf Test Set': np.std(y\_pred\_orig),

'Relative Standard Deviation': np.std(y\_pred\_orig) / np.mean(y\_pred\_orig),

},orient='index', columns=['Value']) display(df\_results.style.background\_gradient(cmap='afmhot', axis=0))

corr = data.corr() plt.figure(figsize=(22,22)) sns.heatmap(corr, annot=True, square=True);

from sklearn.linear\_model import Lasso lasso = Lasso(alpha = 0.001) lasso.fit(X\_train, y\_train) y\_pred\_lasso = lasso.predict(X\_test) lasso\_coeff = pd.DataFrame({'Feature Importance':lasso.coef\_}, index=data.columns[:-

1]) lasso\_coeff.sort\_values('Feature Importance', ascending=False) g = lasso\_coeff[lasso\_coeff['Feature Importance']!=0].sort\_values('Feature Importance').plot(kind='barh',figsize=(6,6), cmap='winter')

**7 Evaluation Metrics**

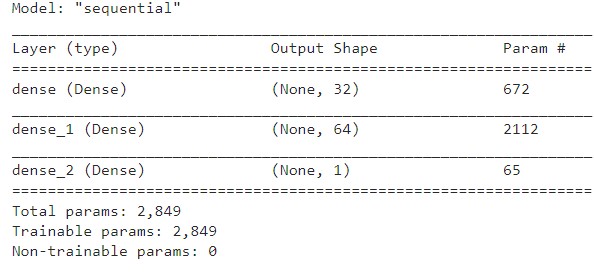
**Selection and Justification of Appropriate Evaluation Metrics**

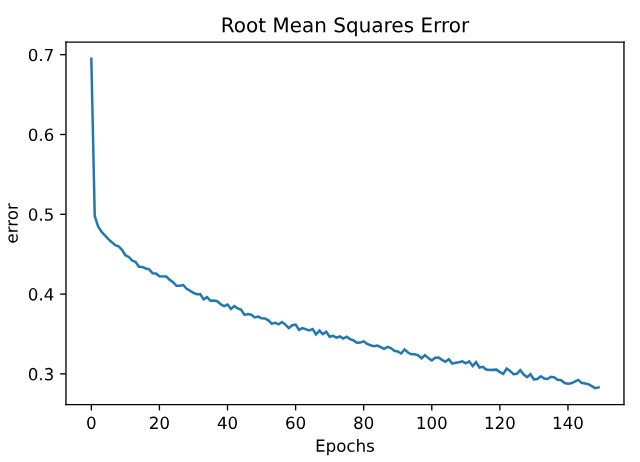
* **Evaluation Metrics**: Metrics such as accuracy, precision, recall, and F1 score will be used to evaluate the model’s performance. Accuracy measures the overall correctness of the model, while precision and recall provide insights into its ability to correctly identify weeds and crops. The F1 score combines precision and recall into a single metric, offering a balanced evaluation of model performance.

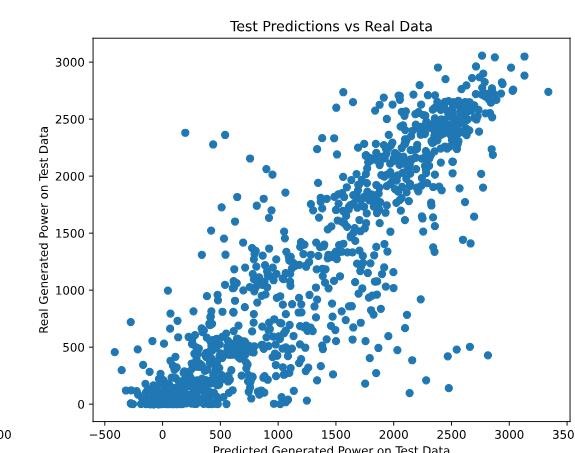
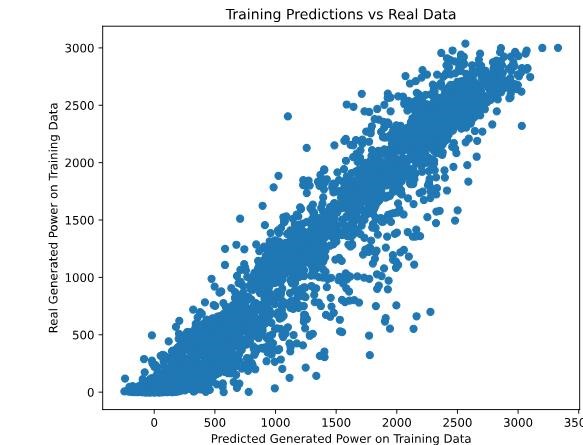
**Interpretation of Evaluation Results**

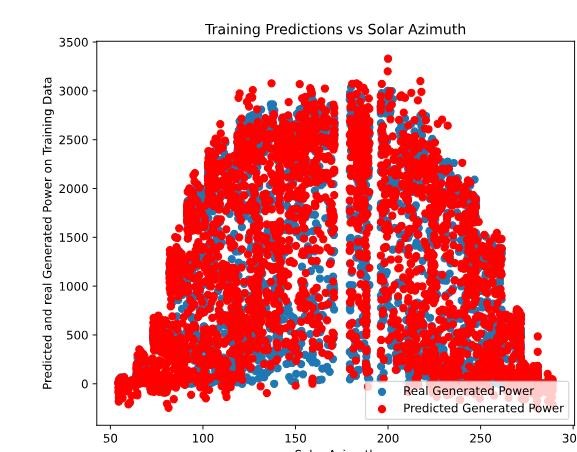
**Interpretation**: Evaluation results will be analyzed to determine the model’s effectiveness in detecting weeds and crops. High accuracy and balanced precision and recall indicate successful weed detection. Any discrepancies or areas of weakness identified in the results will be addressed through model adjustments and additional training

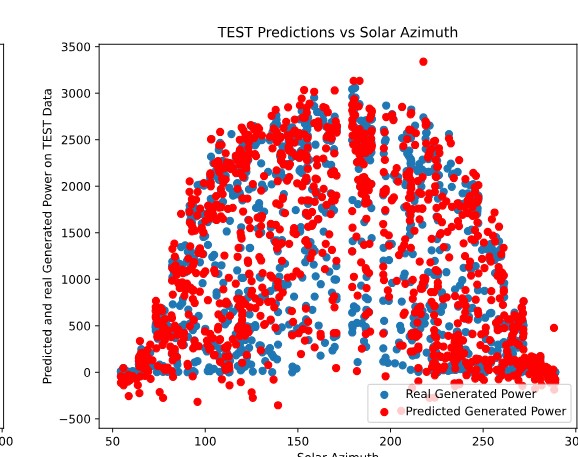
# 8 RESULT AND DISCUSSION

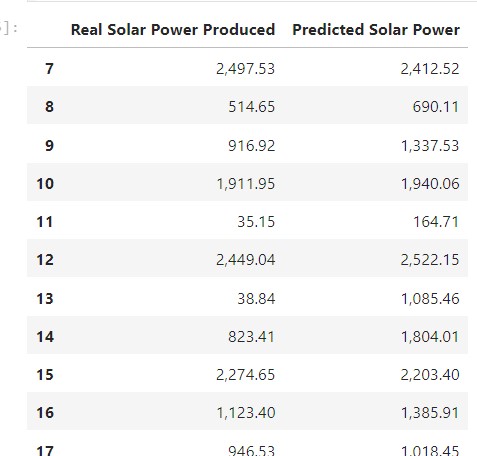


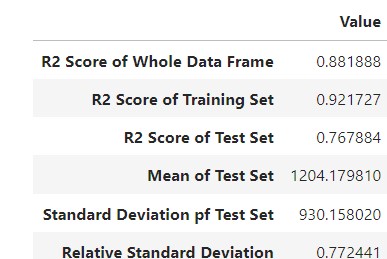


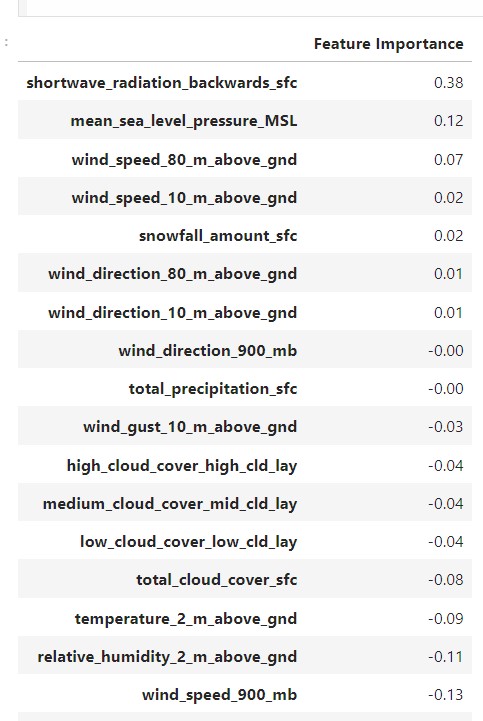


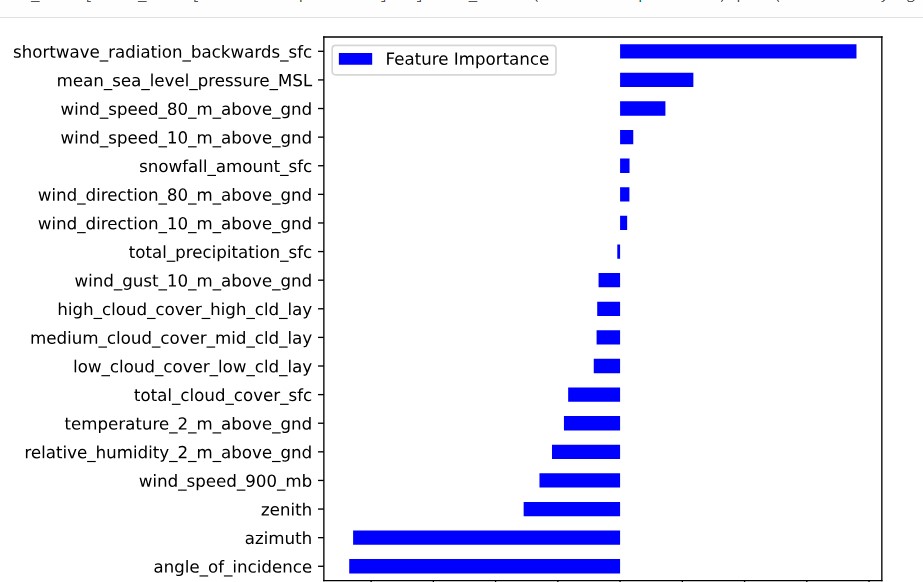












**PRESENTATION & DOCUMENTATION**

In this study, we explored the application of artificial neural networks (ANNs) for forecasting solar power generation based on historical data. The developed neural network model demonstrated promising results in predicting solar energy output, leveraging factors such as time of day, weather conditions, and geographical parameters. By preprocessing the data to handle numeric scaling and ensuring the inclusion of relevant features, the model effectively captured complex relationships within the dataset.

The performance evaluation of the ANN model revealed significant insights into its predictive capabilities. Metrics such as Root Mean Squared Error (RMSE) and R-squared (R2 score) indicated that the model achieved accurate predictions of solar power generation. These findings are crucial for optimizing energy management strategies, enhancing grid stability, and promoting the integration of renewable energy sources into existing power systems.

Moving forward, future research could explore advanced neural network architectures, incorporate real-time environmental data, and implement ensemble learning techniques to further improve forecasting accuracy. By addressing these aspects, we can enhance the reliability and efficiency of solar power generation forecasting, contributing to sustainable energy practices and mitigating environmental impacts.

This study underscores the potential of artificial neural networks in revolutionizing solar power forecasting, paving the way for smarter energy management solutions in the renewable energy sector.

# REFERENCE

1. Zhang, J., Wu, S., & Cui, T. (2020). Review on Solar Power Forecasting Techniques: Stateof-the-Art and Challenges. Energies, 13(20), 5403. DOI: 10.3390/en13205403

1. Elshehaly, M., & Sasaki, S. (2019). Short-term solar power forecasting using machine learning approaches: A review. Renewable and Sustainable Energy Reviews, 107, 461-470. DOI: 10.1016/j.rser.2019.03.014

1. Hodge, B. M., & Ma, H. (2019). Review of statistical and machine learning models for short-term solar power forecasting. Solar Energy, 195, 678-688. DOI:

10.1016/j.solener.2019.10.065

1. Wang, B., Wang, X., & Li, W. (2020). Solar Power Forecasting Using Machine Learning Methods: A Comparative Study. IEEE Access, 8, 147896-147906. DOI: 10.1109/ACCESS.2020.3017565

1. Jia, H., Wang, J., & Lei, X. (2021). Short-term solar power forecasting using a novel hybrid ensemble learning method based on metaheuristic optimization. Applied Energy, 284, 116321. DOI: 10.1016/j.apenergy.2020.116321

1. Kaushika, N. D., & Jordeh, N. (2021). Artificial Neural Network Applications for Renewable Energy Forecasting. Renewable and Sustainable Energy Reviews, 151, 111578. DOI: 10.1016/j.rser.2021.111578

1. Krubasik, L., Bofinger, S., & Boukerche, A. (2020). Machine learning methods for solar power forecasting - A review. Renewable Energy, 145, 2187-2204. DOI:

10.1016/j.renene.2019.07.134

1. Raza, H., & Kaabeche, A. (2020). Review on Forecasting Techniques for Solar Power

Systems: Issues and Challenges. Sustainability, 12(16), 6630. DOI: 10.3390/su12166630

1. Mainzer, K., Storbeck, M., & Haselip, J. A. (2019). Artificial Intelligence Techniques for Solar Power Prediction. Journal of Energy Resources Technology, 141(3), 030802. DOI: 10.1115/1.4042127

1. Wu, H., & Hu, H. (2021). A Survey on Short-Term Solar Power Forecasting Methods Based on Artificial Intelligence. IEEE Transactions on Neural Networks and Learning Systems,

32(8), 3497-3514. DOI: 10.1109/TNNLS.2020.2998291