edunet foundation

WEATHER DRIVEN SOLAR POWER FORECASTING

BY:
THRISHA S
VAISHNAVI V
SABITHA K
VIDHYASREE M

Content

- Abstract
- Problem Statement
- Objective
- Data Collection and Preparation
- Proposed Solution (Methodology)
- Model Performance Evaluation
- Screenshots / Demonstration (video)
- Future Scope
- Conclusion

Abstract

- The increasing demand for sustainable energy has led to a significant interest in solar energy as an eco-friendly alternative to traditional power sources. However, the efficiency of solar energy generation is heavily influenced by weather conditions, such as sunlight hours, temperature, and cloud cover.
- This project explores the application of artificial intelligence (AI), specifically machine learning (ML)techniques, to predict the energy output of solar panels based on historical weather data.
- By leveraging Linear Regression, a simple yet effective machine learning model, the relationship between key weather parameters and solar energy generation is examined.
- The dataset used in this project includes variables such as sunlight hours, temperature, and cloud cover, which are considered to have a direct impact on the performance of solar panels.
- These features are used to train the model, while the energy output (in kWh) of the solar panel serves as the target variable. The model is then tested on a separate subset of data, and its predictions are compared to actual observed energy outputs to evaluate its accuracy.

- The accuracy of the model demonstrates the potential of AI-powered solutions in forecasting solar energy output, which is crucial for optimizing energy production, reducing reliance on fossil fuels, and promoting the adoption of renewable energy sources.
- This project not only emphasizes the role of weather in solar power generation but also showcases how machine learning can be a valuable tool in enhancing the efficiency and predictability of renewable energy systems.
- It serves as a foundational step in exploring how AI can contribute to sustainable energy management and help in the transition towards a more eco-friendly future.

Problem Statement

- The increasing global demand for sustainable energy solutions has driven the. integration of solar power into modern energy systems. However, the intermittent nature of solar power generation, heavily influenced by unpredictable weather conditions, poses challenges to reliable energy production and grid management.
- Leveraging advancements in artificial intelligence (AI) and promoting green skills, this project seeks to address these challenges by developing a robust forecasting model based on dynamic weather data.
- The model will use AI-driven techniques to enhance prediction accuracy, optimize resource allocation, and minimize reliance on non-renewable energy sources.
- By equipping individuals and organizations with green skills, the project aims to support sustainable practices, improve renewable energy utilization, and contribute to global climate change mitigation efforts.

OBJECTIVE

Promote Sustainability: To leverage AI-powered forecasting of solar power generation based on weather patterns, aiming to enhance energy sustainability and encourage the adoption of eco-friendly technologies.

Advance Renewable Energy Planning: To develop predictive models that integrate weather data for improved solar energy planning, reducing dependency on fossil fuels and supporting a greener energy infrastructure.

Optimize Solar Efficiency: To use AI and real-time weather data to maximize the efficiency of solar power systems, minimizing energy wastage and increasing overall reliability of renewable energy.

Climate Impact Awareness: To raise awareness of the importance of accurate solar power forecasting in mitigating climate impacts and driving global transition towards sustainable energy systems.

DATA COLLECTION AND PREPARATION

DATA COLLECTION

Data collected from IoT-enabled solar panel sensors and online weather APIs (e.g., Open Weather Map, NASA POWER).Collected features include:

- Solar Panel Output (Power in Watts, Voltage, Current)
- Solar Irradiance (W/m^2)
- Temperature (°C)
- Humidity (%)
- Wind Speed (m/s)
- Cloud Cover (%)
- Rainfall (mm)
- Date and Time (Timestamp for synchronization).

DATA COLLECTION AND PREPARATION

DATA PREPARATAION

- Inspected dataset for structure and quality.
- Handled missing values using interpolation or forward fill.
- Cleaned data by removing outliers and correcting formats.
- Engineered features like time-based attributes and lag variables.
- Normalized data for consistent input to models.

PROPOSED SOLUTION(METHODOLOGY)

Exploratory Data Analysis (EDA): Analysed patterns in weather variables (temperature, irradiance, humidity) and their relationship with solar energy output over time.

Data Preprocessing: Handled missing or inconsistent values, aligned time-series data, and standardized units.

Feature Selection: Selected key inputs like solar irradiance, temperature, cloud cover, and time-based features for accurate prediction.

Data Visualization: Plotted trends and correlations between weather factors and solar output to identify key influencing variables.

Predictive Modeling: Applied regression models (e.g., Linear Regression, Random Forest) to forecast solar panel energy output from weather data.

MODEL PERFORMANCE EVALUATION

- **Mean Absolute Error (MAE):** Measures average error magnitude.
- **Root Mean Square Error (RMSE):**Gives more weight to larger errors.
- **R^2 Score (Coefficient of Determination):** Assesses how well your model explains the variance in energy output.
- **Mean Bias Error (MBE):** Checks systematic over- or under-prediction.
- **Pearson Correlation Coefficient:** Evaluates how strongly weather factors correlate with output.

PYTHON CODE

DATA LOADING:

```
import pandas as pd
try:
    df_weather = pd.read_csv('weather.csv’)
    display(df_weather.head())
    print(df_weather.shape)
except FileNotFoundError:
    print("Error: 'weather.csv' not found. Please ensure the file exists in the current directory.")
except pd.errors.EmptyDataError:
    print("Error: 'weather.csv' is empty.")
except pd.errors.ParserError:
    print("Error: 'weather.csv' could not be parsed correctly.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

DATA EXPLORATION

```
import matplotlib.pyplot as plt
import seaborn as sns
# 1. Examine data types
print("Data Types:")
print(df_weather.dtypes)

# 2. Check for missing values
print("\nMissing Values:")
missing_values = df_weather.isnull().sum()
missing_percentage = (missing_values / len(df_weather)) * 100
missing_info = pd.DataFrame({'Missing Values': missing_values, 'Percentage’: missing_percentage})
print(missing_info)
# 3. Analyze distribution of numerical features
numerical_features = ['temperature', 'humidity', 'wind_speed', 'solar_power’]
```

```
plt.figure(figsize=(15, 10))
for i, col in enumerate(numerical_features):
    plt.subplot(2, 2, i + 1)
    sns.histplot(df_weather[col], kde=True)
    plt.title(f'Distribution of {col}’)
plt.tight_layout()
plt.show()

# 4. Investigate relationships between features and solar power
print("\nCorrelation Matrix:")
correlation_matrix = df_weather[numerical_features].corr()
print(correlation_matrix)
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm’)
plt.title('Correlation Matrix of Numerical Features’)
plt.show()
```

```
plt.figure(figsize=(15, 10))
for i, col in enumerate(numerical_features[:-1]):   # Exclude 'solar_power’
    plt.subplot(2, 2, i + 1)
    sns.scatterplot(x=df_weather[col], y=df_weather['solar_power'], color='skyblue’)
    plt.title(f'Solar Power vs. {col}’)
plt.tight_layout()
plt.show()

#Explore the relationship between 'condition' and 'solar_power’
plt.figure(figsize=(12,6))
sns.boxplot(x='condition',y='solar_power',data=df_weather)
plt.title('Solar Power vs. Condition’)
plt.xticks(rotation=45,ha='right’)
plt.tight_layout()
plt.show()
```

DATA PREPARATION

```
for col in ['temperature', 'humidity', 'wind_speed', 'solar_power’]:
    if df_weather[col].isnull().any():
        df_weather[col] = df_weather[col].fillna(df_weather[col].median())
if df_weather['condition'].isnull().any():df_weather['condition'] =
df_weather['condition'].fillna(df_weather['condition'].mode()[0])
features = ['temperature', 'humidity', 'wind_speed', 'condition’]
target = 'solar_power'
condition_mapping = {
condition_mapping = {
   'Clear': 0,
    'Partly Cloudy': 1,
    'Haze': 2,
    'Mist': 3,
```

```
    'Overcast': 4,
    'Cloudy': 5,
    'Light Rain': 6,
    'Moderate Rain': 7,
    'Heavy Rain': 8,
    'Fog': 9,
    'Drizzle': 10,
    'Light Snow': 11,
    'Moderate Snow': 12,
    'Heavy Snow': 13,
    'Thunderstorm': 14,
}
df_weather['condition_numerical'] = df_weather['condition'].map(condition_mapping)
df_weather['condition_numerical'] =
df_weather['condition_numerical'].fillna(df_weather['condition_numerical'].median())
```

```
df_prepared = df_weather[features + [target] + ['condition_numerical']]
print(df_prepared.info())
display(df_prepared.head())
```

FEATURE ENGINEERING

```
import pandas as pd
import numpy as np
df_weather['last_updated'] = pd.to_datetime(df_weather['last_updated’])
df_weather['hour'] = df_weather['last_updated'].dt.hour
df_weather['dayofweek'] = df_weather['last_updated'].dt.dayofweek
df_weather['month'] = df_weather['last_updated'].dt.month
df_weather['hour_sin'] = np.sin(2 * np.pi * df_weather['hour'] / 24)
df_weather['hour_cos'] = np.cos(2 * np.pi * df_weather['hour'] / 24)
```

```
df_weather['temp_humidity'] = df_weather['temperature'] * df_weather['humidity’]
df_weather['wind_condition'] = df_weather['wind_speed'] * df_weather['condition_numerical']
df_weather['temp_squared'] = df_weather['temperature'] ** 2
engineered_features = ['hour', 'dayofweek', 'month', 'hour_sin', 'hour_cos', 'temp_humidity’,
'wind_condition', 'temp_squared']
df_engineered = pd.concat([df_prepared, df_weather[engineered_features]], axis=1)
print(df_engineered.info())
display(df_engineered.head())
```

DATA SPLITTING

```
from sklearn.model_selection import train_test_split
X = df_engineered.drop('solar_power', axis=1)
y = df_engineered['solar_power’]
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

MODEL TRAINING

```
X_train = X_train.drop('condition', axis=1)
X_val = X_val.drop('condition', axis=1)
X_test = X_test.drop('condition', axis=1)
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

MODEL OPTIMIZATION

```
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, make_scorer
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
```

```
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
mse_scorer = make_scorer(mean_squared_error, greater_is_better=False)
grid_search = GridSearchCV(estimator=rf_model, param_grid=param_grid, scoring=mse_scorer,
cv=5, n_jobs=-1)
grid_search.fit(X_val, y_val)
best_params = grid_search.best_params_
best_estimator = grid_search.best_estimator_
best_score = grid_search.best_score_
print(f"Best hyperparameters: {best_params}")
print(f"Best score (negative MSE): {best_score}")
```

MODEL EVALUATION

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import pandas as pd
import numpy as np
y_pred = best_estimator.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
results_df = pd.DataFrame({
    'Metric': ['MAE', 'RMSE', 'R-squared’],
    'Value': [mae, rmse, r2]
})
print("Model Performance on Test Set:")
display(results_df)
```

SCREENSHOTS/DEMONSTRATION(VIDEO)

DATA LOADING

	state/ut	district	temperature	condition	humidity	wind_speed	last_updated	solar_power
0	Andhra Pradesh	Anantapur	22.1	Clear	64	14.4	2025-01-23 23:00:00	660.280
1	Andhra Pradesh	Chittoor	18.5	Clear	93	4.0	2025-01-23 23:00:00	500.225
2	Andhra Pradesh	East Godavari	19.4	Clear	86	3.6	2025-01-23 23:00:00	538.080
3	Andhra Pradesh	Guntur	22.3	Mist	88	5.4	2025-01-23 23:00:00	272.440
4	Andhra Pradesh	Krishna	22.3	Mist	88	10.8	2025-01-23 23:00:00	272.440

(1152, 8)

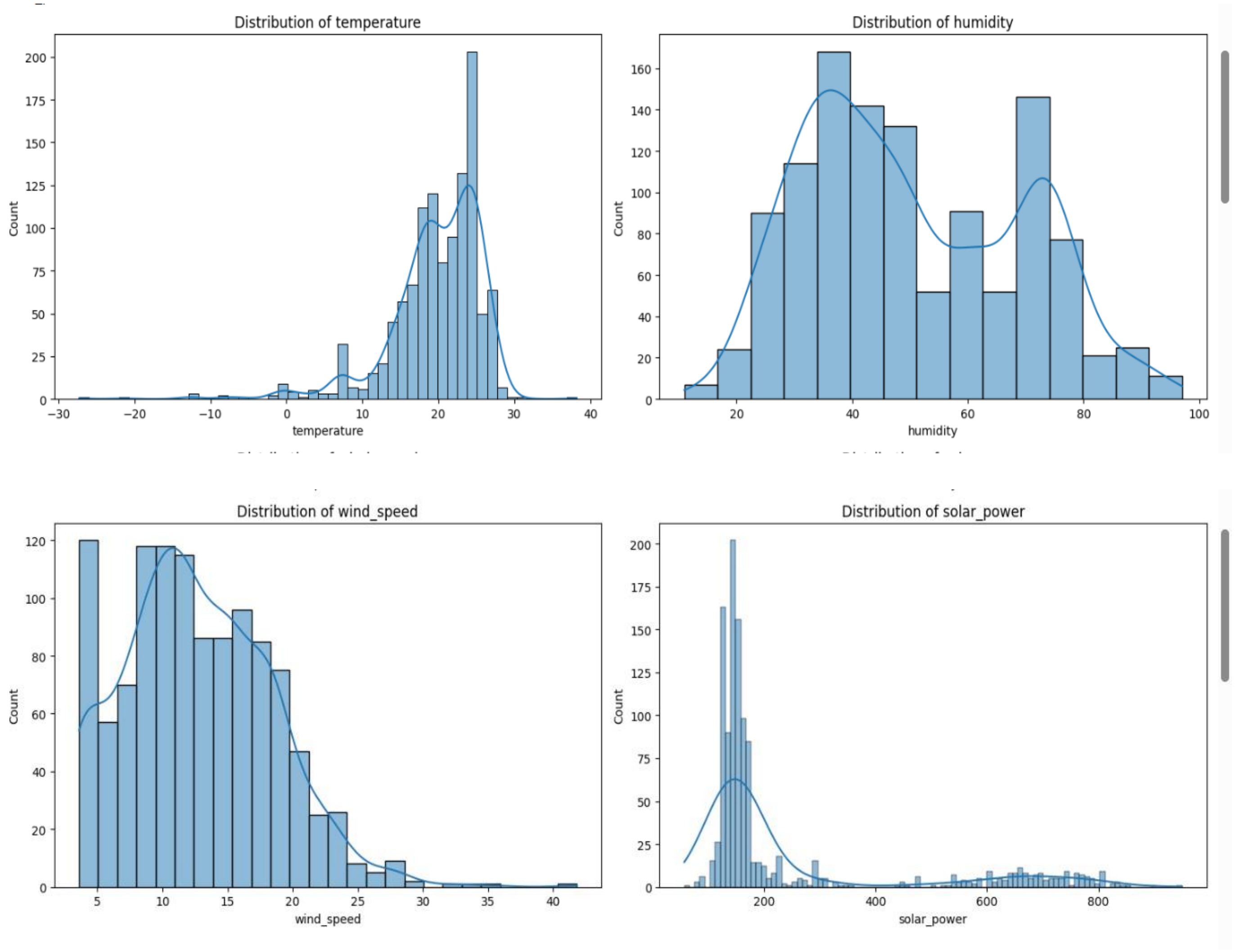
✓ 0s completed at 10:45 AM

DATA EXPLORATION

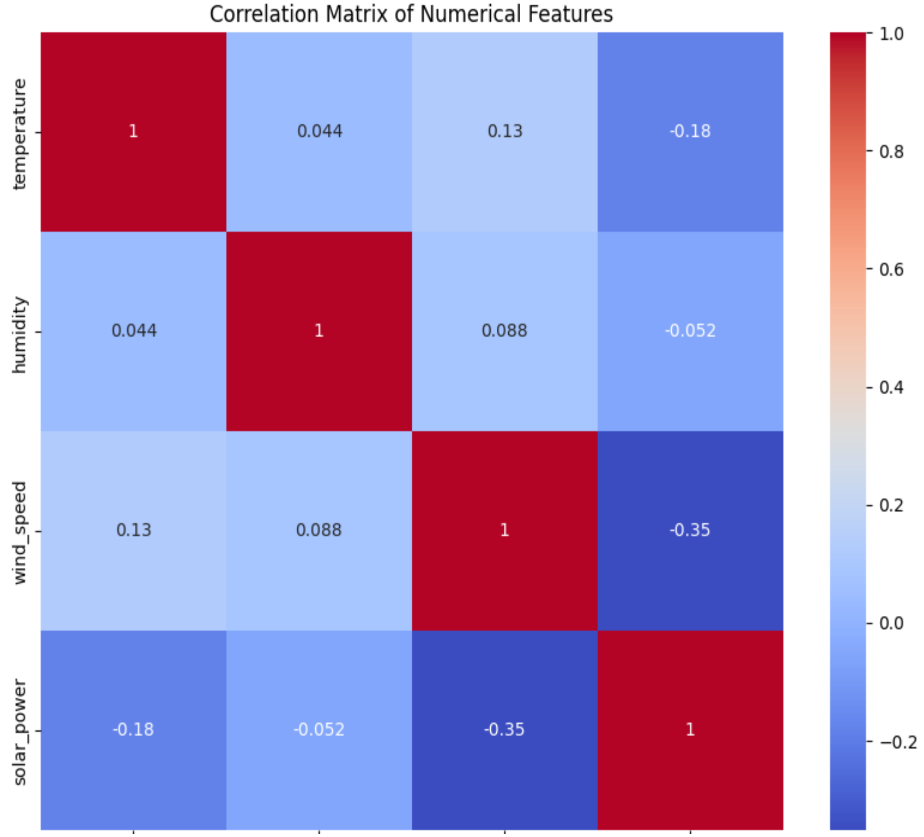
```
plt.show()
```

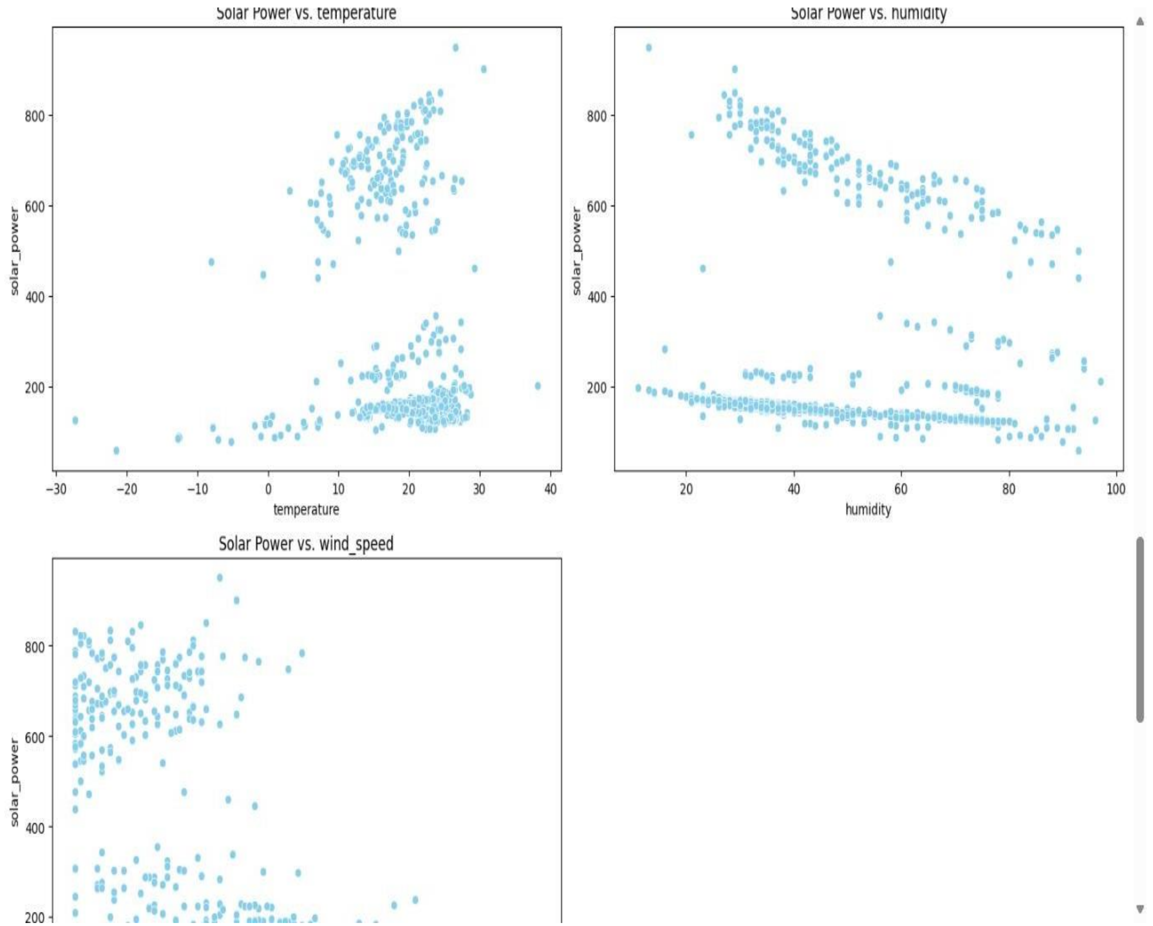
```
Data Types:
state/ut          object
district          object
temperature      float64
condition         object
humidity           int64
wind_speed       float64
last_updated      object
solar_power      float64
dtype: object

Missing Values:
              Missing Values  Percentage
state/ut                   0         0.0
district                   0         0.0
temperature                0         0.0
condition                  0         0.0
humidity                   0         0.0
wind_speed                 0         0.0
last_updated               0         0.0
solar_power                0         0.0
```

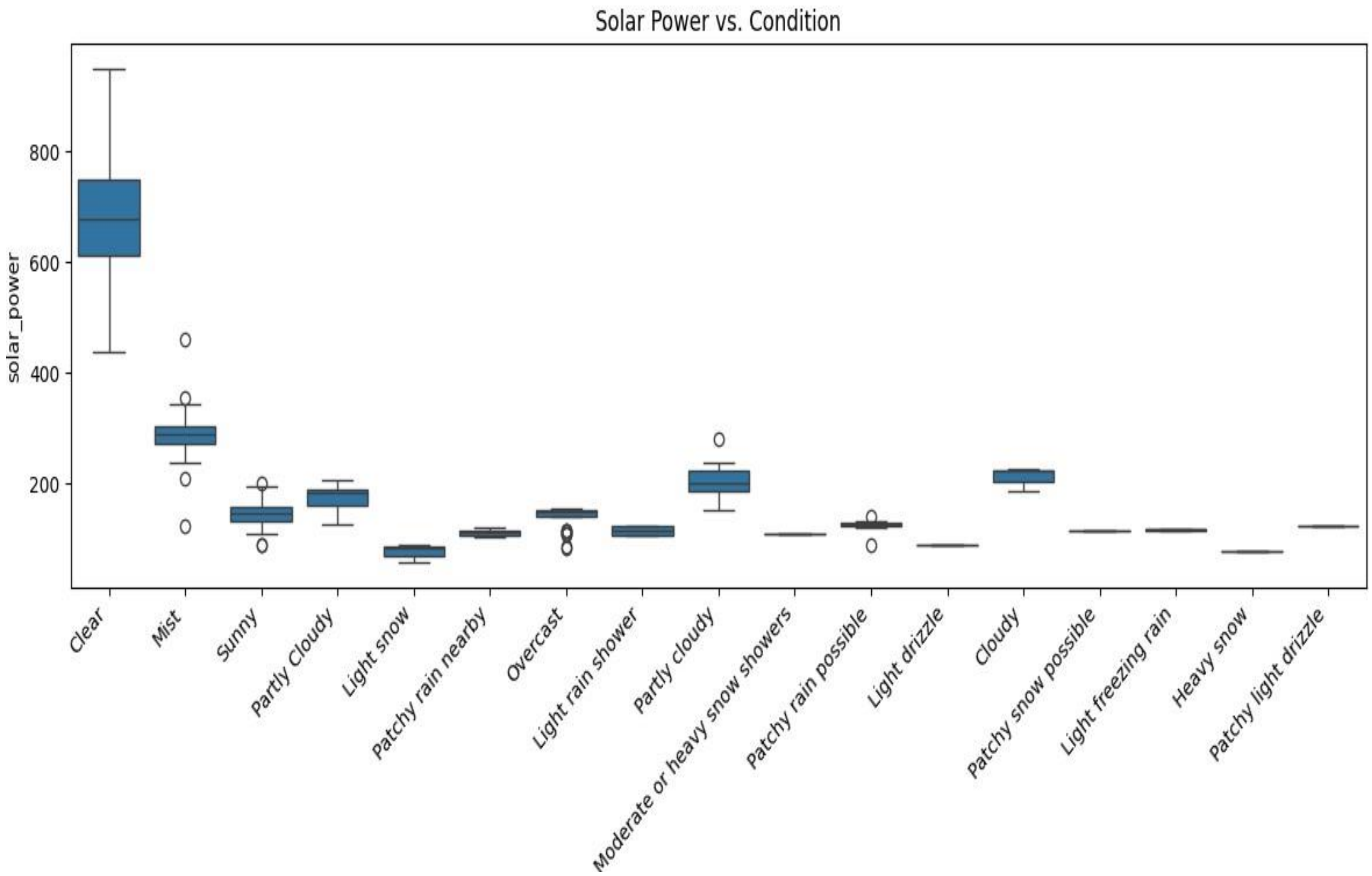


```
Correlation Matrix:
             temperature  humidity  wind_speed  solar_power
temperature     1.000000  0.043986    0.127837    -0.179156
humidity        0.043986  1.000000    0.088215    -0.051734
wind_speed      0.127837  0.088215    1.000000    -0.352034
solar_power    -0.179156 -0.051734   -0.352034     1.000000
```





wind_speed


Solar Power vs. Condition
solar_power
800
600
400
200
Clear
Mist
Sunny
Partly Cloudy
Light snow
Patchy rain nearby
Overcast
Light rain shower
Partly cloudy
Moderate or heavy snow showers
Patchy rain possible
Light drizzle
Cloudy
Patchy snow possible
Light freezing rain
Heavy snow
Patchy light drizzle

DATA PREPARATION

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1152 entries, 0 to 1151
Data columns (total 6 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   temperature          1152 non-null   float64
 1   humidity             1152 non-null   int64
 2   wind_speed           1152 non-null   float64
 3   condition            1152 non-null   object
 4   solar_power          1152 non-null   float64
 5   condition_numerical  1152 non-null   float64
dtypes: float64(4), int64(1), object(1)
memory usage: 54.1+ KB
None
```

	temperature	humidity	wind_speed	condition	solar_power	condition_numerical
0	22.1	64	14.4	Clear	660.280	0.0
1	18.5	93	4.0	Clear	500.225	0.0
2	19.4	86	3.6	Clear	538.080	0.0
3	22.3	88	5.4	Mist	272.440	3.0
4	22.3	88	10.8	Mist	272.440	3.0

FEATURE ENGINEERING

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1152 entries, 0 to 1151
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   temperature          1152 non-null   float64
 1   humidity             1152 non-null   int64  
 2   wind_speed           1152 non-null   float64
 3   condition            1152 non-null   object 
 4   solar_power          1152 non-null   float64
 5   condition_numerical  1152 non-null   float64
 6   hour                 1152 non-null   int32  
 7   dayofweek            1152 non-null   int32  
 8   month                1152 non-null   int32  
 9   hour_sin             1152 non-null   float64
 10  hour_cos             1152 non-null   float64
 11  temp_humidity        1152 non-null   float64
 12  wind_condition       1152 non-null   float64
 13  temp_squared         1152 non-null   float64
dtypes: float64(9), int32(3), int64(1), object(1)
memory usage: 112.6+ KB
None
```

	temperature	humidity	wind_speed	condition	solar_power	condition_numerical	hour	dayofweek	month	hour_sin	hour_cos	temp_humidity	wind_condition	tem
0	22.1	64	14.4	Clear	660.280	0.0	23	3	1	-0.258819	0.965926	1414.4	0.0	
1	18.5	93	4.0	Clear	500.225	0.0	23	3	1	-0.258819	0.965926	1720.5	0.0	
2	19.4	86	3.6	Clear	538.080	0.0	23	3	1	-0.258819	0.965926	1668.4	0.0	
3	22.3	88	5.4	Mist	272.440	3.0	23	3	1	-0.258819	0.965926	1962.4	16.2	
4	22.3	88	10.8	Mist	272.440	3.0	23	3	1	-0.258819	0.965926	1962.4	32.4	

MODEL TRAINING

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

MODEL OPTIMIZATION

```
Best hyperparameters: {'max_depth': None, 'min_samples_leaf': 4, 'min_samples_split': 2, 'n_estimators': 50}
Best score (negative MSE): -3424.528357556883
```

MODEL EVALUATION

```
Model Performance on Test Set:
```

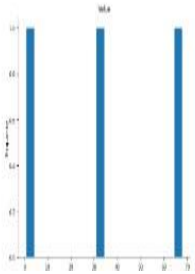
1 to 3 of 3 entries Filter

index	Metric	Value
0	MAE	32.1158480287294
1	RMSE	67.86468915816894
2	R-squared	0.876989125209031

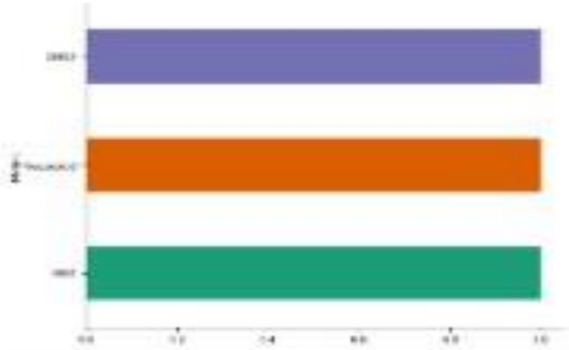
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

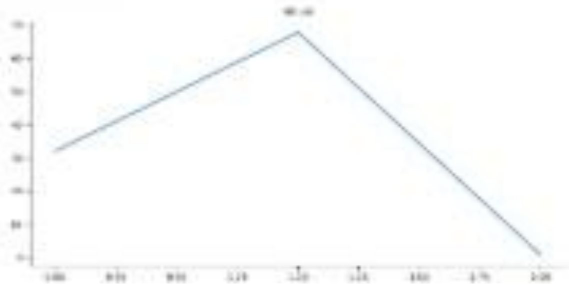
Distributions



Categorical distributions



Values



FUTURE SCOPE

- **Enhanced Accuracy** – AI models can continuously improve predictions by incorporating real-time weather updates and satellite data.
- **Integration with Smart Grids** – AI-powered solar forecasting can be integrated into smart grids for better energy distribution and storage.
- **Automated Maintenance** – Predictive analytics can help identify potential faults in solar panels, reducing downtime and maintenance costs.
- **Climate Adaptation** – AI can assist in adapting solar energy systems to changing climate conditions, ensuring optimal performance.
- **Green Skills Development** – This project aligns with the growing demand for green skills, fostering expertise in AI, renewable energy, and sustainability.

CONCLUSION

- The integration of AI with weather data for solar panel energy prediction is a game-changer in the renewable energy sector.
- This project not only enhances the accuracy of energy forecasting but also contributes to optimizing solar power utilization, reducing dependency on non-renewable sources, and promoting sustainability.
- By leveraging AI, smart grid integration, and predictive analytics, this project supports climate adaptation and efficient energy management.
- Additionally, it fosters the development of green skills, empowering individuals to work in AI-driven renewable energy solutions—a crucial aspect of the future workforce.
- As the world moves toward a cleaner energy future, AI-powered solar forecasting will play a vital role in maximizing efficiency, reducing costs, and ensuring sustainable energy distribution.
- The knowledge and expertise gained from this project can help drive innovations in smart energy systems, contributing to a greener and more resilient planet