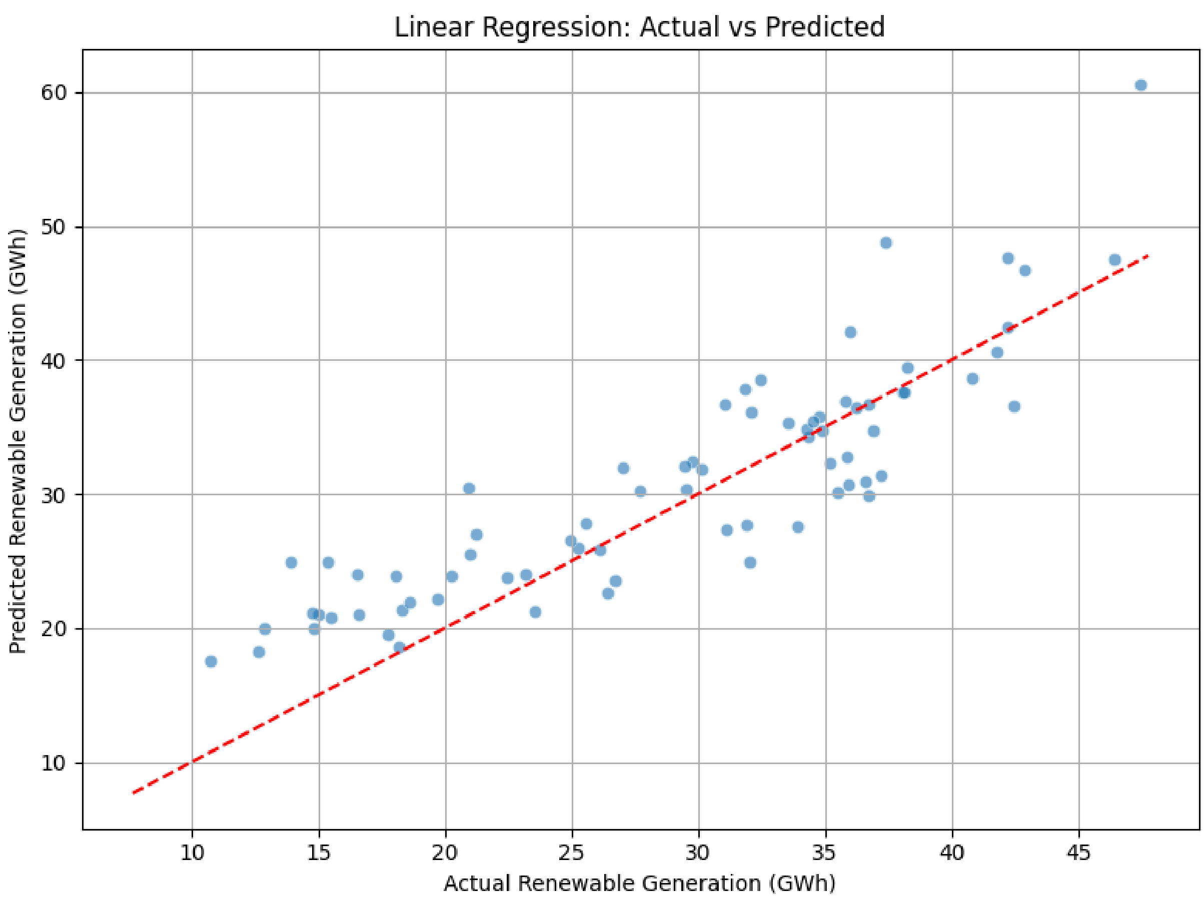
In [142…
```python
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
from sklearn.ensemble import RandomForestRegressor
import numpy as np
import itertools
```

In [146…
```python
# Load dataset
df = pd.read_excel("./datasets/datasets.xlsx", sheet_name="Sheet1")

# Define features and target
features = [
    'Tempmax_C', 'Tempmin_C', 'windspeedmax', 'windspeedmean',
    'solarradiation', 'uvindex', 'cloudcover', 'humidity', 'precip'
]
target = '(Combined) Renewable_Generation_GWh'

X = df[features]
y = df[target]

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# Create and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on test set
y_pred = model.predict(X_test)

# Evaluate the model
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse:.2f}")
print(f"R² Score: {r2:.2f}")

# Display feature coefficients
coefficients = pd.DataFrame({
    'Feature': features,
    'Coefficient': model.coef_
}).sort_values(by='Coefficient', key=abs, ascending=False)
```

```python
print(coefficients)

# Visualize actual vs predicted values
plt.figure(figsize=(8,6))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.6)
plt.xlabel('Actual Renewable Generation (GWh)')
plt.ylabel('Predicted Renewable Generation (GWh)')
plt.title('Linear Regression: Actual vs Predicted')
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
RMSE: 4.83
R² Score: 0.73
           Feature  Coefficient
3    windspeedmean     6.223594
1        Tempmin_C    -2.233962
0        Tempmax_C     1.928870
7         humidity    -1.809670
4  solarradiation     1.732282
5          uvindex    -1.373584
2     windspeedmax     1.072046
6       cloudcover    -0.839232
8           precip     0.467326
```

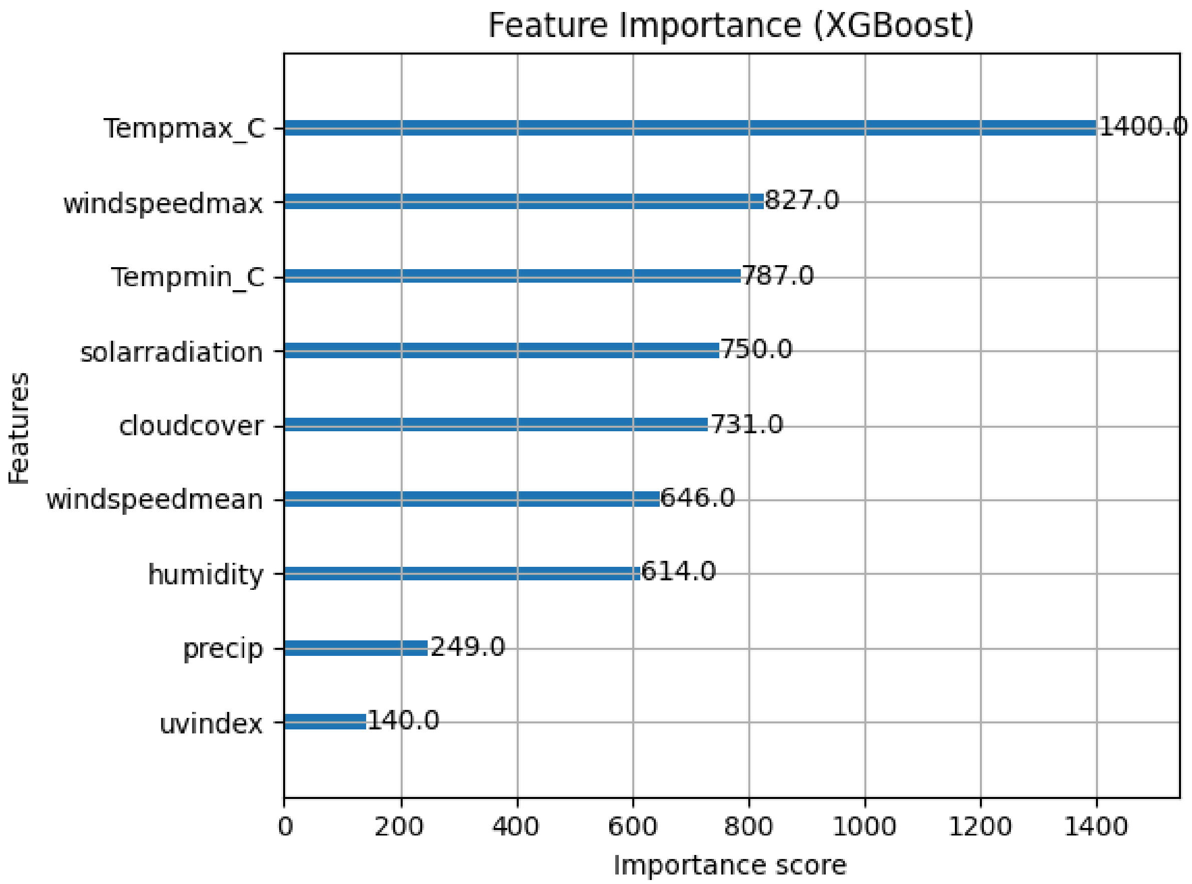


In [141…

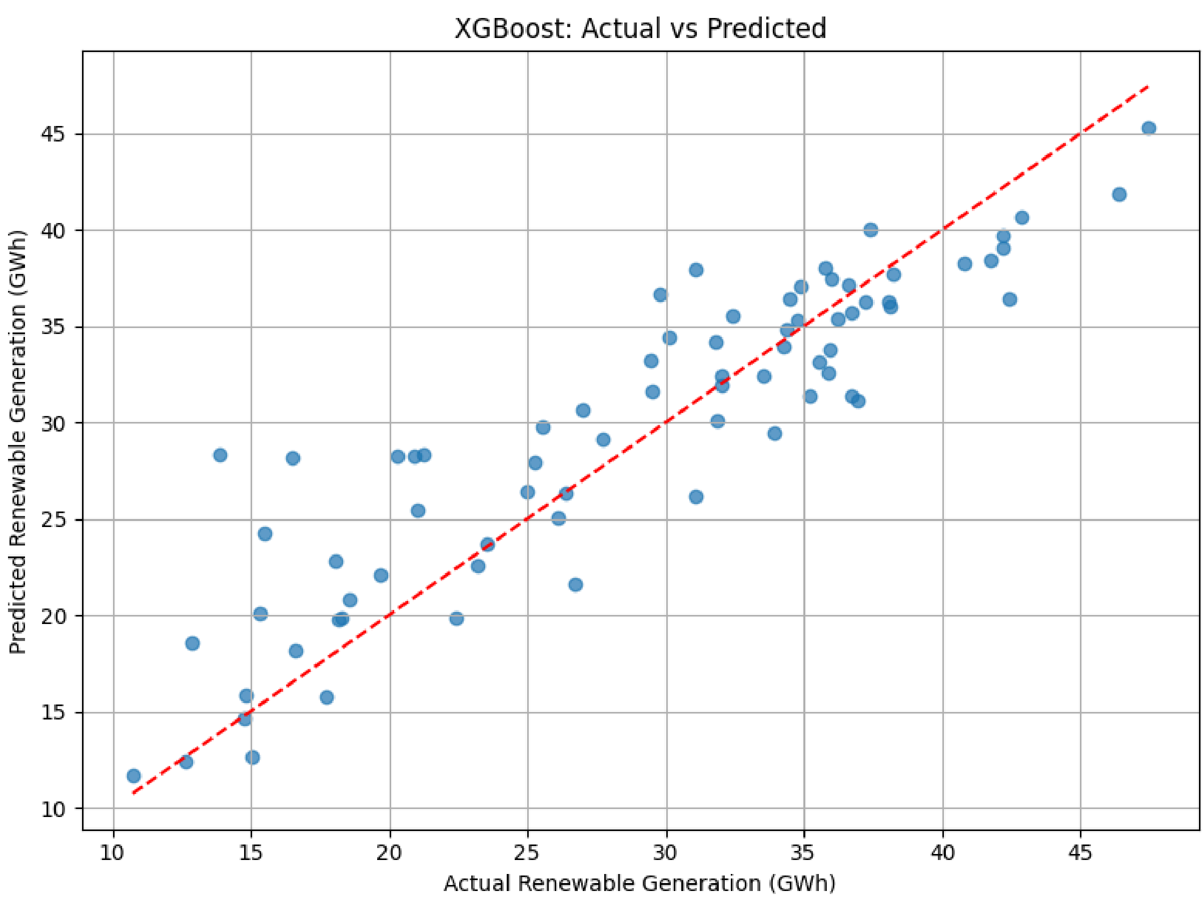
```python
# Select features and target variable
features = [
    'Tempmax_C', 'Tempmin_C', 'windspeedmax', 'windspeedmean',
```

```
    'solarradiation', 'uvindex', 'cloudcover', 'humidity', 'precip'
]
X = df[features]
y = df['(Combined) Solar_GWh'] + df['Wind_GWh']  # total renewable generatio

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Initialize and train XGBoost model
model = xgb.XGBRegressor(
    n_estimators=200,
    eta=0.05,
    max_depth=6,
    random_state=42,
    gamma=0,
    max_delta_step=0,
    alpha=0,
    colsample_bytree=1.0,
    subsample=0.6
)

model.fit(X_train, y_train)

# Make predictions and evaluate the model
y_pred = model.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse:.3f}")
print(f"R² Score: {r2:.3f}")

# Plot feature importance
xgb.plot_importance(model)
plt.title('Feature Importance (XGBoost)')
plt.tight_layout()
plt.show()

# Plot actual vs predicted values
plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, alpha=0.7)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel('Actual Renewable Generation (GWh)')
plt.ylabel('Predicted Renewable Generation (GWh)')
plt.title('XGBoost: Actual vs Predicted')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
RMSE: 4.058
R² Score: 0.808
```



Feature Importance (XGBoost)
Tempmax_C
windspeedmax
Tempmin_C
solarradiation
cloudcover
windspeedmean
humidity
precip
uvindex
1400.0
827.0
787.0
750.0
731.0
646.0
614.0
249.0
140.0
Features
0
200
400
600
800
1000
1200
1400
Importance score



XGBoost: Actual vs Predicted
Predicted Renewable Generation (GWh)
Actual Renewable Generation (GWh)
10
15
20
25
30
35
40
45

In [133…

```
# Define features and target
features = [
    'Tempmax_C', 'Tempmin_C', 'windspeedmax', 'windspeedmean',
    'solarradiation', 'uvindex', 'cloudcover', 'humidity', 'precip'
]
X = df[features]
y = df['(Combined) Solar_GWh'] + df['Wind_GWh']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Extended parameter grid (around 2400 combinations)
param_grid = {
    'eta': [0.01, 0.02, 0.03, 0.04, 0.05],
    'max_depth': [3, 4, 5, 6],
    'gamma': [0, 2, 5, 7, 10],
    'max_delta_step': [0, 5, 10],
    'alpha': [0, 0.2, 0.4, 0.6, 0.8],
    'colsample_bytree': [0.8, 0.9, 0.96, 1.0],
    'subsample': [0.6, 0.7, 0.8, 0.9]
}

# Generate all parameter combinations
keys, values = zip(*param_grid.items())
combinations = [dict(zip(keys, v)) for v in itertools.product(*values)]

print(f"🔍 Total parameter combinations to test: {len(combinations)}")

# Search for the best parameter combination
results = []
best_rmse = float('inf')
best_params = None
best_r2 = None

for idx, params in enumerate(combinations):
    model = xgb.XGBRegressor(
        n_estimators=200,
        random_state=42,
        verbosity=0,
        **params,
        n_jobs=-1
    )
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    rmse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    results.append({
        'Index': idx + 1,
        'RMSE': rmse,
        'R2': r2,
        **params
    })
```

```
    if rmse < best_rmse:
        best_rmse = rmse
        best_params = params
        best_r2 = r2

    if (idx + 1) % 100 == 0:
        print(f"[{idx+1}/{len(combinations)}] RMSE: {rmse:.2f}, R²: {r2:.2f}

# Output the best result
print("\nBest Parameters:")
print(best_params)
print(f"Best RMSE: {best_rmse:.2f}")

# Save results to DataFrame (for visualization or export)
results_df = pd.DataFrame(results)
results_df.to_csv("xgb_grid_search_results.csv", index=False)
```

```
🔍 Total parameter combinations to test: 24000
[100/24000] RMSE: 28.60, R²: 0.67
[200/24000] RMSE: 23.61, R²: 0.73
[300/24000] RMSE: 23.10, R²: 0.73
[400/24000] RMSE: 27.94, R²: 0.67
[500/24000] RMSE: 23.16, R²: 0.73
[600/24000] RMSE: 28.01, R²: 0.67
[700/24000] RMSE: 23.60, R²: 0.73
[800/24000] RMSE: 22.90, R²: 0.73
[900/24000] RMSE: 23.55, R²: 0.73
[1000/24000] RMSE: 23.12, R²: 0.73
[1100/24000] RMSE: 28.05, R²: 0.67
[1200/24000] RMSE: 23.64, R²: 0.72
[1300/24000] RMSE: 28.35, R²: 0.67
[1400/24000] RMSE: 23.02, R²: 0.73
[1500/24000] RMSE: 22.41, R²: 0.74
[1600/24000] RMSE: 27.39, R²: 0.68
[1700/24000] RMSE: 22.55, R²: 0.74
[1800/24000] RMSE: 27.51, R²: 0.68
[1900/24000] RMSE: 22.85, R²: 0.73
[2000/24000] RMSE: 22.22, R²: 0.74
[2100/24000] RMSE: 22.97, R²: 0.73
[2200/24000] RMSE: 22.22, R²: 0.74
[2300/24000] RMSE: 27.55, R²: 0.68
[2400/24000] RMSE: 22.31, R²: 0.74
[2500/24000] RMSE: 28.45, R²: 0.67
[2600/24000] RMSE: 22.83, R²: 0.73
[2700/24000] RMSE: 22.93, R²: 0.73
[2800/24000] RMSE: 27.29, R²: 0.68
[2900/24000] RMSE: 22.89, R²: 0.73
[3000/24000] RMSE: 27.69, R²: 0.68
[3100/24000] RMSE: 22.70, R²: 0.74
[3200/24000] RMSE: 22.95, R²: 0.73
[3300/24000] RMSE: 22.79, R²: 0.73
[3400/24000] RMSE: 22.87, R²: 0.73
[3500/24000] RMSE: 27.67, R²: 0.68
[3600/24000] RMSE: 22.73, R²: 0.74
[3700/24000] RMSE: 28.69, R²: 0.67
[3800/24000] RMSE: 23.81, R²: 0.72
[3900/24000] RMSE: 23.74, R²: 0.72
[4000/24000] RMSE: 27.56, R²: 0.68
[4100/24000] RMSE: 23.14, R²: 0.73
[4200/24000] RMSE: 27.63, R²: 0.68
[4300/24000] RMSE: 23.40, R²: 0.73
[4400/24000] RMSE: 23.79, R²: 0.72
[4500/24000] RMSE: 23.04, R²: 0.73
[4600/24000] RMSE: 23.64, R²: 0.72
[4700/24000] RMSE: 27.79, R²: 0.68
[4800/24000] RMSE: 23.12, R²: 0.73
[4900/24000] RMSE: 20.70, R²: 0.76
[5000/24000] RMSE: 19.31, R²: 0.78
[5100/24000] RMSE: 19.25, R²: 0.78
[5200/24000] RMSE: 19.95, R²: 0.77
[5300/24000] RMSE: 19.52, R²: 0.77
[5400/24000] RMSE: 20.45, R²: 0.76
[5500/24000] RMSE: 19.34, R²: 0.77
```

```
[5600/24000] RMSE: 19.18, R²: 0.78
[5700/24000] RMSE: 19.37, R²: 0.77
[5800/24000] RMSE: 19.26, R²: 0.78
[5900/24000] RMSE: 20.52, R²: 0.76
[6000/24000] RMSE: 19.38, R²: 0.77
[6100/24000] RMSE: 20.29, R²: 0.76
[6200/24000] RMSE: 19.83, R²: 0.77
[6300/24000] RMSE: 19.13, R²: 0.78
[6400/24000] RMSE: 20.24, R²: 0.76
[6500/24000] RMSE: 18.79, R²: 0.78
[6600/24000] RMSE: 20.55, R²: 0.76
[6700/24000] RMSE: 19.43, R²: 0.77
[6800/24000] RMSE: 19.33, R²: 0.77
[6900/24000] RMSE: 18.91, R²: 0.78
[7000/24000] RMSE: 18.87, R²: 0.78
[7100/24000] RMSE: 20.58, R²: 0.76
[7200/24000] RMSE: 18.78, R²: 0.78
[7300/24000] RMSE: 20.30, R²: 0.76
[7400/24000] RMSE: 20.01, R²: 0.77
[7500/24000] RMSE: 19.98, R²: 0.77
[7600/24000] RMSE: 20.48, R²: 0.76
[7700/24000] RMSE: 19.97, R²: 0.77
[7800/24000] RMSE: 20.64, R²: 0.76
[7900/24000] RMSE: 19.75, R²: 0.77
[8000/24000] RMSE: 20.38, R²: 0.76
[8100/24000] RMSE: 19.51, R²: 0.77
[8200/24000] RMSE: 19.83, R²: 0.77
[8300/24000] RMSE: 20.65, R²: 0.76
[8400/24000] RMSE: 19.26, R²: 0.78
[8500/24000] RMSE: 20.86, R²: 0.76
[8600/24000] RMSE: 20.35, R²: 0.76
[8700/24000] RMSE: 20.50, R²: 0.76
[8800/24000] RMSE: 21.38, R²: 0.75
[8900/24000] RMSE: 20.04, R²: 0.77
[9000/24000] RMSE: 21.16, R²: 0.75
[9100/24000] RMSE: 19.95, R²: 0.77
[9200/24000] RMSE: 21.34, R²: 0.75
[9300/24000] RMSE: 19.44, R²: 0.77
[9400/24000] RMSE: 20.56, R²: 0.76
[9500/24000] RMSE: 21.56, R²: 0.75
[9600/24000] RMSE: 20.33, R²: 0.76
[9700/24000] RMSE: 19.32, R²: 0.78
[9800/24000] RMSE: 19.26, R²: 0.78
[9900/24000] RMSE: 18.22, R²: 0.79
[10000/24000] RMSE: 19.28, R²: 0.78
[10100/24000] RMSE: 18.48, R²: 0.78
[10200/24000] RMSE: 19.15, R²: 0.78
[10300/24000] RMSE: 19.29, R²: 0.78
[10400/24000] RMSE: 19.23, R²: 0.78
[10500/24000] RMSE: 18.92, R²: 0.78
[10600/24000] RMSE: 18.22, R²: 0.79
[10700/24000] RMSE: 19.35, R²: 0.77
[10800/24000] RMSE: 19.51, R²: 0.77
[10900/24000] RMSE: 19.40, R²: 0.77
[11000/24000] RMSE: 19.00, R²: 0.78
[11100/24000] RMSE: 18.40, R²: 0.79
```

```
[11200/24000] RMSE: 19.42, R²: 0.77
[11300/24000] RMSE: 18.72, R²: 0.78
[11400/24000] RMSE: 19.14, R²: 0.78
[11500/24000] RMSE: 18.88, R²: 0.78
[11600/24000] RMSE: 18.42, R²: 0.79
[11700/24000] RMSE: 19.30, R²: 0.78
[11800/24000] RMSE: 18.56, R²: 0.78
[11900/24000] RMSE: 19.15, R²: 0.78
[12000/24000] RMSE: 18.67, R²: 0.78
[12100/24000] RMSE: 20.19, R²: 0.76
[12200/24000] RMSE: 19.94, R²: 0.77
[12300/24000] RMSE: 19.47, R²: 0.77
[12400/24000] RMSE: 20.34, R²: 0.76
[12500/24000] RMSE: 19.96, R²: 0.77
[12600/24000] RMSE: 20.61, R²: 0.76
[12700/24000] RMSE: 20.15, R²: 0.77
[12800/24000] RMSE: 20.16, R²: 0.77
[12900/24000] RMSE: 19.97, R²: 0.77
[13000/24000] RMSE: 19.52, R²: 0.77
[13100/24000] RMSE: 21.05, R²: 0.75
[13200/24000] RMSE: 19.56, R²: 0.77
[13300/24000] RMSE: 20.77, R²: 0.76
[13400/24000] RMSE: 20.74, R²: 0.76
[13500/24000] RMSE: 20.08, R²: 0.77
[13600/24000] RMSE: 21.32, R²: 0.75
[13700/24000] RMSE: 19.09, R²: 0.78
[13800/24000] RMSE: 21.56, R²: 0.75
[13900/24000] RMSE: 20.16, R²: 0.77
[14000/24000] RMSE: 21.02, R²: 0.76
[14100/24000] RMSE: 19.08, R²: 0.78
[14200/24000] RMSE: 20.07, R²: 0.77
[14300/24000] RMSE: 21.12, R²: 0.75
[14400/24000] RMSE: 20.14, R²: 0.77
[14500/24000] RMSE: 19.07, R²: 0.78
[14600/24000] RMSE: 18.95, R²: 0.78
[14700/24000] RMSE: 18.80, R²: 0.78
[14800/24000] RMSE: 19.09, R²: 0.78
[14900/24000] RMSE: 19.89, R²: 0.77
[15000/24000] RMSE: 19.16, R²: 0.78
[15100/24000] RMSE: 18.89, R²: 0.78
[15200/24000] RMSE: 19.30, R²: 0.78
[15300/24000] RMSE: 18.96, R²: 0.78
[15400/24000] RMSE: 18.61, R²: 0.78
[15500/24000] RMSE: 19.49, R²: 0.77
[15600/24000] RMSE: 19.61, R²: 0.77
[15700/24000] RMSE: 19.59, R²: 0.77
[15800/24000] RMSE: 18.14, R²: 0.79
[15900/24000] RMSE: 18.53, R²: 0.78
[16000/24000] RMSE: 19.37, R²: 0.77
[16100/24000] RMSE: 18.65, R²: 0.78
[16200/24000] RMSE: 19.27, R²: 0.78
[16300/24000] RMSE: 18.34, R²: 0.79
[16400/24000] RMSE: 18.68, R²: 0.78
[16500/24000] RMSE: 19.11, R²: 0.78
[16600/24000] RMSE: 18.34, R²: 0.79
[16700/24000] RMSE: 19.20, R²: 0.78
```

```
[16800/24000] RMSE: 18.31, R²: 0.79
[16900/24000] RMSE: 20.43, R²: 0.76
[17000/24000] RMSE: 18.56, R²: 0.78
[17100/24000] RMSE: 18.91, R²: 0.78
[17200/24000] RMSE: 20.31, R²: 0.76
[17300/24000] RMSE: 19.32, R²: 0.78
[17400/24000] RMSE: 20.08, R²: 0.77
[17500/24000] RMSE: 18.34, R²: 0.79
[17600/24000] RMSE: 19.55, R²: 0.77
[17700/24000] RMSE: 19.34, R²: 0.77
[17800/24000] RMSE: 19.21, R²: 0.78
[17900/24000] RMSE: 20.55, R²: 0.76
[18000/24000] RMSE: 18.40, R²: 0.79
[18100/24000] RMSE: 21.17, R²: 0.75
[18200/24000] RMSE: 19.36, R²: 0.77
[18300/24000] RMSE: 19.83, R²: 0.77
[18400/24000] RMSE: 21.53, R²: 0.75
[18500/24000] RMSE: 19.65, R²: 0.77
[18600/24000] RMSE: 21.26, R²: 0.75
[18700/24000] RMSE: 19.42, R²: 0.77
[18800/24000] RMSE: 20.48, R²: 0.76
[18900/24000] RMSE: 18.91, R²: 0.78
[19000/24000] RMSE: 19.97, R²: 0.77
[19100/24000] RMSE: 21.20, R²: 0.75
[19200/24000] RMSE: 19.27, R²: 0.78
[19300/24000] RMSE: 19.89, R²: 0.77
[19400/24000] RMSE: 19.65, R²: 0.77
[19500/24000] RMSE: 18.97, R²: 0.78
[19600/24000] RMSE: 19.79, R²: 0.77
[19700/24000] RMSE: 19.33, R²: 0.77
[19800/24000] RMSE: 19.24, R²: 0.78
[19900/24000] RMSE: 19.96, R²: 0.77
[20000/24000] RMSE: 19.66, R²: 0.77
[20100/24000] RMSE: 19.90, R²: 0.77
[20200/24000] RMSE: 18.77, R²: 0.78
[20300/24000] RMSE: 19.35, R²: 0.77
[20400/24000] RMSE: 19.52, R²: 0.77
[20500/24000] RMSE: 19.57, R²: 0.77
[20600/24000] RMSE: 19.90, R²: 0.77
[20700/24000] RMSE: 18.66, R²: 0.78
[20800/24000] RMSE: 19.01, R²: 0.78
[20900/24000] RMSE: 19.70, R²: 0.77
[21000/24000] RMSE: 19.72, R²: 0.77
[21100/24000] RMSE: 18.26, R²: 0.79
[21200/24000] RMSE: 19.38, R²: 0.77
[21300/24000] RMSE: 19.22, R²: 0.78
[21400/24000] RMSE: 17.92, R²: 0.79
[21500/24000] RMSE: 19.14, R²: 0.78
[21600/24000] RMSE: 19.01, R²: 0.78
[21700/24000] RMSE: 19.65, R²: 0.77
[21800/24000] RMSE: 19.25, R²: 0.78
[21900/24000] RMSE: 18.12, R²: 0.79
[22000/24000] RMSE: 19.70, R²: 0.77
[22100/24000] RMSE: 19.75, R²: 0.77
[22200/24000] RMSE: 19.48, R²: 0.77
[22300/24000] RMSE: 19.38, R²: 0.77
```

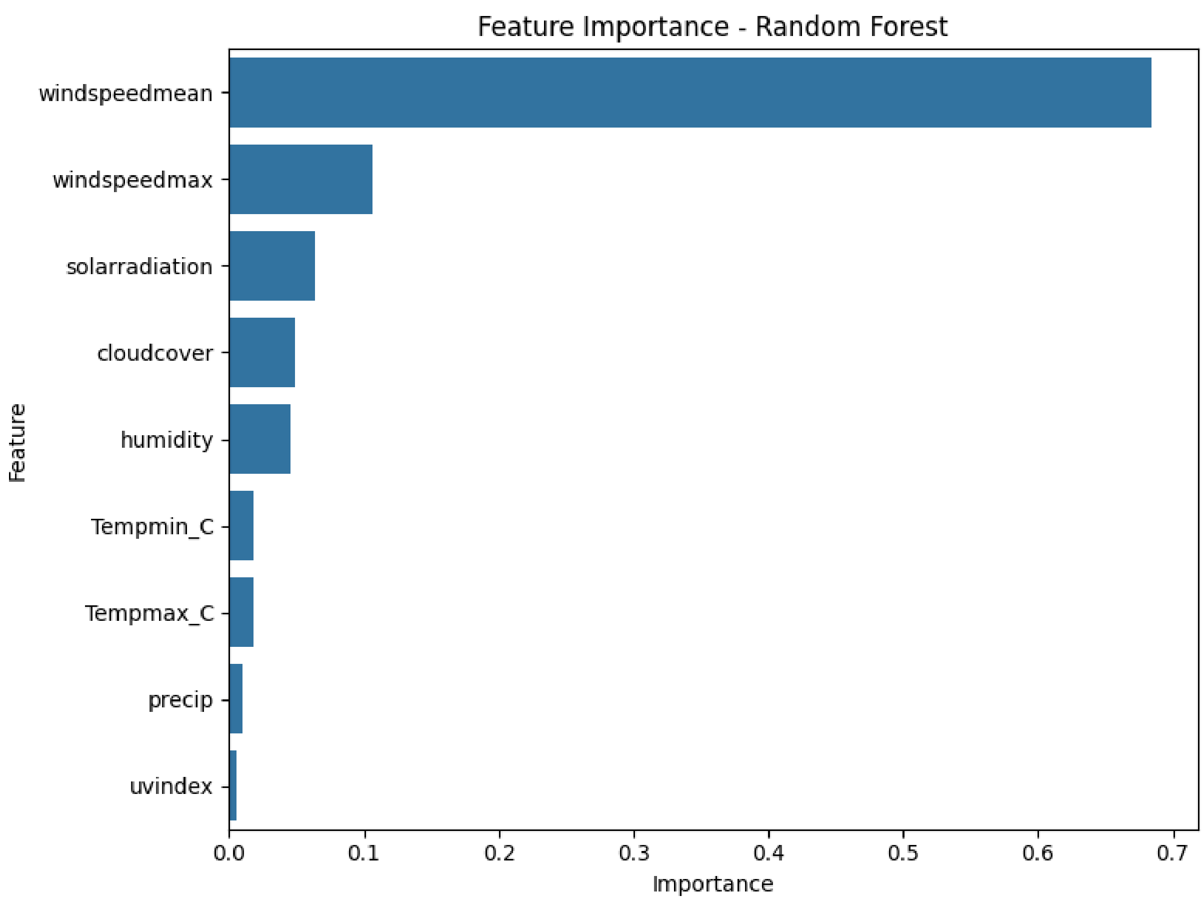
```
[22400/24000] RMSE: 18.84, R²: 0.78
[22500/24000] RMSE: 18.34, R²: 0.79
[22600/24000] RMSE: 18.74, R²: 0.78
[22700/24000] RMSE: 19.42, R²: 0.77
[22800/24000] RMSE: 19.37, R²: 0.77
[22900/24000] RMSE: 20.51, R²: 0.76
[23000/24000] RMSE: 19.83, R²: 0.77
[23100/24000] RMSE: 19.86, R²: 0.77
[23200/24000] RMSE: 19.77, R²: 0.77
[23300/24000] RMSE: 19.11, R²: 0.78
[23400/24000] RMSE: 21.18, R²: 0.75
[23500/24000] RMSE: 19.54, R²: 0.77
[23600/24000] RMSE: 20.35, R²: 0.76
[23700/24000] RMSE: 18.92, R²: 0.78
[23800/24000] RMSE: 18.84, R²: 0.78
[23900/24000] RMSE: 21.05, R²: 0.76
[24000/24000] RMSE: 19.98, R²: 0.77

 Best Parameters:
{'eta': 0.05, 'max_depth': 6, 'gamma': 0, 'max_delta_step': 0, 'alpha': 0,
'colsample_bytree': 1.0, 'subsample': 0.6}
 Best RMSE: 16.47
```

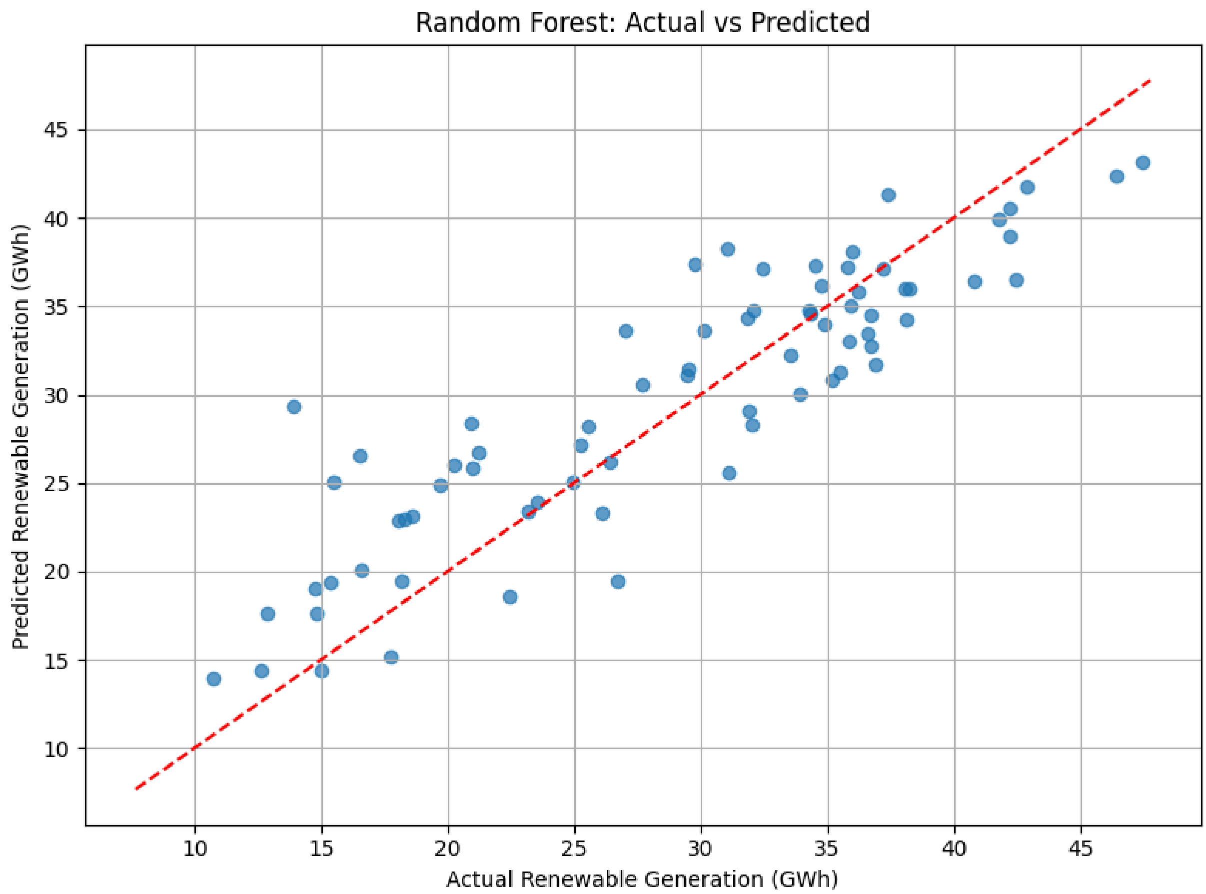
In [145…

```
features = [
    'Tempmax_C','Tempmin_C','windspeedmax','windspeedmean',
    'solarradiation','uvindex','cloudcover','humidity','precip'
]
X = df[features]
y = df['(Combined) Renewable_Generation_GWh']

# Split training and testing data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# Create and train the Random Forest model
rf_model = RandomForestRegressor(
    n_estimators=26,         # Number of trees
    max_depth=11,            # Maximum depth of each tree
    random_state=42,
    n_jobs=-1,               # Parallel computation
    min_samples_leaf=5,
    bootstrap=True
)
rf_model.fit(X_train, y_train)

# Prediction and evaluation
y_pred = rf_model.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f'RMSE: {rmse:.2f}')
print(f'R² Score: {r2:.2f}')

# Visualize feature importance
importances = rf_model.feature_importances_
```

```
feature_df = pd.DataFrame({
    'Feature': features,
    'Importance': importances
}).sort_values(by='Importance', ascending=False)

plt.figure(figsize=(8, 6))
sns.barplot(x='Importance', y='Feature', data=feature_df)
plt.title('Feature Importance - Random Forest')
plt.tight_layout()
plt.show()

# Visualize actual vs predicted
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, alpha=0.7)
plt.plot([y.min(), y.max()], [y.min(), y.max()], '--r')
plt.xlabel('Actual Renewable Generation (GWh)')
plt.ylabel('Predicted Renewable Generation (GWh)')
plt.title('Random Forest: Actual vs Predicted')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
RMSE: 4.36
R² Score: 0.78
```





In [144…

```
for i in range (1,11):
    print(i)
    rf_model = RandomForestRegressor(
        n_estimators=26,
        max_depth=11,
        random_state=42,
        n_jobs=-1,
        min_samples_leaf=i
    )
    rf_model.fit(X_train, y_train)

    y_pred = rf_model.predict(X_test)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    r2 = r2_score(y_test, y_pred)

    print(f'RMSE: {rmse:.2f}')
    print(f'R² Score: {r2:.2f}')
```

```
1
RMSE: 4.38
R² Score: 0.78
2
RMSE: 4.37
R² Score: 0.78
3
RMSE: 4.50
R² Score: 0.76
4
RMSE: 4.47
R² Score: 0.77
5
RMSE: 4.36
R² Score: 0.78
6
RMSE: 4.36
R² Score: 0.78
7
RMSE: 4.36
R² Score: 0.78
8
RMSE: 4.36
R² Score: 0.78
9
RMSE: 4.40
R² Score: 0.77
10
RMSE: 4.52
R² Score: 0.76
```

In [ ]: