**Predicting Renewable Energy Generation, Shortfall, and Anomalies Using Machine Learning**

**Group Name:** Team 7

ADTA 5340 Section 003

Group Members

Varun Kumar Atkuri – 11694018

Irfan Ahmed Shaik – 11641210

Manasa Dontireddy – 11810613

Prudhvi Kolli – 11746825

Charan Rekala – 11702041

**Table of contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No** | **Title** | **Page No** |
| 1 | Abstract | 4 |
| 2 | Business understanding/ Problem Statement | 5 |
| 3 | Literature Review | 6 |
| 4 | Data Understanding and Data Preparation | 7 |
| 4.1 | Data Understanding | 7 |
| 4.2 | Data Preparation | 8 |
| 4.3 | Central Tendency and Dispersion Measures | 9 |
| 5 | Visualizations | 10 |
| 5.1 | Line Plot: Wind Energy Generation Over Time (Germany vs Netherlands) | 10 |
| 5.2 | Line Plot: Wind Renewable Energy Generation Over Time (Germany) | 11 |
| 5.3 | Line Plot: Solar Energy Generation Over Time (Austria vs Netherlands) | 12 |
| 5.4 | Histogram – Distribution of Wind Generation (Germany) | 13 |
| 5.5 | Line Plot – Germany Load vs Renewable Energy Generation | 14 |
| 6 | Data Modelling/Methods | 15 |
| 6.1 | Regression Model: Predicting Next-Day Wind Generation | 15 |
| 6.2 | Classification Model – Wind Shortfall Prediction | 16 |
| 6.3 | DBSCAN Clustering – Anomaly Detection | 18 |
| 7 | Model Evaluation and Discussion | 19 |
| 7.1 | Random Forest Regression – Next Day Wind Generation Prediction | 19 |
| 7.2 | Random Forest Classification – Wind Shortfall Prediction | 21 |
| 7.3 | DBSCAN Clustering – Anomaly Detection | 23 |
| 8 | Deployment | 25 |
| 8.1 | Practical Deployment of Results | 25 |
| 9 | Future Work | 25 |
| 10 | conclusion | 26 |
| 11 | Appendix | 27 |
| 12 | References | 37 |

**1. Abstract:**

The growing dependence on renewable energy has generated the need for accurate forecasting and monitoring of renewable sources like solar power and wind power. This project focuses on Germany’s wind energy market and to develop machine learning models that can forecast next-day wind generation, predict shortfalls, and identify anomalies that may have operational issues. Using the data collected at 15-minute intervals from 2015 to 2020, we applied a structured CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology to guide the analysis.

The key aspect of this project is the feature engineering where leveraging lag values, rolling averages, and time-based attributes are used to capture temporal patterns and production behavior. A Random Forest Regressor was applied for generation forecasting, which achieved a strong predictive performance with low error rates. Shortfall prediction was addressed through classification models, where the Random Forest Classifier demonstrated high precision and solid recall. Unsupervised learning with DBSCAN successfully isolated rare anomalies.

The results show that machine learning methods enhanced the prediction of energy generation forecasting, shortfall and detection anomalies. By improving the accuracy and early event detection, the models support better decision-making for energy planning and contribute for the efficient use of sustainable energy resources.

**2. Business Understanding / Problem Statement**

In recent years energy consumption in the world has been increasing rapidly for industrial and household purposes. This made the global to look into renewable energy resources. Most of the private sector and government sector are interested in investing in renewable energy sources particularly wind and solar. Unlike fossil fuel sources, wind and solar energy heavily depend on environmental conditions. So, it is very difficult to predict their production load. This creates a critical need for future predictions, shortfall identifying, and anomaly monitoring to main grid stability and optimize energy distribution. This helps grid operators and investors to spend more on renewable energy resources.

This project work mainly focuses on Germany Wind renewable energy market using high quality of data collected (15-minute interval) from Open Power System Data (OPSD). The dataset provides rich information about price, wind and solar energy generation actuals, wind and solar load demand values and various time-based features and different EU regional attributes. The data also include various European countries such as Austria, Belgium, Germany, Hungary, Luxembourg and Netherlands provides broad view of energy consumption in neighboring countries.

The core business questions guiding this analysis are:

1. Can we able to predict next-day wind energy generation in Germany using historical and time-based features?
2. Can we predict when wind energy output will fall below expected levels in order to issue early alerts?
3. Can we identify unusual energy behavior in the system that may indicate irregular usage patterns?

The data mining goals are to:

* Apply regression method to predict next day wind energy prediction in Germany.
* Use classification methods to identify shortfall before it occurs.
* Apply unsupervised outlier detection techniques to find rare energy patterns.

By solving these problems, this project aims to support operators and energy planners in decreasing operational risk, optimizing the use of renewable energy sources and reliability in renewable powered energy systems. A complete CRISP-DM methodology will be followed from business understanding through deployment to ensure clear structure and outcome driven approach.

**3. Literature Review**

With the shift towards renewable energy sources, there has been a growth in using machine learning methods to improve forecasting, detect anomalies, and manage energy effectively. The renewable sources like wind enrgy and solar energy are weather-dependent and unpredictable, since then researchers have been exploring more advanced ML models to address these challenges.

**Rasouli, Arabzad, and Askarany (2023)** implemented ML techniques for renewable energy forecasting. They found that Random Forest algorithms, performed well when dealing with the non-linear, noisy, and dynamic nature of energy data. The review concluded that the use of Random Forests, thanks to their ensemble design and strong resistance to overfitting, are particularly effective for short-term forecasting where environmental uncertainty is more.

In a comparison study by **Sabbaghi and Barzegar (2022),** they assessed the performance of Random Forest, Gradient Boosting Machines, and Long Short-Term Memory (LSTM) networks for wind energy forecasting. They found that while deep learning models like LSTM can achieve high accuracy with enough data, Random Forest models offer a balance between performance, interpretability, and computational efficiency. In the end, Random Forests performed better in scenarios needing fast, reliable predictions without heavy hyperparameter tuning.

Anomaly detection is an important part of renewable energy management. It relies on unsupervised-learning methods that can spot outliers without labeled data. **Fang, Zhang, and Chen (2023)** explored DBSCAN (Density-Based Spatial Clustering of Applications with Noise) in energy systems. They found it effective in detecting low-density anomalies in complex, high-volume datasets like those from energy monitoring systems.

Visualization and exploratory analysis are crucial for understanding energy system behavior. **Waskom (2021)** highlighted the advantages of Seaborn, which is a Python visualization library that helps create clear and attractive plots. In this project, Seaborn was used to build line plots, histograms, and scatter plots, helping in the feature understanding and model evaluation.

The current research supports the modeling and methodology choices made in this project. These studies back the use of Random Forests for wind energy forecasting, shortfall and DBSCAN for anomaly detection. Together, these findings make a strong case for how machine learning can significantly boost the reliability, accuracy, and efficiency of renewable energy grids.

**4. Data Understanding and Data Preparation**

The data understanding, data preparation and exploratory data analysis (EDA) steps were undertaken by following process using CRISP-DM methodology. The dataset includes 201604 rows and 61 columns, time-stamped records collected at 15-minute intervals. Following these steps will help us to understand the data better and evaluate it for completeness, quality, and usability for model building.

#### **4.1. Data Understanding**

#### The dataset was explored using .info() and .describe() to understand its structure, datatype and basic statistical measures such as mean, standard deviation, min/max, and quantiles for each column. The (data == 0).sum() method is used to identify columns with zero values in the data. This is one of the important steps in the process to understand the data better, especially in solar and wind generation columns, where zero values represent environmental conditions (e.g., no sunlight at night). The .isnull().sum() method was used to identify columns with NaN values. These steps help us to understand the data better. The dataset datetime column was set as an index, verified and used for generating time-based features and aggregations.

#### **4.2. Data Preparation**

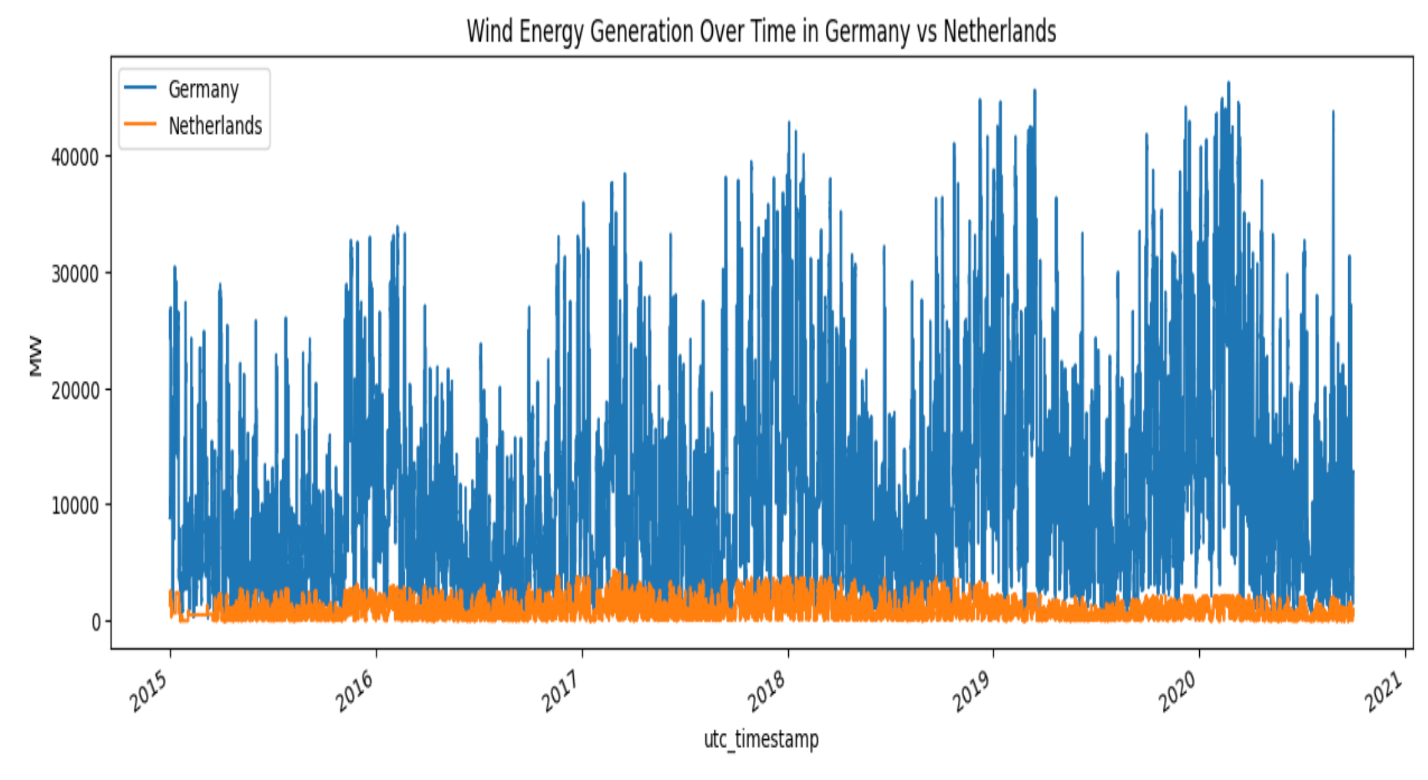
* We used forward fill (.ffill()) and backward fill (.bfill()) methods to handle missing values. By using this method, we can retain the data without disturbing the trend.
* Feature Engineering steps were undertaken such as lag, rolling and time-based features. These features make the model to learn better and provide valuable output for our project work
* Lag features were created wind\_15min\_ago, wind\_24hr\_ago using .shift(1) and .shift(96). The shift(1) will provide value of last 15 minute ago value by calculating 1\*15. In the same way .shift(96) will calculate the last 24 hours ago value using 96\*15 provide 14440 minutes equal to 24 hours ago value. These lag features were particularly used in time series predictions and help us to predict next data wind generation and shortfall prediction.
* Rolling features expected\_wind column was created using .rolling(96).mean() to get a 24-hour moving average for shortfall detection.
* We also extracted time-based features such as hour, day\_of\_week, and month using data.index.hour, data.index.dayofweek and data.index.month to allow model to understand the data clearly.
* A binary classification label (Germany\_shortfall) column was created using logical comparison of actual wind against 80% of expected values.
* After creating lag features, any rows with NaN values resulting from .shift() and .rolling() operations were dropped using .dropna().

**4.3. Central Tendency and Dispersion Measures:**

We calculated key statistics including mean, standard deviation, minimum, maximum, and quartiles. We observed key major columns such as load, wind and solar generation. For example, Austria's wind generation had a mean of 729.27MW and a high standard deviation of 693.58 MW, indicating significant variation due to weather conditions. The median was lower than the mean. These statistics identify variation in wind production and the nature of renewable energy data.

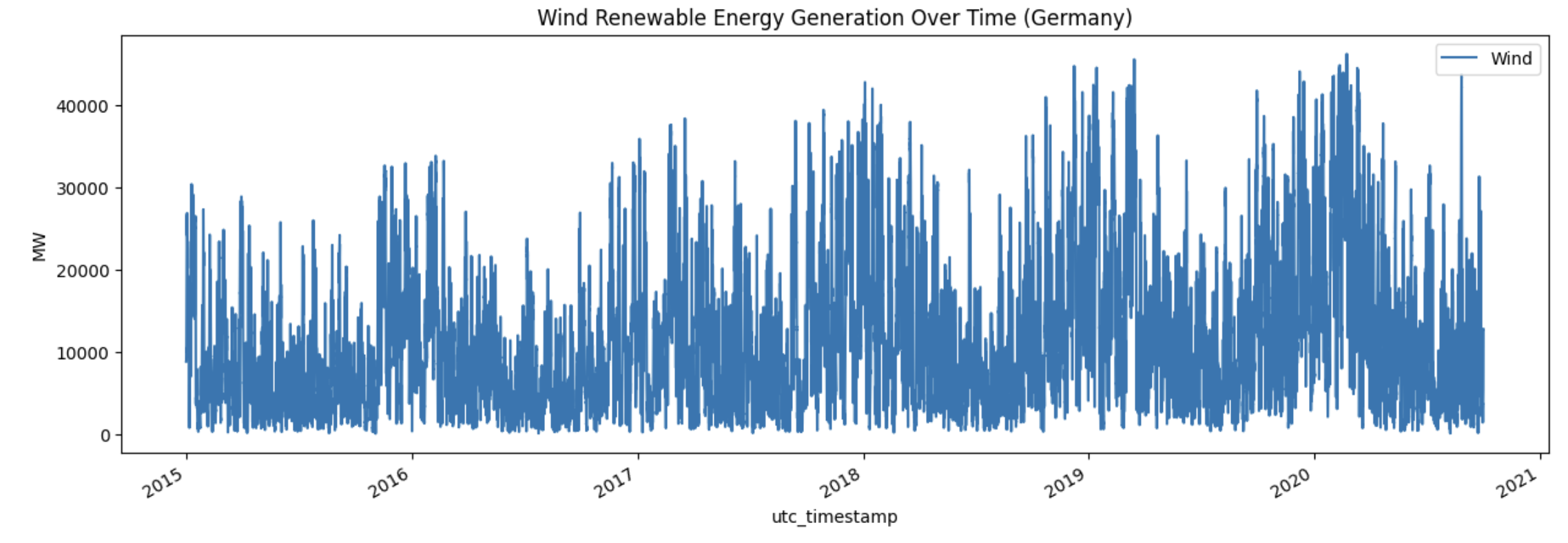
**5. Visualizations**

**5.1. Line Plot: Wind Energy Generation Over Time (Germany vs Netherlands)**

 The line graph provides a wind energy generation comparison between Germany and Netherlands over a five-year period. The x-axis represents utc\_timestamp and y-axis represents wind energy generation in megawatts (MW). The blue line represents Germany wind generation and orange represents Netherlands wind generation.

The graph provides clear picture that Germany provides high wind energy production crossing more than 30000 megawatts MW. The Netherland shows very low and stable wind generation with less than 3000 megawatts MW. Also, we can observe that wind production in Germany is constantly increasing every year, and it reached peak in 2019 and 2020 above 40000 megawatts. This comparison reveals regional difference in wind generation, making Germany is a better choice for modeling shortage and anomaly identification.

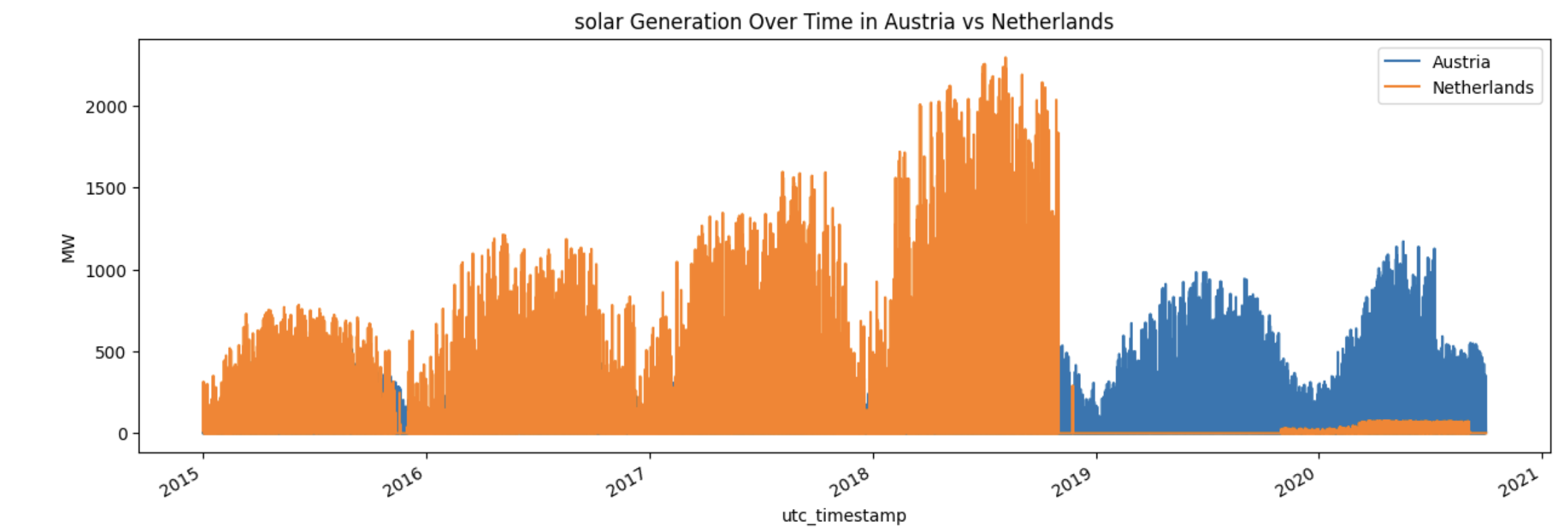
**5.2. Line Plot: Wind Renewable Energy Generation Over Time (Germany)**

 The line graph provides information about the wind energy generation in Germany from 2015 to 2020, based on 15 minutes of interval. The x axis represents the utc\_timestamp and y axis represents the German\_wind\_generation\_actual, measured in megawatts (MW).

There is a clear increase in spikes at the end of the month every year. Suggest seasonal wind patterns. At the same time, the production of wind power generation is unpredictable, with short term increase at all times. According to the plot, wind generation capacity has generally increased over time, whether as a result of improved weather or the growth of infrastructure.

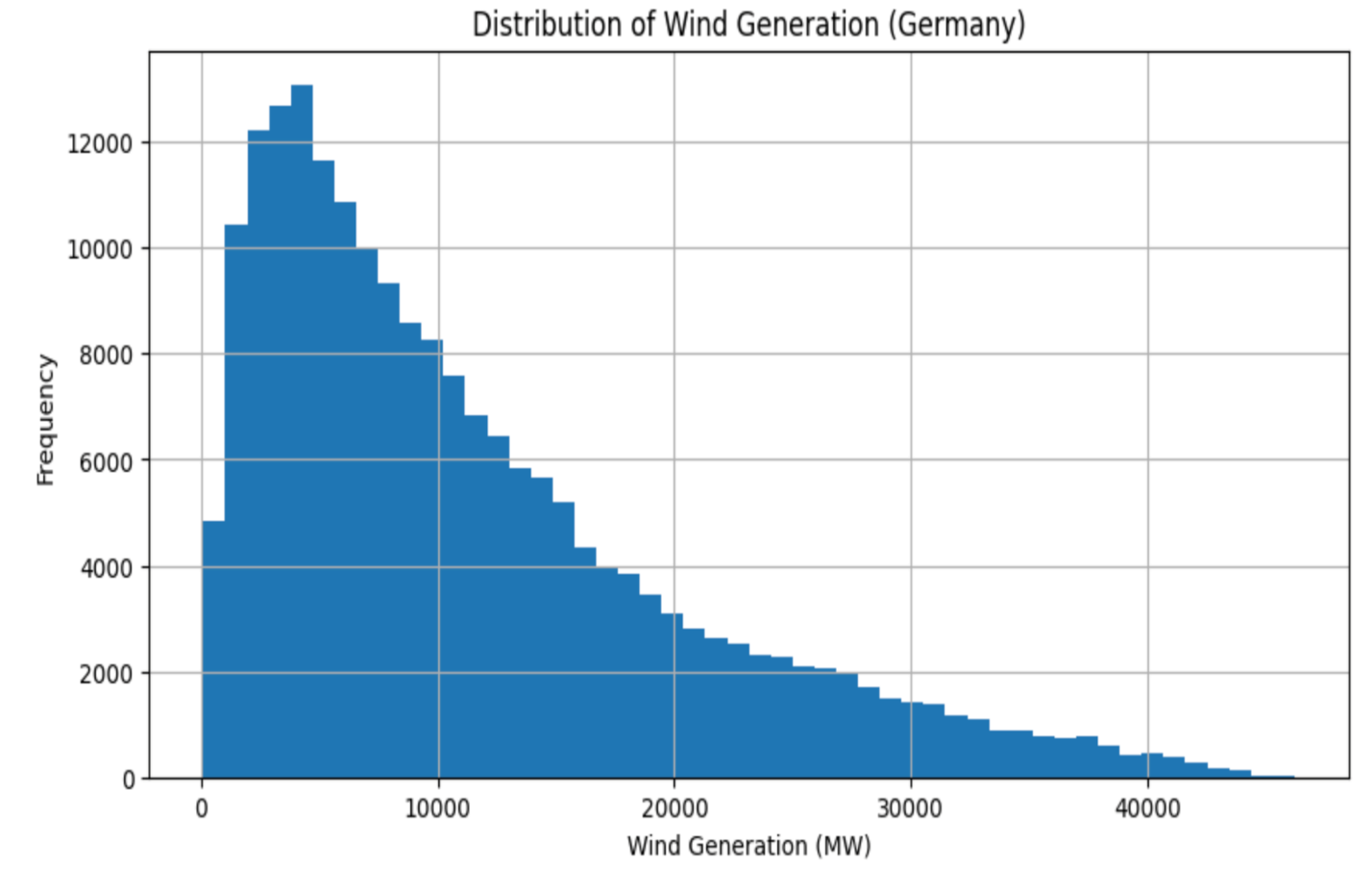
This above line chart provides a clear view of time-based behavior of the wind data, supporting the decision to include lag features and time attributes in predictive modeling.

**5.3. Line Plot: Solar Energy Generation Over Time (Austria vs Netherlands)**

 This line graph visualizes the solar energy generation in Austria and Netherlands over a five-year period. The x-axis represents utc\_timestamp and y-axis represents solar energy generation in megawatts (MW). The blue line represents Austria solar generation and orange represents Netherlands solar generation.

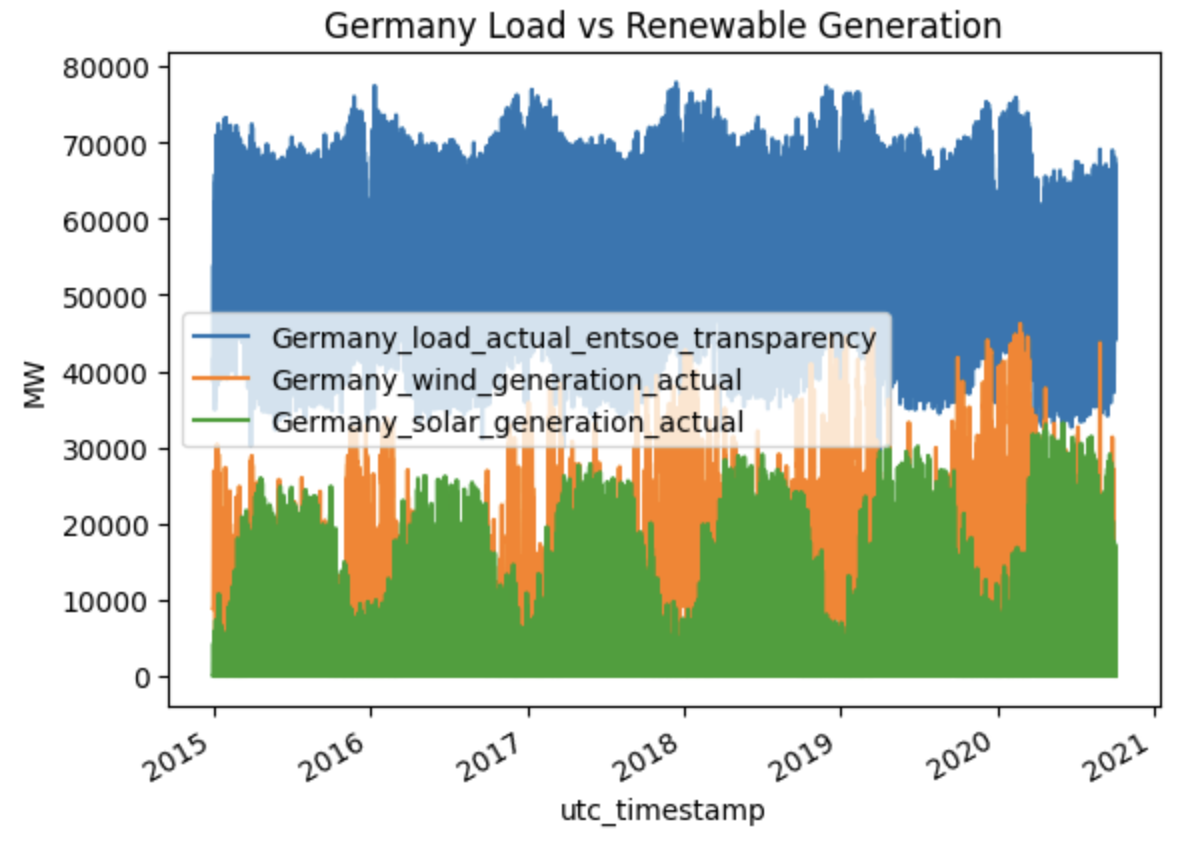
Netherlands constantly produces higher solar generation output compared to Austria. But at the end of 2018 there is a huge drop in solar production in Netherlands. There may be several reasons for reducing in solar power but at the same time Austria maintain constant solar generation throughout the timeline, suggesting better infrastructure or larger solar panel coverage. Also, we can observe the solar production peaks in summer and decrease in winter. These trends give a clear view that month-based features are required for predictive modelling and validate the inclusion of time-based lag variables.

**5.4. Histogram – Distribution of Wind Generation (Germany)**

 This histogram visualizes Distribution of wind generation in Germany throughout the full data collection from 2015 to 2020. The X-axis describes the wind generation in megawatts MW, at interval of 1000 MW. The Y-axis represents frequency, which is 15-minute intervals for each wind generation.

The distribution is right-skewed, with most values ranging between 2,000 to 10,000 MW.

**5.5. Line Plot – Germany Load vs Renewable Energy Generation**

 This multi-line chart compares Germany total energy load with actual solar and wind renewable energy generation from 2015 – 2020. The X-axis represents utc\_timestamp and y-axis represents power generation in megawatts (MW). The blue line Germany\_load\_actual\_entsoe\_transparency represents Germany energy demand. The orange line Germany\_wind\_generation\_actual represents actual wind energy generation. The green line Germany\_solar\_generation\_actual represents actual solar energy generation in Germany.

The energy demand in Germany remains stable over time with minor seasonal drops or rises. Compared to solar, wind energy generation is significantly higher throughout the years, with fewer drops due to climatic changes. Solar energy performs good in summer and lower in winter highlighting a predictable trend. By this visualization we can observe renewable energy wind and solar falls below the expected load, supporting our shortfall classification predictions.

**6. Data Modelling/Methods:**

In this project, a structured modeling approach was followed to address the core business objectives: predicting next-day wind energy generation, detecting shortfalls, and identifying anomalies in the German renewable energy market. Statistical and machine learning methods were carefully selected, applied correctly, and supported by a clear nature of the data and business goals.

A targeted modeling strategy was selected to meet the project's forecasting, classification, and anomaly detection goals using Germany’s wind energy data. Methods were selected based on the nature of each task, and all models were built using a consistent set of engineered features. The three primary modeling approaches are summarized below:

1. Regression Model: Predicting Next-Day Wind Generation
2. Classification Model: Predicting Wind Energy Shortfalls
3. Unsupervised Learning: Anomaly Detection

**6.1. Regression Model: Predicting Next-Day Wind Generation:**

For forecasting next-day wind energy generation in Germany, we choose Random Forest Regressor because of its ability to model non-linear relationships effectively and its ability to handle outliers and noisy data, which are frequently encountered in renewable energy datasets. As an ensemble of multiple decision trees, it minimizes overfitting by averaging results across multiple paths. Based on the complexity of the data (time dependencies, sudden spikes), Random Forest was most suitable to predict next-day forecasts without having heavy parameter tuning.

Featuring engineering plays a crucial role in model performance. The selected dependent variable was used in regression model include wind\_15min\_ago, representing the generation 15 minutes prior, to capture immediate past trends, and wind\_24hr\_ago, reflecting generation exactly one day earlier. Temporal attributes such as hour, day\_of\_week, and month were also included which allow the model to learn daily, weekly, and seasonal patterns inherent in wind energy behavior. This combination of lagged and calendar-based features enables the model to effectively capture complex time-series dependencies influencing wind generation. The target variable is German\_wind\_generation\_actual, representing the actual wind energy produced in Germany, recorded every 15 minutes. This metric is a critical operational indicator for forecasting renewable energy availability.

The data was split into an 80/20 train-test split method. A Random Forest Regressor was initialized with 100 trees (n\_estimators=100) and a fixed random seed (random\_state=42) to ensure reproducibility of results. The model was trained using the selected features (X\_train) and the target (y\_train), where multiple decision trees were built on bootstrap samples and their predictions averaged to form the final forecast. Finally, the trained model was used to predict next-day wind generation values on the testing data (X\_test), providing a strong foundation for operational energy planning.

The regression model was evaluated on the following metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

**6.2. Classification Model – Wind Shortfall Prediction:**

Identifying periods when wind generation falls below expected thresholds is critical for maintaining grid reliability and operational stability. For shortfall prediction, we are choosing a random forest classifier. The advantage of choosing this is Random Forest handles imbalanced classes effectively, a key requirement since shortfall events are relatively rare compared to normal generation periods. It is capable of modeling complex, non-linear interactions between features and internally performs a form of feature selection by prioritizing the most informative inputs. Given the binary nature of the shortfall prediction problem (shortfall or no shortfall), the Random Forest Classifier provides a strong balance between high accuracy and model interpretability.

For feature selection, we use historical trends and expectations based on broader temporal patterns. The features include wind\_15min\_ago, capturing the most recent wind generation trend, and wind\_24hr\_ago, offering historical context from the previous day. The expected\_wind feature, representing the 24-hour rolling mean of wind generation, was critical for defining what constitutes "normal" energy production. Calendar-based features such as hour, day\_of\_week, and month were also incorporated to allow the model to recognize regular seasonality patterns.

The target variable we choose here is Germany\_shortfall, a binary label specifically for this model. Instances were labeled as '1' if the actual wind generation fell below 80% of the expected rolling average, and as '0' otherwise. This threshold was chosen based on domain knowledge to create a meaningful early warning system for operational planning.

The modelling began by creating the shortfall labels based on defined threshold. The dataset was then split into an 80/20 train-test split. A Random Forest Classifier was initialized with 100 trees (n\_estimators=100), a maximum tree depth of 5 (max\_depth=5) to prevent overfitting, and a square-root rule for feature selection (max\_features='sqrt'). The model was trained using the lagged features, expected wind values, and calendar-based time features. Once trained, the classifier was used to predict shortfall labels on unseen test data, allowing evaluation of the model’s ability to issue accurate operational warnings.

The evaluation metric includes accuracy, precision, recall and f1 score for each class (shortfall / normal). Also, confusion matrix visual graph provides correctness of the predictions.

**6.3. DBSCAN Clustering – Anomaly Detection:**

Along with forecasting and shortfall prediction it was important to identify unusual patterns or operational anomalies within the energy system, even without labeled anomaly data. For this the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm was selected. DBSCAN was an ideal choice because it does not require pre-specifying the number of clusters which makes it highly flexible for unknown anomaly patterns. It also naturally detects outliers by identifying low-density regions, and its performance remains stable even when clusters vary in shape and scale.

For anomaly detection, two critical operational features were selected: German\_wind\_generation\_actual, representing wind energy output, and German\_load\_actual\_entsoe\_transparency, representing total energy demand. These variables capture both the supply and demand sides of the system which provides space suitable for identifying deviations from normal operational patterns.

The modeling process began with selecting the features that were extracted and standardized to improve clustering performance. Standardization ensured that both wind generation and load values contributed equally to distance calculations, preventing one feature from dominating the clustering process.

The model was initialized with an epsilon (eps) value of 1500 MW, defining the neighborhood radius, and a min\_samples value of 5, specifying the minimum number of points needed to form a cluster. The model was then fitted to the prepared data, separating core cluster points from noise points based on density.

The resulting scatterplot was the data points colored according to their assigned cluster labels. Normal operational behavior was assigned to cluster '0', outliers’ points that did not belong to any dense region were labeled as '-1'. This approach allowed for easy visual identification of anomalies that may represent sudden operational disruptions, data errors, or rare environmental conditions affecting wind generation and energy load simultaneously.

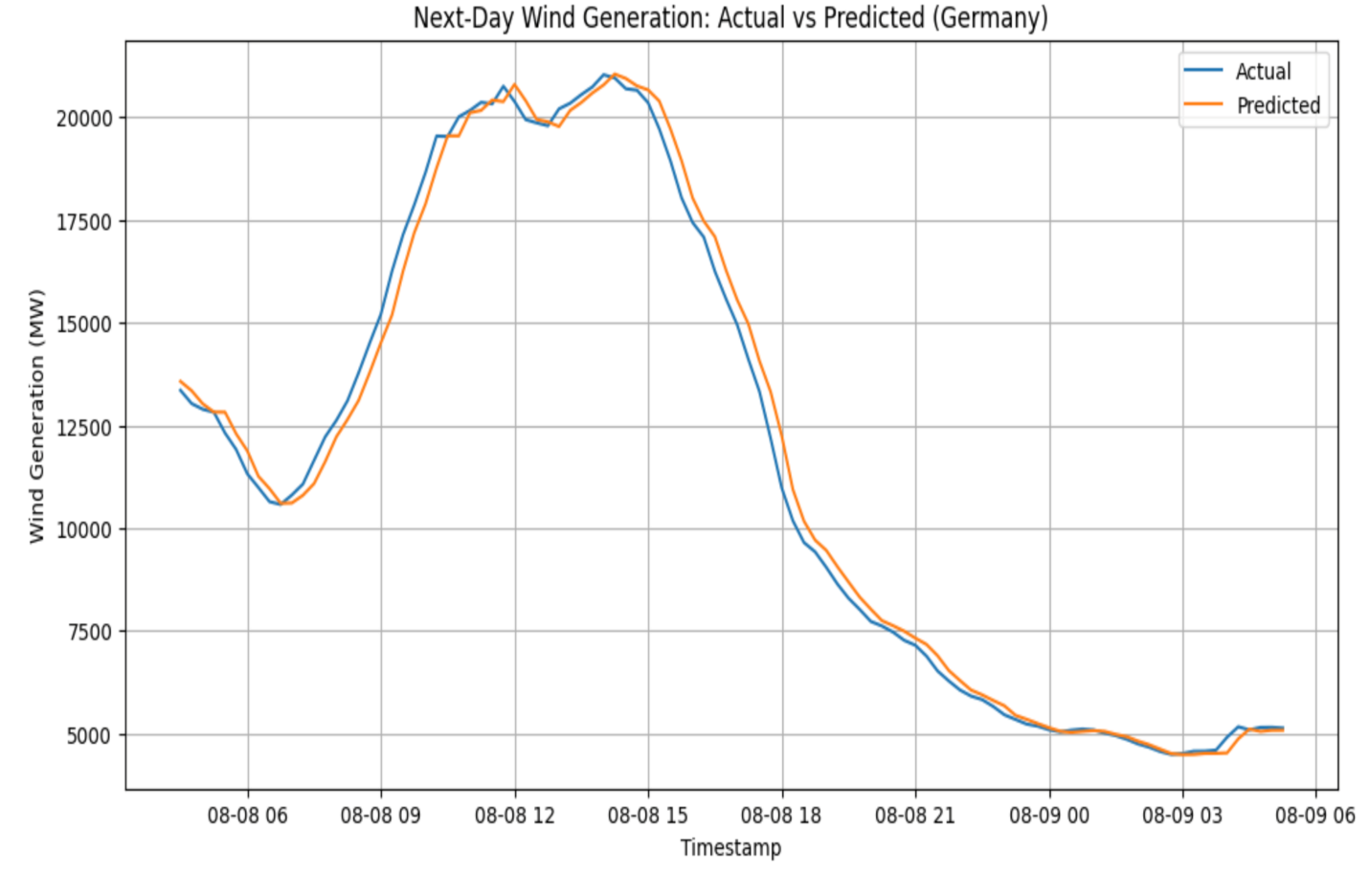
**7. Model Evaluation and Discussion:**

**7.1. Random Forest Regression – Next Day Wind Generation Prediction:**

The random forest regression model achieved best prediction results:

|  |  |
| --- | --- |
| Metric | Value |
| Mean Absolute Error (MAE) | 237.95 MW |
| Root Mean Squared Error (RMSE) | 330.17 MW |
| Mean Wind Generation | 14314 MW |
| % MAE | 1.66% |
| % RMSE | 2.31% |

The mean absolute error indicates that on average the next day wind generation prediction is off by above 200 MW, which is minimal generation capacity. Also, the percentage of MAE 1.66% shows high model reliability related to the total scale of wind production. The model effectively captures the daily, seasonal and different variation in the wind pattern, showing that the lag and time features were very useful.

 The line graph next-day wind generation: actual vs predicted provides a direct visual comparison between actual and model prediction on a particular day. The blue line represents actual wind energy generation. The orange line represents model predicted wind generation. The X-axis represents timestamp, and Y-axis represents energy generation in megawatts (MW).

The graph shows a clear view that the predicted curve is closely related to actual wind generation curve across different hours of the day, identifying both peak and low points of energy production identified correctly. However, minor differences are observed during very sharp drops or rises, which are usually expected in renewable energy.

The strong performance of the model predicting future energy generation and daily generation patterns helps grid operators to plan accordingly, also validating the need of using lag and time-based features.

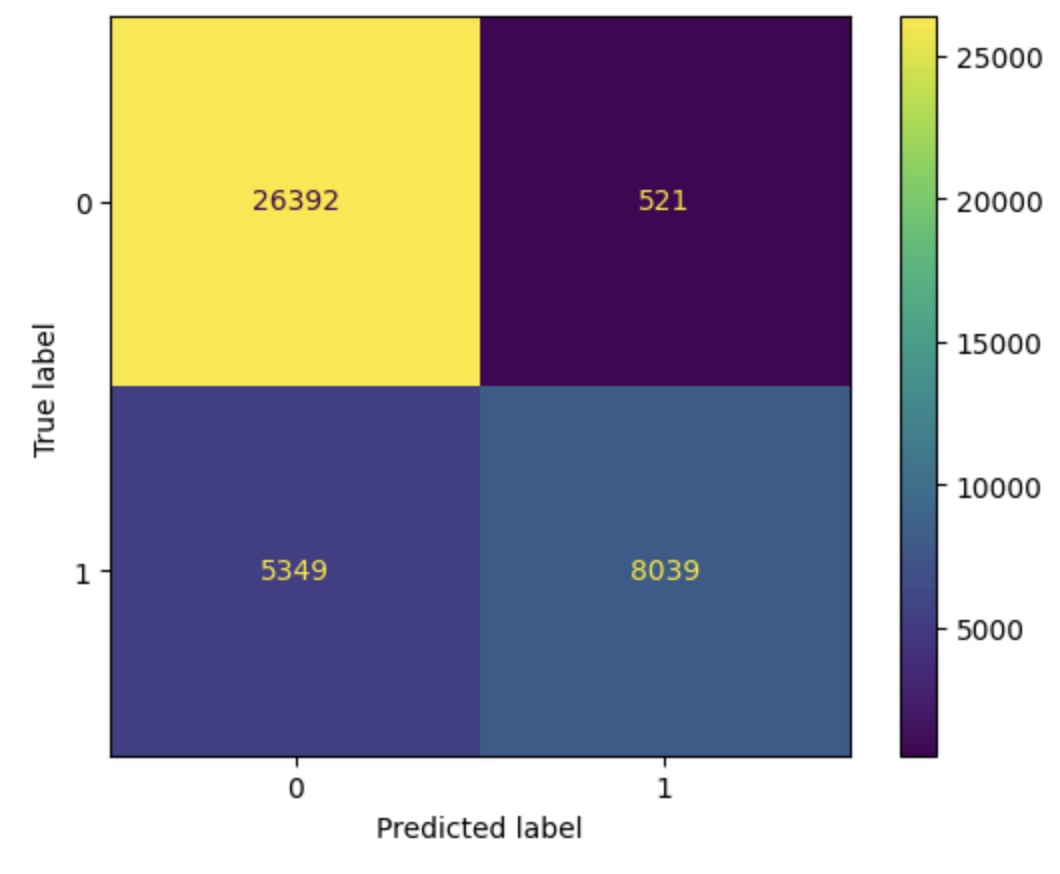
The model answered the first question clearly. Yes, we can accurately predict next-day wind energy generation based on historical patterns.

**7.2. Random Forest Classification – Wind Shortfall Prediction:**

The random forest classification model is able to predict shortfall when wind generation falls below expected level.

|  |  |
| --- | --- |
| Metric | Value |
| Accuracy | 85% |
| Precision (shortfall class) | 94% |
| Recall (shortfall class) | 60% |
| F1 Score (shortfall class) | 73% |

Hight precision 94% ensure that model predicts a shortfall, its highly to be a correct. Also, moderate recall 60% indicate that the model is able to detect shortfall but misses very low events. These are very important in energy planning, where false values may result in unwanted cost and operational planning. Overall, the f1 score 73% indicates a strong balance between precision and recall values. This shows that the model is practically helpful in the real world.

 The confusion matrix shows the classification performance of the random forest model for identifying shortfall in wind generation.

The confusion matrix provides model strong performance in identifying normal energy generation (26392) as True Negative and correctly predicted wind shortfalls (8039) as True Positive. Also able to identify incorrect shortfall (521) when it was normal as False Positive and missed actual shortfall occurred but predicted as normal (5349) as False Negative.

The low false positive (521) indicates that model is very sensitive before saying it as shortfall, which reduces unwanted actions.

The model supports precision with the moderate number of false negatives (5349), which is acceptable because responding to false alarms in energy systems can be more costly than ignoring a few problems.

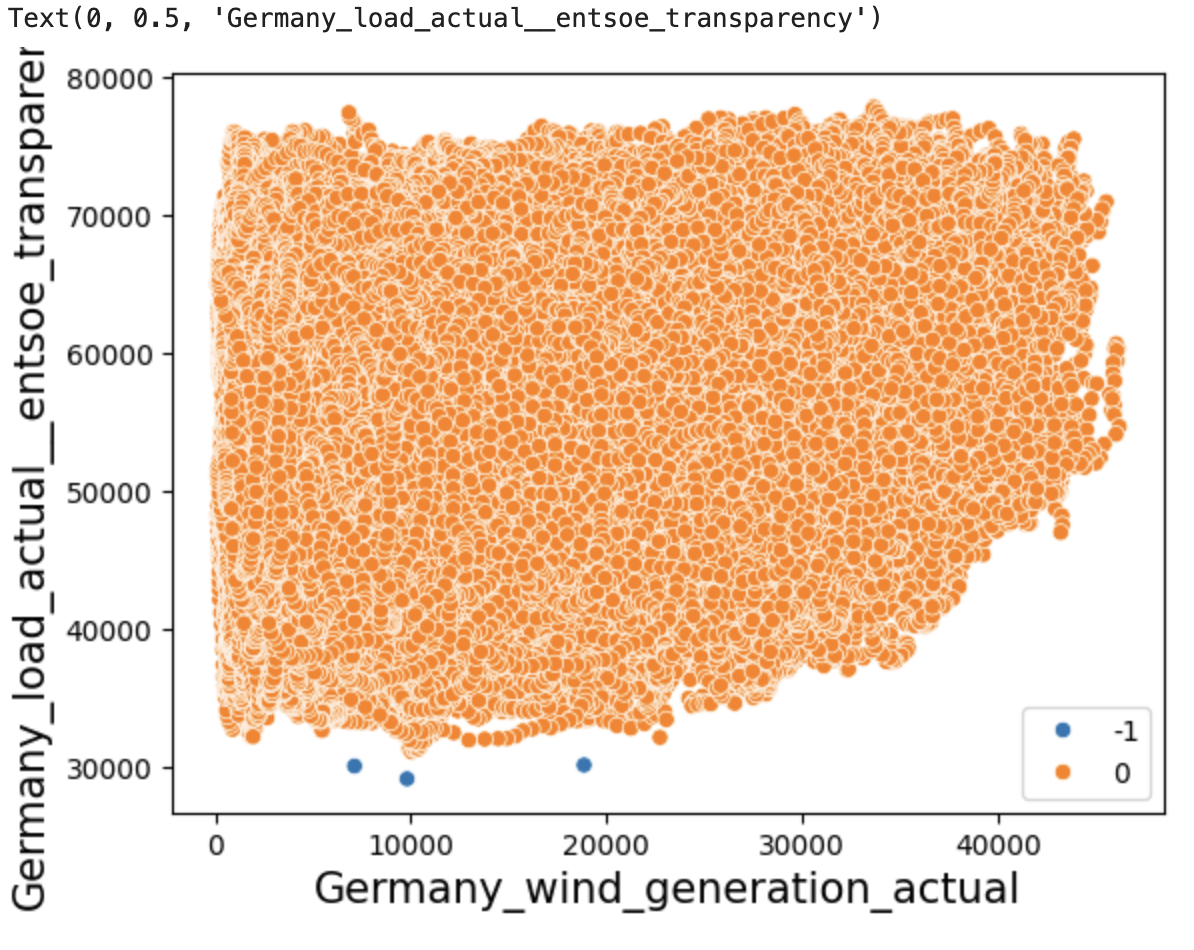
The model successfully answers the second question. Yes, with high precision and balanced performance, yes, we can be able to predict the energy shortfall before it occurs.

**7.3. DBSCAN Clustering – Anomaly Detection:**

The DBSCAN Clustering model identifies outliers in the dataset.

|  |  |
| --- | --- |
| Outcome | Value |
| Total Sample | 201499 |
| Outlier Detection | 3 |

The low number of outliers suggests that wind generation and load behavior were generally stable over the years. Finding outliers in rare situations where energy generation-load relationships were abnormal due to changes in weather condition, outages and delay in updates. DBSCAN didn't need labeled training data to successfully distinguish between regular behavior and anomalous occurrences.

 The scatter plot above represents the clustering of wind generation (X-axis) and actual energy load (Y-axis) using the DBSCAN algorithm. The goal was to detect outliers where renewable generation might behave unusually relative to national energy consumption.

The orange points (0) in the scatter plot are normal points identified by DBSCAN, representing expected operational behavior between Germany wind generation and energy load. The blue points (-1) are the main outliers in the plot. These points are very crucial as they may indicate shortfall in generation or unstable production conditions. The DBSCAN model is able to predict 3 outliers in the data. However, Anomalies are rare but the DBSCAN model is able to detect without depending on labeled data. Also successfully separate the outlier from the normal operation behavior.

This clustering method answers our third question. Yes, DBSCAN provided an unsupervised method to flag outliers in the production pattern.

**8. Deployment:**

**8.1. Practical Deployment of Results:**

The models used in this project provide valuable insights for renewable energy grid operators, planners and stakeholders.

By predicting next day wind generation before 24 hours in advance with a low error margin will help energy planners to schedule backup plans according to the energy market trend. The early shortfall classification model offers a reliable warning system to alert grid operators if renewable generation is expected to fall below operational thresholds, enables immediate action to be taken to maintain the flow.

The unsupervised DBSCAN model identifies unusual patterns in renewable energy without any human intervention, helps operational team to identify unusual patterns and quickly investigate the problem before leading to big outage, which saves huge cost.

**9. Future Work**

Integrating with real time weather data will help the model to predict much more accurate results because renewable energy is heavily dependent on natural conditions. Also, instead of using static 80% threshold for energy shortfall prediction, dynamic thresholds will be more helpful for the prediction because shortfall prediction point will vary based on seasonal or market demand conditions. So, making it dynamically will be useful in the real world. Exploring advanced models like LSTM and Hybrid transformer models could capture more accurate and fast results in real time. Automated Real time a streaming pipeline will help the model to analyze the new data arriving to maintain performance over time.

**10. Conclusion**

The project demonstrated the successful use of machine learning models to address the key challenges in renewable energy management for Germany’s wind energy sector by forecasting next-day wind generation, predicting shortfall events, and identifying operational anomalies, the models which were developed provided actionable insights that can help energy planners and grid operators maintain stability and optimize resource allocation.

The Random Forest Regressor for forecasting next-day wind generation delivered strong predictive performance, capturing complex time dependencies and seasonal patterns in wind generation with minimum error. The Random Forest Classifier for predicting shortfall events proved effective in detecting periods of significant underperformance, offering early warning capabilities with high precision and balanced recall. The DBSCAN clustering approach for anomaly detection effectively isolated rare anomalies without requiring labeled data, highlighting the unusual behaviors that could signal critical operational risks.

To conclude, the project validated that machine learning techniques when paired with proper feature engineering can significantly enhance the reliability and efficiency of renewable energy systems. The findings reinforce the growing potential of data-driven approaches to support the global transition toward sustainable and resilient energy infrastructures.

**11. Appendix:**

#Import Required Libraries

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

from sklearn.ensemble import RandomForestClassifier

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

from sklearn.metrics import classification\_report, confusion\_matrix

from sklearn import metrics

from sklearn.cluster import DBSCAN

from sklearn.preprocessing import StandardScaler

import seaborn as sns

import warnings

warnings.filterwarnings("ignore")

#loading the dataset

data=pd.read\_csv('time\_series\_15min\_singleindex.csv')

#Reading the data

data

#checking basic info

data.info()

#Column with NaN values

data['DE\_LU\_load\_actual\_entsoe\_transparency']

data['HU\_solar\_generation\_actual']

#Droping column with NaN values

data.dropna(axis=1,how='all',inplace=True)

data.info()

#using Forward Fill

data.fillna(method='ffill',inplace=True)

#using Backward fill

data.fillna(method='bfill',inplace=True)

data.dropna(how='all',inplace=True)

data.info()

#Changing data type for date column and makeing it as index

data['utc\_timestamp']=pd.to\_datetime(data['utc\_timestamp'])

data.set\_index('utc\_timestamp', inplace=True)

data.index.dtype

#Droping the column

data.drop('cet\_cest\_timestamp',axis=1,inplace=True)

data.info()

#checking for duplicates in the index

data.index.duplicated().sum()

#checking for duplicates

data.duplicated().sum()

data[data.duplicated()]

#Droping duplicates

data.drop\_duplicates(inplace=True)

print(data.duplicated().sum())

data.describe()

#sum of zeros in dataset

print((data == 0).sum())

#checking for nan values

print(data.isnull().sum())

#Changing columns names for better understanding

|  |  |
| --- | --- |
| data = data.rename(columns={ 'DE\_wind\_generation\_actual': 'Germany\_wind\_generation\_actual', 'DE\_load\_actual\_entsoe\_transparency': 'Germany\_load\_actual\_entsoe\_transparency', 'DE\_solar\_generation\_actual': 'Germany\_solar\_generation\_actual', 'AT\_solar\_generation\_actual': 'Austria\_solar\_generation\_actual','NL\_solar\_generation\_actual': 'Netherlands\_solar\_generation\_actual' })  #printing first 10 rows  print(data.head(10))  data.info()  plt.figure(figsize=(15, 5))  data['Germany\_wind\_generation\_actual'].plot(kind='line', label='Wind')  plt.title("Wind Renewable Energy Generation Over Time (Germany)")  plt.ylabel("MW")  plt.xlabel("utc\_timestamp")  plt.legend()  plt.show()  plt.figure(figsize=(15, 5))  data['Austria\_solar\_generation\_actual'].plot(kind='line', label='Austria')  data['Netherlands\_solar\_generation\_actual'].plot(kind='line', label='Netherlands')  plt.title("solar Generation Over Time in Austria vs Netherlands")  plt.ylabel("MW")  plt.xlabel("utc\_timestamp")  plt.legend()  plt.show()  plt.figure(figsize=(15, 5))  data['Germany\_wind\_generation\_actual'].plot(kind='line',label='Germany')  data['NL\_wind\_generation\_actual'].plot(kind='line', label='Netherlands')  plt.title("Wind Energy Generation Over Time in Germany vs Netherlands")  plt.ylabel("MW")  plt.xlabel("utc\_timestamp")  plt.legend()  plt.show()  plt.figure(figsize=(10, 5))  data['Germany\_wind\_generation\_actual'].hist(bins=50)  plt.title("Distribution of Wind Generation (Germany)")  plt.xlabel("Wind Generation (MW)")  plt.ylabel("Frequency")  plt.show()  plt.figure(figsize=(15, 5))  data[['Germany\_load\_actual\_entsoe\_transparency', 'Germany\_wind\_generation\_actual', 'Germany\_solar\_generation\_actual']].plot(kind='line')  plt.title("Germany Load vs Renewable Generation")  plt.ylabel("MW")  plt.legend()  plt.show()  #Time based features  data['hour'] = data.index.hour  data['day\_of\_week'] = data.index.dayofweek  data['month'] = data.index.month  #Lag features for shift/rolling  data['wind\_15min\_ago'] = data['Germany\_wind\_generation\_actual'].shift(1)  data['wind\_24hr\_ago'] = data['Germany\_wind\_generation\_actual'].shift(96)  data['avg\_expected\_wind'] = data['Germany\_wind\_generation\_actual'].rolling(96).mean()  #Drop NA values from Lag Features  data = data.dropna(subset=['wind\_15min\_ago','wind\_24hr\_ago','avg\_expected\_wind'])  features = ['hour', 'day\_of\_week', 'month', 'wind\_15min\_ago', 'wind\_24hr\_ago']  target = 'Germany\_wind\_generation\_actual'  X = data[features]  y = data[target]  split\_data = int(len(X) \* 0.8)  X\_train = X.iloc[:split\_data]  X\_test = X.iloc[split\_data:]  y\_train = y.iloc[:split\_data]  y\_test = y.iloc[split\_data:]  rf\_regression\_model = RandomForestRegressor(n\_estimators=100,max\_depth=10, random\_state=42)  rf\_regression\_model.fit(X\_train, y\_train)  y\_pred = rf\_regression\_model.predict(X\_test)  mae = mean\_absolute\_error(y\_test, y\_pred)  print("mean\_absolute\_error:")  print(mae)  rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)  print("mean\_squared\_error:")  print(rmse)  #calculating avgerage actual wind generation  mean\_actual = y\_test.mean()  print(mean\_actual)  #Percentage of mae  Percentage\_mae=(mae / mean\_actual) \* 100  print(round(Percentage\_mae,2))  #percentage of rmse  percentage\_rmse=(rmse / mean\_actual) \* 100  print(round(percentage\_rmse,2))  plt.figure(figsize=(12, 6))  plt.plot(y\_test.index[:100], y\_test.values[:100], label='Actual')  plt.plot(y\_test.index[:100], y\_pred[:100], label='Predicted')  plt.title("Next-Day Wind Generation: Actual vs Predicted (Germany)")  plt.xlabel("Timestamp")  plt.ylabel("Wind Generation (MW)")  plt.legend()  plt.grid(True)  plt.show()  #Shortfall Prediction  data['Germany\_shortfall'] = data['Germany\_wind\_generation\_actual'] < (data['avg\_expected\_wind'] \* 0.8)  data['Germany\_shortfall']  #1 = Shortfall, 0 = Normal  data['Germany\_shortfall'] = (data['Germany\_wind\_generation\_actual'] < (data['avg\_expected\_wind'] \* 0.8)).astype(int)  print(data['Germany\_shortfall'])  wind\_shortfall\_features = ['hour', 'day\_of\_week', 'month', 'wind\_15min\_ago', 'wind\_24hr\_ago', 'avg\_expected\_wind']  wind\_shortfall\_target = 'Germany\_shortfall'  X\_wind\_shortfall = data[wind\_shortfall\_features]  y\_wind\_shortfall = data[wind\_shortfall\_target]  split\_data = int(len(X\_wind\_shortfall) \* 0.8)  X\_train\_wind\_shortfall = X\_wind\_shortfall.iloc[:split\_data]  X\_test\_wind\_shortfall = X\_wind\_shortfall.iloc[split\_data:]  y\_train\_wind\_shortfall = y\_wind\_shortfall.iloc[:split\_data]  y\_test\_wind\_shortfall = y\_wind\_shortfall.iloc[split\_data:]  rf\_classifier\_model = RandomForestClassifier(n\_estimators=100,max\_depth=5, max\_features='sqrt',random\_state=42)  rf\_classifier\_model.fit(X\_train\_wind\_shortfall, y\_train\_wind\_shortfall)  y\_pred\_wind\_shortfall = rf\_classifier\_model.predict(X\_test\_wind\_shortfall)  #print(confusion\_matrix(y\_test, y\_pred))  print(classification\_report(y\_test\_wind\_shortfall, y\_pred\_wind\_shortfall))  #Plot the confusion matrix  metrics.ConfusionMatrixDisplay.from\_predictions(y\_test\_wind\_shortfall, y\_pred\_wind\_shortfall)  plt.show()  #Creating a smaller data frame with two variables  data\_outlier = data[['Germany\_wind\_generation\_actual','Germany\_load\_actual\_entsoe\_transparency']]  data\_outlier.describe()  #Initialize DBSCAN model  dbscanModel = DBSCAN(eps=1500, min\_samples=5)  #Fit the model  dbscanModel = dbscanModel.fit(data\_outlier)  clusters= dbscanModel.fit\_predict(data\_outlier)  clusters = pd.Categorical(clusters)  data\_outlier['Cluster'] = clusters  print(data\_outlier['Cluster'].value\_counts())  #Visualize scaled outliers   |  | | --- | | p = sns.scatterplot(data=data\_outlier, x='Germany\_wind\_generation\_actual',y='Germany\_load\_actual\_entsoe\_transparency', hue=clusters)  p.set\_xlabel('Germany\_wind\_generation\_actual', fontsize=15);  p.set\_ylabel('Germany\_load\_actual\_\_entsoe\_transparency', fontsize=15)  plt.title("Outliers detection in German Wind Actual vs Load")  plt.show() | |

**11. References**

Open Power System Data. (2020). *Renewable power generation: Data package*. Retrieved from <https://data.open-power-system-data.org/time_series/>

Rasouli, A., Arabzad, S., & Askarany, D. (2023). Machine learning techniques for renewable energy prediction: A review. Renewable and Sustainable Energy Reviews, 174, 113104. <https://doi.org/10.1016/j.rser.2022.113104>

Sabbaghi, S., & Barzegar, M. (2022). A comparative study of machine learning models for wind power prediction: Random forest, gradient boosting, and LSTM approaches. *Renewable and Sustainable Energy Reviews, 161*, 112348. <https://doi.org/10.1016/j.rser.2022.112348>

Fang, C., Zhang, J., & Chen, W. (2023). Anomaly detection in renewable energy systems using DBSCAN-based clustering. *Renewable Energy, 206*, 1115–1126. <https://doi.org/10.1016/j.renene.2023.02.041>

Waskom, M. (2021). seaborn: Statistical data visualization. *Journal of Open Source Software, 6*(60), 3021. <https://doi.org/10.21105/joss.03021>

Data Science Process Alliance. (2024, December 9). *What is CRISP-DM?* DataScience-PM. Retrieved April 27, 2025, from <https://www.datascience-pm.com/crisp-dm-2/>

Basic Feature Engineering With Time Series Data in Python This tutorial explains how to transform time series data into a supervised learning problem using lag features and rolling statistics. <https://machinelearningmastery.com/basic-feature-engineering-time-series-data-python/>

Practical Guide for Feature Engineering of Time Series Data This guide covers creating lagged variables, moving window statistics, and time-based features for time series forecasting. <https://dotdata.com/blog/practical-guide-for-feature-engineering-of-time-series-data/>

.shift() – Creating Lag Features The .shift() function shifts the index by a specified number of periods, allowing you to create lag features. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shift.html>

.rolling() – Calculating Rolling Statistics The .rolling() function provides rolling window calculations, enabling you to compute statistics like moving averages over a specified window. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html>