CS-GY 6513 BIG DATA

Predictive Modeling for Solar Energy Production in Europe

Code Repository: https://github.com/TekuriSaiAkhil/BIG-DATA

Siri Chandana Errabelli (se2596)
Sai Akhil Tekuri (st5050)
Sai Aravind Yanamadala (sy3902)

Problem Statement:

The objective of our project is to develop a predictive model capable of forecasting solar energy output in real-time. This model leverages extensive large datasets combining historical weather conditions, solar irradiance, and past solar energy production data, sourced from Open Power System Data. This is beneficial as it can help optimize the integration of solar energy into the power grid, improve the management of energy resources, and reduce reliance on non-renewable energy sources. This project addresses a significant big data challenge, requiring robust data handling, real-time analytical capabilities, and advanced predictive modeling techniques.

Goals:

- **Data Preprocessing and Feature Extraction Using PySpark:**
  - Utilize PySpark to handle the preprocessing of historical datasets, including weather conditions, solar irradiance, and past solar energy outputs, and Extract relevant features.
- **Development of a Predictive Model with Spark ML:**
  - Use the Spark ML libraries to build a Machine learning model capable of forecasting solar energy production. This model will be trained on the processed historical data.
- **Real-Time Prediction Using Kafka:**
  - Implement Kafka for streaming real-time weather data from API calls and make real-time predictions of solar energy generation.
- **Data Storage:**
  - Store the real-time predictions in a database system, which will record both the predictions and corresponding timestamps.
- **Data Visualization with Tableau:**
  - Utilize Tableau to create dynamic dashboards that visualize both real-time and historical predictions of solar energy generation

Approach:

To achieve these goals we need a predictive model to predict solar energy production based on weather features and a streaming pipeline to get real-time weather data and produce real-time solar energy predictions based on our prediction model.

Predictive Model Development:

To build a predictive model that predicts solar energy produced we used historical weather data, solar irradiance data, and actual solar energy generated data from 2015 to 2019. The data is recorded at 5-minute intervals. So we have large enough data to build a good prediction model.

1. Data Collection and Preprocessing:

To develop a robust predictive model for solar energy production, we used historical data on weather conditions, solar irradiance, and actual solar energy generation from the years 2015 through 2019 recorded at five-minute intervals. This data is extracted from the Open Power System dataset.

Pre-processing steps included handling missing values and eliminating outliers. Feature extraction involved merging weather information with solar output data based on timestamps. Feature normalization was done to standardize the range of data values for effective model training.

The pre-processed dataset consisted of 500,000 records and comprised the following features, aligned temporally with the target variable **DE_solar_generation_actual**

DE_temperature: Ambient temperature recorded in Germany.
DE_radiation_direct_horizontal: Direct horizontal solar radiation levels.
DE_radiation_diffuse_horizontal: Diffuse horizontal solar radiation levels.

After pre-processing:
1. **Number of records**: 500,000
2. **Features**: 1. DE_temperature
   2. DE_radiation_direct_horizontal
   3. DE_radiation_diffuse_horizontal
3. **Prediction**: DE_solar_generation_actual

This processed dataset was preserved in Parquet format, optimizing both storage efficiency and retrieval speed for subsequent analytical procedures.

2. Model Training:

The training process began with the loading of historical data stored in Parquet format. The dataset is split into 80% training data and 20% test data.

We used Spark MLlib to take advantage of its support for parallel processing, essential for managing the scale and complexity of our data. Three regression algorithms were tested: Linear Regression, Random Forest Regressor, and Gradient Boosting Regressor with hyperparameter tuning. Model performance is measured using Root Mean Square Error (RMSE). The Gradient Boosting Regressor along with hyperparameter tuning was found as the superior model, because of its lowest RMSE on both the training and test datasets.

The Gradient Boosting Regressor achieved an RMSE of 0.23657 on the test data, indicating high predictive accuracy.

Hyperparameters: 1) Best value for numTrees: **150**
2) Best value for maxDepth: **15**

Real-Time Predictions:

1. Data Streaming and Processing:

To facilitate real-time solar energy generation predictions using solar irradiance and weather data, we utilized OpenWeatherMap's Solar Irradiance API to capture various solar and weather metrics. The data retrieval is automated to occur every five minutes, harnessing Kafka for efficient data streaming. Specifically, in the Kafka producer, an API call fetches real-time weather data, including Diffuse Irradiance (DHI), Total Irradiance (GHI), and Air Temperature at 2m above the surface level (AIR_Temp).

Essential features are extracted and formatted during data processing to meet the predictive model requirements. This includes calculating the Radiation Direct Horizontal by subtracting Diffuse Irradiance (DHI) from Total Irradiance (GHI). The data schema organized for further processing is structured as follows:

- DE_Temperature: Represents the air temperature
- DE_Radiation_direct_horizontal: Difference between GHI and DHI.
- DE_Radiation_diffuse_horizontal: Directly corresponds to the DHI value.

2. Real-time Predictive Modeling and Data Storage:

The preprocessed data is then received by the consumer in the form of a message where it gets decoded and split into different features, then our pre-trained machine learning model utilizes these features to predict solar power generation (denoted as DE_SOLAR_GENERATION in MegaWatts). This prediction pipeline is critical for adjusting and optimizing energy distribution in response to real-time environmental changes.

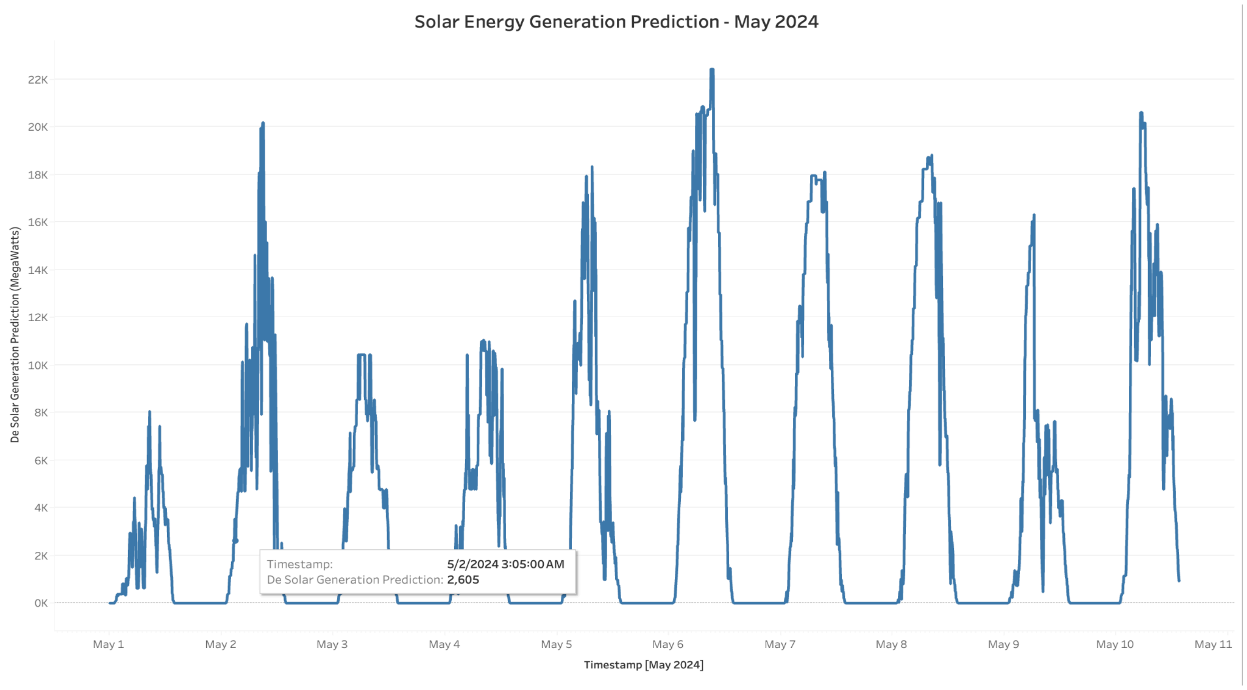
Finally, the predictive outputs, along with their respective timestamps, are stored in AWS RDS. This setup not only ensures data integrity and accessibility but also facilitates subsequent visualization and analysis through Tableau, allowing for a comprehensive understanding of solar generation patterns and their dependencies on weather conditions.

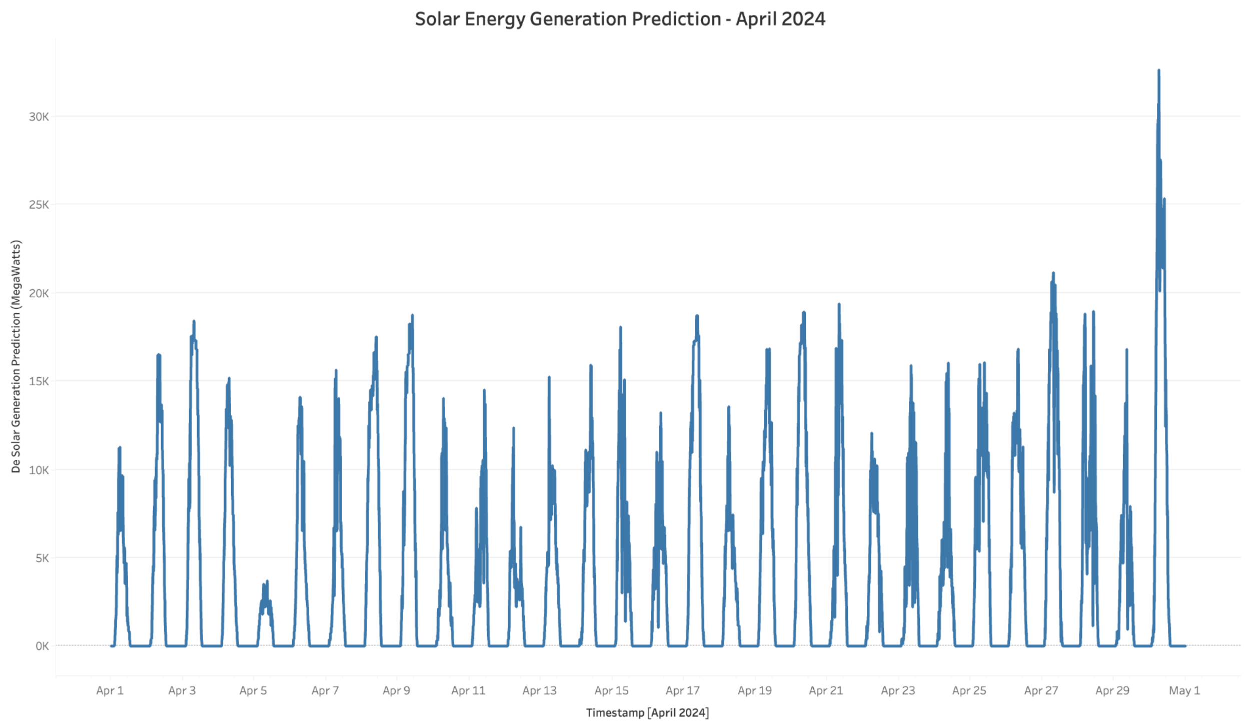
This streamlined and automated data flow enables effective real-time predictive modeling, crucial for operational efficiency in solar power management.

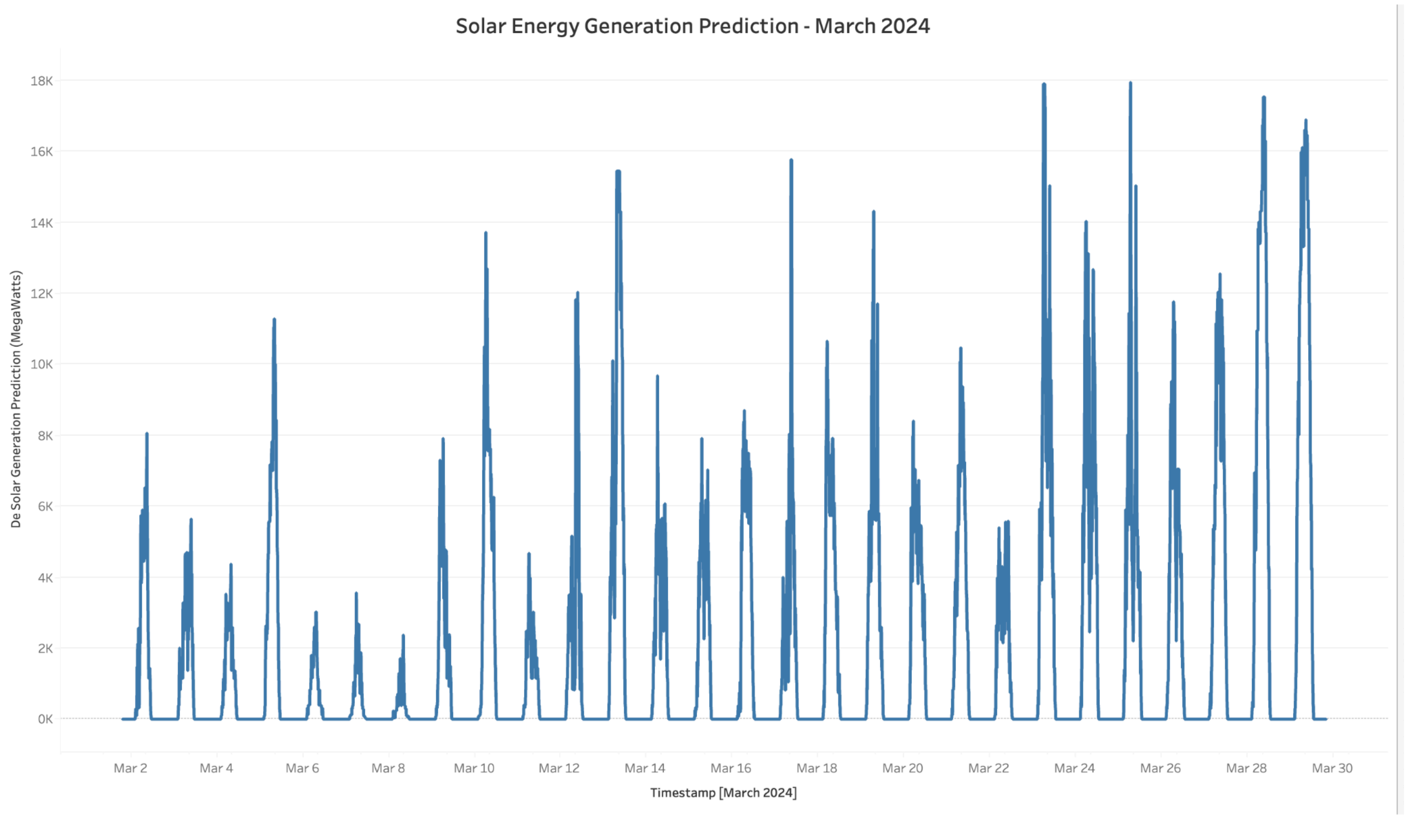
Visualization of Results:

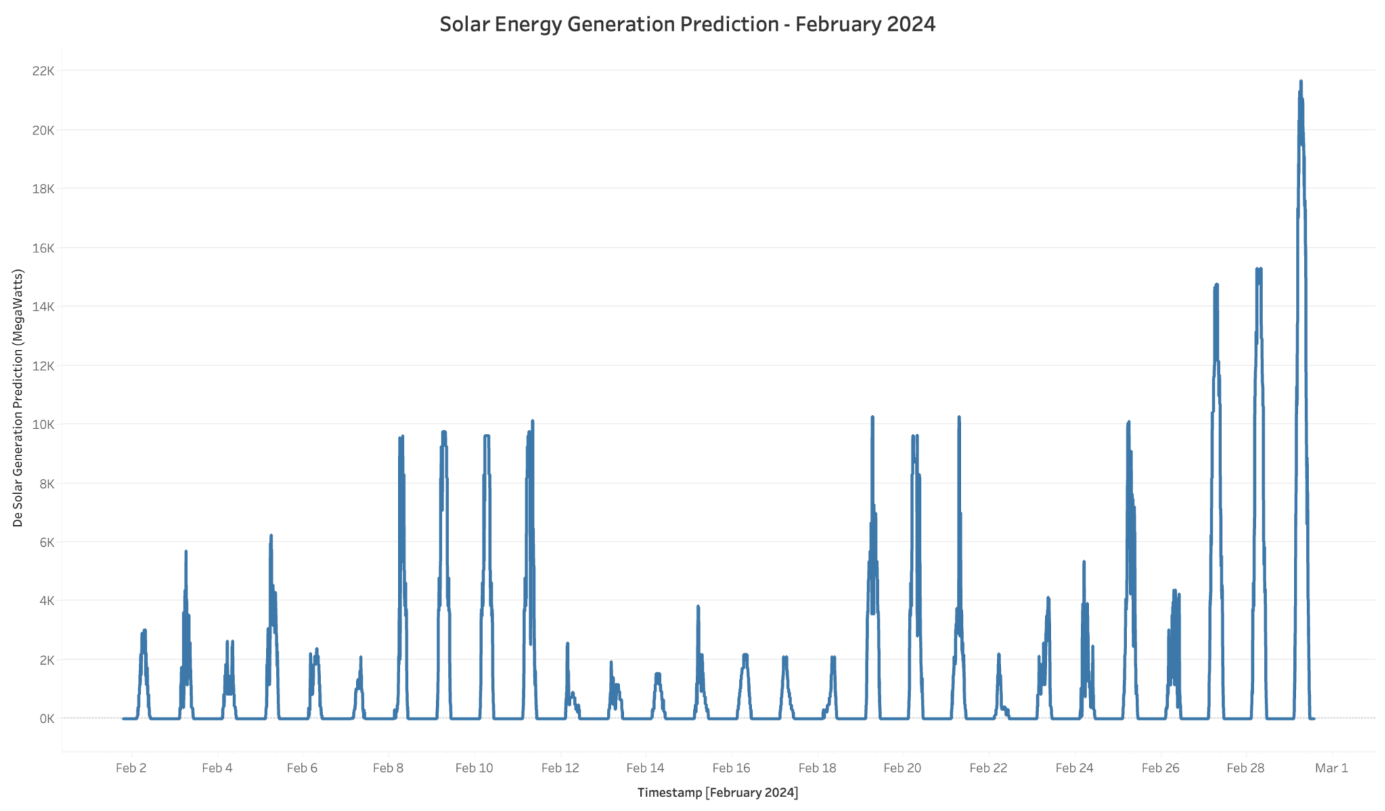
For visualizations, a live connection was established between AWS Relational Database Service (RDS) and Tableau to enable real-time visualization. This integration permitted the dashboard's continual refresh, reflecting new data entries in the database and offering an up-to-date depiction of expected solar energy outputs.

The Tableau dashboard shows a significant pattern in the solar energy generation anticipated by our model. Each day begins with a slow increase in solar generation, which rises to a peak about midday, coinciding with the highest solar irradiation. As the day advances, the generation decreases significantly, slowing off in the evening and eventually ceasing at night in the absence of solar radiation. This temporal trend is similar across the month, demonstrating the predictive model's stability and accuracy in capturing the dynamics of solar energy generation as driven by daily solar radiation patterns.






Solar Energy Generation Prediction - March 2024
De Solar Generation Prediction (MegaWatts)
18K
16K
14K
12K
10K
8K
6K
4K
2K
0K
Mar 2
Mar 4
Mar 6
Mar 8
Mar 10
Mar 12
Mar 14
Mar 16
Mar 18
Mar 20
Mar 22
Mar 24
Mar 26
Mar 28
Mar 30
Timestamp [March 2024]


Solar Energy Generation Prediction - February 2024
De Solar Generation Prediction (MegaWatts)
22K
20K
18K
16K
14K
12K
10K
8K
6K
4K
2K
0K
Feb 2
Feb 4
Feb 6
Feb 8
Feb 10
Feb 12
Feb 14
Feb 16
Feb 18
Feb 20
Feb 22
Feb 24
Feb 26
Feb 28
Mar 1
Timestamp [February 2024]

Technologies:

1. Parquet
2. Spark
3. Spark MLlib
4. Kafka
5. AWS RDS
6. Tableau
7. Open Weather API

References:

1. Data source: Open Power System Data
2. Weather API: Weather API - OpenWeatherMap
3. Spark : PySpark Overview — PySpark master documentation
4. SparkML: GBTRegressionModel — PySpark 3.1.2 documentation
5. Kafka: Kafka 3.7 Documentation