Model Validation Report

Predict Energy Behavior of Consumer/Producer with Solar Panels

This document provides an overview of a model validation and analysis process for the project aimed at predicting the energy behavior of consumer/producers (prosumers) with solar panels in Estonia. The received project, carried out by Enefit, a major energy company in the Baltic region, seeks to develop a prosumer energy forecast model to reduce costs associated with energy imbalances, improve grid reliability, and promote sustainable energy practices. The goal is to create an efficient and sustainable integration of prosumers into the energy system, potentially encouraging more consumers to become prosumers and supporting renewable energy production and use. The validation process includes comparing the challenger model (using PiML) to the submitted model, assessing the performance of the submitted model post-training on the inference dataset, and providing documentation and supporting code on validation findings

Challenger Model (PiML) Comparison

The challenger model – XGBoost is developed using PiML and compared against the submitted model. The challenger model is built using the same feature set as the rogue model to create a like–to–like comparisons. Both the challenger and Rouge models have comparable performance with R2 values of 0.99 and 0.98 respectively. Challenger model is better in terms of Mean Absolute Error (MAE) as it has MAE of 0.0021 as compared to 7.99 of the Rogue Model.

Submitted Model Performance on Inference Dataset

The submitted model is trained on a comprehensive dataset containing various features related to energy consumption and production. The model training includes the use of LightGBM, and feature importance analysis is conducted to identify the most influential features.

The training dataset is preprocessed and features are engineered using functions for timestamp conversion, date processing, and feature extraction. Data preparation involves reading the dataset, processing dates, and extracting relevant features – Top 20 features are selected in the Rogue model. Additionally, external datasets, such as electricity prices, gas prices, and client information, are integrated into the model.

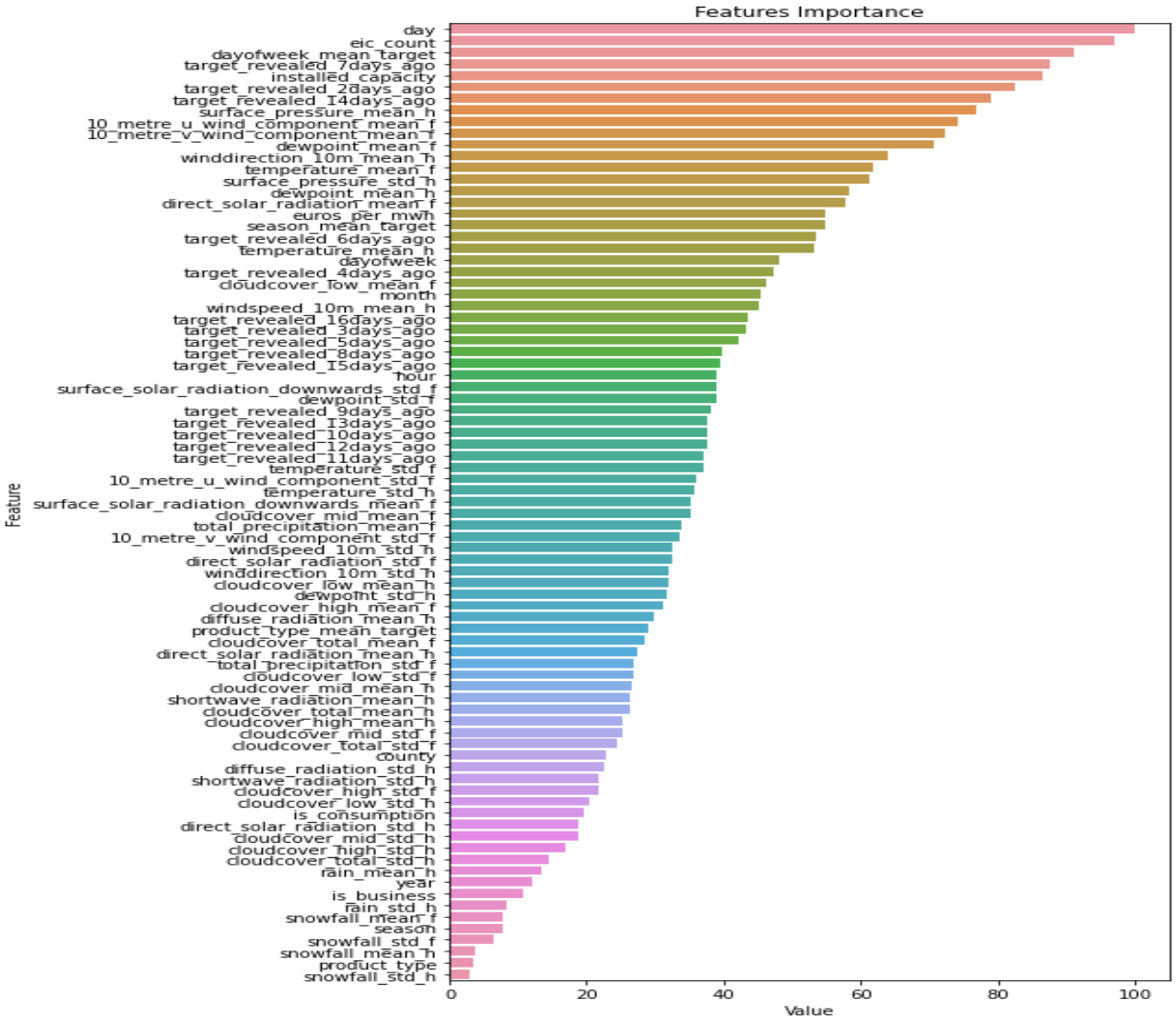
The model training process involves training on the 60% of dataset and evaluating performance on both training and validation sets. Feature importance is analyzed using both LGBM's built-in feature importance and permutation-based feature importance.

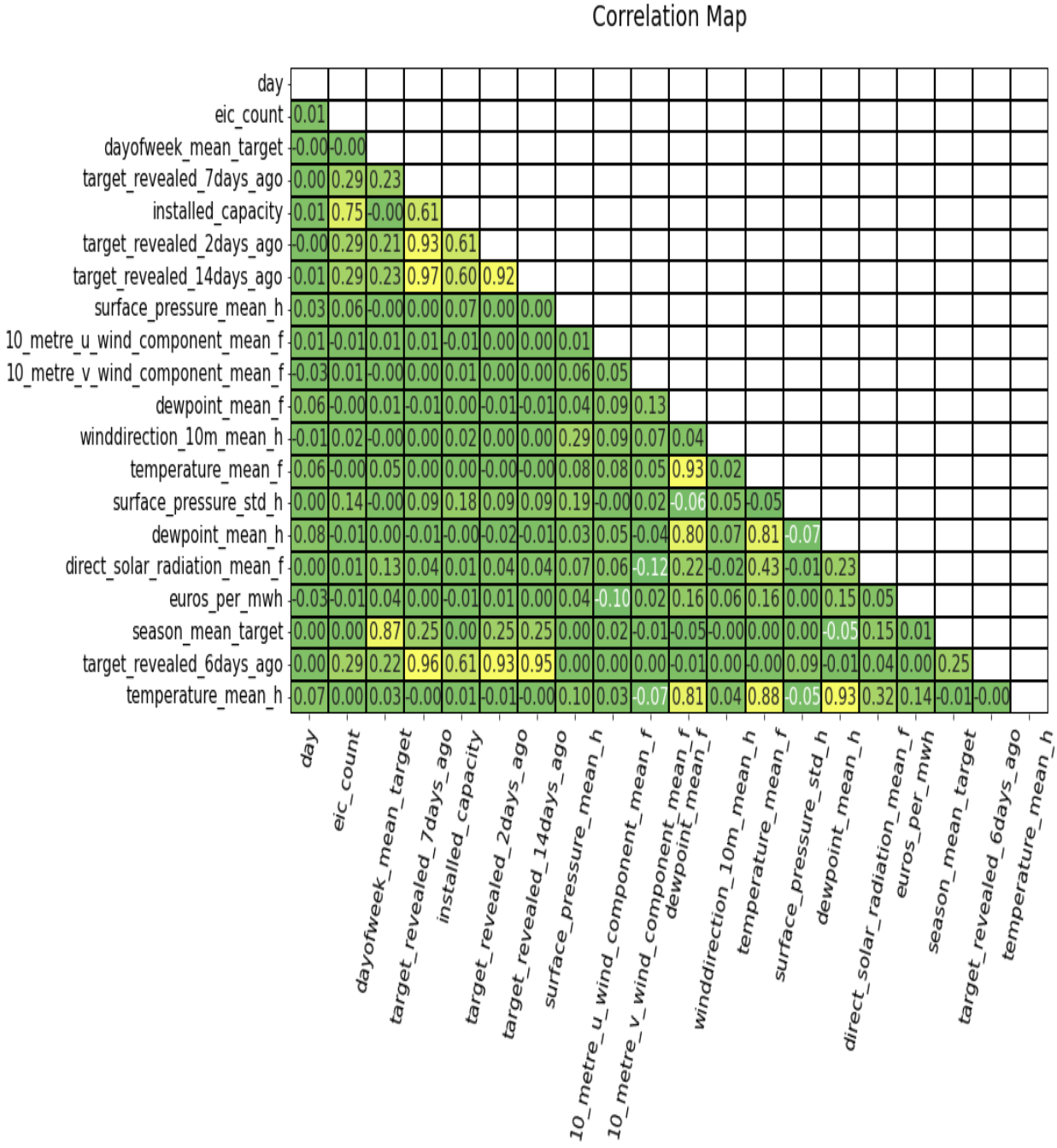
Feature Importance Analysis

Feature importance analysis is conducted using the LightGBM model. The top features are identified based on their importance scores. Additionally, permutation-based feature importance is computed to validate and enhance the understanding of feature importance.

The top 20 features are selected for further analysis, and their correlation is visualized using a heatmap. Understanding feature correlations is crucial for identifying potential redundancies and ensuring that the selected features contribute uniquely to the model.

Correlation heatmap is visualized to understand if there are any features with multicollinearity among them.




Correlation Map
day
eic_count
dayofweek_mean_target
target_revealed_7days_ago
installed_capacity
target_revealed_2days_ago
target_revealed_14days_ago
surface_pressure_mean_h
10_metre_u_wind_component_mean_f
10_metre_v_wind_component_mean_f
dewpoint_mean_f
winddirection_10m_mean_h
temperature_mean_f
surface_pressure_std_h
dewpoint_mean_h
direct_solar_radiation_mean_f
euros_per_mwh
season_mean_target
target_revealed_6days_ago
temperature_mean_h

Model Training and Evaluation

The final model is trained using the top 20 features. The dataset is split into training, validation, and test sets. The model is trained using LightGBM with the following hyperparameters – learning rate of 0.08, 3000 estimators, alpha – 2, lambda – 0 2 jobs and a random state of 12. specified hyperparameters, and its performance is evaluated on both the training and test sets. The mean absolute error (MAE) is used as the evaluation metric.

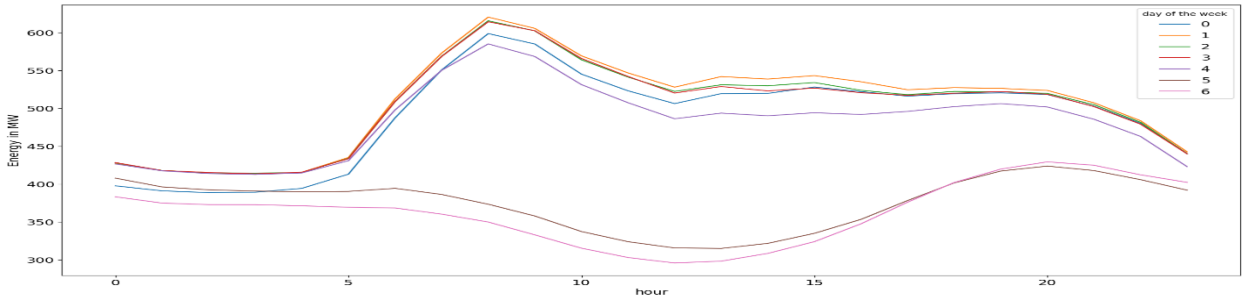
Validation Issues found in the code:

1. **Undefined Debug Variable:**
   Issue: The code uses an undefined debug variable, leading to potential execution errors or unexpected behavior during data loading.
   Impact: This could limit the model's applicability in varied scenarios and affect reproducibility.
   Suggestion: Clearly define the debug variable at the start, or modify the code to function without it, ensuring consistency and clarity in data processing.

2. **Datetime Formatting**:
   Issue: The model currently uses hard-coded values for datetime formatting, which could cause issues if the input datetime format varies.
   Risk: This rigidity may lead to errors or misinterpretations of temporal data, affecting model accuracy.
   Suggestion: Implement a more flexible datetime parsing method that can adapt to different formats, enhancing the model's robustness and adaptability.

3. **Handling Outliers in Target Distribution:**
   Issue: Outliers in the target variable may skew the data distribution, leading to biased predictions.
   Effect: The presence of outliers can significantly impact the model's training process and its ability to generalize.
   Suggestion: Use robust statistical methods to identify and treat outliers, ensuring a more representative data distribution for the model.

4. **Day of the Week Plot Clarification:**
   Issue: The plot uses numerical day representations, making it difficult to interpret and understand weekly patterns.
   Result: This lack of clarity can hinder insights from the data and affect decision-making.
   Suggestion: Replace numerical day labels with actual day names (e.g., Monday, Tuesday) for immediate recognition and clearer data insights.



5. **<u>Season & Product Type Descriptors:</u>**
   Issue: Seasons and product types are currently represented numerically, which lacks intuitive meaning.
   Impact: This abstract representation can confuse users and obscure meaningful patterns in the data.
   Suggestion: Introduce descriptive labels or a mapping dictionary for these features to make the data more accessible and interpretable.

6. **<u>Imputing Zero Values in Key Features:</u>**
   Issue: Features such as temperature and energy price are imputed with zeros, which is unrealistic.
   Consequence: This could lead to significant distortions in predictive accuracy.
   Suggestion: Investigate the reasons for zero values and apply appropriate imputation strategies, like median or mean imputation, or more sophisticated methods like KNN imputation.

7. **<u>Train-Test Split Optimization:</u>**
   Issue: The 60%/40% train-test split may not provide enough training data, potentially causing underfitting.
   Risk: Limits the model's ability to learn from the data fully, impacting predictive performance.
   Suggestion: Consider using a larger portion of data for training or implementing k-fold cross-validation to ensure the model's robustness and generalizability.

8. **Systematic Hyperparameter Tuning:**
   Issue: The model's hyperparameters appear to be chosen arbitrarily.
   Impact: This can lead to suboptimal model performance.
   Suggestion: Employ systematic hyperparameter tuning methods like grid search or randomized search to find the most effective hyperparameters, enhancing model accuracy.

9. **Revisiting Feature Selection:**
   Issue: The permutation-based feature selection code runs indefinitely and the code crashes
   Effect: This could lead to skipping the feature engineering part and affect code reproducibility and issues in creating production ready code.
   Suggestion: Create a permutation-based feature selection code that gives output and helps select important features.

10. **Model Accuracy and Forecasting Approach:**
    Issue: The current model uses regression-based algorithms for time series forecasting.
    Concern: Indicates potential shortcomings in capturing time-dependent patterns and trends.
    Suggestion: Explore specialized time series forecasting models like ARIMA, SARIMA, or LSTM, which might better capture temporal dynamics and improve predictive accuracy.

Conclusion and Recommendations:

The validation process provides insights into the model's performance, feature importance, and potential areas for improvement.

Recommendations for further improvements or considerations may include exploring additional feature engineering, tuning hyperparameters, or experimenting with different models like more sophisticated time-series forecasting models to achieve better predictive performance. It's also essential to continue monitoring the model's performance over time and update it as needed based on new data or changing patterns in the energy domain.

The provided code covers the key aspects of data preprocessing, feature engineering, model training, and evaluation.