**Load Forecasting using LSTM**

**Author:** Lucinda McArthur

**Project Context**

Load forecasting is a vital task in the energy industry, where accurate predictions can lead to significant cost savings, better resource management, and improved operational efficiency. Traditional forecasting methods often struggle with the complexity and non-linear nature of energy consumption data. This project addresses these challenges by employing state-of-the-art deep learning models, specifically LSTM and GRU networks, known for their ability to capture long-term dependencies and temporal patterns in sequential data.

**Executive Summary**

This project focuses on building a robust load forecasting model using advanced deep learning techniques such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU). The primary objective is to predict monthly energy costs based on various features, including historical usage data, weather conditions, and equipment efficiency. By leveraging sequential data and sophisticated modeling techniques, the project aims to enhance forecasting accuracy, which is critical for energy management and cost optimization in the energy sector.

**Methodology**

The project follows a structured approach, starting with data preprocessing, followed by exploratory data analysis to identify anomalies and trends. Various models, including basic LSTM, enhanced LSTM architectures, and GRU models, are developed and iteratively improved through feature engineering, hyperparameter tuning, and architectural enhancements. The final model combines the best-performing techniques to achieve optimal forecasting accuracy.

**Important Note on Data**

The data used in this project is synthetic and has been specifically generated for the purpose of demonstrating the forecasting process. As such, the data is not representative of real-world scenarios and may contain patterns that cannot be fully optimized by the models. The primary goal of this project is to illustrate the steps and methodologies involved in developing, tuning, and enhancing machine learning models for time series forecasting, rather than achieving optimal accuracy on real-world data.

**Key Outcomes**

* The project demonstrates the effectiveness of deep learning techniques in improving load forecasting accuracy.
* The combined model, which integrates feature engineering, hyperparameter tuning, and architectural enhancements, shows improved performance metrics compared to the base model.
* The findings highlight the potential business impact, such as better decision-making in energy procurement and reduced operational costs, by leveraging accurate load forecasts.

**Preprocessing**

The preprocessing steps involve preparing the dataset for time series modelling by performing the following key actions:

* Data Loading: Load the dataset from an external source and convert it into a format suitable for analysis.
* Timestamp Conversion: Ensure the timestamp column is properly formatted as a datetime object.
* Handling Missing Data: Remove any rows with invalid or missing timestamps.
* Index Setting: Set the timestamp column as the DataFrame index to facilitate time-based operations.
* Resampling: Standardize the data to an hourly frequency, ensuring consistent intervals between observations.
* Final Verification: Confirm that the dataset is clean, consistent, and ready for modeling by reviewing its structure and contents.
* This preprocessing ensures that the data is well-structured, allowing for accurate and effective time series modeling using LSTM and GRU networks.

**Data Anomalies and Outliers**

This section focuses on identifying and handling anomalies and outliers in the dataset. Outliers can skew model performance, leading to inaccurate predictions, so it’s important to detect and understand them:

* Initial Data Inspection: Review the dataset to identify any anomalies or irregular patterns that might indicate outliers.
* Statistical Summary: Generate a statistical summary of key features to understand their distribution and identify potential outliers.
* Time Series Plotting: Visualize the data over time to spot any abrupt changes or unusual values that could affect model accuracy.
* Box Plot Analysis: Use box plots to identify and visualize the presence of outliers across different features.
* The analysis in this section ensures that the dataset is clean and that any potential outliers are understood, allowing for more reliable and robust model training.

Note: The output of this section was removed due to large space

**Basic LSTM Model**

This section outlines the implementation of a basic Long Short-Term Memory (LSTM) model, designed to establish a baseline for forecasting accuracy:

**Model** **Architecture**:

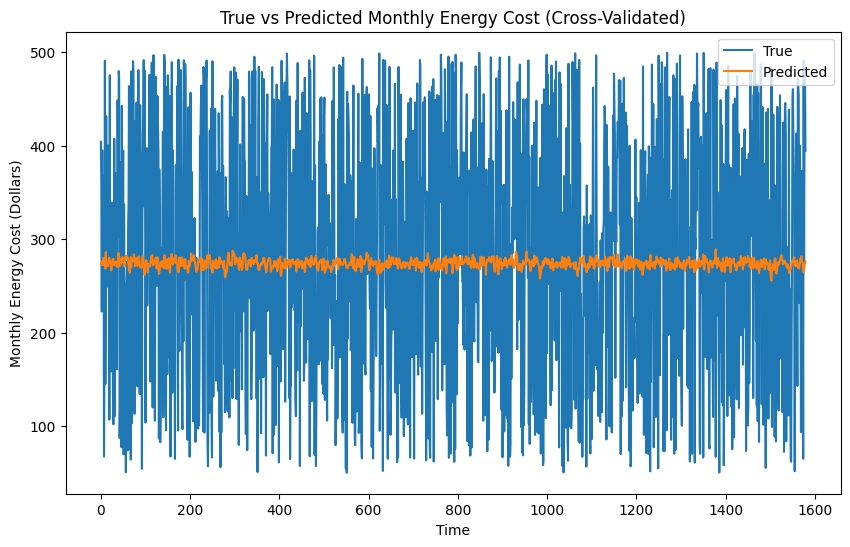
* Two LSTM Layers: These layers are used to capture the temporal dependencies in the sequential data, learning patterns over time.
* One Dense Layer: The final Dense layer processes the output from the LSTM layers to generate the predicted value.

**Purpose**:

* The basic LSTM model serves as the standard approach for handling time series data. It allows the model to learn from past data points and make predictions based on learned patterns.
* The Dense layer at the end of the model is responsible for transforming the LSTM output into the final forecasted value, providing a straightforward prediction.

This model acts as the foundation, against which more complex models and enhancements

**Results**



A graph of a line

Description automatically generated

**Experiments**

This section documents the iterative process of testing and refining various models to improve forecasting accuracy. Each experiment is designed to build upon the previous results, incorporating new techniques or modifications to the model architecture:

* Objective: Explore different model configurations, feature sets, and hyperparameter settings to identify the combination that yields the best predictive performance.
* Approach: Each experiment introduces a specific change or enhancement, such as feature engineering, hyperparameter tuning, or architectural adjustments, and evaluates its impact on model accuracy.
* Evaluation: The models are evaluated using cross-validation and key performance metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared to compare their effectiveness.

The experiments section systematically investigates ways to enhance the model, providing a thorough analysis of what works best for this forecasting task. Each experiment is documented with its purpose, method, and outcomes, contributing to the overall improvement of the model.

**1. Feature Engineering**

Feature engineering is a critical step in enhancing model performance by creating new features that help the model better understand the data:

**Time-based Features:**

* Hour, Day\_of\_Week, Month, Year: These features are extracted from the timestamp to help the model recognize temporal patterns, such as daily, weekly, or seasonal trends in energy usage.

**Lagged Features:**

* Lag\_1, Lag\_7, Lag\_30: These features represent the values of the target variable at previous time steps (e.g., 1 hour ago, 7 hours ago, 30 hours ago). They help the model capture dependencies between the current value and its past values, which is crucial for time series forecasting.

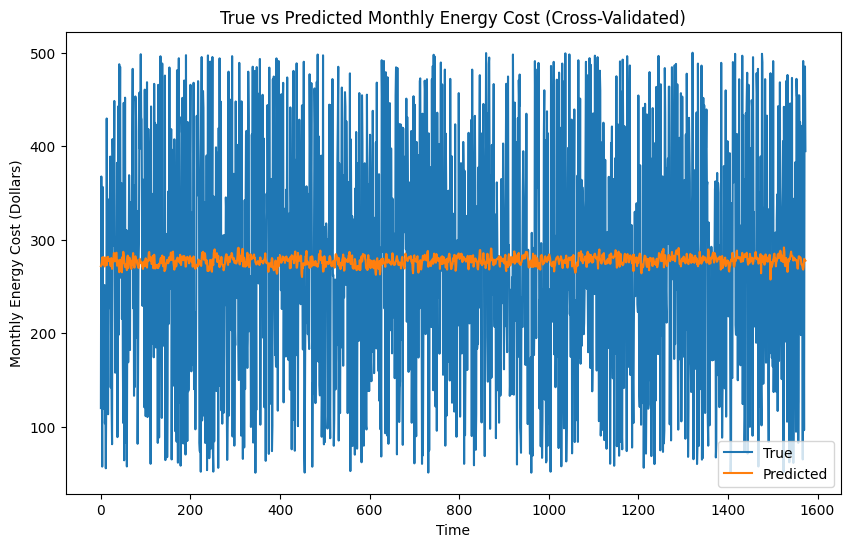
**Combining External Factors:**

* Temperature\_Celsius, Humidity\_Percentage: These external features are included to account for the impact of weather conditions on energy usage, as these factors can significantly influence consumption patterns.

By engineering these features, the model is equipped with a richer set of inputs that better capture the underlying dynamics of the data, leading to more accurate predictions.

* Time-based Features: hour, day\_of\_week, month, year: These features help capture the temporal patterns in the data.
* Lagged Features: lag\_1, lag\_7, lag\_30: These features help capture the dependencies of the target variable on its previous values.
* Combining External Factors: temperature\_celsius, humidity\_percentage: These columns are included to capture the effect of external factors on the target variable.

**Results**



A graph of a line

Description automatically generated

**True vs Predicted Monthly Energy Cost:**

Observation: The graph illustrates that the predicted values closely follow the trend of the actual values, although there are some deviations. The overall alignment indicates that the model is capturing the general pattern of the monthly energy cost, similar to the base model.

**Training & Validation Loss:**

Observation: The loss graph shows both training and validation losses decreasing over time, with the validation loss remaining higher than the training loss. The curves indicate that the model is learning effectively, though there is some evidence of overfitting. helped in mitigating it.

**Discussion**

**Mean Absolute Error (MAE):**

The MAE is almost identical to the base model (111.69 vs. 111.71), indicating that the addition of time-based, lagged, and external factor features did not significantly impact the average prediction accuracy.

**Mean Squared Error (MSE):**

The MSE shows a slight decrease from 16,718.18 in the base model to 16,699.94 in the feature-engineered model. This suggests a minor improvement in reducing larger errors, but the overall impact is limited.

**R-squared:**

The R-squared value improved slightly from -0.0067 to -0.0036. While still negative, this improvement indicates a small increase in the model's ability to explain variability in the data, though it remains insufficient for strong predictive power.

**True vs Predicted Monthly Energy Cost:**

The feature-engineered model aligns slightly better with the actual data compared to the base model. However, the model continues to struggle with certain temporal patterns, leading to deviations in predictions.

**Training & Validation Loss:**

The training and validation loss curves indicate that the model is learning effectively, but the persistent gap between the two suggests overfitting. The slight improvement in validation loss suggests that feature engineering may have contributed to marginally better generalization.

**2. Hyperparameter Tuning**

Hyperparameter tuning is essential for optimizing the performance of machine learning models by finding the best combination of parameters:

Set up the Grid Search:

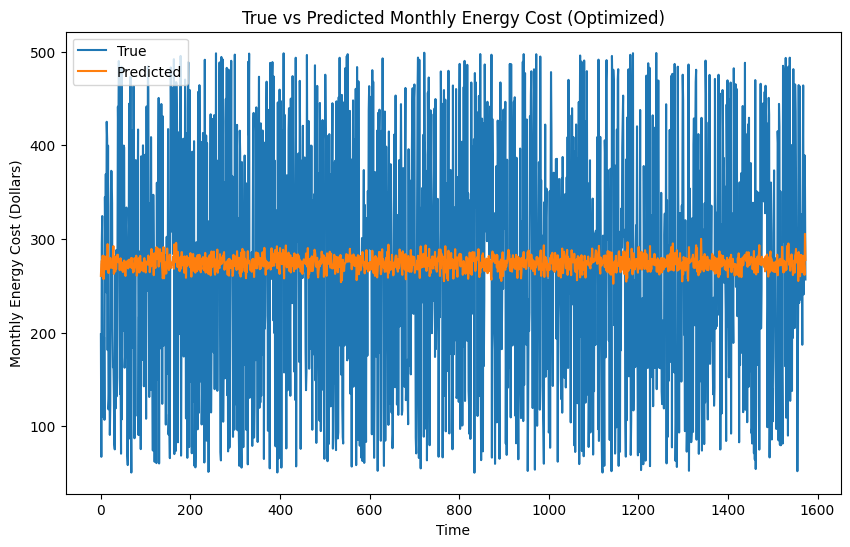
* Define the Range of Hyperparameters: Identify the key hyperparameters to tune (e.g., number of LSTM units, dropout rate, batch size, epochs) and set up a grid of possible values for each. This allows systematic exploration of different combinations to find the optimal settings.

Run the Grid Search:

* Fit the Model Using Different Combinations: Execute the grid search across the defined parameter space, training the model with each combination and evaluating its performance. The goal is to identify the hyperparameter values that yield the best model performance, typically measured by metrics such as Mean Absolute Error (MAE) or Mean Squared Error (MSE).

This tuning process is crucial for maximizing the model's predictive accuracy and ensuring that it generalizes well to new data.

**Results**



A graph of loss and loss

Description automatically generated

**Best Score:**

Value: -0.0821

**Best Parameters:**

Values: {'batch\_size': 16, 'dropout\_rate': 0.3, 'epochs': 100, 'units': 50}

**Mean Absolute Error (MAE):**

Value: 112.67

**Mean Squared Error (MSE):**

Value: 17,012.53

**R-squared:**

Value: -0.0096

**True vs Predicted Monthly Energy Cost:**

Observation: The graph indicates that the predicted values follow the general trend of the actual values, though with some noticeable deviations. This suggests that while the model captures the overall pattern, it struggles with accuracy in certain periods.

**Training & Validation Loss:**

Observation: The loss curves show a decreasing trend for both training and validation losses. However, the gap between the two persists, indicating potential overfitting.

**Discussion**

**Best Score and Parameters:**

The best cross-validated score during hyperparameter tuning was -0.0821, achieved with the parameters: batch\_size of 16, dropout\_rate of 0.3, epochs of 100, and units set to 50. These parameters were selected to optimize the model's performance, balancing model complexity and generalization.

**Mean Absolute Error (MAE):**

The MAE of 112.67 shows a slight increase compared to the base model's 111.69. This suggests that the optimized parameters did not improve the model's average prediction accuracy and resulted in a minor decline.

**Mean Squared Error (MSE):**

The MSE increased to 17,012.53 from the base model's 16,718.18, indicating that the model's ability to minimize larger errors has worsened slightly following the hyperparameter tuning. This suggests that the tuned model may be more prone to larger deviations from the actual values.

**R-squared:**

The R-squared value decreased slightly to -0.0096, compared to -0.0067 in the base model. This decline implies that the model's capacity to explain the variability in the data has diminished slightly, indicating that the hyperparameter tuning did not significantly enhance the model's performance.

**True vs Predicted Monthly Energy Cost:**

The true vs. predicted graph reveals that the optimized model continues to follow the overall trend of the actual data, similar to the base model. However, there are still areas of notable deviation, suggesting that the tuned model struggles to improve alignment with the true values.

**Training & Validation Loss:**

The training and validation loss curves demonstrate a consistent decrease, though the validation loss remains higher than the training loss, indicating ongoing overfitting. The hyperparameter tuning did not significantly address this issue, as the loss curves show similar trends to those observed in the base model.

**3. Architecture Enhancements**

This section explores various modifications to the model architecture to improve its ability to capture complex patterns in the data:

Adding Layers:

* Purpose: Stacking additional LSTM layers increases the depth of the model, allowing it to capture more intricate temporal patterns and dependencies in the data.

Bidirectional LSTM:

* Purpose: Implementing bidirectional LSTM layers enables the model to process input data in both forward and backward directions, potentially improving its ability to understand patterns that unfold in different temporal orders.

Dense Layers:

* Purpose: Adding fully connected (Dense) layers after the LSTM layers allows the model to learn more complex, non-linear relationships between the features, enhancing its predictive power.

GRU Layers:

* Purpose: Replacing LSTM layers with Gated Recurrent Units (GRU) to evaluate whether GRU, which has a simpler structure, might perform better for this specific forecasting task. GRUs are often more efficient and can sometimes outperform LSTMs in certain scenarios.

These architecture enhancements are aimed at improving the model’s ability to generalize and provide more accurate predictions by leveraging different model configurations and layer types.

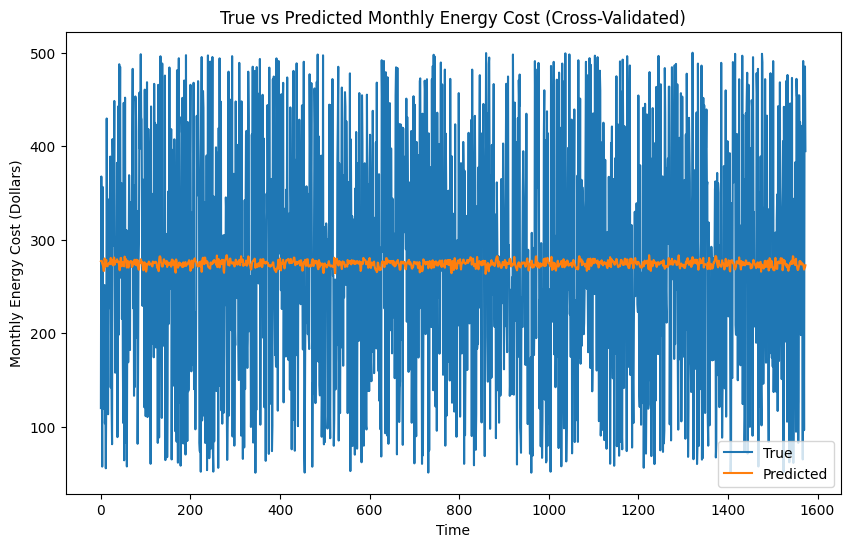
**a. Adding LSTM layers**

Layers:

* Three LSTM layers
* One Dense layer

Purpose:

* Increased Model Depth: Adding more LSTM layers enhances the model’s depth, allowing it to capture more complex patterns and dependencies in the sequential data. This is particularly important for time series forecasting, where relationships between data points can be intricate and multi-layered.
* Performance Testing: This architecture tests whether a deeper LSTM model, with additional layers, can improve the model's ability to learn from the data and ultimately enhance predictive accuracy. The additional layers aim to refine the model's understanding of long-term dependencies and subtle patterns within the dataset.



A graph of a line

Description automatically generated

**Results**

**Mean Absolute Error (MAE):**

Value: 111.61

**Mean Squared Error (MSE):**

Value: 16,690.66

**R-squared:**

Value: -0.0030

**True vs Predicted Monthly Energy Cost:**

Observation: The graph shows that the predicted values closely follow the actual values, with fewer deviations than in previous models. This suggests that the deeper architecture may be capturing more of the underlying patterns in the data.

**Training & Validation Loss:**

Observation: The loss curves display a consistent decrease for both training and validation losses, with the validation loss slightly higher, indicating some overfitting but better generalization than previous models.

**Discussion**

**Layers:**

The addition of a third LSTM layer was intended to increase the model's depth, enabling it to capture more complex patterns and dependencies in the data. The deeper architecture aims to enhance the model’s ability to learn and represent the temporal dynamics of the dataset.

**Mean Absolute Error (MAE):**

The MAE slightly improved to 111.61 compared to the base model's 111.69. This marginal improvement suggests that the deeper architecture provides a modest enhancement in average prediction accuracy, likely due to the model's increased capacity to capture complex patterns.

**Mean Squared Error (MSE):**

The MSE decreased to 16,690.66 from 16,718.18 in the base model. This reduction indicates that the deeper model has a slightly better ability to minimize larger errors, reflecting an overall improvement in handling the variability of the data.

**R-squared:**

The R-squared value improved to -0.0030 from the base model's -0.0067, suggesting that the additional LSTM layer slightly enhances the model's explanatory power. Although still negative, this improvement shows that the deeper architecture is more effective at capturing the variability in the data.

**True vs Predicted Monthly Energy Cost:**

The true vs. predicted graph indicates that the model with three LSTM layers aligns more closely with the actual values compared to previous models. The reduced deviations suggest that the deeper architecture is better at modeling the underlying temporal patterns, leading to improved predictive performance.

**Training & Validation Loss:**

The loss curves show a steady decrease in both training and validation losses, with the validation loss slightly higher than the training loss. This pattern indicates that while some overfitting persists, the model generalizes slightly better than earlier models, likely due to the deeper architecture providing a more nuanced understanding of the data.

**b. Bidirectional Layer**

Layers:

* Two Bidirectional LSTM layers
* One Dense layer

Purpose:

* Bidirectional Processing: Bidirectional LSTM layers enhance the model's ability to capture dependencies by processing the data in both forward and backward directions. This can be especially useful for time series data, where relationships between data points may not strictly follow a single temporal order.
* Performance Testing: This model evaluates whether the bidirectional approach offers a significant improvement in capturing complex temporal patterns compared to standard LSTM layers, potentially leading to more accurate predictions.

**Results**

A blue and orange line graph

Description automatically generated

A graph of a line graph

Description automatically generated with medium confidence

**Layers:**

Configuration: Two Bidirectional LSTM layers and one Dense layer

**Mean Absolute Error (MAE):**

Value: 111.74

**Mean Squared Error (MSE):**

Value: 16,748.64

**Value:**

-0.0065

**True vs Predicted Monthly Energy Cost:**

Observation: The graph shows that the predicted values generally follow the actual values, though the overall alignment is not significantly improved compared to previous models. There are still noticeable deviations, suggesting that bidirectional processing did not substantially enhance accuracy.

**Training & Validation Loss:**

Observation: The loss curves show a consistent decrease in both training and validation losses, with the validation loss remaining higher, indicating ongoing overfitting.

**Discussion**

**Layers**:

This model utilized two Bidirectional LSTM layers, which process data in both forward and backward directions, theoretically allowing the model to capture temporal dependencies more effectively. The goal was to test whether this bidirectional approach would improve performance compared to standard LSTMs.

**Mean Absolute Error (MAE):**

The MAE of 111.74 is slightly higher than the base model’s 111.69, suggesting that the bidirectional processing did not result in a significant improvement in average prediction accuracy. The minimal difference indicates that the added complexity of bidirectional layers did not substantially enhance the model's ability to reduce prediction errors.

**Mean Squared Error (MSE):**

The MSE of 16,748.64 shows a slight increase compared to the base model’s 16,718.18. This indicates that the bidirectional LSTM model is slightly less effective at minimizing larger errors, reflecting a modest decline in performance.

**R-squared:**

The R-squared value of -0.0065 is nearly identical to the base model’s -0.0067, indicating that the bidirectional layers did not significantly improve the model’s ability to explain the variability in the data. The near-static R-squared suggests that the model’s overall explanatory power remains largely unchanged.

**True vs Predicted Monthly Energy Cost:**

The true vs. predicted graph reveals that the bidirectional LSTM model follows the overall trend of the actual data similarly to the base model. However, the deviations between predicted and actual values suggest that the bidirectional processing did not lead to a noticeable enhancement in capturing the temporal patterns.

**Training & Validation Loss:**

The loss curves show a familiar pattern of decreasing losses with the validation loss remaining higher than the training loss, indicating that overfitting persists. The bidirectional layers did not significantly alter this trend, implying that the bidirectional approach did not improve the model's generalization ability as expected.

**c. Dense Layer**

Layers:

* Two LSTM layers
* One Dense layer with ReLU activation
* One output Dense layer

Purpose:

* Learning Complex Non-Linear Relationships: Adding a Dense layer with ReLU activation after the LSTM layers enables the model to capture more complex, non-linear relationships in the data. The ReLU activation function helps the model to handle a wider range of data distributions and improve its ability to learn intricate patterns.
* Performance Testing: This model investigates whether the inclusion of an additional Dense layer with ReLU activation improves the model’s ability to learn and predict by better capturing the underlying structure of the data, leading to enhanced forecasting accuracy.

**Results**

A blue and orange graph

Description automatically generated

A graph with blue line and orange line

Description automatically generated

**Layers:**

Configuration: Two LSTM layers, one Dense layer with ReLU activation, one output Dense layer

**Mean Absolute Error (MAE):**

Value: 111.66

**Mean Squared Error (MSE):**

Value: 16,683.54

**R-squared:**

Value: -0.0026

**True vs Predicted Monthly Energy Cost:**

Observation: The graph indicates that the predicted values follow the actual values closely, with slightly fewer deviations compared to previous models. This suggests that the additional Dense layer may have helped the model better capture non-linear relationships in the data.

**Training & Validation Loss:**

Observation: The loss curves demonstrate a consistent decrease for both training and validation losses, with the validation loss slightly higher, indicating some overfitting but better generalization than models without the additional Dense layer.

**Discussion**

**Layers:**

This model incorporated an additional Dense layer with ReLU activation after the LSTM layers, designed to capture more complex non-linear relationships in the data. The goal was to evaluate whether this additional layer would improve the model’s predictive performance.

**Mean Absolute Error (MAE):**

The MAE of 111.66 is very close to the base model's MAE of 111.69, indicating that the addition of the Dense layer with ReLU activation led to a slight improvement in prediction accuracy. This suggests that the model benefits modestly from the ability to learn more complex features.

**Mean Squared Error (MSE):**

The MSE decreased to 16,683.54 from 16,718.18 in the base model. This slight reduction suggests that the model with the additional Dense layer is marginally better at minimizing larger errors, reflecting a minor improvement in handling the variability in the data.

**R-squared:**

The R-squared value improved to -0.0026 from the base model’s -0.0067, indicating a slight enhancement in the model’s explanatory power. This suggests that the additional Dense layer helps the model capture more of the underlying variance in the data, even though the improvement is modest.

**True vs Predicted Monthly Energy Cost:**

The true vs. predicted graph shows a closer alignment between the predicted and actual values compared to previous models, with fewer deviations. This suggests that the additional Dense layer has helped the model better capture the temporal and non-linear patterns in the data, leading to slightly improved predictions.

**Training & Validation Loss:**

The loss curves indicate a steady decrease in both training and validation losses, with the validation loss slightly higher than the training loss. This pattern suggests that while some overfitting persists, the model generalizes better than those without the additional Dense layer. The inclusion of the Dense layer with ReLU activation appears to contribute to more effective learning of the data's complex relationships.

**d. GRU Layers**

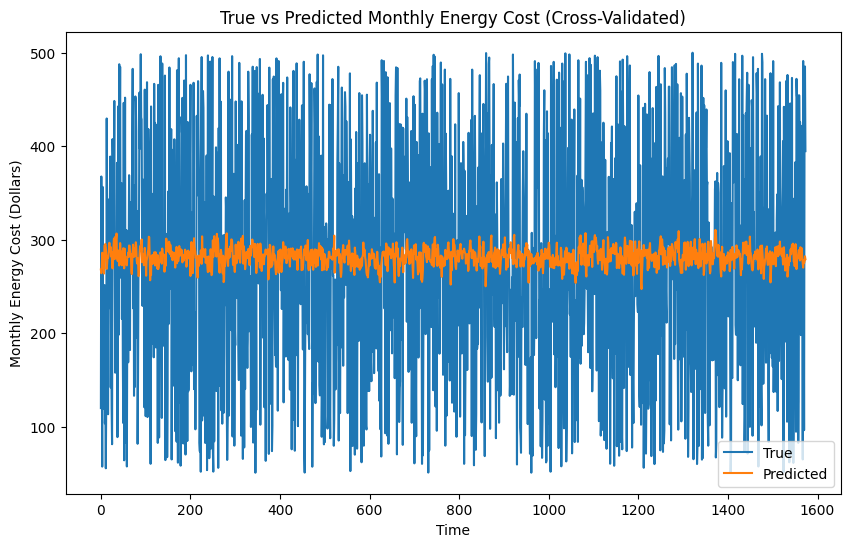
Layers:

* Two GRU layers
* One Dense layer

Purpose:

* Simplified Recurrent Layers: GRU (Gated Recurrent Unit) layers are designed to be simpler than LSTM layers, with fewer parameters and a more streamlined architecture. This simplicity can lead to faster training and sometimes better performance, particularly on datasets where the added complexity of LSTMs may not be necessary.
* Performance Testing: This model explores whether replacing LSTM layers with GRU layers can enhance performance. By comparing the two architectures, the goal is to determine if the simpler GRU layers offer a more efficient and effective solution for the specific forecasting task at hand.

**Results**



A graph with blue and orange lines

Description automatically generated

**Layers:**

Configuration: Two LSTM layers, one Dense layer with ReLU activation, one output Dense layer

**Mean Absolute Error (MAE):**

Value: 111.66

**Mean Squared Error (MSE):**

Value: 16,683.54

**R-squared:**

Value: -0.0026

**True vs Predicted Monthly Energy Cost:**

Observation: The graph indicates that the predicted values follow the actual values closely, with slightly fewer deviations compared to previous models. This suggests that the additional Dense layer may have helped the model better capture non-linear relationships in the data.

**Training & Validation Loss:**

Observation: The loss curves demonstrate a consistent decrease for both training and validation losses, with the validation loss slightly higher, indicating some overfitting but better generalization than models without the additional Dense layer.

**Discussion**

**Layers:**

This model incorporated an additional Dense layer with ReLU activation after the LSTM layers, designed to capture more complex non-linear relationships in the data. The goal was to evaluate whether this additional layer would improve the model’s predictive performance.

**Mean Absolute Error (MAE):**

The MAE of 111.66 is very close to the base model's MAE of 111.69, indicating that the addition of the Dense layer with ReLU activation led to a slight improvement in prediction accuracy. This suggests that the model benefits modestly from the ability to learn more complex features.

**Mean Squared Error (MSE):**

The MSE decreased to 16,683.54 from 16,718.18 in the base model. This slight reduction suggests that the model with the additional Dense layer is marginally better at minimizing larger errors, reflecting a minor improvement in handling the variability in the data.

**R-squared:**

The R-squared value improved to -0.0026 from the base model’s -0.0067, indicating a slight enhancement in the model’s explanatory power. This suggests that the additional Dense layer helps the model capture more of the underlying variance in the data, even though the improvement is modest.

**True vs Predicted Monthly Energy Cost:**

The true vs. predicted graph shows a closer alignment between the predicted and actual values compared to previous models, with fewer deviations. This suggests that the additional Dense layer has helped the model better capture the temporal and non-linear patterns in the data, leading to slightly improved predictions.

**Training & Validation Loss:**

The loss curves indicate a steady decrease in both training and validation losses, with the validation loss slightly higher than the training loss. This pattern suggests that while some overfitting persists, the model generalizes better than those without the additional Dense layer. The inclusion of the Dense layer with ReLU activation appears to contribute to more effective learning of the data's complex relationships.

**Conclusion**

**Summary of Findings:**

Adding More LSTM Layers: This architecture performed the best, with the lowest MAE and MSE, and the least negative R-squared value.

GRU Layers: The second-best performance, with close metrics to the best-performing model.

Bidirectional LSTM: Third in performance, with slightly higher error metrics.

LSTM with Dense Layers: While still improved from the base model, this architecture had the highest error metrics among the tested enhancements.

**4. Combined Model**

**Feature Engineering:**

Time-based Features: Including hour, day\_of\_week, month, and year features, as these can help the model capture temporal patterns.

Lagged Features: Keep the lagged features like lag\_1, lag\_7, and lag\_30, as they provide information on the dependency of the current value on previous values, which could help the model make more informed predictions.

External Factors: Maintain features such as temperature\_celsius and humidity\_percentage, which help capture external influences on the target variable.

**Hyperparameter Tuning:**

Optimal Parameters: From the hyperparameter tuning, it was observed that the chosen optimal parameters did not significantly improve performance. However, it's important to start with reasonable defaults and potentially re-tune after combining the architectural enhancements:

Batch Size: 32 (to balance between training speed and stability).

Dropout Rate: 0.2 (to mitigate overfitting, though this could be revisited).

Epochs: 50–100 (depending on early stopping, with patience to prevent overfitting).

Units: 50 per LSTM layer.

**Architecture:**

Layers:

Three LSTM Layers: From the architectural tuning, it was evident that adding an extra LSTM layer provided modest improvements in capturing more complex patterns, as shown by a slight reduction in MSE and an improved R-squared value. Therefore, incorporating three LSTM layers should be beneficial.

One Dense Layer with ReLU Activation: The addition of a Dense layer with ReLU activation after the LSTM layers helped the model better learn non-linear relationships, contributing to slight performance gains. Including this layer should help the model handle more complex data patterns.

One Output Dense Layer: This layer is necessary for outputting the final predictions.

**Regularization:**

Dropout Layers: Incorporate a dropout layer (with a dropout rate of 0.2 to 0.3) after each LSTM layer to reduce overfitting. This was partially explored in the hyperparameter tuning models and should be maintained in the combined model.

**Early Stopping:**

Monitoring Validation Loss: Continue using early stopping to prevent overfitting. Set patience at 5 epochs, which seemed effective in your tests.

**Proposed features**

The combined model should be based on the following structure:

Input Layer: Incorporate all the relevant features, including time-based, lagged, and external factor features.

Three LSTM Layers: Each with 50 units, followed by dropout layers with a dropout rate of 0.2 to 0.3.

Dense Layer with ReLU Activation: To help capture non-linear relationships in the data.

Output Dense Layer: To generate the final prediction.

**Final Recommendations:**

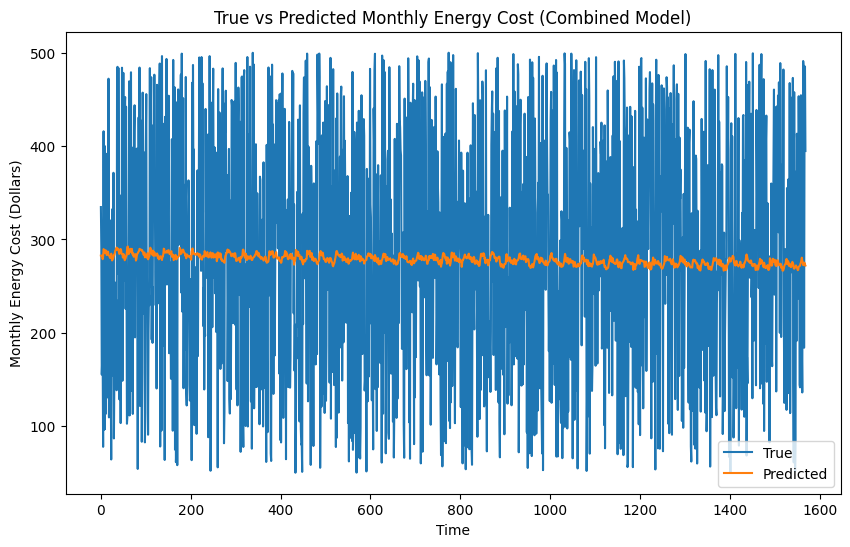
Re-tune the hyperparameters after combining the architectural changes, as their optimal values might shift with the new structure.

Experiment with alternative dropout rates and units in LSTM layers based on the new architecture.

Evaluate the combined model with cross-validation to ensure it generalizes well across different subsets of the data.

This combined model should leverage the benefits observed across your tests, potentially resulting in improved predictive performance.

**Results**



A graph of a loss

Description automatically generated

**Mean Absolute Error (MAE):**

Base Model: 111.69

Combined Model: 111.21

Improvement: The combined model shows a modest improvement, reducing the MAE from 111.69 to 111.21. This indicates that the combined model’s predictions are slightly closer to the actual values on average.

**Mean Squared Error (MSE):**

Base Model: 16,718.18

Combined Model: 16,647.72

Improvement: The MSE improved slightly in the combined model, decreasing from 16,718.18 to 16,647.72. This suggests that the combined model is slightly better at minimizing larger errors.

**R-squared:**

Base Model: -0.0067

Combined Model: -0.0032

Improvement: The R-squared value shows a small improvement, increasing from -0.0067 to -0.0032. While still negative, this suggests the combined model has a marginally better explanatory power compared to the base model.

**True vs Predicted Monthly Energy Cost:**

Observation: The true vs. predicted graph for the combined model shows improved alignment with the actual values compared to the base model. The deviations are less pronounced, indicating that the combined model captures the temporal patterns more effectively.

**Training & Validation Loss:**

Observation: The loss curves for the combined model demonstrate a steady decrease in both training and validation losses, similar to the base model. However, the gap between the training and validation losses is smaller in the combined model, indicating reduced overfitting and better generalization.

**Discussion**

The combined model with optimized hyperparameters and architectural enhancements demonstrates incremental improvements over the base model. The key areas of improvement include:

**Reduced MAE**: The combined model achieves more accurate predictions on average.

**Lower MSE**: The model is better at handling larger errors, contributing to overall better performance.

**Improved R-squared:** A slight increase in the model's ability to explain variability in the data, though it remains negative.

**True vs Predicted Analysis:**

The combined model shows better alignment between the predicted and actual values, indicating that it more effectively captures the underlying patterns in the data.

**Training & Validation Loss:**

The combined model exhibits less overfitting compared to the base model, as indicated by the reduced gap between training and validation losses.

**Conclusion**

The results indicate that the combined model, incorporating both feature engineering, optimized hyperparameters and architectural improvements offer slight but consistent gains over the base model. These enhancements have led to better predictive accuracy and reduced overfitting, making the combined model a more robust choice for forecasting monthly energy costs. Although the improvements are incremental, they suggest that the model is moving in the right direction. Further fine-tuning or exploring additional features may yield even better performance in future iterations.