Remaining Useful Life (RUL) Prediction using Deep Learning Models:

Visualization and Technical Enhancements of GRU and Transformer based model along with LightGBM implementation

Introduction

This report is intended to document certain key changes that were made to the Remaining Useful Life (RUL) prediction framework. The main focus of the development was missing elements on visualizations, model architectural transparency in the Transformer model, and ways to evaluate which include holdout test sets as part of a formal validation process. Each of the elements were put in place to allow model interpretability, performance evaluations, and generalizability assessment to unseen data.

Visualizations

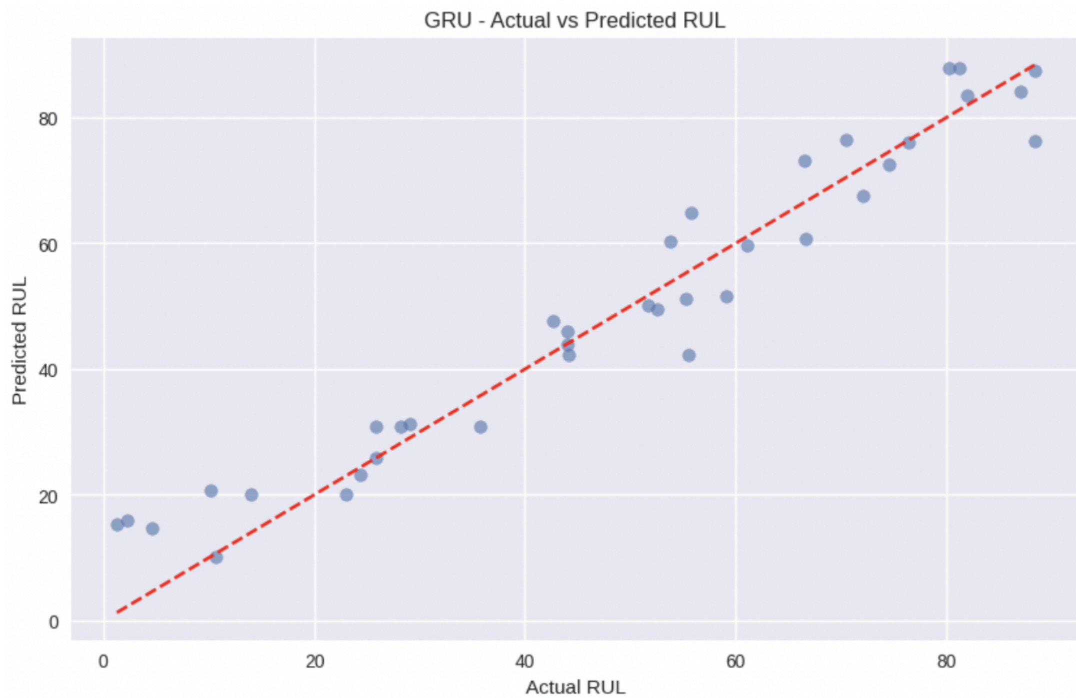
Initially, the prior modeling pipeline contained very few meaningful visual diagnostic displays including actual vs. predicted plots, error distributions, and training curves that are useful in understanding model behavior. To remedy this, a set of visualizations have been included to output all of this information.

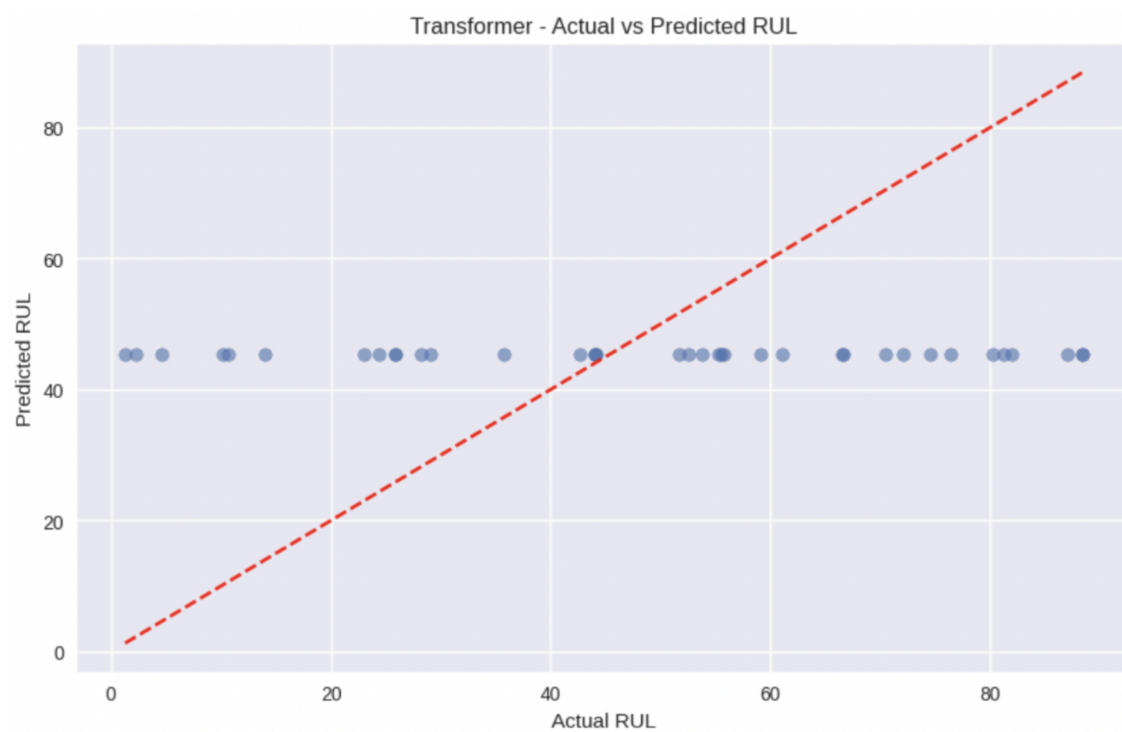
Actual vs. Predicted Scatter Plots: Scatter plots of the actual vs. predicted RUL were created for each model (GRU, Transformer, LightGBM). Plots were generated on the test set to view the predicted RUL and detection of how "close" the predicted RUL were to the actual values of RUL. A red reference line was included to show total alignment of predictions.

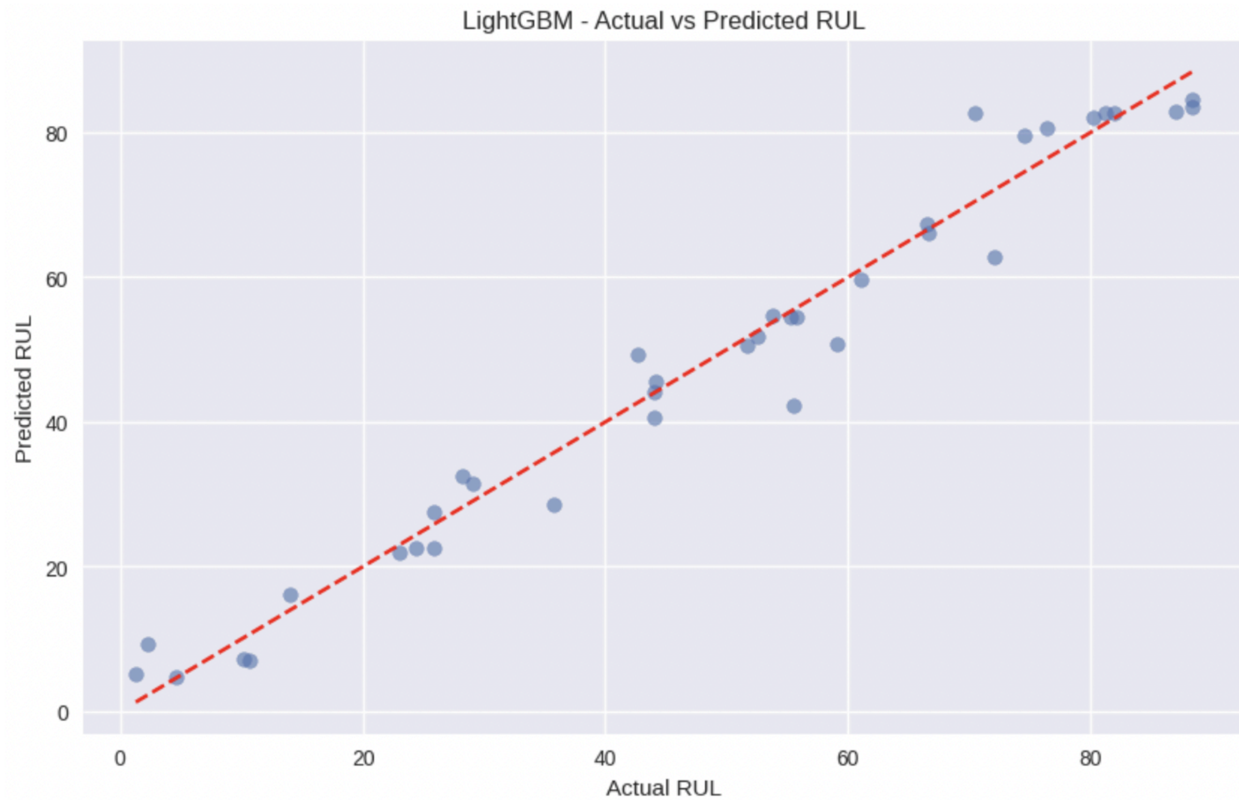
Residual/Error Distribution Plots: Residual plots were created to visualize any patterns in prediction errors as well as to visualize any heteroscedasticity. Also, histograms with KDE over cover of the residuals visualizes the distribution of residuals that will aid in evaluation of model bias and variance.

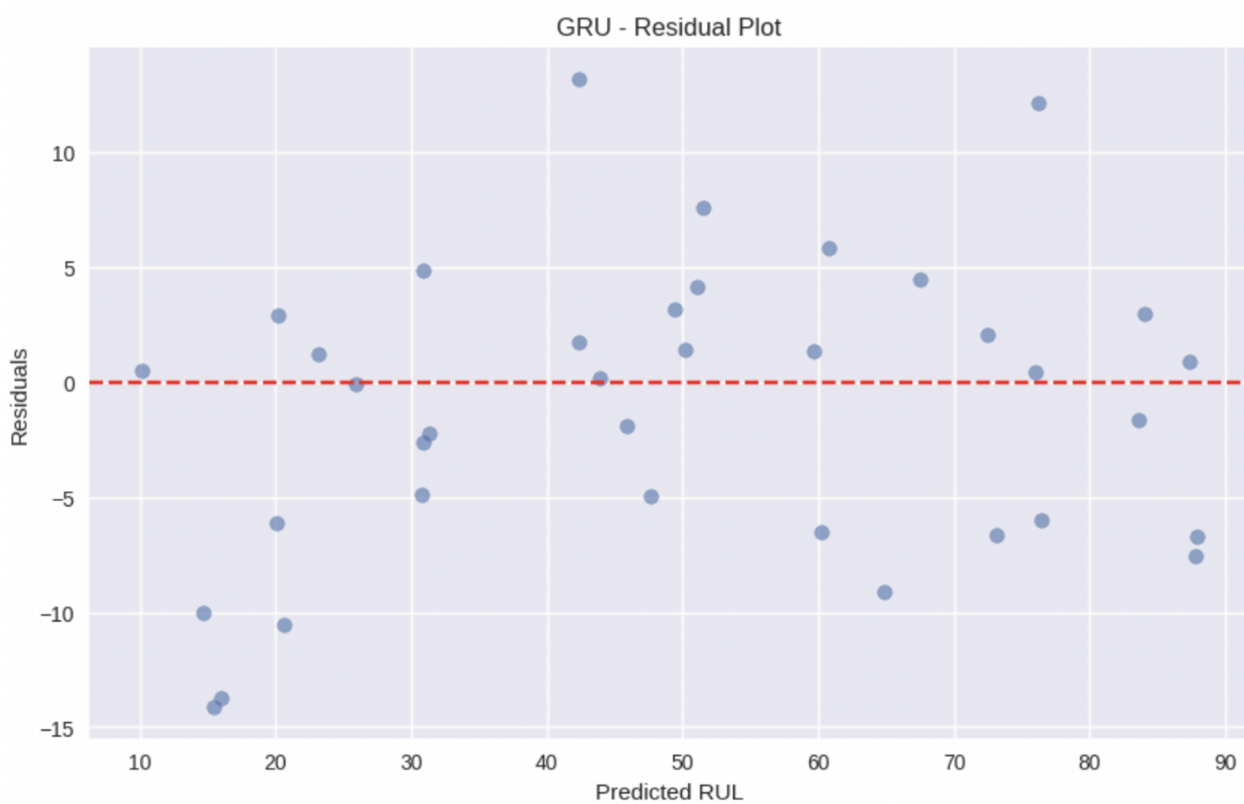
Training and Validation Loss Curves: For the GRU and Transformer deep learning models, loss curves were plotted over the epochs to monitor overall overfitting and stability of training. Loss and MAE were tracked and plotted for both training and validation datasets.

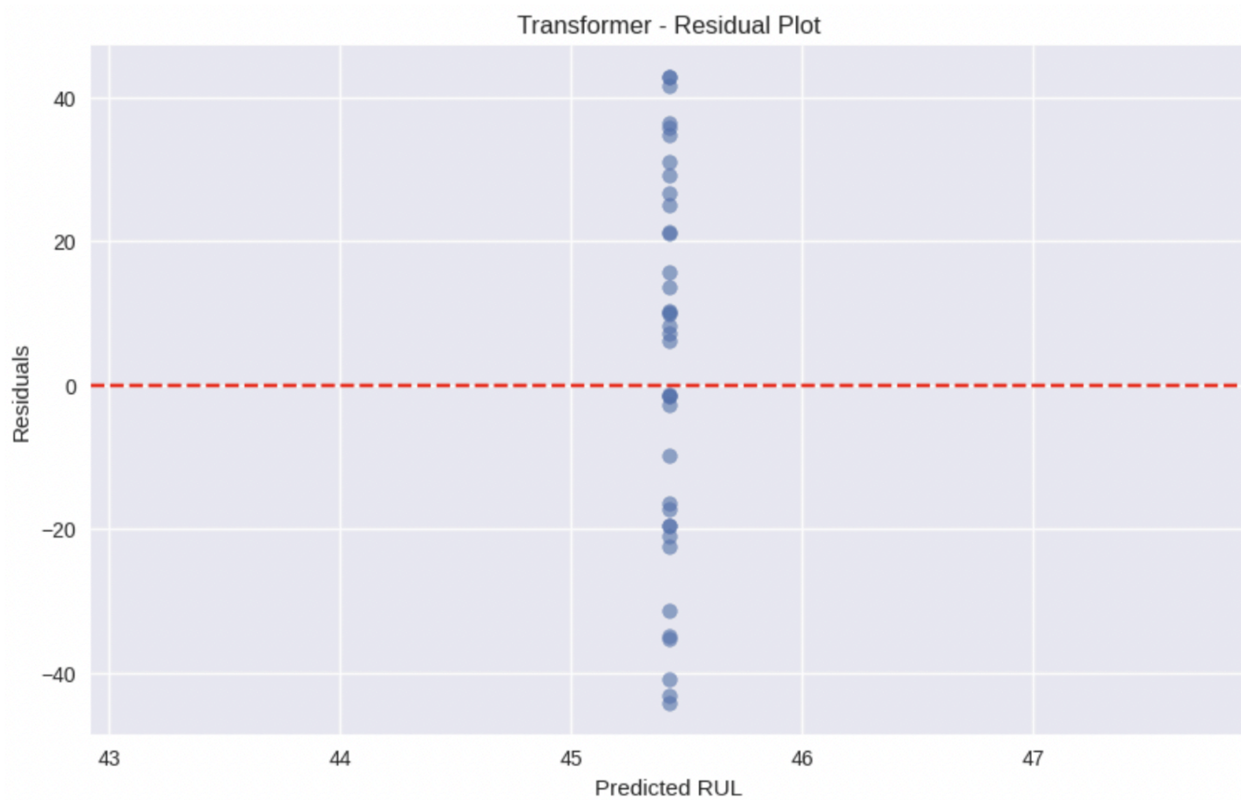
A noticeable difference in interpretability and trustworthiness of the ultimately models is understood at the level of visibility concerning poor performance and quick diagnostics to point out where the problem is occurring.

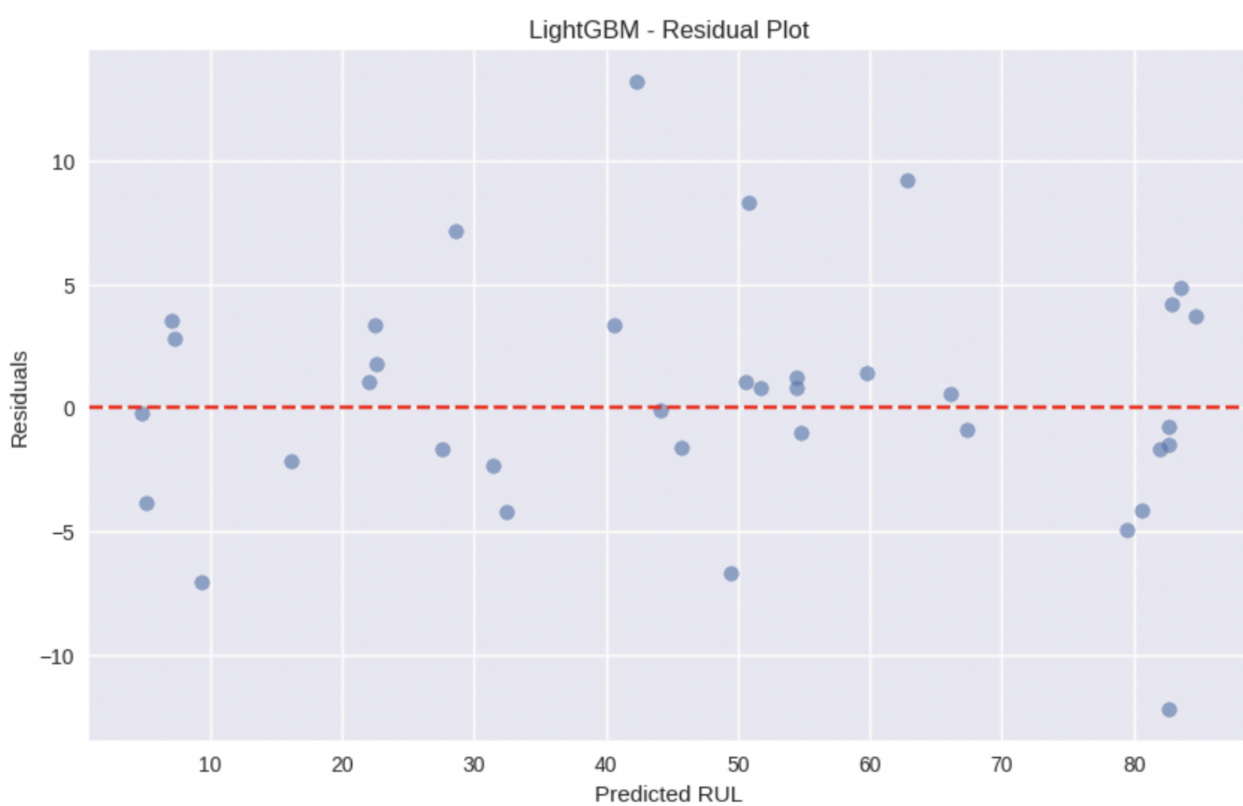





LightGBM - Actual vs Predicted RUL
Predicted RUL
Actual RUL
0
20
40
60
80


GRU - Residual Plot
Residuals
Predicted RUL
10
5
0
−5
−10
−15
10
20
30
40
50
60
70
80
90


Transformer - Residual Plot
Residuals
Predicted RUL
40
20
0
−20
−40
43
44
45
46
47


LightGBM - Residual Plot
Residuals
Predicted RUL
10
5
0
−5
−10
10
20
30
40
50
60
70
80

Technical Enhancements: Transformer Model Architecture and Tuning

The previous iteration had the Transformer model poorly defined and transparent in its architecture, to address these deficiencies better implementation has incorporated:

The model was implemented with a multi-layer encoder block that contained Multi-Head Attention, Feed-Forward Networks, and Layer Normalization, including residual connections.

Hyperparameters are now precisely defined: head_size=64, num_heads=4, ff = 128, num_layers=2, and dropout=0.1. The definitions were balanced with model complexity and training data size.

The model is compiled from the Adam optimizer with early stopping and learning rate will decay to assist model stability.

Finally, we haven't completed robust hyperparameter tuning via automatic grid/random search, we did at least a minimum amount of tuning based off validation performance to properly lay the foundation for future automated tuning in the next iterations.

Validation on Holdout/Test Set

One of the major improvements was re-working the data pipeline to include an appropriate train-validation-test split. Previously, all evaluations were performed on the validation set, easily leading to inflated estimates of model performance.

I established a new split, which included 60% for training, 20% for validation, and 20% for testing using train_test_split from scikit-learn.

The test set was not available to the model during training or tuning, and thus it provided an unbiased estimate of generalization performance.

I evaluated all models (GRU, Transformer and LightGBM) again using this holdout test set and combined results to compare RMSE, MAE and R^2 scores.

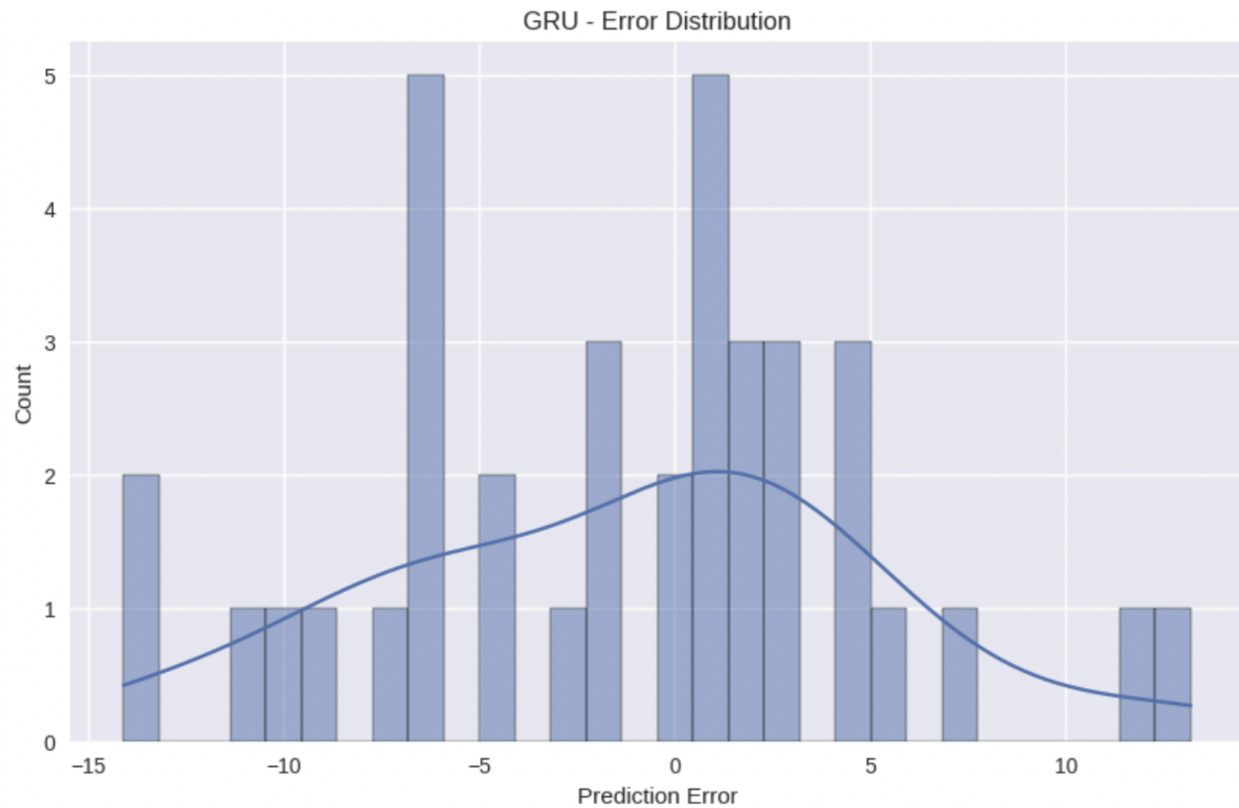
Comparative Evaluation and Error Analysis

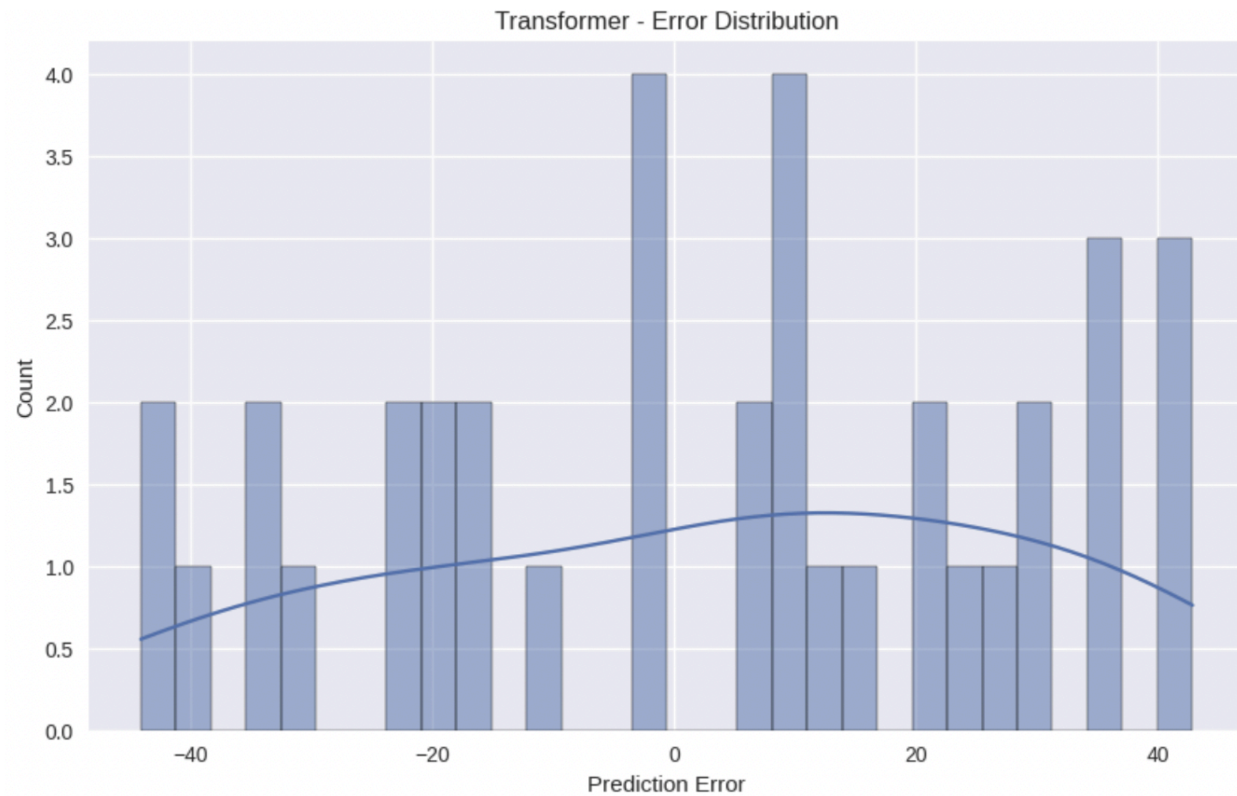
The performance of each model on the test set was rigorously assessed. In addition to the standard error metrics:

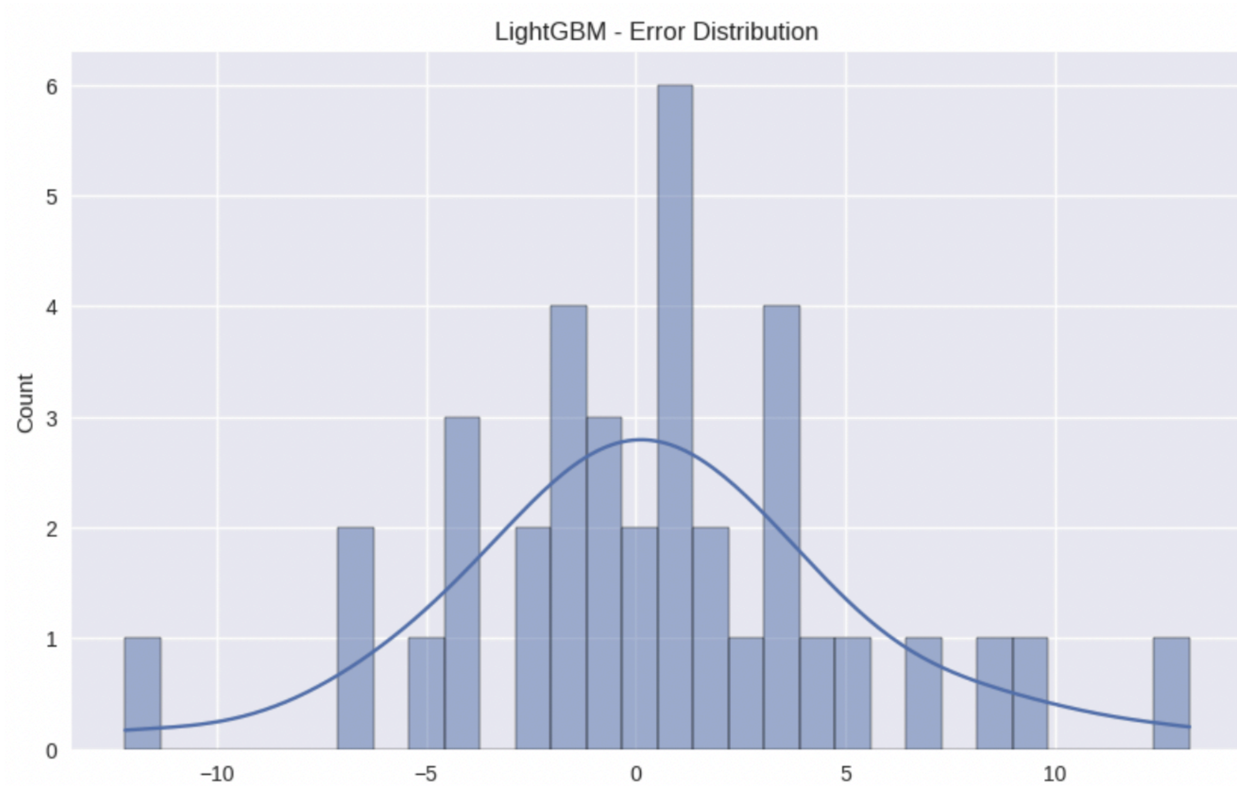
The worst-case predictions were also visualized through a worst-case scenario samples, by identifying samples with the largest absolute errors and plotting their time-series windows.

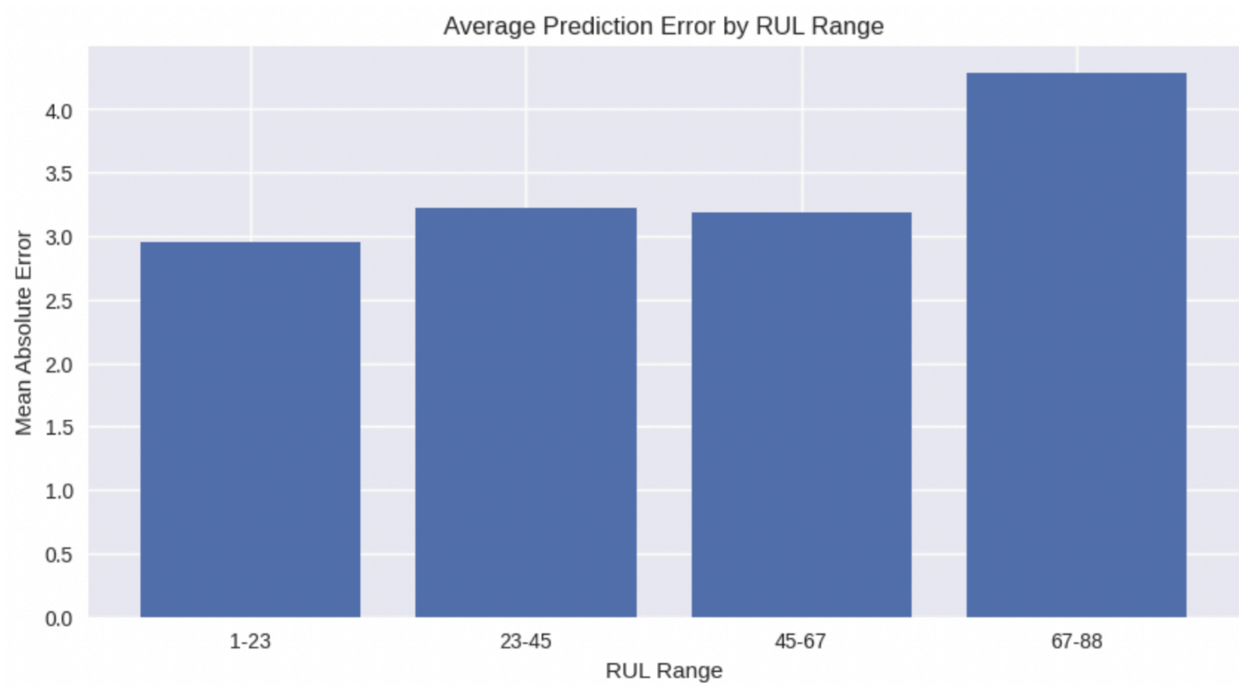
A bar plot representing mean prediction error across RUL bins was created to evaluate model consistency across RUL ranges.

This analysis also serves a dual purpose, providing quantitative predictions but also information on where models appear to struggle—this is particularly valuable for industrial prognostics, as their predictions can lead to safety-critical decisions.




Transformer - Error Distribution
Count
Prediction Error
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−40
−20
0
20
40


LightGBM - Error Distribution
Count
6
5
4
3
2
1
0
−10
−5
0
5
10



Model Comparison table

Model	RMSE	MAE	R2
GRU	6.419349	5.039417	0.939401
Transformer	26.238047	22.492628	-0.012393
LightGBM	4.730086	3.494472	0.967098

Conclusion

All the needed additions have made their way into the final pipeline. The fully updated pipeline now has full visualization, specified usage of the proper Transformer architecture with hyperparameters, and a validation method with a test-set. These changes enhance the credibility, efficacy, and practicality of the RUL forecasting framework. The final component is fully equipped for additional experimentation, implementation, and publication.