Energy Forecasting Model Report

Summary On Project

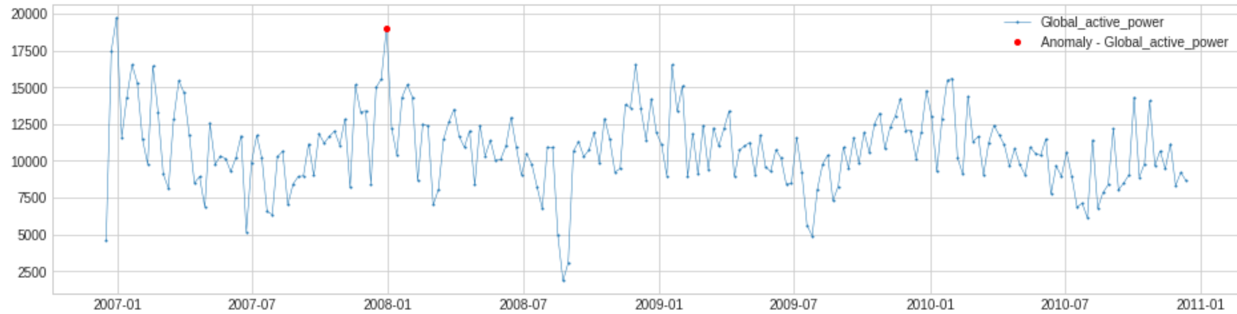
Abstract:

I have explored a multitude of algorithms in this project ranging from conventional statistical methods such as the BATS all the way to the Deep learning models such as NBEATS and Nhits architecture. According to the models tested some of the key findings are

- The Nixtla ARIMA model performs significantly faster than the conventional pmdArima model. However, the model accuracy is not the same and thus how the algorithm reaches the optimal pdq search might not be the same.
- GRU DL method proved to be the best method when forecasting, however it is not as reliable as compared to more conventional tree-based models such as LGBM and XGBoost.
- Ensemble the regressor by changing the base estimator of an Adaboost model to a deep learning model does not always improve performance.
- Ensemble forecasters are the most optimal approach when forecasting data.
- The new complex Exponential smoothing model is one of the better forecasting models. The model has outperformed conventional Statistical Smoothing models.
- It is possible to breakdown a timeseries into different components such as seasonality and trend then forecast the individual components using Tree based models then add each component. This method while powerful requires the decomposition to have a low residual for the most optimal performance. We can have even better performance if we stack the model on top of the forecasted decomposition, thus using a multivariate forecasting approach for both the individual components and the target timeseries.

Data Quality:

- Missing Data Points that needed to be imputed
  Linear interpolation proves to have the best effect on MAPE
- Data Is highly erratic thus makes forecasting difficult (Data Needs to be aggregated to weekly data sample to help reduce error for forecasting.)
- The Data is Stationary (Data does not need differencing)
- Data Is nonlinear thus the use of linear forecasting models will not provide the best forecasts
  The Alternative hypothesis is passes the BDS test thus confirming that the data is not linear.
- Data Contains Anomalies



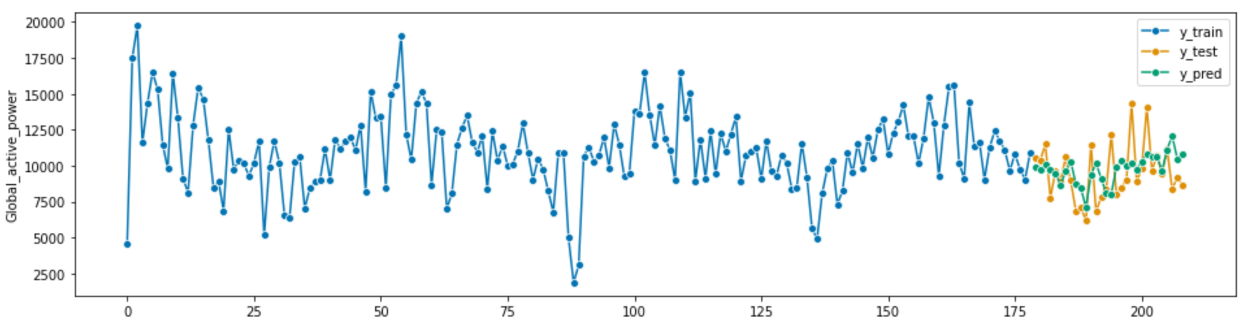
External Data:

External Data Causal Inference:

To justify the use of external data the use of hypothesis testing is robust in evaluating data relevance in timeseries forecasting. However, given that our data is significantly nonlinear as seen from the BDS test, granger causality is not going to do a good job given that it works well on linear data. Thus, the optimal method for evaluating causality is the use of convergent cross mapping from the causal-ccm package. The package uses a novel technique called convergent cross mapping by Sugihara G. Using the statistical test, we concluded that the dataset. For our target timeseries, all the multivariate datasets do have a causal impact on it. However, we also inspected that our target timeseries also affects the other variables thus there is complex relationship in our dataset.

Ensemble Forecasting of Specialized Forecasters based on the complex seasonality in the dataset. (Optimal Method)

We use multiple identical algorithms with different sliding windows to forecast the timeseries. When using this method, we can reduce error since the specialized models will forecast the components effectively while still capturing annual seasonal patterns due to the ensemble.



MAPE:16%

RMSE: 1900

This method has been used in M-Competitions since it proved to be a powerful method when forecasting complex data. Often not it is recommended to go around 5-9 algorithms since this yields the most improvement to forecasting. We will optimize this model even further for better results.

Optimization:

We will use a flexible hyperparameter framework called optuna as it gives us the ability to optimize our model through a Bayesian optimization approach. The algorithm used for the Bayesian optimization is TPE.

Our metrics to optimize are

1. Mean Residual
2. Standard Deviation of Residual
3. RMSE

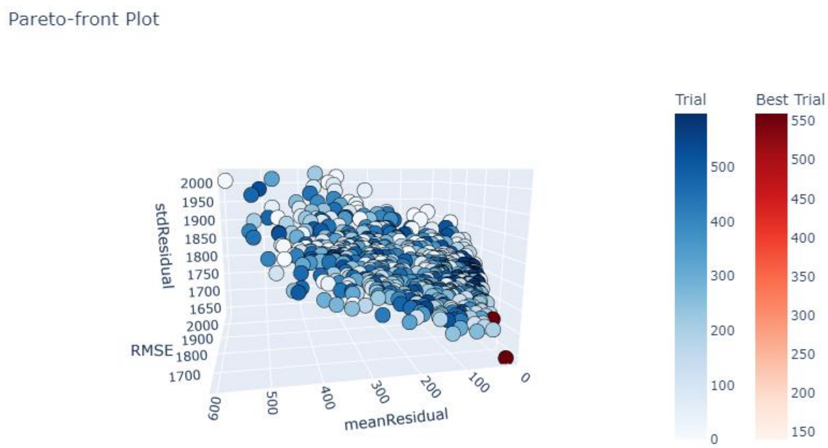
All 3 would help make sure that the model learns as much as possible from the dataset.

We optimize every single sub algorithm in the forecaster while still wrapping all 4 forecasters with LGBM using the feature importance from the LGBM model.

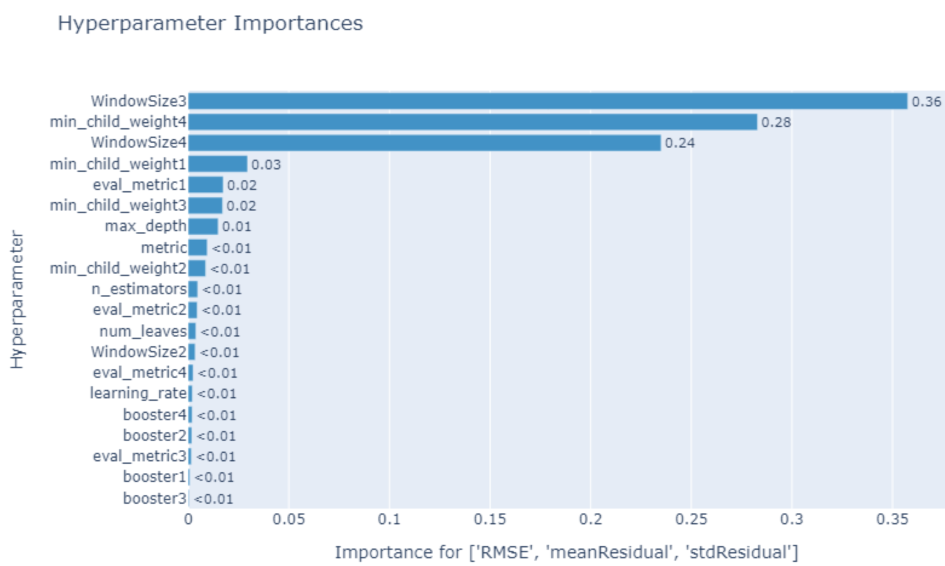
The parameters optimized are booster, evaluation metric, and min_child_weight as well every single sub window for the sub seasonal components.

We also optimize the LGBM model by optimizing the max_depth, num_leaves, number of estimators, learning rate, and the evaluation metric.

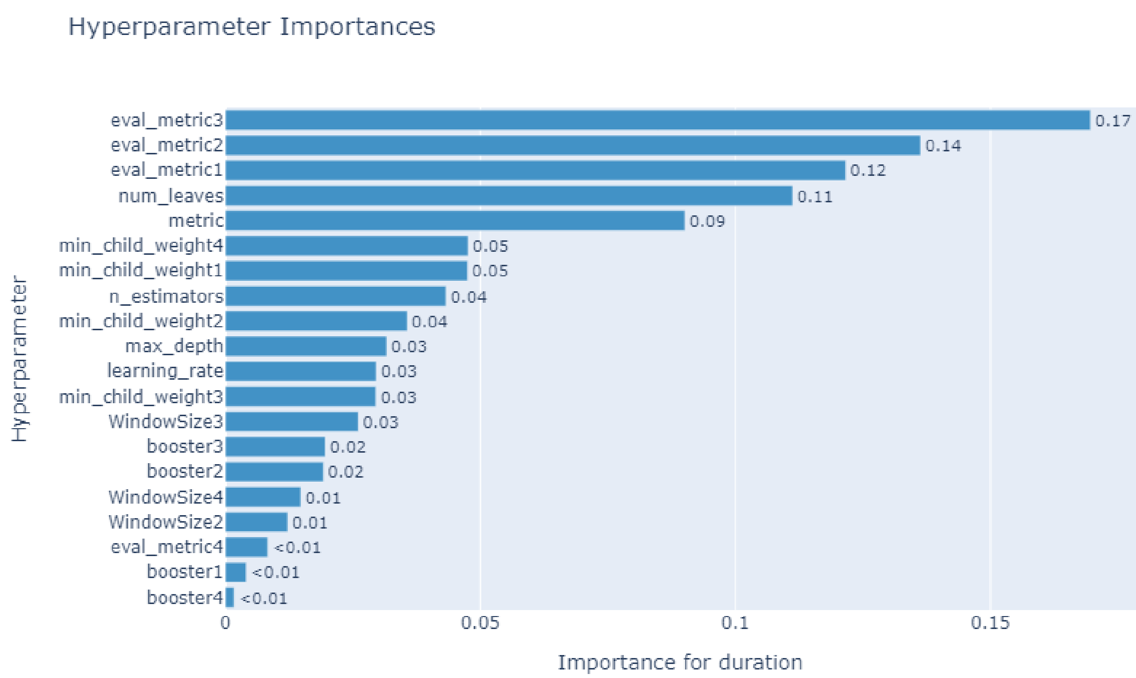
Here is a visualization of the iterations for all 3 metrics



Here is the importance of every single hyperparameter on performance

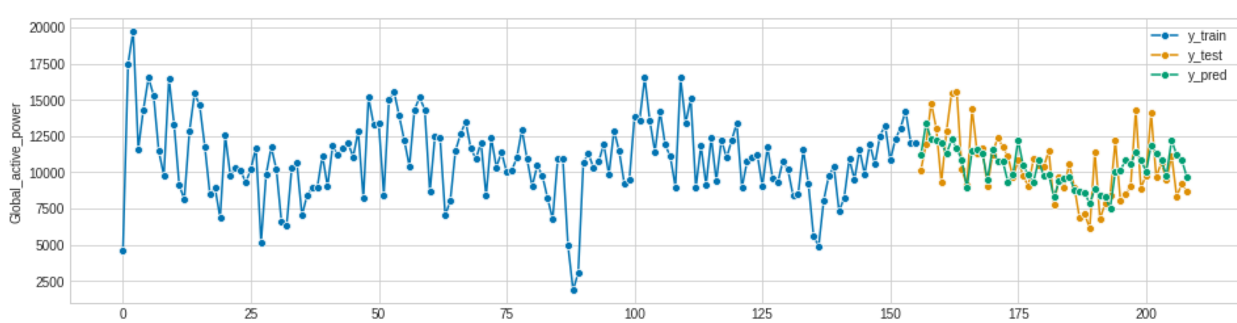


Here is the importance of every single hyperparameter on the duration of time it takes to train the model.



After storing and saving the study, we use the hyperparameter on

our dataset to see how this would improve the model

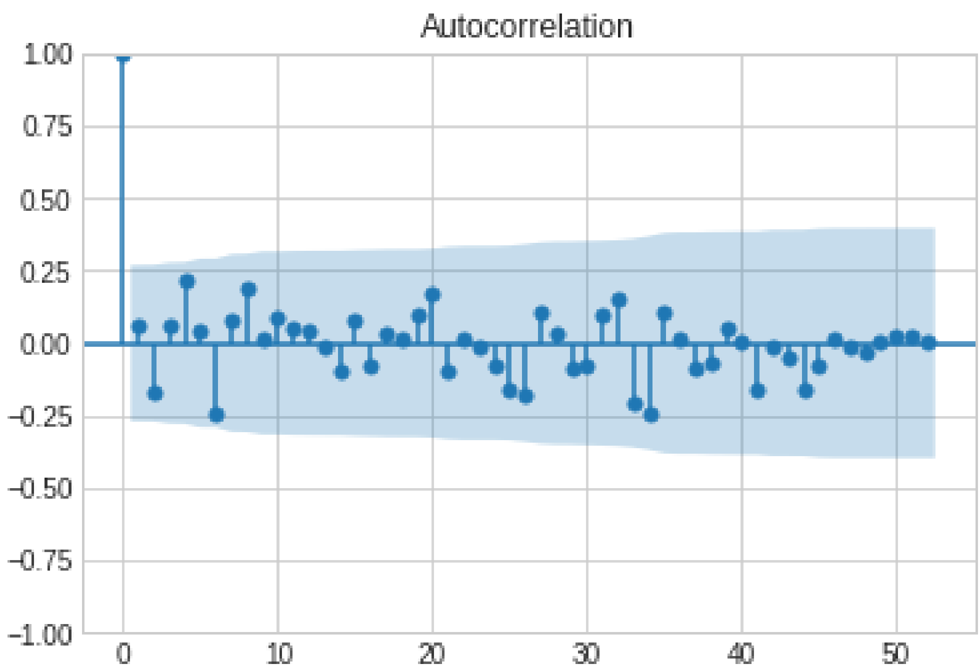


RMSE:1627

MAPE:12.97%

We can see that our model is capable of forecasting well on the dataset, however it is not capable of forecasting the overall spikes.

We also use the autocorrelation to see if our model learned from the dataset


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50

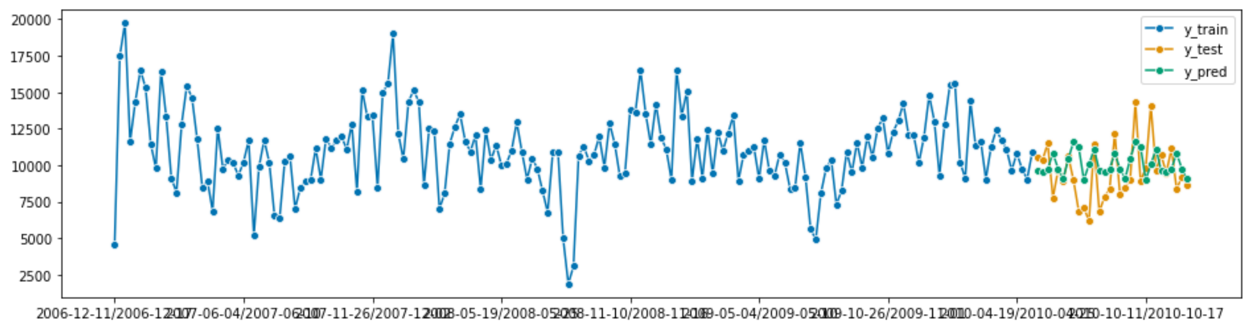
Algorithms and methods used Detailed Breakdown:

Univariate:

Conventional Statistical Models:

BATS/TBATS (Baseline): The algorithm works very well with multiple seasonal datasets. The exponential smoothing has trigonometric, box cox, ARMA errors, trend components, and seasonal components the model adapts to.

BATS:

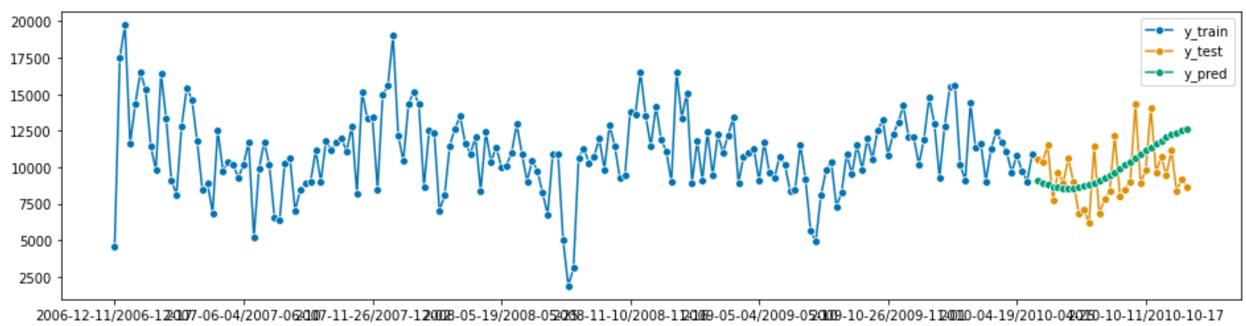


MAPE: 18.57%

RMSE: 2005

Time: 1.5 Mins

TBATS:

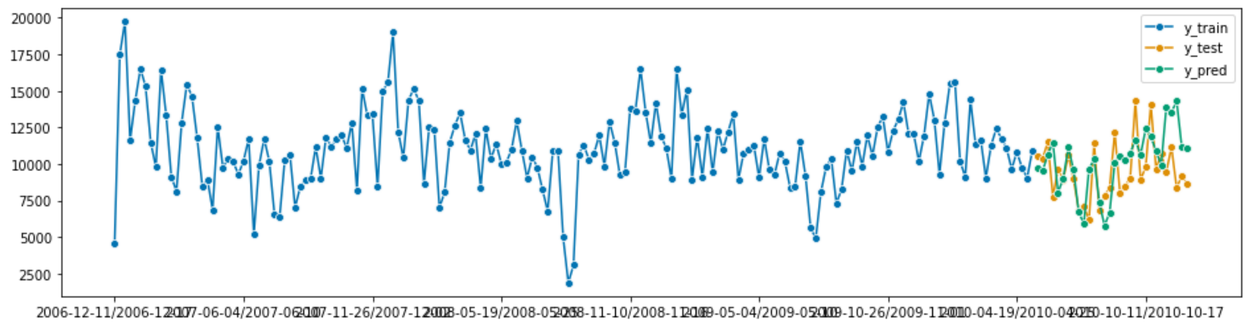


MAPE: 21%

RMSE: 2178

Time: 2 mins

Complex Exponential Smoothing: A new algorithm introduced by Ivan Svetunkov for timeseries forecasting.



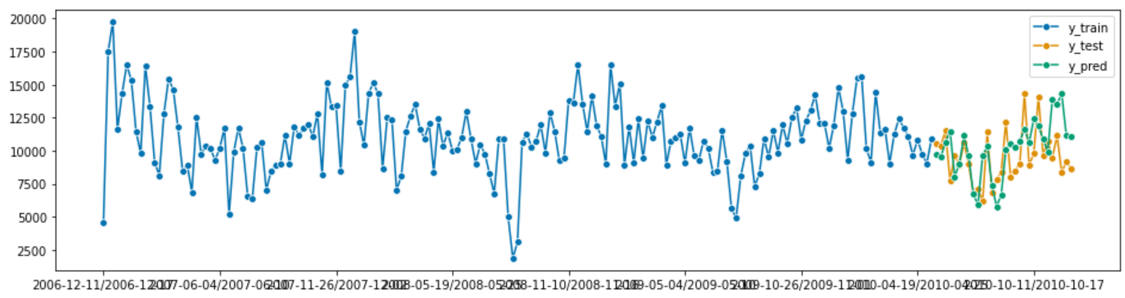
MAPE: 20.6%

RMSE: 2258

Time: 39 Secs

ARIMA (pmdArima & Nixtla): Both models are variants of the auto arima model, however the nixtla variant is faster than the pmdArima variant.

Pmd:

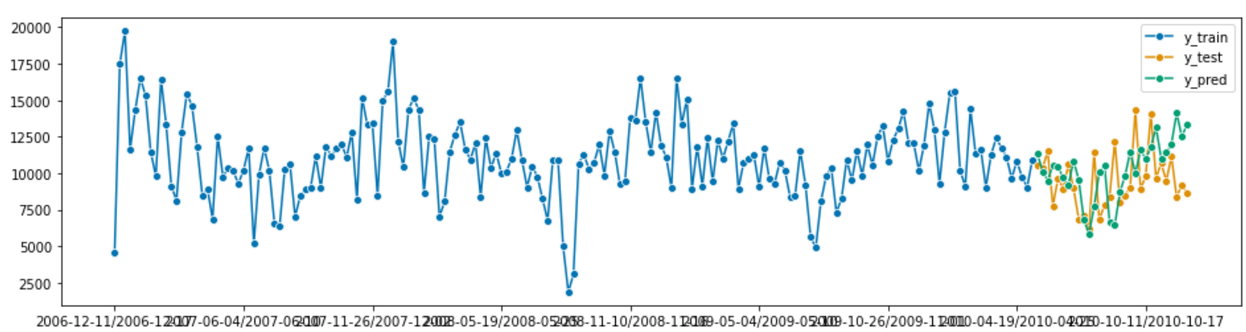


MAPE:20.6%

RMSE:2258

Time: 5 mins

Nixtla:



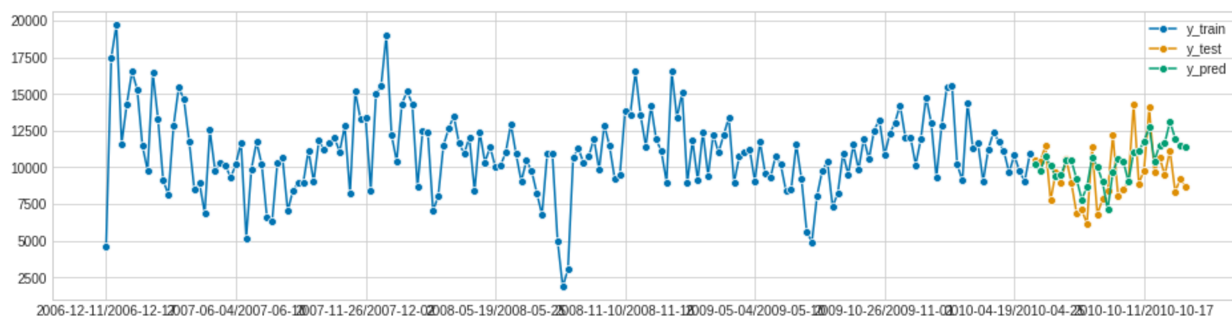
MAPE: 23.8%
RMSE: 2703

Time: 51 Seconds

Sliding Window ML and DL models (Tree based and Neural Networks perform well on nonlinear data thus this method would prove to be the most optimal method for our dataset) The question lies on which algorithm to use.

The use of sliding window model is considerably faster than other conventional algorithms, with the use of neural networks taking considerably more time depending on network complexity as well as epochs.

Baseline: Random Forest: (Both the base random forest was used. However, I have also tried to change the base estimator to other algorithms such as LGBM and there isn't much improvement to the model performance)

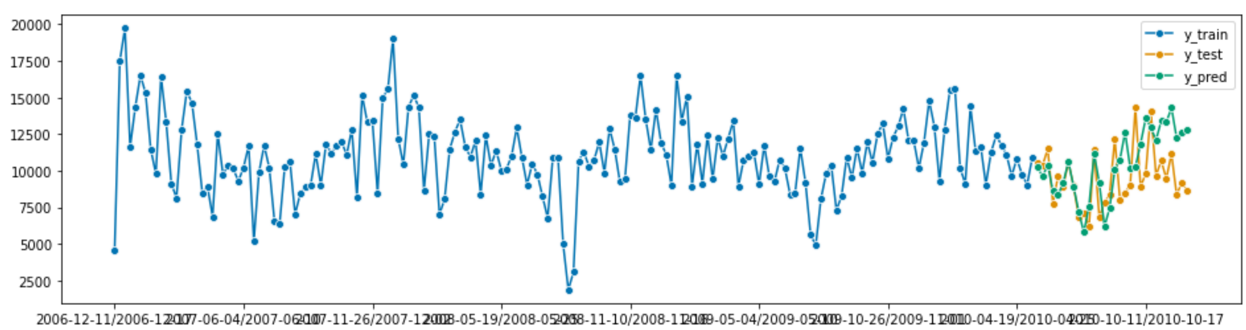


MAPE: 18.36%

RMSE: 1909.62

Gradient Boosting (XGBoost ,LGBM,Catboost,Hisogram Gradient Boosting):

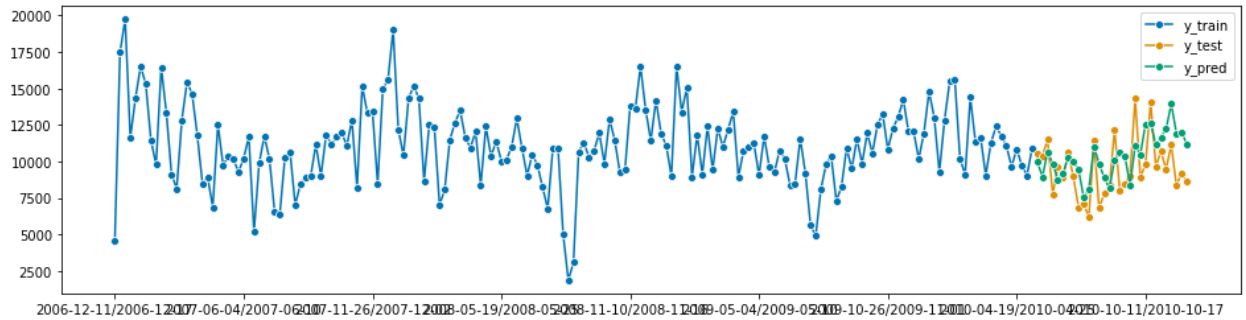
XGBoost:



MAPE: 18.57%

RMSE: 1949 RMSE

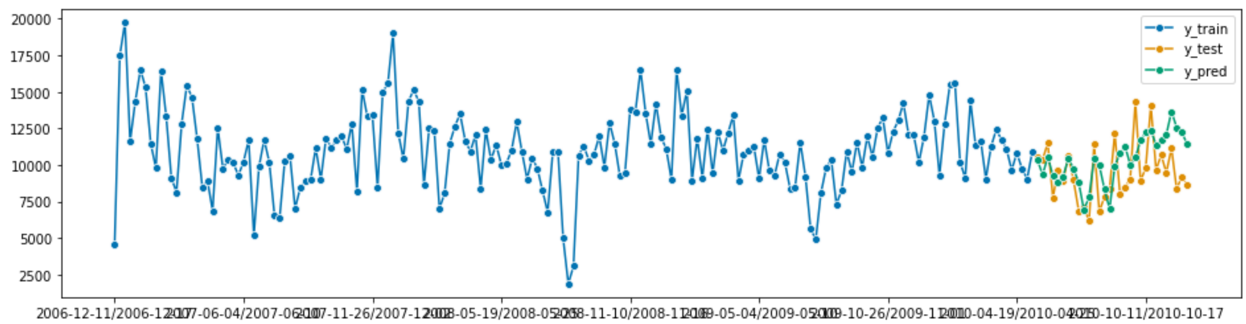
LGBM:



MAPE:21%

RMSE:2387

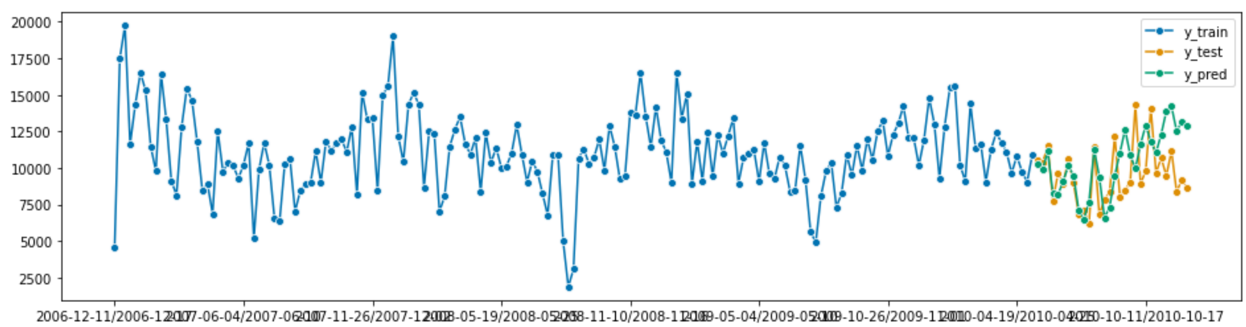
Catboost:



MAPE:19.04

RMSE:2081

HistGradientBoosting:

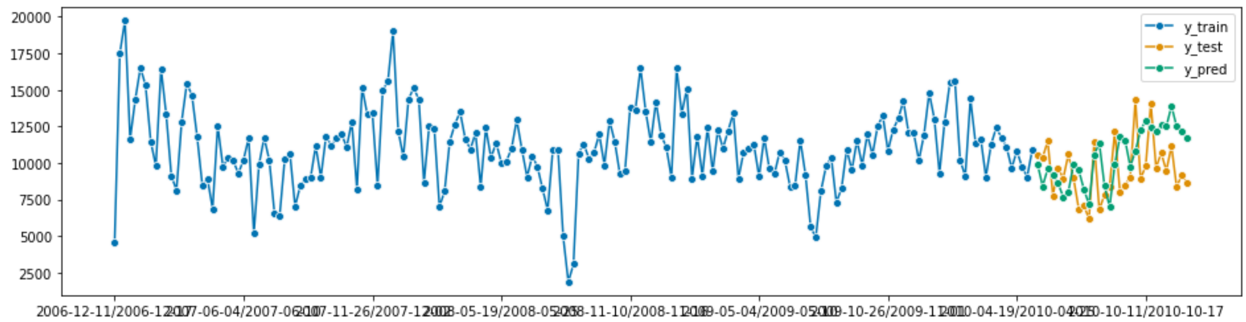


MAPE:21%

RMSE:2441

Deep Learning

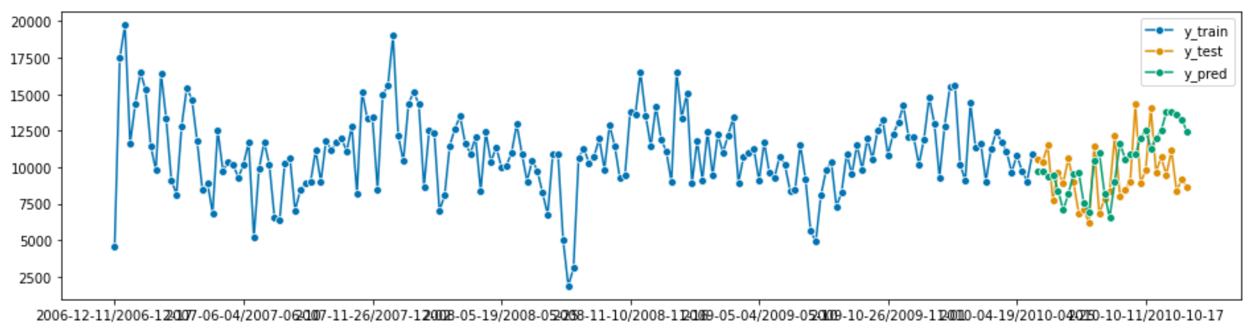
MLP:



MAPE:25%

RMSE:2655

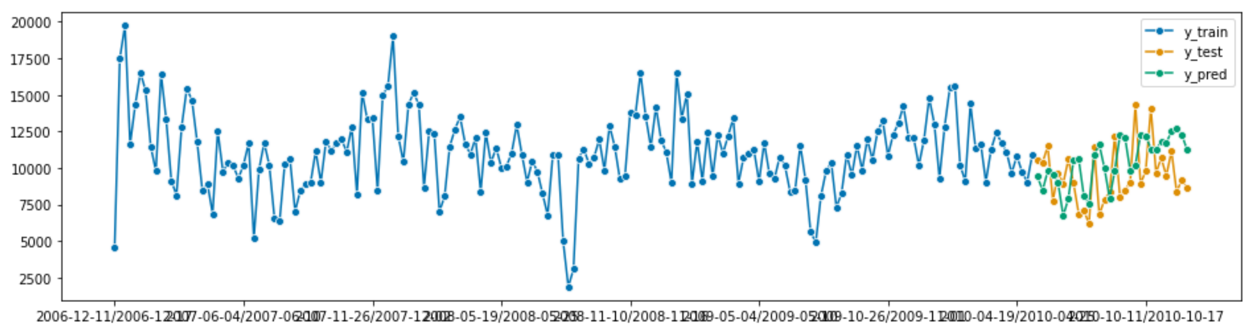
Ensemble Learning of MLP Deep learning models:



MAPE:22%

RMSE:2375

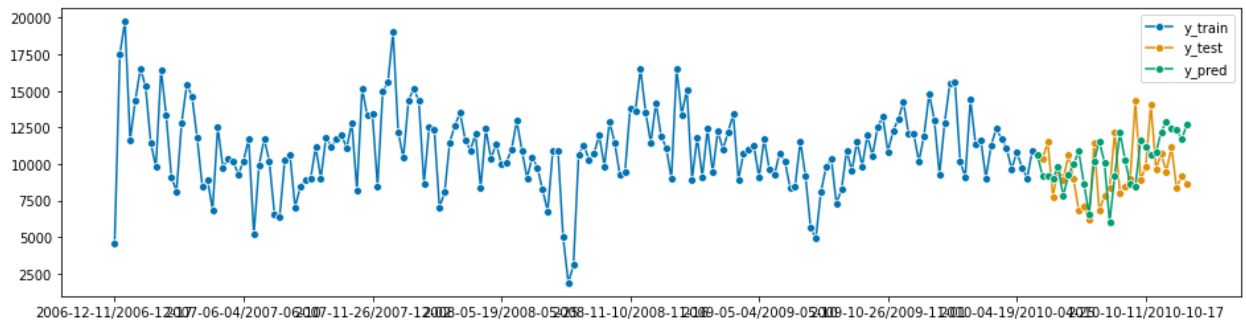
CNN



MAPE:25%

RMSE:2551

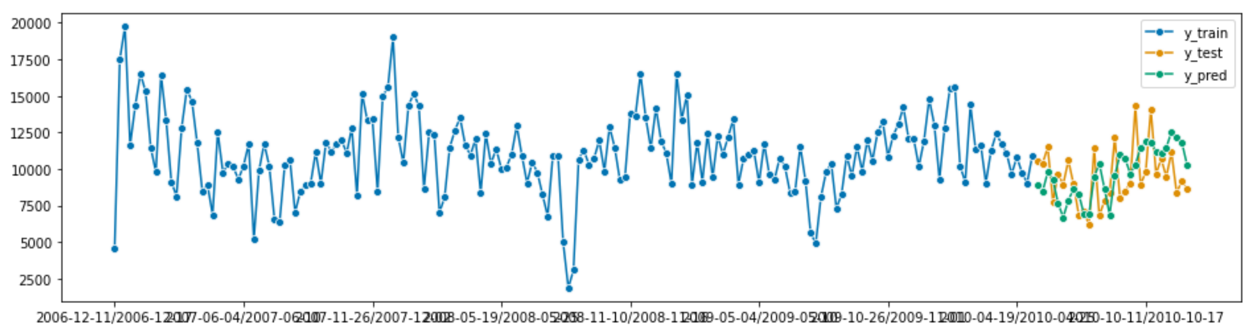
CNN Ensemble



MAPE:23%

RMSE:2521

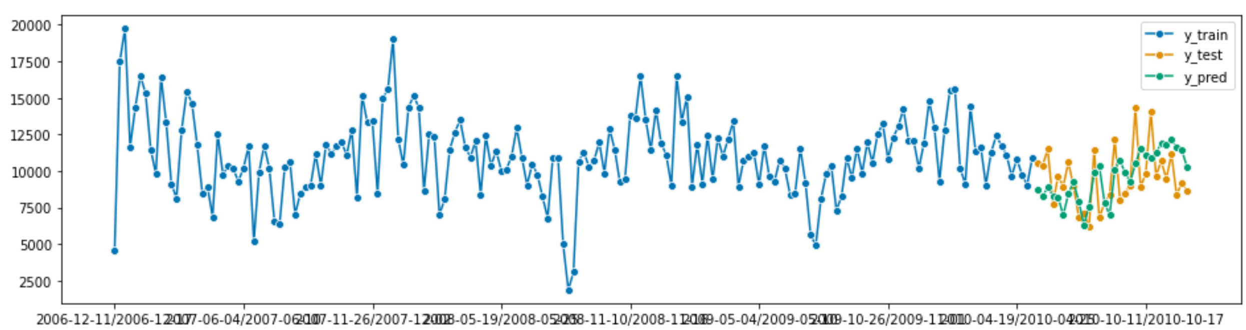
LSTM (Both Regular and Bidirectional layers have little impact on model performance) (Ensemble had no difference in mape)



Mape:20%

RMSE:2134

GRU

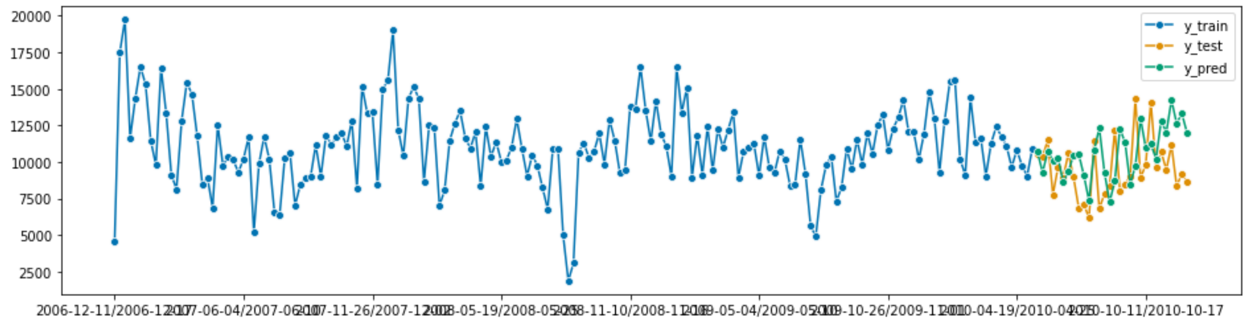


MAPE:18%

RMSE: 2012

GRU Proved to be the best model of all the NNs

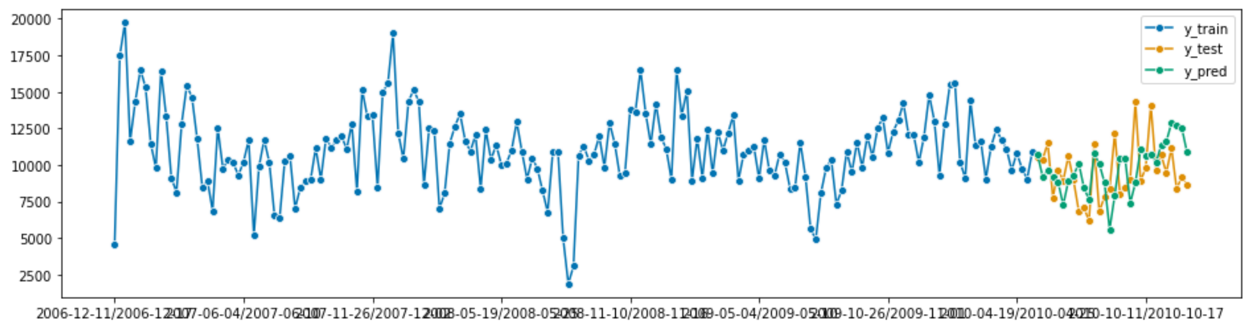
CNN GRU Hybrid Model



MAPE: 25%

RMSE:2699

Ensembled Model:

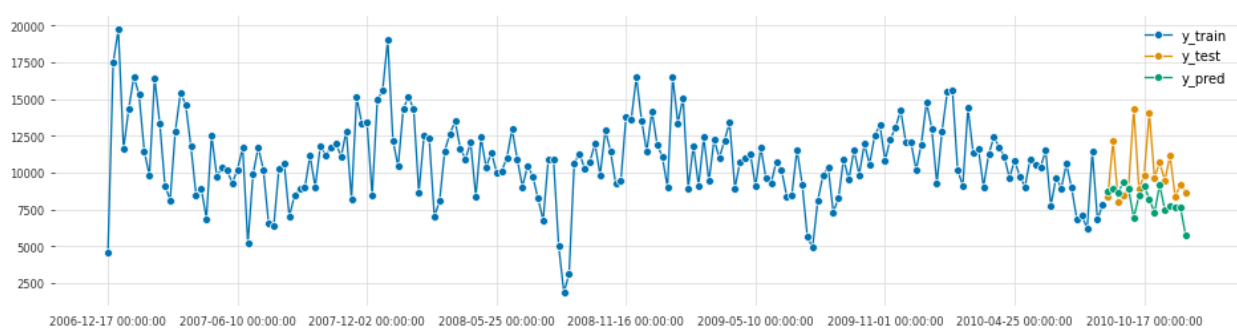


MAPE:21%

RMSE:2373

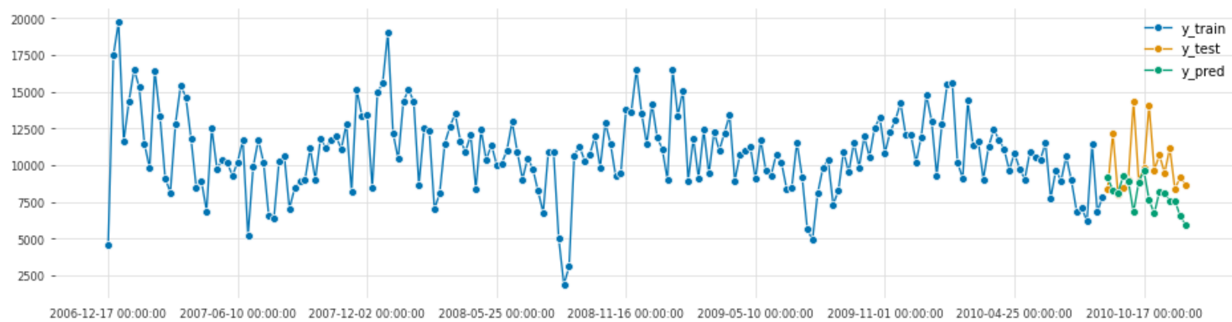
SOTA

DARTS Architecture



MAPE:19%

Nhits



y_train
y_test
y_pred
20000
17500
15000
12500
10000
7500
5000
2500
2006-12-17 00:00:00
2007-06-10 00:00:00
2007-12-02 00:00:00
2008-05-25 00:00:00
2008-11-16 00:00:00
2009-05-10 00:00:00
2009-11-01 00:00:00
2010-04-25 00:00:00
2010-10-17 00:00:00

MAPE:20%

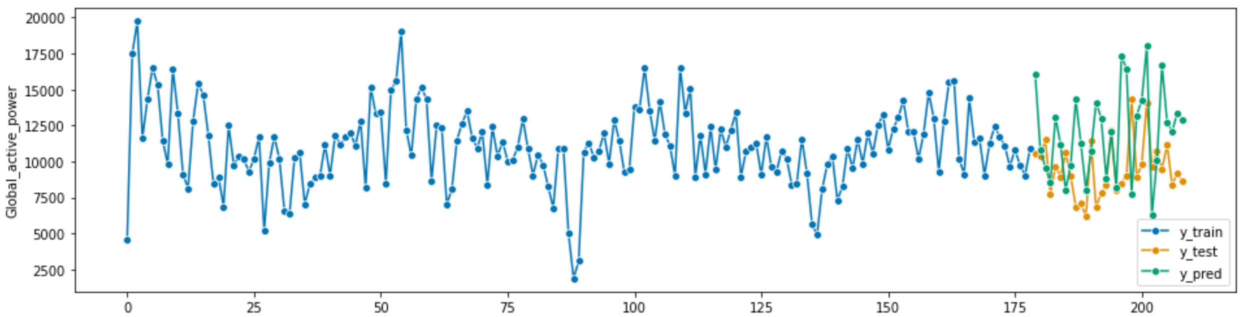
External Data Causal Inference:

To justify the use of external data the use of hypothesis testing is robust in evaluating data relevance in timeseries forecasting. However, given that our data is significantly nonlinear as seen from the BDS test, granger causality is not going to do a good job given that it works well on linear data. Thus, the optimal method for evaluating causality is the use of convergent cross mapping from the causal-ccm package. The package uses a novel technique called convergent cross mapping by Sugihara G. Using the statistical test, I concluded that the dataset. For our target timeseries, all the multivariate datasets do have a causal impact on it. However, we also inspected that our target timeseries also affects the other variables thus there is complex relationship in our dataset.

Multivariate Models & Ensemble forecasting

Multivariate

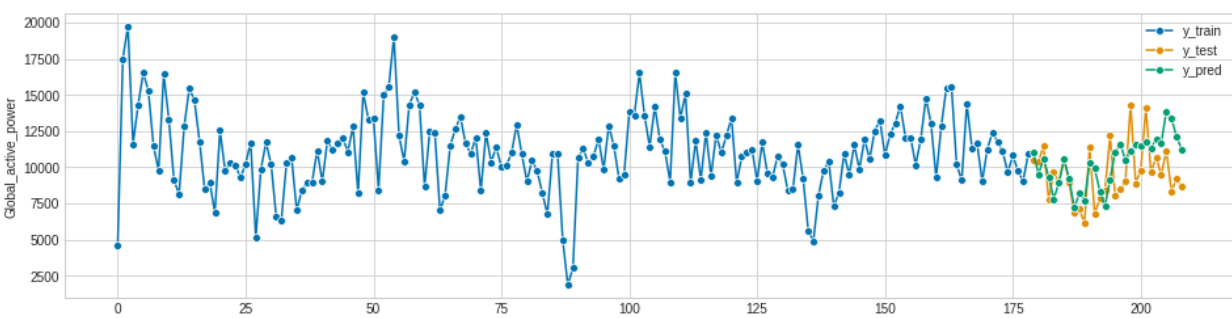
CNNLSTM:



MAPE: 30-40%

RMSE:4335

LGBM:

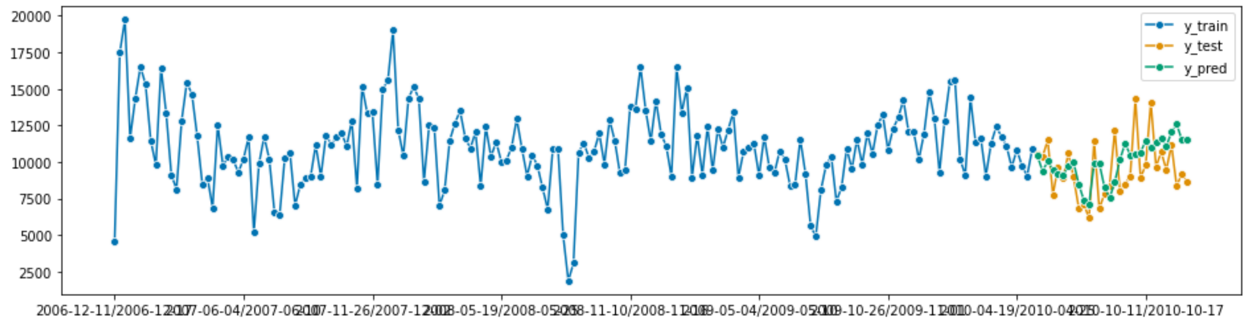


MAPE: 19%

RMSE:2136

Ensemble Forecasting:

ETS+SubSesonal Decompositionsal + Tree Based Model

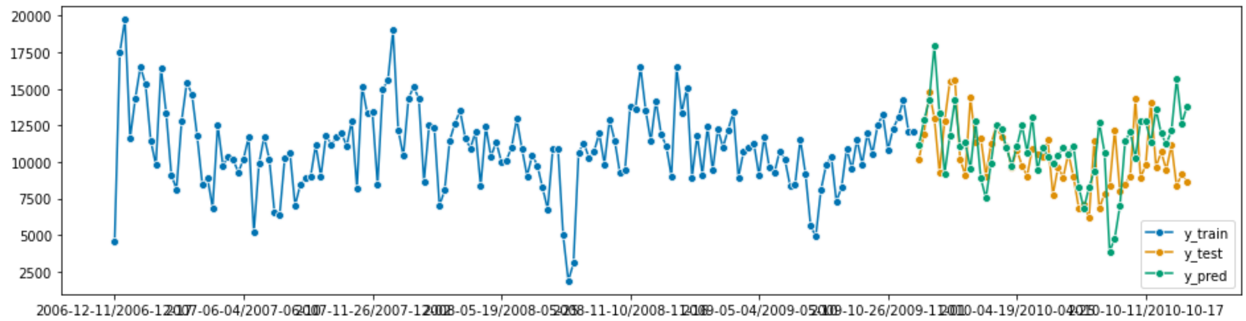


MAPE:18%

RMSE:2003

Forecasting of Individual Components of STL then adding them up together

XGBoost Algorithm is fitted on each component then added up.



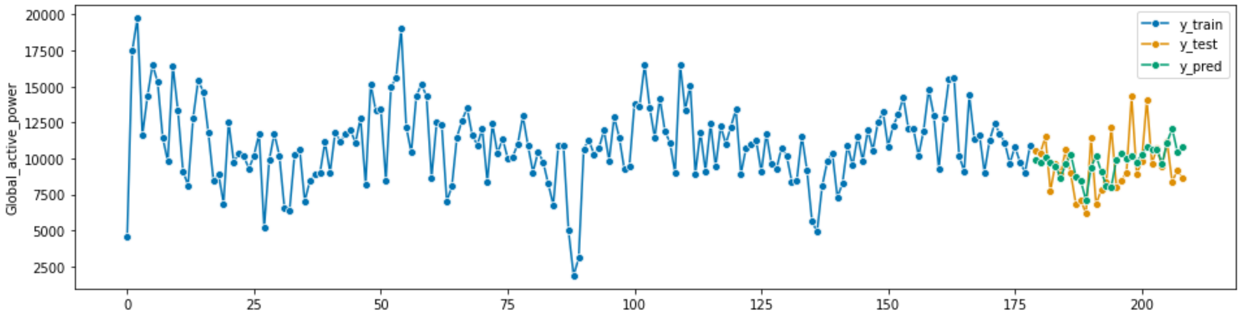
MAPE: 23%

RMSE:2932

(This method while useful in literature did not work well with the dataset we are working with. It is probable that for this dataset to be properly forecasted a multiple seasonal decomposition will be used to forecast the seasonality then add it to the dataset.

Ensemble Forecasting of Specialized Forecasters based on the complex seasonality in the dataset. (Optimal Method)

We use multiple identical algorithms with different sliding windows to forecast the timeseries. When using this method, we can reduce error since the specialized models will forecast the components effectively while still capturing annual seasonal patterns due to the ensemble.



MAPE:16%

RMSE: 1900

This method has been used in M-Competitions since it proved to be a powerful method when forecasting complex data. Often not it is recommended to go around 5-9 algorithms since this yields the most improvement to forecasting. We will optimize this model even further for better results.

Optimization:

We will use a flexible hyperparameter framework called optuna as it gives us the ability to optimize our model through a Bayesian optimization approach. The algorithm used for the Bayesian optimization is TPE.

Our metrics to optimize are

Mean Residual

Standard Deviation of Residual

RMSE

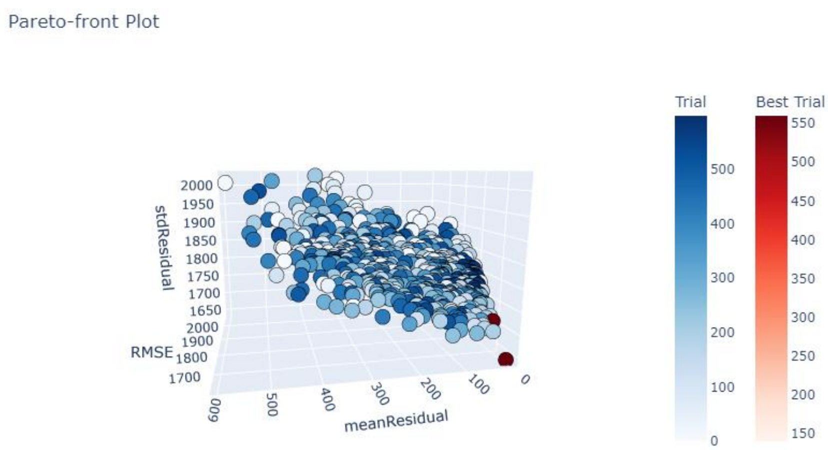
All 3 would help make sure that the model learns as much as possible from the dataset.

We optimize every single sub algorithm in the forecaster while still wrapping all 4 forecasters with LGBM using the feature importance from the LGBM model.

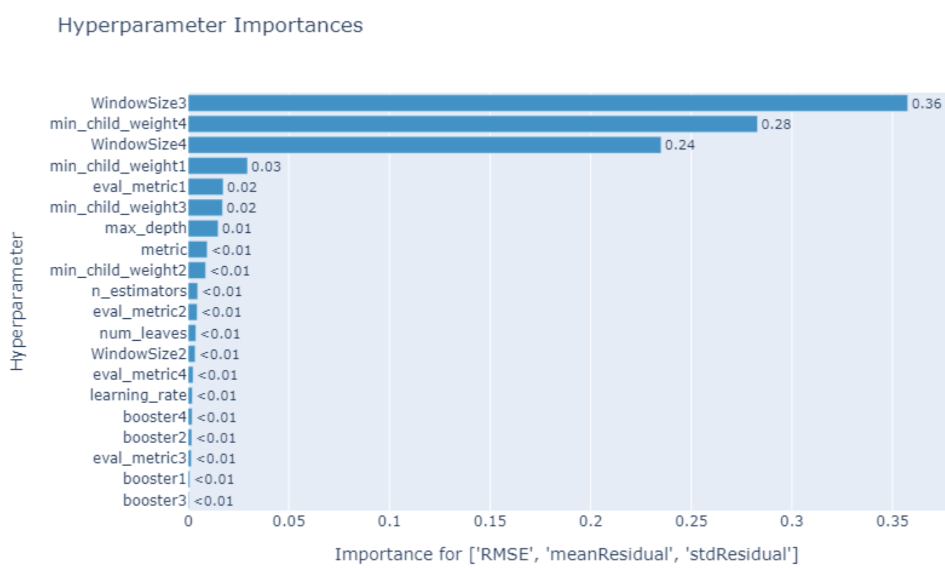
The parameters optimized are booster, evaluation metric, and min_child_weight as well every single sub window for the sub seasonal components.

We also optimize the LGBM model by optimizing the max_depth, num_leaves, number of estimators, learning rate, and the evaluation metric.

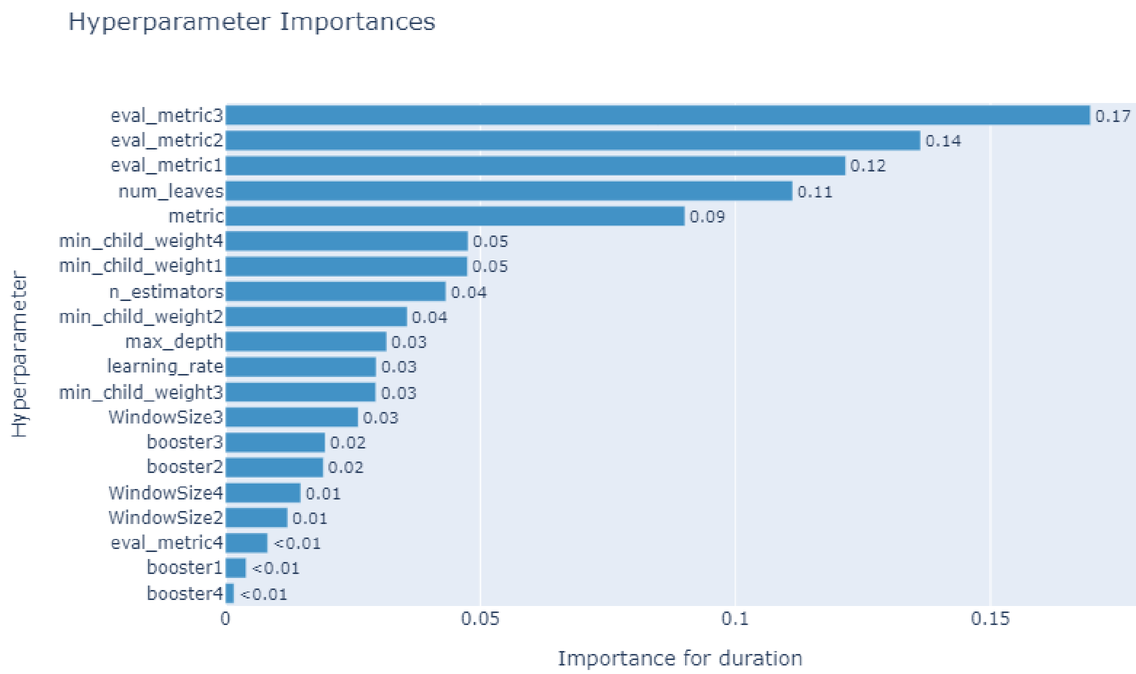
Here is a visualization of the iterations for all 3 metrics



Here is the importance of every single hyperparameter on performance

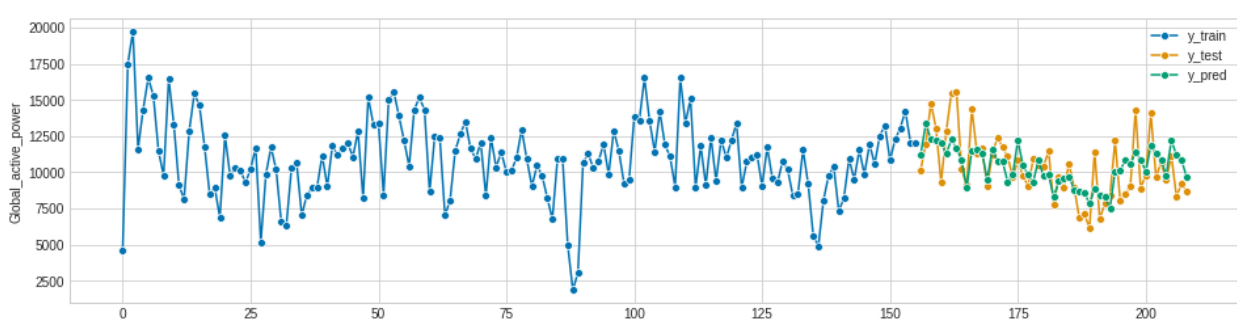


Here is the importance of every single hyperparameter on the duration of time it takes to train the model.



After storing and saving the study, we use the hyperparameter on

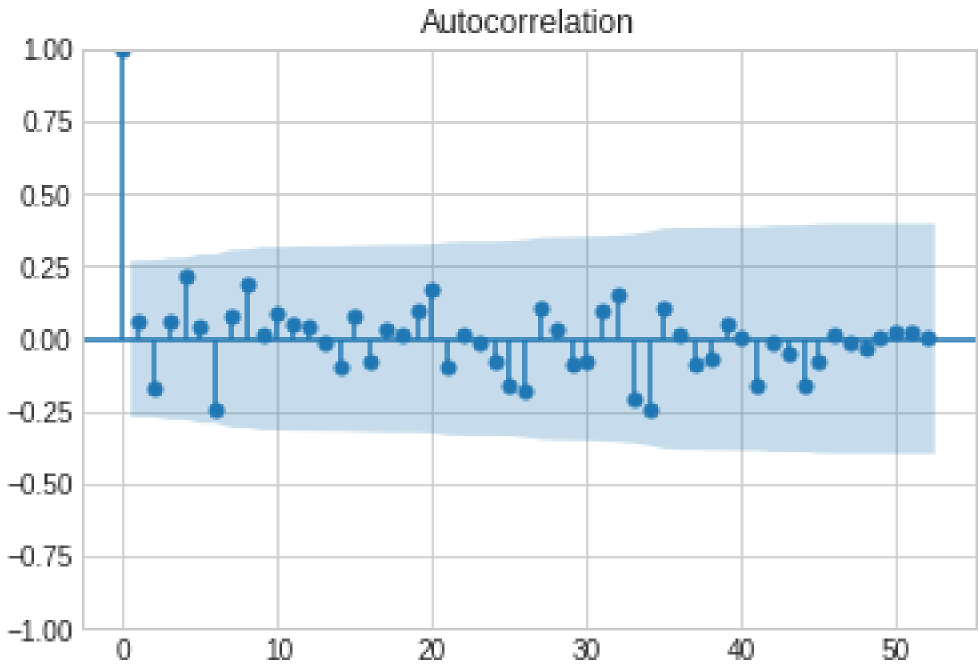
our dataset to see how this would improve the model



RMSE:1627

MAPE:12.97%

We also use the autocorrelation to see if our model learned from the dataset


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50