Forecasting Energy Load for Efficient Utilization of Battery Energy Storage Systems

Tim Prause, Kadisatou Fane, Cosima Birkmaier

Contents

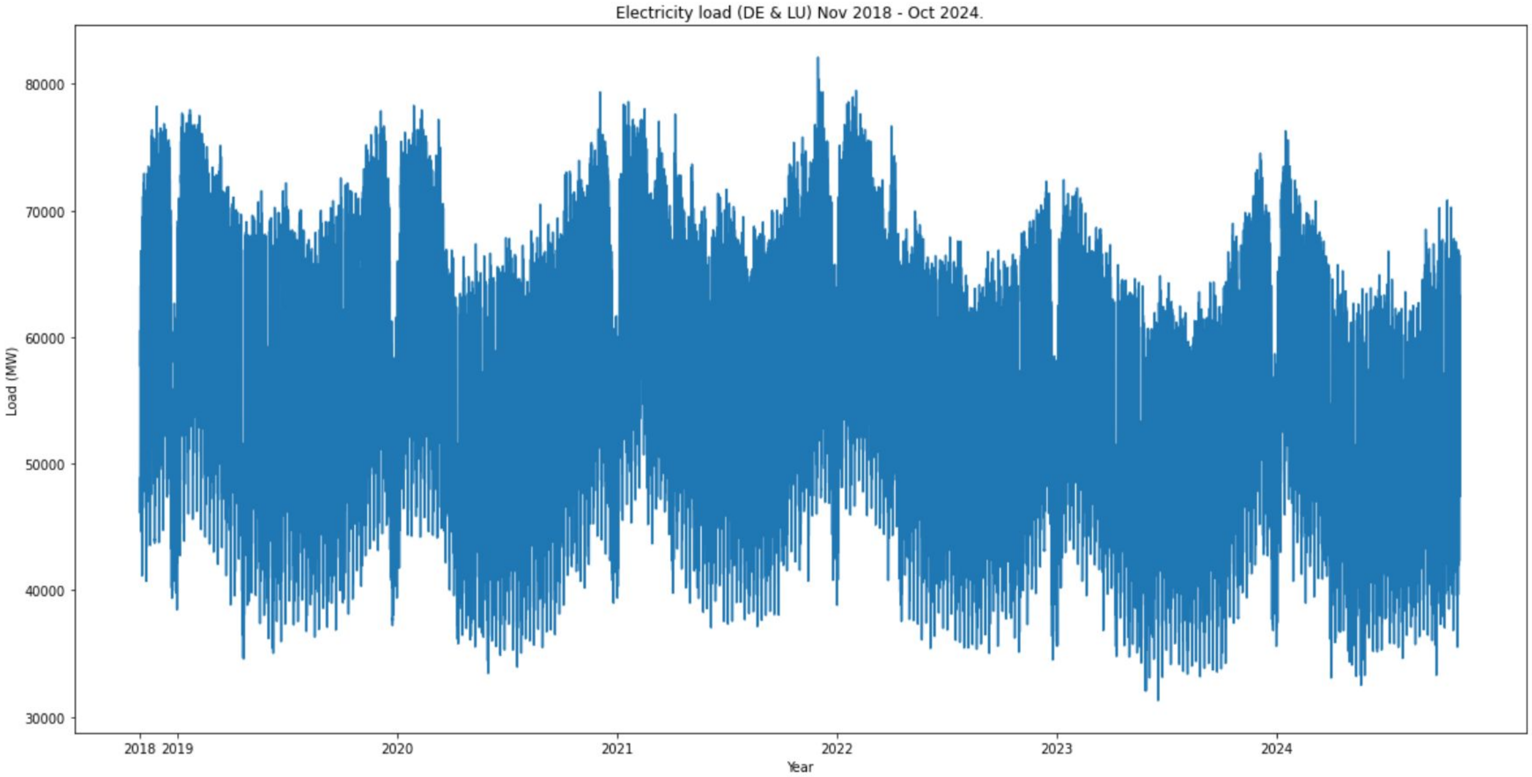
Motivation

- more renewable energy sources
- new challenges for power grids
- electricity generation must be flexibly adapted to power consumption
- one solution: **batteries:**
  - Battery Energy Storage Systems (BESS)
  - charge, when more energy is generated than needed
  - discharge to the grid, when demand is higher than (renewable) generation
  - financially: buy energy, when prices are low, sell it, when prices are high
    - supply and demand
- goal of our project: predict energy load for better battery charging and discharging

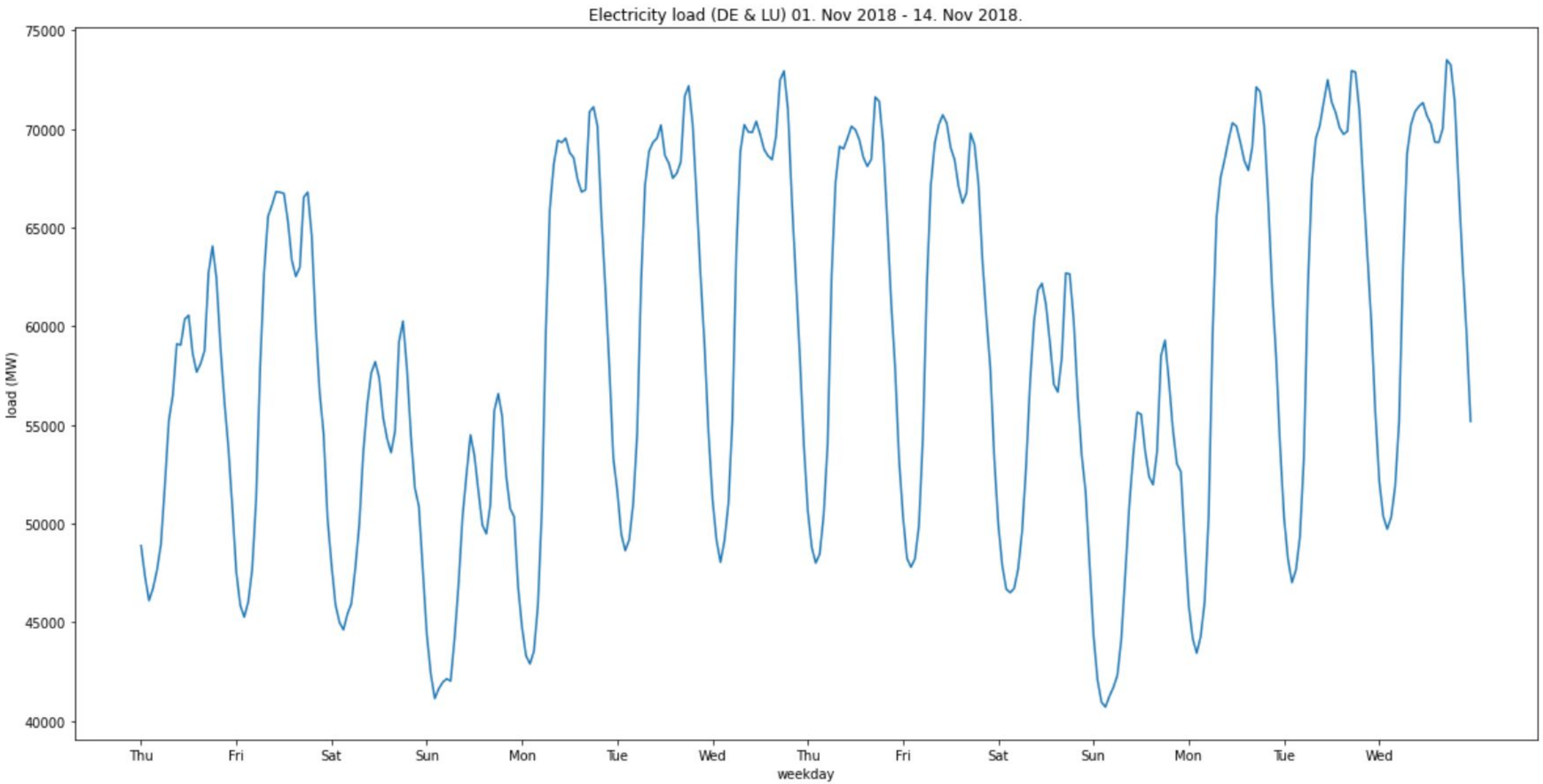
Dataset

- Dataset obtained from Bundesnetzagentur (Federal Agency for Grids)
- Total energy load from Germany and Luxembourg
- From 01.11.2018 to 31.10.2024
- Energy production per production type (e.g. wind, solar...)
- Hourly data
- Added features:
  - Time Features (hour, weekday, holidays...)
  - Energy Prices in Europe
- Final dataset:
  - 52,620 samples
  - 35 features

Data Visualization & Preparation



Data Visualization & Preparation



Train-Test split for all models:

Feature Engineering:

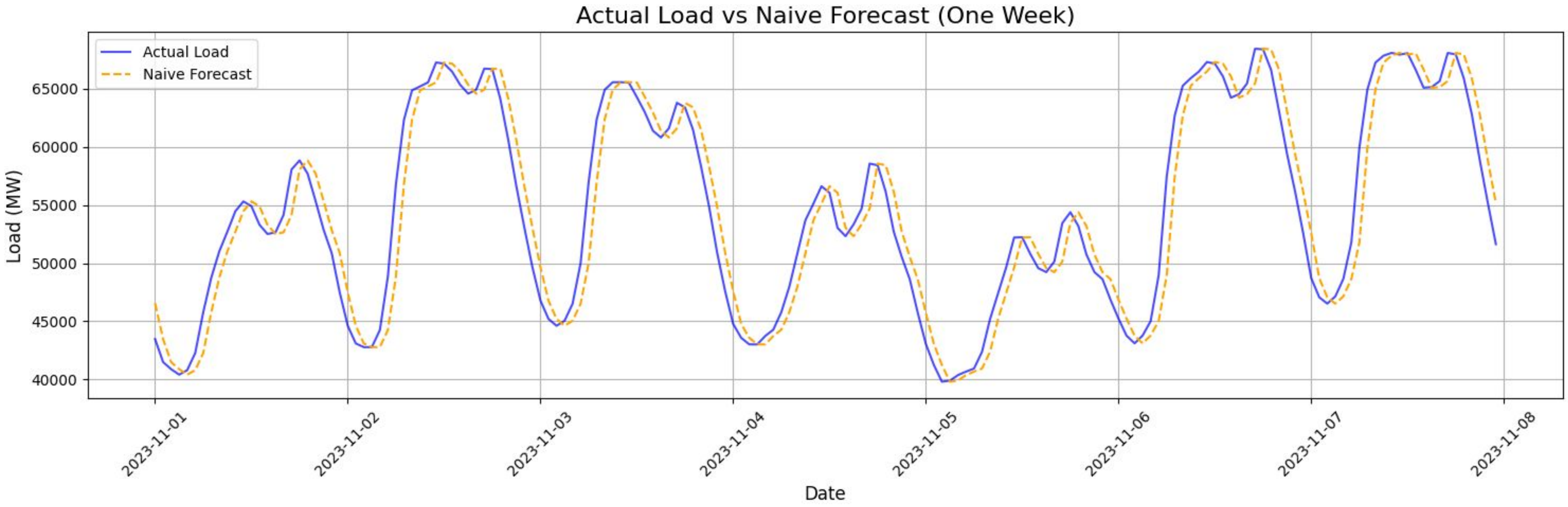
Added features:

- Time Features
    - hour of day in cos and sin
    - weekday cos and sin
    - day of year cos and sin
    - holiday
    - workday True or False
- Energy Prices in Germany
- Energy Production in Germany
    - Wind, solar, coal, gas...

Feature selection:

- Random forest Regressor for feature selection: Feature and importance
    - hour: 0.40
    - day of week: 0.086
    - nuclear energy: 0.028
    - holiday: 0.023
    - coal energy: 0.024
    - ...
- Threshold = 0.1 resulted in 13 remaining features

Baseline Models: Naive Forecast

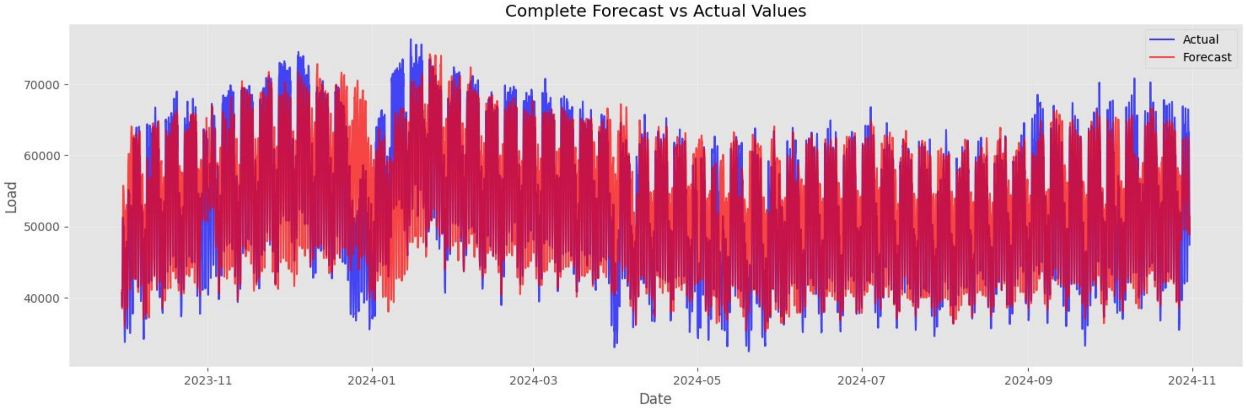


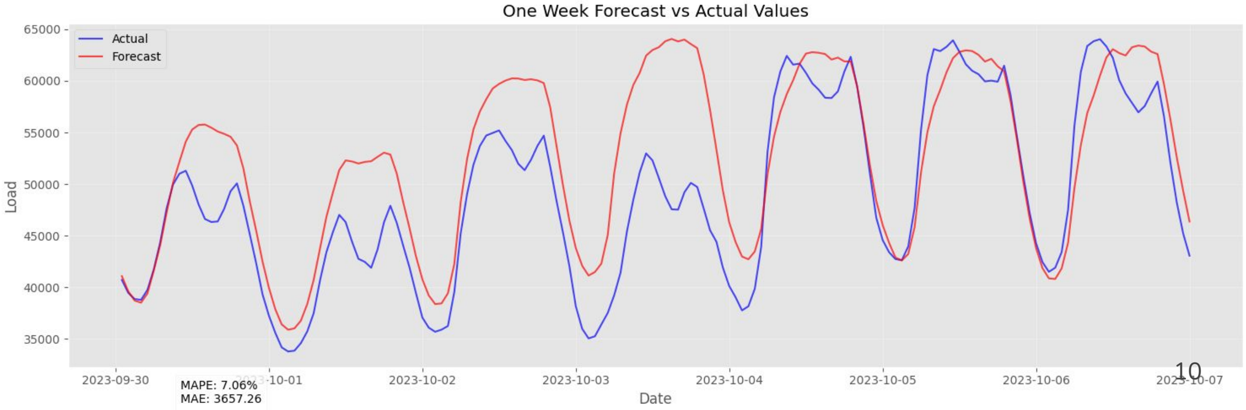
Mean Absolute Percentage Error (MAPE): 3.4592% Mean Absolute Error (MAE): 1807.92

Baseline Models: Sarimax 1

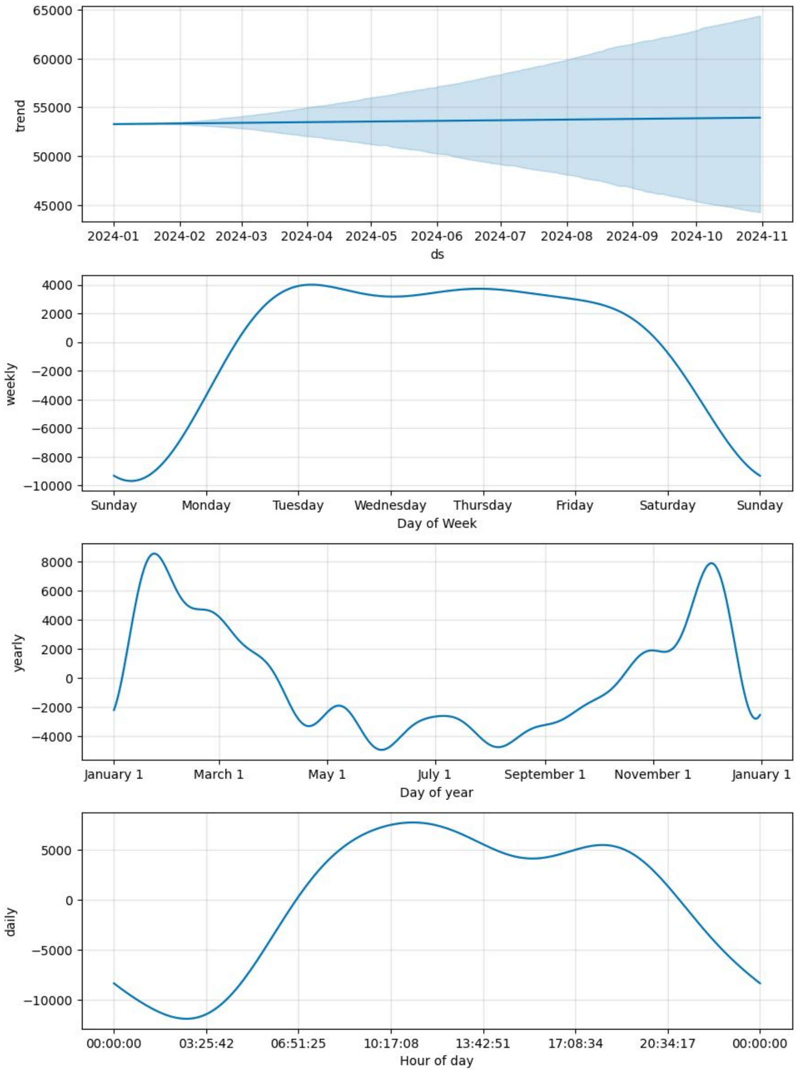
Mean Absolute Percentage Error (MAPE): 7.06%

Mean Absolute Error (MAE): 3657.26



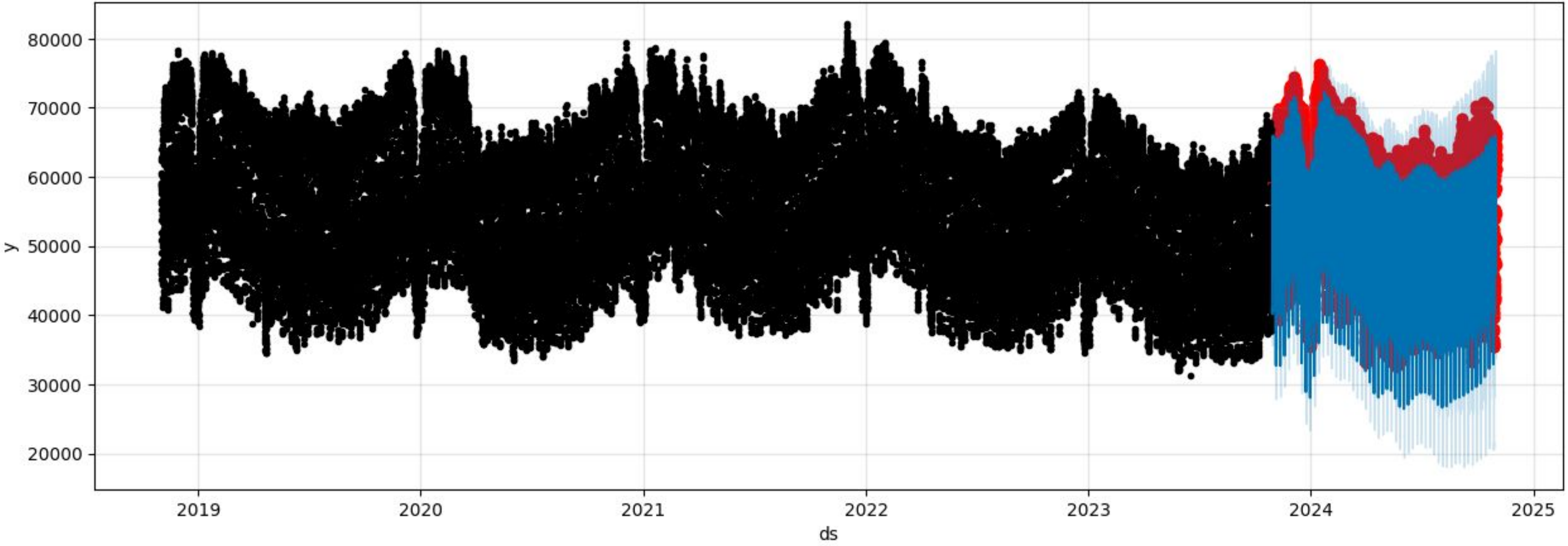


FB Prophet:



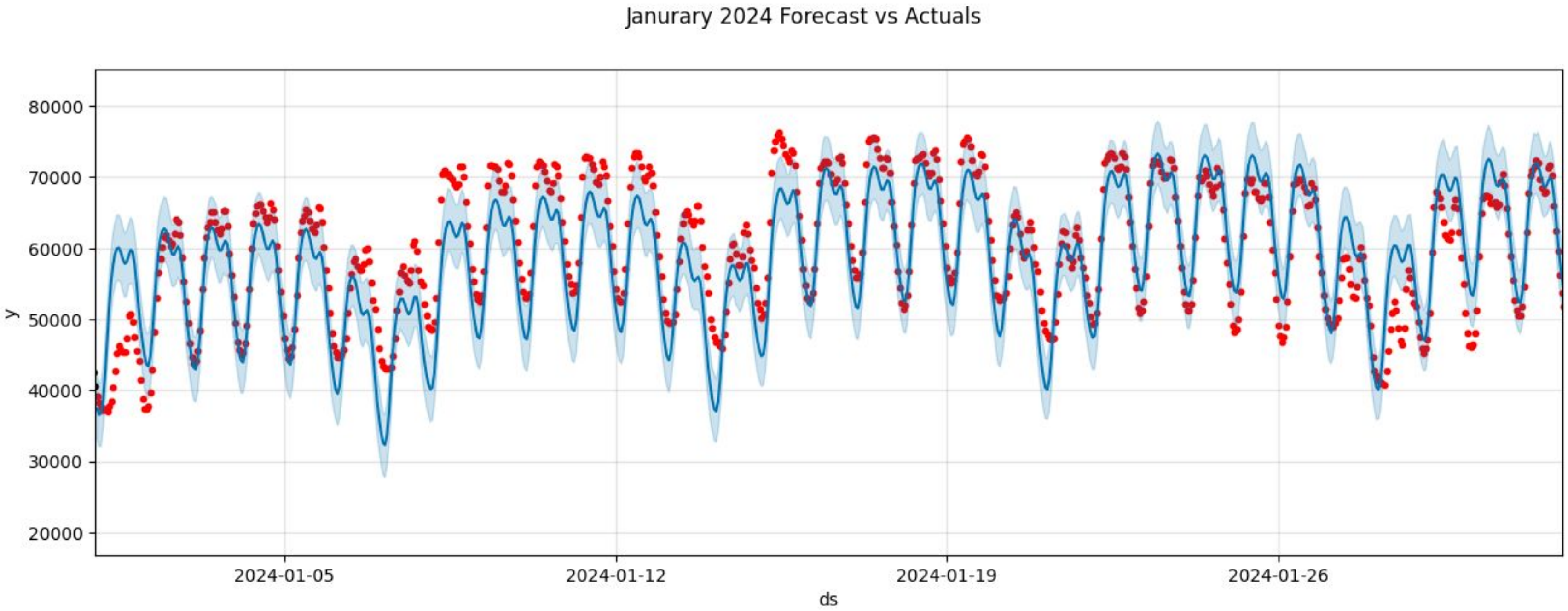
trend
65000
60000
55000
50000
45000
2024-01
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
2024-08
2024-09
2024-10
2024-11
ds
weekly
4000
2000
0
−2000
−4000
−6000
−8000
−10000
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Day of Week
yearly
8000
6000
4000
2000
0
−2000
−4000
January 1
March 1
May 1
July 1
September 1
November 1
January 1
Day of year
daily
5000
0
−5000
−10000
00:00:00
03:25:42
06:51:25
10:17:08
13:42:51
17:08:34
20:34:17
00:00:00
Hour of day

FB Prophet:



80000
70000
60000
50000
40000
30000
20000
y
2019
2020
2021
2022
2023
2024
2025
ds

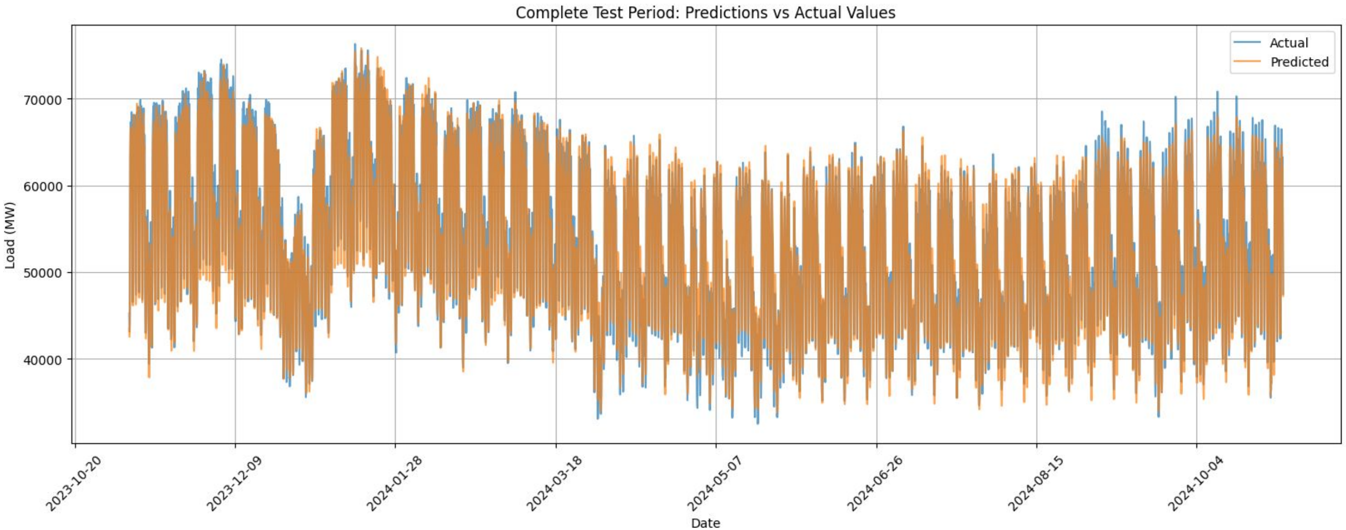
FB Prophet:

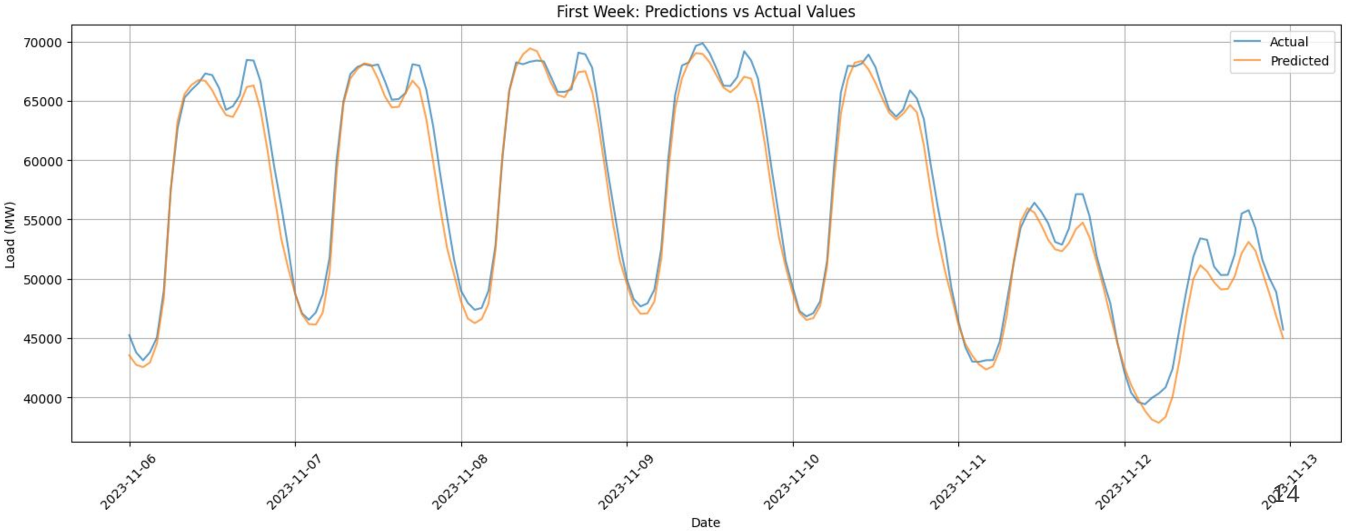


Mean Absolute Percentage Error (MAPE): 5.3319% Mean Absolute Error (MAE): XXX

LSTM:

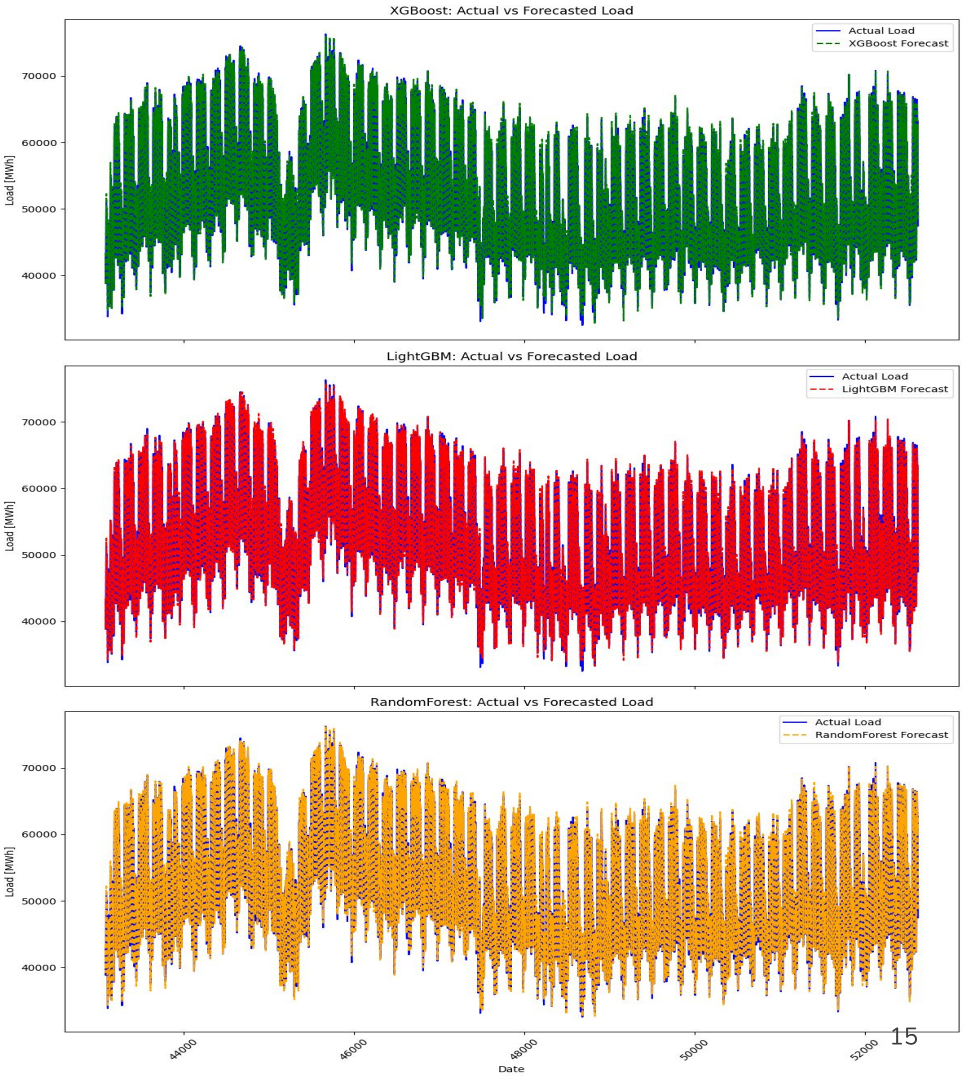
- Architecture:
  - 3 layers
  - 128 neurons
  - 0.2 dropout layer
  - 48 window size
- Test Set MAPE: 1.85%
- Test Set MAE: 957.33





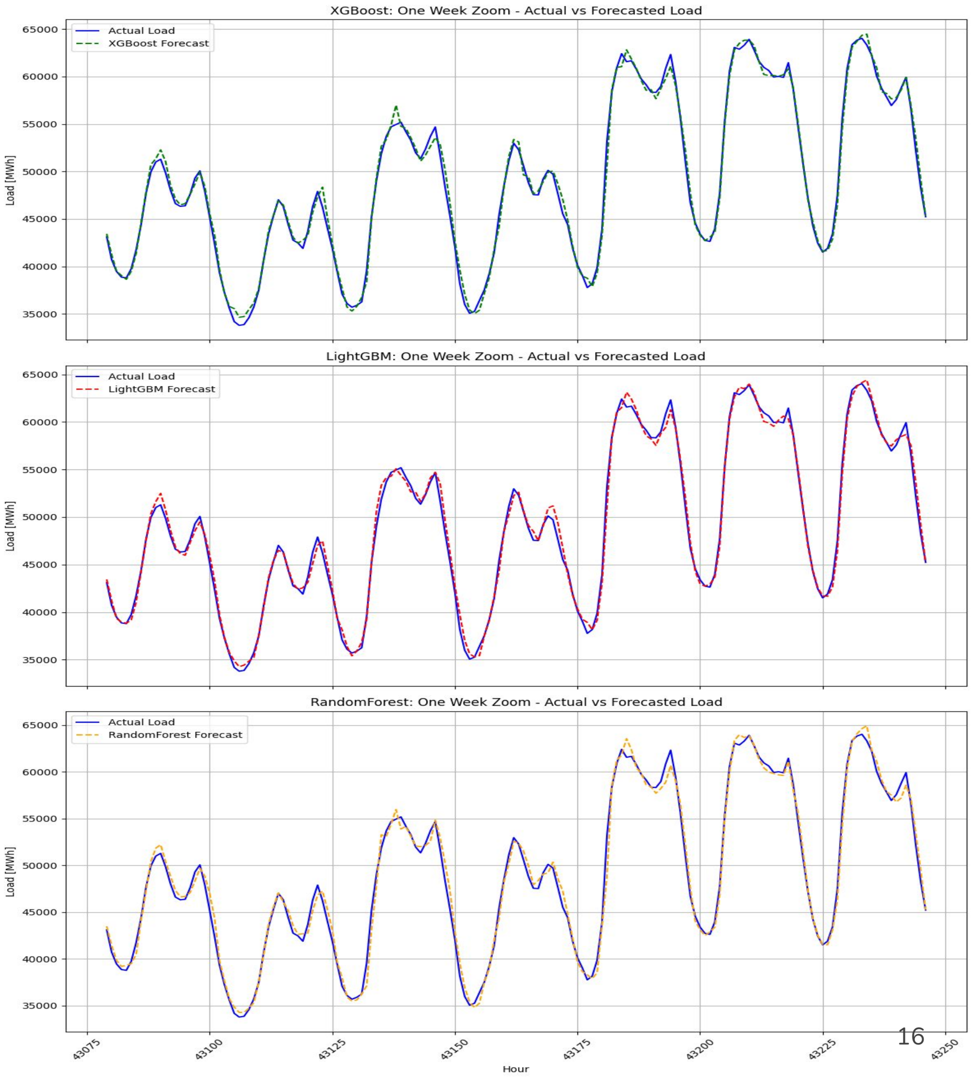
Multivariate Tree Models:

- XGBoost
  - 1000 estimators
  - 30 max depth
  - learning rate 0.01
- LightGBoost:
  - 3000 estimators
  - 35 max depth
  - learning rate 0.05
- RandomForest;
  - 30 estimators
  - 25 max depth



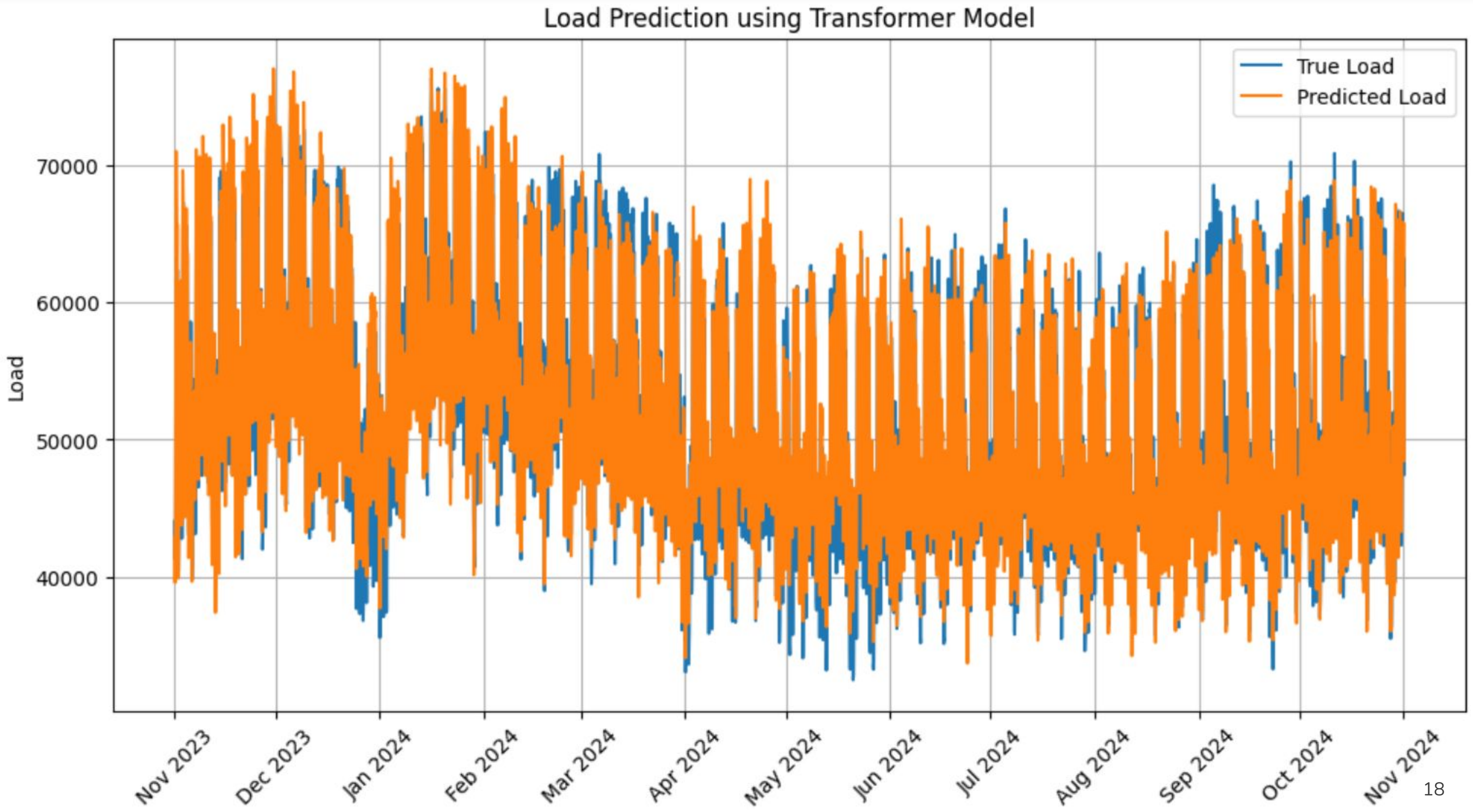
Multivariate Tree Models:

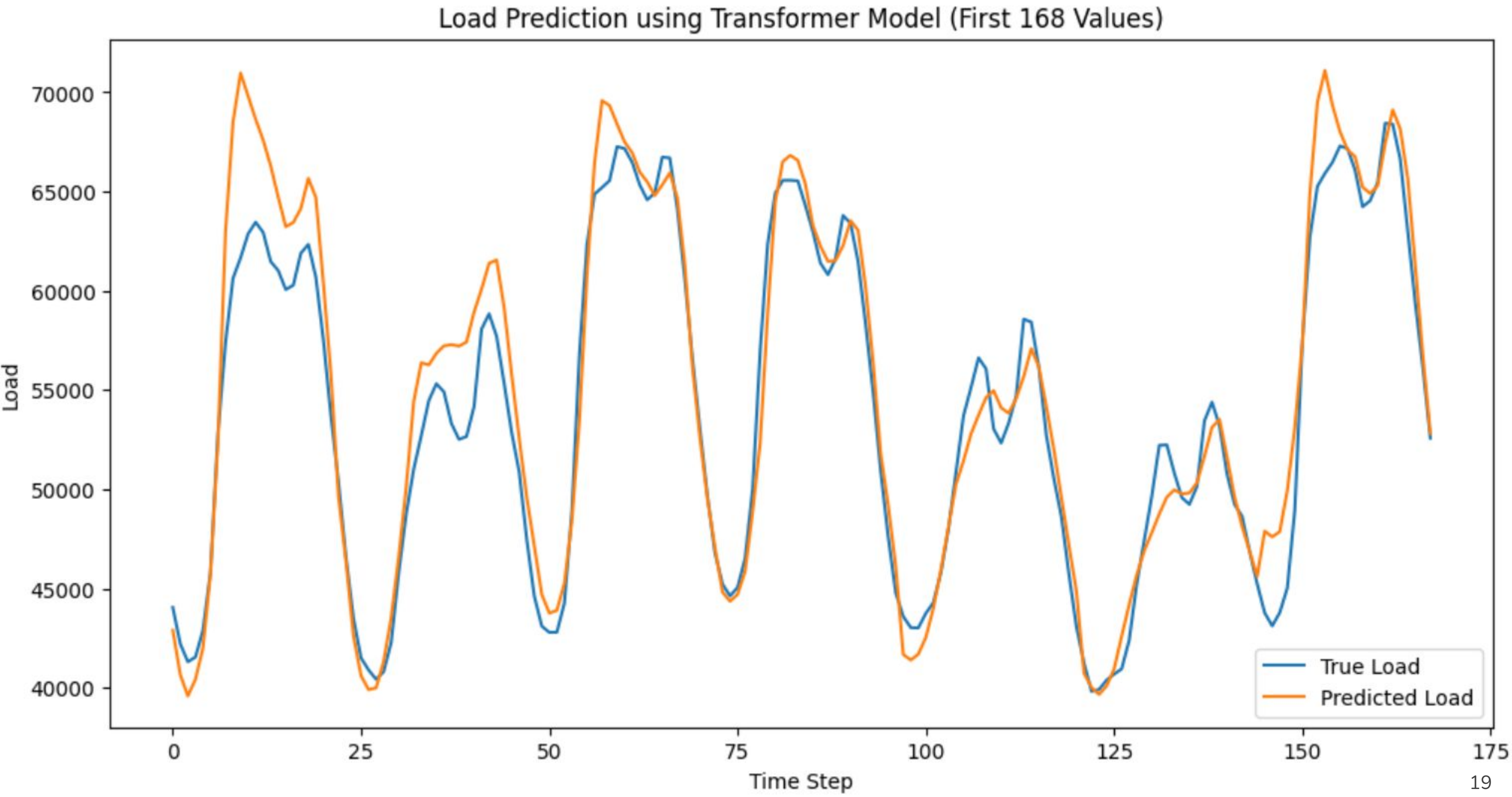
- XGBoost
  - mape: 0.86%
  - mae: 489.51
- LightGBoost:
  - mape: 1.09%
  - mae: 450.10
- RandomForest;
  - mape: 0.89%
  - mae: 515.27



Transformer Model

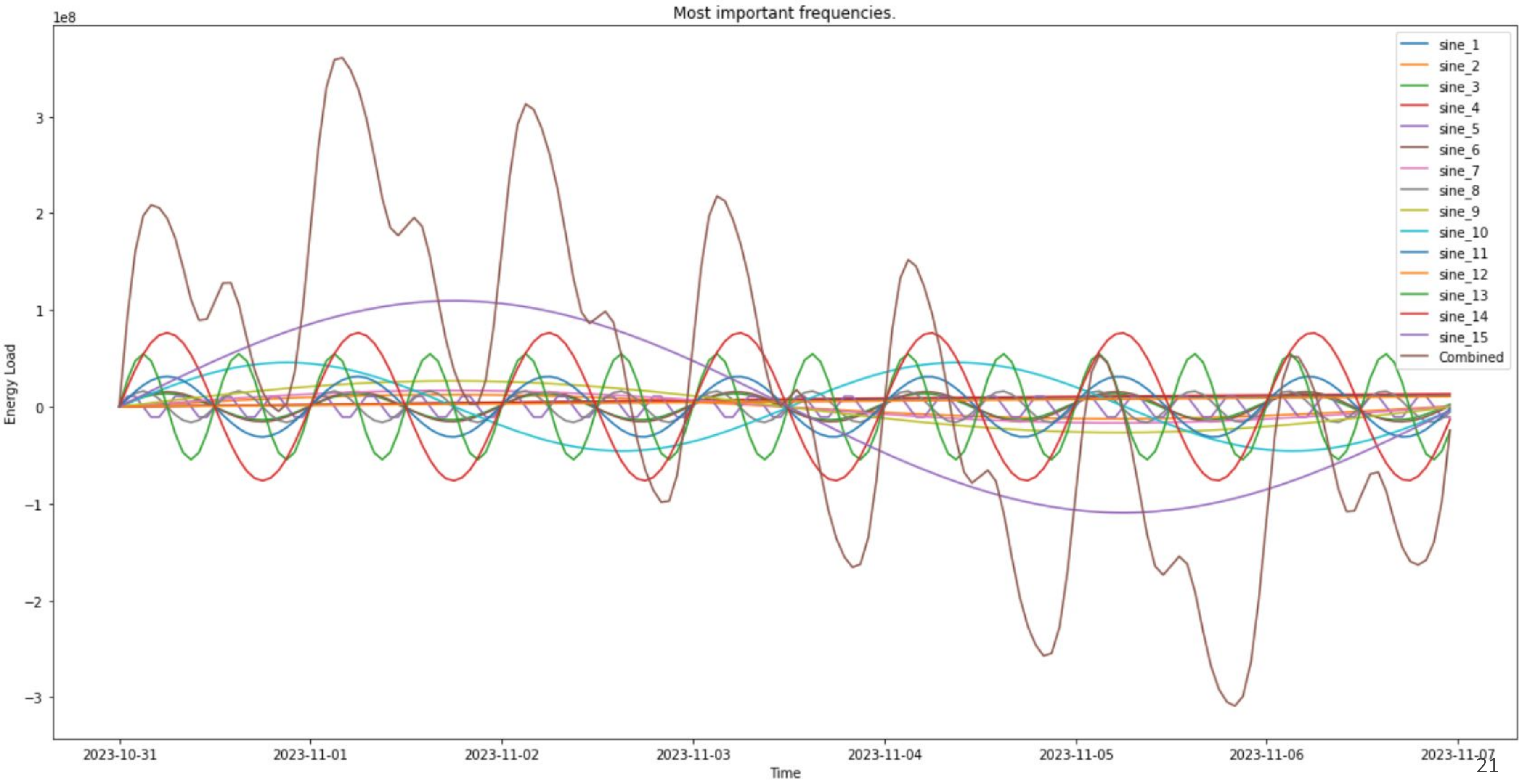
- TensorFlow
- 34,718 parameters (all trainable)
- 4 attention heads
- batch size = 1
- 50 epochs
- training time: 1:04 h
- MAPE: 3.69 %
- MAE: 1,898.06


Load Prediction using Transformer Model
True Load
Predicted Load
Load
70000
60000
50000
40000
Nov 2023
Dec 2023
Jan 2024
Feb 2024
Mar 2024
Apr 2024
May 2024
Jun 2024
Jul 2024
Aug 2024
Sep 2024
Oct 2024
Nov 2024

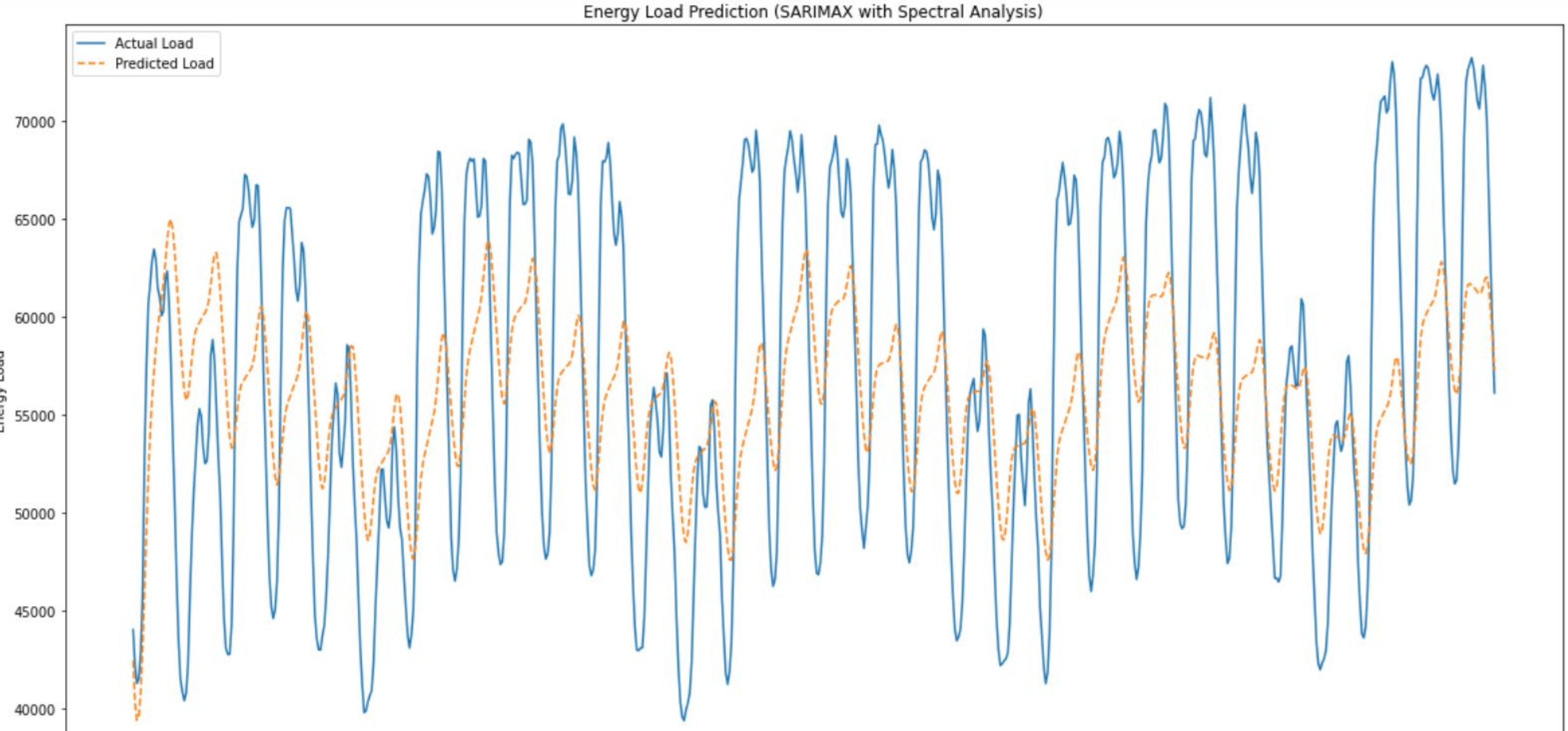

Load Prediction using Transformer Model (First 168 Values)
70000
65000
60000
55000
50000
45000
40000
Load
0
25
50
75
100
125
150
175
Time Step
True Load
Predicted Load

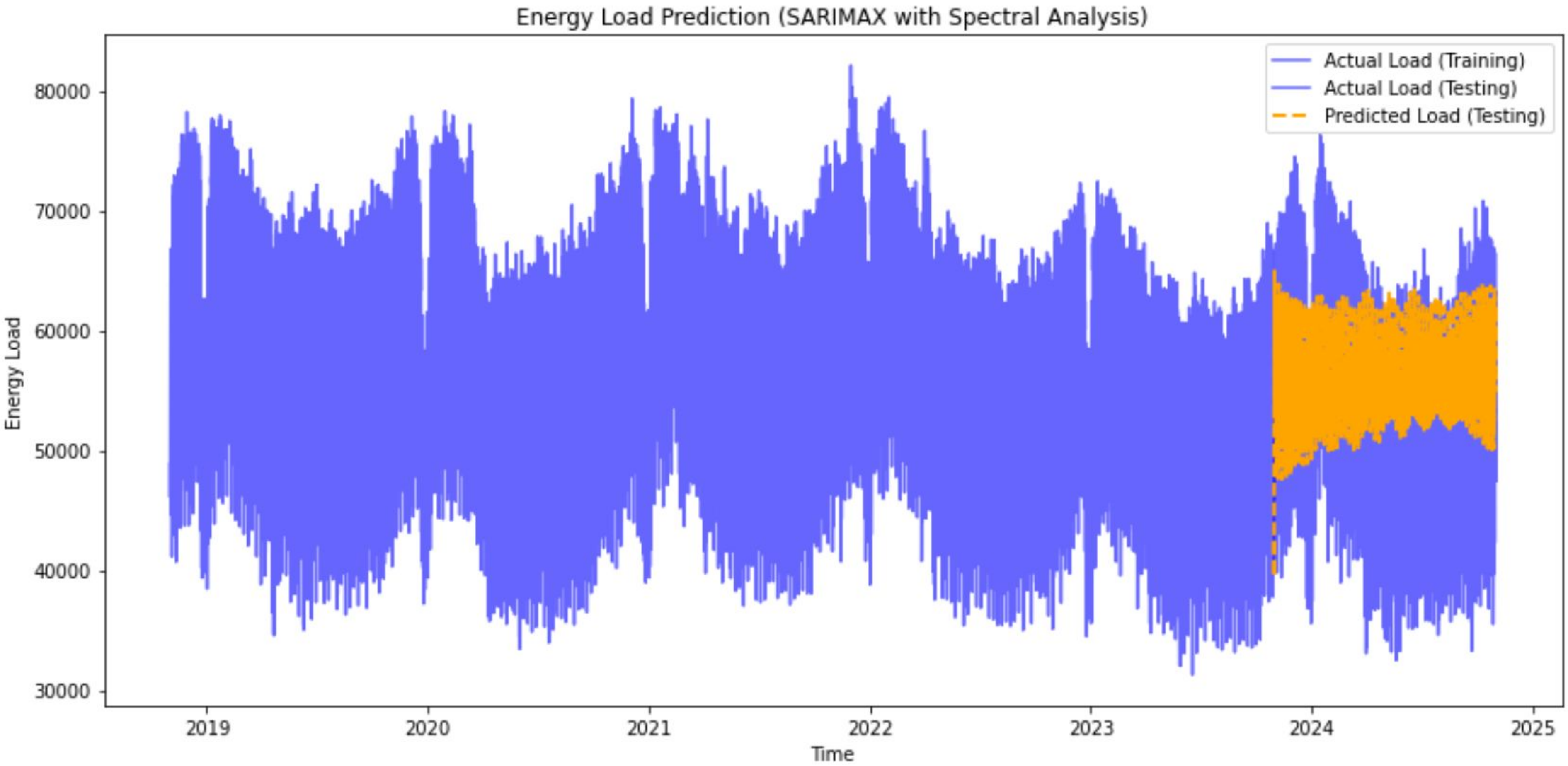
FFT and Spectral Analysis

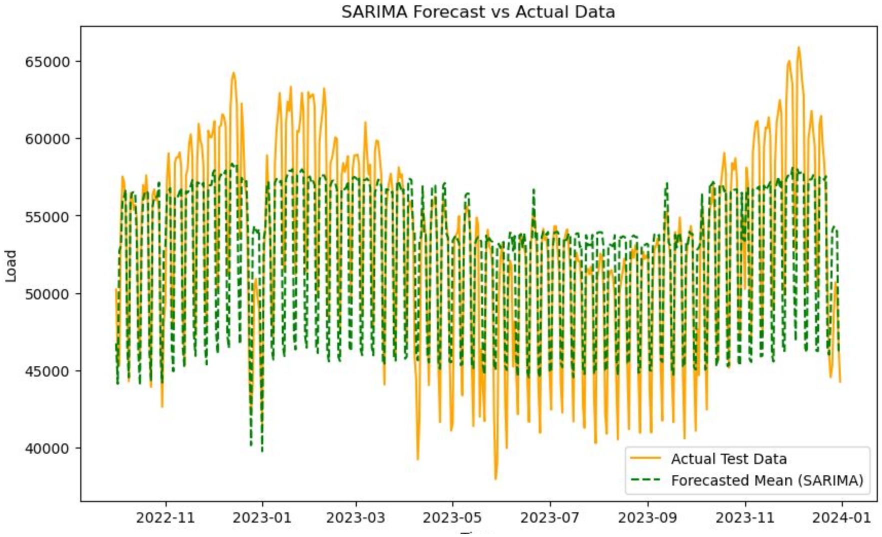
- Perform FFT
- Filter results to obtain most relevant frequencies
- Represent those frequencies with their amplitudes (and phases) as sine waves
- Add the sine waves as new columns to the dataset
- Perform SARIMAX with those columns as exogenous factors
- MAPE: 15.78 %
- MAE: 7,634.24
- Quick computation time (a few minutes for FFT, spectral analysis and SARIMAX)


Most important frequencies.
1e8
3
2
1
0
−1
−2
−3
Energy Load
2023-10-31
2023-11-01
2023-11-02
2023-11-03
2023-11-04
2023-11-05
2023-11-06
2023-11-07
Time
sine_1
sine_2
sine_3
sine_4
sine_5
sine_6
sine_7
sine_8
sine_9
sine_10
sine_11
sine_12
sine_13
sine_14
sine_15
Combined

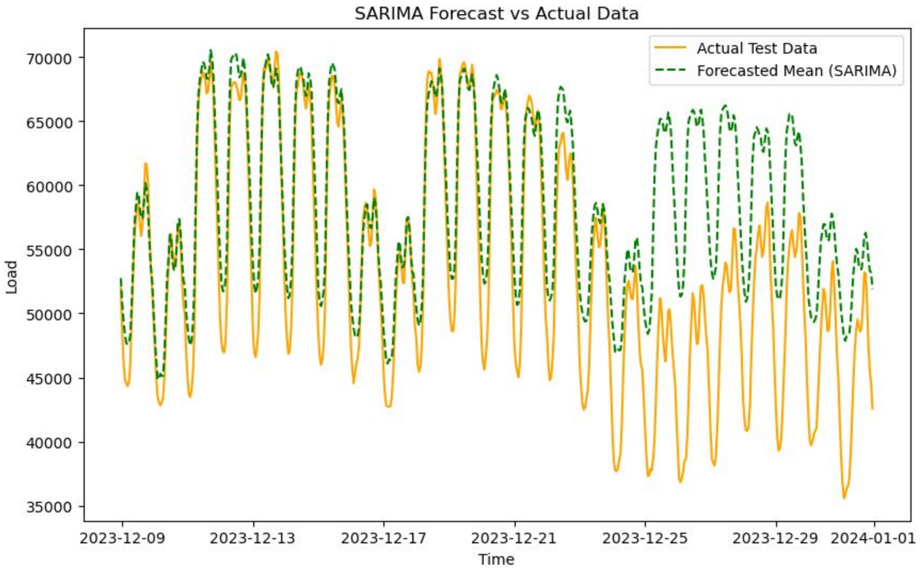
SARIMAX with FFT




Energy Load Prediction (SARIMAX with Spectral Analysis)
Actual Load (Training)
Actual Load (Testing)
Predicted Load (Testing)
Energy Load
80000
70000
60000
50000
40000
30000
2019
2020
2021
2022
2023
2024
2025
Time



Frequency = weekly using average hourly load
Interval: 2019-01-01:2023-12-31 | Data points: 1826
Split: 75%
Model : SARIMAX(4, 1, 0)x(1, 0, [1], 7)
Heteroskedasticity (H): 0.55
AIC:833.944
MAPE :0.048
Runtime: 94.254 seconds



Frequency = 24x7
Interval: 2023-10-01:2023-12-31 | Data points: 2160
Split: 75%
Model : SARIMAX(0, 0, 3)x(1, 0, [1], 168)
Heteroskedasticity (H): 0.31
AIC:-2532.929
MAPE:0.1061
Runtime: 12m59s

Kalman Filter with expansion window: Hourly load Forecast

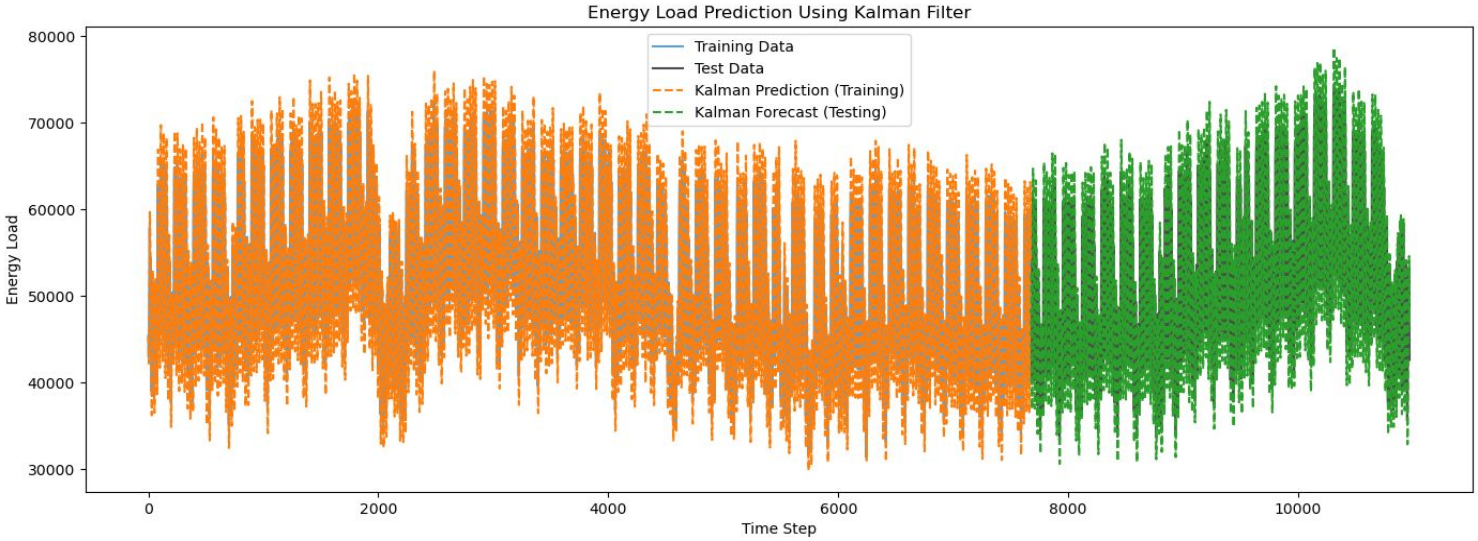
- Regressors: hour, is_weekend, is_holiday
- State variables: short-term trend & seasonality, long-term trend & seasonality
- Hyperparameters: state transition matrix (A), Observation matrix (H), Process noise covariance (Q), Observation noise covariance (R), Initial state vector, Initial state covariance

```
for t in range(forecast_steps):
    # Predict step
    state pred = A @ state prev  # Predict the next state
    P_pred = A @ P_prev @ A.T + Q  # Predict the state covariance

    # Predict energy load (observation)
    energy load pred = H @ state pred
    forecasted_energy_load.append(energy_load_pred[0])

    # Optional: Expanding or Rolling window correction
    if t < len(test data):
        # Use expanding window: update with test data as it becomes
available
        observation_residual = test_data[t] - energy_load_pred  #
Residual
        S = H @ P pred @ H.T + R  # Innovation covariance
        K = P_pred @ H.T @ np.linalg.inv(S)  # Kalman Gain

        # Update states based on test observation
        state updated = state pred + K @ observation residual
        P_updated = (np.eye(n_states) - K @ H) @ P_pred
    else:
        # For rolling window: Use the rolling window states
        state updated = state_pred
        P_updated = P_pred
```



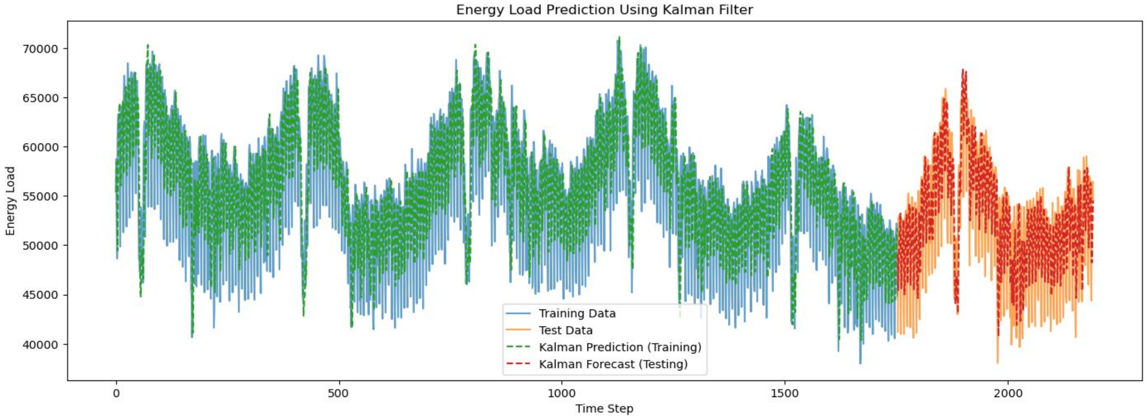
Frequency = hourly

Interval: 2022-10-01: 2023-12-31| Data points: 10969

split: 80%

MAPE: 0.0841

Runtime: 4 seconds



Frequency = daily

Interval: 2018-11-01: 2024-10-31| Data points: 2192

split: 80%

MAPE: 0.1007

Runtime: 4 seconds

GP model: another advanced architecture

- Features: load_lag_1, load_lag_2, load_lag_3, load_lag_4, load_lag_5, load_lag_6, load_lag_7, price, temperature, day_of_week, month
- Hyperparameters:
  - Out layer: **M**ean function, **R**BF kernel, **P**eriodic Kernel, **M**atern Kernel, **G**aussian Likelihood noise variance, Learning rate, Number of iterations
  - In layer: length scale, periodic length,v, output scale
  - RBF Kernel models smooth trends over multiple features.
  - Periodic Kernels captures yearly and weekly seasonality (based on time-related features).
  - Matern Kernel adds flexibility for irregular variations.

Model Comparison

Model	MAPE	MAE	training time (estimated)
Naive Forecast	0.034592	1807.92	—
Sarimax 1	0.0706	3657.26	10 min
FBProphet	0.053319	---	4 min
LSTM	0.0185	957.33	20 min
Univariat XGBoost	0.038462	1988.07	4 min
Multi XGBoost	0.0086	489.51	10 min
Multi LGBoost	0.0109	450.10	3 min
Multi Random Forest	0.0089	515.27	7 min
Transformer	0.0369	1898.06	1 hour
Sarimax with FFT & Sine Waves	0.1578	7634.2403	2 min
Sarimax 2 Daily load forecast	0.048	—	1m34s
Kalman Filter with expansion window	0.0841	—	4s
Gaussian Process	(0.02795)	—	<1 min
Times FM	0.0332	__	< 1 min

Outlook:

- Add more Features
- More fine tuning for each model
- Use models optimized for rolling forecasts
- Forecast Energy price as well
- Forecast different regions
- Consider official and unofficial regional holidays (carnival) and the regions affected
- Consider long “bridge” weekends
- Reproduce the models with alternative set of features

References

- https://www.smard.de/home/downloadcenter/download-marktdaten/?downloadAttributes=%7B%22selectedCategory%22:1,%22selectedSubCategory%22:false,%22selectedRegion%22:false,%22selectedFileType%22:false%7D
- International Energy Agency (2024): Batteries and Secure Energy Transitions. https://iea.blob.core.windows.net/assets/cb39c1bf-d2b3-446d-8c35-aae6b1f3a4a0/BatteriesandSecureEnergyTransitions.pdf
- Lewinson, E. (2022): Three Approaches to Encoding Time Information as Features for ML Models. https://developer.nvidia.com/blog/three-approaches-to-encoding-time-information-as-features-for-ml-models/

Photo credit

Slide 3, top center:

https://unsplash.com/de/fotos/luftaufnahme-des-rasenplatzes-mit-blauen-sonnenkollektoren-Ilpf2eUPpUE

Slide 3, right:

https://pixabay.com/de/photos/windrad-energie-windkraft-5267130/

Slide 4, top center:

https://www.bundesnetzagentur.de/SharedDocs/Bilder/_init_bild_stromleitung.png?__blob=normal&v=3

Slide 4, right:

https://unsplash.com/de/fotos/weisse-windmuhle-auf-gelbblattrigem-blumenfeld-tagsuber-E56cTF65xFw