Approach

- Started with loading the data on Jupyter notebook
- Analyzed the statistics like mean, min, max, and quartiles.
- Checked the null values in the dataset.
- Analyzed the load profile for all the households by plotting the relevant graphs.
- Analyzed outliers in the energy consumption data but did not remove the outliers at this stage because more information was needed like the weather data and occupancy. These factors can be the reason for peaks in the energy data.
- Extracted some of the important features from the time column like hour, day_of_week, and month as these are some of the important features that can affect the load of various households.
- Trained 3 models baseline(persistence model), ARIMA(statistical model), and XGboost.
- Individual models were trained/tested for each household as it has more value instead of a single aggregated model. This was done using for loop.
- XGboost performed exceptionally well for all households, it had higher accuracy and was able to capture the trends as well.

<u>Model Deployment Strategy</u>

1. Deployment using API:
    a. Frameworks like Flask and FastAPI can be used to build API services.
    b. A pre-trained model can be used by building a pickle file
    c. New data can be injected using POST to return predictions for the next 24 hours
2. Batch predictions
    a. Predictions can be triggered at a predefined time every 24 hours using Airflow
    b. The latest historical data can be accessed from a database where all the data is stored. Using this recent data predictions can be done and stored in databases like PostgreSQL or MySQL.
    c. Cloud services like AWS Lambda. S3 bucket and CloudWatch can be used to achieve this. I have performed this job in one of my projects where I did Spotify data processing in batches.

<u>Monitoring the model performance and updating the models</u>

1. Storing actual and predicted values in a database to dynamically compute MAE, RMSE, and MAPE after each prediction window (24 hours).
2. Real-time performance can be evaluated by constantly analyzing these values.
3. Dashboards can be built to monitor any anomalies or deviations.
4. We can set up automated retraining of the model to ensure model prediction does not drift beyond a certain limit/threshold.
5. Another approach for model retraining can be to schedule it every week, with the recent historical data.