faculty

InterCast
Process Presentation

National Grid ESO
21/11/2023

Contents

faculty

Project plan

Forecasting hourly electricity prices for North Sea Link (NO – GB)

Moving Average Method

- Naive model assuming the trend and seasonality components of the time series have already been removed or adjusted for

- Can easily be used in a walk-forward manner over data set

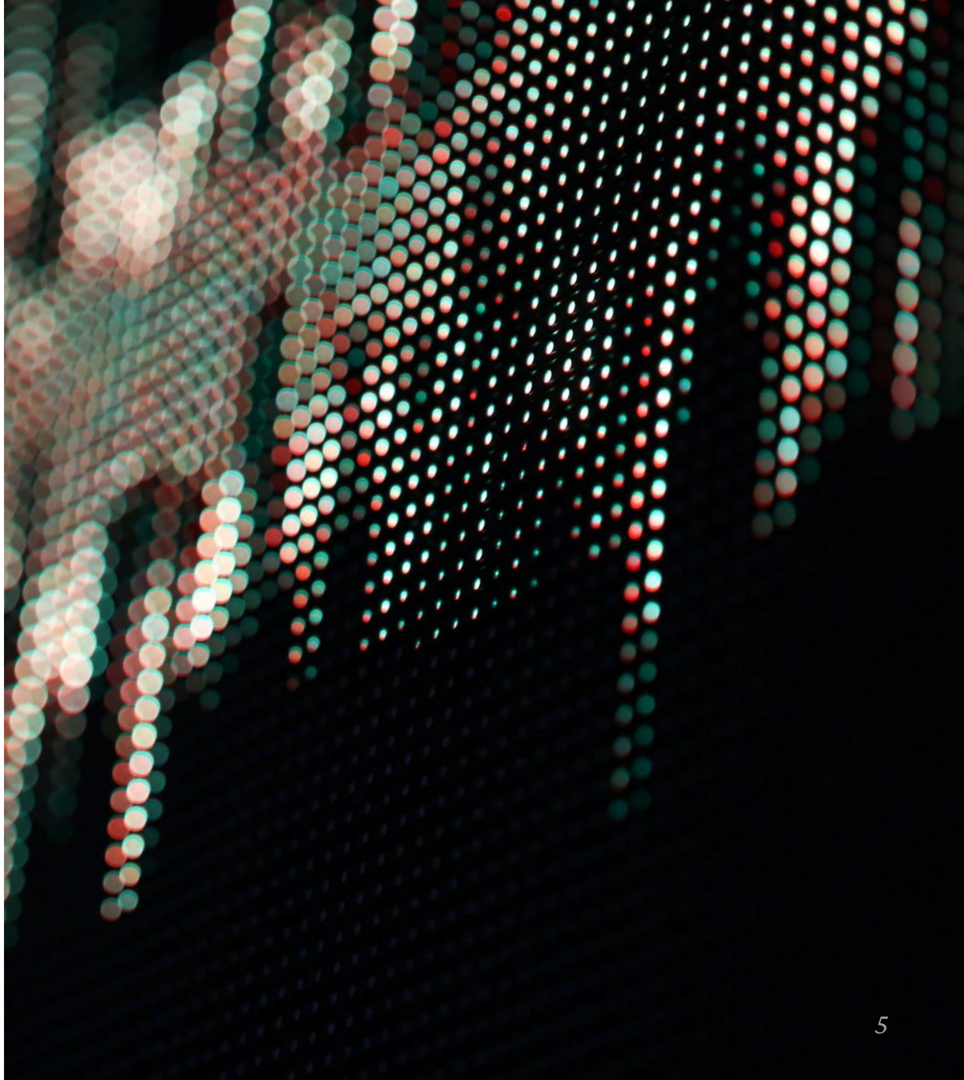
- As new observations are made available (e.g. hourly), the model can be updated and a prediction made for the next hour

Autoregression Model

- A linear regression model that uses lagged variables as input variables
- Particularly valuable for capturing the temporal dependencies and patterns within time series data
- Number of lags as indication for number of considered past values to model the current value

ARIMA

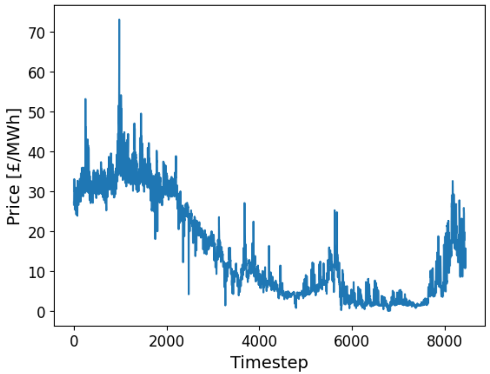
- ‘AutoRegressive Integrated Moving Average’, is a forecasting algorithm based on the idea that the information in the past values of the time series (its own lags and the lagged forecast errors) can alone be used to predict the future values
- Characterized by 3 terms: p, d, q
- Auto_ARIMA: function that performs a search over possible model orders within the constraints provided, and selects the parameters that minimise the given metric



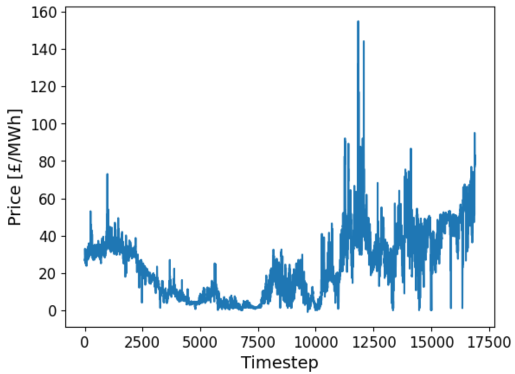
ARIMA cross-validation (price)

- Order parameters
  - Fold 1: **(6, 1, 1)**
  - Fold 1 + 2: **(6, 1, 5)**
  - Fold 1 + 2 + 3: **(2, 1, 3)**

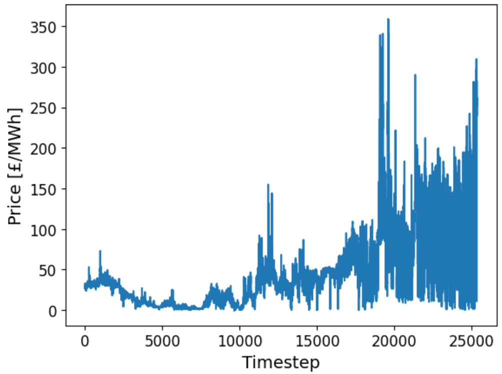
Fold 1	**Fold 2**	**Fold 3**	**Fold 4**	**Validation set**



2019/09/26 – 2020/09/11



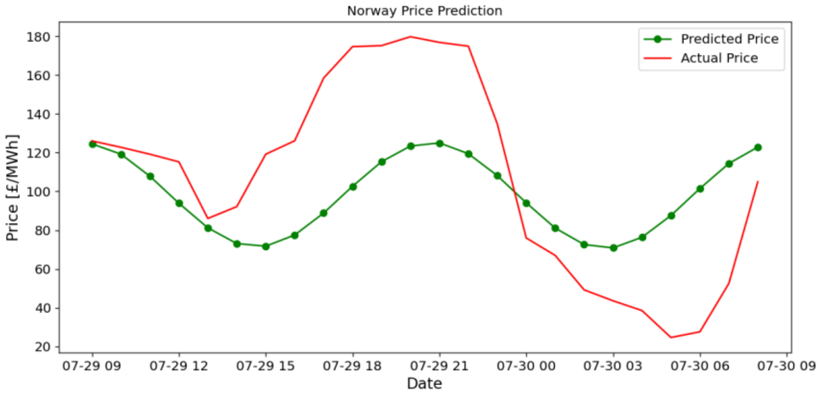
2019/09/26 – 2021/08/29

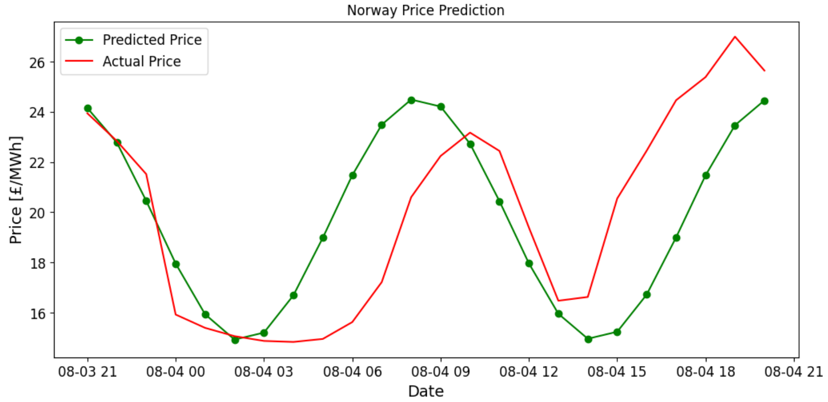


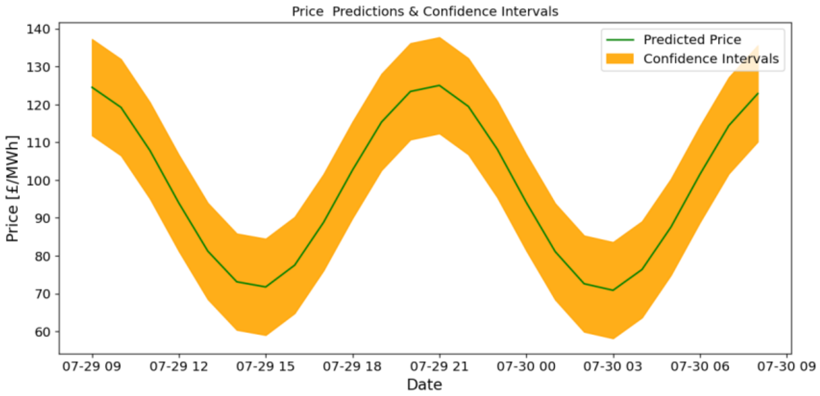
2019/09/26 – 2022/08/16

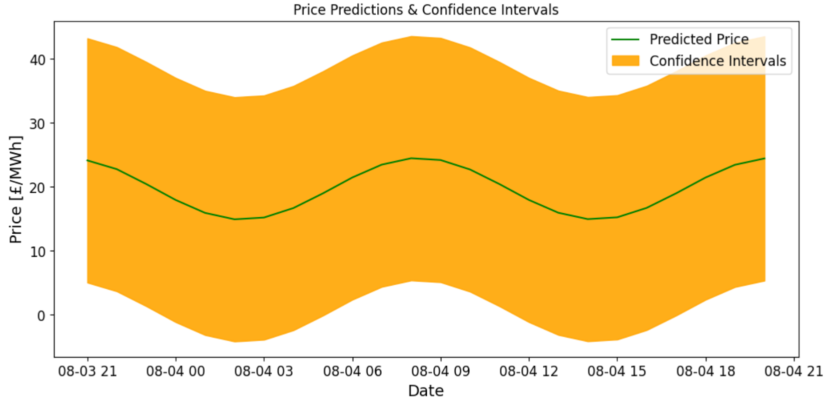
→ Order parameter need to be updated frequently (depending on current economical and political situation)

ARIMA forecast (different days) – order (2, 1, 4)









RNN

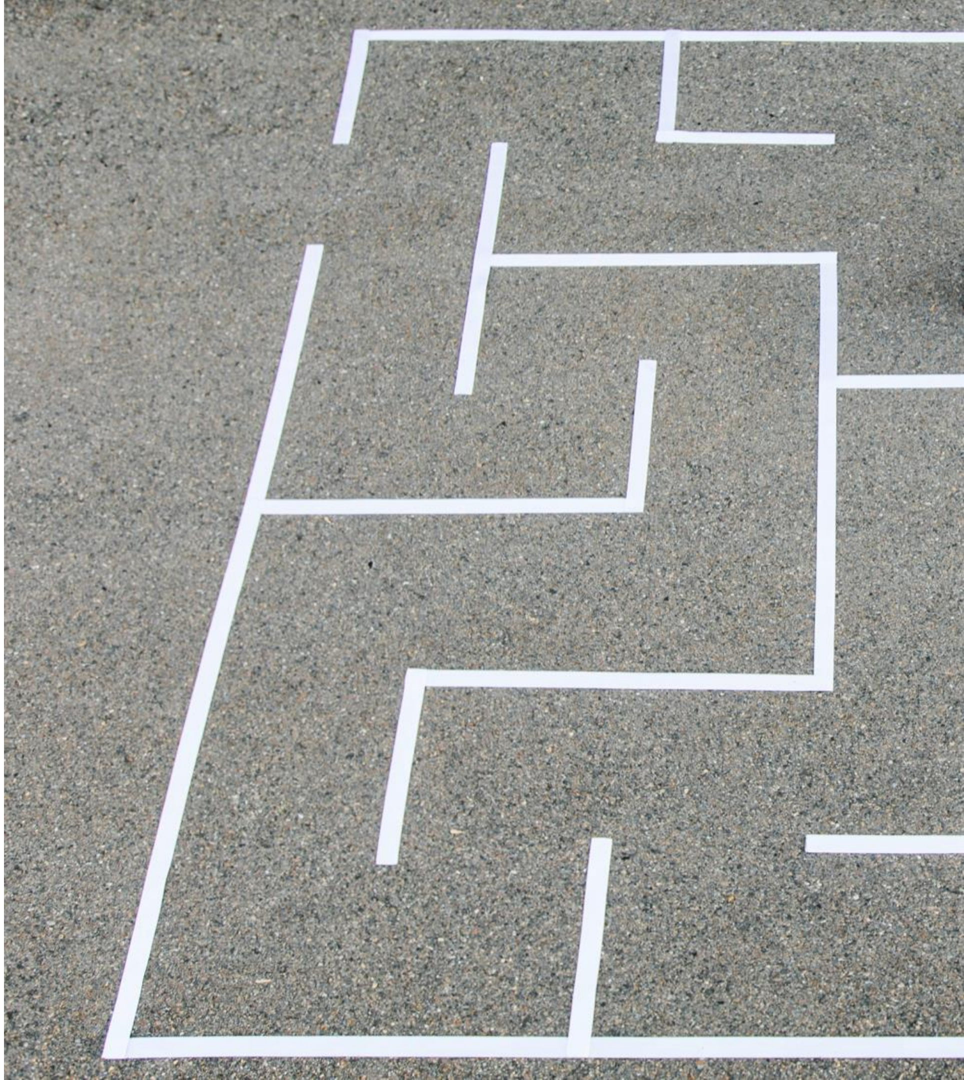
- Type of artificial neural network designed for processing sequential data and capturing patterns over time
- Connections that form directed cycles, allowing them to maintain a memory of previous inputs

Problems with RNNs

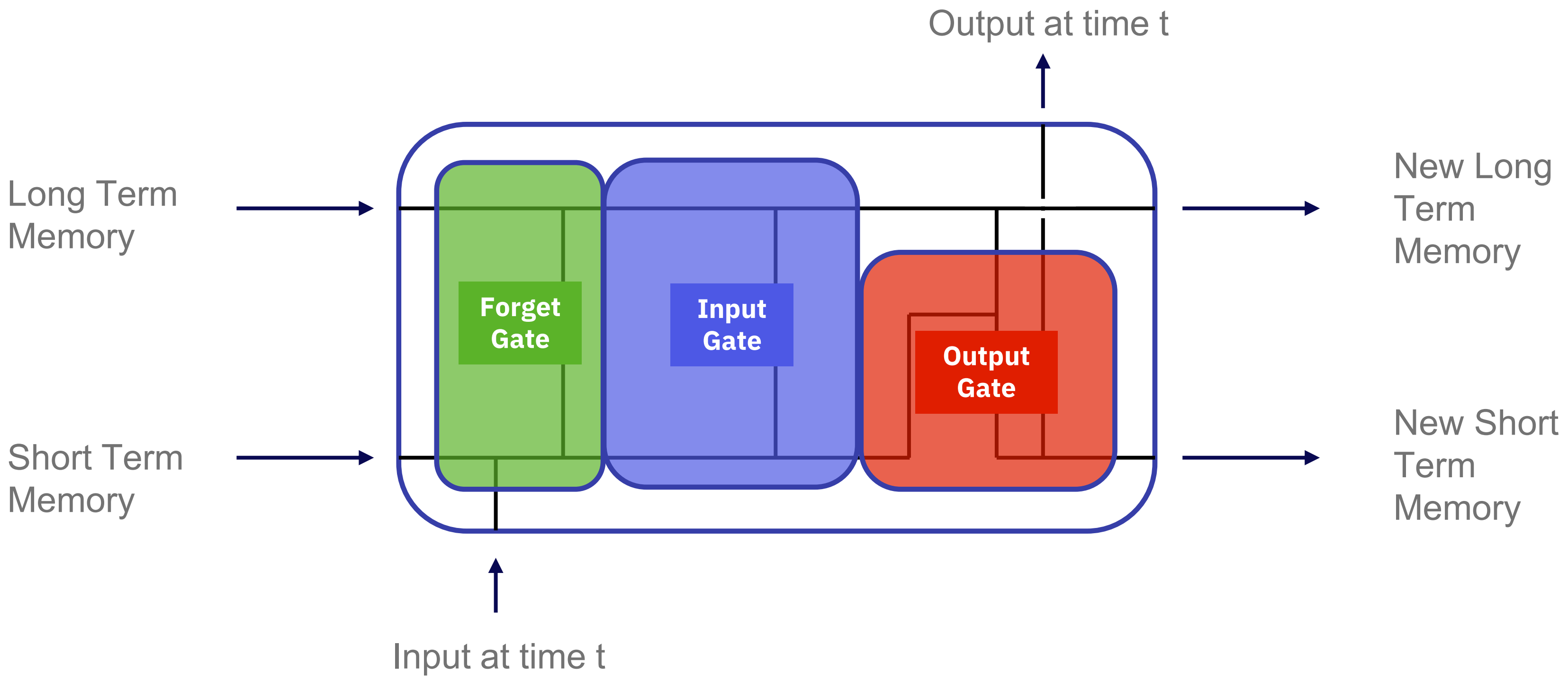
- Because of the length of the time series input, these could slow down training
- Possible solution: shorten the time steps used for prediction → larger trends difficult to predict
- Also RNNs will begin to “forget” the first inputs → “long-term memory” necessary
- **LSTM**s created to address these typical RNN issues

Long Short-Term Memory Model

- Designed to handle the challenges of learning long-term dependencies in sequential data
- Composed of:
  - Memory cell: can store information over long periods
  - Cell state update: allows the LSTM to decide what information to keep, forget, or update
  - Hidden state output: output of the LSTM and is influenced by the cell state



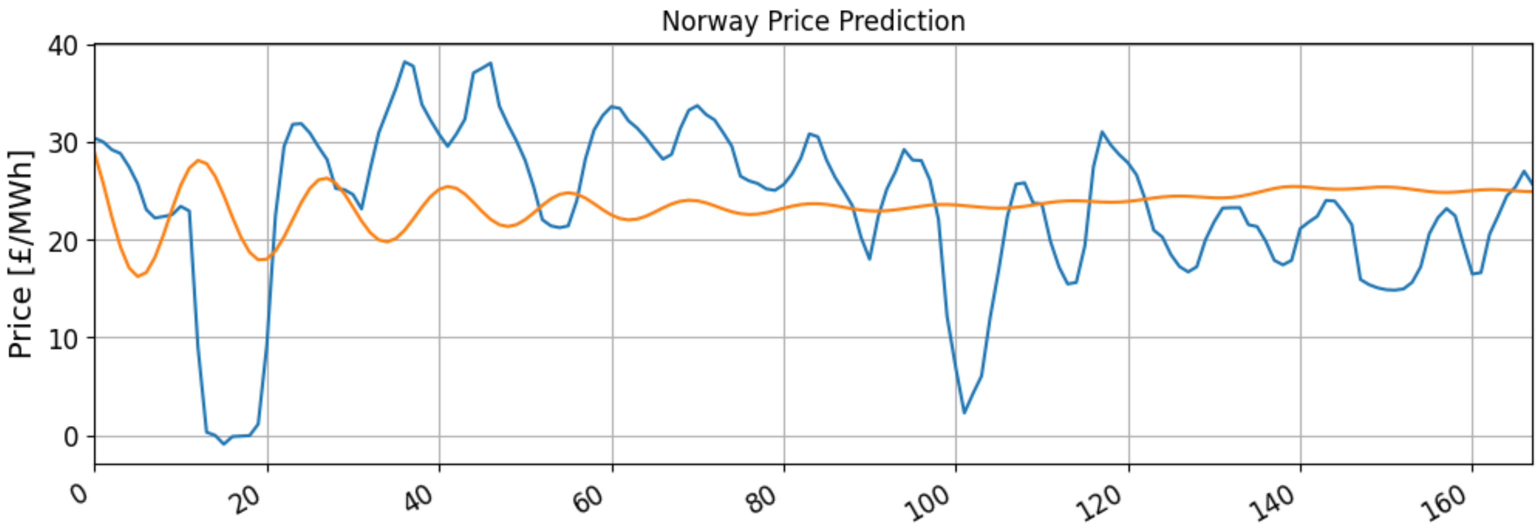
General structure of the memory cell



Explanation of model hyper parameters and training values

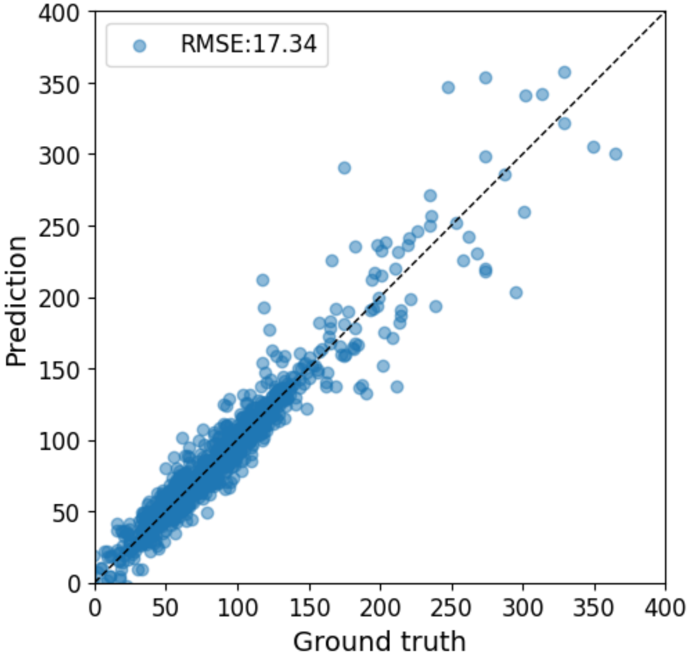
- **Hidden size**: similar to the memory or capacity of the LSTM. More hidden units allow the model to remember more complex patterns, but it also makes the model more computationally intensive.
- **Epoch**: a complete pass through the entire training dataset during the model training process.
- **Learning rate**: size of steps while trying to find the best solution. Too large, and one might overshoot; too small, and it might take forever.
- **Momentum**: similar to the persistence in moving in the same direction. It helps to avoid getting stuck in local minima to find global minima.
- Random **training data tuples** ((24,1) or (168, 24)): represent the input-output pairs for training LSTM models. In the training process, the LSTM model is fed these input sequences (X) during training and is expected to learn the patterns and dependencies in the data to predict the corresponding output sequences (y).

v1 (100 hidden size, 10 epochs, 1000 random training data tuples (24,1), learning rate 0.001)



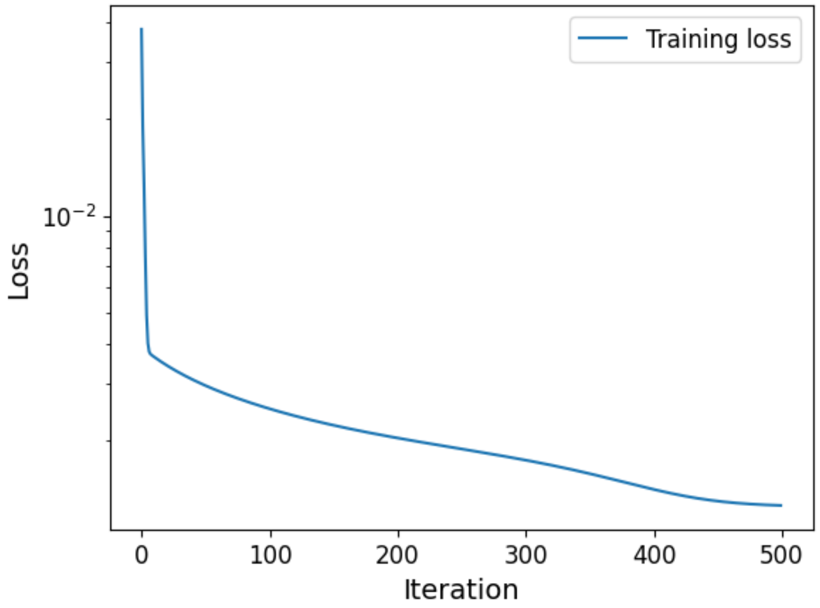
→ Too less epochs the model is trained on

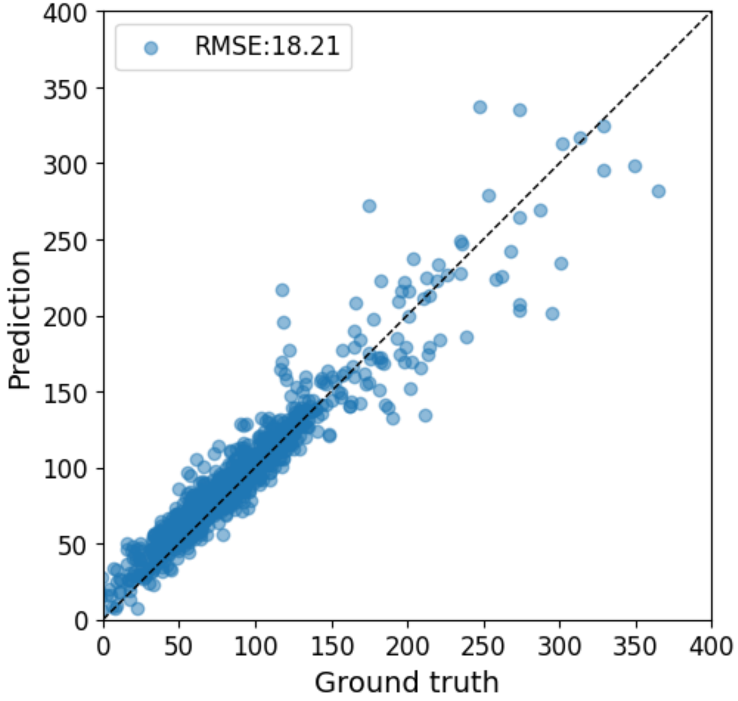
v2 (500 hidden size, 100 epochs, 1000 random training data tuples (24,1), learning rate 0.001, momentum 0.9)



→ Maybe too high hidden size that the model tends to overfit

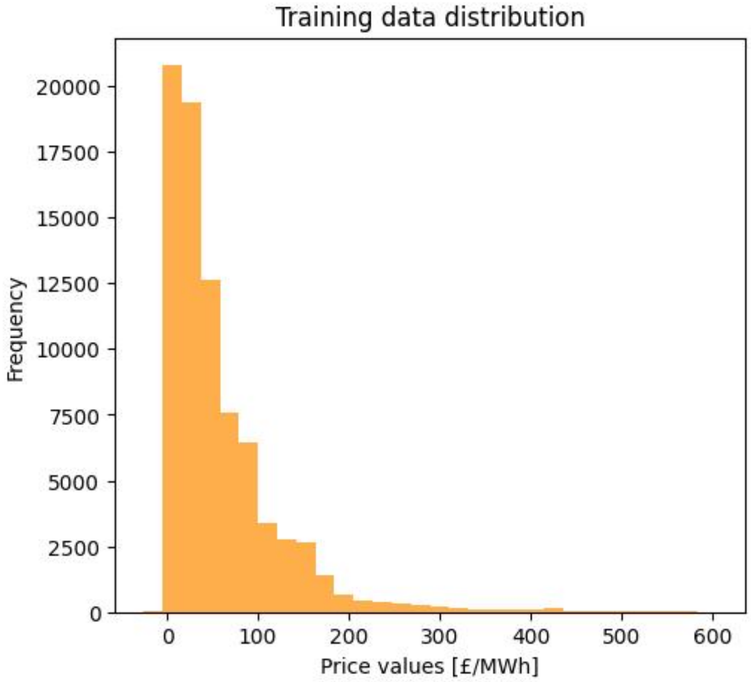
v3 (500 hidden size, 500 epochs, 1000 random training data tuples (24,1), learning rate 0.001, momentum 0.9)

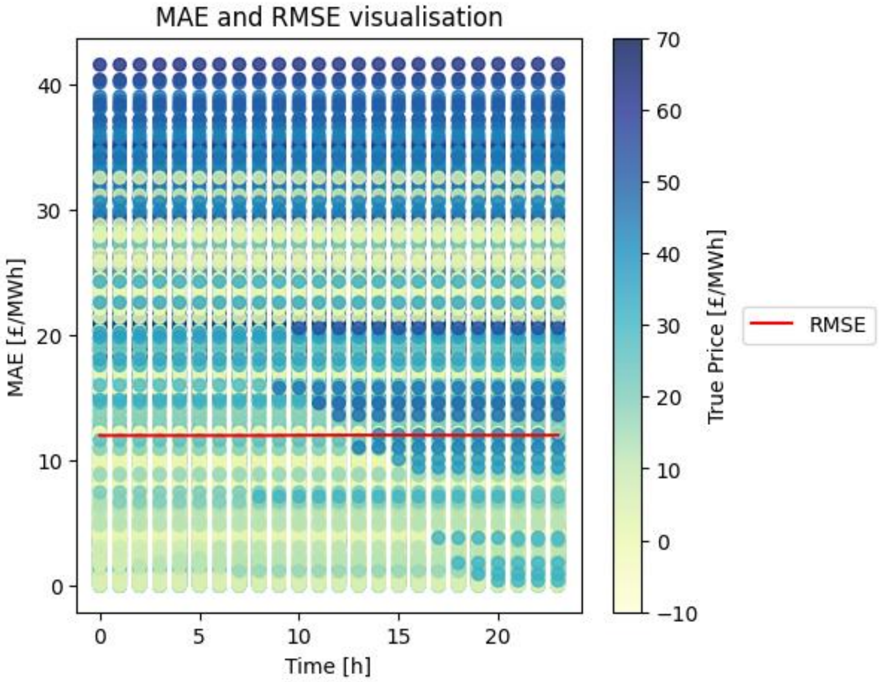




→ Too less training data tuples and too small prediction window

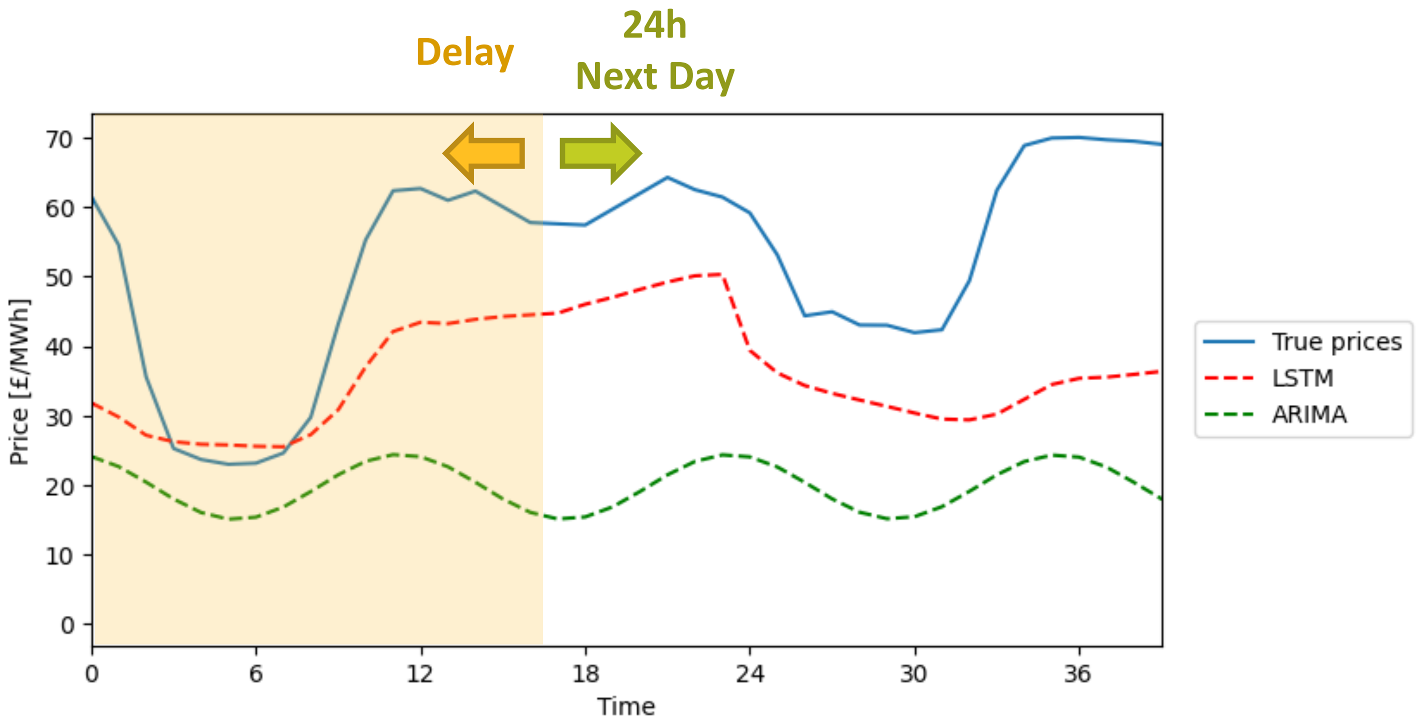
Nearly constant RMSE (12£/MWh) across first 24h prediction (v6)



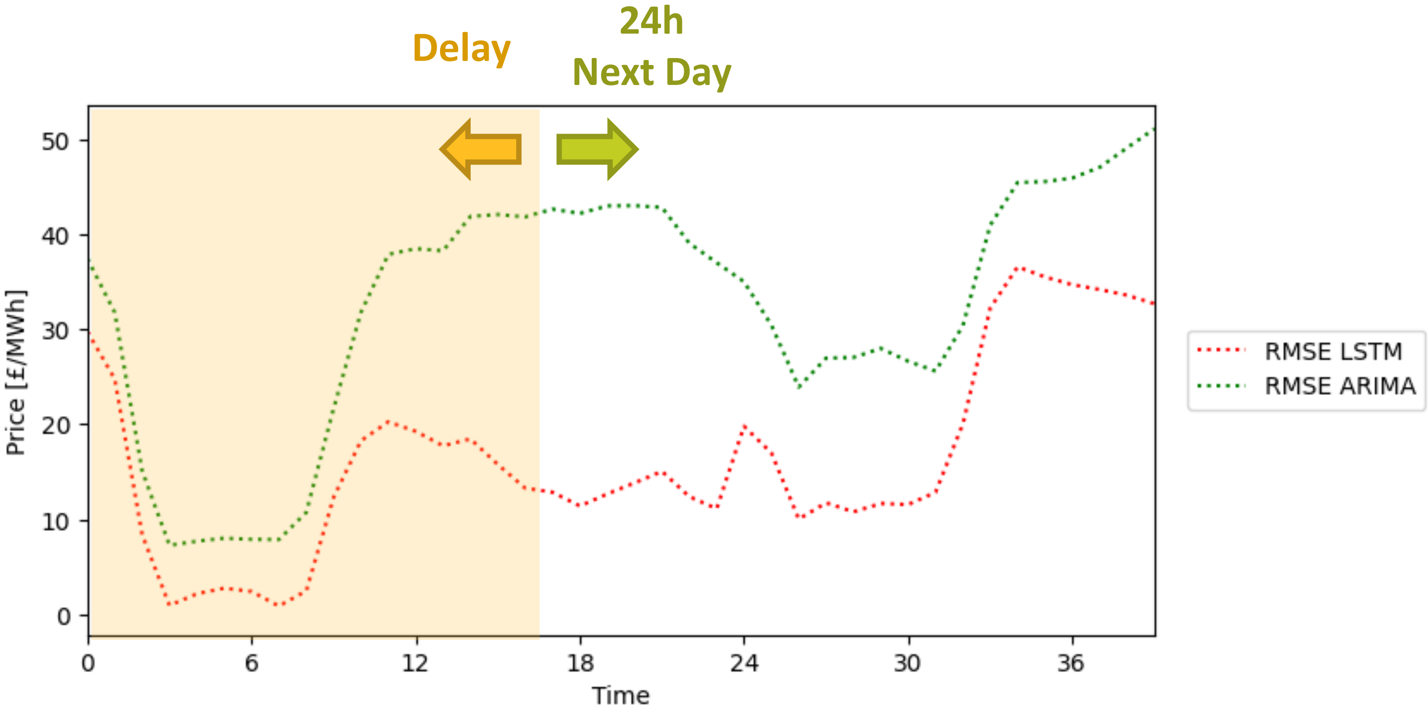


→ Increased training and prediction window (168, 24) with 2000 training data tuples

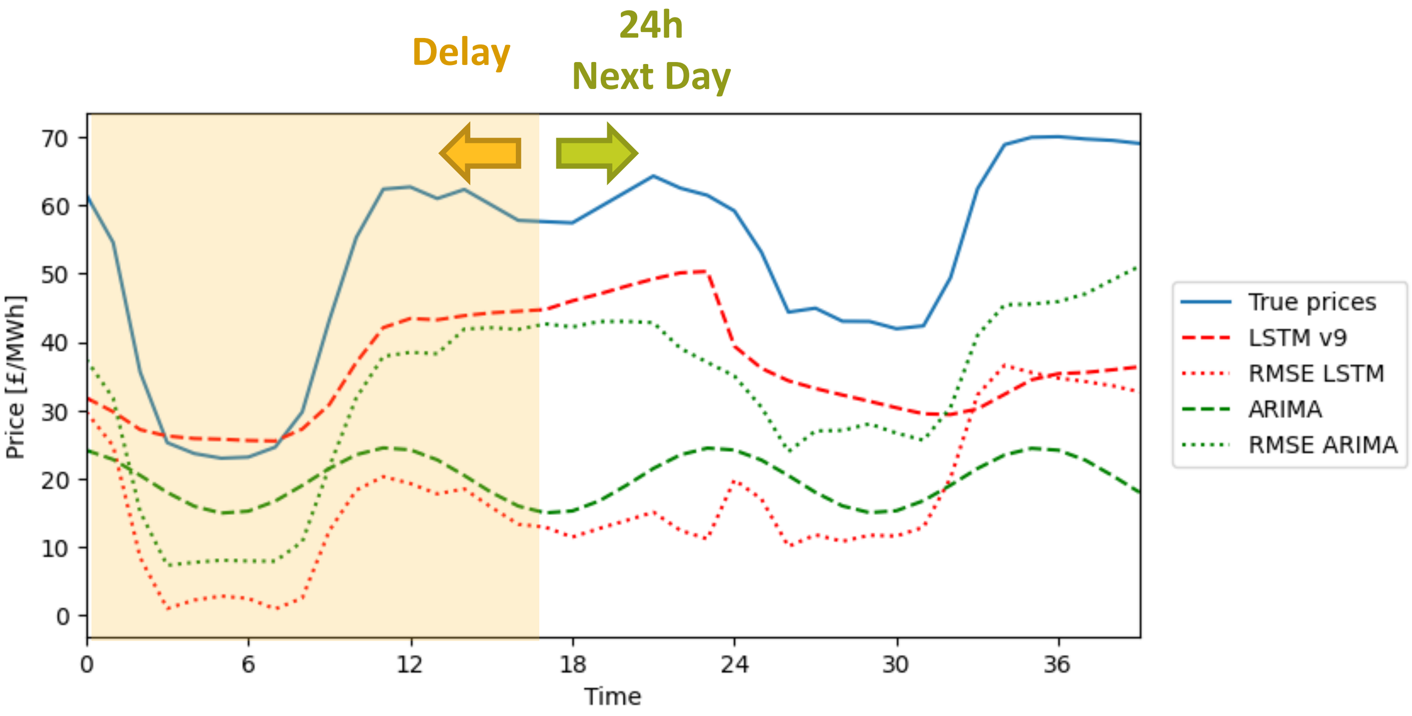
Model comparison ARIMA vs LSTM



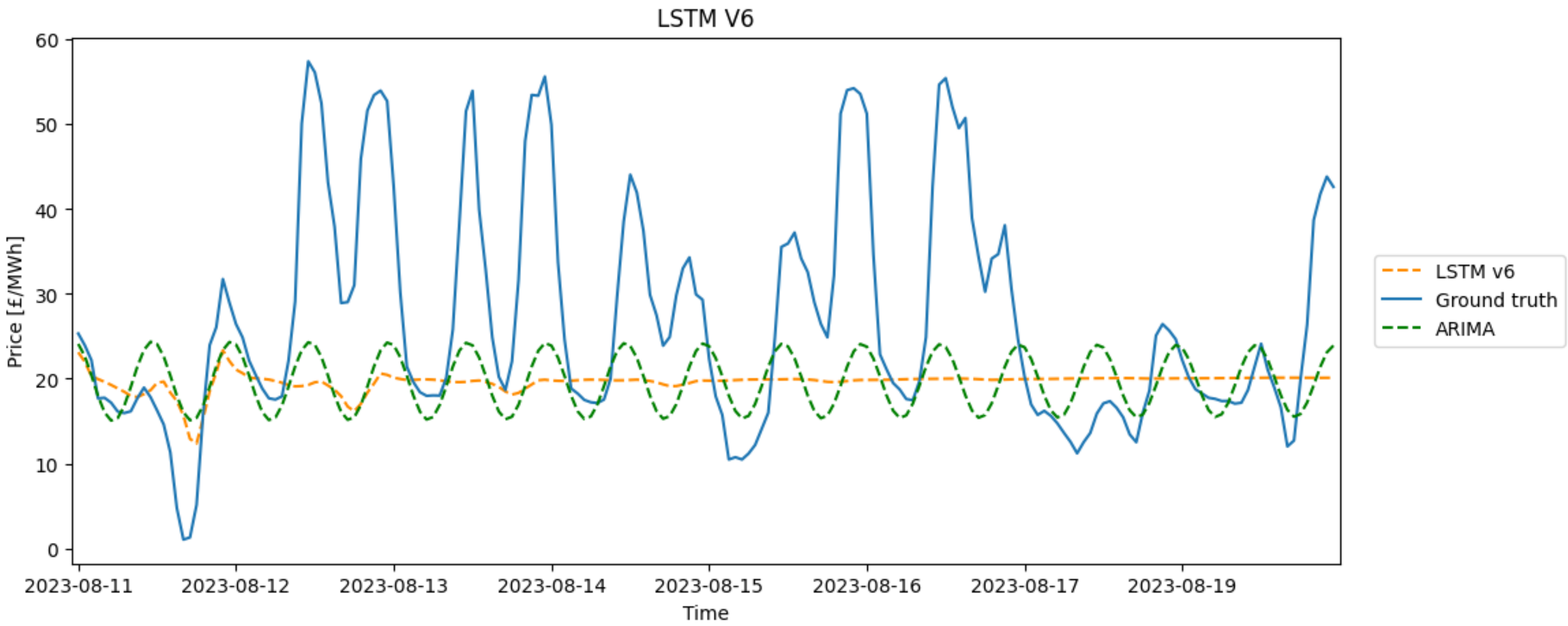
Model comparison ARIMA vs LSTM - RMSE



Model comparison ARIMA vs LSTM (1)



Model comparison ARIMA vs LSTM (2)



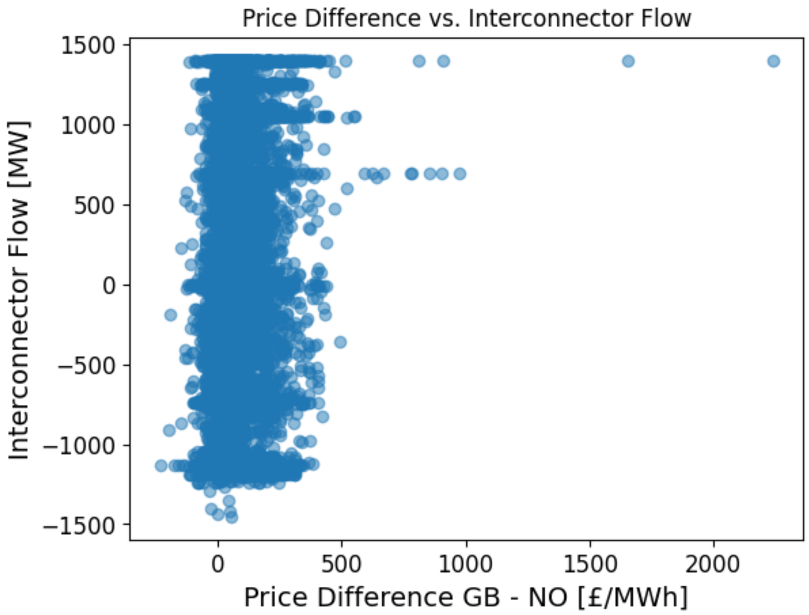
Prediction procedure (general)

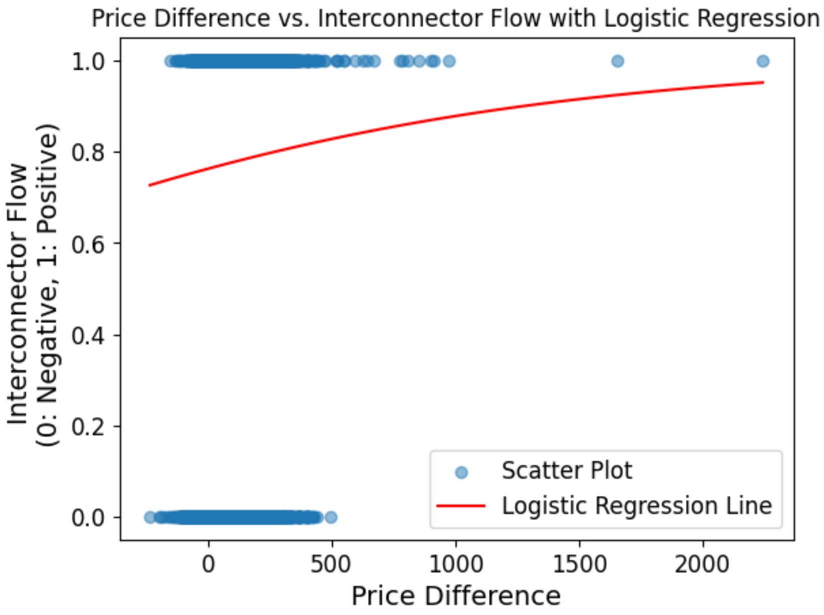
Two main scripts on Azure AI: model training and price prediction

1. Import new data available (no missing values)
2. Script transforms the data, saves the new data table and loads forecasting model
3. LSTM model predicts next 192 hours (8 days)
4. Prediction is stored as new csv table

→ Code review session from 1-3pm today!

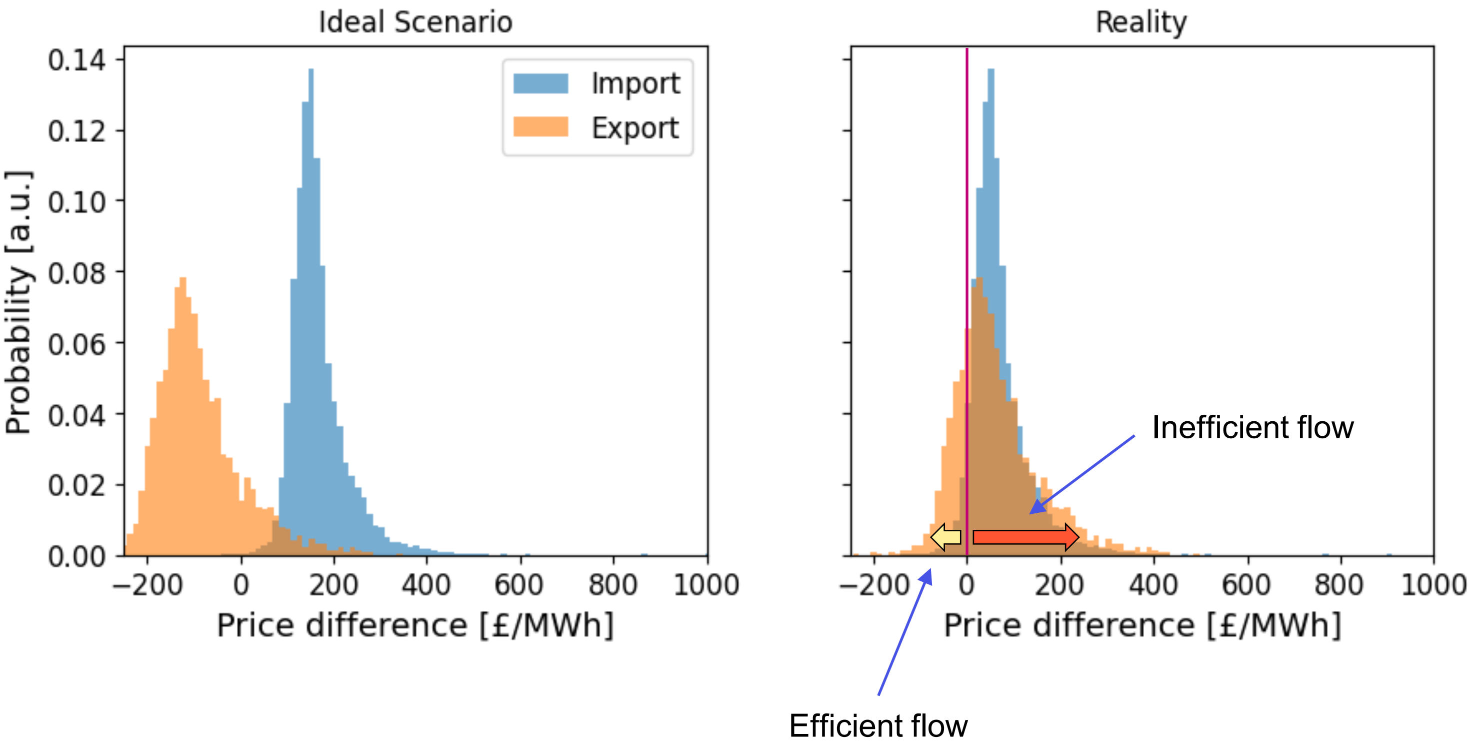
Interconnector flow analysis





→ column ”NSL_uw”: which flow would have been if only price would have been taken into account

Probability distribution analysis



Suggestions

- Continuous monitoring and evaluation of the model's performance → Periodic updates by retraining the model
- Exploring ensemble methods → Investigating the impact of external factors such as weather patterns, geopolitical events, or regulatory changes
- Collaborating with domain experts and stakeholders
- Exploring optimisations or parallelisation techniques to contribute to faster model training and more responsive forecasting

Thank you

If you're interested in finding out more about how Faculty can help you transition to AI independence, get in touch.

lilly.reppin@faculty.ai
160 Old Street, London, EC1V 9BW, UK

faculty