Final Project:

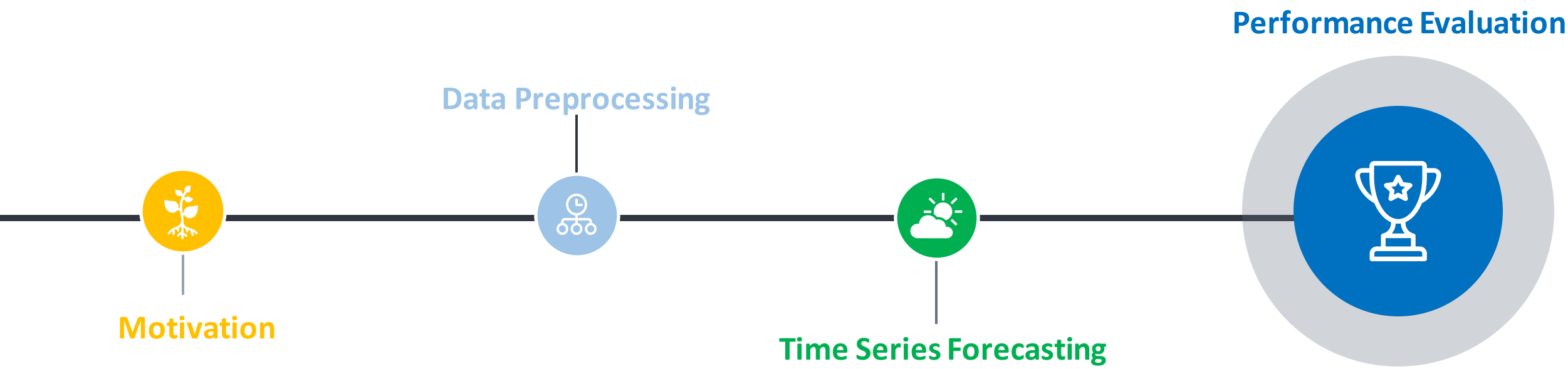
Weather Predictions in Denmark
- Multivariate Timeseries Forcasting

Alice, Erlend, Jonas, Tonje

“It is very difficult to predict - especially the future.”

-Niels Bohr

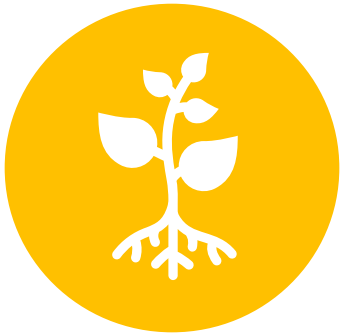
Motivation


Performance Evaluation
Data Preprocessing
Motivation
Time Series Forecasting

Motivation

Goal:

- Predict ***Feels like* Temperature**
  - Measure of how the weather feels to the human body



Important for:

- **Scheduling** of **outdoor work** (construction, agriculture…)
- **Heat demand** and **traffic management**

Approach:

- **Explore** and **compare** **<u>time series forecasting</u>** models
- **Identify challenges** and **limitations**

Dataset - Overview

→ Website:

visualcrossing Weather Data

→ Capabilities:


Historical data


Weather forecasts


Documentation

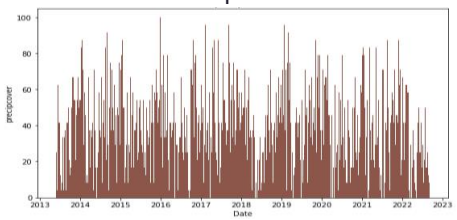
→ Amount of Data:

- ❑ **Hourly** data
- ❑ 1st Jan, **2012** - 31st Dec, **2022**
- ❑ **11** years
- ❑ **96432** events, **20** features

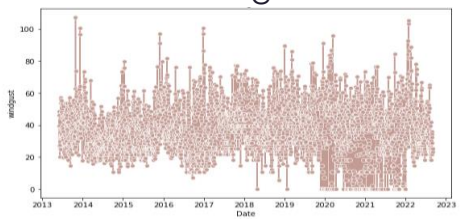
https://www.visualcrossing.com/weather/weather-data-services

Dataset - Features

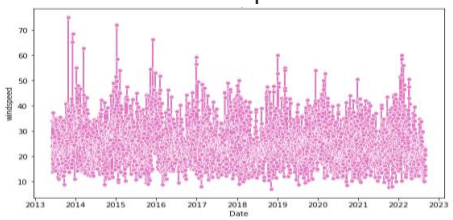
Precip. cover



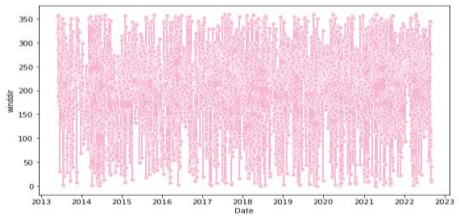
Wind gust



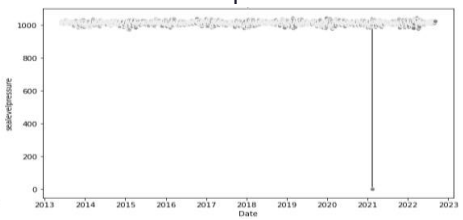
Wind speed



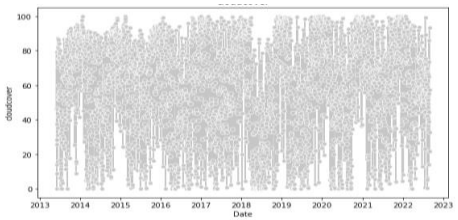
Wind dir.



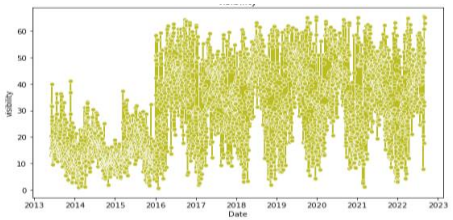
Sea level pressure



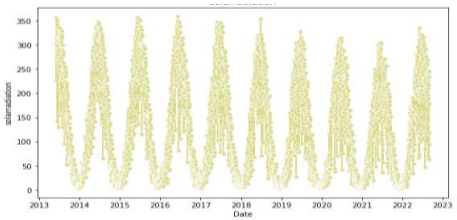
Cloud cover



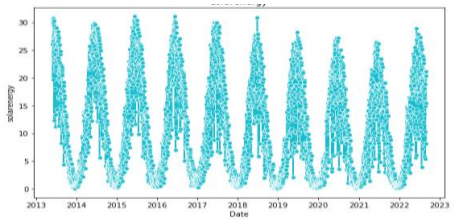
Visibility



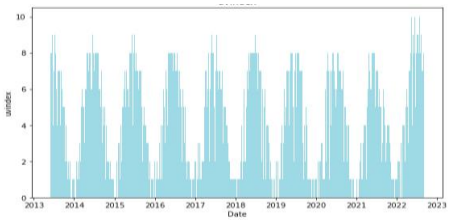
Solar radiation



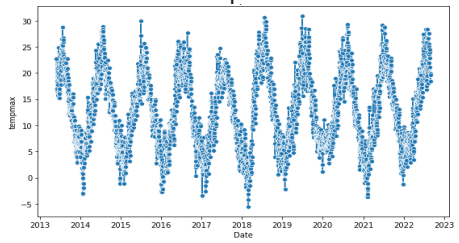
Solar energy



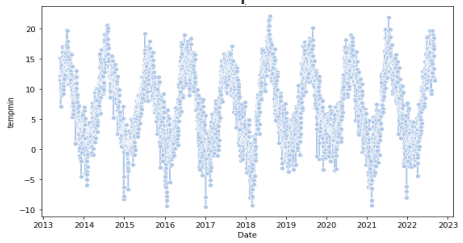
UV index



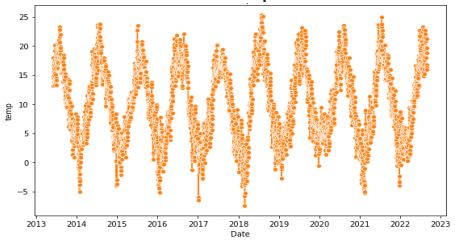
Temp. max



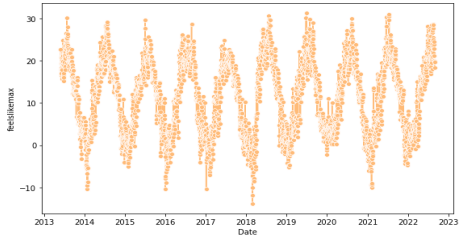
Temp. min



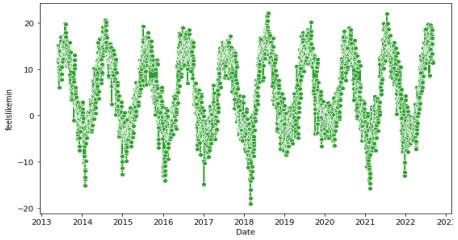
Temp.



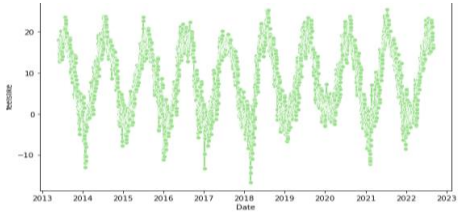
Feels like max



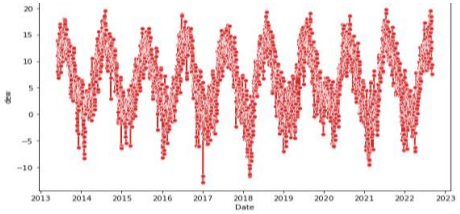
Feels like min



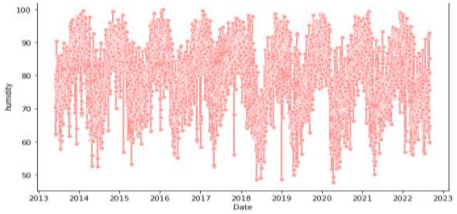
Feels like



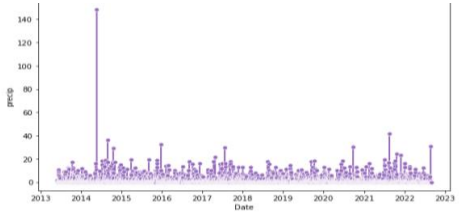
Dew



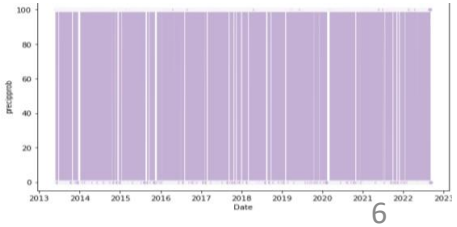
Humidity



Precip.



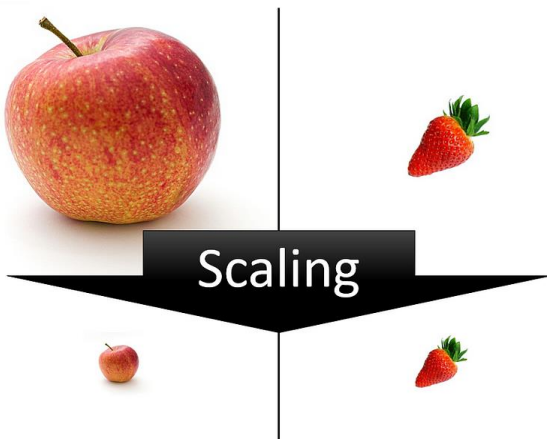
Precip. prob.



2013 to **2022**

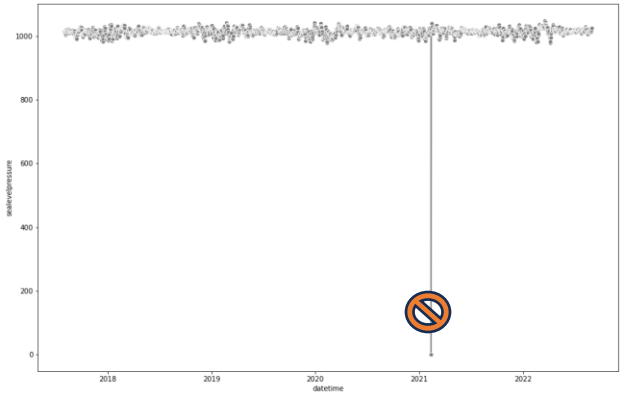
Dataset - Processing

- **Scaling**



- **Removing Outliers**

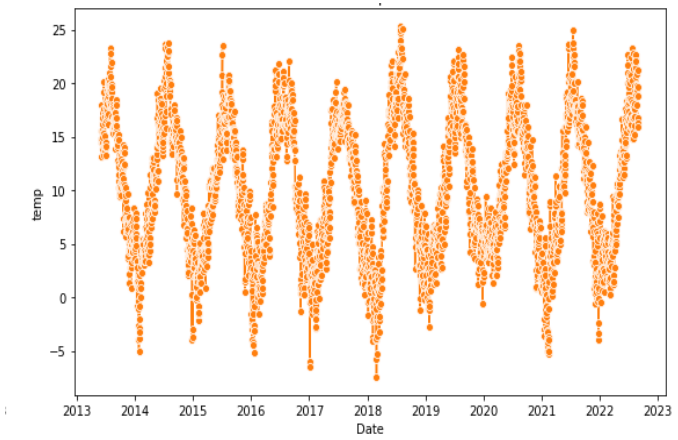
Sea lev el pressure



2013 to **2022**

- **Seasonality**

Temp.

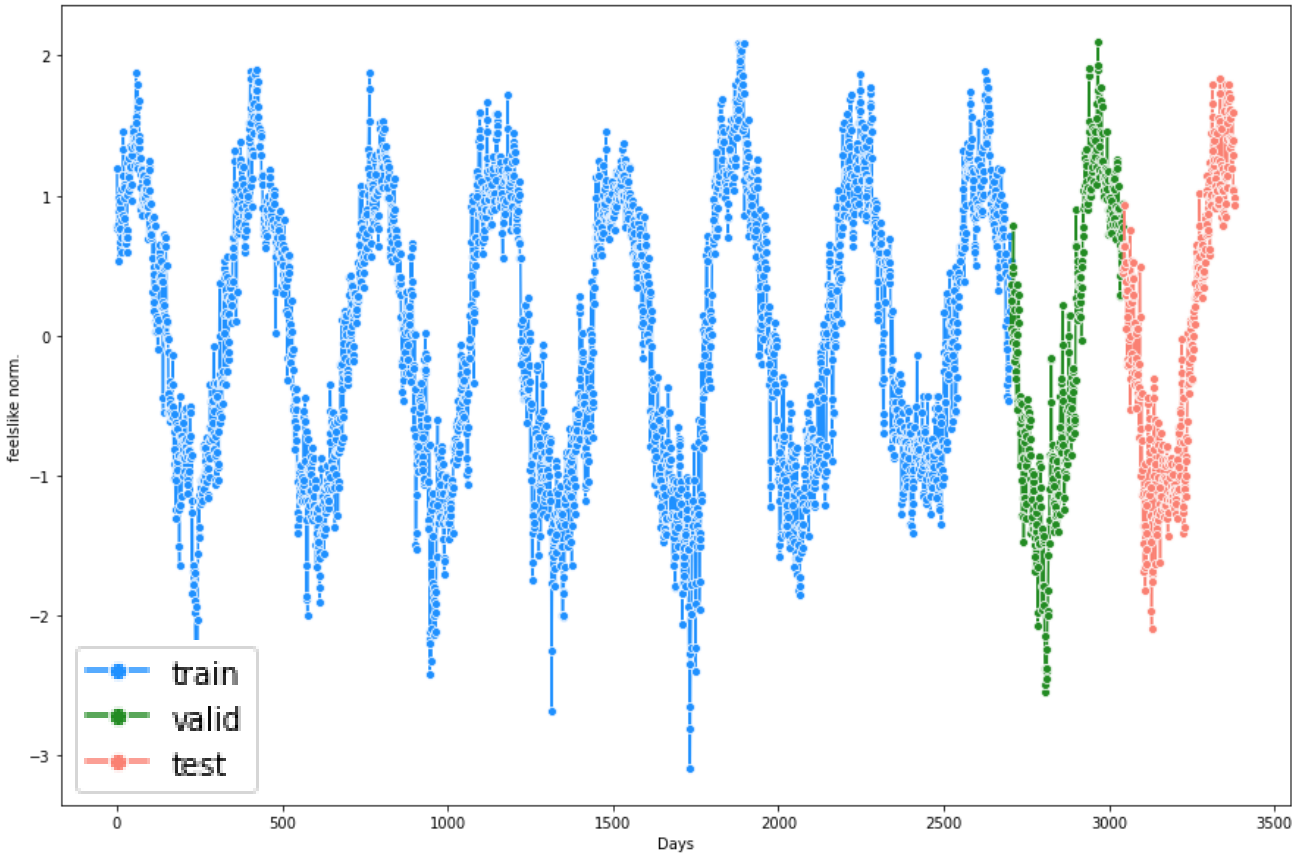


2013 to **2022**

- **<u>XGBoost:</u>**
  - **lag features** from **similar hour** in **day**, **week**, or **year**
- **<u>LSTM/GRU:</u>**
  - **cos/sin wave** with **daily**, **weekly** and **yearly periodicity**

Dataset - Splitting

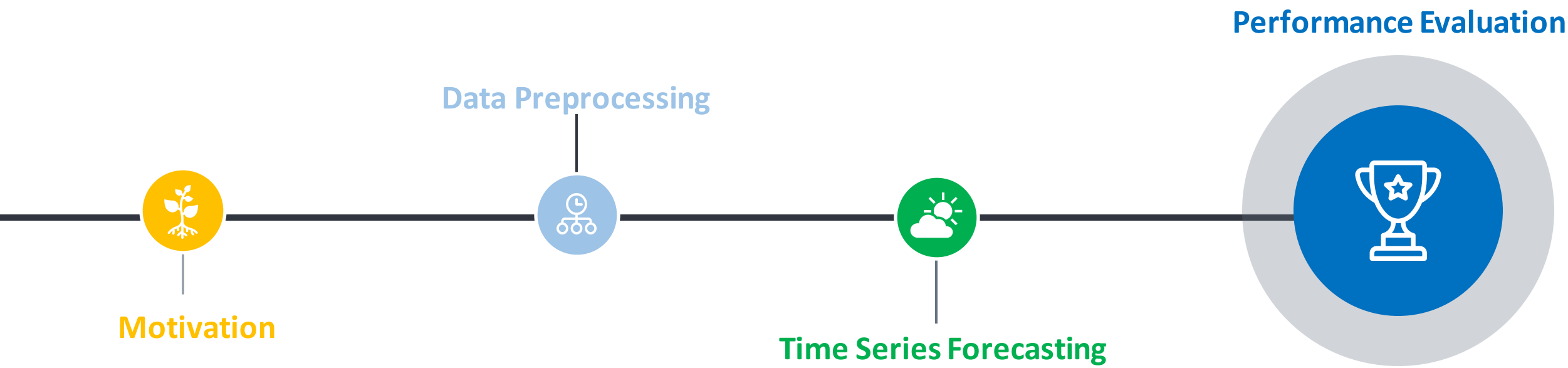
Normalised feels like temp.



2013 to **2022**

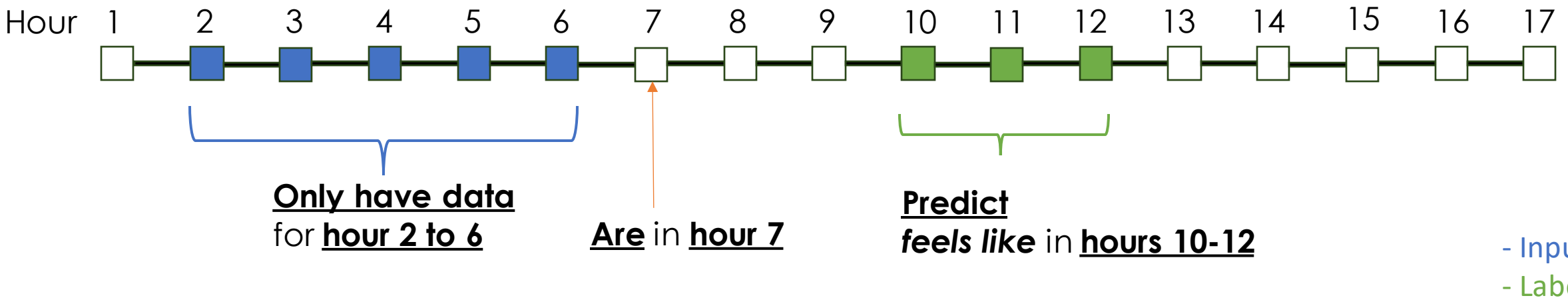
Train-Valid-Test-split
→ without Data leakage

Time Series Forecasting



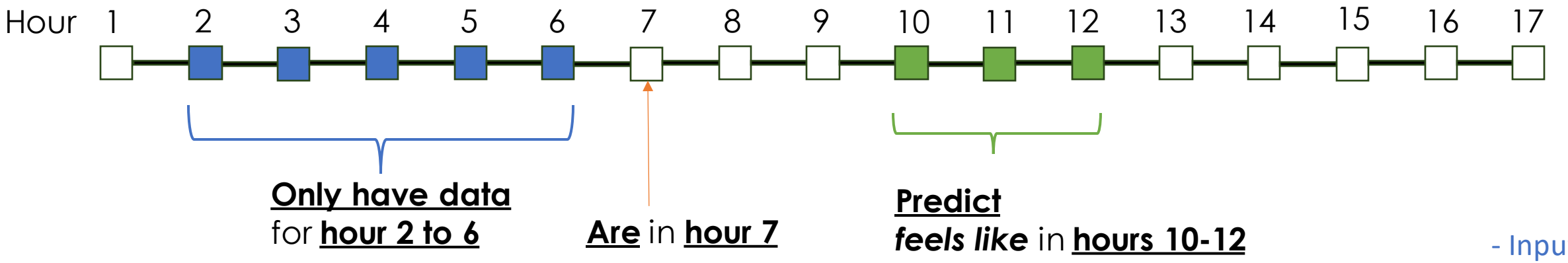
Time Series Forecasting – Example

Example:
Predict *feels like* temp. in **hours 10-12**, given that **we are in hour 7**, and **only have data** for **hour 2-6**



Time Series Forecasting – Example

Example:
Predict *feels like* temp. in **hours 10-12**, given that **we are in hour 7**, and **only have data** for **hour 2-6**



Input, X

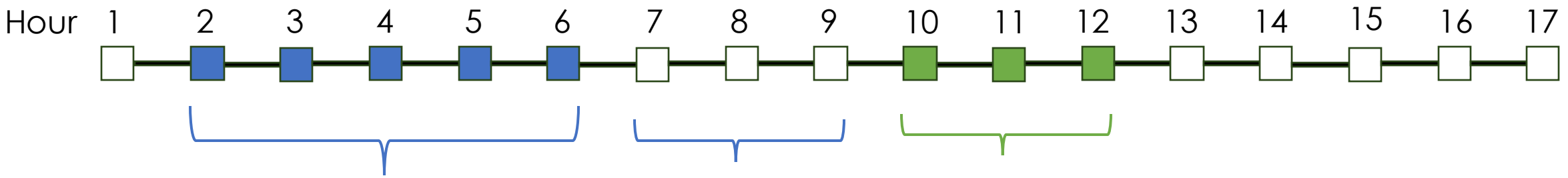
For **hour 2** to **6**, have **20 features**: temperature, humidity…

Output, y

For **hours 10-12**, want **1 output each**: feels like temp.

Time Series Forecasting – Definitions

Definitions:



Window size
Nr. of input time steps

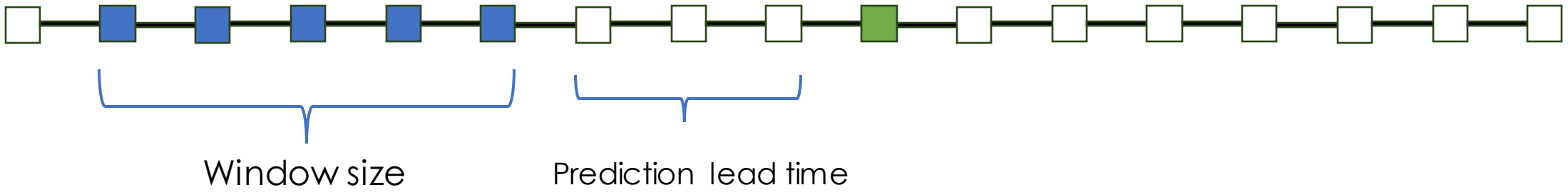
Prediction lead time
Nr. of time steps between in- and output

Prediction horizon
Time steps to predict for

- Input
- Label

Time Series Forecasting

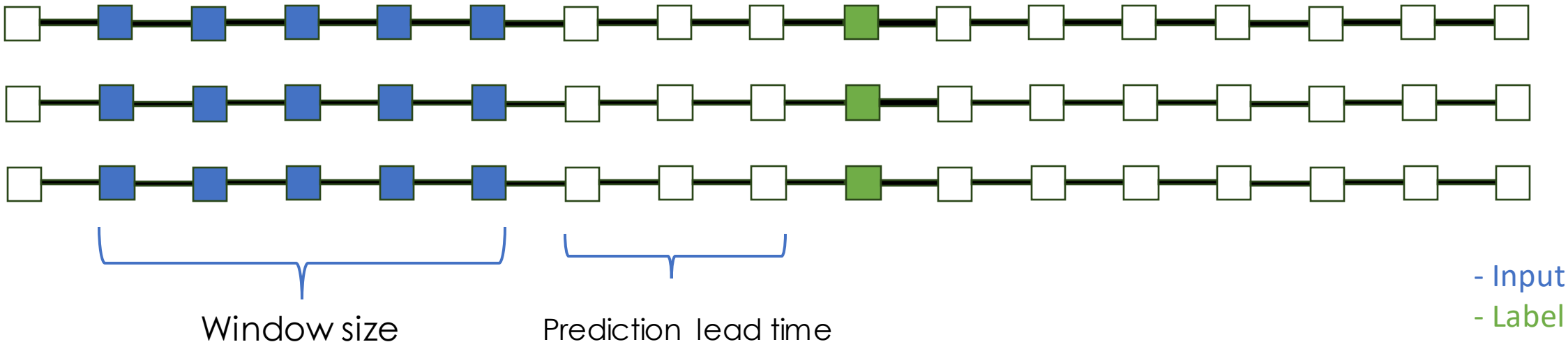
Single variable forecasting:



- Input
- Label

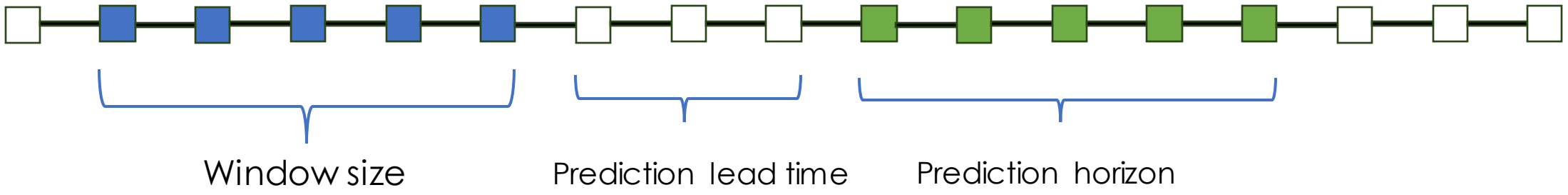
Time Series Forecasting

Multivariate forecasting:



Time Series Forecasting

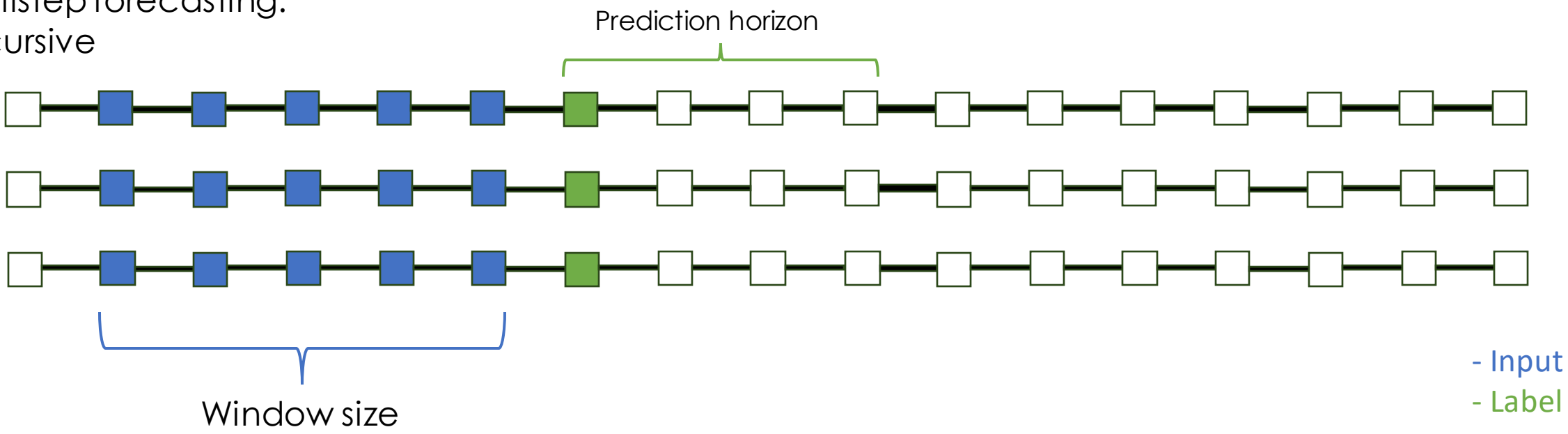
Multistep forecasting:
Naive approach



- Input
- Label

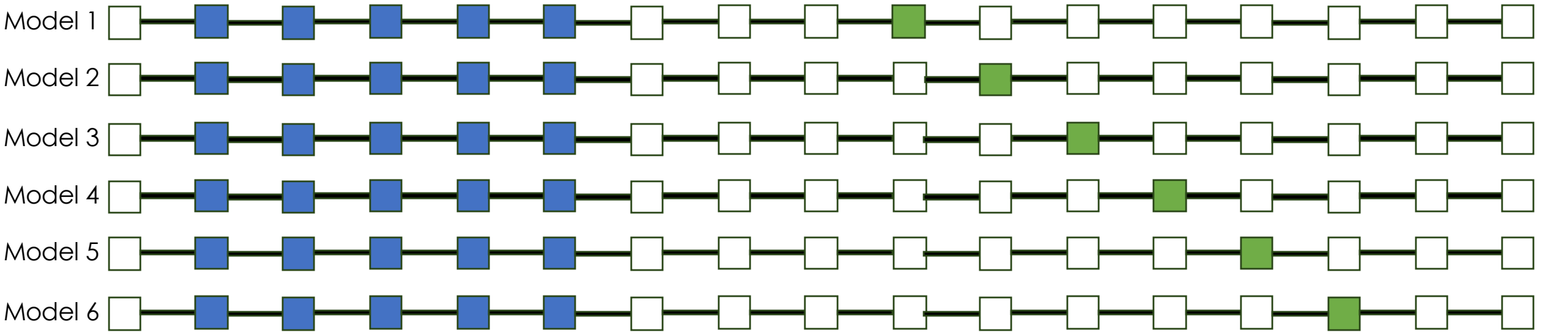
Time Series Forecasting

Multistep forecasting:
Recursive

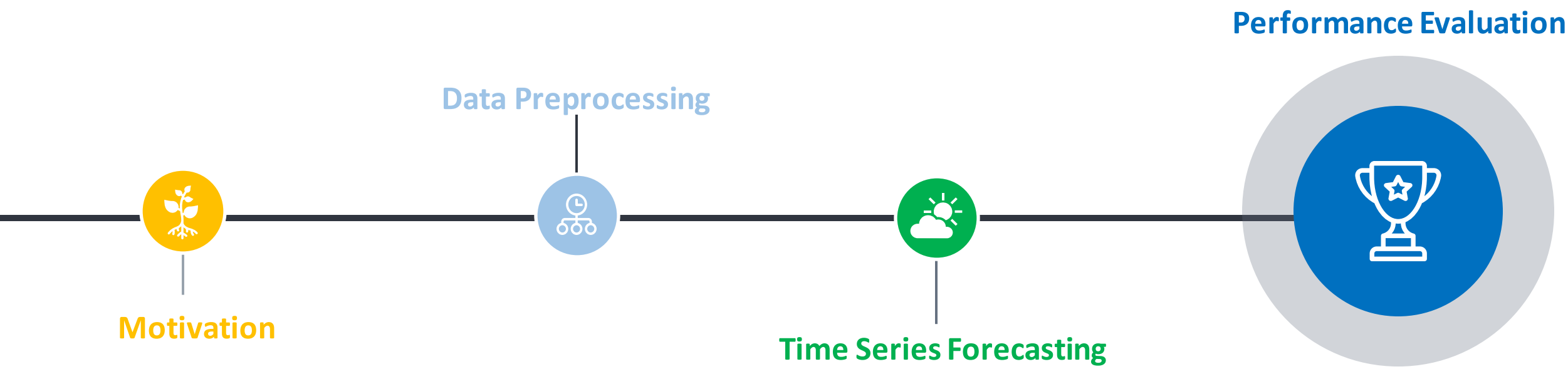


Time Series Forecasting

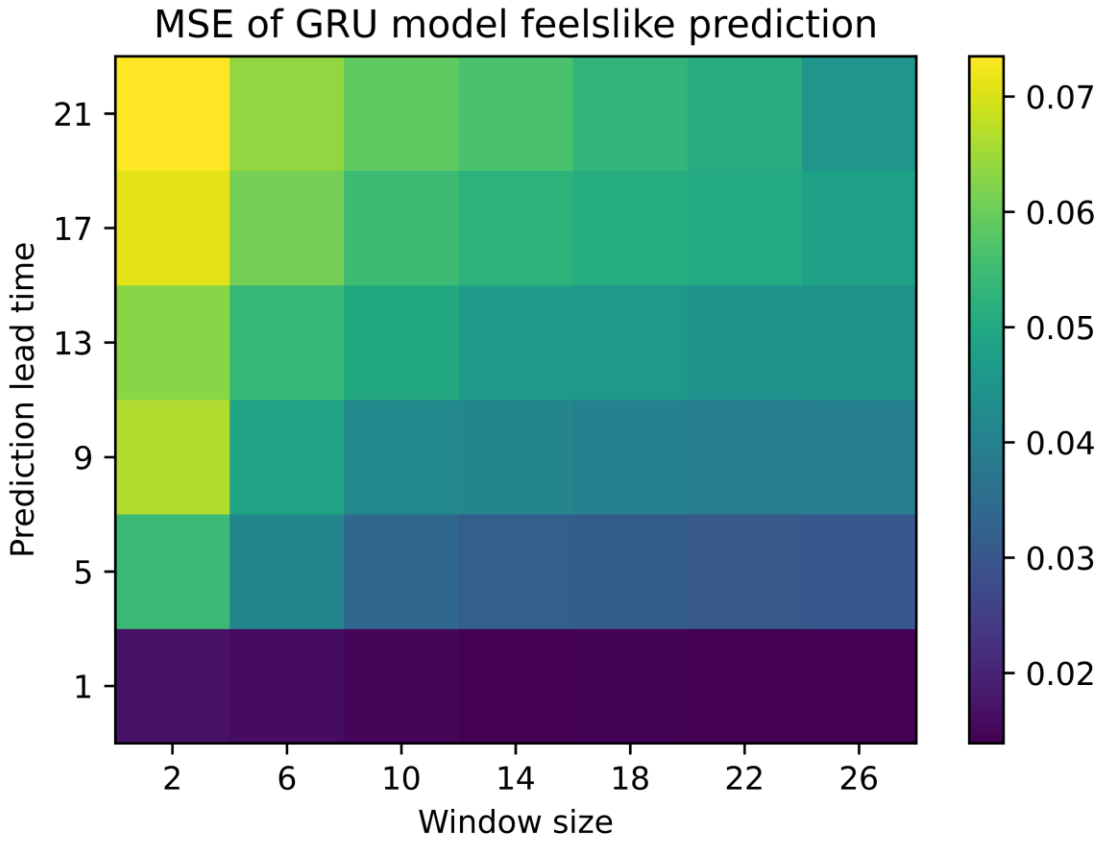
Multistep forecasting:
Ensemble



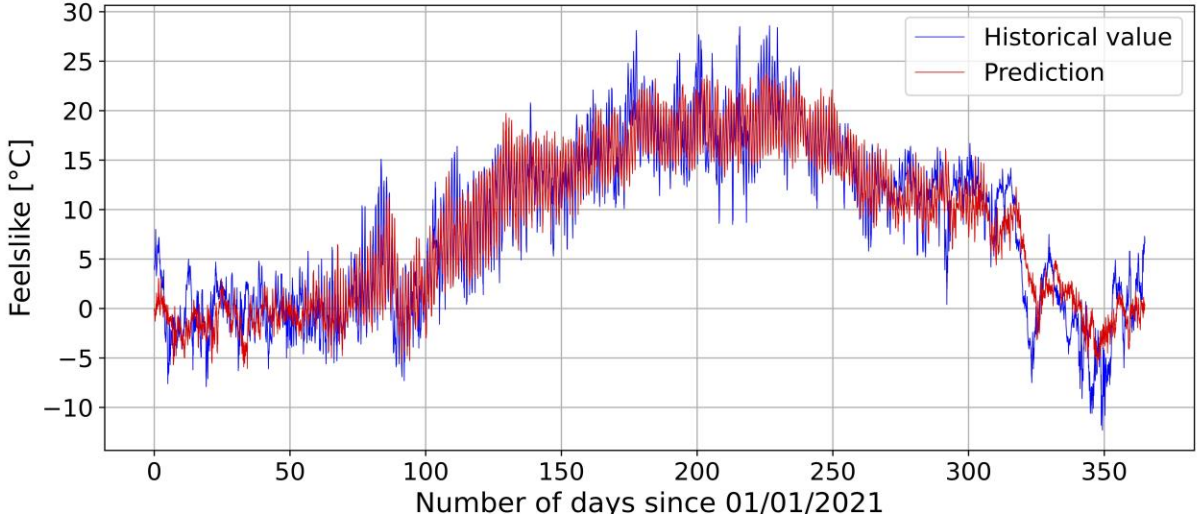
Performance Evaluation & Outlook

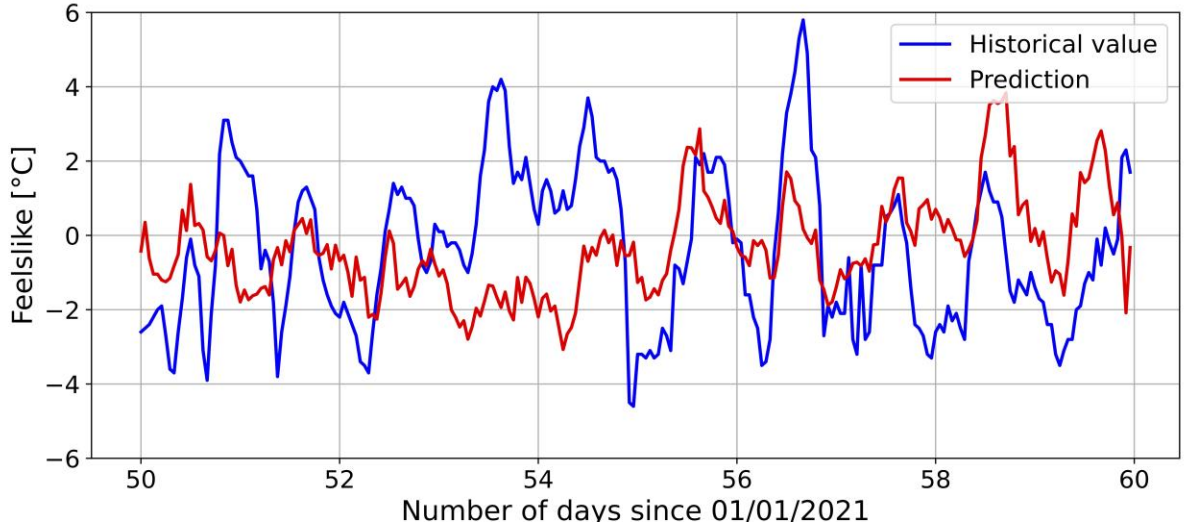


Dependence on Window size and prediction horizon



Performance evaluation: Single-step forecasting



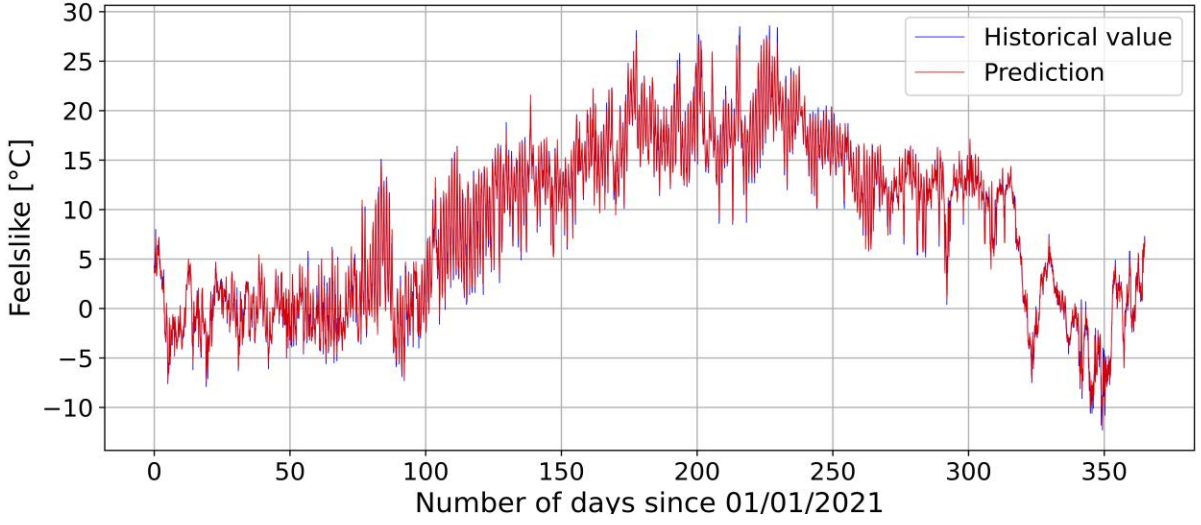


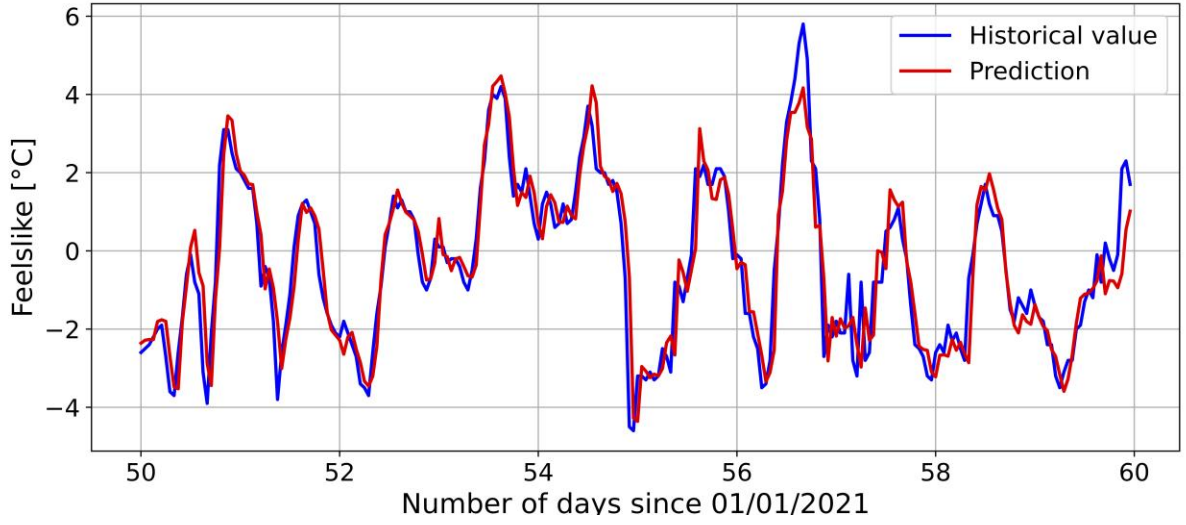
Model
XGBOOST

Prediction lead time
1 hour

RMSE
2.875

Performance evaluation: Single-step forecasting



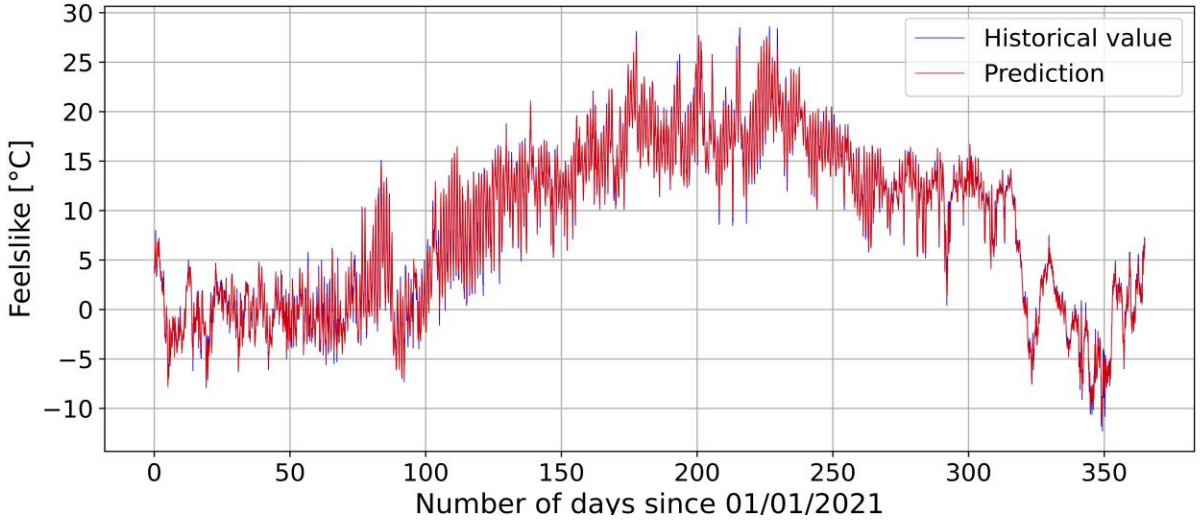


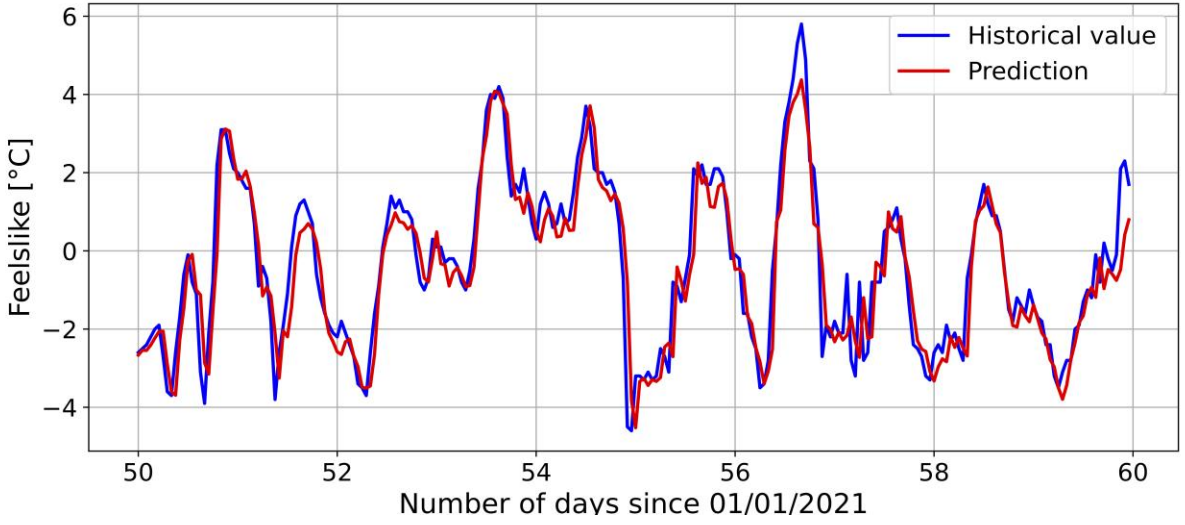
Model
GRU

Prediction lead time
1 hour

RMSE
0.786

Performance evaluation: Single-step forecasting



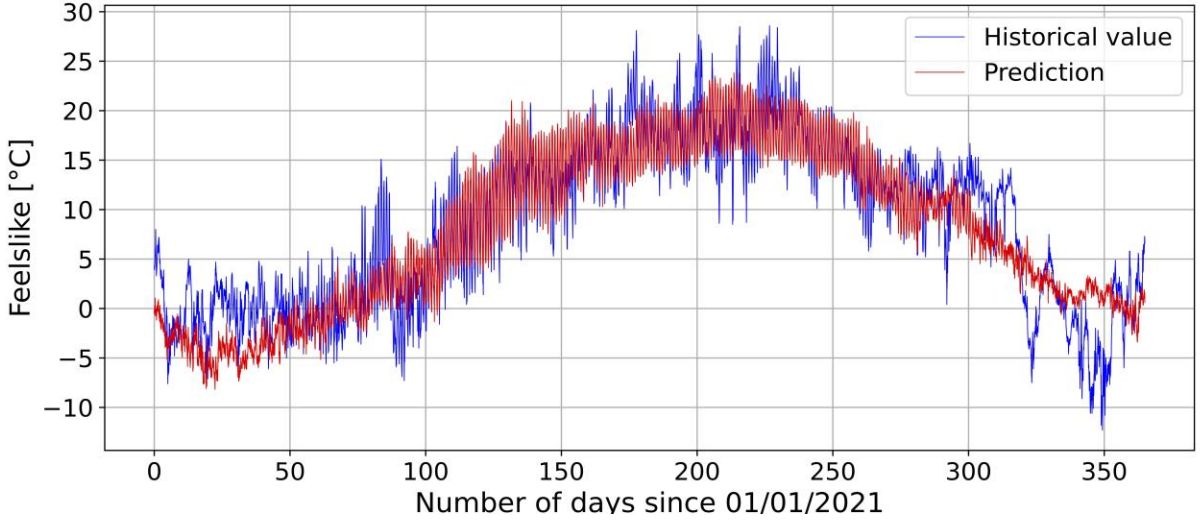


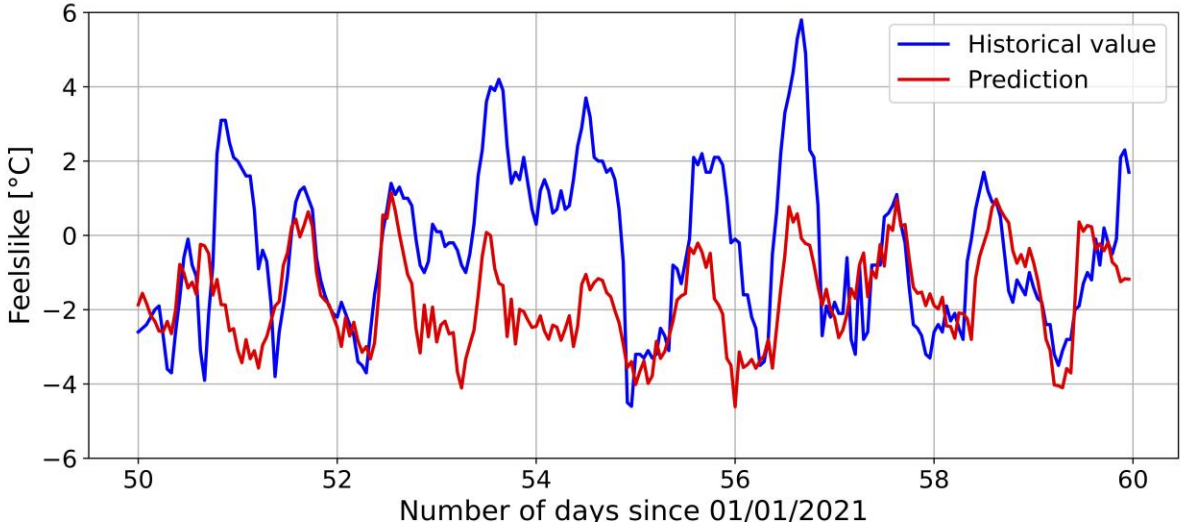
Model
LSTM

Prediction lead time
1 hour

RMSE
0.753

Performance evaluation: Single-step forecasting





Model

XGBOOST

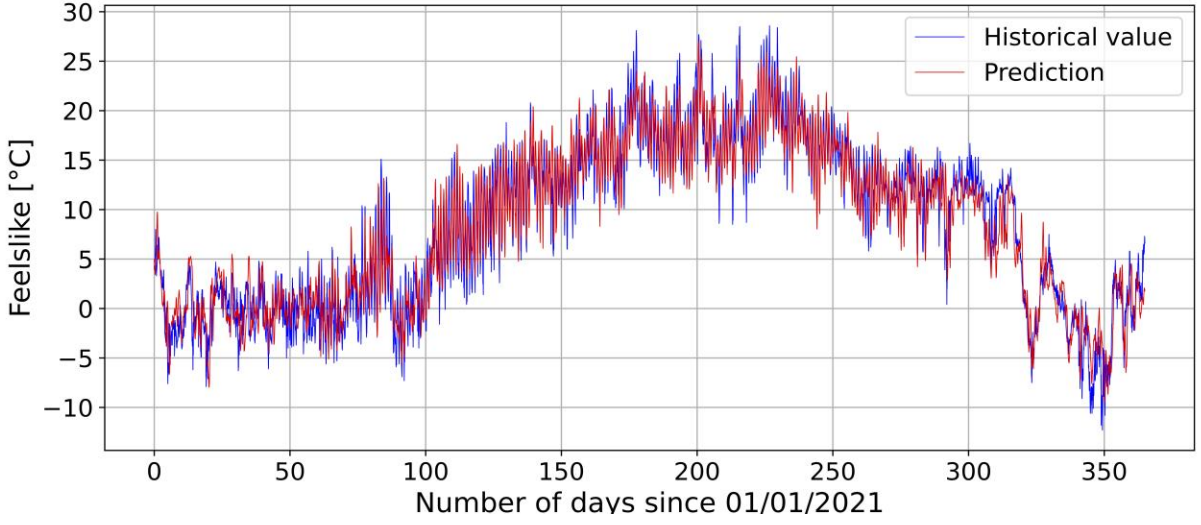
Prediction lead time

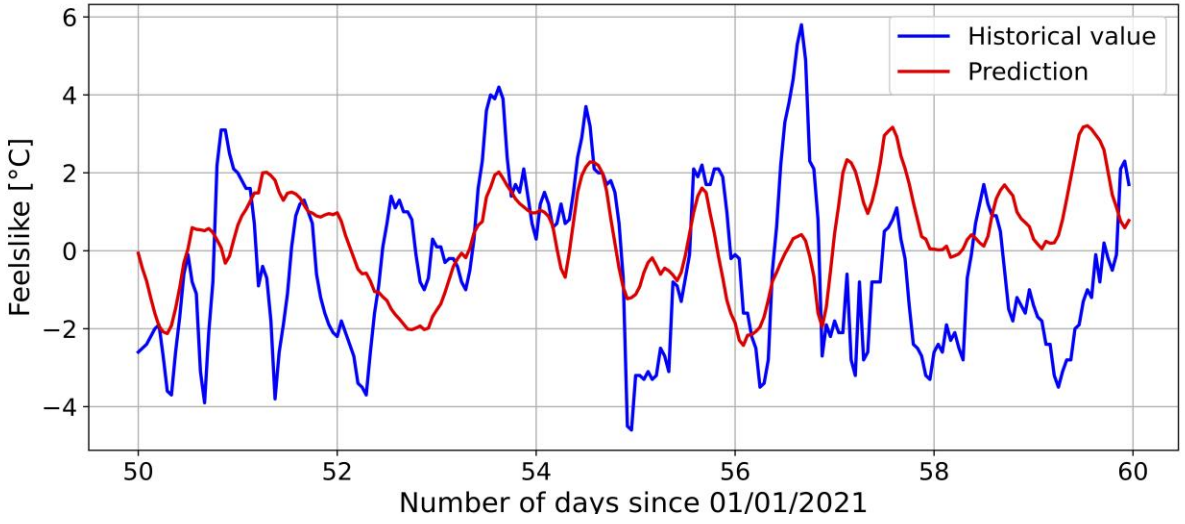
10 hour

RMSE

3.635

Performance evaluation: Single-step forecasting



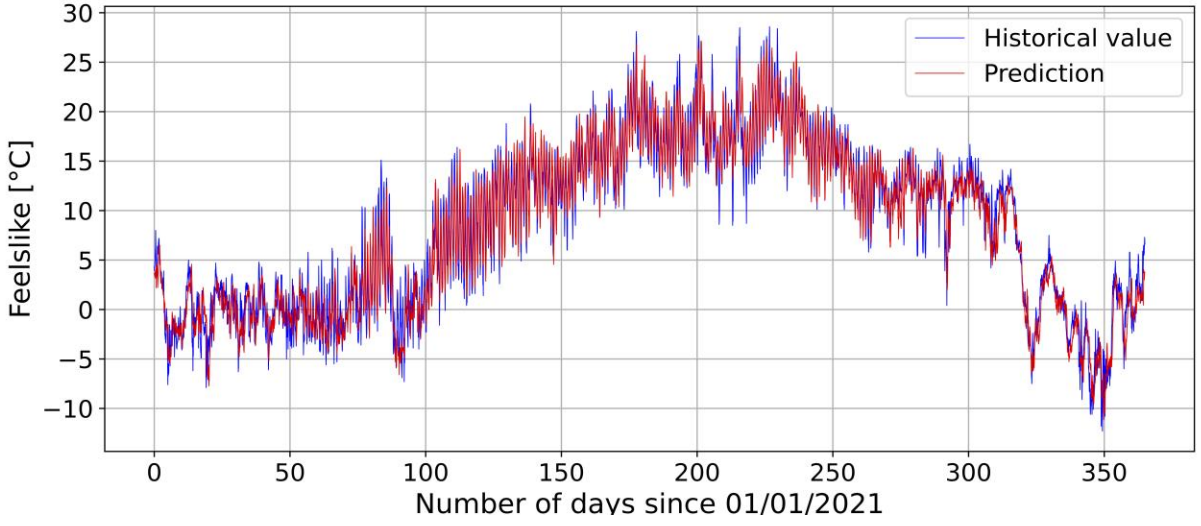


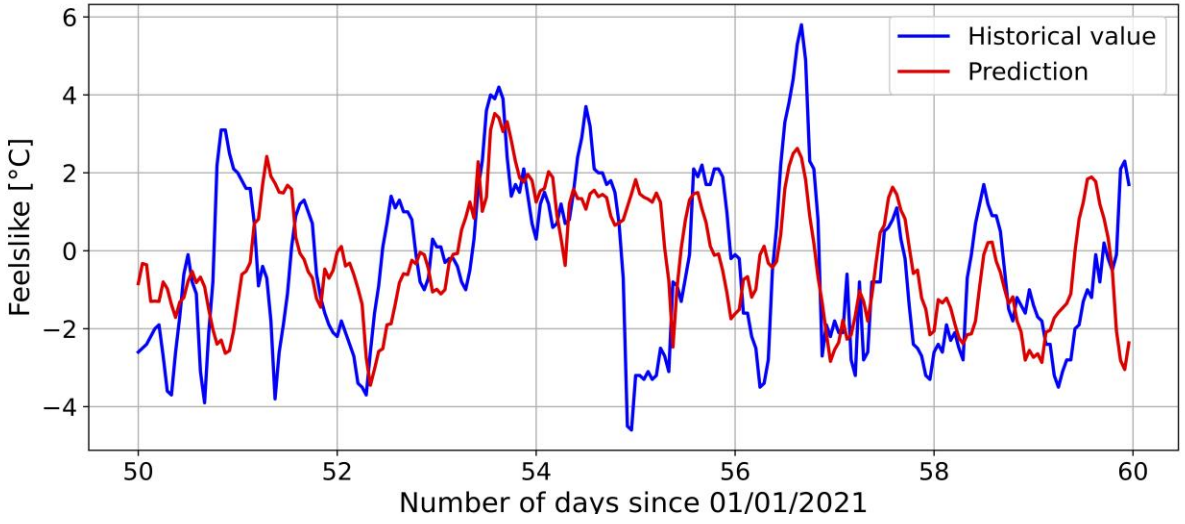
Model
LSTM

Prediction lead time
10 hour

RMSE
2.338

Performance evaluation: Single-step forecasting





Model
GRU

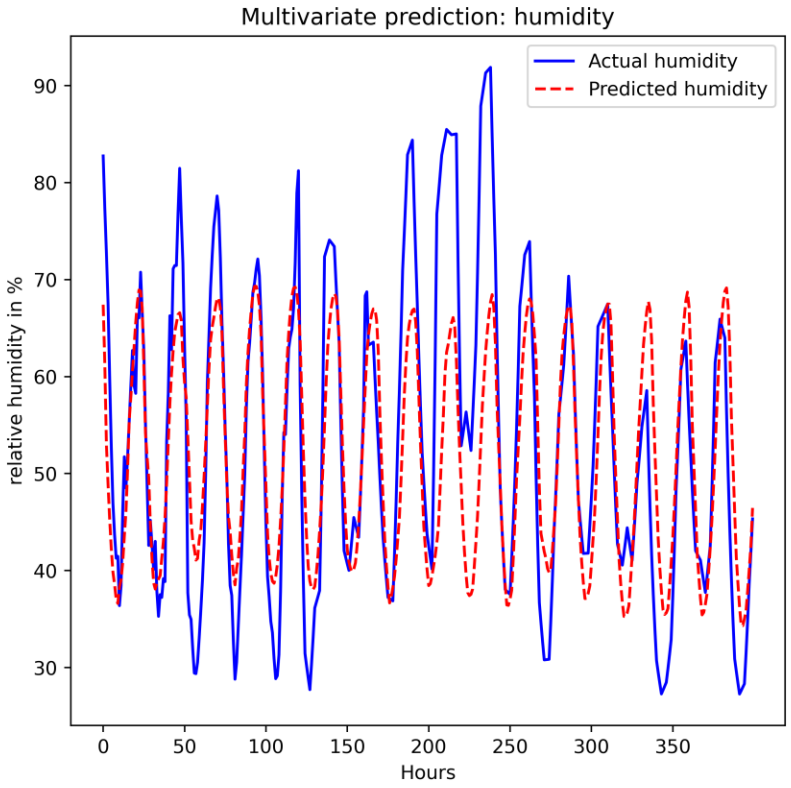
Prediction lead time
10 hour

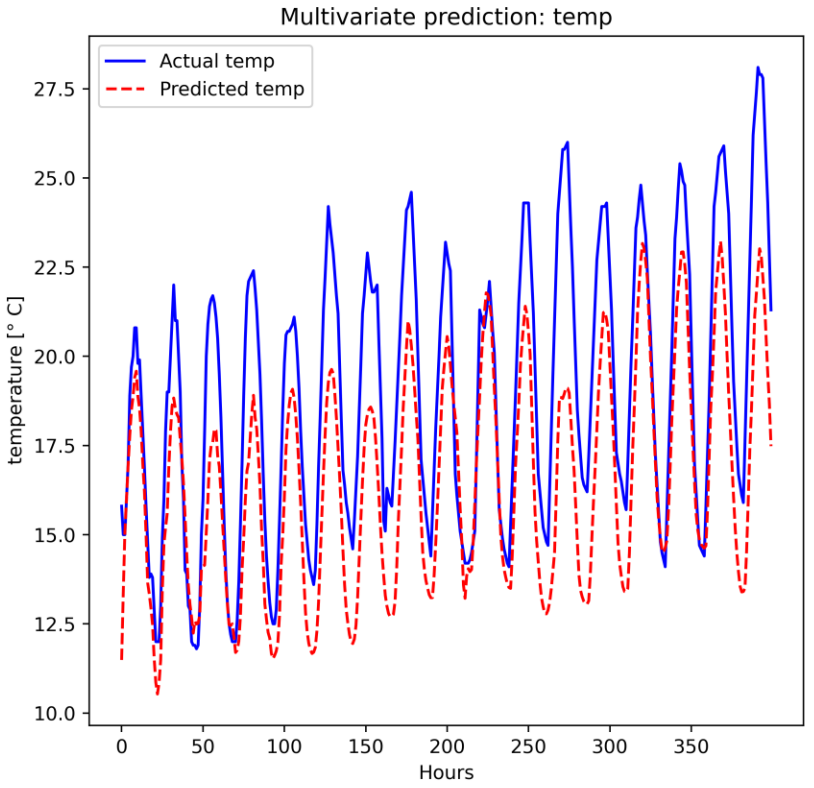
RMSE
1.920

Performance evaluation: Single-step forecasting

Model	RMSE (1 hour)	RMSE (10 hours)	Runtime
XGBoost	2.875	3.635	**fast**
LSTM	**0.753**	2.338	slow
GRU	0.786	**1.920**	slow

Performance Evaluation: Multivariate forecasting





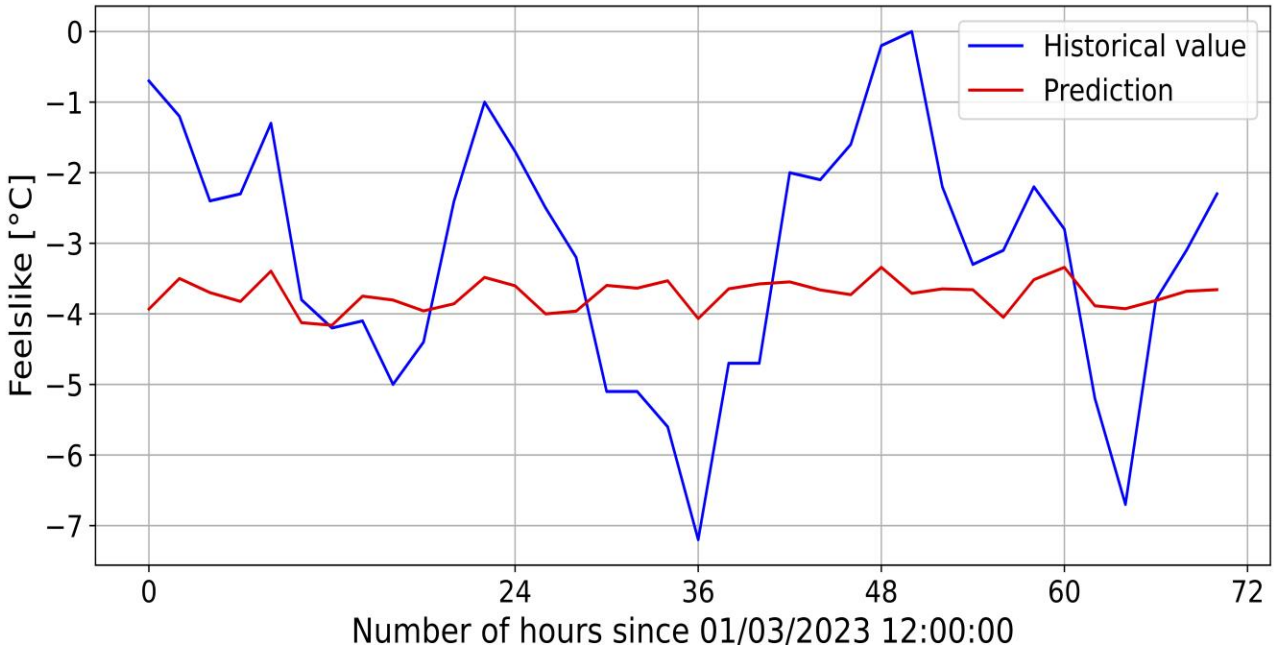
Prediction lead time

1 hour

RMSE

Humidity: 9.23
Temp: 2.81

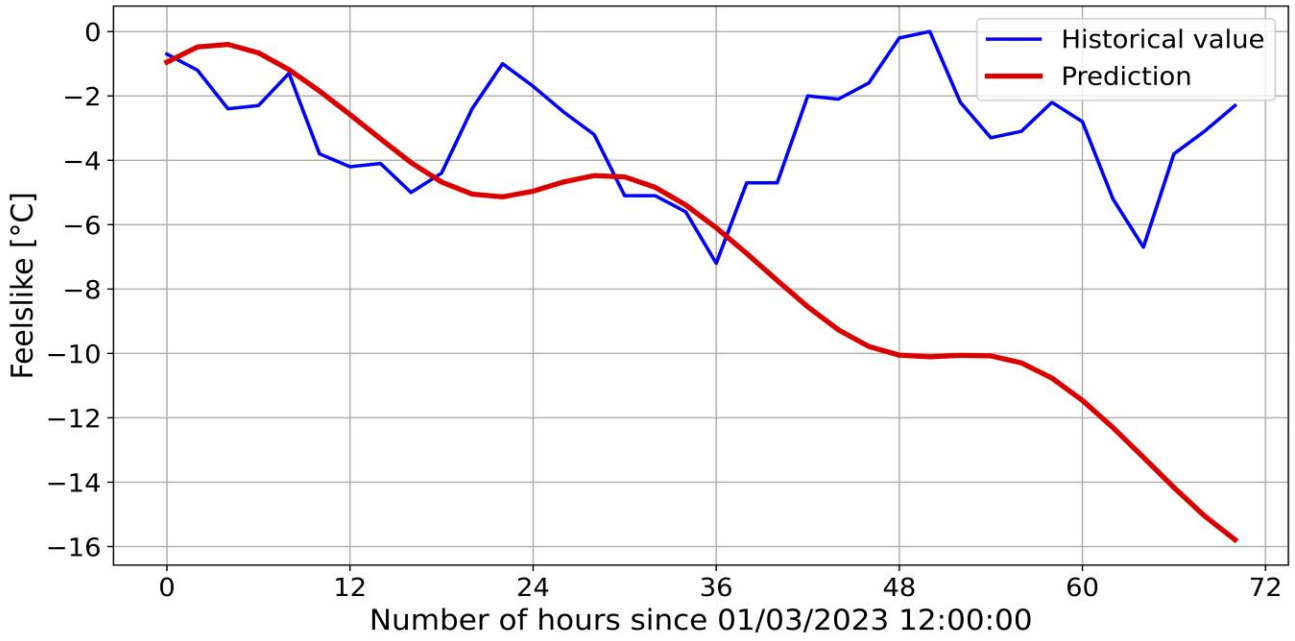
Performance Evaluation: Multistep forecasting



Method
Naive approach

RMSE
6.325

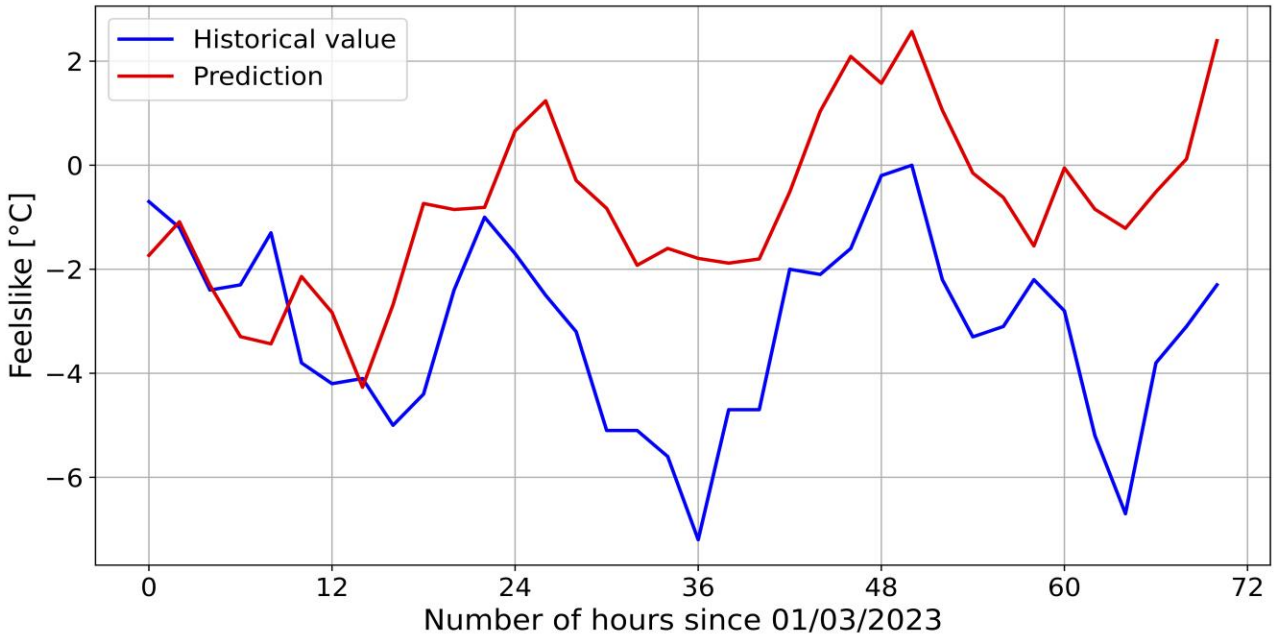
Performance Evaluation: Multistep forecasting



Method
Recursive

RMSE
5.432

Performance Evaluation: Multistep forecasting



Method
Ensemble

RMSE
3.683

Performance Evaluation: Multistep forceasting

Method	RMSE	Runtime
Naive approach	6.325	**fast**
Recursive	5.432	**fast**
Ensemble	**3.683**	slow

How to multi-step Forecasting?

"Naive approach": Make one model with X outputs

"Ensemble": Create X models which predict one timestep each

"Recursive": Train on historical data → Predict one timestep → Add prediction to historical data → Repeat

Method	Pro	Con
Naive approach	Efficient, can capture dependencies between multiple timesteps	Error Propagation, Lack of Flexibility, complex model training (specialized loss functions)
Ensemble	Flexible, error isolation, model diversity (Own model structure, HP etc.)	Increased complexity, difficulty predicting late timesteps reliably
Recursive	Gives the most realistic view?	Harder to implement, prediction error accumulates

Obstacles along the way

- It is essential to not train your model on future data → Data leakage
  - Solution: Plot your Input and label batches to make sure you do not access future labels

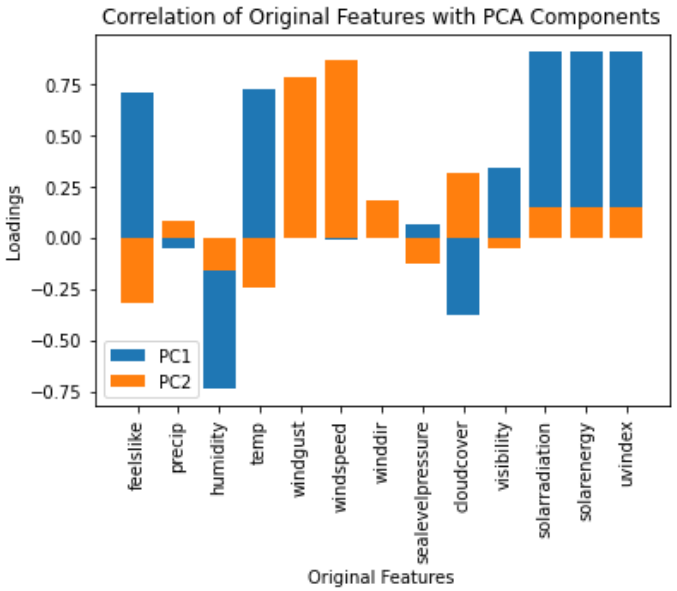
- We started with a Dataset to predict Main Direction of Imbalance in Power System
  - Unsolved problem in forecasting industry
  - Hard to predict → Difficult to evaluate model performance if predictions fail
  - Solution: Use more reliable data first -> verify that models work

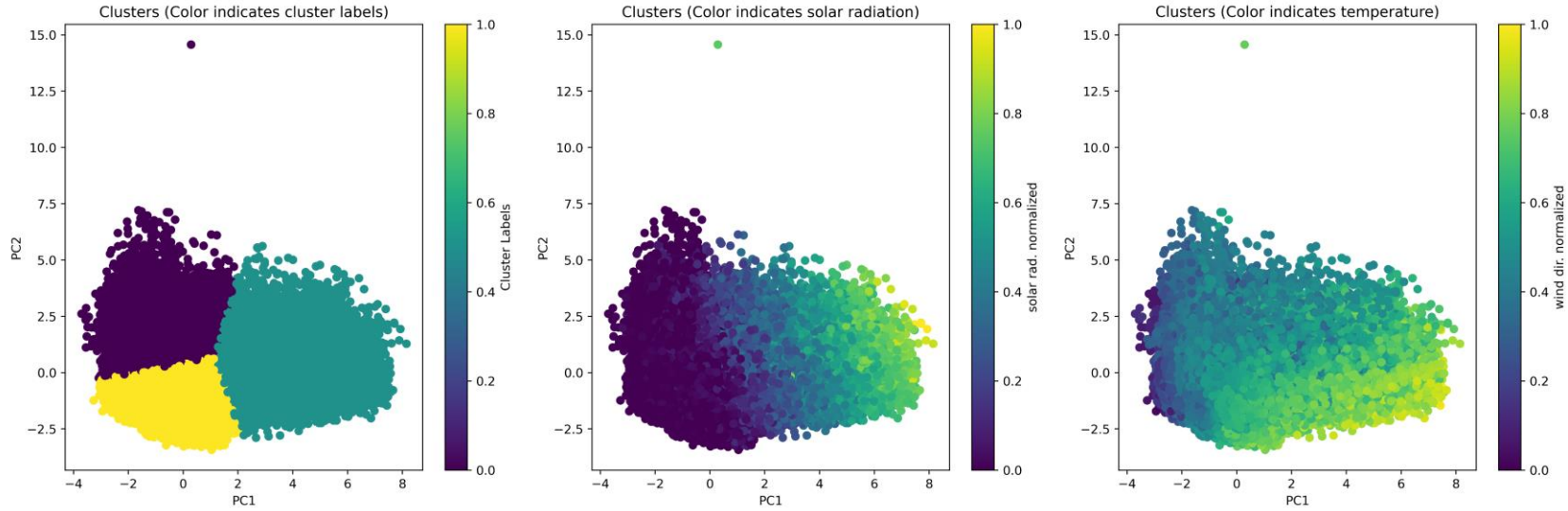
- Outlook
  - Multivariate recursive model – „best of both worlds“

Thank you for your attention!

Appendix

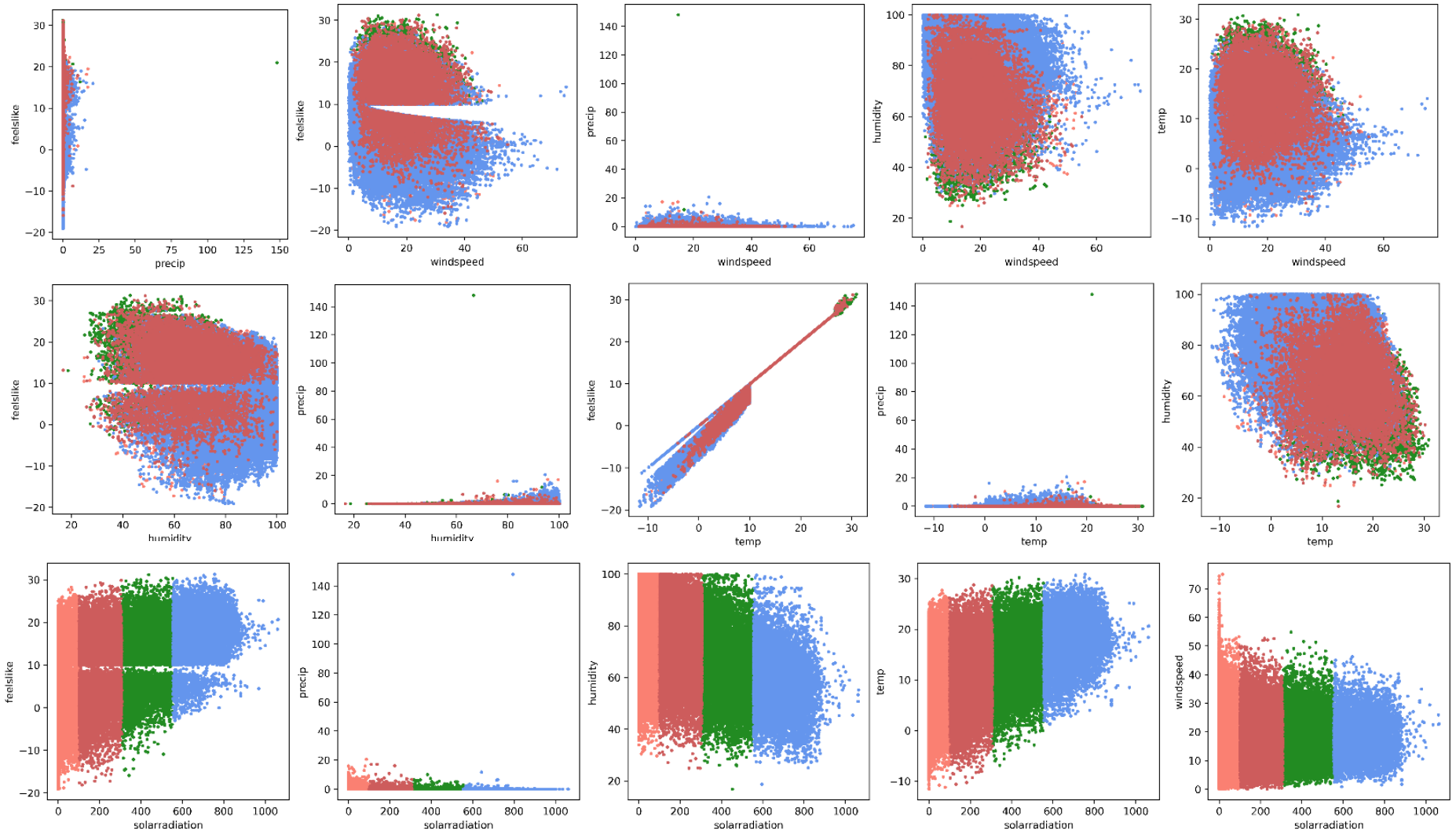
Data Preprocessing: PCA and KMeans





Tried PCA and KMeans to get a feel of the data. PCA1 might correspond to a distinction into warm sunny, and cold cloudy days and PCA2 more into windspeed/direction (just interpretation). Clustering with PCA above and without PCA and only selected features on next slide. Outliers were not removed for this analysis.

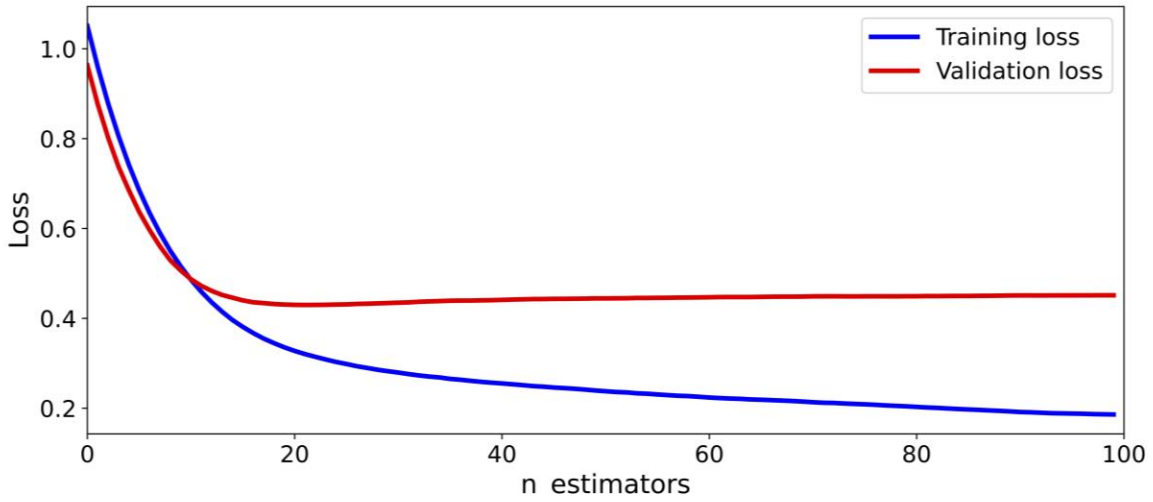
Data Preprocessing: PCA and KMeans

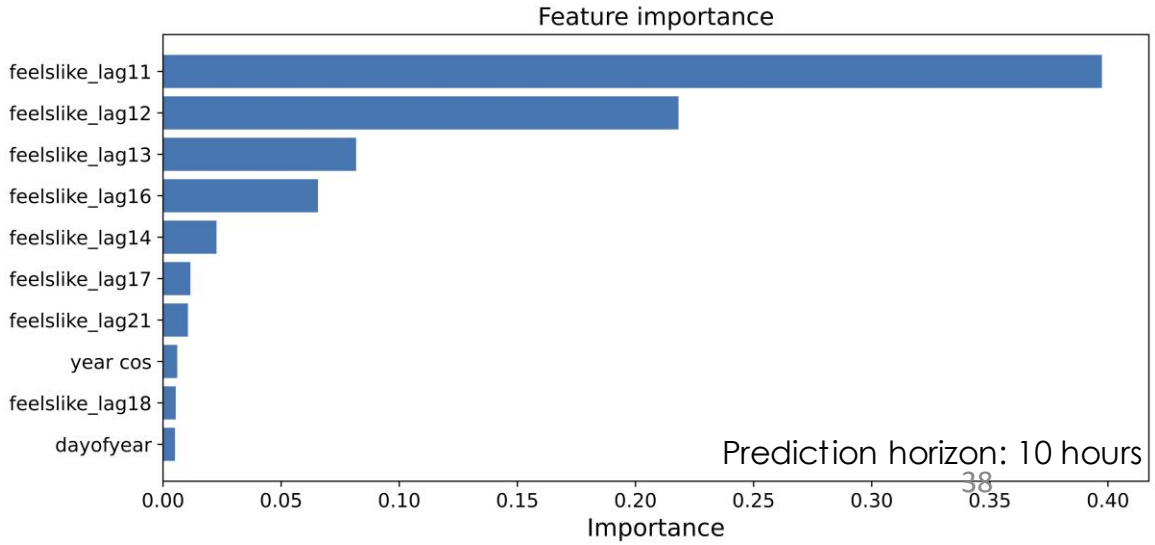


Clustering without PCA, but only 6 selected features. The algorithm seems to cluster the data mainly by solar radiation, probably since it ís a good indicator for the day/night cycle.

XGBoost single variable forecast

- Data preparation:
  - using features from previous time steps for the input features, including the prediction variable (sliding window)
  - Most of the features have daily and yearly periodicity, therefore sine and cosine features
- Hyperparameter optimization : Optuna
  - Learning_rate: 0.0268
  - max_depth: 3
  - N_estimators: 184
- Feature importance: built-in XGBoost feature importance
  - As expected, the best features are the prediction feature of previous timesteps (the best are the closest in time to the prediction)
- Evaluation: The worst of the three models for single variable forecast, but still ok. Also the fastest runtime.

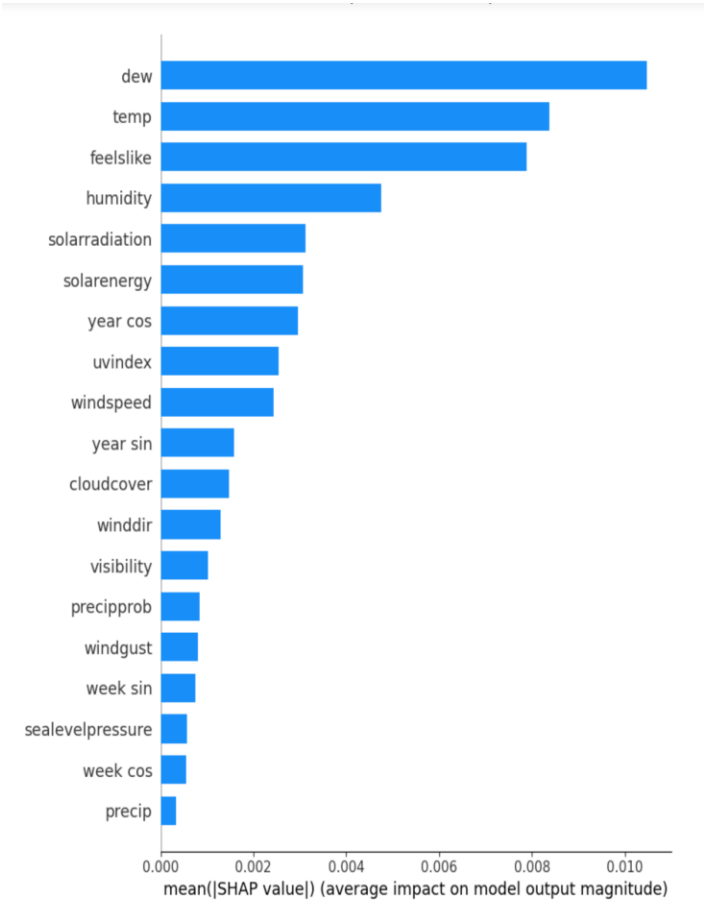


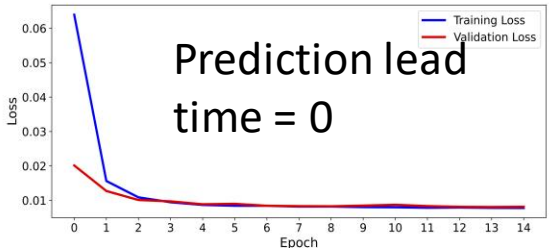


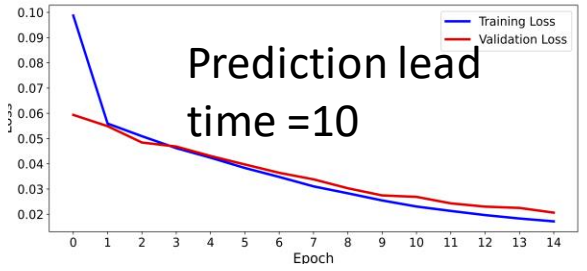
Prediction horizon: 10 hours

LSTM single variable forecast

- Data preparation:
  - Features used: 'temp', 'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'windgust', 'windspeed', 'winddir', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex', 'week sin', 'week cos', 'year sin', 'year cos'
- Hyperparameter optimization : Optuna + Grid search
  - Learning_rate: 0.00138
  - Batchsize: 36
  - Validation split = 0.3
  - Hidden layers: LSTM (96), LSTM (36), DENSE(26), Dense(13)
  - Optimizer, loss function: Adam, MSE
  - Trainable params: 65,023
- Feature importance: SHAP values
  - Dew, temp, feelslike seems to be the most important. Quite different compared to shap values for the GRU model, but the explanation could be that most of the variables are quite correlated.
- Evaluation
  - Good model with 10x runtime compared to XGBoost. Seems comparable to GRU model, but LSTM worked better on small prediction lead times.

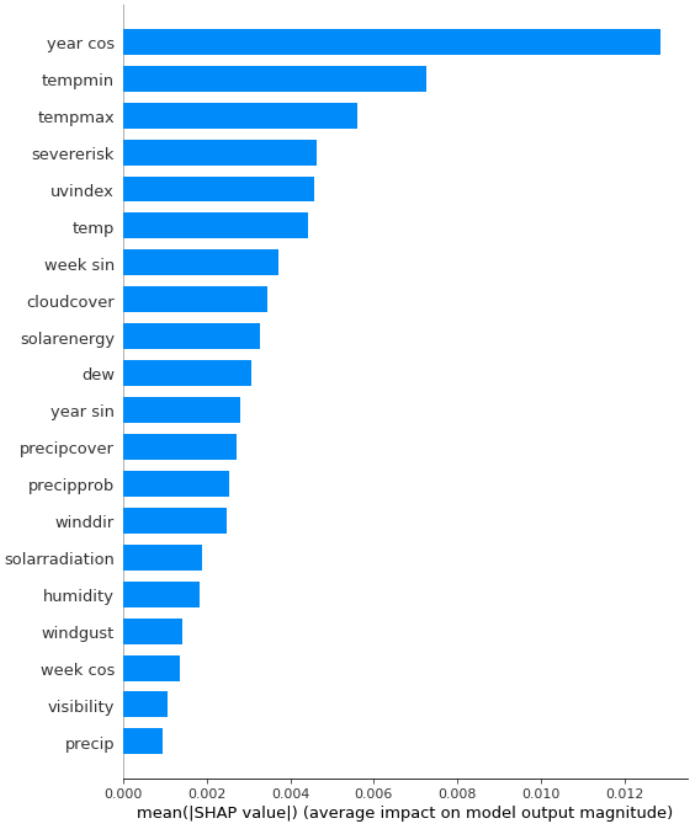


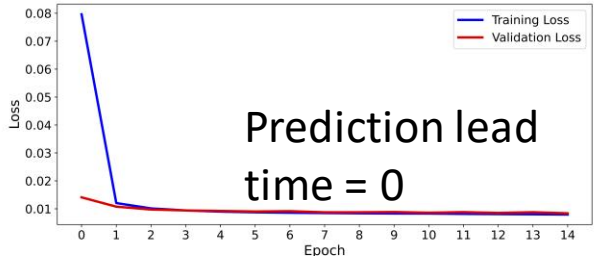


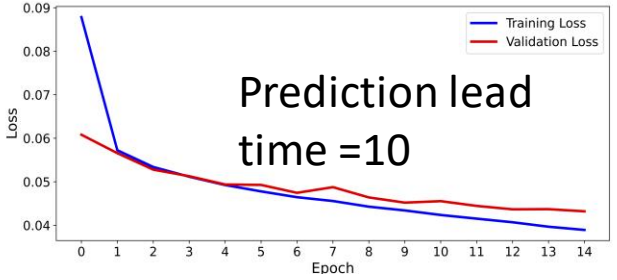


GRU single variable forecast

- Data preparation:
  - Features used: 'temp', 'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'windgust', 'windspeed', 'winddir', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex', 'week sin', 'week cos', 'year sin', 'year cos'
- Hyperparameter optimization : Optuna
  - Learning_rate: 0.00122
  - Batchsize: 45
  - Validation split = 0.3
  - Hidden layers: GRU(94), GRU(50), DENSE(26), Dense(10)
  - Optimizer, loss function: Adam, MSE
  - Trainable params: 55,937
- Feature importance: SHAP values
  - "Year cos" was the most important feature, together with tempmin, tempmax.
- Evaluation
  - Good model with 10x runtime compared to XGBoost. Comparable to LSTM model, but GRU worked better on longer prediction lead times

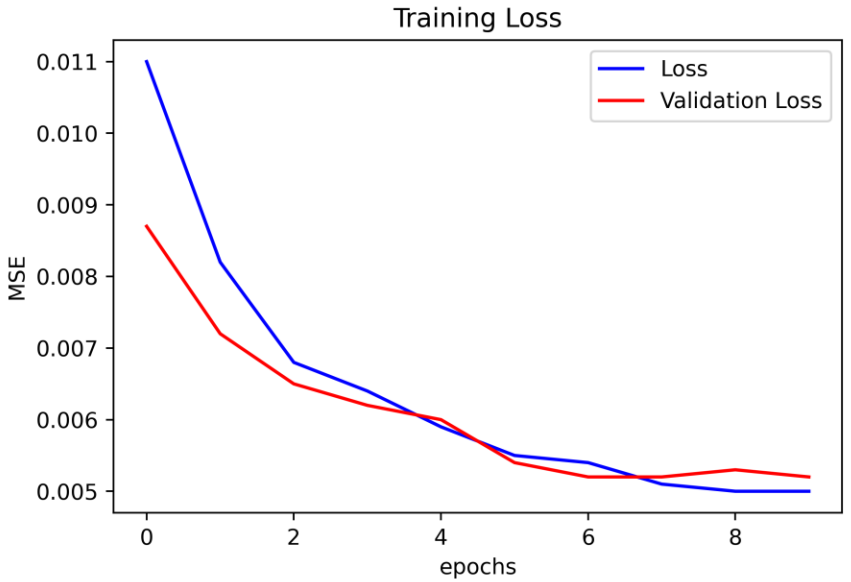


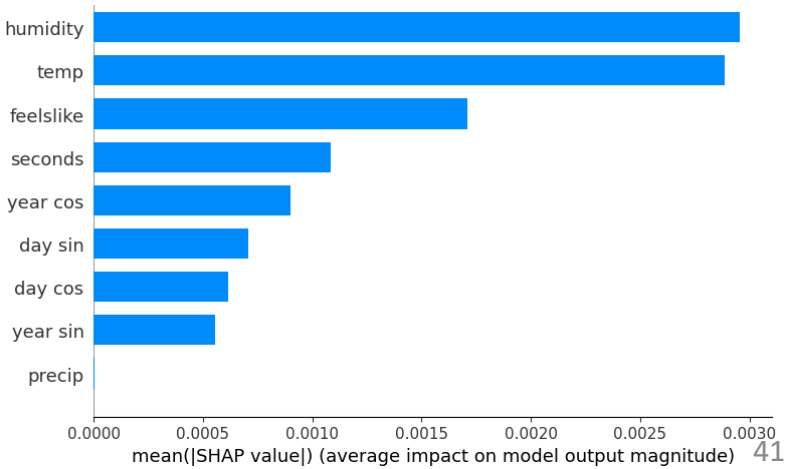




LSTM Multivariate forecast

- Data preparation:
  - Features used: "feelslike", "year cos", "year sin", "day cos", "day sin", "seconds", "precip", "humidity", "temp"
- Hyperparameter optimization : Optuna
  - Learning_rate: 0.0008
  - Batchsize: 36
  - Hidden layers: LSTM (90), LSTM (64), LSTM (32)
  - Optimizer, loss function: Adam, MSE
  - Trainable params: 87,658
- Feature importance: SHAP values
  - As expected, the target variables themselves are most influential. Feelslike correlates strongly with temp and the others let the model train the time dependence.
- Evaluation: Difficult to not overtrain one of the predictions. Could predict temperature reasonably well (as good as XGB) but could only learn overall trend of humidity. Training two models separately seems to be superior, unless you use multivariate forecasting for a recursive multistep forecast





Multistep forecasting methods

- Recursive method:
  - Features used: "feelslike"
  - Architecture + hyperparameters: Same as LSTM single variable forecast
  - Evaluation: Good model, but worked worse than we thought. Problem: Error accumulates.

- Ensemble method:
  - Features used: temp', 'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'windgust', 'windspeed', 'winddir', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex', 'week sin', 'week cos', 'year sin', 'year cos'
  - Architecture: 72 x LSTM single variable forecast
  - Evaluation: Best out of the three multistep methods, but long run time.

- Naive approach:
  - Features used: temp', 'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'windgust', 'windspeed', 'winddir', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex', 'week sin', 'week cos', 'year sin', 'year cos'
  - Architecture: Same as LSTM single variable forecast but with prediction horizon 72 hours. Optimized hyperparameters with randomized grid search.
  - Evaluation: Easy to implement but did not give the best results.

Statement on individual contributions

All participants contributed evenly.