1. Overview

Problem Definition

This is a future data prediction competition with an active training phase and a second period where the solution will be evaluated against future ground truth data.

The goal of the competition is to create an energy prediction model of prosumers to reduce energy imbalance costs.

This competition aims to tackle the issue of energy imbalance, a situation where the energy expected to be used doesn't line up with the actual energy used or produced. Prosumers, who both consume and generate energy, contribute a large part of the energy imbalance. Despite being only a small part of all consumers, their unpredictable energy use causes logistical and financial problems for the energy companies.

Competition webpage

https://www.kaggle.com/competitions/predict-energy-behavior-of-prosumers

Metric

Solutions are evaluated on the Mean Absolute Error (MAE) between the prediction and the observed target. The formula is given by:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i|,$$

where

n is the total number of data points,

y_i is the predicted value for data point i,

x_i is the observed value for data point i.

Available data

train.csv:

Provides the target – the consumption or production amount for the relevant segment for the hour. The segments are defined by the `county`, `is_business`, and `product_type`.

client.csv

For each client (target time series) provides for each day

- The aggregated number of consumption points (EICs - European Identifier Code).
- Installed photovoltaic solar panel capacity in kilowatts.

gas_prices.csv

Provides the lowest and the highest price of natural gas for each day, in Euros per megawatt hour equivalent.

electricity_prices.csv

The price of electricity in euros per megawatt hour, given for each hour.

forecast_weather.csv

Weather forecasts for each hour.

historical_weather.csv

Historical weather for each hour.

weather_station_to_county_mapping.csv

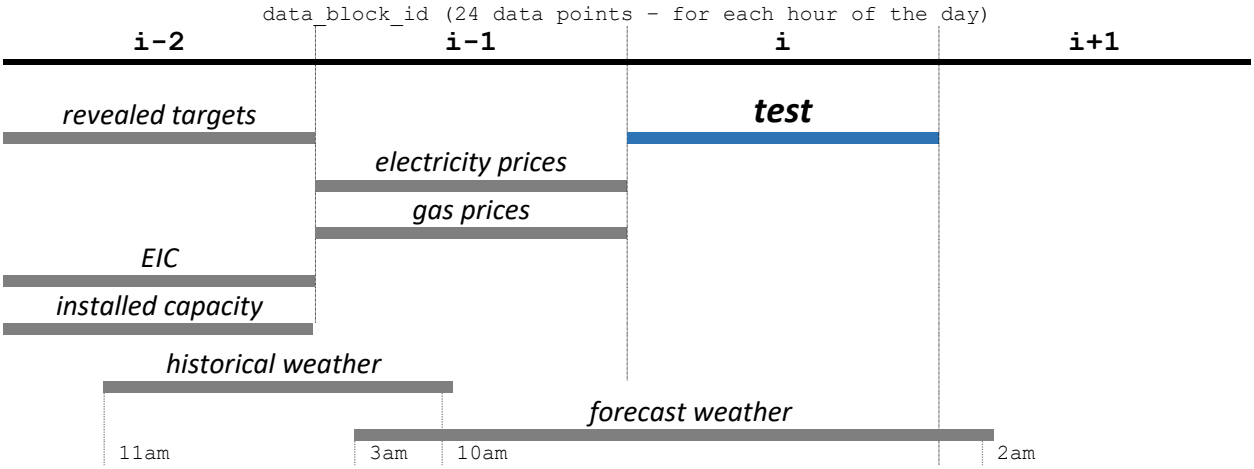
Mapping of coordinates to `county` (id and name).

Strategy for solving the problem

1. Explore available data
2. Create a dataset suitable for prediction
3. Train LightGBM regressor for forecasting

2. Inference Data

- Inference API iteratively provides data for forecasting. Auxiliary data is provided with some delays.
- During one iteration of inference the goal is to forecast 24 data points (one day) for each time series.
- The most recent data provided during the i^{th} iteration of inference:



3. Training Data

EDA Notebooks

`EDA_target.ipynb` – explore target.

`EDA_auxiliary_data.ipynb` – explore client, prices, weather data.

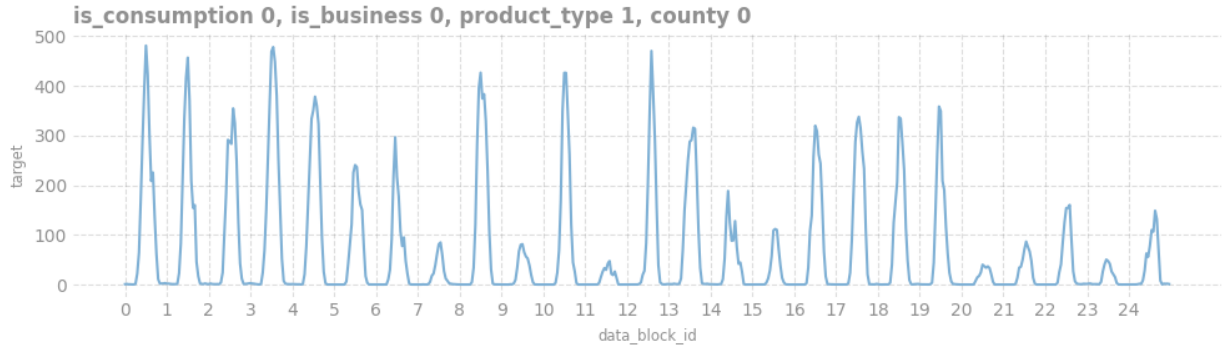
`train_features.py` – assemble all data into a DataFrame suitable for training

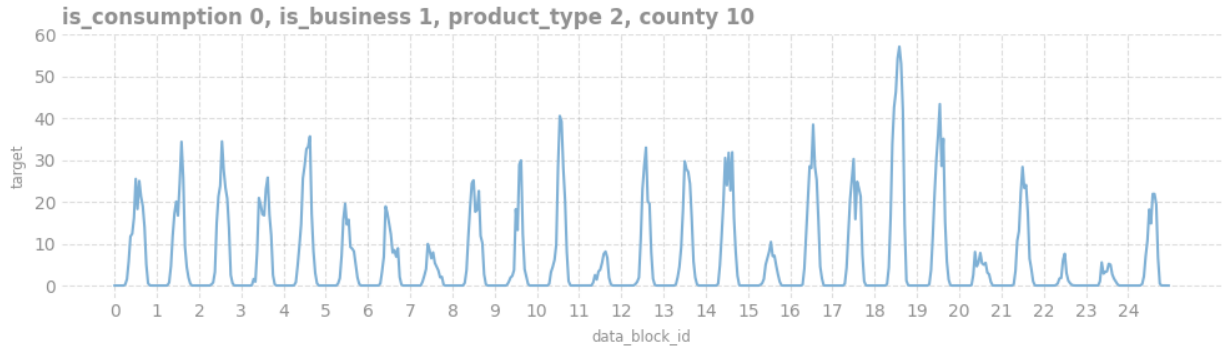
Target

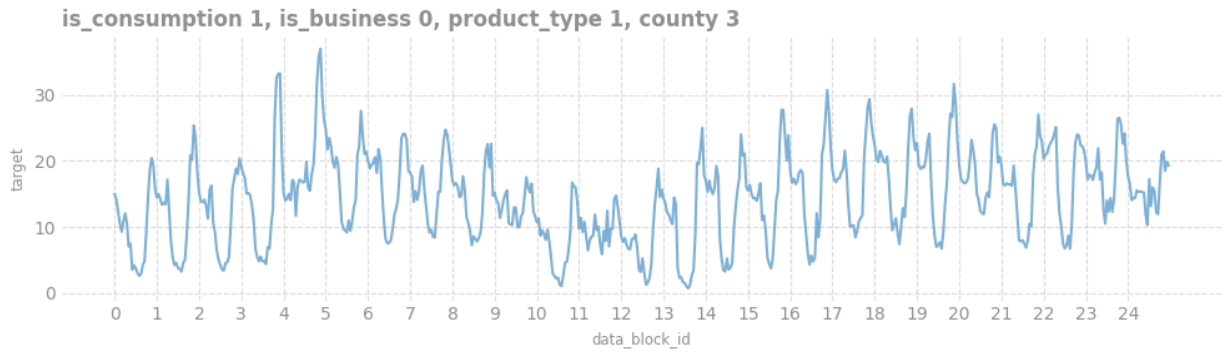
Overview

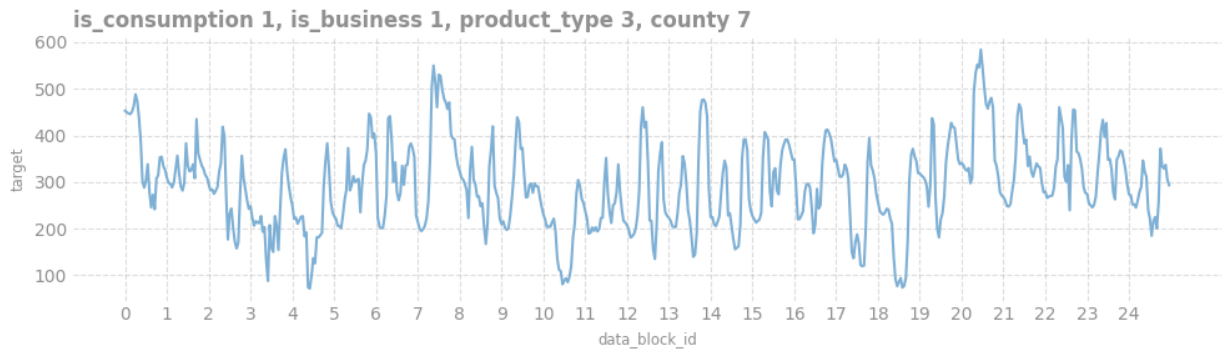
- 138 time series spanning 638 days, values are provided hourly (24 values for each day);
- Each time series is defined by
    - `is_consumption` (binary),
    - `is_business` (binary),
    - `product_type` (4 values),
    - `county` (16 values) variables;
- Target is available for the whole day for each hour for each time series, but some days may be missing;
- 528 NaN values (1 value out of 24 may be missing for some day). Can be inferred as a mean of adjacent values (no 2 consecutive hours are missing).

Examples of target for different time series

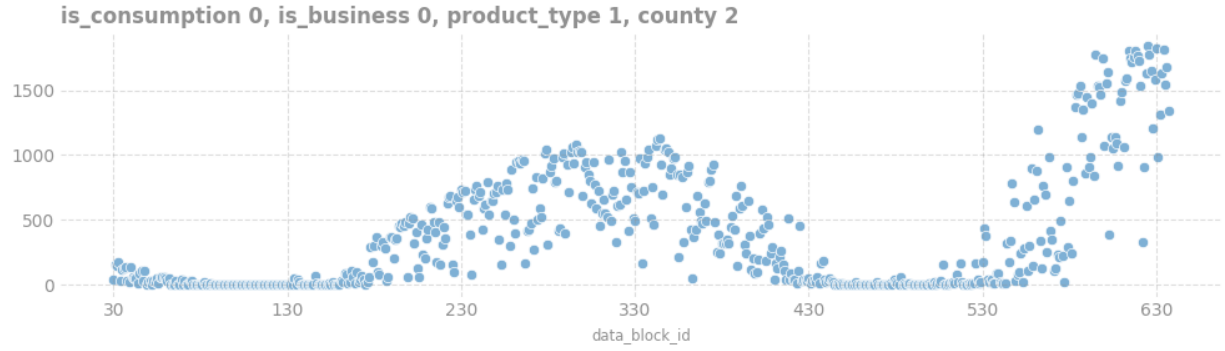


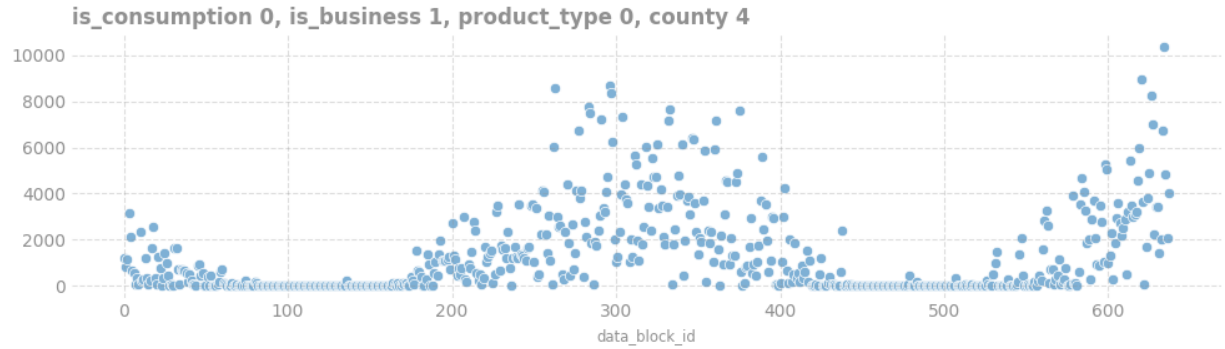


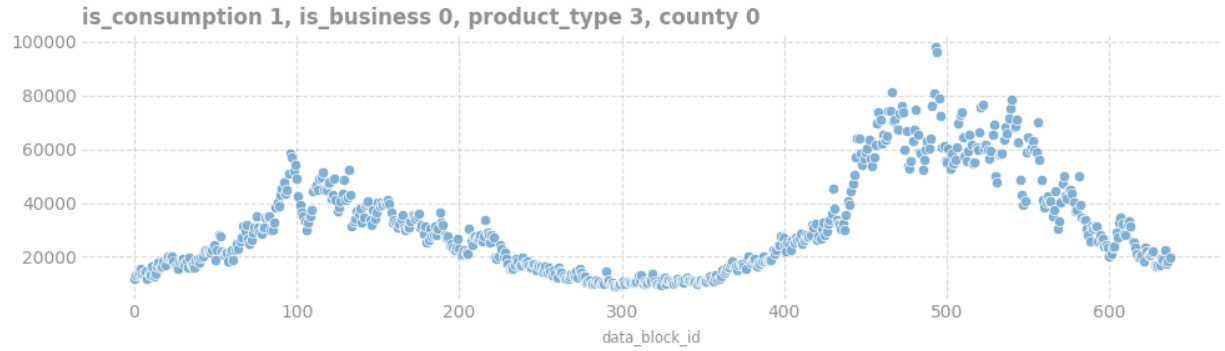


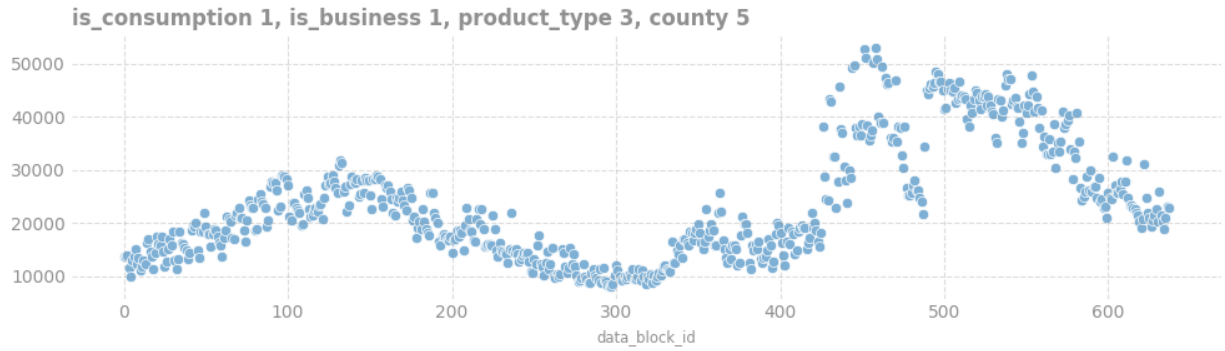


Examples of target for different time series aggregated by day (sum)









Insights

- Strong daily patterns,
- Strong yearly patterns,
- Increase with time,
- Two distinct clusters of time series with `is_consumption` 0 and 1.

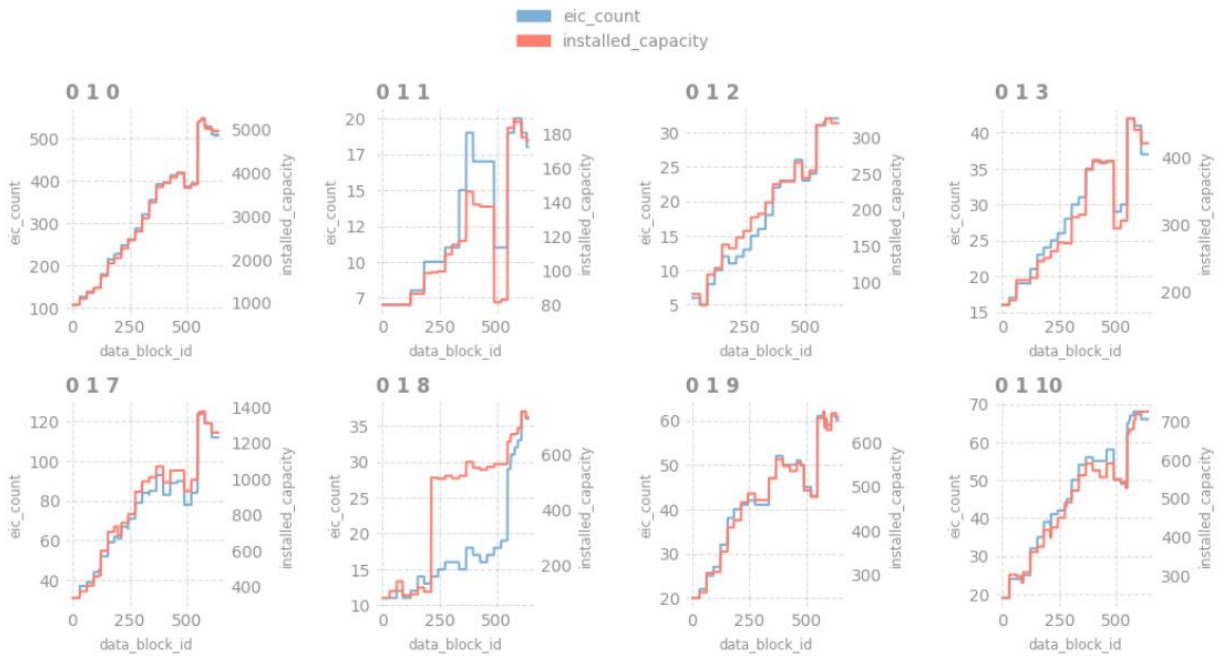
Client

For each set of {`is_business, product_type, county`} the values are provided for each `data_block_id`:

- `eic_count` – integer positive values, the aggregated number of consumption points (EICs - European Identifier Code),
- `installed_capacity` – real positive values, installed photovoltaic solar panel capacity in kilowatts.

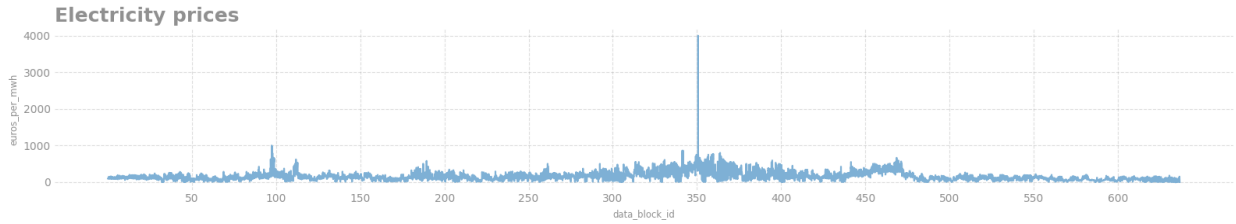
Examples of client data for different time series

The title of each subplot contains values for [`is_business, product_type, county`]

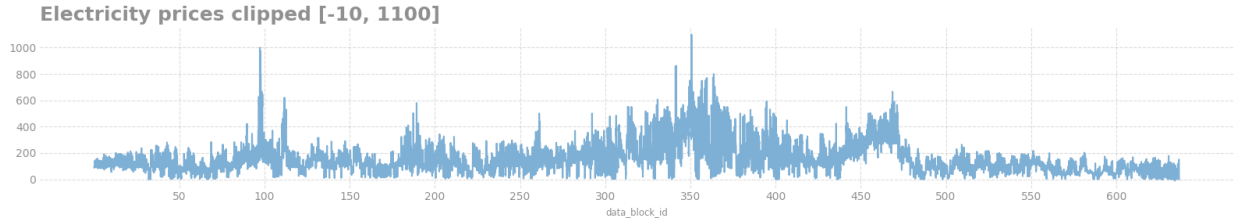


Electricity prices

- Contains the price of electricity (`euros_per_mwh`) for each hour for each day,
- Real values, both positive and negative,

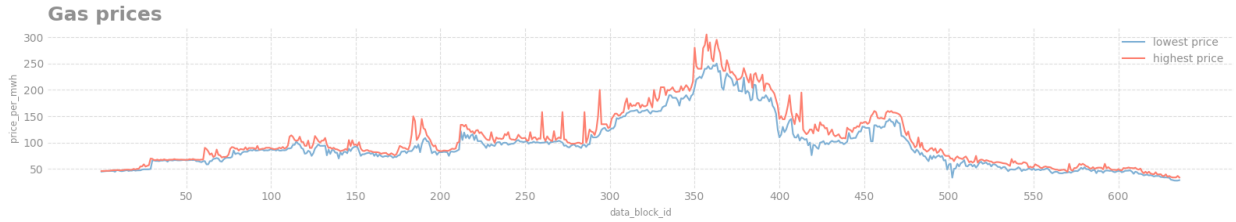


- A single data point with value above 1100:
  - `data_block_id` 351: `euros_per_mwh=4000`,
  - set upper limit on the values of electricity prices.



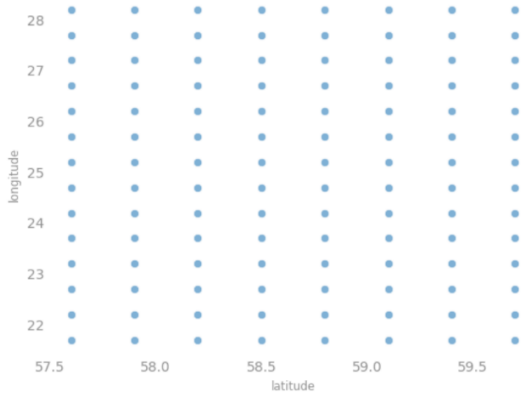
Gas prices

- Contains the lowest and the highest prices of natural gas for each day,
- Real positive values.

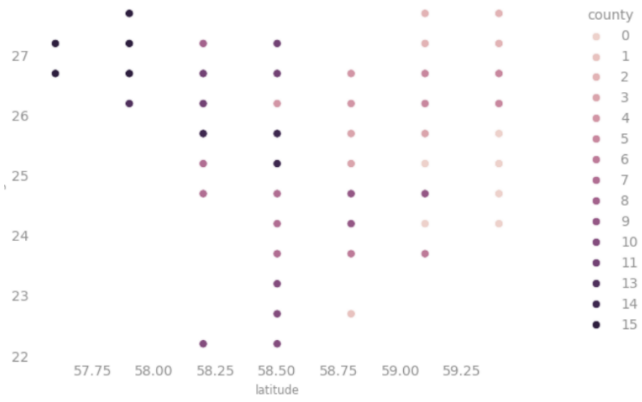


Weather

- A set of weather features are provided hourly for each point on a coordinate grid:



Weather grid



Counties on a grid

- County 12 is missing on the grid.

Historical weather

Features: temperature, dewpoint, rain, snowfall, surface_pressure, cloudcover_total, cloudcover_low, cloudcover_mid, cloudcover_high, windspeed_10m, winddirection_10m, shortwave_radiation, direct_solar_radiation, diffuse_radiation.

Forecast weather

Each day at a given hour forecast of weather features from 1 to 48 hours ahead.

Features: temperature, dewpoint, cloudcover_high, cloudcover_low, cloudcover_mid, cloudcover_total, 10_metre_u_wind_component, 10_metre_v_wind_component, surface_solar_radiation_downwards, direct_solar_radiation, snowfall, total_precipitation.

4. Generated features

- Target lags: target values from previous days on the same hour – 2,3,4,5,6,7,14 and 21 days,
- Gas price,
- Electricity price,
- EIC,
- Installed capacity,
- Mean historical weather features for each hour,
- Mean forecast weather features for each hour,
- Mean forecast weather from the hour before and the hour after the current time.

Correlations to target

The most correlated to target are target lags and installed capacity.

feature	abs corr
target_168	0.967732
target_336	0.962427
target_504	0.956631
target_144	0.951987
target_48	0.936987
target_120	0.932641
target_49	0.932501
target_72	0.929452
target_96	0.927719
target_50	0.921870
installed_capacity	0.608667
eic_count	0.293496
is_consumption	0.204396
product_type	0.174969
is_business	0.162848
county	0.098669
day_cosine	0.064362
surface_solar_radiation_downwards_x	0.061642
surface_solar_radiation_downwards_y	0.061161
surface_solar_radiation_downwards_x_-1	0.059365
surface_solar_radiation_downwards_y_-1	0.059338
surface_solar_radiation_downwards_x_1	0.058922
surface_solar_radiation_downwards_y_1	0.058157
shortwave_radiation	0.056803
direct_solar_radiation_y	0.053580
diffuse_radiation	0.052577
direct_solar_radiation_y_1	0.052458
direct_solar_radiation	0.052280
direct_solar_radiation_x	0.052269
direct_solar_radiation_y_-1	0.051722
. . .	

Clusters of the data

Experiments show that training and inferring each time series separately is less effective than training on a cluster of time series.

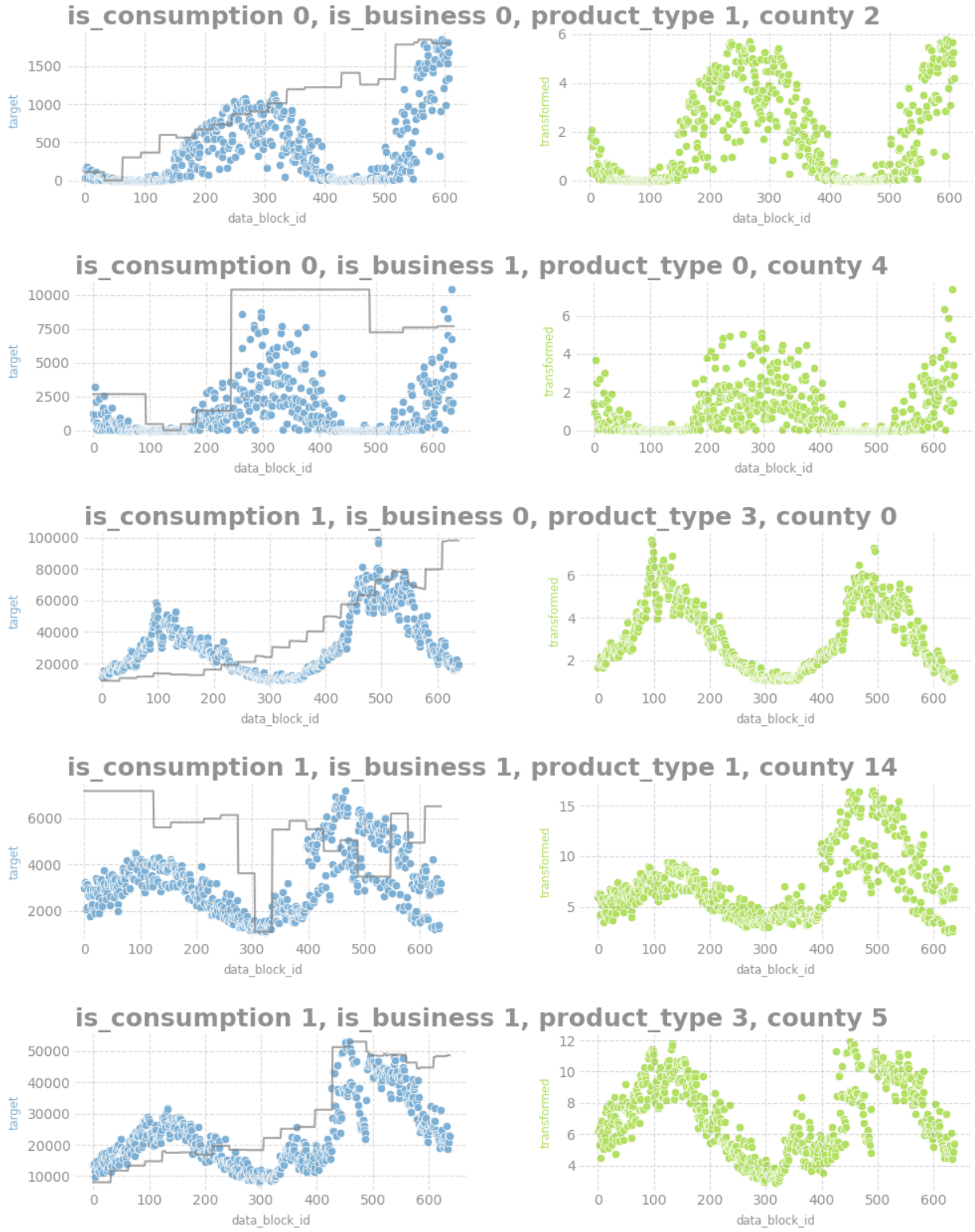
The target time series are naturally clustered using the values of `is_consumption` feature. Further split that results in 4 major clusters is possible using the feature `is_business`.

Target divided by the value of installed capacity

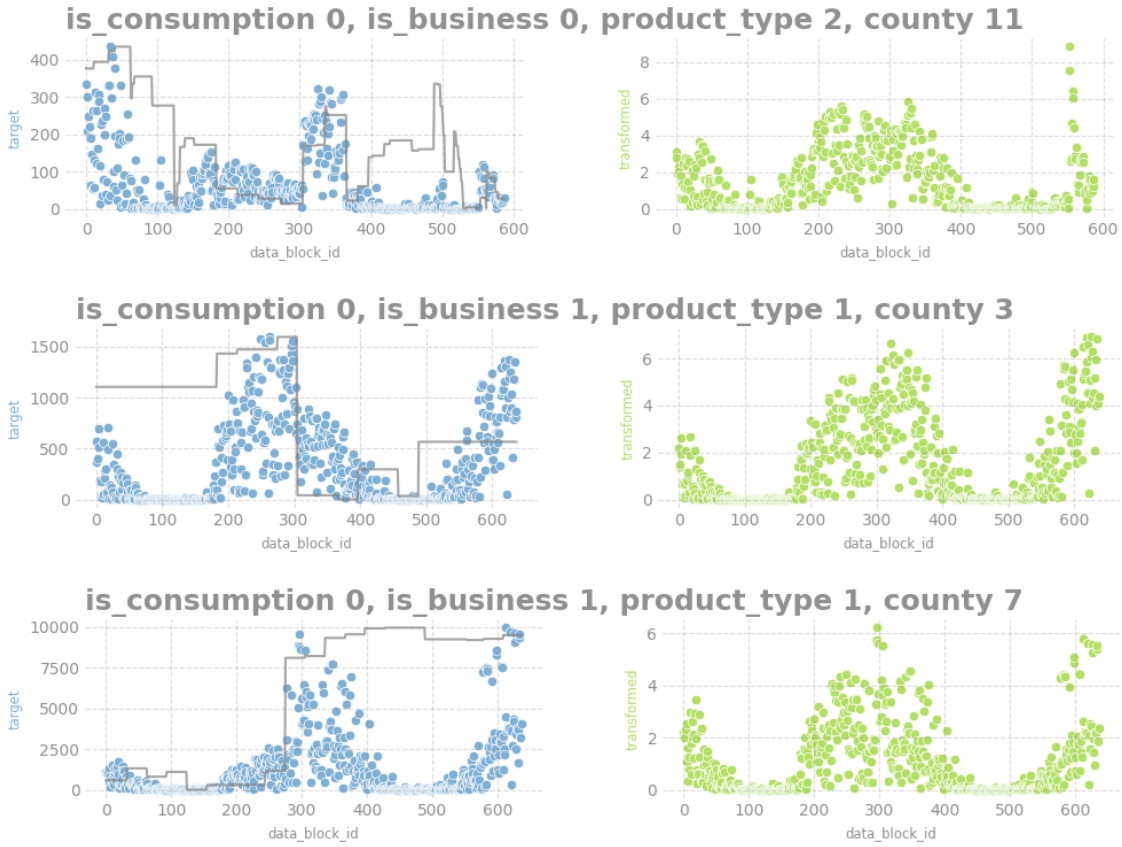
Experiments show that the following transformation produces a normalizing effect:

$$new\_target = \frac{target}{installed\_capacity}.$$

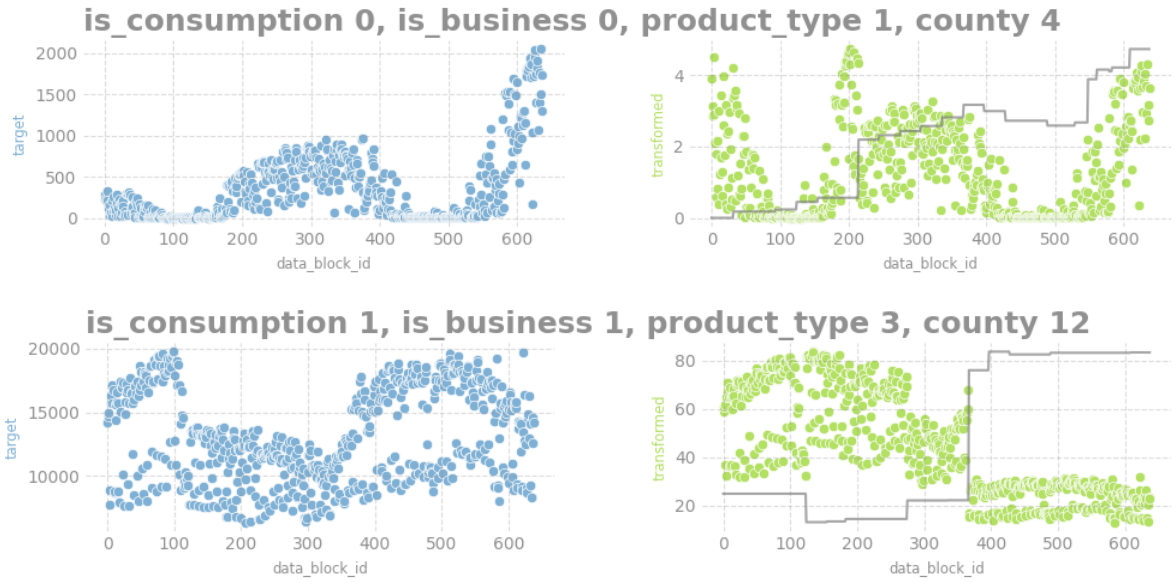
Examples of the normalizing effect of the transform



Examples of improvement of the target after the transform



Examples of deterioration of the target after the transform



Insights

- The target/installed_capacity transform generally produces a normalizing effect for clusters `[is_consumption,is_business]`: [0,0], [0,1], [1,0].
- Experiments show that the cross validation score for the cluster [1,1] is worse with this target transform.

Difference between target and target_lag_48h

A suitable target transform for the cluster [1,1] is the difference between target and its value 48 hours prior:

$$new\_target = target - target\_lag\_48h.$$

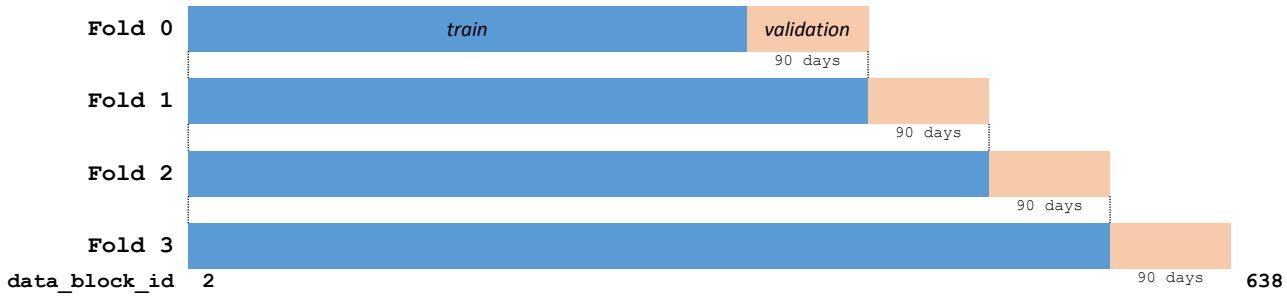
This target transform is less effective for clusters [0,0], [0,1], [1,0] than the transform described in the previous section.

5. Solution overview

According to the conditions of the competition, the evaluation of the solution will be performed over a 3 months long period of data.

Cross validation strategy

- 4 folds,
- Each fold is validated on the following 3 months of data:



Baseline single-model solution

Notebook

`baseline_solution.ipynb`

Model

`LightGBMRegressor` with parameters:

- `random_state=42,`
- `n_estimators=500,`
- `max_depth=5,`
- `num_leaves=31,`
- `objective='mae',`
- `device='gpu'.`

Cross validation score

MAE score for each cluster for each fold:

Cluster	Fold 0	Fold 1	Fold 2	Fold 3	Mean
0 0	50.245	23.542	9.139	96.067	44.369
0 1	57.129	19.334	5.785	83.567	41.291
1 0	9.849	20.013	35.600	32.137	24.411
1 1	106.866	95.063	114.728	101.710	104.518

Insights

- Cluster [1,1] has the worst average score;
- Clusters [0,0] and [0,1] have significantly worse score for the final fold – due to the target's increasing amplitude.

Tuned 4-model solution

Notebook

`tuned_solution.ipynb`

Models

Cluster [0,0]

- Target transformed as $new\_target = target / installed\_capacity$.
- `LightGBMRegressor` with parameters tuned using Weights&Biases sweep.

Cluster [0,1]

- Target transformed as $new\_target = target / installed\_capacity$.
- `LightGBMRegressor` with parameters tuned using Weights&Biases sweep.

Cluster [1,0]

- Target transformed as $new\_target = target / installed\_capacity$.
- `LightGBMRegressor` with parameters tuned using Weights&Biases sweep.

Cluster [1,1]

Mean of the predictions of two models.

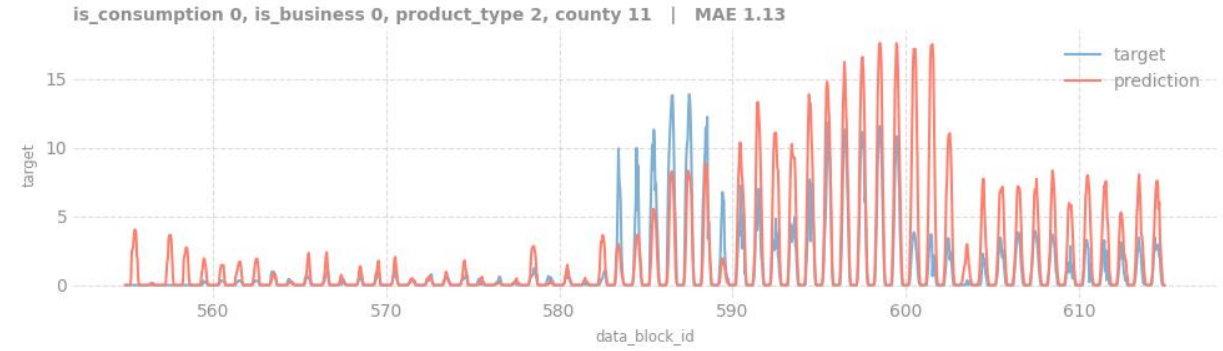
- Model 1:
  - Target transformed as $new\_target = target - target\_lag\_48h$,
  - `LightGBMRegressor` with parameters tuned using Weights&Biases sweep,
  - Trained on data from clusters [1,0] and [1,1].
- Model 2:
  - Original target,
  - `LightGBMRegressor` with parameters tuned using Weights&Biases sweep,
  - Trained on data from clusters [1,0] and [1,1].

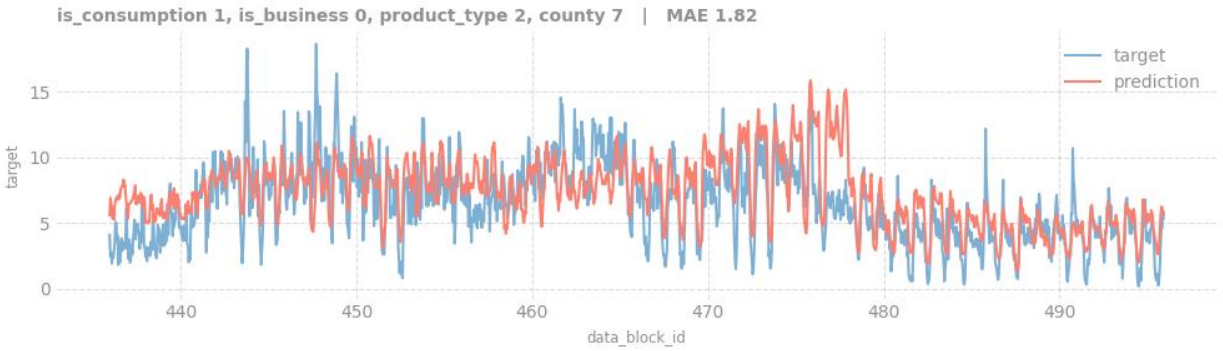
Cross validation score

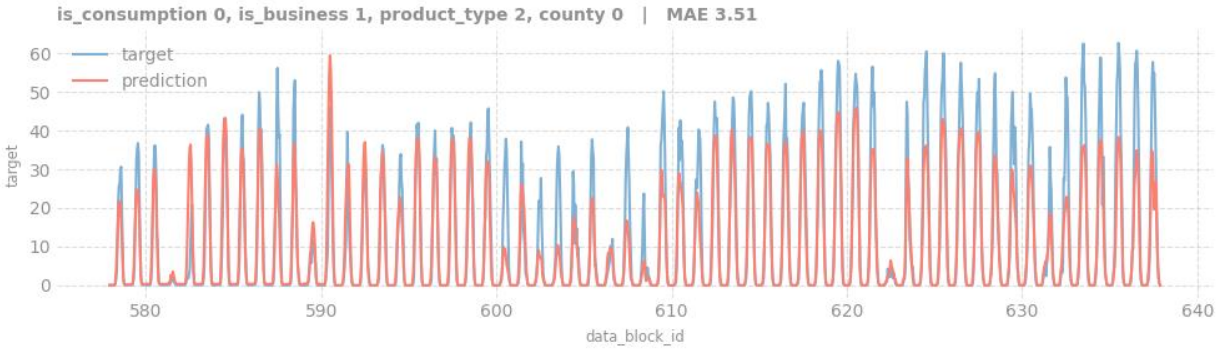
MAE score for each cluster for each fold:

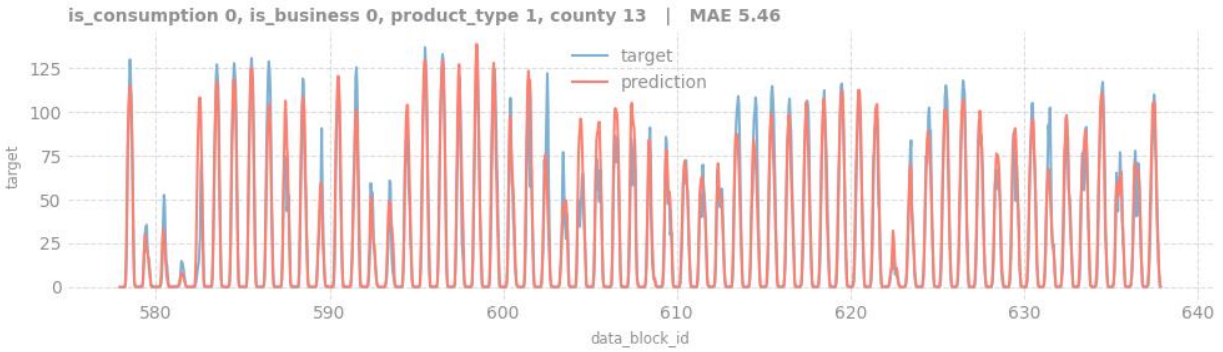
Cluster	Fold 0	Fold 1	Fold 2	Fold 3	Mean
0 0	35.369	16.694	6.652	49.308	26.806
0 1	44.934	14.027	4.956	56.596	30.002
1 0	11.937	15.221	22.132	21.668	17.737
1 1	90.988	80.976	86.503	88.891	86.793

Examples of out-of-fold predictions with the best MAE scores

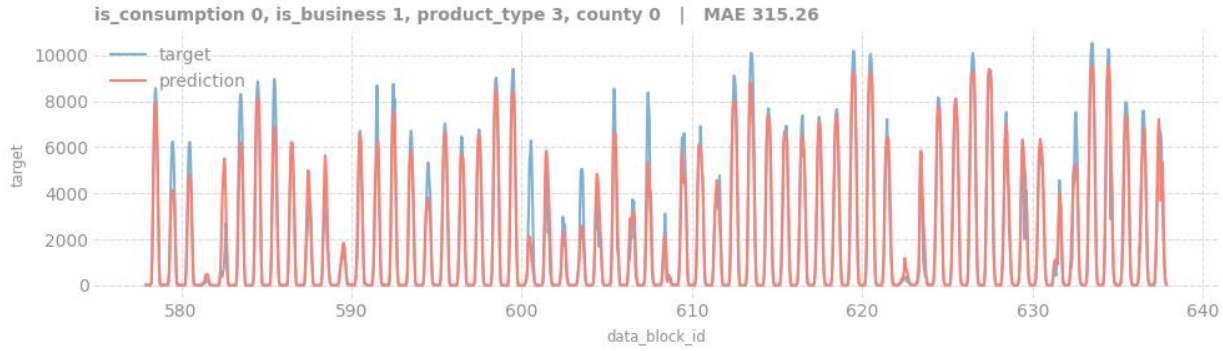


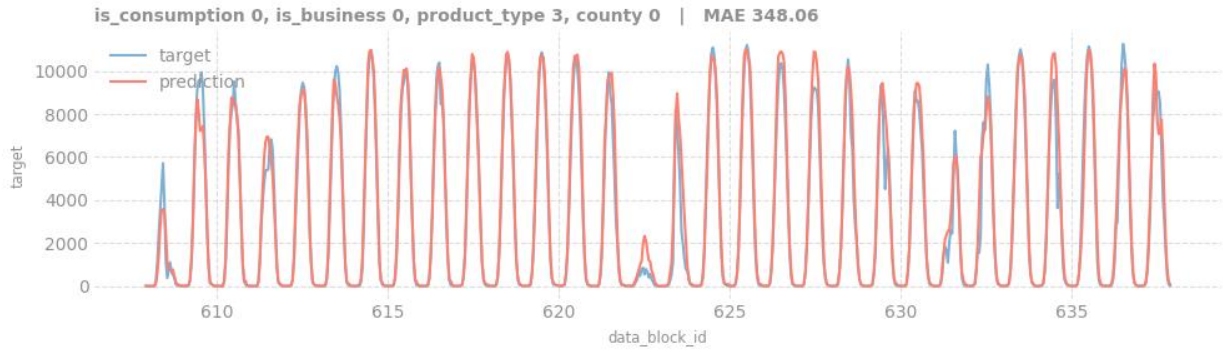


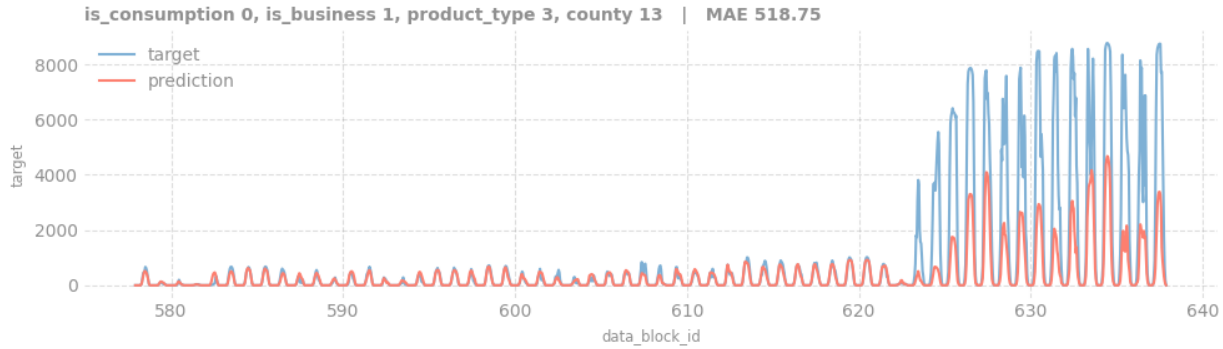


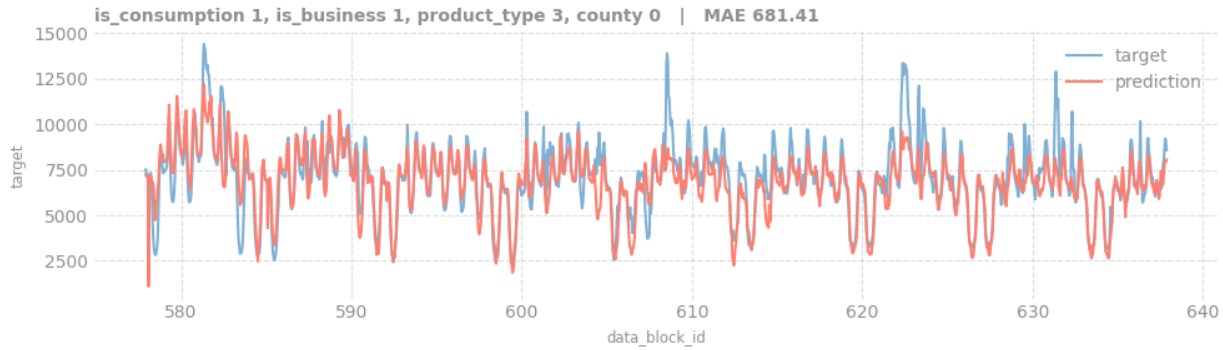


Examples of out-of-fold predictions with the worst MAE scores









6. Unfinished work

Neural Network solution:

+ Possible to predict the whole `data_block_id` as a sequence.

    It might prove beneficial to treat the 24 datapoints of a single time series as a unit and predict them simultaneously.

- Requires normalization of the target.

    As the goal is to forecast a multivariate time series which is not stationary, normalization is not a trivial task for this problem.