**Introduction**

At present, Enefit is attempting to solve the imbalance problem by developing internal predictive models and relying on third-party forecasts. However, these methods have proven to be insufficient due to their low accuracy in forecasting the energy behavior of prosumers. The shortcomings of these current methods lie in their inability to accurately account for the **wide range of variables** that influence prosumer behavior, leading to high imbalance costs.

**Problem Statement**

Enefit is currently facing a significant challenge in managing the imbalance costs arising from inaccurate forecasts of prosumer energy behavior. The existing methods, which include internal predictive models and third-party forecasts, have not achieved a satisfactory level of accuracy. This is primarily due to their inability to effectively incorporate the complex range of variables that influence prosumer behavior. The inadequacy of these forecasting methods has resulted in high imbalance costs for the company. The goal is to develop a more reliable predictive model that can better anticipate the energy behavior of prosumers and reduce the associated financial impacts.

**1. Day-ahead pricing in energy markets**

Day-ahead pricing is a prevalent mechanism within energy markets, such as those for electricity, natural gas, and various other commodities. This system involves establishing prices 24 hours before they take effect.

For example, on January 11, 2024, the market would announce the prices that are to be in effect for January 12, 2024. It is important to note the distinction in how these resources are priced: natural gas prices are set on a daily basis, whereas electricity prices are calculated for each hour of the day.

**2. Understanding the timestamps in datasets**

The datetime entry in a dataset, regardless of its actual label, consistently marks the beginning of a one-hour interval. In contrast, weather-related datasets capture certain measurements—such as temperature or cloudiness—at a specific point in time, which corresponds to the conclusion of that one-hour window.

To put this into perspective with examples:

In datasets concerning energy:

A timestamp of 14:00 indicates that the listed energy prices are applicable for the duration from 14:00 to 15:00.

In datasets related to weather:

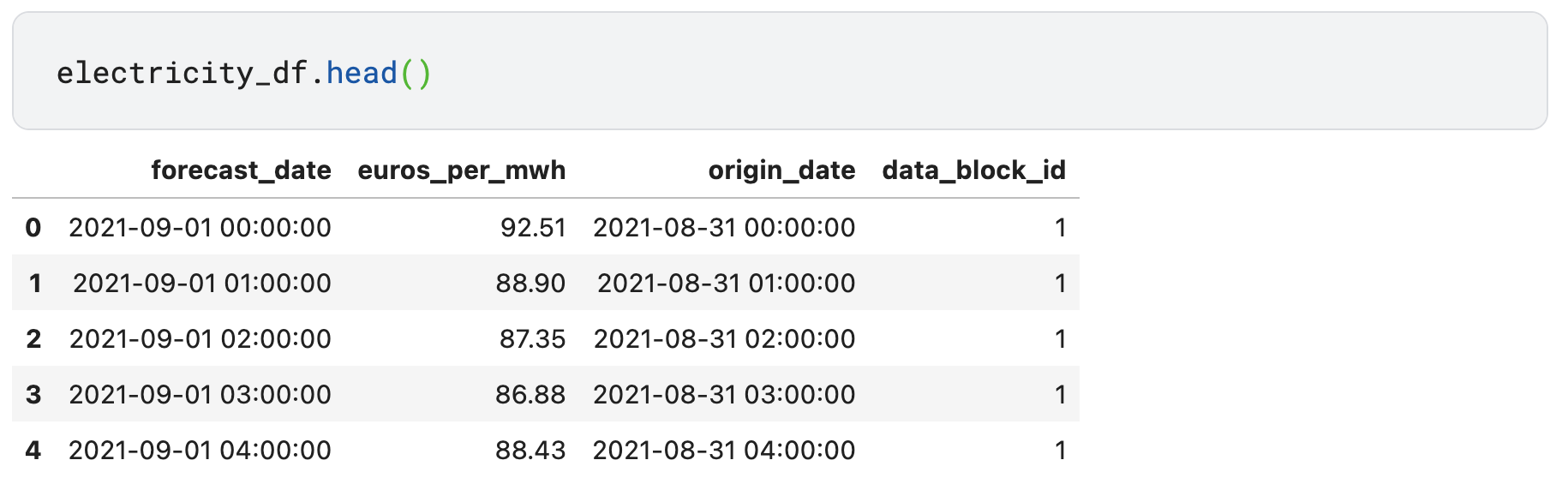
A timestamp of 15:00 signifies that the weather metrics, like temperature, are observed exactly at 15:00, effectively marking the close of the interval that started at 14:00 and ended at 15:00.

**Data Quality Assessment**

Please see data analysis details in the jupyter notebook: https://www.kaggle.com/code/jackren000/predict-energy-behavior-of-prosumers-dataanalysis

electricity\_prices.csv

* origin\_date - The date when forecasting occurs
* forecast\_date - Represents the start of the 1-hour period when the price is valid
* euros\_per\_mwh - The price of electricity on the day ahead markets in euros per megawatt hour.
* data\_block\_id - All rows sharing the same data\_block\_id will be available at the same forecast time.



*################## electricity*

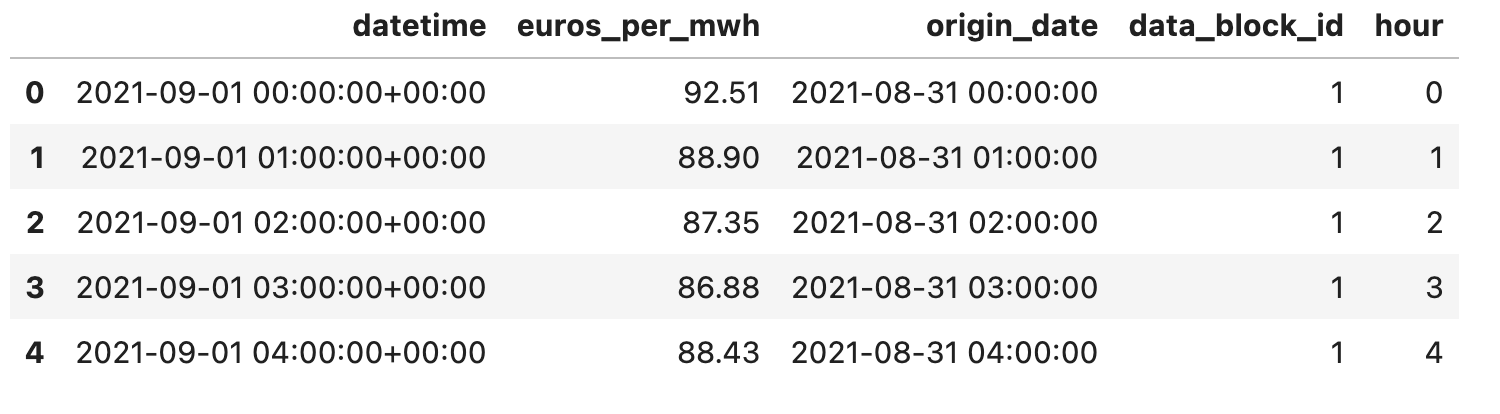
*# rename 'forecast\_date' column to 'datetime' for consistency before merging*

*electricity\_df = electricity\_df.rename(columns={'forecast\_date': 'datetime'})*

*# convert datetime to UTC*

*electricity\_df['datetime'] = pd.to\_datetime(electricity\_df['datetime'], utc=True)*

*# add hour column*

*electricity\_df['hour'] = electricity\_df['datetime'].dt.hour*

Note: add hour column, the price is assigned to forecast\_date, thus rename it into datetime for consistency.

The pseudocode for electricity\_prices.csv:

*################## The pseudocode of electricity\_price dataset*

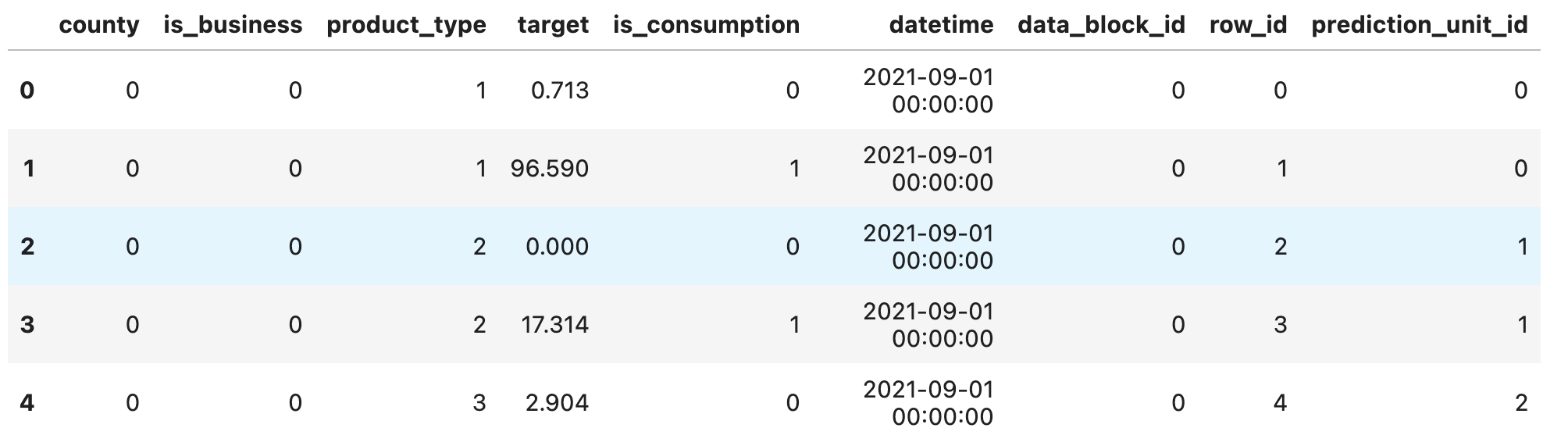
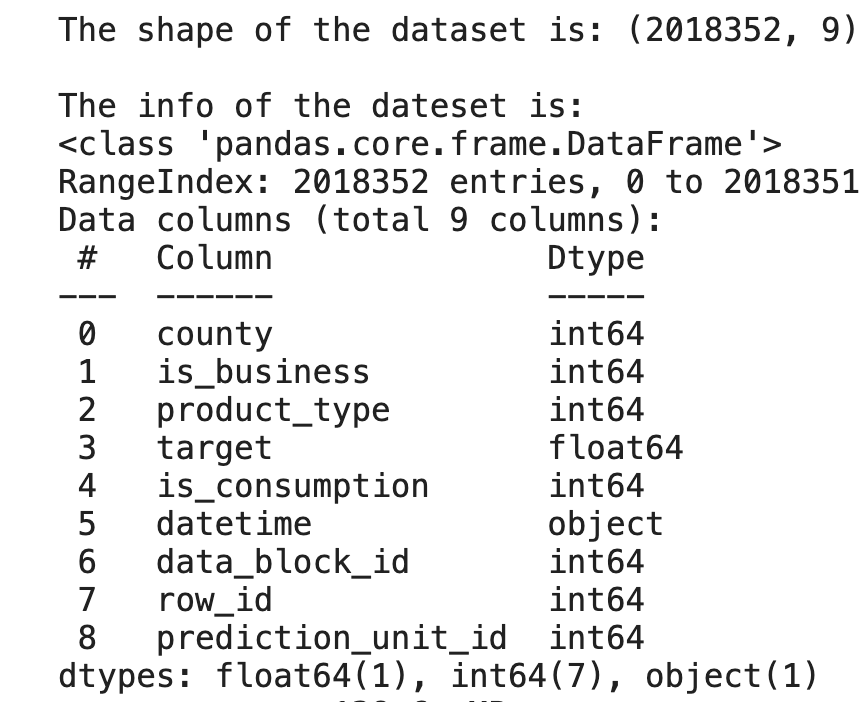
*for year in range(2021, 2024):*

*for month in range(12):*

*for hour in range(24):*

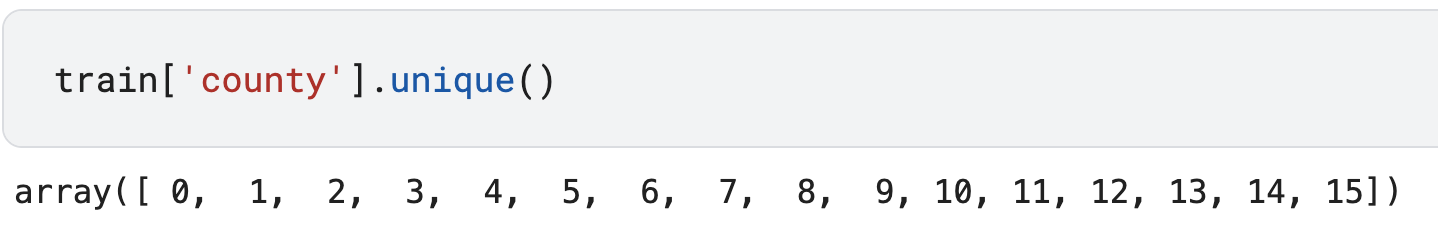
*print(euros\_per\_mwh)*

train.csv

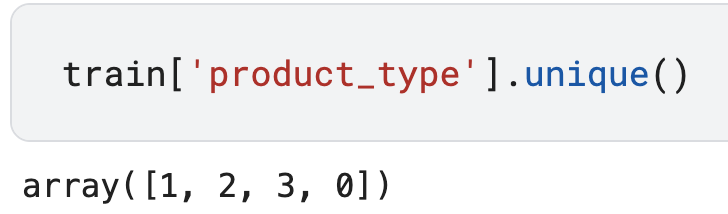


* County - An ID code for the county.

There’re 16 counties inside this dataset.

* 
* Is\_business - boolean for whether or not the prosumer is a business.
* product\_type - ID code with the followin-g contract types:

{0: “combined”, 1: “fixed”, 2: “general service”, 3: “spot”}

* 
* Target - the consumption or production amount for the relevant segment of the hour.

The segments are defined by the county, is\_business, and product\_type.

* Is\_consumption - boolean for whether or not this row’s target is consumption or production.
* Datetime - the estonian time in EET (UTC+2) / EEST (UTC+3).

It describes the start of the 1-hour period on which target is given.

* Data\_block\_id - all rows sharing the same data\_block\_id will be available at the same forecast time. (each number represents a day)
* This is a function of what information is available when forecasts are actually made, at 11 AM
* Row\_id - A unique identifier for the row.
* prediction\_unit\_id - A unique identifier for the county, is\_business, and product\_type combination. New prediction units can appear or disappear in the test set.

Note: The pseudocode for train.csv:

*################## The pseudocode of train dataset*

*for year in range(2021, 2024):*

*for month in range(12):*

*for hour in range(24):*

*for county in range(15):*

*for is\_business in range(1):*

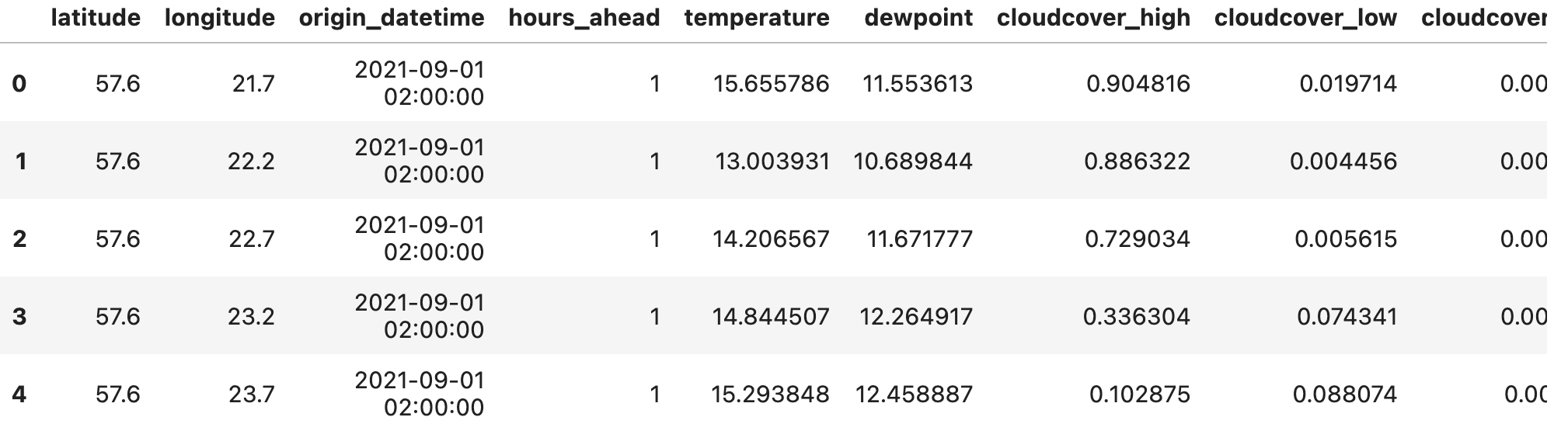
*for product in range(4):*

*print(target)*

forecast\_weather.csv

Weather forecasts that would have been available at prediction time.

* [latitude/longitude] - The coordinates of the weather forecast.
* origin\_datetime - The timestamp of when the forecast was generated.
* hours\_ahead - The number of hours between the forecast generation and the forecast weather. Each forecast covers 48 hours in total.
* temperature - The air temperature at 2 meters above ground in degrees Celsius. Estimated for the end of the 1-hour period.
* dewpoint - The dew point temperature at 2 meters above ground in degrees Celsius. Estimated for the end of the 1-hour period.
* cloudcover\_[low/mid/high/total] - The percentage of the sky covered by clouds in the following altitude bands: 0-2 km, 2-6, 6+, and total. Estimated for the end of the 1-hour period.
* 10\_metre\_[u/v]\_wind\_component - The [eastward/northward] component of wind speed measured 10 meters above surface in meters per second. Estimated for the end of the 1-hour period.
* data\_block\_id
* forecast\_datetime - The timestamp of the predicted weather. Generated from origin\_datetime plus hours\_ahead. This represents the start of the 1-hour period for which weather data are forecasted.
* direct\_solar\_radiation - The direct solar radiation reaching the surface on a plane perpendicular to the direction of the Sun accumulated during the hour, in watt-hours per square meter.
* surface\_solar\_radiation\_downwards - The solar radiation, both direct and diffuse, that reaches a horizontal plane at the surface of the Earth, accumulated during the hour, in watt-hours per square meter.
* snowfall - Snowfall over hour in units of meters of water equivalent.
* total\_precipitation - The accumulated liquid, comprising rain and snow that falls on Earth's surface over the described hour, in units of meters.



Note:

Our dataset contains numerous dates, and our objective is to determine the mean value for each hour across all these dates.

1. fw\_df\_datetime holds the computed average values for each hour, aggregated over all the dates.
2. fw\_df\_datetime includes the hourly average values, broken down by county, for all the dates in the dataset.

historical\_weather.csv

* datetime - This represents the start of the 1-hour period for which weather data are measured.
* temperature - Measured at the end of the 1-hour period.
* dewpoint - Measured at the end of the 1-hour period.
* rain - Different from the forecast conventions. The rain from large scale weather systems of the hour in millimeters.
* snowfall - Different from the forecast conventions. Snowfall over the hour in centimeters.
* surface\_pressure - The air pressure at surface in hectopascals.
* cloudcover\_[low/mid/high/total] - Different from the forecast conventions. Cloud cover at 0-3 km, 3-8, 8+, and total.
* windspeed\_10m - Different from the forecast conventions. The wind speed at 10 meters above ground in meters per second.
* winddirection\_10m - Different from the forecast conventions. The wind direction at 10 meters above ground in degrees.
* shortwave\_radiation - Different from the forecast conventions. The global horizontal irradiation in watt-hours per square meter.
* direct\_solar\_radiation
* diffuse\_radiation - Different from the forecast conventions. The diffuse solar irradiation in watt-hours per square meter.
* [latitude/longitude] - The coordinates of the weather station.
* data\_block\_id

### **Unique Columns in Each Dataset:**

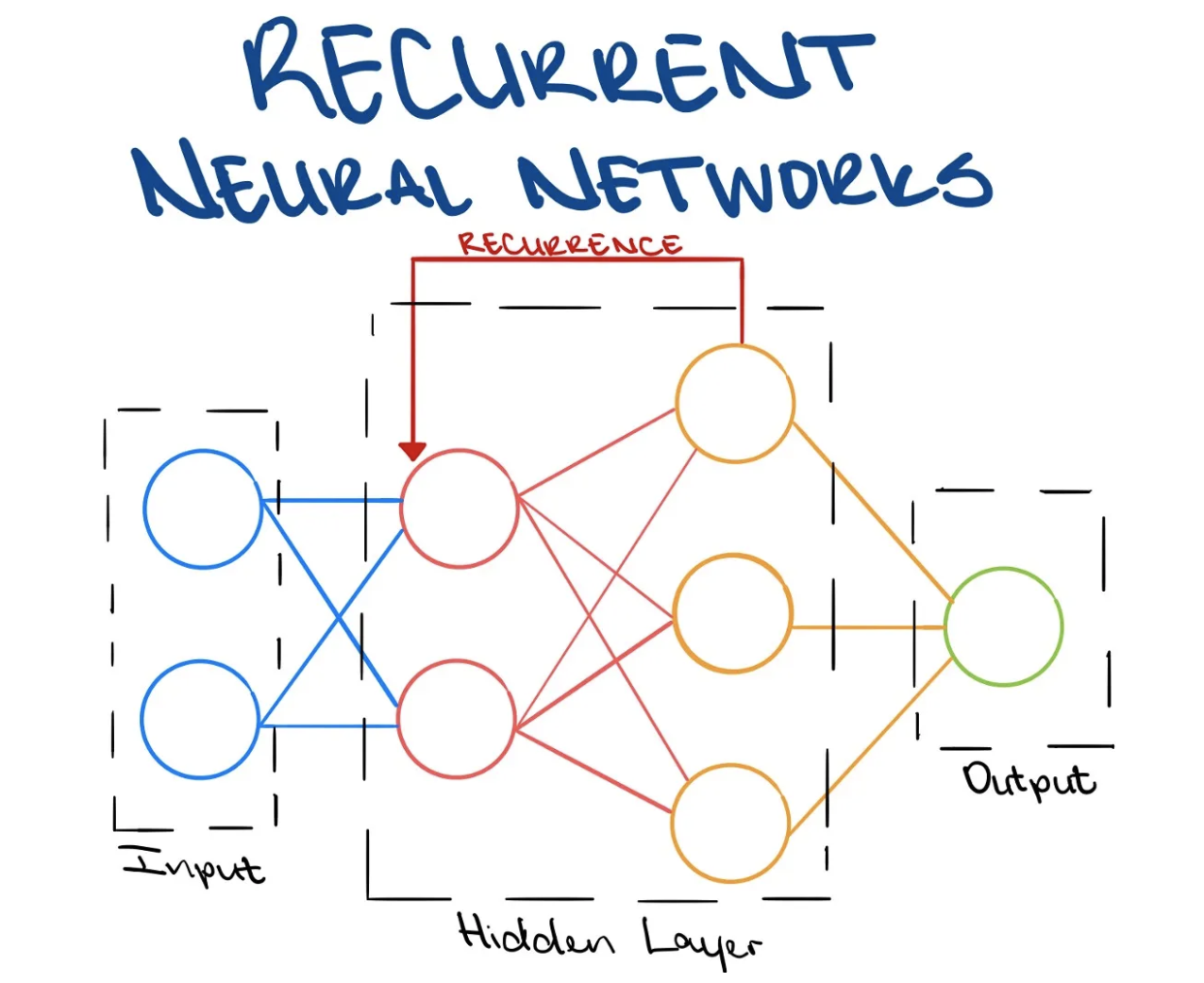
| Unique Forecast Weather Fields | Description |
| --- | --- |
| origin\_datetime | Timestamp indicating when the weather forecast was made |
| forecast\_datetime | Timestamp for the predicted weather conditions |
| hours\_ahead | Number of hours between the forecast generation and the forecasted conditions |
| 10\_metre\_u\_wind\_component | Eastward wind speed component at 10m above surface |
| 10\_metre\_v\_wind\_component | Northward wind speed component at 10m above surface |
| surface\_solar\_radiation\_downwards | Downwards surface solar radiation accumulated during the hour |
| total\_precipitation | Accumulated amount of liquid precipitation over the hour |

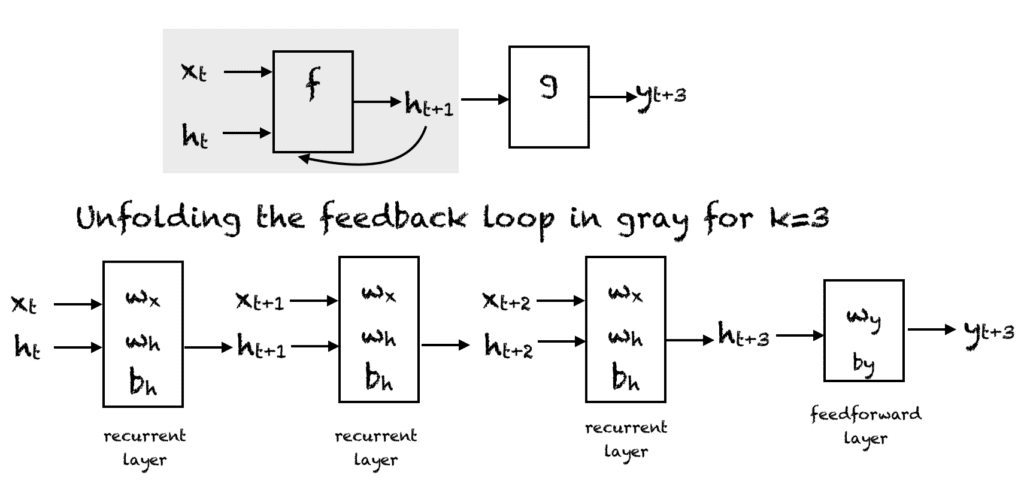
| Unique Historical Weather Fields | Description |
| --- | --- |
| datetime | Actual time when the weather conditions were observed |
| rain | Amount of rain measured over the hour |
| surface\_pressure | Air pressure measured at the surface |
| windspeed\_10m | Wind speed measured at 10 meters above ground level |
| winddirection\_10m | Wind direction measured at 10 meters above ground level |
| shortwave\_radiation | Measured global horizontal irradiation |
| diffuse\_radiation | Measured diffuse solar irradiation |

**Model Building**

Please see building details in the jupyter notebook: <https://www.kaggle.com/code/jackren000/lstm-predict-energy-behavior-of-prosumers>

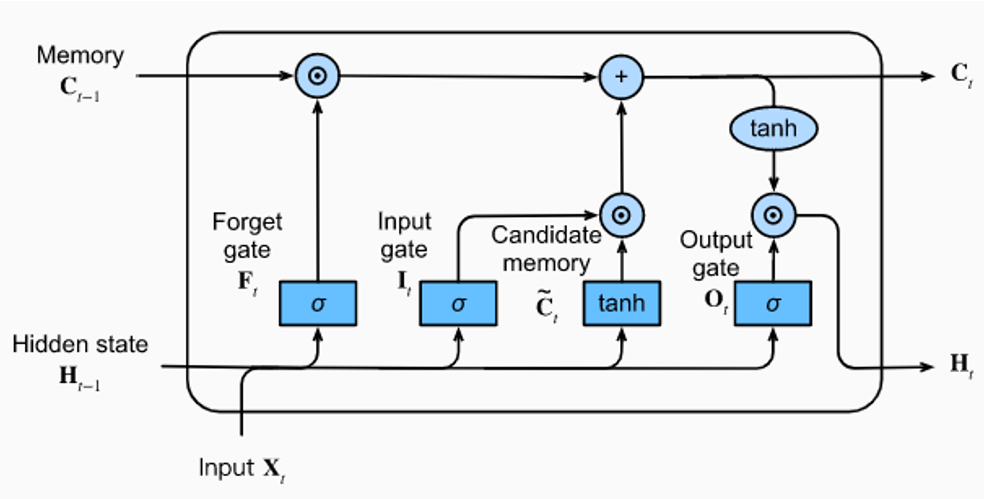
RNN:

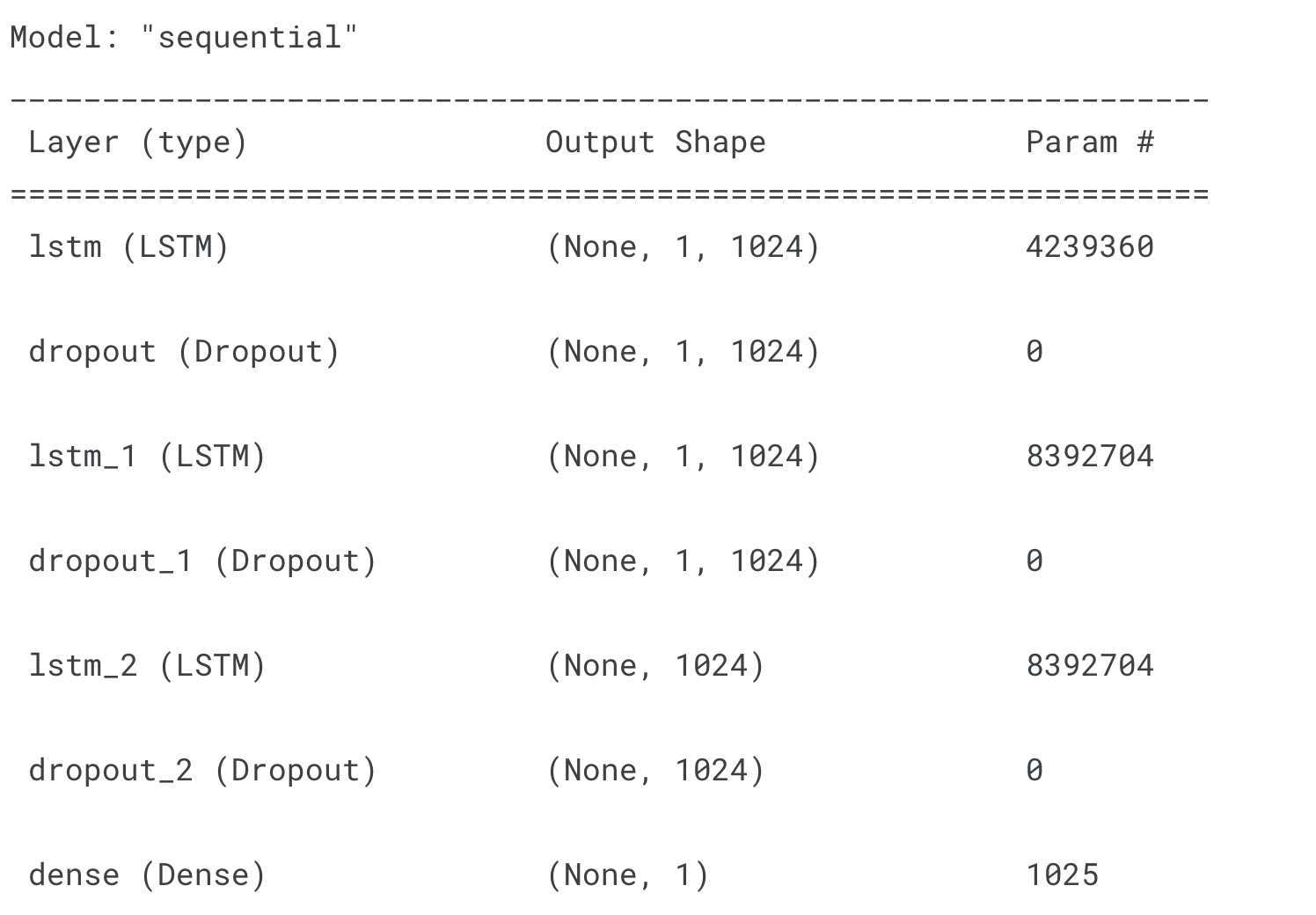




An RNN (Recurrent Neural Network) is a type of artificial neural network designed to recognize patterns in sequences of data such as text, genomes, handwriting and numerical time series data. Unlike feedforward neural networks, RNNs have connections that form directed cycles, which means they can maintain a sort of ‘memory’ of previous inputs in the network’s internal state, which influences the network outputs.

LSTM:

  
LSTM (Long Short-Term Memory) is a type of Recurrent Neural Network that’s designed to remember information for long periods, which is something standard RNNs can struggle with. LSTMs are especially good at making predictions based on sequences of data, like predicting the next word in a sentence or forecasting time series data.



Model 1 :

Train columns: county, is\_business, product\_type, target, is\_consumption, prediction\_unit\_id, month, hour, dayofweek, datetime

MAE: 73.53

Model 2:

Train columns: county, is\_business, product\_type, target, is\_consumption,

Prediction\_unit\_id, month, hour, dayofweek, dayofyear, datetime

MAE: 74.15

Model 3:

Train columns: county, is\_business, product\_type, target, is\_consumption,

Prediction\_unit\_id, month, hour, dayofweek, dayofyear, installed\_capacity, datetime

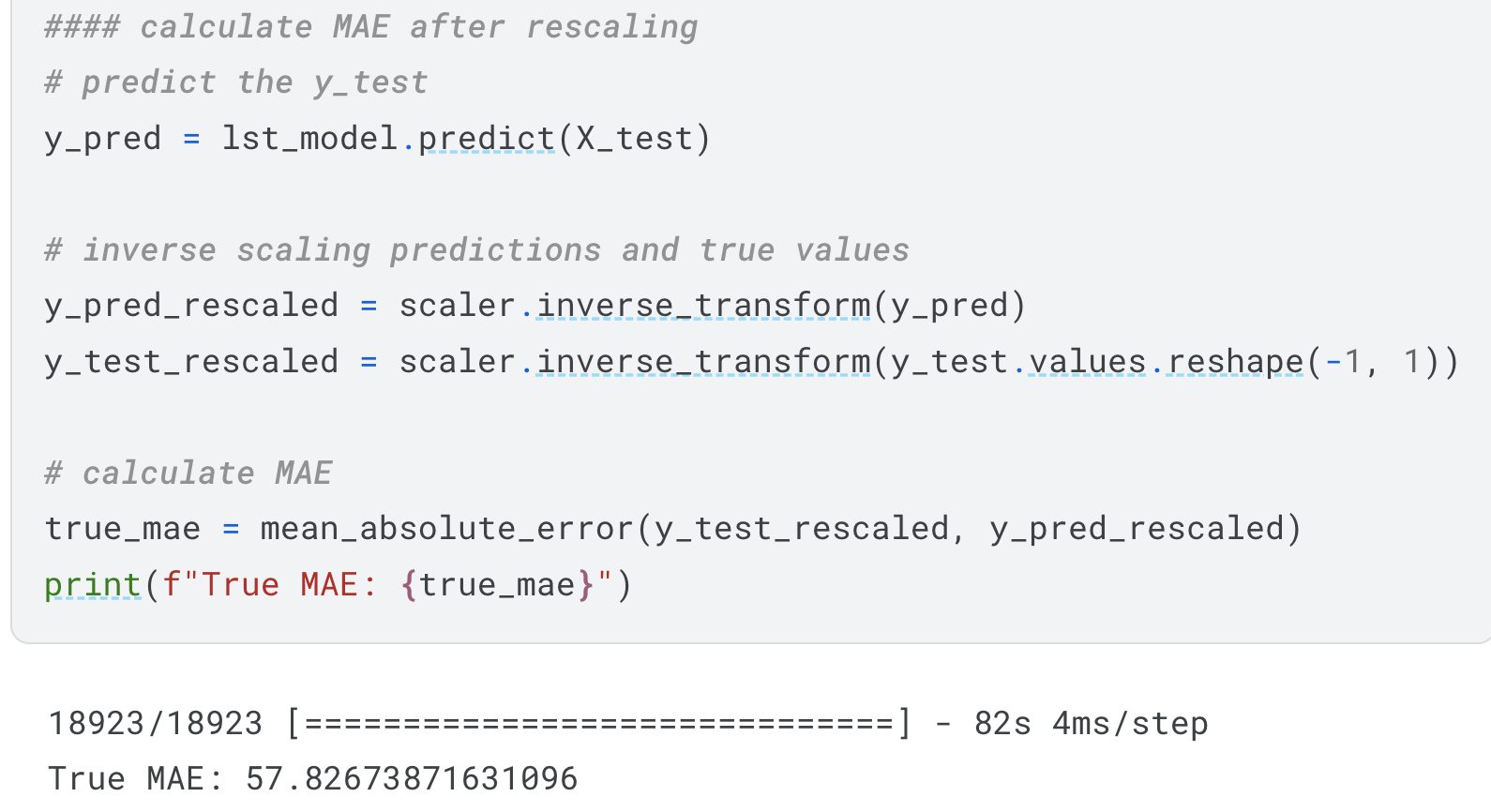
MAE: 58.21

Model 4:

Train columns: county, is\_business, product\_type, target, is\_consumption,

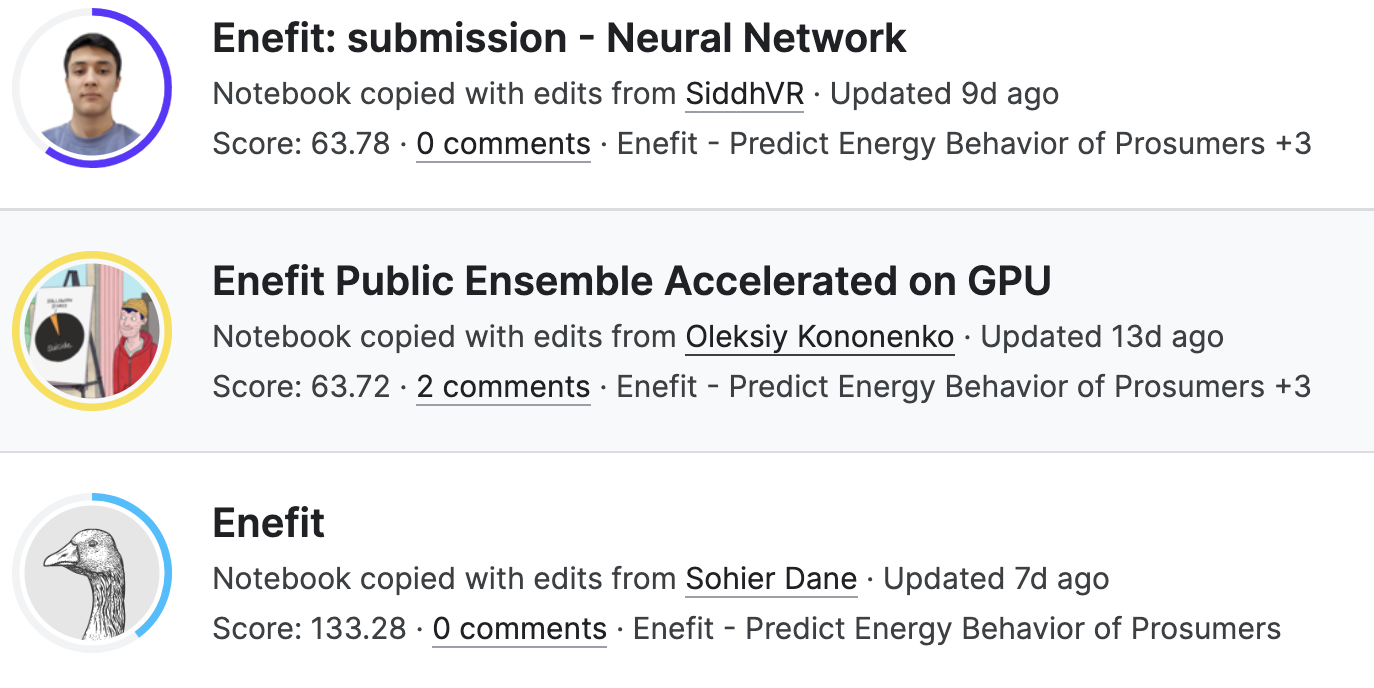
Prediction\_unit\_id, month, hour, dayofweek, dayofyear, installed\_capacity, eic\_count, datetime

MAE: 58.84



**Result**

The outcome is favorable, with a Mean Absolute Error (MAE) of 57.83, which is pretty good when compared to alternative results.

.