Smarter Mobility Data Challenge for a Greener Future

Team *Adorable Interns* : **Nathan Doumèche (MerlinLouis)** **Alexis Thomas (Elial)**
École des Mines de Paris, Université PSL
{nathan.doumeche, alexis.thomas}@minesparis.psl.eu

The aim of this work is to present the method used by the *Adorable Interns* team in the *Smarter Mobility Data Challenge for a Greener Future* organized by the network *Manifeste IA*. Our code and a step-by-step notebook are openly available on GitLab[1] and on GitHub[2].

1 Problem analysis

Context The problem consists in forecasting the states of 91 electric vehicles (EV) stations in Paris within the period T_{test} running from 2021-02-19 to 2021-03-10 with a 15-minute step. Each station contains three plugs and each plug can take one of the states a (available), c (charging), p (passive), or o (other). The state of station $k \in \{1, \dots, 91\}$ at time $t \in T_{test}$ is $y_{t,k} = (a_{t,k},\ c_{t,k},\ p_{t,k},\ o_{t,k}) \in \mathbb{N}^4$ with $a_{t,k} + c_{t,k} + p_{t,k} + o_{t,k} = 3$. We aim at forecasting three groups of times series with a spatial hierarchy. The first level regroups the individual stations $y_{t,k}$. The second one aggregates the stations into the four areas N (north), S (south), E (east), and W (west). We write $A_{t,area} = \sum_{k \in area} y_{t,k}$. The last one is the global level $G_t = \sum_{area} A_{t,area} = \sum_k y_{t,k}$. We give equal importance to all levels. Thus, we want to construct an estimator $\hat{z}$ of the stochastic process $z_t = (y_{t,k},\ A_{t,area},\ G_t)_{1 \le k \le 91,\ area \in \{N,S,E,W\}}$ minimizing the expected test loss $\mathcal{L}(\hat{z}) = \sum_{t \in T_{test}} \mathbb{E}(\|\hat{z}_t - z_t\|_1)$, with $\|\cdot\|_1$ being the L^1 norm.

Data description To this end, we have at hand the data z_t for t in the period T_{data} running from 2020-07-03 to 2021-02-18 with a 15-minute step. We can also use the calendar information (time of day tod, day of week dow, and $trend$) and the station location ($latitude$, $longitude$ and station $area$). The use of external data sets was forbidden, though weather data has been shown to enhance EV load models (Amara-Ouali et al., 2021) and is openly available at https://donneespubliques.meteofrance.fr/.

Data analysis Several challenges arise from the data at hand[3]. First of all, there are a lot of missing values in the data set, whereas most of the machine learning models struggle to handle missing data. An usual way to deal with it would be to filter out the dates or the stations which gather most of the missing data (Hastie et al., 2017), but this is impossible because missing values are spread among the stations and the dates. Another interesting phenomenon at stake is the happening of a change in the data distribution on 2020-10-22. Missing data starts to appear regularly right after this date. We think that this is due to an update in the software communicating with the stations which then starts to detect when stations are malfunctioning. In fact, the stations presenting missing values are the ones which were stuck before the change in either a, i.e. waiting for a car, or o, corresponding to maintenance. These are the two states where no car is parked in the station. Perhaps the malfunctioning stations were avoided by the users, or perhaps the users did try to plug to the station but the plug was unresponsive so the users were undetected. Another challenge regarding the data set was its shortness. Indeed, we expect a yearly seasonal effect due to holiday that cannot be distinguished from a potential trend because we have less than a year of data. Such a trend for the time series z_t could be caused by social and economic factors, such as political incentives to drive EV or Covid-19 restrictions which were shown to strongly impact human mobility (Pullano et al., 2020).

[1] https://gitlab.com/alexis.thomasjutisz/Smart_mobility_challenge
[2] https://github.com/NathanDoumeche/Smart_mobility_challenge
[3] See section 1 of the notebook.

2 Algorithms and method

Models As usual in the supervised learning setting, we need to choose a model $\mathcal{F}$ to construct the estimator $\hat{z}_t = (\hat{y}_{t,k}, \hat{A}_{t,area}, \hat{G}_t) \in \mathcal{F}$. To estimate the full T_{test} period at once, we cannot use online models such as autoregressive models or hidden-state neural networks (RNN, LTSM, transformers...), though they perform well on time series forecasting (Bryan and Stefan, 2021).

Empirical loss Once chosen a model $\mathcal{F}$, we define an empirical loss L on the training data. Then, we use a learning procedure, such as a gradient descent, to find the estimator $\hat{z}$ minimizing L, hoping that this estimator will minimize the expected test loss $\mathcal{L}$ (Hastie et al., 2017). We consider two empirical losses. The first one $L_{equal}(\hat{z}) = \sum_{t \in T_{train}} \|z_t - \hat{z}_t\|_1$ gives equal importance to all data points. The second one $L_{exp}(\hat{z}) = \sum_{t \in T_{train}} \exp((t - t_{max})/\tau)\|z_t - \hat{z}_t\|_1$ with $\tau = 30$ days and $t_{max} =$ 2021-02-19 00:00:00, gives more credit to the most recent observations. This makes it possible to give more weight to the data after the change in the data distribution and to capture the last effect of the trend, while using as much data as possible.

Benchmark phase To compare the performances of the models[4], we define a training period T_{train} covering the first 95% of T_{data}, and a validation period T_{val} covering the last 5%. Models are trained on T_{train} with an empirical loss and their performances are evaluated on T_{val} by $L_{val}(\hat{z}) = \sum_{t \in T_{val}} \|z_t - \hat{z}_t\|_1$. The $Mean$ model estimates $\hat{y}_{t,k}$, $\hat{A}_{t,k}$ and $\hat{G}_t$ by their mean on the training period for each value of (tod, dow). Idem for the $Median$ model. They are robust to missing values since the malfunctioning of a station k only impacts $\hat{y}_{t,k}$. We compare them with a tree-based gradient boosting algorithm specialised in categorical regression called catboost. We call $C(d, i)$ the catboost model of depth d trained with i iterations with L_{equal}, and $C_{exp}(d, i)$ the same model trained with L_{exp}. In this setting, we train twelve catboost models: one for each pair of state (a, c, p, o) and hierarchical level. After hyperparameter tuning, we found $C(4, 150)$ and $C_{exp}(5, 200)$ to be the best models in terms of trade-off between performances and number of parameters (leading to overfitting). These models take advantage of the fact that malfunctioning stations tend to stay in specific states.

Validation phase The organizers of the contest allowed participants to test their models on a subset T_{val} of T_{test}. In this phase, we trained our best models on the whole T_{data} period and sent them to be tested by L_{val}. Figure 1 shows that the hierarchy of the models is preserved.

Model $\mathcal{F}$	Mean	Median	$C(4, 150)$	$C_{exp}(5, 200)$
Benchmark Phase	316	309	292	261
Validation Phase	323	303	233	189

Figure 1: Evaluation of the performances of our models in both phases

Submitted model The model submitted[5] was therefore the $C_{exp}(5, 200)$. This model is interesting for industry because its very low number of parameters ensures robustness and scalability, and because tree-based models are quite interpretable which is paramount for operational uses.

References

Y. Amara-Ouali, Y. Goude, P. Massart, J.-M. Poggi, and H. Yan. A review of electric vehicle load open data and models. Energies, 14:2233, 2021.

L. Bryan and Z. Stefan. Time-series forecasting with deep learning: A survey. Philosophical Transactions of the Royal Society A, 379:20200209, 2021.

T. Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning. Springer Series in Statistics. Springer New York, New York, 2nd edition, 2017.

G. Pullano, E. Valdano, N. Scarpa, S. Rubrichi, and V. Colizza. Evaluating the effect of demographic factors, socioeconomic factors, and risk aversion on mobility during the covid-19 epidemic in france under lockdown: a population-based study. The Lancet Digital Health, 2:e638–e649, 2020.

[4] See section 2 of the notebook.

[5] See section 3 of the notebook. This is the model implemented in the file *main.py*.