Arthur SATOUF
AI engineer student at CY-TECH School

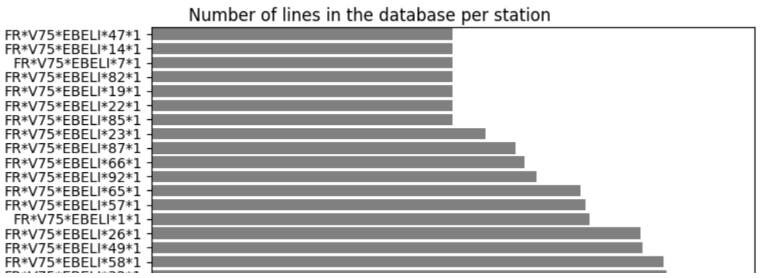
Smarter Mobility Data Challenge

Abstract

The key for this challenge was to ask yourself as many questions as you can imagine. It’s a bit tricky as a competition, as there are 3 hierarchical levels (“Stations”, “Area”, “Global”) and I found that it is not about finding the best model, it is more about making the best strategy to prepare your data to be modeled with a model that is recommended for multi long future steps. After trying and testing several techniques, it turns out it would be best to consider the 3 levels of data sets independently of each other. We would treat them differently especially for filling the missing data and we should apply a different model for each set. A few of my inspiration questions were: How the data sets were exactly built for “Area” and “global”, where and what missing data do we need to recover and how to provide an output simple enough for this stochastic phenomena without any correction for the output manually. In addition, it was very important to connect the targets “Availble”,“Charging”,“pasive” & “other”. I have used CatBoost for the 3 levels. I have filled in a few missing data for Stations with Exponential Moving Average (EMW) with 8 backward and 8 forward and I have filled in all the missing data for “global” and “area”.

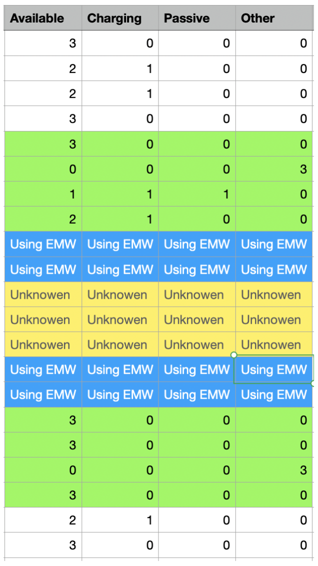
Cleaning and filling the missing values.

After creating some visualization and using “Data Viz” that has been provided by the organizers, I realized that the training data set of “Stations” is missing 166933 rows, which is a lot of data. This is not bad news for “Station” but it is a big issue for “Area” and “Global” because to get “Global” or “Area” you have to sum over stations grouped by a given time and date(and area for “Area”). To solve this we should do the following :



- Generate with Pandas dates from 2020-07-03 00:00:00 to 2021-02-18 23:45:00 per minute and keep only every 15 minutes as the frequency with all stations names.
- Fill 'tod', 'dow', 'trend', 'Latitude', 'Longitude', 'Postcode' and 'area' according to the “data train of station” secondly try the following 5 techniques and do test_train_split using MAE metrics to find the best techniques to fill in our data.
  - Backward fill
  - Forward fill
  - Mean
  - Rolling moving windows mean with ***span*** **=** [2, 12]
  - And Exponential Moving Window (EMW) with ***span*** **=** [2, 12] (backward and forward filling)
- And finally removing or replacing the odd/abnormal values.

For “Stations” model “train_onlyNext4EWM4andBack_EWM_remChar.csv”: By using EMW(4) forward I filled in only the 8 next values of missing data and EMW(4) backward to fill the 8 previous missing data(figure on the right I only did 2 instead of 8) plus I don't need to fill all missing date because we don't really know what really happened after a given time and there are enough data to make a good model. In addition, for a given date and time I would remove all rows that are exactly the same if “Available” ,“charging” ,“passive” or “Other” for all the stations that have the same value for one target. For example at exactly 2020-07-03-17h all the stations that have the same value for charging which is equal to 0,would be removed because after analyzing the data it can be seen that they are quite rare. And round the targets using np.rint() and remove all data that the sum of the target is bigger or smaller than 3.



Available	Charging	Passive	Other
3	0	0	0
2	1	0	0
2	1	0	0
3	0	0	0
3	0	0	0
0	0	0	3
1	1	1	0
2	1	0	0
Using EMW	Using EMW	Using EMW	Using EMW
Using EMW	Using EMW	Using EMW	Using EMW
Unknowen	Unknowen	Unknowen	Unknowen
Unknowen	Unknowen	Unknowen	Unknowen
Unknowen	Unknowen	Unknowen	Unknowen
Using EMW	Using EMW	Using EMW	Using EMW
Using EMW	Using EMW	Using EMW	Using EMW
3	0	0	0
3	0	0	0
0	0	0	3
3	0	0	0
2	1	0	0
3	0	0	0

For “Area” model “data_ewm1.2.csv” : By filling all data with EMW(4) forward. We need all data, in able to sum the stations and regroup the Area. So no missing or unknown data at all

For “global” model “remCharEWM4” : Same as for Area model but I replaced abnormal data of charging (==) by EMW(4) forward.

Modeling with CatBoost

For “Stations”, I’ve created a variable called “labels” which is the sum of targets in string type(example: “3000”, “2100”, “1110”...) so that the sum of the targets will always be 3 as we have only 20 unique combinations of “labels” we can treat it easily as problem as a classification problem with `CatBoostClassifier` otherwise if treated as regression will not always have the sum equal to 3. And using the following inputs `["Station" ,"area" ,"hour", "dow", "tod", "trend"] cat features index = [0,1,2,3].`

For “Area”, I forecast each area independently because the sum of each row are not equal from area to another area and secondly I tested `RegressorChain` from `sklearn.multioutput` and it gave a good result (the concept is we predict one target then we use it as input to predict the next target and so on) so we used `CatBoostReg` so we end with `4(area) * 4(models) = 16` models so that we would prevent having a bigger or smaller sum than the right sum number of the sum of the targets for each Area.

For “global”, I used a regular `CatBoostRegressor` with `col_act= ["hour" ,"tod", "min", "dow", "trend"]` and `cat_features_index = [0,3].`

All together, I have tested many algorithms like fully connected NN, LSTM, 2 Transformers, S-ARIMA-X, XGboost, Lightgbm and Random forest. It turns out that CatBoost is the fastest algorithm and most accurate to forecast many steps ahead in the future. Then of course I have done some grid_search to find the best hyperparameters using GPU on Collab. I split my data into Train-Test sets and treated and tested the 3 levels separately. In each submission I only focused on one level to test. Note: I believe for input data for “Area” and “Global” it would be the same but i did not have enough time to test that.