III. Model documentation and write-up

You can respond to these questions either in an e-mail or as an attached file (any common document format is acceptable such as plain text, PDF, DOCX, etc.) **Please number your responses.**

1. Who are you (mini-bio) and what do you do professionally?

Graduated in 2016 from Polytech Lille in Software Engineering and Statistics. Currently moving from a DataScientist role in a consulting firm to a more machine learning oriented role in an online marketplace.

2. High level summary of your approach: what did you do and why?

I do not predict directly the value but value / value_mean where value_mean is the mean value of the building during the Tn last available timestamp. With Tn the size of the timestamp forecast (the number of timestamps that we are calculating the metric over for this forecast n). It is a "multiplicative model"
Match each timestamp to be predicted to the closests:

a. Last available values
b. Last available values on (time)
c. Last available values on (time, day of week)
d. Last available values (time, close) (with the same business day (close/open))

Compare weather to the above matching to the timestamp to be predicted.
All values must also be normalized.
Each timestamp can be predicted from different t (with 0 < t <= Tn), therefore each timestamp generate up to Tn learning points. It generates more than one billions points which I couldn't handle locally. Therefore I sampled around X millions rows (therefore my models are trained only on X% of the possible learning data) them given their weight in the metrics.
Hierarchical modeling: a model is trained by Building
Simple average of 3 LightGbm models with different set of parameters (number of leaves)
Time series related training process: train and validate models with time split in order to avoid any leak (in my case determine the number of iterations), then re-train on full data before predicting
In the end, there is 6 models trained (3 set of parameters, once on train/valid, once on full data) on all 300 Building/SiteId. It takes around 2? hours to train.

3. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

Rolling joins thanks to data.table package. Eg :I assign to the learning dataset (the final dataset used for training) the last past available feature engineered set based on last closed (business days) on keys (SiteId, Forecastid, time, closed, Timestamp_last_day).

Timestamp_last_day is the rolling join. Learn more about rolling joins on this great blog post.

```
setkeyv(fe_backward_last_closed, c("SiteId", "ForecastId", "time", "closed", "Timestamp_last_day"))
setkeyv(df_learning, c("SiteId", "ForecastId", "time", "closed", "Timestamp_last_day"))
df_learning = fe_backward_last_closed[df_learning, roll=T]
```

Same but not on the last available timestamp value but the nearest one.

```
setkeyv(weather, c("SiteId", "Timestamp_lead"))
setkeyv(df_learning, c("SiteId", "Timestamp_lead"))
df_learning = weather[df_learning, roll="nearest"]
```

Smart sampling maybe

Fast prototyping

Exemple de FE

```
fe_backward_last_day[ , c("value_time_mean_7", "value_time_mean_3",
                          "temperature_time_mean_7", "temperature_time_mean_3",
                          "value_4_time_mean_7", "value_4_time_mean_3") :=
                        list(roll_mean(Value, n = 7, align = "right", fill = NA, na.rm = T),
                             roll_mean(Value, n = 3, align = "right", fill = NA, na.rm = T),
                             roll_mean(Temperature, n = 7, align = "right", fill = NA, na.rm = T),
                             roll_mean(Temperature, n = 3, align = "right", fill = NA, na.rm = T),
                             roll_mean(value_4, n = 7, align = "right", fill = NA, na.rm = T),
                             roll_mean(value_4, n = 3, align = "right", fill = NA, na.rm = T)),
                      by = c("ForecastId", "time")] %>%
  setnames(c("Value", "Temperature", "value_4", "Timestamp"),
           c("value_time_mean_1", "temperature_time_mean_1", "value_4_time_mean_1", "Timestamp_last_day")) %>%
  .[, c("time", "size", "timestep_minutes", "step_day", "step_hour") := NULL]
```

4. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

   Not sure if it worked on not, and in this end I decided to be more conservative and to not use the following feature engineering.
   Replace day of week by :
   a. If holiday, the first closed day of week
   b. If holiday next day, the last opened day of week before closing
   c. If holiday previous day, the first opened day of week before closing

5. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

   I only used R with Rstudio.

6. How did you evaluate performance of the model other than the provided metric, if at all?

   I tried to quickly have a local metric correlated to the provided one. Getting a robust local validation scheme is needed in machine learning competition, however since the metrics was unknown to me, not easy to replicate, and that I was short on time, I had a first local metrics version, then prioritized feature engineering before updating it later in the competition. At the end, I accepted my okayish and not perfect local validation metrics, then improved it during competition. I prioritized feature engineering but ideally, having at the beginning of the competition the perfect local validation scheme and not changing it is a must.

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

   I faced memory failed allocation problem with Lightgbm on Ubuntu when tried to set up a config on Google Cloud Platform. But not on my Mac.
   Run all the solution on my Mac with 16 Go RAM, I however used swap memory.

8. Do you have any useful charts, graphs, or visualizations from the process?

   See Rmarkdown document

9. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

   a. Improve local validation scheme
   b. Simply train on full data and not a small portion of it (adjust observations' weight during model training)
   c. When there is all missing value in a ForecastId, replace the value_mean with the last available from previous Forecastid – and therefore allow model prediction based on time, weather, etc rather than simply imputing the mean of the previous series.
   d. Add a rolling linear trend information
   e. Improve normalization approach (I have the feeling it should be a multiple of week cycle and not the size of forecast)
   f. Improve weather deduplication / choose of data
   g. Mean/sd temperature of previous X hours in each fe_XXX might help a little bit (avoid suddent heat/cold, we might have accumulated previously)