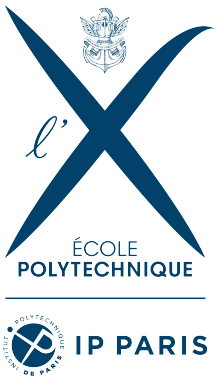


École Polytechnique

Master of Science and Technology, Data Science for Business

MAP 536: Final Report

Authors:

Ghali Chraibi, Tim Valencony

Institution:

Ecole Polytechnique, Institut Polytechnique de Paris, Palaiseau France

Academic year 2023/2024

Contents

1 Introduction

1.1 The Challenge

This project aims at predicting the hourly bicycle traffic in some specific locations in Paris. For that matter, the study is done using a dataset containing hourly measurements of bike counts from 56 counters installed in strategic bike lanes throughout the French capital. The best submission, resulting in the first place on both private and public leaderboard with corresponding RMSEs [3] of 0.4386 and 0.4579, is a **CatBoostRegressor(iterations=2000, depth=11, eval metric='RMSE', learning rate=0.05)** after cleaning the initial training set from defectuous measurements, performing feature engineering to compute columns on date, on location of counters in respect to other transport means, on specific hourly weather data combined with a PCA [8] and lastly fixing any anomaly in the predicted test set. All of the code and work described in this report can be found in this Github repository.

1.2 Data Exploration

Before getting any further in the modelling part, a thorough exploration of the data is conducted with the objective of gathering relevant insights, helping in the data cleaning and feature engineering processes. Each of the $496,827$ records are described by the following 12 variables: *counter id, counter name, site id, site name, bike count, date, counter installation date, coordinates, counter technical id, latitude, longitude, log bike count* being the target variable. Few observations can be drawn after inspecting Table 1.

Feature	*counter id, counter name*	*site id, site name, counter technical id, coordinates, lat, long*	*bike count*
Distinct Modalities	56	30	998
Description	Identification at the lowest level. Two counters for the same *site id* have different behavior compared to the logged-scales and the overall trends. To avoid any information loss, group by counter rather than site.	These features have 1.00 pairwise correlation coefficients. To avoid any noise, and considering *counter id* is more granular, it is recommended to drop them after using them to construct new features.	Average value is 60.2 and 8.3% are zeros. Distinguishing real zeros from anomalies is key to not train the model on faulty measurements. Analysis leads in dropping some observations for both counters of *'20 Avenue de Clichy'* and *'152 Boulevard de Montparnasse'*.[1]

Table 1: Data exploration table, description of the features.

2 The Full Pipeline

2.1 Baseline Tests

First, a point of reference on which to evaluate more complex models is needed, checking their real impact compared to naive implementations. The naive implementations proposed are in Table 2, where the RMSE is tested on a cross validation [2] of the full training set with 5-*fold* time series split, as well as a validation set previously separated from the train set. The baseline is calculated using a naive implementation of predicting the average of the training set as constant output.

It is important to note here that the execution time is calculated on a laptop under macOS Ventura $13.4.1$ with Apple M1 Chip containing 8 cores, $8422.121.1$ system firmware version, and 8GB of RAM.

From those results, one can clearly see that there are two top contenders: *CatBoost* [6] and *XGBoost* [4]. The former is more accurate on the validation set that is split from the train set according to the date (as the data is a time series). The latter is more accurate on the cross validation score. Furthermore, the *CatBoost* model is more computationally heavy. Despite

[1] The deletion of these records in the training dataset permitted us to improve our best submission score on the public leaderboard from 0.6002 to 0.5975 before any reverse engineering

that, *CatBoost* is chosen as it is more suited for the task at hand, since the model is finally evaluated on a test set where the dates are following those found in the train set.

Model	Validation RMSE	5-CV RMSE	Execution Time
Baseline	1.49	N/A	0s
Ridge Regression	0.70	0.858 ± 0.0698	1.67s
Random Forest	0.97	1.05 ± 0.0597	195.02s
Naive LGBM 3	0.60	0.758 ± 0.0582	16.22s
Naive CatBoostRegressor	0.47	0.762 ± 0.0841	50.77s
Naive XGBRegressor	0.49	0.739 ± 0.0621	4.09s

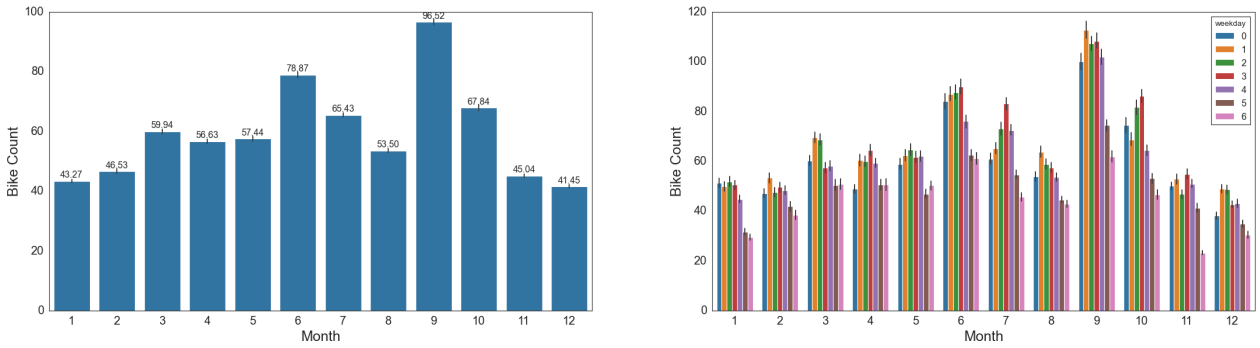
Table 2: Comparison Table of Model Performance

2.2 Feature Engineering

2.2.1 Date encoding to compute holidays and Covid related features: `_encode_date()`

Naturally, the first features engineered are derived from encoding the date column. For visualisation and modelling purposes, the new variables *year, month, day, weekday, hour, week of year* and *season* are treated as categorical.

Figures 1a and 1b bring attention to the influence of the month and weekdays in the bike usage.



(a) (b)

Figure 1: a) Average bike count per month. b) Average bike count per day in the week, per month.

- *Weather and Seasonality*: Plots 1a and 1b show that summer and warmer weather positively impact bike traffic. (82% increase from January to June). The opposite observation can be made for colder weather by noticing a 57% decrease from September to December.
- *Holidays*: Figure 1b shows that summer holidays negatively influence the bike usage trend, as well as weekends. The maximum peak is attained in September, when people get back to work.

Using the *holidays* [5] library, a binary column *'holidays'* is created to capture non-working days including weekends. Lastly, by noticing that records were counted during the Covid pandemic, the column *'france stay at home'* is generated using the *lockdowndates* [10] library. The same observation as for weekends are made but with a slight difference with noticeable peaks from 6 to 8 AM and from 4 to 6 PM for the people still having to work.

2.2.2 Encoding Cyclical Values

Cyclical features, such as the day of the week or the hour of the day, are clearly important [7]. Portraying them in a way that the model understands their cyclicality seems like a good idea at first. Thus, the choice for this study is to encode them using a Fourier [12] approach. Features like *weekday*, *hour*, *month*, *day*, *hour*, *season*, and *weekyear* are encoded using scaled sinus and cosinus functions. Taking as example the hours in a day, the transformation is the following, adding two columns per feature:

$$\cos_{\text{hour}}(h) = \cos(\frac{2\pi h}{H}) \; ; \; \sin_{\text{hour}}(h) = \sin(\frac{2\pi h}{H})$$

where H is the number of hours in a day.

However, at the time this report is written, which is after the final submission, it is acknowledged that leaving the features as is for the model is a better choice. Indeed, considering a decision tree model, encoding the features generally brings more harm than good, as it cannot handle multiple features at a single decision point. The cyclicality can only be grasped in more than two dimensions, and thus at least two columns. This argument is even supported by Table 3 which shows the prediction on the validation set as better without the cyclical data.

Encoding	Validation RMSE Score	5-CV RMSE Score
None	0.43	0.71 ± 0.166
Fourier	0.46	0.637 ± 0.0356

Table 3: Comparison Table of Performance with and without Cyclical Encoding

2.2.3 Counters' proximity to transport means: `_closest_transport() + _count_stations()`

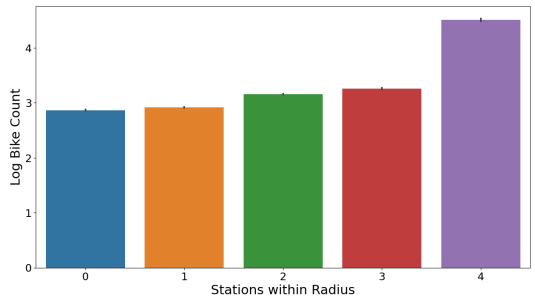


Figure 2: Average log bike count compared to the number of stations with the given radius.

Thanks to a dataset [11] gathering all the coordinates of the transport stations in Paris, two columns are built:

- *'closest metro distance'*: returns the distance to the closest station for each counter.
- *'num stations within radius'*: returns the number of stations within a radius of 400m for each counter. The threshold is arbitrarily set considering that this could be a reasonable walking distance to get to another transport mean.

Per Figure 2, the visualisations show that the more stations in the radius, the more bike usage on average.

2.2.4 Weather encoding: `_add_weather_data()`

The original external dataset provided is not used for the lack of granularity and the inherent approximation if the 3 hours data were to be interpolated to hourly observations. To solve these problems, hourly data is captured per counter thanks to the *meteostat* [9] library, leveraging the available coordinates in the initial dataset.

After doing so, each observation now has 9 new specific variables describing weather data such as temperature, humidity, precipitation, and wind speed. Finally, to avoid any data leakage, all these features are shifted to describe the past hour's weather.

Good practice is to make a summary of all the values received through the *meteostat* library. In order to do this, principal component analysis (PCA) reduces the dimensionality of the data [8] to $k \in \{2, ..., 9\}$ evaluable components. The idea behind this is, even though interpretability is somewhat entirely lost, similar weather states are grasped if their five-dimensional vectors are close in MSE, forming clusters of weather states. This, in turn, is interpreted by the model and seperation values are determined equivalently in the tree. This method is chosen over making the categories by hand as it is easier for the model to comprehend and the thresholds are determined heuristically. The best number $k = 5$ of components is determined by an iterative process.

2.3 Model Tuning

2.3.1 Hyperparameter Tuning

Iterations	Learning rate	Depth	Subsample	Colsample by level	Minimum data in leaf	Grow policy
$\{1,000; 2,000\}$	$(10^{-4}, 0.2)$ (Log scale)	$\{5, ..., 11\}$	$(0.05, 1)$	$(0.05, 1)$	$\{1, ..., 100\}$	['Depthwise', 'Symmetric-Tree']

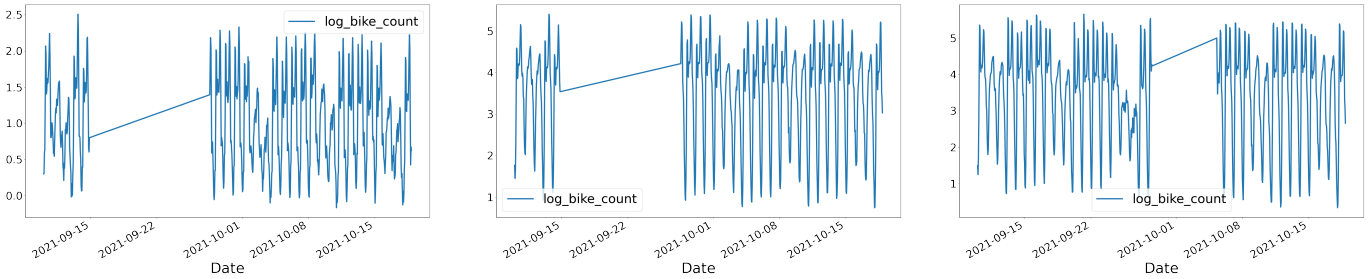
Table 4: Table of the Hyperparameter Layout

Hyperparameter tuning is done using the *optuna* [1] library, with a total number of trials set to 200. Table 4 summarizes the ranges that correspond to each parameter.

The hyperparameter tuning is run on Google Cloud platform for performance and computational issues, as the cost of running it on personal laptops is too great. One should note that the retained hyperparameters on this operation are not all used. When plugging in the optimal values and running on Kaggle, this leads to overfitting. Thus, only the number of iterations, learning rate, and depth are kept in this case. The final optimal parameters are: iterations $= 2,000$, learning rate $= 0.05$, and depth $= 11$. The execution time for such a model considerably increases.

2.3.2 Output Analysis

This part is dedicated to irregularities in the output that can be taken advantage of in the submission. After close examination, irregularities are identified in three counters: *'Totem Cours la Reine E-O', '28 Boulevard Diderot E-O', and '28 Boulevard Diderot O-E'*. Those irregularities mimic a similar behaviour than identified previously in the training set, where missing values are present for certain counters.



(a) (b) (c)

Figure 3: Log bike count per day for *a) '28 Boulevard Diderot E-O', b) '28 Boulevard Diderot O-E', and c) 'Totem Cours la Reine E-O'*.

In the training set, those missing values are sometimes the predicament of following values set to zero. Figure 3 shows plots of the location of the missing values for each counter in question.

In the end, those irregularities are handled by plugging zeros in for all dates situated after each beginning of missing date value. This brute force-way is chosen after trials to scale the predicted values down, remove a threshold off the predictions, and set values to zero only for a specific limited time after the irregularity, were unsuccessful. By trial and error, *'Totem Cours la Reine E-O'* is left untouched as removing part of it gives a worse RMSE. All in all, this post-processing technique significantly decreases the RMSE on the Kaggle competition, and gives a serious edge over predictions that do not use this technique.

3 Conclusion

In conclusion, this report gives a road map to a respectable RMSE for the bike challenge. However, improvements can always be made, beginning with the features. The features used in training the models were never compared in terms of their usefulness in the decision tree, they were always selected using the metric of a decrease in RMSE on the validation set. Nonetheless, the *CatBoost* library provides useful tools for feature importance and selection which could have been put to the advantage of a better model. Moreover, data on roadblocks around the counters, programmed maintenance on the counters or incidents/traffic in the close metro stations could have been leveraged to enrich the *CatBoostRegressor* since it could arguably highly affect the bike usage.

In addition, as stated previously in the report, the encoding used for the features in the final submission was not perfect, and could have been studied more thoroughly. To add to the previously made comments, the *CatBoost* library provides some built-in one hot encodings, and it is specifically stated in the documentation not to encode the categorical columns in the preprocessing.

Finally, the hyperparameter tuning led to over fitting. It is with no doubt due to too many hyperparameters put in the equation when giving the model to *optuna*. If the concentration had been on fewer parameters, and the evaluation metric had been the validation score, not the 5-fold cross validation score, then the over fitting might not have taken place.

4 References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[2] Daniel Berrar et al. Cross-validation., 2019.

[3] Tianfeng Chai and Roland R Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250, 2014.

[4] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[5] Pypi Collective. holidays. https://github.com/vacanza/python-holidays, 2023.

[6] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[7] Kathleen Hornsby, Max J Egenhofer, and Patrick Hayes. Modeling cyclic change. In *International Conference on Conceptual Modeling*, pages 98–109. Springer, 1999.

[8] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[9] Christian Lamprecht. Meteostat, the weather's record keeper. https://meteostat.net, 2023.

[10] Toby Phillips. covid-policy-tracker. https://github.com/OxCGRT/covid-policy-tracker, 2023.

[11] David San. Paris metro map. https://github.com/davidsan/paris-metro-map, 2013.

[12] Georgi P Tolstov. *Fourier series*. Courier Corporation, 2012.