Report: Predicting cyclist traffic in Paris

Raphaël Amzallag (Ecole Polytechnique) - Henri Vignial (Ecole Polytechnique)

17/12/2023

Github Repository: link

1 Introduction

The project relied on the analysis of a dataset which logged bike counter data in Paris over 496827 instances on 12 features. This data as well as any additional information would be the basis for the implementation of machine learning models to predict future fluctuations in the number of bikes at the counters studied.

2 Exploratory data analysis

The first step taken to decrypt the dataset provided was to check for missing values, which are not present in the file. It then appeared that there were some redundant variables. Indeed, 'counter_id', 'counter_name', 'site_id', 'site_name' as well as 'counter_technical_id' provided no obvious exploitable information for the analysis. To keep track of the counters studied, only the latitude and longitude were initially kept to take the locations into account. However, further analysis revealed that keeping 'counter_id' provided better results, especially once the variable had been encoded. Finally, 'bike_count' was deleted and 'log_bike_count' kept as the output variable since prediction accuracy was the goal, and not interpretability. Indeed, the logarithm transformation often helps to minimize variance deviations which often increases overall precision.

3 Additional data

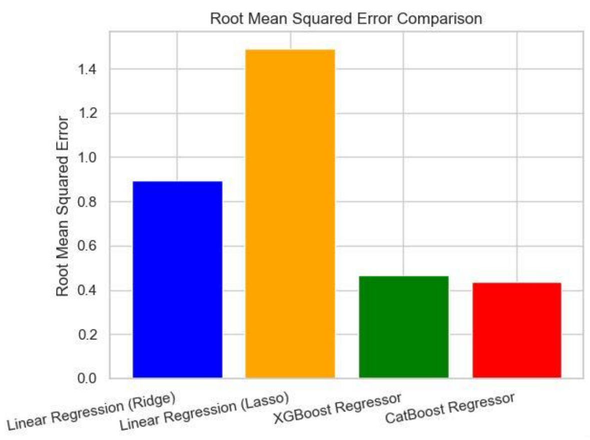
In order to provide the most accurate prediction of the number of bikes at any given time, it became necessary to incorporate additional datasets to take into account variables which might have an effect on the outcome. The first priority was to integrate the weather, and a few parameters seemed paramount such as precipitation, temperature, wind speed as well as visibility. As such, these were the only columns kept on the dataset that was selected, which offered day-by-day data on the necessary features. On top of the variables mentioned above, the 'icon' column was kept as it provided a simple classification of the overall weather on the day, which would be encoded later.
As many Parisians ride their bicycles to go to work, it seemed important to take into account holidays into the model. To do so, the 'jours-feries-france' library was implemented since it gives access to the bank holidays per year and geographical zone. Additionally, the library: 'vacances-scolaires-france' was used in order to integrate public holidays. Finally, since the data provided covered periods during the Covid pandemic, the dates of lockdowns and curfews were added to the model. At first, an existing library was used to get easy access to the relevant dates. However, the final model includes a manual calculation of the lockdown and curfew variables for increased ease of use.

4 Preprocessing

First, the date variable was separated into its individual components for easy manipulation. Then, the days were separated into weekdays and weekends since the initial analysis showed differences in cyclist behavior between these instances. Furthermore, the variables such as hours, days, weeks and months were encoded by using sine and cosine functions. Indeed, machine learning models struggle to take into account the cyclical nature of these parameters, so the overall outcome should be more precise. The numerical values were further encoded with a standard scaler to remove the mean and scale variance to 1 to enable the model to treat all variables similarly by preventing some values from dominating others because of their initial size.
Finally, categorical variables were processed using one-hot encoding, to avoid ordinal assumptions and improve model compatibility. This was an effective method since none of the columns considered had an excessive number of values.

5 Model selection, hyper-parameter tuning

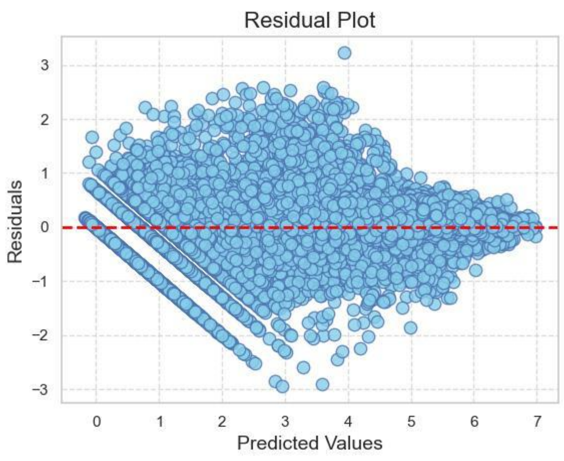
Upon consolidating all external data, the focus shifted towards constructing a predictive model for 'log_bike_count'. The initial phase involved crafting four models—Lasso, Ridge, XGBoost, and CatBoost—each without hyper-parameters. This approach provided valuable insights into the individual performance of each regressor, with the root mean square error (RMSE) depicted in the figure below. The visual representation indicated beyond doubt that CatBoost outperformed its counterparts.



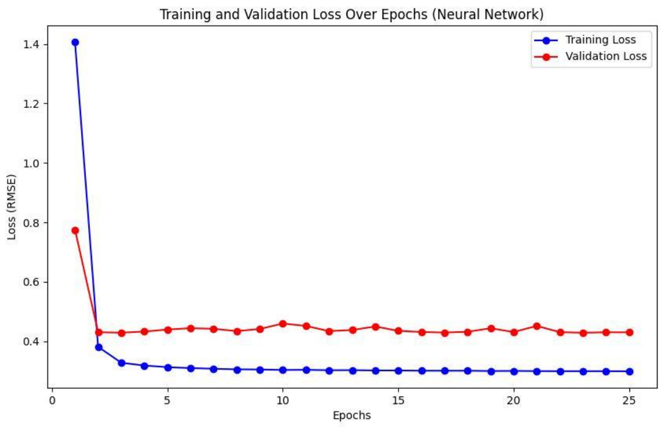
Subsequently, a GridSearch was undertaken to fine-tune the hyper-parameters of the chosen CatBoost model; this resulted in increased overall performance.
The gridsearch procedure was focused on the 'depth', 'iterations', 'rsm' (random subspace method) and 'subsample' parameters. In the end, the calculations outputted the following values, respectively: 12, 1000, 0.3 and 0.7 which seemed to accurately capture the complexities of the data studied and thus to improve the RMSE on the training set. For each parameter, the ranges considered were: for 'depth', between 5 and 13, for 'iterations' between 500 and 1500 every 250, for 'rsm' from 0.25 to 0.4 every 0.05 and for subsample from 0.5 to 0.8 every 0.1.

6 Residual analysis and ensemble methods

Following this improvement, the focus shifted to assessing the model's residuals (cf. figure below). The lack of randomness in the residual plot led to the conclusion that the model failed to capture certain relationships within the features.



In order to improve model performance, ensemble methods, particularly experimenting with stacking models were a logical next step. The initial strategy focused on enhancing CatBoost by integrating a Dense Neural Network, aiming to uncover latent relationships within the dataset. Although the model exhibited a commendable performance on the training set, the test set results unveiled a notable disparity (cf. figure below). This discrepancy suggests challenges beyond overfitting, which could translate to potential issues such as underfitting, suboptimal model complexity, or issues with data quality. Consequently, a decision was made to retain the standalone CatBoost model, given its consistent superiority. It was also recognised that further exploration and refinement would be necessary to address the challenges observed.



7 Further improvements, future outlook

Despite the satisfactory performance of the model, many additional techniques could be considered to increase prediction accuracy. Indeed, subsequent observations revealed a malfunction in specific counters during the studied period. This anomaly could have been leveraged by employing a classification model to distinguish between functioning (labeled as 1) and malfunctioning (labeled as 0) states. Subsequently, a CatBoost model could have been applied, incorporating the additional feature derived from the classification results. An alternate solution would have been to remove the malfunctioning counters altogether to avoid skewing the results.

Alternative procedures could further be explored. Incorporating a time series model, such as Meta's Prophet, could have facilitated the development of a multivariate time series model. Additionally, the exploration of a Long Short-Term Memory (LSTM) Neural Network was considered, since they are known to be better suited to capture temporal matters in time series data. However, implementing such a model would have been too time-consuming given the project’s constraints.

Finally, collaboration and overall workflow could have been improved to increase efficiency and accountability. While Git was utilized for tracking updates, a more structured organizational approach could have been implemented to systematically track results. One strategy that could be implemented in future projects would be to dedicate a notebook to each submission, which would streamline the documentation of methods and results.