Python for Data Science

kaggle Challenge - Bike count in Paris

Adrien SENGHOR - Victor SOTO
"*attaquants comme bappé*" team

December 20th, 2024
APM_51460_EP (2024-2025)

THE GOAL AND POTENTIAL CHALLENGES AT FIRST GLANCE

THE GOAL 🎯

- A prediction…
  - **the log hourly traffic at each bike counter in Paris**, for a given period (2020-09-01 01:00 to 2021-09-09 23:00)
- …with accuracy
  - **measured by the root-mean-square error (RMSE)** between the predicted log-count and the evaluated target

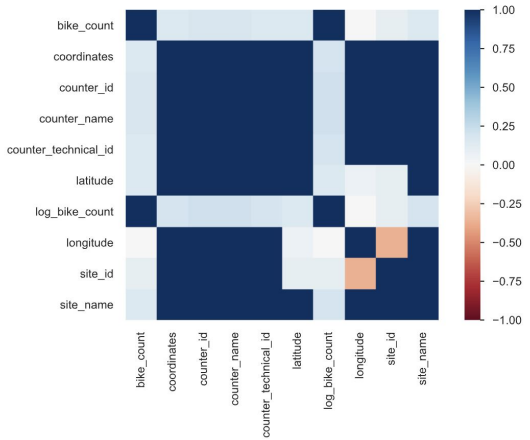
THE CHALLENGES at first glance

- Data exploration (missing values, etc.)
- External data integration
- Model selection and finetuning

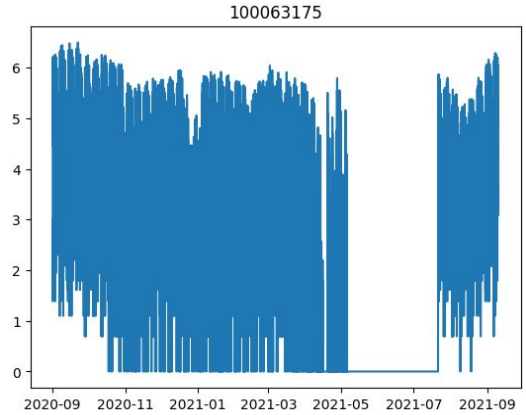
-> EASY!!

EDA - TRAINING SET

Heatmap for X features



Log count problem with counters



KEY INSIGHTS

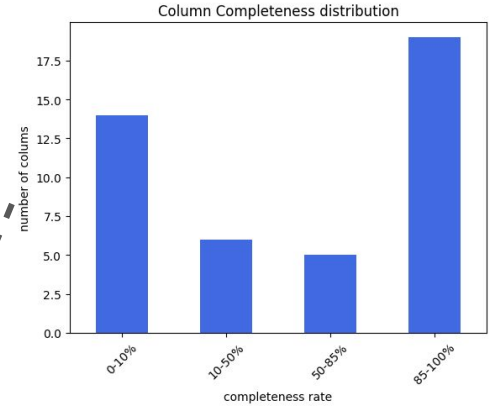
- Clean set with **redundant features** (counter location information): overfitting risk
- y data with **missing values** for identified counters, probably due to problems with the sensors on a certain period

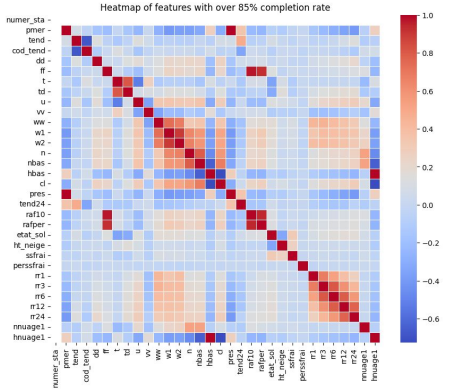
EDA - EXTERNAL DATA

- Large dataset
  - 59 columns
  - open dataset sourced from SYNOP

- Varying data quality across columns
  - Presence of empty and incomplete columns
  - Significant number of missing values across features

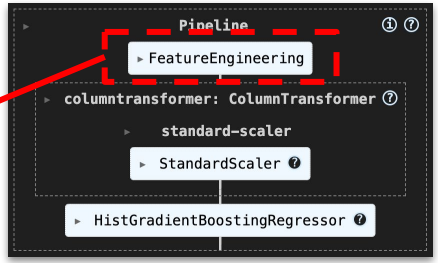
- Independent features
  - Minimal correlation between features
  - Substantial amount of data requiring preprocessing



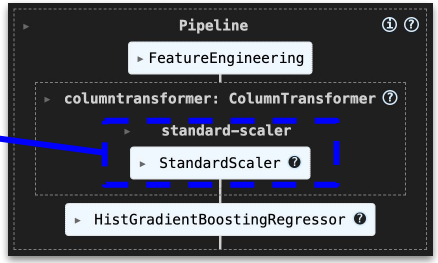


FEATURE ENGINEERING

- Training data
  - **X["date"] column**
    - **New datetime features**
    - **Sine/cosine transformation**[1]
  - "Plugged-in" external data
    - Public holidays and Covid dates added
- External data -> difficult to select the right features
  - Attention to the chronology of the 2 datasets
  - Attention to the missing values (most frequent modality filling)



- Encoding -> no useful categorical features spotted
  - Only **numerical features** kept (no OneHot/OrdinalEncoder)
  - **StandardScaler**[2] applied



[1] Three approaches to encoding time informations as features for ML models, Nvidia developer blog
[2] StandardScaler module from scikitlearn is used to normalize and scale the values from a column

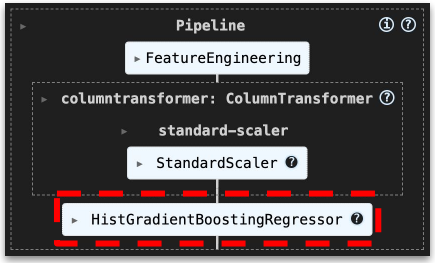
MODEL AND HYPERPARAMETERS SELECTION

- Naive approach for **feature selection**

- **Model comparison**
  - Complete FeatureEngineering pipeline (previous slide)
  - Default parameters
  - **5-fold Cross-Validation**
  - RMSE comparison

Model	RMSE
Ridge	0.90
RF	0.98
CatBoost	0.71
HGB	0.72

- **Model finetuning**
  - **HistGradientBoostingRegressor**
  - Hyperparameter optimization with OPTUNA
  - difficult to select the right ranges to explore



RESULTS

- Hyperparameter finetuned
  - on learning_rate, max_iter, max_depth and min_samples_leaf

- **BEST PREDICTION - 0.61 RMSE**
  (🏆 13th position on the Kaggle on the private Leaderboard[1])

- GitHub structuration
  - FINAL_model_HGB_Kaggle.py using the FINAL_FeatureEngineering.py file

[1] MSDB 2024, Kaggle

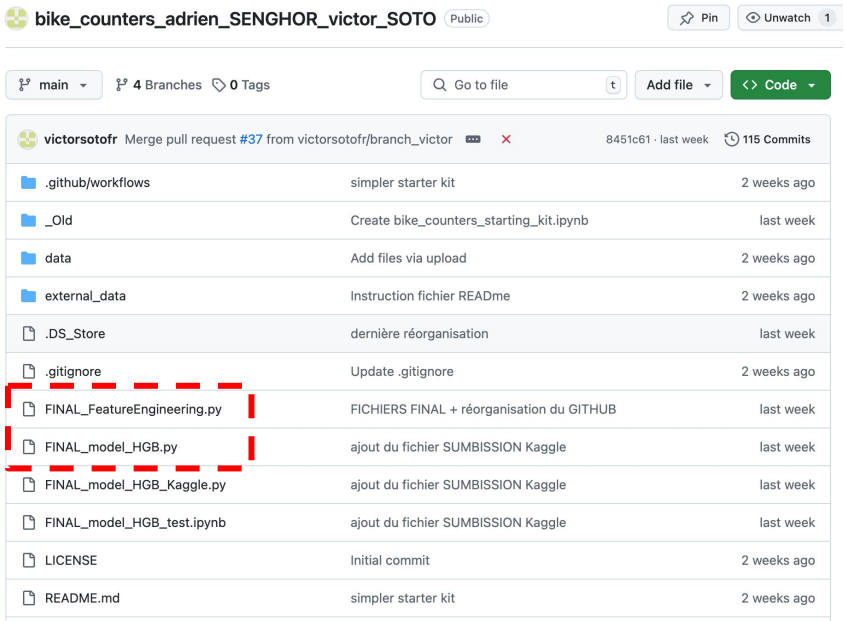
Thank You!

Adrien SENGHOR - Victor SOTO

Appendix

OUR GITHUB

https://github.com/victorsotofr/bike_counters_adrien_SENGHOR_victor_SOTO.git



FEATURE ENGINEERING (1/2)

DIFFICULTIES

- Select the right features
- Spot the useful correlations
- Not over-engineer our model (stack features and hope for the best incorporation within the model)

IDEAS (1/2) 💡

- Add more time to explore the external data
- Do **L1-regularization**, ie Lasso regression, before applying the final model
    - It enables a preselection of the most useful features

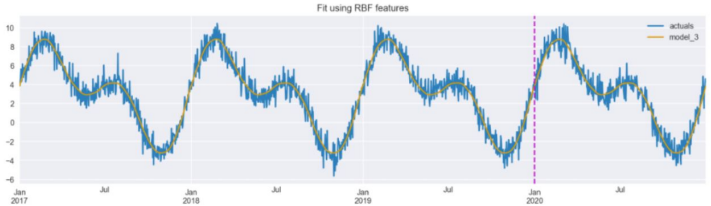
FEATURE ENGINEERING (2/2)

IDEAS (2/2) 💡

- Datetime feature
  - **Radial basis encoding** - not fully understood, difficult application and integration in the pipeline

```
model_3 = LinearRegression().fit(X_3.iloc[:TRAIN_END],
                                 y.iloc[:TRAIN_END])

results_df["model_3"] = model_3.predict(X_3)
results_df[["actuals", "model_3"]].plot(figsize=(16,4),
                                        title="Fit using RBF features")
plt.axvline(date(2020, 1, 1), c="m", linestyle="--");
```



[1] Three approaches to encoding time informations as features for ML models, Nvidia developer blog

MODEL SELECTION

DIFFICULTIES

- Have a first idea of what could be the best model right away
- Training without spending too much time
- Finetune the final model's hyperparameters with the right ones and right ranges for them

IDEAS 💡

- **AutoML**[1]

[1] AutoML