Stage 2 – Machine Learning Project
Metro Traffic Volume Forecasting

MAZEPA Noémie
MORLET Lorrain
MARCELINO Auriane
MARTIN Aymeric

ESILV DIA5

Contents

1 Previous Methods and Limitations

1.1 Methods in Stage 1

Stage 1 focused on predicting metro traffic volume using preprocessing steps like encoding holidays, adding temporal features (e.g., season, part of the day), and ranking weather conditions. Visualizations were used to analyze data trends and correlations.

Several models were tested:

- **Linear Regression and Ridge**: Served as baselines but failed to capture non-linear relationships.
- **K-Neighbors Regressor (KNN)**: Proximity-based predictions but sensitive to outliers.
- **Random Forest Regressor (RFR) and Gradient Boosting Regressor (GBR)**: Performed best, with GBR achieving an R^2 of 0.970 and an MSE of 117,955.06.

1.2 Key Limitations

- **Non-linear Data Handling**: Linear models performed poorly ($R^2 \approx 0.388$).
- **Imbalanced Data**: Rare features like holidays and snowfall created challenges.
- **Computational Overhead**: Ensemble models required significant resources.
- **Feature Gaps**: Limited external data integration and preprocessing improvements.

2 Improvement Assumptions

2.1 Objectives of Stage 2

The main objectives driving Stage 2 were:

- To improve the capacity of models to handle non-linear relationships in the dataset, as linear models underperformed in Stage 1.
- To refine the handling of imbalanced features, particularly rare events such as holidays or extreme weather conditions.
- To implement systematic tuning of hyperparameters for better model optimization and generalization.
- To introduce and evaluate advanced algorithms to outperform existing Stage 1 models.

2.2 Improvements Made

- **Advanced Models Added:** New algorithms, such as XGBoost, LightGBM, and CatBoost, were introduced to handle non-linear patterns more effectively than the Gradient Boosting and Random Forest models used in Stage 1. These algorithms manage categorical data and imbalances better, especially in features like holidays and weather conditions.
- **Improved Hyperparameter Tuning:** Optuna was used for automated and efficient hyperparameter optimization. This marked an upgrade from Stage 1's grid search approach, allowing more precise tuning of key parameters like learning rate, depth, and number of estimators.
- **Feature Engineering Refinements:** Adjustments to feature preprocessing included scaling and encoding improvements to support the new algorithms and mitigate the sensitivity to outliers observed with models like KNN in Stage 1.
- **Ensemble Techniques for Robustness:** Ensemble techniques combining models (e.g., boosting and bagging) were implemented to reduce variance and overfitting, addressing the performance instability observed in Stage 1.

3 New Algorithm Description

3.1 New Algorithms

In Stage 2, the project implemented advanced algorithms to improve upon the models used in Stage 1. The new algorithms were selected to handle the dataset's non-linear patterns, imbalanced data, and categorical features more effectively. Below is a description of these algorithms:

- **XGBoost (Extreme Gradient Boosting)**:
  - **Description:** XGBoost is a high-performance, scalable tree-based boosting algorithm that enhances model accuracy while maintaining efficiency. It improves upon traditional gradient boosting by implementing regularization techniques to prevent overfitting.
  - **Key Features:** Regularization parameters to control model complexity, parallel computation for faster training, handles missing values effectively.
  - **Reason for Use:** XGBoost was chosen for its ability to model non-linear relationships and manage high-dimensional data efficiently.
- **LightGBM (Light Gradient Boosting Machine)**:
  - **Description:** LightGBM is a gradient boosting framework optimized for speed and memory usage. It employs histogram-based algorithms to improve computational efficiency and scalability.
  - **Key Features:** Leaf-wise tree growth for better accuracy and effective handling of large datasets with categorical features.
  - **Reason for Use:** LightGBM was selected for its computational efficiency, making it suitable for large-scale data while maintaining high prediction accuracy.
- **CatBoost (Categorical Boosting)**:
  - **Description:** CatBoost is a gradient boosting library specifically designed to handle categorical features without requiring explicit preprocessing like one-hot encoding.

  - **Key Features:** Built-in support for categorical variables and ordered boosting to reduce overfitting.
  - **Reason for Use:** CatBoost's ability to handle categorical features directly addresses the imbalances observed in the "holiday" and "weather condition" columns.

- **Improved Ensembles**:
  - **Description:** Ensemble approaches combining bagging and boosting techniques were employed to enhance model robustness and prediction accuracy.
  - **Key Features:**
    * Bagging with Random Forest: Combines results from multiple decision trees trained on random subsets of data.
    * Boosting with AdaBoost and Random Forest: Uses a sequential approach where misclassified samples are given more weight in subsequent iterations.
  - **Reason for Use:** These techniques aim to improve generalization and reduce the variance observed in single-model implementations.

3.2 Limitations

Despite the significant improvements introduced in Stage 2, there are still limitations to consider regarding the new methodologies and algorithms. These limitations align with the challenges inherent in the dataset and the methods employed.

- **Computational Complexity:** Advanced algorithms like XGBoost, LightGBM, and CatBoost require substantial computational resources, especially during hyperparameter tuning using Optuna.

  Ensemble techniques combining bagging and boosting add further overhead, making real-time implementation more challenging.

- **Data-Specific Challenges:**
  - Imbalanced Data: Although algorithms like CatBoost and LightGBM are designed to handle categorical imbalances, extreme scenarios (e.g., holidays or rare weather events) might still affect pre-

dictive accuracy. Over-reliance on boosting can lead to overfitting specific underrepresented cases.

  - Outliers: Stage 2 preprocessing addressed many outliers, but the presence of extreme values in weather-related features (e.g., rainfall anomalies) may still skew predictions for certain models.

- **Model Complexity and Interpretability:** Ensemble methods, particularly boosting frameworks, result in highly complex models with reduced interpretability. While performance improves, understanding the underlying feature contributions becomes more difficult.

- **Scalability:** While LightGBM is optimized for large datasets, combining multiple ensemble techniques (e.g., boosting and bagging) in one pipeline may hinder scalability for very large or streaming datasets.

- **Risk of Overfitting:** Boosting algorithms inherently focus on reducing errors in difficult samples, which could lead to overfitting on noise or anomalies despite regularization.

  Fine-tuning hyperparameters mitigates this risk but does not eliminate it entirely.

4 Methodology

4.1 New Algorithm Implementation and Hyperparameters

In Stage 2, advanced machine learning models and more sophisticated hyperparameter optimization techniques were implemented to address the limitations of Stage 1. This section describes the steps involved in implementing these algorithms and highlights the specific hyperparameters used for each model.

4.1.1 Implementation Process

- **Dataset Preprocessing**: The preprocessing pipeline was updated to better support the advanced algorithms introduced in Stage 2. This included scaling, encoding categorical features, and handling missing values.

- **Model Selection**: Models introduced in Stage 2 include:
  - XGBoost
  - LightGBM
  - CatBoost
  - Ensemble techniques combining bagging (Random Forest) and boosting (e.g., AdaBoost with Random Forest).

  Each model was trained and evaluated on the same train/test split as in Stage 1 for fair comparison.

- **Hyperparameter Optimization**: Optuna was employed to systematically tune hyperparameters for XGBoost, LightGBM, and CatBoost. This approach allowed for an efficient exploration of the hyperparameter space.

4.1.2 Model-Specific Hyperparameters

- **XGBoost (XGBRegressor)**
  - Learning Rate: 0.8
  - Max Depth: 5
  - Number of Estimators: 300

- **LightGBM (LGBMRegressor)**
  - Learning Rate: 0.2
  - Max Depth: 5
  - Subsample: 0.8

- **CatBoost (CatBoostRegressor)**
  - Learning Rate: 0.099
  - Depth: 10
  - Subsample: 0.461
  - Min Data in Leaf: 100
  - Colsample_bylevel: 0.756

- **Ensembles with Bagging and Boosting**
  - Random Forest with Bagging: Number of base estimators: 20. Max samples and features adjusted for variance reduction.
  - Boosted Random Forest: AdaBoost applied on a Random Forest model to leverage strengths of both bagging and boosting.

4.1.3 Key Differences from Stage 1

- **Algorithm Variety**: Stage 1 primarily relied on Random Forest, Gradient Boosting, and simpler models like Linear Regression and Ridge. Stage 2 introduces more advanced models such as XGBoost, LightGBM, and CatBoost, which are designed for large, complex datasets.
- **Hyperparameter Tuning**: Stage 1 used grid search for hyperparameter tuning, while Stage 2 employed Optuna, significantly increasing efficiency and exploration depth.
- **Feature Engineering Adjustments**: Enhanced preprocessing steps, including scaling and encoding, were tailored to support Stage 2 models.

5 Results

5.1 Metrics

Model	**MSE**	R^2
AdaBoostRegressor	569,084.77	0.858
GradientBoostingRegressor	89,337.02	0.978
XGBRegressor	114,806.77	0.971
LGBMRegressor	114,087.88	0.971
CatBoostRegressor	83,835.91	0.979
RandomForestRegressor_Bagging	164,738.39	0.959
RandomForestRegressor_Boosting	125,312.05	0.969

Table 1: Performance Metrics for Stage 2 Models

5.2 Interpretation

- Best Performing Model: The CatBoostRegressor is the best performing model with the highest R^2 score (0.979) and the lowest MSE (83,835.91), indicating the most accurate predictions.
- Second Best Performing Model: The GradientBoostingRegressor also performed exceptionally well with a high R^2 score (0.978) and a low MSE (89,337.02).
- Middle Ground: The XGBRegressor, LGBMRegressor, and RandomForestRegressorBoosting performed well with high R^2 scores and moderate MSE, but were outperformed by the CatBoostRegressor and GradientBoostingRegressor.
- Worst Performing Model: The AdaBoostRegressor performed the worst with a relatively low R^2 score (0.858) and high MSE, indicating less accurate predictions compared to other models.

6 Overfitting

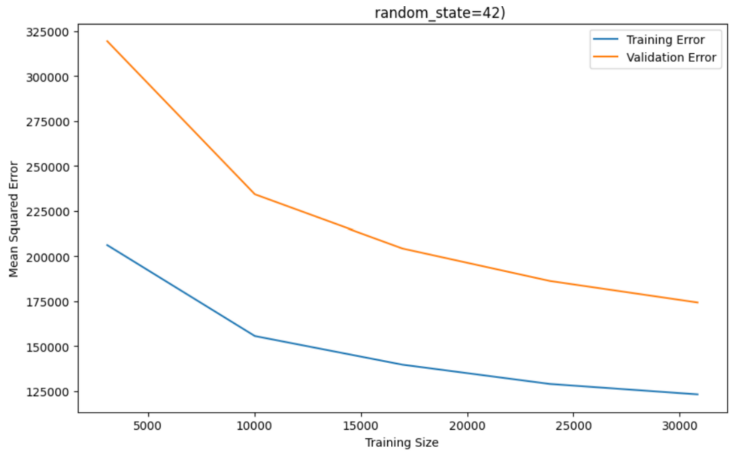

Figure 1: Learning Curve BaggingRegressor

This is the learning curve for the train and test split of the **BaggingRegressor** model. As we can see, the orange line refers to the validation error.

The two curves stay close all along the training process. Moreover, the validation error does not diverge from the training error. This confirms that there is no overfitting for this model. For the other models, we observed the same results. Therefore, there is no overfitting in our models.

7 Evaluation

These results suggest that ensemble methods like **CatBoostRegressor** and **GradientBoostingRegressor** are highly effective for this particular dataset. Among these, **CatBoostRegressor** stands out as the top performer, making it the preferred choice for this regression task.

7. Discussion and Conclusion

7.1 Stage 1 Models

Model	**MSE**	**R^2**
RandomForestRegressor	127,354.99	0.9681
GradientBoostingRegressor	117,955.06	0.9705
LinearRegression	2,445,271.58	0.3884
Ridge	2,445,268.29	0.3884
KNeighborsRegressor	227,094.39	0.9432

Table 2: Performance of Stage 1 Models

7.1 Stage 2 Models

7.2 Interpretation and Comparison

- **GradientBoostingRegressor**: Performs well in both stages, with Stage 2 showing a slight improvement in both MSE and R^2.
- **CatBoostRegressor**: In Stage 2, it outperforms all models in both stages in terms of MSE and R^2, making it the best-performing model overall.

Model	MSE	R^2
AdaBoostRegressor	569,084.77	0.8577
GradientBoostingRegressor	89,337.02	0.9777
XGBRegressor	114,806.77	0.9713
LGBMRegressor	114,087.88	0.9715
CatBoostRegressor	83,835.91	0.9790
RandomForestRegressor_Bagging	164,738.39	0.9588
RandomForestRegressor_Boosting	125,312.05	0.9687

Table 3: Performance of Stage 2 Models

- **LinearRegression and Ridge**: These models in Stage 1 perform poorly compared to other models, indicating that linear models may not be suitable for this dataset.
- **AdaBoostRegressor**: In Stage 2, it has the highest MSE and lowest R^2 among the Stage 2 models, suggesting it is the least effective model in this stage.
- **RandomForestRegressor**: Shows consistent performance across both stages, with slight variations in MSE and R^2. The boosting variant in Stage 2 performs similarly to the original model in Stage 1.

Overall, the models in Stage 2, particularly **CatBoostRegressor** and **GradientBoostingRegressor**, show better performance compared to the models in Stage 1. This indicates that more advanced ensemble methods and boosting techniques can provide significant improvements in predictive performance.