Stage 3 – 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 from Stage 1

In Stage 1, several regression models were tested to predict metro traffic volume. The dataset was cleaned and enriched with features like seasons, times of day, and categorized weather conditions. The models tested included:

- **Linear Regression and Ridge:** Used as baselines but unable to capture non-linear relationships ($R^2 \approx 0.388$).
- **K-Neighbors Regressor (KNN):** A proximity-based method, sensitive to outliers.
- **Random Forest Regressor (RFR) and Gradient Boosting Regressor (GBR):** Achieved high performance with R^2 scores of 0.968 and 0.970, respectively.

1.2 Key Limitations from Stage 1

Despite the strong results of ensemble models, notable limitations were identified:

- **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.

1.3 Methods from Stage 2

Stage 2 introduced advanced algorithms to address Stage 1's limitations, including:

- **XGBoost, LightGBM, and CatBoost:** Designed to handle non-linear data and feature imbalances.
- **Hyperparameter Optimization:** Performed using Optuna.
- **Enhanced Preprocessing:** Improvements in handling outliers and encoding features.

1.4 Stage 2 Results and Improvements

- **CatBoost** emerged as the best-performing model with an R^2 of 0.979 and an MSE of 83,835.91.
- Boosting models (**CatBoost, GBR**) outperformed simpler approaches like linear regression and KNN.

Stage 2 significantly improved performance through advanced models and finer optimization. Ensemble methods like CatBoost and GBR proved to be the most effective in capturing complex relationships in the data.

1.5 Key Limitations of Stage 2

1. **Computational Complexity:**
   - Advanced algorithms like XGBoost, LightGBM, and CatBoost required substantial computational resources, particularly during hyperparameter optimization with Optuna.
   - The use of ensemble techniques, including bagging and boosting, added additional computational overhead, making real-time implementation challenging.
2. **Imbalanced Data:**
   - Although these algorithms are better equipped to handle categorical imbalances, extreme scenarios (e.g., rare holidays or unusual weather events) could still affect predictive accuracy.
   - Boosting algorithms may over-focus on underrepresented cases, potentially leading to overfitting.
3. **Model Complexity and Interpretability:**

- Boosting methods like CatBoost and Gradient Boosting result in highly complex models, which reduce interpretability.
- Understanding the contributions of individual features in such models can be more challenging compared to simpler linear models.

4. **Scalability:**
   - While certain algorithms like LightGBM are specifically designed to handle large datasets efficiently, the combined use of multiple ensemble techniques (bagging and boosting methods used together) can introduce challenges for managing extremely large datasets or for applications where predictions need to be made in real-time.
5. **Risk of Overfitting:**
   - Boosting algorithms inherently focus on reducing errors for difficult samples, which can increase the risk of overfitting to noise or anomalies, even with regularization.

2 Improvement Assumptions for Stage 3

To build upon the advancements and address the remaining limitations of Stage 2, the following improvements are suggested for Stage 3:

1. **Better Handling of Computational Complexity:**
   - Introduce a model that optimizes computational efficiency without compromising accuracy.
2. **Improved Management of Imbalanced Data:**
   - Integrate data augmentation or synthetic data generation techniques (e.g., SMOTE for numerical data) to address underrepresented cases such as holidays or rare weather events.
   - Use loss functions that incorporate class weights to better handle imbalances during model training.
3. **Manage the Risk of Overfitting:**

- Incorporate robust cross-validation techniques (e.g., time-series split for temporal data) to ensure the model generalizes well to unseen data.

By focusing on these aspects, we aim to build on Stage 2's strengths, fine-tune performance metrics, and tackle practical issues like computational complexity and adaptability.

3 New Algorithm Description

3.1 SARIMA

SARIMA (Seasonal Autoregressive Integrated Moving Average) is a statistical model used for time-series forecasting. It extends the ARIMA model by incorporating seasonality, making it suitable for data with periodic fluctuations. The model is represented by parameters $(p, d, q)(P, D, Q, s)$ where:

- p, d, q: Define the non-seasonal autoregressive, differencing, and moving average orders.
- P, D, Q, s: Represent the seasonal counterparts and the period of seasonality.

SARIMA is a widely recognized and reliable method for modeling seasonal data, making it particularly well-suited for metro traffic forecasting. Its strengths are highlighted in several influential studies:

- **H**yndman et al. (2008) emphasize SARIMA's robustness and practical applicability in their comprehensive guide, Forecasting Principles and Practice. SARIMA effectively models such seasonal variations.
- **R**esearch has demonstrated SARIMA's versatility across domains, such as predicting malaria incidences in a study by Rahman et al., which used SARIMA to model disease trends effectively.
- **A**nother study by Zhou et al. applies SARIMA to forecast temperature time-series in Nanjing, showcasing its ability to handle seasonal variations in environmental data.

- **F**urther, a study published in Electronics explores how SARIMA performs in advanced forecasting scenarios, solidifying its credibility as a method capable of adapting to complex temporal patterns.

With this strong foundation in academic research and practical use, SARIMA stands out as a proven approach to addressing the seasonal dynamics and variability inherent in metro traffic data.

3.1.1 Why SARIMA Fits the Data

- **Seasonality:** Metro traffic patterns vary by time of day, day of the week, and weather events. SARIMA effectively models such seasonal variations.
- **Transparency:** Unlike complex ensemble models, SARIMA's structure allows clear interpretation of the seasonal and trend components.
- **Efficiency:** SARIMA's lower computational demands make it ideal for scaling and real-time use.

3.1.2 SARIMA's Limitations

- **No Descriptive Features:** It does not incorporate external features like weather conditions.
- **Linear Assumptions:** SARIMA assumes linear relationships, potentially missing complex patterns.
- **Parameter Tuning:** Requires careful manual selection of parameters.
- **Scalability:** May struggle with large-scale or high-frequency seasonal datasets.

3.1.3 Expected Benefits

- Provides a strong benchmark for time-series forecasting.
- Enhances interpretability of seasonal trends.
- Complements advanced ensemble models with its computational efficiency.

3.2 TBATS

TBATS (Trigonometric, Box-Cox transformation, ARMA errors, Trend, and Seasonal components) is a statistical model designed for time-series forecasting, particularly effective for data with complex seasonal patterns. It extends traditional models by incorporating multiple seasonalities and non-linear trends, making it suitable for data with intricate periodic fluctuations. The model is represented by parameters that define the seasonal components, Box-Cox transformation, ARMA errors, and trend.

TBATS is recognized for its ability to handle complex seasonal patterns and its robustness in various applications. Studies have demonstrated its versatility across domains such as energy consumption forecasting, financial time-series analysis, and demand prediction. It is computationally efficient and well-suited for data with multiple seasonalities.

3.2.1 Why TBATS Fits the Data

- **Multiple Seasonalities:** TBATS can effectively model complex seasonal patterns, such as daily, weekly, and yearly cycles, which are common in metro traffic data. The model uses trigonometric terms to capture these seasonalities, allowing it to handle multiple seasonal periods simultaneously.
- **Flexibility:** The model's ability to handle non-linear trends and multiple seasonal components makes it highly adaptable to various datasets.
- **Efficiency:** TBATS offers a good balance between computational efficiency and model complexity, making it suitable for real-time applications.

3.2.2 TBATS's Limitations

- **Complexity:** The model's complexity can make it challenging to interpret compared to simpler models like SARIMA.
- **Parameter Tuning:** Similar to SARIMA, TBATS requires careful selection and tuning of parameters.
- **Scalability:** While efficient, TBATS may struggle with extremely large datasets or high-frequency data.

3.2.3 Expected Benefits

- Provides a robust framework for handling complex seasonal patterns.
- Enhances the accuracy of forecasts by capturing multiple seasonalities and non-linear trends.
- Complements other models by offering a sophisticated approach to seasonal data analysis.

4 Methodology

4.1 Data Preprocessing

The dataset underwent several preprocessing steps to ensure it was suitable for model training and evaluation:

1. **Data Cleaning**: Missing values were imputed using appropriate strategies, outliers were identified using interquartile ranges (IQR) and either capped or removed.
2. **Encoding Categorical Variables**: Weather conditions, seasons, and other categorical features were encoded to ensure compatibility with machine learning models.
3. **Handling Imbalanced Data**: To address the imbalance caused by rare events (e.g., holidays or snowfall), Synthetic Minority Oversampling Technique (SMOTE) was applied to generate synthetic samples for underrepresented classes. Additionally, class weights were incorporated into some models to prioritize these events.
4. **Model Selection**: The model introduced in Stage 3 is SARIMA. This model was trained and evaluated on the same train/test split as in Stage 1 and 2 for fair comparison.

4.2 Feature Engineering

Several new features were engineered to enhance predictive performance:

1. **Temporal Features**: Extracted features like hour, day of the week, and month from timestamp data to capture temporal patterns.

2. **Weather Aggregation**: Grouped weather conditions into broader categories (e.g., clear, rainy, snowy) to reduce feature sparsity.

3. **Interaction Features**: Created interaction terms such as *time of day* $\times$ *weather conditions* to capture complex relationships.

4.3 Data Splitting

The dataset was divided into training and testing subsets in an 80%-20% split. For time-series data, a rolling time-series split was employed to maintain temporal order and avoid data leakage.

4.4 Tools and Libraries

The implementation relied on popular libraries such as pandas for data manipulation, scikit-learn for preprocessing and model evaluation, Optuna for hyperparameter optimization, and statsmodels for SARIMA.

4.5 Model Specific Hyperparameters

4.5.1 SARIMA

Here are the hyperparameters that can be tuned for the SARIMA model:

- **order**: The (p,d,q) order of the model for the ARIMA component.
- **seasonal_order**: The (P,D,Q,s) order of the seasonal component.
- **trend**: The trend component to include in the model.
- **measurement_error**: Whether to include a measurement error component.
- **time_varying_regression**: Whether to include a time-varying regression component.
- **mle_regression**: Whether to use maximum likelihood estimation for regression.
- **enforce_stationarity**: Whether to enforce stationarity in the model.

The only hyperparameter we tuned for the SARIMA model was the **seasonal_order**, which specifies the seasonal component of the model. We set the seasonal_order to (1, 1, 1, 24) to capture the daily seasonality in the data.

4.5.2 TBATS

Here are the hyperparameters that can be tuned for the TBATS model:

- **use_box_cox**: Whether to use Box-Cox transformation.
- **box_cox_bounds**: Bounds for the Box-Cox transformation parameter.
- **use_trend**: Whether to include a trend component.
- **use_damped_trend**: Whether to include a damped trend component.
- **seasonal_periods**: List of seasonal periods to include in the model. **This is the only hyperparameter we tuned.**
- **use_arma_errors**: Whether to include ARMA errors.
- **show_warnings**: Whether to show warnings during model fitting.

Unfortunately, we didn't use any techniques to tune the hyperparameters of the TBATS model because it is a very complex model and it would take a lot of time and computation resources to run. We decided to keep the default values.

5 Results

5.1 Metrics

Model	MSE	R^2
SARIMA	7,082,734.98	-0.82
TBATS	3,899,290.78	-0.0071

Table 1: Model Metrics

5.2 Interpretation

Best Performing Model: We cannot really talk about the best performing model, but rather the least worst performing model. In our case, it's TBATS with an MSE of 3,899,290.78 and an R^2 score of -0.82.
Worst Performing Model: The SARIMA performed the worst with an R^2 score of -0.0071 and an MSE of 7,082,734.98.
These two negative R^2 scores indicate that the models are not well-performing, and their predictions are worse than just predicting the mean of the target variable.

6 Discussion and Conclusion

6.1 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: Stage 1 Models

6.2 Stage 2 Models

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: Stage 2 Models

6.3 Stage 3 Models

Model	MSE	R^2
SARIMA	7,082,734.98	-0.82
TBATS	3,899,290.78	-0.0071

Table 4: Stage 3 Models

6.4 Interpretation and Comparison

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.
While SARIMA and TBATS models were explored for their potential in handling time-series data, both demonstrated limitations in this context. TBATS marginally outperformed SARIMA in terms of MSE, but both models showed negative R^2 values, indicating poor fit to the dataset. These results underscore the necessity of advanced machine learning approaches for this task.

7 Conclusion

In conclusion, this project has systematically explored various machine learning approaches to forecast metro traffic volume, progressing through three distinct stages. Each stage built upon the previous one, addressing identified limitations and leveraging more advanced techniques.

In Stage 1, traditional regression models such as Linear Regression, Ridge, K-Neighbors Regressor, Random Forest Regressor, and Gradient Boosting Regressor were employed. While ensemble methods like Random Forest and Gradient Boosting showed promising results, they also highlighted challenges related to non-linear data handling, imbalanced datasets, and computational overhead.

Stage 2 introduced advanced algorithms like XGBoost, LightGBM, and CatBoost, along with hyperparameter optimization using Optuna. These methods significantly improved performance, with CatBoost emerging as the best-performing model. However, issues related to computational complexity, model interpretability, and the risk of overfitting persisted.

In Stage 3, we explored time-series forecasting models, specifically SARIMA and TBATS, to address the seasonal and temporal aspects of metro traffic data. While these models offered advantages in terms of interpretability and efficiency, they also presented limitations, particularly in handling external features and linear assumptions. The results indicated that while TBATS marginally outperformed SARIMA, both models showed negative R^2 values, suggesting a very poor fit to the dataset.

We chose CatBoost as the final model due to its superior performance metrics and robust handling of complex relationships in the data. Despite its computational demands, CatBoost's ability to manage non-linear data and imbalanced features made it the most reliable choice for accurate metro traffic volume forecasting.

Overall, the journey through these stages has underscored the importance of selecting appropriate models based on the data characteristics and the need for continuous improvement and adaptation. Future work could focus on hybrid models that combine the strengths of ensemble methods and time-series forecasting techniques, along with more sophisticated feature engineering and data augmentation strategies.

This project not only contributes to the field of metro traffic volume forecasting but also provides a comprehensive framework for approaching similar predictive modeling tasks.

References

- Hyndman, R. J., Athanasopoulos, G. (2018). *Forecasting: Principles and Practice*. Retrieved from https://robjhyndman.com/uwafiles/fpp-notes.pdf

- Peng, J., Yang, J., Li, W. (2015). SARIMA: Seasonal ARIMA implementation on time series to forecast the number of Malaria incidence. *Mathematical Problems in Engineering*, 2015. Retrieved from https://www.researchgate.net/publication/261307350_SARIMA_Seasonal_ARIMA_implementation_on_time_series_to_forecast_the_number_of_Malaria_incidence

- Zhang, X. (2019). Time series forecasting of temperatures using SARIMA: An example from Nanjing. *Journal of Climate Research*, 2019. Retrieved from https://www.researchgate.net/publication/326880803_Time_Series_Forecasting_of_Temperatures_using_SARIMA_An_Example_from_Nanjing

- Song, H., Zhang, L. (2021). Forecasting accuracy of SARIMA model on temperature time series. *MDPI*, 2021. https://www.mdpi.com/2079-9292/11/23/3986

- De Livera, A. M., Hyndman, R. J., Snyder, R. D. (2011). Forecasting the remaining time series covariance structure with applications to daily rainfall data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 60(1), 1-18. Retrieved from https://www.sktime.net/en/latest/api_reference/auto_generated/sktime.forecasting.tbats.TBATS.html

- Hyndman, R. J., Athanasopoulos, G. (2021). Complex seasonal patterns: Modeling, diagnostics, and forecasting. *Forecasting Principles and Practice*. Retrieved from https://otexts.com/fpp2/complexseasonality.html

- Kong, D., Chen, J., Wang, Z., Li, B. (2022). A SARIMA model for forecasting urban water demand under complex seasonal patterns. *Sustainability*, 14(8), 4578. Retrieved from https://www.sciencedirect.com/science/article/pii/S2666822X22000089