DS340 Final Paper - Predicting Bluebikes Usage and Demand in Greater Boston

Authors
Jessica Cannon, Ayu Izumo

Introduction

As urban populations grow, cities increasingly rely on sustainable transportation solutions to address challenges such as traffic congestion, environmental pollution, and limited parking infrastructure. Bike-sharing systems, like Boston's Bluebikes, play a vital role in promoting sustainable urban mobility. However, these systems face operational challenges, particularly in managing the distribution of bikes across stations to meet fluctuating user demand. The focus of this research is to determine which machine learning model, if any, most effectively predicts demand across Boston's Bluebikes system. Specifically, this study seeks to address the following questions:

- Which models best predict system usage across the 13 different districts where the Bluebikes system operates in Greater Boston?
- What features of the data, as well as the model, are most effective in predicting demand across the Bluebikes system?

Answering these questions is crucial for optimizing bike distribution, reducing operational costs, and improving the overall user experience. A better understanding of demand patterns can help Bluebikes administrators proactively allocate resources and ensure the system operates efficiently, meeting the needs of its growing user base. This paper provides an analysis of various machine learning approaches, identifying the features and models that best predict Bluebikes usage. By leveraging predictive analytics, this research aims to contribute practical insights to the operation of bike-sharing systems, fostering their continued success in urban environments.

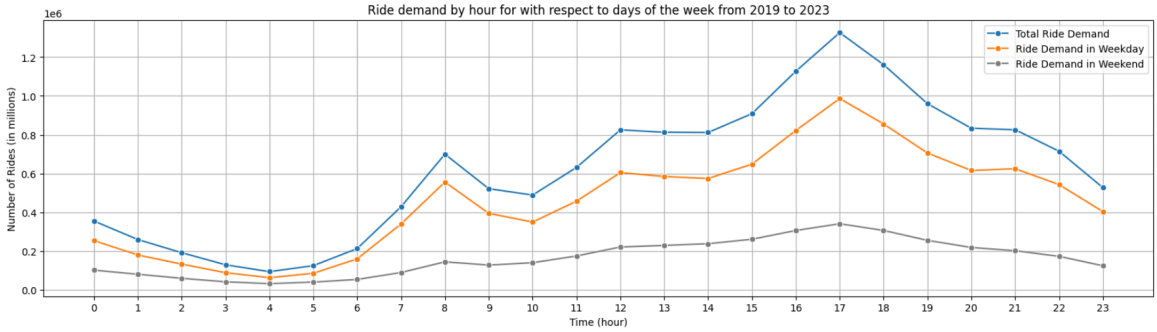
Methodology

This study employed a structured approach to predict Bluebikes usage and demand, comprising several key steps.

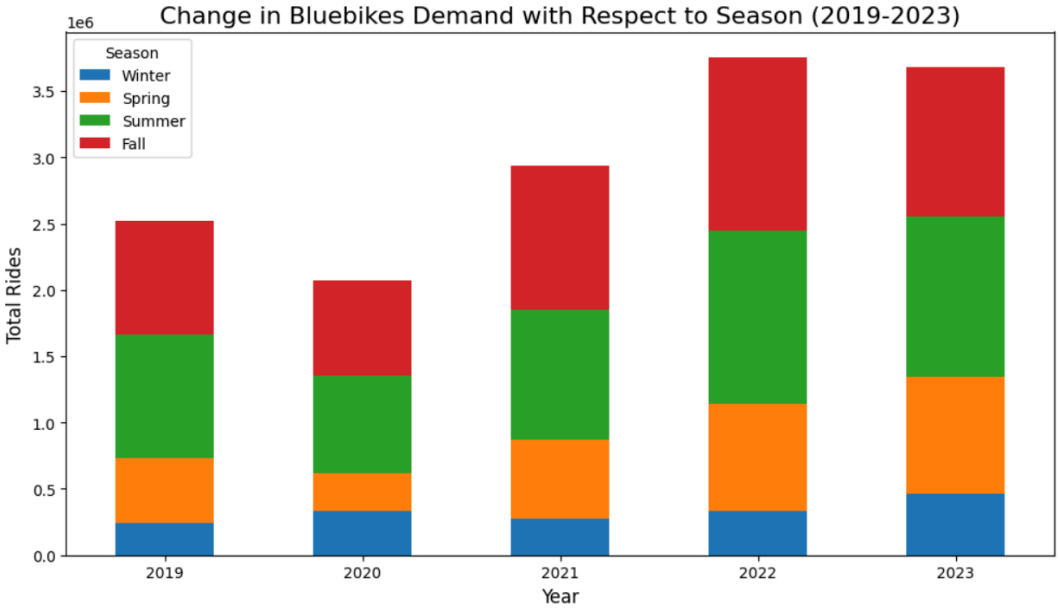
1. Data Preprocessing, Cleaning, and Integration
   We started by collecting trip history data from the official Bluebikes System website. This included monthly datasets spanning the years 2019 to 2023, each detailing individual rides that occurred during the respective month. To create one comprehensive dataset to work on, we combined these individual datasets. This initially proved to be difficult, as the datasets from March 2023 onwards had different column names compared to those of previous months. However, we worked around this by conditioning on one of the new dataset column names (specifically 'ride_id') to rename and rearrange the new datasets' columns and properly concatenate them with the previous data. Further data cleaning ensured consistency and removed erroneous or incomplete entries that could have impacted our results.

2. Exploratory Data Analysis

We conducted an exploratory data analysis to examine demand variations across time (e.g., seasonal trends) and regions (e.g., usage patterns within the 13 districts).



The plot above visualizes ride demand per hour with respect to total demand (blue), weekday demand (orange), and weekend demand (gray), where a weekday is classified as Monday-Friday and the weekend is either Saturday or Sunday. From this, we discovered that more demand for Bluebikes is seen on weekdays compared to weekends, and that the demand tends to reach its peak around 5PM.



By analyzing bike demand trends across seasons, we observed a steady increase in overall Bluebikes usage since 2020. Demand was highest during the summer and fall seasons, with a significant decline during the winter months. Additionally, a separate district-level analysis revealed that Boston and Cambridge were the most popular locations for Bluebikes usage, with over 7 million and 4 million total rides, respectively. In contrast, Malden and Medford had the lowest usage, with approximately 4,000 and

7,000 rides, respectively. These insights were instrumental in guiding our feature engineering and model design.

3. Model Selection

   In order to compare models, our dataset was split into training and testing sets based on a split date (2023-01-01 - gave an approximate 80/20 split). The training set consists of historical data before the split date, while the testing set contains data from the split date onward. The target variable, 'Totalrides' (per day), was extracted from the dataset, and the data for both training and testing sets was converted to numpy arrays for use in the models. The following four models were chosen for initial training and evaluation against each other:

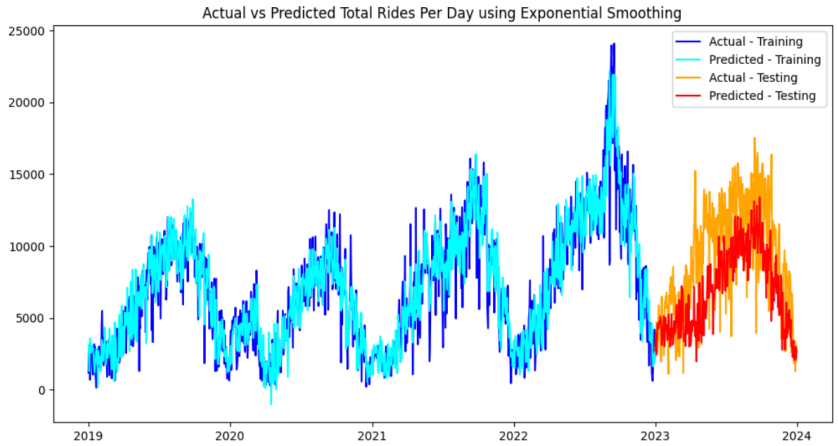
   - Exponential Smoothing
     - To ensure that the models below were performing well, we first constructed an Exponential Smoothing model to use as an easy baseline. This model utilized an additive trend component (indicating that the data exhibits consistent increases or decreases over time), a damped trend (allows the trend to gradually decrease over time, helping to avoid overfitting to extreme trends), and a seasonality period of 365 to reflect the yearly cycles of our data.
   - Recurrent Neural Network (RNN)
     - Before training this model and the LSTM, the data was normalized using a MinMaxScaler to improve model performance and convergence. Since RNNs rely on sequential data, the data was reshaped into sequences using a lookback window of time_steps = 30. A helper function, create_sequences(), was created to generate these sequences.
     - The model was built using the Sequential API in Keras. It consists of two SimpleRNN layers (50 units each and the tanh activation function), two Dropout layers (one after each RNN layer, dropout rate of 20%), and a single dense layer with 1 neuron to output the next value in the time series. The model is compiled with the Adam optimizer and a learning rate of 0.001, with mean squared error (MSE) as the loss function, and was trained for 100 epochs with a batch size of 32.
   - Long Short-Term Memory (LSTM)
     - The construction of the LSTM model directly follows that of the RNN, except that two LSTM layers are used instead of two SimpleRNN layers. This is done to ensure equal and fair comparisons to each other.
   - XGBoost
     - The dataset was prepared for supervised learning by creating lag features, which capture the historical values of the target variable ('Totalrides') over a specified lookback period (lag = 30 days). This is achieved through the function create_lag_features(), which generates lag features by shifting the

target variable (Totalrides) for 1 to 30 days. The input features consist of these lag features. This was done so that the model could be better comparable to the recurrent approaches above.

- An XGBoost Regressor model (XGBRegressor) is used for forecasting. The model is initialized with the following hyperparameters: n_estimators=1000, learning_rate=0.01, max_depth=6, subsample=0.8, colsample_bytree=0.8, and random_state=42. These hyperparameters were selected to balance model complexity and generalization. The model is trained for 1000 boosting rounds.

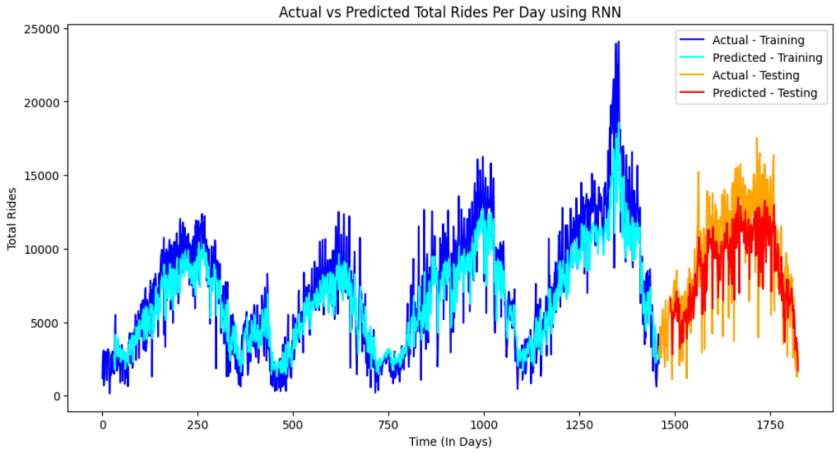
4. Model Evaluation
   Each of the models above was evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) metrics. These metrics provided quantitative measures to identify the best-performing model.
5. Model Training and Fine-tuning
   The selected model was further trained using our Bluebikes dataset, this time making predictions for rides across specified districts as opposed to rides across the entire dataset. Hyperparameters were fine-tuned to improve accuracy and robustness. Additionally, various data augmentation techniques were tested to enhance model performance.
6. Prediction Visualization
   To even better assess the models' predictive capabilities, the models' predicted values were visualized alongside the actual values given in the dataset. These visualizations provided a more comprehensive overview of our results, facilitated comparison, and demonstrated each model's effectiveness in capturing demand patterns.

Results



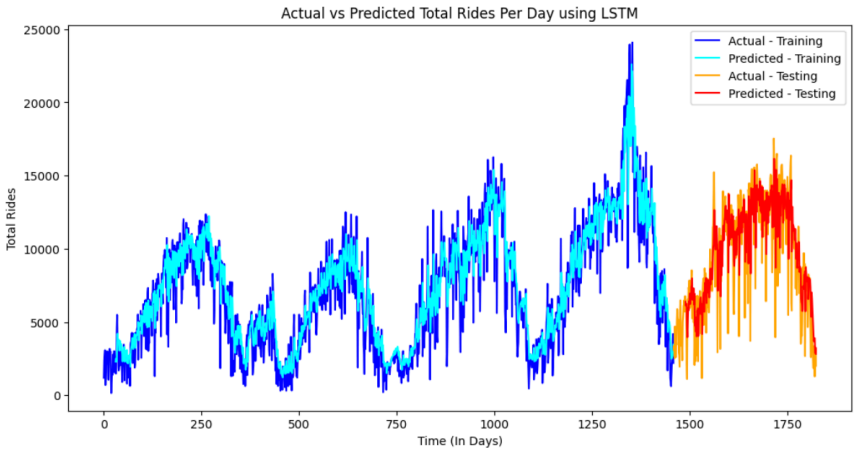

```
Train RMSE: 1755.85
Test RMSE: 3872.83
Train MAE: 1244.88
Test MAE: 3290.10
```

Exponential Smoothing Model Prediction Visualization + Evaluation Metrics



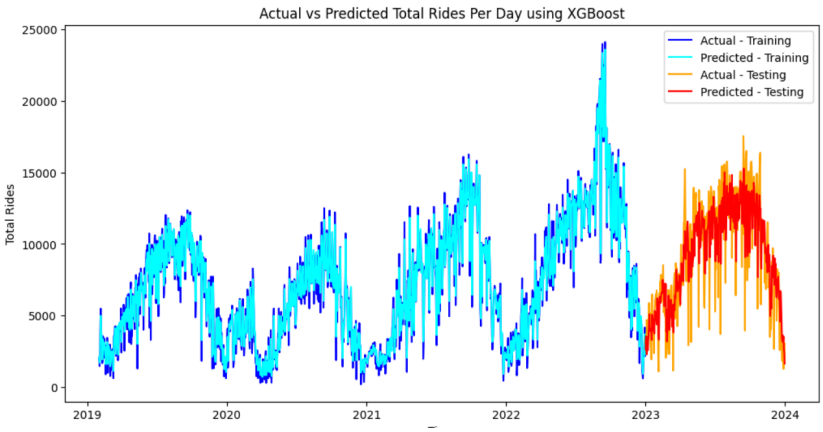

```
Train RMSE: 1871.10
Test RMSE: 2677.72
Train MAE: 1425.19
Test MAE: 2182.28
```

Recurrent Neural Network Prediction Visualization + Evaluation Metrics




```
Train RMSE: 1714.14
Test RMSE: 2454.30
Train MAE: 1238.43
Test MAE: 1788.63
```

Long Short-Term Memory Prediction Visualization + Evaluation Metrics

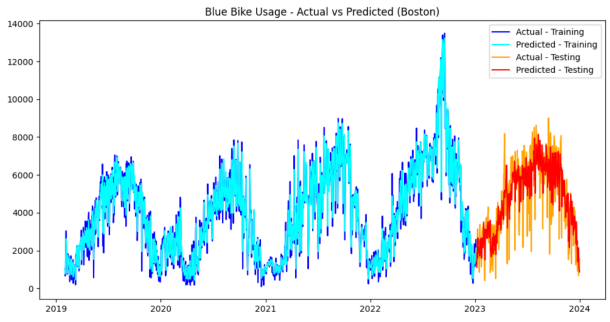


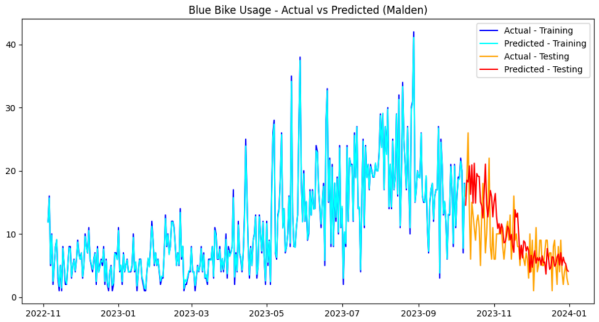

```
Train RMSE: 411.50
Test RMSE: 2303.17
Train MAE: 309.83
Test MAE: 1735.34
```

XGBoost Prediction Visualization + Evaluation Metrics

The Exponential Smoothing baseline model was significantly outperformed by all three of the machine learning models. It achieved a test RMSE of 3872.83 and a test MAE of 3290.10,

highlighting its limited predictive capability. Among the machine learning models tested—XGBoost, LSTM, and RNN—XGBoost demonstrated the best performance, surpassing both LSTM and RNN in accuracy. While LSTM performed better than RNN, as expected, it was surprising that XGBoost achieved the highest accuracy, given that LSTMs are generally considered more suitable for large-scale time-series forecasting tasks. Specifically, XGBoost achieved the lowest test RMSE (2303.17) and test MAE (1735.34), compared to LSTM's RMSE of 2454.30 and MAE of 1788.63, and RNN's RMSE of 2677.72 and MAE of 2182.28. We believe XGBoost may have performed best due to its ability to quickly and effectively handle structured data with predictable patterns, which aligned well with the nature of our data.






```
--- Forecasting for District: Boston ---
Train RMSE: 247.84
Test RMSE: 1227.64
Ratio: 4.95
```

```
--- Forecasting for District: Malden ---
Train RMSE: 0.35
Test RMSE: 4.75
Ratio: 13.43
```

For district-level predictions, we switched from a split date (2023-01-01) to a manual 80/20 train-test split. This adjustment was necessary because not all districts had data extending back to January 2019, requiring careful handling of data availability. XGBoost excelled at predicting demand across districts, particularly in areas with extensive trip data. For instance, in Boston, which had data from January 2019, XGBoost exhibited a smaller test RMSE/train RMSE ratio of 4.95, compared to 13.43 for Malden, where data only began in November 2022.

```
Train RMSE: 1121.70
Test RMSE: 1424.90
```

Prediction Scores for Boston using LSTM

```
Train RMSE: 6.33
Test RMSE: 4.17
```

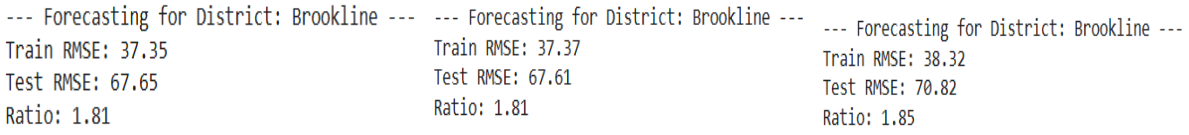
Prediction Scores for Malden using LSTM

For an additional comparison, we evaluated the prediction scores of the LSTM model against those of XGBoost. While the LSTM performed worse overall across the districts, it consistently produced lower test/train RMSE ratios compared to XGBoost. For instance, the LSTM achieved ratios of approximately 1.27 for Boston and 1.52 for Malden, whereas XGBoost recorded ratios of 4.95 and 13.43, respectively. These results suggest that, even without fine-tuning, LSTMs (and RNNs) are inherently better at mitigating overfitting and tend to make more consistent predictions across districts.

To fine-tune the XGBoost model, we optimized its hyperparameters to achieve the best results. The final model parameters were:

- max_depth: 3 (larger values led to overfitting)
- learning_rate: 0.01 (lower rates performed worse in districts with limited data)
- n_estimators: 1000
- subsample: 0.8
- colsample_bytree: 0.8

Fine-Tuned XGBoost Model Brookline Predictions With and Without Data Augmentation



Without | With Noise Injection | With Scaling

We also experimented with two data augmentation techniques: noise injection and scaling. For noise injection, a noise factor of 0.02 was applied to the target variable Total Rides. While the improvements were minor (e.g., Brookline's test RMSE improved slightly from 67.65 to 67.61), this approach demonstrated marginal gains in performance. Conversely, scaling the target variable with a factor of 0.05 resulted in worse predictions, as Brookline's test RMSE increased to 70.82. Based on these findings, noise injection was deemed the most effective augmentation strategy for optimizing model performance.

Conclusions

From our analysis, we concluded that XGBoost outperforms recurrent models like LSTM and RNN in predicting Bluebikes usage and demand across Greater Boston districts. This result challenges the assumption that LSTM, typically preferred for large-scale time-series forecasting, would excel in this domain. While XGBoost achieved the lowest RMSE and MAE across all models, LSTM demonstrated superior generalization with lower test/train RMSE ratios, indicating its consistency in predicting demand, even in districts with less historical data. Our team learned valuable lessons from this experience, particularly the importance of robust data preprocessing when integrating multiple datasets with varying structures. We found that fine-tuning models and incorporating domain knowledge—such as seasonality and district-level trends—are crucial for high performance. Our exploration of data augmentation highlighted the nuanced impact of such methods: noise injection yielded marginal improvements, while scaling detracted from performance. These findings underscore the value of predictive analytics in optimizing Bluebikes operations and providing actionable insights for better resource allocation. Future research could explore hybrid models that combine the strengths of XGBoost and LSTM for even greater predictive accuracy.