Optimizing NYC Bike Share: Hourly Demand Forecasting with Big Data

CSC5003 Data Exploration and Analysis Project

Chenwei Wan

Télécom Paris, Institut Polytechnique de Paris

February 4, 2025

Table of Contents

Bike-Sharing System

- Bike-sharing systems are widely adopted worldwide, particularly in major cities like New York City and Paris
- Users can *rent* or *return* a bike at any station by swiping their membership, generating a rental or return record that includes their user ID, bike ID, station name, and timestamp

Figure: Citi Bike pay station in Midtown Manhattan (Wikipedia)

Demand vs Supply

1 Introduction

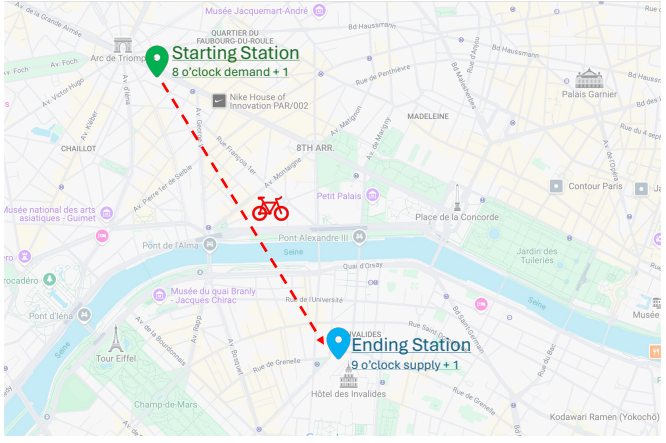


Figure: Demand & supply of bike sharing stations.

Jamming Situation

- Some stations may face a shortage of bikes
- While others may receive more incoming bikes than the demand, potentially exceeding their capacity
- To address this issue, system operators continuously reposition bikes among stations using trailers
- **However**, reacting only after stations become jammed or starved is often **too late**, especially when large-scale repositioning is required

Figure: Transporting bikes on a bicycle trailer (Wikipedia)

Business Objective

- Therefore, **forecasting** the demand and supply of bike stations is a promising solution
- By predicting stations with an excess supply and those experiencing shortages, bikes can be **proactively** transported before jamming situations occur

Challenges of Forecasting

- The demand and supply of individual bike stations can fluctuate significantly
- Impact of a combination of complex factors, including *time*, *weather conditions*, and *events*

Our Big Data System

- We aggregate **multiple data sources**, including *trip data* generated by city bike users, *hourly weather condition data*, and *city geometric data*
- By cleaning and enriching large-scale data, we are able to train **a time series prediction model** *offline*, which demonstrates superior performance across several metrics
- We develop **an interactive dashboard** using D3.js to visualize the demand and supply dynamics of different station clusters in the bike-sharing system

Table of Contents

2 Data

Data Source

2 Data

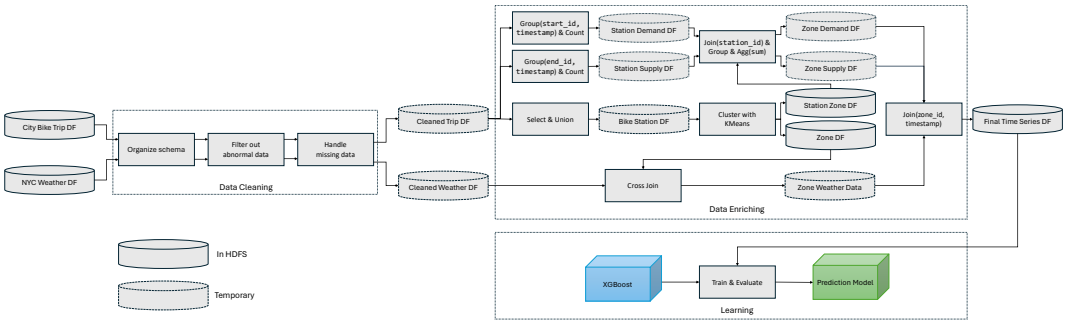
- Complete 2023 **trip data** released by *Citi Bike* Company (1.54GB, CSV)
- **Hourly weather data** of New York City in 2023 from *Kaggle* (1.1MB, CSV)
- **Geometric data** for NYC and Jersey City in GeoJSON format from *GitHub and the Jersey City government database*

Table of Contents

3 System Pipeline

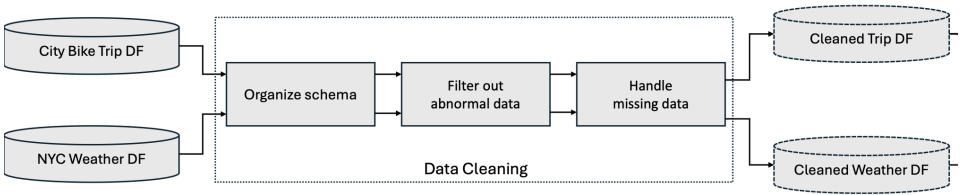
Overview

3 System Pipeline



Stage 1: Data Cleaning

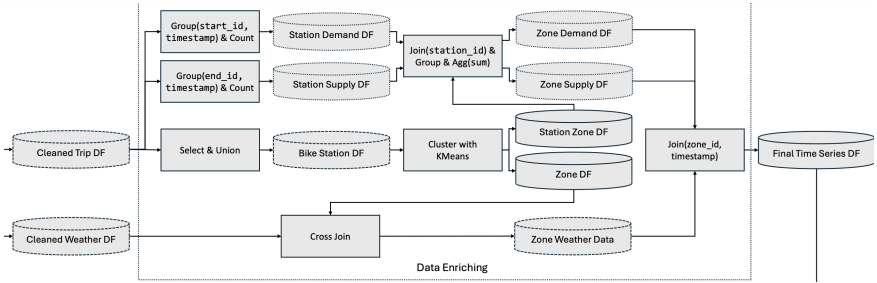
3 System Pipeline



Stage 2: Data Enriching

3 System Pipeline

- We **cluster** the thousands of bike stations into 50 zones using the **K-Means** algorithm based on their longitudes and latitudes
- This aggregation *regularizes* bike usage within each zone, thereby improving prediction accuracy



Stage 2: Data Enriching

3 System Pipeline

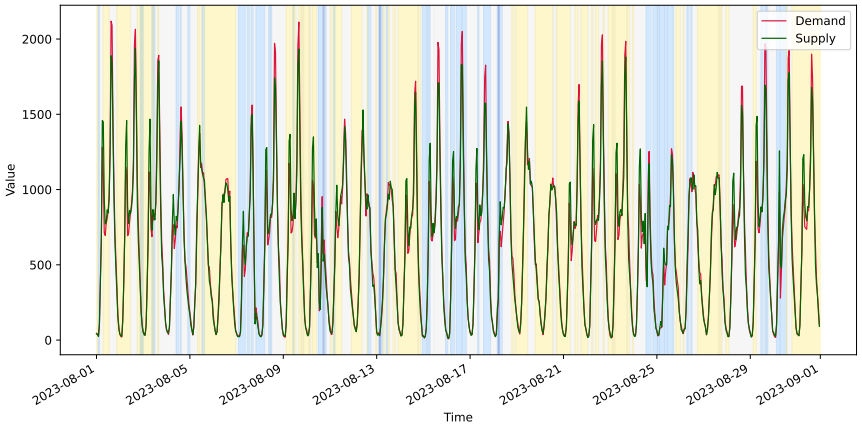


Figure: Demand & Supply time series sample for zone 27 in August 2023.

Stage 2: Data Enriching

3 System Pipeline

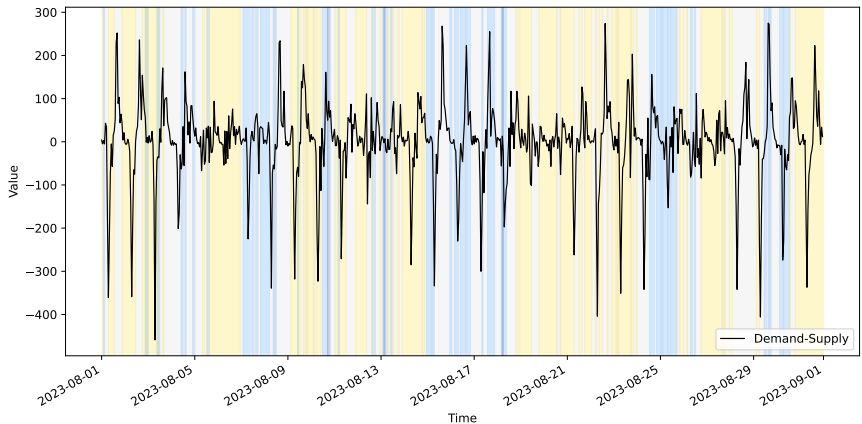


Figure: Demand-Supply difference time series sample for zone 27 in August 2023.

Stage 3: Learning

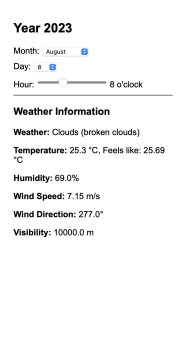
- With the time series data, we train an **XGBoost** model
- The features used in the forecasting model:
  - Zone Features: `longitude`, `latitude`
  - Weather Features: `temperature`, `visibility`, `weather_main_index`
  - Temporal Features: `hour`, `day_of_week`, `is_weekend`
  - Time Series Features: `lag_1`, `lag_24`, and `lag_168`, `rolling_3h_mean`, `delta_1h` ...

	Full Features	w/o Weather
RMSE↓	**4.8112**	4.8921
MAE↓	**0.8511**	0.9063
R^2↑	**0.9815**	0.9809
MedAE↓	**0.1644**	0.1831

Interactive Visualization

3 System Pipeline

- Backend: Scala + Akka HTTP + Spark
- Frontend: D3.js



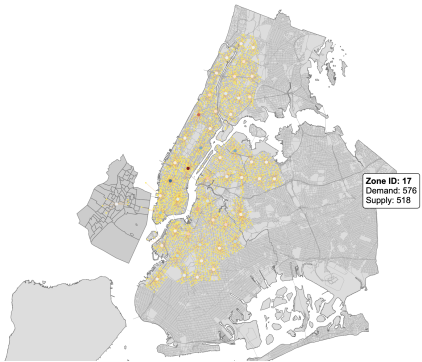


Table of Contents

4 Demo

Table of Contents

5 Summary

Future Work

- We can use **Spark Streaming** to ingest real-time trip and weather data, and process them into time series format using the pipeline proposed above. Predictions can then be made using our offline-trained XGBoost model.
- We can periodically retrain the XGBoost model on newly accumulated streaming data within the Spark framework to adapt to recent trends.