LARGE SCALE COMPUTING ON THE CLOUD
DIVVY BIKE TRIP DATA

Group 6

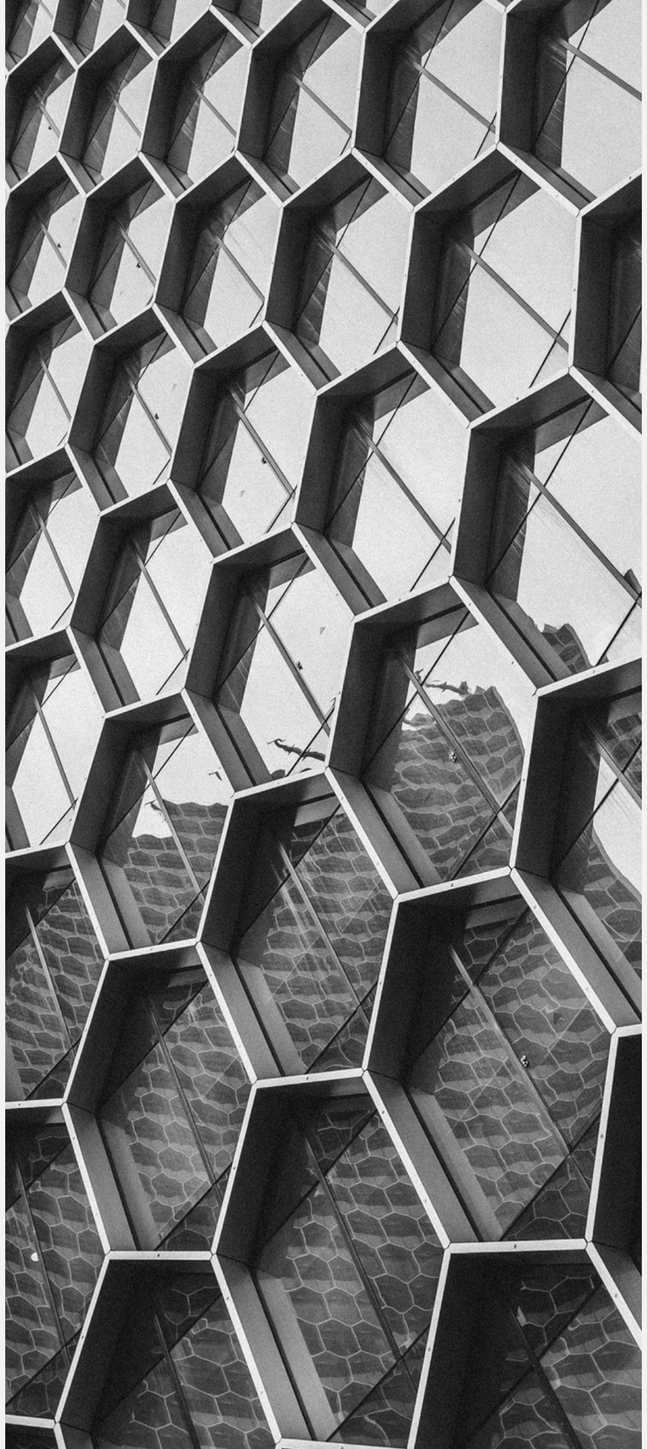
Calis Nguyen, Tianjin Duan,
Emanuel Telles Chaves,
Jinge Zhou, Yizhi Wang, Tianyi Zhao

Business Context + Pain points

Client: Divvy Operations — Chicago's bike-share operator with 600+ stations

Current Problem: Demand-Supply Mismatch
Morning rush: Residential stations empty → riders can't start trips
Evening commute: Downtown stations full → bikes can't be returned
Result: Potential rides lost due to availability issues

The Opportunity
Shift from reactive to predictive, forecasting demand 15 minutes ahead
Smarter rebalancing, deploying trucks to prevent stockouts before they happen
Data-driven operations: 5.9 million trips/year = rich historical patterns



Dataset Overview

Divvy Bike-Sharing Trip Dataset
13 attributes, 5.9M rides
Time range: Nov 2024 – Nov 2025

Add features:

1. **season: winter(Dec,Jan,Feb) spring(Mar,Apr,May) summer(Jun,Jul,Aug) fall(Sep,Ort,Nov)**
2. **day_type(holiday, weekend, weekday)**
3. **peak_period(morning peak 7-9am, evening_peak 16-18pm, night_low 1-5am, others)**
4. **weekday_name**

```
Index(['ride_id', 'rideable_type', 'started_at', 'ended_at',
       'start_station_name', 'start_station_id', 'end_station_name',
       'end_station_id', 'start_lat', 'start_lng', 'end_lat', 'end_lng',
       'member_casual', 'day_type', 'peak_period', 'weekday_name', 'season'],
      dtype='object')
```

5 V

Volume

13 months of trip data (~5–6M rides)
requiring batch ingestion and scalable processing.

Variety

Mixed data types (time, geolocation, station IDs, bike/member categories)
+ engineered fields (season, peak, ride_length).

Velocity

Monthly release cycles and near real-time operational relevance
automated retrieval and concatenation.

Veracity

Data quality risks (missing stations, invalid lat/lon & timestamps, unrealistic durations)
addressed via structured cleaning rules.

Value

Enables predictive, data-driven bike rebalancing.
Supports station planning, fleet allocation, and peak-hour management.
Reduces lost rides via short-term demand forecasting.
Leverages historical data to segment users and optimize seasonal strategies.
Turns large, multi-type data into actionable operational insights.

Methods - Forecasting Bike Demand for Rebalancing

1. Goal (Prediction Target)
   - Predict next 15-minute pickups per station
     - Label: y_next = pickups at next 15-min bucket (/ station)

2. Data Setup (from our pipeline)
   - Aggregation:
     - trips → (station_id, time_bucket=15min) → pickups
   - Scale controls：
     - Use Top 200 busiest stations (by total pickups)
     - Optional training downsample to 500k rows

3. Train/Test Split
   - Train: 2024-11-01 → 2025-09-30 23:45
   - Test: 2025-10-01 → 2025-11-29 22:45

4. Baselines (Operational Benchmarks)
   - Last-bucket baseline:
     - pred_last_bucket = pickups(t)
   - Last-week same time:
     - pred_last_week = pickups(t − 7 days) using lag 672

5. ML Model (Gradient-Boosted Trees)
   - LightGBM regression
     - Categorical: station_id, season)

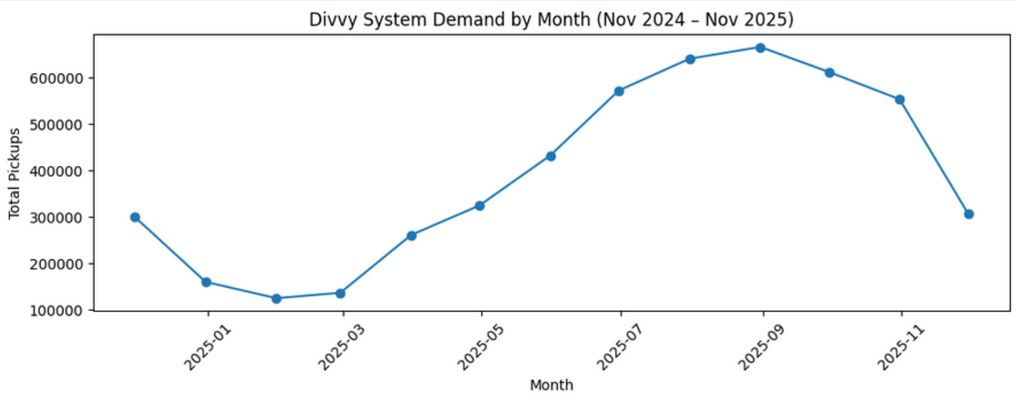
6. Feature Engineering
   - Time: hour, day-of-week, weekend, month/season
   - Lags: 1, 2, 4, 672
   - Rolling mean: 4, 8 (past-only)

7. Evaluation Metrics
   - MAE (primary; average absolute error in pickups)
   - RMSE (penalizes large misses; for peak-demand rebalancing)

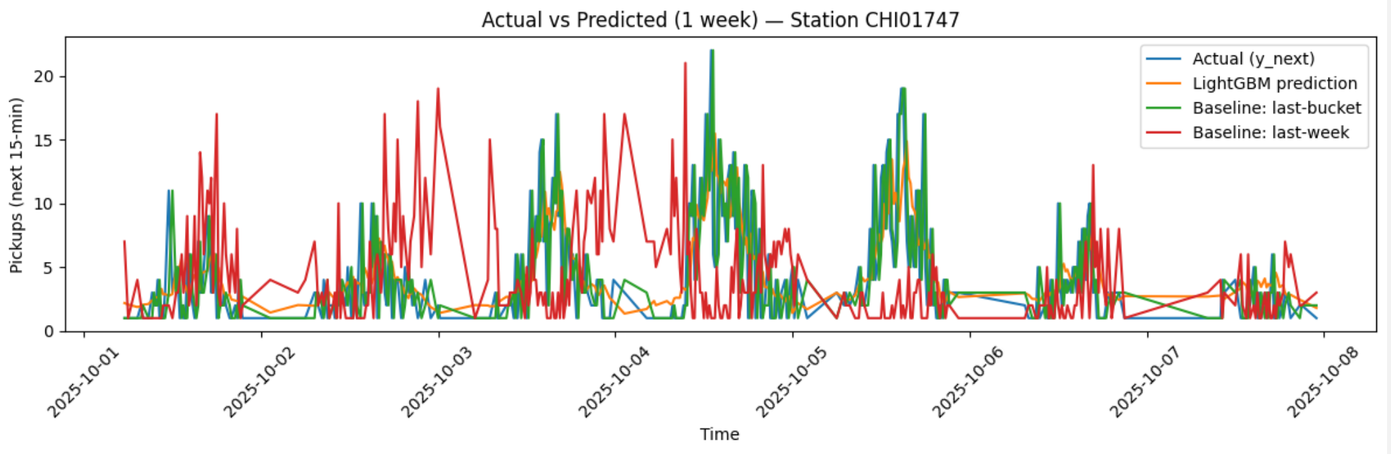
	model	n	MAE	RMSE
2	pred_model	249421	0.878846	1.475442
0	pred_last_bucket	249421	0.923603	1.774006
1	pred_last_week	249421	1.497015	4.304283

```
MAE improvement vs last-bucket: 4.85%
RMSE improvement vs last-bucket: 16.83%
```

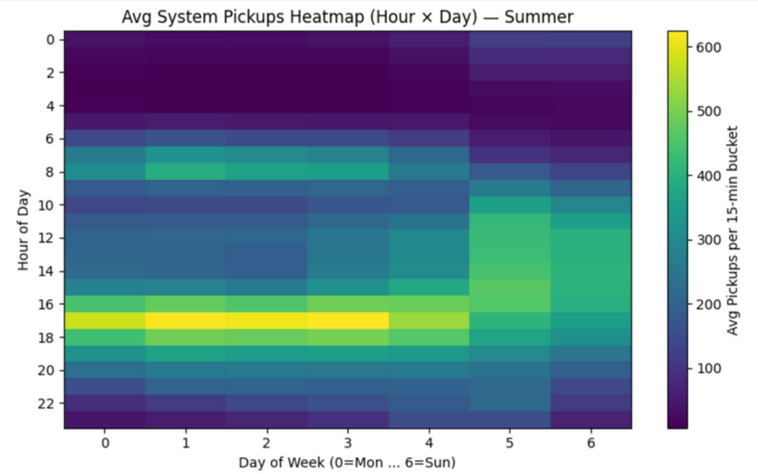
Methods - Seasonality & Model Fit

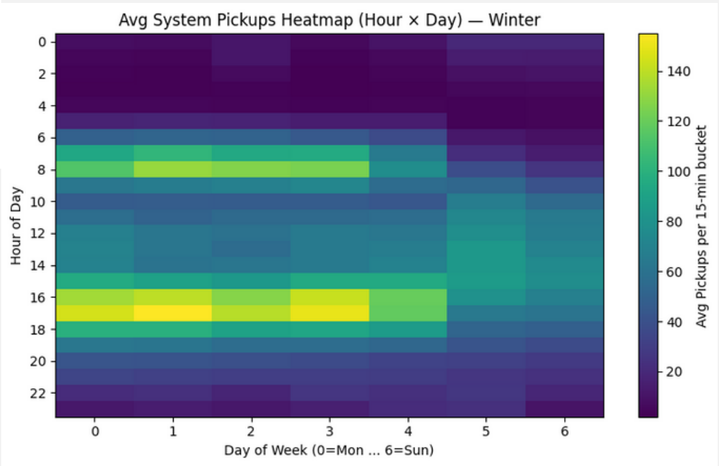


System demand is seasonal (summer peak, winter low).

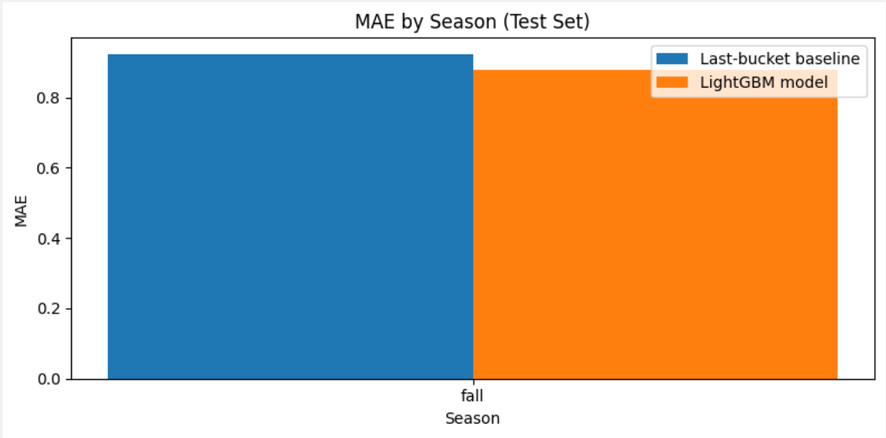


Model tracks short-term fluctuations better than baselines





Demand concentrates at specific hours/days; patterns shift by season



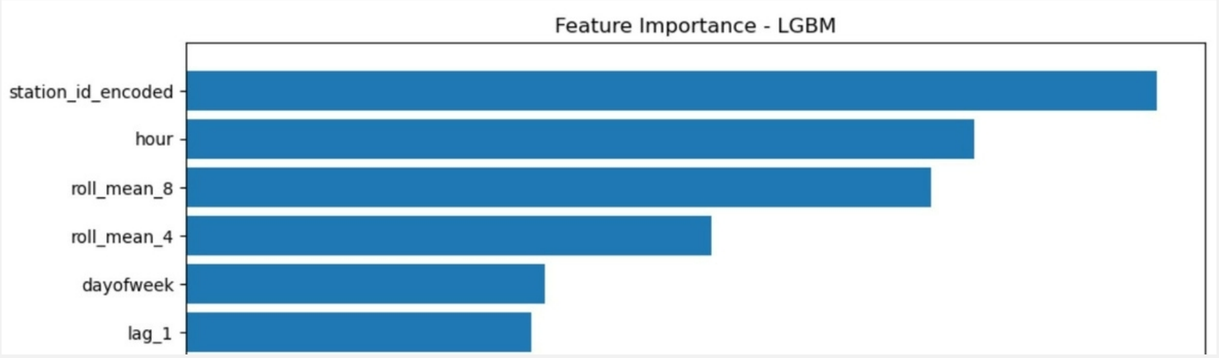
Model improves accuracy vs baseline across seasons.

AWS Pipeline - LGBM

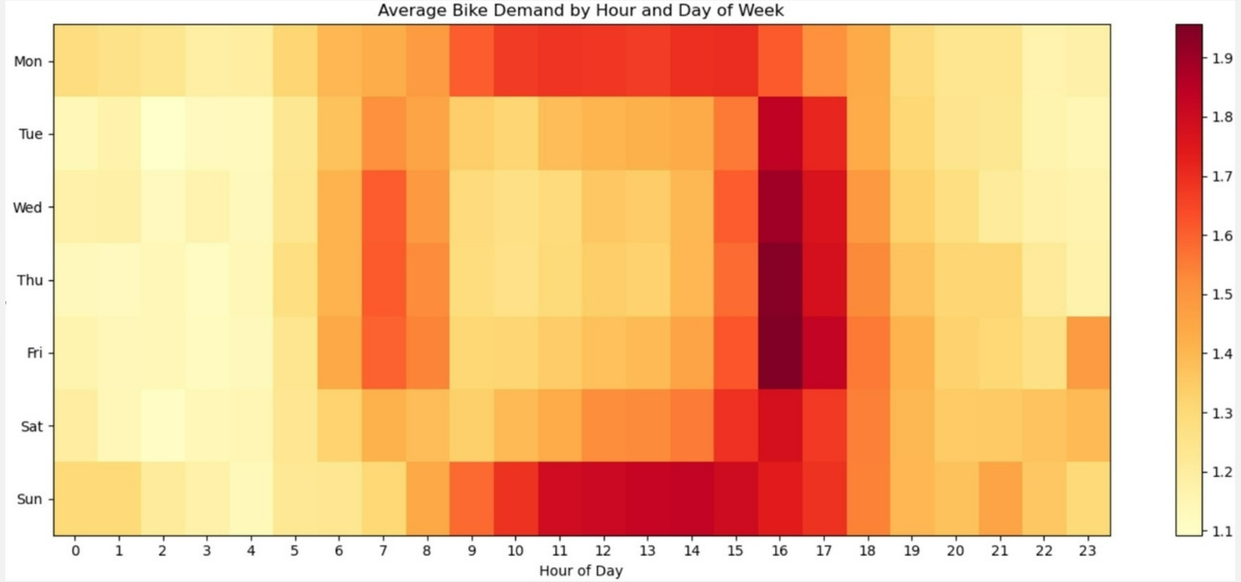
S3

EMR

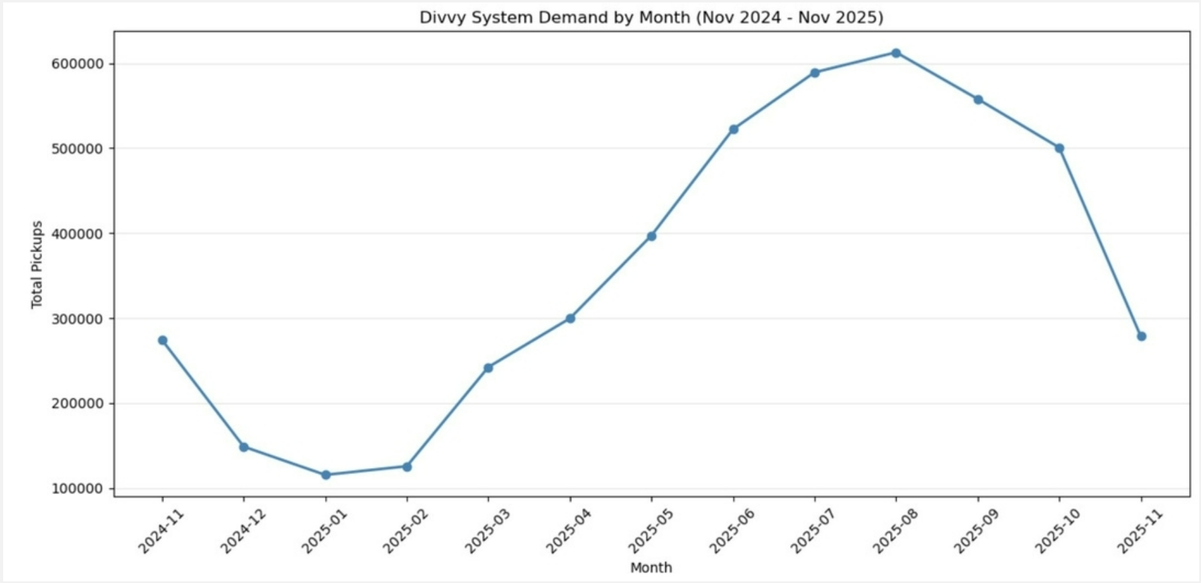
SageMaker AI



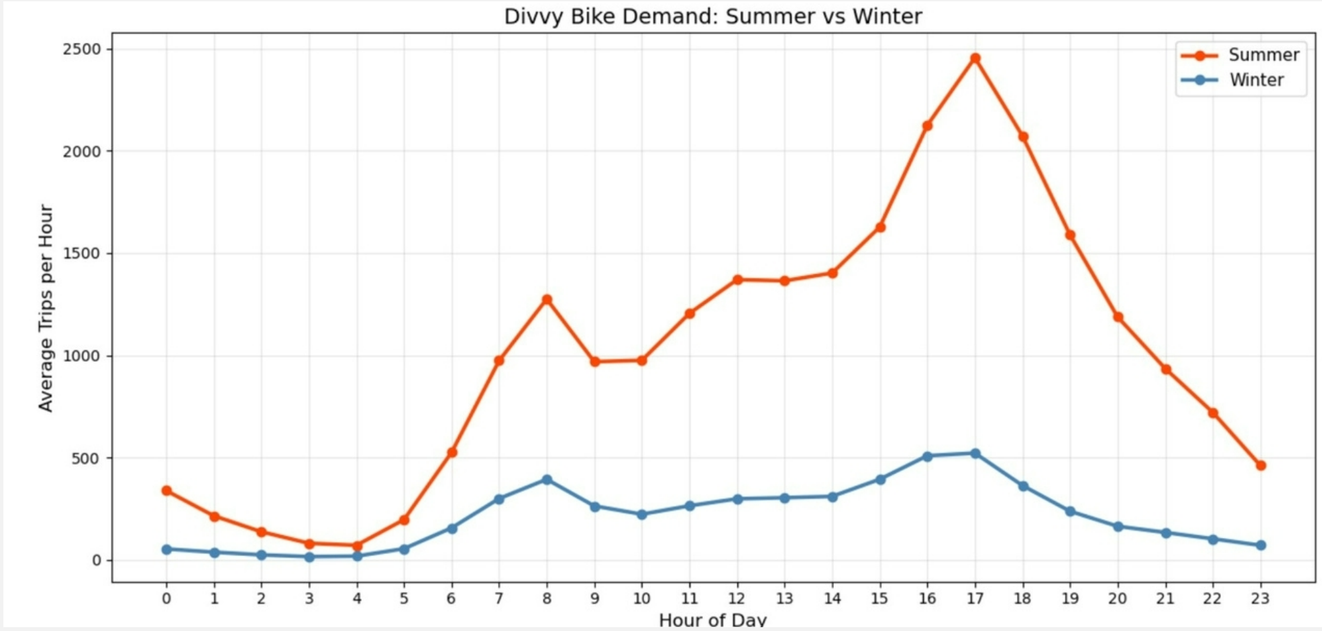
Station demand varies greatly.



Peak Hours: 3-5 pm **Peak Days: Sun, Mon**



Hourly and monthly demand is seasonal



Results

- **AWS outperformed Colab—more data, better results**
- **MAE improved ~5% vs baseline (0.6482 vs 0.6539)**
- **RMSE improved ~17%—fewer big misses at peak hours**
- **Model beats both baselines across the board**

Metric	Colab	AWS
Model MAE	0.8788	**0.6482** ✅
Model RMSE	1.4754	**0.9775** ✅
Last-Bucket MAE	0.9236	0.6539
Last-Bucket RMSE	1.7740	1.2554
Last-Week MAE	1.4970	0.8862
Last-Week RMSE	4.3043	1.5597
Improvement %	4.85%	0.87%

	model	n	MAE	RMSE
2	pred_model	249421	0.878846	1.475442
0	pred_last_bucket	249421	0.923603	1.774006
1	pred_last_week	249421	1.497015	4.304283

```
MAE improvement vs last-bucket: 4.85%
RMSE improvement vs last-bucket: 16.83%
```

Factor	Colab	AWS
Stations used	Top 200 only	**All stations**
Training rows	500k (downsampled)	**All rows**
Data mix	High-demand stations only	Mix of high and low demand

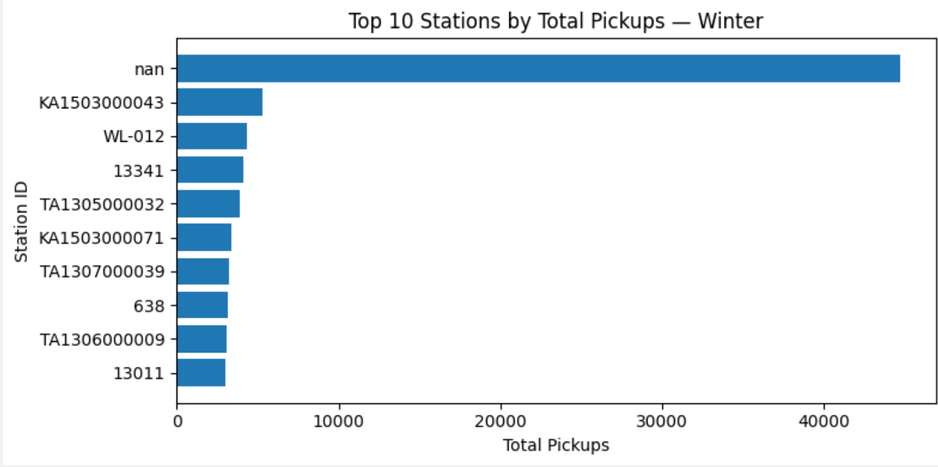
Limitations & Further Steps

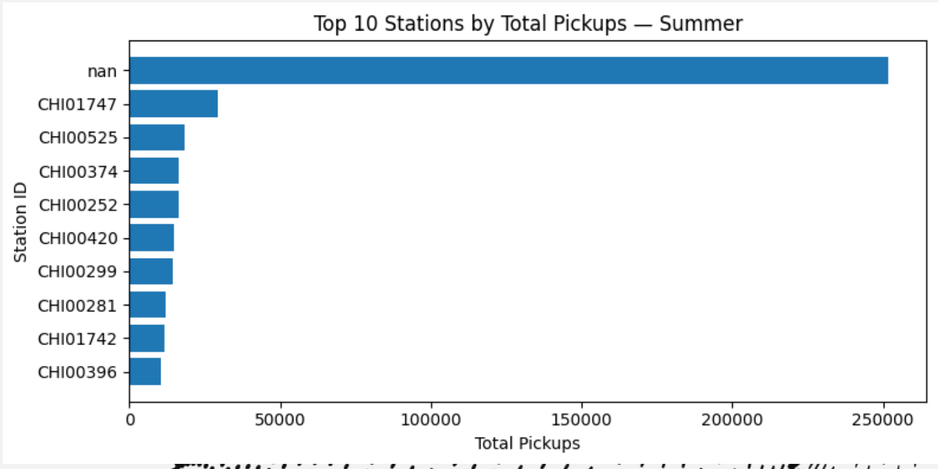
Limitations:

- No weather data (rain and temperature strongly affect ridership)
- No special events data (concerts, sports games, festivals)
- Model trained on Top 200 stations only
- Single-model approach with no per-station customization
- Test period limited to fall season (Oct–Nov 2025)
- station_id missing values observed

Further Steps:

- Integrate real-time weather API
- Add event calendar data
- Extend to full 600+ station network
- Implement station-cluster models
- Deploy as real-time streaming pipeline
- A/B test with actual rebalancing operations





References & Appendix

Divvy Bikes. (n.d.). System data. Divvy. Retrieved October 24, 2023, from https://divvybikes.com/system-data

Code for Data Description, Data Cleaning and Methods:
https://colab.research.google.com/drive/1vqD_tezqQRCpdNZuluM7uSNajjBz2S0U?usp=sharing

AWS SageMaker and Spark Methods:
https://drive.google.com/file/d/1tEG0JoMolV6AII9k9V9CyBoVSAroPS3V/view?usp=drive_link
https://drive.google.com/file/d/1kbNt0lKizYE3i1qUgFoASa5T4QWIj-Fe/view?usp=drive_link

THANKS FOR WATCHING