Dynamic Pricing Engine for Urban Parking Lots

Author: Anvi Aggarwal

Program: Summer Analytics 2025 – Consulting & Analytics Club

Project Overview

This project solves the issue of wasteful parking lot use in cities by introducing a smart, real-time Dynamic Pricing Engine. From a 73-day dataset (collected at 30-minute intervals) from 14 parking lots, the task is to maximize space utilization, minimize congestion, and enhance pricing equitability on the basis of demand drivers such as occupancy, traffic, queue length, vehicle type, and competition.

Technologies Used

- Python, Pandas, NumPy (core logic and data handling)
- Bokeh (interactive visualization)
- Google Colab (development environment)
- GitHub (version control)
- Pathway (simulated or actual streaming pipeline)

Dataset Summary

The dataset includes:

- SystemCodeNumber, Capacity, Occupancy, QueueLength
- TrafficConditionNearby, VehicleType, IsSpecialDay
- Latitude, Longitude, LastUpdatedDate, LastUpdatedTime

A Timestamp column was created by combining the date and time columns, and traffic levels were mapped to numeric values.

Implemented Models

Model	Description
Model 1: Linear Pricing	Adjusts price directly based on occupancy using a simple linear formula.

Model 2: Demand-Based Pricing	Factors in queue length, traffic conditions, vehicle type, and special day indicators for a more dynamic and realistic pricing strategy.
Model 3: Competitive Pricing	Extends demand-based pricing by incorporating competition from nearby lots using geospatial logic and provides rerouting suggestions when appropriate.

Demand Function

Demand Score =

0.012 × (Occupancy % × Base Price) +

0.07 × QueueLength +

0.004 × (Traffic - 50) +

2.5 × IsSpecialDay

All prices are clipped between \$5 and \$20 (0.5× and 2× of base price \$10).

Assumptions

- Vehicle type influences parking behavior
- Traffic levels correlate with demand
- Proximity is calculated using latitude and longitude
- Special events increase parking urgency and price tolerance

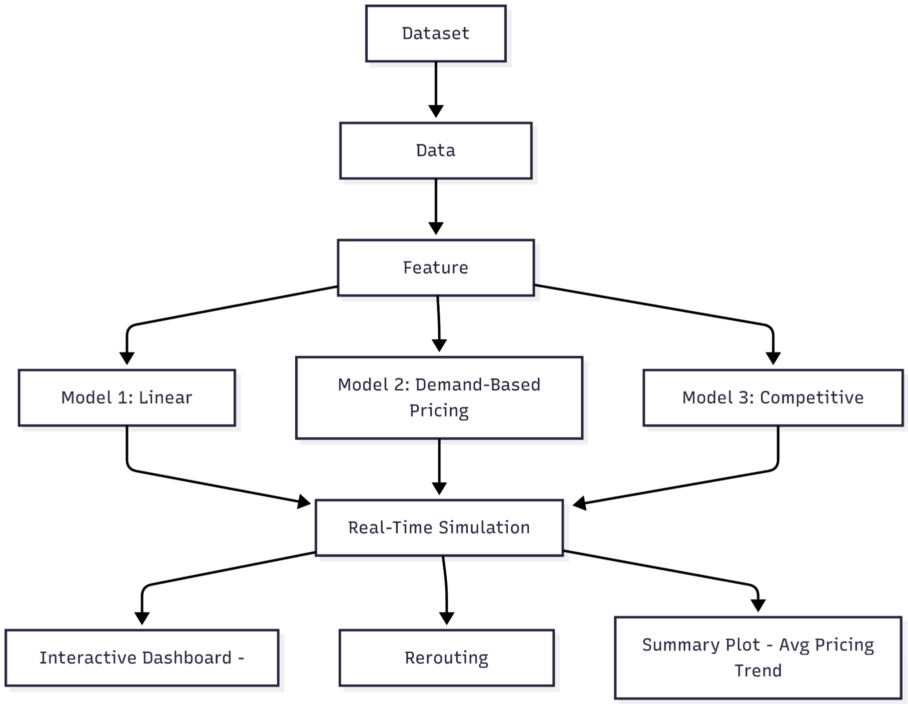
Price Behavior with Demand & Competition

- Model 1 reacts only to occupancy.
- Model 2 includes multiple variables to price more responsively.
- Model 3 blends Model 2's result with nearby average prices:
- Final Price = 0.7 × Model 2 + 0.3 × Nearby Avg Price

System Architecture

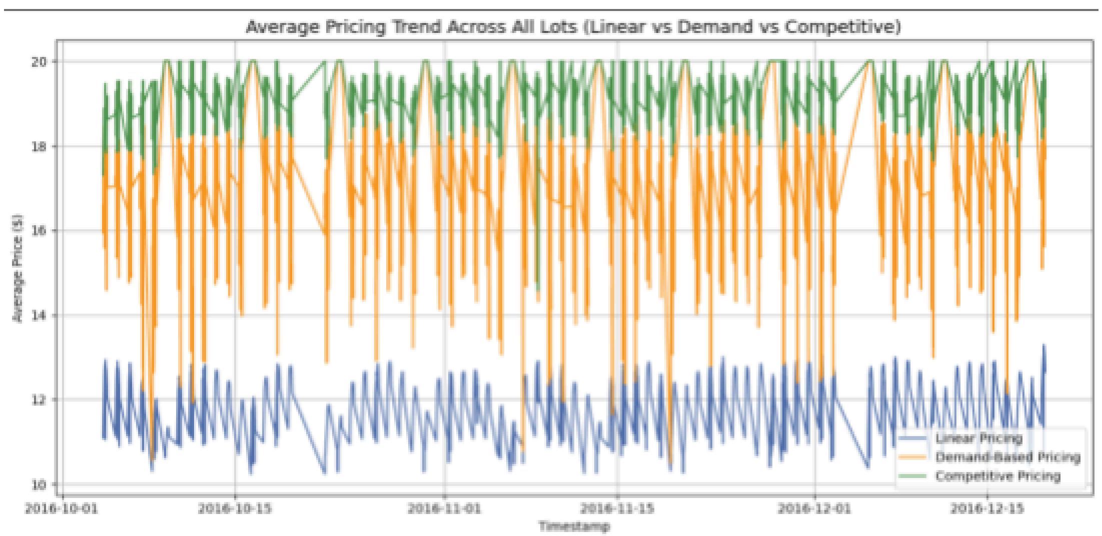
The architecture consists of the following stages:

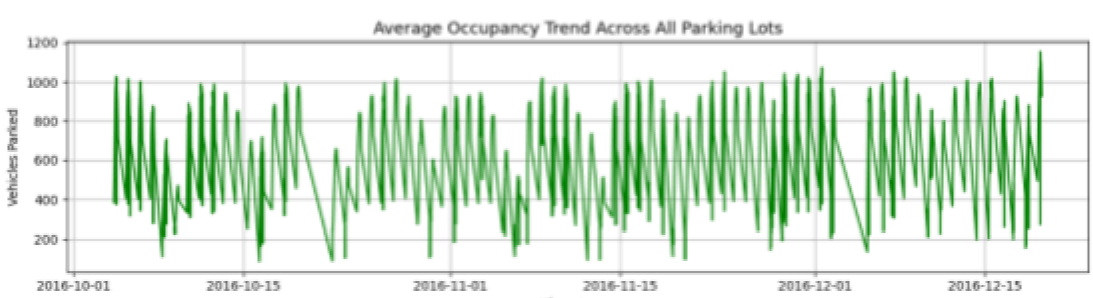
1. Data ingestion and preprocessing
2. Feature engineering
3. Model-based pricing (linear, demand, competitive)
4. Simulation over timestamp batches
5. Interactive dashboard and reroute recommendations
6. Summary trend visualizations for model comparison



Summary Visualization

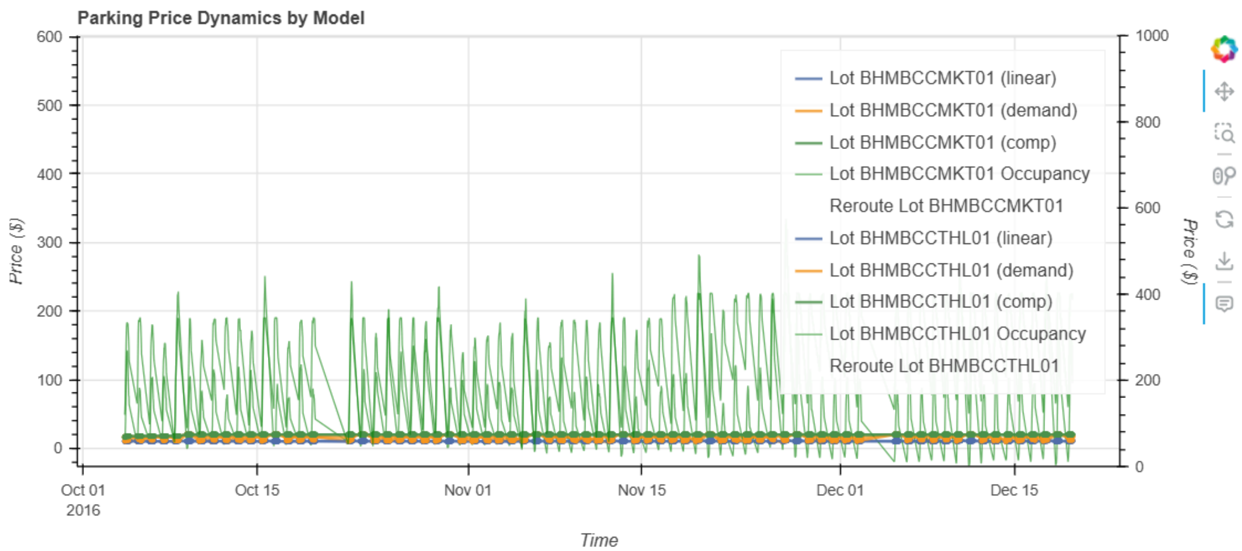
A summary plot compares the pricing behavior of all three models over time, highlighting how dynamic models respond more effectively to real-world changes compared to the linear baseline. The competitive model showed strategic increases and rerouting when nearby lots were congested.





Interactive Dashboard and Rerouting

An interactive dashboard built with Bokeh allows users to visualize pricing changes in real-time for selected lots. Rerouting suggestions are generated based on geospatial proximity and pricing advantages when the current lot exceeds threshold occupancy or cost.



GitHub Repository

🔗 [Click here to view Github Repo]

Conclusion

This project emulates a dynamic pricing engine that responds in real-time to demand and competition. Each model increases the realism and complexity. The ultimate solution incorporates rerouting, competitive awareness, and demand sensitivity — all implemented in Google Colab using explainable Python logic.