Dynamic Pricing for Urban Parking Lots

Capstone Project Report - Summer Analytics 2025

Overview

Urban parking spaces are a limited resource, and static pricing models often result in inefficient usage. This project aims to implement a **dynamic pricing engine** for urban parking lots that adjusts pricing in real-time based on multiple factors such as occupancy, traffic conditions, special days, and competitor prices.

Objectives

- Design a pricing engine that updates prices in real-time.
- Build three levels of pricing models:
- Baseline Linear Pricing
- Demand-Based Dynamic Pricing
- Competitive Pricing based on geographical proximity
- Visualize pricing evolution and comparison with competitor lots using Bokeh.

Dataset

- **Source**: CSV dataset containing 14 parking locations, sampled over 73 days.
- **Fields** include: `occupancy`, `queue`, `vehicle_type`, `traffic`, `special_day`, `latitude`, `longitude`.
- **Note**: Since `lot_id` was not provided, synthetic IDs were generated using unique `(latitude, longitude)` combinations.

Step-by-Step Methodology

1. Data Preprocessing

- Timestamps were parsed and formatted correctly.
- Unique parking lot identifiers were created by combining latitude and longitude.
- Initial data preview and inspection ensured that values were usable for modeling.

2. Model 1: Baseline Linear Pricing

A simple linear model increases the price based on occupancy:

```
new_price = previous_price + alpha * (occupancy / capacity)
```

- `alpha` is a tuning parameter that controls the sensitivity to occupancy.
- Used to set a reference baseline.

3. Model 2: Demand-Based Dynamic Pricing

A weighted model incorporating several features:

```
Demand =
  α * (occupancy / capacity) +
  β * queue -
  γ * traffic +
  δ * special_day +
  ε * vehicle_weight
```

- Demand is normalized with a tanh function.
- Price bounds: between 0.5x and 2x base price.

4. Model 3: Competitive Pricing Model

- Based on proximity using geodesic distance (1 km radius).
- Adjust prices depending on relative pricing with competitors:
- If full and competitors are cheaper: reduce price.
- If competitors are more expensive: increase price.
- Logic ensures smoother transitions and real-world competitiveness.

Real-Time Simulation

- **Simulated over 10 time steps**, each 30 minutes apart.
- For each parking lot, simulated occupancy and pricing updates.
- Data streamed into real-time line plots.

Bokeh Visualizations:

Each parking lot features:

- **Blue Line**: The lot's current dynamically computed price.
- **Red Dashed Line**: Average competitor price within 1 km radius.

All plots update in real-time, simulating a dynamic environment.

Example Output (Simulation Summary)

- Base Price: \$10.00
- Minimum Price (due to low demand): \~\$5.00
- Maximum Price (due to peak demand/competition): \~\$20.00

Plots show:

- Gradual increase in price as occupancy and demand grow.
- Competitive responses when nearby lots are cheaper.

Tools Used

- **Python** (NumPy, Pandas)
- **Bokeh** for interactive visualizations
- **Geopy** for distance calculations
- **Pathway (planned)** for real-time data ingestion and deployment

---

Future Improvements

- Integrate **Pathway** to simulate real-time ingestion pipeline.
- Introduce **rerouting logic** to nearby lots.
- Build **interactive dashboard** with filters and time control.

Conclusion

This dynamic pricing engine leverages historical patterns, real-time data, and competitive intelligence to price urban parking efficiently. The modular model approach allows gradual sophistication, and the visualization confirms that pricing behavior is responsive and smooth.

Appendix

- Dataset columns: `timestamp`, `occupancy`, `queue`, `vehicle_type`, `traffic`, `special_day`, `latitude`, `longitude`
- Visuals exported via Bokeh (interactive in notebook)
- Code available in accompanying Colab notebook