Dynamic Pricing for Urban Parking Lots

Objective

This project aims to develop a **real-time dynamic pricing system** for 14 urban parking lots using a variety of real-world signals like vehicle occupancy, queue lengths, traffic conditions, and nearby competition. The goal is to create a smart pricing engine that:

- Reacts to demand in real time
- Makes pricing fair and competitive
- Suggests rerouting when congestion occurs

Dataset Overview

- **Number of Records:** 18,368 entries
- **Time Window:** 30-minute intervals from 8:00 AM to 4:30 PM
- **Duration:** 73 days of data
- **Parking Lots:** 14 unique locations
- **Key Features:**
    - `Capacity`, `Occupancy`, `QueueLength`
    - `VehicleType`, `TrafficConditionNearby`, `IsSpecialDay`
    - `Latitude`, `Longitude`

      `Timestamp` column
    - `LastUpdatedDate`, `LastUpdatedTime` → merged into `Timestamp`

Preprocessing

- Combined date and time into a unified
- Mapped:
    - `VehicleType` * numerical weight: `car=1`, `bike=0.5`, `truck=1.5`
    - `TrafficConditionNearby`: `low=0`, `medium=1`, `high=2`
- Normalized:
    - `OccupancyRate` = `Occupancy / Capacity`

- `QueueLength` and demand features using `MinMaxScaler`
- Computed distances between parking lots using the **Haversine formula**

Pricing Models

Model 1: Linear Pricing Model

➤ Formula:

$$\text{Price}_{t+1} = \text{Price}_t + \alpha \times \left(\frac{\text{Occupancy}}{\text{Capacity}}\right)$$

- **Base Price:** $10
- **α (alpha):** 2
- **Price Range:** [$5, $20]

Purpose:

A simple baseline that increases price linearly as occupancy rises.

Model 2: Demand-Based Pricing

➤ Demand Function:

$$\text{Demand} = \alpha \cdot \text{OccRate} + \beta \cdot \text{Queue} + \gamma \cdot \text{VehicleWeight} + \delta \cdot \text{SpecialDay} - \epsilon \cdot \text{TrafficLe}$$

- **Weights used:**
  - $\alpha = 0.4$
  - $\beta = 0.3$
  - $\gamma = 0.1$
  - $\delta = 0.2$
  - $\varepsilon = 0.3$

➤ Pricing Formula:

$$\text{Price} = \text{BasePrice} \cdot (1 + \lambda \cdot \text{NormalizedDemand})$$

- **λ (lambda):** 1
- **Price Range:** [$5, $20]

Assumptions:

- Higher demand increases price proportionally
- Each vehicle type contributes differently to lot pressure
- Special event days naturally raise demand

Model 3: Competitive Geo-Aware Pricing

➤ Core Logic:

- Identify nearby lots (within 1km using latitude & longitude)
- Adjust price based on:
  - Own occupancy rate
  - Competitor average price

➤ Rules:

```
if occupancy > 90% and avg_competitor_price < own_price:
    price -= 1
elif avg_competitor_price > own_price:
    price += 1
```

- All prices are clamped between $5 and $20

➤ Rerouting Logic:

If a lot is 90%+ full and a cheaper, less crowded lot is nearby → suggest reroute:

```python
SuggestReroute = 1 if conditions met else 0
```

Visualizations

All models were visualized using **Bokeh**, with an interactive dashboard that includes:

- Dropdown to select parking lot (SystemCodeNumber)
- Plot 1: Model 3 Pricing over time
- Plot 2 : Occupancy rate over time
- Plot 3 : Reroute signals as binary bars

These graphs update **dynamically** as a new lot is selected, simulating a real-time urban monitoring tool.

Hover tools show detailed information
Plots are scrollable, zoomable, and real-time style

Assumptions

- Prices are updated every 30 minutes
- Demand is estimated from normalized features
- Nearby competitors are within 1 km only

- Maximum price is $20; minimum is $5
- Queue length is a strong proxy for pressure
- Special days have a +20% demand boost

Conclusion

This project simulates a real-world smart parking system that:

- Dynamically adjusts pricing using intelligent demand signals
- Reacts to nearby competition
- Provides rerouting suggestions to reduce urban congestion

It's scalable, explainable, and deployable – built with Python, Bokeh, and a deep blend of analytics + strategy.