**Dynamic Pricing Engine for Urban Parking Spaces/ Armaan**

**Project Overview**

Urban parking spaces are scarce, in-demand, and often mismanaged due to static pricing models. Fixed rates lead to either overcrowded lots or underutilized capacity. To improve efficiency, this project develops a **real-time dynamic pricing engine** for 14 urban parking lots using historical and live data. The models are designed using **basic economic theory**, **demand estimation**, and built-from-scratch **machine learning logic** using only numpy and pandas.

**Data Description**

The dataset covers:

* **14 parking lots** over **73 days**, sampled every 30 minutes (18 times/day).
* **Features include**:
  + **Occupancy**, **Capacity**, **Queue Length**
  + **Vehicle Type** (Car/Bike/Truck)
  + **Special Day** Indicator (Holiday/Event)
  + **Traffic Conditions Nearby** (mostly missing)
  + **Latitude/Longitude** (used for competition modeling in Model 3)

**Preprocessing Steps**

* Combined LastUpdatedDate and LastUpdatedTime into a Timestamp.
* Converted numerical fields: Occupancy, Capacity, QueueLength, IsSpecialDay.
* Mapped VehicleType to weights: Car = 1.0, Bike = 0.6, Truck = 1.5.
* Calculated OccupancyRate = Occupancy / Capacity.
* Dropped rows with missing or zero capacity.

**Model 1: Baseline Linear Pricing**

A simple model where price increases linearly with occupancy:

Pricet=10+α⋅(OccupancyCapacity)\text{Price}\_{t} = 10 + \alpha \cdot \left(\frac{\text{Occupancy}}{\text{Capacity}}\right)

* α=5\alpha = 5, base price = $10
* Price is bounded between $5 and $20
* **Intuition**: As a lot fills up, it becomes more expensive.

**Model 2: Demand-Based Pricing**

This model uses a **custom demand function** combining real-world features:

Demand=2.0⋅OccupancyRate+1.0⋅QueueLength+0.3⋅IsSpecialDay+0.8⋅VehicleTypeWeight\text{Demand} = 2.0 \cdot \text{OccupancyRate} + 1.0 \cdot \text{QueueLength} + 0.3 \cdot \text{IsSpecialDay} + 0.8 \cdot \text{VehicleTypeWeight}

* Demand is **normalized** between 0 and 1
* Pricing Formula:

Price=10⋅(1+0.6⋅NormalizedDemand)\text{Price} = 10 \cdot (1 + 0.6 \cdot \text{NormalizedDemand})

* Prices again bounded between $5 and $20
* **Improvements over Model 1**:
  + Reacts to more contextual factors
  + Captures demand volatility
  + Allows differentiation across lots and time

**Assumptions Made**

* TrafficConditionNearby was dropped (column had only NaN).
* All lots start with a base price of $10.
* Vehicle types were assumed to take different space (affects pricing).
* No direct competitor pricing data was given — this was not modeled explicitly in Model 2.

**Price Variation Behavior**

* In Model 1, **price increases linearly with fill rate**.
* In Model 2, **price reacts to demand fluctuations**, queue buildup, and special events.
* **High demand** results in prices pushing toward the upper bound.
* Prices are **smoothed and bounded** to avoid erratic jumps.

**Tools Used**

* Language: Python
* Libraries: numpy, pandas, bokeh
* Notebook: Google Colab
* No external ML libraries used

**Conclusion**

This project successfully simulates a basic real-time pricing engine that adjusts prices based on occupancy and contextual signals. While simple, it lays the foundation for more complex systems that can use competition, prediction, and reinforcement learning to improve pricing efficiency in urban infrastructure.