Dynamic Pricing for Urban Parking Lots

Project Overview

This project builds a **dynamic pricing of urban parking lots using real-time data** and economic principles. Given 14 parking locations and multiple time-varying inputs (like occupancy, queue length, vehicle type, traffic, and special events), the objective is to create a pricing model that adjusts the parking price intelligently over time.

The model ingests streaming data(obtained from csv file), evaluate demand and competition, and update prices accordingly using custom logic built only with **NumPy**, **Pandas**, and **Pathway**.

Project Architecture & Workflow

Steps Taken

1. **Data Ingestion**
   - Imported CSV(dataset.csv) with Pandas and streamed via Pathway.
   - Converted LastUpdatedDate + LastUpdatedTime to unified timestamps.
   - Calculated Occupancy Ratio = Occupancy / Capacity.
2. **Model 1 – Baseline Linear Model**
   - Applied formula: **Price(t+1) = Price(t) + α × (Occupancy / Capacity)**
   - Initialized price at $10 and α = 0.6 (tunable)
   - Used recursive logic (via Python loop) to compute the price based on previous price.
3. **Model 2 – Demand-Based Pricing**
   - Constructed a **demand function** using multiple features:
     Demand = α × (Occupancy / Capacity) + β × QueueLength − γ × Traffic + δ × IsSpecialDay + ε × VehicleTypeWeight

     Variables:

     1. **Occupancy / Capacity**: Indicates fullness of the lot.
     2. **QueueLength**: Adds urgency — more cars waiting means higher demand.
     3. **TrafficLevel**: Higher congestion lowers demand.
     4. **IsSpecialDay**: Special days like holiday likely raise demand.
     5. **VehicleTypeWeight**: e.g., cars (1.0), bikes (0.5), trucks (1.5), cycle(0.1).
   - Normalized demand between 0 and 1.
   - Applied smoothed pricing:
     Price(t) = Base × (1 + λ × Demand)
   - Ensured price is bounded between 0.5x and 2x.

4. **Model 3 – Competitive Pricing (Optional)**
    - Calculated **proximity** between 2 lots using Haversine formula(not implemented in codes, but used an online calculator).
    - Chose 2 lots **BHMBCCTHL01**(own lot) and **BHMBCCMKT01**(competitor lot)
    - Compared prices and applied competition logic
        1. If lot full & nearby cheaper → reduce price or reroute.
        2. If lot empty & nearby expensive → increase price moderately.
5. **Visualization with Bokeh**
    - Rendered dynamic price plots.
    - Compared own price vs. competitor lots.

Assumptions

- Demand varies linearly with features.
- Trends over the dataset are consistent.
- Competition logic based on location(proximity).
- Small price changes can affect lot occupancy.
- Base price=10, Tunable Weights $\alpha = 0.6$, $\lambda = 0.7$, $\beta = 0.3$, $\gamma = 0.2$, $\delta = 0.5$, $\varepsilon = 0.4$

Price Behaviour

Model 1: Price keeps on rising **(being dependent on previous price).**

Model2: Rises with higher occupancy, longer queues, special day**.** Drops with low occupancy, higher traffic level,

Model3: Keeps the prices competitive (below competitors' price making it attractive)

Tech Stack Used

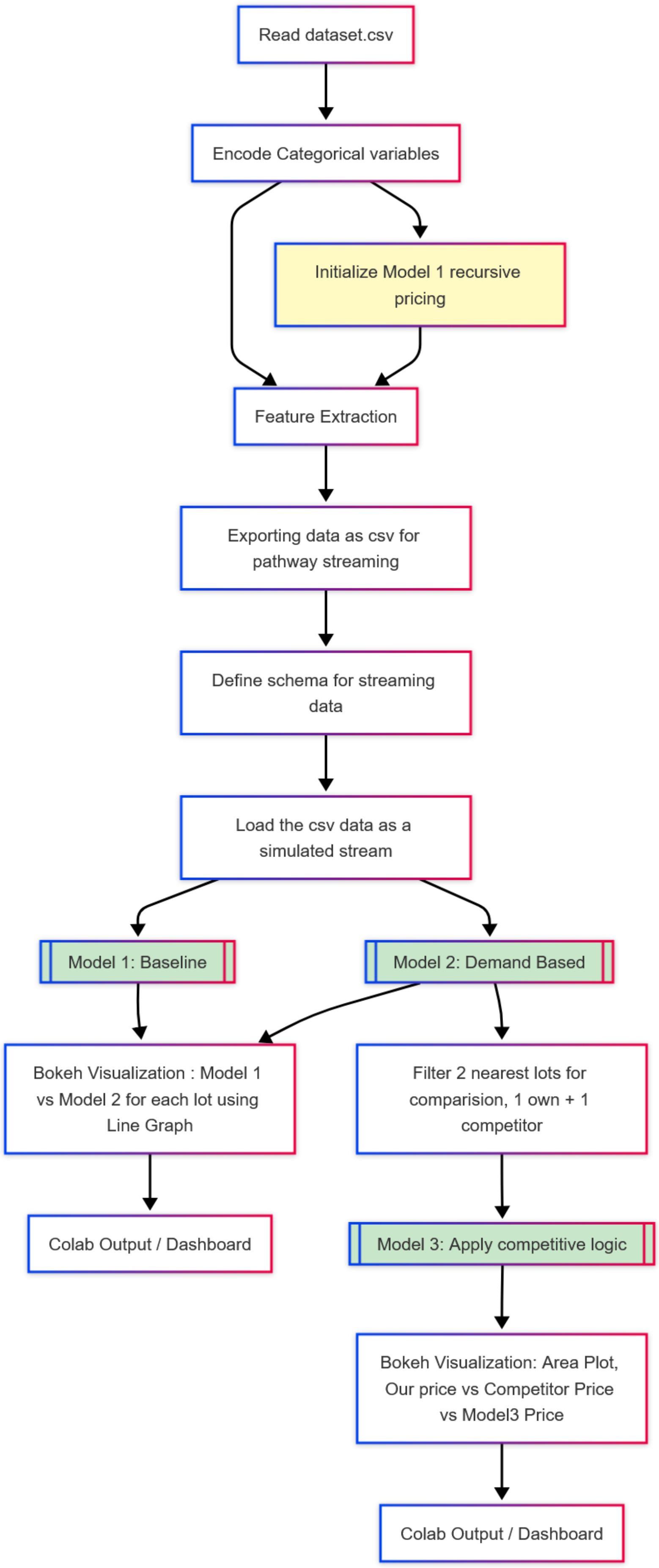
Programming: Python (NumPy, Pandas)

Data Processing: Pathway

Visualization: Bokeh

Notebook: Google Colab

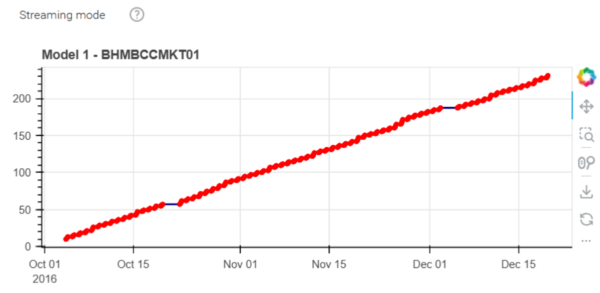
Architecture Diagram

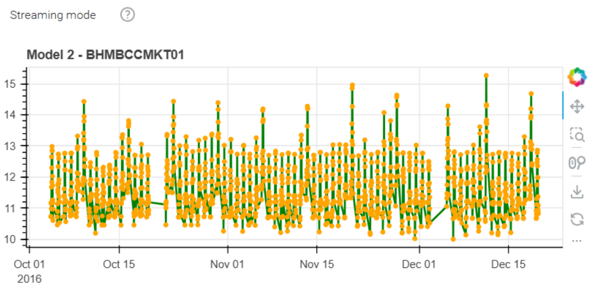

Read dataset.csv
Encode Categorical variables
Initialize Model 1 recursive pricing
Feature Extraction
Exporting data as csv for pathway streaming
Define schema for streaming data
Load the csv data as a simulated stream
Model 1: Baseline
Model 2: Demand Based
Bokeh Visualization : Model 1 vs Model 2 for each lot using Line Graph
Filter 2 nearest lots for comparision, 1 own + 1 competitor
Colab Output / Dashboard
Model 3: Apply competitive logic
Bokeh Visualization: Area Plot, Our price vs Competitor Price vs Model3 Price
Colab Output / Dashboard

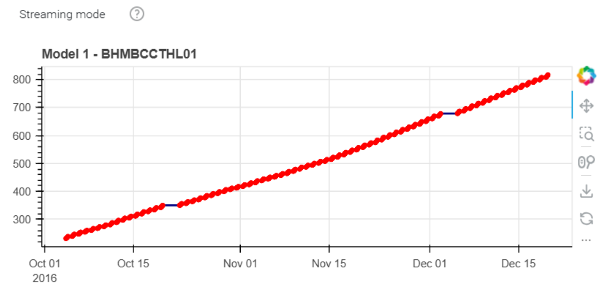
Visualizations

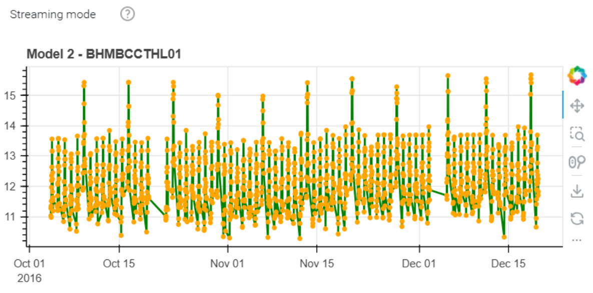
Model 1 vs Model 2

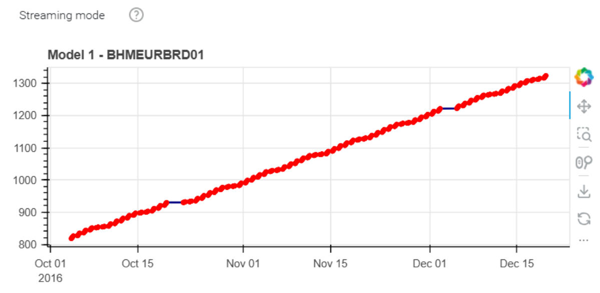
Model 1 vs Model 2 (for first 4 lots), Remaining lots can be visualized in the colab output dashboard

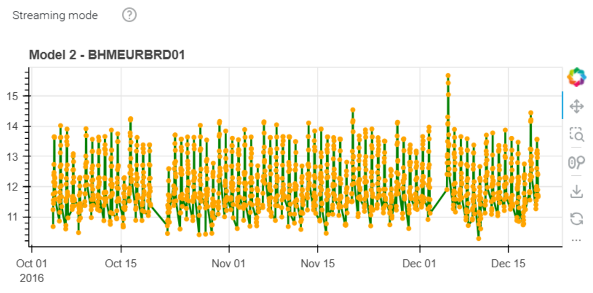


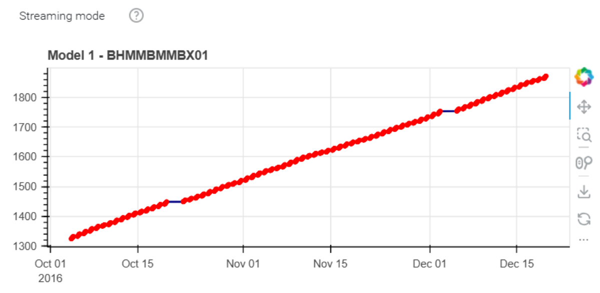


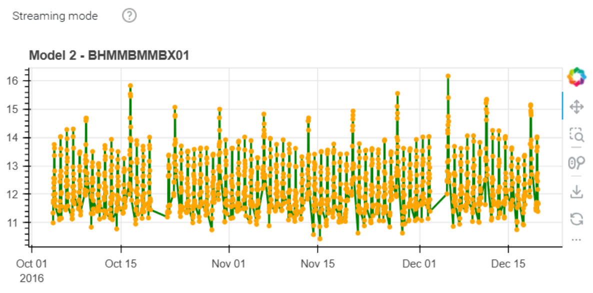










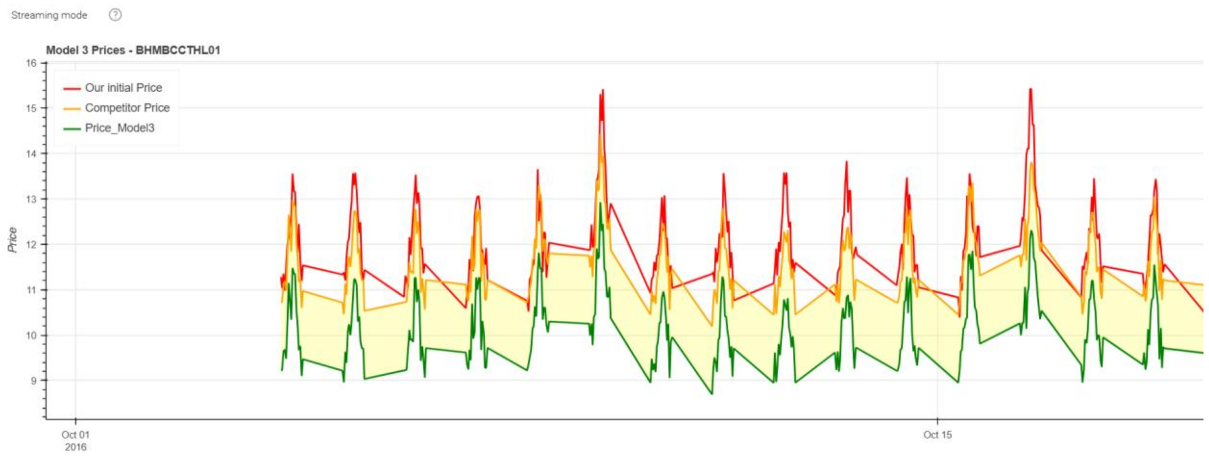


Model 3: Competition Logic (BHMBCCTHL01 vs BHMBCCMKT01)

Keeps the prices competitive (below competitors' price making it attractive)

Only a chunk of the plot is shown for clarity, the whole plot can be visualized in the colab output.

Red line is our lot’s(BHMBCCTHL01) initial price, orange line is the competitor’s (BHMBCCMKT01) price and green line is the adjusted price based. **Yellow region shows the gap**.


Streaming mode
Model 3 Prices - BHMBCCTHL01
Our initial Price
Competitor Price
Price_Model3
Price
Oct 01
Oct 15