Dynamic Pricing Algorithms & Rules Document

1. Objective

The objective of the dynamic pricing algorithm is to maximize revenue per class while maintaining high attendance by adjusting class prices dynamically based on demand, time, class popularity, and price sensitivity.

2. Inputs to the Algorithm

- **Base Price** – Original class price
- **Forecasted Demand** – Predicted attendance as a percentage of capacity
- **Class Capacity** – Maximum number of participants
- **Time Slot** – Peak or off-peak hours
- **Class Type** – Popular or regular class
- **Price Elasticity** – Sensitivity of demand to price change

3. Key Pricing Rules

1. **High Demand Rule**
    - If forecasted demand > 80% of capacity
    - Increase price by 10–20%
2. **Low Demand Rule**
    - If demand < 40%
    - Reduce price by 10% to increase occupancy
3. **Peak Hour Rule**
    - Classes between 6–9 AM and 6–9 PM
    - Add a peak hour premium
4. **High-Demand Class Rule**
    - Popular class types (Yoga, HIIT, Strength)
    - Higher base price is allowed
5. **Elasticity Control Rule**
    - Price increase is capped using price elasticity
    - Prevents overpricing and customer drop-off

4. Step-by-Step Dynamic Pricing Algorithm

Step 1: Start with Base Price

The algorithm begins with the original class price.

Step 2: Calculate Demand Ratio

Demand Ratio=Forecasted AttendanceClass Capacity\text{Demand Ratio} = \frac{\text{Forecasted Attendance}}{\text{Class Capacity}}Demand Ratio=Class CapacityForecasted Attendance

Step 3: Apply Demand-Based Adjustment

- If Demand Ratio > 0.8 → Increase price
- If Demand Ratio < 0.4 → Decrease price
- Else → Keep base price

Step 4: Apply Time-Based Adjustment

- If class time falls in peak hours → Add premium

Step 5: Apply Class Popularity Adjustment

- Popular class → Allow higher pricing range

Step 6: Apply Elasticity Cap

- Final price increase is limited based on price elasticity
- Ensures demand does not fall sharply

Step 7: Output Final Dynamic Price

The final adjusted price is generated for the class.

5. Algorithm Logic (Pseudo Code)

Input: base_price, demand_ratio, elasticity, is_peak, is_popular

price = base_price

IF demand_ratio > 0.8:
    increase = MIN(0.2, ABS(elasticity))
    price = price * (1 + increase)

ELSE IF demand_ratio < 0.4:
    price = price * 0.9

IF is_peak:
    price = price * 1.05

IF is_popular:
    price = price * 1.05

RETURN final_price

6. Python Implementation

```
def dynamic_price(base_price, demand_ratio, elasticity, is_peak, is_popular):
    price = base_price

    # Demand-based pricing
    if demand_ratio > 0.8:
        price *= (1 + min(0.2, abs(elasticity)))
    elif demand_ratio < 0.4:
        price *= 0.9

    # Peak hour pricing
    if is_peak:
        price *= 1.05

    # Popular class premium
    if is_popular:
        price *= 1.05

    return round(price, 2)
```

7. Simulation & Expected Impact

- High-demand classes generated higher revenue
- Low-demand classes saw increased attendance
- Price fluctuations remained within acceptable limits
- Overall revenue per class increased

8. Business Benefits

- Dynamic revenue optimization
- Better class utilization
- Customer-friendly pricing
- Scalable across locations and class types

9. Conclusion

This dynamic pricing algorithm combines demand forecasting, price elasticity analysis, and well-defined business rules to create a practical, data-driven, and scalable pricing system for fitness classes. By dynamically adjusting prices based on real-time demand, peak hours, class popularity, and customer price sensitivity, the model ensures optimal balance between maximizing revenue and maintaining high class occupancy**.**