RM 294 - Optimization II — Project 2

Dynamic Programming

Ramzi Kattan (rk32373), Kennedy Zapalac(kmz459), Dameli Aziken (da35254), Varsha Manju Jayakumar (vm26476)

Table of Contents

Executive Summary

This report presents a comprehensive optimization and simulation-based analysis of airline ticket overbooking strategies using dynamic programming and forward-looking policy evaluation. The goal is to determine the optimal number of coach seats to oversell and the best pricing strategy that maximizes long-term profitability while minimizing operational risk from bumping passengers.

The core contributions of this analysis are:

- **Dynamic Programming Framework:** A backward recursion model evaluates daily ticket pricing strategies under various constraints, including seat capacities, demand uncertainty, and customer show-up probabilities.
- **Flexible Pricing Control:** Multiple scenarios are compared — from fixed coach/first-class pricing to policies that include the option to pause coach ticket sales altogether.
- **Realistic Overbooking Costs:** Terminal cost computations use binomial modeling of passenger show-ups to accurately estimate upgrade and bumping penalties.
- **Seasonality-Aware Demand Modeling:** Demand probabilities increase over time, reflecting real-world customer behavior as departure approaches. This significantly improves the accuracy and responsiveness of the model.
- **Stochastic Forward Simulation:** Policies are tested through Monte Carlo simulations to assess their real-world performance over thousands of hypothetical flight runs. Key metrics such as profit volatility, overbooking frequency, bumping incidents, and class-wise seat utilization are captured.
- **Comprehensive Policy Comparison:** Results from different overbooking levels and policy types are directly compared using both quantitative tables and visual analytics, identifying the optimal strategy in both static and seasonal demand settings.

We tested four distinct scenarios: (1) Basic Overbooking with a fixed 5-seat overbooking limit; (2) Fixed Optimal Overbooking with 9 seats of overbooking; (3) Optional Coach Sales allowing strategic pausing of coach ticket sales; and (4) Optional Coach Sales under Seasonal Demand incorporating time-dependent purchase probabilities. Our analysis reveals that Scenario 3 (Optional Coach Sales) achieves the highest expected profit at $42,139.89, closely followed by Scenario 2 at $42,134.62. However, these profit-maximizing strategies result in denied boardings on over 70% of flights. For airlines prioritizing customer satisfaction, Scenario 1 offers significantly better passenger experience with denied boardings on only 25.7% of flights, albeit at a lower profit of $41,886.16. Crucially, Scenario 4 demonstrates how changing business assumptions can dramatically impact outcomes—the same optimal policy from Scenario 3, when applied in an environment with seasonal demand patterns, delivers the lowest profit ($41,830.46) among all tested scenarios. This underscores the importance of accurately modeling business conditions, as even an optimal policy can underperform basic approaches when key assumptions change.

Importantly, across all tested scenarios and assumptions, our dynamic programming model consistently identifies the optimal decision policy that maximizes expected profit. A key finding is that the optimal overbooking level remains stable at 9 coach seats when the airline must sell tickets daily. However, when given the flexibility to strategically pause coach ticket sales on select days (Scenario 3), the airline can set a higher nominal overbooking limit while still maintaining effective control. With this flexibility, even if the overbooking capacity is nominally set to 15+ seats, the optimal policy naturally limits actual overbooking to around 9 seats by selectively choosing not to sell on certain days. This strategic selling approach increases profits compared to mandatory daily sales because it allows the airline to prioritize selling tickets on the most advantageous days. Airlines can immediately implement this insight by adjusting their booking systems to allow for day-to-day sales decisions rather than rigid daily quotas. The practical value of this model lies in its ability to quantify the precise trade-offs between competing objectives, allowing airlines to make data-driven decisions aligned with their strategic goals.

Presets

Before beginning our analysis of the optimal ticket pricing strategy, we identified these presets pictured below as reasonable assumptions on which to base our analysis. Realistically, these presets will vary from flight to flight. We will likely want a larger variety of prices we charge, different airplanes have different capacities, and demand may be much more complex. For example, we assumed coach and first-class demand to be independent. The strength and relevance of this analysis are highly dependent on the preset values used in the model (e.g., show-up probabilities, pricing options, and overbooking penalties). These presets can be easily modified at the top of the code file to reflect different business environments or assumptions. Future applications could explore alternative pricing tiers, route-specific show-up rates, or seasonal variations in penalties to further refine policy recommendations for specific markets or customer segments.

```
allow_no_coach_sale=False # For scenario 3
seasonal_demand=False    # For scenario 4
coach_seats=100          # Fixed across all scenarios
first_class_seats=20     # Fixed across all scenarios
coach_prices=[300, 350]  # Fixed across all scenarios
first_class_prices=[425, 500] # Fixed across all scenarios
coach_show_prob=0.95    # Fixed across all scenarios
first_class_show_prob=0.97 # Fixed across all scenarios
coach_demand_prob=[0.65, 0.30] # Base probabilities
coach_demand_increase=0.03 # Coach demand increase when first-class sold out
first_class_demand_prob=[0.08, 0.04] # Base probabilities
bump_cost=50             # Cost to bump to first-class
denied_cost=425          # Cost to deny boarding
days=365                 # days until flight
discount_rate=0.17       # used to discount profit from selling tickets at time t
```

General Mathematical Formulation

The dynamic programming framework for each scenario incorporates several interconnected components. The **state variables** track the system's current condition: day index from 0 to 365 (t), total coach tickets sold at time t (ct), and total first-class tickets sold at time t (ft). **Decision variables** represent the daily pricing choices that management must make for both cabins. On each day, you pick a pricing pair. Coach can be either low ($300) or high ($350), and first-class can be either low ($425) or high ($500). This gives you 4 possible combinations each day: (1) low coach and low first-class price, (2) low coach and high first-class price, (3) high coach and low first-class price, and (4) high coach and high first-class price. In Scenario 3, the model introduces a third Coach pricing action: no sale, allowing the system to skip selling a coach ticket on any day. This expands the action space and enables more flexible, state-sensitive strategies. In Scenario 4,the same flexible pricing control is used, but probabilities are adjusted dynamically using a **seasonality multiplier**, which increases sale likelihoods as the departure date approaches. This allows the model to pace sales more effectively.The **dynamics** describe how the system evolves over time, with state transitions following stochastic processes based on pricing decisions and resulting sales probabilities. In other words, the number of tickets sold at time t+1 is dependent on the price chosen at time t according to this equation:

Dynamics: (t, c_t, f_t) ------> (t+1, c_{t+1}, f_{t+1})

$$c_{t+1} = \begin{cases} c_t + 1, & \text{Coach tickets sold at } t+1: \\ & \text{with probability } P_c(p_c, f_t) \text{ if } c_t < \text{cap}_c \\ c_t, & \text{with probability } (1 - P_c(p_c, f_t)) \text{ or if } c_t = \text{cap}_c \end{cases}$$

$$f_{t+1} = \begin{cases} f_t + 1, & \text{First-class tickets sold at } t+1: \\ & \text{with probability } P_f(p_f) \text{ if } f_t < 20 \\ f_t, & \text{with probability } (1 - P_f(p_f)) \text{ or if } f_t = 20 \end{cases}$$

Where:

- c_t = Number of coach tickets sold at time t
- f_t = Number of first-class tickets sold at time t
- c_{t+1} = Number of coach tickets sold at time $t+1$
- f_{t+1} = Number of first-class tickets sold at time $t+1$
- cap_c = Maximum allowed coach tickets (105 for the +5 oversold scenario)
- p_c = Price of coach ticket, either $\$300$ or $\$350$
- p_f = Price of first-class ticket, either $\$425$ or $\$500$
- $P_c(p_c, f_t)$ = Probability of selling a coach ticket at price p_c given f_t first-class tickets sold:
  - $P_c(300, f_t) = 0.65$ if $f_t < 20$, and 0.68 if $f_t = 20$
  - $P_c(350, f_t) = 0.30$ if $f_t < 20$, and 0.33 if $f_t = 20$
- $P_f(p_f, c_t)$ = Probability of selling a first-class ticket at price p_f:
  - $P_f(425) = 0.08$
  - $P_f(500) = 0.04$

The **Bellman equation** forms the mathematical heart of our approach, determining the optimal decision by recursively maximizing expected discounted profit at each state while considering all possible future outcomes.

$$V(t, c_t, f_t) = \max_{p_c, p_f} \{\mathbb{E}[R_t + \delta \cdot V(t+1, c_{t+1}, f_{t+1})]\}$$

Where:

- $V(t, c_t, f_t)$ = Maximum expected discounted profit from state (t, c_t, f_t)
- c_t = Number of coach tickets sold at time t
- f_t = Number of first-class tickets sold at time t
- p_c = Price of coach ticket, either $\$300$ or $\$350$
- p_f = Price of first-class ticket, either $\$425$ or $\$500$
- R_t = Revenue at time t
- δ = Daily discount factor = $1/(1 + 0.17/365)$

As noted previously, there are 4 possible price combinations each day. Therefore, in this simple scenario, we can expand the Bellman equation to demonstrate what we are maximizing over more explicitly. The four pricing combinations are:

1. $(p_c, p_f) = (300, 425)$: Low coach price, low first-class price
2. $(p_c, p_f) = (300, 500)$: Low coach price, high first-class price
3. $(p_c, p_f) = (350, 425)$: High coach price, low first-class price
4. $(p_c, p_f) = (350, 500)$: High coach price, high first-class price

Now, the new bellman equation is:

$$V(t, c_t, f_t) = \max\left\{V^{(300,425)}(t, c_t, f_t), V^{(300,500)}(t, c_t, f_t), V^{(350,425)}(t, c_t, f_t), V^{(350,500)}(t, c_t, f_t)\right\}$$

Each pricing strategy's value can be expressed as the expectation of all possible outcomes (sell both, only sell coach, only sell first class, and don't sell either). Here is an example for $300 coach price and $425 first-class price.

$$\begin{aligned} V^{(300,425)}(t, c_t, f_t) = & \\ & P_c(300, f_t) \cdot P_f(425) \cdot [300 + 425 + \delta \cdot V(t+1, c_t+1, f_t+1)] + \\ & P_c(300, f_t) \cdot (1 - P_f(425)) \cdot [300 + \delta \cdot V(t+1, c_t+1, f_t)] + \\ & (1 - P_c(300, f_t)) \cdot P_f(425) \cdot [425 + \delta \cdot V(t+1, c_t, f_t+1)] + \\ & (1 - P_c(300, f_t)) \cdot (1 - P_f(425)) \cdot [\delta \cdot V(t+1, c_t, f_t)] \end{aligned}$$

Similar equations apply for the other three price combinations, with the appropriate price values substituted. The probability values from the assumptions are:

- $P_c(300, f_t) = 0.65$ if $f_t < 20$, and 0.68 if $f_t = 20$
- $P_c(350, f_t) = 0.30$ if $f_t < 20$, and 0.33 if $f_t = 20$
- $P_f(425) = 0.08$
- $P_f(500) = 0.04$

If coach tickets sold reach the overbooking level or if first-class tickets sold reaches capacity, then the probabilities become:

- If $c_t = \text{cap}_c$, then $P_c(300, f_t) = P_c(350, f_t) = 0$
- If $f_t = 20$, then $P_f(425) = P_f(500) = 0$

In order to solve the Bellman equation, you must determine the terminal condition since this equation loops recursively from t=365 to t=0.

Finally, the **terminal condition** handles the actual flight day (t=365), where no more sales occur but costs are incurred based on passenger show-up patterns. This condition uses binomial

distributions to calculate expected costs from upgrading passengers to first-class ($50 per upgrade) or denying boarding entirely ($425 per denied passenger), capturing the full financial impact of the overbooking policy. We loop over all possible values of passengers showing up for all possible tickets sold at time T=365 to complete the terminal value calculation. For example, you could have sold 105 coach tickets ($c_{365} = 105$) and 20 first-class tickets ($f_{365} = 20$) at T.

Given the tickets sold, the expectation of your costs is calculated over all possible combinations of 0 to 105 coach passengers and 0 to 20 first class passengers showing up. You must then calculate the terminal value for all possible combinations of tickets sold because this will be used to determine the optimal pricing decision at T-1=364 and so on in the Bellman equation.

$$V(365, c_{365}, f_{365}) = -\sum_{i=0}^{c_{365}} \sum_{j=0}^{f_{365}} P(C = i) \cdot P(F = j) \cdot \text{Cost}(i, j)$$

Where:

- C = Number of coach passengers who show up (binomial with probability 0.95)
- F = Number of first-class passengers who show up (binomial with probability 0.97)
- $P(C = i) = \binom{c_{365}}{i}(0.95)^i(0.05)^{c_{365}-i}$
- $P(F = j) = \binom{f_{365}}{j}(0.97)^j(0.03)^{f_{365}-j}$
- $\text{Cost}(i, j)$ = Overbooking cost for i coach show-ups and j first-class show-ups

This cost function accounts for the three possible scenarios:

1. Coach show-ups don't exceed capacity (100 seats) → No cost
2. Some coach passengers can be bumped to available first-class seats → $50 per passenger
3. After filling available first-class seats, additional passengers must be denied boarding → $425 per passenger

Cost function is defined as :

$$\text{Cost}(i, j) = \begin{cases} 0, & \text{if } i \leq 100 \\ 50 \cdot \min(i - 100, 20 - j), & \text{if } 100 < i \leq 100 + (20 - j) \\ 50 \cdot (20 - j) + 425 \cdot (i - 100 - (20 - j)), & \text{if } i > 100 + (20 - j) \end{cases}$$

Adaptations

Allow no coach sale

The first adaptation to the general formulation above is to not sell coach tickets on any given day. This "no-sale coach option" introduces a third decision into the dynamic programming framework and allows the airline to strategically pause bookings, especially on low-demand days or when overbooking risk is high. Instead of finding the maximum value with respect to 4 pricing combinations, there are now 6 potential pricing combinations with the addition of: (1) don't sell coach and high first-class price and (2) don't sell coach and high first-class price. When you are calculating the expected value for these states, you only have to get the weighted average for selling or not selling first class.

Seasonal demand

As the departure date approaches, demand for both coach and first-class tickets increases. To model this, we scale daily sale probabilities using the formula:

$$Adjusted\ Probability = Base\ Probability * (0.75 + \frac{t}{730}),$$

where t is the day (0 to 364).

This realistically reflects heightened urgency and willingness to purchase in the days leading up to the flight. This does not change the mathematical formulation, but it changes the inputs used for P_c and P_f daily.

Code Implementation

For more details on the code implementation, visit the appendix. For now, here is a high level understanding of the code used.

calculate_expected_profit()

Calculates expected discounted profit for one run under a specified overbooking level and pricing policy. This function is flexible and can be used for all future scenarios by changing the overbooking_level, allow_no_coach_sale, and seasonal_demand parameters.

Core Logic:

- Builds value function using dynamic programming.
- Uses Bellman recursion

- Ensures you can't sell more tickets once you have reached the maximum tickets for coach or first-class
- Returns `V[0,0,0]` as the expected profit, `V` as the value matrix (366 x 106 x 21 matrix recording value function at each state=t,c,f), and `U` the policy matrix (366 x 106 x 21 matrix recording encoded optimal decision at each state=t,c,f).

U-Matrix Encodings

The U matrix stores the optimal action to take for each ticket sales state. The action values differ slightly depending on whether the model allows not selling a coach ticket.

U-Matrix (Must Sell Coach Ticket)

Used in scenarios where the airline is required to sell one coach ticket every day:

When both ticket types are available:

- 0 = Low coach, Low first-class (LL)
- 1 = Low coach, High first-class (LH)
- 2 = High coach, Low first-class (HL)
- 3 = High coach, High first-class (HH)

When first-class is sold out but coach isn't:

- 4 = Low coach only (L-)
- 5 = High coach only (H-)

When coach is sold out but first-class isn't:

- 6 = Low first-class only (-L)
- 7 = High first-class only (-H)

When both are at capacity:

- 8 = no sales possible

U-Matrix (Option to Not Sell Coach Tickets)

Used in flexible models where the airline may skip coach ticket sales on some days:

When both ticket types are available:

- 0 = Low coach, Low first-class (LL)
- 1 = Low coach, High first-class (LH)
- 2 = High coach, Low first-class (HL)
- 3 = High coach, High first-class (HH)
- 4 = No coach, Low first-class (NL)
- 5 = No coach, High first-class (NH)

When first-class is sold out but coach isn't:

- 6 = Low coach only (L-)
- 7 = High coach only (H-)
- 8 = No coach sales (N-)

When coach is sold out but first-class isn't:

- 9 = Low first-class only (-L)
- 10 = High first-class only (-H)

When both are at capacity:

- 11 = No sales possible

simulate_flight()

Simulates the full ticketing and flight process for one run using a given U policy matrix, returning profit, sales, and overbooking outcomes for that flight.

Core Logic:

- Simulates a single flight (365 days of ticket sales + flight day)
- Each day, reads the current state and selects a pricing action from the U policy matrix
- Randomly decides whether a ticket is sold for coach or first-class based on the demand distributions
- On the flight day, simulates which passengers show up (using show-up probabilities and binomial distribution)
- Applies overbooking costs on day 365 if more than 100 coach passengers show up
- Tracks the prices used and sales over time as well as the final profit, cost, passengers showed, and number of bumped to first-class or denied service

simulate_flights()

Runs forward simulations to test how the pricing strategy performs in real-world-like conditions.

Core Logic:

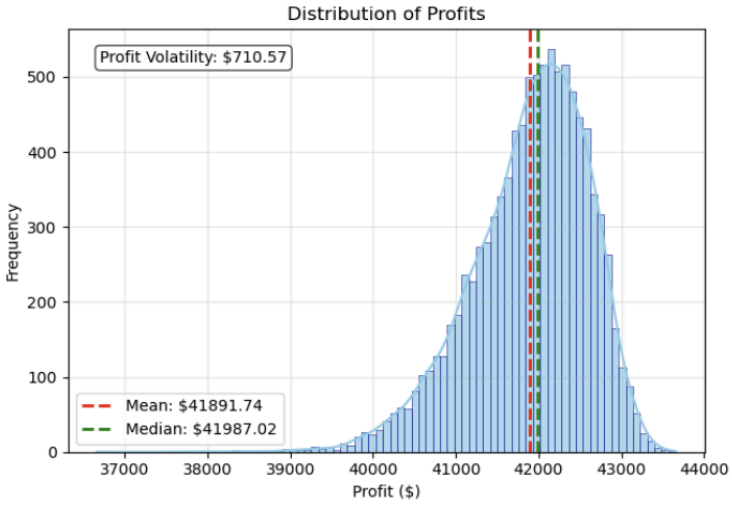
- Uses the simulate_flight() for n simulations and records the results across all simulations.
- Outputs summary statistics and visualizations across all simulations.

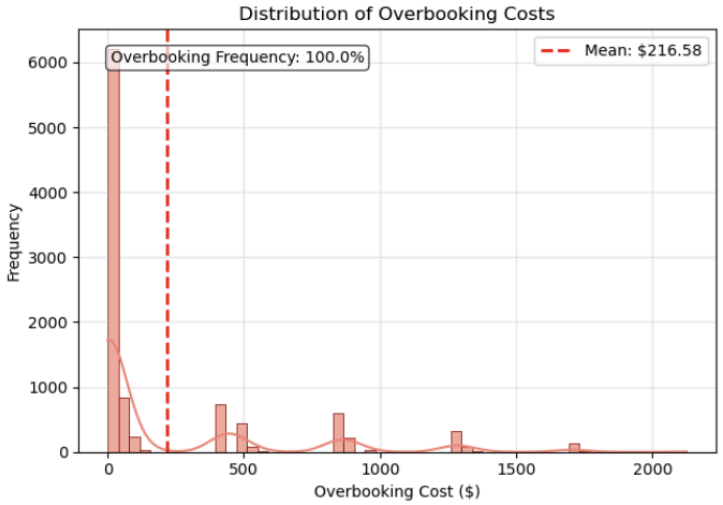
Results

Scenario 1: Overbooking Coach by 5 seats

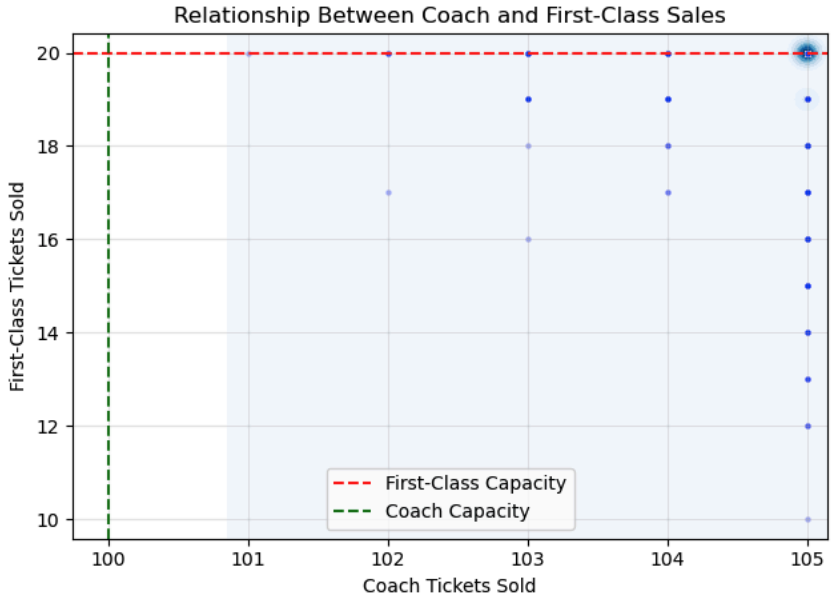
In this scenario, we examine a fixed overbooking policy where the airline allows selling up to 5 additional coach tickets beyond the physical capacity of 100 seats. The airline offers two pricing tiers for each cabin: coach tickets at either $300 (low) or $350 (high), and first-class tickets at $425 (low) or $500 (high). Customer behavior follows predictable patterns, with coach tickets selling at higher probabilities (65% at low price, 30% at high price) compared to first-class tickets (8% at low price, 4% at high price). Historical data suggests show-up rates of 95% for coach passengers and 97% for first-class passengers. Our objective is to determine the expected discounted profit from implementing this fixed overbooking strategy, and to evaluate its performance through extensive forward simulation.

After using the Bellman equation, we found the **expected profit** when using the optimal pricing policy and allowing 5 overbooked coach seats was **$41,886.16**. We ran 10,000 simulations using the optimal policy derived from the dynamic program and we found the average profit from these 10,000 simulations was very close, $41,891.74. Using simulations to analyze this scenario allows us to get more granularity when analyzing this pricing and overbooking policy. For example, here is the distribution of profits and costs from the simulations. Although the average profit was $41,891.74, there was a small standard deviation of $710.57. This means we earn less than $40,470.6 2.5% of the time (2 standard deviations below the mean). Furthermore, 100% of flights were overbooked, but overbooking penalties were generally low, averaging $216.58 per flight. Note these costs are expressed on the day of departure rather than discounted, but they are discounted when they are incorporated into profit.

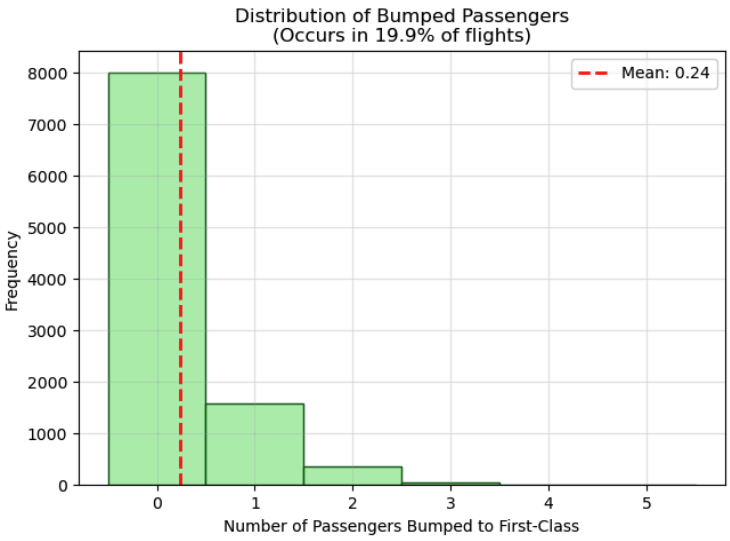


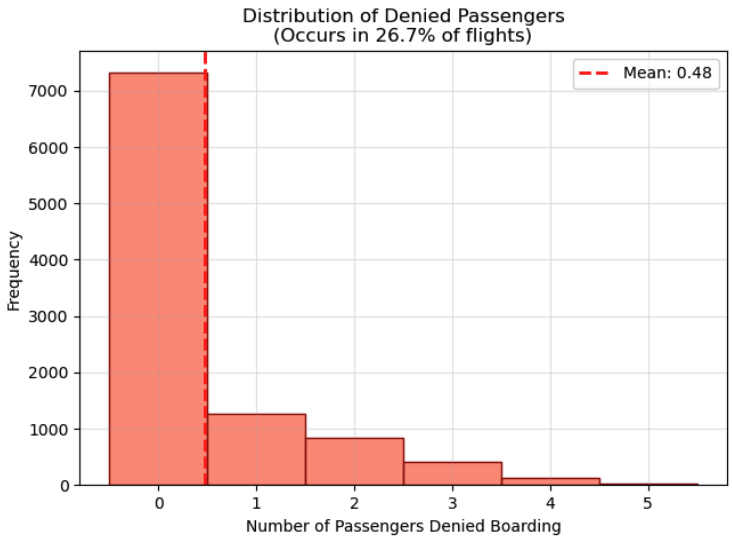


Here, you can also see the relationship between the number of coach tickets sold and the number of first class tickets sold with the seat capacities marked. More frequent ticket combinations are marked with darker, larger circles. There seems to be more variation in the number of first-class tickets sold, ranging from 10-20, whereas 101-105 coach tickets are always sold with 105 being the most frequent.

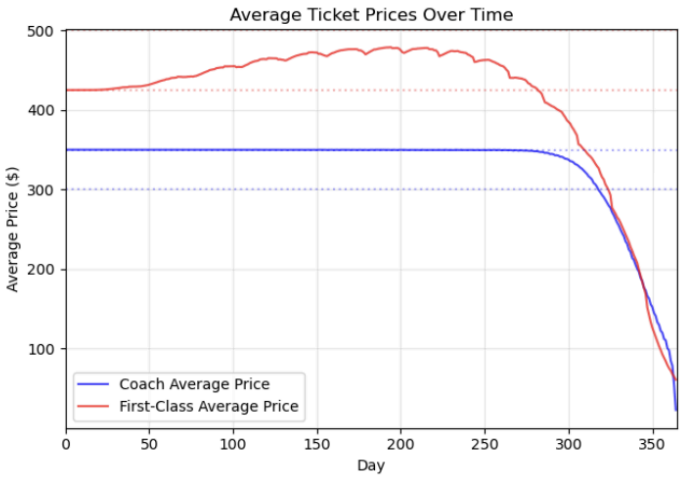


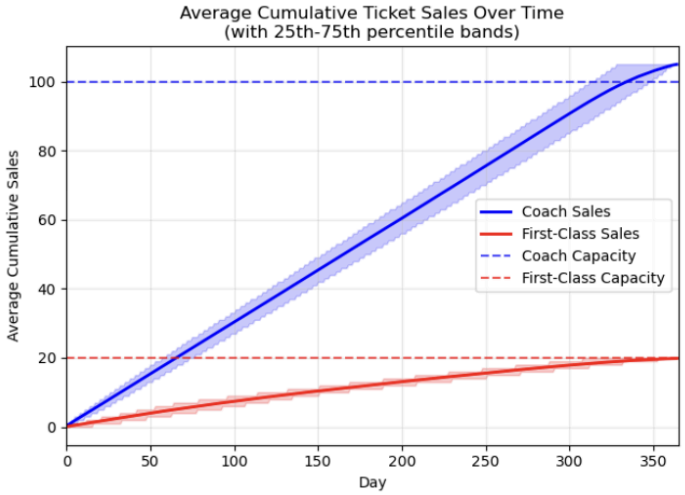
Even though we always oversold the coach cabin, we had very low overbooking costs since not all of the passengers who buy tickets show up. We bumped coach passengers to first-class in 19.9% and denied coach passengers in 26.7% of the 10,000 simulations. On average, we bump 0.24 passengers and deny 0.48 passengers. We never bump or deny more than 5 passengers since we can only overbook 5 seats in this scenario.



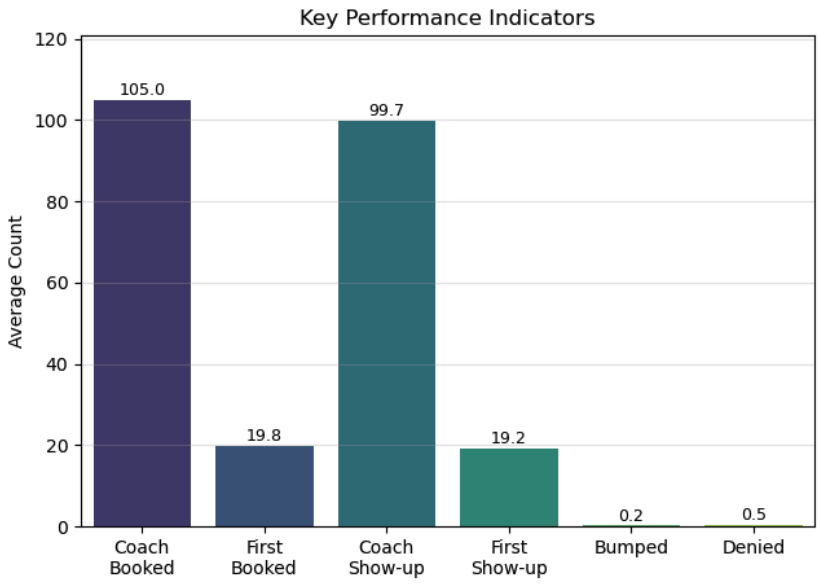


We also analyzed the average pricing strategy over time and the resulting ticket sales over time. Typically, we sell coach tickets at the high price and first-class tickets at the low price up to day 50. Around day 50, we start selling first-class tickets at the higher price, increasingly selling more tickets at the high price until day 250. In comparison, it appears we always sell coach tickets at the highest price, until the average price starts decreasing around day 300. The average price probably declines because we are reaching capacity and unable to sell coach tickets (coach price = $0), rather than we sell at the low price. The actual optimal price for each day can be determined using the U-matrix rather than looking at these averages though. The average total tickets sold over time steadily increases using these optimal policies, nearing capacity at around 300 days. Coach tickets aren't oversold until around day 330.



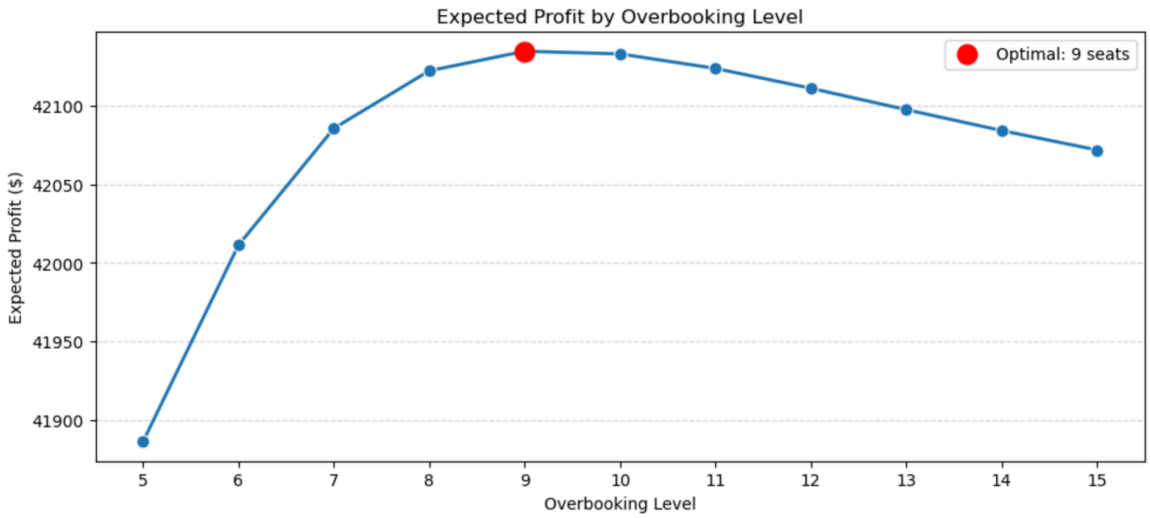


Overall, these are the key performance indicators when simulating overbooking 5 seats. The strategy achieves near-full utilization of seats with minimal bumping.



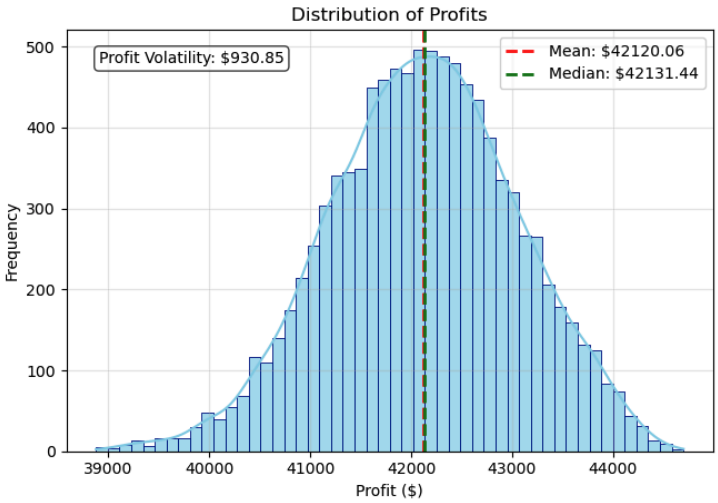
Scenario 2: Compare Profits Across Overbooking Levels

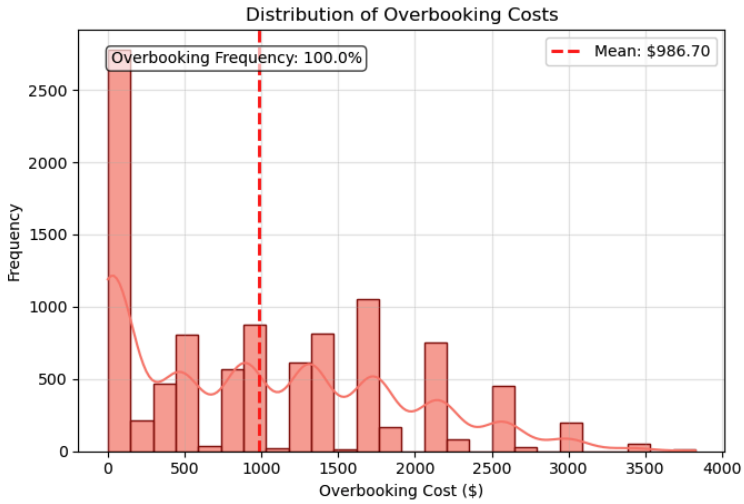
In scenario 2, we evaluate how expected profit changes as the number of overbooked seats increases from 5 to 15. The goal is to determine the optimal overbooking level that maximizes profit without incurring excessive penalties. The mathematical formulation for scenario 2 is the exact same as scenario 1, except the maximum number of tickets sold for coach changes. The results follow.



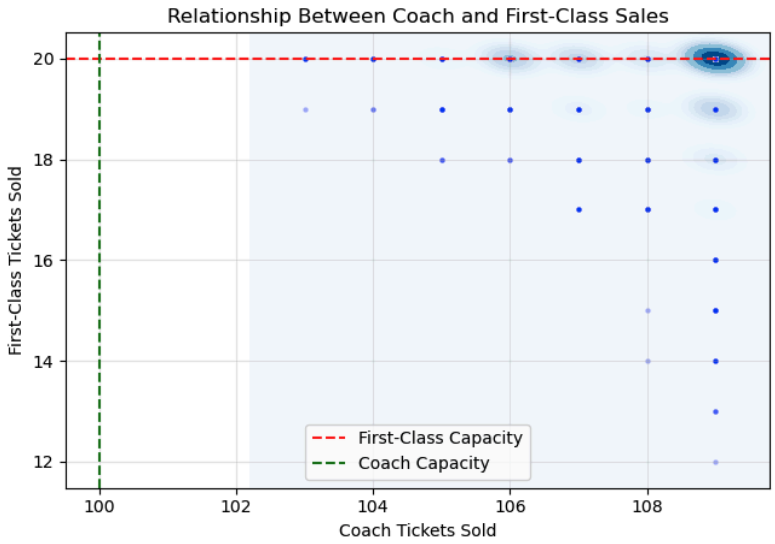
The profit increases steadily from 5 to 9 overbooked seats peaking at 109 coach tickets sold. Beyond this point, profits start to decline due to rising overbooking costs from bumped and denied passengers. The **optimal overbooking level is 9 seats** (highlighted with red dot) with an **expected profit of $42,134.62.**

After finding the optimal overbooking level, we also ran 10,000 simulations to assess the profits, costs, and bumped and denied passengers when overselling 9 seats. These simulations demonstrate the opportunity to increase profits to $42,120.06 by using an overbooking level of 9 seats instead of 5, a $228.32 (0.5%) increase. The below distribution of profits show a profit volatility of $930.85, meaning we expect to earn below $40,258.36 2.5% of the time (2 standard deviations below the mean). This is $212.24 lower than when the overbooking level is 5. However, we also stand to gain an additional $668.88 in the best case scenarios because 2.5% of the time we earn above $43,981.76 in this scenario versus $43,312 in the previous scenario. Also, you'll notice that this distribution is better characterized by a bell curve, whereas the previous distribution had a slight negative skew. Again, 100% of flights were overbooked and we incurred much higher overbooking penalties, $986.70 per flight on average.

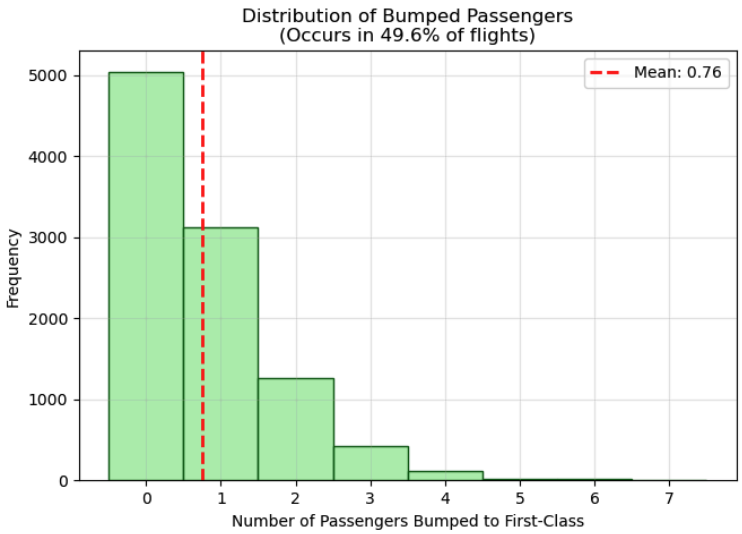


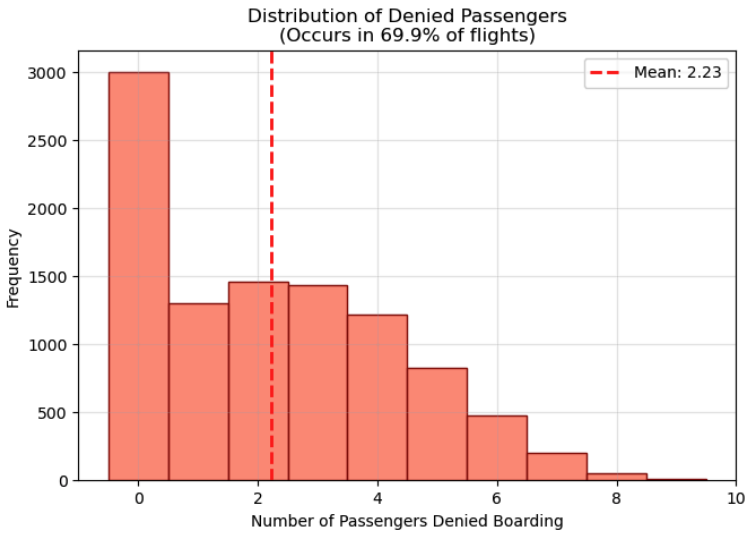


The below graph shows that we very often elect to sell at full capacity, including the 9 oversold seats. This graph shows more variation in the number of coach and similar variation in the number of first-class seats sold as compared to the 5 seat policy. 103-109 coach seats are always sold with 109 being the most frequent by far. 12-20 first class seats are sold with 20 being the most frequent.



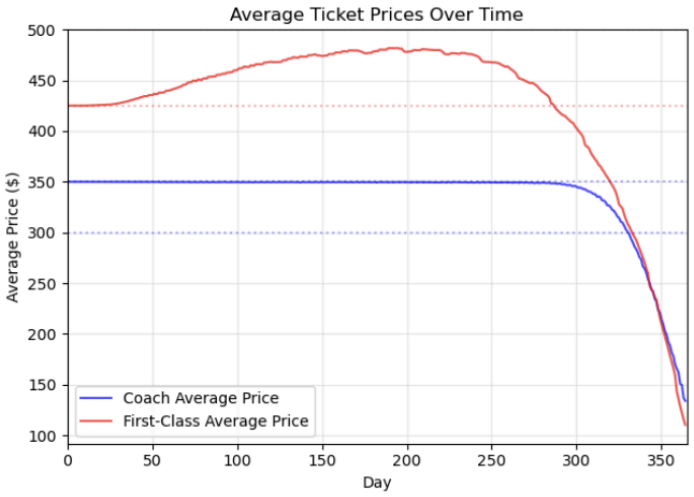
In this policy, we see that the number of people who don't show up to the flight does not offset the effects of overbooking by 9 seats. We bump coach passengers to first class in 50% of flights and deny passengers in 70% of flights in the 10,000 simulations. In some rare cases, we are forced to deny all 9 overbooked passengers off the flight.

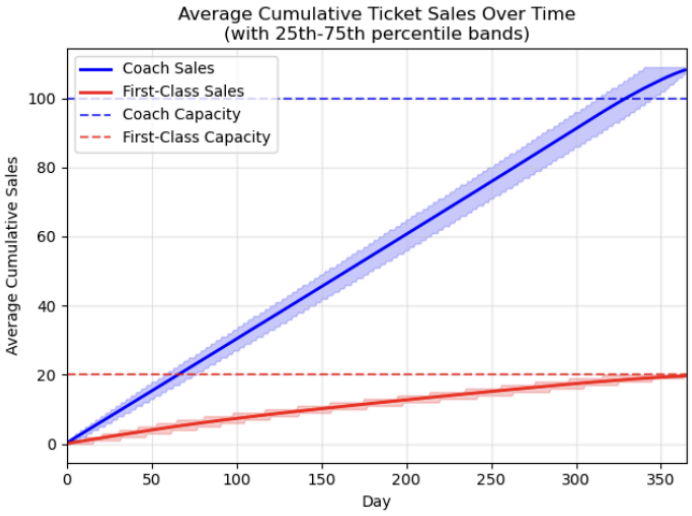




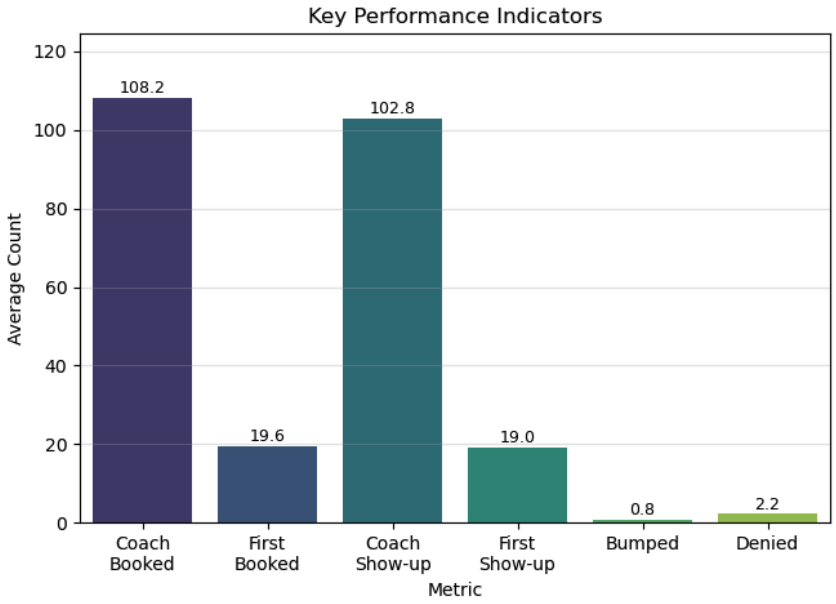
Here, the pricing policy appears to favor consistently high prices for both coach and first-class tickets. First-class tickets are sold at the low price until around day 50, after which the average price increases steadily, reaching a peak around day 200 and slowly declining afterward. This suggests an increasing shift toward the high price for first-class tickets starting around day 50. Coach prices, on the other hand, remain constant at the high price nearly the entire time. There is a slight decline in average coach prices after day 300, likely because the system reaches full coach capacity around day 325–330. This causes the average coach ticket price to drop (due to

coach tickets becoming unsellable and counted as $0), rather than indicating a switch to low-price sales. From the cumulative ticket sales plot, we observe that coach tickets are sold at a consistent rate until capacity is reached just before day 350. First-class tickets are sold more gradually and steadily throughout the time horizon, reaching full capacity right at day 350. This steady increase in sales for both classes reflects a more balanced selling strategy compared to the earlier scenario, with slightly smoother transitions in pricing over time. As with the first case, the actual optimal prices per day are determined using the U-matrix, and these average plots provide general trends rather than daily decisions.





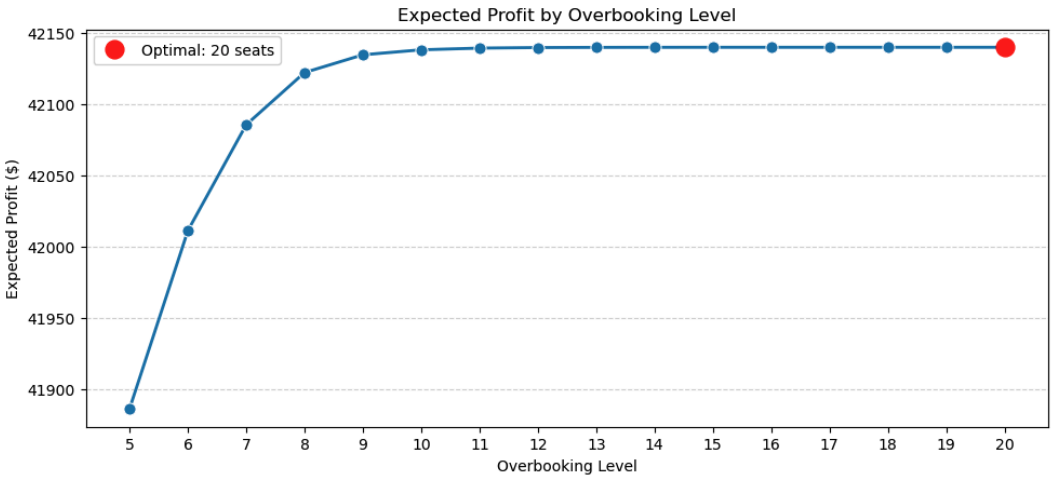
Overall, this strategy achieves near-max utilization of both cabins. When selecting an overbooking strategy, it is crucial to consider how denying passengers could affect overall brand recognition and customer experience. Although we began overbooking flights by 9 seats instead of 5, the average number of customers bumped (0.8) or denied (2.2) is still manageable.



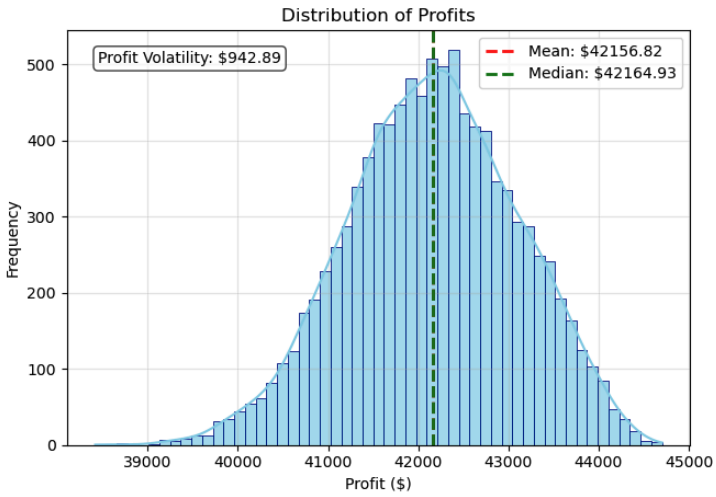
Scenario 3: Optimal Overbooking with the Option to Not Sell Coach Tickets

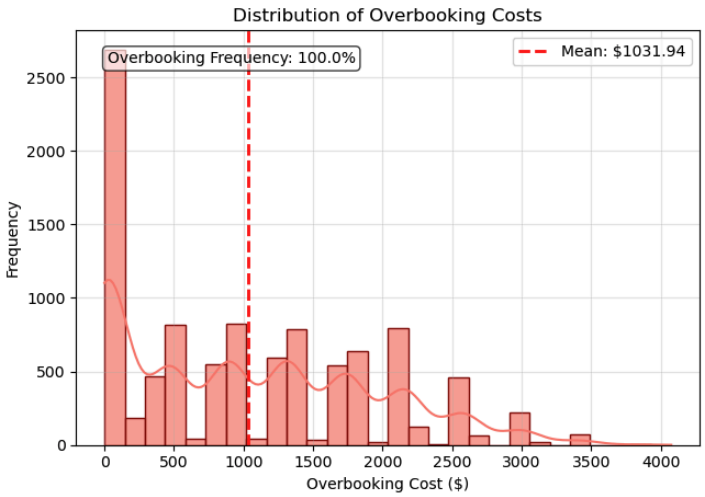
This scenario evaluates the performance of a more flexible pricing and booking strategy, where in addition to choosing between high and low coach ticket prices, the airline can now choose to not sell coach tickets at all on any given day. This flexibility gives the airline greater control over overbooking risk and demand pacing, making the policy more robust and adaptable to real-world booking patterns and risk tolerance.

First, we had to determine if the optimal overbooking level remained the same under this scenario. As you'll see below, the expected profit levels off around 10 or level oversold seats. The optimal overbooking level is slightly higher than the 9 seats before in scenario 2 because you can now decide to stop selling once you have overbooked if it's not profitable to continue selling. With the **optimal overbooking level of 20** here (it's just cents above all 11+ overbooking levels), the **expected profit is $42,139.89.**

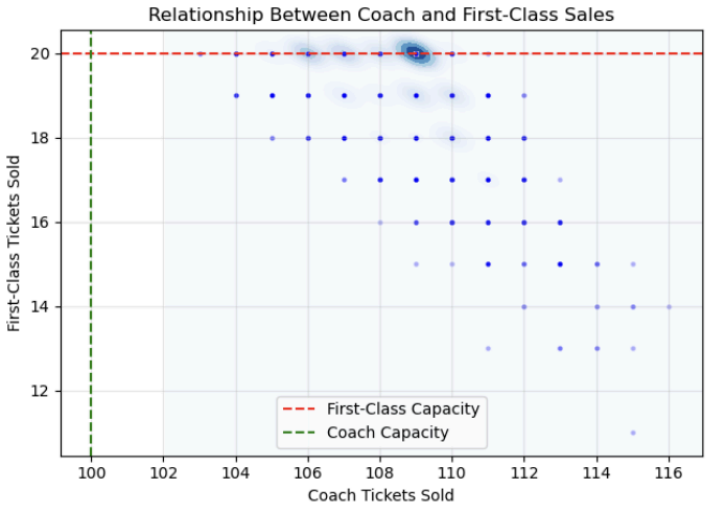


Once we knew the optimal overbooking level was 20, we ran 10,000 simulations using this overbooking level. The profit histogram below shows a tight bell-shaped distribution centered around $42,156.82 with a standard deviation of $942.89. The profit in this scenario is slightly higher but so is the volatility. The gains from implementing the no coach sale policy are minimal; the average profit is $36.76 greater when using an overbooking level of 20 with no coach sale versus an overbooking level of 9 with required daily sales. Furthermore, sales are below $40,271.04 2.5% of the time, and sales are above $44,042.6 2.5% of the time.These gains over many flights add up though. As for the histogram of overbooking costs, it shows that while costs are frequent (since every flight is overbooked), they are typically moderate — with a mean around $1,031.94. The costs are slightly higher than in scenario 2 (required sales) with a difference of $42.24. Despite the almost double overbooking level instated, the costs do not increase proportionally due to the option to not sell.

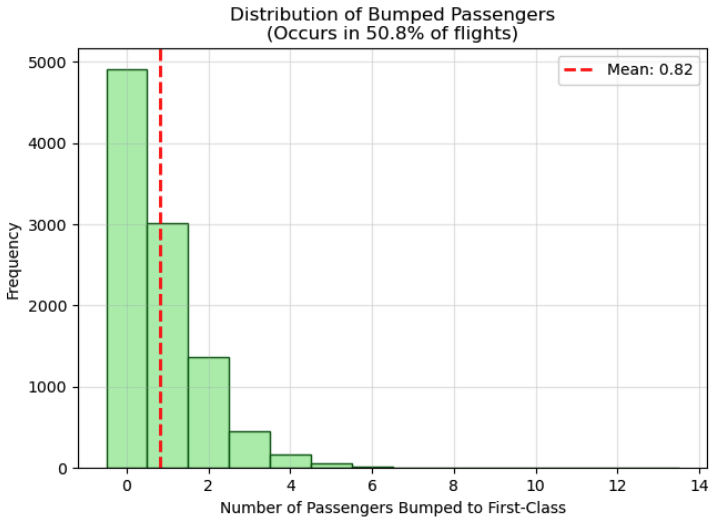


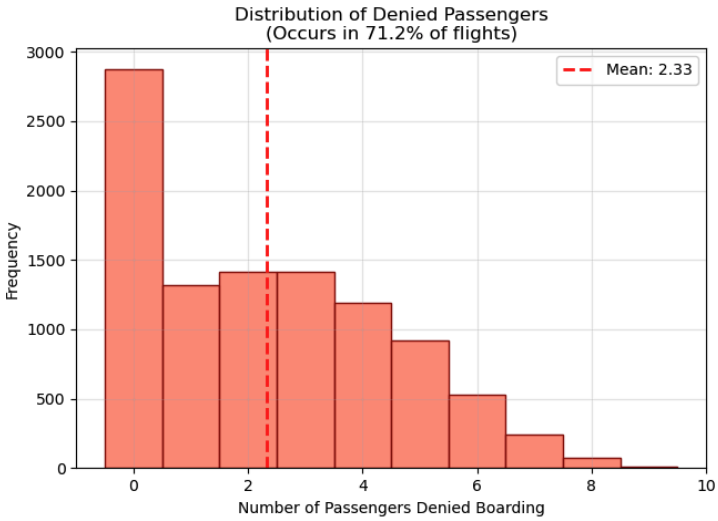


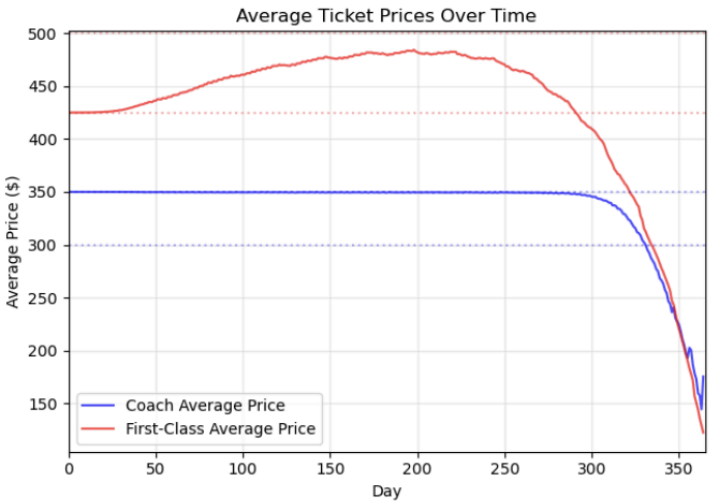
Here we can see why the costs don't proportionately increase with the overbooking level. Despite the ability to oversell 20 seats, we typically only oversell 9 seats as indicated by the graph below. Here we have more variation in the number of coach seats sold (103-116) than in scenario 2 since the overbooking policy is set to 20 instead of 9. The number of first class tickets sold remains in the same range of 12-20, with 20 being the most frequent.

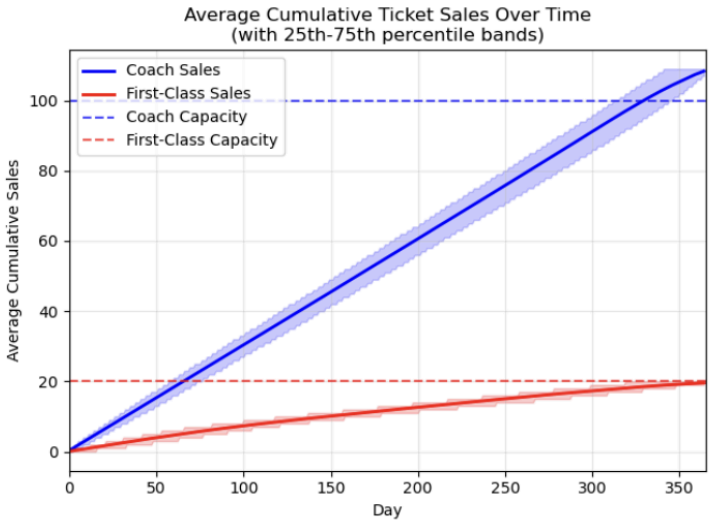


As for the passenger experience, the bumped-passenger histogram shows that about half of all flights at least one passenger upgraded to first-class and just over 70% of flights have at least one passenger denied. Bumped passengers average 0.82 while denied boardings average 2.33 per flight. The increased frequency of bumps and denials reflects a more aggressive overbooking strategy, which the airline can afford to do when they have the option not to sell tickets.



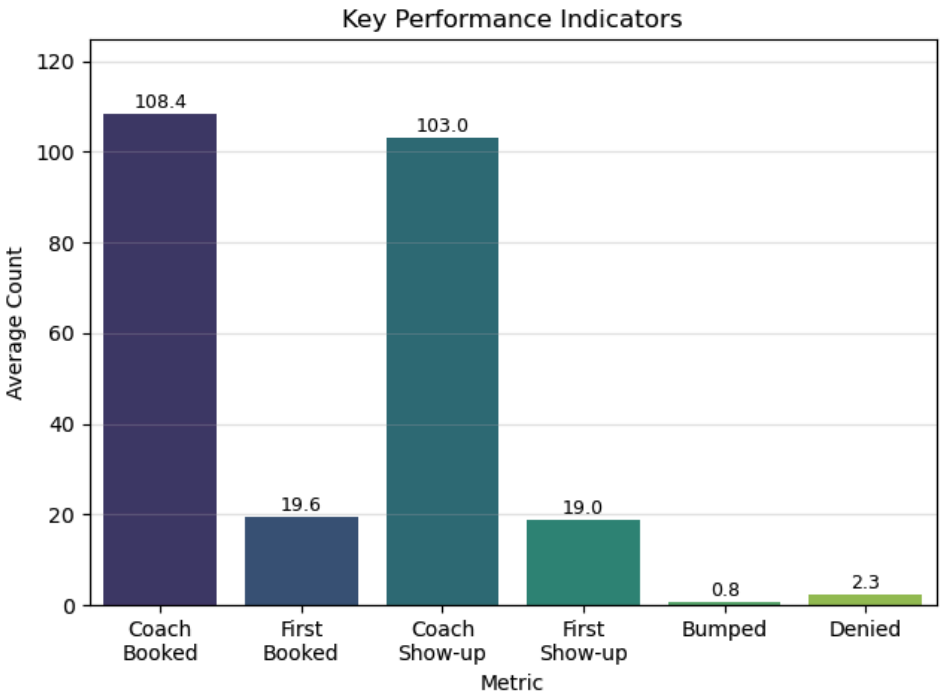






Scenario 3 demonstrates that adding the option to pause coach ticket sales offers **a meaningful tactical advantage**. It slightly increases profit over the required sale benchmark (Scenario 2) by enabling smarter pacing of coach sales, which can help **defer overbooking pressure** and better align supply with demand.

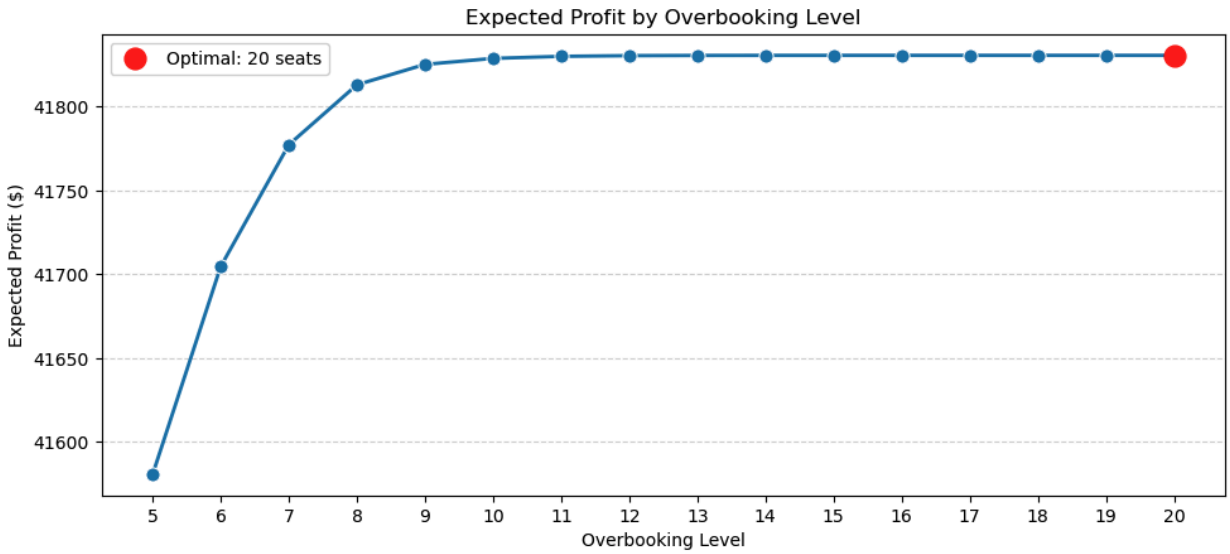
Despite its flexibility, this policy **does not significantly reduce customer impact**, as the frequency of denied boarding (71.2% vs. 69.9%) is slightly higher. However, this is offset by maintaining nearly full seat occupancy and maximizing ticket revenue over the sales window.



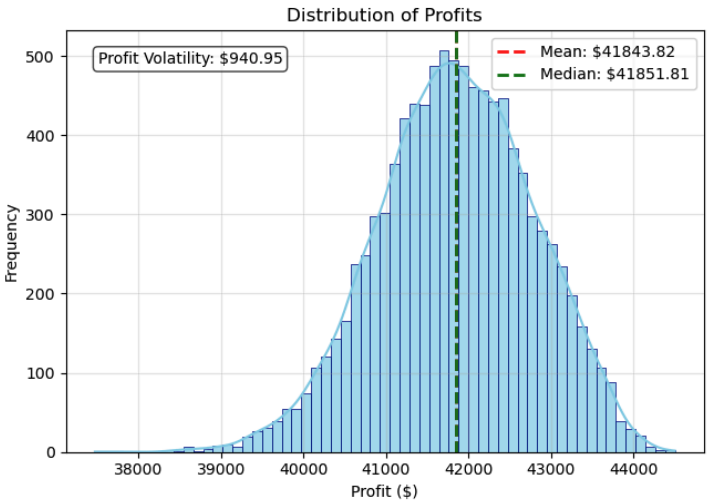
Scenario 4 : Incorporating Seasonality in Demand

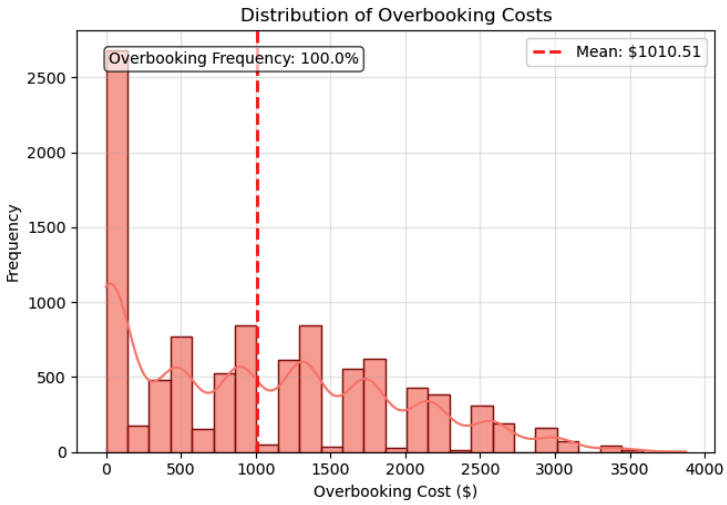
In this section we need to evaluate the performance of our optimal pricing and overbooking policy under realistic market behaviors. Specifically, we explore the effects of seasonality in demand while allowing the airline to opt out of selling coach tickets on certain days — a policy we refer to as the "no-sale coach option." As the departure date approaches, demand for both coach and first-class tickets increases. This realistically reflects heightened urgency and willingness to purchase in the days leading up to the flight. Combined with the no-sale coach option, this policy attempts to balance late-stage revenue maximization with early-stage inventory control.

In the below figure, we see that as overbooking level increases, expected profit increases at a slowing rate until leveling off after 10 seats. The reason we see expected profit at this steady level instead of decreasing is because although we allow overbooking beyond 10 seats, the option to stop selling coach tickets allows us to forgo overbooking costs. Meaning that beyond 10 extra allowed seats, we potentially still elect to sell around 110 coach seats. We chose to use an overbooking level of 20 going forward for this scenario.

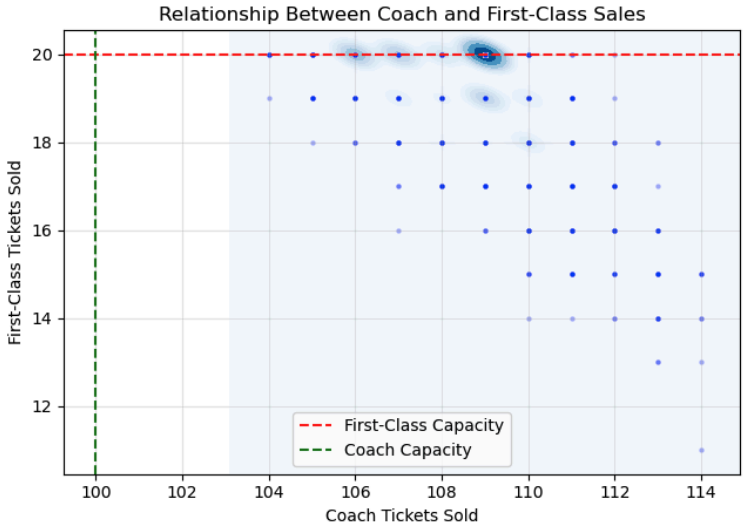


Seasonal demand boosts revenue potential by raising late-stage bookings, while the option to pause coach sales avoids unnecessary overbooking risk. However, the likelihood of early sales are also reduced, potentially reducing profit. As we discovered, the **expected discounted profit is $41830.46**, which is slightly **lower** than the expected profit from scenario 3 that doesn't incorporate seasonal demand but uses the same overlooking level and incorporates no-sale coach option. Using the optimal overbooking level of 20 determined for this scenario, we once again ran 10,000 simulations. Accordingly, profit is symmetrically centered around the mean of $41,843.82 with a standard deviation of $940.95. The mean is slightly lower, but the standard deviation is very similar (~$2 difference). Once again, all flights are overbooked, and the average overbooking cost is very similar: $1,010.51. The overbooking cost has a very right skewed distribution though notable extreme values.

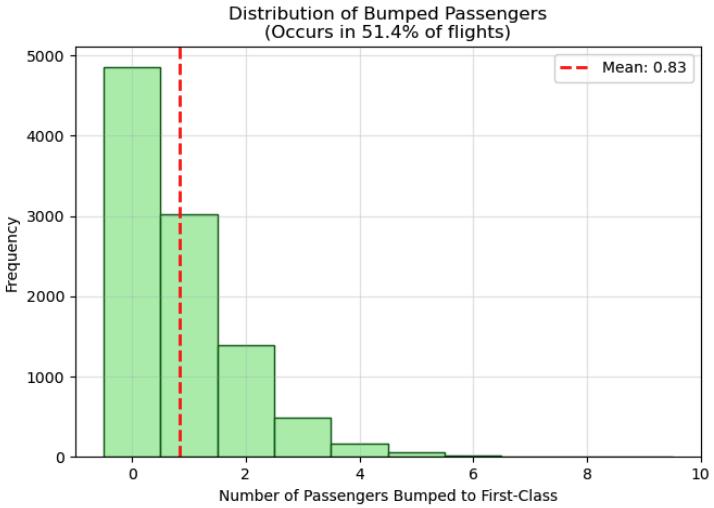


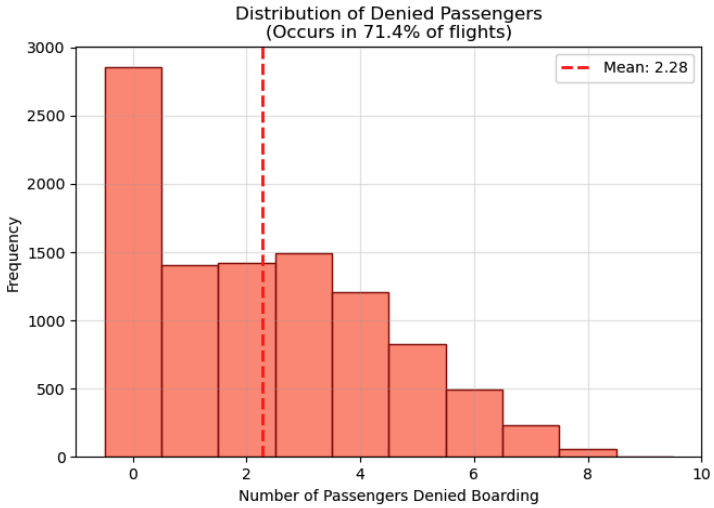


Just like in scenario 3, we also see that typically 109 coach tickets are sold even though we have the option to sell 120 tickets. The range of coach tickets sold is also similar, going from 104 to 114, and so is the range of first-class ticket sold, going from 11 to 20 with 20 being the most frequent.



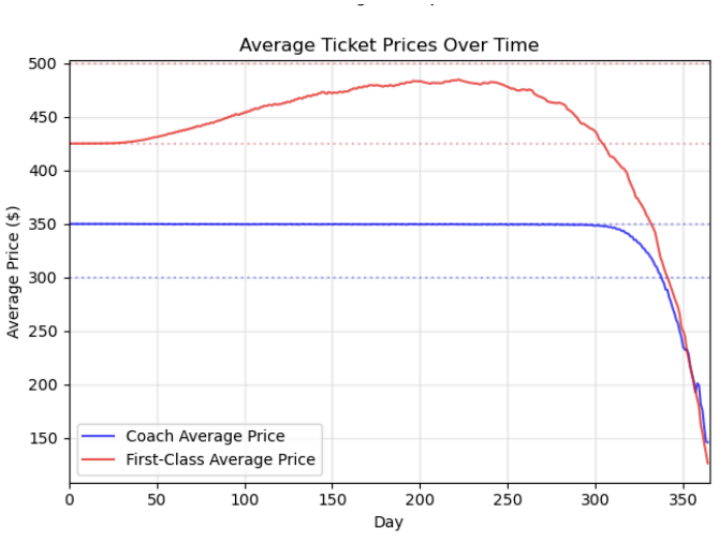
The customer experience is very similar to scenario 3 too. At least one passenger is bumped around 51.4% of the time and at least one passenger is denied around 71.4% of the time. The mean passengers bumped is 0.83 whereas the mean passengers denied is 2.28. We should consider if this level of denial is acceptable for our customer experience standards.

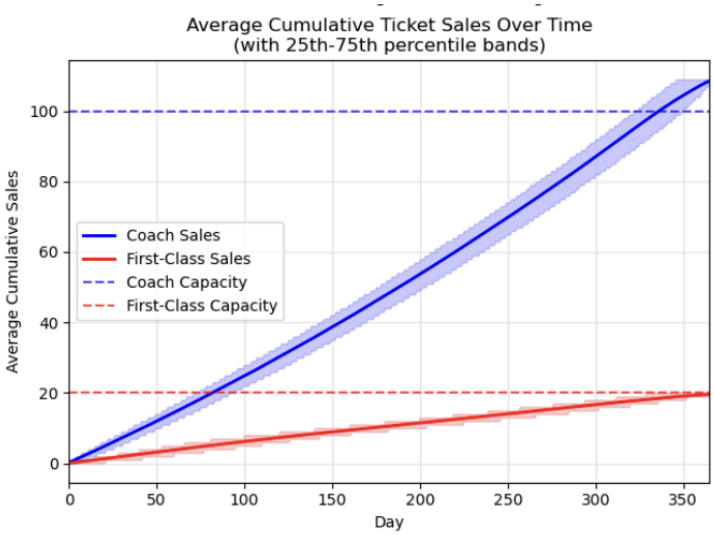




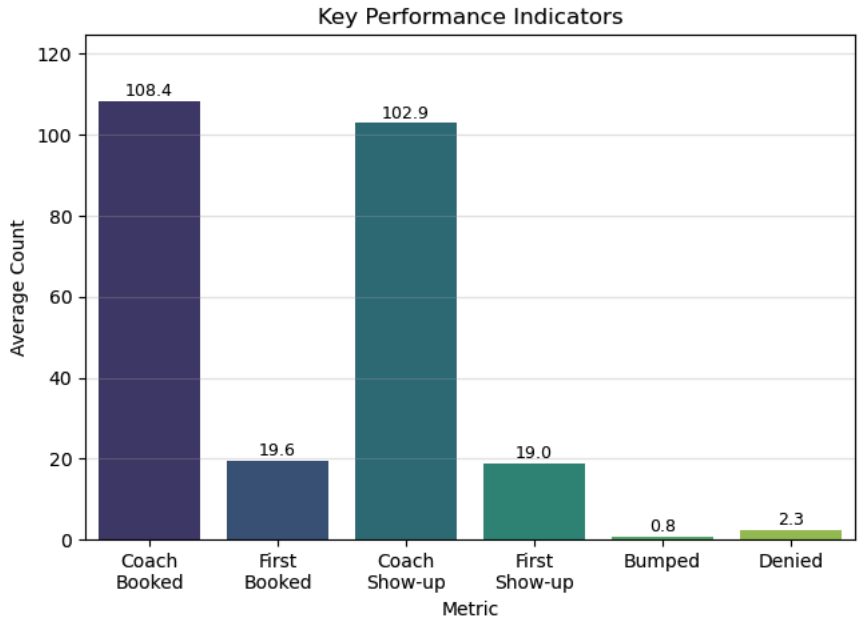
As for the ticket prices and sales over time, seasonal demand drives **slightly more aggressive ticket sales toward the end** to fill remaining seats as demand increases. The ticket prices for

both classes remain higher on day 300 in this scenario than in scenario 3. Furthermore, on day 50, around 13 coach tickets are sold for the seasonal demand scenario depicted here whereas around 18 tickets are sold for the static demand depicted in scenario 3. Given more sales are made later and later sales are discounted at a higher rate than earlier sales, scenario 4 has a lower lower profit than scenario 3. The option to not sell likely helps the airline to sell as many tickets as possible while minimizing overbooking costs.





Concluding this comparison, the main difference from including seasonal demand with the optimal policy from scenario 3 is decreased expected profit because more ticket sales occur later. These later sales are discounted more when calculating the profit, reducing the total overall profit even though the ticket sales, show-ups, and passengers bumped and denied remain nearly identical. You can compare the key performance indicators here with those in scenario 3 to see this for yourself.



Part 5. Simulation Comparison

With optimal overbooking and pricing strategies derived from our dynamic programming model, we now assess how these strategies perform under **realistic forward simulation**. We analyze both financial outcomes and customer experience using a Monte Carlo simulation of **10,000 flights** per policy.

I. Summary

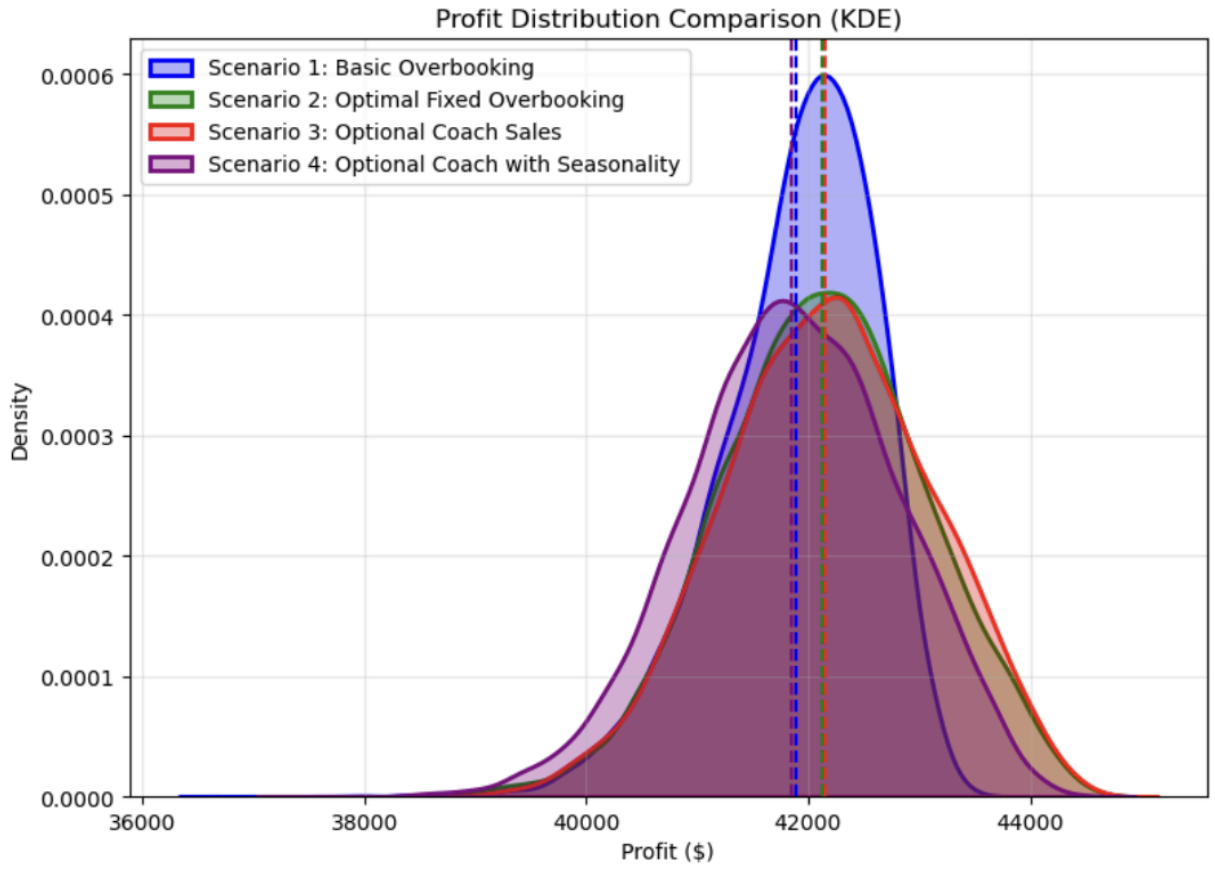
Based on the forward simulation results, Scenario 3, which includes the option not to sell coach tickets on a certain days, emerges as the most profitable strategy with an average profit of **$42,156.82,** representing a **0.63%** percent **improvement** over the baseline (Scenario 1). Although it comes with slightly higher volatility compared to Scenario 1, it performs favorably against Scenario 2 (the fixed optimal overbooking policy) by offering both **greater profit** and **enhanced control through selective ticket sales**. While scenario 4 does not implement new policies, it demonstrates how our expected profit may change under different assumptions and circumstances, namely seasonal demand. The average profit is now lower even though the same policy as scenario 3 is implemented due to later, more discounted sales.

```
Policy Comparison Summary:
```

Metric	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Expect.discounted profit	$41891.74	$42120.06	$42156.82	$41843.82
Profit Volatility	$710.57	$930.85	$942.89	$940.95
% Flights Overbooked	100.00%	100.00%	100.00%	100.00%
% Flights with Denied	26.68%	69.92%	71.22%	71.40%
% Flights with Bumping	19.87%	49.57%	50.85%	51.38%
Avg passengers Bumped	0.24	0.76	0.82	0.83
Avg passengers Denied	0.48	2.23	2.33	2.28
Avg. Coach Tickets Sold	104.96	108.16	108.39	108.36
Avg. First-Class Sold	19.80	19.61	19.57	19.56

```
Profit improvement over baseline: 0.55%, 0.63%, -0.11%
```

II. Profit Distributions



All policies yield profits centered around $41,800–$42,100, with slightly differing shapes and spreads.

- **Scenario 3 (Optional Coach Sales)** exhibits the **highest average profit** and a slightly right-shifted curve, indicating its **superior revenue performance** among the policies.

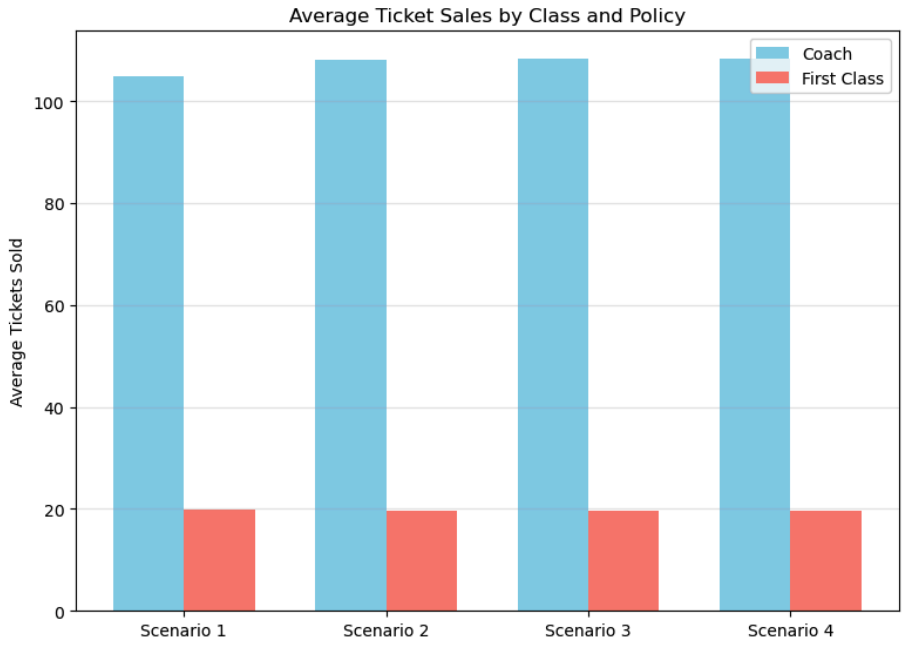
- **Scenario 1 (Basic Overbooking)** displays the **narrowest, most peaked distribution**, reflecting **lower volatility** but also a **lower profit ceiling**.

- **Scenario 4 (Optional Coach with Seasonality)** has the **lowest average profit**, due to reduced demand initially and increased demand later in the sales cycle.

- **Scenario 2 (Fixed Optimal Overbooking)** maintains a strong central peak just below Scenario 3's, offering **high and consistent profit**, though not the absolute maximum.

Overall, Scenario 3 has high profitability and manageable volatility, suggesting that introducing flexibility in coach ticket sales enhances revenue potential without excessive risk from increased overbooking levels.

III. Ticket Sales



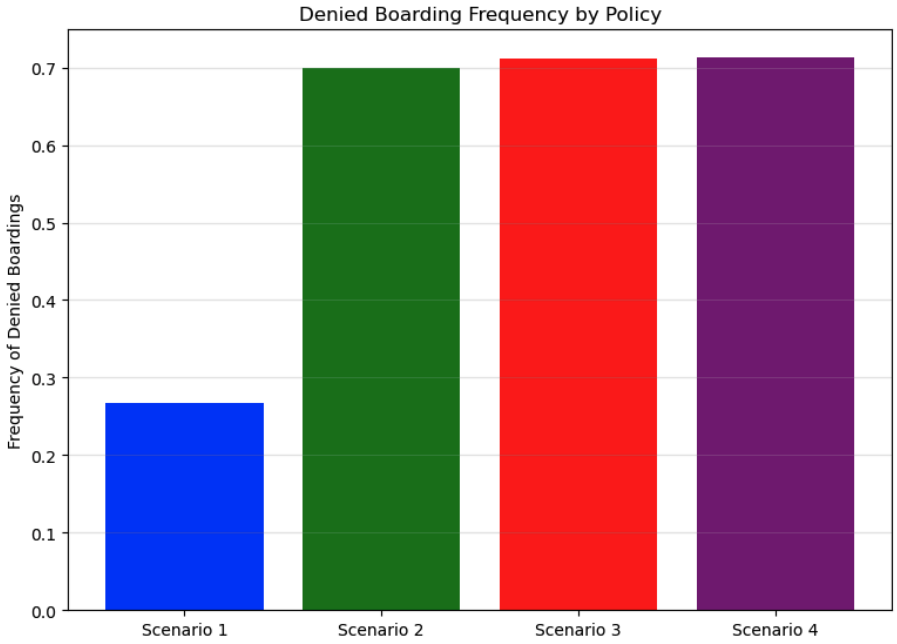
- Sales are consistently at or near capacity for both classes in all scenarios.
- The **Optional Coach Sale (Scenarios 3 & 4)** policies do not significantly reduce total ticket sales, confirming that flexibility does **not come at the cost of capacity utilization**.

Giving the airline the ability to **pause coach sales** does not reduce total bookings. Instead, it

enables **more strategic pacing** of those bookings, only overbooking when ticket sales have been low.

IV. Customer Experience

Profit is not the only important factor in determining the optimal policy though. We must also consider the customer experience. If scenario is the optimal policy, are we willing to deny service over 70% of the time to at least one customer? If we would prefer to foster customer relationships, perhaps we should consider reducing the overbooking at the expense of immediate profits as seen in the plot below.



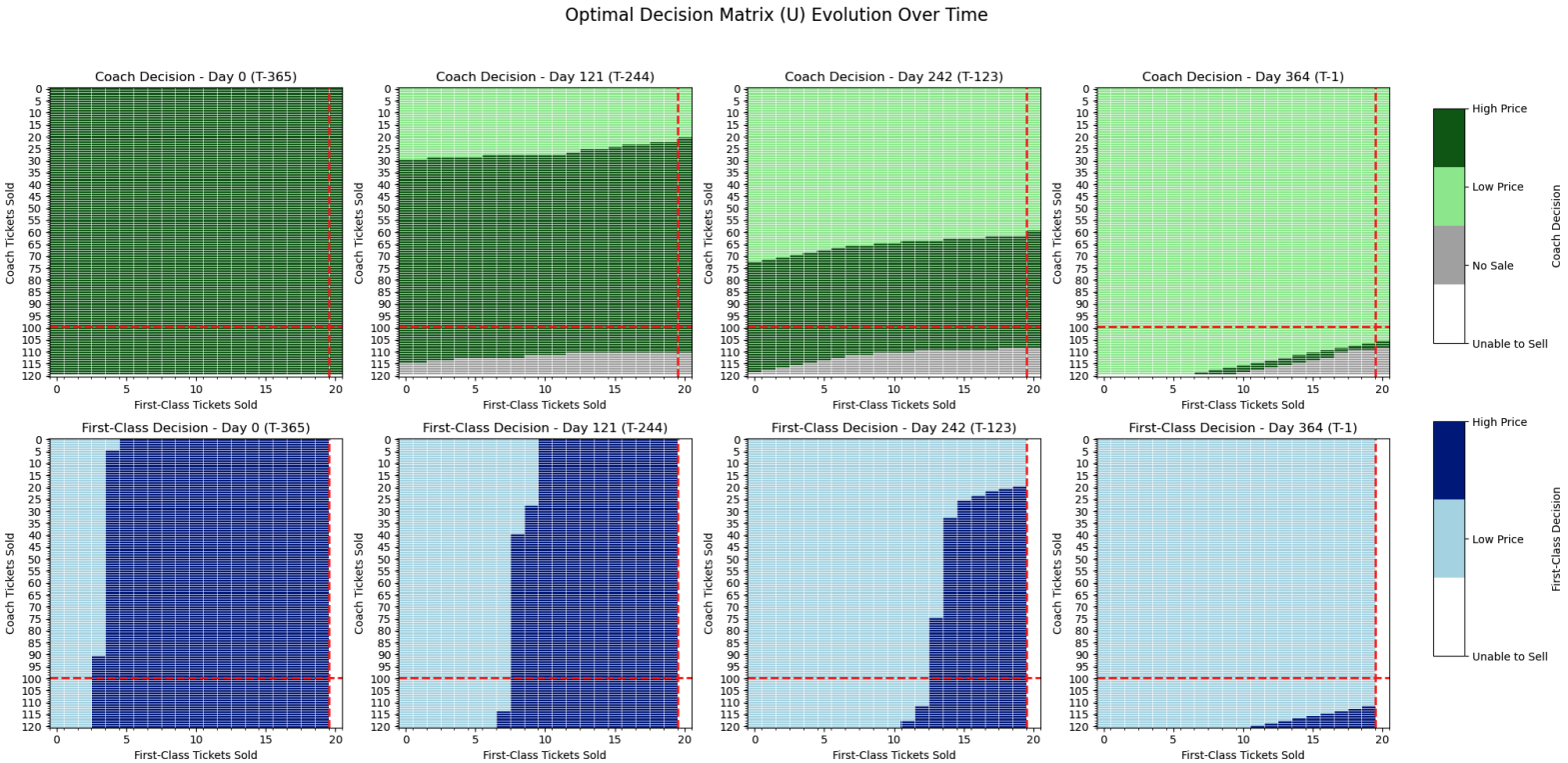
- **Scenario 1** results in the **lowest denial rate (~26.68%)**, due to conservative ticket selling.
- **Scenarios 2–4** all show **high denial frequencies (~69–71%)**, despite their profitability. This reflects the aggressive overbooking strategies baked into each optimized policy.

Policies that prioritize profit often do so at the expense of **passenger disruption**, which could result in reputational cost or regulatory concern. The trade-off here is clear and measurable.

Best Policy Analysis

Given our best policy was to allow overbooking and no coach sale for some days (scenario 3), we have created a figure to display the best pricing decision at each time depending on the number of first-class tickets sold and coach tickets. The best pricing decision comes from the

U-matrix for scenario 3. Below, you will see the pricing decision for 4 evenly spaced time points over the 365 day selling period. Each column represents a time period, and the row represents the pricing decision for coach (top) and first-class (bottom). On day 0 (leftmost column), you should always sell coach tickets at the high price, regardless of how many seats have been sold (which will be 0 at time 0). You should also sell first-class tickets at the low price at time 0. Even though you would sell first-class tickets at the high price if you had already sold 5 first-class tickets, that will never happen at time 0. Looking ahead to day 121, you should sell coach tickets at the high price if you have already sold at least around 30 coach tickets, though this number changes slightly depending on how many first-class tickets you have sold. At the same time, you should buy first-class tickets at the high price if you have sold more than about 8 first-class tickets, but again the exact number depends on how many coach tickets you have sold. As you move through the next few time frames, the low coach and first-class price are recommended when you have sold a relatively low number of tickets for that time point. At the last time period, you want to sell any remaining tickets, so you will prefer to sell at the low price, particularly when you aren't anywhere near capacity. The decision not to sell coach begins happening in an effort to manage further overbooking costs when you have oversold coach tickets and are nearing capacity in first-class. Overall, this figure provides a good overview of the pricing decision throughout time depending on the number of tickets sold. You should generally sell at the high price for coach unless tickets sales are relatively low for that time period, and you should begin selling first-class tickets at the low price and transition to the high price if you have sold a relatively high number of first-class seats. You can refer to the U-matrix output by the optimal pricing policy for exact pricing decisions though.



Conclusion

The forward simulation analysis across all four overbooking and pricing policies reveals a clear hierarchy in terms of profitability, volatility, and passenger experience. Among all strategies, the **Optional Coach Sale policy without seasonality (Scenario 3)** achieves the **average profit** of **$42,156.82**, narrowly outperforming the **Fixed Optimal Overbooking policy (Scenario 2)** at **$42,120.06**. While both strategies significantly improve upon the **Basic Overbooking benchmark (Scenario 1)**, which yields **$41,891.74**, these financial gains are offset by increases in customer disruption. Note: these are the average profits reported from the simulations, which have some variation. The same pattern emerges with the expected profits calculated using the Bellman equation though.

As confirmed in the Monte Carlo analysis, profit-focused strategies like Scenarios 2 and 3 lead to **denied boardings in over 69–71% of flights**, compared to just **26.68% under Scenario 1**, where bumping is also minimized (average of **0.24 passengers per flight**). This demonstrates the trade-off between profitability and customer satisfaction. Notably, **Scenario 4**, which combines coach-sale flexibility with seasonal demand modeling, results in the **lowest profit among the optimized policies ($41,843.82)**, but achieves the **least profit volatility** (**$940.95**), indicating its suitability for environments where **financial stability is prioritized** over absolute revenue.

Across all scenarios, seat utilization remains high, with average coach ticket sales ranging from **104.96 to 108.36** and first-class consistently near full capacity. This confirms that **selectively pausing coach sales** does not reduce total volume but instead enables more efficient and responsive demand shaping.

In summary, the choice of overbooking and pricing policy ultimately depends on the airline's strategic priorities. If **maximizing profit** is paramount, Scenario 3 is ideal. If **minimizing volatility** or **balancing customer satisfaction** is more critical, Scenario 1 offers a compelling alternative. The evidence presented through both numerical results and simulation graphics provides a robust foundation for making this operational trade-off.

Appendix

Exhibit 1: Calculate expected profit parameters

```
# Can calculate expected profit for any of the scenarios above using this function
def calculate_expected_profit(
    overbooking_level,        # Coach seats that can be overbooked
    allow_no_coach_sale=False, # Change for scenario 3
    seasonal_demand=False,    # Change for scenario 4
    # Set defaults for constant variables across scenarios based on presets
    coach_seats=coach_seats,          # Fixed across all scenarios
    first_class_seats=first_class_seats,     # Fixed across all scenarios
    coach_prices=coach_prices,  # Fixed across all scenarios
    first_class_prices=first_class_prices, # Fixed across all scenarios
    coach_show_prob=coach_show_prob,     # Fixed across all scenarios
    first_class_show_prob=first_class_show_prob, # Fixed across all scenarios
    coach_demand_prob=coach_demand_prob, # Base probabilities
    coach_demand_increase=coach_demand_increase, # Coach demand increase when first-class sold out
    first_class_demand_prob=first_class_demand_prob, # Base probabilities
    bump_cost=bump_cost,             # Cost to bump to first-class
    denied_cost=denied_cost,          # Cost to deny boarding
    days=days,                 # Fixed across all scenarios
    discount_rate=discount_rate        # Fixed across all scenarios
):
```

The only parameters you should change from scenario to scenario are the first 3 (overbooking_level, allow_no_coach_sale, and seasonal_demand). While the other parameters can be changed on a case by case basis, they are currently set to be the same across all scenarios for comparability.

Exhibit 2: calculate_expected_profit Bellman equation setup

```
# Initial state
t = days  # days until plane departs
c = 0    # coach tickets sold
f = 0    # first class tickets sold
C = coach_seats  # number of coach seats
F = first_class_seats   # number of first class seats
delta = 1/(1+discount_rate/days)

# Choice variables
p_c = coach_prices
p_f = first_class_prices  # first-class prices

# Dynamics
P_c = coach_demand_prob  # probability of selling coach ticket at each price
P_c_adj = np.array(P_c) + coach_demand_increase # coach probability increases if first-class sells out
P_f = first_class_demand_prob  # probability of selling first-class ticket at each price

# Terminal condition
T = days
c_cap = C + overbooking_level  # coach tickets that can be sold

# All possible state values
tValues = np.arange(T+1)  # all possible days until take off (0-365)
cValues = np.arange(c_cap+1)  # 0 to (100 + overbooking_level) coach tickets sold
fValues = np.arange(F+1)  # 0-20 first-class tickets sold

# Count possible state values
tN = len(tValues)  # 366
cN = len(cValues)  # (101 + overbooking_level)
fN = len(fValues)  # 21

# Set up V and U matrices
V = np.zeros((tN, cN, fN))  # 3D matrix
U = np.zeros((tN, cN, fN))
```

Exhibit 3: calculate_expected_profit() terminal value calculation

```
# Terminal condition calculation
for c_T in range(cN):
    for f_T in range(fN):
        exp_cost = 0
        for i in range(c_T + 1):
            for j in range(f_T + 1):
                P_i = stats.binom.pmf(i, c_T, coach_show_prob)
                P_j = stats.binom.pmf(j, f_T, first_class_show_prob)
                if i <= C:  # less coach passengers show up than coach seats available
                    cost = 0
                elif i <= C + (F - j):  # all extra coach passengers can be bumped to first-class
                    cost = bump_cost * (i - C)
                else:  # some passengers must be denied boarding
                    cost = bump_cost * (F - j) + denied_cost * (i - C - (F - j))
                exp_cost += P_i * P_j * cost
        V[tN-1, c_T, f_T] = -exp_cost
```

Exhibit 4: calculate_expected_profit() solving the Bellman equation

```
# Solve the bellman equation
for t in reversed(range(tN-1)):  # day 364 to 0
    # set the demand dependent on time if using seasonal demand
    if seasonal_demand:
        P_ct = np.array(P_c) * (.75 + t/730) # demand of coach at time t
        P_ct_adj = np.array(P_c_adj) * (.75 + t/730) # demand of coach at time t if first-class sold out
        P_ft = np.array(P_f) * (.75 + t/730) # demand of first class at time t
    else:
        P_ct = P_c # demand of coach at time t (constant)
        P_ct_adj = P_c_adj # demand of coach at time t if first-class sold out
        P_ft = P_f # demand of first class at time t
    for c in range(cN):  # 0 to (100 + overbooking_level) tickets sold
        for f in range(fN):  # 0 to 20 tickets sold
            if c == c_cap and f<F: # Can't sell more coach tickets, but can sell first-class
                # Low first-class price
                valueL = P_ft[0] * (p_f[0] + delta * V[t+1, c, f+1]) + (1-P_ft[0]) * delta * V[t+1, c, f]
                # High first-class price
                valueH = P_ft[1] * (p_f[1] + delta * V[t+1, c, f+1]) + (1-P_ft[1]) * delta * V[t+1, c, f]
                V[t, c, f] = max(valueL, valueH)
                if allow_no_coach_sale:
                    U[t, c, f] = np.argmax([valueL, valueH]) + 9  # 9 for -L, 10 for -H
                else:
                    U[t, c, f] = np.argmax([valueL, valueH]) + 6  # 6 for -L, 7 for -H

            elif c == c_cap and f == F:  # Both at capacity
                V[t, c, f] = delta * V[t+1, c, f]
                if allow_no_coach_sale:
                    U[t, c, f] = 11  # No sale possible
                else:
                    U[t, c, f] = 8
            elif c < c_cap and f == F:  # First-class is sold out but coach isn't
                # Low coach price
                valueL = P_ct_adj[0] * (p_c[0] + delta * V[t+1, c+1, f]) + (1-P_ct_adj[0]) * delta * V[t+1, c, f]
                # High coach price
                valueH = P_ct_adj[1] * (p_c[1] + delta * V[t+1, c+1, f]) + (1-P_ct_adj[1]) * delta * V[t+1, c, f]
                # No coach
                valueN = delta * V[t+1, c, f]

                if allow_no_coach_sale:
                    V[t, c, f] = max(valueL, valueH, valueN)
                    U[t, c, f] = np.argmax([valueL, valueH, valueN]) + 6  # 6 for L-, 7 for H-, 8 for N-
                else:
                    V[t, c, f] = max(valueL, valueH)
                    U[t, c, f] = np.argmax([valueL, valueH]) + 4  # 4 for L-, 5 for H-
```

```
            else:  # Both ticket types can potentially be sold
                # Calculate value for all four (six) price combinations (six combinations when no sale is allowed)

                # Low coach, Low first-class (LL)
                # expected profit calculate across both, one, or neither ticket purchased
                value_LL = P_ct[0] * P_ft[0] * (p_c[0] + p_f[0] + delta * V[t+1, c+1, f+1]) + \
                           P_ct[0] * (1-P_ft[0]) * (p_c[0] + delta * V[t+1, c+1, f]) + \
                           (1-P_ct[0]) * P_ft[0] * (p_f[0] + delta * V[t+1, c, f+1]) + \
                           (1-P_ct[0]) * (1-P_ft[0]) * delta * V[t+1, c, f]

                # Low coach, High first-class (LH)
                # expected profit calculate across both, one, or neither ticket purchased
                value_LH = P_ct[0] * P_ft[1] * (p_c[0] + p_f[1] + delta * V[t+1, c+1, f+1]) + \
                           P_ct[0] * (1-P_ft[1]) * (p_c[0] + delta * V[t+1, c+1, f]) + \
                           (1-P_ct[0]) * P_ft[1] * (p_f[1] + delta * V[t+1, c, f+1]) + \
                           (1-P_ct[0]) * (1-P_ft[1]) * delta * V[t+1, c, f]

                # High coach, Low first-class (HL)
                # expected profit calculate across both, one, or neither ticket purchased
                value_HL = P_ct[1] * P_ft[0] * (p_c[1] + p_f[0] + delta * V[t+1, c+1, f+1]) + \
                           P_ct[1] * (1-P_ft[0]) * (p_c[1] + delta * V[t+1, c+1, f]) + \
                           (1-P_ct[1]) * P_ft[0] * (p_f[0] + delta * V[t+1, c, f+1]) + \
                           (1-P_ct[1]) * (1-P_ft[0]) * delta * V[t+1, c, f]

                # High coach, High first-class (HH)
                # expected profit calculate across both, one, or neither ticket purchased
                value_HH = P_ct[1] * P_ft[1] * (p_c[1] + p_f[1] + delta * V[t+1, c+1, f+1]) + \
                           P_ct[1] * (1-P_ft[1]) * (p_c[1] + delta * V[t+1, c+1, f]) + \
                           (1-P_ct[1]) * P_ft[1] * (p_f[1] + delta * V[t+1, c, f+1]) + \
                           (1-P_ct[1]) * (1-P_ft[1]) * delta * V[t+1, c, f]

                # No coach, Low first-class (NL)
                # expected profit calculate across first-class purchased or not
                value_NL = P_ft[0] * (p_f[0] + delta * V[t+1, c, f+1]) + \
                           (1-P_ft[0]) * delta * V[t+1, c, f]

                # No coach, High first-class (NH)
                # expected profit calculate across first-class purchased or not
                value_NH =  P_ft[1] * (p_f[1] + delta * V[t+1, c, f+1]) + \
                           (1-P_ft[1]) * delta * V[t+1, c, f]

                if allow_no_coach_sale:
                    values = [value_LL, value_LH, value_HL, value_HH, value_NL, value_NH]
                    V[t, c, f] = max(values)
                    U[t, c, f] = np.argmax(values)  # 0 = LL, 1 = LH, 2 = HL, 3 = HH, 4 = NL, 5 = NH
                else:
                    values = [value_LL, value_LH, value_HL, value_HH]
                    V[t, c, f] = max(values)
                    U[t, c, f] = np.argmax(values)  # 0 = LL, 1 = LH, 2 = HL, 3 = HH
```

Notice the adjustments made using the if/else statements if allow_no_coach_sale=True and seasonal_demand=True. When allow_no_coach_sale=True, the maximum value is calculated over 3 additional values: when the coach is sold out (valueN) or both tickets can potentially be sold (valueNH and valueNL) where N represents no coach sale. Seasonal demand affects the probability of a sale each day (P_ct and P_ft).

Exhibit 5: calculate-expected_profit() no coach sale implementation

```
for c in range(cN):  # 0 to (100 + overbooking_level) tickets sold
    for f in range(fN):  # 0 to 20 tickets sold
        if c == c_cap and f<F: # Can't sell more coach tickets, but can sell first-class
            # Low first-class price
            valueL = P_ft[0] * (p_f[0] + delta * V[t+1, c, f+1]) + (1-P_ft[0]) * delta * V[t+1, c, f]
            # High first-class price
            valueH = P_ft[1] * (p_f[1] + delta * V[t+1, c, f+1]) + (1-P_ft[1]) * delta * V[t+1, c, f]
            V[t, c, f] = max(valueL, valueH)
            if allow_no_coach_sale:
                U[t, c, f] = np.argmax([valueL, valueH]) + 9  # 9 for -L, 10 for -H
            else:
                U[t, c, f] = np.argmax([valueL, valueH]) + 6  # 6 for -L, 7 for -H

        elif c == c_cap and f == F:  # Both at capacity
            V[t, c, f] = delta * V[t+1, c, f]
            if allow_no_coach_sale:
                U[t, c, f] = 11  # No sale possible
            else:
                U[t, c, f] = 8
        elif c < c_cap and f == F:  # First-class is sold out but coach isn't
            # Low coach price
            valueL = P_ct_adj[0] * (p_c[0] + delta * V[t+1, c+1, f]) + (1-P_ct_adj[0]) * delta * V[t+1, c, f]
            # High coach price
            valueH = P_ct_adj[1] * (p_c[1] + delta * V[t+1, c+1, f]) + (1-P_ct_adj[1]) * delta * V[t+1, c, f]
            # No coach
            valueN = delta * V[t+1, c, f]

            if allow_no_coach_sale:
                V[t, c, f] = max(valueL, valueH, valueN)
                U[t, c, f] = np.argmax([valueL, valueH, valueN]) + 6  # 6 for L-, 7 for H-, 8 for N-
            else:
                V[t, c, f] = max(valueL, valueH)
                U[t, c, f] = np.argmax([valueL, valueH]) + 4  # 4 for L-, 5 for H-
```

Exhibit 6: calculate_expected_demand() seasonal demand implementation

```
# Solve the bellman equation
for t in reversed(range(tN-1)):  # day 364 to 0
    # set the demand dependent on time if using seasonal demand
    if seasonal_demand:
        P_ct = np.array(P_c) * (.75 + t/730) # demand of coach at time t
        P_ct_adj = np.array(P_c_adj) * (.75 + t/730) # demand of coach at time t if first-class sold out
        P_ft = np.array(P_f) * (.75 + t/730) # demand of first class at time t
    else:
        P_ct = P_c # demand of coach at time t (constant)
        P_ct_adj = P_c_adj # demand of coach at time t if first-class sold out
        P_ft = P_f # demand of first class at time t
```