Optimization II
Project 2 - Airline Optimal Dynamic Pricing
Group 21

Kaushik Kumaran (kk34537), Soumya Swarupa Nayak (sn25829), Suchit Das (sd38448),
Vishal Gupta (vg22846)

Introduction

For the airlines it is essential to put as many people as possible in the flights to maximize the revenue that requires pricing the tickets aptly so that maximum number of passengers buys the tickets, and even if they do not show overall revenue can be maximized by overbooking to fill up the seats left vacant due to no-show at the day of the trip. Creating such a pricing policy involves setting up the price for different classes – coach and first – for each day from the listing till the day of the trip. Identifying the optimal pricing policy requires looking at the overall profit for all choices of coach and first-class prices as well as the total overbooking allowed for both classes.

In the analysis we will compare two strategies, first *varied overbooking strategy* using different overbooking levels and then choosing the one with the highest expected profit. The second *no-sale flexibility overbooking* strategy involves overbooking the coach to fill up the first class as well, but we will also have the flexibility not to put the flight for booking for a day. In both cases we will arrive at dynamic pricing policy by using dynamic programming to maximize profit at each day to set the prices, high/low/no-sale, this can involve transferring some coach passengers to first class and kicking off some passengers from either of the two classes in case more people show up for seats on the flight because of overbooking.

The recommendation for the best strategy will be based on the performance of both approached on the maximum expected discounted profit, overbooking cost, overbooking percentage, and a few other metrics.

Mathematical Formulation

Discounted profit depends on both cost and revenue, that will be discounted to arrive at profit at the start of the listing date. The revenue will be from the cost of selling the coach and first-class tickets each day, though costs will be incurred only on the day of flight when the airline must upgrade coach passengers or kick off passengers from the flight due to overbooking.

The formulation for varied overbooking strategy is below -

State variables

1. Coach tickets c_i
2. First tickets f_i
3. day from flight listing t

Choices

1. Price of coach class ticket P_c - high H_c or low L_c
2. Price of first class ticket P_f - high H_f or low L_f

Value function

$$V(c, f, t) = \sum (profit@t + i) * \delta^i \; for \; i \in [0, T - t]$$

where δ is the discounting factor for net present value

At the day of the flight, T, there will not be any more flight booking hence revenue will be zero, hence profit will be composed on cost due to class upgrade or kicking the passengers from the flight

$$V(c, f, T) = -E[cost \; if \; sold \; c \; coach \; tickets \; and \; f \; first \; class \; tickets]$$

Expected cost at the day of flight can be written as below -

$$E[cost \; for \; c \; coach \; tickets \; and \; f \; first \; class \; tickets] =$$

$$\sum_{i=0}^{c} \sum_{j=0}^{f} p(i|c) * p(j|f) * overbooking \; cost \; to \; accomodate \; i \; in \; coach \; and \; f \; in \; firstclass$$

Bellman equation

$$V(c, f, t) = \max_{p} E[revenue@t|p] + E[V(c^{'}, f^{'}, t + 1)|p]$$

$$where \; p \in (H_c, H_f), (H_c, L_f), (L_c, H_f), (L_c, L_f)$$

Both revenue and value function for each state will depend on the price choice p. There are four possible outcome for any price choice p -

1. ticket sale for both classes
2. ticket sale for coach but not sale for first,
3. ticket sale for first class but not coach
4. no sale for both classes

Expected revenue & value function at t will be as below -

$$E[revenue|p] = (c_i + f_i) * p(coachsale) * p(firstsale) +$$

$$c_i * p(coachsale) * p(nofirstsale) +$$

$$f_i * p(nocoachsale) * p(firstsale) +$$

$$0 * p(nocoachsale) * p(nofirstsale)$$

$$E[value function|p] = V(c + 1, f + 1, t + 1) * p(coachsale) * p(firstsale) +$$

$$V(c + 1, f, t + 1) * p(coachsale) * p(nofirstsale) +$$

$$V(c, f, t + 1) * p(nocoachsale) * p(firstsale) +$$

$$V(c, f, t + 1) * p(nocoachsale) * p(nofirstsale)$$

Boundary condition

Since no more can be sold for a class if all the seats including the overbooking are sold, hence value function will only be just a discounted value function for next day, there will be a three boundary condition as below

$$V(C, F, t) = V(C, F, t + 1) * \delta$$

$$V(c, F, t) = V(c, F, t + 1) * \delta$$

$$V(C, f, t) = V(C, f, t + 1) * \delta$$

where C - total coach seats + overbooking level and F - total first class seats

For varied overbooking, the formulation will remain the same except that there will be two more choices available due to the no-sale option for coach tickets on any day. More details regarding the high/low price and other factors for both strategies are in their respective sections.

Strategy 1: Optimal Overbooking

Policy objective

Under this policy the airline can overbook the flight. The airline must incur a cost on overbooking if more passengers show up than seats available, so we must find the optimal pricing policy and overbooking value that leads to the maximum profit. The airline has 365 days to sell 100 coach seats and 20 first-class seats which can be overbooked by 5 to 15 seats.

	Coach seats/ Probability of sale	First-class seats/ Probability of sale
Low price	$300 (0.65)	$425 (0.08)
High price	$350 (0.30)	$500 (0.04)

Each coach ticket holder shows up to the flight with probability 95%, and each first-class ticket holder shows up to the flight with probability 97%.

Estimation of the cost on the day of the flight

Cost associated with overbooking in coach and first class will be incurred only on the day of the flight. It can be calculated for all values of tickets sold in coach(c) and in first-class(f).

- There will be no cost incurred if the tickets sold for both are less than the seats available.
- If the number of tickets sold in first class is more than the available seats, then all the passengers must be bumped off the plane, cost of which will be [number of passengers bumped off the plane] *$425
- If the number of tickets sold on the coach is more than the available seats, there will be two types of costs i.e.,
    - Transfer cost: If first class seats are available to transfer the passengers from coach to first class, cost will be [number of passengers bumped off to first class] *$50
    - If first class seats are not available to transfer, cost will be [number of passengers bumped off the plane] *$425
- If the number of tickets sold in both coach and first-class is more than the available seats, cost will be [number of passengers in both coach and first-class bumped off the plane] *$425

Probability array can be calculated using binomial distribution with tickets sold and probability of showing up in each class.

```
# getting probabilities, binom.pmf(r, n, p)
prob_arr[i, j] =  binom.pmf(i, c, showup_probs['c']) * binom.pmf(j, f, showup_probs['f'])
```

$$\text{total cost} = \sum_{i=0}^{c} \sum_{j=0}^{f} \text{(Cost of i,j passengers)*(Probability of i,j passengers showing up)}$$

Expected revenue and value function

$$V(c, f, t) = \max_{P} \text{E(Revenue today} + \delta \text{V tomorrow|Price choices)}$$

Since the airline can charge high or low prices for both coach and first-class: price choices = [('H', 'H'), ('H', 'L'), ('L', 'H'), ('L', 'L')]. The function for revenue today and value function for tomorrow are as below –

Expected revenue for today -

```
# expected revenue
# given the price of coach and first class, calculating the expected revenue based on
# probability of sale
def exp_rev(c, f, cp, fp, V, prices, sell_probs):
    rev = 0
    if (c == V.shape[0] - 1) & (f == V.shape[1] - 1):
        rev = 0
    elif (c == V.shape[0] - 1) & (f < V.shape[1] - 1):
        rev = (
            prices['f'][fp] * sell_probs['f'][fp] +
            0 * (1 - sell_probs['f'][fp])
        )
    elif (c < V.shape[0] - 1) & (f == V.shape[1] - 1):
        rev = (
            prices['c'][cp] * sell_probs['c'][cp] +
            0 * (1 - sell_probs['c'][cp])
        )
    else:
        rev = (
            (prices['c'][cp] + prices['f'][fp]) * sell_probs['c'][cp] * sell_probs['f'][fp] +
            prices['c'][cp] * sell_probs['c'][cp] * (1 - sell_probs['f'][fp]) +
            prices['f'][fp] * (1 - sell_probs['c'][cp]) * sell_probs['f'][fp] +
            0 * (1 - sell_probs['c'][cp]) * (1 - sell_probs['f'][fp])
            )

    return rev
```

Value function for tomorrow –

```
def exp_value_fn(c, f, t, cp, fp, V, sell_probs):

    if (c == V.shape[0]-1) & (f == V.shape[1]-1):
        value_fn = V[c, f, t+1]
    elif (c == V.shape[0]-1) & (f < V.shape[1] - 1):
        value_fn = (
            V[c, f+1, t+1] * sell_probs['f'][fp] +
            V[c, f, t+1] * (1 - sell_probs['f'][fp])
        )
    elif (c < V.shape[0]-1) & (f == V.shape[1] - 1):
        value_fn = (
            V[c+1, f, t+1] * sell_probs['c'][cp] +
            V[c, f, t+1] * (1 - sell_probs['c'][cp])
        )
    else:
        value_fn = (
            V[c+1, f+1, t+1] * sell_probs['c'][cp] * sell_probs['f'][fp] +
            V[c+1, f, t+1] * sell_probs['c'][cp] * (1 - sell_probs['f'][fp]) +
            V[c, f+1, t+1] * (1 - sell_probs['c'][cp]) * sell_probs['f'][fp] +
            V[c, f, t+1] * (1 - sell_probs['c'][cp]) * (1 - sell_probs['f'][fp])
        )
    return value_fn
```

Dynamics with Bellman equation to get optimal policy for each state

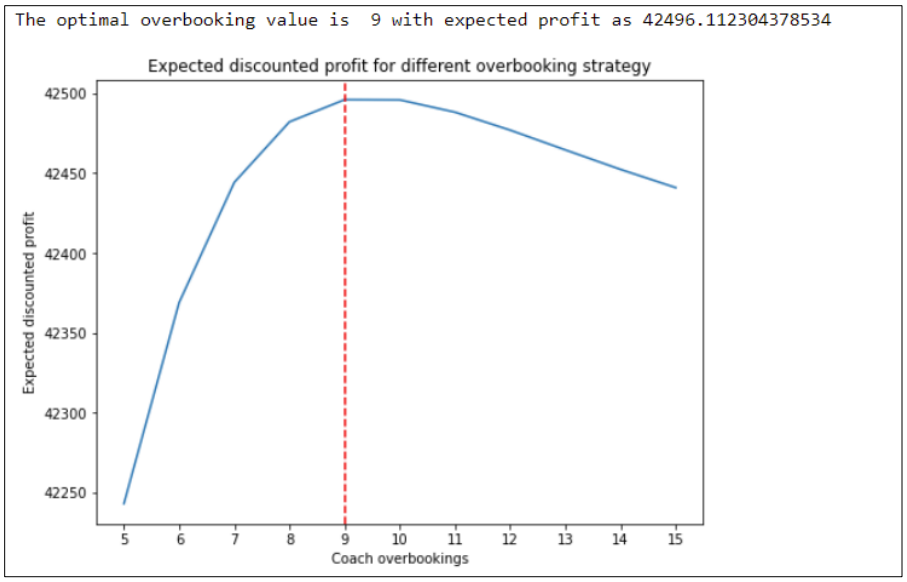
We will calculate the negative of expected cost for value function on the last day for all values of tickets sold in coach and first class.

```
def bellman_dynamic_value_fn(V, U, sell_probs, prices):
    price_choices = [('h', 'h'), ('h', 'l'), ('l', 'h'), ('l', 'l')]

    for t in reversed(range(V.shape[2]-1)):
        for c in range(V.shape[0]):
            for f in range(V.shape[1]):
                rev_choice = np.array(
                    [exp_rev(c, f, cp, fp, V, prices, sell_probs) for cp, fp in price_choices])

                value_fn_choice = np.array(
                    [exp_value_fn(c, f, t, cp, fp, V, sell_probs) for cp, fp in price_choices])

                total_value = rev_choice + disc_daily*value_fn_choice

                V[c, f, t] = np.max(total_value)
                U[c, f, t] = np.argmax(total_value)

    return V, U
```

Based on the Bellman equation, we can calculate the profit value for each of the overbooking values (ranging from 5 to 15 seats) to find the optimal overbooking policy providing highest return in terms of profit.

```
coach_overbooked_list = range(5, 16)
discounted_value_list = []
first_overbooked = 0
V_list = []
U_list = []

for coach_overbooked in coach_overbooked_list:
    print('Getting discounted value for overbooking coach seats of ', coach_overbooked)
    V_init, U_init = value_state_init(coach_overbooked, first_overbooked, overbooking_cost, showup_probs)
    V, U = bellman_dynamic_value_fn(V_init, U_init, sell_probs, prices)
    V_list.append(V)
    U_list.append(U)
    discounted_value_list.append(V[0, 0, 0])
```

So, the airline gets the maximum profit of $42496 when 9 seats are overbooked. Maximum discounted profit for different overbooking is below -



Strategy 2: No-sale flexibility with complete overbooking

Policy Objective

Under this policy the airlines are allowed to overbook in similar lines with the optimal overbooking policy. However, there is an added caveat to the policy in the sense that it allows airlines to choose to sell no coach tickets on a day if it wants. In a way the airlines have the option to force the demand to 0 on any day. This allows the airlines to stop over-selling the tickets not just based on how many they have already sold, but also on how many days they have left to sell. Like the original policy here we are assuming that there are 365 days until the plane departs. The seat distribution is such that there are 100 seats on the coach and 20 seats in the first-class. The high – low price distribution for different seats along with the probabilities for the ticket being sold can again be given as:

	Coach seats/ Probability of Ticket Sold	First-class seats/ Probability of Ticket Sold
Low price	$300 (0.65)	$425 (0.08)
High price	$350 (0.30)	$500 (0.04)

Apart from the Low price and High price option, under the new policy the airline will be allowed to sell no coach tickets if it wants. So now there will be 3 choices for the coach tickets – High Price, Low Price, and No Sale. In accordance with the above, our price distribution of the coach seats now looks as follows:

	Coach seats/ Probability of Ticket Sold	First-class seats/ Probability of Ticket Sold
Low price	$300 (0.65)	$425 (0.08)
High price	$350 (0.30)	$500 (0.04)
No Sale	$0 (0)	NA

The no sale choice does not apply for the first-class tickets. The estimation of the costs is calculated in line with the cost estimations for the optimal overbooking policy and can be enumerated as below:

- There are no costs involved if the no of tickets is less than the number of seats available
- If the number of tickets sold in the First-Class is more than the number of tickets available, then all the passengers are bumped off with the cost being: [No of Passengers bumped off] *425
- If the number of tickets sold in Coach seat are more than the number of seats available then we can either transfer the passengers to first class with the cost being: [No of passengers transferred] * $50 or if First-Class seats are not available then the passengers need to be bumped off in which case the cost comes out as: [No of passengers bumped off] * $425
- If the number of tickets sold in both coach and first-class is more than the available seats, cost will be [number of passengers in both coach and first-class bumped off the plane] *$425

Dynamics with the Bellman Equation

We will be calculating the expected cost for values function on the last day for all values of tickets sold in First Class and Coach tickets with the added benefit of no sale value. The function for the Bellman equation can be given as:

```
def bellman_dynamic_nosale_value_fn(V, U, sell_probs, prices):
    price_choices = [('h', 'h'), ('h', 'l'), ('l', 'h'), ('l', 'l'), ('ns', 'h'), ('ns', 'l')]

    for t in reversed(range(V.shape[2]-1)):
        for c in range(V.shape[0]):
            for f in range(V.shape[1]):
                rev_choice = np.array(
                    [exp_rev(c, f, cp, fp, V, prices, sell_probs) for cp, fp in price_choices])

                value_fn_choice = np.array(
                    [exp_value_fn(c, f, t, cp, fp, V, sell_probs) for cp, fp in price_choices])

                total_value = rev_choice + disc_daily*value_fn_choice

                V[c, f, t] = np.max(total_value)
                U[c, f, t] = np.argmax(total_value)

    return V, U
```

Based on the Bellman equation, we can go ahead and calculate the discounted values for overbooking coach seats of 20 to get the optimal return in terms of profit.

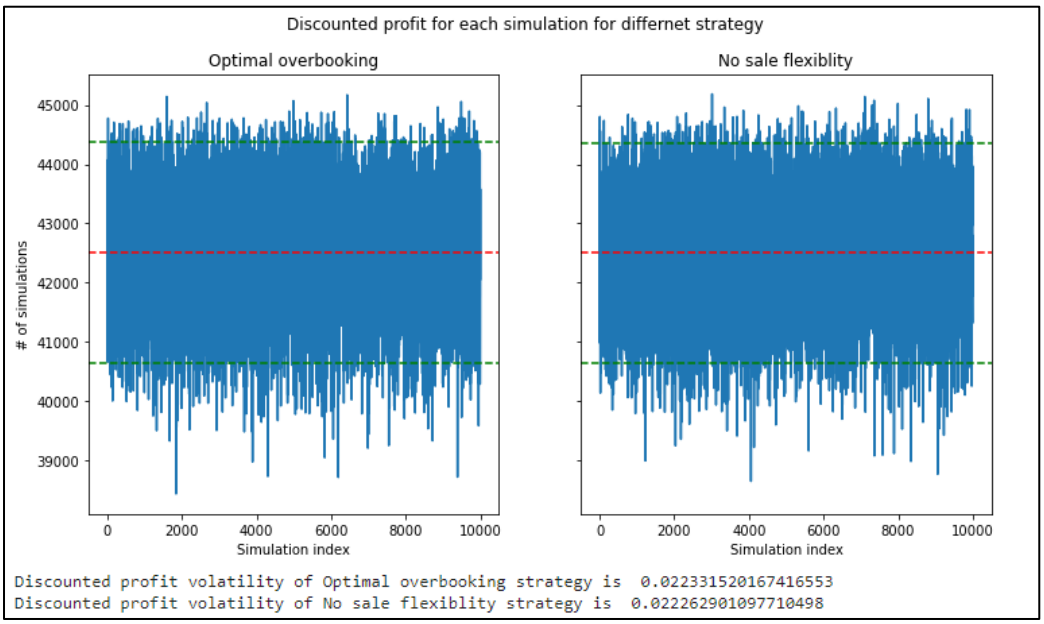
```
coach_overbooked = 20
first_overbooked = 0

sell_probs_ns = sell_probs.copy()
sell_probs_ns['c']['ns'] = 0

prices_ns = prices.copy()
prices_ns['c']['ns'] = 0

print('Getting discounted value for overbooking coach seats of ', coach_overbooked)
V, U = value_state_init(coach_overbooked, first_overbooked, overbooking_cost, showup_probs)
V, U = bellman_dynamic_nosale_value_fn(V, U, sell_probs_ns, prices_ns)
discounted_value_no_sale = V[0, 0, 0]
```

Comparison of two strategies

To compare the outcomes of both strategies, 10,000 simulations will be performed and performance metrics like overbooking cost, price volatility, etc. will be evaluated. Each simulation will involve sampling out from the probability distribution for a passenger buying a ticket for a day given the prices for both classes. The simulations will also involve sampling for number of passengers showing up on the day of the flight from binomial distribution given the number of tickets sold, the based on the optimal pricing policy obtained through dynamic programming.

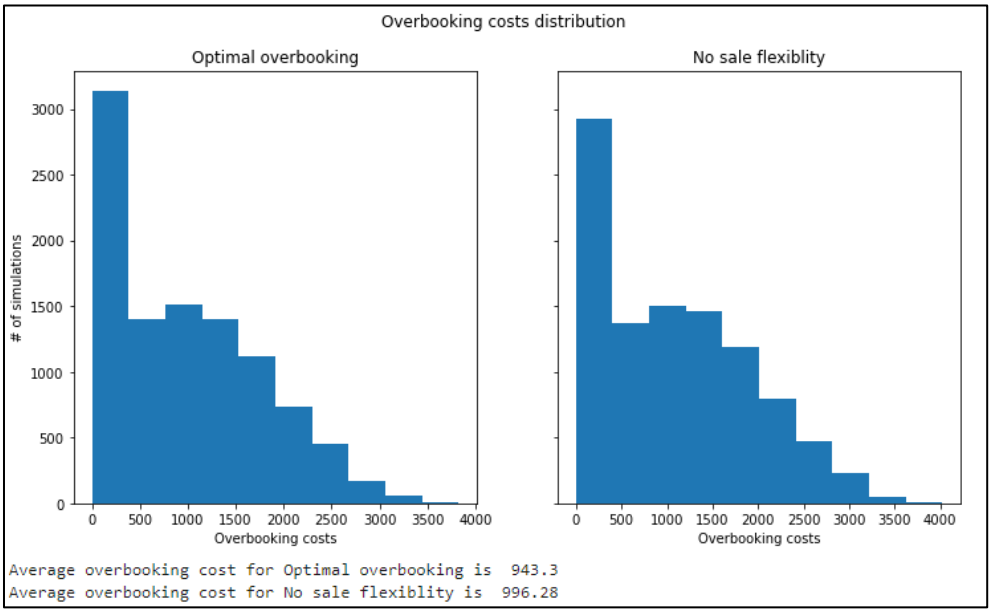
The next step is to compare the effectiveness of both the strategies:

1. Firstly, we compare the discounted profit with both the optimal overbooking strategy and the no sale flexibility strategy. The figure below shows the results for the simulation



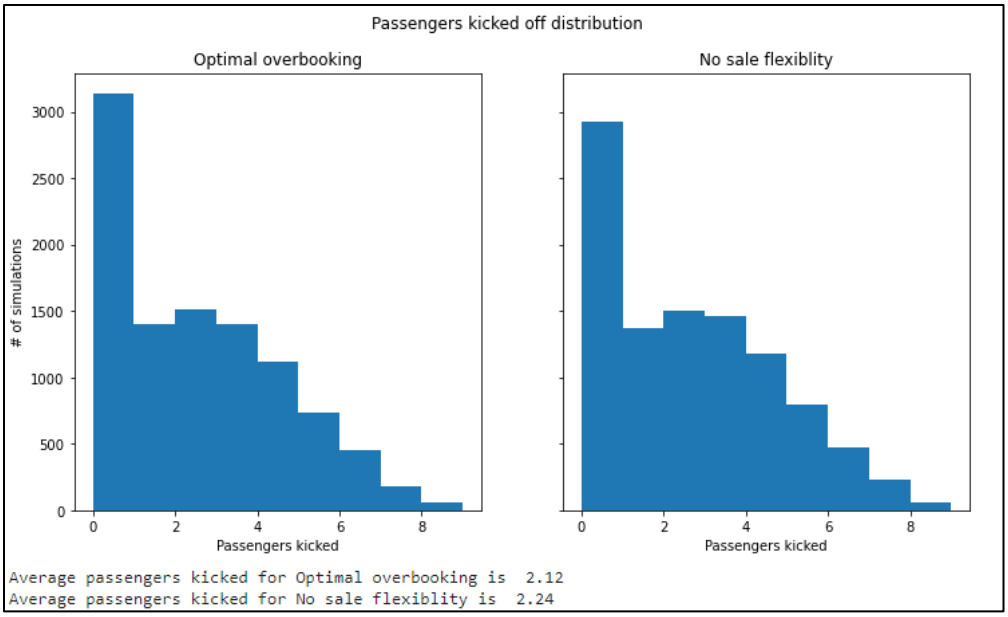
Inference: Although the mean profits for 'No sale flexibility' strategy are slightly higher than the 'Optimal overbooking' strategy, we see that the upper and lower bounds are similar. Also, the profit volatility, defined as the ratio of standard deviation to mean, is similar for the strategies at about 2.2%, leading to similar confidence intervals. Looking at the confidence intervals we can say the expected profit for both strategies does not have a statistically significant difference.

2. Next, we compare the **overbooking costs** for both the strategies.

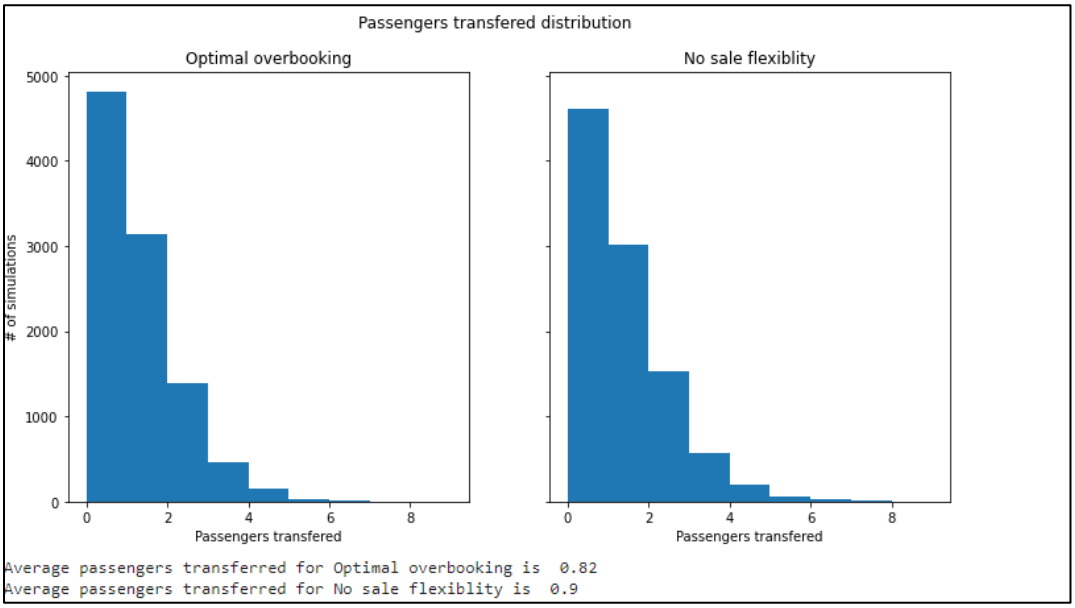


Inference: We see that the overbooking cost of No Sale Flexibility is higher while compared to the optimal overbooking strategy

3. The next comparison is the **kickoff** distribution as well as the **transfer** distribution.



We see that the average passengers kicked off for no sale flexibility is quite higher than passengers kicked off for optimal overbooking



The average passengers transferred for no sale flexibility is higher than passengers transferred for optimal overbooking.

Other Findings:

- The probability of passengers getting kicked off or transferred in optimal overbooking strategy is 83.1% while it is 82.2% for no sale strategy
- The probability of passengers getting kicked off in optimal overbooking strategy is 68.6% while it is 70.7% for no sale strategy

Conclusion

Although we see that the overall expected profit is slightly higher in the case of the *No Sale Flexibility* strategy, the overbooking cost, kickoff rates and the transfer rates are better than in the *Optimal Overbooking* strategy. Thus, from the customer retention standpoint, given that the overall profitability is quite similar and there is no statistically significant difference, our **recommendation is to go with the Optimal Overbooking strategy** since their kickoff rates and transfer rates are lower thereby providing a better customer experience.