Project 2: Airlines Dynamic Ticket Pricing

Group 8 Members: Praneet Kumar Alamuri (pa22222), Sai Bhargav Tetali (srt2578), Rudraksh Garg (rg44778), Parthiv Borgohain (pb25347)

INTRODUCTION

The business for an airline company seems simple: put as many passengers as possible on the flight to maximize revenue. However, they also need to regularly optimize the air fares so that the maximum customers buy tickets from them. Additionally, the revenue can also be maximized by overbooking the flight, that is, to sell more tickets than available on the flight since there is a possibility of no shows on the day of travel or vacancies that might be present in first class.

An airline company thus needs to have a separate pricing policy for both the classes i.e. coach and first class and they also need to dynamically change the prices of both the classes on a daily basis. The decision on pricing is thus based on 2 factors - permissible overbooking quantity and the overall profit that can be expected. We will explore 2 different strategies to arrive at the price of the tickets. First, we will look into a strategy that has a hard cutoff point of how many passengers can be overbooked and will keep selling tickets until the date of the flight or no more tickets are available. Second, we will look into a strategy that allows the airline to stop ticket sales in order to optimize the number of passengers have can overbook a flight. Both these strategies will be solved using dynamic programming where we will have to know what price a first-class and a coach ticket should be on a given day in order to reach the maximum expected profit.

PROBLEM DESCRIPTION

For our airline company, we are looking to book a flight that will depart a year from now (365 days). This flight has 2 different classes: first-class and coach. The sales of the first-class tickets and the coach tickets are independent from each other. This means that the demand of each class will not influence the other class. The only exception to this is if first-class tickets are sold-out, then the probability that a customer will purchase a coach ticket will rise by 3% regardless of price. We are also told that we will not overbook first-class tickets and only overbook coach tickets. In the event that more coach customers show up for the flight than number of coach seats, we would experience two potential costs. First, if there is a vacant seat in first-class, due to unsold tickets or absent customers, then we can upgrade that coach customer to first class for a cost of $50 for each coach customer that can be upgraded to first-class. Second, if this is not possible, because first-class is full, either before or after upgrading coach customers, we would need to offer vouchers for a cost of $425 for each remaining overbooked customer.

We also know that coach customers are 95% likely to show up for their flight and first-class customers are 97% likely to show up. We also have 100 seats available in coach and 20 seats available in first-class. For the coach seats, we have 2 different prices: $300 or $350. For first class, we also have 2 different prices: $425 or $500. For the coach seats, there is a 65% chance a customer will buy a ticket if set at the low price and 30% chance if set at the high price. When first-class tickets sell out, these increase to 68% and 33% respectively. For the first-class seats, there is a 8% chance a customer will buy a ticket if set at the low price and 4% chance if set at the high price. We also assume that the discount rate is 17% per year.

We would want to find the best decision for setting the first-class ticket price and setting the coach ticket price at any given day away from departure of the flight. There would be a total of 4 decisions:

- Low price Coach, Low price First
- High price Coach, Low price First
- Low price Coach, High price First
- High price Coach, High price First

With this, there would also be 9 different situations for each day as well:

- Sold Low Coach, Sold Low First
- Sold High Coach, Sold Low First
- Sold Low Coach, Sold High First
- Sold High Coach, Sold High First
- Not Sold Coach, Sold Low First
- Not Sold Coach, Sold High First
- Sold Low Coach, Not Sold First
- Sold High Coach, Not Sold First
- Not Sold Coach, Not Sold First

We want to explore 2 different strategies as explained above. The first strategy would set the number of allowed overbooked tickets to be from 5 to 15 tickets. We would run each of these problems to see which overbooking policy would give us the most expected profits. The second strategy would be to allow dynamic programming algorithm to find the optimal number of overbooking given expected costs and stop selling tickets at any time in order to maximize profits. We would have a soft cap at 20 tickets overbooked. This would add an additional decision action of not selling tickets on a specific day only for coach customers. These decisions would be

- Low price Coach, Low price First
- High price Coach, Low price First
- Low price Coach, High price First
- High price Coach, High price First

- Not Sell Coach, Low price First
- Not Sell Coach, High price First.

The 9 daily situations, as listed above, would be the exact same. In this case, with the addition of not selling coach tickets, we can represent that as the Not Sold Coach states.

FORMULATION

<u>Strategy 1</u>

State Variables

The state variables would be number of coach seats available, time, and number of first-class seats available. In this case, we can represent this as

(sc, t, sf)

Choice Variables

The choice variables would be as explained above as decisions. In this case, we represent this as:

[(coach price low, first price low), (coach price high, first price low), (coach price low, first price low), (coach price high, first price high)]

Dynamics

In order to change from one state to another, we would need to find the number of tickets sold to today and the number of tickets sold already for both coach and first-class. Time will always increase from one state to another. The possible next states can be represented as:

(sc, t+1, sf) - unsold coach and first-class ticket today
(sc-1, t+1, sf) - sold coach ticket, but not first-class ticket today
(sc-1, t+1, sf-1) - sold both coach and first-class ticket
(sc, t+1, sf-1) - sold first-class ticket, but not coach

Value function

$$v(sc, t, sf) = max(E(\sum_{i=0}^{T-t} (profits@t + 1)\,\gamma i)$$

Bellman equation

Before we write the Bellman equation, we can do some manipulation of the variables in order to simplify our equation.

Assuming the following variables:

P is the number of dollars we get if a coach ticket is sold
x is the probability that a coach ticket will be sold
Q is the number of dollars we get if a first-class ticket is sold
y is the probability that a first-class ticket is sold

There are 4 observable scenarios:

- Both of them get sold
- Coach sold and first-class not sold
- Coach not sold and first-class sold
- Both of them don't get sold.

With these being independent event, we can then represent each day of revenue as:

$$\begin{gathered} xy(P+Q)+x(1-y)(P)+(1-x)(y)(Q)\ +\ (1-x)(1-y)(0) \\ = xyP+xyQ+xP-xyP+yQ-xyQ \\ = xP+yQ \end{gathered}$$

This can be included into the Bellman equation by also discounting future possible dates as well:

$$v(sc,t,sf)=xP+yQ\ +\gamma(\ldots)$$

With this in mind, we can construct our Bellman equation.

<u>If both coach and first-class tickets are available:</u>

$$\begin{aligned} LL = \ & prob\_coach\_sale\_low\ * price\_coach\_low\ + prob\_first\_sale\_low\ * price\_first\_low \\ & + \\ & \gamma * (prob\_coach\_no\_sale\_low\ * prob\_first\_no\_sale\_low\ *\ v(sc,t+1,sf) \\ & + prob\_coach\_sale\_low\ * prob\_first\_no\_sale\_low\ *\ v(sc-1,t+1,sf) \\ & + prob\_coach\_sale\_low\ * prob\_first\_sale\_low\ *\ v(sc-1,t+1,sf-1) \\ & + prob\_coach\_no\_sale\_low\ * prob\_first\_sale\_low\ *\ v(sc,t+1,sf-1)) \end{aligned}$$

$$LH = prob\_coach\_sale\_low * price\_coach\_low + prob\_first\_sale\_high * price\_first\_high + \gamma * (prob\_coach\_no\_sale\_low * prob\_first\_no\_sale\_high * v(sc, t+1, sf) + prob\_coach\_sale\_low * prob\_first\_no\_sale\_high * v(sc-1, t+1, sf) + prob\_coach\_sale\_low * prob\_first\_sale\_high * v(sc-1, t+1, sf-1) + prob\_coach\_no\_sale\_low * prob\_first\_sale\_high * v(sc, t+1, sf-1))$$

$$HL = prob\_coach\_sale\_high * price\_coach\_high + prob\_first\_sale\_low * price\_first\_low + \gamma * (prob\_coach\_no\_sale\_high * prob\_first\_no\_sale\_low * v(sc, t+1, sf) + prob\_coach\_sale\_high * prob\_first\_no\_sale\_low * v(sc-1, t+1, sf) + prob\_coach\_sale\_high * prob\_first\_sale\_low * v(sc-1, t+1, sf-1) + prob\_coach\_no\_sale\_high * prob\_first\_sale\_low * v(sc, t+1, sf-1))$$

$$HH = prob\_coach\_sale\_high * price\_coach\_high + prob\_first\_sale\_high * price\_first\_high + \gamma * (prob\_coach\_no\_sale\_high * prob\_first\_no\_sale\_high * v(sc, t+1, sf) + prob\_coach\_sale\_high * prob\_first\_no\_sale\_high * v(sc-1, t+1, sf) + prob\_coach\_sale\_high * prob\_first\_sale\_high * v(sc-1, t+1, sf-1) + prob\_coach\_no\_sale\_high * prob\_first\_sale\_high * v(sc, t+1, sf-1))$$

$$v(sc, t, sf) = max(LL, LH, HL, HH)$$

```
else:
    # value if you set both coach and first-class ticket prices low
    valueLL = pcL[1]*pricecL + pfL[1]*pricefL + delta* (pfL[0]*pcL[0]*V[sc,t+1,sf]
                                                        +pfL[0]*pcL[1]*V[sc-1,t+1,sf]
                                                        +pfL[1]*pcL[1]*V[sc-1,t+1,sf-1]
                                                        +pfL[1]*pcL[0]*V[sc,t+1,sf-1])

    # value if you set coach ticket price low and first-class ticket price high
    valueLH = pcL[1]*pricecL + pfH[1]*pricefH + delta* (pfH[0]*pcL[0]*V[sc,t+1,sf]
                                                        +pfH[0]*pcL[1]*V[sc-1,t+1,sf]
                                                        +pfH[1]*pcL[1]*V[sc-1,t+1,sf-1]
                                                        +pfH[1]*pcL[0]*V[sc,t+1,sf-1])

    # value if you set coach ticket price high and first-class ticket price low
    valueHL = pcH[1]*pricecH + pfL[1]*pricefL + delta* (pfL[0]*pcH[0]*V[sc,t+1,sf]
                                                        +pfL[0]*pcH[1]*V[sc-1,t+1,sf]
                                                        +pfL[1]*pcH[1]*V[sc-1,t+1,sf-1]
                                                        +pfL[1]*pcH[0]*V[sc,t+1,sf-1])

    # value if you set both coach and first-class ticket prices high
    valueHH = pcH[1]*pricecH + pfH[1]*pricefH + delta* (pfH[0]*pcH[0]*V[sc,t+1,sf]
                                                        +pfH[0]*pcH[1]*V[sc-1,t+1,sf]
                                                        +pfH[1]*pcH[1]*V[sc-1,t+1,sf-1]
                                                        +pfH[1]*pcH[0]*V[sc,t+1,sf-1])


    V[sc,t,sf]=max(valueLL,valueLH,valueHL,valueHH) # value funciton maximizes expected profit
```

<u>If first-class tickets are available but not coach:</u>

$$Lf = prob\_first\_sale\_low * price\_first\_low + \gamma * (prob\_first\_no\_sale\_low * v(sc, t+1, sf) + prob\_first\_no\_sale\_low * v(sc, t+1, sf-1))$$

$$Hf = prob\_first\_sale\_high * price\_first\_high + \gamma * (prob\_first\_no\_sale\_low * v(sc, t+1, sf) + prob\_first\_no\_sale\_low * v(sc, t+1, sf-1))$$

$$v(sc, t, sf) = max(Lf, Hf)$$

```
elif sc == 0 and sf!=0:# there are first-class tickets available but not coach
    # value if you set the first-class ticket price low
    valueLf = (pfL[1])*pricefL + delta* ((pfL[0])*V[sc,t+1,sf] + (pfL[1])*V[sc,t+1,sf-1])
    # value if you set the first-class ticket price high
    valueHf = (pfH[1])*pricefH + delta* ((pfH[0])*V[sc,t+1,sf] + (pfH[1])*V[sc,t+1,sf-1])


    V[sc,t,sf]=max(valueLf,valueHf) # value function maximizes expected revenue
```

<u>If coach tickets are available but not first class:</u>

$$Lc = (prob\_coach\_sale\_low + 0.03) * price\_coach\_low + \gamma * ((prob\_coach\_no\_sale\_low - 0.03) * v(sc, t+1, sf) + (prob\_coach\_sale\_low + 0.03) * v(sc-1, t+1, sf)$$

$$Hc = (prob\_coach\_sale\_high + 0.03) * price\_coach\_high + \gamma * ((prob\_coach\_no\_sale\_high - 0.03) * v(sc, t+1, sf) + (prob\_coach\_sale\_high + 0.03) * v(sc-1, t+1, sf)$$

$$v(sc, t, sf) = max(Lc, Hc)$$

```
elif sc !=0 and sf==0: # there are coach tickets available but not first-class. This increases the
    # probability of sale coach tickets by 0.03.
    # value if you set the coach ticket price low
    valueLc = (pcL[1]+0.03)*pricecL + delta* ((pcL[0]-0.03)*V[sc,t+1,sf] + (pcL[1]+0.03)*V[sc-1,t+1,sf])
    # value if you set the coach ticket price high
    valueHc = (pcH[1]+0.03)*pricecH + delta* ((pcH[0]-0.03)*V[sc,t+1,sf] + (pcH[1]+0.03)*V[sc-1,t+1,sf])

    V[sc,t,sf]=max(valueLc,valueHc) # value function maximizes expected profit
```

<u>If both coach and first-class are not available:</u>

$$v(sc, t, sf) = \gamma * (v(sc, t+1, sf)$$

```
if sc==0 and sf==0: # is the flight full (0 seats left)
    V[sc,t,sf]=delta*V[sc,t+1,sf] # if so, you can't make any more money
```

We can formulate all of this into just one Bellman equation function by having identity functions that represent cases given if tickets are able in either coach or first, however, in our formulation and engineering of the problem, we felt that this would be the most approachable way to solve the problem by separating into these different situations.

Terminal Conditions

Because we might incur a cost at the time of departure, we would need to set that as a terminal condition. In this problem, we are unsure how many customers would actually show up for the flight as it is a probability as explained in the problem description. To solve this, we would need to get the expected cost at the day of the flight using a binomial probability mass function.

We get each probability of every possible number of passengers that would show up for coach and first-class independently. With this expected probability we would add this to a cost variable for that specific number of coach and first class tickets sold in the value function.

For each number of coach tickets (sc) and first-class tickets (sf) booked and for each possible number of coach (C) and first-class passengers (F) that actually show up:

Assuming the following variables:

C is the number of coach customer that actually show up for the flight
F is the number of first-class customers that actually show up for the flight
x is the probability that C customers will arrive
y is the probability that F customers will arrive

As a reminder, we :

- Upgrade a coach passenger for $50 if there are available seats in first class,
- Give a voucher to a coach passenger for $425 if there are not no more seats available on the flight
- Have 100 seats in coach
- Have 20 seats in first class
- Set cost as a running negative sum value for the number of coach and first-class tickets sold

With this in mind, we can formulate the cost in three cases:

If the number of coach passengers who showed up is less than the number of coach seats available:

$$cost = cost - 0$$

If the number of coach passengers who showed up is more than the number of coach seats but number of overbooked passengers less than the number of empty first-class seats, we can upgrade all coach passengers:

$$cost = cost - x * y * 50(C - 100)$$

If the number of coach passengers who showed up is more than the number of coach seats and the number overbooked is more than the number of empty first-class seats, we can upgrade some coach passengers to fill first-class and then we can give vouchers to the remaining overbooked coach passengers:

$$cost = cost - x * y * (50(20 - Y) + 425(C - 100 - (20 - Y))$$

After all of these costs are calculated and summed up for a specific number of coach and first-class tickets sold, we can add this to our value matrix:

$$v(sc, t, sf) = cost$$

```
for sc in range(scN):
    tc = scN - sc-1 # number of coach tickets booked
    for sf in range(sfN):
        tf = sfN - sf-1 # number of first-class tickets booked
        cost = 0
        for tc_act in range(tc+1):# looping over each possible number of coach passengers actually showing up
            # on the day of departure
            pr_tc_act = scipy.stats.binom.pmf(tc_act,tc,0.95)# probability of 'tc_act' number of
            # people showing up out of 'tc'
            for tf_act in range(tf+1):# looping over each possible number of first-class passengers actually
            #showing up on the day of departure
                pr_tf_act = scipy.stats.binom.pmf(tf_act,tf,0.97)# probability of 'tf_act' number of
                # people showing up out of 'tf'
                if tc_act<=Mc:# if the number of coach passengers who showed up is less than the capacity of
                    # coach then there will be no cost.
                    cost = cost+0
                else:
                    if tc_act-Mc <= Mf-tf_act:# if the number of coach passengers who showed up is higher than the
                    #capacity of coach but lower than the seats remaining in first-class then the cost would be the
                    # cost associated with bumping all of them to first-class
                        cost = cost - pr_tc_act*pr_tf_act*costs['ctof']*(tc_act-Mc)
                    else:
                    # if the number of coach passengers who showed up is higher than the capacity of coach and
                    # higher than the seats remaining in first-class then the cost would be the
                    # cost associated with bumping some of them till first-class is filled and bumping the
                    # remaining passengers off the plane.
                        cost = cost - pr_tc_act*pr_tf_act*(costs['ctof']*(Mf-tf_act)
                                                    + costs['ctoout']*(tc_act-Mc-(Mf-tf_act)))
        V[sc,tN-1,sf] = cost
```

In our decision matrix, U, we also represent the last day when the flight departs as a day that we can not sell anymore tickets. This is similar to saying that both the first-class and coach tickets are sold out. More details regarding our U matrix is explained below.

Additional formulation notes:

For our decision matrix, U, we have 9 different cases that we set for each decision:

- Sell Coach Low, Sell First Class Low: 11
- Sell Coach Low, Sell First Class High: 12
- Sell Coach Low, First Class Sold Out: 13
- Sell Coach High, Sell First Class Low: 21
- Sell Coach High, Sell First Class High: 22
- Sell Coach High, First Class Sold Out: 23
- Coach Sold Out, Sell First Class Low: 31
- Coach Sold Out, Sell First Class High: 32
- Coach Sold Out, First Class Sold Out: 33

We set the value for U given the number of seats available in coach and first and the time til the flight as the index of the max value of different price levels for both coach and first-class tickets:

If both coach and first-class tickets are available:

$$v(sc, t, sf) \ = \ max(LL, LH, HL, HH)$$
$$U(sc, t, sf) \ = \ int((argmax(LL, LH, HL, HH))/2) + 1) * 10 + int((argmax(LL, LH, HL, HH))\%2) \,! = 0) + 1$$

```
V[sc,t,sf]=max(valueLL,valueLH,valueHL,valueHH) # value funciton maximizes expected profit

U[sc,t,sf]=(int(np.argmax([valueLL,valueLH,valueHL,valueHH])/2)+1)*10\
+int((np.argmax([valueLL,valueLH,valueHL,valueHH])%2)!=0)+1
```

Possible values: 11, 12, 21, 22

If coach tickets are available but not first class:

$$v(sc, t, sf) \ = \ max(Lc, Hc)$$
$$U(sc, t, sf) \ = \ (argmax(Lc, Hc) \ + \ 1) * 10 + 3$$

```
V[sc,t,sf]=max(valueLc,valueHc) # value function maximizes expected profit
U[sc,t,sf]=(np.argmax([valueLc,valueHc])+1)*10+3
```

Possible values: 13, 23

If first-class tickets are available but not coach:

$$v(sc, t, sf) \ = \ max(Lf, Hf)$$
$$U(sc, t, sf) \ = \ 30 + argmax(Lc, Hc) \ + \ 1$$

```
V[sc,t,sf]=max(valueLf,valueHf) # value function maximizes expected profit
U[sc,t,sf]=30+np.argmax([valueLf,valueHf])+1
```
Possible values: 31, 32

<u>If both coach and first-class are not available:</u>

$$v(sc,t,sf) \ = \gamma * (\,v(sc,t+1,sf)$$
$$U(sc,t,sf) \ = \ 33$$

```
V[sc,t,sf]= delta*V[sc,t+1,sf] # if so, you can't make any more money
U[sc,t,sf]=33 # No tickets can be sold.
```
Possible values: 33

All of these situations give the tens place information regarding the price we should set the coach tickets if available and the ones place inflation regrading the price we should set the first-class tickets if available

<u>Strategy 2</u>

In strategy 2, instead of a hard cap of tickets that we can overbook, we would allow the dynamic programming algrothgim to find where to stop selling tickets to reduce costs while maximizing profits. This would introduce another decision to stop selling coach tickets at any given state. This would be similar to assuming that coach tickets are sold out as seen in strategy 1. Therefore, there are only a few slight changes between strategy 1 and strategy 2.

State Variables

The state variables would be the same as Strategy 1

(sc, t, sf)

Choice Variables

There would be an addition of 2 decision variables from strategy 1:

[(coach price low, first price low), (coach price high, first price low), (coach price low, first price low), (coach price high, first price high) , (don’t sell coach, first low), (don’t sell coach, first high]

Dynamics

The dynamics would be similar as an unsold coach ticket could also be considered that the ticket was never put up on sale in the first place as a decision.

(sc, t+1, sf) - unsold coach and first-class ticket today
(sc-1, t+1, sf) - sold coach ticket, but not first-class ticket today
(sc-1, t+1, sf-1) - sold both coach and first-class ticket
(sc, t+1, sf-1) - sold first-class ticket, but not coach

Value function

$$v(sc, t, sf) \; = \; max(E(\sum_{i=0}^{T-t} \quad (profit@t + 1)\, \gamma i)$$

Bellman equation

The Bellman equation would only be slightly different in 2 cases: if both coach and first-class tickets are available and if coach tickets are available but not first-class.

<u>If both coach and first-class tickets are available:</u>

Along with LL, LH, HL, HH, as we have calculated before, we would introduce 0L and 0H

Recall the simplification of the Bellman equation inputs as:

$$v(sc, t, sf) = xP + yQ \; + \gamma(\dots)$$

Where P is the number of dollars we get if a coach ticker is sold
x is the probability that a coach ticket will be sold
Q is the number of dollars we get if a first-class ticket is sold
y is the probability that a first-class ticket is sold

In this case, we would not sell any coach tickets, so x would be be 0. Therefore:

$$v(sc, t, sf) = (0)P + yQ \; + \gamma(\dots) \; = yQ \; + \gamma(\dots)$$

We can use this in order to solve for 0L and 0H

$$0L = prob\_first\_sale\_low * price\_first\_low + \gamma * (prob\_coach\_no\_sale\_low * v(sc, t+1, sf) + prob\_coach\_no\_sale\_low * v(sc, t+1, sf-1))$$

$$0H = prob\_first\_sale\_high * price\_first\_high + \gamma * (prob\_coach\_no\_sale\_low * v(sc, t+1, sf) + prob\_coach\_no\_sale\_low * v(sc, t+1, sf-1))$$

```
# value if you decide not to sell coach and set first-class ticket price low
value0L = pfL[1]*pricefL + delta* (pfL[0]*V[sc,t+1,sf]+pfL[1]*V[sc,t+1,sf-1])

# value if you decide not to sell coach and set first-class ticket price high
value0H = pfH[1]*pricefH + delta* (pfH[0]*V[sc,t+1,sf]+pfH[1]*V[sc,t+1,sf-1])
```

Note that this is the same as the Lf and Hf as calculated in the case that there are first class tickets but not coach. This would give us the following equation:

$$v(sc, t, sf) = max(LL, LH, HL, HH, 0L, 0H)$$

<u>If coach tickets are available but not first class:</u>

Along with Lc, Hc, as we have calculated before, we would introduce 0c.

$$0c = 0 + \gamma * (v(sc, t+1, sf))$$

```
# value if you decide not to sell the coach ticket
value0c = 0 + delta*V[sc,t+1,sf]
```

This would be similar to the case that we do not have either first-class or coach tickets available. This would give us the following equation:

$$v(sc, t, sf) = max(Lc, Hc, 0c)$$

Terminal Conditions

The terminal conditions would be the same as strategy as the costs have not changed.

Additional formulation notes:

Because we have changed only 2 cases (if both coach and first-class tickets are available and if coach tickets are available but not first-class), we would only need to make changes to these cases for the U matrix. As stated before, we can represent the decision to not sell coach tickets as a state where the coach tickets are sold out (30's).

If both coach and first-class tickets are available:

$$v(sc,t,sf) \;=\; max(LL,LH,HL,HH,0L,0H)$$
$$U(sc,t,sf) \;=\; int((argmax(LL,LH,HL,HH,0L,0H))/2) + 1) * 10 + int((argmax(LL,LH,HL,HH,0L,0H))\%2)\,!=0) + 1$$

```
V[sc,t,sf]=max(valueLL,valueLH,valueHL,valueHH,value0L,value0H) # value funciton maximizes expected profit

U[sc,t,sf]=(int(np.argmax([valueLL,valueLH,valueHL,valueHH,value0L,value0H])/2)+1)*10\
+int((np.argmax([valueLL,valueLH,valueHL,valueHH,value0L,value0H])%2)!=0)+1
```

Possible values: 11, 12, 21, 22, 31, 32

If coach tickets are available but not first class:

$$v(sc,t,sf) \;=\; max(Lc,Hc,0c)$$
$$U(sc,t,sf) \;=\; (argmax(Lc,Hc,0c) \;+\; 1) * 10 + 3$$

```
V[sc,t,sf]=max(valueLc,valueHc,value0c) # value function maximizes expected profit
U[sc,t,sf]=(np.argmax([valueLc,valueHc,value0c])+1)*10+3
```

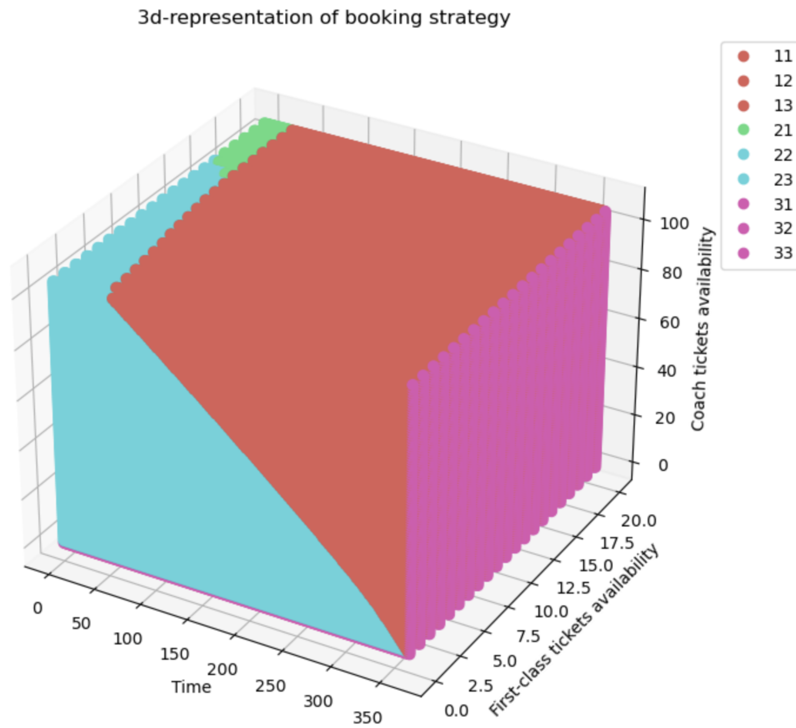
Possible values: 13, 23, 33

RESULTS

The results shown below are just for one run. These are not simulation runs. The simulation runs will be discussed in the next section.

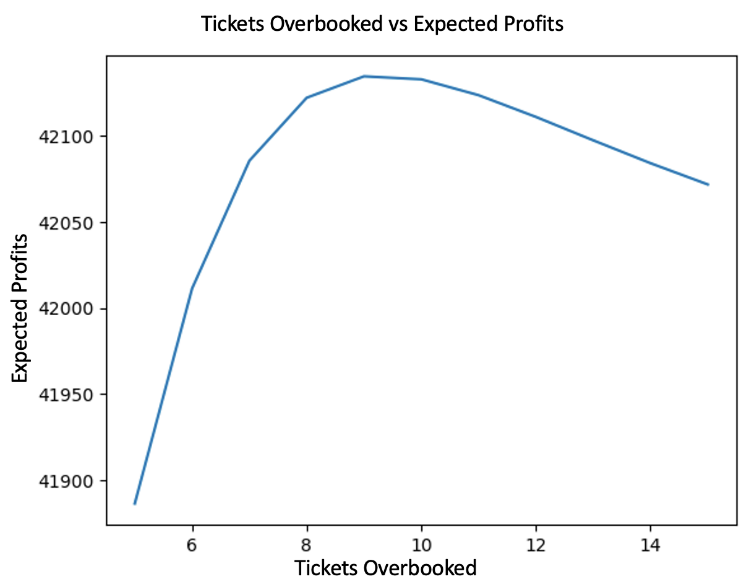
Strategy 1

The most optimal results for each given day til the flight and available coach and first-class tickets can be given with the following 3-D graph:



The decision legend can be understood by referring to the U matrix as detailed in the Formulation section above.

For strategy 1, we ran overbooking policies between 5 - 15 tickets. By viewing the profits from each one of these policies, we can find the most optimal overbooking policy. Below are the results of these runs:



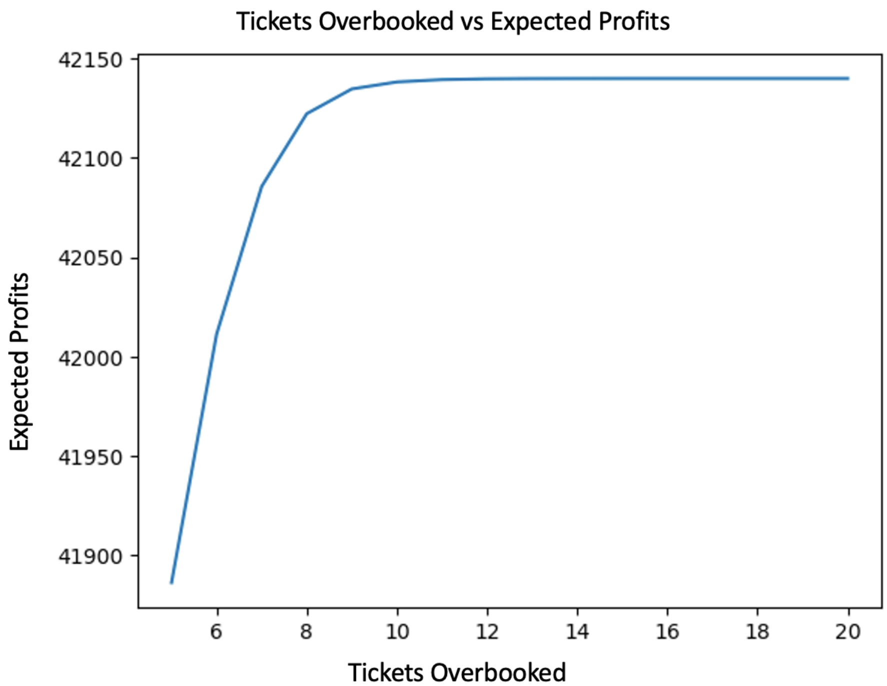
Based on this plot, we see that selling 9 overbooked tickets would give us the most expected profits. Below are the exact values:

Tickets Overbooked	Expected Profit Rounded to Nearest Cent
5	$41886.16
6	$42011.22
7	$42085.54
8	$42122.17
9	$42134.62
10	$42132.90
11	$42123.67
12	$42111.03
13	$42097.42
14	$42084.11
15	$42071.74

With 9 overbooked tickets, we get $42134.62 in expected profits.

Strategy 2

For strategy 2, we allow the algorithm to decide when to stop overbooking coach tickets. In this case, we would expect the algorithm to flat line expected profits after a certain number of ticket overbooked. The graph is seen below:



As expected, we see that after around 10 tickets, we would decide to stop overbooking tickets to maximize profits. To confirm the exact number, we would need to look more granular and look at the exact expected profit for each overbooked ticket number:

Tickets Overbooked	Expected Profit Rounded to Nearest Cent
5	41886.16
6	42011.22
7	42085.54
8	42122.17
9	42134.62
10	42138.14

11	42139.33
12	42139.71
13	42139.83
14	42139.87
15	42139.89
16	42139.89
17	42139.89
18	42139.89
19	42139.89
20	42139.89

We can see here that after 15 tickets, we are not getting any more profit at the nearest cent. With this strategy, we are getting $42139.89 in expected profit.

<u>Comparison of Strategies</u>

With strategy 1, we are getting an expected profit of $42134.62 and with strategy 2, we are getting an expected profit of $42139.89. Strategy 2 gives us a $5.27 increase in profits. This shows that, even though there might be only a slight improvement in profits, strategy 2 performs better than strategy 1. Because this is the result of just one random run, we would need to run simulations and get the mean values of profit for each strategy.

SIMULATION AND ANALYSIS

In order to assess the effectiveness of the two approaches, a total of 5000 simulations were conducted to evaluate their profitability. To recap, below is the optimal booking policy for the 2 strategies:

Strategy 1 : No option of not selling coach tickets despite being available; overbook by 9 seats

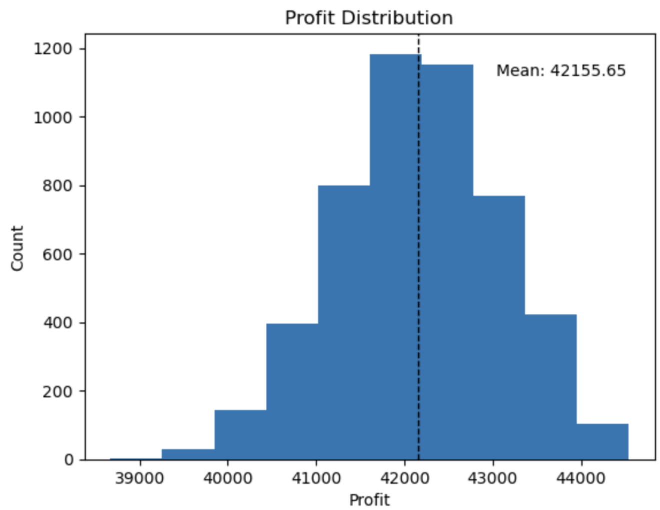
Strategy 2 : Option of not selling coach tickets on any given day; overbook by 20 seats

These simulations involved sampling from the probability distribution of passenger ticket purchases based on the prices of both classes. Additionally, the simulations involved sampling from a binomial distribution to determine the number of passengers who would show up on the day of the flight based on the number of tickets sold and the optimal pricing policy obtained through dynamic programming.

Which strategy is more profitable?

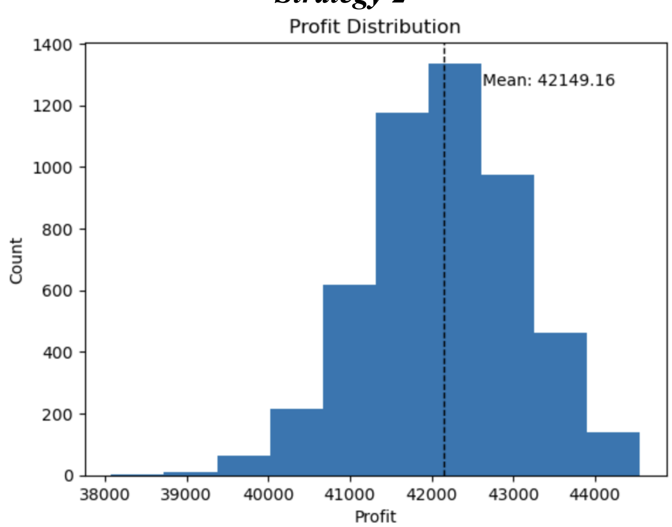
Firstly, we compare the profits obtained from both the strategies by looking at the distribution plots of the profits:

Strategy 1



On an average, we are able to achieve a profit of **$42,155.65**

Strategy 2



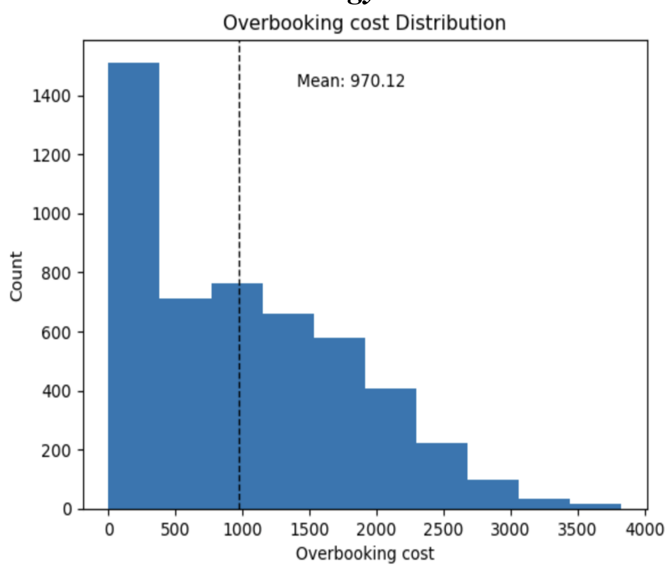
On an average, we are able to achieve a profit of **$42,149.16**

Despite observing slightly higher mean profits for the first' strategy than the 'Optimal overbooking' strategy, the upper and lower bounds of the profits are relatively comparable. The two distributions look fairly identical as well. This indicates that the difference between the two strategies is not statistically significant.

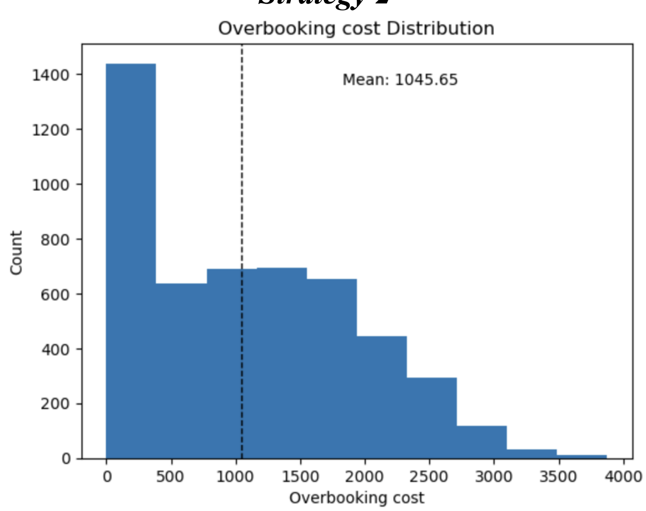
What is the Overbooking cost associated with each strategy?

Strategy 2 seems to have a higher overbooking cost. We look at the distribution plot of the overbooking cost during the 5000 simulations:

Strategy 1



Strategy 2

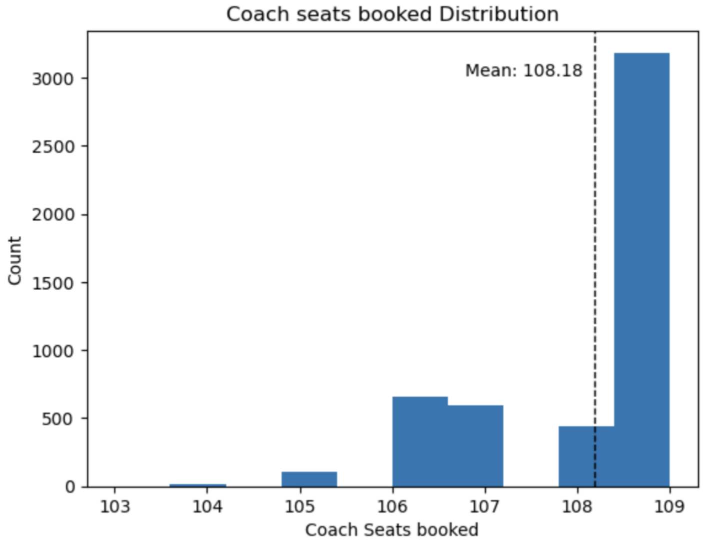


Clearly, the second strategy has much higher overbooking costs associated with it. The distribution plots also suggest that the higher values are more common in the second strategy.

Despite similar profits, we will be spending more on overbooking costs if we utilize the second strategy.

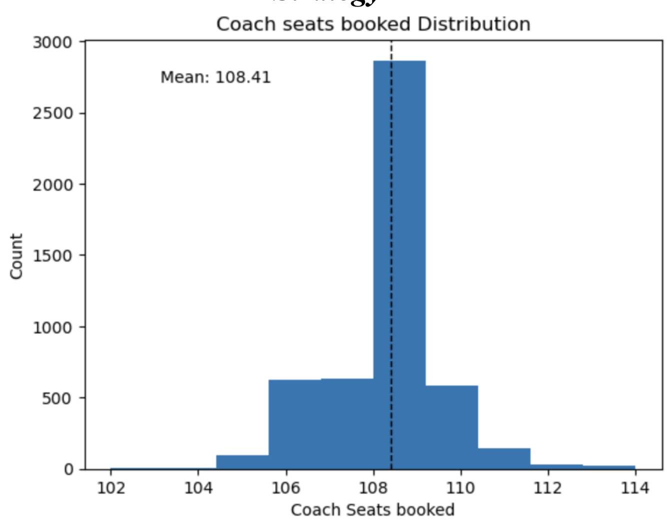
How many coach seats are booked?

Strategy 1



The coach seats are overbooked if the option to not sell them is unavailable.

Strategy 2

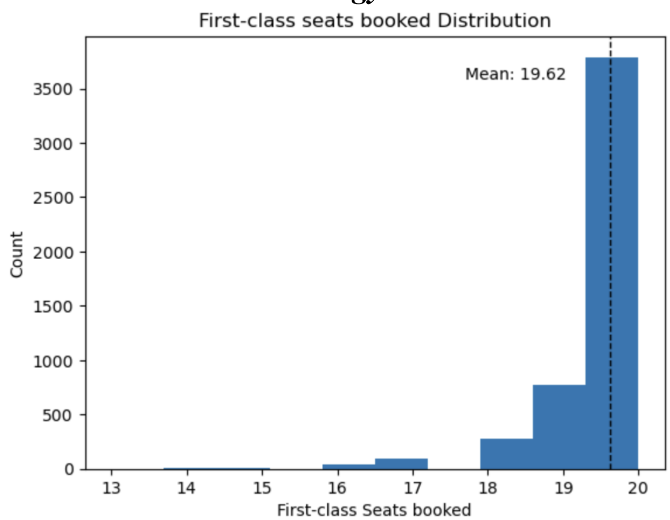


Coach seats are rarely overbooked after 108 as that is where we attain the maximum profits.

Clearly, we can observe that there is no significant difference in the mean coach seats booked between the two strategies. However, the distributions seem very different.

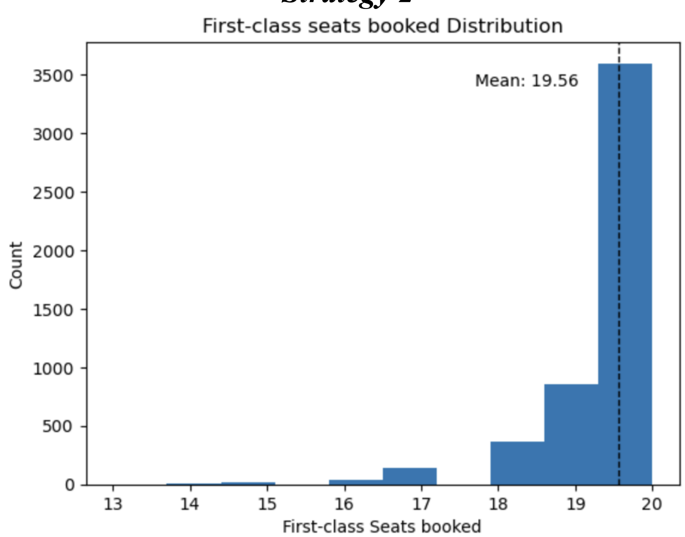
How many first-class seats are booked?

Strategy 1



Even if it looks like the probability of selling a first-class ticket is low, it mostly gets fully booked due to the higher number of days to book them than the number of first-class tickets.

Strategy 2

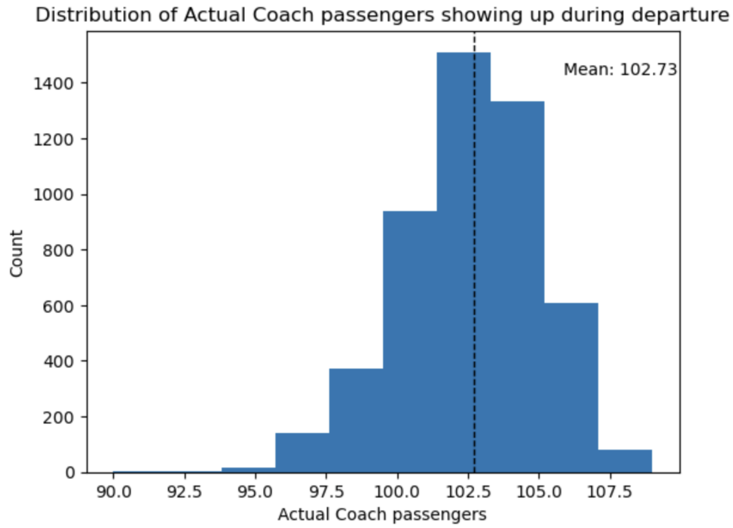


First-class mostly gets fully booked due to the higher number of days to book them than the number of first-class tickets.

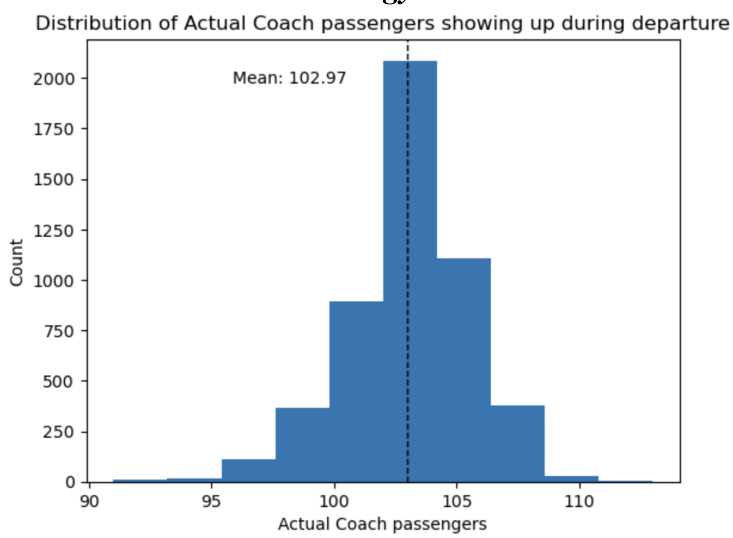
Clearly, here too there does not seem to be any significant difference between the two means using the two strategies.

How many coach passengers actually show up?

Strategy 1



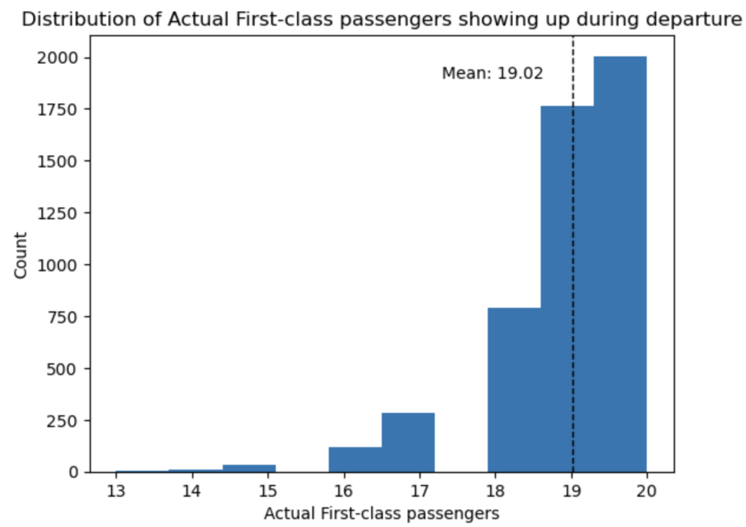
Strategy 2



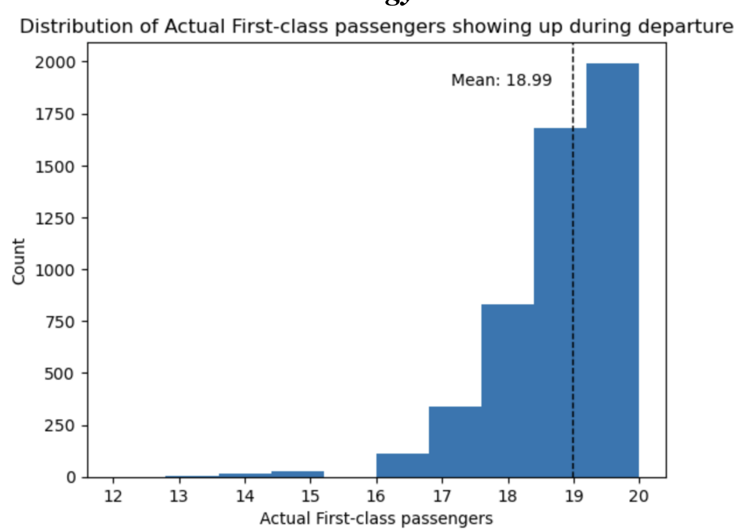
Slightly more passengers turn up on average in the second strategy although this seems to be a very minute difference. The first distribution seems to have more values centered around the mean while the second strategy distribution seems to have more outlier values.

How many first-class passengers actually show up?

Strategy 1



Strategy 2

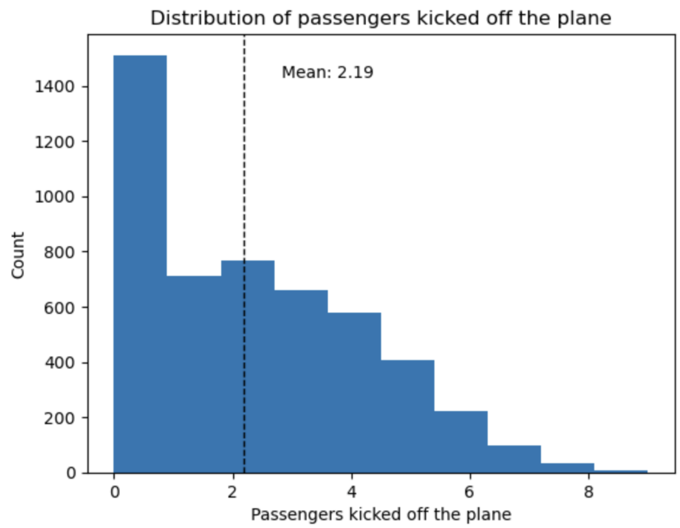


The number of first class passengers turning up on average seems to be practically the same in both the strategies.

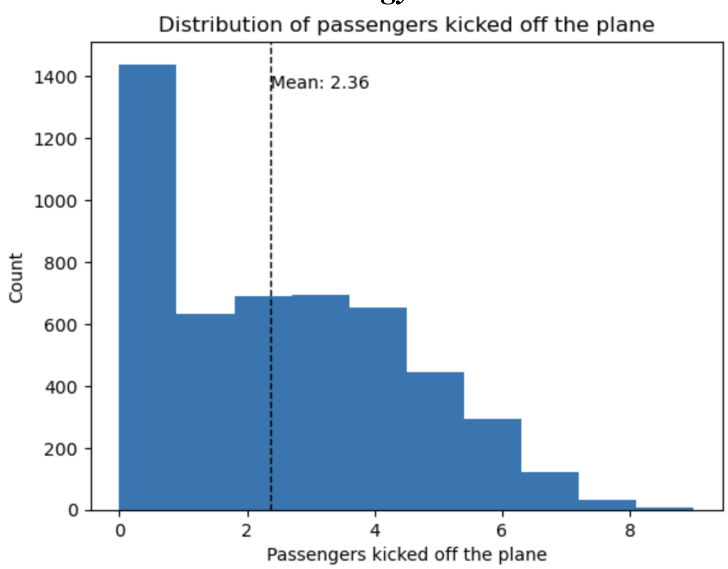
How many passengers are kicked off the plane?

We compare the number of passengers kicked off using the two strategies. We look at the distribution plot of the no. of passengers kicked off during the 5000 simulations:

Strategy 1



Strategy 2



The second strategy appears to have a notably higher number of passengers being removed compared to the first strategy. Therefore, it is crucial to pay special attention to the second strategy as a larger number of displaced passengers can result in potential PR crises and negative publicity.

CONCLUSIONS

The analysis performed by our team provides valuable insights into how airlines can optimize their overbooking policies to maximize expected discounted profit. By using dynamic programming techniques, we were able to examine different pricing policies and the number of tickets to offer for sale on a particular flight. The results indicate that following the first strategy with 9 overbooked tickets can significantly increase expected discounted profit.

It is important to note that while overbooking may be a profitable strategy for airlines, it can also be a contentious issue for customers. Therefore, airlines must strike a balance between maximizing profits and providing a positive customer experience. They can do this by being transparent about their overbooking policies, offering incentives for customers who voluntarily give up their seats, and providing excellent customer service.

Overall, the findings of this analysis can be used by airlines to optimize their overbooking policies and increase their profitability. However, airlines must also take into account the potential impact of these policies on their customers and strive to maintain a positive reputation and customer satisfaction.