International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

# Designing a Data-Driven Decision Model for Optimising Hotel Operations: An Integrated Study of Demand Forecasting, Workforce Allocation, Queueing Systems and Resource Utilisation

**Ruveer Sarwal[1], Raghu Raja Mehra[2]**

[1]Student Researcher, Amritsar, Punjab, India

[2]Faculty Mentor, Amritsar, Punjab, India

**Abstract:** ***Hotels take four important decisions every day: how many guests to expect, how many staff to put on duty, how many service counters to open, and how much of the building to keep running. In most hotels these four decisions are taken separately. This paper builds a single decision model that links them together. A forecasting model predicts tomorrow's room demand, the forecast decides how many staff each department needs, a queueing model checks whether the planned counters will keep waiting time below a target, and a resource index tracks how well rooms, staff, energy and water are being used. Because hotels rarely share day-by-day operating data, the model is tested on a clearly labelled simulated dataset of 365 days for a 180-room hotel; the data are artificial and are not taken from any real hotel. On a 60-day test period, a random forest gave the most accurate forecast (average error 6.34 rooms, 5.59%), just ahead of Holt-Winters smoothing. Replacing a fixed roster with the forecast-based plan raised staff utilisation from 81.7% to 91.8%, cut the average check-in wait from 11.34 minutes to 2.00 minutes, reduced daily staffing cost by 13.0% and improved the resource index from 72.6 to 79.5. Most of the improvement comes from moving staff to the busy hours rather than from hiring fewer people. The results are illustrative and would need testing on real hotel data before they can be generalised.***

**Keywords:** hotel operations; demand forecasting; staff scheduling; queueing theory; resource utilisation; decision model; simulation

## 1. Introduction

A hotel collects a large amount of information every single day. Every booking, every check-in, every meal served and every unit of electricity used is recorded somewhere. Yet the decisions that matter most for cost and for guest experience are usually taken by habit. The duty manager looks at the arrivals list, remembers what last weekend was like, and keeps the same roster that has been used for months.

This creates a simple problem. On quiet days the hotel pays for staff it does not need. On busy days the same number of staff cannot handle the rush, and guests see the shortage directly as a long queue at reception. A hotel cannot store its service the way a factory stores stock, so this mismatch cannot be corrected later. It shows up as idle hours on one side and unhappy guests on the other.

Each part of this problem has already been studied on its own. There is a large literature on forecasting hotel demand, another on staff rostering, another on queues, and another on energy and water use in hotels. What is much rarer is a study that joins them. A forecast is only useful if it changes the roster. A roster only helps guests if it changes how many counters are open. And the same demand figure should also decide how many floors are opened and cooled. The value lies in the link between the four decisions, not in any one of them.

This paper builds that link. It develops a four-part decision model, tests it end to end on a simulated hotel, and reports honestly where the model helps and where it does not.

## 2. Background and Problem Statement

Consider a hotel with 180 rooms, a restaurant, a housekeeping department and a front office that runs three shifts. Arrivals are not spread evenly through the day. Most check-ins happen in a window of about three hours in the late afternoon, and most departures happen in a shorter window in the morning. Demand also changes across the week and across the year, with weekend and festival peaks that managers can describe but few hotels actually measure.

Against this changing demand, the staffing plan usually stays the same. A roster is made for the month, sized for a fairly busy day, and then repeated. Two costs follow. The first is extra staff on ordinary days. The second, and more damaging, is too few staff during the peak. Waiting time does not rise smoothly as a queue gets busier; it rises slowly at first and then very steeply once the counters are almost fully occupied. A roster that works comfortably at 70% counter utilisation can produce a crowded lobby at 90%.

The research problem can therefore be stated in four parts. Using past daily data, how can a hotel (i) predict short-term demand reasonably well, (ii) turn that prediction into department-level staffing, (iii) check that the staffing meets a stated waiting-time target at each service point, and (iv) release physical resources in line with the same forecast?

## 3. Literature Review

a) ***Forecasting hotel demand***

Forecasting is the foundation of revenue management, and the methods used have changed over time. Claveria, Monte and Torra [1] showed the value of non-linear methods for hospitality demand, and Caicedo-Torres and Payares [2] applied machine learning to occupancy prediction. Pereira and Cerqueira [3] compared a large set of methods for short forecast horizons and found that machine-learning models are competitive but not always better than well-chosen statistical models. Ampountolas [4] reached a similar conclusion when comparing SARIMAX and neural models for daily hotel demand. António, de Almeida and Nunes [5] published one of the few openly available hotel demand datasets, which has supported a good deal of later work. Hyndman and Koehler [6] remain the standard reference for the accuracy measures used in this paper.

***b) Staff scheduling***

Staff scheduling is a classical operations-research problem. The reviews by Ernst et al. [7] and Van den Bergh et al. [8] describe the usual approach: minimise labour cost subject to coverage, working-hour and skill limits. Hospitality applications are fewer than in nursing or airline scheduling, and in most of them the demand figure is simply assumed rather than forecast and tested.

***c) Queues in service operations***

Queueing models connect staffing to waiting. The multi-server model and Little's law [9] give the relationships used here. In hospitality, queueing has been used for capacity planning [10] and to study self-service check-in kiosks [11], which shorten queues under certain conditions. Maister [12] adds the behavioural side: guests judge a wait by how it feels, not only by how long it is.

***d) Resources and analytics***

Energy and water dominate hotel resource use, and Bohdanowicz and Martinac [13] provide benchmark intensities per occupied room that are still widely cited. On the management side, Mariani et al. [14] review business intelligence in hospitality, and Kimes [15] first set out the idea of adjusting capacity decisions to demand.

**Table 1:** Studies and the decision parts they cover.

| Study | Main focus | DF | WA | QS | RU |
|---|---|---|---|---|---|
| Kimes [15] | Yield management | ✓ | – | – | ✓ |
| Ernst et al. [7] | Staff rostering | – | ✓ | – | – |
| Bohdanowicz [13] | Resource use | – | – | – | ✓ |
| Kokkinou [11] | Self check-in | – | – | ✓ | – |
| Claveria [1] | Forecasting | ✓ | – | – | – |
| António [5] | Demand data | ✓ | – | – | – |
| Pereira [3] | ML forecasting | ✓ | – | – | – |
| Chetthamrongchai [10] | Queue capacity | – | – | ✓ | ✓ |
| This study | Integrated model | ✓ | ✓ | ✓ | ✓ |

DF = demand forecasting, WA = workforce allocation, QS = queueing systems, RU = resource utilisation.

## 4. Research Gap, Objectives and Questions

Three gaps follow from the review. First, most studies look at one part only, so a better forecast is never converted into a better roster. Second, each study measures success in its own units- forecast error, roster cost, or waiting time- so cost and guest experience are never compared on the same scale. Third, resource use is usually treated as an environmental reporting task rather than as an outcome of the same daily demand decision.

The objectives are: (O1) to design an integrated decision framework; (O2) to compare demand forecasting models; (O3) to build a staffing model driven by the chosen forecast; (O4) to analyse guest queues; (O5) to build a combined resource utilisation index; and (O6) to compare the whole model against normal fixed-roster practice.

The study asks: (RQ1) Which forecasting method predicts daily demand best? (RQ2) How much does forecast-based staffing change cost and staff utilisation? (RQ3) What happens to waiting time and queue length? (RQ4) Does joining the four parts help more than using them separately, and where does it fail?

## 5. The Proposed Decision Model

The framework has five steps, shown in Fig. 1. Data are collected and cleaned; demand is forecast; the forecast is converted into workload; the workload decides staffing and the number of open counters; and the plan is checked against a waiting-time target before it is published. Realised results are then compared with the plan, and the difference is fed back into the forecasting model.

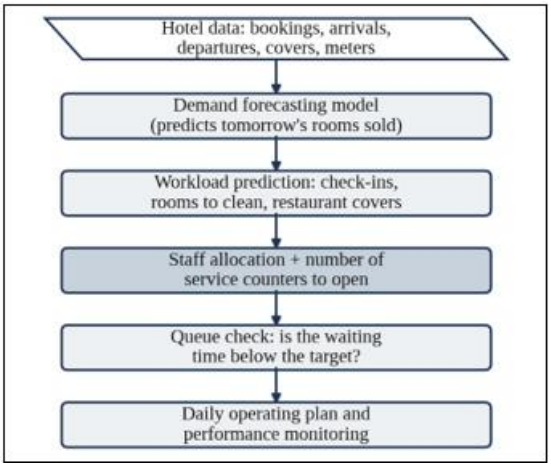


**Figure 1:**The integrated data-driven hotel operations decision model. Each step uses only the output of the step above it, so any single part can be improved without rebuilding the rest

Two design rules are followed. The first is that each step passes on only a small, clear output- a demand figure, a staffing figure, a queue setting. The second is that the model is never allowed to minimise cost alone. It must also meet a stated waiting-time target, otherwise the cheapest answer would always be the smallest possible team.

## 6. Demand Forecasting

Let D(t) be the number of rooms sold on day t. The task is to predict D(t+1) using only information available at the end of day t. Four families of models are compared, and the process is shown in Fig. 2.

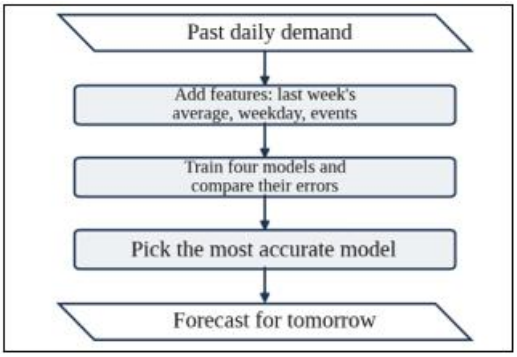


**Figure 2:** Forecasting workflow. Models are compared again at every update, so the method is not fixed forever.

The simplest method is the moving average, which predicts tomorrow as the average of the last k days:
*Forecast = (D(t) + D(t–1) + ... + D(t–k+1)) / k,  with k = 7*

Holt-Winters exponential smoothing improves on this by tracking three things separately: the current level, the trend, and a repeating weekly pattern. Recent days are given more weight than older days. A seasonal ARIMA model instead describes demand using its own past values and past errors, with an extra term repeating every seven days.

The fourth family treats forecasting as ordinary prediction from a table of features. A random forest builds many decision trees on different samples of the data and averages their answers. Eleven features are used: demand 1, 7 and 14 days ago; the average of the last 7 and 28 days; day of week, month and a weekend flag; a festival flag; and yesterday's room rate and check-in count.

Accuracy is measured in three ways. MAE is the average size of the error in rooms. RMSE is similar but gives extra weight to big misses. MAPE expresses the error as a percentage:

*MAPE = average of |Actual – Forecast| ÷ Actual × 100*

## 7. Workforce Allocation

The staffing step turns the demand forecast into a roster for five departments: front office, housekeeping, restaurant, maintenance and support. Fig. 3 shows the sequence.

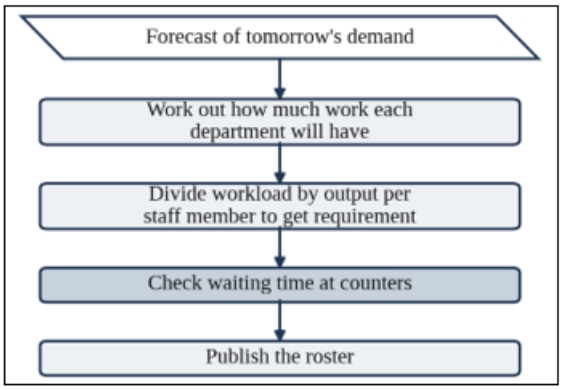


**Figure 3:** Workforce allocation flowchart. The waiting-time check happens before the roster is published, so staffing and queue quality are decided together.

For departments whose work is proportional to volume, the requirement is the predicted workload divided by an output standard:
*Staff needed = Predicted workload ÷ Output per staff member*

One housekeeping attendant is assumed to service 14 room-units per shift and one restaurant staff member to handle 30 covers per day. Maintenance and support have a fixed core team plus a part that grows with occupancy. The front desk is different: its size comes from the queueing model of Section VIII, because what matters there is waiting time, not output per person.

The daily plan is chosen to minimise total cost:
*Cost = Wage cost + Overtime cost + Delay penalty*

subject to four conditions: every department must have enough staff for its predicted workload; no department may fall below a minimum crew; overtime is capped; and the waiting time at every service point must stay below the target. Wages are taken as INR 145 per regular hour and INR 240 per overtime hour, and the delay penalty is INR 165 for each minute of average waiting above the five-minute target. The penalty is not a cash payment; it is a way of putting a price on a poor guest experience.

## 8. Queueing Analysis

Four service points are studied: reception check-in, the check-out desk, restaurant seating and the concierge desk. Each is modelled as a single line served by several counters, as shown in Fig. 4.

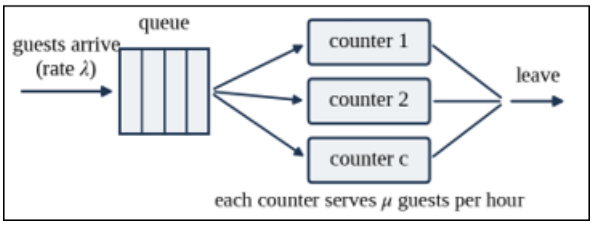


**Figure 4:** The check-in system: guests join one line and go to whichever of the c counters becomes free first

Two quantities describe each service point. The arrival rate λ is the number of guests arriving per hour during the peak

window. The service rate μ is the number of guests one counter can serve per hour. With c counters open, utilisation is

$$\rho = \lambda \div (c \times \mu)$$

Utilisation must stay below 1, otherwise the queue keeps growing. Standard multi-server queueing formulae then give the average number of guests waiting, L(q), and the average waiting time is obtained from Little's law [9]:

$$\text{Waiting time } W(q) = L(q) \div \lambda$$

Because a real arrival peak lasts only a few hours, waiting times are capped at 35 minutes in the simulation. In practice guests give up, are handled by a manager, or are served out of turn. This cap makes the results more conservative, not less.

The arrival rate is taken from the forecast. If a share of the day's check-ins falls into a window of a few hours, then the arrival rate is that share of predicted check-ins divided by the length of the window. Check-in is assumed to take about eight minutes, so one counter serves roughly 7.5 guests per hour.

## 9. Resource Utilisation

Utilisation of any resource is written as a percentage of what is available:

$$\text{Utilisation (\%)} = \text{Used capacity} \div \text{Available capacity} \times 100$$

For rooms this is simply occupancy. For staff it is required hours divided by rostered hours. Energy and water are handled slightly differently: they are compared against a benchmark consumption per occupied room [13], so that a hotel using less energy per occupied room scores higher. The six parts are then combined into one Resource Utilisation Index (RUI):

$$RUI = 0.25\ Rooms + 0.25\ Staff + 0.15\ Energy + 0.10\ Water + 0.15\ Housekeeping + 0.10\ Restaurant$$

The weights are a management judgement rather than a measured quantity. Changing them by half in either direction moves the index by less than 2.5 points and never changes which plan scores higher.

## 10. Methodology and Dataset

**The dataset used in this study is completely artificial.** It was created by the authors to test the framework. It does not come from any real hotel, and no conclusion about any actual property should be drawn from it. Because the data were generated by a known process, the behaviour of the model can be checked against a known answer, which real data would not allow. The simulated hotel has 180 rooms and is observed for 365 days. Demand is built from a yearly seasonal pattern with two peaks, a day-of-week effect, six festival blocks, a small upward trend and random noise. All other variables are generated from occupancy using simple relationships: room turnover gives check-ins, check-ins and departures give front-desk transactions, and occupancy and covers give housekeeping load and utility use. Fig. 5 shows the demand series. Tables 2 and 3 list the variables and their summary statistics.

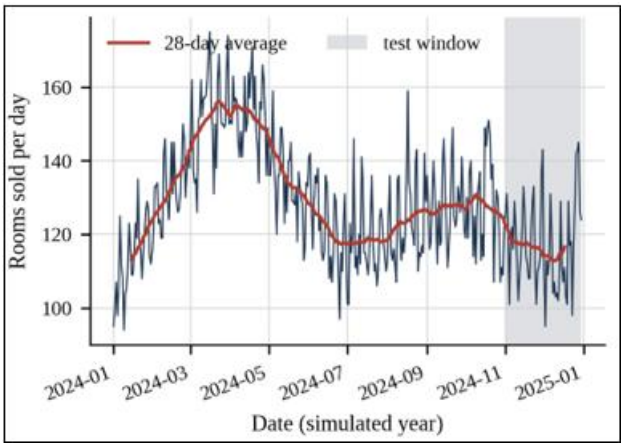


**Figure 5:** Simulated daily demand for 365 days, with the 28-day average and the 60-day test window. The data are artificial

**Table 2:** Main variables in the simulated dataset.

| Variable | Meaning | Unit |
|---|---|---|
| Occupancy rate | Rooms sold ÷ rooms available | 0–1 |
| Rooms sold | Rooms occupied that night | rooms/day |
| Average daily rate | Average price per room | INR |
| Room revenue | Rooms sold × rate | INR/day |
| Check-ins | Guests arriving | per day |
| Check-outs | Guests leaving | per day |
| Front-desk load | Check-ins + check-outs | per day |
| Restaurant covers | Meals served | per day |
| Housekeeping load | Room-units to clean | per day |
| Staff available | Total staff rostered | per day |
| Arrival rate λ | Check-ins per peak hour | per hour |
| Service time | Time per check-in | minutes |
| Waiting time | Average queue wait | minutes |
| Queue length | Guests waiting | guests |
| Staff utilisation | Work hours ÷ rostered hours | fraction |
| Energy use | Electricity consumed | kWh/day |
| Water use | Water consumed | kL/day |
| Guest score | Constructed experience proxy | 1–10 |

**Table 3:** Summary statistics of the simulated data (365 days)

| Variable | Mean | SD | Min | Max |
|---|---|---|---|---|
| Occupancy rate | 0.71 | 0.09 | 0.52 | 0.97 |
| Rooms sold | 128.3 | 16.8 | 94 | 175 |
| Room rate (INR) | 4509 | 462 | 3677 | 5846 |
| Check-ins | 68.1 | 12.1 | 38 | 113 |
| Front-desk load | 134.1 | 21.1 | 90 | 219 |
| Restaurant covers | 180.4 | 40.0 | 106 | 331 |
| Housekeeping load | 158.5 | 20.9 | 115 | 228 |
| Waiting time (min) | 15.6 | 10.7 | 1.0 | 35.0 |
| Staff utilisation | 0.86 | 0.08 | 0.71 | 1.15 |
| Energy (kWh) | 4389 | 298 | 3507 | 5293 |
| Water (kL) | 60.3 | 7.0 | 46.2 | 82.8 |
| Guest score | 8.22 | 1.01 | 5.72 | 9.90 |

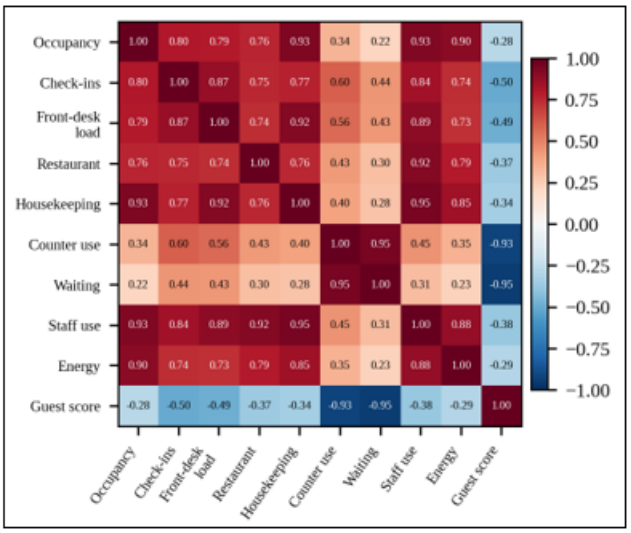


**Figure 6:** How the main variables move together. Waiting time is only weakly related to how many guests arrive, but strongly related to how busy the counters are.

Fig. 6 shows how the variables move together. Occupancy, housekeeping load, front-desk load and energy form one tightly linked group, because all four are driven by the same volume of guests. Waiting time behaves differently. Its correlation with the number of arrivals is only 0.44, but its correlation with counter utilisation is 0.95. This is the most important pattern in the data: crowding is not caused by many guests, it is caused by many guests relative to the number of open counters.

The 365 days are split in order: the first 305 days are used to build the models and the last 60 days are kept aside for testing. Forecasts are made one day ahead. After each test day the true value is added and the statistical models are updated, which copies the information a manager would really have. Two policies are then compared on the same 60 days. The conventional policy uses a fixed roster of 41 staff with two counters open at each service point, and one extra agent at reception and in the restaurant on Fridays and Saturdays. The proposed policy sizes each department from the forecast and opens the smallest number of counters that keeps waiting below five minutes. Both policies face the same real demand, so the proposed policy also suffers from its own forecast errors.

## 11. Results

### 1) *Forecast accuracy*

Table 4 and Figs. 7 and 8 show the results. The random forest is the most accurate (MAE 6.34 rooms, MAPE 5.59%), but only just: Holt-Winters smoothing is behind by 0.15 percentage points, which is small enough that a different test period could reverse the order. Seasonal ARIMA is further behind and the moving average is clearly the weakest, as expected from a method that ignores weekly and festival patterns. The boosted trees model does worse than the random forest, because with only 305 training days it fits the festival blocks too closely.

**Table 4:** Forecast accuracy on the 60-day test window.

| Model | MAE | RMSE | MAPE | Accuracy |
|---|---|---|---|---|
| Moving average (7 d) | 9.51 | 11.68 | 8.02% | 91.98% |
| Holt-Winters | 6.69 | 8.31 | 5.74% | 94.26% |
| Seasonal ARIMA | 7.58 | 8.92 | 6.58% | 93.42% |
| **Random forest** | **6.34** | **8.26** | **5.59%** | **94.41%** |
| Boosted trees | 8.06 | 10.81 | 7.22% | 92.78% |

MAE and RMSE are in rooms per day.

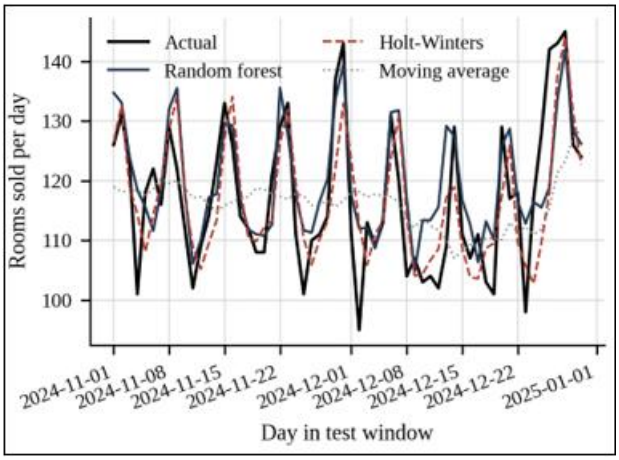


**Figure 7:** Actual and forecast demand over the test window. All methods follow the weekly cycle; they differ mainly at the festival peaks.

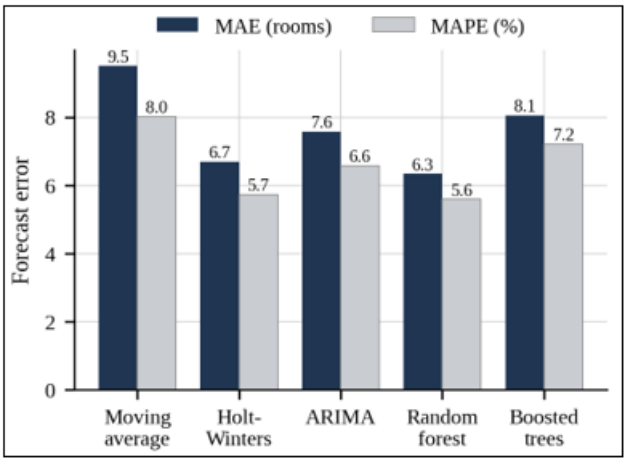


**Figure 8:** Comparison of forecast errors. Lower bars are better

Fig. 9 explains why the ranking is so close. The random forest takes about 32% of its information from the average of the last seven days and another 37% from demand one and fourteen days earlier. In other words, recent level and weekly rhythm explain most of hotel demand — and that is exactly what Holt-Winters already assumes. The real advantage of the random forest is that it can also use extra information such as festival flags and room rates, which a simple smoother cannot.

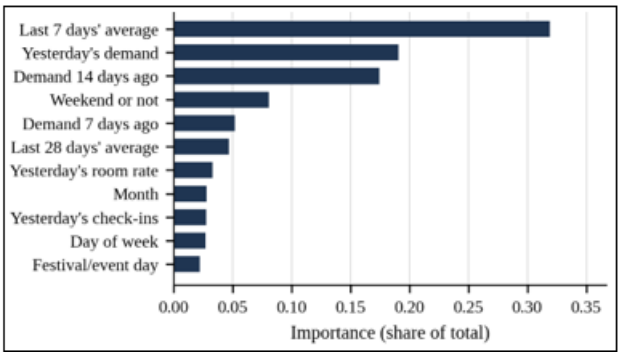


**Figure 9:** Which inputs the random forest relies on most

### *2) Staffing*

Table 5 and Fig. 10 compare the fixed roster with the forecast-based plan. Average staffing falls from 41.0 to 36.5 per day, a drop of 11.0%, while the actual requirement averages 33.5. The fall is not spread evenly. Housekeeping and the restaurant shrink, because the fixed roster was sized for a busier period than the test window turned out to be. The front office does not shrink at all: it stays at 6.0 staff, but more of them are placed inside the arrival peak and fewer outside it. The right way to read this result is redeployment rather than job cuts.

**Table 5:** Staffing under the two policies.

| Department | Needed | Fixed | Proposed | Change |
|---|---|---|---|---|
| Front office | 6.0 | 6 | 6.0 | 0% |
| Housekeeping | 10.6 | 14 | 11.7 | –16% |
| Restaurant | 6.0 | 8 | 6.7 | –16% |
| Maintenance | 4.9 | 6 | 5.0 | –17% |
| Support | 6.0 | 7 | 7.0 | 0% |
| **Total** | **33.5** | **41.0** | **36.5** | **–11%** |
| Staff utilisation | – | 81.7% | 91.8% | +10.1 pp |

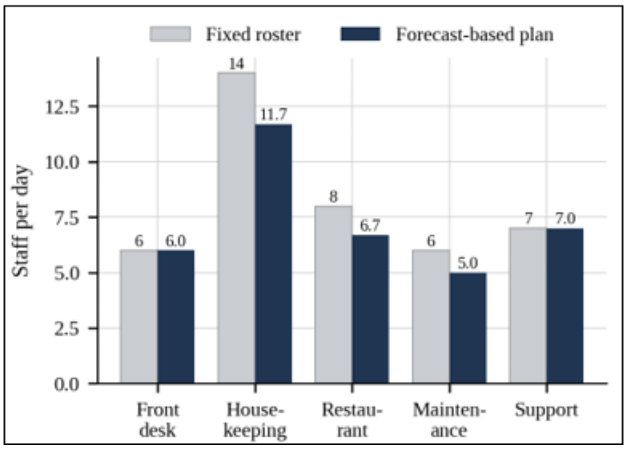


**Figure 10:** Staff deployed by department. The fixed roster over-provides in housekeeping and the restaurant.

### *3) Queues*

Table 6 and Figs. 11 and 12 report the four service points. At reception about 12.5 guests arrive per hour during the peak, while one counter serves 7.5 per hour. The fixed roster opens 2.3 counters on average, so utilisation is 0.74 and the average wait is 11.34 minutes. The proposed plan opens 3.0 counters, utilisation falls to 0.55 and the wait falls to 2.00 minutes. The restaurant behaves in the same way. The check-out desk and the concierge barely change, and this is the most useful part of the table: both were already comfortable, so the model correctly leaves them alone. A model that improved everything everywhere would suggest the comparison was unfair.

**Table 6:** Queue results by service point (test-window averages)

| Service point | λ | μ | Counters | ρ | Wait (min) | Queue |
|---|---|---|---|---|---|---|
| Check-in | 12.5 | 7.5 | 2.3→3.0 | 0.74→0.55 | 11.34→2.00 | 2.53→0.46 |
| Check-out | 13.2 | 12.0 | 2.0→2.1 | 0.55→0.53 | 2.81→2.08 | 0.73→0.49 |
| Restaurant | 25.7 | 6.0 | 5.3→6.0 | 0.81→0.71 | 8.98→2.59 | 5.02→1.20 |
| Concierge | 5.2 | 6.0 | 2.0→2.0 | 0.43→0.43 | 2.41→2.33 | 0.22→0.21 |

Each cell shows fixed roster→proposed plan. λ and μ are guests per hour; queue is the average number of guests waiting.

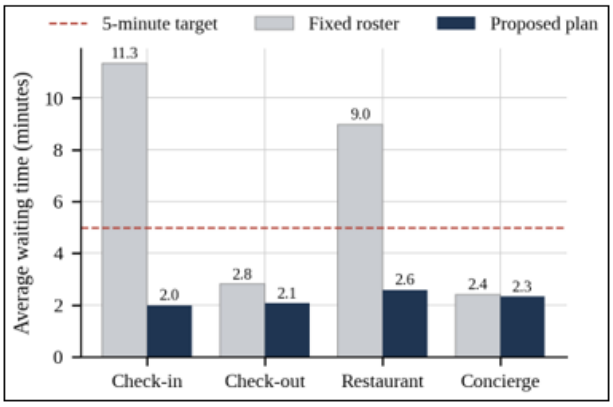


**Figure 11:** Average waiting time before and after optimization

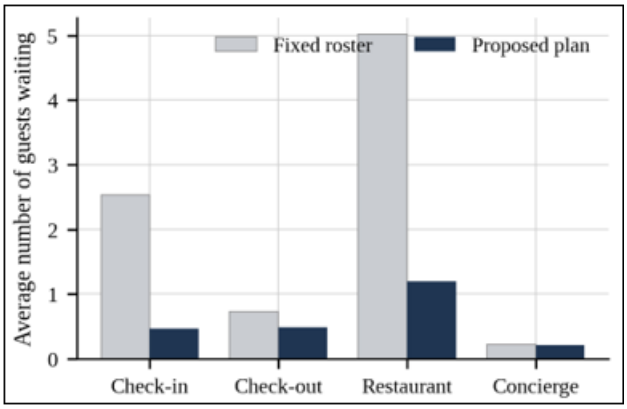


**Figure 12:** Average queue length at each service point

Over the 60 test days the five-minute target was missed on 42 days under the fixed roster and on only 3 days under the proposed plan. Those three failures all happened on days when demand was under-forecast, so one counter too few was opened. This is exactly how a forecasting error becomes

visible to a guest, and it is the strongest reason for treating forecasting and staffing as one problem instead of two.

### *4) Resource use*

Table 7 and Fig. 13 show the six parts of the resource index. Room utilisation is the same under both policies, because the model does not sell rooms- a useful check, since a framework that appeared to raise occupancy would only be measuring its own assumptions. The gains come from staff, housekeeping, energy (36.2 down to 33.6 kWh per occupied room) and water. The overall index rises from 72.6 to 79.5.

**Table 7:** Resource utilisation index.

| Component | Weight | Fixed | Proposed |
|---|---|---|---|
| Room utilisation | 0.25 | 65.1% | 65.1% |
| Staff utilisation | 0.25 | 81.7% | 91.8% |
| Energy efficiency | 0.15 | 72.0% | 77.6% |
| Water efficiency | 0.10 | 83.2% | 88.1% |
| Housekeeping use | 0.15 | 73.8% | 88.5% |
| Restaurant capacity | 0.10 | 57.2% | 65.2% |
| Overall index (RUI) | – | 72.6 | 79.5 |

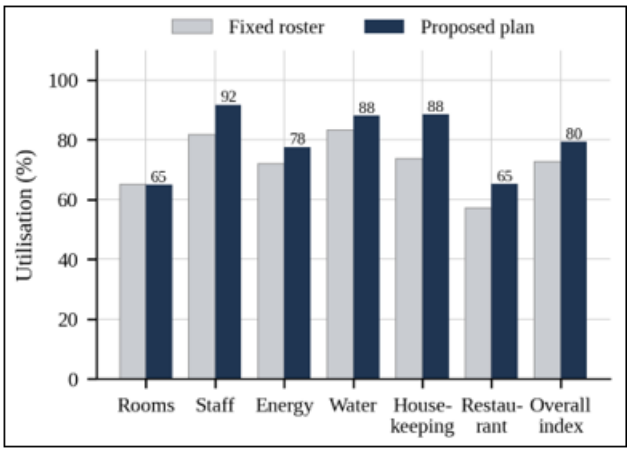


**Figure 13:** Resource use before and after optimization

### *5) How many counters are enough?*

Fig. 14 and Table 8 show what happens when the number of counters and the level of demand are varied around a busy reference level of 16.2 arrivals per hour. With two counters the wait is very long. Adding a third counter brings it to about 5 minutes, and a fourth brings it below one minute. The table also shows why fixed rosters fail: with three counters, demand 30% higher than expected pushes the wait from 5 minutes to over 20, whereas with four counters the same surprise raises it only to about 3 minutes. Extra capacity is best understood as insurance against forecast error rather than as a service upgrade.

**Table 8:** Waiting time (minutes) for different counters and demand

| Demand level | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 30% below normal | 10.7 | 1.3 | 0.3 | 0.1 |
| 15% below normal | 26.0 | 2.5 | 0.5 | 0.1 |
| Normal (16.2/hour) | 33.2 | 5.0 | 0.9 | 0.2 |
| 15% above normal | 35.0 | 10.8 | 1.7 | 0.4 |
| 30% above normal | 35.0 | 26.0 | 2.9 | 0.7 |

Columns give the number of counters open at reception.

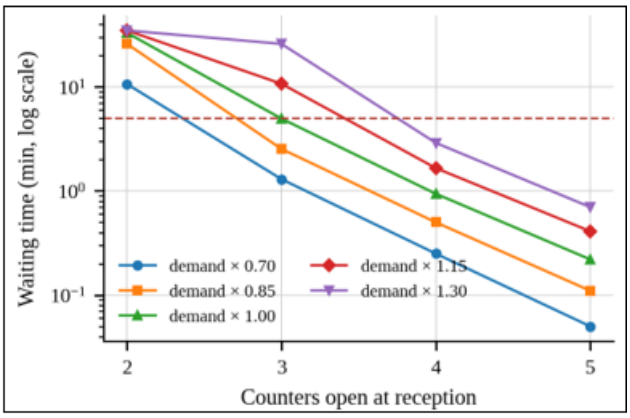


**Figure 14:** Waiting time falls sharply with each extra counter. The dashed line is the five-minute target.

### *6) Three demand scenarios*

Three days were taken from the test window- a quiet day (57% occupancy), a normal day (63%) and a peak day (79%)- and the model was applied to each. Table 9 and Fig. 15 give the recommended plans. Total staffing rises from 33 to 47, check-in counters from three to four and restaurant stations from five to nine, while the expected wait stays between 0.8 and 1.7 minutes.

The peak day carries the most important result in the paper. The recommended cost of INR 54,520 is higher than the fixed roster's INR 51,711 for the same day. The model does not save money on busy days; it spends more, buys the capacity that protects service, and recovers the difference on the many ordinary days when the fixed roster is too large. Judged only on peak days, the framework would look like a failure.

**Table 9:** Recommended plans for three demand scenarios

| Item | A: Low | B: Normal | C: Peak |
|---|---|---|---|
| Occupancy | 56.6% | 63.3% | 79.1% |
| Rooms sold | 102 | 114 | 142 |
| Check-ins | 48 | 61 | 91 |
| Restaurant covers | 141 | 152 | 284 |
| Front office staff | 6 | 6 | 7 |
| Housekeeping | 10 | 10 | 15 |
| Restaurant staff | 5 | 6 | 11 |
| **Total staff** | **33** | **34** | **47** |
| Check-in counters | 3 | 3 | 4 |
| Expected wait (min) | 0.8 | 1.7 | 1.5 |
| Fixed-roster wait | 5.6 | 15.6 | 19.2 |
| Proposed cost | 38,280 | 39,440 | 54,520 |
| Fixed-roster cost | 47,650 | 49,314 | 51,711 |

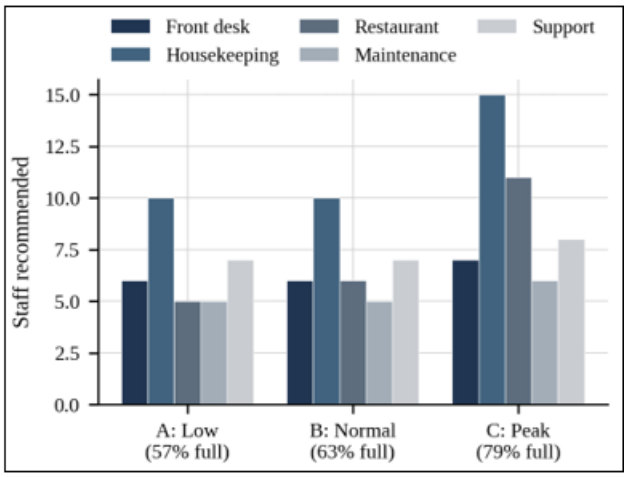


**Figure 15:** Recommended staffing in the three scenarios

## 12. Overall Comparison

Table 10 and Fig. 16 bring the results together. Forecast accuracy improves by 2.4 percentage points against the simple benchmark, staff utilisation by 10.1 points and the resource index by 6.9 points. Daily staffing cost falls by 13.0%, from INR 48,798 to INR 42,458. Average check-in waiting time falls by about 82%.

**Table 10:** Conventional practice against the proposed model

| Indicator | Conventional | Proposed | Change |
|---|---|---|---|
| Forecast accuracy | 91.98% | 94.41% | +2.4 pp |
| Staff per day | 41.0 | 36.5 | –11.0% |
| Staff utilisation | 81.7% | 91.8% | +10.1 pp |
| Wage cost | 47,560 | 42,282 | –11.1% |
| Overtime | 70 | 157 | +124% |
| Delay penalty | 1,168 | 19 | –98% |
| **Total cost** | **48,798** | **42,458** | **–13.0%** |
| Check-in wait (min) | 11.34 | 2.00 | –82% |
| Queue length (guests) | 2.53 | 0.46 | –82% |
| Days off target | 42 | 3 | –93% |
| Resource index | 72.6 | 79.5 | +6.9 |
| Guest score (1–10) | 8.55 | 9.41 | +0.86 |

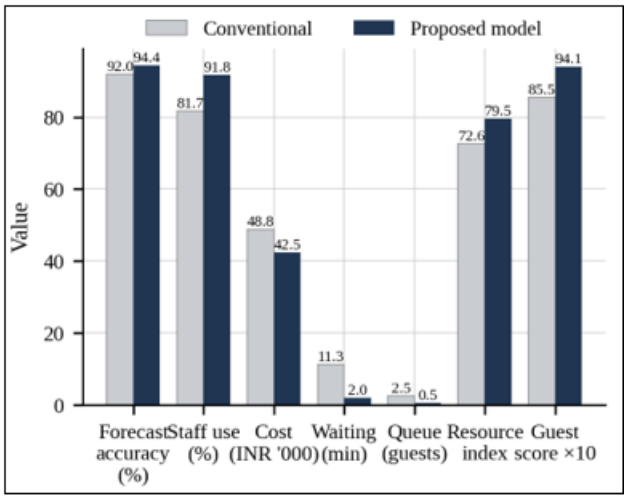


**Figure 16:** Overall comparison. Lower is better for cost, waiting time and queue length; higher is better for the rest

Two warnings belong beside these numbers. First, the waiting-time improvement is large because the fixed roster runs close to saturation at reception, where waiting time rises very steeply. A hotel where the manager already opens a third counter when the lobby fills would see a much smaller gain. Second, the guest score is a constructed variable in the simulation and is partly defined by waiting time, so it is reported for completeness and not as evidence.

## 13. Discussion

The forecasting comparison should not be read as proof that machine learning is better. The gap between the random forest and Holt-Winters is 0.15 percentage points. What the importance chart shows is that daily hotel demand is mostly recent level plus weekly rhythm, which a smoother already captures. A hotel with a stable weekly pattern should use exponential smoothing; a hotel whose demand is driven by weddings, conferences and festivals should invest in the extra features, because that is where the machine-learning advantage actually lies.

The chain from a forecast to a guest's experience is short. An error of about six rooms becomes an error of roughly one guest per hour in the arrival rate. That sounds trivial, but near the staffing boundary it is enough to change the number of counters, and the difference between three and four counters is the difference between a five-minute wait and a one-minute wait. The value of forecast accuracy is therefore uneven: it is worth almost nothing in the middle of a staffing band and a great deal at its edge.

A second point concerns where the gains actually sit. The model does not improve every part of the hotel. The concierge desk and the check-out counter were already working comfortably below saturation, so the optimisation left them almost unchanged. Room utilisation did not move at all, because selling rooms is a marketing decision, not an operations one. Almost the entire benefit comes from two places: the reception counter during the afternoon peak, and the size of the housekeeping and restaurant teams on ordinary days. This is worth saying plainly, because integrated models are often presented as if every part of the system improves together.

The results show both lower cost and better service, which normally suggests an unfair comparison. The scenario analysis is the answer to that objection. On the peak day the proposed plan costs more. What the model really does is move spending from days when staff are not needed to days when they are. Whether that is worthwhile depends entirely on how much a hotel believes a minute of guest waiting is worth. At the assumed penalty the trade is clearly good; at a much lower valuation, the fixed roster becomes defensible again.

## 14. Managerial Implications

Four practical points follow. First, roster the peak, not the day: front-office headcount hardly changed, yet waiting time fell sharply because the same people were placed differently. Second, write down a service-level target; without it, the optimisation collapses into pure cost cutting. Third, treat the extra counter as insurance- it is bought not for the expected day but for the day that turns out busier than expected. Fourth, aim for a utilisation band rather than a maximum: at

91.8% the plan is efficient but has little slack, and the small rise in overtime is the price of that leanness.

None of this needs a large technology project. Bookings, arrivals, departures, covers, cleaning lists and meter readings already exist in any property management system. What is usually missing is the daily habit of turning them into tomorrow's plan and then checking the result.

## 15. Limitations

The limitations are real and are listed in Table 11. The most important is that the data are simulated, so every number reported here is a property of the simulation. The direction of the effects follows from queueing and scheduling theory and is more reliable than the sizes. The test window is only 60 days. The queueing model assumes random arrivals and a single line, whereas real check-in times are more regular than assumed and guests sometimes leave the queue. Staff are treated as interchangeable within a department, ignoring skills, contracts and fatigue. The guest score is constructed, not measured. Finally, the fixed-roster benchmark is a simplified version of real practice.

**Table 11:** Limitations and what they mean

| Limitation | Effect on the results |
|---|---|
| Simulated data | Sizes are illustrative; direction is more reliable |
| 60-day test window | No seasonal change is observed |
| Queue assumptions | Real service times and guest behaviour are simplified |
| Staff treated alike | Skills and contracts ignored; savings are an upper limit |
| Constructed guest score | Not evidence about real satisfaction |
| Simplified benchmark | A reacting manager would narrow the gap |
| Single hotel | Size and market are not varied |

Future work should test the same pipeline on operating data from a real hotel. Other useful extensions include real-time data from door counters and kiosks, which would allow re-planning within the day; linking the model to room pricing, so that demand is shaped as well as served; and extending it to a group of hotels that can share staff.

## 16. Conclusion

This paper asked whether linking demand forecasting, staffing, queue management and resource use into one decision loop helps more than using them separately. On a simulated year of operations in a 180-room hotel, the integrated model raised staff utilisation from 81.7% to 91.8%, cut the average check-in wait from 11.34 to 2.00 minutes, reduced days missing the service target from 42 to 3, lowered daily staffing cost by 13.0% and raised the resource index from 72.6 to 79.5. A random forest gave the most accurate forecast, narrowly ahead of Holt-Winters smoothing.

The more useful conclusions are not the numbers. Crowding at a hotel counter depends on the ratio of arrivals to open counters, not on arrivals alone, which is why moving existing staff within the day achieved more than adding staff to the day. A forecast is worth something only where it changes a decision. And the model does not help everywhere: it spends more on peak days, changes nothing where service was already comfortable, and depends on how the hotel values a guest's time. Testing on real hotel data is the necessary next step.

## References

[1] Claveria, O., Monte, E., & Torra, S. (2015). A new forecasting approach for the hospitality industry. International Journal of Contemporary Hospitality Management, 27(7), 1520–1538. https://doi.org/10.1108/IJCHM-06-2014-0286

[2] Caicedo-Torres, W., & Payares, F. (2016). A machine learning model for occupancy rates and demand forecasting in the hospitality industry. Lecture Notes in Computer Science, 10022, 201–211. https://doi.org/10.1007/978-3-319-47955-2_17

[3] Pereira, L. N., & Cerqueira, V. (2022). Forecasting hotel demand for revenue management using machine learning regression methods. Current Issues in Tourism, 25(17), 2733–2750. https://doi.org/10.1080/13683500.2021.1999397

[4] Ampountolas, A. (2021). Modeling and forecasting daily hotel demand: A comparison based on SARIMAX, neural networks, and GARCH models. Forecasting, 3(3), 580–595. https://doi.org/10.3390/forecast3030037

[5] António, N., de Almeida, A., & Nunes, L. (2019). Hotel booking demand datasets. Data in Brief, 22, 41–49. https://doi.org/10.1016/j.dib.2018.11.126

[6] Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. International Journal of Forecasting, 22(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

[7] Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. European Journal of Operational Research, 153(1), 3–27. https://doi.org/10.1016/S0377-2217(03)00095-X

[8] Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., & De Boeck, L. (2013). Personnel scheduling: A literature review. European Journal of Operational Research, 226(3), 367–385. https://doi.org/10.1016/j.ejor.2012.11.029

[9] Little, J. D. C. (1961). A proof for the queuing formula: L = λW. Operations Research, 9(3), 383–387. https://doi.org/10.1287/opre.9.3.383

[10] Chetthamrongchai, P., Opulencia, M. J. C., Mironov, S., Aleynikova, M. Y., & Kostyrin, E. V. (2022). Hotel capacity planning using queuing systems and meta-heuristic algorithms. Mathematical Modelling of Engineering Problems, 9(5). https://doi.org/10.18280/mmep.090505

[11] Kokkinou, A., & Cranage, D. A. (2013). Using self-service technology to reduce customer waiting times. International Journal of Hospitality Management, 33, 435–445.

[12] Maister, D. H. (1985). The psychology of waiting lines. In J. A. Czepiel, M. R. Solomon, & C. F. Surprenant (Eds.), The service encounter (pp. 113–123). Lexington Books.

[13] Bohdanowicz, P., & Martinac, I. (2007). Determinants and benchmarking of resource consumption in hotels-Case study of Hilton International and Scandic in Europe. Energy and Buildings, 39(1), 82–95. https://doi.org/10.1016/j.enbuild.2006.05.005
[14] Mariani, M., Baggio, R., Fuchs, M., & Höpken, W. (2018). Business intelligence and big data in hospitality and tourism: A systematic literature review. International Journal of Contemporary Hospitality Management, 30(12), 3514–3554. https://doi.org/10.1108/IJCHM-07-2017-0461
[15] Kimes, S. E. (1989). Yield management: A tool for capacity-constrained service firms. Journal of Operations Management, 8(4), 348–363.
[16] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324
[17] Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: Principles and practice (3rd ed.). OTexts.