Case Study

Value Based Pricing in Hospitality Systems

THE PROBLEM

Revenue Management Systems (RMS) are a key component of hospitality systems. Their primary purpose is to provide room rate recommendations. These systems apply statistical analysis, forecasting models and, in some cases, machine learning models to a combination of historical reservation data (representing historical demand), current inventory (representing current demand), a demand calendar (representing anticipated demand hot spots) and other sources of data to produce room rate recommendations. These systems typically operate in batch performing this computation a few times per day. The output of this computation - the room rates - is fed to CRS/PMS either directly via electronic means or indirectly via humans that mediate the rates before handing the rates off to the CRS/PMS for use in the pricing and booking flow.

An RMS's job is to maximize room yield. The intention is to set a room rate that maximizes demand and room profitability. However, it does so by purely focusing on room and market metrics - historical demand, current demand, and a demand calendar that can potentially call out anticipated demand that may not be discoverable from the historical data. What is missing is the customer. When providing a rate to a customer, the pricing algorithm also needs to factor in customer history, propensities, spend patterns, preferences and even sentiment to provide a rate that is personalized to the customer. This drives loyalty and increases demand while maximizing revenue and profitability for the hotel not only for the room but across all products offered by the hotel.

This is best illustrated via an example.

Let's consider a hotel that offers gaming (casino), hosts fight nights and sells rooms. The hotel has a customer that loves fight nights, always spends less than $300/night on a room and spends, on average, $200/night on the casino floor on each trip. A fight night is a high demand night and so the RMS will price rooms at a higher rate. Let's say it prices it at $500. This rate is higher than the customer's budget resulting in the customer not booking the room. What the hotel has effectively lost is not only room revenue but also fight night ticket revenue, potential dining revenue and gaming revenue. However, if the pricing algorithm factored in the customer value, it can, for example, rate the room at $300 ($500 - average customer spend). The rate of $300 is within the customer's budget resulting in the customer booking the trip and spending an additional $200 on the gaming floor, buying a fight night ticket and spending money on dining and other activities. In other words, by factoring customer value in the pricing algorithm, the system can increase demand and ensure enough spend to maximize overall hotel profitability.

THE CORE ISSUE

The natural question that arises from above is that if the issue is simply that the customer value is missing in the pricing algorithms, then why not just enhance the RMS to add in customer value and produce a price that is individualized per customer. The issue is one of volume. RMSs operate in batch. In other words, they produce data a few times a day that is handed in batch to the eCommerce/CRS/PMS systems for use in the pricing/booking flow. Without customer value factored in, the amount of data produced is relatively small. For example, for a hotel chain with 10 properties, 500 room types per property and a booking window of 365 days, the RMS would output a total

of 1.825 million rates for each pricing run. This is a small number and can be updated in the CRS/PMS database by the RMS relatively easily. However, when you factor in customer, then a rate needs to be produced for each customer, hotel, room type and date combination. With an active customer base of 10M customers, this number grows to 18.25 trillion rates. This is a non-trivial number that significantly increases the cost of compute, the cost of storage and the time it takes to compute, to a point that is likely not feasible for agile hotel operations.

THE SOLUTION

To solve this problem, one needs the ability to execute the customer value-based pricing logic in *real time* instead of applying it in batch. Doing this in real time means that the overlay logic needs to be applied at the time the customer is active on a booking channel (website, partner site, contact center, etc.) and is requesting for a price. The data footprint for such an implementation is significantly smaller than compared to the batch approach. The data needed for this computation is a combination of (1) the rack rates produced by the RMS and (2) the customer data that is relevant to the overlay logic. Continuing with the example above, one just needs to store the 1.825M rates produced by the RMS and the 10M customer records to implement this approach. The booking flow identifies the customer, the property, the room, and the date(s) for which the rates need to be produced. It then pulls up the appropriate customer record and the appropriate RMS generated rate record and runs the pricing logic to produce the final, individualized rate for the customer.

The smaller data footprint in this approach makes it feasible to generate individualized prices. However, the data and compute architecture required to implement this solution needs to be quite sophisticated. To drive engagement, the industry demands millisecond level pricing response time. Additionally, to drive conversion, the front ends that serve the customer have become increasingly sophisticated. Controls such as smooth scrolling calendar type rate displays, and rate comparison matrices result in heavyweight pricing calls to the backend that require rates to be produced for several room types for each date in the entire booking window without compromising on the millisecond level response times. Finally, “look-to-book” ratios in the industry are small which translates to high pricing throughputs even if the booking rate is relatively small. To meet these demands, implementing value-based pricing in real-time requires a rather sophisticated data and compute architecture in the transactional plane.

INTRODUCING RUMI™

Rumi™, from N5 Technologies, is a platform that is designed to enable large amounts of data to be processed in real-time directly in transactional systems, such as CRS, PMS, and eCommerce systems. Rumi™ introduces a unique architectural approach and computing model in which data and its associated business logic are co-located in a single processing node. This approach enables data intensive compute logic, such as the value-based pricing logic described above, to be implemented in real-time instead of batch. More general information about Rumi can be found in the Rumi analysis by Bloor Research.

THE RUMI™ SOLUTION

A traditional architecture, in which the data and compute planes are separated by a network, is not capable of solving this value-based pricing issue in real-time. This is because the time it takes to fetch the data needed for a pricing calculation plus the compute time to derive the rate will far exceed the pricing response time requirements. This is what makes Rumi™ ideal for this solution since, by co-locating the data and business logic, Rumi™ eliminates the data fetch times completely thus bringing in the overall pricing calculation times to well within the response time SLAs.

It is due to this unique ability of Rumi™ that a **world leading, Fortune 500 gaming and hospitality company that is based out of Las Vegas USA**, selected N5 as the foundation for its new loyalty-based web site of which value-based pricing was a core, foundational capability. The site used Rumi™ to leverage customer reservation history, customer gaming spend, customer propensities and preferences and custom created offers to dynamically generate prices, select offers and produce content that is individualized for each customer in real time. The site was a massive success resulting in a **25x increase in loyalty driven revenue within just the first 9 months of deployment**. This resulted in the company deciding to use Rumi™ as the foundation for a brand new, in-house built CRS that would cater to all the booking channels.

The architectural ability of Rumi™ to be able to analyze and use raw data to produce value added compute directly in the transactional plane is a unique and extremely powerful capability. Over time, in addition to customer specific data, channel specific data was also added to the set of data available to the transactional business logic. The availability of all this data for use in the transactional plane unlocked several additional use cases and benefits beyond value-based pricing that was only possible due to this unique ability of Rumi™. The following are some of the key capabilities that were unlocked by Rumi™:

- Customer value-based pricing
- Dynamic selection of offers for pricing and redemption
- Transactional pricing for OTA
- Dynamic pricing for product bundles (upwards of 10 products)
- Consistent pricing for all channels – OTA, Loyalty, Property, Contact Center, Front Desk

The system is currently in production dynamically calculating and presenting upwards of 20,000 rates per second.