FORECASTING HOTEL DEMAND USING MACHINE LEARNING APPROACHES

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

by

Yueqian Zhang

August 2019

ABSTRACT

A critical aspect of revenue management is a firm's ability to predict future demand. Historically hotels have used pick-up based models owing to the complexities of trying to build casual models of demands. Machine learning approaches are slowly attracting attention owing to their outstanding predicting power and flexibility in modeling relationships.

This study provides an overview of approaches to forecasting hospitality demand using machine learning models, including Neural Network, Nearest Neighbors, Tree, and Support Vector Machine. The out-of-sample performances of the above approaches are illustrated by using two sets of data: one from a single hotel with long booking windows up to 12 months, the other from 24 hotels with 14 days advanced bookings and additional information including pricing, location, etc.

This research appears to be the first study in academia applying machine learning approaches in hotel demand forecast. The empirical findings prove that machine learning approaches outperform traditional models, especially given long booking history. The proposed models are valuable for practitioners in improving forecast accuracy and optimizing revenue, and lay the groundwork for future research into refining machine learning models in hotel revenue management.

BIOGRAPHICAL SKETCH

Yueqian (Rachel) Zhang was born and grew up in Lanzhou, China. She attended college at Dongbei University of Finance and Economics in Dalian, China with double majors in Hotel Management and English. In her senior year, Rachel studied aboard at Université Blaise Pascal in Vichy, France. After graduating from college, Rachel worked in consultancy in Beijing for a year and she very much enjoyed it.

Rachel started her study at Cornell University in Fall 2017. While at Cornell, Rachel explored her interests in quantitative analysis and data science, and developed her thesis in applying machine learning in hotel revenue management.

To my parents, Yueqing Tan and Aiguo Zhang

To my alma mater, Cornell

ACKNOWLEDGEMENTS

Writing the acknowledgements is always the nicest part. A lot of people have contributed efforts to this research, and I relish this opportunity to thank them.

I would like to firstly thank my committee chair, advisor, mentor, Chris Anderson, for his infinite patience and support through this journey. Every single word in this thesis concentrates his efforts, from the conceptualization, the search for data, to the over and over polishing and refining. His expertise in the interdisciplinary area of hospitality and data science has steered me through many challenging circumstances. I am lucky to learn from and work with him.

My committee members, Yao Cui and Yang Ning, have also brought unique perspectives and insights to this research. I also appreciate Dr. Deniz Akdemir's help in talking me through the statistical models and share his insights. The very initial idea of this research was sparkled by Lutz Finger's and Felix Theommes' classes, and I would like to thank them for being awesome teachers. Many thanks to Dr. Michelle Cox, Alison Shea, Zihan Hu, and the anonymous users on StackOverflow have contributed valuable efforts to this research. Professor Yanjun Xie has always been my role model in the area (or in peotry and literature) since my first day at college, and I would like to thank him for the inspiration.

I am incredibly grateful for the cheer my friends brought to me at or beyond Cornell. Atlas and Issy are the most inspirational friends, best project teammates, and the most active bubble tea cohort I could ever ask for. I am grateful to have met Zishuo Li from college and received his encouragement throughout all life stages. Buyun, thanks for being with me on the adventure of trying to blend in

the engineers' world as two hotelies, and thanks for feeding me. Alexa, Susie, Montse... the limited space here can never let me fully express my gratitude, but I am so happy to have spent all the lovely time with you. Particular gratitude is due to Shura Gat, with whom the conversations are some of my best memories at Cornell or in the U.S. My internship at Cornell Women's Resource Center has been one of my most empowering experiences and I am truly grateful for that. Special thanks to Frédéric Chopin and Stewart Adams, whose contribution to either Nocturne and Waltz or Ibuprofen, have got me through many stressful moments or unpredictable toothache attacks.

But just like the theme of this research: uncertainty itself is fascinating, and it worth endless attempts to explore its beauty. My journey at Cornell has been a practice of this exploration, and it empowered me to continue to learn, grow, and challenge myself in a broader world. These two years have been the best time of my life. Cornell, thank you for having me.

This thesis is dedicated to my parents, without whom I could never be in the U.S. pursuing my dream. Mom and Dad, I owe you the world for the boundless love and support you continue to give me. Your unflagging support and unwavering faith are the best gifts I've ever had. I love you two deeply from the bottom of my heart.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the hotel industry, accurate forecast of demand is an essential component in revenue management. Models such as times series, advance booking models, and other combined models are commonly used in hotel demand forecasting. However, the models either fail in recognizing the complex relations between historical bookings and final arrivals, or are not able to capture information from the full booking curves.

On the other hand, machine learning approaches are gaining popularity because of their outstanding performances in forecasting. By applying machine learning models, researchers are able to recognize the non-parametric patterns from data without setting rigorous statistical assumptions.

This research discusses the feasibility of applying machine learning approaches in hotel demand forecast. Two sets of empirical studies are designed: one with booking data from a single hotel with long booking histories, the other from multiple hotels with up to 14 days booking windows and other information including price, location, review score, etc. The results indicate that machine learning approaches outperform pick-up based models especially given long historical data.

Although lacking a systematic understanding of how machine learning approaches contribute to hotel demand forecast, the unique characteristics of some

algorithms may play a vital role in the outperforming results. Firstly, the amount of transaction data in hotel industry increase sharply in the recent decade, and it provides the foundation for machine learning models. Machine learning models require a large number of data points to fit, and hotel industry becomes a perfect setting thanks to the recent rapid development of digitization. The continuing rise of digital platforms and interactions is creating a considerable amount of data for hotel managers and researchers to wade through. As GloabalData (2019) forecasts, there are 8.32 billion hotel rooms nights available all over the world, and over 50% of the hotel bookings worldwide are made online (Hospitality Technology, 2017). The large amount of data provides opportunities for machine learning algorithms to capture intricate patterns and build more stable models.

Secondly, machine learning models can capture the complex and non-parametric relationships between historical bookings and hotel demand. Traditional models such as pick-up based models or regression only have decent accuracy when the shape of the function between predictors and responses is linear, which is hard to validate in hotel demand forecast situation. Machine learning models, in comparison, simulate the arbitrary function according to data at hand, and therefore have the potential to capture the complex relations better.

Last but not least, machine learning models are capable of dealing with high dimensional data, and using booking curves to predict hotel demand is the perfect setting for machine learning to practice. When conducting forecasts, each reservation on hand can be regarded as an independent variable to model the trend in

booking patterns. However, this valuable information cannot be accommodated by pick-up based models or regression. Besides, using multiple reservations on hand in linear regression models can result in multicollinearity. In comparison, some machine learning algorithms can tackle high-dimension data easily. For instance, the K-Nearest Neighbor (K-NN) algorithm calculates the distances between the predicted target and a few neighbors, then takes the average of the nearest distances. This algorithm avoids the restrictions on dimensionality and can function well with long historical booking windows.

There are a few machine learning algorithms which are analogous to classical methods but utilized more information to improve performances. For instance, K-NN outperforms advance booking models by recognizing patterns using full historical booking curves; Random Forest samples out the highly correlated features to avoid overfitting. Even machine learning models have been widely used and proved efficiency in various areas, there has been little publication in hotel industry's academic literature regarding the use of machine learning models in hotel forecasting.

This study provides insights into how machine learning approaches can be applied in hotel demand forecasts. The empirical results confirm the potential of machine learning models in hotel demand forecast. The findings should make an important contribution to the field of hotel revenue management to both industry managers and academic researchers.

The remainder of this thesis proceeds as follows: Chapter 2 presents the lit-

erature review which goes over the classical hotel demand forecast models and machine learning application in related areas. Chapter 3 lays out the main models and theoretical dimensions behind the methods. Chapter 4 introduces the two sets of empirical studies and interprets the initial results. Chapter 5 concludes the main takeaways and offers implications for further research.

CHAPTER 2

LITERATURE REVIEW

2.1 Classical Hotel Demand Forecasting Models

One of the most salient properties which differentiates hotel products from other retail products is advance booking. Advance booking information includes valuable insights on demand prospects, changing trends, booking patterns, etc. Therefore, models which capture the characteristics of advance bookings have always played vital roles in hotel demand forecast.

Advance booking models consider hotel reservations over a range of horizon for a specific stay night. This type of models estimates the increments of future reservations and aggregate the increments into realized demand, as part of the final reservations (Lee, 2018). The booking curve illustrates the accumulation of reservations on hand (ROH) for a specific future date of stay. Advance booking models are also named as ”pick-up” models since the number of bookings is ”picked up” from a specific time point to another. The forecast is calculated by adding the pick-up in a similar condition (e.g. same hotel, same day of week, same season) to the ROH (Weatherford and Kimes, 2003).

Pick-up models are widely used in the hotel industry since they exploit the unique characteristics of reservations throughout the booking window (Zakhary et al., 2008). L'heureux (1986) discusses the classical pick-up models in the airline

context. He calculates the average and weighted average of flight reservations between dates for departed flights for a particular day of week to predict the future pick-up for the same flight number on the same day of week. This concept is quickly applied in the hospitality industry since both airline and hotel industry share the common characteristics of advance reservations.

Zakhary et al. (2008) discuss the main types of pick-up models. From the perspective of the relationship between current bookings and final arrivals, additive pick-up models assume ROH on a certain day before arrival (DBA) is independent of the final arrivals. Therefore, the final demand is forecast as the sum of current bookings and the average pick-up between now and the day targeted. Multiplicative pick-up approaches, on the other hand, assume current bookings are proportional to the final arrivals, and thus the current bookings are multiplied by an average pick-up ratio to get the final forecast.

From the perspective of data completion, pick-up models can be categorized into classical models and advanced models. Traditional pick-up methods only utilize completed booking curves in forecasting. For instance, if today is January 5th and we would like to predict the arrivals for February 5th, classical pick-up methods only allow us using the bookings for days before today (where the curves are completed). The reservations for dates after January 5th are not included since they are incomplete. In comparison, advanced pick-up method uses both complete and incomplete booking information. When calculating pick-ups, Zakhary et al. (2008) suggest there are two methods: simple average method which simply

takes the algebraic average of the pick-ups, and weighted average method which assigns different weight to different pick-ups when taking the average.

Pick-up based models are widely used in empirical studies, but the results are various when compared to other models. Weatherford and Kimes (2003) compare the performances of additive and multiplicative pick-up models with simple exponential smoothing, Holt's Double Exponential smoothing, moving average, linear regression, and logistic regression. They test the models on both small roadside hotels and large business hotels. Results indicate that exponential smoothing and pick-up methods perform most robustly. Tse and Poon (2015) visually defined the relationship between time and ROHs at a certain week as quadratic. They break down the booking window into several segments and fit quadratic regressions separately. The week t is the predictor used to forecast the ROHs at *(90-t)* days before the final arrival day. Chen and Kachani (2007) combine exponential smoothing with advanced pick-up models, and compare the hybrid models with linear regression, advanced pick-up models, and simple exponential smoothing models. Results show that exponential smoothing yields the lowest error rate when predicting final room arrivals. Lee (2018) simulates hotel arrivals by establishing a non-homogeneous Poisson process then extends the models by including booking features such as large variance of the demand, the correlation between early and late bookings, etc. This research uses a daily booking data which ranges from 364 arrivals dates with a booking window of 0-28 days. The results show that Poisson mixture models outperform standard models and linear regression models.

However, very few research in applying pick-up based models have accommodated all booking curves into consideration. Schwartz and Hiemstra (1997) develop booking curve similarity approach to conduct forecast by comparing the incomplete curves of the forecast day to each of the complete curves. The models take four most recent ROH and generate the index by calculating the distances between the incomplete and complete curves. They test the performance of this model in various booking windows and compare the results with times series models and polynomial regression. The curve similarity model outperforms other models significantly. The machine learning models that our research will be using follow the same logic of Schwartz and Hiemstra (1997)'s article, which will be clarified in the following sections.

There have been other models explored by researchers in hotel demand forecast. Time series is another mainstream approach widely applied in the hotel industry. Time series models (also called historical models) seek time patterns (trends, cycles, and seasonal fluctuations) in the single series of historical data, and then models the patterns mathematically. In the hotel industry, time series models consider total room bookings on each night (i.e. the final number of rooms sold or arrivals) as a series of observations, then extend the time series to get forecasts for the future.

There has been a long history of applying time series models in hotel demand forecast. Andrew et al. (1990) use Box-Jenkins and exponential smoothing models to predict hotel occupancy rates. Monthly occupancy rates of one major-city

hotel are used. Even if Box-Jenkins outperformed exponential smoothing models marginally, the authors suggest that exponential smoothing might be more feasible considering its interpretability. Lim et al. (2009) use Holt-Winters triple exponential smoothing and Box-Jenkins models to forecast the total monthly hotel guest arrivals in New Zealand.

Some other researchers add advance booking information to time series. Rajopadhye et al. (2001) use the Holt-Winters process to estimate long-term forecast, and estimate the short term forecast by dividing the ROH on a specific day in the booking window by a historical ratio of the current booking numbers to actual arrivals (analogous to the multiplicative pick-up models which will be mentioned later). Pereira (2016) adds double and complex seasonal patterns to exponential smoothing models. He also adds the trigonometric framework to keep track of several seasonal complexities in the hotel demand forecast. The performances are measured in 1, 2, 4, and 8 weeks horizon and different room types.

However, as stated by (Schwartz and Hiemstra, 1997), a significant drawback of the time series approach is its complexity when constructing the model. For instance, Box-Jenkins models need manual intervention to identify and diagnose parameters, which requires statistical training and relevant mathematical background. This requirement is difficult to reach among hotel practitioners, and the whole procedure also consumes a massive amount of time. Besides, hotel transactions have a unique structure with advance booking, limited capacity, and the forecast period, which time series models also fail to accommodate.

Many research attempts to explore the additional effect exogenous to the system, such as local events, weather change, unemployment rate, etc. Schwartz et al. (2016) include hotel competitive set's predicted occupancy as an input of the daily occupancy forecasting. They randomly generate hotel occupancy data for the target hotel and hotels in the competitive set, and use an evolutionary algorithm to reach the lowest forecast error. The models use a simple linear combination of the target hotel's forecast and an aggregated forecast of the competitors, and then applies the evolutionary algorithm to find the optimal coefficient. Zakhary et al. (2011) estimate the key characteristics affecting hotel arrival and occupancy, then use Monte Carlo simulation to forecast future arrivals. They identify reservations, cancellations, length of stay, no shows, group reservations, etc. as the key features and conduct estimation accordingly. However, it is challenging to quantify the specific scale of the effect from the input to output, and it is extremely difficult to control all practical factors in real empirical studies.

2.2 Machine Learning in Forecasting

Machine learning models have proved their capabilities in forecasting in the last decade (Ahmed et al., 2010). The research which develops machine learning models can be traced back to 1980s from the exploration of neural network models. Machine learning approaches have been widely used in predicting business failure (Gepp et al., 2010; Li and Sun, 2012; Lin et al., 2011), stock price (Alkhatib et al.,

2013; Tsai and Wang, 2009), currency exchange rate (Galeshchuk, 2016; El Shazly and El Shazly, 1999), etc.

However, hotel was historically not one of the industries considered to be at the forefront of technological innovation (GlobalData, 2017). There has been little discussion on applying machine learning approaches in the hotel industry. Most of the applications of machine learning techniques in the industry focus on hotel online review analysis. For instance, Ma et al. (2018) calculate the effect of hotel online reviews with user-provided photo using text mining techniques. Moro et al. (2017) use support vector machine to predict customer online review score given users profile. Phillips et al. (2015) use Artificial Neural Network (ANN) to investigate relationships among online client reviews, hotel characteristics, and revenue per available room (RevPar). Besides online review analysis, Yang et al. (2015) use projection pursuit regression (PPR), ANN, SVM, and boosted regression to predict hotel success indicators (RevPar, profit, labor productivity, and efficiency score) given hotels location. Corazza et al. (2014) apply supervised Multi-Layer Perceptron (MLP) ANN to simulate the procedure of hotel online booking. Instead of establishing a forecasting model according to the hotels structure, the authors utilize ANN to figure out datas internal pattern based on existed customer bookings. They construct a function to respond to customers reservation requests and provide alternate solutions.

In summary, there remains a paucity of research on applying machine learning technique on demand forecasting in hotel settings.

CHAPTER 3

METHODOLOGY

This chapter introduces the primary methodology applied in this study and the theoretical background of the models. The models covered in this session include: 1) Advance booking models (Additive and Multiplicative); 2) Regression models; 3) Machine learning models (K-NN, weighted K-NN, Decision Tree, Random Forest, SVM, and Neural Network).

3.1 Pick-up Models

Pick-up Models, also widely known as advance booking models, use the accumulative reservations over time for a particular stay night to predict the final arrivals. The main idea of pick-up models is using the pick-up method to estimate the increments of reservations for a future day and then aggregate these increments to obtain a forecast of the total arrivals (Zakhary et al., 2008).

Additive pick-up models regard the final arrivals independent of the current ROH and calculate final arrivals by adding pick-ups to the current ROH. Suppose we are forecasting the arrival demand Y_t on stay date t, the forecast conducted by additive advance booking models equals to:

$$ROH_{0,t} = ROH_{today} + P_{i,t} \tag{3.1}$$

where i stands for the forecast is made on i days ahead. In other words, there are i days between today and the forecasting stay date. $P_{i,t}$ refers to the average pick-up between today and t, namely what more reservations for t will emerge over the following i days. $ROH_{0,t}$ is the dependent variable which describes the ROH on the arrival day (on DBA=0).

The pick-up in additive pick-up models is calculated by:

$$P_i = \frac{\sum_{i=1}^{n} (TRUE_t - ROH_{i,t})}{n} \tag{3.2}$$

$$P_{i,DOW} = \frac{\sum_{i=1}^{n_{DOW}} (TRUE_t - ROH_{i,t})}{n_{DOW}} \tag{3.3}$$

Equation 3.6 describes the case that pick-ups are differentiated by the day of week (DOW). In this case, there are different pick-up value to add on depending on the DOW. n_{DOW} stands for the number of observations for a certain DOW.

Different from additive models, multiplicative pick-up models regard the ROH as a certain ratio of the final arrivals. In this case, the final arrival on a future date t is calculated by $\frac{R_i}{PR_i}$ where the PR_i is the average of the ratio of ROH to the final arrival on i days before arrival.

Notice that multiplicative models are sensitive to zeros. When all ROHs on a certain DBA equal to 0 (which might be normal when the booking window is

long), the pick-up ratio is 0 since it is calculated by the average of the ratio of all training observations to true demand. Similarly, in the test set, if an ROH on the nearest DBA is 0, then the forecast will be 0 as well regardless of the value of the pick-up ratio.

Suppose we are forecasting the arrival day *ROH* on stay date t, the forecast should equal to:

$$ROH_{0,t} = \frac{ROH_{today}}{PR_{i,t}} \tag{3.4}$$

where i stands for the forecast is made on i days ahead, and $PR_{i,t}$ refers to the average pick-up ratio between today and t, namely the multiplier of ROH for the targeting day t.

The pick-up ratio in multiplicative pick-up models is calculated by:

$$PR_i = \frac{\sum_{i=1}^{n} (ROH_{i,t}/ROH_{0,t})}{n} \tag{3.5}$$

$$PR_{i,DOW} = \frac{\sum_{i=1}^{n_{DOW}} (ROH_{i,t}/ROH_{0,t})}{n_{DOW}} \tag{3.6}$$

Equation 3.6 indicate the case that pick-up ratios are differentiated by the day of week (DOW).

3.2 Linear Regression

Linear regression assumes the relationship between the ROH on a specific DBA and the final arrival number is linear. Linear regression can either consider the newest ROH instead of the whole booking curve (Equation 3.7), or can include every known ROH on the booking curve (Equation 3.8):

$$ROH_{0,t} = \beta_0 + \beta_1 ROH_i + \epsilon_i \tag{3.7}$$

$$ROH_{0,t} = \beta_0 + \beta_1 ROH_i + \beta_2 ROH_{i+1} + ... + \beta_p ROH_p + \epsilon_i \tag{3.8}$$

where R_i is the ROH on the ith DBA, and p represents the largest DBA in the dataset.

However, either of the models have drawbacks. For the first regression which takes only the newest ROH into account, the historical information before the current day is lost. Instead of using the whole book curve to make forecasts, the first regression is closer to forecasting with a "booking point." Although the second regression makes up this problem by including every predictor on the booking curve, it suffers from multicollinearity since ROH is calculated accumulatively and thus highly correlated with each other. Therefore the coefficients of the second model must be interpreted with caution.

Since linear regression simulates both additive (intercept) and multiplicative (coefficient) relations between the predictor and response, linear regression can be comparable to the combination of additive and multiplicative models. In other words, additive advance booking models is the special case for the regression where $\beta_1 = 1$, and the final arrival is simply the current ROH plus a pick-up, which is the β_0. Multiplicative advance booking models is the circumstance where $\beta_0 = 0$ for the regression.

3.3 Neural Network

The core logic of neural network is extracting linear combinations of the inputs as derived features, then models the observations in interest as a nonlinear function of the fitted features. The neural network, therefore, can be regarded as a multi-step regression.

Neural Network has evolved to a large variety of models and learning approaches, but most commonly it refers to single hidden layer back-propagation network or single layer perceptron (Friedman et al., 2001).

A neural network usually takes two stages to build. Typically there is only one output at the end. Derived features Z_m are generated from the linear combinations of the inputs, and then the observation in interest is modeled as a linear combination function of Z_m:

$$Z_m = \sigma(\alpha_{0m} + \alpha_M^T ROH_m), m = 1, ..., M,$$
$$ROH_{0,t} = \beta_{0k} + \beta_K^t Z, k = 1, ..., K, \tag{3.9}$$
$$f_k(ROH_m) = g_k(ROH_{0,t}), k = 1, ..., K,$$

where Z = (Z_1, ..., Z_M). The activation function $\sigma(v)$ takes various forms. Normal functions includes sigmoid, Gaussian, radial, etc. The output function $g_k(ROH_{0,t})$ transforms the result vector. Z_m, the derived features, are called hidden units because Z_m are an expansion of the original inputs ROH_m and therefore do not originally exist. Weights are the unknown parameters neural network models seek to optimize.

Before establishing the neural network model, scaling the inputs is necessary. The value of inputs determines the value of the weights in the first layer and can have a substantial effect on the forecast solution. Therefore the standardization of all inputs is necessary. This procedure ensures all inputs being treated equally in the regularization stage. A general approach is the min-max normalization, as illustrated in 3.10:

$$SROH_i = \frac{ROH_i - min(ROH_i)}{max(ROH_i) - min(ROH_i)} \tag{3.10}$$

where $SROH_i$ is the scaled response value for original response ROH_i.

Additionally, choosing the appropriate number of hidden units and layers is vital in neural network. If the data is linearly separable, there is no need to use hidden layers as the activation function can be directly applied to input layer to

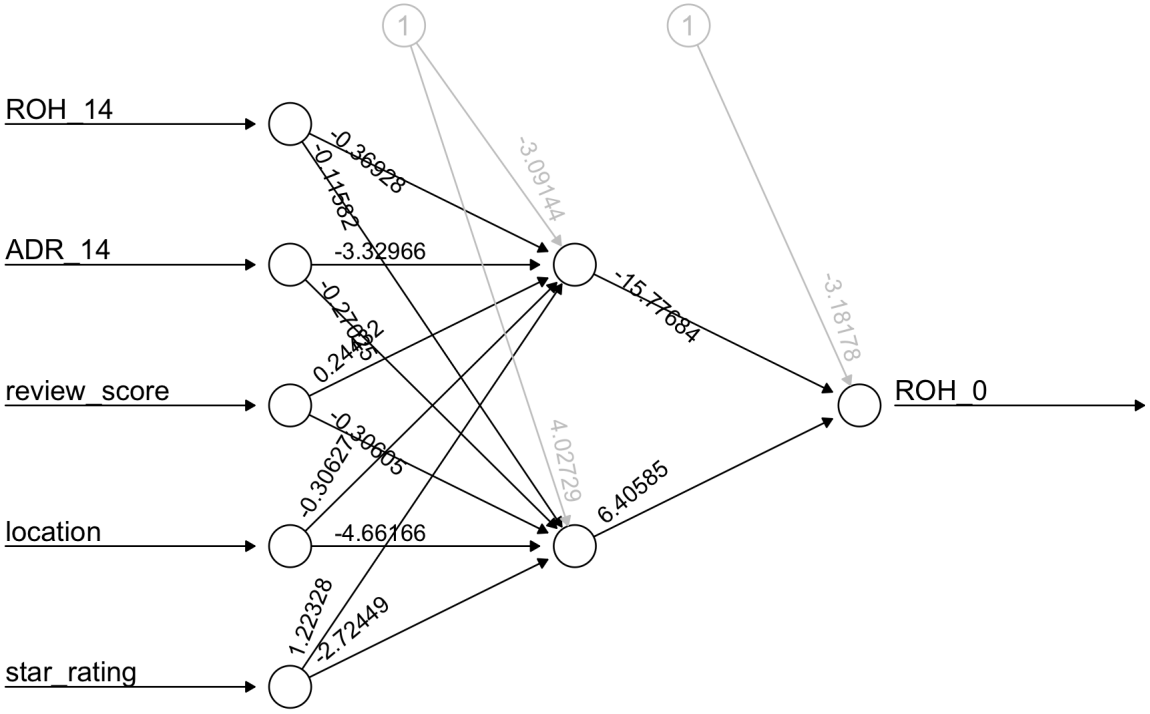
solve the problem. With too few hidden units, the neural network might not be able to capture the non-parametric patterns in the data. On the opposite, too many hidden units might shrink the extra weights to zero and might result in overfitting.

There is no standard procedure of selecting the optimal number of hidden units and layers, however, there are a few common rules to consider: the number of hidden layer units should be around 2/3 of the size of the input layer and the size of the hidden layer neurons should not exceed the size of either input layer or output layer (Karsoliya, 2012).

Neural network models are usually illustrated by Neural Interpretation Diagram (NID). The weights of each predictor on the next layer are illustrated by the arrows and number linking two neurons. Figure 3.1 illustrates a neural network models in hotel demand forecast. Positive effects of input are depicted through positive input-hidden coefficient and positive hidden-output weights, or negative input-hidden and negative hidden-output coefficients (Olden and Jackson, 2002). In other words, the sign of the multiplication of two connection weights indicates the effect that a predictor generates on the response.

Similar to other non-parametric machine learning models, neural network can learn and capture non-linear and complex relationships. Another advantage of neural network is it does not impose any restrictions on the input variables. Additionally, neural network is effective in high dimensional settings.

Figure 3.1: Illustration of Neural Network Using Hotel Reservation Sample



Notes: This is the illustration of Neural Network model established using the dataset with multiple hotels' information including historical booking, pricing, review score, location, and star rating. This graph takes DBA=14 as the example, where ROH_{14}, ADR_{14}, review score, location, and star rating are included as the input neurons. The numbers on the connection lines indicate the weights between neurons. The grey line and numbers represent the errors.

However, the interpretation of the weights on NID can be subjective and complicated. Additional hidden layers will further complicate the interpretation. In cases where the number of input predictors is large, it is challenging to decipher the relationships virtually. Besides, neural network models are also not intuitive

and require expertise to tune, which may be challenging to achieve especially in the hotel industry.

3.4 Nearest Neighbors Models

The nearest neighbor algorithm is one of the most straightforward non-parametric decision rules. Nearest neighbor can be used in the classification problem, which assigns an unclassified observation into the category to the nearest sample. It can also be applied in predicting the test value by taking the average of the k closest neighbors. Given the focus of this current research (the hotel demand, a continuous numeric variable), this section will target the regression perspective.

To calculate the distance between the targeting test value x and existing training observation y_i in a p dimension space, the simplest instance is calculating their Euclidean distance.

Assuming Y_a and Y_b represent two observation on stay date a and b. Y_a and Y_b both have up to p ROHs which indicates the ROH on a specific DBA. Therefore the Euclidean distance between Y_a and Y_b is the length of the line segment connecting both points. Suppose $Y_{t1} = (ROH_{a1}, ROH_{a2}, ..., ROH_{ap})$ and $Y_b = (ROH_{b1}, ROH_{b2}, ..., ROH_{bp})$, the distance between Y_a and Y_b is given by equation 3.11:

$$
\begin{aligned}
d(Y_a, Y_b) &= \sqrt{(ROH_{1a} - ROH_{1b})^2 + (ROH_{2a} - ROH_{2b})^2 + ... + (ROH_{pa} - ROH_{pb})^2} \\
&= \sqrt{\sum_{i=1}^{p} (ROH_{ia} - ROH_{ib})^2}
\end{aligned}
\tag{3.11}
$$

Other distance metrics such as city block, Chebychev, Mahalanobis, Minkowski, etc. are also common in K-NN modeling.

After calculating all the distances between the target observation and all observations in the training set, K-NN uses observations in training set τ closest in input space to x to form $\hat{ROH}_{0,t}$. In the K-NN case, the estimated $\hat{ROH}_{0,t}$ is defined as:

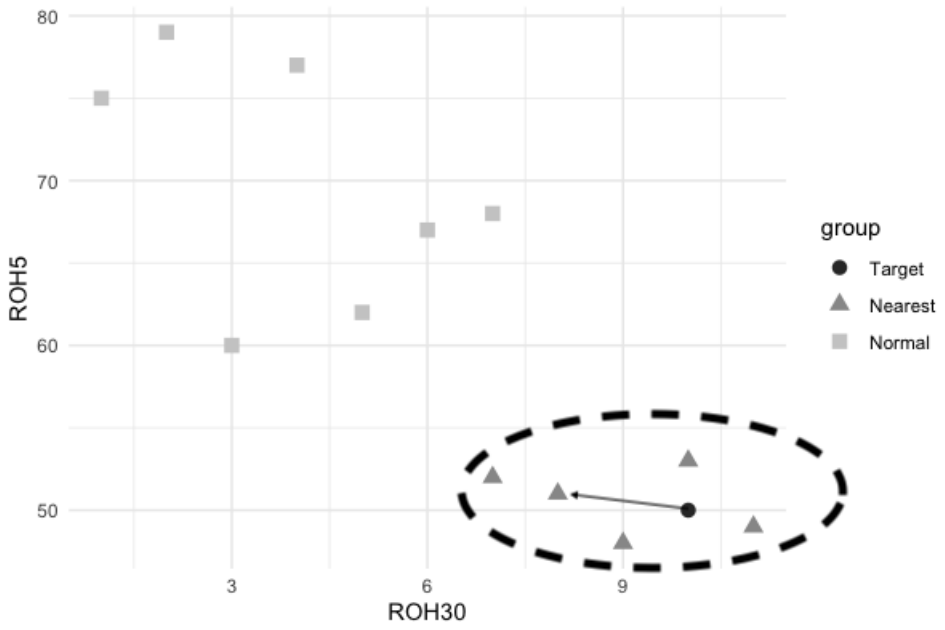
$$
\hat{ROH}_{0,t}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} (ROH_{0,k}) \tag{3.12}
$$

where $N_k(x)$ is the neighborhood of x defined by the k observations x_1, x_2, ..., x_k with the smallest distances in the training set. After locating the closest neighbors, we simply take the average to get the estimated value.

Figure 3.2 is an illustration of this procedure: the model calculates the distance d between the target observation and all of the observations in the training set, and select the five nearest samples with the smallest d. For instance, the target $Y_a = (ROH_{30}, ROH_5) = (10, 50)$, and the triangle observation the target links to with an arrow, has a location of $Y_b = (ROH_{30}, ROH_5) = (8, 51)$, and their distance

$d(Y_a, Y_b) = \sqrt{(ROH_{5a} - ROH_{5b})^2 + (ROH_{30a} - ROH_{30b})^2} = \sqrt{(10-8)^2 + (50-51)^2} = 5$. The five nearest neighbors with the smallest distances are illustrated in grey triangle plots. The predicted ROH_0 of the target is calculated by taking the average of the five nearest neighbors' response: $\widehat{ROH}_{0,a} = \frac{1}{5}\sum(ROH_{0,k})$.

Figure 3.2: Illustration of K-Nearest Neighbors Using Historical Booking Sample



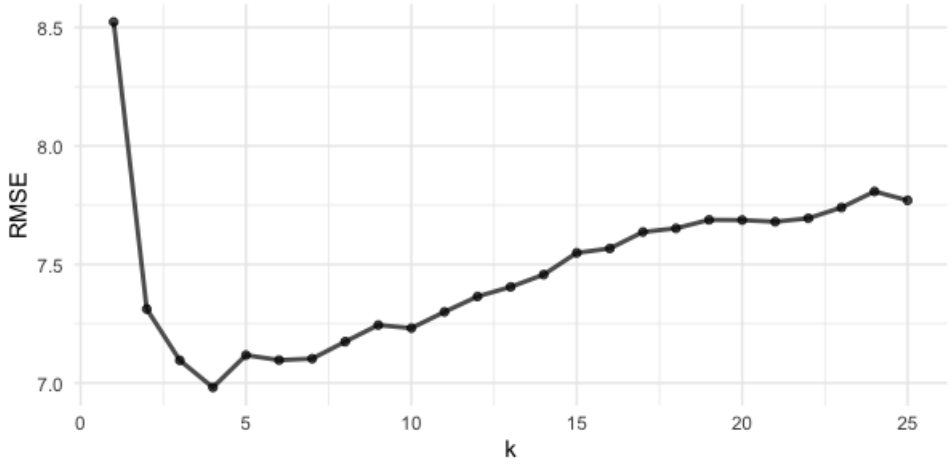
Notes: This is the 2-dimension illustration of K-Nearest Neighbor model established with booking curves. The x-axis is the ROH_{30} for a certain stay date, and the y-axis indicates ROH_5 for a certain stay date. This graph illustrates 5-nearest neighbor case.

As the discussions above, an essential aspect of the K-NN model is to find an appropriate value of k. Generally speaking, the larger the value of K, the more inflexible the models would be, the larger the variance.

Suppose K takes the maximum value (the number of observations in the training set), in this case, the predicted value of the test point would be the arithmetic average of all objects in the training set. In the opposite, when K equals to 1, the predicted value for the targeting subject will be its "the only nearest neighbor" in the training set. In this case, the models are highly possible to commit the fallacy of over-fitting, since the predictions are highly dependent on the training sample. The selection of K needs to consider the trade-off between variance and bias.

Resampling is a common method to select the optimal model parameter. Resampling repeatedly drawing samples from the training set and refitted the models to fetch additional information. K-fold cross-validation approach is a common approach in resampling (James et al., 2014). This approach randomly divides the training set into k folds with approximately equal size. The first fold is used as the validation set, and the rest $k - 1$ folds are used to fit the models. After repeating the procedure for k times, the K value which generates the smallest error is selected to apply in the model. Notice that the small k distinct from the value of K (the number of nearest neighbors) we are pursuing. Figure 3.3 illustrates this procedure where a list of K value is tested in the randomly split folds and the test error is recorded. In this case, the K value of 4 is selected since it generates the lowest error.

Figure 3.3: Illustration of the Parameter Selection of K-NN



K-NN algorithm makes forecasts based on the average of the nearest neighbors, and each neighbor has the same influence on the prediction. However, under some circumstances, it makes more sense that closer neighbors should be given more weight when making forecasts.

Dudani (1976) brought up "distance-weighted K-NN rule" which suggested a weighting function which varies with the distance between the predicted sample and neighbors, and heavier weight is given to closer neighbors.

Suppose $K(t)$ is the weighting function of the distance d with the maximum in $d = 0$ (the predicting observation overlap with the neighbor), and decreases with d grows. This function $K(t)$ is usually called a kernel weight (Altman, 1992). The kernel weights can be simulated by various forms of functions, for instance Hechenbichler et al. (2004) :

- Rectangular kernel: $\frac{1}{2} \cdot I(|d| \leq 1)$

- Triangular kernel: $(1 - |d|) \cdot I(|d| \leq 1)$

- Quadratic kernel: $K(t) = 6(\frac{1}{4} - t^2)$

- Cosine kernel: $\frac{\pi}{4} cos(\frac{\pi}{2} d) \cdot I(|d| \leq 1)$

- Gauss kernel: $\frac{1}{\sqrt{2\pi}} exp(-\frac{d^2}{2})$

- Inversion kernel: $\frac{1}{|d|}$

The detailed discussion of kernel is beyond the scope of this research. In the empirical study, the specific kernel used is automatically chosen by computer software.

3.5 Tree Models

Tree-based models partition the feature space into a group of rectangles and fit a simple model in each space. A decision tree is composed of multiple judgment nodes, representing a mapping relationship between the attributes and values.

To build a decision tree, we start by finding the optimal split for attributes which generates the largest information gain. Currently, the primary measure metrics for information gain are entropy (for ID3 and C4.5) and the Gini impurity (for CART algorithm) (Yao et al., 2018).

We divide the set of possible values for predictors $ROH_1, ROH_2, ..., ROH_p$ into J distinct and non-overlapping regions. Then for observations falling into the region R_j, we make the prediction which equals to the mean of the response values for all the training observations within R_j. By trying stratifying at different values, we find the segmentations with $R_1, R_2, ..., R_j$ which minimize the residual sum of squares (RSS), given by $\sum_{j=1}^{J} \sum_{i \in R_j} (ROH_{0,i} - \hat{ROH}_{0,j})^2$, where $\hat{ROH}_{0,j}$ is the mean of responses for training observations within the jth region.

Tree models simplify the stratifying procedure by conducting recursive binary splitting. The models first select the predict X_i and the cut point s such that splitting the predictor space into the regions $R_1(j, s) = \{ROH|ROH_j < s\}$ and $R_2(j, s) = \{ROH|ROH_j \geq s\}$. We see the value of j and s which minimize equation 3.13:

$$\sum_{i:x_i \in R_1(j,s)} (ROH_i - \hat{ROH}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (ROH_i - \hat{ROH}_{R_2})^2 \tag{3.13}$$

We then repeat the process, seeking for the best predictor and the best cut point to split the data further minimizing the RSS within each of the resulting regions. We do this partition on one of the previously identified regions.

This recursive process continues until a stopping criterion is reached, and this is the procedure of tree pruning. There are two possible methods to select the optimal subtree. We can start building the tree and stop at certain thresholds such as current regions contain no more than a certain number of observations or the

marginal RSS decrease reaches a point. Alternatively, we could also grow a very large tree first, then prune it back by cross-validation.

Decision tree models have many attractive features. A major advantage of the decision tree is its interpretability. Conducting forecasts using decision tree models closely mirrors human-being's decision-making process. This benefit is particularly useful in hotel management practice since it is easy to understand by the general audience without a strong statistical background. Another advantage of the decision tree is it can efficiently deal with qualitative predictors without creating dummy variables, which is superior to K-NN and neural network models.

However, tree models usually do not have the same predicting accuracy due to its binary splitting, and they can be very non-robust. A tiny change in the training data can cause a substantial change in the tree models, which might cause the problem in actual hotel demand forecasting. Further steps to increase the stability of the decision tree can be applied, and random forest in the next section is a commonly used approach.

Hod (1998) firstly proposed combining multiple trees constructed in randomly selected subspaces can significantly improve generalization accuracy. Breiman (2001) brought up the term of "random forest" which modifies the bootstrapping and establishes an extensive collection of de-correlated trees.

To solve the problem that tree models can be non-robust, we consider adding bootstrap procedure (or bagging) to reduce the variance of the tree models. In

practice, a bootstrap sample Z (out of N) is retrieved from the training data. Then, we randomly select m variables from the p variables, and pick the best feature splitting point among m, with two smaller nodes. The value of m is usually 1/3 of p. Now we generate a random-forest tree T_b from the bootstrapped data. This tree-generating process is recursively repeated until the minimum node size n_{min} is reached.

Suppose all of the trees consisting a space $\{T_b\}_1^B$, then the prediction for a new point x is:

$$\hat{f}_{rf}^{\,B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \tag{3.14}$$

The variance of tree models can be significantly decreased since the bootstrapping process captures the complex interaction structures in the data. Even when the trees are grown deep enough, the bias is relatively low (Friedman et al., 2001).

The predictor-subsetting feature of the random forest can be beneficial in the hotel demand predicting case. To conduct forecasts using the booking curve, the ROH on the newest DBA is usually the strongest predictor of the final arrivals. Even after bootstrapping, the trees will always use the newest ROH to build the models, and the bagged models are highly correlated with each other. In other words, bootstrapping will not lead to a substantial variance reduction over a single tree. However, random forest eliminates this predictor at on average $(p-m)/p$ of the time, which gives other predictors more chances and reduce the variances

for the overall models.

The drawback associated with improving accuracy by using the random forest is the interpretability is lost. By randomly selecting part of the variables and repeating the process, it is difficult to decipher the specific effect from a predictor to the response. In practice, random forest can be more likely to be used merely as a ”black box” forecasting machine, and it is hard to derive extra insights from the models itself.

3.6 Support Vector Machine

Tree models split data in a binary manner, which generate virtual ”square boxes” when the tree grows. However, for some data, the flat surface (as the box) cannot categories data efficiently, and more flexible, a non-parametric hyperplane is necessary. Support Vector Machine (SVM) is one widely used machine learning algorithm which is able to split the data using more flexible boundaries.

SVM was first proposed in the 1990s by Vapnik (1995) and other researchers. The logic of SVM is maximizing the distance between two classes and find the non-parametric boundary.

Suppose a data matrix X has n training observations in a p-dimensional space, and all of the samples can be categorized into two classes. To accurately classify a targeted test observation also with p features, we would like to develop a classifier

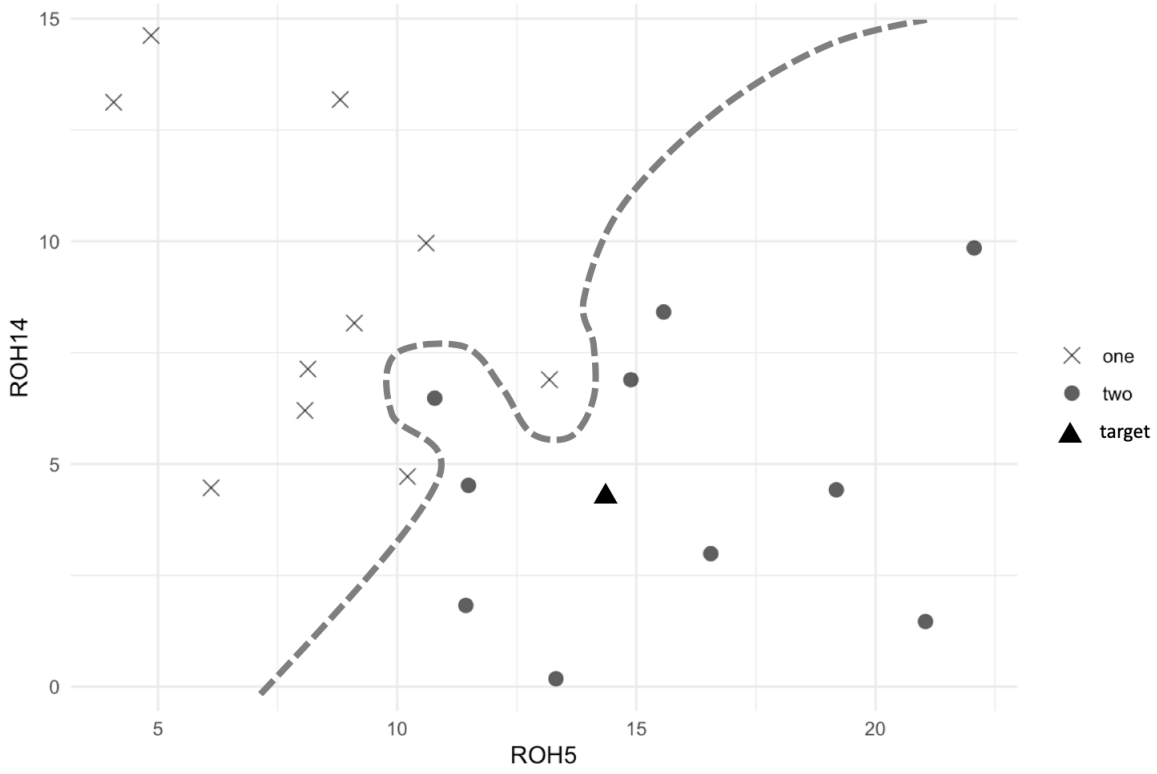
that will correctly classify the test observation using the feature measurements.

There can be infinite possibilities of setting up the classifier if both classes do not overlap. In this case, we select the hyperplane which is farthest from the training observations around, namely with the largest margin. Then the test observation is classified by its location of the hyperplane. However, the two classes might be overlapping, and it is impossible to find a separating hyperplane. In this case, support vector classifier, an outcome of the maximal margin, optimizes the solution of the problem:

$$\begin{gathered} ROH_{0,i}(\beta_0 + \beta_1 ROH_{i1} + ... + \beta_p ROH_{ip}) \leq M(1 - \epsilon_i), \\ \text{maximizing M where } \beta_0, \beta_1, ..., \beta_p \subseteq M, \\ \text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1 \end{gathered} \tag{3.15}$$

where β_0, β_1, ..., $\beta_p \subseteq M$, M is the width of the margin and β_i defines the classifier. By plugging in the features of the targeting test observation in the equation above, we can the classification of the observation based on the sign of $f(x) = \beta_0 + \beta_1 ROH_1 + ...\beta_p ROH_p$. Figure 3.4 illustrates a case in two dimensional space.

Figure 3.4: Illustration of Support Vector Machine Using Historical Booking Sample



Notes: This is the 2-dimension illustration of Support Vector Machine (SVM) model established with hotel booking curves. The x-axis is the ROH_5 for a certain stay date, and the y-axis indicates ROH_{14} for a certain stay date. The SVM is generated under the max margin rule which maximize the distance from the boundary to support vectors. The estimated demand for the target (black triangle) is by taking the average of the observations on the right side of the boundary.

The optimization procedure is beyond the scope of this current research. However, it can be attempted to achieve through enlarging the feature space using dif-

ferent terms of the predictors. Some simple terms my be quadratic (ROH_1^2) and polynomial ($ROH_1 * ROH_2$), and the possible interaction terms form $ROH_j * ROH'_j$ for $j \neq j'$. There are other popular choices to construct transformed space, for instance, the dth-Degree polynomial, Gaussian, radial basis, neural network, etc. Those kernels transform the input space into a new feature space with a higher dimension, where it is easier to find a separable hyperplane.

Generally speaking, the more support vectors used in the model, the more complex the model is. Linear hard margin SVM only needs two support vectors to define the boundary, while soft-margin linear SVM needs a couple of more, but every "twist" and "turn" in non-linear SVM need a support vector. In this case, the accuracy depends on the trade-off between a highly complex model which may overfit the data, and a simple boundary which may classify observations wrong. In the extreme case, every observation in the training data can be a support vector, and the SVM is completely overfitted.

One of the main advantages of SVM is its ability to be universal approximators of any multivariate function to any desired degree of accuracy (Kecman and Wang, 2005). The kernel trick of SVM makes it possible to models the unknown, highly nonlinear, complicated patterns between the predictors and response. This characteristic of SVM is especially suitable for hotel demand forecasting. Hotel demand is affected by many tangible and intangible factors such as internal characteristics (chain scale, location, brand, etc.) and external factors (macroeconomy, conferences, festivals, etc.). The relationships between these factors and the final

arrivals are hard to describe and also extremely challenging to validate. SVM can capture those ”invisible” patterns and give pretty accurate forecasts.

A common disadvantage of SVM is its parameters are extremely hard to interpret. By fitting a kernel, SVM fits a higher dimensional space and the marginal contribution of each predictor is hard to capture. Besides, the searching procedure for an optimal kernel function usually takes long training time, which might not be efficient for hotel practitioners to apply.

CHAPTER 4

EMPIRICAL STUDY

A set of empirical studies are adopted to evaluate the effectiveness of machine learning approaches. There are two sets of data applied in the empirical research: one from a single hotel with up to one-year advance booking information; the other from multiple hotel properties from the same market with shorter booking windows up to 14 days. Each set of data is randomly split into training and test sets to validate the models' performances.

4.1 Single Hotel with Long History

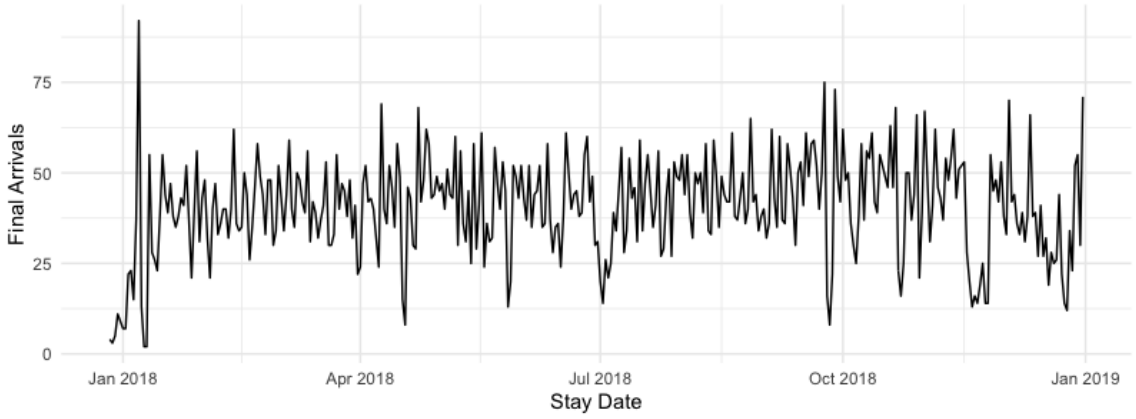
Hotel managers strive to predict future demand of the hotel property to make pricing and other arrangements in order to optimize revenue. Large hotel company or hotels have been in operations for years usually have rich data either in the amount of transactions, or historical data range. In this situation, information from historical booking (booking curves) and chronological demand changes are valuable. Therefore, the first empirical study (Empirical Study 1) is designed for forecasting demand for a single hotel given booking curves with long booking windows.

The data used in this empirical study is fetched through a hotel revenue management company. The dataset includes information of the hotel's client arrivals (in terms of rooms sold) and booking curves (advance booking). The arrival date

ranges from 12/27/2017 to 12/31/2018, with booking date from one day ahead up to 396 days ahead.

Figure 4.1 illustrates the number of final arrivals on each day of the subject hotel. It can be seen from the graph that the final arrivals fluctuate significantly.

Figure 4.1: Empirical Study 1: Stay Date Arrivals



Notes: The x-axis indicates the check-in date of the hotel reservations. The y-axis describes the number of accumulated reservations on the check-in day, namely the number of final arrivals for the hotel.

Bookings happen for a specific date is accumulated in terms of ROH, and for each arrival date, the ROH consists of the booking curve for this date. The ROH on different DBA is the independent variable in each model, and the final arrival on each stay date is the dependent variable.

80% of the stay dates are randomly selected in the training set, while the rest 20% are in the test set. There are 296 days in the training set, and 74 in the test set.

Models are fitted with the samples in the training set. The accuracy is calculated by the errors between the predicted values and actual values from the test set.

This study examines in 12 different forecasting horizons: 1, 2, 3, 4, 5, 6, 7, 14, 21, 30, 60, 90, and beyond. In the hotel industry, the dynamic pricing plan is usually set according to the cutoffs above. Aggregating on those horizons can adequately cover different rates of reservation accumulation on the booking curve. It is noticeable that during early periods when the booking day is far away from the stay date, the reservations accumulate very slow. Therefore, a wider horizon in earlier periods allows information to accumulate. When the stay dates are approaching closer, the booking window is broken into smaller horizons for closer attention.

In this empirical study, I use nine different models to forecast hotel demand with historical booking information.

4.1.1 Pick-up Models

Observations in the training set are used to calculate the ”pick-ups” which will be applied on the test set to verify performances. Additive pick-up model takes the average of the absolute value difference between current ROH and ROH_0.

By visually examining Figure 4.2, we can find that the pick-ups differentiate significantly by the day of week (DOW). Therefore, I take DOW into account when making forecasts using pick-ups.

Figure 4.2: Empirical Study 1: Additive Pick-ups (Differentiate by DOW)

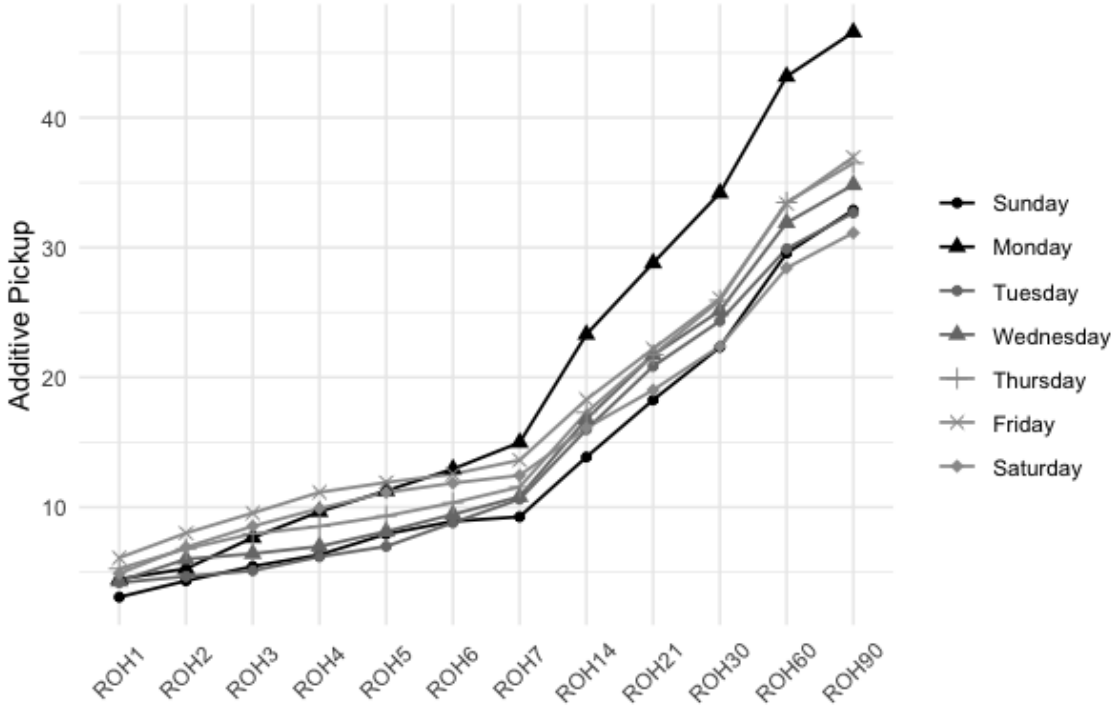


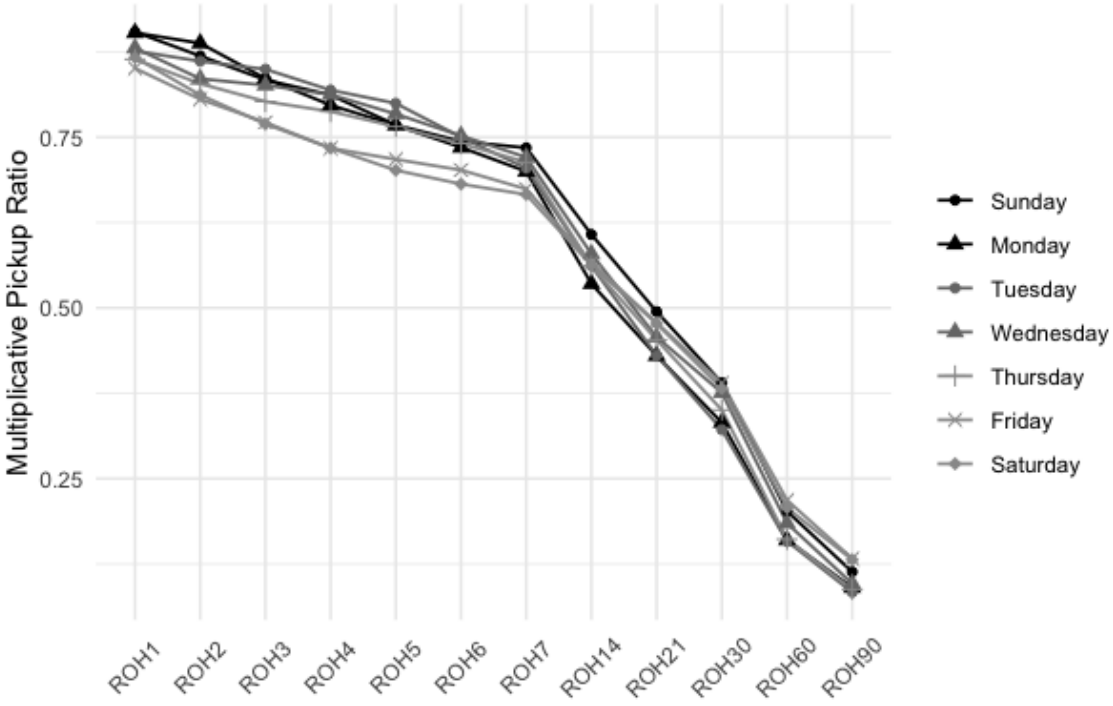
Table 4.1 shows the value of pick-ups in additive advance booking models. Taking DBA=5 as the example, if the observation in question is a Sunday, then the forecast is conducted by adding 7.93 on the ROH of that day at DBA=5.

For multiplicative pick-up models, the pick-up ratio is calculated using the observations in the training set. The final forecasts are conducted by dividing the target ROH by the pick-up ratio instead of adding pick-ups. The pick-up ratios are in the range of $(0, 1)$. The larger the DBA, the smaller the pick-up ratio (as in Figure 4.3.

Table 4.1: Empirical Study 1: Additive Pick-ups

DOW	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_{14}	ROH_{21}	ROH_{30}	ROH_{60}	ROH_{90}
Sunday	3.07	4.33	5.44	6.33	7.93	8.91	9.26	13.9	18.3	22.3	29.6	32.9
Monday	4.46	5.26	7.65	9.65	11.26	12.94	14.98	23.3	28.8	34.2	43.2	46.6
Tuesday	4.18	4.67	5.10	6.18	6.97	8.79	10.62	16.0	20.8	24.3	29.9	32.7
Wednesday	4.29	6.00	6.42	6.98	8.12	9.44	10.80	16.8	21.7	25.1	31.9	34.8
Thursday	5.26	6.79	7.95	8.53	9.32	10.34	11.55	17.3	21.7	25.9	33.5	36.5
Friday	6.09	8.00	9.55	11.14	11.90	12.55	13.62	18.3	22.2	26.1	33.4	37.0
Saturday	4.89	6.92	8.53	9.89	11.13	11.85	12.45	16.1	19.0	22.4	28.4	31.1

Figure 4.3: Empirical Study 1: Average Multiplicative Pick-up Ratios



Notice that even the ratio curves seem not too significantly differ from each other, small differences in the denominator can generate a huge variance. Therefore, I consider DOW when conducting forecasts with multiplicative pick-up models.

The value of pick-up ratios on each DOW is displayed in Table 4.2. Taking DBA=5 as the instance, if the targeting test observation is a Monday, the forecast for this observation would be the current ROH divided by 0.77.

Table 4.2: Empirical Study 1: Multiplicative Pick-up Ratios

DOW	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_{14}	ROH_{21}	ROH_{30}	ROH_{60}	ROH_{90}
Sunday	0.90	0.87	0.83	0.81	0.77	0.74	0.73	0.61	0.49	0.39	0.20	0.11
Monday	0.90	0.89	0.84	0.80	0.77	0.74	0.70	0.54	0.43	0.33	0.16	0.09
Tuesday	0.88	0.86	0.85	0.82	0.80	0.75	0.70	0.56	0.43	0.32	0.16	0.08
Wednesday	0.88	0.84	0.83	0.81	0.78	0.75	0.72	0.58	0.46	0.38	0.18	0.10
Thursday	0.86	0.83	0.80	0.79	0.77	0.74	0.71	0.56	0.45	0.35	0.16	0.09
Friday	0.85	0.81	0.77	0.73	0.72	0.70	0.67	0.57	0.48	0.39	0.22	0.13
Saturday	0.87	0.81	0.77	0.73	0.70	0.68	0.67	0.56	0.48	0.38	0.21	0.13

4.1.2 Linear Regression

There are two ways to model the relationship between ROHs and stay date arrivals using the regression model. The model can either use multiple ROHs on the booking curves as predictors (as equation 4.1), or use the newest ROH as the single predictor (as equation 4.2). The results of both models are illustrated in

Table 4.3 and Table 4.4.

$$ROH_0 = \beta_0 + \beta_1 ROH_i + \epsilon \tag{4.1}$$

$$ROH_0 = \beta_0 + \beta_1 ROH_i + \beta_2 ROH_{i+1} + ... + \beta_p ROH_p + \epsilon \tag{4.2}$$

Table 4.3: Empirical Study 1: Regression DBAs Results (using Partial Booking curves)

	DBA 1	DBA 2	DBA 3	DBA 4	DBA 5	DBA 6	DBA 7	DBA 14	DBA 21	DBA 30	DBA 60	DBA 90
(Intercept)	3.50***	5.02***	5.43***	5.91***	6.76***	7.37***	7.75***	12.55***	17.83***	21.94***	28.63***	34.26***
	(0.51)	(0.58)	(0.66)	(0.71)	(0.76)	(0.82)	(0.91)	(1.14)	(1.25)	(1.33)	(1.33)	(1.11)
ROH_1	1.35***											
	(0.13)											
ROH_2	−0.10	1.37***										
	(0.19)	(0.15)										
ROH_3	−0.22	−0.32	1.25***									
	(0.19)	(0.23)	(0.17)									
ROH_4	−0.03	−0.09	−0.14	1.40***								
	(0.21)	(0.24)	(0.28)	(0.19)								
ROH_5	0.13	0.20	0.03	−0.25	1.31***							
	(0.19)	(0.22)	(0.25)	(0.27)	(0.18)							
ROH_6	−0.16	−0.15	−0.06	0.12	−0.01	1.49***						
	(0.17)	(0.20)	(0.23)	(0.25)	(0.27)	(0.19)						
ROH_7	0.10	0.05	0.02	−0.13	−0.12	−0.29	1.40***					
	(0.13)	(0.16)	(0.18)	(0.19)	(0.21)	(0.23)	(0.09)					
ROH_{14}	−0.06	−0.11	−0.10	−0.09	−0.12	0.00	−0.18	1.37***				
	(0.07)	(0.09)	(0.10)	(0.11)	(0.12)	(0.13)	(0.14)	(0.12)				
ROH_{21}	0.05	0.12	0.07	0.06	0.08	−0.07	−0.05	−0.04	1.39***			
	(0.07)	(0.08)	(0.09)	(0.10)	(0.11)	(0.12)	(0.13)	(0.18)	(0.15)			
ROH_{30}	−0.08	−0.16*	−0.11	−0.16	−0.14	−0.14	−0.14	−0.03	−0.06	1.42***		
	(0.07)	(0.08)	(0.09)	(0.10)	(0.11)	(0.11)	(0.13)	(0.17)	(0.20)	(0.14)		
ROH_{60}	0.03	0.09	0.07	0.13	0.03	0.07	0.07	−0.26	−0.17	−0.22	2.03***	
	(0.10)	(0.12)	(0.14)	(0.15)	(0.16)	(0.17)	(0.19)	(0.25)	(0.30)	(0.34)	(0.30)	
ROH_{90}	−0.01	0.06	0.02	−0.01	0.05	−0.02	−0.09	−0.12	−0.27	−0.13	−0.77*	1.43***
	(0.10)	(0.11)	(0.13)	(0.14)	(0.15)	(0.16)	(0.18)	(0.24)	(0.29)	(0.33)	(0.37)	(0.19)
R^2	0.96	0.94	0.92	0.91	0.89	0.87	0.84	0.71	0.58	0.46	0.27	0.16
Adj. R^2	0.95	0.94	0.92	0.90	0.89	0.87	0.84	0.71	0.58	0.45	0.27	0.16
Num. obs.	296	296	296	296	296	296	296	296	296	296	296	296
RMSE	3.10	3.66	4.17	4.54	4.92	5.35	5.91	7.90	9.48	10.78	12.46	13.38

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

As shown in Table 4.3, even though taking multiple ROHs in the model, it is still the newest ROH which plays a significant role. Other predictors besides the newest ROH are insignificant, and some of them have negative values, which indicates the risk of multicollinearity. A model with severe multicollinearity could be highly unstable in the forecast.

As the statistics of the regression models in Table 4.3 and Table 4.4 show, the model performance (measured in R^2 and adjusted R^2) are similar, which means only including the newest ROH can have the comparable performance as modeling with multiple ROHs. The former also avoid the risk of multicollinearity since only one predictor is included. Therefore, in this empirical study, I use the newest ROH in the regression model (as in Table 4.4)as the benchmark.

In this set of regression models, the values of both the coefficient β_1 and intercept β_0 increase with the length of the booking window. In other words, the further future day the forecast is targeting on, the larger intercept and coefficient of the current ROH will be added on and multiplied by. This results fit the common sense in the hotel industry and especially align with the results from additive and multiplicative pick-up models as in Figure 4.2 and Figure 4.3. The ROH far away from the arrival day is usually small, and will increase gradually with the stay date approaches.

Table 4.4: Empirical Study 1: Regression Models Results (only using the Newest ROH)

	DBA 1	DBA 2	DBA 3	DBA 4	DBA 5	DBA 6	DBA 7	DBA 14	DBA 21	DBA 30	DBA 60	DBA 90
(Intercept)	3.70***	4.95***	5.40***	5.92***	6.66***	7.38***	8.02***	12.85***	18.13***	22.12***	29.41***	34.26***
	(0.50)	(0.58)	(0.65)	(0.71)	(0.76)	(0.83)	(0.91)	(1.14)	(1.24)	(1.31)	(1.28)	(1.11)
ROH1	1.03***											
	(0.01)											
ROH2		1.03***										
		(0.02)										
ROH3			1.06***									
			(0.02)									
ROH4				1.08***								
				(0.02)								
ROH5					1.10***							
					(0.02)							
ROH6						1.11***						
						(0.03)						
ROH7							1.14***					
							(0.03)					
ROH14								1.20***				
								(0.05)				
ROH21									1.20***			
									(0.06)			
ROH30										1.26***		
										(0.08)		
ROH60											1.48***	
											(0.15)	
ROH90												1.43***
												(0.19)
R^2	0.95	0.94	0.92	0.90	0.89	0.86	0.83	0.70	0.57	0.45	0.26	0.16
Adj. R^2	0.95	0.94	0.92	0.90	0.89	0.86	0.83	0.70	0.57	0.45	0.26	0.16
Num. obs.	296	296	296	296	296	296	296	296	296	296	296	296
RMSE	3.12	3.67	4.15	4.55	4.93	5.41	6.00	7.98	9.54	10.79	12.53	13.38

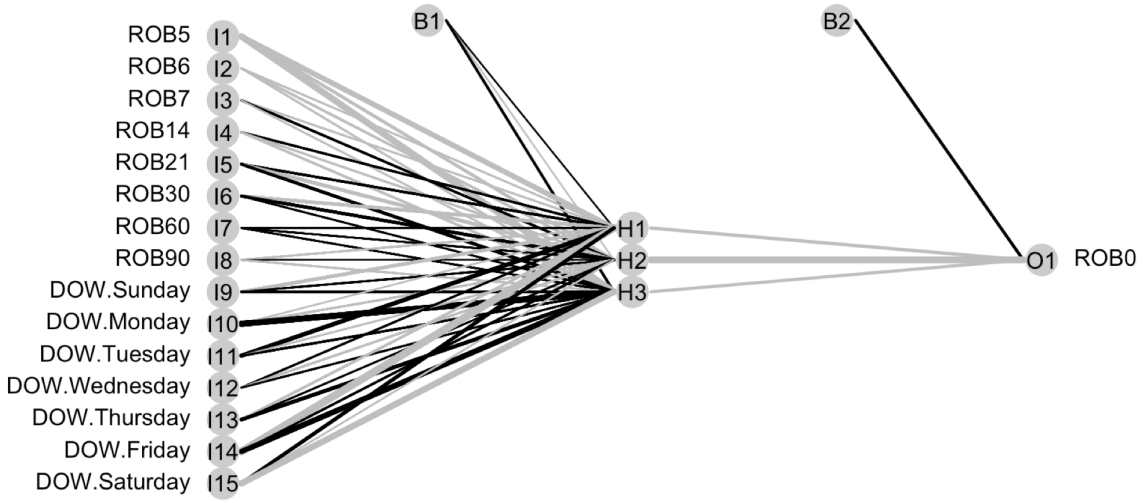
$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Comparing the R^2 and Adjusted R^2 results, we can find that the further the prediction is conducted, the lower the explained variance is. It aligns with the practice as well: when a prediction is made long way before the tarting day, less information is given and more unpredictable change may happen between now and future. The results show that when the forecasts are conducted 30 days before, the variance explained by the regression models falls below 50%.

4.1.3 Neural Network

The neural network is fitted through R's *neuralnet* package (Fritsch et al., 2019). Taking DBA=5 as an example, the corresponding neural network models has one hidden layer with three neurons. Figure 4.4 illustrates the relationship among input neurons and the output. Black lines indicate the positive coefficient, while gray lines represent negative relations. The thicker the line is, the stronger the relationship is.

Figure 4.4: Empirical Study 1: Neural Interpretation Diagram



Notes: This is the illustration of Neural Network model established using the dataset with one hotel's long booking history. This graph takes DBA=5 as the example. The black lines indicates positive connection weights, while the grey lines indicates negative connection weights.

It is noticeable that all coefficients from the hidden neuron to output are negative, therefore, if the connection between input and hidden neurons are negative, the effects of the input on output are on the opposite, positive. From Figure 4.4 we can find that ROH_5 has a stronger connection than the rest of the ROHs, and Monday, Friday, and Saturday have strong impacts on the final arrivals. Table 4.5 displays the value of the weights from the input neurons to the three neurons in the hidden layer. It is shown in the table that DOW and ROH $_5$ have stronger connections with the hidden layer.

The coefficient of H_1, H_2, H_3 to the output neuron is 15.1, 10.7, and 13.5. The intercept of the hidden layer to the output is 9.78. The error of this model is 91.5. The reached threshold is 0.0095, and it takes 3,290 steps to reach this threshold.

Table 4.5: Empirical Study 1: Neural Network Model Neuron Weights

	H1	H2	H3
Intercept	1.098	-1.34	-1.803
ROH_5	-2.332	14.42	4.621
ROH_6	-5.519	6.38	1.239
ROH_7	1.503	-6.43	-1.632
ROH_{14}	3.165	-7.08	10.528
ROH_{21}	-1.430	-8.81	0.499
ROH_{30}	1.528	11.11	-0.759
ROH_{60}	-0.372	-2.91	-3.090
ROH_{90}	0.771	9.87	1.781
Sunday	1.237	-3.07	11.378
Monday	-4.635	4.63	-18.908
Tuesday	-0.626	-8.55	-4.329
Wednesday	-3.368	1.27	-6.359
Thursday	3.195	24.31	-1.463
Friday	-0.927	19.12	-11.118
Saturday	-0.623	-1.92	-4.148

Notes:
This table displays the value of weights in the empirical study where one single hotel's data is used. The H1, H2, H3 columns indicate the effect coefficients of input neurons (in the first column) on three neurons in the hidden layer. This table illustrates an example of DBA=5.

4.1.4 K-NN

As the model's name itself indicates, K-NN attempts to find the "nearest neighbor" to conduct the forecast. In this current study, the empirical models try to find the most similar booking curves of the target by calculating the Euclidean distance and make the prediction by calculating the average of the closest booking curves. I use R's *caret* package to establish the model (Kuhn et al., 2019).

When establishing the model, it is essential to find an appropriate value for the number of nearest neighbors (the value of K). In this empirical study, I randomly select 8 K values, then apply 10-fold cross-validation to find the best K. This action randomly breaks all the booking curves in training set into ten folds, fits the models with 9 of those folds, and tests the model's performance in the last fold. This procedure is repeated 3 times to find the optimal k. RMSE (Root Mean Square Error) is used as the criteria to find the optimal models.

We are still taking DBA=5 as an example. When using the booking curves up to 5 days before arrival, the computer randomly selects a range of values of K, in this case, K takes 5, 7, 9, 11, 13, 15, 17, 19. 21, and 23. Then by using each K, the models are tested in the 10-fold cross-validation, and the RMSE is recorded. In this empirical study, we can find that when K takes the value of 7, it generates the lowest RMSE (as in Table 4.6). Therefore, when establishing the real models for DBA=5, we use k=7 which means searching for seven nearest neighbors in the training booking curves for each observation in the test set.

Table 4.6: Empirical Study 1: RMSE of K-NN models with various K values

k	5	7	9	11	13	15	17	19	21	23
RMSE	8.39	8.33	8.35	8.52	8.62	8.65	8.82	8.87	8.93	9

Note:

Taking DBA=5 as the example.

All this procedure is repeated when DBA takes another value and a brand new model is established. This parameter selection procedure is repeated every time a model is fitted. In other words, the K for each model can be different. Table 4.7 displays the value of K used in each of the 12 models.

Table 4.7: Empirical Study 1: Parameter Selection (K) for K-NN models

DBA	1	2	3	4	5	6	7	14	21	30	60	90
K	7	9	5	5	7	7	5	5	7	9	13	15

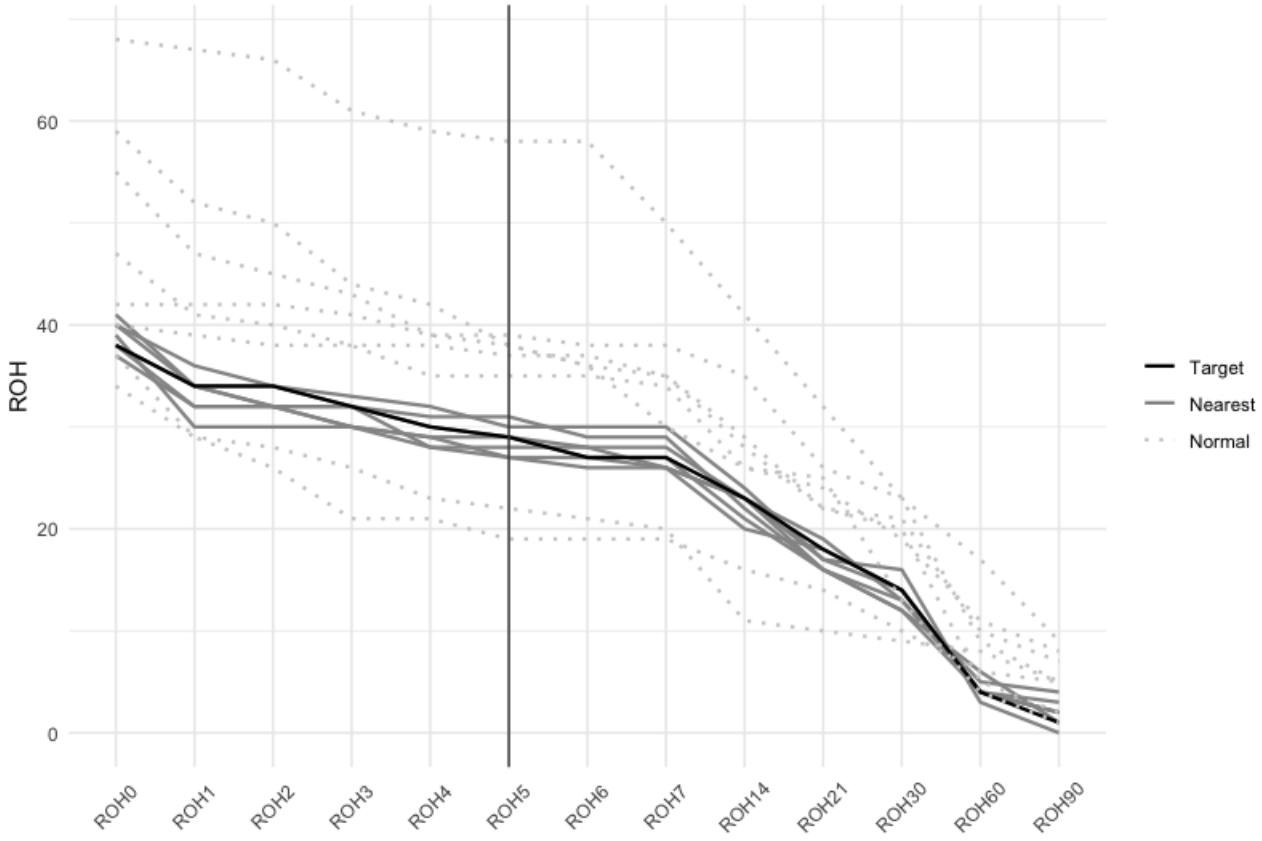
Note:

At each DBA, a new model is established. Before buidling the model, a cross-validation is conducted to select the optimal k value which generates the lowest RMSE. This chart displays the results of the selected k.

By using the empirical data, Figure 4.5 illustrates the nearest booking curves the K-NN model finds when DBA=5. The solid black line is the target in the test set, and the solid grey lines are the seven nearest neighbors the models selects. I also randomly select a few not-so-close observations in the training set and plot them with dotted lines. It can be seen from the figure that the target and its nearest

neighbors are geographically close.

Figure 4.5: Empirical Study 1: 7-Nearest Neighbors for Booking Curves



Notes: This graph illustrates the K-NN model from the one single hotel with long booking history dataset. This model takes DBA=5 as the example. The solid black line is the target in the test set, and the solid grey lines are the seven nearest neighbors the models selects. The dotted grey lines are other randomly selected reservations. To make the forecast for the target, the model selects the seven nearest curves by using only partial curves on the right side of the black vertical line. Then after finding the most similar curves, the model makes the forecast by taking the average of the ROH_0 of the seven nearest neighbors.

The model selects the seven nearest curves by using only the ROHs on DBA 5 and beyond. In the graph, only curves on the right side of the black vertical line are used to fit the model. Then after finding the most similar curves, the model makes the forecast for the target by taking the average of the ROH_0 of the seven nearest neighbors.

4.1.5 Weighted K-NN

Compared to K-NN, weighted K-NN calculates the distances using different kernel functions. Similar to parameter selection for K-NN, the value of K is selected through cross-validation. A range of kernels is considered when selecting K: rectangular, triangular, epanechinikov, Gaussian, rank, and optimal. The criteria for choosing the optimal K is the Mean Absolute Error (MAE). The Weighted K-NN models are fitted using R's *kknn* package (Schliep and Hechenbichler, 2016).

Similar to the K-NN model, weighted K-NN makes the forecast by finding the "nearest" booking curves for the target, but the "nearest" is measured by more flexible kernel shapes. Taking DBA 5 as an example, the model conducts 10-fold cross-validation to test 20 K values and a range of kernel shapes. As 4.8 indicates, when K=11 and the kernel is triangular, the model has the lowest MAE. The triangular kernel is described as equation 4.3:

$$(1 - |d|) * I(|d| \leq 1) \tag{4.3}$$

where d represents the Euclidean distance.

Table 4.8: Empirical Study 1: Parameter Selection of Weighted K-NN models

k	rectangular	triangular	epanechnikov	gaussian	rank	optimal
1	7.94	7.94	7.94	7.94	7.94	7.94
2	6.27	6.87	6.84	6.29	6.60	7.27
3	5.91	6.35	6.32	5.90	6.13	6.71
4	5.87	6.12	6.08	5.84	5.93	6.38
5	5.76	5.89	5.86	5.72	5.80	6.17
6	5.79	5.77	5.74	5.70	5.74	6.01
7	5.70	5.75	5.73	5.63	5.69	5.89
8	5.66	5.68	5.67	5.58	5.64	5.82
9	5.77	5.64	5.63	5.67	5.64	5.76
10	5.75	5.59	5.59	5.62	5.63	5.72
11	5.78	5.58	5.59	5.65	5.63	5.69
12	5.89	5.58	5.60	5.72	5.65	5.67
13	5.93	5.61	5.63	5.76	5.69	5.67
14	5.89	5.61	5.63	5.73	5.71	5.66
15	5.92	5.63	5.66	5.74	5.73	5.67
16	5.96	5.63	5.65	5.77	5.75	5.67
17	6.02	5.66	5.69	5.82	5.77	5.68
18	6.11	5.69	5.72	5.89	5.80	5.69
19	6.16	5.71	5.74	5.92	5.83	5.71
20	6.18	5.72	5.75	5.93	5.86	5.72

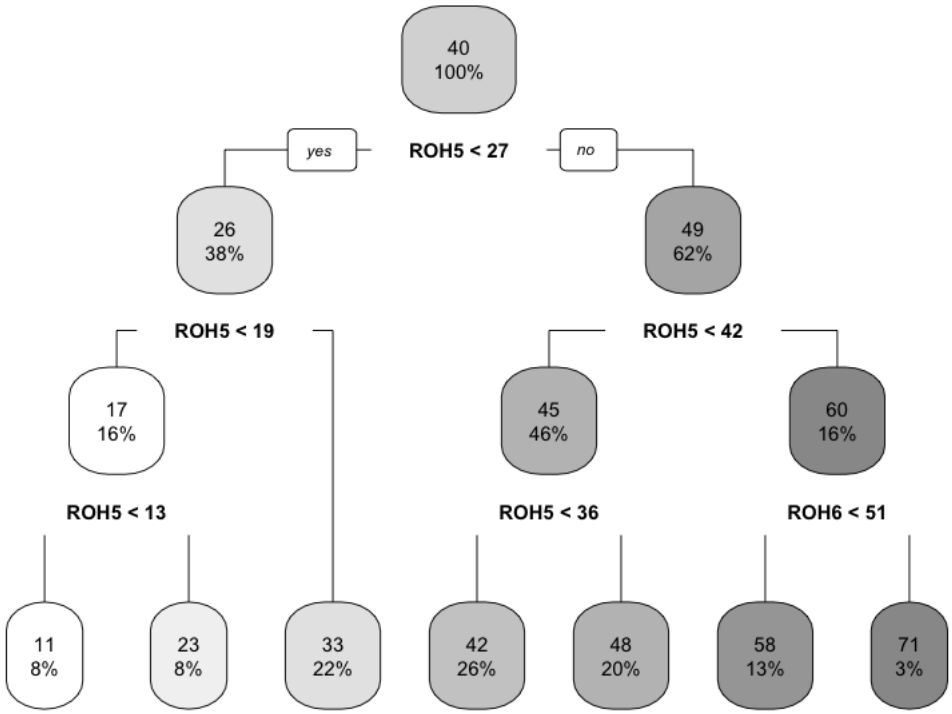
Note:

Taking DBA=5 as the example.

4.1.6 Decision Tree

Decision tree consists of a series of splitting nodes starting at the top of the tree. Figure 4.6 illustrates a decision tree models at DBA=5.

Figure 4.6: Empirical Study 1: Decision Tree Illustration



Notes: This graph illustrates the decision tree model from the one single hotel with long booking history dataset. This model takes DBA=5 as the example. The numbers in rounded rectangles indicate the average value of the responses under each node. The percentage values in rounded rectangles represent the percentage of the observations under each node takes to the whole training set.

The top node assigns observations with ROH no more than 27 to the left branch, and the predicted value for this subset is given by the average response value in training set with ROH larger or equal to 27. In this case, the forecast is 49 and there are 62% of the observations in this group. Continuing on the node, if the booking curve has ROH on DBA5 larger or equal to 42, and ROH on DBA6 larger or equal to 51, the predicted value is 71.

The tree model tries out different possible splitting methods then find the optimal node which minimizes the Residual Sum of Squares (RSS). Usually, trees follow along with top-down approach which begins at the top of the tree, and then successively split the predictor space further down. The splitting procedure stops when the RSS decrease reaches a threshold, which is set as 0.01 in this model empirical study.

Decision tree also illustrates the ”importance” of variables when splitting the data. The more top the split is located, the more ”important” the split variable is. In these models, the newest ROH information plays a significant essential role to conduct the final forecast. A new predictor ROH on DBA=6 appears at the bottom of the tree. However, this high dependency on ”strong” predictors ignores the information provided by other parts of the booking curve. Random forest model in the next section provides a solution to this problem.

4.1.7 Random Forest

Different from the decision tree which starts with all predictors, random forest randomly selects a subset of the whole predictor sets, then fits the models accordingly. The split is only allowed to use each of the predictors once, which eliminates the problem that the decision tree is highly dependent on a few variables.

I use R's *caret* (Kuhn et al., 2019) and *randomForest* (Liaw and Wiener, 2002) packages to build and visualize the random forest models. By default, the random forest models fits 100 trees per time.

Before the random forest is established, a 10-fold cross-validation is conducted to select the optimal number of predictors sampled for splitting at each node (Table 4.9). In the DBA=5 example, a range of numbers are tested, and 10 predictors are sampled for the model to choose at each split. The trees are only allowed to fit a model using the ten predictors selected this time, and this predictor subsetting is called bagging. The strongest indicator ROH_5 might not be even in the predictor pool at some split, and this approach ensures other parts on the booking curve are taken into consideration.

Table 4.9: Empirical Study 1: Parameter Selection of Random Forest

mtry[1]	RMSE	R^2	MAE	RMSESD	R^2 SD	MAESD
2	5.52	0.867	4.40	0.906	0.032	0.531
3	5.17	0.882	4.17	0.845	0.026	0.496
4	5.07	0.886	4.06	0.843	0.027	0.557
5	5.00	0.890	4.01	0.824	0.026	0.561
6	4.99	0.890	3.98	0.835	0.026	0.575
7	4.94	0.893	3.95	0.815	0.025	0.583
8	4.94	0.893	3.94	0.826	0.026	0.602
9	4.90	0.894	3.92	0.804	0.026	0.590
10	4.88	0.895	3.90	0.805	0.026	0.607
11	4.88	0.895	3.90	0.781	0.025	0.599
12	4.89	0.895	3.88	0.771	0.025	0.589

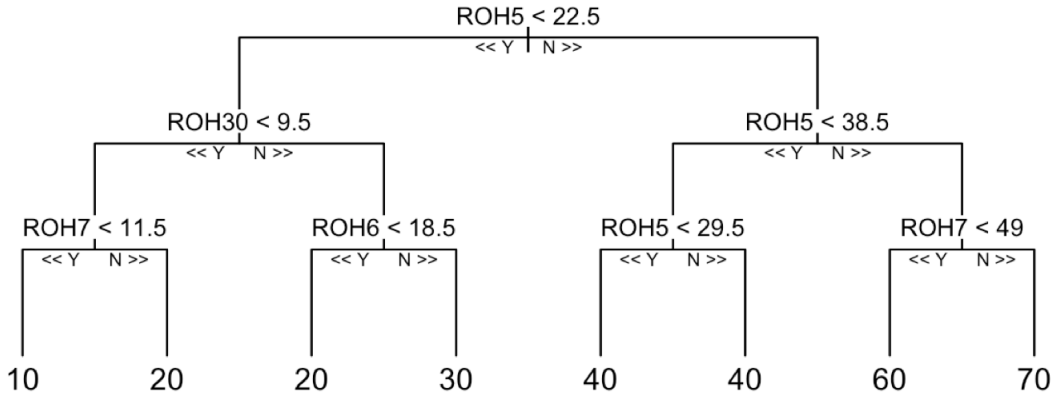
Note:

This graph takes DBA=5 as the example.

[1] The number of variables randomly sampled as candidates at each split.

Figure 4.7 shows a randomly selected tree from the forest for DBA=5. Compared to the decision tree model which heavily depends on the newest ROH, the tree fitted in random forest include other predictors such as ROH_{30}, ROH_6, and ROH_7.

Figure 4.7: Empirical Study 1: A Tree Sample from the Random Forest



Notes: This graph selects one sample from the random forest model generated from the one single hotel with long booking history dataset. This graph takes DBA=5 as the example.

Random forest also evaluates the importance of variables. The level of importance is measured by the mean decrease in accuracy (in terms of mean squared error, MSE). As in Table 4.10, when involving ROH_5 in the forest, the MSE drops 21,554, which is significantly larger than other predictors. Some ROHs on the further time range and DOW cannot significantly increase the model accuracy.

Table 4.10: Empirical Study 1: Variable Importance by Random Forest

	IncNodePurity[1]
DOW	846
ROH_5	21554
ROH_6	15887
ROH_7	10965
ROH_{14}	5357
ROH_{21}	3550
ROH_{30}	1950
ROH_{60}	877
ROH_{90}	859

[1] Measured in terms of mean squared error, MSE.

4.1.8 Support Vector Machine

Among all the ROH on various DBAs, SVM attempts to find the non-linear boundary to divide observations and make forecasts. Each SVM conducts feature selection on kernel shapes over linear, polynomial, radial basis, and sigmoid.

The SVM models fitted for DBA=5 uses radial basis kernel to conduct forecast. 239 support vectors are used to define the boundary. Compared to 296 training samples, this is a pretty complicated and highly non-parametric model. The radial kernel can be described in equation $exp(-\gamma|u - v|^2)$, and in this models gamma = 0.003. A small gamma value indicates a wide kernel, which means the hyperplane

is relatively flat.

4.2 Multiple Hotels with Short Booking Window

Third-party agencies such as consulting firms and revenue management company need forecasting modelS to differentiate according to hotel characteristics. The data used in this empirical study includes transaction data from 18 hotels located in Las Vegas, Nevada. The stay dates of the transactions range from 01/01/2018 to 06/19/2018.

There are 2,744 booking curves in the dataset. I randomly select 80% of the observations, 2,195 records, as the training set. The rest 20%, 549 pieces of observations are assigned in the test set. This process is repeated as a robustness test to see if model performance is consistent on different data.

There are 24 hotel properties in this dataset. I scraped the customer review score and location of each hotel from tripadvisor.com. The customer review is measured from a scale from 1 to 5, with 5 as the highest satisfaction level. The location is defined by "the distance from the property to downtown" and is in units of miles. The information of hotel properties is given in Table 4.11. As the table shows, the volume of transactions and room prices are different for each hotel.

Table 4.11: Empirical Study 2: Hotel Information

ProductName	ROH_0[1]	var(ROH_0)[2]	ADR_0[3]	var(ADR_0)[4]	star	review	location
Aliante Casino and Hotel	1.46	0.635	39.3	1486	4	4.5	12.63
Bally's	6.03	12.617	73.9	3381	4	3.8	0.08
Caesars Palace	6.43	29.695	145.8	25239	4.5	4.1	0.19
Green Valley Ranch Resort and Spa	1.30	0.493	66.2	4354	4	4.4	7.99
Hard Rock Hotel	4.33	9.134	75.6	2821	4	3.9	1.21
HRH Tower at Hard Rock Hotel & Casino	1.33	0.479	61.0	4040	4	3.9	1.21
M Resort	1.10	0.099	25.4	346	4	4.6	10.35
Mandalay Bay	7.49	44.223	109.1	4078	4	4.2	1.57
MGM Grand	10.27	51.575	115.8	4587	4	3.9	0.93
Mirage	6.62	25.347	101.1	3585	4	4.2	0.53
New York - New York	8.17	23.676	91.2	3170	4	4.1	0.80
Palms Casino Resort	2.78	5.953	67.0	2391	4	3.9	1.23
Palms Place Hotel and Spa	1.49	0.725	67.8	5662	4	4.2	1.46
Paris Las Vegas	10.22	50.200	103.2	3887	4	4.0	0.22
Park MGM Las Vegas	9.88	65.494	74.5	2252	4	3.6	0.74
Planet Hollywood	5.12	8.549	100.0	6803	4	3.9	0.29
Red Rock Casino Resort and Spa	2.15	2.905	72.5	3344	4.5	4.5	9.44
Rio	9.60	39.933	72.6	2850	4	3.8	0.79
South Point	2.36	2.623	59.9	2690	4	4.4	7.10
The Cromwell	1.10	0.151	114.0	13220	4.5	4.2	0.03
The Signature at MGM Grand	3.18	5.766	86.4	4054	4.5	4.6	0.61
Treasure Island - TI	11.56	62.049	88.2	2289	4	4.1	0.61
Tropicana Las Vegas - A Doubletree by Hilton	2.61	3.679	44.4	1795	4	4.0	0.97
Trump International Hotel and Tower	1.52	0.960	90.2	9624	4.5	4.5	0.98

[1] ROH_0 represents the average ROH on the arrival day across all stay dates for a hotel.

[2] var(ROH_0) is variance of ROH_0.

[3] ADR_0 is the average daily rate on the arrival day for a hotel.

[4] var(ADR_0) is variance of ADR_0.

The dependent variable in this empirical study is the ROH_0, which describes the ROH on DBA 0, namely the final arrivals. The predictors in this dataset are:

1. ADR_i: average daily rate on i days before arrival, the mean transaction price of a certain arrival date. For instance, there could be a couple of reservations with different prices for January 1st at DBA=5. The ADR takes the average prices of all transactions for January 1st on DBA 5. i is in the range of (1, 14).
2. ROH_i: reservations on hand on i days before arrival. ROH describes the accumulated reservations for a specific stay date in the future. Each ROH_i is a predictor in this dataset.
3. Star: the chain scale star rating of the hotel. The hotels in this dataset are luxury hotels with 4 or 4.5 stars.
4. Review score: customer review score on the scale of 1-5.
5. Location: the distance in units of miles from the property to the city center.

In this empirical study, I compare the performances of pick-up models (both additive and multiplicative), regression, neural network, K-NN and weighted K-NN, decision tree and random forest, and SVM.

4.2.1 Pick-up Models

Table 4.12 illustrates the pick-up and pick-up ratios calculated from the training set. The larger DBA, the larger value of pick-up since more reservations are ex-

pected to happen during the booking window. Similarly, the larger the DBA, the smaller the ratio the current ROH is to be levered.

Table 4.12: Empirical Study 2: Pick-ups and Pick-up Ratios

DBA	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Pick-up	0	1.11	1.93	2.49	2.93	3.33	3.7	4.05	4.35	4.65	4.89	5.15	5.38	5.59	5.8
Pick-up ratio	1	0.791	0.653	0.563	0.5	0.445	0.388	0.337	0.288	0.243	0.206	0.163	0.124	0.085	0.044

4.2.2 Linear Regression

There are two ways to build linear regression models: one is using all historical information on the booking curves, the other one only uses the newest ROH and pricing. Results show that in the first model, only the newest ROH is significant among all other ROHs in most cases. It is the same for pricing information: prices on other time slot do not play significant roles in the model. After establishing the second regression and comparing its performance with the first model, I found that both of the models have a similar R^2 and Adjusted R^2. The full results of both sets of regression models can be found in Table A.4 and Table A.5.

Therefore, in this empirical study, I only use the newest ROH, the relevant ADR, and hotel information to construct the regression model, as in equation 4.4:

$$\hat{ROH}_{0,t} = \beta_0 + \beta_1 ROH_i + \beta_2 ADR_i + \beta_3 Star + \beta_4 Review + \beta_5 Location + \epsilon_i \tag{4.4}$$

where $\hat{ROH}_{0,t}$ is the estimated arrivals on day t. $ROH_{i,t}$ is the ROH on i DBA for

the arrival date t. $ADR_{i,t}$ is the average daily rate for day t at DBA i.

14 different regression models are established in this study. In terms of coefficients, results indicate that ROH always plays a significant role in the models with positive coefficients.

Table 4.13 shows the results of the regression model constructed at DBA=5. As the table shows, all of the predictors are significant in this model, and this model explains 74% of the variance of the data. When controlling all other factors unchanged, the ROH_5 increases one, and the final arrivals will increase 1.47. When other factors remain constant, the one-dollar increase of ADR comes with 0.01 more hotel arrivals. Other predictors seem to have negative effects on stay day arrivals. For instance, when other factors remain unchanged, a 4.5-star hotel averagely has 0.75 less final arrivals than a 4-star hotel.

Table 4.13: Empirical Study 2: Taking DBA=5 as the example

	DBA=5
(Intercept)	5.69***
	(1.48)
ROH_5	1.47***
	(0.02)
ADR_5	0.01***
	(0.00)
review	−0.95*
	(0.37)
location	−0.10**
	(0.03)
star 4.5	−0.75***
	(0.22)
R^2	0.74
Adj. R^2	0.74
Num. obs.	2195
RMSE	2.97

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

From the results, we can find in Table A.5 that both of the R^2 and adjusted R^2 decrease when the booking extends. This fits the practical situation in hotel bookings: the closer the forecasting date, the more accurate the forecast usually is.

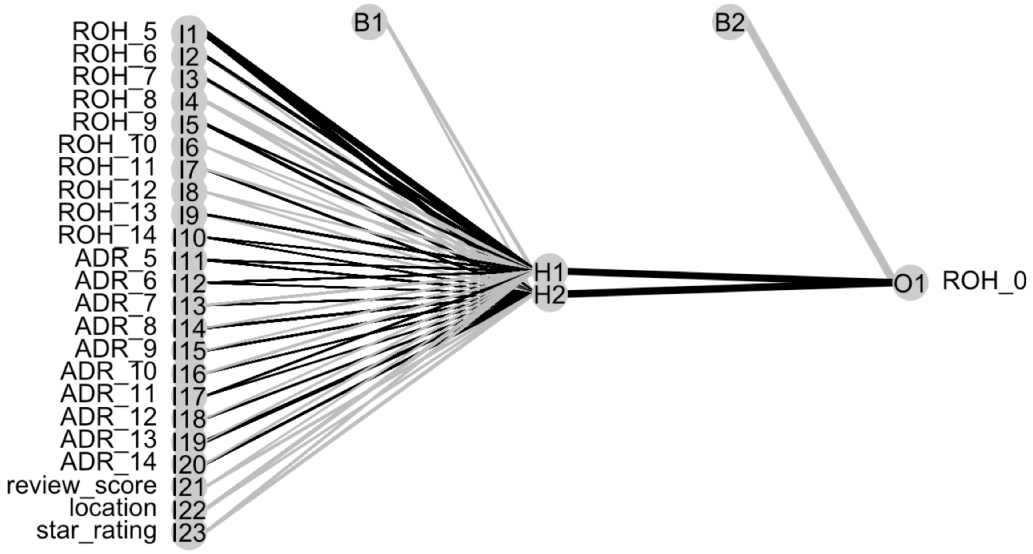
4.2.3 Neural Network

Before fitting the neural network models, I scale the dataset since the value ranges of the predictors are different. Scaling ensures the averagely similar effects among the predictors even if some predictors have large values.

Figure 4.8 visualizes the neural network models when taking DBA=5. This diagram shows the relative impact of ROH, price, star rating, location, and review score in predicting hotel demand. This impact is determined in three steps: input and the first hidden layer, the first hidden layer and the output layer.

Black lines indicate positive correlations, while the grey lines represent negative coefficients. It is visually difficult to decipher the specific effect from one input on the output since each neuron has complex connections to multiple other neurons, and the hidden layer also shifts the effect additionally.

Figure 4.8: Empirical Study 2: Neural Network Illustration



Notes: This graph illustrates the DBA=5 example from the neural network models. The dataset used here includes multiple hotels with information including historical booking, pricing, review score, location, and star rating. The black connection lines between the neurons indicate positive correlations, and the grey lines indicate negative correlations.

Table 4.14 displays the weights of between each input neuron and the neurons in the hidden layer. As shown in the chart, ROH_5 has the strongest connection between itself and the two hidden neurons, and the effect is positive. In this model, the error is 664.56, and the model used 114 steps to reach the threshold 0.48.

Table 4.14: Empirical Study 2: Neural Network Weights

	H1	H2
Intercept	-3.187	-1.613
ROH_5	6.366	5.314
ROH_6	2.192	-0.258
ROH_7	1.838	-1.251
ROH_8	-2.436	-1.712
ROH_9	0.533	1.926
ROH_{10}	-0.074	-0.055
ROH_{11}	-1.133	0.883
ROH_{12}	-0.384	-0.587
ROH_{13}	1.170	-0.615
ROH_{14}	0.265	0.144
ADR_5	0.505	0.669
ADR_6	0.435	0.188
ADR_7	-1.074	0.321
ADR_8	-0.461	1.145
ADR_9	-0.086	0.777
ADR_{10}	-0.971	0.236
ADR_{11}	0.551	0.032
ADR_{12}	-0.301	0.556
ADR_{13}	-0.274	2.436
ADR_{14}	-0.538	1.036
review_score	-0.019	-1.471
location	-2.474	-0.713
star	-1.118	-2.007

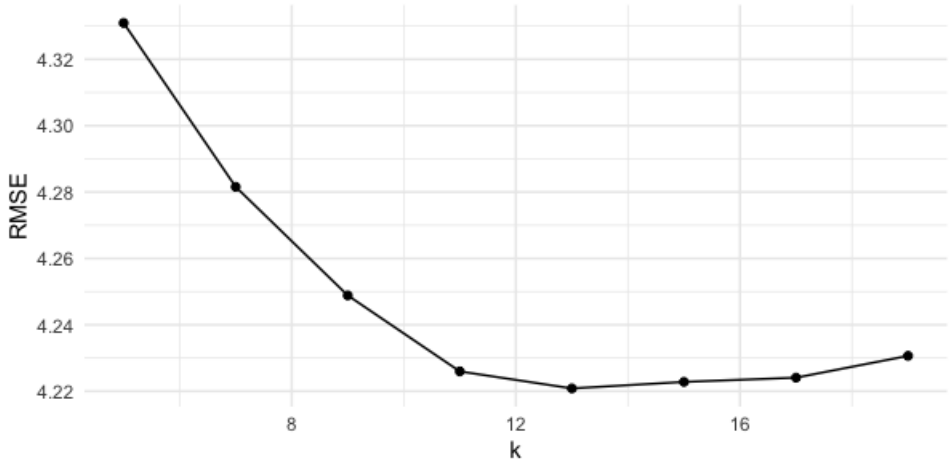
Note:

Taking DBA=5 as the example. H1 column indicates the coefficients of input neurons on the first neuron in the hidden layer, and H2 column shows the coefficients of inputs on the second neuron in the hidden layer.

4.2.4 K-NN

Similar to the previous study, I apply parameter selection before establishing the K-NN models. I use 10-fold cross-validation and repeat the procedure three times to select the optimal value of K. 8 randomly selected K values are tested. Taking DBA=5 as an example, 5, 7, 9, 11, 13, 15, 17, and 19 are tested through cross-validation using the training set. Figure 4.9 shows that when k takes the value of 13, the models' RMSE reaches the lowest. Then the K-NN model for DBA=5 uses k=13.

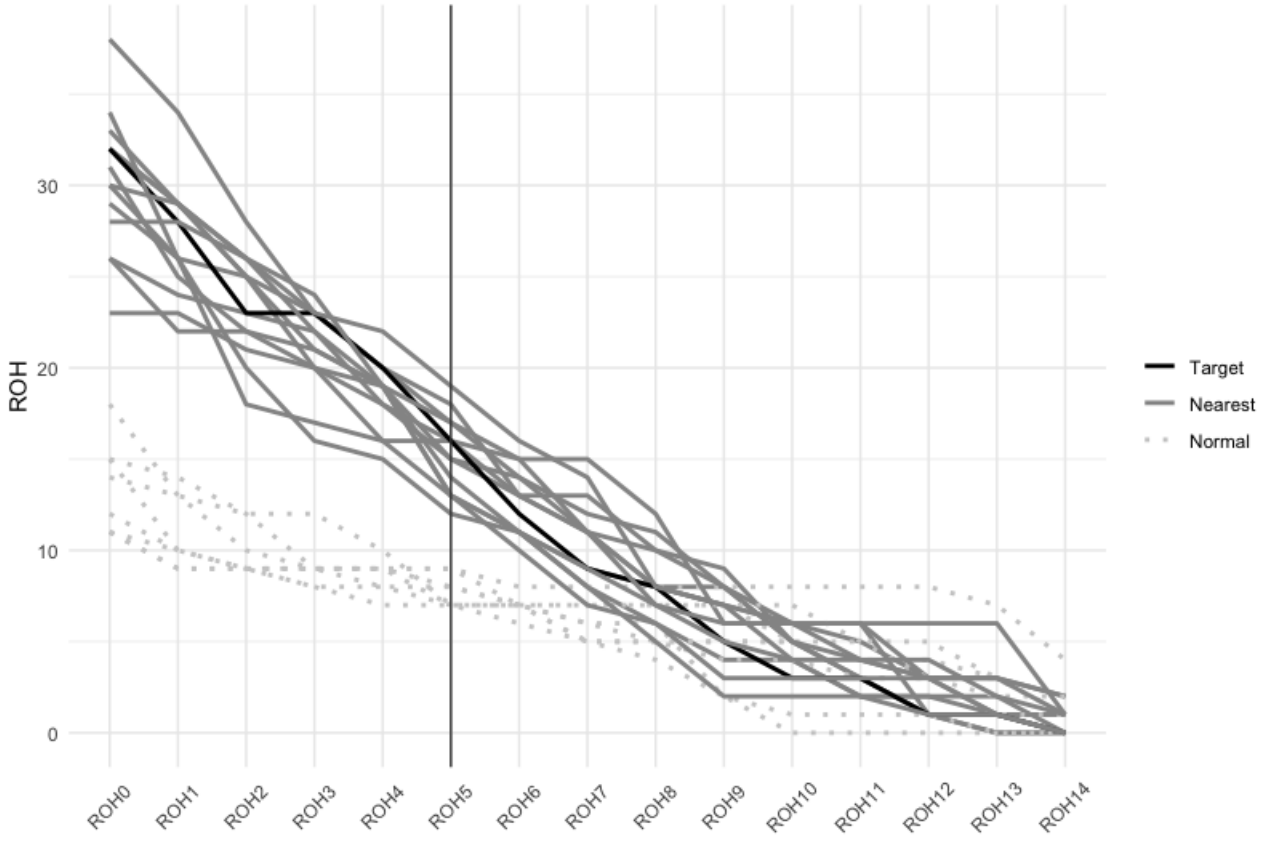
Figure 4.9: Empirical Study 2: Parameter Selection for K-NN



There are 24 predictors and one response in this model, however, it is extremely hard to visualize dimensions over 3. Therefore, I partially plot out the booking curves of the sample. As in Figure 4.10, the solid black line is the observation in question and the 13 grey solid lines are the nearest neighbors the K-NN

model finds. Notice that in this model, K-NN searches for nearest neighbors by the shape of the booking curves and pricing curves after DBA=5. In other words, from the booking curves' perspective, K-NN only uses partial curves on the right side of the black vertical line, and then make the forecast for the targeted by taking the average of the ROH_0s of the nearest neighbors.

Figure 4.10: Empirical Study 2: K-NN Illustration



Notes: K-NN model searches for nearest neighbors by the shape of the booking curves after DBA=5. In other words, it only uses partial curves on the right side of the black vertical line, then make forecast for the target by taking the average of the ROH_0 of the nearest neighbors.

This picture is merely an illustration in 2 dimensions. The model is actually established in a 24-dimension space where not only booking curves, but also pricing curves, star rating, review score, and location are considered. Table 4.15 shows the detailed information of the selected observation in interest (black booking curve), the 13 nearest neighbors (solid grey line), and other chosen curves (grey dotted line). Even though Table 4.15 is just an illustrative example only using booking curves, we can find that one unique perspective about K-NN is that it does not limit the searching to similar hotel or days. Instead, it focuses on exploring similar internal patterns within the data, which is valuable when a massive amount of data is available.

Table 4.15: Empirical Study 2: Example of the 13-Nearest Booking Curves by K-NN

ProductName	StayDate	ROH_0	ROH_1	ROH_2	ROH_3	...	ROH_{12}	ROH_{13}	ROH_{14}	group
Rio	2018-02-16	32	28	23	23	...	1	1	0	Target
Caesars Palace	2018-04-06	32	29	25	21	...	2	2	0	Nearest
Treasure Island - TI	2018-01-05	33	29	26	22	...	3	3	1	Nearest
Park MGM Las Vegas	2018-01-26	32	29	25	23	...	3	3	2	Nearest
MGM Grand	2018-04-02	29	26	25	20	...	3	1	0	Nearest
MGM Grand	2018-02-09	31	25	22	21	...	1	0	0	Nearest
Park MGM Las Vegas	2018-02-09	26	22	22	20	...	2	1	1	Nearest
Treasure Island - TI	2018-03-09	30	29	26	24	...	6	6	1	Nearest
Treasure Island - TI	2018-03-29	28	28	26	23	...	4	2	1	Nearest
Park MGM Las Vegas	2018-02-19	26	24	23	22	...	3	3	2	Nearest
MGM Grand	2018-02-16	38	34	28	23	...	1	1	1	Nearest
Rio	2018-03-29	34	26	20	16	...	1	1	0	Nearest
Treasure Island - TI	2018-03-23	23	23	21	20	...	3	1	0	Nearest
Paris Las Vegas	2018-03-08	30	26	18	17	...	1	0	0	Nearest
MGM Grand	2018-01-14	11	9	9	9	...	3	3	1	Normal
Park MGM Las Vegas	2018-01-25	14	13	12	12	...	1	0	0	Normal
Mirage	2018-03-02	18	13	10	9	...	8	7	4	Normal
Rio	2018-03-09	15	10	9	9	...	0	0	0	Normal
Paris Las Vegas	2018-03-09	12	10	9	8	...	5	3	1	Normal
Rio	2018-01-19	15	14	12	9	...	3	2	2	Normal
MGM Grand	2018-01-31	11	9	9	9	...	4	3	2	Normal
Planet Hollywood	2018-02-18	11	10	9	8	...	3	1	1	Normal

4.2.5 Weighted K-NN

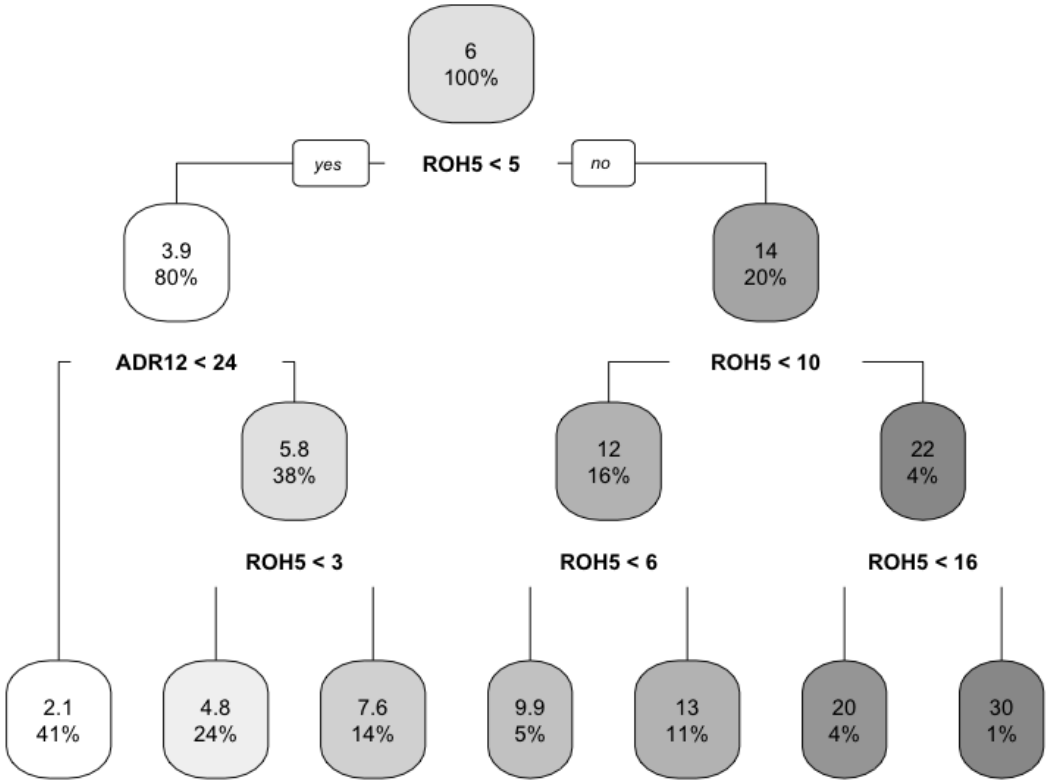
Same as the previous section, the value of k in the Weighted K-NN models is selected through leave-one-out cross-validation. I tested 20 randomly selected k values and test different shapes of kernels (rectangular, triangular, epanechinikov, Gaussian, rank, and optimal) to select the optimal k. The value with the lowest MAE is applied in models building.

In the DBA=5 models, weighted K-NN models use k=12 when building the models since it generates the lowest MAE. The kernel used is an optimal kernel.

4.2.6 Decision Tree

As Figure 4.11 illustrates, ROH plays a significant role in the tree fitting, and no other variable is involved. In other words, this decision tree models only split on different ROH values to conduct final forecast. For instance, if a ROH_i is smaller than 4.5, then it traces left and facing the $ADR_{12} < 24$ split. If the $ADR_{12} < 24$, then the left nodes then reached a bottom node with a forecast of 2.1, and the forecasts with this value consist of 31% of the whole test set.

Figure 4.11: Empirical Study 2: Decision Tree Illustration



Notes: This graph illustrates the decision tree model from a dataset with multiple hotels and information including historical bookings, pricing, location, etc. This model takes DBA=5 as the example. The numbers in rounded rectangles indicate the average value of the responses under each node. The percentage values in rounded rectangles represent the percentage of the observations under each node takes to the whole training set.

Even though ADR_{12} appears in the tree model, it is apparent that ROH_5 still plays a significant role in most of other splits. In other words, this decision tree

model does not take into other parts of the booking curve, pricing curve, and other information into consideration, and a large amount of valuable information is lost.

4.2.7 Random Forest

To find the number of variables randomly sampled as candidates at each split, I apply 10-fold cross-validation and repeat the procedure three times before establishing the random forest models. The number of variables sampled at each split is a vital parameter in models building since it affects final accuracy and is decided empirically by the dataset. In this empirical study, since we only have 4 independent variables (ADR, ROH, review score, location), I use Grid Search approach which examines a vector of candidate values in optimizing models performance. In this case, the tuning process tests the performance of models when selecting 1, 2, 3, 4 variables each time and decide the best parameter by comparing the RMSE. Taking DBA=5 as an example, the optimal parameter is 21 with an RMSE of 2.773.

4.2.8 Support Vector Machine

SVM seeks the non-parametric boundary to divide data points then conduct forecasts accordingly. A range of various shapes of kernels is tested and selected. Taking DBA=5 as the example, the SVM generates 1,377 support vectors and uses

radial kernel which can be express in the equation:

$$\exp(-\gamma|u - v|^2), \tag{4.5}$$

where $\gamma = 0.04$. This small gamma value indicates the kernel used in this models is relatively wide, and therefore the hyperpline is relatively flat.

4.3 Results

The empirical study explores a detailed understanding of the capabilities of machine learning models in hotel demand forecasts. It is also useful to obtain further in-depth information on how and why those models outperform or fail to outperform traditional approaches. This section evaluates the models from three perspectives: bias, accuracy, and variance. Bias is measured by the Mean Error (ME), accuracy is illustrated by the Mean Absolute Error (MAE), and variance is displayed in terms of the Standard Deviation of Error (SDE).

$$ME = \sum_{i=1}^{N} \frac{(x_i - \hat{x}_i)}{N} \tag{4.6}$$

$$MAE = \sum_{i=1}^{N} \frac{|x_i - \hat{x}_i|}{N} \tag{4.7}$$

$$SDE = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(e_i - \bar{e})^2}, \tag{4.8}$$

where $e_i = x_i - \hat{x}_i$, $\bar{e} = \frac{1}{N}e_i$

Evaluation of bias (ME) is fundamental in hotel revenue management since it is essential for hotel dynamic pricing. Ideally, we prefer models with a systematic ME of 0. Printing out ME of each model also helps practitioners to understand whether a model will usually over-estimate or underestimate the demand.

From the statistical performance, models with smaller test errors are preferred since it indicates higher accuracy. MAE here is an indicator of accuracy. SDE describes the variance of the error term. The lower the error variance, the less disperse the errors are, the more stable a model performs.

Good test performance of a forecasting model seeks the balance between bias and variance. When the statistical performances are relatively similar, interpretability is another factor to consider when evaluating the models. This chapter compares the results from the two empirical studies and provides a further discussion on models evaluations.

The following section displays the performances of the nine models: additive pick-up, multiplicative pick-up, regression , neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, random forest, and support vector machine. A robustness test is designed to verify the stability of the models. An

aggregated discussion concludes all the findings in the studies.

4.3.1 Empirical Study 1

Table 4.16 shows the aggregated overall model performance in bias (measured in ME), accuracy (measured in MAE), and variance (measured in SDE) in empirical study 1. Figure 4.12 - 4.14 illustrate how the bias, accuracy, and variance changes separately with DBA increases. The x-axis of Figure 4.12 - 4.14 is the ROH on a certain DBA, and the y-axis refers to the performance criteria. The plots report the results on a summarized basis with DBA's cutoff at day 1, 2, 3, 4, 5, 6, 7, 14, 21, 30, 60, 90, and beyond. This cutoff is applied because this is the common interval used in hotel dynamic pricing.

Table 4.16: Empirical Study 1: Summary Statistics

	Traditional Models			Machine Learning Models					
	apk	mpk	reg	nn	knn	wknn	dtree	rf	svm
ME	-0.46	0.605	-0.041	-13.4	-0.11	-0.431	0.502	-0.021	0.026
MAE	4.87	8.651	5.174	15.71	6.229	6.058	5.94	5.366	3.362
SDE	6.352	11.23	6.543	14.92	7.985	7.706	7.255	6.842	4.293

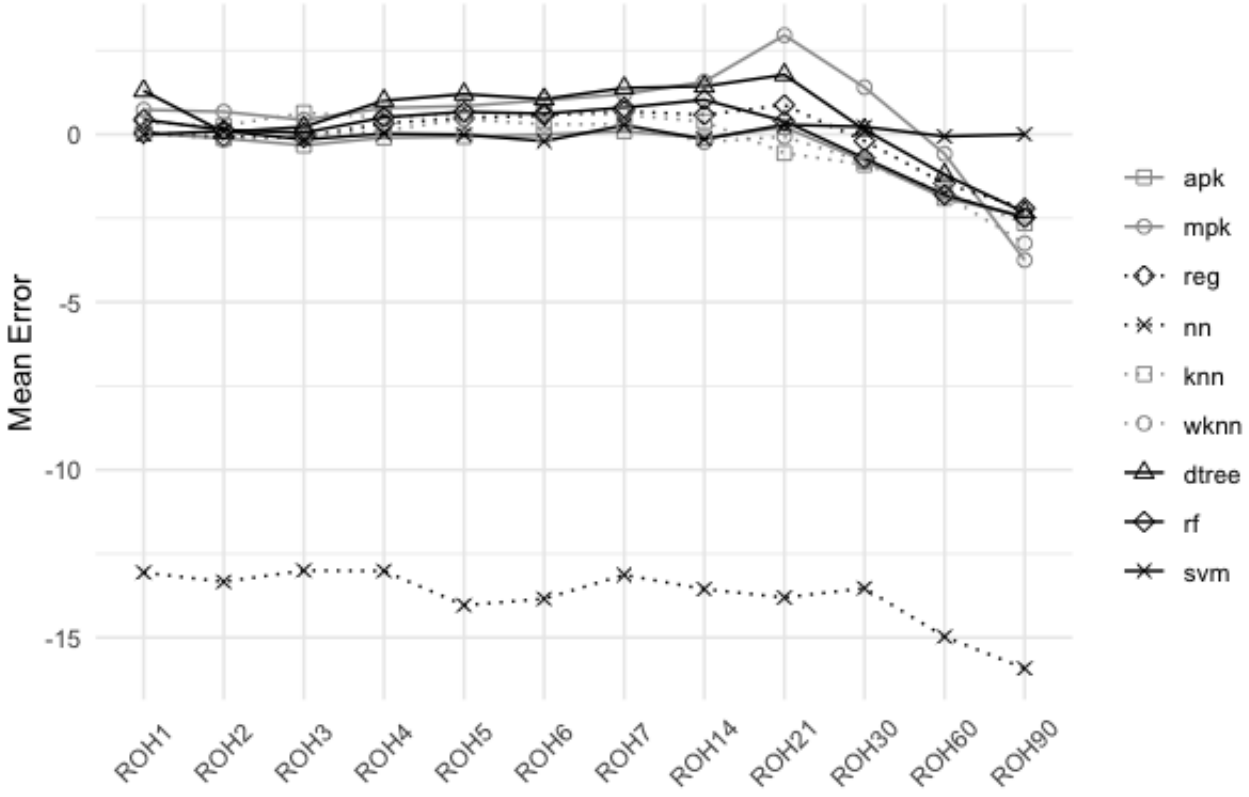
Note:

This table concludes the model performances over the whole booking window, from DBA 1 to 90. The models used in this study is (from left to right): additive pick-up model, multiplicative pick-up model, linear regression, neural network, k-nearest neighbor, weighted k-nearest neighbor, decision tree, random forest, and support vector machine.

In terms of bias, most of the models have an overall bias close to 0. Only the multiplicative pick-up, decision tree, and SVM have positive biases, which indicates those models might tend to over-estimating demand. Random forest and SVM have the lowest overall biases.

Figure 4.12 gives a closer inspection of how the models perform in terms of bias. SVM keeps stable performances with biases close to zero. Except for neural network which consistently underestimates the models with a bias of around -13, all other models have similar prediction power in term of bias.

Figure 4.12: Empirical Study 1: Bias (ME) of Models

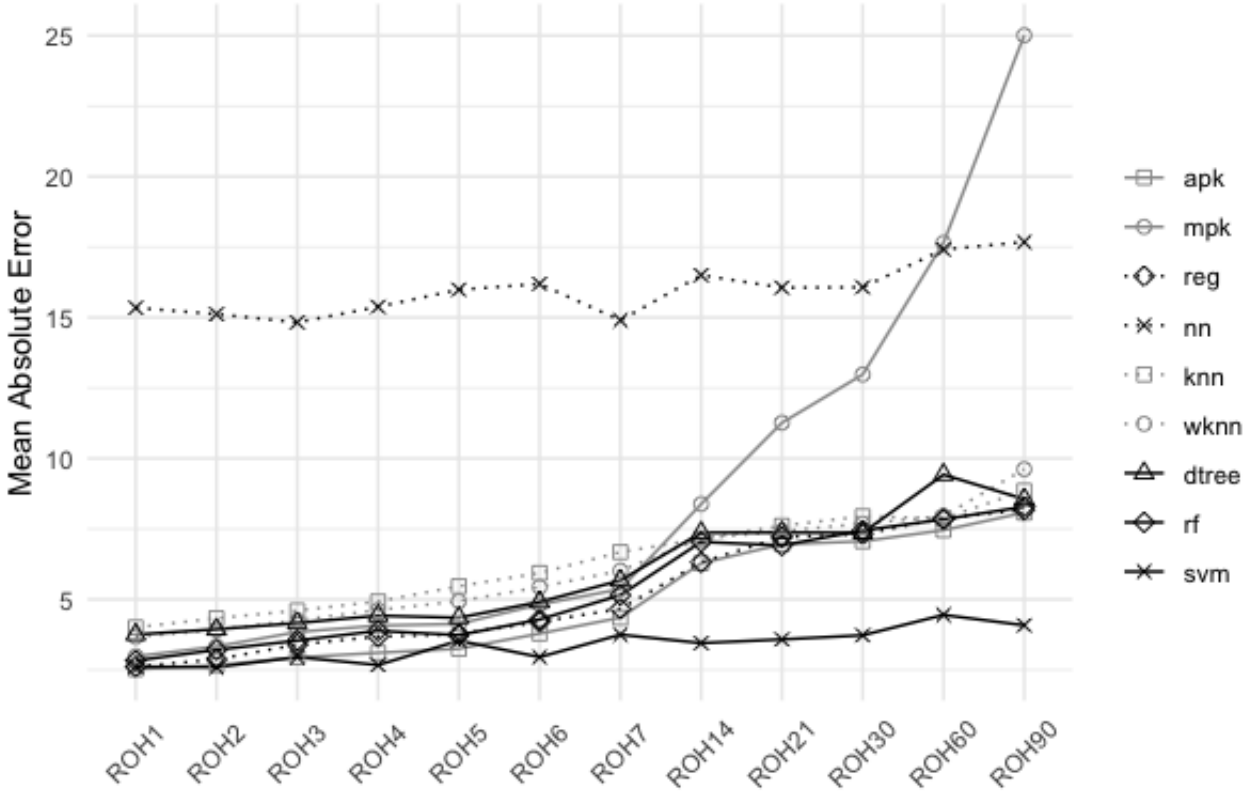


In terms of accuracy, models lose their forecast accuracy with the length of booking window increase. This general trend makes sense in the hospitality industry since the longer the time between now and future, the more challenging to make forecast since less information is given.

SVM stands out with the lowest MAE of 3.362, which outperforms all other models by at least 1.5 number of rooms. Figure 4.12 indicates that SVM does have a lower error rate than the rest of the models, especially when most of the

models have sudden decrease in accuracy after DBA7, SVM still performs stable and keeps its MAE under 5.

Figure 4.13: Empirical Study 1: Accuracy (MAE) of Models



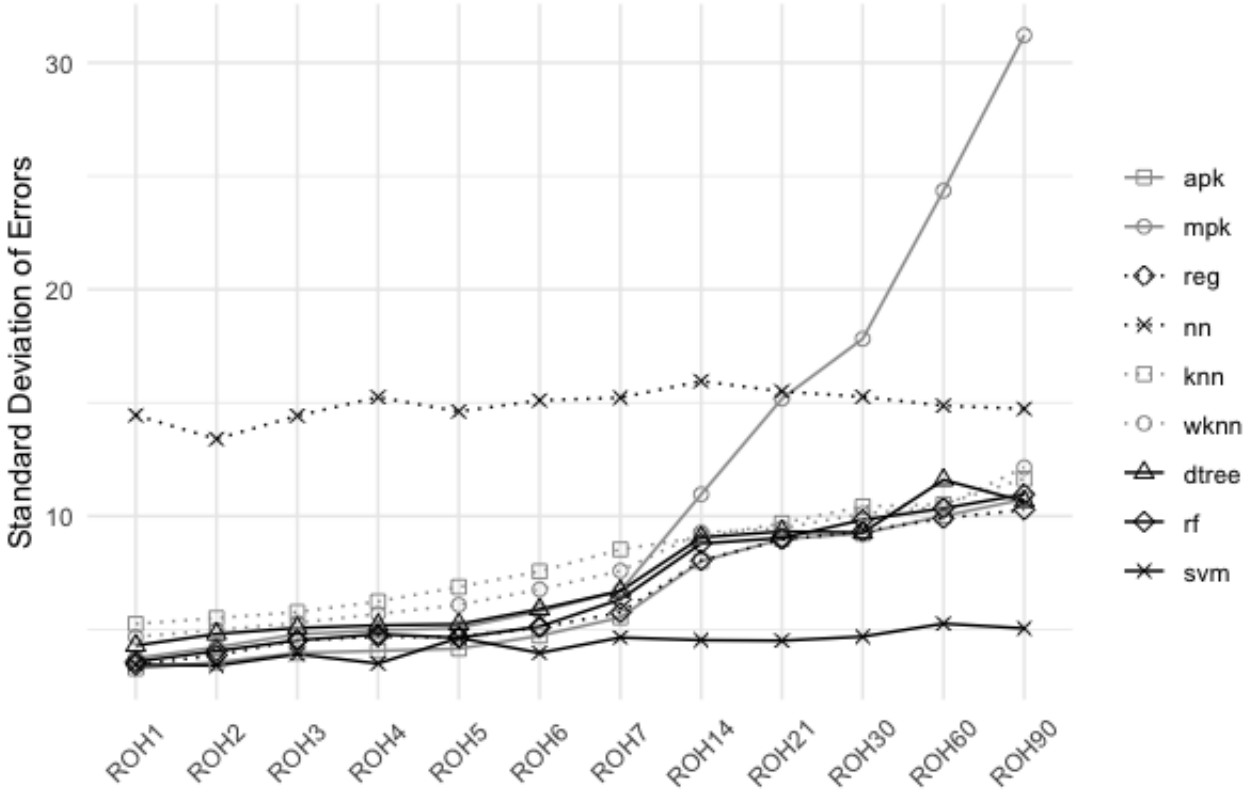
There is a significant jump for multiplicative pick-up models after DBA 7. The MAE of multiplicative pick-up models increases sharply from around 5 at DBA 7 to 8 at DBA 14 and finally reached 25 when DBA equals to 90. It is understandable given multiplicative pick-up models makes forecasts on a ratio basis, and the far-

ther the stay date is, the smaller the average pick-up ratio is, where a small change can generate a huge impact on the final forecast.

Neural network performs poorly in accuracy, with an average MAE above 15. The MAE of other models concentrate at around 5 when DBA is smaller than 7, and gradually increase to the range of 7 to 10 afterward.

To describe the forecast stability of the models, Figure 4.14 plots out the standard deviation of errors of each model. As can be seen from Figure 4.14, SVM has the lowest variance of errors across all DBAs. The SDE of SVM slightly grows from 3 at DBA 1 to 5 at DBA 90. The SDE of pick-up models, regression, nearest neighbor, and tree models grow steadily from around 5 to 10. It is noticeable that those models experience a jump in SDE after DBA 7 while SVM remains steady.

Figure 4.14: Empirical Study 1: Error Variance (SDE) of Models



Due to the large absolute error generated by multiplicative models, the variance of error rockets after DBA 7. Neural network remains a high SDE at around 15 but the variance itself stays table.

To have a better understanding on how SVM outperforms other models, Table A.1, A.2, and A.3 display the results of the error decrease ratio of SVM to other models. As shown in the tables, SVM outperforms other models in various ratio,

and the differences enlarge alongside the days advance. It is noticeable that even though SVM leads the overall bias ratio, the absolute ME of additive pick-up and regression are lower than SVM's on some specific days. In terms the accuracy, SVM has a 31% lower overall MAE than additive pick-up and 35% lower than regression. Similar to error variance, SVM outperforms other models in terms of SDE from 32.4% to 71.2%.

4.3.2 Empirical Study 2

The second empirical study aims to explore the models performance in multiple hotel demand forecasting with shorter booking window. The results in the following section are reported in each DBA from 1 to 14. Table 4.17 displays the summary statistics of the models performances in empirical study 2.

Table 4.17: Empirical Study 2: Summary Statistics

	Traditional Models			Machine Learning Models					
	apk	mpk	reg	nn	knn	wknn	dtree	rf	svm
ME	-0.463	-0.556	-0.341	-3.93	-0.258	-0.178	-0.303	-0.076	-0.68
MAE	3.22	3.78	2.63	4.63	2.62	2.31	2.57	2.12	2.13
SDE	4.42	5.45	3.76	5.75	4.03	3.6	3.86	3.27	3.8

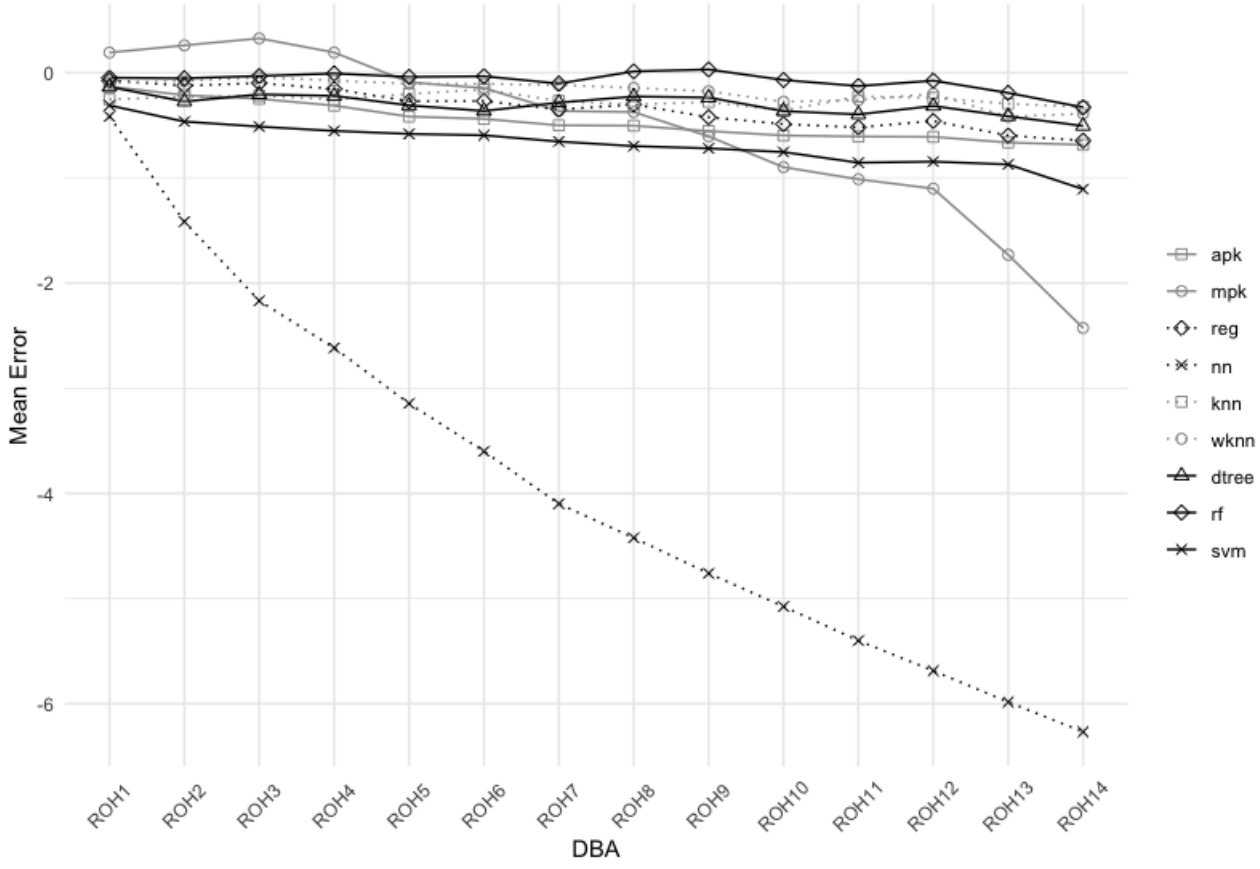
Note:

This table concludes the model performances over the whole booking window, from 1 to 14 days. The models used in this study is (from left to right): additive pick-up model, multiplicative pick-up model, linear regression, neural network, k-nearest neighbor, weighted k-nearest neighbor, decision tree, random forest, and support vector machine.

In terms of bias, it is interesting to find that all of the models have overall negative biases, which means they underestimate the ROH_0. Among those, Random forest has the smallest overall bias of -0.076, followed by the weighted K-NN with a bias of -0.178, and K-NN with a bias -0.258. Neural Network and multiplicative pick-up perform relatively weak in this study, with average biases of -3.93 and -1.67 respectively. All other models keep an overall bias withing (-1, 0).

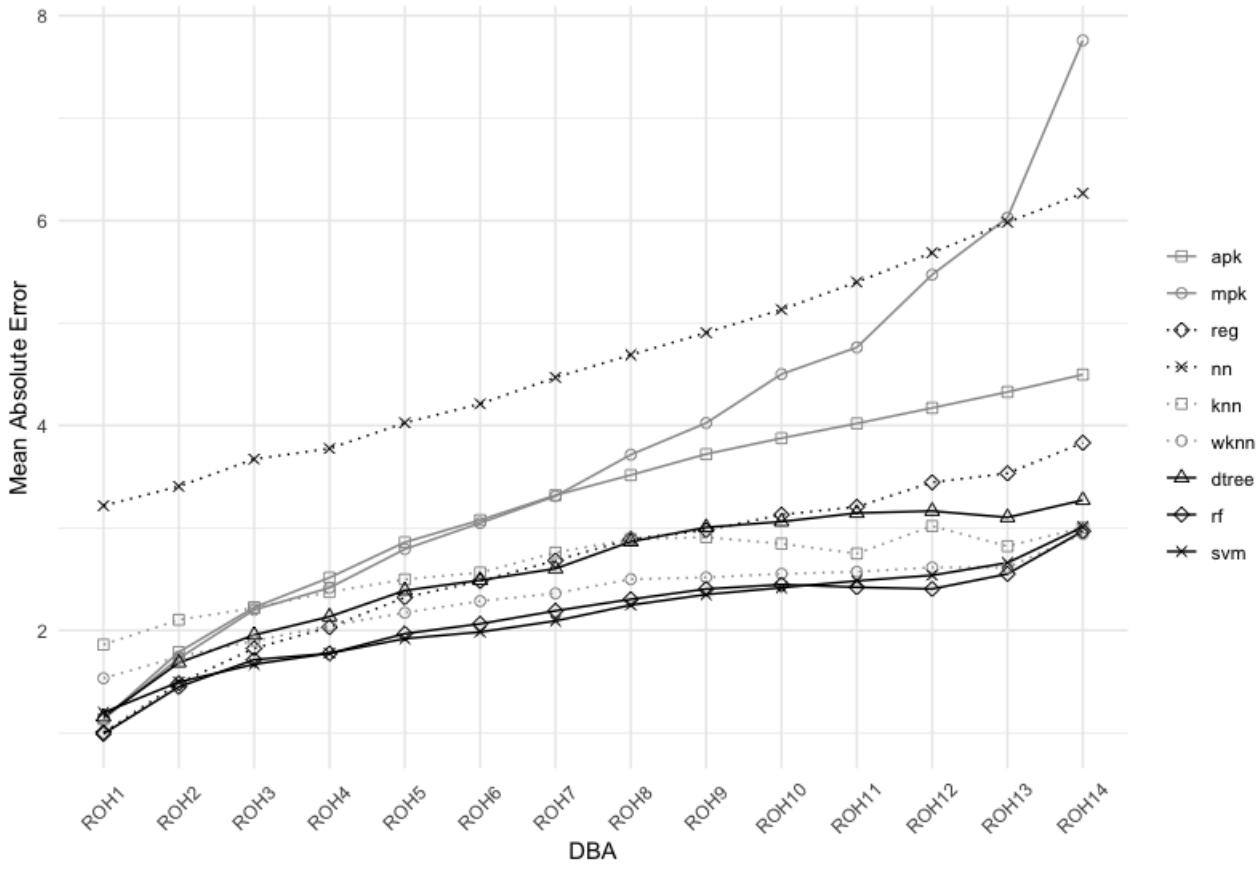
Figure 4.15 shows that all models generally have downward trends when DBA increases. Random forest remains a flat trend with biases close to 0 when DBA changes. Neural network performs badly with an accelerating negative bias when DBA moves further.

Figure 4.15: Empirical Study 2: Bias (ME) of Models



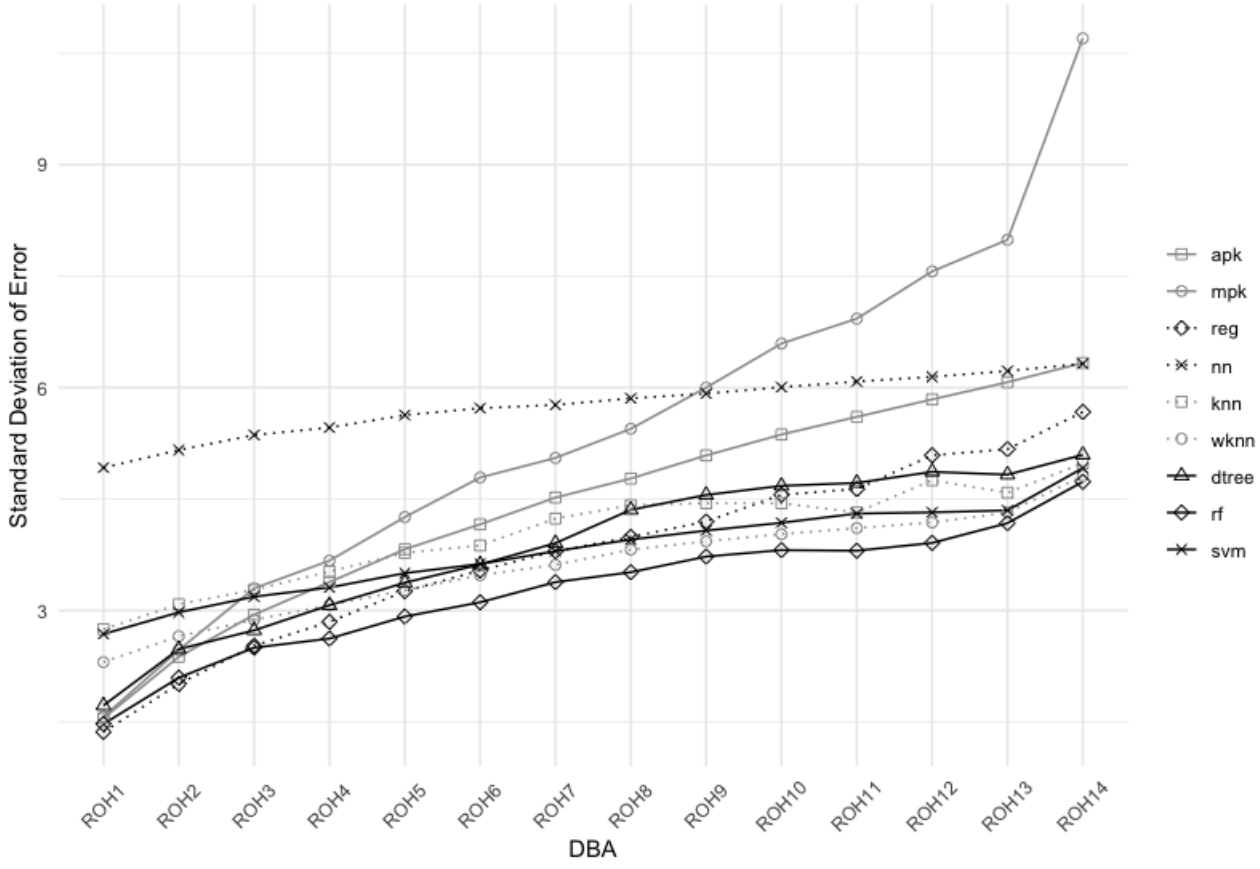
In terms of accuracy, Figure 4.16 shows that the MAE of models increases with DBA moves further. Random forest, SVM, and weighted K-NN outperform other models in accuracy. Neural network and pick-up models perform relatively bad in terms of accuracy.

Figure 4.16: Empirical Study 2: Accuracy (MAE) of Models



Random forest has the lowest error variance across the DBA. Regression has low error variance within the first 7 DBAs, but its performance deteriorates afterwards. Weighted K-NN and SVM show higher stability in error variance after DBA=7.

Figure 4.17: Empirical Study 2: Error variance (SDE) of Models



To have a more direct sense of how random forest outperforms other models, I compare its ME, MAE, and SDE with other models and generate Table A.6, A.7, and A.8. In terms of bias, random forest has over 50% lower ME (absolute value) than other models in most scenario. From the perspective of accuracy, random forest has a 34.2% lower absolute MAE than additive pick-up, and 19.5% lower MAE than regression. Random forest slightly outperforms SVM with less than

1% in MAE. In terms of error variance, random forest has around 13.1% lower SDE than regression, and 26% lower SDE than additive pick-up.

4.3.3 Robustness Test

To test the stability and robustness of the models, I design a follow-up test which divides the training and test set in a different seed.

All the findings from this robustness test are in line with the previous studies on the original training-test sets. As Table 4.18 shows, SVM dominates the empirical study 1 in all metrics. In empirical study 2, Random forecast outperforms all other models, and weighted K-NN also has a close performance. More detailed information on how performance metrics changes with DBA for each model can be found in Figure A.1 and Figure A.2.

The models on the new split data do not perform significantly different from the original hold-out, which proves that all the models have a certain level of robustness and stability.

Table 4.18: Robustness test: Summary Statistics

		apk	mpk	reg	nn	knn	wknn	dtree	rf	svm
	ME	0.366	1.22	0.2	-11.6	0.136	-0.112	0.042	-0.043	-0.009
Empirical Study 1	MAE	5.72	9.09	6.04	17.5	7.49	7.19	6.42	5.91	3.83
	SDE	7.44	12	7.61	18.5	9.58	9.17	8.04	7.48	5.03
	ME	-0.877	-1.67	-0.341	-3.95	-0.3	-0.178	-0.303	-0.076	-0.68
Empirical Study 2	MAE	3.19	3.39	2.63	4.64	2.6	2.31	2.57	2.12	2.13
	SDE	4.42	4.49	3.76	5.75	4.02	3.6	3.86	3.27	3.8

4.3.4 Discussion

Given pure historical booking information and long booking window, SVM outperforms all other models in either bias, accuracy, or error variance. This outstanding performance is also verified in the robustness test. By generating support vectors to establish a virtual hyperplane, SVM efficiently reduces the dimensionality of the problem, and accurately makes forecast based on historical bookings. In the hospitality industry, when a hotel has long historical reservation windows, SVM's ability to deal with high-dimension data is extremely helpful.

SVM's advantage in accommodating high-dimension data is proven by the results between pick-up based models and SVM. Pick-up models and the regression model used in this research only take the newest information into consideration. Therefore, valuable information, such as the booking patterns reflected from the

booking curves, is neglected.

SVM's superior predicting power may be attributed to its kernel function. The kernel function can capture the non-linear patterns reflected in the booking curves. The relationship between historical bookings and future arrivals are usually non-linear due to many internal and external factors. SVM can adopt different shapes of kernels to accommodate this non-linearity.

The advantage of kernels is also revealed when controlling the specific algorithm. For instance, weighted K-NN outperforms K-NN in most cases. The difference between the two models is weighted K-NN applies different shapes of kernels to calculate the distance, while K-NN uses Euclidean distance which is comparable to straight line distance. By adopting a range of kernels, the weighted K-NN has the potential to find similar patterns by putting more weight to the more similar observations, which gives more flexibility to model fit.

Another outperforming model in the empirical study is random forest. Given hotel property information and short historical booking information (including booking curves and pricing curves), random forest leads the performance in either bias, accuracy, or error variance. One possible explanation for random forest's superiority is its bagging function.

Bagging is a unique approach widely used in machine learning to improve model performance. Tree models automatically select the split which decreases chaos of the data the most, but this spit is usually the strongest predictor in the

model. In the practical case, decision tree always uses the newest ROH to establish the model, which results in ignoring other historical booking information. In this case, decision tree is comparable to pick-up models and regressions since only the newest ROH is used. Random forest solves this problem by forcing the model to consider other "not-that-important" ROHs, and ensures other parts of the booking curve is considered as well.

It is noticeable that neural network has a poor performance in this study, and a possible explanation might derive from the parameter selection. Even though neural network has the advantages of indicating weights between the input and output directly, it requires strong statistical background or a decent amount of computational resources to select an appropriate parameter. For instance, there is no universal rule to select the number of hidden layers and the number of neurons in a hidden layer. Repetitive cross-validation test is one way to find the parameter which performs better, but it is a computationally intensive procedure to achieve. In both of the studies, I use some general rules to decide the size of the hidden layer, but the results appear to be poor.

Although the statistical performance of SVM and random forest are superior to other models, it is important to bear in mind that machine learning models are hard to interpret and are complicated to build. Little information can be derived from the fitted machine learning model due to the high complexity and non-parametric patterns. In comparison, practitioners are able to extract incremental value over the booking window and specific coefficient weights from pick-

up based and regression models. Additionally, facing large datasets, machine learning models tend to take times larger computational time than pick-up based models. Practitioners need to balance the trade-off between accuracy and interpretability considering the specific situation and make decisions accordingly.

CHAPTER 5

CONCLUSION

This study is the first academic research which explores the application of machine learning approaches in hotel demand forecasting. The emerging amount of hotel transaction data lays a strong foundation for applying machine learning methods in the hotel industry. The non-parametric relationship between booking curves and final demand is also a problem machine learning approaches are good at to solve.

The empirical studies test a range of models and compare their performances using real hotel transactions. The result of this study reveals that machine learning has the potential to outperform classical forecasting methods, especially when given long booking windows. Both additive and multiplicative pick-up models only use one single point from the booking curve. Therefore, historical booking patterns cannot be included in the modeling, and valuable information is lost.

Machine learning models, instead, can incorporate information from the whole booking curve since the models have the ability to accommodate high-dimension data. Even with different activation functions, machine learning approaches also share a common advantage of being able to capture non-parametric patterns. These unique properties can be helpful in modeling the booking patterns and make accurate forecasts.

Particularly, Support Vector Machine and Random Forest are two outstanding

models in the empirical results. Support Vector Machine's kernel function has the ability to generate a highly flexible hyperplane to classify the booking curves and make forecasts. Random forest's bagging function requires all parts of the booking curves are considered. Those characteristics lead them outperforming other models in the practical test, and indicate strong potentials for further use.

However, machine learning models need to be applied with caution. Practitioners need to take into account the computational time and complexity of machine learning models. To choose the suitable model for practical use, revenue managers need to further compare the accuracy improvement brought by machine learning models and the cost of interpretability and computational power.

In summary, The empirical results confirm the potential of introducing machine learning techniques in hotel demand prediction. Furthermore, this study provides new insights into hotel revenue management in both academia and industry. Even though machine learning models have the drawback in lacking interpretability, their performances in accuracy and stability can help hotel managers better forecast the demand and optimize hotel revenue. For researchers, this present study explores, for the first time, the use of machine learning in hotel demand forecast, and proves the capability of this new approach. This research provides a valuable opportunity to advance quantitative research in hotel revenue management and is hoped to serve as a starting point for further study.

Due to practical constraints, this research cannot provide a comprehensive investigation under situations with longer (multiple years) time frame. It is also be-

yond the scope of this study to accommodate other important factors in hotel revenue management, such as the external events, seasonality, cancellation, etc. This research serves as an initial attempt in introducing machine learning approaches in hotel demand forecast, and leaves expansive space for further exploration.

APPENDIX A

APPENDIX OF CHAPTER 4

Table A.1: Empirical Study 1: Comparison of Model Performances in Bias (Using Support Vector Machine as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_{14}	ROH_{21}	ROH_{30}	ROH_{60}	ROH_{90}	Overall
apk	-0.598	-0.186	-0.526	-0.816	-0.867	27.683	1.918	0.291	0.331	-0.718	-0.968	-0.999	-0.943
mpk	-0.984	-0.854	-0.638	-0.977	-0.989	-0.800	-0.774	-0.910	-0.903	-0.841	-0.894	-1.000	-0.957
reg	-0.691	0.292	0.778	-0.944	-0.980	-0.640	-0.608	-0.756	-0.672	0.120	-0.958	-0.999	-0.358
nn	-0.999	-0.993	-0.988	-0.999	-0.999	-0.984	-0.979	-0.990	-0.978	-0.982	-0.996	-1.000	-0.998
knn	-0.963	-0.850	-0.687	-0.976	-0.987	-0.788	-0.488	-0.468	-0.521	-0.779	-0.961	-0.999	-0.740
wknn	-0.831	-0.385	1.948	-0.827	-0.979	-0.318	-0.104	-0.391	3.165	-0.728	-0.966	-0.999	-0.940
dtree	-0.991	0.561	-0.302	-0.982	-0.992	-0.807	-0.805	-0.901	-0.838	0.651	-0.949	-0.999	-0.948
rf	-0.973	0.101	1.078	-0.965	-0.983	-0.701	-0.657	-0.861	-0.604	-0.712	-0.966	-0.999	0.260
svm	0	0	0	0	0	0	0	0	0	0	0	0	0

Notes:
This table compares the mean error (ME) among models over different booking window stage. The results are from the empirical study using one single hotel with long booking windows. The benchmark is the ME of Support Vector Machine. The results are calculated by

$$\frac{|ME_{SVM}| - |ME_i|}{|ME_i|}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall ME (absolute value) comparison between SVM and other models.

Table A.2: Empirical Study 1: Comparison of Model Performances in Accuracy (Using Support Vector Machine as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_{14}	ROH_{21}	ROH_{30}	ROH_{60}	ROH_{90}	Overall
apk	0.036	-0.025	0.004	-0.139	0.088	-0.220	-0.139	-0.451	-0.485	-0.472	-0.403	-0.495	-0.310
mpk	-0.122	-0.221	-0.232	-0.344	-0.144	-0.387	-0.301	-0.589	-0.682	-0.713	-0.748	-0.837	-0.611
reg	0.001	-0.096	-0.131	-0.274	-0.060	-0.298	-0.197	-0.454	-0.504	-0.491	-0.435	-0.501	-0.350
nn	-0.831	-0.831	-0.798	-0.829	-0.771	-0.804	-0.749	-0.798	-0.767	-0.754	-0.740	-0.765	-0.786
knn	-0.368	-0.407	-0.389	-0.465	-0.369	-0.502	-0.454	-0.534	-0.521	-0.555	-0.432	-0.532	-0.473
wknn	-0.288	-0.338	-0.307	-0.422	-0.284	-0.457	-0.377	-0.518	-0.518	-0.515	-0.441	-0.576	-0.445
dtree	-0.303	-0.342	-0.290	-0.396	-0.187	-0.399	-0.339	-0.532	-0.514	-0.494	-0.528	-0.523	-0.434
rf	-0.085	-0.171	-0.173	-0.303	-0.058	-0.306	-0.273	-0.516	-0.503	-0.499	-0.427	-0.509	-0.373
svm	0	0	0	0	0	0	0	0	0	0	0	0	0

Notes:
This table compares the mean absolute error (MAE) among models over different booking window stage. The results are from the empirical study using one single hotel with long booking windows. The benchmark is the MAE of Support Vector Machine. The results are calculated by

$$\frac{MAE_{SVM} - MAE_i}{MAE_i}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall MAE (absolute value) comparison between SVM and other models.

Table A.3: Empirical Study 1: Comparison of Model Performances in Error Variance (Using Support Vector Machine as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_{14}	ROH_{21}	ROH_{30}	ROH_{60}	ROH_{90}	Overall
apk	0.051	-0.039	-0.015	-0.137	0.114	-0.160	-0.160	-0.434	-0.499	-0.493	-0.475	-0.530	-0.324
mpk	-0.076	-0.191	-0.184	-0.294	-0.087	-0.316	-0.307	-0.587	-0.703	-0.737	-0.784	-0.838	-0.618
reg	0.001	-0.117	-0.137	-0.253	-0.008	-0.220	-0.192	-0.437	-0.497	-0.497	-0.469	-0.509	-0.344
nn	-0.761	-0.744	-0.734	-0.765	-0.675	-0.736	-0.700	-0.718	-0.714	-0.695	-0.642	-0.665	-0.712
knn	-0.348	-0.384	-0.359	-0.442	-0.349	-0.485	-0.469	-0.508	-0.528	-0.562	-0.498	-0.562	-0.475
wknn	-0.262	-0.315	-0.262	-0.384	-0.239	-0.414	-0.388	-0.512	-0.524	-0.536	-0.496	-0.584	-0.443
dtree	-0.196	-0.290	-0.229	-0.325	-0.116	-0.327	-0.307	-0.500	-0.517	-0.495	-0.547	-0.526	-0.408
rf	-0.038	-0.147	-0.141	-0.260	-0.007	-0.220	-0.269	-0.488	-0.520	-0.520	-0.485	-0.540	-0.372
svm	0	0	0	0	0	0	0	0	0	0	0	0	0

Notes:
This table compares the error variance among models over different booking window stage. The criteria used is the standard deviation of errors (SDE) of the models. The results are from the empirical study using one single hotel with long booking windows. The benchmark is the SDE of Support Vector Machine. The results are calculated by

$$\frac{SDE_{SVM} - SDE_i}{SDE_i}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall SDE (absolute value) comparison between SVM and other models.

Table A.4: Empirical Study 2: Regression using Booking Curves and Pricing Curves

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
(Intercept)	1.09	1.97	2.62*	2.94*	5.65***	6.39***	8.42***	9.14***	10.89***	12.85***	14.16***	14.54***	15.54***	17.25***
	(0.68)	(1.02)	(1.25)	(1.42)	(1.59)	(1.71)	(1.87)	(1.98)	(2.12)	(2.22)	(2.30)	(2.46)	(2.54)	(2.73)
ROH_1	1.33***													
	(0.03)													
ROH_2	−0.27***	1.57***												
	(0.05)	(0.05)												
ROH_3	0.05	−0.32***	1.70***											
	(0.06)	(0.09)	(0.08)											
ROH_4	−0.06	0.03	−0.12	1.86***										
	(0.06)	(0.10)	(0.12)	(0.09)										
ROH_5	−0.01	−0.21*	−0.49***	−0.52***	1.71***									
	(0.07)	(0.10)	(0.13)	(0.14)	(0.11)									
ROH_6	−0.01	0.04	0.03	0.09	0.07	2.15***								
	(0.07)	(0.11)	(0.13)	(0.15)	(0.17)	(0.12)								
ROH_7	0.09	0.24*	0.26	−0.03	−0.36*	−0.52**	1.92***							
	(0.08)	(0.12)	(0.14)	(0.16)	(0.18)	(0.20)	(0.15)							
ROH_8	0.01	−0.01	−0.07	0.01	0.10	0.03	0.07	2.40***						
	(0.09)	(0.13)	(0.16)	(0.18)	(0.21)	(0.22)	(0.24)	(0.16)						
ROH_9	−0.03	−0.26	−0.11	−0.14	−0.19	−0.19	−0.53*	−0.77**	2.15***					
	(0.09)	(0.14)	(0.17)	(0.19)	(0.22)	(0.24)	(0.26)	(0.27)	(0.20)					
ROH_{10}	0.04	0.08	0.08	0.15	0.03	−0.15	0.08	0.06	−0.43	1.90***				

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
	(0.09)	(0.14)	(0.17)	(0.20)	(0.22)	(0.24)	(0.26)	(0.28)	(0.30)	(0.20)				
ROH_{11}	−0.11	−0.00	−0.04	−0.10	−0.06	−0.12	−0.06	−0.27	0.10	0.16	2.51***			
	(0.10)	(0.15)	(0.18)	(0.20)	(0.23)	(0.25)	(0.27)	(0.28)	(0.30)	(0.32)	(0.20)			
ROH_{12}	−0.07	−0.20	−0.27	−0.15	−0.21	−0.04	−0.22	−0.35	−0.86**	−1.17***	−1.56***	1.20***		
	(0.10)	(0.15)	(0.18)	(0.20)	(0.23)	(0.25)	(0.27)	(0.29)	(0.30)	(0.32)	(0.33)	(0.23)		
ROH_{13}	−0.06	−0.04	0.04	0.07	0.28	0.32	0.35	0.78*	1.05**	1.26***	1.23***	1.09**	2.17***	
	(0.11)	(0.16)	(0.19)	(0.22)	(0.25)	(0.27)	(0.29)	(0.31)	(0.33)	(0.35)	(0.36)	(0.38)	(0.27)	
ROH_{14}	0.11	−0.03	−0.07	−0.38	−0.60*	−0.73*	−0.85**	−1.10***	−1.33***	−1.43***	−1.56***	−1.59***	−1.51***	0.63*
	(0.11)	(0.17)	(0.21)	(0.24)	(0.27)	(0.29)	(0.31)	(0.33)	(0.36)	(0.37)	(0.39)	(0.41)	(0.43)	(0.26)
ADR_1	−0.00**													
	(0.00)													
ADR_2	0.00	−0.00												
	(0.00)	(0.00)												
ADR_3	0.00	0.00	0.00											
	(0.00)	(0.00)	(0.00)											
ADR_4	0.00	0.00	−0.00	0.00										
	(0.00)	(0.00)	(0.00)	(0.00)										
ADR_5	−0.00	0.00	0.00	0.00	0.00									
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)									
ADR_6	0.00	0.00	0.00	0.00	−0.00	−0.00								
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)								
ADR_7	0.00	−0.00	−0.00	0.00	0.00	0.00	0.00*							
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)							
ADR_8	−0.00	−0.00	−0.00	−0.00	0.00	0.00	0.00	−0.00						

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)						
ADR_9	−0.00	−0.00	−0.00	0.00	0.00	0.00	0.00	0.00	0.00					
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)					
ADR_{10}	−0.00	−0.00	−0.00	−0.00	−0.00	0.00	−0.00	0.00	0.00	0.00				
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)				
ADR_{11}	0.00	0.00	−0.00	0.00	0.00	0.00	0.00	0.01*	0.00	0.01*	0.01*			
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)			
ADR_{12}	−0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01*	0.01***	0.01***	0.02***	0.02***		
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)		
ADR_{13}	0.00	0.00	0.00	0.00	0.00	0.00*	0.00	0.00	0.00	0.00	0.01**	0.01***	0.02***	
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	
ADR_{14}	−0.00	0.00	0.00	0.00*	0.01*	0.01*	0.01*	0.01***	0.01***	0.01***	0.02***	0.02***	0.03***	0.04***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
review_score	−0.18	−0.34	−0.46	−0.54	−1.17**	−1.32***	−1.81***	−1.94***	−2.32***	−2.78***	−3.07***	−3.12***	−3.28***	−3.46***
	(0.16)	(0.24)	(0.29)	(0.33)	(0.37)	(0.40)	(0.43)	(0.46)	(0.49)	(0.52)	(0.54)	(0.57)	(0.59)	(0.64)
location	−0.03*	−0.06**	−0.07**	−0.08**	−0.07*	−0.09**	−0.09*	−0.10**	−0.10*	−0.10*	−0.09*	−0.13**	−0.15**	−0.23***
	(0.01)	(0.02)	(0.02)	(0.03)	(0.03)	(0.03)	(0.04)	(0.04)	(0.04)	(0.04)	(0.04)	(0.05)	(0.05)	(0.05)
star 4.5	0.21*	0.48***	0.56***	0.75***	0.76***	0.85***	1.01***	1.10***	1.11***	1.09***	1.11***	1.41***	1.52***	1.56***
	(0.09)	(0.14)	(0.17)	(0.19)	(0.21)	(0.23)	(0.25)	(0.27)	(0.29)	(0.30)	(0.31)	(0.33)	(0.34)	(0.37)
R^2	0.96	0.90	0.85	0.80	0.75	0.71	0.66	0.61	0.55	0.51	0.47	0.40	0.35	0.25
Adj. R^2	0.95	0.90	0.85	0.80	0.75	0.71	0.65	0.61	0.55	0.50	0.47	0.39	0.35	0.25
RMSE	1.24	1.86	2.28	2.59	2.91	3.14	3.43	3.64	3.90	4.10	4.26	4.54	4.70	5.06

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table A.5: Empirical Study 2: Regression using the Newest ROH and Price

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
(Intercept)	0.66	1.17	2.41*	3.39*	5.69***	6.52***	8.72***	9.53***	10.72***	11.37***	13.08***	13.59***	14.46***	16.73***
	(0.63)	(0.95)	(1.15)	(1.32)	(1.48)	(1.61)	(1.74)	(1.87)	(2.00)	(2.11)	(2.23)	(2.30)	(2.37)	(2.49)
ROH1	1.12***													
	(0.01)													
ADR1	0.00													
	(0.00)													
review score	−0.02	−0.03	−0.27	−0.45	−0.95*	−1.09**	−1.59***	−1.68***	−1.89***	−1.99***	−2.28***	−2.38***	−2.51***	−2.96***
	(0.16)	(0.24)	(0.29)	(0.33)	(0.37)	(0.41)	(0.44)	(0.47)	(0.51)	(0.53)	(0.56)	(0.58)	(0.60)	(0.63)
location	−0.04**	−0.07***	−0.09***	−0.10***	−0.10**	−0.12***	−0.12**	−0.15***	−0.16***	−0.17***	−0.20***	−0.22***	−0.22***	−0.25***
	(0.01)	(0.02)	(0.02)	(0.03)	(0.03)	(0.03)	(0.04)	(0.04)	(0.04)	(0.04)	(0.05)	(0.05)	(0.05)	(0.05)
star 4.5	−0.24*	−0.55***	−0.66***	−0.83***	−0.75***	−0.86***	−0.98***	−1.24***	−1.23***	−1.22***	−1.27***	−1.52***	−1.80***	−1.76***
	(0.09)	(0.14)	(0.17)	(0.20)	(0.22)	(0.24)	(0.26)	(0.28)	(0.30)	(0.32)	(0.33)	(0.35)	(0.36)	(0.37)
ROH2		1.22***												
		(0.01)												
ADR2		0.00***												
		(0.00)												
ROH3			1.29***											
			(0.01)											
ADR3			0.01***											
			(0.00)											
ROH4				1.37***										
				(0.02)										

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
ADR4				0.01***										
				(0.00)										
ROH5					1.47***									
					(0.02)									
ADR5					0.01***									
					(0.00)									
ROH6						1.55***								
						(0.03)								
ADR6						0.01***								
						(0.00)								
ROH7							1.60***							
							(0.03)							
ADR7							0.02***							
							(0.00)							
ROH8								1.68***						
								(0.04)						
ADR8								0.02***						
								(0.00)						
ROH9									1.76***					
									(0.05)					
ADR9									0.02***					
									(0.00)					
ROH10										1.87***				
										(0.06)				

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10	Model 11	Model 12	Model 13	Model 14
ADR10										0.02***				
										(0.00)				
ROH11											2.04***			
											(0.08)			
ADR11											0.02***			
											(0.00)			
ROH12												1.80***		
												(0.10)		
ADR12												0.03***		
												(0.00)		
ROH13													1.89***	
													(0.14)	
ADR13													0.03***	
													(0.00)	
ROH14														0.92***
														(0.25)
ADR14														0.04***
														(0.00)
R^2	0.95	0.90	0.85	0.80	0.74	0.70	0.65	0.59	0.53	0.48	0.41	0.37	0.34	0.27
Adj. R^2	0.95	0.89	0.85	0.80	0.74	0.70	0.64	0.59	0.53	0.48	0.41	0.37	0.34	0.27
RMSE	1.26	1.90	2.30	2.65	2.97	3.23	3.50	3.77	4.04	4.24	4.49	4.64	4.78	5.02

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table A.6: Empirical Study 2: Comparison of Model Performances in Bias (Using Random Forest as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_8	ROH_9	ROH_{10}	ROH_{11}	ROH_{12}	ROH_{13}	ROH_{14}	Overall
apk	-0.637	-0.750	-0.875	-0.977	-0.905	-0.923	-0.795	-0.975	-0.945	-0.884	-0.791	-0.876	-0.712	-0.519	-0.836
mpk	-0.752	-0.795	-0.906	-0.963	-0.565	-0.770	-0.717	-0.966	-0.949	-0.923	-0.875	-0.932	-0.889	-0.864	-0.863
reg	-0.353	-0.568	-0.686	-0.954	-0.854	-0.875	-0.706	-0.958	-0.927	-0.859	-0.756	-0.836	-0.679	-0.490	-0.777
nn	-0.885	-0.962	-0.986	-0.997	-0.987	-0.991	-0.975	-0.997	-0.994	-0.986	-0.977	-0.987	-0.968	-0.947	-0.981
knn	-0.807	-0.775	-0.862	-0.975	-0.844	-0.873	-0.710	-0.957	-0.916	-0.808	-0.590	-0.773	-0.428	-0.141	-0.750
wknn	-0.518	-0.322	-0.310	-0.903	-0.626	-0.683	-0.118	-0.911	-0.826	-0.749	-0.512	-0.624	-0.542	-0.154	-0.573
dtree	-0.645	-0.805	-0.850	-0.968	-0.872	-0.908	-0.638	-0.943	-0.869	-0.811	-0.680	-0.762	-0.538	-0.349	-0.750
rf	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
svm	-0.847	-0.885	-0.940	-0.987	-0.932	-0.944	-0.843	-0.982	-0.957	-0.908	-0.852	-0.911	-0.780	-0.703	-0.888

Notes:

This table compares the mean error (ME) among models over different booking window stage. The results are from the empirical study using multiple hotels with up to 14 days historical bookings, and information including pricing, location, star rating, and client review score. The benchmark is the ME of Random Forest. The results are calculated by

$$\frac{|ME_{RF}| - |ME_i|}{|ME_i|}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall ME (absolute value) comparison between SVM and other models.

Table A.7: Empirical Study 2: Comparison of Model Performances in Accuracy (Using Random Forest as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_8	ROH_9	ROH_{10}	ROH_{11}	ROH_{12}	ROH_{13}	ROH_{14}	Overall
apk	-0.126	-0.190	-0.230	-0.294	-0.312	-0.329	-0.340	-0.345	-0.354	-0.369	-0.398	-0.424	-0.410	-0.340	-0.342
mpk	-0.126	-0.161	-0.221	-0.265	-0.295	-0.323	-0.338	-0.380	-0.403	-0.456	-0.492	-0.561	-0.577	-0.617	-0.439
reg	-0.013	-0.024	-0.060	-0.126	-0.152	-0.168	-0.183	-0.203	-0.193	-0.218	-0.245	-0.302	-0.278	-0.225	-0.195
nn	-0.692	-0.575	-0.533	-0.530	-0.511	-0.510	-0.510	-0.509	-0.510	-0.523	-0.552	-0.577	-0.574	-0.526	-0.543
knn	-0.472	-0.304	-0.233	-0.244	-0.214	-0.207	-0.191	-0.185	-0.159	-0.140	-0.125	-0.182	-0.093	-0.008	-0.185
wknn	-0.354	-0.169	-0.097	-0.132	-0.094	-0.097	-0.072	-0.079	-0.045	-0.041	-0.059	-0.080	-0.024	0.009	-0.083
dtree	-0.143	-0.140	-0.123	-0.168	-0.176	-0.171	-0.158	-0.196	-0.200	-0.201	-0.230	-0.240	-0.178	-0.092	-0.177
rf	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
svm	-0.173	-0.032	0.026	-0.001	0.025	0.040	0.047	0.025	0.023	0.012	-0.025	-0.052	-0.041	-0.013	-0.006

Notes:

This table compares the mean absolute error (MAE) among models over different booking window stage. The results are from the empirical study using multiple hotels with up to 14 days historical bookings, and information including pricing, location, star rating, and client review score. The benchmark is the MAE of Random Forest. The results are calculated by

$$\frac{MAE_{RF} - MAE_i}{MAE_i}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall MAE (absolute value) comparison between SVM and other models.

Table A.8: Empirical Study 2: Comparison of Model Performances in Error Variance (Using Random Forest as the Benchmark)

	ROH_1	ROH_2	ROH_3	ROH_4	ROH_5	ROH_6	ROH_7	ROH_8	ROH_9	ROH_{10}	ROH_{11}	ROH_{12}	ROH_{13}	ROH_{14}	Overall
apk	-0.049	-0.118	-0.151	-0.224	-0.237	-0.252	-0.252	-0.264	-0.268	-0.290	-0.321	-0.331	-0.312	-0.253	-0.260
mpk	-0.062	-0.151	-0.242	-0.286	-0.314	-0.351	-0.331	-0.354	-0.379	-0.422	-0.451	-0.483	-0.477	-0.558	-0.400
reg	0.077	0.042	-0.010	-0.079	-0.105	-0.121	-0.109	-0.119	-0.112	-0.163	-0.179	-0.232	-0.192	-0.166	-0.131
nn	-0.701	-0.594	-0.534	-0.520	-0.481	-0.456	-0.413	-0.400	-0.371	-0.365	-0.374	-0.364	-0.329	-0.251	-0.432
knn	-0.465	-0.322	-0.239	-0.252	-0.227	-0.206	-0.185	-0.189	-0.159	-0.142	-0.131	-0.166	-0.091	-0.048	-0.187
wknn	-0.361	-0.210	-0.130	-0.146	-0.108	-0.105	-0.064	-0.080	-0.052	-0.053	-0.074	-0.066	-0.032	-0.017	-0.093
dtree	-0.144	-0.155	-0.085	-0.146	-0.135	-0.141	-0.134	-0.193	-0.182	-0.185	-0.193	-0.197	-0.135	-0.071	-0.152
rf	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
svm	-0.451	-0.295	-0.215	-0.208	-0.167	-0.142	-0.110	-0.111	-0.086	-0.088	-0.116	-0.095	-0.039	-0.037	-0.139

Notes:

This table compares the error variance among models over different booking window stage. The criteria used is the standard deviation of errors (SDE) of the models. The results are from the empirical study using multiple hotels with up to 14 days historical bookings, and information including pricing, location, star rating, and client review score. The benchmark is the SDE of Random Forest. The results are calculated by

$$\frac{SDE_{RF} - SDE_i}{SDE_i}$$

where i indicates each of the additive pick-up, multiplicative pick-up, linear regression, neural network, k-nearest neighbors, weighted k-nearest neighbors, decision tree, and random forest. The last column calculates the overall SDE (absolute value) comparison between SVM and other models.

Figure A.1: Robustness Test: Model Performances of Empirical Study 1

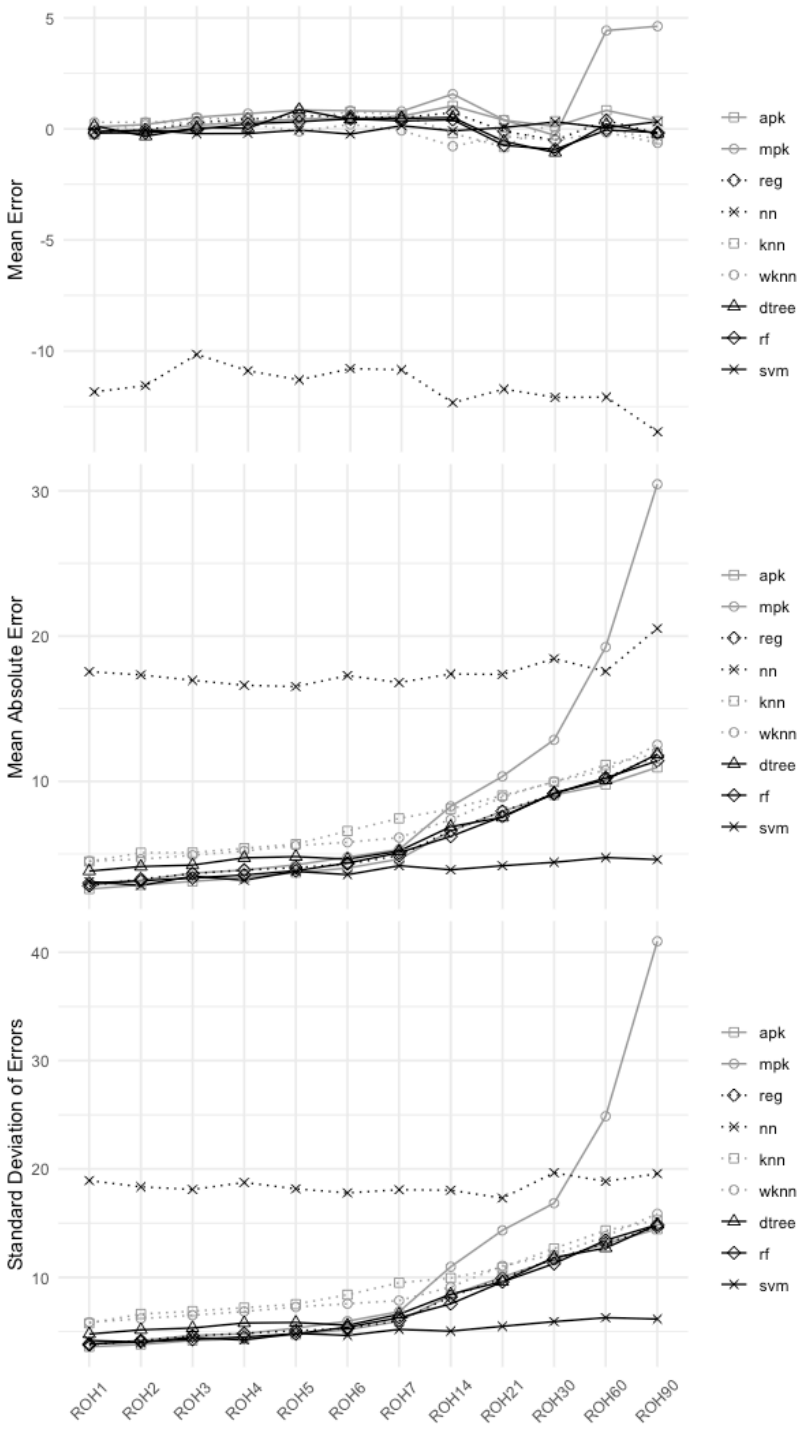
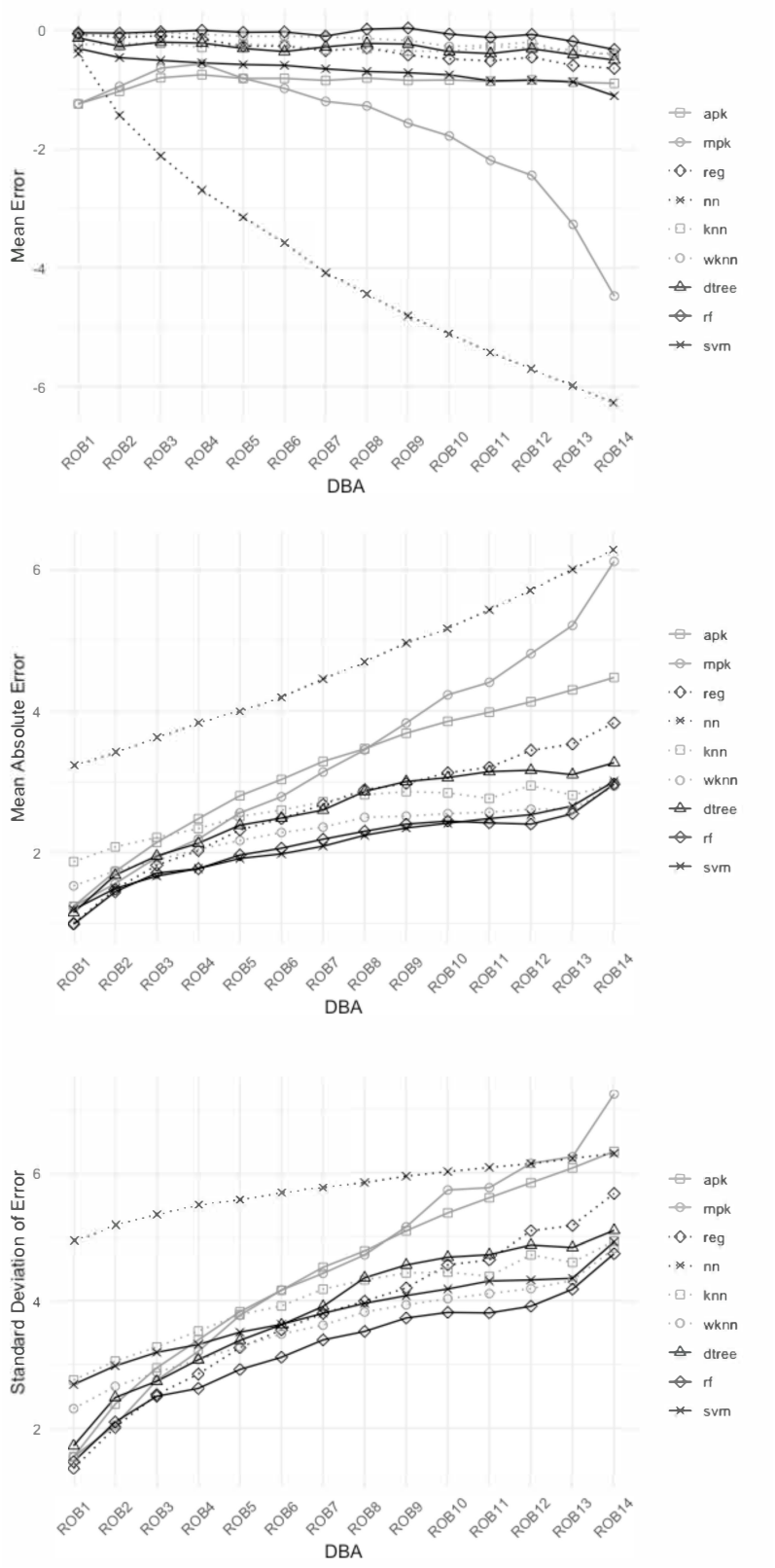


Figure A.2: Robustness Test: Model Performances of Empirical Study 2



BIBLIOGRAPHY

Ahmed, N. K., Atiya, A. F., Gayar, N. E., and El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29(5-6):594–621.

Alkhatib, K., Najadat, H., Hmeidi, I., and Shatnawi, M. K. A. (2013). Stock price prediction using k-nearest neighbor (knn) algorithm. *International Journal of Business, Humanities and Technology*, 3(3):32–44.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185.

Andrew, W. P., Cranage, D. A., and Lee, C. K. (1990). Forecasting hotel occupancy rates with time series models: An empirical analysis. *Hospitality Research Journal*, 14(2):173–182.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Chen, C. and Kachani, S. (2007). Forecasting and optimisation for hotel revenue management. *Journal of revenue and pricing management*, 6(3):163–174.

Corazza, M., Fasano, G., and Mason, F. (2014). An artificial neural network-based technique for on-line hotel booking. *Procedia Economics and Finance*, 15:45–55.

Dudani, S. A. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(4):325–327.

El Shazly, M. R. and El Shazly, H. E. (1999). Forecasting currency prices using a genetically evolved neural network architecture. *International review of financial analysis*, 8(1):67–82.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Fritsch, S., Guenther, F., and Wright, M. N. (2019). *neuralnet: Training of Neural Networks*. R package version 1.44.2.

Galeshchuk, S. (2016). Neural networks performance in exchange rate prediction. *Neurocomputing*, 172:446–452.

Gepp, A., Kumar, K., and Bhattacharya, S. (2010). Business failure prediction using decision trees. *Journal of forecasting*, 29(6):536–555.

GloabalData (2019). Global hotel: number of rooms nights available by hotel category.

GlobalData (2017). How can artificial intelligence, machine learning, and big data boost efficiency in the industry? *Case study: Machine learning in the hotel industry*.

Hechenbichler, K., Schliep, K., and Wilson, A. (2004). Weighted k-Nearest-Neighbor Techniques and Ordinal Classification. *Discussion paper*, 399.

Hod, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844.

Hospitality Technology (2017). Robots in hospitality: Five trends on the horizon.

James, G., Witen, D., Hastie, T., and Tibshirani, R. (2014). *An Introduction to Statistical Learning with Applications in R*, volume 64.

Karsoliya, S. (2012). Approximating number of hidden layer neurons in multiple hidden layer bpnn architecture. *International Journal of Engineering Trends and Technology*, 3(6):714–717.

Kecman, V. and Wang, L. (2005). Support vector machines: theory and applications.

Kuhn, M., Jed, W., Steve, W., Andre, W., Chris, K., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., the R Core Team, Benesty, M., Lescarbeau, R., Ziem, A., Scrucca, L., Tang, Y., Candan, C., and Hunt., T. (2019). *caret: Classification and Regression Training*. R package version 6.0-84.

Lee, M. (2018). Modeling and forecasting hotel room demand based on advance booking information. *Tourism Management*, 66:62–71.

L'heureux, E. (1986). A new twist in forecasting short-term passenger pickup. In *AGIFORS PROCEEDINGS*.

Li, H. and Sun, J. (2012). Forecasting business failure: The use of nearest-neighbour support vectors and correcting imbalanced samples–evidence from the chinese hotel industry. *Tourism Management*, 33(3):622–634.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3):18–22.

Lim, C., Chang, C., and McAleer, M. (2009). Forecasting h (m) otel guest nights in new zealand. *International journal of hospitality management*, 28(2):228–235.

Lin, F., Yeh, C.-C., and Lee, M.-Y. (2011). The use of hybrid manifold learning and support vector machines in the prediction of business failure. *Knowledge-Based Systems*, 24(1):95–101.

Ma, Y., Xiang, Z., Du, Q., and Fan, W. (2018). Effects of user-provided photos on hotel review helpfulness: An analytical approach with deep leaning. *International Journal of Hospitality Management*, 71:120–131.

Moro, S., Rita, P., and Coelho, J. (2017). Stripping customers' feedback on hotels through data mining: The case of las vegas strip. *Tourism management perspectives*, 23:41–52.

Olden, J. D. and Jackson, D. A. (2002). Illuminating the black box: a randomization approach for understanding variable contributions in artificial neural networks. *Ecological modelling*, 154(1-2):135–150.

Pereira, L. N. (2016). An introduction to helpful forecasting methods for hotel revenue management. *International Journal of Hospitality Management*, 58:13–23.

Phillips, P., Zigan, K., Silva, M. M. S., and Schegg, R. (2015). The interactive effects of online reviews on the determinants of swiss hotel performance: A neural network analysis. *Tourism Management*, 50:130–141.

Rajopadhye, M., Ghalia, M. B., Wang, P. P., Baker, T., and Eister, C. V. (2001). Forecasting uncertain hotel room demand. *Information sciences*, 132(1-4):1–11.

Schliep, K. and Hechenbichler, K. (2016). *kknn: Weighted k-Nearest Neighbors*. R package version 1.3.1.

Schwartz, Z. and Hiemstra, S. (1997). Improving the accuracy of hotel reservations forecasting: curves similarity approach. *Journal of Travel Research*, 36(1):3–14.

Schwartz, Z., Uysal, M., Webb, T., and Altin, M. (2016). Hotel daily occupancy forecasting with competitive sets: a recursive algorithm. *International Journal of Contemporary Hospitality Management*, 28(2):267–285.

Tsai, C. and Wang, S. (2009). Stock price forecasting by hybrid machine learning techniques. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, page 60.

Tse, T. S. M. and Poon, Y. T. (2015). Analyzing the use of an advance booking curve in forecasting hotel reservations. *Journal of Travel & Tourism Marketing*, 32(7):852–869.

Vapnik, V. N. (1995). The nature of statistical learning. *Theory*.

Weatherford, L. R. and Kimes, S. E. (2003). A comparison of forecasting methods for hotel revenue management. *International journal of forecasting*, 19(3):401–415.

Yang, Y., Tang, J., Luo, H., and Law, R. (2015). Hotel location evaluation: A combination of machine learning tools and web gis. *International Journal of Hospitality Management*, 47:14–24.

Yao, Z., Xu, X., and Yu, H. (2018). Floor Heating Customer Prediction Model Based on Random Forest. *Proceedings - 17th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2018*, pages 573–578.

Zakhary, A., Atiya, A. F., El-Shishiny, H., and Gayar, N. E. (2011). Forecasting hotel arrivals and occupancy using monte carlo simulation. *Journal of Revenue and Pricing Management*, 10(4):344–366.

Zakhary, A., El Gayar, N., and Atiya, A. F. (2008). A comparative study of the pickup method and its variations using a simulated hotel reservation data. *ICGST international journal on artificial intelligence and machine learning*, 8:15–21.