Predicting hotel booking cancellations using machine learning

Abstract

Hotel room cancellations have been a persistent problem plaguing the hotel industry. The adverse impacts of unpredictable and abrupt booking cancellations not only cost hotels huge financial losses, but also bring about operational difficulties in the employment of staff and the provision of logistical supplies. Therefore, our project aims to employ and compare the performance of machine learning algorithms in determining the most suitable model to predict cancellations for two particular hotels in Gothenburg, Sweden, and to identify underlying factors resulting in these cancellations via vari- able selection. These machine learning algorithms include Logistic Regression (with Regularization), Extreme Gradient Boosting, Adaptive Boosting, Artificial Neural Network (ANN), Random Forest, Decision Tree, and Na¨ıve Bayes Classifier. We conclude that ANN best modeled our dataset, attaining a high accuracy of 99.61% and 99.81% sensitivity. This is followed by XGBoost with sensitivity 99.14%. Additionally, deposit type, arrival date of month and lead time are major factors affecting whether a hotel guest would cancel his or her booking.

Keywords: machine learning; hotel booking cancellation; binary classification

Contents

1 Introduction

1.1 Background and Problem Discussion

Last-minute cancellations are often a bane in many service-oriented industries—aviation, fine-dining, and hotel industries. Cancelled reservations, especially those that cannot be replaced with another guest immediately not only cost huge financial losses for these companies, but also generate problems like understaffing and oversupply of logistical resources. In fact, the total financial loss incurred due to hotel cancellations amounted to up to 2.7 million Euros over all hotels in Europe in 2015 (Antonio et al., 2017). As bookings signify a contract between a customer and the hotel (Talluri & Van Ryzin, 2004), a cancellation of a booking is a termination of this contract prior to the guest's arrival. For simplicity, we consider no-shows to be a cancellation, as they are indeed cancellations but without the guest giving prior notice.

Unsurprisingly, given the troubling nature of cancellations, forecasting customer behaviour has beenaround for more than 40 years, when the aviation industry used customer behaviour mapping techniques to reduce unpredictable behaviours and therefore mitigate potential financial losses (Chiang et al., 2007). The hotel industry first began adopting these methods in 1980, with an emphasised focus on cancellations that are nearer to the booking date as these seemingly last-minute cancellations tend to be more volatile and therefore more impactful to the hotelier (Uysal & O'Leary, 1986). Many of these forecasting tech- niques, however, take only a novice approach to this prediction problem and as (Antonio et al., 2017) put it, are "poorly represented in the scientific literature", especially because machine learning techniques are underutilised. (Antonio et al., 2017) goes further to discuss the limitations of such studies, one of which being the external validity problem of low predictive power when applying the results of one specific hotel to another. Hence, we seek to perform our prediction on two different hotels and assess the validity ofthe results obtained.

1.2 Outline

This paper's approach is as follows: First, we seek to conduct exploratory data analysis on a dataset of hotel booking cancellations in Gothenburg, Sweden. Next, we attempt to investigate whether machine learning techniques enable us to predict hotel bookings based on the explanatory variables as provided in the dataset, and to compare the predictive accuracy of different deep learning models to determine the most suitable model for cancellation forecasting. These variables are typical factors affecting bookings which are obtained from a Hotel Property Management system (PMS) database and summarised in Figure 1 below. Additionally, we seek to answer the research question of which variable(s) play the most significant role in predicting customer cancellation behaviour. Lastly, we discuss the limitations of our findings and propose recommendations for future studies.

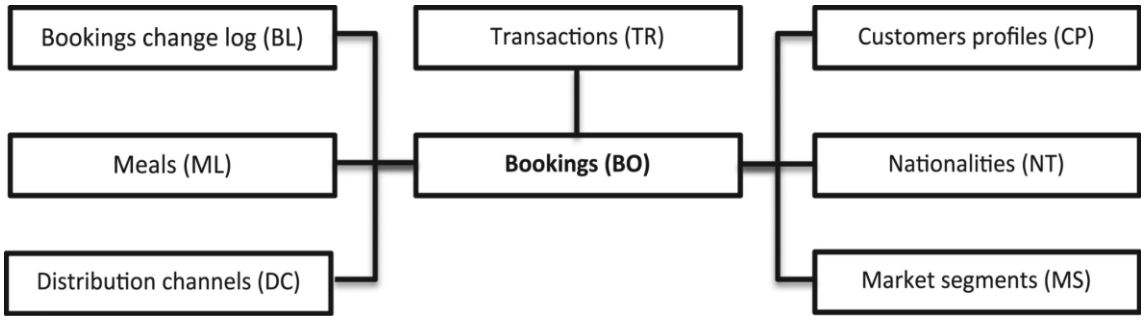

Figure 1: Functions of a hotel Property Management System

2 Exploratory Data Analysis (EDA)

2.1 Dataset Description

The raw dataset "hotel bookings.csv" was obtained from an online source on Kaggle and contains customers' particulars and booking information for two hotels, a city hotel and a resort hotel, in Gothenburg,

No	Feature	Description
		Factor Variables
1	*hotel*	Type of hotel; City Hotel / Resort Hotel
2	*arrival date month*	Month of arrival date; from "January" to "December"
3	*meal*	Customers' meal type
4	*country*	Country of origin
5	*market segment*	Market segment designation
6	*distribution channel*	Booking distribution channel
7	*reserved room type*	Room type reserved
8	*assigned room type*	Room type assigned
9	*deposit type*	Whether a deposit was made or not
10	*agent*	Travel agency ID that made the booking
11	*company*	Company ID that made the booking
12	*customer type*	Whether the booking was part of a group or not
13	*reservation status*	Whether customers checked out / No show / Cancelled
14	*reservation status date*	Date at which the last status was set
		Integer Variables
15	*is canceled*	Indicator of booking cancellations (1) or not (0)
16	*lead time*	No. days between booking date and arrival date
17	*arrival date year*	Year of arrival
18	*arrival date week number*	Week number of arrival date
19	*arrival date day of month*	Day of the month of arrival
20	*stays in weekend nights*	No. weekend nights guest stayed or booked to stay at hotel
21	*stays in week nights*	No. weekday nights guest stayed ot booked to stay at hotel
22	*adults*	No. adults
23	*children*	No. children
24	*babies*	No. babies
25	*is repeated guest*	Indicator of repeated guest (1) or not (0)
26	*previous cancellations*	No. previous booking cancellations made
27	*previous bookings not cancelled*	No. previous bookings not cancelled
28	*booking changes*	No. changes made to booking
29	*days in waiting list*	No. days booking was in waiting list before confirmation
30	*required car parking spaces*	No. car parking spaces required by customer
31	*total of special requests*	No. special requests made by customer (e.g. twin bed or high floor)
		Numeric Variable
32	*adr*	Average daily rate; calculated by dividing sum of all lodging transactions by the total no. of staying nights

Table 1: Summary of variables in dataset

Sweden. The dataset consists of 119390 observations and 32 variables, comprising factor, integer and numeric variables described in Table 1.

The response variable of interest is ***is_canceled***, a binary variable which is an indicator of booking cancellations ('1' when booking is cancelled and '0' when booking is not cancelled). An approximate 37% of the observations corresponds to the booking cancellation class, while the remaining 63% in the non-cancelled class. Hence, we seek to perform a binary classification on booking cancellation based onthe predictors and models trained.

2.2 Data Cleaning & Feature Engineering

The following data cleaning steps have been applied to the raw data "hotel_bookings.csv":

1. All "NULL" and "NA" entries are replaced with "0". Such null entries are only found in columns of three variables- *agent, company and country*. Observations with values "0" for *agent*/*company* indicates that booking was not done through an agent/company respectively.

2. Remaining empty entries are identified and assigned the value "0".

3. Rows containing "0" for all three variables *adults, children* and *babies* indicate non-existing cus- tomers and are removed.

4. The following five variables are removed:

   i) reservation_status

   ii) arrival_date_year

   iii) agent

   iv) company

   v) country

	City Hotel	Resort Hotel										
hotel	1	2										
	Janauary	February	March	April	May	June	July	August	September	October	November	December
arrival_date_month	1	2	3	4	5	6	7	8	9	10	11	12
	BB	FB	HB	SC	Others							
meal	1	2	3	4	5							
	Aviation	Complementary	Corporate	Direct	Groups	Offline TA/TO	Online TA					
market_segment	1	2	3	4	5	6	7					
	Corporate	Direct	GDS	TA/TO	Others							
distribution_channel												
	A	D	E	F	G	B	Others					
reserved_room_type	1	2	3	4	5	6	7					
	A	D	E	F	G	C	Others					
assigned_room_type	1	2	3	4	5	6	7					
	No deposit	Non Refund	Refundable									
deposit_type	1	2	3									
	Contract	Group	Transient	Transient-Party								
customer_type	1	2	3	4								
	Janauary	February	March	April	May	June	July	August	September	October	November	December
reservation_month	1	2	3	4	5	6	7	8	9	10	11	12

Table 2: Encoded Factor Variables

The variable *reservation_status* and the response variable *is_canceled* are essentially the same things since we regard no-shows as cancellations. We exclude the variable *arrival_date_year* and use the other period-relevant predictors. Since both *agent* and *company* are just IDs of the respective agent and company where bookings were made, column entries are numeric and that no specific agent nor company names can be identified. Hence, their effects on booking cancellations cannot be interpreted and directed to a named agent or company. Lastly, some countries only appear once and this does not fit well when we split our data, hence *country* is removed.

5. The variable *reservation_status_date* is split into two variables- a factor variable *reservation_month* and a numerical variable *reservation_day*.

We classified the 28 variables into two groups- numerical (18 variables) and factor (10 variables). The factor variables are encoded as shown in Table 2. In the numerical group, we identified variables with high variance and they are scaled using the natural logarithm transformation. These variables include *lead_time, days_in_waiting_list & adr*. The dimensions of our cleaned dataset "rawdata" are 119,210 by 28.

2.3 Correlation Plot

Variable names are shortened as shown in Table 3. These shortened names are used for the correlation plot in Figure 2

is_canceled	**cancel**	*lead_time*	**lead**	*arrival_date_week_number*	**week arr**
arrival_date_day_of_month	**day arr**	*stays_in_weekend_nights*	**wkend**	*stays_in_week_nights*	**week**
is_repeated_guest	**rep**	*previous_cancellations*	**prev can**	*previous_bookings_not_canceled*	**prev not**
booking_changes	**change**	*days_in_waiting_list*	**wait**	*required_car_parking_spaces*	**car park**
total_of_special_requests	**req**	*reservation_day*	**day res**		

Table 3: Shortened Variable Names

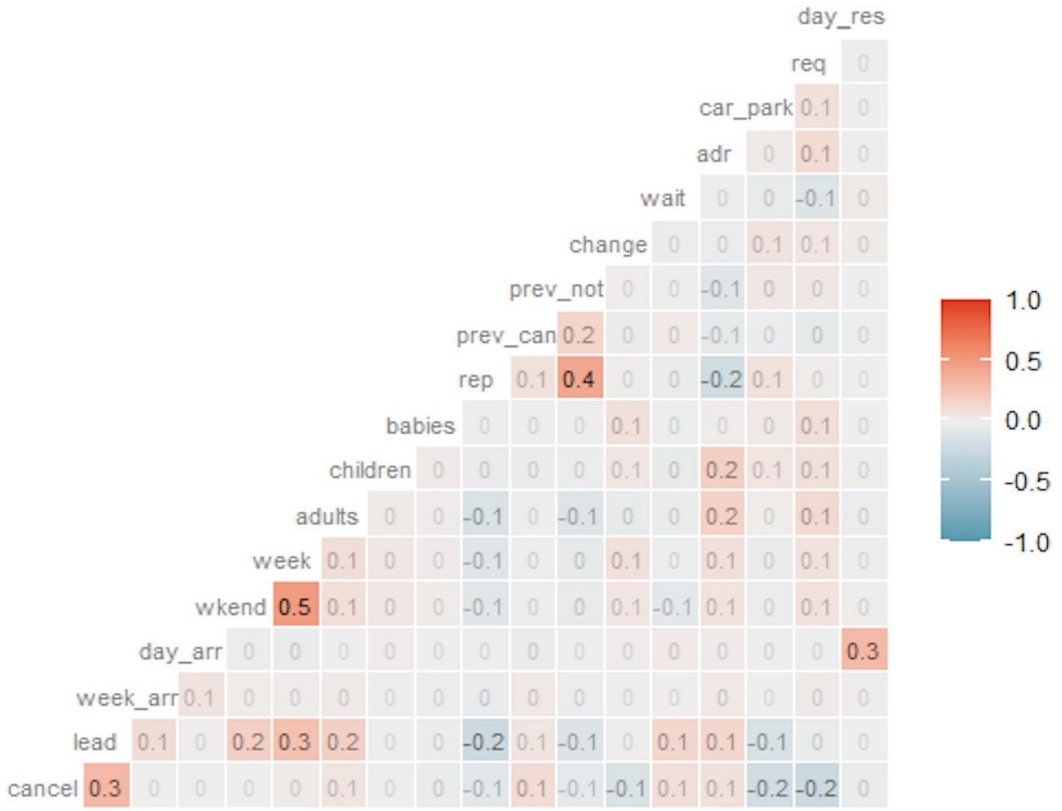

Figure 2: Correlation plot between Numeric Variables

To investigate potential correlations between the variables, we generated a correlation plot. Correlations are not extremely high, as shown in Figure 2. Hence, we can conclude that the chance for the problem of multicollinearity is low. The pair of variables with the highest correlation are *stays in weekend nights* & *stays_in_week_nights* (corr =0.5). The next pair of variables with the highest correlation (0.4) is*is repeated guest* & *previous bookings not cancelled*. However, these correlations are not strong. Hence,we decided to retain all the variables.

2.4 How does the price vary per night over the year?

Figure 3 shows the variation in hotel prices per night over the year. In general, hotel prices are more volatile for resort hotels as compared to city hotels, with sharp price hikes of around $100 per night between May to August. This exploratory analysis enables us to determine whether higher price variations over a year in specific months significantly affect hotel booking cancellations.

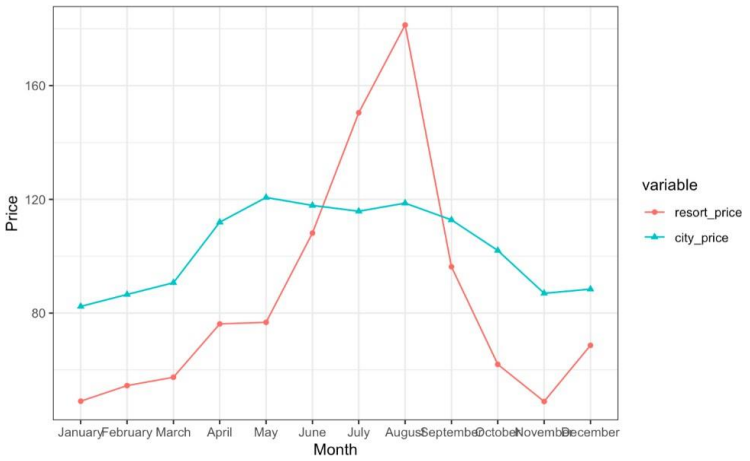

Figure 3: Hotel price variations over the year

2.5 Which are the most busy months?

Figure 4 depicts the frequency of guest check-ins over the course of 3 years from 2015-2017. The months of July and August record the highest guest check-ins, and a possible reason for this is that these are the summer vacation months. This exploratory analysis is important to our study because trends in guest check-ins by month could give us an insight as to whether higher hotel cancellation counts coincide with vacation months as a result of guests shopping for cheaper accommodation options.

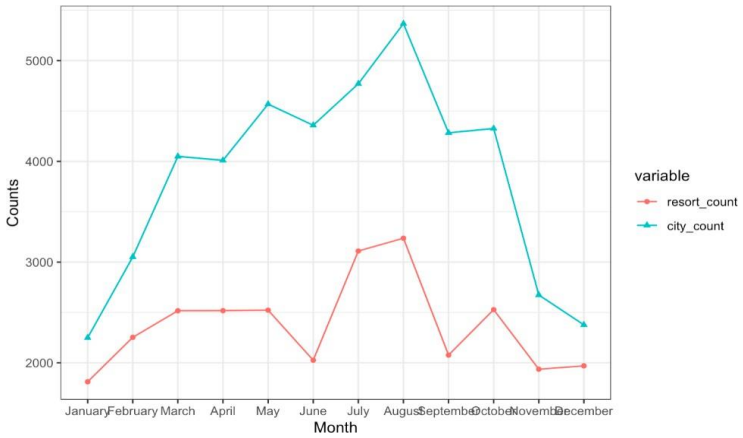

Figure 4: Time-plot of guest check-ins for both hotels

2.6 Data Split

To avoid model overfitting, we split our cleaned dataset randomly into 70% training and 30% test sets. The training set will be used for modelling and test set is used to evaluate models' performance.

3 Modelling

This section discusses various machine learning models to predict hotel booking cancellations. The choice of parameters for each model is explained in the respective sub-sections and further methods were implemented to address overfitting.

3.1 Penalized Logistic Regression

3.1.1 Description

Since our study is concerned with binary classification of hotel booking cancellations, we first employed the sigmoid logistic regression model on the binary response variable *is_canceled*, with the output range [0, 1]. We applied regularization to the regression, due to our dataset having many variables. A 5-fold cross-validation is implemented to overcome the issue of overfitting. This reduces model complexity and performs variable selection simultaneously. We consider three penalized regressions:

- Least Absolute Shrinkage and Selection Operator (LASSO)
- Adaptive LASSO (ALASSO)
- Elastic Net

Variables are normalised to ensure large coefficients (β_j) do not dominate the penalty term (λ) in the loss function. The ALASSO regression aims to reduce bias of LASSO, by incorporating respective weights (w_j) of the predictors into the penalty term and performing different regularization for each coefficient. w_j is computed as $1/|\hat{\beta}_j|^{\gamma}$, where $\hat{\beta}$ is an initial estimate of the coefficients obtained from the ridge regression, and $\gamma > 0$ is the adjustment. The Elastic Net regression is a combination of ridge and lasso regression. Ridge regression is not discussed here as it is incapable of variable selection. A 5-fold cross-validation was performed to tune the hyperparameters λ and α. The optimal values of the hyperparameters were selected for the best model, which was used for prediction on the test data.

3.1.2 Results and Discussion

The top three variables with the highest estimated coefficients and thus the most effect on cancellations are:

LASSO:	*previous_cancellations*	*deposit_typeNon Refund*	*required_car_parking_spaces*
ALASSO:	*required_car_parking_spaces*	*deposit_type*	*previous_cancellations*
Elastic Net:	*deposit_typeNon Refund*	*previous_cancellations*	*required_car_parking_spaces*

Table 4: Variable importance in penalized regressions

All three penalized models selected the same top three variables with the most effect on cancellations and they seem to be equally important in determining booking cancellation. Our results show that the penalised regression models achieved a simpler model without compromising much on model performance in terms of accuracy. Additional metrics were calculated to compare the performances.

	Optimal Hyperparameter(s)	**Performance Metrics**			
		Accuracy	**Sensitivity**	**Specificity**	**F1 score**
LASSO	lambda = 0.001	0.8261	0.6329	0.9413	0.8375
ALASSO	lambda = 0.00553	0.8123	0.5869	0.9429	0.8258
Elastic Net	lambda = 0.01 ; alpha = 0.25	0.8153	0.5952	0.9465	0.8313

Table 5: Performance Metrics & Optimal Hyperparameter(s) of penalized regressions

LASSO performed the best in terms of accuracy, sensitivity and F1 score. Despite this, we believe that the regularized models are preferred as it is less likely to overfit the training data. Coefficients of the LASSO regression is show in Figure 5, where the variables with the highest coefficients are summarisedin Table 4.

	s1
(Intercept)	-3.469094e+00
hotelResort Hotel	9.812174e-02
arrival_date_monthFebruary	2.054941e-01
arrival_date_monthMarch	-1.725265e-01
arrival_date_monthApril	-1.312787e-01
arrival_date_monthMay	-3.280973e-02
arrival_date_monthJune	8.424162e-02
arrival_date_monthJuly	3.349271e-02
arrival_date_monthAugust	-1.426773e-01
arrival_date_monthSeptember	1.824408e-02
arrival_date_monthOctober	-7.970286e-02
arrival_date_monthNovember	1.441477e-04
arrival_date_monthDecember	1.456806e-03
mealFB	3.647431e-02
mealHB	-2.815400e-02
mealSC	1.252908e-02
mealUndefined	-7.824451e-02
market_segmentComplementary	1.110876e-01
market_segmentCorporate	.
market_segmentDirect	.
market_segmentGroups	.
market_segmentOffline TA/TO	-2.422497e-01
market_segmentOnline TA	3.216828e-01
market_segmentUndefined	1.045465e-02
distribution_channelDirect	-7.192213e-02
distribution_channelGDS	-2.721176e-02
distribution_channelTA/TO	.
distribution_channelUndefined	2.355429e-02
reserved_room_typeB	1.441378e-02
reserved_room_typeC	8.040605e-02
reserved_room_typeD	3.303442e-01
reserved_room_typeE	2.457630e-01
reserved_room_typeF	1.389003e-01
reserved_room_typeG	1.587339e-01
reserved_room_typeH	.
reserved_room_typeL	4.542301e-03
assigned_room_typeB	-4.204819e-02
assigned_room_typeC	-1.550379e-01
assigned_room_typeD	-4.473534e-01
assigned_room_typeE	-3.034667e-01
assigned_room_typeF	-2.595505e-01
assigned_room_typeG	-2.291206e-01
assigned_room_typeH	-3.773466e-05
assigned_room_typeI	-1.248436e-01
assigned_room_typeK	-5.145960e-02
assigned_room_typeL	6.259561e-03
deposit_typeNon Refund	1.627146e+00
deposit_typeRefundable	2.037305e-03
customer_typeGroup	-1.089874e-02
customer_typeTransient	2.117039e-01
customer_typeTransient-Party	.
reservation_month2	8.896706e-02
reservation_month3	-3.294487e-02
reservation_month4	-2.453329e-01
reservation_month5	-3.118393e-01
reservation_month6	-3.373821e-01
reservation_month7	-4.973534e-01
reservation_month8	-6.142302e-01
reservation_month9	-3.466670e-01
reservation_month10	-3.686813e-01
reservation_month11	-2.089577e-01
reservation_month12	-5.191863e-02
lead_time	5.874463e-01
arrival_date_week_number	2.528596e-01
arrival_date_day_of_month	.
stays_in_weekend_nights	1.024844e-02
stays_in_week_nights	.
adults	4.884781e-02
children	1.030705e-01
babies	1.416644e-02
is_repeated_guest	-3.025666e-02
previous_cancellations	1.908794e+00
previous_bookings_not_canceled	-5.066393e-01
booking_changes	-2.212366e-01
days_in_waiting_list	-1.329508e-02
adr	2.309929e-01
required_car_parking_spaces	-1.142282e+00
total_of_special_requests	-5.588278e-01
reservation_day	-4.470985e-02

Figure 5: LASSO Coefficients

3.2 Extreme Gradient Boosting (XGBoost)

3.2.1 Description

Extreme Gradient Boosting (XGBoost) is a classification model that is composed of the summation over K basic classification models. (Chen & Guestrin, 2016). It is an extension of the gradient boosting ensemble algorithm and renowned for its performance accuracy and scalability. Additionally, XGBoost

also enables us to carry out feature selection in terms of relative importance to the final model. Assuming our t^{th} iteration tree model is

$$\hat{y}_i(t) = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i(t-1) + f_t(x_i) \quad (1)$$

Where $\hat{y}_i(t)$ represents the prediction result for sample i after the t^{th} iteration. $f_t(x_i)$ represents the t^{th} tree model. Based on the model, we can write its loss function as

$$L = \sum_{i=1}^{n} l(y_i, \hat{y}_i(t)) \quad (2)$$

To prevent overfitting, XGBoost introduces regularization to penalize complex models by minimizing the objective function

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i(t)) + \sum_{k=1}^{t} \Omega(f_i) \quad (3)$$

Where $\sum_{k=1}^{t} \Omega(f_i)$ is the summation of t^{th} tree's complexity, representing the regularization part. From (1) and (2), we derive

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i(t-1) + f_t(x_i)) + \sum_{k=1}^{t} \Omega(f_i) = \sum_{i=1}^{n} l(y_i, \hat{y}_i(t-1) + f_t(x_i)) + \Omega(f_i) + constant \quad (4)$$

Since we can determine the structure of the previous (t-1) tree models, their complexity can be taken as a constant.

Recall the second order Taylor expansion:

$$f(x + \Delta x) = f(x) + \frac{df(x)}{dx}\Delta x + \frac{1}{2} f''(x)\Delta x^2 \quad (5)$$

If we consider $f(x)$ as the loss function $l(y_i, y_i(t-1)+f_t(x_i))$, where x represents the previous (t-1) trees' predicting value and Δx is the t^{th} tree model we are training. Then we can rewrite our loss function as:

$$l(y_i, \hat{y}_i(t-1)) + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \quad (6)$$

Where g_i is the first-order derivative of our loss function and h_i is the second order gradient. Substituting (4) into (3) and omitting all the constant values, we obtain

$$Obj^t = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + \Omega(f_t) \quad (7)$$

Therefore, we only need to solve the last step's loss function's first-order and second-order derivative and use that to solve the objective function. The algorithm then constructs each iteration's new tree model and the final model comes from the combination of these trees.

To apply XGBoost in R, the "xgboost" package will be used. The following data and parameters should be tuned:

1. Input data: xgboost only allows numerical data in matrix form as input variable.
2. Label: Response variable in num format
3. Booster: The boosting model we want to use: gbtree or gblinear
4. Eta: learning rate, used for preventing overfitting
5. Max_depth: the maximum depth of a single tree
6. Objective: The learning objective, in our example, we set it as binary:logistic

7. Nrounds: Number of iterations

In our example, we set booster='gbtree', max_depth=6, eta=0.3 and Nrounds=100.

3.2.2 Results and Discussion

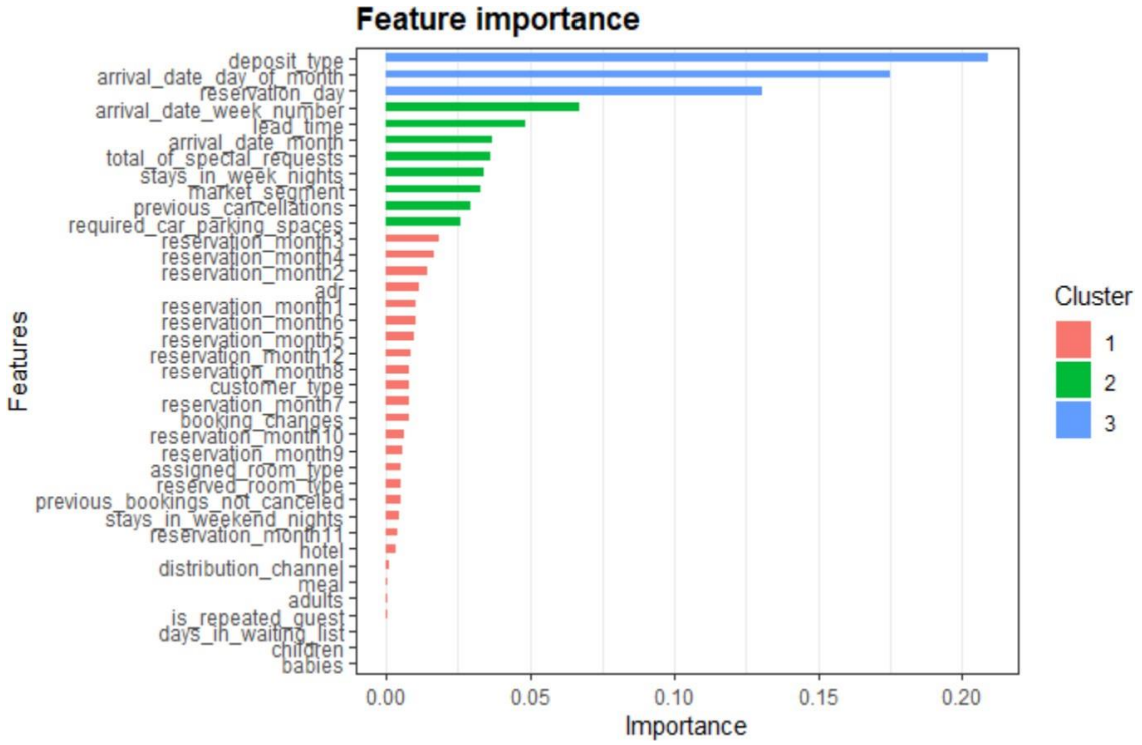

Figure 6: Feature Importance for XGBoost

The confusion matrix shows the predicting accuracy of XGBoost reaches 99.67%. According to the feature importance graph, we can infer the top 5 important variables are *deposit_type, arrival_date_day_of_month, reservation_day, arrival_day_week_number* & *lead_time*. This graph can help us select useful variables when applying other machine learning models. Overall, XGBoost can make accurate predictions efficiently. Like other boosting methods, it is also sensitive to outliers. Excluding outliers is necessary before applying XGBoost.

3.3 Adaptive Boosting(AdaBoost)

3.3.1 Description

Adaptive Boosting (AdaBoost) is one typical embedding machine learning method and usually collabo- rates with decision trees (Kurama, 2019). Its adaptation lies in the fact that the weight of the misclassified samples of the previous weak classifier will increase for training the next classifier, while the weight of correctly classified samples will decrease. At the same time, a new weak classifier will be added in each iteration. Weak classifier means a classifier that can do prediction with low accuracy. The final model is the linear combination of those weak classifiers.

We can write the final strong classifier as follows:

$$F(x) = \sum_{t=1} \alpha_t f_t(x)$$

Where x is input vector, $F(x)$ is the strong classifier. $f_t(x)$ is the weak classifier and α_t is the weight of classifier t. For binary classification, the objective function will be sign($F(x)$).

Given I samples: (x_1, y_1), (x_2, y_2) . . . (x_I, y_I), where x_i is the characteristic, y_i is the class label. First, we will initialize all the samples' weights to be the same, which is

$$\omega_i^0 = \frac{1}{I}, \quad i = 1, 2 \ldots I$$

We set the iteration times be T. Then for t=1...T, we train some weak classifiers and choose the one with the least error rate e_t to be our weak classifier $f(t)$. The error rate can be calculated as $e_t = P(f_t(x_i) \quad y_i)$ The weight of weak classifier t is

$$\alpha_t = \frac{1}{2} log(\frac{1 - e_t}{e_t})$$

After each iteration, we can update the weight of the samples:

$$\omega_i^t = \omega_i^{t-1} \exp(-y_i \alpha_t f_t(x_i)) / Z_t$$

Where Z_t is the total sum of all samples' weights:

$$Z_t = \sum_{i=1}^{n} \omega_i^{t-1} \exp(-y_i \alpha_t f_t(x_i))$$

After finishing the T iterations, we can get each time's classifier and its corresponding weight, the combination of them will formulate our strong classifier model.

In R language, the package 'adabag' will be used for training AdaBoost model. The following data and parameters need to be modified when applying Adaboost:

1. The response variable should be in factor format.
2. mfinal: The iteration times, default value is 100.

3.3.2 Results and Discussion

mfinal	accuracy	Time.in.min
10	0.9000364	1.379933
30	0.9659145	3.949753
50	0.9756452	6.881846
100	0.9843414	14.855089

Table 6: ADAboost's accuracy with different 'mfinal' values

For 'adabag', the hyper-parameter is the number of iterations (mfinal). While a larger number of iterations can give us a more accurate prediction model, it also increases the running time. In order to find an optimal balance between predicting accuracy and efficiency. We tried different mfinal value (10,30,50,100) and listed their results below.

From Table 6, we chose mfinal value as 100. It reaches 98.4% test accuracy, spending 14.9 minutes. Although Aaboost can make an accurate prediction, it is very sensitive to outliers and noise. Hence, eliminating those outliers is necessary during the data cleaning.

3.4 Artificial Neural Network (ANN)

3.4.1 Description

In our Artificial neural networks (ANNs) model, all parameters are tuned via backpropagation based on the error rate obtained in the previous epoch (i.e. iteration.). Since our objective is binary classification, the activation function of the final output unit is sigmoid. Other hypertuned features of our model are given in the table below.

Layer 1	units= 50, activation= relu, regularizer= regularizer_l1(0.001), dropout rate= 0.01
Layer 2	units= 50, activation= relu
Layer 3	units= 2, activation= sigmoid
optimizer	adam
loss	Binary crossemtropy
epochs	20

Table 7: Artificial neural network model features

3.4.2 Results and Discussion

Accuracy	0.9961
Loss	0.0289
Validation accuracy	0.9981
Validation loss	0.0201
Sensitivity	0.9981
Specificity	0.9983

Table 8: ANN result

The final results are shown in Table 8. Given that we have a large dataset (83447 train data and 35763 test data) and low model complexity (3 layers only), and the difference between the training set loss and the validation does not increase with epoch, we can conclude that our model does not overfit.

Hypertuning was done through flags.

Dropout rate	0.01,0.05,0.1,0.2
Number of units in layer 1	50,100,150
Number of units in layer 2	50,100,150
Regularizer_l1	0.0001,0.001,0.01

Table 9: Possible Parameters

All possible parameters combination for tuning are shown above. This will produce 144 possible combinations of parameters. We specified in the 'tuning_run' function that we only want to try 10% of the possible combinations (sampled) to reduce the running time. The optimal combination of parameters shown in Table 7 is not necessarily the best one. It is possible to further increase the model accuracy by conducting further grid search and hyperparameter tuning.

3.5 Random Forest

3.5.1 Description

In our project, through tuning the hyper-parameters by grid research, we select *mtry* =15, *splitrule* = gini, *min.node.size* = 5 and *num.trees*=500 for the hyper-parameters in the final model of Random Forest.

3.5.2 Results and Discussion

The accuracy of Random Forest is 95.98%. Setting the positive class to be 'is_ canceled=1', the sensitivity is 89.70% and specificity is 99.66%.

3.6 Decision Tree

3.6.1 Description

We pruned our decision tree model by adding a penalty term to the loss function. Through this regular- ization process, we performed a grid search for the optimal value of the penalty's coefficient.

3.6.2 Results and Discussion

The optimal value of the penalty term's coefficient is found to be 0.01. We applied that value to construct our final decision tree model, which gives an overall accuracy of 80.6%. The sensitivity is 0.6119 and the specificity is 0.9192.

3.7 Naïve Bayes Classifier

3.7.1 Description

Na¨ıve Bayes is a probabilistic machine learning algorithm based on Bayes Theorem, and it is used in a wide variety of classification tasks (Finnstats, 2021). The Bayes Theorem is defined as:

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

Where:

- X = set of events $\{x_1, x_2, \ldots, x_k\}$
- C_i= Class, i
- P(X)= Probability of event, X
- P(C_i)= Probability of Class, i
- P($C_i|X$)= Probability of class C_i given the occurrence of event X
- P($C_i|X$)= Probability of event X given the occurrence of class C

The fundamental Na¨ıve Bayes assumption is that each feature makes an:

- Independent and
- Equal

contribution to the outcome.

For instance, we assume that *previous_cancellations* and *required_car_parking_spaces* are unrelated, which is intuitively the case. Additionally, every feature contributes equally to predicting the outcome as to whether a guest would cancel his or her hotel booking, and knowing only one or a few features like *market_segment* is insufficient to predict the outcome perfectly.

The Na¨ıve Bayes classifier is constructed by taking into account how each feature contributes towards a certain class prediction, in our case-cancel or no cancel, using conditional probability. It is therefore an interpretable model that is simple (both intuitively and computationally), fast, and also performs well with small training data, and scales well to large data sets. In real life, however, we note that these assumptions may not hold since *market_segment* and meals *assigned_meal_type* are unlikely to be purely independent from each other (hence the name Na¨ıve).

Applying the chain rule and the assumption, we derive:

$$\begin{aligned} P(C_i|X) &= P(x_1, x_2, \ldots, x_k|C_i)P(C_i) \\ &= P(x_1|C_i)P(x_2, \ldots, x_k|C_i)P(C_i) \\ &= (P(x_1|C_i)P(x_2|C_1, x_1)P(x_3, \ldots, x_k|C_i)P(C_i) \\ &= P(x_1|C_i)P(x_2|C_1, x_1)\ldots P(x_k|C_i, x_1, x_2, \ldots, x_{k-1})P(C_i) \\ &= P(C_i) \prod_{j=1} P(x_j|C_i) \end{aligned}$$

In our case, the response variable has only two outcomes, cancelled-1 or non-canceled-0. The optimization problem therefore becomes:

$$\arg\max \; P(C_i) \prod_{j=1} P(x_j|C_i)$$

To maximise the probability of each class. The caret package in R enables us to tune the following hyperparameters in order to increase the Na¨ıve Bayes model performance using a tuning grid for grid search

- **usekernel** parameter allows us to use a kernel density estimate for continuous variables versus a Gaussian density estimate,
- **adjust** allows us to adjust the bandwidth of the kernel density (larger numbers mean more flexible density estimate),
- **fL** allows us to incorporate the Laplace smoother.

3.7.2 Results and Discussion

Using a tuning grid to find the best combination of hyperparameters by running the train data through several iterations, we chose the model where usekernel=FALSE, fL=0, adjust=0.75. The final output shows that we built a Naive Bayes Classifier that can predict whether a guest would cancel his or her hotel booking, with an accuracy of approximately 51.0% on the test data.

The greatest weakness of the na¨ıve Bayes classifier is that it relies on an often-faulty assumption of equally important and independent features which results in biased posterior probabilities. Although this assumption is rarely met, in practice, this algorithm works surprisingly well. This is primarily be- cause what is usually needed is not a propensity (exact posterior probability) for each record that is accurate in absolute terms but just a reasonably accurate rank ordering of propensities.

For example, we may not care about the exact posterior probability of attrition, we just want to know for a given observation, if the posterior probability of attriting is larger than not attriting. Even when the assumption is violated, the rank ordering of the records' propensities is typically preserved. Consequently, na¨ıve Bayes is often a surprisingly accurate algorithm; however, on average it rarely can compete with the accuracy of advanced tree-based methods (random forests & gradient boosting machines) but is definitely worth having in your toolkit.

4 Results & Discussions

In evaluating the models' performance, we consider accuracy, sensitivity, and specificity as the perfor- mance metrics as shown in Table 10. We focus more on sensitivity as the goal is to identify cancellations (Positive Class) so that hotels can better manage their resources and save costs.

Models	Accuracy(%)	Sensitivity(%)	Specificity(%)
Logistic Regression	82.88	64.47	93.86
LASSO	82.61	63.29	94.13
XGBoost	99.67	99.14	99.98
AdaBoost	98.30	95.70	99.83
Artificial Neural Network	99.61	99.81	99.83
Random Forest	95.98	89.70	99.66
Decision Tree	80.58	61.19	91.92
Naïve Bayes Classifier	50.70	25.19	94.98

Table 10: Performance for different models

Na¨ıve Bayes Classifier performs the worst due to its strong independence assumption. The low accuracy of the regressions could be due to the linear classifier, where the classification boundary is a hyperplane. Hence we considered a more complex classifier in the ANN model. Models involving decision tree per- form better, in particular XGBoost, with the highest accuracy. ANN comes in second, with only a mere

difference. ANN also obtained the highest sensitivity. Hence we conclude that ANN is the best model in classifying booking cancellations.

XGBoost reported the top 5 significant variables as: *deposit type, arrival date day of month, reservation day, arrival day week number* & *lead time* (Figure 6). Deposit type affects cancellation the most and this is in line with the results from penalized regressions, in Figure 4. This indicates that guests are more particular with making deposits in hotel bookings. The factor variable *previous cancellations* is selected in all three penalized regressions and positively correlated with the response variable. This is intuitive, since guests who cancelled before are more likely to cancel again, or not even make any bookings at all.

5 Limitations & Conclusion

As the world continues recovering from the COVID-19 pandemic and international travel resumes, so too are hotel booking cancellations expected to become a persistent problem. The findings of this study indicate that lead time, deposit type, and month of arrival are major factors affecting whether a hotel guest would cancel his or her booking. Nonetheless, it is worthwhile to consider that this dataset is a record of booking cancellations in Sweden, so the results might not be externally valid for other hotels in other countries.

Additionally, our data mining algorithms are not without their shortfalls: XGBoost for instance as a gradient boosting algorithm is very sensitive to outliers since every classifier is forced to fix the errors in the predecessor learners. Naive Bias Classifier, on the other hand, might result in 0 frequency phenomenon and hence required a Laplace tuner to tuner the hyperparameters. In view of these limitations, researchers might consider data mining problems of similar nature with more hotels, and more predictors based on the existing literature on hotel booking cancellations.

6 References

Chiang, W.C., Chen, J. C. & Xu, X. (2007), *An overview of research on revenue management: current issues and future research*, International journal of revenue management 1(1), 97–128.

Uysal, M. & O'Leary, J. T. (1986), *A canonical analysis of international tourism demand*, Annals of Tourism Research 13(4), 651–655.

Antonio, N., de Almeida, A. & Nunes, L. (2017), *Predicting hotel booking cancellations to decrease uncertainty and increase revenue*, Tourism & Management Studies 13(2), 25– 39.

Chen, T., & Guestrin, C. (2016). *XGBoost: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Conferences. https://dl.acm.org/doi/pdf/10.1145/2939672.2939785

Freund, Y., & Schapire, R. E. (2005). *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*. ScienceDirect. https://doi.org/10.1006/jcss.1997.1504

Kurama, V. (2019). *A Guide to AdaBoost: Boosting To Save The Day*. PaperspaceBlog. https://blog.paperspace.com/adaboost-optimizer/

Finnstats (2021). *Naive Bayes Classification in R*. R-bloggers. https://www.r-bloggers.com/2021/04/naive-bayes-classification-in-r/

7 Code and dataset availability

All code for exploratory data analysis and model building for our project is available at https://drive.google.com/drive/folders/1kUctXULPHO5VChjWkOvSJ3WvLQpsEN5v?usp=share_link

The raw dataset "hotel bookings.csv" was obtained from an online source on Kaggle at https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand

8 Appendix

8.1 Additional Exploratory Data Analysis

This part includes some more data inference that helps the hotel better understand its customer group.

8.1.1 Where do most guests come from?

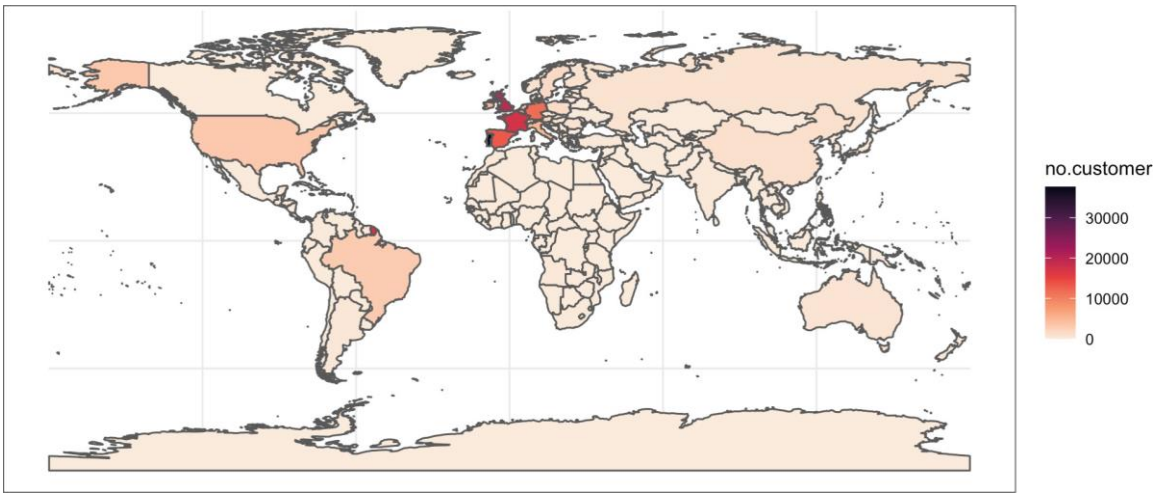

Figure 7: Customers' nationality density plot

The world map above was plotted based on the column "country" and the total customer coming from the country (i.e. corresponding sum of "adults, children, and baby"). It can be seen from the world map that people from all over the world have stayed in these two hotels. Most of them come from Western Europe, with the largest number coming from Portugal.

8.1.2 How much do guests pay for a room per night?

Figure 8 depicts a box plot of hotel prices per night according to room type. A potential area of interest could be whether hotel prices affected cancellations because the reserved room type deviated from the assigned room type, which is specific to our problem of finding major contributing variables of hotel cancellations.

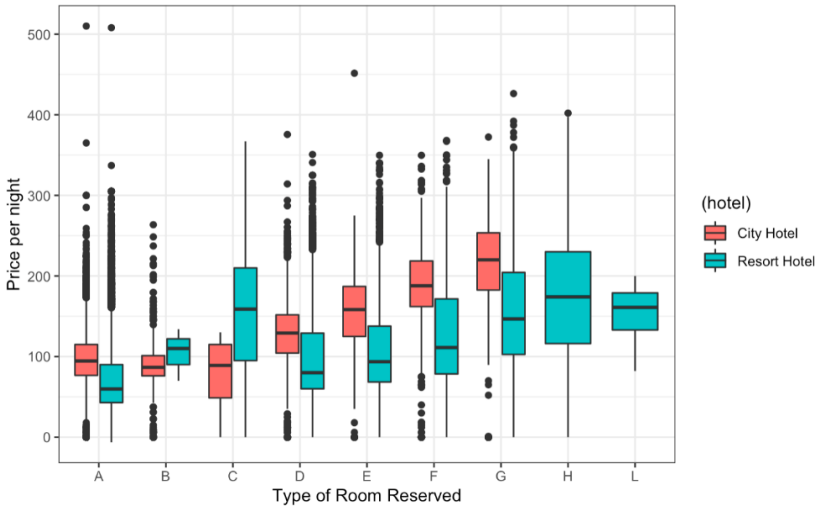

Figure 8: Box Plot of Hotel Room prices according to room type

8.1.3 How long do people stay at the hotels?

Figure 9 below shows the duration of stay of hotel guests in total nights against the frequency count of how many stays occur according to the number of total nights for both City and Resort hotels.

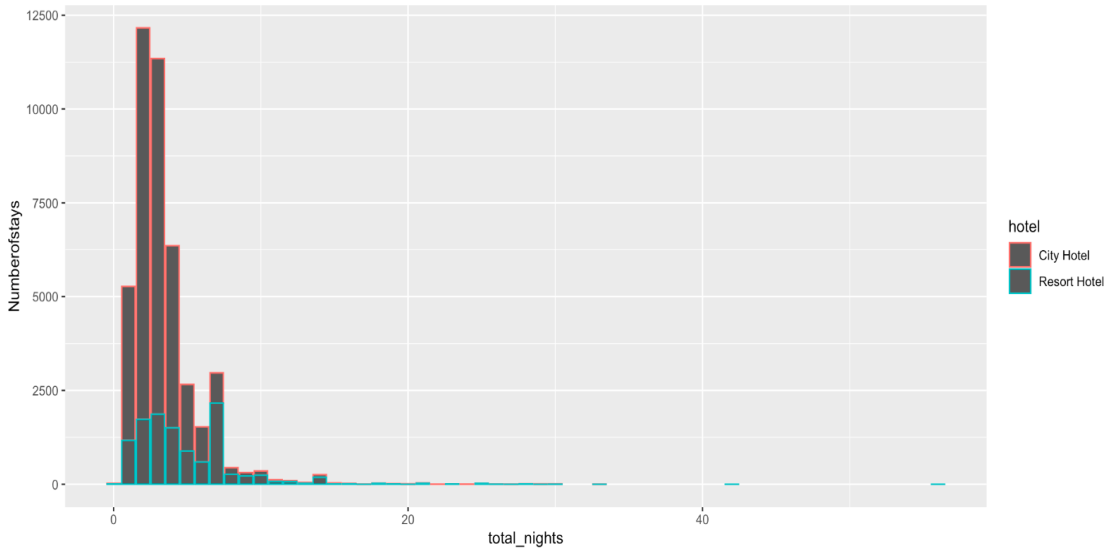

Figure 9: Bar Plot of hotel guest's duration of stay in both types of hotels

We observe that on average, most guests tend to stay 2 to 3 nights per visit. This observation allows us to draw inferences about whether the duration of stay affects booking cancellation during variable selection. Additionally, City hotels generally record a higher number of stays for each number of total nights than Resort hotels. This can be explained by the distribution of data points for City and Resort hotels-67% and 33% respectively.