Prediction of Hotel Reservation Cancellation

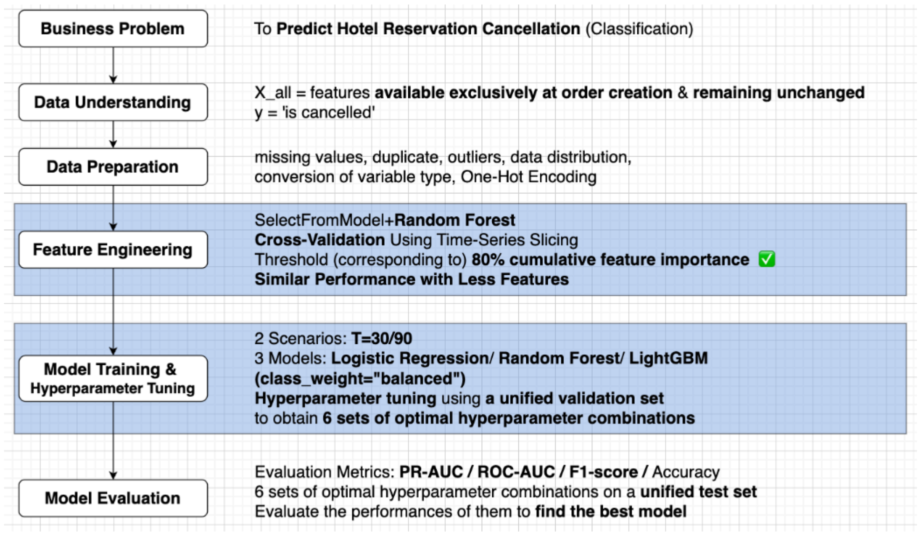
City University of Hong Kong

Semester A (2025-2026)

IS6400 - Business Data Analytics

Student Name	**SID**
Hongyun YANG	59025295
Yiwen XU	59999852
Ziqian GE	59562408
Ying WANG	59459510
Yimin ZHA	59760658
Wenting Yu	60099865

1. Executive Summary



Business Problem
To Predict Hotel Reservation Cancellation (Classification)
Data Understanding
X_all = features available exclusively at order creation & remaining unchanged
y = 'is cancelled'
Data Preparation
missing values, duplicate, outliers, data distribution,
conversion of variable type, One-Hot Encoding
Feature Engineering
SelectFromModel+Random Forest
Cross-Validation Using Time-Series Slicing
Threshold (corresponding to) 80% cumulative feature importance ✅
Similar Performance with Less Features
Model Training & Hyperparameter Tuning
2 Scenarios: T=30/90
3 Models: Logistic Regression/ Random Forest/ LightGBM
(class_weight="balanced")
Hyperparameter tuning using a unified validation set
to obtain 6 sets of optimal hyperparameter combinations
Model Evaluation
Evaluation Metrics: PR-AUC / ROC-AUC / F1-score / Accuracy
6 sets of optimal hyperparameter combinations on a unified test set
Evaluate the performances of them to find the best model

Fig 1.1 Process Flow Diagram

The Process Flow Diagram has revealed the whole workflow of this course project. Having defined the business problem to be classifying whether a hotel reservation will be canceled, we identify features only available at order creation and maintain unchanged as the independent variables and the dependent variable 'is cancelled' (y).

In the stage of **data preparation**, we've conducted ordinary steps such as handle missing values, duplicates, outliers or so to assure data quality.

For **feature engineering**, we employed a Random Forest to assess feature importance, performed time-series slicing cross-validation, and selected features accounting for 80% of cumulative importance. It turns out achieved comparable performance with fewer features, demonstrating an effective feature selection.

In **model training & hyper-parameter tuning**, we explore two scenarios (T=30 and T=90) and three models: Logistic Regression, Random Forest, and LightGBM (with class_weight="balanced"). We used a parametric grid search and a unified validation set for hyper-parameter tuning, yielding 6 optimal parameter combinations.

Finally, in **model evaluation**, we assess models using metrics like PR-AUC, ROC-AUC, F1-score, and Accuracy. All 6 parameter sets were tested on a unified test set to determine the best-performing model.

2. Description of the Specific Business Problem

2.1. Background of the Industry

The hotel industry has long faced a high-impact problem – high cancellation rates. In a typical ‘perishable inventory’ industry like hotels, once a room night is canceled and cannot be resold, the inventory is lost forever and the revenue it brings cannot be recovered. Last-minute cancellations will directly disrupt the hotel's operational processes, including room status management, price optimization, staff scheduling and resource allocation. These uncertainties have caused hotels to continuously struggle between cost control and customer experience, creating obvious operational pain points.

Key operational pain points include revenue erosion, cost inefficiency, collaboration silos and customer experience. As for revenue erosion, if a room is not sold in time, even if the room price is high, it will only generate 0 revenue. This irreversible loss has a huge impact on hotel revenue. As for tendency to higher cost, temporary cancellations force a series of reactive responses from hotels, which definitely cause higher cost, including re-pricing, reallocation of resources for cleaning, front desk, catering. These not only increase labor and management costs of communication and operation, but also reduce operational efficiency. From customers’ perspectives, if customers find that there is no room available when they arrive, as hotels usually use overbooking strategies to reduce expected cancellation rates, they tend to leave bad reviews and have bad experiences.

2.2. Project Objectives

This project aims to use historical booking data to analyze the patterns behind cancellation behavior, thereby helping hotels better manage uncontrollable risks. Our goals are to predict high-risk cancellations in advance and develop more targeted intervention strategies for hotels. By solving these problems, we hope to improve the hotel's operational efficiency, resource allocation accuracy, and ultimately improve guest experience.

3. Description of the Data Collected

3.1. Data Source

The data used in this project comes from Hotel Booking on Kaggle Website. This data set is currently one of the most widely used public data in hotel industry analysis, covering all booking records of City Hotel and Resort Hotel from July 1, 2015, to August 31, 2017, including successful stays and canceled orders, with a total of 119,390 records and 36 variables. The data covers multi-dimensional features such as booking information, customer characteristics, deposit types, booking channels and final booking status, providing a reliable database for analyzing hotel cancellation behavior, customer pattern identification, pricing strategy and operational optimization.

3.2. Variable Description

For subsequent modeling and analysis, this section divides all variables in the data set into three categories according to their properties: target variables, numerical variables and categorical variables. This classification is helpful for understanding the variable and also provides a framework for data cleaning, feature engineering and model training.

Table 3.1 Target Variable

Variable	Meaning
is_canceled	if the booking was canceled (1) or not (0).

Table 3.2 Numerical Variables

Variable	Meaning
lead_time	The number of days between booking and check-in
arrival_date_year	Year of arrival date
arrival_date_week_number	Week number of the arrival date
arrival_date_day_of_month	Day of the month of the arrival date
stays_in_weekend_nights	Number of weekend nights
stays_in_week_nights	Number of wee nights
adults	Number of adults
children	Number of children
babies	Number of babies
is_repeated_guest	Whether it is a repeat customer
previous_cancellations	Number of previous bookings cancelled by the customer prior to the current booking
previous_bookings_not_canceled	Number of previous bookings not cancelled by the customer prior to the current booking
booking_changes	Number of reservation changes
agent	travel agency ID

company	ID of the company
days_in_waiting_list	Number of days the booking was on the waiting list before it was confirmed to the customer
adr	Average Daily Rate
required_car_parking_spaces	Number of parking spaces required by the customer
total_of_special_requests	Number of special request

Table 3.3 Categorical Variables

Variables	Meaning
hotel	City / Resort
arrival_date_month	Month of arrival date: January to December
meal	Types of meal plans (BB, HB, FB)
country	Country of origin
market_segment	- Online TA (Travel Agency) - Offline TA/TO (Tour Operator) - Direct - Groups - Corporate - Undefined
distribution_channel	Booking distribution channel TA-Travel Agents; TO-Tour Operators

reserved_room_type	The room type reserved by the guest
assigned_room_type	The actual allocated room type
deposit_type	- Non-Refund - Refundable - No Deposit
customer_type	- Transient (Individual guest booking without contract) - Transient-Party (Group of transient guests) - Contract - Group (Group booking)
reservation_status	Canceled/Check-Out/No-Show

4. Data Preparation

Data preparation is a critical pre-processing step in building machine learning models, aimed at transforming raw, disorganized data into a clean, structured, and high-quality dataset to ensure effective model learning. The data preparation in this study includes three key steps: data cleaning, variable transformation, and encoding & structuring.

4.1. Data Cleaning: Handling noise in the data

4.1.1. Handling Missing Values

Missing values in the children, agent, and company fields were filled with 0. The reasoning is as follows: missing values in children generally indicate no children, while missing values in agent and company indicate that the booking was made directly, without involving an agent or a company. Filling with 0 aligns with real-world business scenarios and facilitates subsequent binary conversion.

4.1.2. Removing Duplicate Records

Duplicate records were checked and confirmed to be absent, ensuring the independence of each observation.

4.1.3. Handling Numerical Outliers

Boxplots were drawn for all numerical variables, and outliers were identified in the adr variable. These outliers were replaced with the median value. The median is less sensitive to outliers, effectively mitigating their impact while preserving the dataset size.

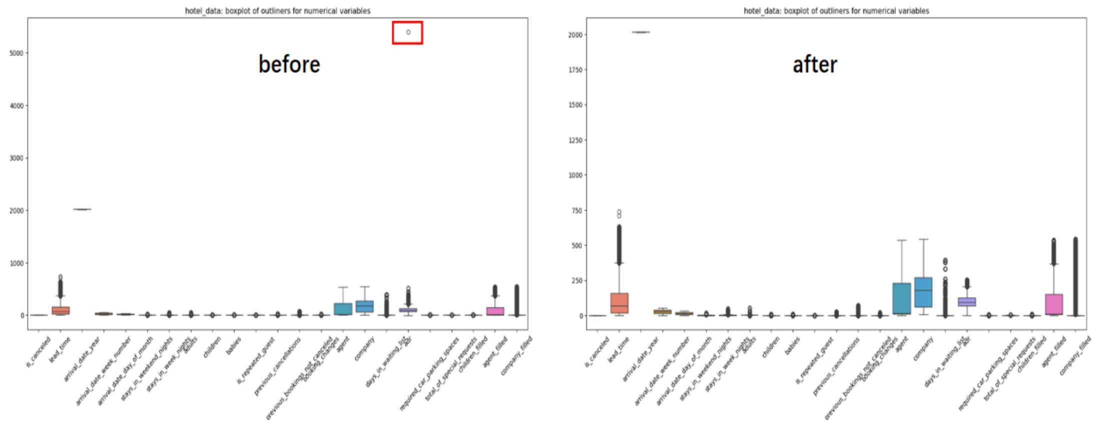


Fig 4.1 Illustration of before and after data cleaning for adr.

4.1.4. Examining and Correcting Data Distribution

Histograms were plotted for all numerical variables, revealing a right-skewed distribution in the lead_time variable. To address this, we applied a log1p transformation. This approach was chosen because the lead_time variable contains a significant number of zero values, making direct logarithmic transformation infeasible. The log1p transformation helps make the distribution more normal, thereby improving model stability and performance.

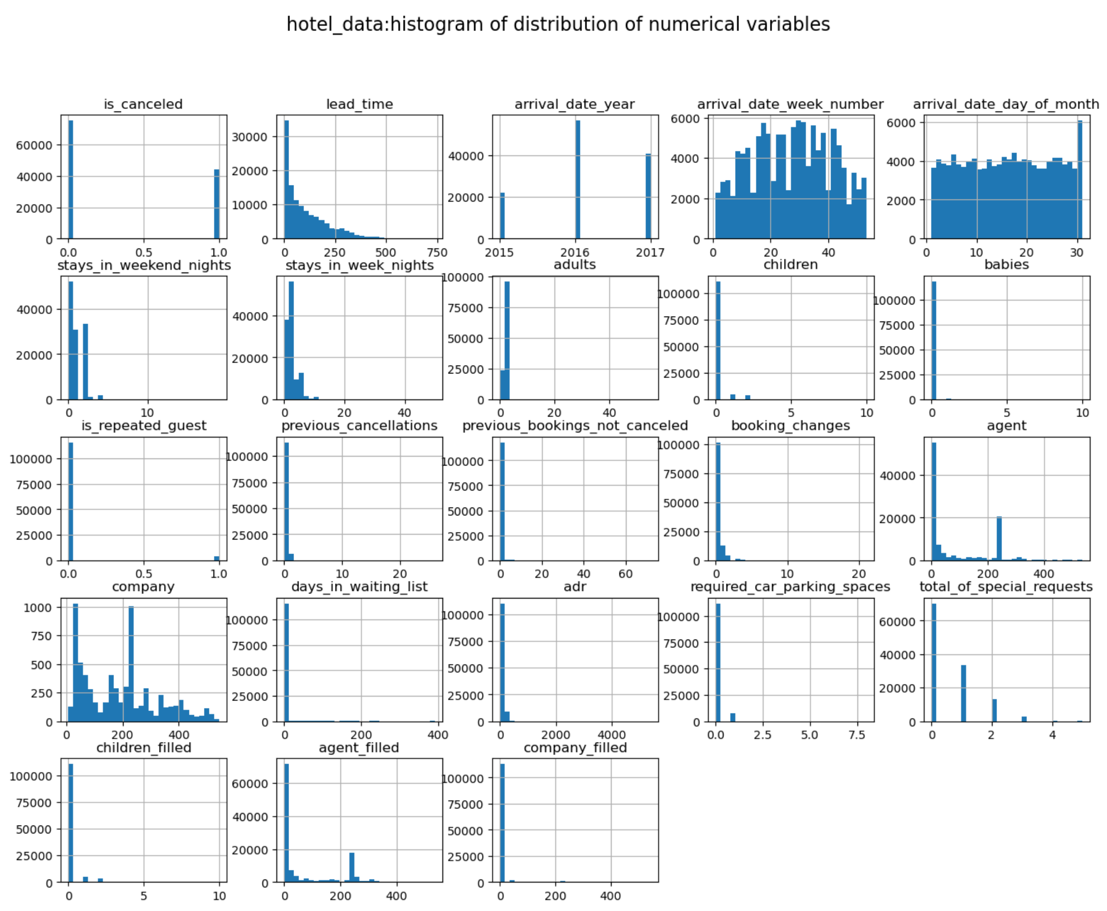


Fig 4.2 Histogram of distribution of numerical variables.

4.2. Variable Transformation: Converting variables for model compatibility

4.2.1. Numerical Mapping

We mapped the categorical feature meal (meal type) into numerical values, creating a new numerical feature called daily_meals. The mapping is as follows: SC (Self Catering) = 0，Undefined = 0，BB (Bed & Breakfast) = 1，HB (Half Board) = 2，FB (Full Board) = 3. The reason for this mapping is that meal types in the hotel business inherently follow an ordinal progression, representing increasing levels of service: SC (self-catering, 0) < BB (bed and breakfast, 1) < HB (half board, 2) < FB (full board, 3). By using numerical mapping, machine learning models can interpret values like "2" as greater than "1", thereby learning that half board represents a higher service level than bed and breakfast alone. This ordinal information would be lost if one-hot encoding were applied.

4.2.2. Binary Conversion

We created binary features via_agent and via_company (yes=1, no=0) based on the categorical features agent_filled and company_filled. This transformation simplifies the original question of "booking through a specific agency/company" to "whether booked through an agency/company", enhancing feature simplicity and interpretability while ensuring these features can be properly processed by machine learning models.

4.3. Encoding & Structuring: Formatting data for model consumption

4.3.1. One-Hot Encoding

We applied one-hot encoding to five categorical variables: market_segment, distribution_channel, reserved_room_type, deposit_type, and customer_type. These are nominal variables where categories have no ordinal relationship. One-hot encoding creates independent binary columns for each category, preventing models from incorrectly assuming numerical relationships between categories.

4.3.2. Time Data Processing

We first converted English month names into numerical values to prepare for generating complete timestamps. Subsequently, we consolidated the time information scattered across separate columns for year, month, and day into a standardized datetime-type feature. The reason for this processing is that a complete timestamp serves as the foundation for time series analysis and time-ordered data splitting, which is crucial for preventing data leakage that would occur if future data were used to predict past events.

5. Feature Engineering

In the feature engineering, the aim of this part is selecting the useful variables. First, based on the prepossessed data, we include approximately 30 feature variables in X, while setting whether the user cancels as the target variable Y. We use the random forest model for data-level screening, then calculate the cumulative importance. Features with cumulative importance exceeding 80% are regarded as effective features, while those below the 80% threshold are recorded as noise variables. Subsequently, we sort the features in descending order of importance and finally obtain the top 10 most significant features. The reasons for selecting the method. Random Forest is an ensemble learning method that improves model performance by constructing multiple decision trees and integrating their prediction results. During the training process, the model can calculate the contribution degree of each feature to the prediction results, namely feature importance. Compared with other models, the random forest model consists of multiple decision trees, offering higher accuracy and insensitivity to outliers, thereby better improving the accuracy of feature importance screening.

In the feature validation part, our goal is to evaluate the performance of the random forest model on the feature subset. We adopt the time series cross-validation method. By comparing the correct prediction scores of the screened data and the original metadata, we ultimately determine whether the feature selection is effective. In time series cross-validation, we split the data into 5 folds. Data before a certain fold's time point is used for training, while the subsequent data fold is used for validation. Unlike traditional cross-validation, time series cross-validation follows a sequential order, ensuring that future data cannot be used to predict the past and ensure that no data leak.

In the result of the times series cross-validation, there are two figures. Accuracy_score, which means the proportion of correct predictions made by the random forest model. The other one, F1_score, the harmonic mean of precision and recall. The result shows that the time series cross-validation accuracy mean score for the 10 features in the figure is 0.64 and the time series cross-validation accuracy mean score for all features in the figure is 0.65.

Table 5.1 Results of Different Features Selection.

Feature Set	Mean Accuracy	F1-Score
Selected (80% cumulative importance)	0.64	0.37
Full Features (all original)	0.65	0.38

It can be seen from the results that the random forest model shows comparable performance on the original data set and the dataset with selected features. Additionally, a large number of noise variables are eliminated, thus the feature selection is effective.

Our group final selected top 10 important features are 'log_lead_time', 'stays_in_weekend_nights', 'stays_in_week_nights', 'previous_cancellations', 'booking_changes', 'adr', 'required_car_parking_spaces', 'total_of_special_requests', 'deposit_type_Non Refund', 'customer_type_Transient'.

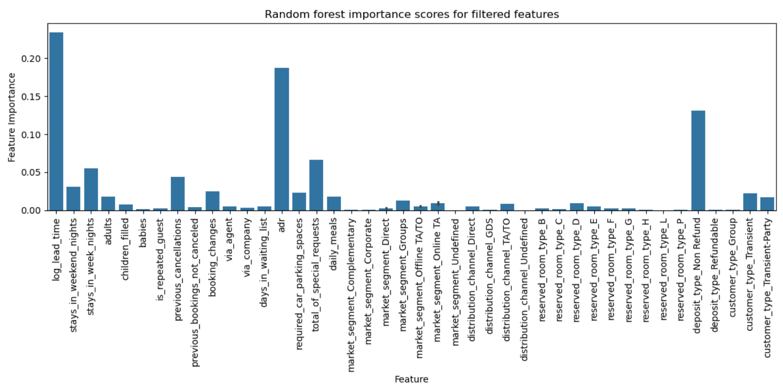


Fig 5.1 Importance of Different Features.

Since there may be autocorrelation among the top 10 important features, we use a Correlation Heatmap to examine the relationships between variables. As shown in Fig 5.2, the maximum correlation between any two variables is 27%, and the rest are even lower. Therefore, the autocorrelation is not significant, and all 10 features are retained without changes.

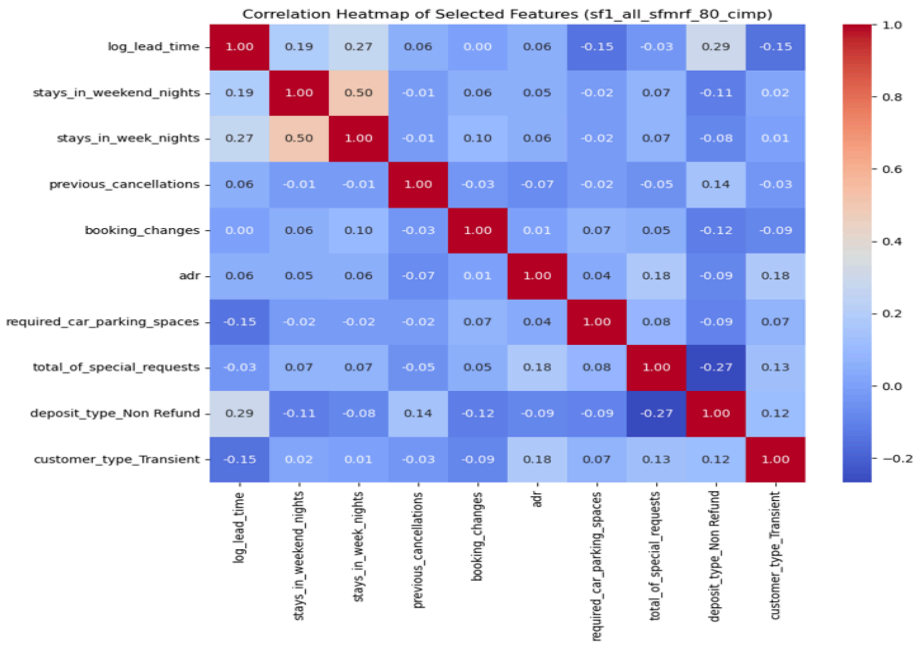


Fig 5.2 Illustration of the Correlation Heatmap of Selected Features.

6. Model Training & Hyper-parameter Tuning

After selecting the top 80% of the most critical feature variables, we began model training based on these features. We adopted two partitioning methods for the overall dataset to address two different model prediction strategies: requiring the model to predict data 30 days later (T=30) or data 90 days later (T=90), which are used for short-term and long-term predictions respectively. These two strategies align with patterns encountered in actual business operations, namely the need to predict whether guests will cancel orders in advance to allow sufficient time to take countermeasures.

We divided the complete dataset into training sets, validation sets and test sets. The training set was used for cross-validation to select the optimal parameters, the validation set was used to verify the accuracy of each model under the optimal parameter combination, and the accuracy on the test set will serve as the final evaluation criterion for the model's performance.

We considered this task as a binary classification task with some time factors. Since whether users cancel orders depends more on static time points than historical data, we used three classification models in total, rather than choosing any time-series models, especially because time-series models are more difficult to train than ordinary classification models. The three selected classification models are Random Forest, Logistic Regression, and LightGBM classifier.

Logistic Regression uses the Sigmoid function as the activation function before the final output to make binary classification, identifying the linear relationship between features and the target and distinguishing data on the hyperplane. Random Forest generates decision trees randomly, selects data and features randomly, and determines the final classification result through voting. LightGBM classifier achieves higher accuracy by sequentially optimizing a series of decision trees, with each new tree fitting the error of previous results. Theoretically, Logistic Regression is relatively simple to train and has strong interpretability; Random Forest is more robust to interference and less prone to overfitting; while LightGBM achieves the highest theoretical accuracy.

We set different parameter choices for each model to search for their optimal configuration: 1) For Logistic Regression, we set different regularization penalty coefficients, maximum number of iterations, and the solvers; 2) For Random Forest, we set different numbers of decision trees and the maximum depth of each tree; 3) For LightGBM, we set different numbers of decision trees, the maximum depth of each tree, the learning rate, the data sampling ratio, and the feature sampling ratio. The specific parameters and value options are shown in the following three tables. We use the same searching range for different T values:

Table 6.1 Hyper-parameters Searching Options for Logistic Regression.

Logistic Regression	C	Solver	Max iter	penalty
T=30/T=90	[0.01, 0.1, 1, 10, 100]	[liblinear, lbfgs]	[100, 200, 300]	L2

Table 6.2 Hyper-parameters Searching Options for Random Forest.

Random Forest	N estimators	Max depth
T=30/T=90	[100, 200]	[None, 10, 20]

Table 6.3 Hyper-parameters Searching Options for LGBM.

LGBM Classifier	N estimators	Max depth	Learning rate	Subsample	Colsample bytree
T=30/T=90	[100, 200]	[3, 5, 7]	[0.01, 0.1, 0.2]	[0.8, 1.0]	[0.8, 1.0]

For each set of parameters, we use GridSearchCV from the sklearn to search for the optimal parameter settings. It performs k-fold validation on the input dataset, testing a specific parameter configuration for each k-fold validation. For any model, a complete GridSearchCV will traverse every possible parameter combination and perform k-fold validation on each parameter configuration. Taking Table 4.1 as an example, for logistic regression, each optimal parameter searching requires testing 30 parameter combinations.

Through experiments, we obtained six different sets of optimal parameters, respectively for two data partitioning methods (T=30 and T=90) and three different models. The detailed parameter configurations are shown in the table below:

Table 6.4 Best Hyper-parameter setting for Logistic Regression

Logistic Regression	C	Solver	Max iter	penalty

T=30	1	lbfgs	200	L2
T=90	100	lbfgs	100	L2

Table 6.5 Best Hyper-parameter setting for Random Forest

Random Forest	N estimators	Max depth
T=30/90	100	10

Table 6.6 Hyper-parameters Searching Options for LGBM.

LGBM Classifier	N estimators	Max depth	Learning rate	Subsample	Colsample bytree
T=30/T=90	200	5	0.01	0.8	1.0

The tables above show that for LGBM and Random Forest, the hyperparameters required to predict data at different time intervals are basically the same. However, for logistic regression tasks, a stronger regularization coefficient and fewer iterations are needed. In sklearn's Logistic Regression module, the regularization coefficient is the reciprocal of C, so a larger C indicates a weaker regularization. A stronger regularization means stronger constraints on the model parameters during training to prevent excessively large parameters from generating overly strong feature collection capabilities, leading to overfitting. In this task, predicting data 30 days later is significantly easier than predicting data 90 days later, making it more prone to overfitting, especially since the logistic regression model is inherently less robust to noise than LGBM and Random Forest models. This argument aligns with the experimental results: the regularization coefficient for logistic regression is 1 at T=30, while it reaches 100 at T=90. In the optimal parameters of LGBM, a larger number of decision trees and a smaller learning rate are mutually reinforcing: more decision trees result in stronger feature extraction capabilities, while a smaller learning rate often requires more trees, thus ensuring the model's generalization ability.

We compared the accuracy on different datasets using the optimal hyperparameter combinations for each model on the validation set. During the validation phase, we used F1 score and accuracy as evaluation metrics, but in the testing phase, we also compared the model's ROC-AUC curve and PR-AUC curve. The four evaluation metrics mentioned above all reflect the model's classification accuracy, but there are subtle differences. Accuracy reflects the classification results in a relatively balanced way, that is, the proportion of data correctly classified out of the total number of predictions, treating all classes equally. Therefore, it sometimes cannot objectively describe the model performance when the class distribution is imbalanced. The F1 score focuses on the classification accuracy of the positive class and is more suitable for scenarios with imbalanced classes. The ROC-AUC curve focuses on the model's generalization ability to different classes under relatively balanced class conditions. The PR-AUC curve, similar to the F1 score, focuses more on the accuracy of positive samples, showing the trade-off between precision and recall for positive sample predictions.

In this task, order cancellations accounted for 37% of all samples, thus exhibiting a slight class imbalance. Therefore, metrics such as the F1 score can comprehensively evaluate the model's performance. In the validation phase, we compared the performance of different models on different datasets. The comparison results are shown in the table below:

Table 6.7 Evaluation of different models on different datasets

Model	Accuracy	F1 score
T30 Logistic Regression	0.6935	0.6844
T90 Logistic Regression	0.7427	0.7203
T30 Random Forest	0.7196	0.6674
T90 Random Forest	0.7481	0.6904
T30 LGBM Classifier	0.7198	0.6440
T90 LGBM Classifier	0.7350	0.6819

Of the six models above, the Random Forest model, which predicts 90 days in advance, achieved the best accuracy, while the Logistic Regression model, which predicts 90 days in advance, achieved the best F1 score. Overall, since the F1 score better assesses model performance under class imbalance, it may be more objective and important than the Accuracy value. It is worth noting that on both evaluation metrics, all models with T=90 performed better than those with T=30. This indicates that the further in advance users book hotels, the better we can predict their behavior, bringing more convenience to hotel operations.

7. Model Evaluation

In the previous analysis, we obtained the optimal parameters of each model and their performance on the validation set. Finally, to identify the optimal model for hotel cancellation prediction, we evaluated the practical predictive capabilities of all models on a unified test set. Specifically, we adopted Accuracy, F1-score, ROC-AUC, and PR-AUC as evaluation metrics for each model and compared their performance. The results are presented in Table 1.

Table 7.1 Performance of different models on test set.

Model	Accuracy	F1-Score	ROC-AUC	PR-AUC
T30_LR	0.720034	0.566112	0.773115	0.654277
T30_RF	0.715189	0.505848	0.782458	0.661203
T30_LGBM	0.702970	0.431452	0.753387	0.609134
T90_LR	0.719402	0.570600	0.773669	0.650080
T90_RF	0.710975	0.491852	0.776455	0.653443
T90_LGBM	0.702970	0.431452	0.751890	0.607364

For hotels, it is necessary to both reduce false negatives of canceled bookings which result in room vacancies and minimize false positives of non-canceled bookings which leave guests without accommodation. Thus, F1-Score is the most critical evaluation metric, as it directly reflects the balance between these two types of misclassifications.

After comparing the six models, we ultimately selected the Logistic Regression model with T=90 as the optimal model for hotel cancellation prediction, for the following specific reasons:

1) The overall performance of LightGBM is poor. Both its short-term and long-term prediction models rank the lowest across the four evaluation metrics, making it unsuitable for our business scenario.

2) The ROC-AUC and PR-AUC of Random Forest are slightly higher than those of Logistic Regression, but these metrics lack practical significance in our business scenario. In contrast, regarding the core metric F1-Score, Random Forest's short-term model achieves 0.506 and its long-term model 0.492, both of which are lower than those of Logistic Regression. This indicates that Logistic Regression is more suitable than Random Forest for the hotel cancellation prediction scenario.

3) When comparing Logistic Regression models with different prediction horizons, their performance is very close. We thus focus primarily on the core metric: the F1-score of the model with T=90 is 0.5706, which is 0.0045 higher than that of T=30. Although this improvement is modest, it carries significant implications for hotels: each reduction in false negatives (missed canceled bookings) corresponds to avoiding the loss of one room's revenue, while each decrease in false positives (misclassified non-canceled bookings) reduces one guest complaint. This 0.0045 increase in F1-score directly translates to "cost reduction + reputation enhancement." Additionally, the long-term model (T=90) provides predictions 90 days in advance, granting hotels a longer response time and more actionable measures to implement. Therefore, the T=90 Logistic Regression model is deemed the most suitable

choice.

8. Conclusion

To address the business problem of predicting hotel reservation cancellations, we selected the T=90 logistic regression model with parameters {“C”: 100, “max_iter”: 100, “penalty”: “l2”, “solver”: “lbfgs”} as the optimal solution.

Through the above analysis, we have identified ten key variables influencing hotel reservation cancellations, specifically including:

- log_lead_time：The number of days between booking and check-in
- stays_in_weekend_nights：Number of weekend nights
- stays_in_week_nights：Number of week nights
- previous_cancellations：Number of previous bookings cancelled by the customer prior to the current booking
- booking_changes：Number of reservation changes
- adr：Average Daily Rate
- required_car_parking_spaces：Number of parking spaces required by the customer
- total_of_special_requests：Number of special request
- deposit_type_Non Refund：deposit_type
- customer_type_Transient：customer type of Individual guest booking without contract

In response to the above key variables, we provide corresponding recommendations to help the hotel to optimize its operations, as detailed in the Table 5.2 :

Table 8.1 Some strategies for features

Features	Strategy

log_lead_time	1. Establish a reasonable advance booking window to accommodate varying operational requirements 2. Implement a secondary confirmation process for bookings made well in advance of the check-in date
stays_in_weekend_nights	1. Optimize weekend room allocation
stays_in_week_nights	1. Optimize weekly room allocation
previous_cancellations	1. Differentiated deposit and booking policies tailored to users based on cancellation behaviour 2. Promotional offers for users maintaining consistent and stable bookings
booking_changes	1. Establish booking changes policy, specifying free amendment allowances 2. Mark high-frequency change orders for advance coordination of room availability and service resources
adr	1. Design differentiated discount plans and cancellation policies for room types at varying price levels 2. Offer value-added services to high-priced customers
required_car_parking_spaces	1. Optimize the allocation of parking space resources
total_of_special_requests	1. Coordinate resource allocation in advance to ensure services are delivered effectively
deposit_type_Non Refund	1. Design a tiered non-refundable deposit plan 2. Provide value-added services for users with non-refundable deposits
customer_type_Transient	1. Operate base on preferences of the customer segment

References

[1] Antonio, Nuno, Ana de Almeida, and Luis Nunes. "Hotel booking demand datasets." Data in brief 22 (2019): 41-49.

[2] Peng, Hanchuan, and Chris Ding. "Minimum redundancy and maximum relevance feature selection and recent advances in cancer classification." Feature Selection for Data Mining 52 (2005).

[3] Radovic, Milos, et al. "Minimum redundancy maximum relevance feature selection approach for temporal gene expression data." BMC bioinformatics 18.1 (2017): 9.

[4] Zhao, Zhenyu, Radhika Anand, and Mallory Wang. "Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform." 2019 IEEE international conference on data science and advanced analytics (DSAA). IEEE, 2019.