1 Description of the Dataset

For this project, we analyze a comprehensive dataset [1] which contains customers' information collected from several hotels. For this dataset, the dimension is 119390×32. Within the 32 features, *is_canceld* is what we want to predict. It is 1 if a customer canceled the reservation before checking in and 0 otherwise. The ratio between label 0 and label 1 on the whole dataset is $0.77 : 0.23$ so it is kind of unbalanced. So we need to use more metrics other than only using accuracy to measure the performance of the models and set a baseline by simply predicting all the test samples as 0. For the remaining 31 features, they have different data types including integer, float, and string. More details can be found in Table 1 in Appendix.

For features with data type string, we are curious about their content. Since this is a real-world dataset, so they may not only contain categorical data like month and room type, but also have user generated natural language data like comments. Different methods should be used to pre-process these two kinds of string data. Luckily, in this dataset, all the string-type features can be regarded as categorical data. However, there is still a problem that for feature *country* and *reservation_status_date*, the numbers of label are extremly large, which are 174 and 923 respectively. As a result, if we use straightforward approaches, such as dummy variables to handle these two features, it's important to highlight that this would markedly increase the dataset's dimensionality since dummy variables create binary columns for each category within a categorical feature.

The consequence of this substantial increase in dimensionality could lead to challenges in subsequent analyses and modeling. It may result in the curse of dimensionality, where the model's performance deteriorates due to increased computational complexity and the sparsity of data points in the high-dimensional space. Additionally, higher dimensionality can lead to overfitting, as the model might start capturing noise in the data rather than the underlying patterns. Therefore, we might use some other methods to deal with these two features, which will be discussed later in Section 5.

2 Problem Statement

2.1 The Problem

The hospitality industry has undergone significant transformations in recent years, with the advent of online booking platforms providing consumers with unprecedented convenience and choice. Despite these advancements, one persistent challenge faced by hotels is the issue of booking cancellations. Understanding and predicting customer behavior in this context is of paramount importance for optimizing operations and enhancing customer satisfaction. By identifying key features and relationships, our goal is to develop accurate classification models such as logistic regression, KNN (k-NearestNeighbor), SVM (Support Vector Machine), decision tree, and neural networks, that assist hotel management in proactively managing reservations and providing a more seamless booking experience.

In this project, we presented the methodology employed, the features considered, and the results obtained from our predictive models. By shedding light on the factors influencing booking cancellations, this research contributes valuable insights to the hospitality industry, ultimately supporting the development of strategies to reduce cancellations and optimize hotel operations.

2.2 The Evaluation Metrics

To comprehensively evaluate the performance of the models we will build, we used five metrics: accuracy (with range 0-1), recall (0-1), precision (0-1), F1-score(0-1), time cost (>0). As mentioned above, for this project, our goal is to predict whether customers will cancel their reservations. If the model predicts that a customer will cancel the reservation with a high probability, the hotel can implement some methods to decrease such probability like asking the customer to make a deposit for this reservation. However, this will result in a decrease in customer experience. As we know, recall is defined as

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{1}$$

which indicates the capability of a model to find out all the positive samples while precision, which is defined as

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2}$$

evaluates the ability of making prediction precisely. So in the scenario of this project, recall is connected with the benefit of the hotel as they want to figure out all the customers that will cancel the reservation while precision indicates

[1] `https://www.kaggle.com/datasets/urmilsojitra/hotel-booking-cancellation`

customer experience since no one wants to be classified into the group that will cancel the reservation by mistake and pay a deposit. As a result, we wanted to find the model with a high recall and a not low precision. So we prioritized evaluating the model performance based on recall, followed by F1-Score, which is defined as

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision + Recall}} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} \tag{3}$$

This allows us to safeguard hotel interests while minimizing the impact on user experience as much as possible.

3 Main Section:

3.1 Pre-processing

Before modeling, we needed to first pre-process the dataset. To start with, we calculated the missing rate for each feature. There are only three features that contain missing values, see Table 2. With respect to *company*, as the missing rate is higher than 0.9, we deleted it directly. For *country* and *agent*, we could either fill the missing value by some state-of-art algorithms or delete the missing value directly. For this dataset, since the number of samples is enough and the missing rate is not high, so we chose to delete rows with missing values.

Then we changed the feature *arrival_date_month* from string to integer by using 1 to 12 respectively. What's more, we also dropped features that will lead to information leak like *assigned_room_type* and *reservation_status*. This is easy to be understood since the assigned room type can be known only if the customer have checked in. And the reservation status even directly indicates whether the customers have canceled their reservation. Besides, we also did standarization on the data.

3.2 Models

In this part, we are going to compare the performance of baseline and five different models, including

- Logistic Regression
- SVM
- KNN
- Decision Tree
- Neural Network

As you can see, some of these models have certain hyper-parameters that should be choosen before modeling. For threshold of logistic regression, we chose it by plotting out the ROC (Receiver Operating Characteristic) curve. For other hyper-parameters, we chose them by 5-fold Cross Validation (CV).

3.2.1 Logistic Regression

~~Logistic Regression is a statistical method utilized for classification tasks, estimating the probability of a data point belonging to a specific class through the logistic function. With respect to the selection of threshold for logistic regression, we first set the threshold to 0.5 and then plotted out the ROC curve. Next, we chose the threshold that corresponds to the point that is the nearest to the upper left corner on ROC curve. The result are given below in Fig. 1 in Appendix. So we set the threshold for logistic regression as 0.4.~~

3.2.2 SVM & KNN & Decision Tree

SVM is a powerful supervised machine learning algorithm can be used for classification tasks, aiming to find an optimal hyperplane that best separates different classes in a high-dimensional space. KNN is also an algorithm that can be used to deal with classification tasks, relying on the proximity of data points to make predictions. It assigns a new data point to the majority class or computes a weighted average based on the k-nearest neighbors in the feature space. Decision Tree is a tree-like model used for both classification and regression tasks, where each internal node represents a decision based on a feature, leading to subsequent nodes until a final prediction is reached at the leaf nodes, making it interpretable and easy to understand.

For these three models, we used 5-fold CV to select the optimal hyper-parameters. Besides, we also guaranteed that all the chosen hyper-parameters do not locate at the boundary. For SVM, the optimal kernel is rbf with C = 2000 and gamma = 0.00001. For KNN, the optimal number of neighbors K = 3. For decision tree, we used gini index as the

criterion and set the min_samples_leaf to be 1, min_samples_split to be 2, max_depth to be 'None', which means nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. Besides, all these optimal parameters are guaranteed not to locate on the boundary.

3.2.3 Neural Network

Neural Network is a powerful machine learning model inspired by the human brain, consisting of interconnected nodes organized in layers, capable of learning complex patterns and relationships in data for tasks such as classification. It is a little different from models mentioned above. For this project, since neural network is already a complex enough model, so even a neural network with the most simple structure can perform much better than other methods. As a result, compared to hyper-parameter selection, we were more concerned about the overfitting issue. We can use methods like early stop, drop out, regularization and etc. to deal with the overfitting problem in neural network. Fortunately, in this project, our neural network performs well on both training data and test data, which indicates that our model is not overfitted. The structure of our network is shown below in Fig. 2. We used two hidden layers with dimension 100 and 50 respectively with activation function ReLU. For the output layer, we used sigmoid function since this is a binary classification problem. Besides, the loss function we used is binary cross entropy and the optimizer is Adam.

3.2.4 Performance

Table. 3 displays the performance of different models. Here the baseline is simply predicting all the samples as 0. From Table. 3, we can see that the best model for predicting whether customers will cancel their reservations is neural network, which aligns with our expectations. This can be attributed to the large size of the considerable number of samples. Moreover, the relationships between features are not necessarily linear. Neural networks excel in capturing complex relationships between features and selectively learning relevant patterns, allowing them to outperform other models. However, we are still not stisfied with the result as the recall is lower than 0.95. We want to further enhance the model performance. So we also did feature engineering on this dataset, which will be discussed immediately in the next section.

4 Feature Engineering

From the previous section, we know that the best model is Neural Network. Although we know that neural network is good at capturing the complex relationships between features, this dataset is too sparse to be analyzed if we use dummy variables for all the categorical data. Besides, the high dimensionality will also lead to high time cost. As a result, we want to perform dimensionality reduction on this dataset.

At first, we proposed to implement feature selection by Lasso regression and feature extraction by PCA to reduce the dimension. Later, after carefully checking the content of the categorical features, we found that the high dimensionality is mainly caused by two two features, *country* and *reservation_status_date*. So in our perspectives, if we can reduce the number of labels for these two features, the dimension of the whole dataset will be reduced significantly. To test the performance of these dimensionality reduction methods, we used the dataset returned from these ways to train the Neural Network gotten in previous section.

4.1 Feature Selection and Feature Extraction

For feature selection, we first use dummy variables for all categorical features. Then we used Lasso regression to figure out features with non-zero parameters. And we only kept these features to get our new dataset. With respect to feature extraction, we only use PCA here. We do not use Kernel PCA since the dimension of our dataset is quite large, which will lead to the crash of Colab session. We set the threshold to be 0.95 and get our new dataset. Finally, we used our new dataset to train the model and get the results in Table. 4. We can see that the performance of the original model is the best. This result is not satisfactory so that we want to reduce the dimension by other methods.

4.2 Country

Beyond the feature selection and feature extraction methods mentioned above, we can also reduce the dimensionality by reducing the number of dummy variables needed for feature *country* and feature *reservation_status_date*. In this part, we want to deal with *country* by using only the most important countries, creating new features that can explain the variance caused by *country* to replace it, or using word embedding and clustering to group these countries and label them respectively by the result of clustering. First, we outputted the four most important countries by decision tree.

They are Portugal (PRT), Spain (ESP), Germany (DEU), and France (FRA). We gradually add the number of countries used and test their performance respectively.

Second, we thought the variance of *country* are mainly caused by their corresponding GDP and location. So we replace *country* by their GDP in 2015 and continents respectively. But this method has a drawback that these two new features are given by us subjectively, so they may not contain all the variance of *country*. So we want use word embedding to capture the information contained in a country.

We first used API of OpenAI to do word embedding. As the embedding dimension is 1536, we then implemented Kernel PCA to reduce the dimension. Next we did clustering on the low dimension data and aimed to label the countries by the result of clustering. To chooss which dimension we want to reduce to and how many clusters to use, we plotted the elbow plot like Fig. 3. We then tested the optimal number of clusters in dimension 1, 2, 3 respectively and finally found that dimension 2 with 5 clusters can guarantee our model to have the highest recall. We think it is reasonable to label the country in this way since when OpenAI was training the word embedding model, the content in the vector of each country name has already contained the context information, which covers the religion, economics, politics, culture, and etc. of a country.

From Table. 5, we can summarize that with word embedding and clustering, the model has the highest recall. So we will generate a new dataset in this way. If you prefer the integral performance, using the most important three countries to replace *country* is a better choice.

4.3 Reservation Status Date

For *reservation_status_date*, we found that data in this feature are actually some dates. So we can extract the year, month, and day. We tried to keep only year, and year with month, and all these three respectively. From Table. 6, we can see that the model performance is the best if we keep all these three.

5 Business Insights

As for how can hotels benefit from our model, we proposed three suggestions. First, since the recall of our model is 0.9783, we can figure out most of the customers that will cancel the reservation. With these predicted results, hotels can take specific measures against customers who are likely to cancel their reservations, such as collecting a deposit at the time of booking or offering free room upgrades for timely arrivals during the off-peak season to reduce the revenue loss caused by cancellation.

Second, the high precision of this model also guarantees that we won't mistakenly predict a large number of guests who wouldn't cancel their reservations as potential cancellations. This significantly enhances the user experience by sparing them from many cumbersome measures that are only intended for users with a high probability of canceling reservations.

Third, as our model exhibits both high recall and high precision, resulting in a high F1-score, ensuring that we can largely trust its predictive outcomes. The model accurately identifies the majority of individuals who are likely to cancel reservations while minimizing false positives. This allows hotels to adopt more aggressive and higher-risk strategies based on our model's predictions. Following the example of airlines, hotels can oversell rooms according to booking forecasts. This approach not only minimizes potential losses for the hotel but also preserves the customer experience. Even for customers predicted to cancel, no deposits are collected, and full refunds are approved upon cancellation requests, thereby maintaining a positive customer.

6 Conclusion

To sum up, the Neural Network with structure shown in Fig. 2 emerged as the most effective model for predicting reservation cancellations, leveraging its capacity to capture complex relationships in a large and sample-rich dataset. However, with a recall slightly below 0.95, further enhancements were sought through feature engineering. Various dimensionality reduction methods were explored, including Lasso regression, PCA, and replacing *country* by other features, but the most impactful approach involved label different countries by the result given by word embedding for country names and clustering. With a nearly perfect performance, this model can help hotels strategically address cancellations, offering targeted measures without compromising the user experience, and presenting opportunities for more proactive and risk-aware operational strategies.

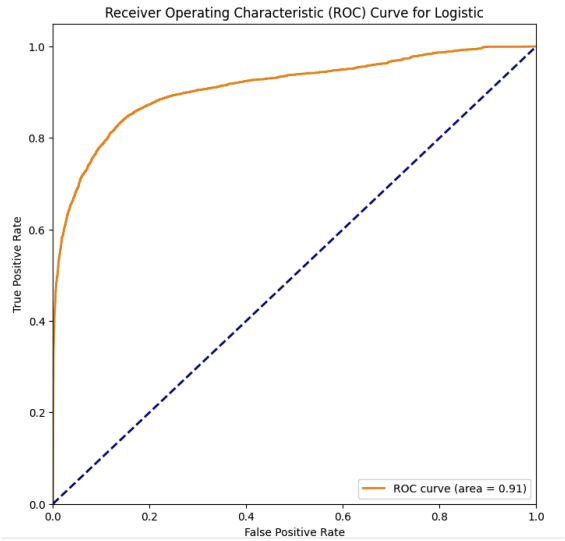
7 Appendix

Table 1: Feature Data Types

Data Type	Count
Integer	15
Float	4
String	12

Table 2: Missing rate

Feature	Missing Rate
Country	0.0041
Agent	0.1369
Company	0.9431



~~Figure 1: Roc Curve for Logistic and Lasso Regression~~

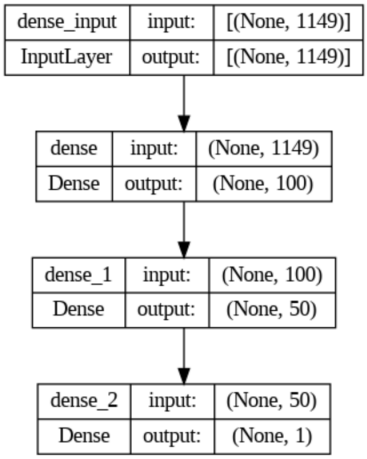


Figure 2: Structure of Neural Network

Table 3: Performance of Different Models

Pre-processing	Accuracy	Recall	Precision	F1-Score	Time Cost (s)
Baseline	0.6079	0.0000	0.0000	0.0000	0.0001
Logistic	0.8275	0.8752	0.7353	0.7992	**6.9225**
KNN	0.8363	0.7567	0.8129	0.7838	38.3395
SVM	0.8205	0.7202	0.7991	0.7576	2312.1248
Decision Tree	0.8623	0.8152	0.8306	0.8228	9.6647
Neural Network	**0.9639**	**0.9411**	**0.9660**	**0.9534**	5s/Epoch

Table 4: Performance of Neural Network on Dataset with Different Pre-processing Methods

Pre-processing	Accuracy	Recall	Precision	F1-Score
Original	**0.9639**	**0.9411**	0.9660	**0.9634**
PCA	0.9603	0.9261	**0.9714**	0.9482
LASSO	0.8991	0.7784	0.9563	0.8582

Table 5: Performance of Neural Network on Dataset with Different Pre-processing Methods on Country

Pre-processing	Accuracy	Recall	Precision	F1-Score
Original	0.9639	0.9411	0.9660	0.9634
Delete	0.8860	0.8494	0.8584	0.8539
PRT	0.9706	0.9602	0.9646	0.9624
P+ESP	0.9702	0.9520	0.9713	0.9616
P+E+DEU	**0.9715**	0.9487	**0.9780**	**0.9631**
P+E+D+FRA	0.9711	0.9462	0.9791	0.9624
GDP	0.9706	0.9488	0.9757	0.9620
Continent	0.9712	0.9512	0.9748	0.9629
GDP+Continent	0.9676	0.9488	0.9682	0.9584
Word Embedding	0.9705	**0.9530**	0.9712	0.9620

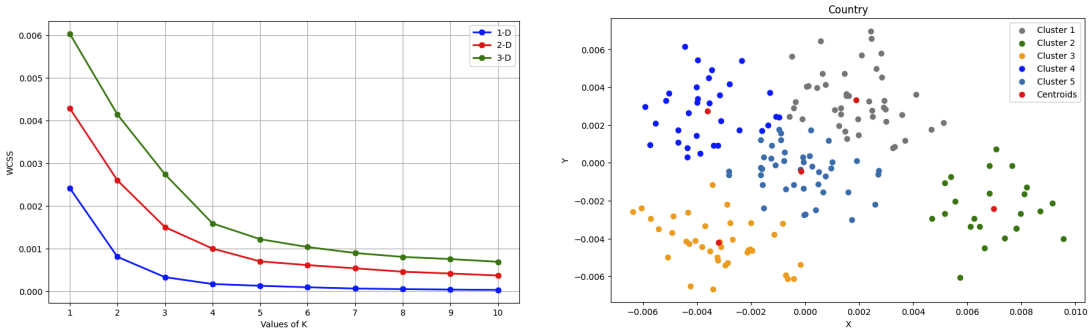


Figure 3: Elbow Plot(Left) and Clustering Result(Right)

Table 6: Performance of Neural Network on Dataset with Different Pre-processing Methods on Reservation Status Date

Pre-processing	Accuracy	Recall	Precision	F1-Score
Original	0.9639	0.9411	0.9660	0.9634
Delete	0.8475	0.8105	0.8025	0.8065
Year	0.8598	0.7644	0.8623	0.8104
Y+Month	0.9367	0.8826	0.9524	0.9162
Y+M+Day	**0.9984**	**0.9978**	**0.9982**	**0.9980**