

**Project Title:**

Optimising Hotel Reservation Management for Revenue Maximization and Overbooking Prevention

**Section & Team No:**

G2T2

**Prepared for:**

Professor May Wang

| **Name:** | **Admission No.:** |
| --- | --- |
| Adrianus Tjoatja Widjaja | 01391444 |
| Davin Shee | 01392785 |
| Eng Jun Hong | 01393746 |
| Hannah Loh Qiaoan | 01443701 |
| Mohammad Nursyamiir Bin Othman | 01374623 |
| Nurul Khalisah Bte Kamaruzaman | 01394824 |
| Rachel Ooi | 01404562 |

**Table of Contents**

[**1. Introduction 3**](#_6t5g684kktpf)

[**2. Literature Review 5**](#_1aj7369wtfdu)

[**3. Dataset 11**](#_rwrdrv5c3wt1)

[3.1. Data Preprocessing and Exploratory Data Analysis 11](#_4mmnqhtlqf6s)

[3.1.1 Missing Values 11](#_ncdezfxn4pqb)

[3.1.2 Identifying Relationships 12](#_vo9rivkg6neu)

[**4. Methodology 14**](#_uvvemljpx3ro)

[4.1. Feature Selection 14](#_pbym26vkmbg6)

[4.1.1 Chi-Square Test of Independence 14](#_yy84vnyxvfcb)

[4.1.2 Correlation Coefficient 15](#_xmmu1qvl6q2z)

[4.1.3 Final Features Selected 15](#_ey34glxxeqec)

[4.2. Further Data Preprocessing of Post Feature Selection 16](#_9tx1kdv9pg6j)

[4.2.1 One-hot Encoding for Categorical Variables 16](#_jxof6elcalrx)

[4.2.2 Normalisation for Numerical Variables 16](#_9al0mrcc7wbf)

[4.3. Main Performance Metrics 16](#_8zhvjzes4ga2)

[4.3.1 Why Precision over Accuracy/Recall? 16](#_8sanc3mpsyff)

[4.5. Hyperparameter Tuning 17](#_ifqt6h2us0a1)

[4.5.1 Random Search 17](#_vgrrpzsbqbak)

[4.6. Data Modelling 17](#_s0vv03dukn3d)

[4.6.1 Logistic Regression 17](#_knshrpz4x3w)

[4.6.2 Decision Tree 20](#_w4aryraz3vc8)

[4.6.3 Random Forest 22](#_gdo5ziqly9qi)

[4.6.4 Extreme Gradient Boost (XGBoost) 24](#_h4vgpzaxuqjz)

[4.6.5 FeedForward Neural Network (FNN) 26](#_pzfhnf0evj2)

[4.6.6 Support Vector Machine (SVM) 30](#_kzzd6jv042qh)

[**5. Results & Discussion 32**](#_w82whu4p2dq1)

[5.1 Voting Ensemble 33](#_o35w42m0m1tm)

[**6. Future Work and Conclusion 35**](#_c07bs3pmyuv4)

[6.1. Conclusion 35](#_nkz2cjhtl6w0)

[6.2. Future Work 37](#_uaeg5c4f6esf)

[**7. References 38**](#_awivdb10pny3)

[**8. Appendix 44**](#_jsxgocrmzizt)

# 

# **1. Introduction**

## 1.1 Problem Statement

Maximising hotel revenue while preventing the pitfalls of overbooking is a complex yet critical challenge in the hospitality industry. The aim is to develop an advanced classification model that accurately predicts hotel reservation cancellations. This model will enable the establishment to strategically manage reservations, effectively allocate resources, and enhance overall revenue generation.

## 1.2 Context

The COVID-19 pandemic has had devastating impacts on the global economy (Mou, 2020). One of the industries that had been adversely affected was the tourism industry (Behsudi, 2020). That being said, in the past two to three years, the world has recovered considerably from the debilitating effects of COVID-19, particularly in various parts of the economy. As part of returning to the “new normal”, people around the world wanted to resume tours and travels around the globe. Indeed, after travel restrictions were gradually eased for various countries, the demand for travel has been booming (The Straits Times, 2023). With demand for travel, comes demand for travel accommodations. However, there are various problems that hoteliers face, when it comes to meeting demands from would-be travellers. One of the biggest issues that hotels face is that of booking cancellations. For hotels, their main business goal can be neatly encapsulated as “making the right room available for the right guest and the right price at the right time via the right distribution channel” (Mehrotra & Ruttley, 2006). Intuitively, “making the right room available” would entail striving to uphold good room inventory management practices. However, this is significantly hampered by the spate of cancellations that hotels have to deal with by customers. Consequently, this has a knock-on effect on the profitability of the hotel’s business, when hotels have to deal with revenue loss as a result of unsold rooms, decreased profit margins, and even damage to their social reputation (Noone & Lee, 2010). This leads to the hotel overbooking problem, overbooking occurs when a hotel accepts more reservations than it has rooms available, in anticipation of cancellations. While this strategy can boost revenue, it can also lead to customer dissatisfaction and a tarnished reputation if cancellations are not accurately predicted.

## 1.3 Motivation

As we have stated earlier, the key issue we are targeting is the revenue loss for hotels as a result of hotel booking cancellation. In light of the frenzy due to pent-up frustrations from the inability to travel during COVID-19 lockdowns, we can intuitively expect that the volume of bookings (and by extension, cancellations) would increase. Being able to solve a pain point that hotels face in the COVID-19 “endemic” era would contribute to greater economic recovery overall. There is thus a greater imperative to try and solve this problem. But apart from that, successfully dealing with this issue would result in greater operational efficiency overall (Hayes & Miller, 2011). Things like resource allocation, staffing, reducing waste, enhanced customer experience, are all potential benefits that can be obtained if the issue of predicting whether a customer is likely to cancel their booking can be successfully tackled. With this in mind, our group has decided, after taking reference from the vast literature on this issue, to build our own classification models, evaluate amongst them, in order to discover the best model that has the best performance.

# **2. Literature Review**

We recognise that the problem that is being attempted to solve is not novel, as can be seen from the number of papers available on this subject. Hence, we aim to value-add to the existing literature by selecting publications where we can not only learn from, in terms of methodology or findings, but also to point out the possible limitations of such studies. From here, we are then able to point out potential areas of exploration that we can then apply to our proposed models.

## 2.1. Predicting hotel cancellations using machine learning

Data

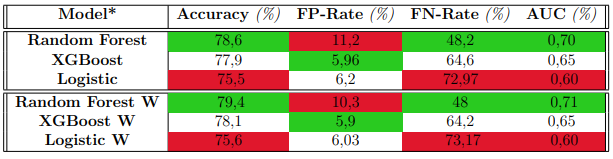
Obtained from a hotel base in Gothenburg, Sweden, between 2016-01-01 and 2020-12-31. During COVID-19 in 2020, customer behaviour changed significantly, thus they removed the affected observations. This resulted in a data-set with dimensions of 1,638,735 x 24. (Appendix, Table A&B)

Methodology

They further preprocessed the variables by re-categorizing or adjusting them. For variables with categories that contained less than 1% of the observations were merged, either with the most similar subcategory, or as a new subcategory ‘other’. This was done to avoid inaccurate conclusions. They also removed variables that were redundant as their information was contained in another variable and thus not needed. A 50-50 train-test split was used, training data is further processed into 5 folds using K-fold for cross validation. (Appendix, Figure C)

Models:

1. Logistic Regression
   1. Tuned parameters: none
2. Random Forest
   1. Tuned parameters: number of trees, max depth, number of features used in each split
3. XGBoost
   1. Tuned parameters: max depth, min split loss, learning rate.

Results

*Figure 1: Model performance, W in the last 3 rows indicates weather data included*

| **Findings** | **Area of Exploration** |
| --- | --- |
| Random Forest performed the best and the use of weather data increased the model’s accuracy. | Explore the models used on our dataset, and see if we are able to achieve the same pattern of results. |
| A total of 24 attributes were used, which raises concerns of overfitting, curse of dimensionality and high computational cost. | Training of models with lesser attributes while trying to obtain similar or higher performance. |

*Table 1: Findings and Area of Exploration*

1. **Prediction of hotel booking cancellations: Integration of machine learning and probability model based on interpretable feature interaction**

Data

The dataset consists of two sets of Personal Name Records (PNR) – one with 85,274 records from two hotels in Portugal, and the other with 4,238 records from a travel service company. After pre-processing, there are 85,274 samples with 27 independent variables in the first dataset and 4235 samples and 15 independent variables in the second dataset (Appendix, Figure E).

Methodology

The study proposes a model integrating Bayesian Networks (BNs) and Lasso Regression to improve performance in predicting hotel booking cancellations:

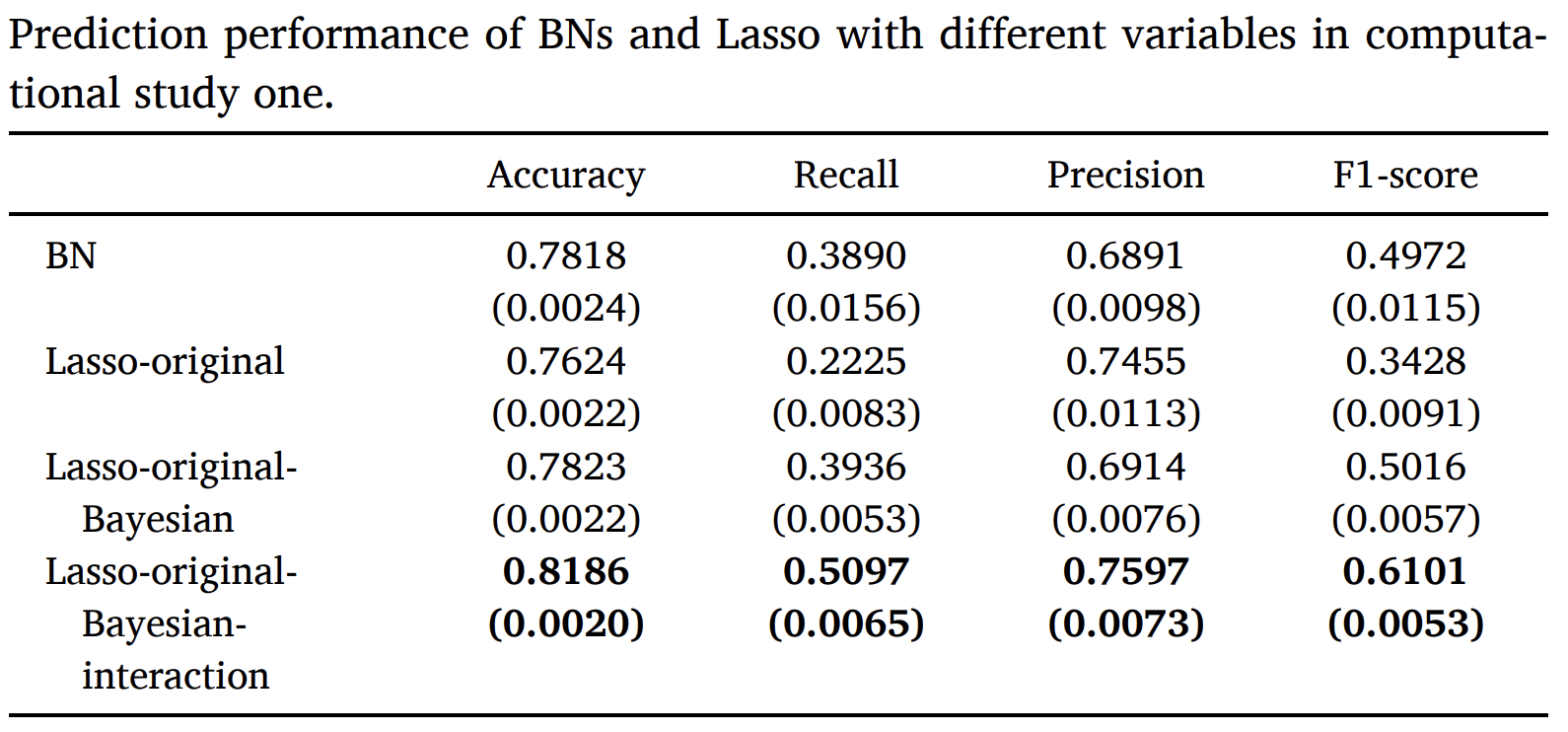
1. PNR data was collected and pre-processed
2. Original features were identified by manual extraction methods
3. Fuzzy multi-dimensional representation method employed, variables inserted into proposed feature interaction model
4. Bayesian Network model was built with determined structure and parameters, generating estimation results
5. Preceding features combined to train Lasso Regression, producing final prediction results and feature importance.

The first dataset is separated via a 70-30 train-test split, while the second dataset is separated via a 80-20 train-test split.

Models

1. Bayesian Network
2. Lasso Regression
3. Other models used for comparison include Artificial Neural Networks (ANNs) and Extreme Gradient Boosting (XGB)

Results



*Figure 2: Model performance for the first dataset, numbers set in boldface represent the best values.*

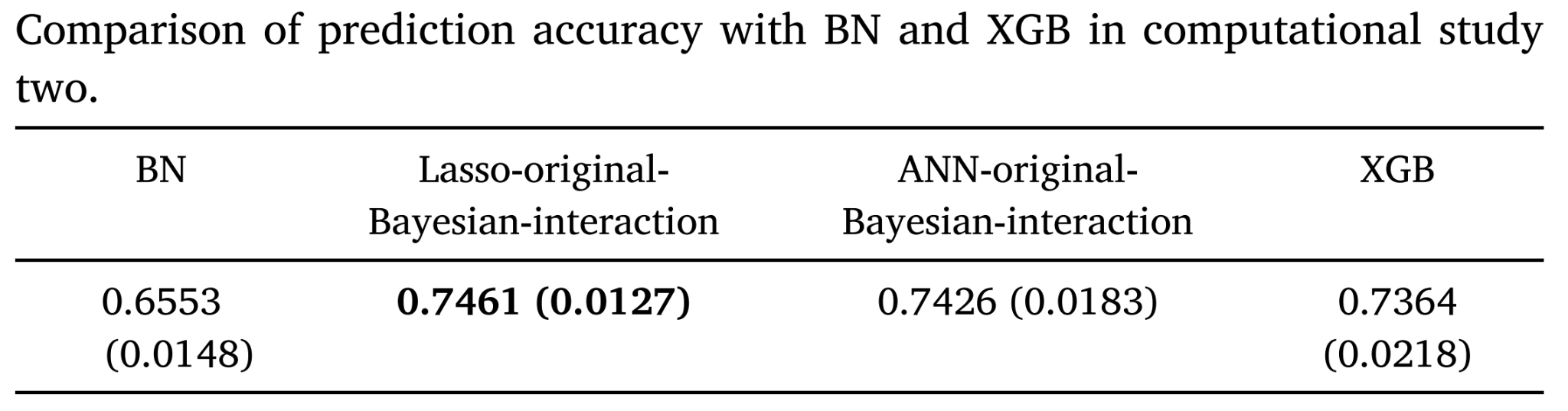


Figure 3: Model performance for the second dataset, numbers set in boldface represent the best values.

| **Findings** | **Area of Exploration** |
| --- | --- |
| The empirical results of the study show that integrating various ML models had led to the best prediction performance. | Explore the concept of combining multiple ML models to generate more accurate models similar to what was done in this study. |

*Table 2: Findings and Area of Exploration*

1. **Predicting hotel booking cancellations to decrease uncertainty and increase revenue**

Data

The data was obtained from 4 confidential hotels in Algarve, Portugal between 2013 to 2015. They had a total of 73,141 observations which they took directly from the hotels’ Property Management System(PMS) database. The attributes they ultimately used for the models are unknown, but the initial attributes from the PMS are listed in the appendix(Figure D&E).

Methodology

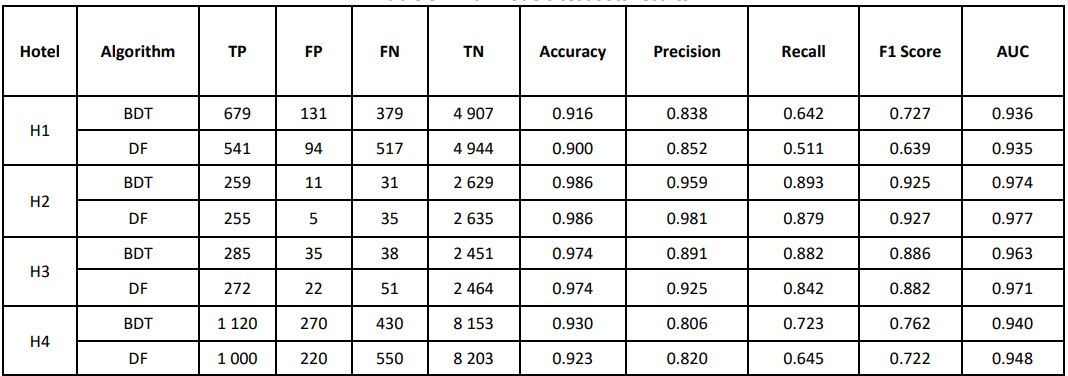
They first preprocessed the data, by using dataQualityR to check for missing values, outliers and skewed distributions. They then did feature engineering, followed by normalisation, but later found out that the original data performed better. They also used a mutual information feature selector filter to perform feature selection. They then employed a 70-30 train-test split for hyperparameter tuning and evaluation of their models. They then used K-fold cross validation of 10 folds to evaluate the tuned models. The parameters in which they hypertuned were not mentioned. They also used traditional machine learning algorithms like random forest (which they called decision forest algorithm).

Models:

1. Boosted Decision Tree
2. Decision Forest
3. Decision Jungle
4. Locally Deep Support Vector Machine
5. Neural Network

Results

These are the results of their hypertuned models. Results before hypertuning can be found in the appendix(Figure F)



*Figure 4: Final Model performance*

| **Findings** | **Area of Exploration** |
| --- | --- |
| Study shows that the use of traditional machine learning algorithms, such as random forest, when built properly with the right parameters and relevant features, have been proven to work well. | Instead of trying to explore many complex, deep learning algorithms, we should focus on improving the performance of traditional machine algorithms, which have proven to work well for classification problems. |

*Table 3: Findings and Area of Exploration*

# **3. Dataset**

The data was originally from an article titled “Hotel booking demand datasets” (Antonio et al., 2018). Subsequently, it was ported over to Kaggle where it was accessed and downloaded from (Mostipak, 2019).

The dataset consists of data for two different types of hotel, resort & city hotel. In totality, it consists of 32 attributes and 119390 records. The data is not synthetic, and was extracted from the databases of the hotels, with personal identifiable information removed. Working with real-life data is desirable and preferred, as we can have greater confidence in our methodology described below when it comes to future real-life applications.

The attributes as well as their description can be found in Figure D in the Appendix below.

## 

## 3.1. Data Preprocessing and Exploratory Data Analysis

### 3.1.1 Missing Values

Out of the whole dataset, 4 attributes have been found to have missing values, namely “children”, “country”, “agent”, and “company”. These attributes have 4, 488, 16340, and 112593 missing values respectively. We used various methods to handle these missing values according to their data type.

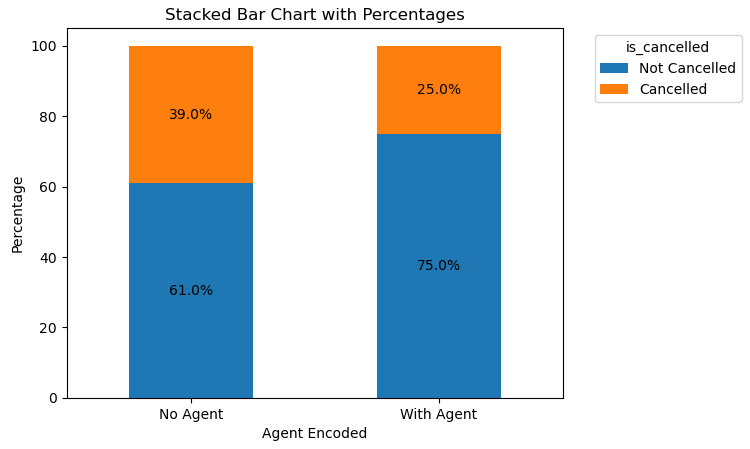
Since attribute “children” is a numerical variable, we used an imputation method, by filling in the missing values with the median of all the “children” values. As for the attribute “country”, since it is a categorical variable, we decided to remove all rows with missing values. Our rationale behind this was due to the small percentage of only 0.4% of all rows containing missing values for attribute “country”.

For attributes “agent” and “company”, missing values meant that the reservation was done without an agent and company respectively. We dealt with the missing values by creating new attributes labelled “agent\_encoded” and “company\_encoded” in order to split up the original attributes into binary form. This resulted in missing values being labelled “0” and filled values being labelled “1” to represent the following:

* 0: no agent; 1: with agent
* 0: no company; 1: with company

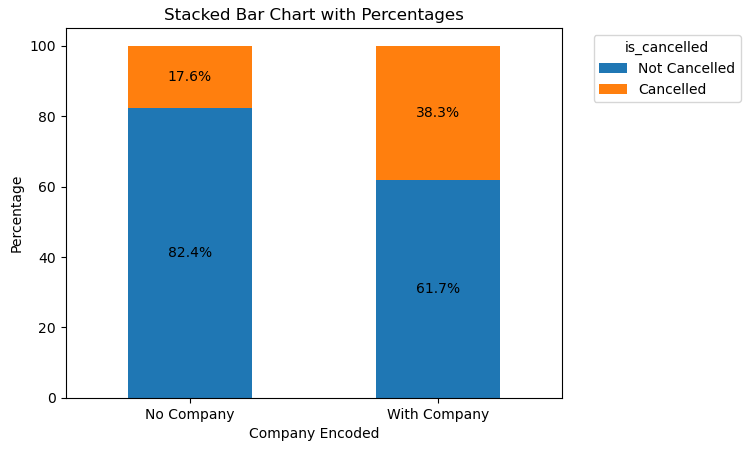
### 3.1.2 Identifying Relationships

After which, we plotted bar charts to better understand and identify relationships for “agent\_encoded” and “company\_encoded”.



*Figure 5: Bar chart on the distribution of booking cancellations with regards to the presence of an agent*

From Figure 5, we found that there was a higher percentage of cancellations when the hotel booking was made without an agent as compared to bookings made without one.

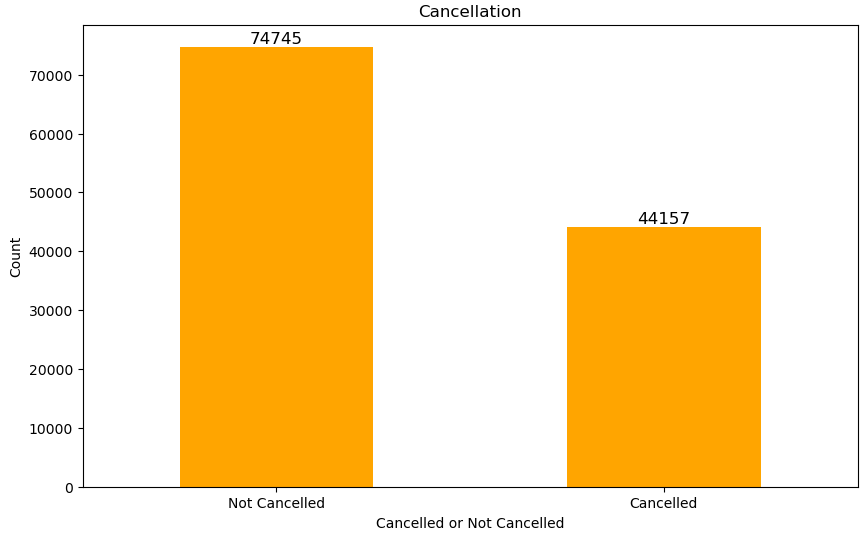


*Figure 6: Bar chart on the distribution of booking cancellations with regards to the presence of a company*

In contrast, from Figure 6, we found that there was a higher percentage of cancellations when the hotel booking was made with a company as compared to bookings made without one.

The importance of these two attributes “agent\_encoded” and “company\_encoded” is further explored in the following sections.

After cleaning our data, we then plotted a bar chart to visualise the distribution of cancelled bookings in our dataset.



*Figure 7: Bar chart on the distribution of booking cancellations*

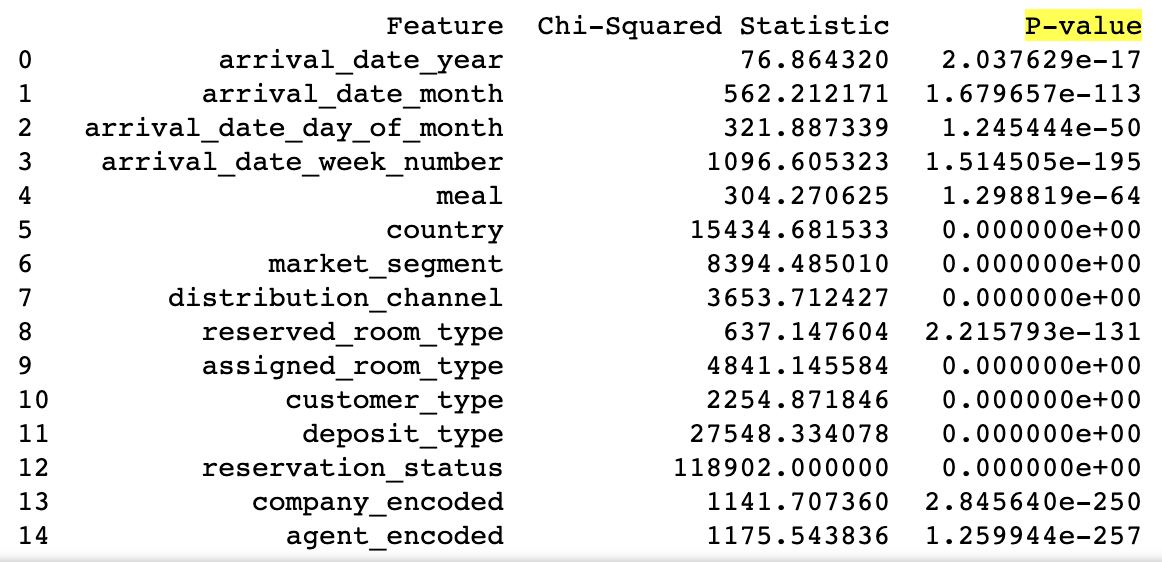
From Figure 7, we observed that the proportion of cancelled bookings and non-cancelled bookings is not too skewed.

# **4. Methodology**

## 4.1. Feature Selection

### 4.1.1 **Chi-Square Test of Independence**

The Pearson’s Chi-Square Test of Independence is a statistical test used for categorical data to determine the relationship between two categorical variables (Turney, 2023). The team tested the relationship between ‘is\_canceled’ and other categorical variables, and obtained the results in Figure 6:



*Figure 8: Chi-Square Test Statistics*

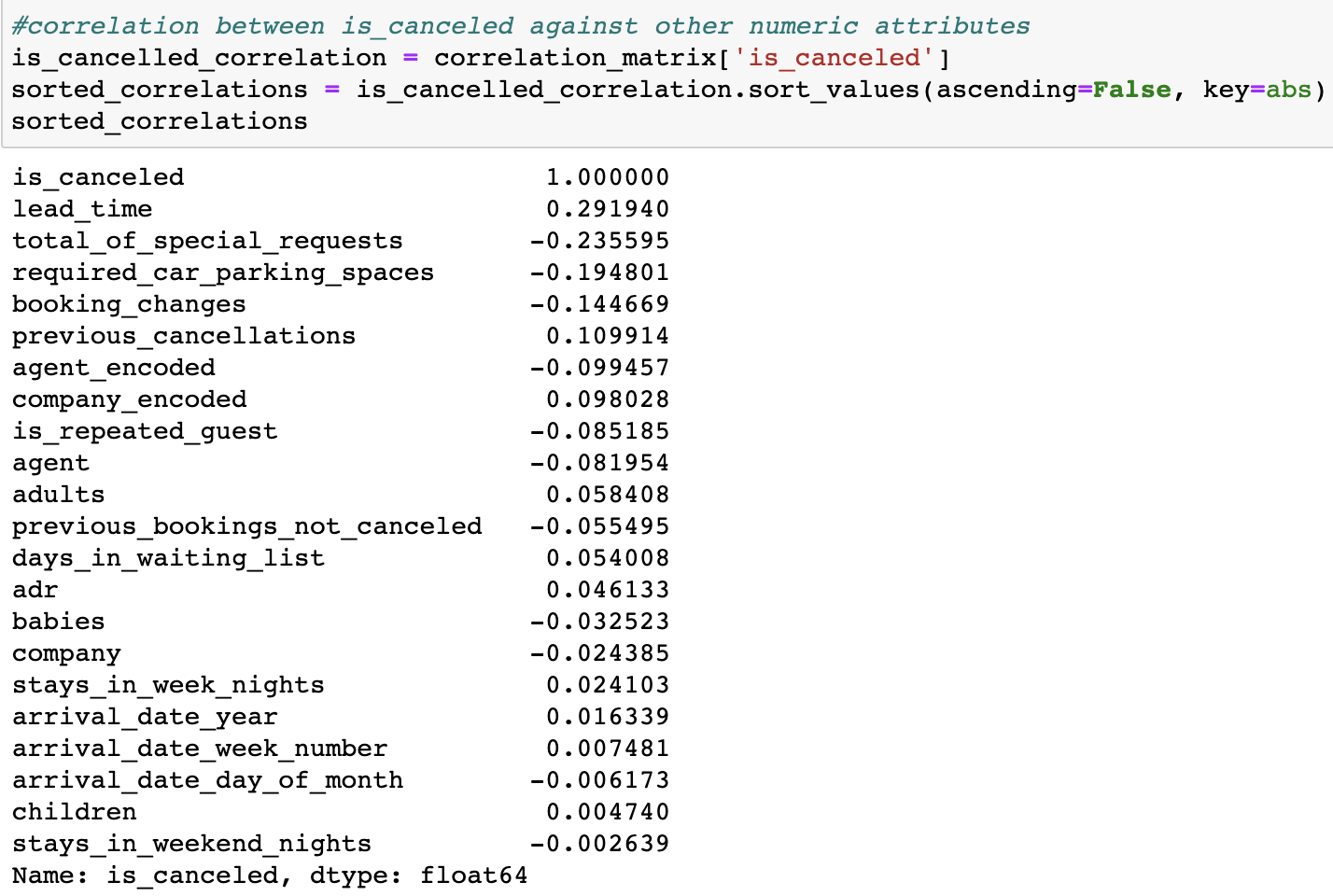
Based on Figure 8, it is crucial to select features that have obtained a relatively high Chi-Squared Statistic (more than 1000) and low p-value (near 0) as that indicates that therelationship they have with ‘is\_canceled’ is not independent.

The selected features are:

1. arrival \_data\_week\_number
2. country
3. market\_segmet
4. distribution\_channel
5. assigned\_room\_type
6. customer\_type
7. deposit\_type
8. reservation\_status
9. company\_encoded
10. agent\_encoded

### 4.1.2 **Correlation Coefficient**

The Pearson’s Correlation Coefficient is a numerical measure of the linear correlation between two variables (NGO, 2019). The team tested the correlation between ‘is\_canceled’ and other numerical variables, and obtained the results in Figure 7:



*Figure 9: Correlation Coefficient Statistics*

Based on Figure 9, we found that ‘lead\_time’ has the highest correlation of 0.29.

### 4.1.3 **Final Features Selected**

The team selected the features based on the analysis that we have performed above. The team has decided to remove ‘reservation\_status’ even though the feature had obtained a high Chi-Square value and low p-value. This is because the feature is a historical feature with 1 of these 3 values - ‘canceled’, ‘checked out’ or ‘no show’, which is directly related to the labelling of whether the reservation is cancelled or not.

Therefore, these are the 10 features that the team has selected to train the model:

1. arrival \_date\_week\_number
2. country
3. market\_segmet
4. distribution\_channel
5. assigned\_room\_type
6. customer\_type
7. deposit\_type
8. company\_encoded
9. Agent\_encoded
10. lead\_time

## 4.2. Further Data Preprocessing of Post Feature Selection

### 4.2.1 One-hot Encoding for Categorical Variables

With the final 10 features that the team selected, we performed one-hot encoding to deal with the categorical variables. This method was applied to categorical variables “country”, “deposit\_type”, “market\_segment”, “assigned\_room\_type”, “distribution\_channel”, and “customer\_type”. The breakdown of the variables after conducting one-hot encoding can be found in Table C in the Appendix.

### 4.2.2 Normalisation for Numerical Variables

We noted that there were 2 non-binary attributes, lead time and arrival\_date\_week\_number, amongst the attributes we selected. We decided to do Z-score normalisation on these 2 attributes. This is so that it will prevent possible bias when used in our models. This also helps with algorithms that use distance-based metrics in their computations for a fair comparison between features.

## 4.3. Main Performance Metrics

### 4.3.1 **Why Precision over Accuracy/Recall?**

Specific to our problem, in order to circumvent revenue losses due to cancellations, the hotel may adopt overbooking strategies to manage room occupancy (C.-C. Chen et al., 2011; C.-C. Chen & Xie, 2013; Mehrotra & Ruttley, 2006; Smith et al., 2015; Talluri & Van Ryzin, 2004).

However, they run the risk of having to deny guests with confirmed reservations due to overbooking. This may then result in severe reputational harm and revenue (Noone & Lee, 2011), and even potential future revenue loss because of lost return guests (Mehrotra &

Ruttley, 2006).

Seeing as the costs of having too many false positives is much too high, we have thus decided to look at the precision metric, over accuracy – which is too general a metric as it does not take into consideration the nuances of the dataset, as well as the underlying business context – and recall, which focuses more on the occurrences of false negatives.

## 4.5. Hyperparameter Tuning

Hyperparameter tuning is crucial when it comes to optimising the performance and generalisation capabilities of your machine learning models. There is no one-size-fits-all set of parameters that can guarantee the best performance for your models. It depends on the dataset, problem, and models that you are utilising which will require different configurations for it to achieve optimal results. That being said, we have to be mindful of the possibilities of overfitting. Overfitting occurs when a model is excessively complex and fits the training data too closely, leading to poor performance when presented with new, unseen data.

### 4.5.1 **Random Search**

We used RandomSearchCV for our hypertuning process. Random Search (RS) is a simple technique that performs random trials in a search space. Its use can reduce the computational cost when there is a large number of possible settings being investigated. Usually, RS performs its search iteratively in a predefined number of iterations. RS has obtained efficient results in optimization for Deep Learning (DL) algorithms (Mantovani et al., 2019).

## 4.6. Data Modelling

### 4.6.1 **Logistic Regression**

Logistic Regression (LR) determines the relationship between features and the probability of a particular outcome (Goyal, 2021). It uses the Sigmoid function to get the probability of different classes and use these to classify binary targets. We will specifically be using the Binary Logistic Regression in which the target variable has only two possible outcomes; 0

and 1. In our case, 0 and 1 indicate that it is not cancelled and is cancelled respectively.

We trained and tested the model as such:

1. Build the LR Model and train on Training Set
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Perform hyperparameter tuning on the values for the 'C' parameter, which controls the regularization strength in logistic regression.
   2. RandomizedSearchCV is employed to search for the best hyperparameters within the specified distribution. It will perform 20 iterations of fitting the logistic regression model to the data with different parameter combinations. The evaluation metric used is 'precision,' and a 10-fold cross-validation is employed.
5. Evaluate the model post-tuning

These are the performance metrics captured before and after tuning the hyperparameters of the Logistic Regression Model:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.77 | |
| Macro Precision | 0.77 | |
| Precision | 0.77 | 0.78 |
| Recall | 0.91 | 0.54 |
| F1 Score | 0.83 | 0.64 |

Table 4: Performance Metrics of LR **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.77 | |
| Macro Precision | 0.78 | |
| Precision | 0.75 | 0.81 |
| Recall | 0.93 | 0.50 |
| F1 Score | 0.83 | 0.62 |

Table 5 : Performance Metrics of LR **after** Hyperparameter Tuning

As observed, the Logistic Regression model performed slightly better post hyperparameter tuning. The precision score of the model predicting cancelled hotel bookings increased from 78% to 81%. This means that of all the hotel bookings that the model predicts will be cancelled, it is correct 81% of the time. While that in itself is not too bad a precision score, as we continue to explore more models, we will soon see the LR model being outperformed by the other models.

This could be attributed to some of LR’s limitations, one of which is its construction of linear boundaries (Ranjan Rout, 2020). If the relationship between features and the likelihood of hotel booking cancellations is highly non-linear, other algorithms that can model non-linear relationships (e.g., decision trees, random forests, or support vector machines with non-linear kernels) might therefore be more suitable.

Also, due to its assumption of independence of errors (Stoltzfus, 2011), in scenarios where bookings may be interdependent or influenced by external factors that are not included in the model, logistic regression may not capture these dependencies well. Time-series models or more advanced methods would be better worth exploring.

As such, we went on to explore other various models that may be better suited for this classification problem.

### 

### 4.6.2 **Decision Tree**

Decision tree (DT) methodology is a commonly used data mining method for establishing classification systems based on multiple covariates or for developing prediction algorithms for a target variable. This method classifies a population into branch-like segments that construct an inverted tree with a root node, internal nodes, and leaf nodes (Song & Lu, 2015).

Our group decided to utilise the DT methodology in our classification problem due to the advantages that are unique to this model, specifically its efficiency in dealing with large, complicated datasets without imposing a complicated parametric structure due it being non-parametric in nature. Notably, implementing a DT model requires little data preparation due to its ability to handle missing values as well as its robustness to outliers, making it a fuss-free method overall (Song & Lu, 2015). Additionally, DT is a white box machine learning model, meaning that if a given situation is observable in the model, the condition is easily explained and interpreted by boolean logic (scikit-learn, n.d.). Given that our project deals with a binary classification problem, using a DT model that is derived from historical data, makes it easy to predict the result for future records of hotel booking cancellations (Song & Lu, 2015).

We trained and tested the model as such:

1. Build the DT Model and train on Training Set
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Decision tree designed without limitations on depth or impurity in a split will create a very complex tree, with a leaf for each observation. Again, this means perfect results on the training set but a classic case of overfitting that will fail on unseen data. To avoid this, we carried out hyperparameter tuning by inputting DT classifier parameters that could reduce overfitting (Decision trees: Complete guide to decision tree analysis 2023).
   2. Hyperparameter tuning is carried out following the standardised number of iterations, random state, and cross-validation value inputted in RandomizedSearchCV as specified in section 4.3.
5. Evaluate the model post-tuning

These are the performance metrics captured before and after tuning the hyperparameters of the Decision Tree Model:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.81 | |
| Macro Precision | 0.80 | |
| Precision | 0.84 | 0.76 |
| Recall | 0.86 | 0.74 |
| F1 Score | 0.85 | 0.75 |

Table 6: Performance Metrics of DT **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.75 | |
| Macro Precision | 0.85 | |
| Precision | 0.71 | 0.99 |
| Recall | 1.00 | 0.33 |
| F1 Score | 0.83 | 0.49 |

Table 7: Performance Metrics of DT **after** Hyperparameter Tuning

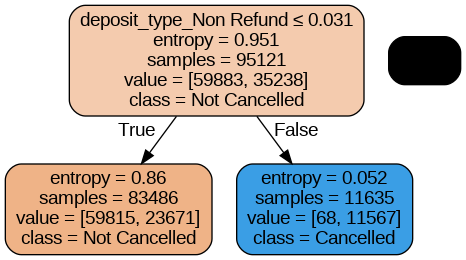


Figure 10: Decision Tree After Hyperparameter Tuning

After conducting hyperparameter tuning of the DT model, it is seen that the precision score of the model predicting cancelled hotel bookings increased from 76% to 99%. This means that of all the hotel bookings that the model predicts will be cancelled, it is correct 99% of the time.

The high predictive score post tuning is due to Classification and Regression Tree (CART) being much more sensitive to hyperparameter tuning as compared to other methods like J48 Classification (Mantovani et al., 2019).

However, even though the DT achieved an almost perfect precision score post hyperparameter tuning, this extremely high score brought forth the possibility of the model being more overfitted than before. It is also important to note that hyperparameter configurations that lead to a model with high predictive performance for a given dataset may not lead to high predictive performance for other datasets (Mantovani et al., 2019).

### 4.6.3 **Random Forest**

Random Forest (RF) is a supervised machine learning algorithm that leverages an ensemble of decision trees to facilitate predictive modelling. Random Forest models are widely used due to their ability to handle a range of classification problems effectively (Cutler et al., 2012).

RF incorporates an Out-of-Bag (OOB) estimation method, which is a unique feature. This technique leverages the data not used in the bootstrap sample for each tree to estimate the model's performance without the need for a separate validation set (Li et al., 2018). This OOB estimation helps us assess the model's predictive accuracy and generalisation performance efficiently, saving valuable time and resources.

RF algorithms are also a suitable approach for imputing missing data (Tang, F., Ishwaran, H., 2017). Though we have taken the steps to deal with missing rows in our dataset, the model generally has the ability to handle mixed types of missing data, adapt to interactions and nonlinearity, and have the potential to scale for big data scenarios.

The Random Forest model was trained and tested via the following steps:

1. Build a Random Forest Model with default classification parameters, and train on Training Set
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Because the Random Forest model is an ensemble method that makes use of multiple decision trees, running the Random Forest model was computationally very costly and time-consuming. As such, the team was only able to present the evaluation for the Random Forest Model **before** hyperparameter tuning. In this final report, we will include the evaluation of the model **before** & **after** hyperparameter tuning.
5. Evaluate the model post-tuning

These are the performance metrics captured before and after tuning the hyperparameters of the Random Forest Model:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| OOB Score | 0.82 | |
| Accuracy | 0.82 | |
| Macro Precision | 0.82 | |
| Precision | 0.85 | 0.78 |
| Recall | 0.88 | 0.73 |
| F1 Score | 0.86 | 0.76 |

Table 8: Performance Metrics **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| OOB Score | 0.75 | |
| Accuracy | 0.74 | |
| Macro Precision | 0.85 | |
| Precision | 0.71 | 1.00 |
| Recall | 1.00 | 0.32 |
| F1 Score | 0.83 | 0.48 |

Table 9: Performance Metrics **after** Hyperparameter Tuning

After conducting hyperparameter tuning, The Random Forest Model showed a fairly large improvement in Precision – the Macro Precision rose from 0.82 to 0.85 (+0.03). However, we noted that the Precision for ‘Is Cancelled’ after hypertuning was a perfect score of 1.00, which showed that the model may have been subjected to overfitting for this dataset.

The Random Forest model is also much slower to train compared to other models like a single Decision Tree model – for a large dataset like ours, an ensemble method like Random Forest can be very time consuming to train (Beheshti, 2022).

### 4.6.4 **Extreme Gradient Boost (XGBoost)**

Extreme Gradient Boosting is a powerful supervised machine learning algorithm that uses an ensemble of decision trees and gradient boosting to make predictions. The model provides parallel tree boosting and is the leading machine learning library for classification problems of our sort (NVIDIA, n.d.).

The XGBoost model has high predictive accuracy as it excels in capturing intricate data relationships and handles complex patterns in data effectively (Guest\_Blog, 2023). Aside from the model’s great performance, it also handles non-linear relationships within the data (Hachcham, 2023). This capability is especially valuable for our project, predicting hotel cancellations, where the data contains complex, non-linear dependencies that other models might struggle to capture. The model is also computationally efficient, making it suitable for large datasets. The model implements parallel processing (speed) and takes advantage of GPU acceleration for even faster training (Jain, 2023).

We trained and tested the model as such:

1. Build the XGBoost Model and train on Training Set
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Particularly for the XGBoost model, we decide to perform 40 iterations (instead of 20) to obtain much accurate hyperparameter values to achieve better predictive results.
5. Evaluate the model post-tuning

These are the performance metrics captured before and after tuning the hyperparameters of the XGBoost model:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.81 | |
| Macro Precision | 0.81 | |
| Precision | 0.81 | 0.82 |
| Recall | 0.91 | 0.65 |
| F1 Score | 0.86 | 0.72 |

Table 10: Performance Metrics **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.81 | |
| Macro Precision | 0.81 | |
| Precision | 0.81 | 0.81 |
| Recall | 0.91 | 0.65 |
| F1 Score | 0.86 | 0.72 |

Table 11: Performance Metrics **after** Hyperparameter Tuning

After tuning the hyperparameters, the Precision Score of the XGBoost model showed a slight drop from 0.82 to 0.81. The drop in the precision for identifying cancelled hotel bookings after hyperparameter tuning could likely be due to insufficient tuning iterations - 40 iterations of tuning produced much better Precision Score than 20 iterations, but it is still not the best. However, running all the possible iterations is extremely computationally intensive and takes a long time to process.

Contrary to the excellent performance and productivity of the XGBoost model, the model has its limitations as well. The XGBoost model is a complex algorithm that requires some degree of technical expertise to implement to produce the optimal model. Finding the best hyperparameters often involves performing Grid Search or Random Search, which requires significant computational resources, resulting in the process to be time consuming (Krayonnz, 2023). The model may also tend to overfit, especially when the hyperparameters are not tuned correctly (Rithp, 2023). Although the XGBoost model is fast and efficient, the model can be extremely memory-intensive, especially when the model works with large datasets. This can pose a significant challenge when training and testing the model on computers with limited memory, leading to slower performance (DMLC XGBoost, 2020).

### 4.6.5 **FeedForward Neural Network (FNN)**

Feedforward neural networks (FNNs), also known as multilayer perceptrons (MLPs), are a fundamental type of artificial neural network used in machine learning and deep learning for a variety of tasks. In the context of solving binary classification problems, such as hotel booking cancellations, FNNs prove to be an invaluable tool. These networks are adept at capturing and understanding intricate patterns within data, which is crucial when distinguishing between booking cancellations and successful reservations. FNNs' deep architecture, leveraging multiple layers and backpropagation for learning, allows them to uncover hierarchical data representations, enabling more effective feature learning and abstraction in the binary classification process (Turing, 2022).

One of the key reasons for employing FNNs in binary classification tasks is their universal approximation capabilities. Feedforward neural networks can approximate any continuous function with remarkable precision. This quality makes them exceptionally versatile, as they can adapt to the complexities of real-world data and tailor their predictive power to the specific nuances of hotel booking cancellations. Furthermore, FNNs bring the element of non-linearity into the classification process. By using multiple layers of non-linear activation functions, they can model intricate non-linear relationships and dependencies that often exist in binary classification problems, enhancing their ability to recognize and differentiate the various factors contributing to booking cancellations, be it seasonal trends, customer behaviour, or other variables (Turing, 2022).

We trained and tested the model as such:

1. Build the FNN Model and train on Training Set
   1. Set the input dimension of FNN to match the number of selected features
   2. 1 hidden layer with 64 neurons (ReLU activation function)
      1. Use ReLU mainly because of computational efficiency. Also, it promotes sparsity in the network by deactivating neurons with negative inputs and avoids the vanishing gradient problem, as it does not saturate for large positive inputs (Brownlee, 2020).
   3. Output layer (Sigmoid activation function)
      1. The sigmoid activation function scales output values between 0 and 1, facilitating their direct interpretation as the probability of an input belonging to the positive class in binary classification (Vishwakarma, 2023).
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Perform hyperparameter tuning on the number of units (neurons) in the first hidden layer, number of additional hidden layers, and learning rate
   2. Run RandomSearch for 20 trials with 10 epochs for each trial.
5. Evaluate the model post-tuning

These are the performance metrics captured before and after tuning the hyperparameters of the FNN model:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.80 | |
| Macro Precision | 0.79 | |
| Precision | 0.82 | 0.76 |
| Recall | 0.87 | 0.68 |
| F1 Score | 0.84 | 0.72 |

Table 12: Performance Metrics **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.77 | |
| Macro Precision | 0.83 | |
| Precision | 0.74 | 0.93 |
| Recall | 0.98 | 0.43 |
| F1 Score | 0.84 | 0.59 |

Table 13: Performance Metrics **after** Hyperparameter Tuning

After hyperparameter tuning, there is an increase in Macro Precision from 0.79 to 0.83, and increase the precision of identifying cancelled bookings, from 0.76 to 0.93, indicating an improvement in the model's ability to correctly identify bookings that have been cancelled. While the tuning showed great improvement for our target metric, it is important to note the drops in other performance metrics (recall and ultimately F1). The trade-off of having low recall means that there are a significant number of false negatives, where actual cancellations are being wrongly classified as non-cancellations.

Parameters provided by the hypertuning may also not be the best as we used RandomSearch() of 20 iterations instead of all the possible iterations, due to it being too computationally intensive and time-consuming, especially for deep neural networks like FNN.

Furthermore, it is good to note that FNN does have its limitations as well. FNN is prone to overfitting, especially when the model gets too complex. Deeper FNN with more hidden layers are also prone to exploding and vanishing gradients which can greatly inhibit the model’s learning capability (Bohra, 2021). FNN is also sensitive to changes in hyperparameters; slight changes to the hyperparameters will significantly impact model performance. This is the reason why it is so tricky and hard to find the optimal hyperparameters for our FNN model. Last but not least, FNN, like any other deep learning models, requires large amounts of diverse labelled data in order to perform well. Even though our current dataset is reasonably large, an even larger and more diverse dataset would significantly improve the training performance of the FNN model (CaseGuard Video Redaction Software, 2022).

A possible solution to this can be investing in more extensive and diverse labelled training data to help address the trade-off issue, allowing the model to better generalise while maintaining high precision. It may be worth considering hybrid models that combine deep neural networks with traditional machine learning algorithms, as this can sometimes provide a more robust and interpretable solution. Finally, we can then further fine tune the hyperparameters with more iterations that focus on improving precision while also maintaining a high accuracy and F1 score.

### 4.6.6 **Support Vector Machine (SVM)**

SVM is a powerful supervised algorithm that works best on smaller but complex datasets. Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks, but generally, they work best in classification problems.

SVM works by trying to the hyperplane that best separates the classes while maximising the margin between them. This makes SVM generally more robust to overfitting. SVM also has different kernels which allows it to work well for non-linear data as well. That being said, it is sometimes difficult to choose which kernels to use as it requires domain knowledge and careful experimentation. While SVM performs well in high-dimensional spaces, making them suitable for complex data with many attributes, it is important to note that it can be computationally intensive and time-consuming when large datasets are used. This was something that we underestimated during our project and we had to rethink the ways of how we wanted to hypertune our parameters (Saini, 2023).

We trained and tested the model as such:

1. Build a SVM Model as the baseline, and train on Training Set
2. Predict on Test Set
3. Evaluate the Model
4. Perform Hyperparameter Tuning
   1. Because of our large dataset, 118902 x 213, we noted that even doing RandomSearchCV would take up too much time. We resulted in manually iterating through the kernels with the other parameters set to their default values first. We then chose the kernels that showed promising results to further tune other parameters, C, Gamma, Degree and Coef0 where applicable.
5. Evaluate the model post tuning

These are the performance metrics captured before and after tuning the hyperparameters of the SVM model using the Radial Basis Function (RBF) kernel as our baseline:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.79 | |
| Macro Precision | 0.81 | |
| Precision | 0.78 | 0.83 |
| Recall | 0.93 | 0.57 |
| F1 Score | 0.85 | 0.68 |

Table 14: Performance Metrics **before** Hyperparameter Tuning

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.76 | |
| Macro Precision | 0.85 | |
| Precision | 0.73 | 0.98 |
| Recall | 0.99 | 0.38 |
| F1 Score | 0.95 | 0.55 |

Table 15: Performance Metrics **after** Hyperparameter Tuning

After manually tuning in 2 stages, we found out that the kernel: RBF, C: 0.1 and Gamma: 100 resulted in the best performance in terms of precision. As we can see from Table 14 and Table 15 from above, the macro precision increased by 0.04. We can also see that while the precision for Not Cancelled(0) decreased by 0.05 to 0.73, Is Cancelled(1) increased tremendously by 0.15 to 0.98. While the tuning showed great improvement for our target metric, it is important to note the drops in other performance metrics. This may also not be the best parameters as we did not implement RandomizedSearchCV or GridSearchCV due to it being too computationally intensive and time-consuming. Instead, manually iterate the values of C and Gamma in magnitudes of 10.

We can see from our hyperparameter tuning that while SVM can produce some good results for our classification problem, its disadvantage of being computationally intensive and time-consuming is inherent when used for big datasets. When tuning the values of C and Gamma, larger values for either or both parameters made the model take up a lot more time as the model tries to find a more complex boundary separating the training data. Too large values of C and Gamma however, will cause problems of overfitting while smaller values have better generalisation but may cause problems of underfitting. Hence, it is important to experiment with these parameters to find an optimal model.

Since performing hyperparameter tuning on such a large dataset is very computationally intensive and time-consuming, a possible solution would be to split the datasets into smaller subsets or get a representative sample and hypertune it from there. However, some potential drawbacks and considerations is that these smaller subsets or samples have to be representative to the population otherwise, the parameters will not generalise well for the full dataset.

# **5. Results & Discussion**

Table 16 shows the ranking (in descending order) of all the models based on the Precision Scores derived from the models above:

| Model | Precision Score  (is\_canceled) |
| --- | --- |
| 1. Random Forest | 1.00 |
| 1. Decision Tree | 0.99 |
| 1. Support Vector Machine (SVM) | 0.98 |
| 1. Feedforward Neural Network (FNN) | 0.93 |
| 1. Extreme Gradient Boosting (XGBoost) | 0.81 |
| 1. Logistic Regression | 0.81 |

Table 16: Ranking of Models based on the Precision Score

The team decided to narrow the selection of models to the top 3 models that had obtained higher Precisions. Table 17 shows the top 3 models selected by the team:

| Model | Precision Score  (is\_canceled) |
| --- | --- |
| 1. Support Vector Machine (SVM) | 0.98 |
| 1. Feedforward Neural Network (FNN) | 0.93 |
| 1. Extreme Gradient Boosting (XGBoost) | 0.81 |

Table 17: Top 3 Models

It is worth noting that even though the Random Forest model and Decision Tree model had obtained near perfect Precision Scores, the team chose not to select them as we realise that these models are extremely prone to overfitting, which will bring about more difficulties when determining the globally optimal tree. In addition, though the Logistic Regression model had obtained a Precision Score of 0.81, which matches what the XGBoost model had obtained, the team did not select it because we felt that the Logistic Regression model was generally more sensitive to outliers in the dataset.

For SVM and FNN, even though it obtained a very high precision in identifying cancelled bookings, there is a tradeoff of having a relatively low recall and F1 score, hence in an attempt to counter these setbacks, we decided to use a voting ensemble to get a relatively high precision score while maintaining a high Recall and F1 Score.

## 5.1 Voting Ensemble

Using the top 3 models selected above, we performed a Hard Voting Ensemble by combining the predictions of the models. The predictions obtained by the ensemble will then simply be the majority vote of the individual classifiers (Ahmed, 2023). Table 18 shows the predictions produced by the Hard Voting Ensemble:

| Metrics | Y\_Test, Y\_Pred | |
| --- | --- | --- |
| Not Cancelled (0) | Is Cancelled (1) |
| Accuracy | 0.80 | |
| Macro Precision | 0.81 | |
| Precision | 0.80 | 0.82 |
| Recall | 0.92 | 0.61 |
| F1 Score | 0.85 | 0.70 |

Table 18: Predictions by Hard Voting Ensemble

Based on Table 18, we obtained a Precision Score of 0.83. Through the implementation of the Hard Voting Ensemble method, we managed to create a model that is capable of identifying that 83% of the bookings that are classified as "cancelled," are genuinely cancelled. In other words, out of the bookings the model predicts as cancellations, 83% of them are accurate, highlighting the model's effectiveness in reducing false positive predictions and enhancing its precision in identifying true cancellations. Through the ensemble method, we also saw an improvement in the Recall and F1 Score compared to that of SVM and FNN, which addresses our initial concern.

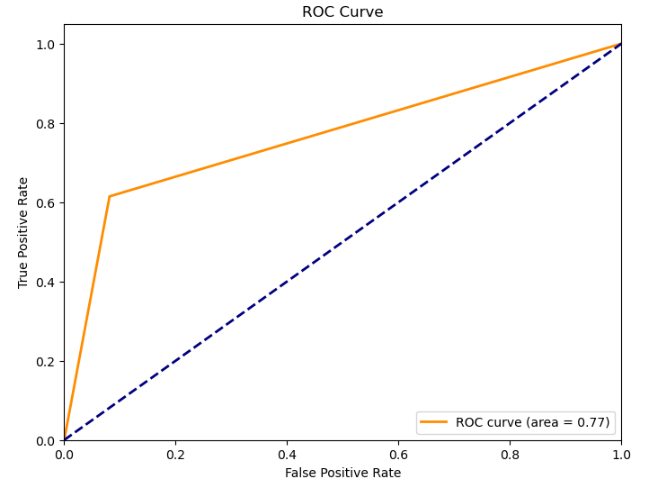


Figure 11: ROC curve for voting ensemble

We also managed to obtain a good AUC score of 0.76, which suggests that the model has moderate discriminatory power (Bhandari, 2023). This indicates that our binary classification model is reasonably effective at distinguishing between the positive and negative classes, though there is much room for improvement. We believe that improving recall metric, while maintaining our high precision, will greatly improve the AUC score, and thus improve the model’s performance in distinguishing between classes.

# **6. Future Work and Conclusion**

## 6.1. Conclusion

In conclusion, even though our voting ensemble has a lower precision value as compared to certain individual models, we believe that the integration of the 3 models will help with generalising and prediction of future data. It can also help reduce overfitting and also increase robustness to noise and outliers.

For our performance against literature review, as shown above, we were able to successfully explore the areas of consideration we set out in our proposal. Furthermore, we were also able to validate our initial hypotheses with regards to the methods set out in the literature review.

* Literature review 1
  + Compared to the findings in literature review 1, our research has yielded commendable results. We have achieved high precision and a relatively high recall, leading to low false positive and false negative rates, which are in line with the outcomes reported in literature 1. Additionally, our model has demonstrated an improved accuracy of 0.8, surpassing the accuracy of all models documented in literature 1. These accomplishments were made possible through rigorous data preprocessing and feature selection. Notably, we reduced the feature dimensions to just 10, which is significantly fewer than the 24 features used in the literature review. This underscores that we can not only achieve superior metric performance but also effectively mitigate overfitting and enhance generalisation in our model.
* Literature review 2
  + Compared to the findings in literature review 2, our utilisation of a voting ensemble approach, which combines both traditional machine learning methods and deep learning techniques, has resulted in strong overall performance, marked by consistently high metrics.
* Literature review 3
  + Compared to the findings in literature review 3, we are able to get good results from traditional machine learning methods, which could perform even better than our deep learning method, especially after applying hyperparameter tuning to them.
  + However, compared to the literature review, which uses data from 4 different hotels in a single PMS, our datasets only comprise data from two distinct types of hotels, predominantly city hotels and resort hotels. Although this dual-hotel dataset is more diverse than using just a single hotel type, our objective is to further enrich the diversity of our dataset. This expansion is aimed at enhancing our model's performance and generalisation capabilities, enabling it to excel in predicting future unseen external data.

We also acknowledge that there could be possible overfitting of models, especially with Decision Tree model which resulted in near-perfect precision. Although good on test data, it may not perform so well for future data.

## 6.2. Future Work

To optimise the performance of complex models, we propose two key strategies. First, we recommend extending the number of iterations during hyperparameter tuning, especially for models with a multitude of hyperparameters. This thorough exploration of the hyperparameter space enhances the model's configuration, leading to improved performance. Second, we emphasise the importance of cross-dataset validation by acquiring new datasets from reliable sources. Validating the model's performance on unseen data from diverse distributions ensures its robustness and generalisation. This approach provides a comprehensive assessment of the model's reliability, even when faced with varying data characteristics. By incorporating these strategies, we aim to fortify the accuracy and adaptability of our complex models.

In addition to hyperparameter tuning and cross-dataset validation, we advocate broadening our research horizon by exploring adjacent sectors such as airline ticket cancellation and cruise booking cancellation. By applying our expertise to these related domains, we can build or refine models that address analogous challenges, harnessing the knowledge gained in our existing research to enhance model performance. Furthermore, we underscore the importance of knowledge transfer, emphasising the invaluable lessons that can be applied across sectors. Collaboration with experts in these fields and fostering a culture of knowledge sharing can lead to innovative models capable of addressing a wide array of complex scenarios. By adopting these approaches, we aim to stay at the forefront of research and model development, effectively adapting to emerging challenges and opportunities in our ever-evolving landscape.

# **7. References**

Ahmed, I. (2023, May 31). What is Hard and Soft Voting in Machine Learning? - Ilyas Ahmed - Medium. Medium.<https://ilyasbinsalih.medium.com/what-is-hard-and-soft-voting-in-machine-learning-2652676b6a32>

Antonio, N., Almeida, A., & Nunes, L. Hotel booking demand datasets. Retrieved November 2, 2023, from<https://www.sciencedirect.com/science/article/pii/S2352340918315191>

Antonio, N., De Almeida, A. M., & Nunes, L.. (2017). Predicting Hotel Booking Cancellation to Decrease Uncertainty and Increase Revenue. Retrieved from<https://www.researchgate.net/publication/310504011_Predicting_Hotel_Booking_Cancellation_to_Decrease_Uncertainty_and_Increase_Revenue>

Beheshti, N. (2022, March). Random Forest Regression. Retrieved from<https://towardsdatascience.com/random-forest-regression-5f605132d19d>.

Behsudi, A. (2020, December) Tourism-dependent economies are among those harmed the most by the pandemic. Retrieved November 2, 2023, from<https://www.imf.org/en/Publications/fandd/issues/2020/12/impact-of-the-pandemic-on-tourism-behsudi>

Bhandari, A. (2023, August 31). Guide to AUC ROC Curve in Machine Learning : What is specificity? Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/#:~:text=The%20Area%20Under%20the%20Curve,the%20positive%20and%20negative%20classes.

Bohra, Y. (2021, June 23). The challenge of Vanishing/Exploding gradients in

deep neural networks. Analytics

Vidhya.

https://www.analyticsvidhya.com/blog/2021/06/the-challenge-of-vanishing-exploding-gradients-in-deep-neural-networks/

Brownlee, J. (2020, August 20). A gentle introduction to the

rectified Linear Unit (RELU).

MachineLearningMastery.com.

https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

CaseGuard Video Redaction Software. (2022, August 9). Deep

learning and Feed-Forward neural networks. CaseGuard.

https://caseguard.com/articles/feed-forward-neural-networks-new-deep-learning-models/

Chen, C. C., Schwartz, Z., & Vargas, P. (2011). The search for the best deal: How hotel cancellation policies affect the search and booking decisions of deal-seeking customers. International Journal of Hospitality Management, 30(1), 129-135.

Chen, S., Ngai, E., Ku Y., Xu Z., Gou X., & Zhang, C. (2023, July). Predicting Hotel Booking Cancellation to Decrease Uncertainty and Increase Revenue. Retrieved from<https://www.sciencedirect.com/science/article/abs/pii/S0167923623000349>

Cutler et. al. (2012, January). Random Forests. Retrieved from<https://link.springer.com/chapter/10.1007/978-1-4419-9326-7_5>

Decision trees: Complete guide to decision tree analysis. Explorium. (2023, August 6). https://www.explorium.ai/blog/machine-learning/the-complete-guide-to-decision-trees/

DMLC XGBoost. (2020, January 8). RAM memory keeps increasing with iterative training of XGB models. XGBoost.<https://discuss.xgboost.ai/t/ram-memory-keeps-increasing-with-iterative-training-of-xgb-models/1331>

Gartvall, E., & Sk˚anhagen. O (2021, September). Predicting hotel cancellation using machine learning. Retrieved from<https://gupea.ub.gu.se/bitstream/handle/2077/70742/gupea_2077_70742_1.pdf?sequence=1>

Goyal, C. (2021, May 28). Logistic Regression Questions | Questions On Logistic Regression. Analytics Vidhya https://www.analyticsvidhya.com/blog/2021/05/20-questions-to-test-your-skills-on-logistic-regression/

Guest\_Blog. (2023, March 30). Introduction to XGBOOST Algorithm in Machine Learning. Analytics Vidhya.<https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>

Hachcham, A. (2023, August 11). XGBoost: Everything you need to know. [neptune.ai](http://neptune.ai).<https://neptune.ai/blog/xgboost-everything-you-need-to-know>

Hayes, D. K., & Miller, A. A. (2011). Revenue management for the hospitality industry. Hoboken, NJ, USA: John Wiley & Sons, Inc

Jain, A. (2023, October 25). Mastering XGBoost Parameter Tuning: A Complete Guide with Python Codes. Analytics Vidhya.<https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/#Advantages_of_XGBoost>

Krayonnz : Social Learning network. (2023, April 15).<https://www.krayonnz.com/user/doubts/detail/623b2b7235e21e005f953106/what-are-the-advantages-and-disadvantages-of-XGBoost>

Li et. al. (2018, September). A principle component analysis-based random forest with the potential nearest neighbor method for automobile insurance fraud identification. Retrieved from<https://www.sciencedirect.com/science/article/abs/pii/S1568494617304386>.

Mantovani, R. G., Horváth, T., Cerri, R., Junior, S. B., Vanschoren, J., &amp; de Carvalho, A. C. P. de L. F. (2019, February 12). An empirical study on hyperparameter tuning of decision trees. arXiv.org. https://arxiv.org/abs/1812.02207

Mehrotra, R., & Ruttley, J. (2006) Revenue management. American Hotel and Lodging Association.

Mostipak, J. (2019) Hotel Booking Demand. Retrieved November 4, 2023, from<https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand>

Mou, J. (2020) Research on the Impact of COVID19 on Global Economy. Retrieved November 2, 2023, from<https://iopscience.iop.org/article/10.1088/1755-1315/546/3/032043>

NGO, Hoang Anh. (2019, December 5). Correlation between numerical variables.<https://rpubs.com/hoanganhngo610/556786#:~:text=A%20correlation%20coefficient%20is%20a,variable%20with%20a%20known%20distribution>.

Noone, B. M., & Lee, C. H. (2010). Hotel overbooking: The effect of overcompensation on customers’ reactions to denied service. Journal of Hospitality & Tourism Research, 35(3), 334–357.

NVIDIA. (n.d.). What is XGBoost? NVIDIA Data Science Glossary.<https://www.nvidia.com/en-us/glossary/data-science/xgboost/>

Ranjan Rout, Amiya. (2020, August 25). Advantages and disadvantages of logistic regression. GeeksforGeeks. https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/

Rithp. (2023, January 29). The main parameters in XGBoost and their effects on model performance. Medium.<https://medium.com/@rithpansanga/the-main-parameters-in-xgboost-and-their-effects-on-model-performance-4f9833cac7c>

Saini, A. (2023, October 27). Guide on Support Vector Machine (SVM) Algorithm. Analytics Vidhya.<https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/#h-what-is-a-support-vector-machine>

scikit-learn. (n.d.). 1.10. Decision Trees. scikit. https://scikit-learn.org/stable/modules/tree.html

Smith, S. J., Parsa, H. G., Bujisic, M., & van der Rest, J. P. (2015). Hotel cancelation policies, distributive and procedural fairness, and consumer patronage: A study of the lodging industry. Journal of Travel & Tourism Marketing, 32(7), 886-906

Song, Y.-Y., &amp; Lu, Y. (2015, April 25). Decision tree methods: Applications for classification and prediction. Shanghai archives of psychiatry. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4466856/

Stoltzfus, J. C. (2011). Logistic regression: A brief primer. Academic Emergency Medicine, 18(10), 1099–1104. https://doi.org/10.1111/j.1553-2712.2011.01185.x

Talluri, K. T., Van Ryzin, G., & Van Ryzin, G. (2004). The theory and practice of revenue management (Vol. 1). Boston: Kluwer Academic Publishers.

Tang, F., Ishwaran, H. (2017, June). Random forest missing data algorithms. Retrieved from<https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11348>.

The Straits Times. (2023, May 26) Travel industry looks to Chinese tourists to cap post-Covid-19 rebound. Retrieved November 2, from<https://www.straitstimes.com/world/travel-industry-looks-to-chinese-tourists-to-cap-post-covid-19-rebound>

Turing. (2022, February 11). Understanding feed forward neural networks with

maths and statistics.<https://www.turing.com/kb/mathematical-formulation-of-feed-forward-neural-network#advantages-of-feed-forward-neural-networks>

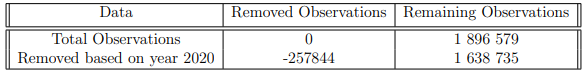
Turney, S. (2023, June 22). Chi-Square Test of Independence | Formula, Guide & Examples. Scribbr.<https://www.scribbr.com/statistics/chi-square-test-of-independence/>

Vishwakarma, S. (2023, October 31). How to understand sigmoid

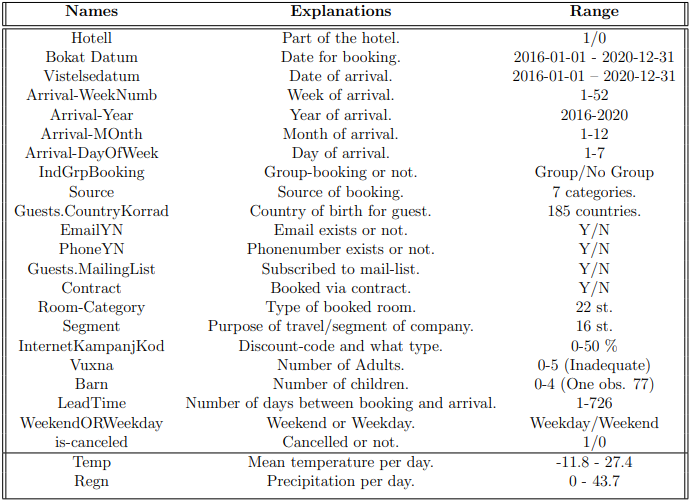
function in artificial neural networks? Analytics Vidhya.

https://www.analyticsvidhya.com/blog/2023/01/why-is-sigmoid-function-important-in-artificial-neural-networks/#:~:text=The%20sigmoid%20function%20is%20defined,classification%20and%20logistic%20regression%20problems.

# **8. Appendix**

.

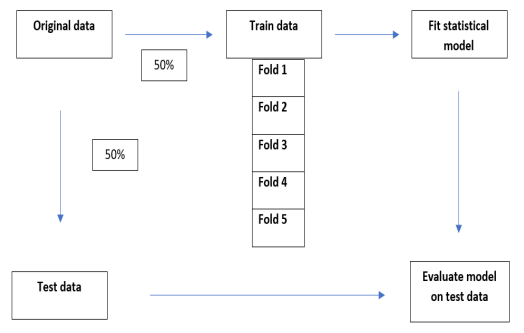
*Table A*



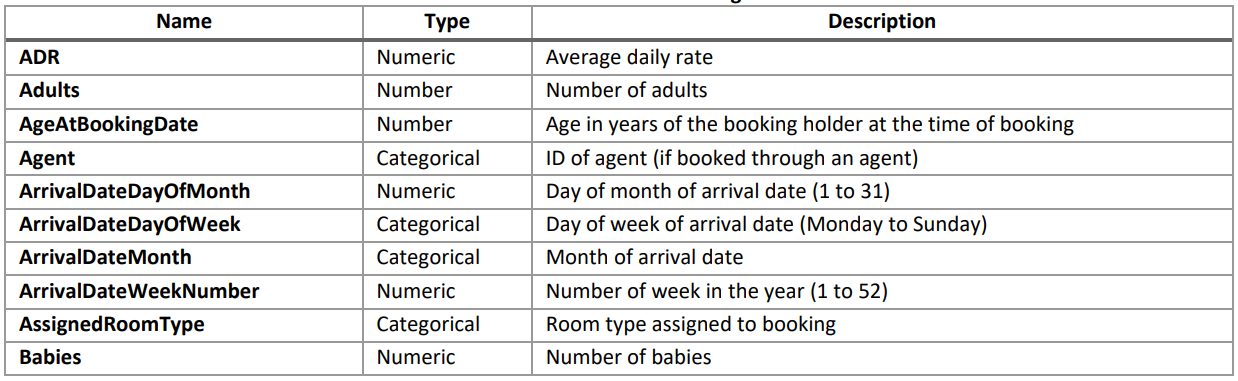
*Table B*

| **Variable/Feature** | **Number of Variables After One-hot Encoding** |
| --- | --- |
| country | 177 (country\_ABW, country\_AGO, country\_AIA, country\_ALB, country\_AND, country\_ARE, country\_ARG, country\_ARM, country\_ASM, country\_ATA, ... ,country\_UMI, country\_URY, country\_USA, country\_UZB, country\_VEN, country\_VGB, country\_VNM, country\_ZAF, country\_ZMB, country\_ZWE) |
| deposit\_type | 3 (deposit\_type\_No Deposit, deposit\_type\_Non Refund, deposit\_type\_Refundable) |
| market\_segment | 8 (market\_segment\_Aviation, market\_segment\_Complementary, market\_segment\_Corporate, market\_segment\_Direct, market\_segment\_Groups, market\_segment\_Offline TA/TO, market\_segment\_Online TA,  market\_segment\_Undefined) |
| assigned\_room\_type | 12 (assigned\_room\_type\_A, assigned\_room\_type\_B, assigned\_room\_type\_C,  assigned\_room\_type\_D, assigned\_room\_type\_E, assigned\_room\_type\_F,  assigned\_room\_type\_G,  assigned\_room\_type\_H,  assigned\_room\_type\_I,  assigned\_room\_type\_K,  assigned\_room\_type\_L,  assigned\_room\_type\_P) |
| distribution\_channel | 5 (distribution\_channel\_Corporate, distribution\_channel\_Direct,  distribution\_channel\_GDS, distribution\_channel\_TA/TO,  distribution\_channel\_Undefined) |
| customer\_type | 4 (customer\_type\_Contract, customer\_type\_Group, customer\_type\_Transient, customer\_type\_Transient-Party) |

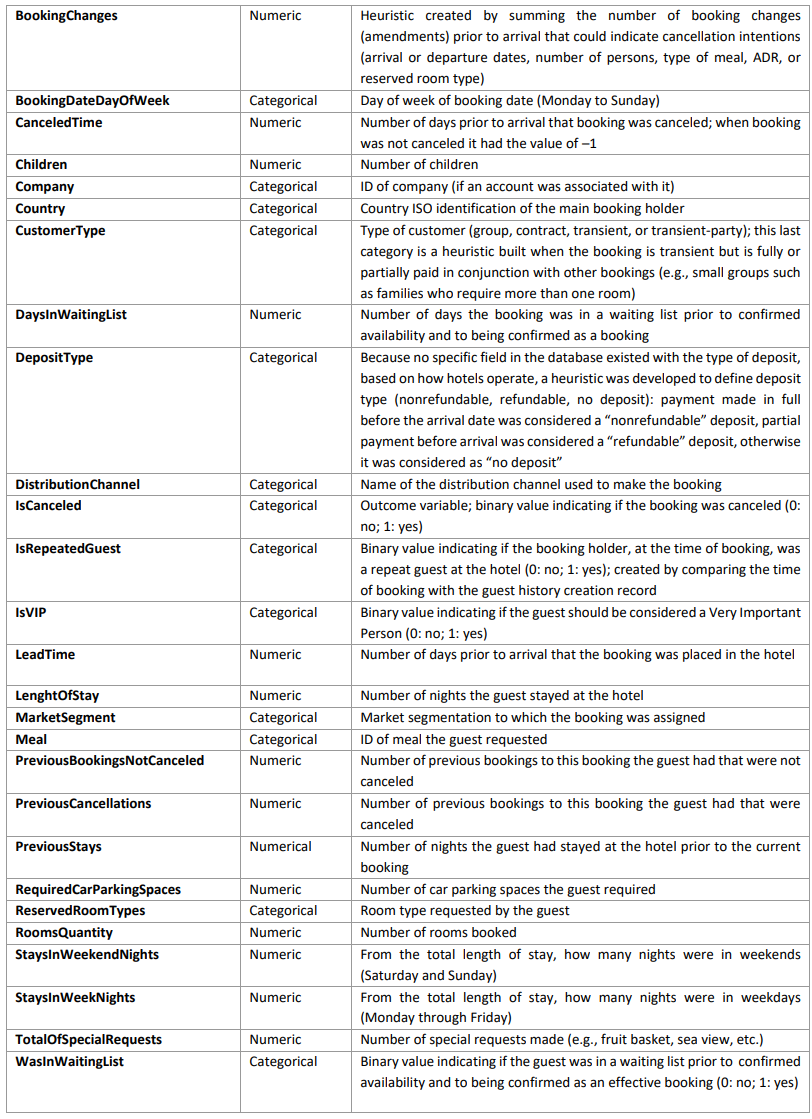
*Table C*



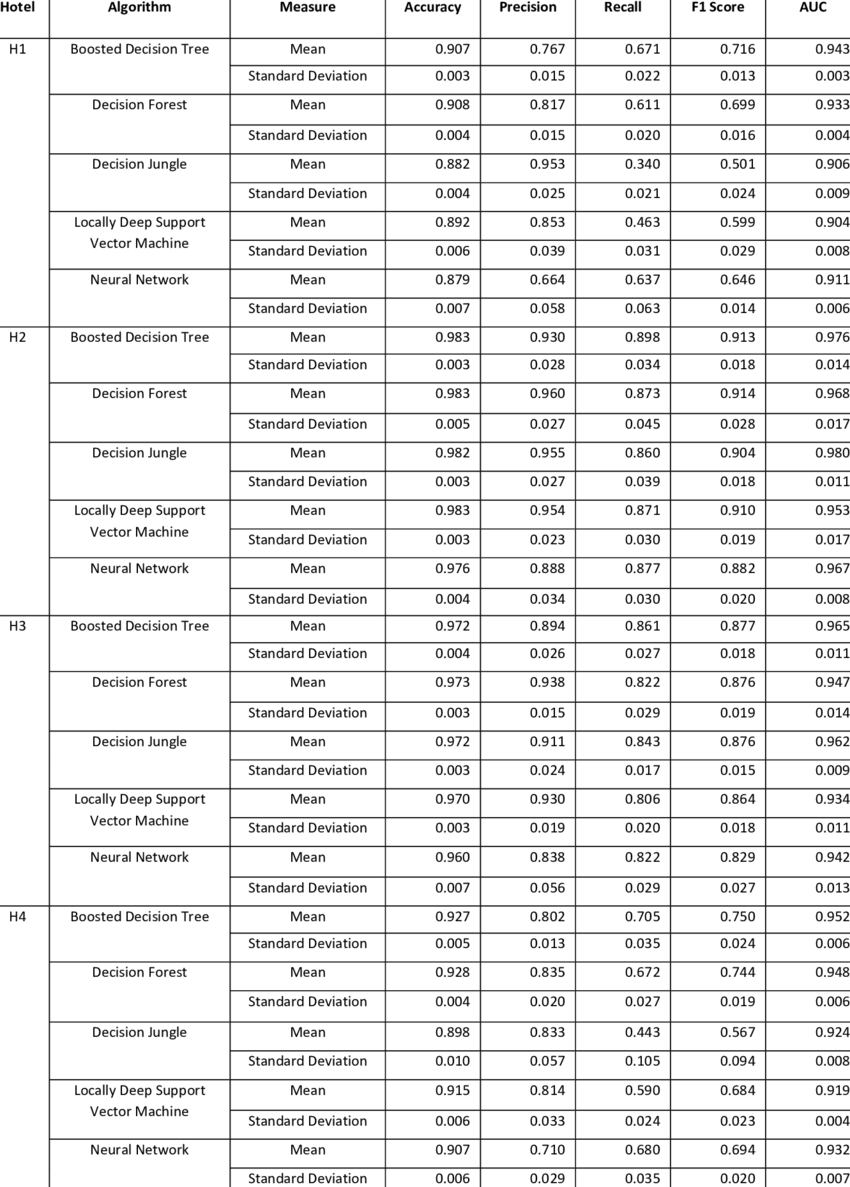
*Figure C*



*Figure D*



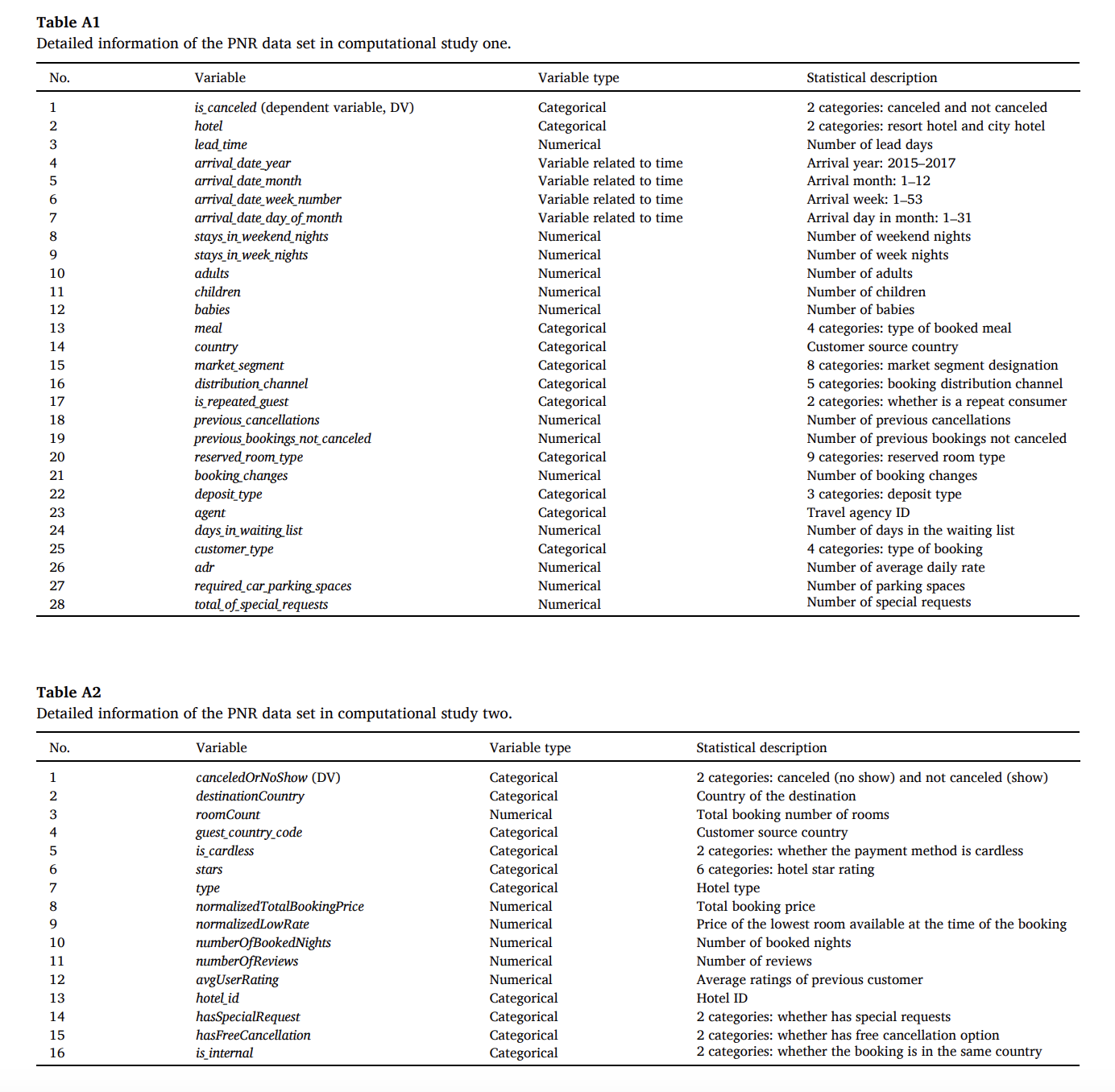
*Figure E*



*Figure F*

| Column Name | Description |
| --- | --- |
| hotel | Hotel (H1 = Resort Hotel or H2 = City Hotel) |
| is\_canceled | Value indicating if the booking was cancelled (1) or not (0) |
| lead\_time | Number of days that elapsed between the entering date of the booking into the PMS and the arrival date |
| arrival\_date\_year | Year of arrival date |
| arrival\_date\_month | Month of arrival date |
| arrival\_date\_week\_number | Week number of year for arrival date |
| arrival\_date\_day\_of\_month  stays\_in\_weekend\_nights | Day of arrival date  Number of weekend nights (Saturday or Sunday) the guest stayed or booked to stay at the hotel |
| stays\_in\_week\_nights | Number of week nights (Monday to Friday) the guest stayed or booked to stay at the hotel |
| adults | Number of adults |
| children | Number of children |
| babies | Number of babies |
| meal | Type of meal booked. Categories are presented in standard hospitality meal packages: Undefined/SC – no meal package; BB – Bed & Breakfast; HB – Half board (breakfast and one other meal – usually dinner); FB – Full board (breakfast, lunch and dinner) |
| country | Country of origin. Categories are represented in the ISO 3155–3:2013 format |
| market\_segment | Market segment designation. In categories, the term “TA” means “Travel Agents” and “TO” means “Tour Operators” |
| distribution\_channel | Booking distribution channel. The term “TA” means “Travel Agents” and “TO” means “Tour Operators” |
| is\_repeated\_guest | Value indicating if the booking name was from a repeated guest (1) or not (0) |
| previous\_cancellations | Number of previous bookings that were 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 |
| reserved\_room\_type | Code of room type reserved. Code is presented instead of designation for anonymity reasons. |
| assigned\_room\_type | Code for the type of room assigned to the booking. Sometimes the assigned room type differs from the reserved room type due to hotel operation reasons (e.g. overbooking) or by customer request. Code is presented instead of designation for anonymity reasons. |
| booking\_changes | Number of changes/amendments made to the booking from the moment the booking was entered on the PMS until the moment of check-in or cancellation |
| deposit\_type | Indication on if the customer made a deposit to guarantee the booking. This variable can assume three categories: No Deposit – no deposit was made; Non Refund – a deposit was made in the value of the total stay cost; Refundable – a deposit was made with a value under the total cost of stay. |
| agent | ID of the travel agency that made the booking |
| company | ID of the company/entity that made the booking or responsible for paying the booking. ID is presented instead of designation for anonymity reasons |
| days\_in\_waiting\_list | Number of days the booking was in the waiting list before it was confirmed to the customer |
| customer\_type | Type of booking, assuming one of four categories: Contract - when the booking has an allotment or other type of contract associated to it; Group – when the booking is associated to a group; Transient – when the booking is not part of a group or contract, and is not associated to other transient booking; Transient-party – when the booking is transient, but is associated to at least other transient booking |
| adr | Average Daily Rate as defined by dividing the sum of all lodging transactions by the total number of staying nights |
| Required\_car\_parking\_spaces  total\_of\_special\_requests | Number of car parking spaces required by the customer  Number of special requests made by the customer (e.g. twin bed or high floor) |
| reservation\_status | Reservation last status, assuming one of three categories: Canceled – booking was canceled by the customer; Check-Out – customer has checked in but already departed; No-Show – customer did not check-in and did inform the hotel of the reason why |
| reservation\_status\_date | Date at which the last status was set. This variable can be used in conjunction with the ReservationStatus to understand when was the booking canceled or when did the customer checked-out of the hotel |

*Figure G*



*Figure H*