Predicting Hotel Booking Cancellations with Tree-Based Models

Chris Thomas Alexander
Electrical, Computer and Energy Engineering Department
University of Colorado, Boulder
Boulder, USA
Chris.Alexander@colorado.edu

***Abstract*—The consistent occurrence of booking cancellations poses a revenue challenge for the hotels and service industry. To address this issue, forecasting the number of cancellations during different seasons based on data can be a valuable tool. Implementing tree-based machine learning algorithms enables accurate predictions. Additionally, by learning and tuning hyperparameters such as tree depth and number of trees, which directly influence these predictions, the confidence and performance of the model can be improved. This approach helps mitigate the impact of cancellations and assists in making informed decisions within the industry.**

***Index Terms*—Data Pre-processing, Exploratory Data Analysis, Decision Tree, Random forest, Hyper-parameter tuning, RandomizedSearchCV.**

I. Introduction

Cancellation of bookings in the hotel and service industry has become a persistent problem, leading to diminishing revenue for businesses. The uncertainty surrounding cancellations and their impact on financial stability necessitates effective strategies to mitigate this issue. One promising approach is to forecast the number of cancellations during different seasons based on historical data, enabling proactive measures and better resource allocation. Machine learning algorithms, specifically tree-based models, have shown promising results in making accurate predictions in various domains. By leveraging these algorithms, it becomes possible to develop a robust cancellation prediction model.

Additionally, tuning the hyper-parameters, such as tree depth and number of trees, further enhances the predictive capabilities and confidence of the model. This paper aims to explore the application of tree-based machine learning algorithms for predicting booking cancellations in the hotel and service industry. Through empirical analysis and experimentation, we seek to develop a reliable and accurate prediction model that can assist industry stakeholders in managing and mitigating the financial impact caused by cancellations. The findings from this research will contribute to improving revenue management strategies and optimizing resource allocation in the face of uncertain booking cancellations.

II. Dataset

The data was obtained from the website Kaggle [1], comprising 19 features and 36,275 bookings for the years 2017 and 2018. With the exception of the lead time feature, all the features in the dataset are categorical. Notably, the dataset lacks information regarding the location, hotel names, or weather conditions during the bookings, which are significant factors affecting cancellations. However, despite these omissions, the provided features should provide sufficient information for prediction purposes, enabling the development of a generalized model.

A. Data Pre-processing

Before the data can be used it needs to be examined and cleaned so that the machine learning model can perform well [2]. the following step have been done on the dataset:

- Removing entries with NaN
- Removing duplicate entries
- Removing outlier

III. Exploratory Data Analysis

Exploratory Data Analysis (EDA) was performed on the dataset to gain deeper insights and assess whether the observed trends align with common intuition.

A. Lead time histogram

Observing the histogram in Figure 1, it becomes apparent that the majority of individuals book their hotel rooms within a 30-day timeframe prior to their stay. The histogram also reveals the presence of outliers exceeding 320 days, which were subsequently eliminated during the data pre-processing phase. This step was taken to address extreme values that could potentially skew the analysis and model performance.

B. Arrival month histogram

By examining the histogram depicted in Figure 2, a noticeable trend is observed where the density of bookings is skewed towards the left side. This indicates that a significant proportion of individuals tend to book hotel reservations during the latter months of the year. This inference suggests that there is a higher demand for hotel accommodations towards the end of the year, which aligns with the common practice of people taking vacations and traveling during the holiday season.

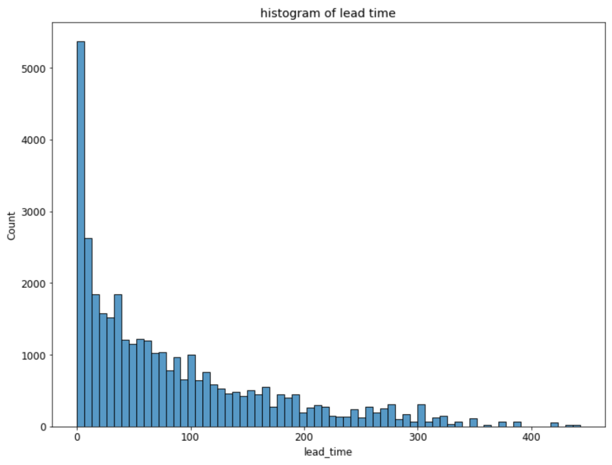

Fig. 1. lead time histogram

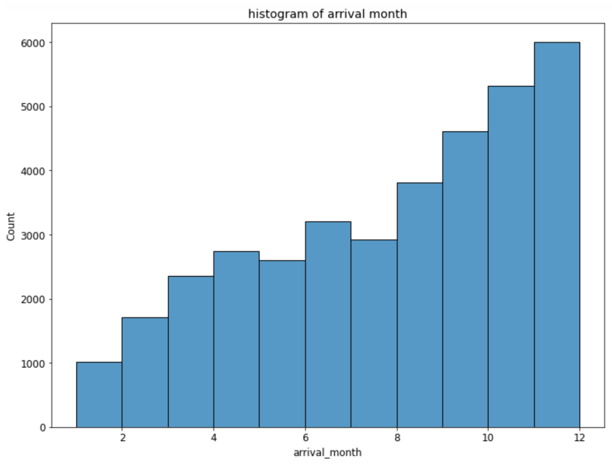

Fig. 2. Arrival month histogram

C. Market Segment

Based on the information provided in Figure 3, it can be deduced that the majority of data points or bookings are made online.

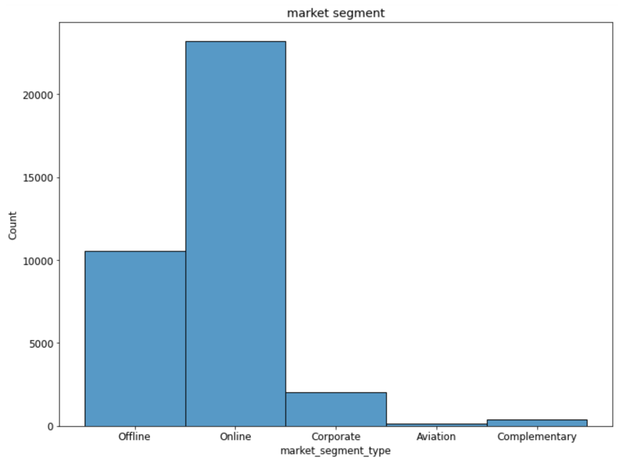

Fig. 3. Market histogram

D. Number of week nights

Based on the analysis of Figure 4, it is evident that a common duration of stay is approximately 2.5 nights. This implies that a significant number of individuals tend to stay for a duration of 2 nights and a half day.

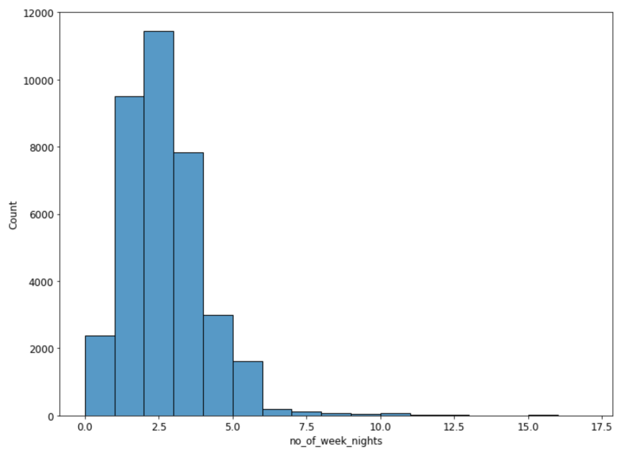

Fig. 4. Number of weeknights histogram

IV. Tree-based Prediction Models

A. Decision Tree

A decision tree is a machine learning algorithm that uses a tree-like structure to make predictions by recursively splitting the data based on feature values. It offers several benefits, including interpretability, intuitive representation of decision-making, the ability to handle both categorical and numerical features, and suitability for classification and regression tasks. Decision trees are valuable for feature selection, data exploration, and providing insights into the underlying patterns of the data [3].

The initial experimental implementation involved evaluating the raw performance of the scikit-learn decision tree algorithm, as presented in Table I. In this study, the performance of the predictive model was assessed using accuracy as the evaluation metric. The accuracy measure quantifies the proportion of correctly classified instances, providing an indication of the model's overall predictive capability.

Decision tree Training accuracy is	0.995
Decision tree Test accuracy is	0.863

TABLE I
Accuracy of unconstrained Decision tree

The unconstrained decision tree demonstrates promising results with high training accuracy. However, the high training accuracy suggests overfitting, indicating that the model may not generalize well to new test data.

Confusion matrix	Not-Cancelled	Cancelled
Not-Cancelled	4364	499
Cancelled	460	1708

TABLE II
CONFUSION MATRIX FOR DECISION TREE WITHOUT CONSTRAINTS

To tackle overfitting in tree-based models, one method is tree pruning, which involves limiting the tree depth. This helps prevent the model from capturing noise or specific patterns that may not generalize well to new data, resulting in a more simplified and generalized tree model. Pruning mitigates overfitting and enhances performance on unseen test data.

Decision tree with depth 20 training accuracy is	0.961
Decision tree with depth 20 test accuracy is	0.866

TABLE III
ACCURACY FOR DECISION TREE WITH DEPTH 20

The performance of a decision tree pruned to a depth of 20 is demonstrated in Table III and Table IV. It is noteworthy that while the training accuracy decreases, the test accuracy remains unchanged. This finding indicates that the pruned model achieves better generalization and is less susceptible to overfitting.

Confusion matrix	Not-Cancelled	Cancelled
Not-Cancelled	4391	472
Cancelled	468	1700

TABLE IV
CONFUSION MATRIX FOR DECISION TREE WITH DEPTH 20

Another method to address overfitting is by employing bagging, which involves generating multiple weak decision tree learners to form a strong algorithm. Bagging helps mitigate overfitting by reducing the impact of individual trees and promoting a more robust and generalized model. Random Forest, a tree-based algorithm, inherently incorporates bagging by constructing an ensemble of decision trees. This technique will be further discussed in the upcoming subsection.

B. Random Forest

Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. Each tree is trained on a random subset of the data, and the final prediction is determined by aggregating the predictions of all the trees. Random Forest is used over Decision tree to mitigates over-fitting by introducing randomness in two ways: by randomly selecting subsets of features for each tree and by bootstrapping the data, which creates diverse training sets. The combination of these randomized trees helps to reduce overfitting by averaging out individual tree biases and improving the generalization ability of the model [4].

Random forest algorithm using sci-kit learn [9] was used to check its performance against the algorithm written from scratch in the next subsection.

Random forest Using Sci-kit learn - training accuracy is	0.94
Random forest Using Sci-kit lear-test accuracy is	0.898

TABLE V
ACCURACY FOR RANDOM FOREST WITH DEPTH 17 AND 1200 TREES

As depicted in Table V, the Random Forest algorithm demonstrates superior performance compared to a decision tree pruned with a maximum depth of 20. Hence a random forest is better than a decision tree as it is also not sensitive to changes in training data.

C. Random Forest Using Sci-kit learn Decision Tree

An attempt was made to implement Random Forest from scratch, but due to several bugs encountered in the decision tree component of the algorithm, a middle ground approach was adopted. Instead of building the decision tree from scratch, the decision tree implementation from scikit-learn was utilized to execute the Random Forest algorithm from scratch.

Random forest from scratch - training accuracy is	0.934
Random forest from scratch- test accuracy is	0.894

TABLE VI
ACCURACY FOR RANDOM FOREST WITH DEPTH 17 AND 1200 TREES

Upon comparing Table V and Table VI, it is evident that the performance of the Random Forest algorithm implemented from scratch is comparable to that of the Random Forest implementation from the scikit-learn library

V. LEARNING

This section encompasses personal learnings derived from the experimentation with decision trees and Random Forest algorithms

A. Effect of One-Hot Encoding on Random Forest

In contrast to many models where One-Hot encoding often improves performance by providing higher resolution for the dataset, this approach does not hold true for the Random Forest algorithm. Random Forest tends to perform poorly when applied to datasets with high cardinality categorical variables [5]. To evaluate this, a test setup was implemented by running the Random Forest model on datasets that were both one-hot encoded and label encoded.

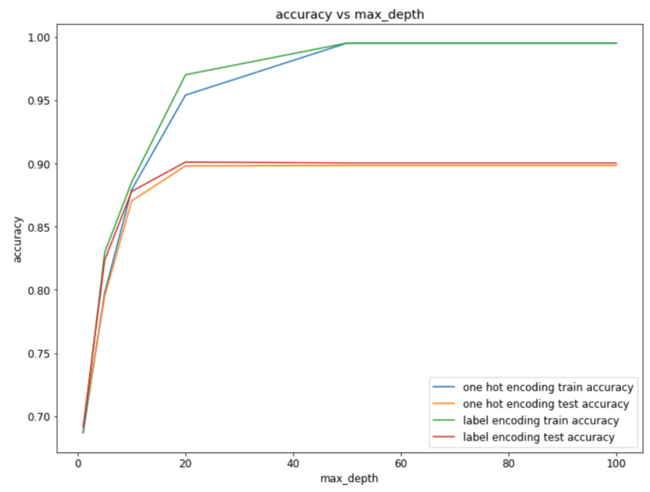

Fig. 5. Accuracy vs Max depth for One-Hot and label encoded data

Figure 5 illustrates the performance variation of both one-hot encoded and label-encoded datasets as the complexity of the model increases

Random forest with one hot encoding training accuracy is	0.798
Random forest with one hot encoding test accuracy is	0.795

TABLE VII
ACCURACY FOR RANDOM FOREST WITH DEPTH 5 AND 1200 TREES

Table VII and Table VIII present the accuracy values of two models at a depth of 5. The notable difference in accuracy between the two tables indicates that the model's performance is adversely affected when the data is one-hot encoded.

Random forest with label encoding training accuracy is	0.83
Random forest with label encoding test accuracy is	0.823

TABLE VIII
ACCURACY FOR RANDOM FOREST WITH DEPTH 5 AND 1200 TREES

In Random Forest, the algorithm splits the trees based on information gain, and the presence of sparsity in the data can impact this process. One-hot encoding introduces sparsity into the dataset, leading to most trees in the forest having longer branches compared to others. On the other hand, label encoding results in a more evenly spread distribution of branches in the underlying trees, which is generally more favorable for the model. It is important to note that scikit-learn random forest implementations can handle categorical data directly, without the need for explicit encoding.

B. Feature Scaling

According to theory, unlike many other prediction models, tree-based models such as decision trees and random forests are not sensitive to the scale of features. Therefore, there is no need to scale features to a normal distribution, as these models can effectively handle varying feature scales without any adverse effects on their performance [6].

Random forest without standard scalar training accuracy is	0.885
Random forest without standard scalar test accuracy is	0.877

TABLE IX
ACCURACY FOR RANDOM FOREST WITH DEPTH 10 AND 1200 TREES

Table IX and Table X show accuracy results for the performance of random forest on standard scaled and without scaled datasets. It was noted that although the results should be the same, there seemed to be a small degree of variation in the test error. Nevertheless, it was concluded that the model performed relatively similarly on both datasets.

Random forest with standard scalar training accuracy is	0.885
Random forest with standard scalar test accuracy is	0.871

TABLE X
ACCURACY FOR RANDOM FOREST WITH DEPTH 10 AND 1200 TREES

C. Polynomial Feature

Going into this experiment it was assumed that the performance would not vary since tree-based algorithms inherently perform feature selection. However, from Figure 6 the training error was observed to reduce, which may be due to the combination of important features in higher polynomial data causing a slightly better fit. As a compromise between computation and generalization, a polynomial of degree 1 was found to be good enough for the model.

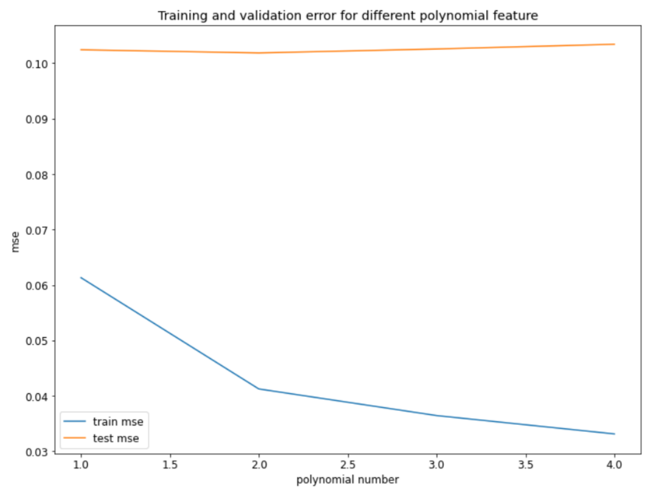

Fig. 6. Error vs Polynomial featured data

D. Ideal Number of Trees

The parameter n_estimators determines the number of trees in the forest. It is generally preferable to have a larger number of trees to learn the data effectively. However, a trade-off exists as an excessively high number of trees can considerably slow down the training process. Therefore, we conduct a parameter analysis to identify the optimal value that strikes a balance between performance and computational efficiency.

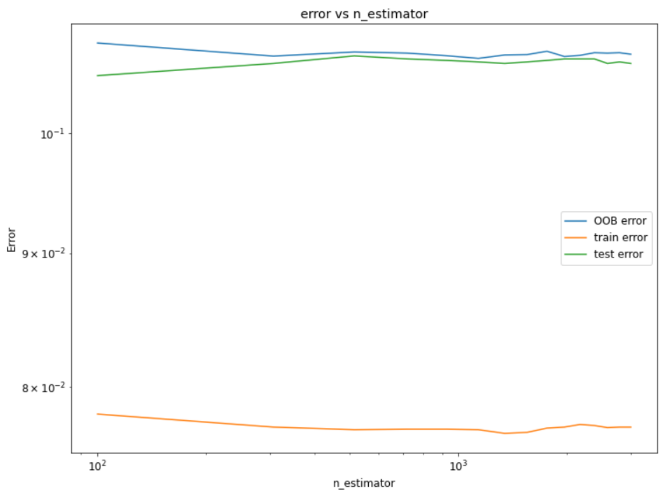

Fig. 7. Error vs Number of trees in Random forest

From Figure 7 it is evident that increasing the number of trees did not reduce the error by a large margin, and a value around 1000 was deemed ideal in terms of confidence in the model and computation.

E. Effect of Depth of Trees

The max_depth parameter determines the depth of each tree in the forest. By increasing the depth, the tree becomes capable of extracting more intricate patterns and finer details from the dataset.

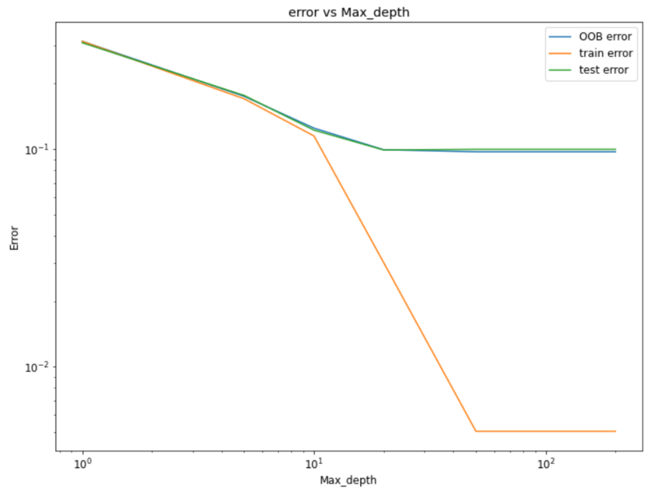

Fig. 8. Error vs Maximum depth of tree

Figure 8 shows that any tree in the forest with a depth greater than 20 starts to over-fit the data. so choosing a depth between 10-20 is ideal for this dataset.

F. Parameter Tuning

To find the best parameters for the random forest model, randomizedsearchCV was used to take a distribution of different parameters and then pick each parameter with replacement. [7] [8].

Randomized SearchCV training Accuracy	0.994
Randomized SearchCV Test Accuracy	0.905

TABLE XI
ACCURACY FOR RANDOM FOREST WITH TUNED PARAMETERS - DEPTH 150 AND 1964 TREES

By examining the accuracy and parameter values in Table XI, it is apparent that the accuracy is high, indicating the possibility of overfitting. Additionally, the parameter value is also high, which disregards computational limitations

Confusion matrix	Not-Cancelled	Cancelled
Not-Cancelled	4586	253
Cancelled	438	1978

TABLE XII
CONFUSION MATRIX FOR RANDOM FOREST WITH TUNED PARAMETERS - DEPTH 150 AND 1964 TREES

Randomized searchCV, does not consider the computational constraints when selecting parameters. However, based on the analysis conducted in previous sections, it has been determined that setting the depth of the trees to 17 and using 1200 trees results in a well-fitted and generalized model. This observation is confirmed by the performance depicted in Table XIII.

Random forest Using Sci-kit learn - training accuracy is	0.94
Random forest Using Sci-kit learn - test accuracy is	0.898

TABLE XIII
ACCURACY FOR RANDOM FOREST PARAMETER FROM PAST ANALYSIS - DEPTH 17 AND 1200 TREES

Confusion matrix	Not-Cancelled	Cancelled
Not-Cancelled	4623	240
Cancelled	478	1690

TABLE XIV
CONFUSION MATRIX FOR RANDOM FOREST PARAMETER FROM PAST ANALYSIS - DEPTH 17 AND 1200 TREES

CONCLUSION

In conclusion, this paper has undertaken a comprehensive process that involved cleaning, exploring, and analyzing a dataset sourced from Kaggle to identify various trends in the hotel industry. Additionally, Decision Tree and Random Forest algorithms were thoroughly studied and analyzed, considering different model parameters. Through this analysis, the best-performing model was identified, exhibiting a remarkably high accuracy in predicting hotel booking cancellations.

ACKNOWLEDGMENT

I would like to express my gratitude to Professor Mahesh Varanasi and Teaching Assistant Altug Karakurt for their guidance and support during the project. Additionally, I would like to acknowledge the insightful discussions and motivation provided by my fellow peers during this project.

References

[1] https://www.kaggle.com/datasets/ahsan81/hotel-reservations-classification-dataset

[2] https://www.obviously.ai/post/data-cleaning-in-machine-learning

[3] https://towardsdatascience.com/decision-tree-in-machine-learning-e380942a4c96

[4] https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[5] https://towardsdatascience.com/one-hot-encoding-is-making-your-tree-based-ensembles-worse-heres-why-d64b282b5769

[6] https://towardsdatascience.com/how-data-normalization-affects-your-random-forest-algorithm-fbc6753b4ddf

[7] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[8] A. M. Coroiu, "Tuning model parameters through a Genetic Algorithm approach," 2016 IEEE 12th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 2016, pp. 135-140, doi: 10.1109/ICCP.2016.7737135.

[9] https://scikit-learn.org/stable/index.html

[10] https://pandas.pydata.org/pandas-docs/stable/index.html