An analysis of hotel reservation data

Group 1

Chua Kang Wei	**(A0173557W)**
Frederick Liew Yong Lun	**(A0168902Y)**
Lee Yong Jie, Richard	**(A0170235N)**
Nicholas Alexander	**(A0171494Y)**

Presentation recording:
https://nus-sg.zoom.us/rec/share/67MOkySC6AbaXumTEPtk4Gipw88_F9JuyhpQeydpW4KmBJ3QvDEvkBDA6aceegM.gePEwxsGXmhjiFBt
Password: QFHU83@Q

DSA4211 High-Dimensional Statistical Analysis
Department of Statistics and Applied Probability
National University of Singapore
2020/2021

1 Introduction

One of the problems that the hotel industry may face nowadays is to match the (unlimited) needs of the customers to the limited rooms available in the hotel. Customers may book for a stay in advance and cancel it after, potentially costing the company time/effort preparing for the reservations.

The initial dataset was first passed through a Sure Independence Screening (SIS) procedure to determine the more significant variables, and were subsequently analysed with three different methods, listed below.

- Logit models
- K-nearest neighbours
- Tree-based methods

2 Data

The dataset was obtained from the "Hotel booking demand" problem from Kaggle, obtainable at `https://www.kaggle.com/jessemostipak/hotel-booking-demand`. This dataset has $n = 119390$ and $p = 31$.

The predictors consist of numerous details concerning the booking of each customer's stay, which can be classified into a few general categories:

- Durations - Arrival dates, how many days was the booking made in advance, how long the stay is
- Stay information - Room type (reserved/assigned), average daily price, adults/children/baby count
- Reservation channel - How the booking was made (if it was made through an agent/company)
- Past booking history - Whether the customer has cancelled reservations before, how many bookings were cancelled before

5 of the rows have NA values, and one of the columns `reservation_status` is highly correlated to the label of the dataset and was dropped from the dataset. The final dataset has $n = 119385$ and $p = 30$ after the adjustments.

The label from the dataset (`is_cancelled`) is a binary variable and hence classification methods will be explored heavily in this problem.

3 Problem Statement

Given the numerous attributes that come with each booking, are we able to predict if the particular booking is cancelled?

4 Sure Independence Screening

With the already high n from the dataset, considering all possible predictors in the subsequent models may affect the runtime of the algorithms greatly. A Sure Independence Screening test was conducted on the original dataset to determine the more impactful predictors to be used for the methods later on.

The SIS procedure was run on the original dataset with a LASSO penalty, and the best subset of predictors was selected via a 10-fold cross validation for a maximum of 5 iterations. The 9 variables that SIS selected were:

- `lead_time`
- `adr`
- `required_car_parking_spaces`
- `total_of_special_requests`
- `market_segment_Groups`
- `market_segment_Offline_TA_TO`
- `deposit_type_Non_Refund`
- `agent_240`
- `customer_type_Transient_Party`

The dataset (pruned with the above 9 predictors) was then utilized in all the subsequent models.

5 Logit models

A logistic regression model was fitted across all the 9 variables that SIS selected. The P values in the final model were all significant ($< 2 \times 10^{-16}$) apart from the P value of 0.692 obtained with the coefficient for `required_car_parking_space`. The final logit model obtained is:

$$\begin{aligned}\log \frac{p(x)}{1-p(x)} = &- 9.159 + 0.00348 * lead\_time + 0.01147 * country + 0.472 * market\_segment \\ &+ 0.5532 * reserved\_room\_type - 0.5496 * assigned\_room\_type \\ &+ 3.983 * market\_segment\_offline\_TA\_TO + 0.0051583 * \_deposit\_type\_non\_refund \\ &- 0.1783 * required\_care\_parking\_space - 0.6067 * customer\_type\_transient\_party\end{aligned}$$

The ROC was also plotted out with the training dataset. (Figure 1)

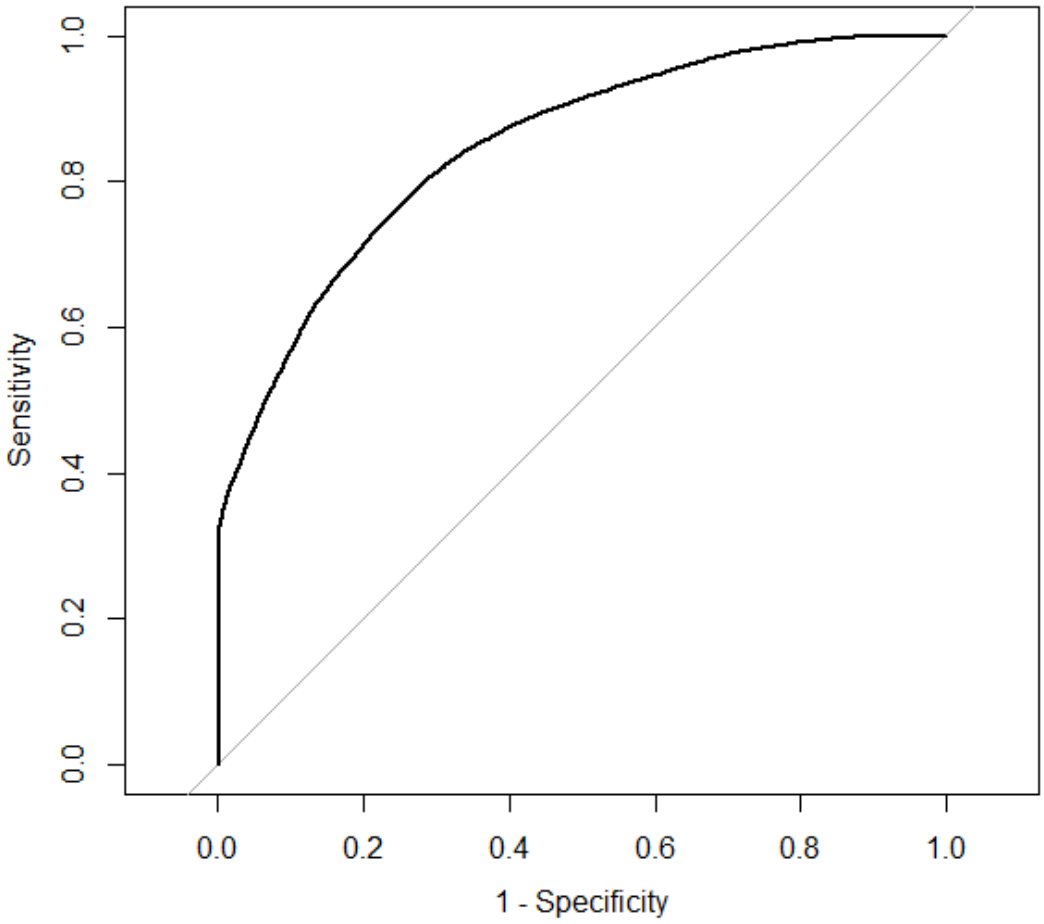

Figure 1: ROC curve for the Logit model (training dataset)

The AUC obtained was 0.8473. To obtain the threshold it is assumed that the cost of obtaining a false negative and a false positive to be the same. However, some may suggest that the cost of predicting a false negative (`is_canceled` to be 0 (not cancelled) even though it is 1 (cancelled)) to be more severe since the hotels may lose revenue for cancellations. Nevertheless, we measured the threshold which will maximize the sensitivity and specificity with the Youden statistic and the threshold was clamped between 0.3 and 0.8 to avoid significantly small specificities and sensitivities.

The Youden statistic is measured by taking the sum of the true positive rate and the false positive rate, then taking 1 off the sum. The maximised threshold obtained from the training dataset was 0.36. The confusion matrices were plotted out for the training and test datasets with the logit model. (Figures 2 and 3)

$$Youden\ statistic = TPR + FPR - 1$$

		Actual	
		0	1
Predicted	0	46359	9036
	1	13680	26433

Figure 2: Confusion matrix for the Logit model (training dataset) (TPR = 0.745, TNR = 0.77)

		Actual	
		0	1
Predicted	0	11642	2235
	1	3484	6516

Figure 3: Confusion matrix for the Logit model (test dataset) (TPR = 0.745, TNR = 0.77)

The logistic regression model performs considerably well with a high AUC and high sensitivity and specificity. The test accuracy obtained was 76.04%.

6 K-nearest neighbours

Using the training set that consists of 80% of the total samples, 10-fold cross validation was used to determine the optimal k for the k-nearest neighbours algorithm. The average CV accuracy can be seen in Figure 4, and the plot against k can be seen in Figure 5

k value	Average CV accuracy
1	100.00
2	99.98429
3	99.97801
4	99.95707
5	99.93822
6	99.93194
9	99.90263
17	99.81991
45	99.61678
309	98.44411

Figure 4: Average CV accuracy for the KNN algorithm for various k

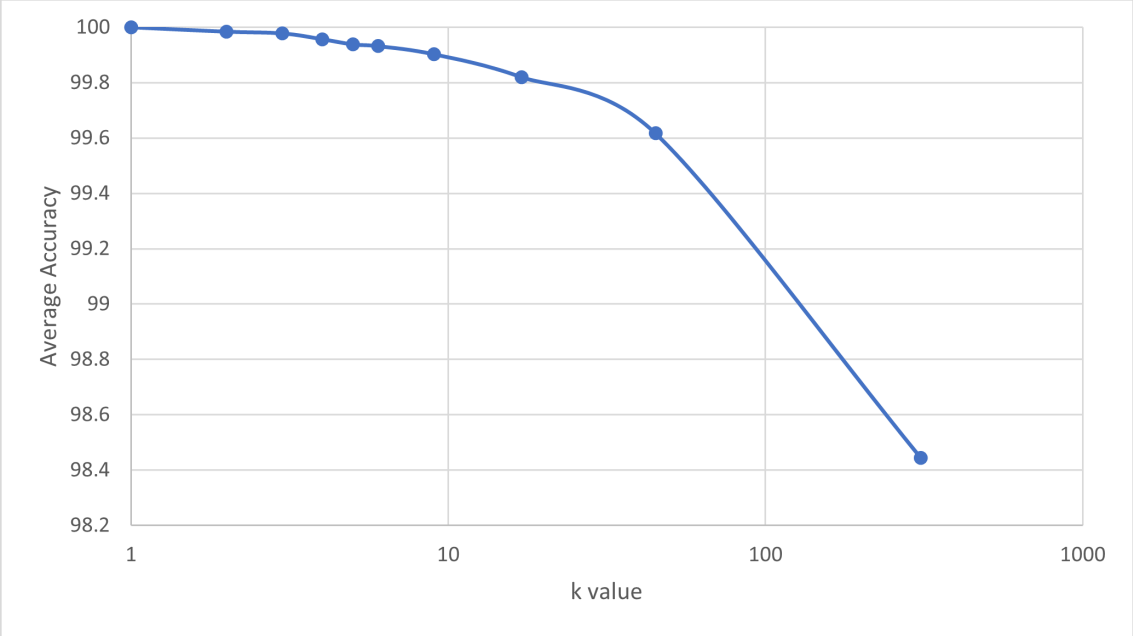

Figure 5: Plot of average CV accuracy against k value

As shown, the average accuracy across the folds decreased as k increased, which is surprising as the optimal k usually lies near the square root of the total number of samples.

Upon obtaining the optimal k value of 1, we trained a model with the full training set, before using the model to predict on the test set. The following confusion matrix for the test dataset was obtained. (Figure 6)

		Actual Outcome	
		Not cancelled	Cancelled
Predicted outcome	Not cancelled	15012	2
	Cancelled	2	8861

Figure 6: Confusion matrix for the KNN model (test dataset)

The final accuracy obtained was 99.983%, which could be due to the high number of training samples, allowing for a better approximation with the relatively small number of predictors. One of the issues with the k-nearest neighbours model is that there is no way to modify it to either reduce the number of false positives or negatives in exchange for a lower total accuracy. This would prevent the hotels from being able to take excess bookings on rooms that customers which are predicted to cancel have booked. However, a model with high accuracy would still allow the hotel to keep a smaller amount of rooms on hold for the amount of customers who have been predicted to cancel.

7 Tree-based methods

7.1 Decision trees

A decision tree was first fitted on the dataset using entropy as the splitting method. The residual mean deviance obtained was 0.9162 and the misclassification error rate was 0.2178. A total of 6 variables were used in tree construction as seen in Figure 7.

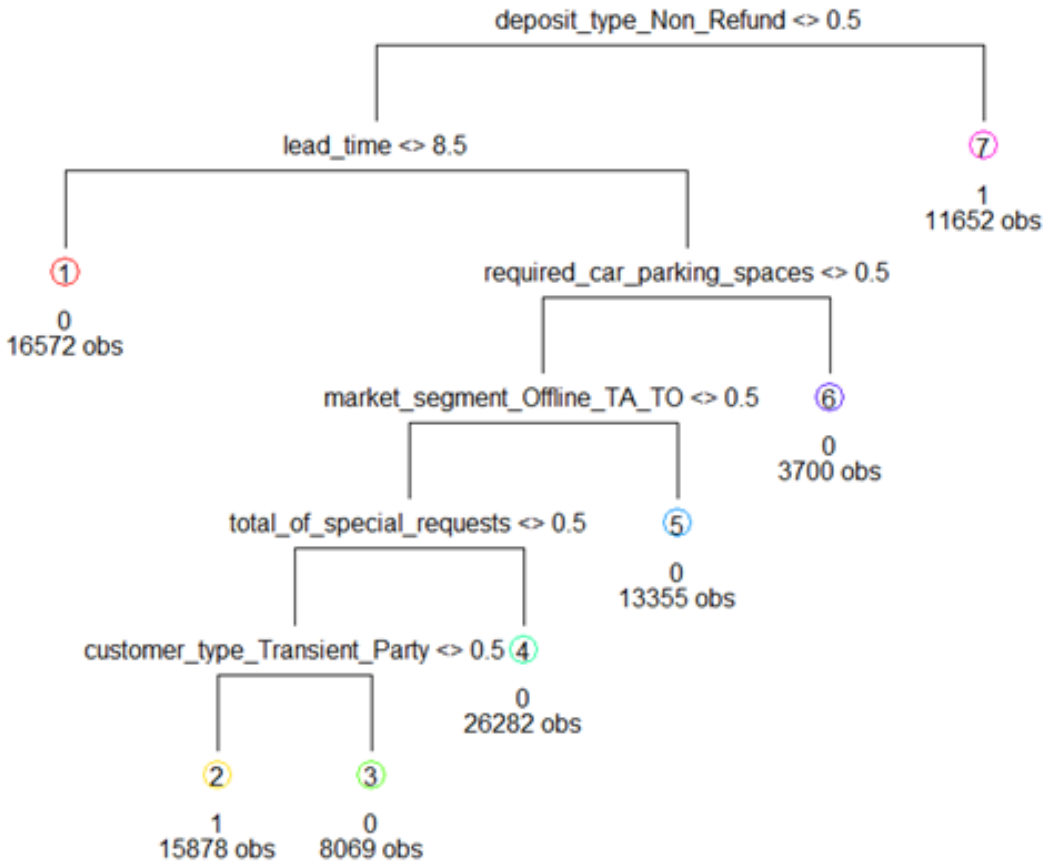

Figure 7: Plot of test error against B for various decision tree methods

Another decision tree was fitted using Gini index as the splitting method, but the resulting tree contains 71 terminal nodes which is much higher than 7 terminal nodes when entropy was used. The residual mean deviance of the fitted tree was 0.8772. The complexity of the tree is much higher which makes it harder to interpret, even though the training error rate is slightly lower at 0.212 instead of 0.218 when entropy was used.

To check if there is a further need to prune the decision tree that relied on entropy, we used cost complexity pruning and 10-fold cross-validation to determine the optimal tree complexity level. As seen in Figure 8, the cross-validation error was lowest at size 7.

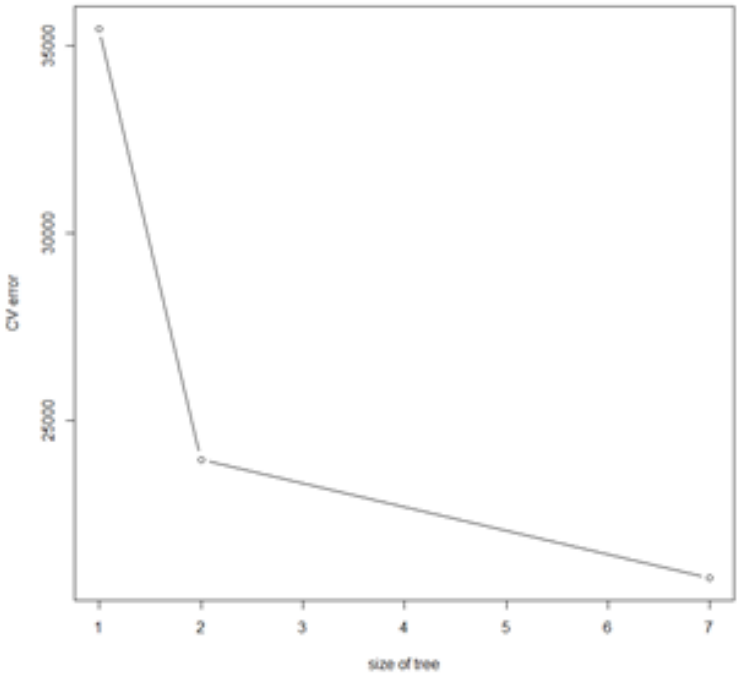

Figure 8: Plot of CV error against the size of the tree for pruning

It was also noted that the training error for the decision tree is 0.218, whereas the test error is 0.216. This suggests that there is minimal overfitting or underfitting.

		Actual	
		0	1
Predicted	0	53607	14371
	1	6432	21098

Figure 9: Confusion matrix for the decision tree model (training dataset) (Error rate = 0.218)

		Actual	
		0	1
Predicted	0	13508	3528
	1	1618	5223

Figure 10: Confusion matrix for the decision tree model (test dataset) (Error rate = 0.216)

7.2 Random Forest

We attempted fitted a random forest model to the dataset using $B = 100$ to $B = 500$ decision trees. As seen from Figure 11, using $m = p$ (bagging) produced the smallest test error at around 0.182. In comparison, using $m = \frac{p}{2}$ produced test error at around 0.188 to 0.19 and using $\sqrt{p}$ produced a test error of around 0.195. As such, bagging seems to be the best method and it was also observed that increasing B further did not reduce the test error significantly.

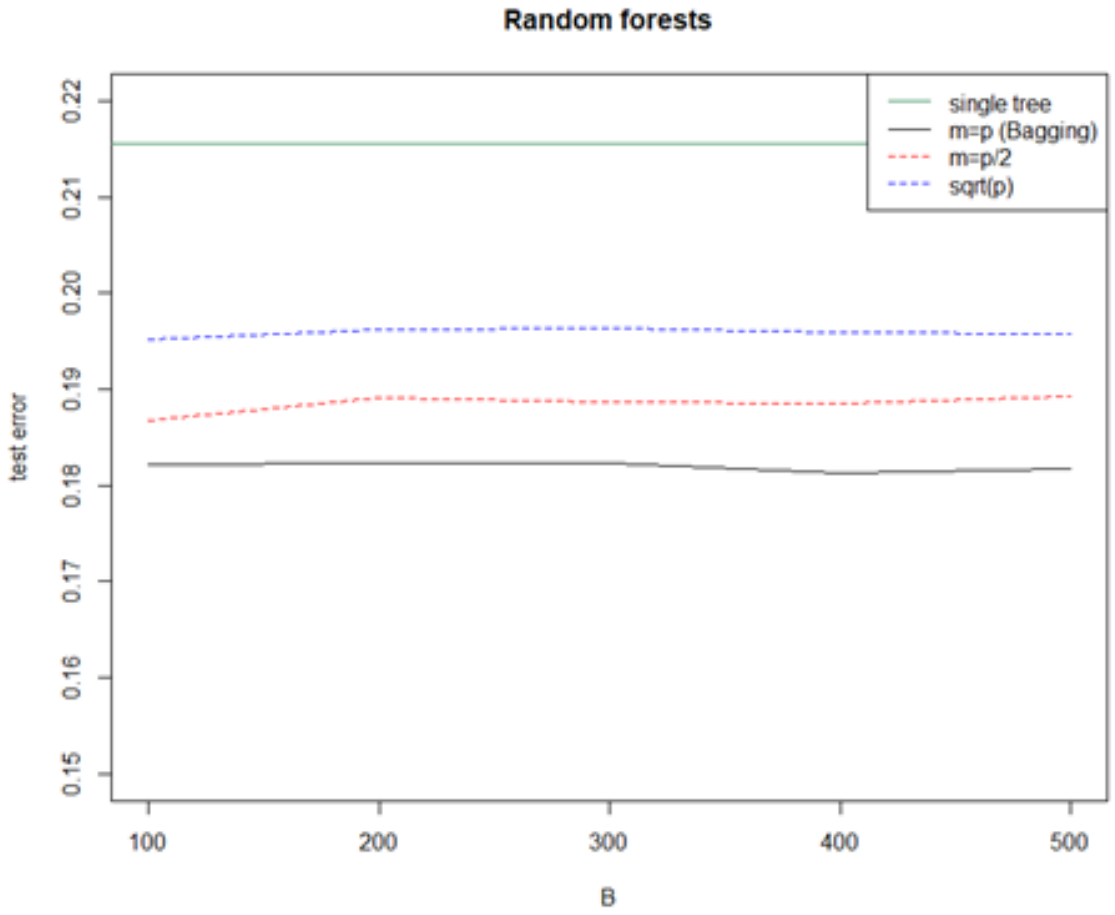

Figure 11: Plot of test error against B for various decision tree methods

8 Evaluation and comparisons

8.1 Accuracies

Model	Test accuracies
Logit	76.04%
KNN (1-nearest neighbour)	99.983%
Decision tree (Entropy)	78.4%
Random forest (Bagging)	81.8%

Figure 12: Summary of test accuracies for the various models

The KNN is the clear winner in terms of accuracy here and its near perfect accuracy may be attributed to the high n of the dataset. The random forest method does significantly well as well, edging out an extra 3-6% in accuracy over the last two models, logit and (simple) decision trees.

8.2 Runtimes

All timings were taken on a i7-7700 3.60GHz processor with 8 logical cores and 16 Gb of memory.

Model	Runtime(s)
Logit	1.23
KNN (10 k values * 10 folds/k)	333.8
KNN (Single k value)	22.68
Decision tree (Entropy)	0.41
Decision tree (Gini)	0.71
Decision tree (CV Pruning)	5.02
Random forest (3 * 5 runs)	280.34

Figure 13: Summary of runtimes for the various models

Note: The evaluation of each individual fold for KNN as well as for each set of hyperparameters for the Random Forest was done with the help of parallelization. 8 cores were utilized for KNN while only 6 cores were utilized for the Random Forest execution (primarily due to the limited memory of the machine. While it may seem impressive that KNN only took an average of 3 seconds per run, running it alone in sequential provides a much different runtime than expected.

The KNN model took the longest for a single run taking about 23 seconds, it is only with the help of parallelization that a manageable runtime could be achieved to determine (via cross validation) the best parameters to use. Random forest was also slow in that regard, taking an average of 19 seconds per set of hyperparameters. Although the rest of the models produce less than ideal accuracies as seen previously, they run near instantaneously which can be quite ideal for companies that do not have the luxury of time to crunch out their datasets.

9 Conclusion

In summary, various models were fitted to try to predict whether a particular booking from a customer will be cancelled.

The K-nearest neighbours method performed the best with the dataset that we had, achieving a near perfect test accuracy. The other two models achieved relatively high accuracies as well between 75% - 82%. The runtimes for the better-performing methods, however, leave much to be desired, taking about 22 seconds for a single KNN run.

All in all, predicting if a particular booking will be cancelled can be done with a very high accuracy, but this analysis is not without its drawbacks.

On the topic of the KNN's runtimes, it may be a cause for concern for other potential users of these methods since in a real life scenario, the amount of data streaming in may be a lot higher than the dataset provided, and there are possibly many more predictors that the hotel may take into account. Having such a slow-running algorithm to determine if certain customers should be prioritized may have adverse effects on the way the hotel does operations, and care must be taken.

In addition, certain variables presented to us in the dataset may not be immediately available to use from the start, and there might be imperfect information should the customer decide to opt for a different service halfway through their stay.

Especially in the local context, there may also be a different system for booking and one must consider the possible cultural differences when attempting to apply similar methods here. The significance of each of the predictors may be vastly different from the analysis being done in this report.