HOTEL BOOKING CANCELLATION PREDICTION

EAS 506 : STATISTICAL DATA MINING 1

PROJECT REPORT

Submitted to - Dr Saptarshi Chakraborty

FALL 2022

Bharatwaj Majji (50442312)
Jayanth Puthineedi (50442725)
Vishnu Bhadramraju (50441735)

Table of Contents

CHAPTER 1

1.1 ABSTRACT

Every year there is significant growth regarding online travel. The majority of contributions to online travel come from the hotel industry. Advance reservation for hotels is a wonderful solution but online cancellation of booking is currently one of the problems that hotel management systems are facing. It's a risk for hotels when an advance reservation gets cancelled at the last minute, the hotel cannot do much but give the room at a lesser price or even face a total loss in case of no further booking. So the opportunity to earn some higher revenue is lost.

Therefore, there is an opportunity for a prediction model to fill this gap and help hotels minimize profit loss. Our aim is to find a solution that allows hotels to accurately predict the demand and help them overcome revenue issues with the cancellation and improve their inventory allocation procedures by evaluating and analyzing machine learning models such as Logistic Regression, KNN, Decision Tree, Random Forest and AdaBoost.

Keywords: Logistic Regression, KNN, Decision Tree, Random Forest, AdaBoost.

CHAPTER 2

2.1 SIGNIFICANCE OF THE PROBLEM

One of the significant factors that affect demand management in the hospitality industry is booking cancellation. Even though the advance booking option helps the hotels to plan the stay of customers in advance, the cancellation option puts the risk on the management. Last-minute cancellations of booking through the hotel management in a situation to sell the room for a lesser price or not at all. Booking cancellations impede the hotel management's inventory to go as planned which is very important in revenue management for hotels.

2.2 POTENTIAL

How great it would be if the hotels had a model to predict if the guest will make it to the hotel or cancel their booking. This prediction can help hotels save and effectively plan other logistics like food and personnel arrangements. This might help hotels to make additional money by offering the room likely to be cancelled to other customers. Our main goal is to come up with a solution that allows hotels to correctly predict the demand and help them overcome revenue issues with cancellation and maximize profits. The dataset we selected will aim at the development of a model to predict the likelihood of a hotel booking being cancelled. The variables chosen for this model are not only limited and can be used for cancellation prediction problems but also can be used for solving more problems.

2.3 DATA SOURCES

The dataset is taken from Kaggle. The dataset consists of data from two different hotels, H1 is a resort and the other is a hotel H2 in the city. The data is gathered between 1st July 2015 and 31st August 2017.

2.4 DATASET DESCRIPTION

- There are 31 features: hotel, lead time, guests, company, car parking spaces, meal etc.
- The dataset contains 0.1 million records.
- The dataset contains both categorical and numerical columns

  hotel*: indicate which hotel H1 or H2*

  is_canceled*: show if the booking is cancelled or not. 1 if cancelled 0 if not cancelled*

  lead_time*: The no.of days between booking and arrival date*

  arrival_date_year*: year of arrival date*

  arrival_date_month*: month of arrival date*

  arrival_date_week_number*: week number of the year of arrival date*

  arrival_date_day_of_month*: day of month*

  stays_in_weekend_nights*: number of weekend nights that guests booked the hotel.*

  stays_in_week_nights*: number of weeknights that guests booked the hotel*

  adults*: Number of adults staying*

  children*: Number of children staying*

  babies*: Number of babies staying*

  meal*: This indicates the type of meal chosen. BB(Breakfast), FB(Full board), HB(Halfboard)*

  country*: country of customer*

  market_segment*: it says which market segment. TA means Travel Agents and TO means Tour Operators.*

  distribution_channel*: it says about the booking distribution channel.*

  is_repeated_guest*: indicates if the guest is repeated (1) or not(0)*

  previous_cancellations*: shows the number of previous cancellations by the customer.*

  previous_bookings_not_canceled*: shows the number of previous bookings not cancelled by the customer.*

  reserved_room_type*: this specified the code of the room type reserved.*

  assigned_room_type*: this specified the code of the room type assigned.*

  booking_changes*: indicates the number of changes made to the original* *booking.*

deposit_type: *indicates what type of deposit the customer chose. No Deposit, Non-Refund, Refundable*

agent*: id of an agent who made the booking*

company*: id of the company that made the booking*

days_in_waiting_list: *number of days the booking was in waitlist before getting confirmed*

customer_type: *specifies the type of booking Contract, Transient-party, Group, Transient*

adr: *average daily rate*

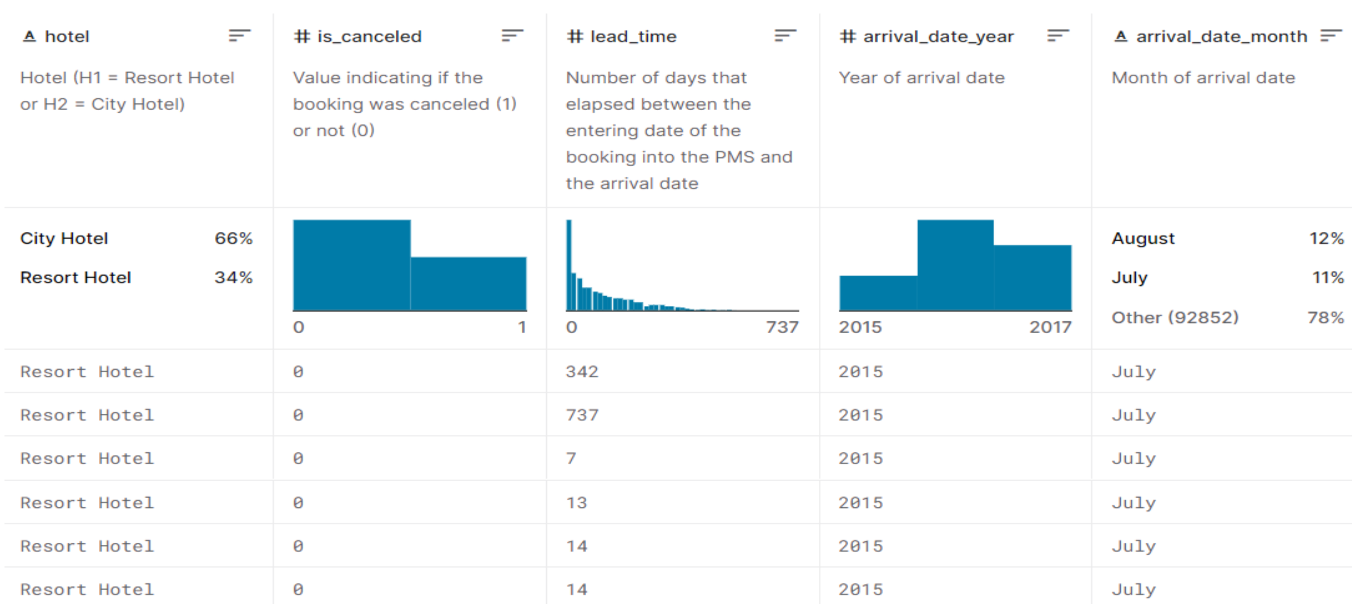
required_car_parking_spaces: *number of car parking space required*

total_of_special_requests: *gives the count of special requests made by the customers.*

reservation_status: *gives the latest status of the booking*

reservation_status_date: *the date of the last status change recorded.*

2.5 Overview



hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month
Hotel (H1 = Resort Hotel or H2 = City Hotel)	Value indicating if the booking was canceled (1) or not (0)	Number of days that elapsed between the entering date of the booking into the PMS and the arrival date	Year of arrival date	Month of arrival date
City Hotel 66% Resort Hotel 34%	0 – 1	0 – 737	2015 – 2017	August 12% July 11% Other (92852) 78%
Resort Hotel	0	342	2015	July
Resort Hotel	0	737	2015	July
Resort Hotel	0	7	2015	July
Resort Hotel	0	13	2015	July
Resort Hotel	0	14	2015	July
Resort Hotel	0	14	2015	July

CHAPTER 3

3.1 Data Visualization and Pre-analysis

In this chapter we preprocessed and analyzed our dataset for meaningful insights. To gain accurate insights we did a little data cleaning to remove unnecessary noise from the data. The following are the data cleaning methods we adopted for our data set.

- Handled missing values
- Replaced missing values with average mean and most frequent values.
- Analyzed pairwise association between highly correlated features.
- Combine multiple features to get a new feature.
- Removed redundant columns.

Handling missing values:

We learnt that some of our columns in the dataset contain NA values and missing values. We checked our data set to learn what all columns with NA values and missing values were, and we found that the columns children, country, agent and company are the columns with missing and NA values.

```
**Columns with MissingValues**
```{r}
cat("Columns with NA values - ", names(which(sapply(df, function(x) any(is.na(x))))), "\n")
cat("Columns with NULL values - ", names(which(sapply(df, function(x) any(x=='NULL')))), "\n")
```

Columns with NA values - children
Columns with NULL values - country agent company
```

Replacing missing values with average mean and most frequent values:

After finding the missing values and NA values columns. Here we replaced the columns with average mean and frequent values.

```
**Handle Missing Values columns**
```{r}
df$children = ifelse(is.na(df$children), 0, df$children)
df$country = ifelse(df$country == 'NULL', 'Unknown', df$country)
df$agent = ifelse(df$agent == 'NULL', 0, df$agent)
df$company = ifelse(df$company == 'NULL', 0, df$company)
df$guests_stayed = df$adults + df$children + df$babies
df$nights_stayed = df$stays_in_week_nights + df$stays_in_weekend_nights
```

```

## Combining multiple features to get a new feature

After handling missing values we got to learn that some columns represent similar types of information. So, we combined multiple features to get new features.
For example, we formed a new feature for guests stayed instead of having three different features such as adults, children, and babies.

```
df$guests_stayed = df$adults + df$children + df$babies
df$nights_stayed = df$stays_in_week_nights + df$stays_in_weekend_nights
```
```

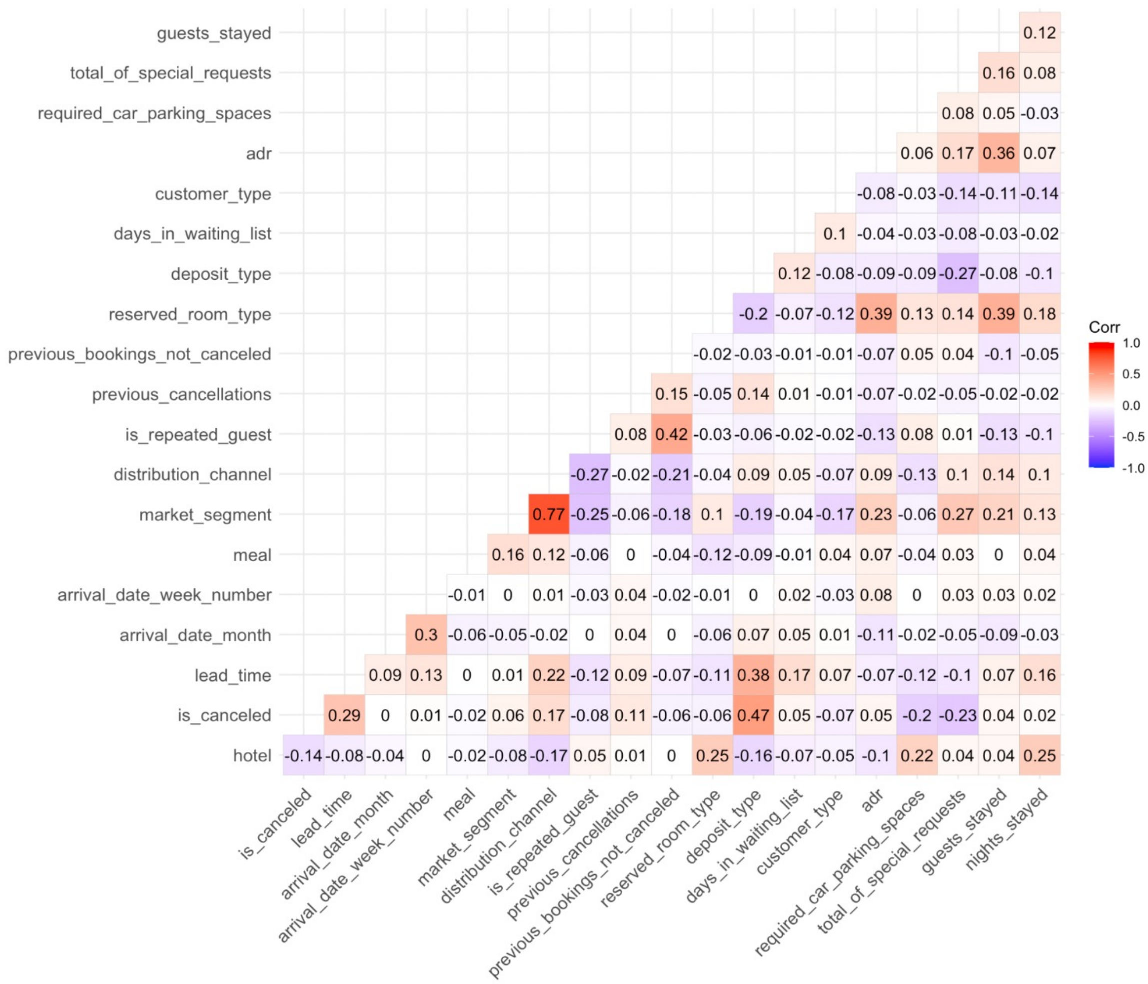
Analyzed pairwise association between highly correlated features

Here we analyzed the features to find out highly and negatively correlated features. The following are the features we acknowledged from our dataset's
Highest correlated features:

1. Lead Time
2. ADR(Average Daily Price)
3. Previous Cancellations
4. Days in Waiting List

Negatively correlated features:

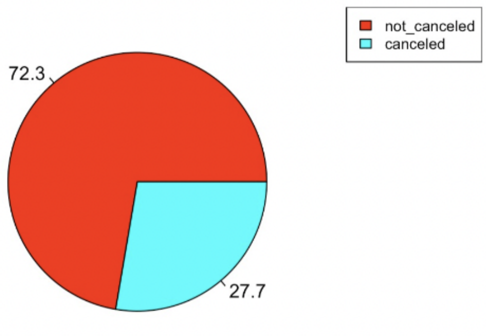
5. Total Special Requests
6. Required Parking Spaces
7. Booking changes
```



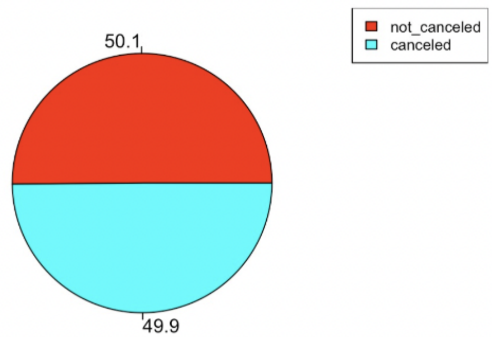
The above figure explains the correlation matrix among various features in the dataset.

Analysis of correlated Features (Pair-Wise Association)

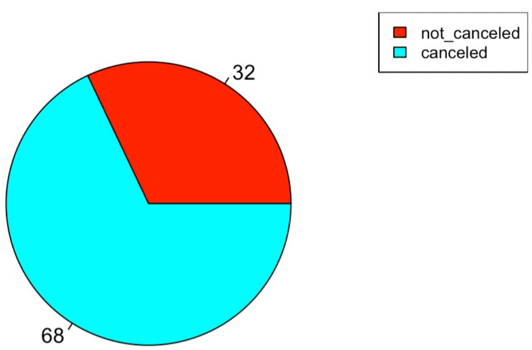
Lead Time vs Cancelations:



Lead Time <**100** days



Lead Time >=**100** days & < **365** days



Lead Time >= **365** days

From this we can understand that as lead_time increases the chances of booking cancellation as well increases.

Previous Cancellations vs Cancelations:

Never Previously Cancelled Bookings	**33.9%**
Previously cancelled only once	**94.4%**
Previously cancelled more than 10 times	**99.3%**

The table shows the percentage of cancellations by three different types of booking. As the number of previous cancellations increases the chances of booking cancellations as well increase.

Total of Special Requests vs Cancelations (highly negatively correlated feature):

Even though it is a negatively correlated feature, as the number of special requests increases the booking cancellation percentage decreases.

Parking Spaces vs Cancelations (highly negatively correlated feature):

Non-Cancelled Bookings:

required_car_parking_spaces <int>	V1 <int>
0	67750
1	7383
2	28
3	3
8	2

Cancelled Bookings:

required_car_parking_spaces <int>	V1 <int>
0	44224

From this, we can understand the model can tune in such a way that if the number of required spaces is zero the booking can be cancelled which is not the case ideally. So, we can ignore this feature while modelling.

Hotel vs Cancelations:

Here we are checking the feature Hotel against our target cancellation feature with the respective month. The hotel feature contains mainly two variables, which are hotel and resort.

Description: df [12 × 2]

	resort_cancel <dbl>	city_cancel <dbl>
January	0.1481988	0.3966809
February	0.2562037	0.3828802
March	0.2287170	0.3694642
April	0.2934331	0.4632353
May	0.2877213	0.4437561
June	0.3307061	0.4469217
July	0.3140171	0.4087537
August	0.3344912	0.4009796
September	0.3236808	0.4202703
October	0.2751055	0.4297173
November	0.1891670	0.3812256
December	0.2382931	0.4211036

From the above stats, we can understand that wrt monthly city hotels have more booking cancellations compared to resort hotels according to arrival months.

Meal vs Cancelations:

meal <chr>	percent_cancellations <dbl>
BB	0.3738490
FB	0.5989975
HB	0.3446035
SC	0.3723944
Undefined	0.2446536

From this, we can understand that FB meal is the most frequently cancelled booking. And meal Undefined can relate to SC no-meal.

Market Segment vs Cancelations:

market_segment <chr>	percent_cancellations <dbl>
Aviation	0.2194093
Complementary	0.1305518
Corporate	0.1873466
Direct	0.1534190
Groups	0.6106204
Offline TA/TO	0.3431603
Online TA	0.3672114

From the above stats we can understand that cancellations are higher for Groups, Offline and Online TA/TO travel and tour operator bookings.

Distribution Channel vs Cancelations:

distribution_channel <chr>	percent_cancellations <dbl>
Corporate	0.2207578
Direct	0.1745988
GDS	0.1917098
TA/TO	0.4102585
Undefined	0.8000000

From the above stats we can understand that cancellations are higher for TA/TO travel and tour operator bookings

CustomerType vs Cancelations:

customer_type <chr>	percent_cancellations <dbl>
Contract	0.3096173
Group	0.1022530
Transient	0.4074632
Transient-Party	0.2542987

From the above stats we can understand that cancellations are higher for Transient customer_type bookings.

DepositType vs Cancelations:

deposit_type <chr>	percent_cancellations <dbl>
No Deposit	0.2837702
Non Refund	0.9936245
Refundable	0.2222222

From the above, we can see that non-refund bookings have 99 percent cancellations which is weird since ideally non-refund transactions tend to have lower cancellations. Looks like the values of cancelled and not-cancelled must have swapped up for non-refund transactions. Let us check this while modelling.

CHAPTER 4

4.1 DATA CLEANING AND DATA PREPARATION

In this chapter we performed data cleaning and data preparation after our pre-analysis. After the pre-analysis, we identified a few unwanted columns and some redundancy in our data. Removing the anomalies from the data aids our model to become more generalized rather than falling into the overfitting and underfitting traps. We performed the below process after our pre-analysis.

- Drop unwanted columns.
- Removing not valid rows.
- Encode categorical features.
- Standard scale the numerical features.

Drop unwanted columns:

```
df <- subset(df, select = -c(agent, company, booking_changes, arrival_date_day_of_month, arrival_date_year))
df <- subset(df, select = -c(reservation_status, reservation_status_date, assigned_room_type, country) )
```

- **Numerical Columns:**
  - agent & company: These columns are uninformative since they contain discrete codes for the agents and company using which the booking is made.
  - booking_changes: Could constantly change over time and has not much effect on the predictor.
  - arrival_date_day_of_month & arrival_date_year: Prevents the model from generalising, since we have arrival_week information that would be sufficient.
- **Categorical Columns:**
  - reservation_status: It has values Check-Out, Cancelled and No-Show which means not-cancelled and cancelled considering this feature can cause the model to overfit.
  - reservation_status_date: The date when the reservation_status is last changed is not relevant.

- assigned_room_type: This is irrelevant and over reserved_room_type makes more sense since the booking can be cancelled only before checking in which means the room is assigned.
- Country: There are many countries and not uniformly distributed so there are higher chances that this model can prevent the model from generalising.

Removing not valid rows.

We observed a few rows in our guest feature contain zero. Here we removed the rows with zero guests.

```
**Remove rows with zero guests**
```{r}
df <- filter(df, adults+children+babies>0)
```
```

After our pre-analysis, we found that these columns are redundant. So, we removed these columns before feeding the data into our models.

Encode categorical features:

Here we encoded our categorical features in our dataset. In our dataset after cleaning, we got a few categorical features that need to be encoded before feeding those features into the models.

```
**Encode categorical data**
```{r}
df$hotel <- as.numeric(as.factor(df$hotel))  # Convert categories to numbers
df$arrival_date_month <- as.numeric(as.factor(df$arrival_date_month))
df$meal <- as.numeric(as.factor(df$meal))
df$market_segment <- as.numeric(as.factor(df$market_segment))
df$distribution_channel <- as.numeric(as.factor(df$distribution_channel))
df$reserved_room_type <- as.numeric(as.factor(df$reserved_room_type))
df$deposit_type <- as.numeric(as.factor(df$deposit_type))
df$customer_type <- as.numeric(as.factor(df$customer_type))
```
```

Standard scale of the numerical features:

Here we scaled our numerical features before we feed them into the models.

```
**Scale the dataset**
```{r}
df$lead_time <- scale(df$lead_time)
df$adr <- scale(df$adr)
```
```

CHAPTER 5

5.1 METHODS

In this chapter, we will discuss the methods and models that we performed on our data. We believe that these methods helped us to gain some significant insights from our data.

Sampling Methods

In our application we utilized the cross-validation and bootstrap sampling methods on our dataset to attain good accuracy on our dataset.

- **Cross Validation:**
  We applied cross-validation on three models in our application. They are
  - Logistic Regression,
  - KNN
  - Decision Tree
- **Bootstrap:**
  - Random forest
  - Adaboost

```
**Cross Validation for Logistic Regression**
```{r}
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
folds = createFolds(train$is_canceled, k = 10)
cv = lapply(folds, function(x) {
  training_fold = train[-x, ]
  test_fold = train[x, ]
  log_classifier = glm(formula=is_canceled ~ ., family=binomial,
data=training_fold)
  prob_pred = predict(log_classifier, test_fold, type='response')
  y_pred = ifelse(prob_pred > 0.5, 1, 0)
  cm=table(y_pred, test_fold$is_canceled)
  accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
  return(accuracy)
})
accuracy = mean(as.numeric(cv))
accuracy
```
```

The above picture describes the application of cross-validation on logistic regression.

5.2 MACHINE LEARNING MODELS

In our Hotel booking cancellation prediction, we used five different models to predict the cancellation factor. We tried different machine learning models to find out the best model for predicting cancellation. Below are five different models we used in our application. They are

- Logistic Regression
- KNN (K-Nearest Neighbors)
- Decision Tree
- Random Forest
- AdaBoost

Logistic Regression

```
**Logistic Regression**
```{r}
log_classifier = glm(formula=is_canceled ~ ., family=binomial, data=train)
summary(log_classifier)

prob_pred = predict(log_classifier, test, type='response')
y_pred = ifelse(prob_pred > 0.5, 1, 0)

# Making the Confusion Matrix
cat("Prediction vs Actual table for Logistic Regression below -")
cm=table(y_pred, test$is_canceled)
cm
cat("Test error rate for Logistic Regression -", mean(y_pred != 
test$is_canceled))
```
```

The application of logistic regression on our dataset yielded an accuracy of **79.4%.**

KNN (K-Nearest Neighbors):

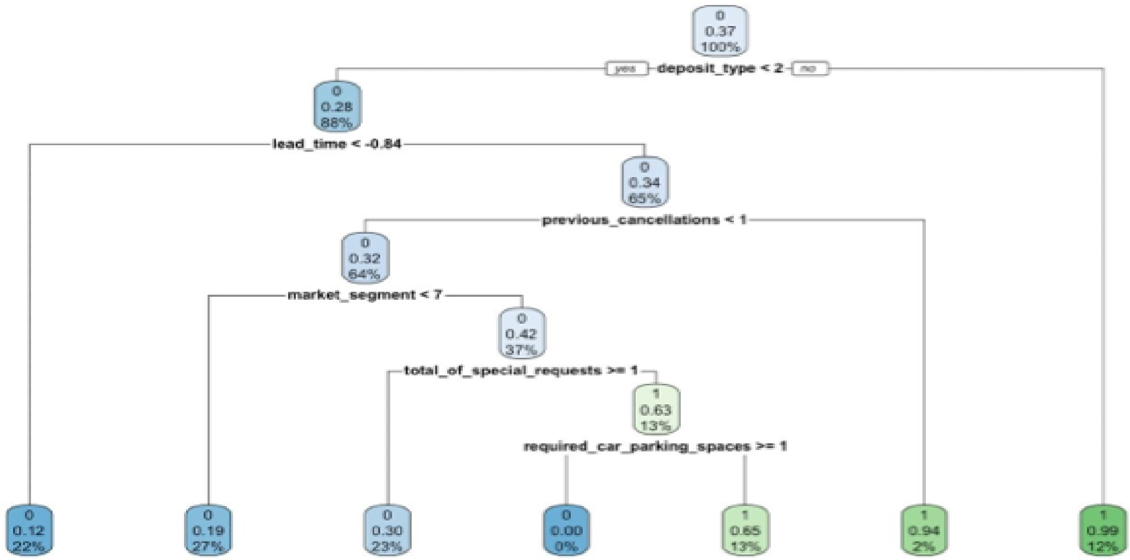
```
y_pred = knn(train=subset(train, select = -c(is_canceled)),
             test=subset(test, select = -c(is_canceled)),
             cl=train$is_canceled,
             k = 5,
             prob = TRUE)
cm <- table(test$is_canceled, y_pred)
cat("Prediction vs Actual table for Test KNN below -", "\n")
cm
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for KNN -", accuracy, "\n")
cat("Test error rate for KNN -", mean(y_pred != test$is_canceled))
```

The application of KNN on our dataset yielded an accuracy of **79.7%.**

Decision Tree:

```
tree.pred.test <- predict(tree.fit, test, type='class')
cat("Confusion Matrix for trees - \n")
cm <- table(tree.pred.test, y_test)
cat("Test error for trees -", mean(tree.pred.test != y_test))
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for Decision Tree -", accuracy, "\n")
rpart.plot(tree.fit)
```

The application of decision-tree on our dataset yielded an accuracy of **80.6%**.



Random Forest :

```
**Random Forest**
```{r}
oj.randomForest <- randomForest(is_canceled ~ ., data=train, importance=TRUE,
ntree = 500)
oj.randomForest.pred.test <- predict(oj.randomForest, test, type = "class")
cat("Confusion Matrix for randomForest - \n")
table(oj.randomForest.pred.test, test$is_canceled)
```
```

The application of random forest on our dataset yielded an accuracy of **85.2%**

Adaboost:

```
adaboost.pred <- predict(adaboost, test, type='response') %>% round()
cm <- table(adaboost.pred, test$is_canceled)
cat("Confusion Matrix for AdaBoost - \n")
print(cm)
accuracy = (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
cat("Test Accuracy for AdaBoost -", accuracy, "\n")
```

The application of AdaBoost on our dataset yielded an accuracy of **81.1%**

	var <chr>	rel.i <dt
deposit_type	deposit_type	53.9776556
lead_time	lead_time	11.9996973
total_of_special_requests	total_of_special_requests	8.2236348
market_segment	market_segment	6.8892176
previous_cancellations	previous_cancellations	6.8076200
required_car_parking_spaces	required_car_parking_spaces	5.5535621
customer_type	customer_type	2.3249470
adr	adr	1.6872330
previous_bookings_not_canceled	previous_bookings_not_canceled	1.3319520
nights_stayed	nights_stayed	0.4867930
arrival_date_month	arrival_date_month	0.2354475
arrival_date_week_number	arrival_date_week_number	0.1288771
meal	meal	0.1221856
guests_stayed	guests_stayed	0.0931385
reserved_room_type	reserved_room_type	0.0651408
days_in_waiting_list	days_in_waiting_list	0.0645454
distribution_channel	distribution_channel	0.0083517
hotel	hotel	0.0000000
is_repeated_guest	is_repeated_guest	0.0000000

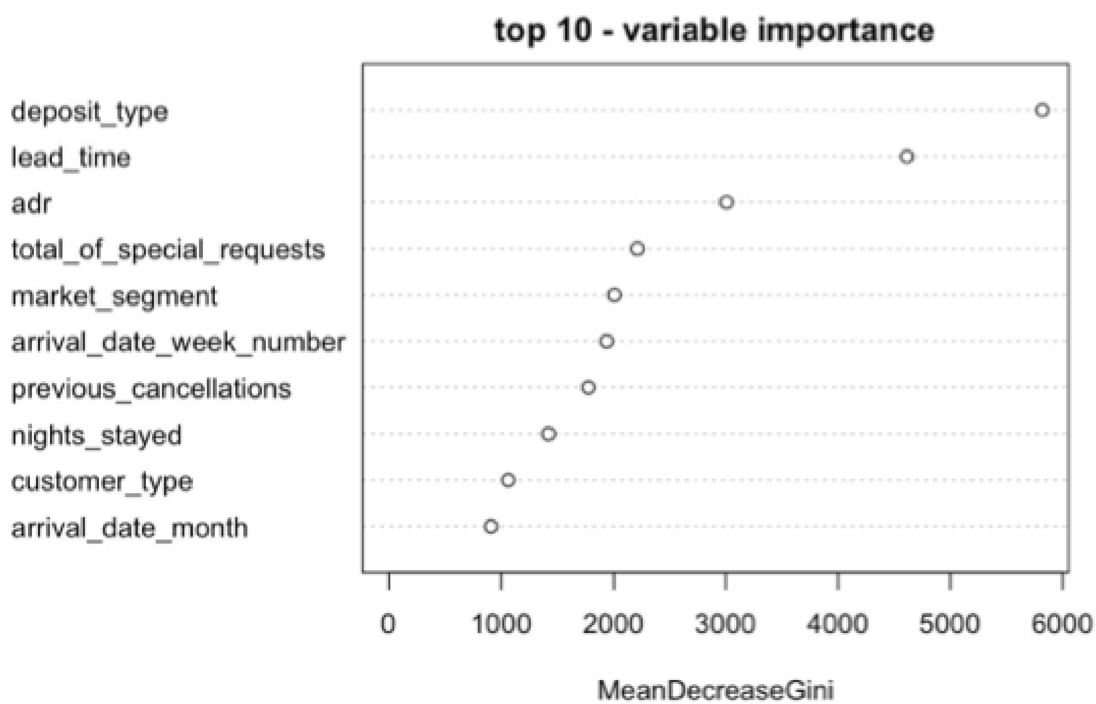
19 rows

CHAPTER 6

6.1 FEATURE SELECTION:

In this chapter, we tried to model our algorithms with the top features we got from the previous implementation. We checked the top features from the algorithms. Below are the top features shown by the algorithms we got implemented previously.

Random Forest:



The features on the Y-axis are the top features that our Random Forest algorithm predicted.

6.2 FEATURES CONSIDERED

After checking the feature importance of various algorithms we implemented we get to learn that these are the top features we considered for model implementation

- Deposit Type
- Total Special Requests
- Lead Time
- ADR
- Market Segment

CHAPTER 7

7.1 RESULTS AND DISCUSSIONS

In this chapter, we will discuss the results that we acquired from our implementation. We implemented models with feature selection and without feature selection. The below tables show the results we got during the implementation of these algorithms.

Classifications Models	Without Feature Selection	With Feature Selection
Logistic Regression	Train: 79.4% Test: 79.4% (CV)	Train: 76.7% Test: 76.8% (CV)
KNN	Train: 80.3% Test: 79.7% (CV)	Train: 79.9% Test: 79.8% (CV)
Decision Tree	Train: 80.6% Test: 80.6% (CV)	Train: 79.4% Test: 79.1% (CV)
Random Forest	Train: 90.5% Test: 85.2%	Train: 82.3% Test: 82.2%
AdaBoost	Train: 81.3% Test: 81.1%	Train: 80.3% Test: 79.6%

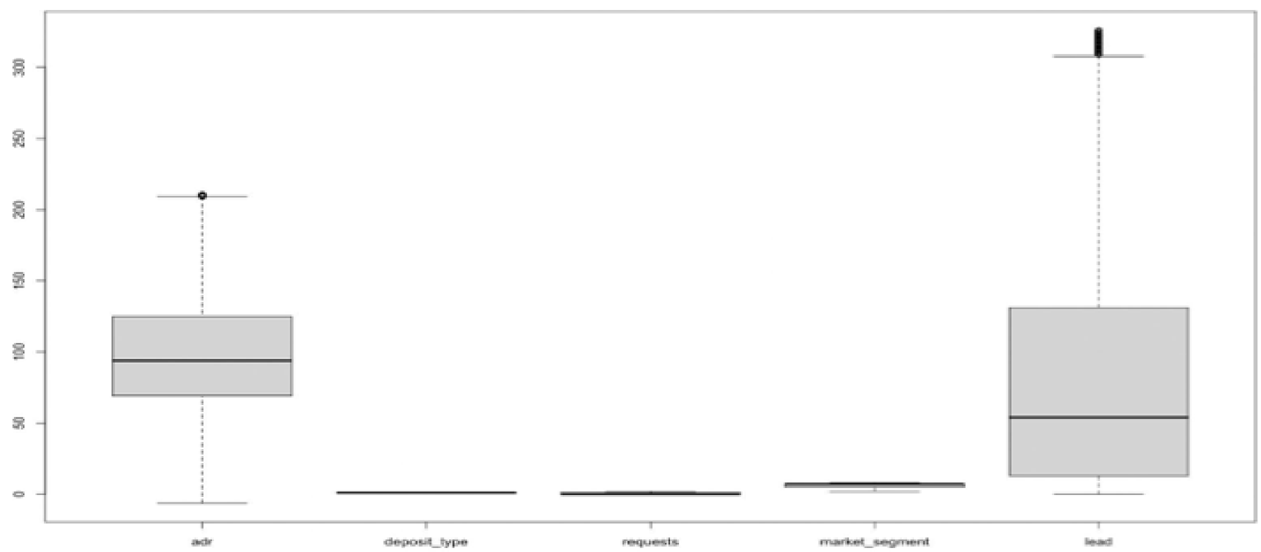
- The table contains the details of accuracy by different algorithms we implemented and their accuracy.
- We can observe no or less difference between without and with feature selection.
- So there is some scope to generalize the model, which means we can go ahead **with feature selection**.

Here, we can see that out of all models we implemented on our dataset the **Random Forest** algorithm tends to perform better than the rest of our algorithms.

So, we chose to predict the Hotel booking cancellation through the Random Forest algorithm.

7.2 Improvisation(Remove Outliers)

We believe that there is a chance for improvisation on our model for better prediction accuracy. So, we tried to check the outliers and other problems with our data. We learn that there are few outliers on our selected features. The below box plot shows the outliers on our selected features.



From this, we can understand there are no outliers for deposit_type, requests, or market segment and there are few to no outliers on ADR and there are few outliers on lead_time.

7.3 FINAL MODEL DECISION:

RANDOM FOREST WITH FEATURE SELECTION:

We implemented our Random forest model with outliers and without outliers for checking the behaviour of prediction and accuracy. The below table shows the behaviour of our model with explained instances.

Model	Without Outliers	With Outliers
Random Forest	Train: 80% Test: 80.08%	Train: 82.3% Test: 82.2%

- There is little or no change in accuracy with and without outliers.
- Unsure of removing outliers due to their influence on the model
- Unable to determine if the outliers are actual outliers even if they fall out of the IQR

CHAPTER 8

8.1 CONCLUSION

We tried implementing various machine learning models for our application. We learnt that the Random forest model performed well from other models we implemented. Reduction of features maintains the accuracy of the model wrt all features and helps to generalize the model for the new datasets. We are unsure of removing outliers due to their influence on the model. Unable to determine if the outliers are actual outliers even if they fall out of the IQR From the best model, the features deposit_type, lead_time, ADR, special_requests and market_segment are the most influential features.

CHAPTER 9

9.1 GITHUB REPOSITORY

We pushed our project into the Git repository.
https://github.com/jayanthjay12/EAS506_StatisticalDataMining1_Team5

9.2 CONTRIBUTIONS

Bharatwaj Majji -

- Analysed all the Pairwise Associations on the dataset
- Applied Logistic Regression on the cleansed data.
- Applied Random Forest to check performances.
- Contributed to PPT
- Contributed to Documentation

Jayanth Puthineedi -

- Understood dataset and performed Data Visualisation with some analysis.
- Applied KNN and Decision Trees algorithms for the classification.
- Evaluated models by applying Cross Validation for Logistic Regression, KNN and Decision trees models.
- Project setup in GIT
- Contributed to the Documentation

Vishnu Bhadramraju -

- Performed data Pre Processing on the dataset.
- Applied Adaboost classifier on the dataset.
- Removed outliers to evaluate the model performances.
- Contributed to PPT.
- Contributed to Documentation.

9.3 REFERENCE

- Dataset: https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand?datasetId=511638
- Course Documents