Business Statistics Project Report

On

"Developing a Classification Model Using Random Forest to find the Passenger Status in a Spaceship Accident Scenario"

Submitted in partial fulfillment for award of

Master's in Business Administration

Degree

In

General Management

By

Team 09

Shreya K	22BM63066
Deepak Singh	22BM63042
Sarath R	22BM63106

Under the guidance of

Professor Sujoy Bhattacharya

Vinod Gupta School of Management

Indian Institute of Technology, Kharagpur

Kharagpur, India

November, 2022

Table of Contents

1. Introduction

In order to understand the complexities of the world, humans have always resorted to field of statistics in order to formulate the real life events as a workable model. Starting from the ages of manual calculations, the field has advanced into a much advanced independent dimension, commonly referred as Data Analytics. This has presented an opportunity to utilize and implement the learnings in the field of Business, where Executives deal with huge amount of data in daily basis in order to arrive at a meaningful decision. The purpose of a Business Analytics is to assist the decision making by backing it up with a quantifiable data and providing a viable solution to the problems faced by the target entity. It utilizes various data analytics tools in order to predict and analyze the available data.

One such tool is Random Forest. Random Forest is a supervised learning algorithm that uses ensemble methods to solve both regression and classification problems. It operates by constructing many decision trees at training time and outputs the mean/mode of prediction of individual trees. We have used this algorithm for our model construction.

2. Problem Statement

The problem statement pertains to a fictional scenario in year 2912 where a spaceship TITANIC, transporting passengers from 3 different planets to 3 different destinations meets a space anomaly, resulting in the teleporting of several passengers into alternative dimension. The objective is to assist the rescue team by analyzing the database of damaged ship and identifying whether passenger has been teleported or not. We have been provided with different attributes associated with passengers in order to develop a training model and implement the algorithm in test dataset, returning the results in form of Passenger ID along with the status whether he/she was teleported or not.

3. Methodology

The given problem statement pertains to a case of **Classification Model** with requirement of predicting the status of passengers, that is, essentially classifying if the passenger has been teleported or not by utilising the given dataset.

- **Decision Making Tool: Random Forest**

Why Random Forest over Decision Tree?

Random forest adds additional randomness to the model while growing trees. When splitting a node, it searches for the best feature among a random subset of features instead of looking for the most important feature. Thus, it reduces the overfitting problem in decision trees and lessens the variance, improving accuracy.

Target Variable

TRUE : Transported

FALSE : Not Transported

Since our **target** variable is **categorical** and our **independent** dataset is **discrete and categorical**, primarily, the problem pertains to a ***Classification Model*** that can be easily formulated as well as solved using Random Forest.

The final ***evaluation*** of our decision-making model is based on the ***Classification Accuracy***, calculated through:

Classification Accuracy:

$$\frac{Accurate\ Predictions}{Total\ Number\ of\ Outcomes/Predictions}$$

- **Approach**

Our approach can be segregated under two stages:

I. Formulating problem

II. Developing Model and Analysis

I. Formulation

Formulation involved defining our variables as independent and target variables and segregating them further on the basis of their utility in our Test Model. ***Figure 3.1*** depicts various input variables that were provided with the problem statement along with the target variable that was to be determined.

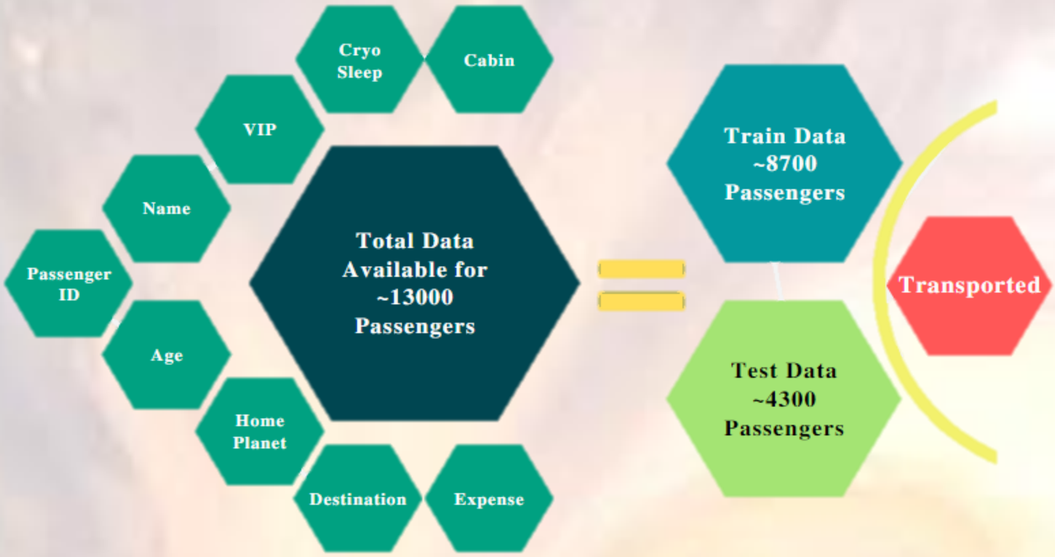

Fig. 3.1 Model Development - Outline

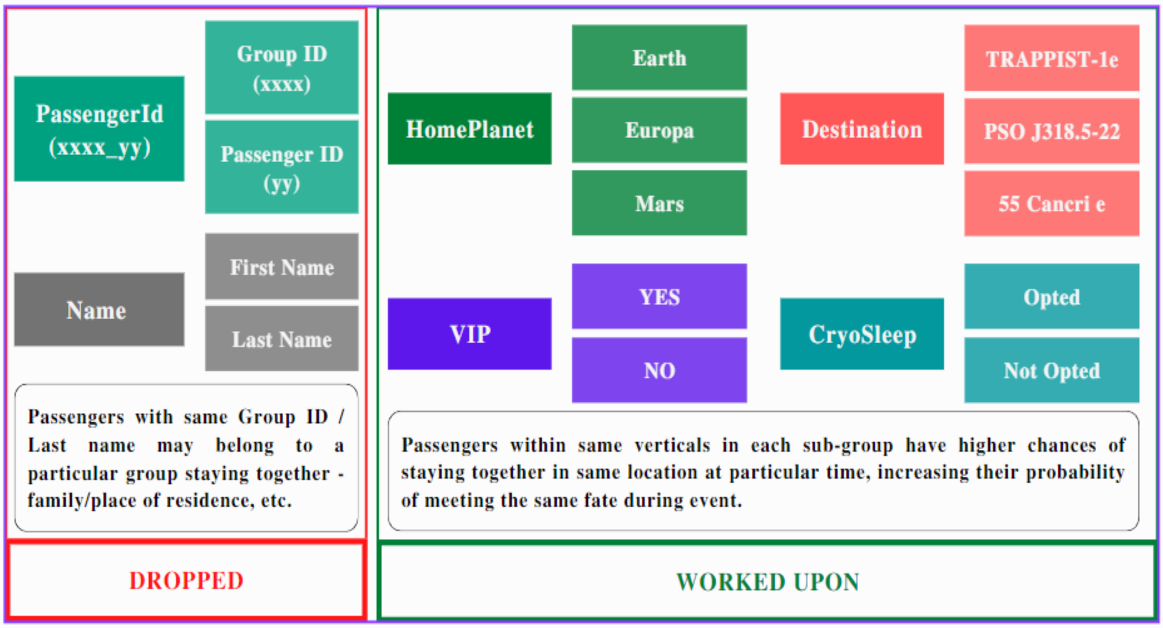

Fig. 3.2.1 Dataset Available

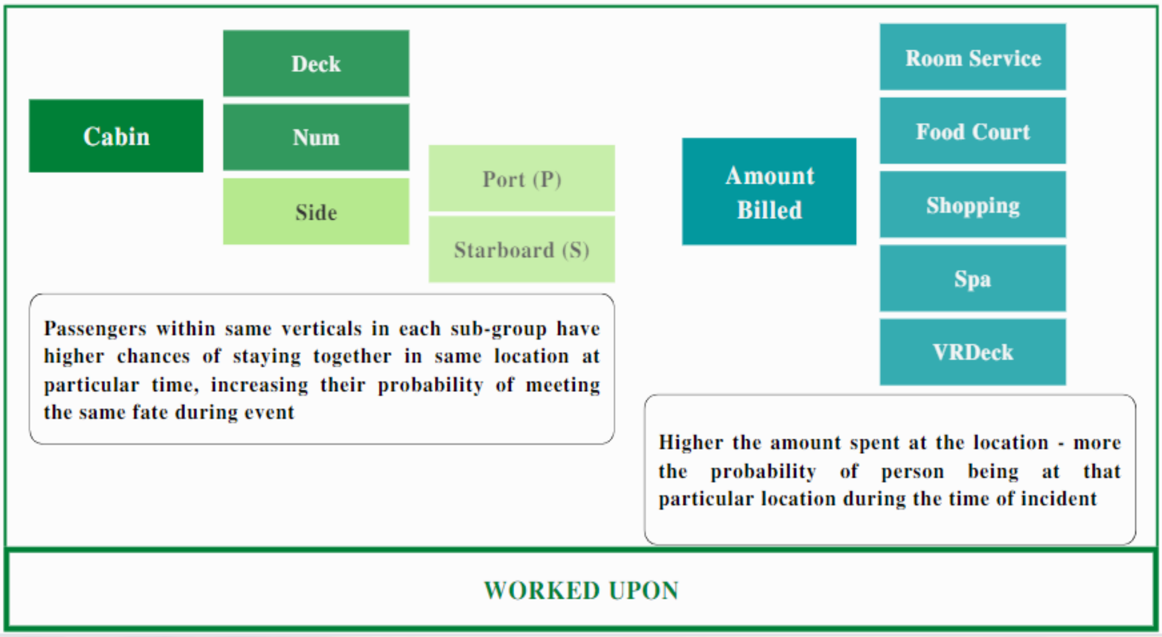

Fig. 3.2.2 Dataset Available

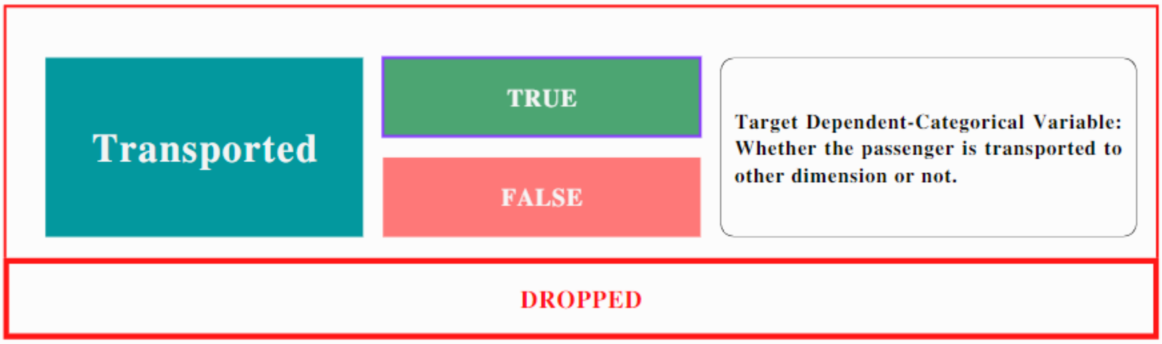

Fig. 3.3 Target Variable

- **Dataset Explained:**

  The dataset retrieved from damaged ship was primarily segregated into following groups:

- **Passenger ID :** **xxxx_yy**

  Data Type : **Numerical**

  Significance :

  Passenger ID depicted the Unique ID assigned to the passengers for the journey. Entire database was with reference to this ID and thus final outcome was also assigned to it. Also, the ID comprised of both Group ID (xxxx) that was assigned to a particular group travelling together along with unique number associated with the passenger (yy).

Since it was unique to the passenger, the dataset could not have contributed in training our model and hence was dropped from our test dataset.

- **Name**

  Data Type : **Categorical**

  Significance :

  Name comprises of First and Last name of an individual. The last name might reflect the person belonging to a particular group, example a family, same residential location, etc.

  Considering each name unique to an individual, the dataset has been dropped out of our test model.

- **Age**

  Data Type : **Numerical**

  Significance :

  Person belonging to different age groups have different capacities to respond to a given emergency situations. Hence the data has been retained in our working model.

- **VIP**

  Data Type : **Categorical**

  Significance :

  Since all the VIP persons might be placed together by the crew, hence they can be grouped together. The data has been retained.

- **Home Planet**

  Data Type : **Categorical**

  Significance :

  The dataset classifies passengers among three categories on the basis of their origin planet: Earth, Europa, and Mars. As persons belonging to same locations often are clustered together, their chances of meeting the same fate are higher during any adverse event. Hence, data has been retained in our model.

- **Destination Planet**

  Data Type : **Categorical**

  Significance :

  The dataset classifies passengers among three categories on the basis of their destinations. Persons can be grouped on the basis of their destinations, which raises their chances of meeting the same fate during any adverse event. Hence, data has been retained in our model.

- **Cryo-sleep**

  Data Type : **Categorical**

  Significance :

  Data classifies passengers on the basis of their choice to opt for Cryo-sleep. All passengers opting for cryo-sleep have high probability of being placed together, and hence ending up in similar situation as others in same group. Thus, the data has been retained in our model.

- **Cabin**

  Data Type : **Categorical**

  Significance :

  Passengers have been segregated on the basis of their cabin's position in the ship – as Port side or Starboard Side, along with the number of the cabin occupied. Persons on the same side of the ship and closer to the assigned number have higher chances of meeting the same fate. Thus, the data has been retained in our model.

- **Expenses**

  Data Type : **Categorical**

  Significance :

  Expenses at a particular location can be used to determine the probable location of the passenger at the time of the event, under the assumption that more the amount a passenger spends at a particular location, higher the amount of time spent by her at that location.

- **Transported**

  Data Type : **Categorical**

  Significance :

This is our Target Variable, and it signifies whether the passenger was teleported or not. Since this was our target variable, it has been segregated to the variable 'Y' signifying that it is our target variable.

II. Model Development

Various **stages** involved in the process were:

a. Platform Selection : *Google Collab*

Selecting Library : *pandas, sklearn, matplotlib, seaborn*

Import Libraries

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import drive
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
```

Fig. 3.4 Importing Libraries

b. **Importing** source dataset: *Train dataset + Test Dataset*

Mount Drive

```python
[2] drive.mount("/content/gdrive")
```

```
Mounted at /content/gdrive
```

Import Dataset

```python
[3] dataset = pd.read_csv('/content/gdrive/My Drive/You & Me/Sem 1/BA/BA Project - Spaceship Titanic/train.csv')
    testDataset = pd.read_csv('/content/gdrive/My Drive/You & Me/Sem 1/BA/BA Project - Spaceship Titanic/test.csv')
```

Fig. 3.5 Importing Dataset from Google Drive

c. **Analyse** the given datasets for developing a descriptive statistic (to be used in assigning values to vacant columns) and determining the unfilled rows and columns.

- Info() method gives us the number of non-null entries and the data type of each feature.
- Describe() method gives us the descriptive statistics of the dataset.

Analyze Data

```
[ ]  dataset.info()
     testDataset.info()
     dataset.describe()
     testDataset.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8693 entries, 0 to 8692
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   PassengerId   8693 non-null   object
 1   HomePlanet    8492 non-null   object
 2   CryoSleep     8476 non-null   object
 3   Cabin         8494 non-null   object
 4   Destination   8511 non-null   object
 5   Age           8514 non-null   float64
 6   VIP           8490 non-null   object
 7   RoomService   8512 non-null   float64
 8   FoodCourt     8510 non-null   float64
 9   ShoppingMall  8485 non-null   float64
 10  Spa           8510 non-null   float64
 11  VRDeck        8505 non-null   float64
 12  Name          8493 non-null   object
 13  Transported   8693 non-null   bool
dtypes: bool(1), float64(6), object(7)
memory usage: 891.5+ KB
```

Fig. 3.6 Data Type and Non-null Entries in Training Dataset

```
[ ]  <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 4277 entries, 0 to 4276
     Data columns (total 13 columns):
      #   Column        Non-Null Count  Dtype
     ---  ------        --------------  -----
      0   PassengerId   4277 non-null   object
      1   HomePlanet    4190 non-null   object
      2   CryoSleep     4184 non-null   object
      3   Cabin         4177 non-null   object
      4   Destination   4185 non-null   object
      5   Age           4186 non-null   float64
      6   VIP           4184 non-null   object
      7   RoomService   4195 non-null   float64
      8   FoodCourt     4171 non-null   float64
      9   ShoppingMall  4179 non-null   float64
      10  Spa           4176 non-null   float64
      11  VRDeck        4197 non-null   float64
      12  Name          4183 non-null   object
     dtypes: float64(6), object(7)
     memory usage: 434.5+ KB
```

Fig. 3.7 Data Type and Non-null Entries in Testing Dataset

	Age	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck
count	4186.000000	4195.000000	4171.000000	4179.000000	4176.000000	4197.000000
mean	28.658146	219.266269	439.484296	177.295525	303.052443	310.710031
std	14.179072	607.011289	1527.663045	560.821123	1117.186015	1246.994742
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	19.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	26.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	37.000000	53.000000	78.000000	33.000000	50.000000	36.000000
max	79.000000	11567.000000	25273.000000	8292.000000	19844.000000	22272.000000

Fig. 3.8 Descriptive Statistics of Training Dataset

d. Understanding the correlation between different features

In order to understand how closely the features are related to each other and to the target variable, we are computing the correlation among them. Then we are plotting the correlation on a heatmap to get a visualization of the same. We can take decision regarding which features to keep or drop in the subsequent stages of model development.

Analyzing Correlation Between Features via Heatmap

```
[5] correlation = dataset.corr()
    correlation
```

	Age	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Transported
Age	1.000000	0.068723	0.130421	0.033133	0.123970	0.101007	-0.075026
RoomService	0.068723	1.000000	-0.015889	0.054480	0.010080	-0.019581	-0.244611
FoodCourt	0.130421	-0.015889	1.000000	-0.014228	0.221891	0.227995	0.046566
ShoppingMall	0.033133	0.054480	-0.014228	1.000000	0.013879	-0.007322	0.010141
Spa	0.123970	0.010080	0.221891	0.013879	1.000000	0.153821	-0.221131
VRDeck	0.101007	-0.019581	0.227995	-0.007322	0.153821	1.000000	-0.207075
Transported	-0.075026	-0.244611	0.046566	0.010141	-0.221131	-0.207075	1.000000

Fig. 3.9 Correlation Map Between Features

```
sns.heatmap(correlation, annot = True, fmt=".2f")
```

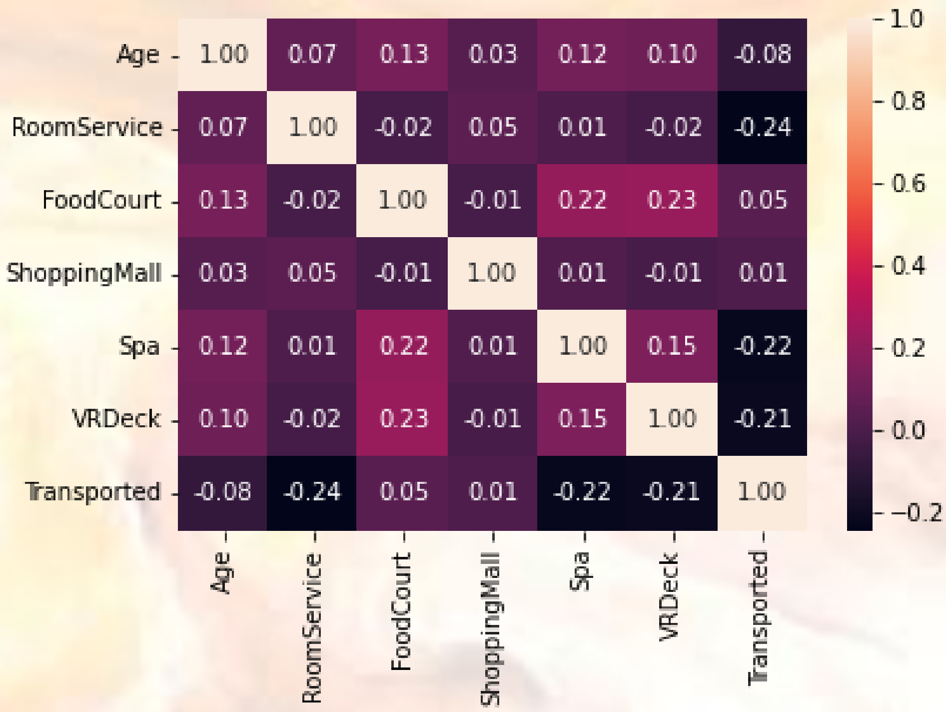

Fig. 3.10 Heatmap Map Between Features highlighting Correlation

e. Modifying the dataset

- *Dropping* Passenger ID and Name (unique to passengers – no pattern developed)

Modify Train and Test Datasets

1. Drop PassengerId and Name from Train and Test Dataset

```
[7] dataset.drop(['PassengerId'], axis = 1, inplace = True)
    dataset.drop(['Name'], axis = 1, inplace = True)

    testDataset.drop(['PassengerId'], axis = 1, inplace = True)
    testDataset.drop(['Name'], axis = 1, inplace = True)
```

Fig. 3.11 Dropping Passenger Id and Name from Training and Test Datasets

- *Splitting* Cabin data into respective datasets. We are doing this because, cabin is a combination of Deck, Number and Side each of which represent a position in the spaceship thereby having a contributing factor in determining if the passenger will be transported or not.

2. Split Cabin into Deck, Num and Side

```
[8] dataset[['Deck', 'Num', 'Side']] = dataset['Cabin'].str.split('/', expand = True)
    dataset.drop(['Cabin'], axis = 1, inplace = True)

    testDataset[['Deck', 'Num', 'Side']] = testDataset['Cabin'].str.split('/', expand = True)
    testDataset.drop(['Cabin'], axis = 1, inplace = True)
```

Fig. 3.12 Splitting Cabin into Further Features

- For missing *numerical data*, we performed *Mean Imputation* on the dataset.

3. Perform Mean Imputation for Missing Numerical Features

```
[9] numericalColumns = ['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa', 'VRDeck']
    for col in numericalColumns:
        dataset[col].fillna((dataset[col].mean()), inplace=True)
        testDataset[col].fillna((testDataset[col].mean()), inplace=True)
```

Fig. 3.13 Mean Imputation

- For missing *categorical* values, we performed the *Mode Imputation* on the dataset

4. Perform Mode Imputation for Missing Categorical Features

```
[10] categoricalColumns = ['HomePlanet', 'CryoSleep', 'Destination', 'VIP', 'Deck', 'Num', 'Side']
     for col in categoricalColumns:
         dataset[col].fillna((dataset[col].mode()[0]), inplace=True)
         testDataset[col].fillna((testDataset[col].mode()[0]), inplace=True)
```

Fig. 3.14 Mode Imputation

e. Further the categorical dataset was converted into numerical dataset in order to develop a relation among our independent variable.
   - Affirmative/Negative Responses converted into binary responses.
   - Home and Destination planets were assigned numerical values.

5. Convert Categorical Data into Numerical Data

```
labelencoder = LabelEncoder()
for col in categoricalColumns:
    dataset[col] = labelencoder.fit_transform(dataset[col])
    testDataset[col] =labelencoder.fit_transform(testDataset[col])
```

Fig. 3.15 Conversion of Categorical to Numerical Data

```
[12] dataset.head()
```

	HomePlanet	CryoSleep	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Transported	Deck	Num	Side
0	1	0	2	39.0	0	0.0	0.0	0.0	0.0	0.0	False	1	0	0
1	0	0	2	24.0	0	109.0	9.0	25.0	549.0	44.0	True	5	0	1
2	1	0	2	58.0	1	43.0	3576.0	0.0	6715.0	49.0	False	0	0	1
3	1	0	2	33.0	0	0.0	1283.0	371.0	3329.0	193.0	False	0	0	1
4	0	0	2	16.0	0	303.0	70.0	151.0	565.0	2.0	True	5	1	1

Fig. 3.16 Converted Training Dataset

```
testDataset.head()
```

	HomePlanet	CryoSleep	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Deck	Num	Side
0	0	1	2	27.0	0	0.0	0.0	0.0	0.0	0.0	6	820	1
1	0	0	2	19.0	0	0.0	9.0	0.0	2823.0	0.0	5	927	1
2	1	1	0	31.0	0	0.0	0.0	0.0	0.0	0.0	2	0	1
3	1	0	2	38.0	0	0.0	6652.0	0.0	181.0	585.0	2	1	1
4	0	0	2	20.0	0	10.0	0.0	635.0	0.0	0.0	5	1029	1

Fig. 3.17 Converted Testing Dataset

f. Further dataset was further split into *independent features* and the *target variables*:

Split Data into Dependent Features and Target Variable

```
[14] Y = dataset['Transported']
     dataset.drop(['Transported'], axis = 1, inplace = True)
     X = dataset
```

```
[15] X
```

	HomePlanet	CryoSleep	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Deck	Num	Side
0	1	0	2	39.0	0	0.0	0.0	0.0	0.0	0.0	1	0	0
1	0	0	2	24.0	0	109.0	9.0	25.0	549.0	44.0	5	0	1
2	1	0	2	58.0	1	43.0	3576.0	0.0	6715.0	49.0	0	0	1
3	1	0	2	33.0	0	0.0	1283.0	371.0	3329.0	193.0	0	0	1
4	0	0	2	16.0	0	303.0	70.0	151.0	565.0	2.0	5	1	1

Fig. 3.18 Independent Features Represented by 'X'

```
Y

0        False
1         True
2        False
3        False
4         True
```

Fig. 3.19 Dependent Feature Represented by 'Y'

g. Finally *running* the *Random Forest model*

Implement Random Forest Model

```
[19] classifier = RandomForestClassifier()
     classifier.fit(X, Y)
     Y_pred = classifier.predict(testDataset)
     result = round(classifier.score(X, Y) * 100, 2)
     result

     99.94
```

Fig. 3.20 Random Forest Implementation

4. Results and Discussion

Our model returned a Classification Accuracy of ***99.94 %***. Therefore, the accuracy of the model is very good. With the use of Random Forest Algorithm over Decision Trees, the potential of over-fitting has decreased significantly though it may occur due to the limited amount of input data available with us. This can result in conveying false information for some passengers, and needs to be corrected by including some more information in the dataset.

5. Conclusion

The undertaken study helps us in developing a basic understanding about Business Analytics as a tool for decision-making. It further exposes the limitations of the Random Forest as a tool with limited dataset. The undertaken study can be used in other decision-making scenarios as well, provided there is sufficient data available for model to run successfully, without *over-fitting* or *under-fitting* our model.

The field of Analytics is equipped with several other tools to deal with similar decision-making dilemmas, which can be used along with Random Forest in order to arrive at a better conclusion.

6. Bibliography

- *Spaceship Titanic: Kaggle link*
- *Coding and model development : Google Collab link*
- *Data Set:*

train.csv test.csv

7. Appendix: Contribution Table

Task / Roll No.	22BM63042	22BM63066	22BM63106
Problem Identification	✓	✓	✓
Methodology Development	✓	✓	✓
Coding	✓	✓	✓
Analysis	✓	✓	✓
Report Compilation	✓	✓	✓
Contribution (%)	**33**	**34**	**33**