STAT 4241 Statistical Machine Learning
Final Project
Spaceship Titanic Model

Name: Judy Wu
UNI: dw2936

Table of Contents

1. Introduction *3*

2. Data Description and Pre-processing *3*

2.1 Data Description **3**

2.2 Data Pre-processing **4**

2.2.1 Composite Feature 4

2.2.2 Missing Values 4

2.2.3 Skewness 5

2.2.5 Categorical Features Encoding and Standardization 5

2.2.6 Split Dataset 5

3. Statistical Models *5*

3.1 Linear Discriminant Analysis **5**

3.3 Naïve Bayes **6**

3.4 KNN **6**

3.5 Random Forest **7**

3.6 Gradient Boosting **7**

4. Results *8*

5. Limitation and Potential Improvement *9*

5.1 Imbalanced Data **9**

5.2 Hyperparameter Tuning **9**

5.3 Name **9**

Appendix A: Important Outcomes of Exploratory Data Analysis *10*

A.1 Overview of Dataset **10**

A.2 Imbalanced Data **10**

A.2.1 HomePlanet 10

A.2.2 Destination 10

A.2.3 VIP 11

A.3 Skewness **11**

A.4 Correlation **13**

1. Introduction

Inspired by the Titanic, it has always been a hot topic to discover the survival patterns of passengers on wrecked ships through their information. This dataset is based on a fictional background, recording the interstellar passengers' information on a spaceship that traveling to three newly habitable exoplanets. This spaceship had an accident and half of the passengers on board were transported to an alternate dimension. This project aims at constructing binary classification models to predict whether passengers on the spaceship were transported to an alternate dimension, which would be beneficial for retrieving and rescuing the lost passengers.

The following report uses Python as coding language and explores the recovered data from the damaged spaceship computer with missing values and constructs binary classification models using algorithms such as linear discriminant analysis, K-Nearest-Neighborhoods, Naïve Bayes, Logistic Regression, Random Forest, and Gradient Boosting and gets the highest accuracy of around 80%.

2. Data Description and Pre-processing

2.1 Data Description

The dataset is the training set of the Kaggle competition Spaceship Titanic. It consists of 8639 passengers' personal information with 13 features and one column of target "Transported" that we want to predict (Table 1). We split the dataset and use 70% as the training set and 30% as the test set.

The dataset includes both numeric features and categorical features and has 1.9% of cells missing. In addition to missing values, there are also problems of multicollinearity and imbalanced data, and skewness in the dataset. The exploratory data analysis (Appendix A) shows that multicollinearity exists between the numeric features which may cause problems when constructing models (Figure 1). For some categorical features like HomePlanet, the distribution among classes is quite imbalanced. Moreover, the features indicating passengers' cost on luxury facilities are highly skewed. We will fix these problems in the pre-processing procedure.

Feature	Type	Meaning
PassengerId	Numeric	A unique ID for each passenger
HomePlanet	Categorical	The planet the passengers departed from, contains three different classes, Earth, Mars, and Europa.
CryoSleep	Categorical	Whether the passenger was elected to be in cryosleep or not, containing True and False.
Cabin	Categorical	The cabin number where passengers staying in. It consists of the deck, number, and side information.
Destination	Categorical	The destination where passengers going to. It contains three different classes.
Age	Numeric	The age of passengers.
VIP	Categorical	Whether the passenger is VIP or not. The value is True or False.
RoomService, FoodCourt, ShoppingMall, Spa, VRDeck	Numeric	These features indicate the cost of passengers on the luxury facilities.
Name	Categorical	The name of each passenger.
Transported	Categorical	The target feature indicates whether the passenger was transported or not

Table 1 The Data Field Description

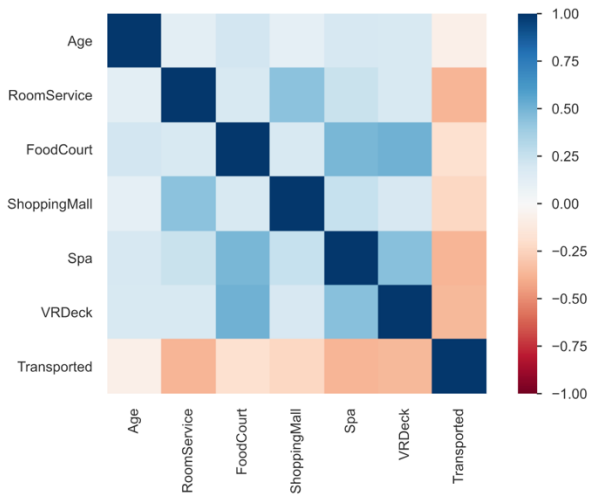

Figure 1 Correlation heatmap of numeric features

2.2 Data Pre-processing

2.2.1 Composite Feature

Since the feature "Cabin" contains three different information including the deck, number of the deck, and the side where the passenger was staying, we split this column into three features, "Deck", "Num" and "Side" for later use and drop the original "Cabin" feature to avoid redundancy.

2.2.2 Missing Values

a) Numeric Features

There are six numeric features in the dataset. Among them, the five features about the cost of luxury facilities are highly skewed, with over 60% zeros (Figure 2). Hence, we fill the missing values of these columns by 0. The other feature "Age" is mainly distributed between 15 and 45 but with some extreme values. Hence, we fill the missing values of age with the median of available data.

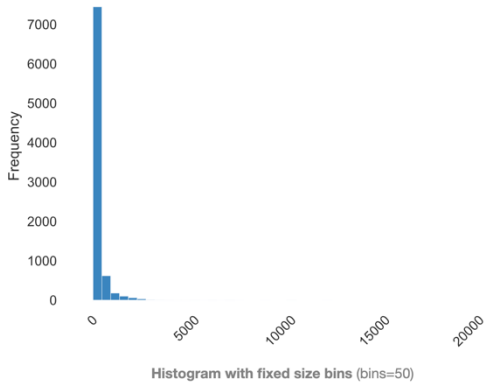

Figure 2 Histogram of Shoppingmall

b) Categorical Features

For the categorical features, we fill most of the missing values by the class that appears most frequently. Moreover, for the feature "Name" and "PassengerId", since it seems that no useful information is provided in the columns, we directly drop the features.

2.2.3 Skewness

Since the skewness exists in some numeric features' distribution, we need to prune the extreme values to reduce the skewness. When the 95th percentile is close to the maximum value, the tail has more interesting information than what we want to discard. Hence, we substitute the extreme values with the 95th percentile when they are larger than ten times the median value, which can reduce the skewness and avoid excessive prune at the same time.

Moreover, since the data of these features distributes in a wild range but mainly concentrates around 0. Hence, the Log function was also applied to compress large numbers to a smaller range. This was applied to all numeric features that have more than 50 unique values to reduce the skewness. This constraint can filter out the numeric type features that exist as categorical features.

2.2.5 Categorical Features Encoding and Standardization

The cost of luxury facilities was summed up and split into 6 classes to indicate the level of consumption. This information is stored in a new categorical feature "Luxury". After this, all categorical features were encoded as dummy variables for later use by pd.get_dummies() function. This method can encode categorical features into interpretable dummy variables. Moreover, Min-Max scaling was also applied to the dataset to scale the wildly-spread data into the same range, which may be helpful for modeling.

2.2.6 Split Dataset

Train_test_split function from sklearn.model_selection was conducted to split the dataset into two parts. 70% of data were used as the training set and the left part was used as the test set to see the models' performance.

3. Statistical Models

Since this is a binary classification problem, we tried different types of classification models such as LDA, Logistic regression, Naïve Bayes, KNN, random forest, and Gradient Boosting algorithms to construct classification models. To select the best parameters for models, we conducted 10-fold repeated stratified cross-validation to tune models. Furthermore, to evaluate the performance of the models, we used metrics accuracy, recall and precision rate, recall rate, ROC AUC score, and F1-Score.

Note that due to the random sampling process, the result shown below may be slightly different from the result in Python script file.

3.1 Linear Discriminant Analysis

We first fit a toy LDA model with default parameter values and use the GridSearchCV function to find the best performance hyperparameters. For cross-validation, we used the repeated stratified 10-fold algorithm with 3 repeats. The result shows that the LDA model using solver "lsqr", which stands for least square, and shrinkage parameter "auto" has the best performance, which has about 77.15% accuracy for the test set and 77.22% accuracy for the training set. The confusion matrix is shown in Figure 3. From the matrix, we can tell that there are relatively more un-transported passengers who are misclassified to transported class, which is around 12.81%. The precision is 77.15% meaning that among all chosen of all chosen positive examples, 77.15% of the data is genuinely positive.

3.2 Logistic Regression

Logistic Regression as a popular binary classification was tried here. To tune the hyperparameter, we also apply grid search cross-validation on logistic regression to find the

best performing hyperparameter and we chose to use the repeated stratified 10-fold algorithm with 3 repeats for cross-validation. The result shows that when choosing L1 penalty with solver “liblinear” and Cs=10 or L2 penalty with solver “lbfgs” and C=1000 has the best performance with training accuracy of 77.43% and test accuracy of 77.72%. The confusion matrix is shown in Figure 4. In the prediction, around 11.39% of data are false-positive and 10.89% are true negative. This model performs similarly to the Logistic Regression model.

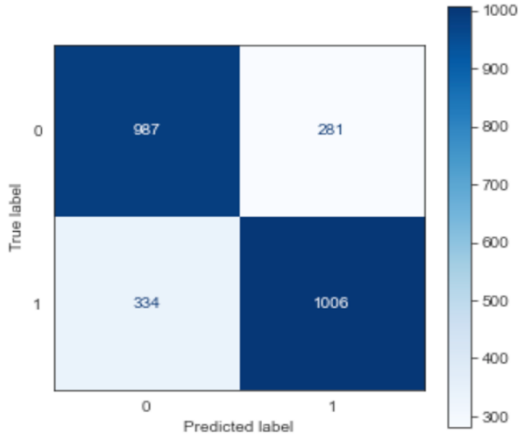

Figure 3 Confusion Matrix of the best LDA model after tuninig

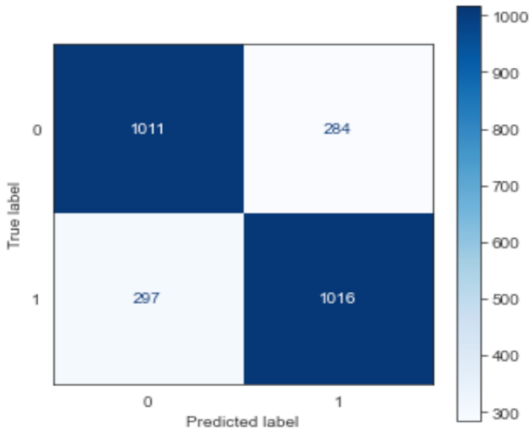

Figure 4 Confusion Matrix of Best Performing Logistic Regression Model

3.3 Naïve Bayes

We also tried the Gaussian Naïve Bayes model on the training set using the default hyperparameters and tuned hyperparameters. However, the model did not perform well, only having a training accuracy of 63.16% and a test accuracy of 63.23%.

3.4 KNN

K Nearest Neighborhoods use the information of K neighborhoods to classify data points. To choose the value of K, a 10-fold cross-validation was conducted. We used the cross_val_score function to get the accuracy score of each model and chose the K having the highest score, which was 14 (Figure 5). Then we constructed14-NN model based on training data and got a model with 75.92% training accuracy and 78.50% test accuracy (Figure 6). In the prediction, more un-transported passengers were misclassified to transported class, which means around 15.57% of data are false-positive and only 7.06% are true negative. Although this model gave compatible accuracy with LDA and logistic regression, it showed different behavior for classification outcomes.

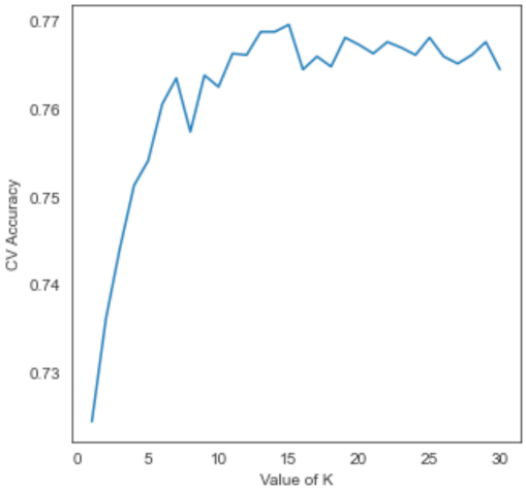

Figure 5 Cross Validation Scores for K from 1 to 30

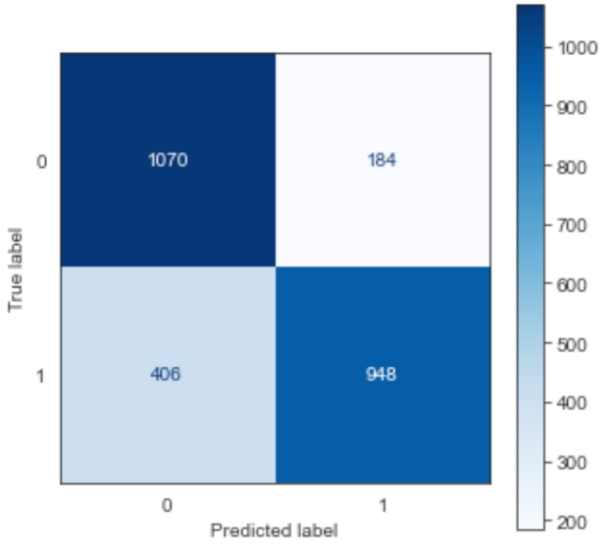

Figure 6 Confusion Matrix of 14-NN model

3.5 Random Forest

For the random forest algorithm, we tried to set the number of decision trees (n-estimators) to be 100, 200,500, and 1000 and bootstrap to be True and False. According to the accuracy each model gave, we set the number of decision trees to be 500 and set bootstrap to be true to get higher accuracy. In this model, we get a training accuracy of 99.95% and a test accuracy of 79.22%. The confusion matrix of this model is shown below (Figure 7). The false-positive rate is 12.5% and the true negative rate is 8%, which is pretty low compared to the models above.

3.6 Gradient Boosting

Gradient Boosting algorithm is an ensemble of weak prediction models decision tree which may be able to give good performance. We tried a gradient boosting algorithm with the number of decision trees (n_estimators) equal to 100, 200, 500, and 1000. The model with n_estimators = 500 performed good training accuracy as well as generalization ability, giving the training accuracy 83.37% and test accuracy 80.67%. The confusion matrix of the model is shown in Figure 8. The false-positive rate and true negative rate are around 10% and relatively lower than the linear models and KNN model.

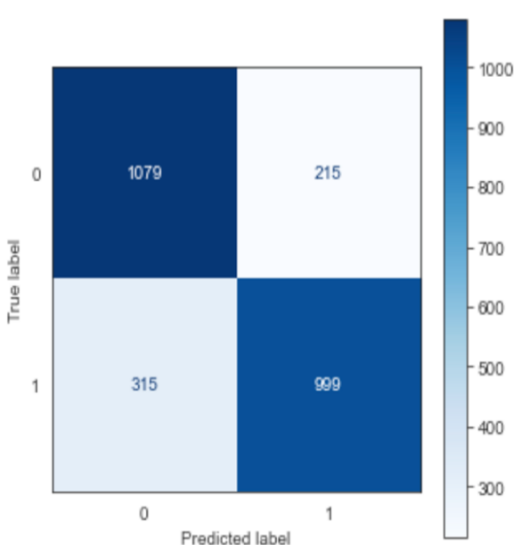

Figure 7 The Confusion Matrix of Random Forest Model

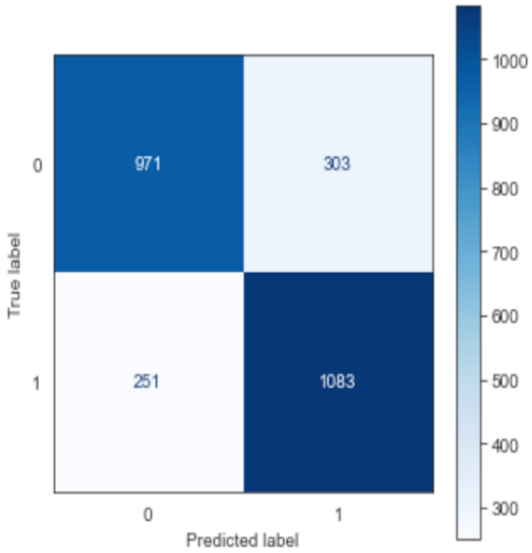

Figure 8 The Confusion Matrix of Gradient Boosting Model

4. Results

The result of the models above is summarized in the table below (Table 2). From the table, we can see that the Gradient Boosting model combining 200 decision trees performed the best among the models we have, giving the test accuracy of 80.67%. Random Forest Model also performed quite well, giving an accuracy of about 79.21%. Other algorithms such as LDA, Logistic regression and KNN give similar accuracy around 77% which is also quite good. Looking at other metrics like ROC AUC, the linear models like the linear discriminant analysis model and logistic model perform quite well. Moreover, the random forest model, as well as the gradient boosting model, give the ROC AUC around 0.80 which is quite good.

In conclusion, we can choose the Gradient Boosting model or random forest model to predict whether passengers were transported to another dimension or not with higher accuracy. If we need a higher ROC AUC score, we can also choose linear models like LDA or the Logistic regression model.

	Accuracy	Recall	Precision	F1-Score	MCC score	ROC AUC
LDA	0.764187	0.764187	0.764888	0.764222	0.528981	0.764569
LDA_Tuned	0.771472	0.771472	0.771517	0.771450	0.542958	0.848032
Logistic_l1	0.777224	0.777224	0.777267	0.777226	0.554485	0.849566
Logistic_l2	0.774923	0.774923	0.774933	0.774925	0.549841	0.849525
KNN	0.759202	0.759202	0.764163	0.758011	0.523262	0.759042
Random Forest	0.792178	0.792178	0.794656	0.792003	0.587016	0.793014
Gradient Boosting	0.806748	0.806748	0.809216	0.806646	0.616102	0.807724
Gaussian Bayes	0.632285	0.632285	0.698397	0.594833	0.318800	0.626746

Table 2 Summary of Statistical Models Performance

5. Limitation and Potential Improvement

5.1 Imbalanced Data

As shown in the exploratory data analysis, the distribution of categorical features is quite imbalanced. For example, in HomePlanet data, 52.9% of passengers are from earth, and around 20% of passengers are from Europa and Mars. A similar problem exists in Destination, CryoSleep, and VIP columns. This means that when training the model, most of the information comes from the most common classes and cannot learn enough from other classes of data.

To solve this problem, down-sampling or up-weighting may be conducted on the dataset to solve this problem. The oversampling method SMOTE (Synthetic Minority Oversampling Technique) can also be used to reduce the imbalance among classes. These methods may be able to improve the performance of our models.

5.2 Hyperparameter Tuning

Although we conducted grid search cross-validation when training LDA, Logistic regression, and KNN models, there was a limitation when we tuned the hyperparameter of random forest and gradient boosting. More specifically, we only tuned the number of decision trees included and whether bootstrapping was conducted. Other parameters like max_depth and min_samples_split were not considered in the model selection process. If more hyperparameter choices were included, the model performance of random forest and gradient boosting maybe even better.

5.3 Name

In the data pre-processing procedure, we directly drop the columns containing the name. However, it seems that the information can be contained in the last name or even the first name of passengers. If more exploration was done on the names, model prediction performance may be improved.

Appendix A: Important Outcomes of Exploratory Data Analysis

A.1 Overview of Dataset

Number of Variables	14
Number of Observations	8693
Missing Values Overview (%)	
PassengerId	0.00
HomePlanet	2.31
CryoSleep	2.50
Cabin	2.29
Destination	2.09
Age	2.06
RoomService	2.08
FoodCourt	2.11
ShoppingMall	2.39
Spa	2.11
VRDeck	2.16
Name	2.30
Transported	0.00

A.2 Imbalanced Data

A.2.1 HomePlanet

As in Figure 9, in the column HomePlanet, more than 50% of passengers are from Earth and only 20% from Mars, and 20% from Europa. This means that when constructing a machine learning model, most information comes from passengers from the Earth and not enough information can be learned from other classes.

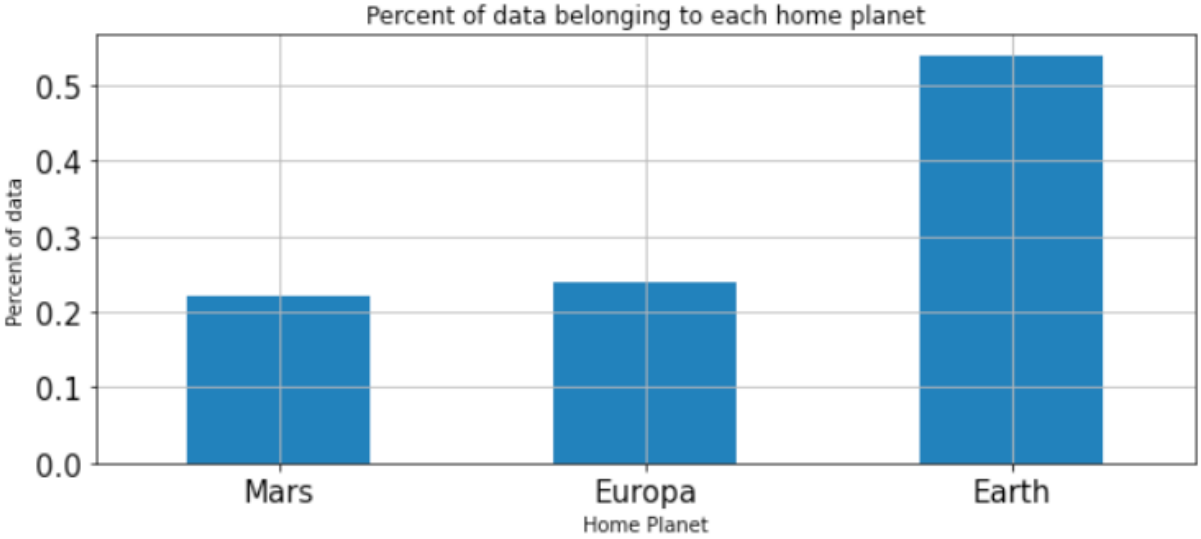

Figure 9 Distribution of HomePlanet

A.2.2 Destination

A similar situation happens in column Destination (Figure 10), and it is even more obvious. More than 70% of passengers have the destination of TRAPPIST-1e and less than 10% of passengers have the destination of PSO J318.5-22, which is quite an imbalance.

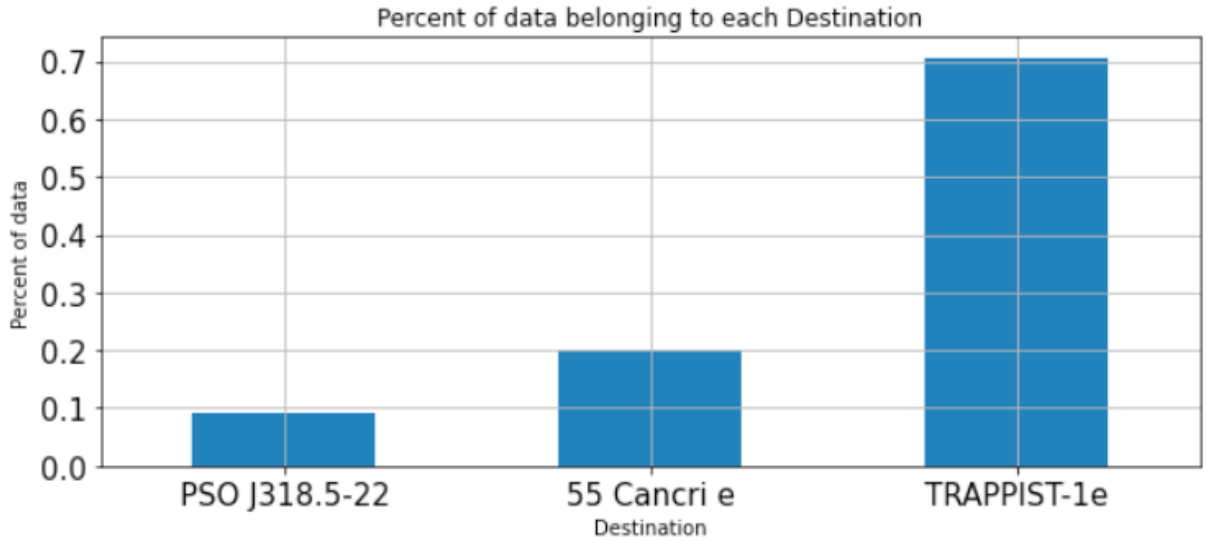

Figure 10 Distribution of Destination

A.2.3 VIP

The imbalance is quite extreme when it comes to column VIP. Over 95% of passengers are not VIPs. This means that we don't have enough information about VIP passengers.

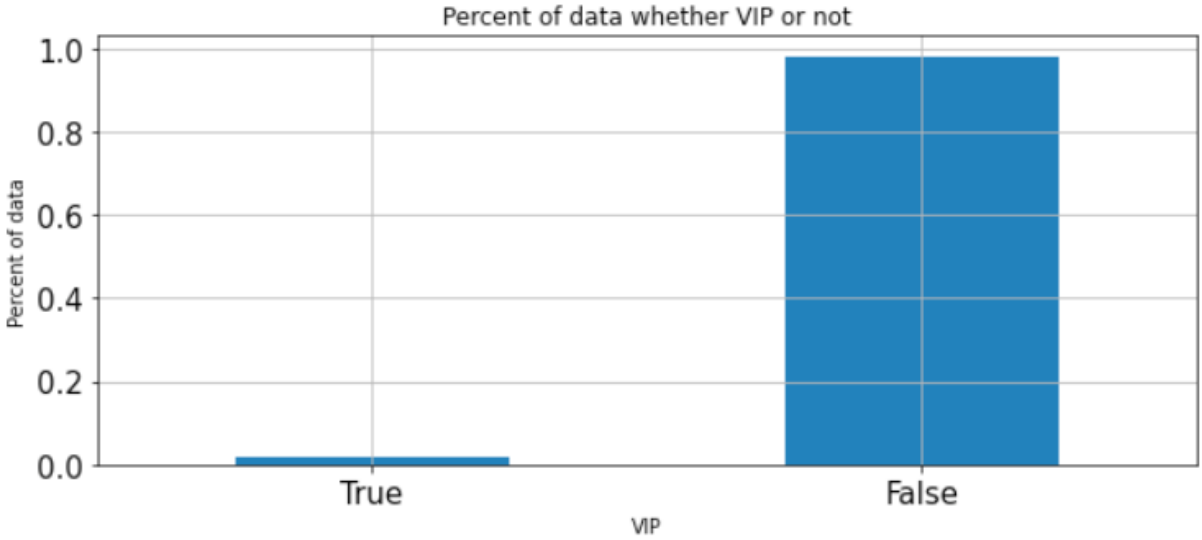

Figure 11 Distribution of VIP

A.3 Skewness

Looking at the statistics of features RoomService, FoodCourt, ShoppingMall, Spa, and VRDeck (Table 3), we see that the data is mainly concentrated under 100.0 but has extreme values that are over 10000. This may cause a negative effect on model performance.

	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck
Count	8512.00	8510.00	8485.00	8510.00	8505.00
Mean	224.69	458.08	173.73	311.14	304.85
Std	666.72	1611.49	604.70	1136.71	1145.72
Minimum	0.00	0.00	0.00	0.00	0.00
25%	0.00	0.00	0.00	0.00	0.00
50%	0.00	0.00	0.00	0.00	0.00
75%	47.00	76.00	27.00	59.00	46.00
Maximum	14327.00	29813.00	23492.00	22408.00	24133.00

Table 3 Descriptive Statistics of Luxury Facility features

The distribution can also be seen by the histogram of these features in Figure 12.

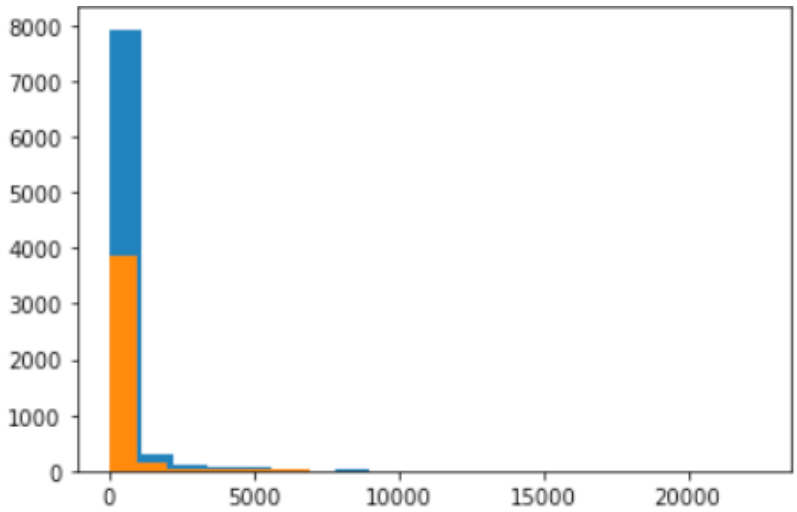

Figure 12 Histogram of Luxury Facility Features

A.4 Correlation

After filling missing values and encoding categorical features, we plotted the correlation matrix heatmap (Figure 13). It shows that a high correlation exists between several features. For example, the correlation between those luxury facilities' costs is about 0.4. HomePlanet also shows a high correlation with deck variables. This may cause problems when we construct some types of models.

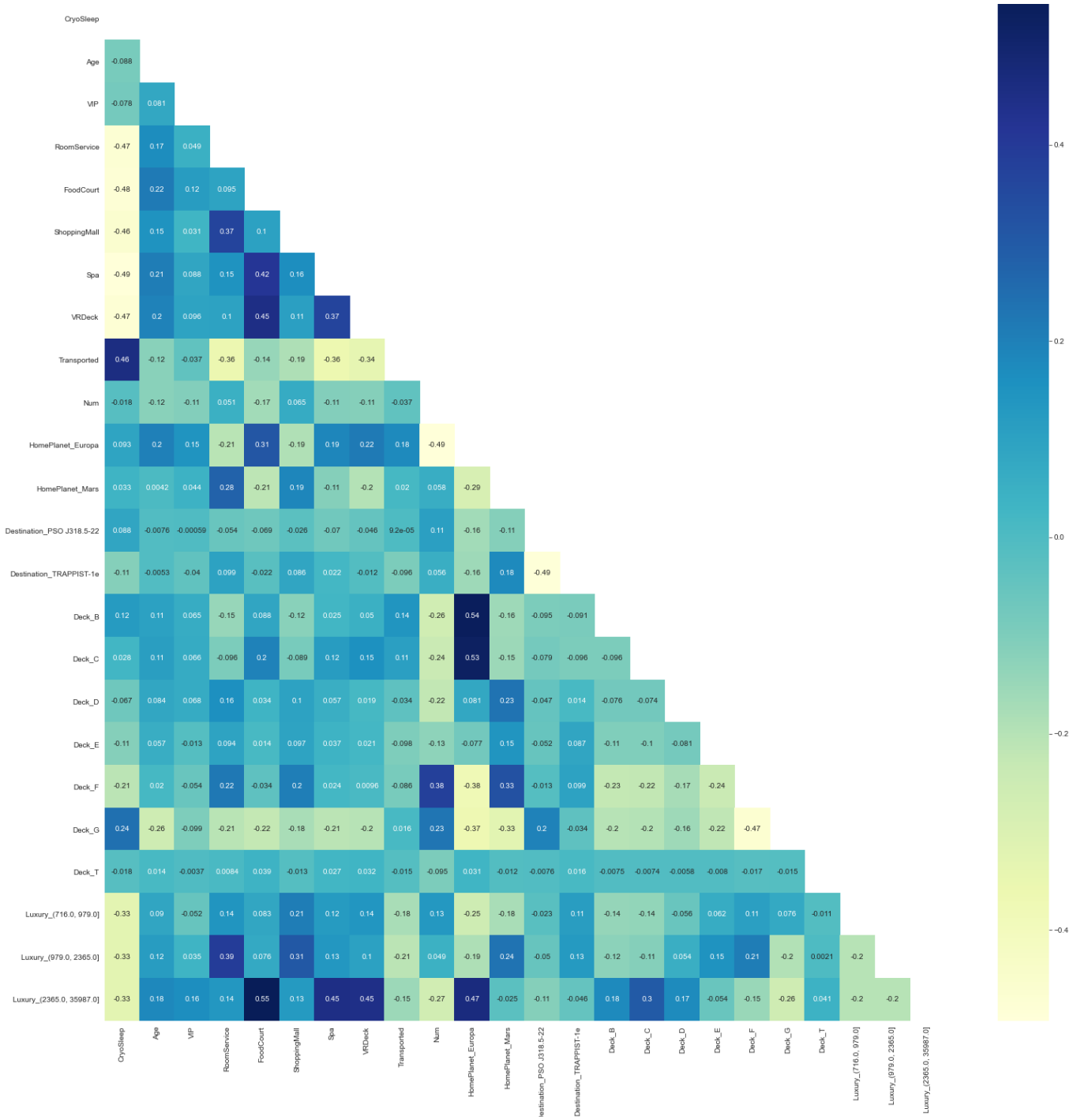

Figure 13 Correlation Heatmap after Data Manipulation