Nicholas SEBALD

Ruicong WANG

Mohamed KANNOU

Qinyi MIAO

Fulin ZHANG

TABLE OF CONTENTS

Introduction

This report explores space travel through data exploration techniques as well as the application of relevant machine learning models to make useful predictions, based on information available in our dataset.

Precisely, we will attempt to make binary classification for our target variable Transported which is boolean and equals 1 if the traveler has been transported to another planet. The dataset comprises 13 other variables; 8 qualitative and 5 quantitative. Based on variables such as HomePlanet, CyroSleep, Destination and Age, we will try to predict whether the traveler made it through the interplanetary voyage.

The information in our dataset is fictional and tries to imagine what space travel could look like in the future. Nevertheless, the underlying logic behind our prediction could be applied to maritime, air and land travel as well, provided of course that we will need more variables than we are using in this dataset to make realistic predictions.

Data Cleaning

After loading the dataset we copy them to ensure that changes made to the new DataFrame do not affect the original data. By displaying the DataFrame information, we can easily find the data types and the number of null values, and we drop the column ‘Name’ that we don’t need to manipulate later, and count the total null values of ‘Cabin’ column, that is used to do feature engineering and feature preprocessing in the following parts.

Data Exploration

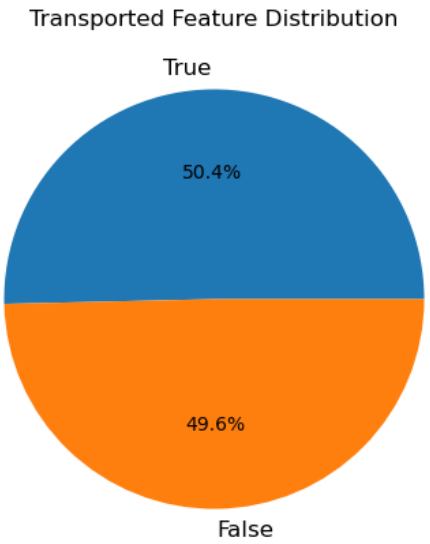


Figure 1. Target Distribution

The pie chart describing the distribution of our Target Feature 'Transported'. It indicates that the values are fairly balanced as there's almost as many "True" values as "False" ones.

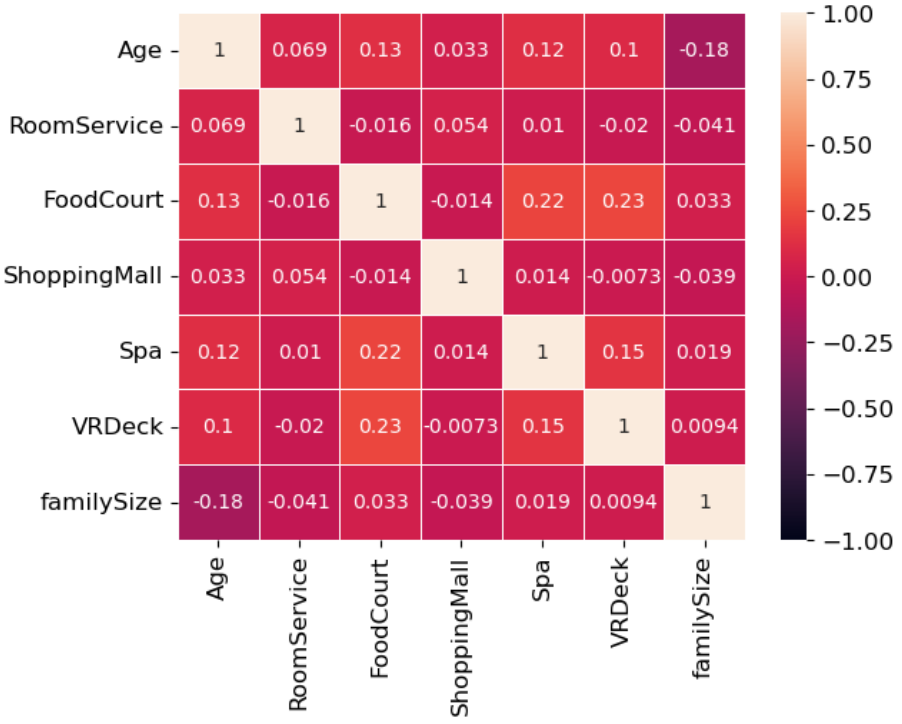


Figure 2. Quantitative Variables Heatmap

The above correlation matrix gives an overview on the correlations between all variables in our dataset. We see that those values are low and it indicated that most variables in our dataset have little to no correlation.

We now investigate the correlation between our variables two by two.

1. **Family Size vs Transported:**
    a. **Anova Table:**

	Source	DF	Sum of Squares	Mean Square	F	PR(>F)
0	False	1	76.191188	76.191188	59.768286	1.186047e-14
1	True	1	75.094787	75.094787	59.768286	1.186047e-14

Figure 3. Anova Table - Family Size v. Target

The results show that there is a statistically significant difference between the group means for both "False" and "True" levels of the independent variable, as the p-values are both less than 0.05. This means that we can reject the null hypothesis that the group means are equal.

b. Boxplot:

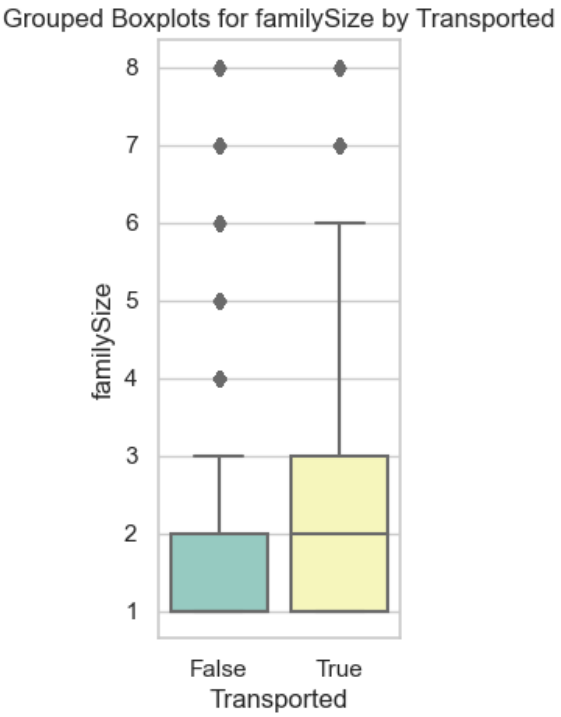


Figure 4. Box Plot of Anova result - Family Size v. Target

This boxplot further supports the fact that there's a statistically significant difference between the means for Family Size groups for different Transported values

2. **Homeplanet vs Transported:**

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.19560087242990837

Chi-Squared Statistic: 324.9013834000382,

p-value: 2.809059985251311e-71

These results suggest that there is a statistically significant association between Homeplanet and Transported variables, but the strength of the association is weak to moderate.

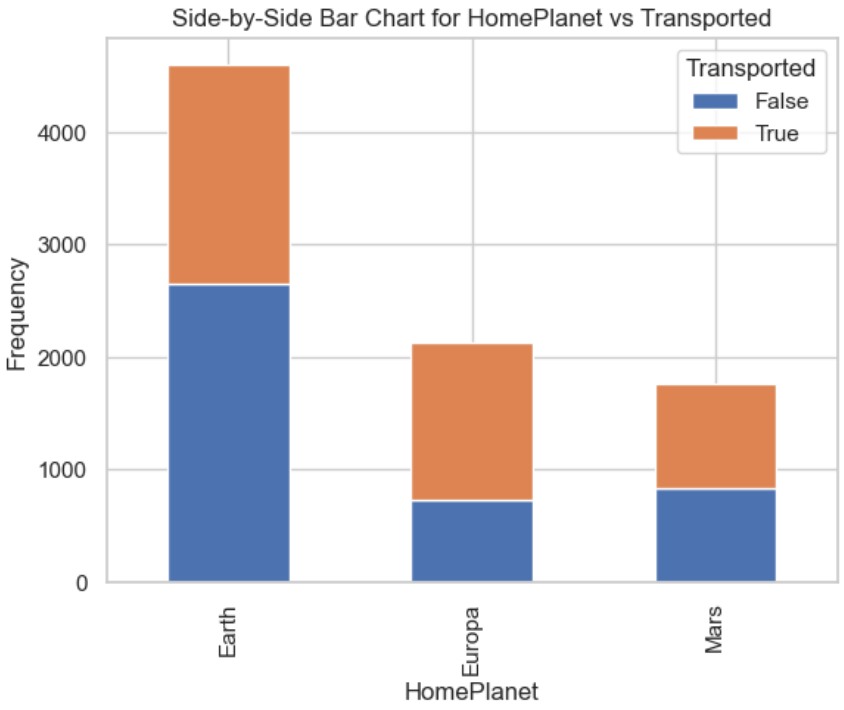


Figure 5. Bar Chart - Home Planet v. Transported

The following bar chart indicates that travelers who are from planet Europa were more relatively likely to be transported, followed by Mars inhabitants and lastly Earth inhabitants.

This stacked bar chart further supports our findings as we see that those from Europa who were transported largely outnumber those who weren't, those transported from Mars also

slightly outnumber those who weren't, and inversely, those from Earth who weren't transported outnumber those who were transported.

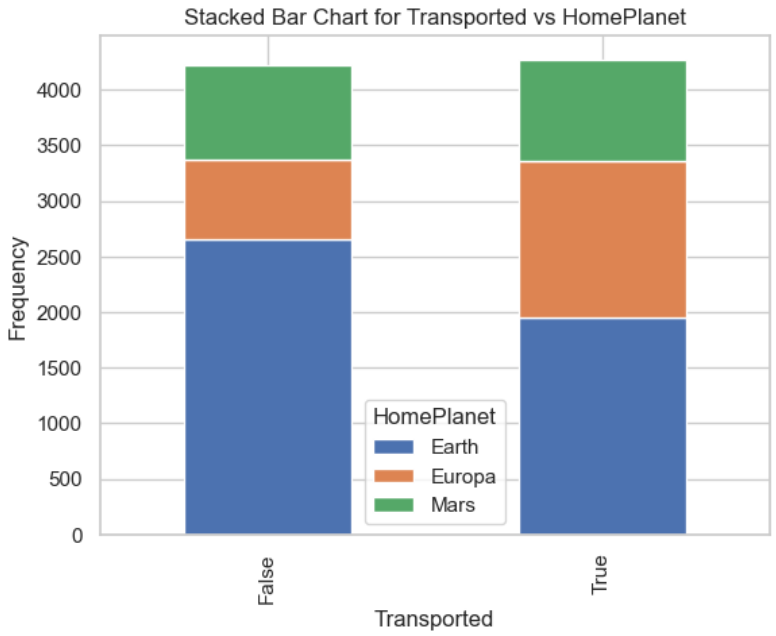


Figure 6. Stacked Bar Chart - Home Planet v. Transported

3. Cryosleep vs Transported:

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.4683988058935883

Chi-Squared Statistic: 1859.6127129888841

p-value: 0.0

These results suggest that there is a statistically significant moderate association between Cryosleep and Transported.

This side-by-side bar chart for Cryosleep vs Transported shows that among those transported, a high proportion underwent cryosleep. It also shows that most of those who weren't transported didn't undergo cryosleep. Based on the bar chart, If you did cryosleep, you were more likely to be transported and vice versa.

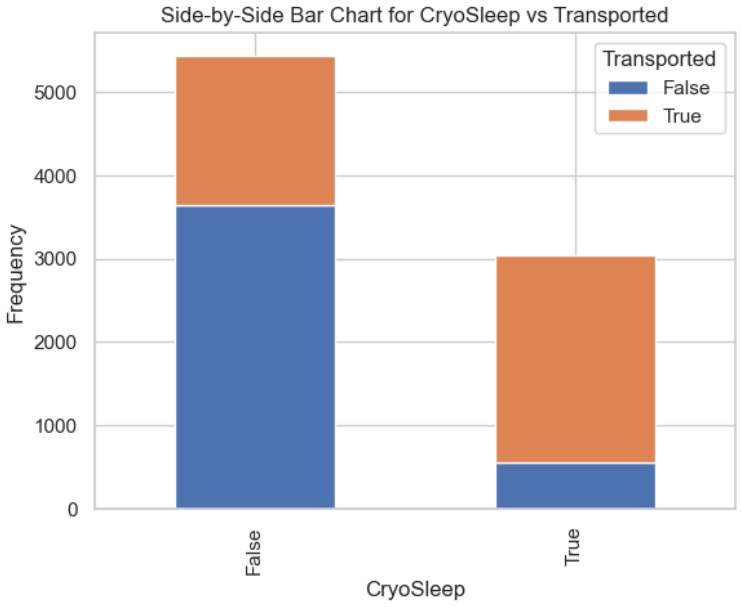


Figure 7. Stacked Bar Chart - CryoSleep v. Transported

This stacked bar chart supports our earlier observations. More than half of travelers who were transported underwent cryosleep. Whereas the majority of those who weren't transported didn't undergo cryosleep.

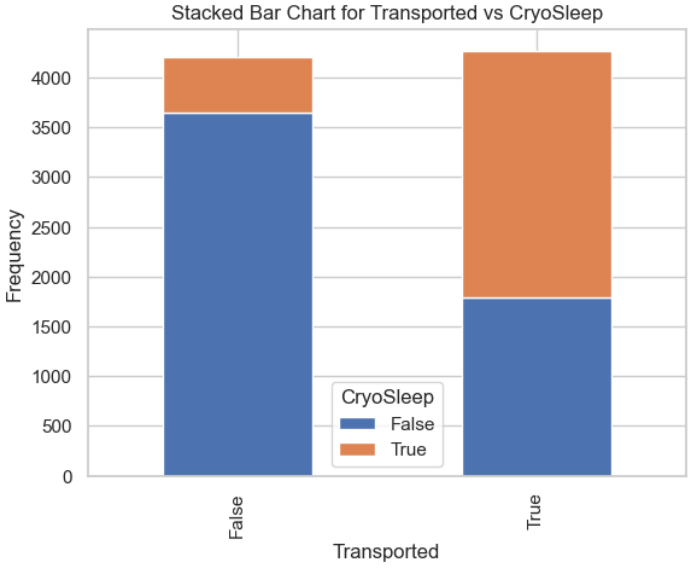


Figure 8. Stacked Bar Chart - Transported v. CryoSleep

4. **Destination vs Transported:**

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.11180584580691638

Chi-Squared Statistic: 106.39215684982227

p-value: 7.892901466137099e-24

These results suggest that there is a statistically significant association between Destination and Transported variables, but the strength of the association is weak to moderate.

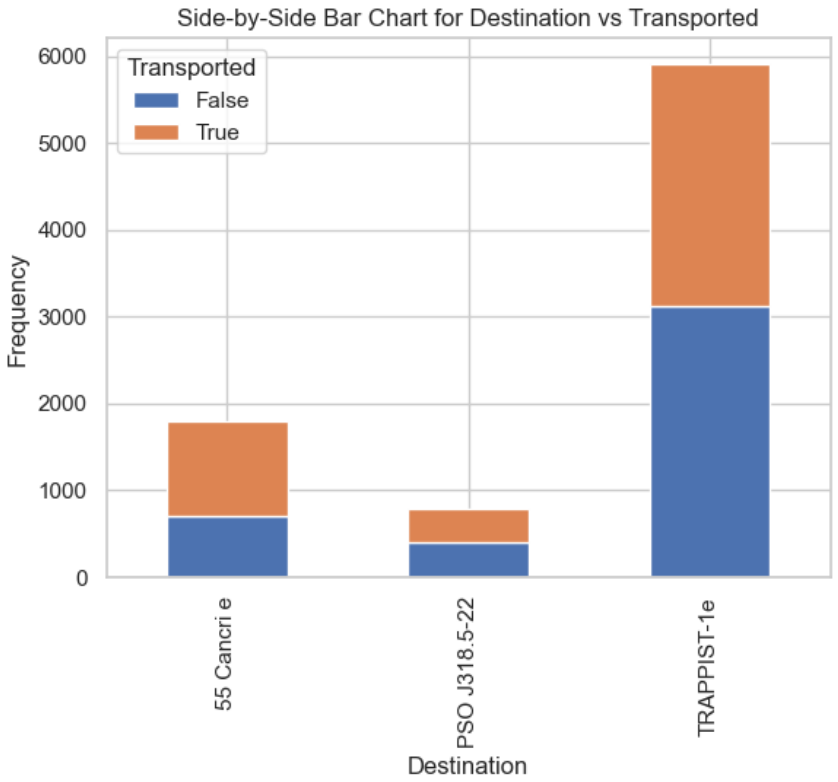


Figure 9. Stacked Bar Chart - Destination v. Transported

Based on this side-by-side bar chart for destination vs transported, we see that the travelers who are going to 55 Cancri e were more likely to to be transported with a higher proportion for those who did vs those who did not. This is followed by those heading to PSO J318.5-22 with a higher proportion of those who were transported, and finally the inverse is true for those heading to TRAPPIST-1e as more than half didn't make it to their destination.

5. VIP vs Transported:

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.036871183678923294

Chi-Squared Statistic: 11.542020738162797

p-value: 0.0006804064556968345

These results suggest that there is a statistically significant association between Destination and transported variables, but the strength of the association is very weak to non-existent.

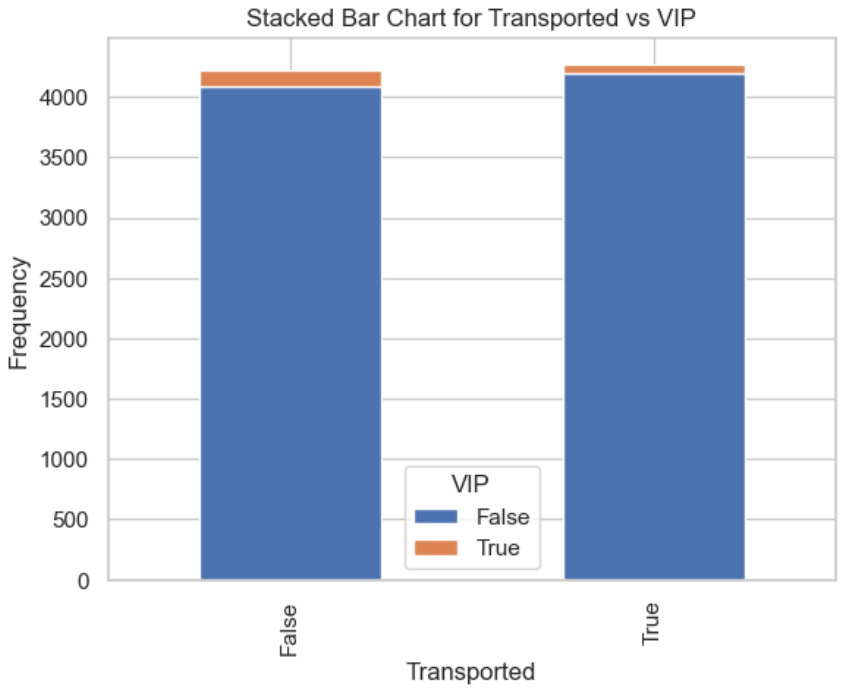


Figure 10. Stacked Bar Chart - VIP v. Transported

While there may be some relationship between the variables, it is not meaningful.

Surprisingly, this stacked bar char shows that those who were VIP were less likey to be transported. However, VIP passengers were a small portion of passengers.

6. **Deck vs Transported:**

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.214904943695003

Chi-Squared Statistic: 392.2880411997481

p-value: 1.0743307258871414e-80

These results suggest that there is a statistically significant association between Deck and Transported variables, but the strength of the association is weak to moderate.

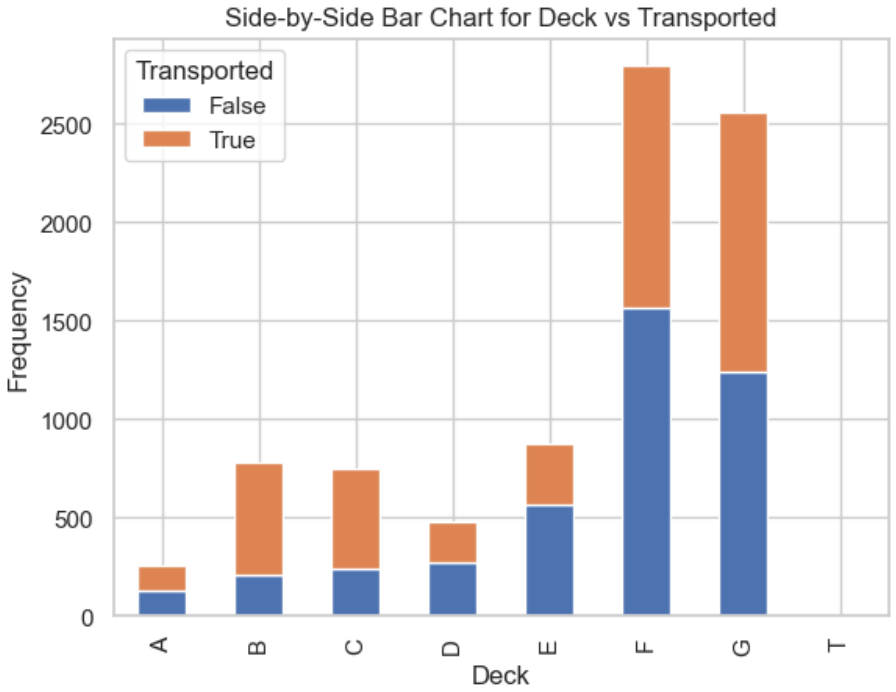


Figure 11. Stacked Bar Chart - Deck v. Transported

Those in deck B, deck C and deck G were significantly more likely to be transported with a higher proportion making it to destination. Those in deck A had a balanced proportion for being transported and those in D, E and F were less likely to make it to destination. Those in deck T were the least likely to be transported.

7. **Side vs Transported:**

Calculating the contingency table between homeplanet and transported variables and testing the association between the two variables produced the following values:

Cramer's V: 0.10353965983288271

Chi-Squared Statistic: 91.0595970786772

p-value: 1.3940936254458288e-21These results suggest that there is a statistically significant association between Side and Transported variables, but the strength of the association is weak to moderate.

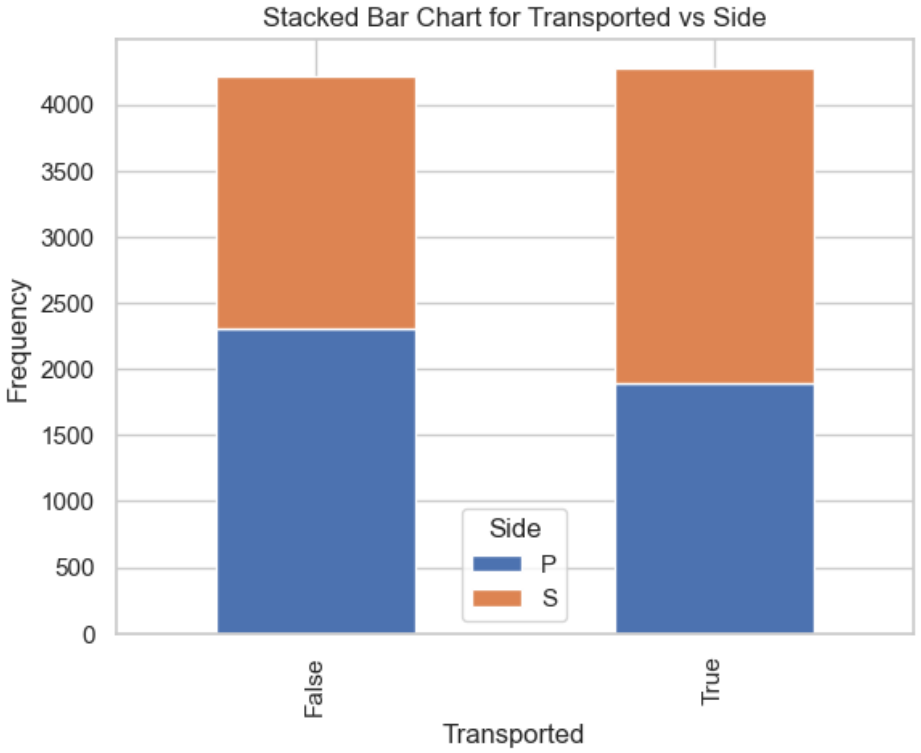


Figure 12. Stacked Bar Chart - Transported vs. Ship Side (Port - Starboard)

We see that those on the S side were more likely to be transported, and inversely those on the side P were less likely to be transported. Nevertheless, there is a big difference between the two groups even though it's statistically significant.

Feature engineering

For the first step, we created three new columns named ‘Deck’ , ‘Num’ and ‘Side’ which were extracted from column ‘Cabin’. The ‘Deck’ column contains the content of ‘Cabin’ before the first slash. The ‘Num’ column contains the content of ‘Cabin’ between the first and second slashes. The ‘Side’ column contains the content of ‘Cabin’ after the second slash. The reason we create these three columns is that we want know if they are correlated with ‘Transported’

```
df["Deck"] = df["Cabin"].apply(
    lambda x: x.split("/")[0] if isinstance(x, str) and "/" in x else np.nan
)
df["Num"] = df["Cabin"].apply(
    lambda x: x.split("/")[1] if isinstance(x, str) and "/" in x else np.nan
)
df["Side"] = df["Cabin"].apply(
    lambda x: x.split("/")[2] if isinstance(x, str) and "/" in x else np.nan
)
```

Figure 13. Splitting “Cabin” data to 3 columns

For the second step, we created two new columns which are called ‘familyId’ and ‘familySize’. For the column ‘familyId’ we extract numbers from ‘PassengerId’ and try to find the numbers before underscore. For the ‘familySize’ we try to count how many passengers have the same familyId’, because we want to know how big the family is. The reason we create these two columns is that we want know if they are correlated with ‘Transported’

```
df["familyId"] = df["PassengerId"].str.extract(r"(\d+)_")
df["familySize"] = df.groupby("familyId")["PassengerId"].transform("count")
```

Figure 14. Splitting “PassengerI” feature

For the third step, we removed columns ‘PassengerId’, ‘Cabin’ and ‘familyId’, because we already use them to extract what we need, so now they are useless, we don’t need them anymore.

```
df.drop(columns=["PassengerId", "Cabin", "familyId"], axis=1, inplace=True)
```

Figure 15. Column drop

For the final step, we need to check if the columns we created already appeared and the columns we removed already disappeared on the table.

Model Selection

First step we defined our features and outcomes. We set ‘Transport’ as our target variable as we wanted to predict if the passenger was transported or not.

Second, we split our data into test data and training data. The training data was used to train our models and the test data to evaluate the performance of our models.

Third step we divided our variables into categorical variables and numeric variables.

Fourth step we checked the unique variables in the categorical variables and then we checked which features have many variations and which have fewer. From the answers we can see that 'HomePlanet', 'CryoSleep', 'Destination', 'VIP', 'Deck', 'Side' have fewer variations.

Fifth step, we need to check the missing values, once we find it we will use the mean value to fill it. This step is to make sure everything is fine and we can do the next step.

Sixth step, we preprocess our data to make sure every column is without NaN and transfer the categorical data to numerical data.

Seventh step, we chose Logistic Regression, Decision Tree Classifier, Random Forest Classifier and KNN as our prediction models.

Last step we start to training our models, from the results we can see as follows:

Classification Report for Logistic Regression:

Training set:

	precision	recall	f1-score	support
False	0.80	0.77	0.79	3452
True	0.78	0.81	0.80	3502
accuracy			0.79	6954
macro avg	0.79	0.79	0.79	6954
weighted avg	0.79	0.79	0.79	6954

Test set:

	precision	recall	f1-score	support
False	0.79	0.78	0.79	863
True	0.79	0.80	0.79	876
accuracy			0.79	1739
macro avg	0.79	0.79	0.79	1739
weighted avg	0.79	0.79	0.79	1739

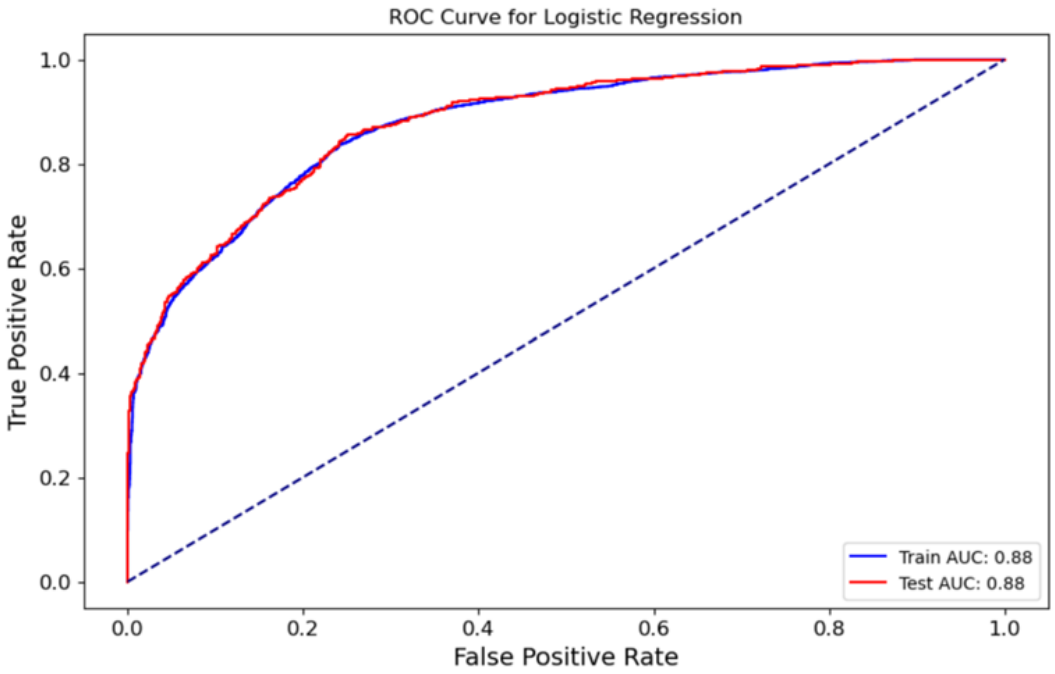


Figure 16. Logistic Regression Score Output

1. The Logistic Regression model performs consistently on both the training and test data with an accuracy and AUC of approximately 0.79 and 0.88, respectively.

2. Given that both training and test metrics are close, it indicates the model is neither overfitting nor underfitting.

3. An AUC of 0.88 is relatively high, indicating that the model has a good measure of separability and is skilled in distinguishing between the positive and negative classes.

4. The Precision of False is 0.80 and recall is 0.77. This means that of all instances the model predicted as False, 80% were actually False. Also, of all actual False instances, the model correctly identified 77%. The Precision of True is 0.78 and recall is 0.81. Of all instances predicted as True, 78% were actually True. Of all actual True instances, the model correctly identified 81%. The model is 79% accurate on the training data. The model is also 79% accurate on the test data, consistent with the training set. The area under the curve (AUC) for the training data is 0.88. AUC ranges from 0 to 1, with a value of 1 indicating a perfect classifier and 0.5 indicating a random classifier. An AUC of 0.88 is a strong score. The AUC for the test data is also 0.88, suggesting that the model performs similarly on both the training and test data.

Classification Report for Decision Tree:

Training set:

	precision	recall	f1-score	support
False	1.00	1.00	1.00	3452
True	1.00	1.00	1.00	3502
accuracy			1.00	6954
macro avg	1.00	1.00	1.00	6954
weighted avg	1.00	1.00	1.00	6954

Test set:

	precision	recall	f1-score	support
False	0.73	0.75	0.74	863
True	0.75	0.73	0.74	876
accuracy			0.74	1739
macro avg	0.74	0.74	0.74	1739
weighted avg	0.74	0.74	0.74	1739

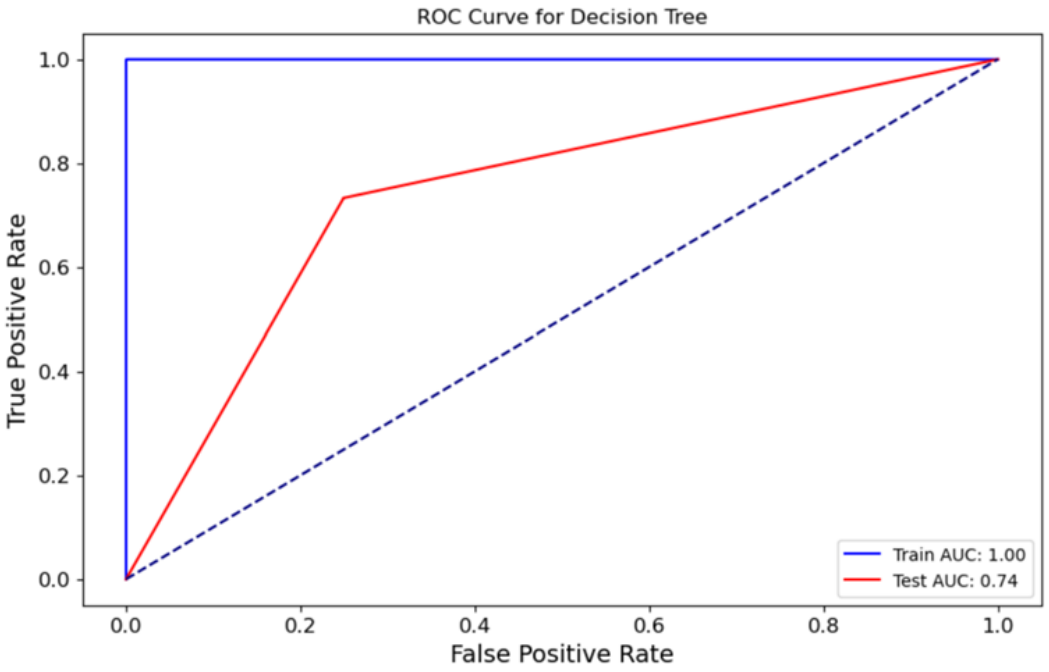


Figure 17. Decision Tree Classifier Score Output

1. Overfitting: The Decision Tree model seems to be overfitting on the training data. This is evident from the perfect scores (1.00) on the training set compared to the scores on the test set, which are lower (around 0.74).

2. Test Performance: The model's performance on the test set is reasonably good, with an accuracy and AUC of 0.74. However, there's a notable difference between training and test performances, indicating potential overfitting.

3. ROC Curve: The curve for the training set goes straight up, indicating perfect classification. The test curve, on the other hand, deviates from the top left corner, showing that the model isn't perfect on unseen data.

4. For the training set precision, recall, and F1-score for both 'True' and 'False' labels are 1.00. This indicates perfect classification on the training set. The accuracy of the model on the training set is also 1.00, which means the model has classified all the training data correctly. For the test set precision for 'False' is 0.73 and for 'True' is 0.75. Recall for 'False' is 0.75 and for 'True' is 0.73. The F1-scores are 0.74 for both. The accuracy of the model on the test set is 0.74. The ROC curve visualizes the performance of a binary classifier, plotting the true positive rate (sensitivity) against the false positive rate (1-specificity). The AUC (Area Under Curve) for the training set is 1.00, indicating perfect classification. The AUC for the test set is 0.74. With such a high training set score, we were wary of overfitting.

Classification Report for Random Forest:

Training set:

	precision	recall	f1-score	support
False	1.00	1.00	1.00	3452
True	1.00	1.00	1.00	3502
accuracy			1.00	6954
macro avg	1.00	1.00	1.00	6954
weighted avg	1.00	1.00	1.00	6954

Test set:

	precision	recall	f1-score	support
False	0.78	0.84	0.81	863
True	0.83	0.76	0.80	876
accuracy			0.80	1739
macro avg	0.81	0.80	0.80	1739
weighted avg	0.81	0.80	0.80	1739

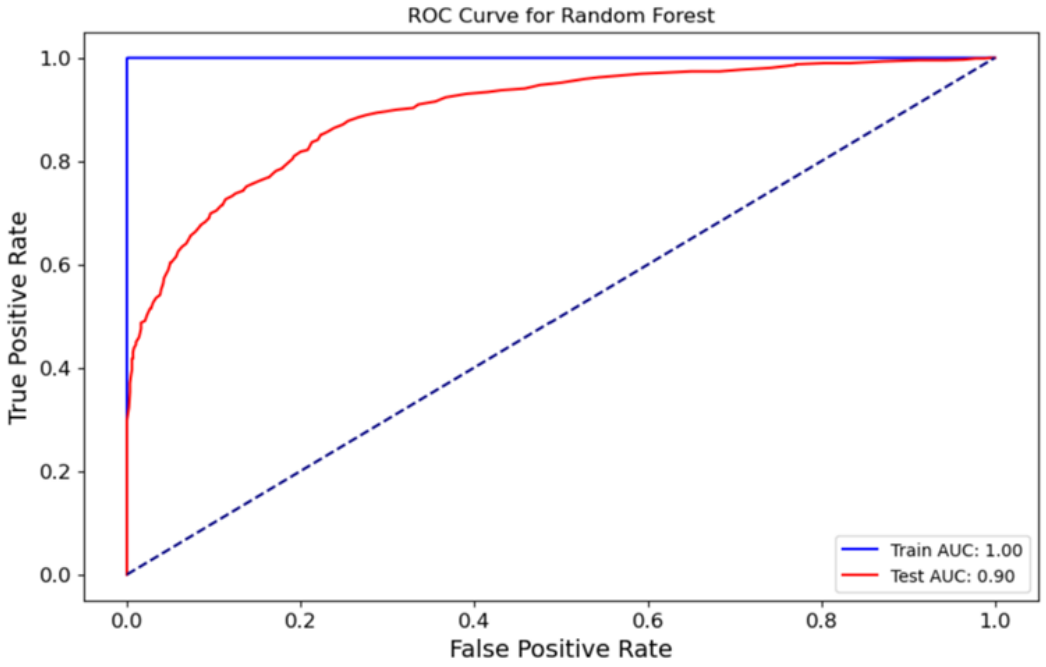


Figure 18. Random Forest Classifier Score Output

1. Overfitting: The Decision Tree model seems to be overfitting on the training data. This is evident from the perfect scores (1.00) on the training set compared to the scores on the test set, which are lower (around 0.74).

2. Test Performance: The model's performance on the test set is reasonably good, with an accuracy and AUC of 0.74. However, there's a notable difference between training and test performances, indicating potential overfitting.

3. ROC Curve: The curve for the training set goes straight up, indicating perfect classification. The test curve, on the other hand, deviates from the top left corner, showing that the model isn't perfect on unseen data.

4. The model achieved perfect precision, recall, and f1-score values of 1.00 for both the classes (`True` and `False`). The overall accuracy of the model on the training set is 1.00. For the test set, the overall accuracy on the test set is 0.80. The AUC value for the training set is 1.00, which means the model has perfect discriminative power on the training data. The AUC value for the test set is 0.90, which is quite high and indicates good discriminative power on the test data.

Classification Report for KNN:

Training set:

	precision	recall	f1-score	support
False	0.84	0.79	0.81	3452
True	0.80	0.85	0.83	3502
accuracy			0.82	6954
macro avg	0.82	0.82	0.82	6954
weighted avg	0.82	0.82	0.82	6954

Test set:

	precision	recall	f1-score	support
False	0.79	0.75	0.77	863
True	0.76	0.80	0.78	876
accuracy			0.77	1739
macro avg	0.77	0.77	0.77	1739
weighted avg	0.77	0.77	0.77	1739

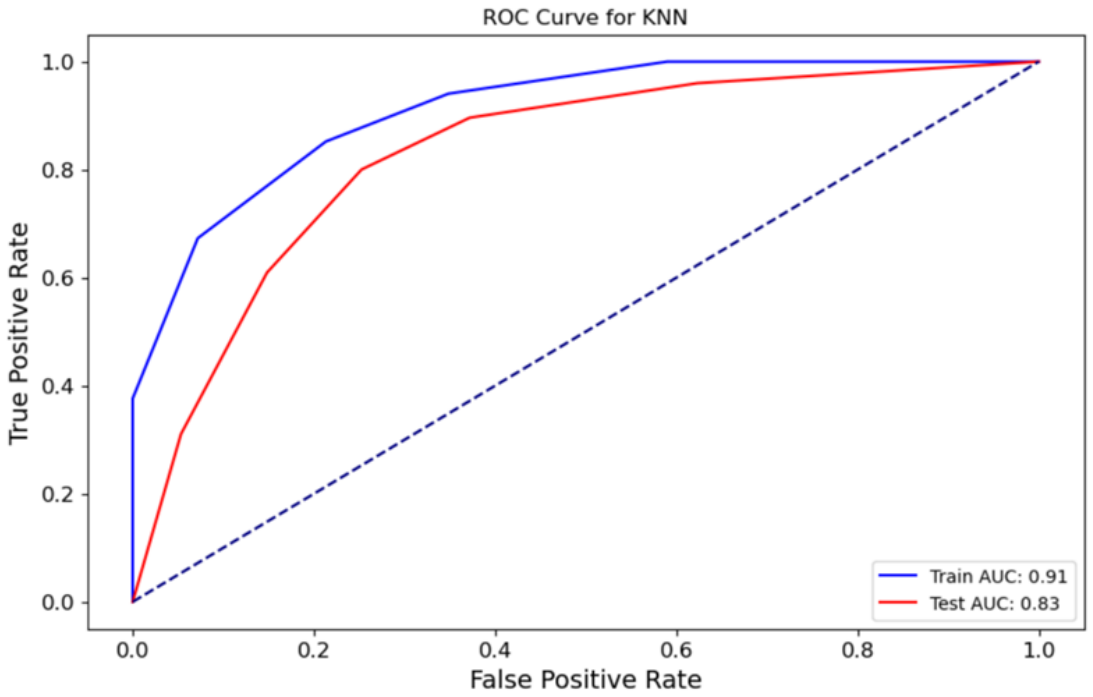


Figure 19. KNN Classifier Score Output

1. The KNN model seems to perform well on both the training and test sets, with accuracy scores of 82% and 77% respectively.

2. There's a slight overfitting observed, as the training metrics are a bit higher than the test metrics. This is a common challenge and may require regularization, more data, or tuning to address.

3. The AUC for both the training and test sets are good, with the training set having an exceptionally high AUC of 0.91.

4. This tells us the proportion of correct predictions among the total number of instances. An accuracy of 0.82 means the model correctly predicted the outcome for 82% of instances in the training set. AUC values range from 0 to 1, with higher values indicating better classifier performance. In this case, the training AUC is 0.91, indicating very good classifier performance on the training data. The test AUC is 0.83, which is slightly lower but still indicates strong classifier performance on the test data.

Hypertuning

First step is pipeline creation. The two main tasks are preprocessing, which is when we cleaned and organize our data and the preparation for the next step and classification, where we start to do the prediction.

Second step is to choose the KNN model to search for the best combination of settings according to the previous evaluation. The result of the score of performance is 0.7725, so we can summarize that with the best parameters (using the Manhattan distance metric, considering 9 neighbors, and giving each neighbor equal weight), the K-Nearest Neighbors algorithm was able to correctly predict outcomes approximately 77.25% of the time.

We went through two phases of evaluating Random Forest Classifier parameters. After trying 720 different combinations of parameters, the model found that a forest of 200 trees, with a maximum depth of 30, requiring at least 2 samples to make a final decision (leaf) and at least 2 samples to consider further splitting, gave the best results, predicting correctly 80.73% of the time.

After our next attempt experimenting with 500 distinct combinations of parameters, the model identified that a forest consisting of 231 trees, with no preset maximum depth, requiring a minimum of 3 samples for both leaf formation and node splitting, produced the best results, predicting correctly about 80.85% of the time. These results were in line with what was produced before hyper tuning, thus the effort proved not very useful.

Dimensionality Reduction

We tried to run the SVM Model, but it took us a longtime, which may be due to the model's features and extensive dataset, so we decided to use PCA to do dimensional reduction to increase the efficiency.

Before we did PCA, we need to do some preprocessing. First, we need to transform variables X(which exclude the target 'Transported', and we divide it by three subgroups they are respectively 'num', 'low_card_cat' and 'high_card_cat') by preprocessor, which is used to fill missing values and do one-hot encoding.

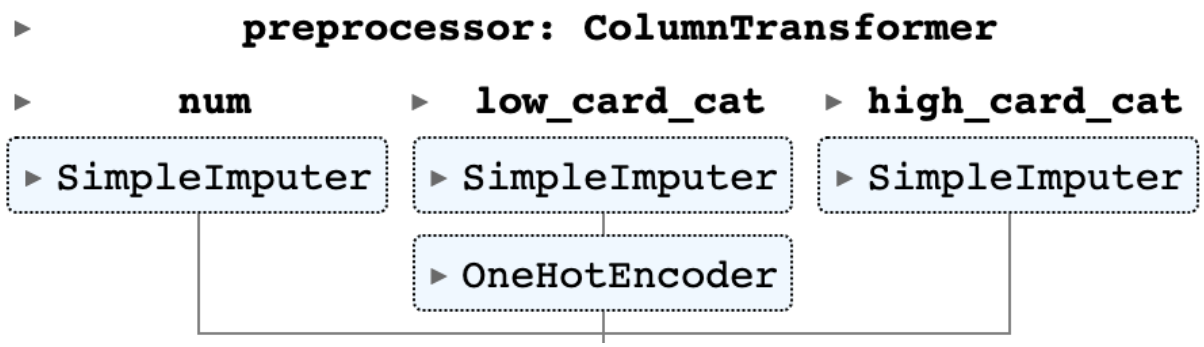
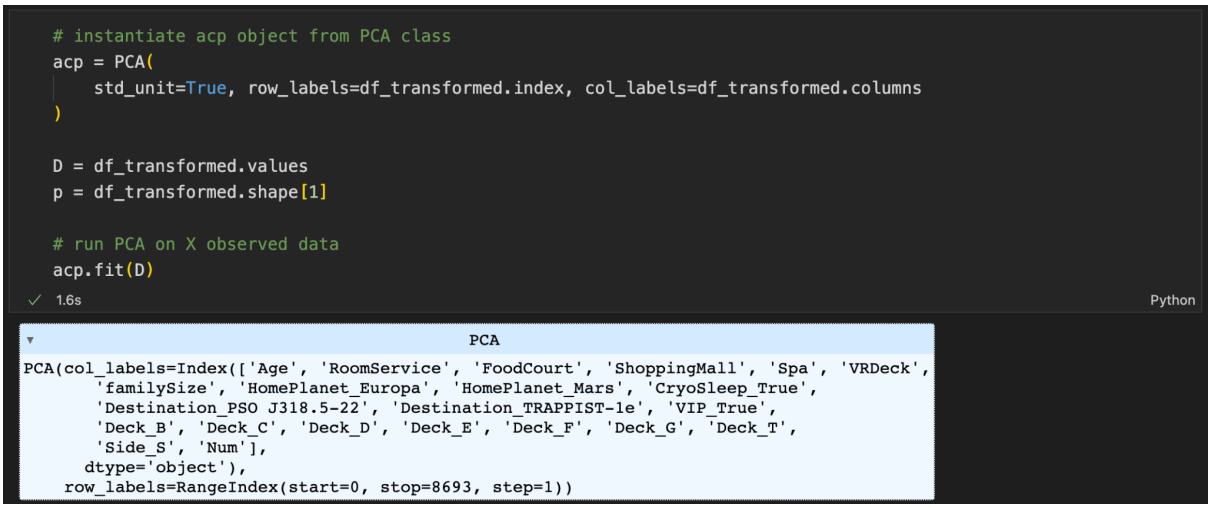


Figure 20. Preprocessor visualization

After the previous step, we saw the new df_transformed dataframe has been well encoded and has no missing value, but the type of the features are all 'object' so we change them to type 'float 64'.

We can now do PCA for these features, in the parameter of PCA, std_unit means standardize the data before PCA, because it makes features comparable, and PCA is sensitive to the scaling of the data.



```
# instantiate acp object from PCA class
acp = PCA(
    std_unit=True, row_labels=df_transformed.index, col_labels=df_transformed.columns
)

D = df_transformed.values
p = df_transformed.shape[1]

# run PCA on X observed data
acp.fit(D)
```

✓ 1.6s Python

PCA

```
PCA(col_labels=Index(['Age', 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa', 'VRDeck',
       'familySize', 'HomePlanet_Europa', 'HomePlanet_Mars', 'CryoSleep_True',
       'Destination_PSO J318.5-22', 'Destination_TRAPPIST-1e', 'VIP_True',
       'Deck_B', 'Deck_C', 'Deck_D', 'Deck_E', 'Deck_F', 'Deck_G', 'Deck_T',
       'Side_S', 'Num'],
      dtype='object'),
    row_labels=RangeIndex(start=0, stop=8693, step=1))
```

Figure 21. PCA code reference

After we instantiate acp object from PCA class, we can get the Eigenvalues of the data, and plot Scree Plot, which is used to determine the number of factors to retain in an exploratory factor analysis or principal components to keep in a principal component analysis. According to the Elbow Method, as we can see in the diagram there isn't a significant change in the rate of decrease (If we use Kaiser-Guttman's Rule to decide the number of factors that we need to retain that will be 6 and this 6 factors will explain nearly 50% of variance), this can also be proved it the Explained variance plot, it's clearly to see that the line increase steadily. If we want to explain 95% of the variance in our modeling, we will preserve the first 17 principal components. The small reduction in variables seemed insignificant for future efficiency, so we decided to not dimensionally reduce and move forward with different models.

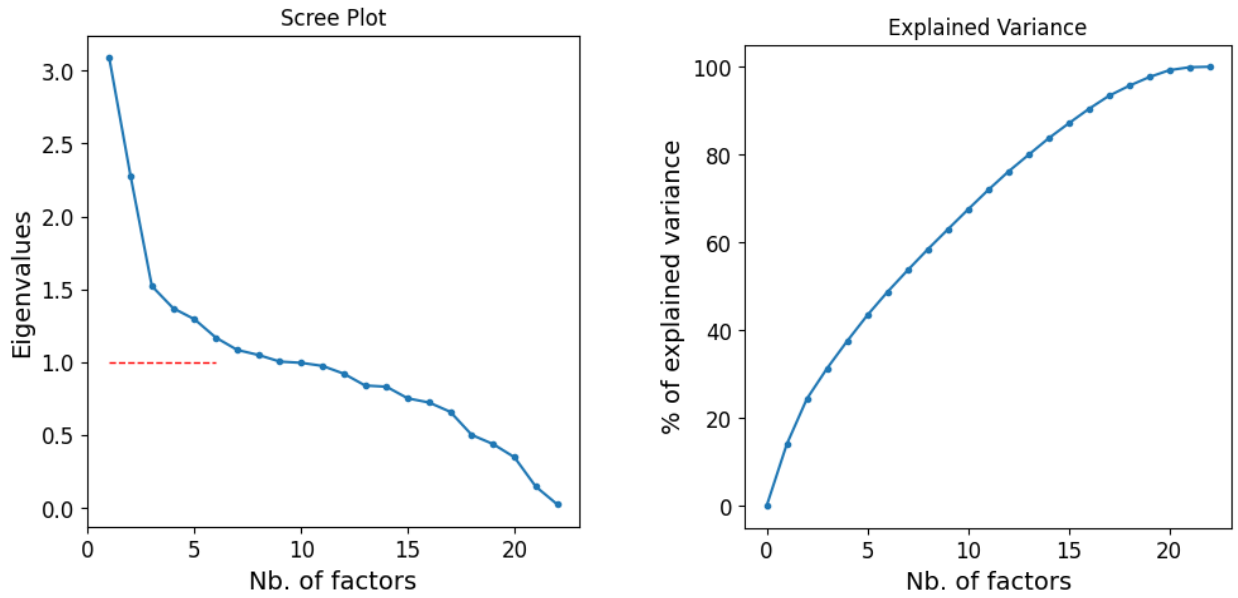


Figure 22. PCA - Scree Plot and Explained Variance based on Principal Components

Clustering

A good clustering would have close distance within a cluster and large distance between clusters. In this part, we tried K-Means to do clustering. We use df_transformed preprocessor to transform out data first, after instantiate and K-Means and predict clusters, we create a new column 'Cluster' to store different subgroup of the data (the number of cluster is 12 as we code on the parameter 'n_clusters=12' of K-Means) and we see the scatter plot of feature 'Age' and feature 'RoomService' their distribution of different clusters, but we didn't get a meaningful discovery, then we drop the 'Cluster' column and move to next step which is exploring the cluster between feature 'familySize' and feature 'Age'.

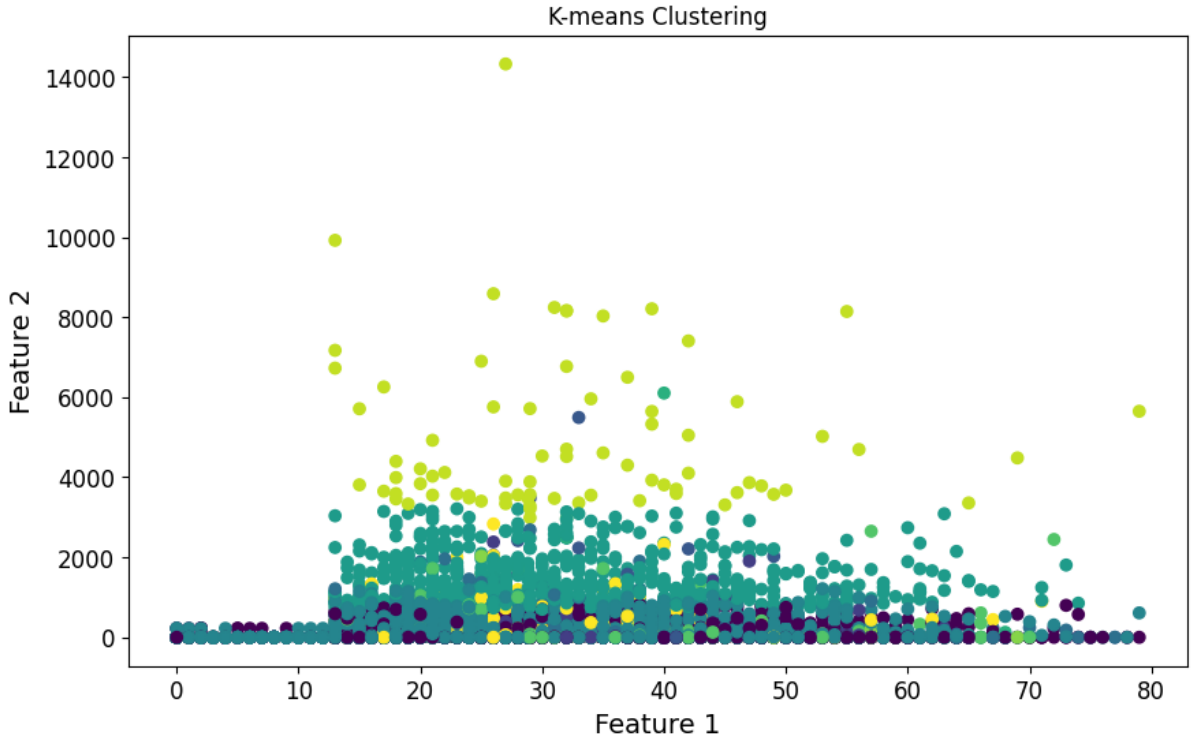


Figure 23. Clustering output - Age v. Room Service

We plot distortion for different values of k between 2 and 28 to see whether there is an 'Elbow' to find the optimal number of clusters in our modeling, but as the plot shows, we didn't find the point where the inertia begins to decrease more slowly, because this line is a smoothly descending curve.

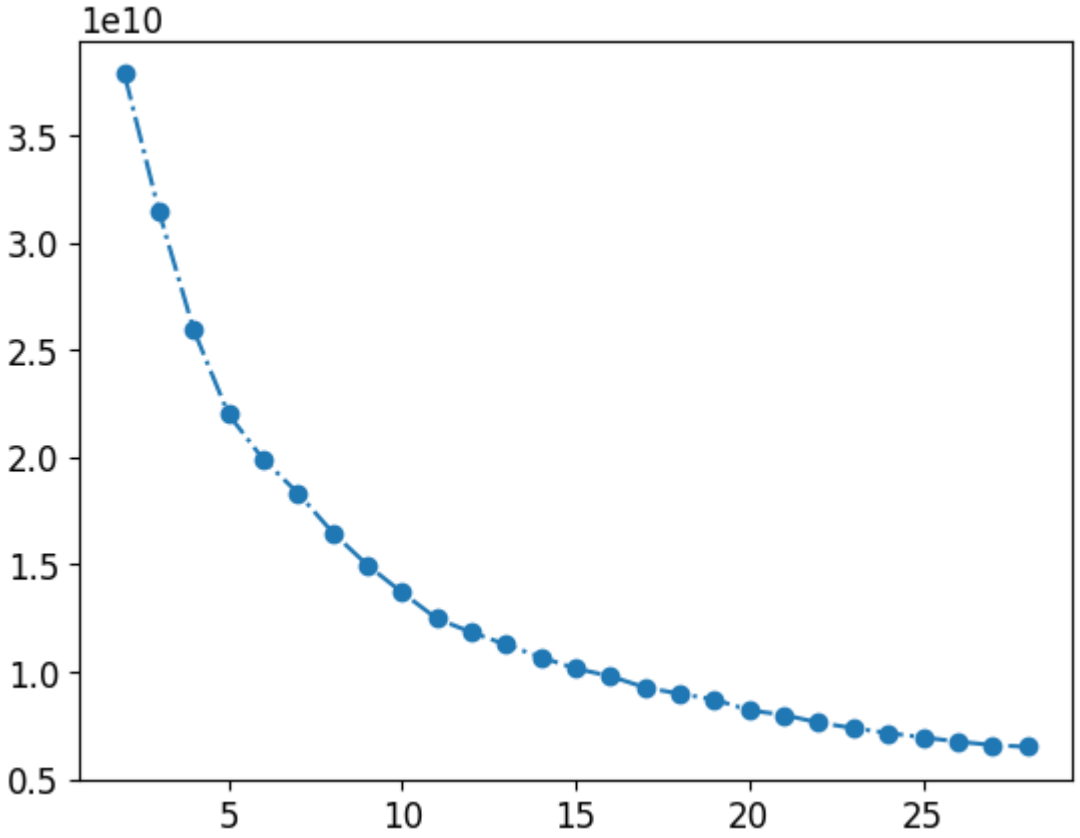


Figure 24. Elbow Plot for ideal cluster count

We also tried to use 'Silhouette plot' to visualize how well the data belongs to the cluster, after getting the silhouette score, the distribution is not ideal for clustering. We do see a nice grouping past the average, however, we are also seeing some negative values, which would need to be explored.

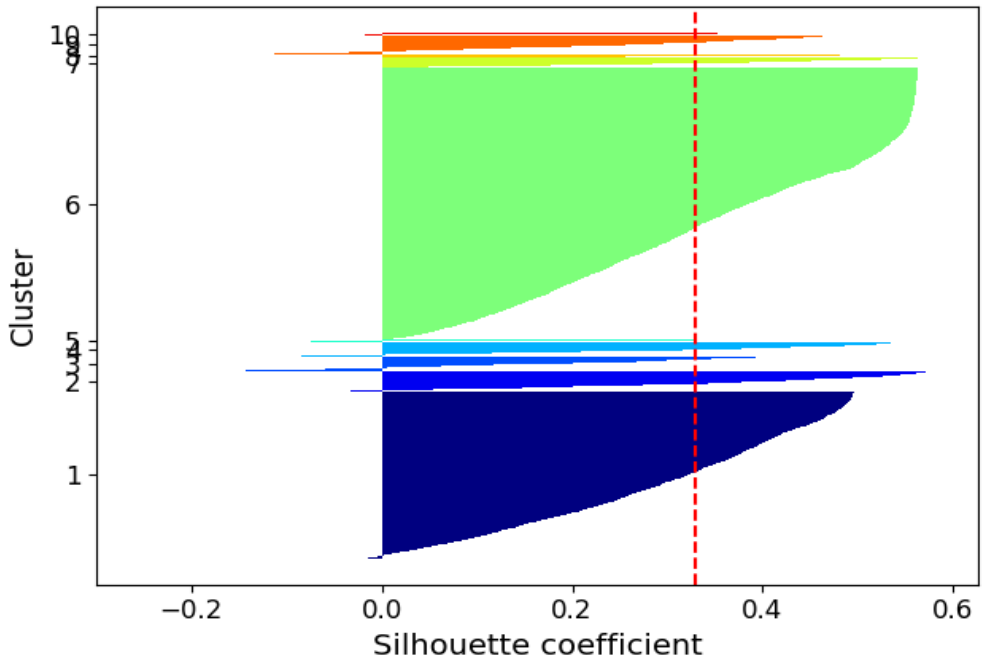


Figure 25. Silhouette coefficient plot

Results

Based on the previous scores, the final model we chose is the Random Forest Classifier. After a series of hyperturning, we derived the importance of each feature value as follows:

Feature low_card_cat__CryoSleep_True: 0.13978371249321797

Feature num__Spa: 0.125759232844655

Feature num__RoomService: 0.12373104923319786

Feature num__VRDeck: 0.11547911116436434

Feature num__FoodCourt: 0.08858460744292337

Feature high_card_cat__Num: 0.08689672978743225

Feature num__ShoppingMall: 0.0746938536276062

Feature num__Age: 0.07126115819771499

Feature low_card_cat__HomePlanet_Europa: 0.02876561999019842

Feature num__familySize: 0.02502028413925926

Feature low_card_cat__Side_S: 0.02063176445576493

Feature low_card_cat__Deck_G: 0.01983014964361373

Feature low_card_cat__HomePlanet_Mars: 0.016942205332084223

Feature low_card_cat__Deck_F: 0.01525930649485904

Feature low_card_cat__Deck_E: 0.011658108374865066

Feature low_card_cat__Destination_TRAPPIST-1e: 0.011080809111257634

Feature low_card_cat__Deck_B: 0.00847861364343979

Feature low_card_cat__Deck_C: 0.007778511658315425

Feature low_card_cat__Destination_PSO J318.5-22: 0.0043789442926881715

Feature low_card_cat__Deck_D: 0.0027880164885257593

Feature low_card_cat__VIP_True: 0.0011982115840166374

Feature low_card_cat__Deck_T: 0.0

Based on the results from the Random Forest feature importance analysis, several insights can be drawn about the features' impact on predicting the target variable.

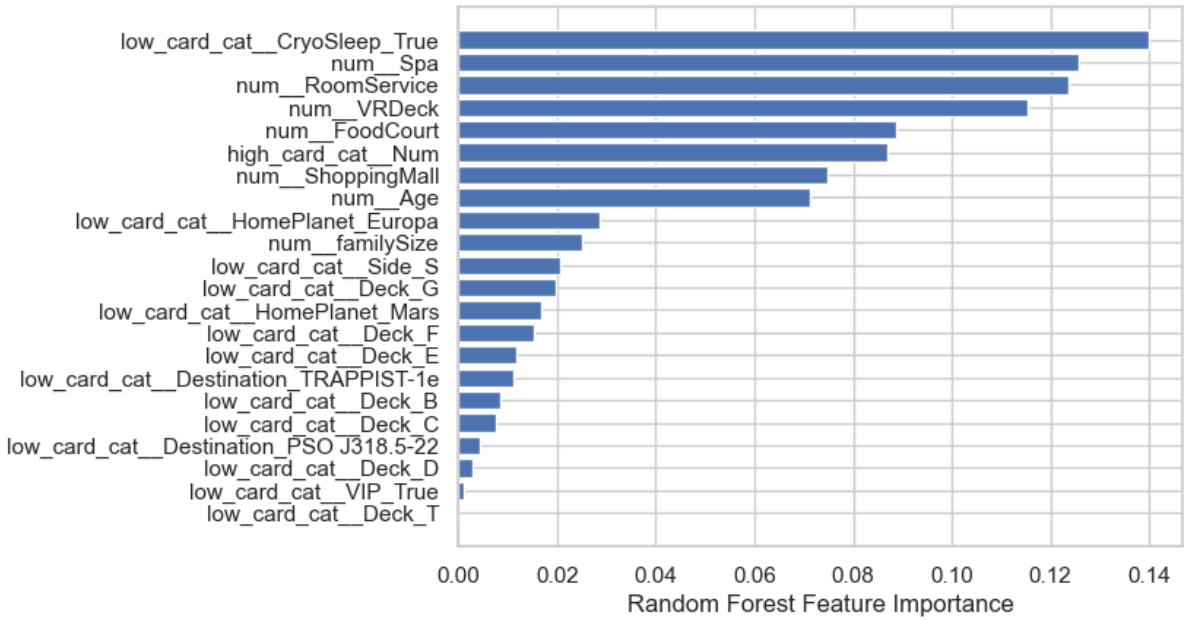


Figure 26. Random Forest main features

- CryoSleep_True (13.98%): Being in a cryosleep state is the most influential factor in the model. This might imply that passengers in cryosleep have distinctive characteristics or outcomes compared to those who are not, making it a crucial factor in predictions.
- Spending Habits (Approximately 45.54% Combined): Features related to spending habits such as Spa (12.58%), RoomService (12.37%), VRDeck (11.55%), FoodCourt (8.86%), and ShoppingMall (7.47%) collectively make up a significant portion of the model's predictive power. This highlights that a passenger's expenditure on various amenities aboard is a vital indicator of their behavior or status, making it a key component in predictions.
- Age (7.13%) and Family Size (2.50%): The age of passengers plays a notable role, and it is imperative to understand how different age groups might influence the prediction. Family size also impacts predictions, though its influence is lesser compared to age.
- High Cardinality Feature - Num (8.69%): This feature is among the top influential features. Its interpretation depends on its context. If 'Num' is an identifier, its importance might be misleading and could indicate a potential overfit. If it carries meaningful information, it warrants further investigation.
- Location and Accommodation:
    - HomePlanet and Deck Categories: Features related to a passenger's origin (HomePlanet_Europa: 2.88%, HomePlanet_Mars: 1.69%) and their accommodation (Deck_G: 1.98%, Deck_F: 1.53%, etc.) show varying

importance. This indicates that the model picks up on specific patterns related to these categories, but they are less influential compared to other features.
    - Side of the Ship (Side_S: 2.06%): This feature has a moderate influence, indicating that the side of the ship where a passenger is located does play a role in predictions, albeit not as significant as other features.
- Destination and VIP Status:
    - Destination (Destination_TRAPPIST-1e: 1.11%, Destination_PSO J318.5-22: 0.44%): Different destinations have a minor impact on the model's predictions.
    - VIP Status (VIP_True: 0.12%): Being a VIP seems to have a very low impact on the model's predictions, suggesting that VIP status does not significantly differentiate passengers in the context of the prediction task.
- Deck_T (0.00%): The 'Deck_T' feature shows no importance in the model, implying that it does not contribute to the model's decision-making process.

In conclusion, spending habits, age, cryosleep status, and certain aspects of location and accommodation play significant roles in the model's predictions. On the other hand, VIP status, specific destination categories, and the 'Deck_T' category have negligible impacts. These insights could guide further feature engineering, data collection, and model refinement efforts to improve prediction performance.

Random Forest

While the visualized decision tree provides an in-depth insight into the decision-making process of a single model, it is crucial to recognize that Random Forest operates as an ensemble algorithm. This means it constructs multiple decision trees during training and merges them to produce more accurate and stable predictions than any individual tree could achieve.

In the context of Random Forest, each tree in the ensemble is trained on a random subset of the data and makes its predictions independently. The final prediction of the Random Forest model is then derived by aggregating the predictions from all the individual trees. This aggregation could be in the form of majority voting for classification tasks or averaging for regression.

Given this ensemble nature of Random Forest, the interpretability of the model becomes a complex issue. While we can delve into a single tree to understand its decision paths and the importance of various features, this tree does not represent the entirety of the Random Forest model. Each tree in the ensemble contributes a piece to the overall prediction, and the importance of features can vary across different trees.

Therefore, while the decision tree visualization is a valuable tool for understanding the decision-making process at a granular level, it does not capture the comprehensive picture

presented by the entire Random Forest model. The ensemble approach inherently trades off interpretability for predictive accuracy and robustness, as it mitigates the risk of overfitting and ensures more reliable predictions across diverse sets of data.

To address the challenge of interpretability in Random Forest and other ensemble methods, techniques such as feature importance scores and permutation importance have been developed. These techniques aggregate information across all the trees in the ensemble to provide a holistic view of feature importance, helping to identify the most significant drivers of the model's predictions.

In summary, while individual decision trees offer valuable insights, it is essential to consider the ensemble nature of Random Forest in drawing conclusions about the model's behavior. Emphasizing aggregated metrics of feature importance and acknowledging the trade-offs between interpretability and predictive performance is crucial in providing a balanced and comprehensive data analysis report.

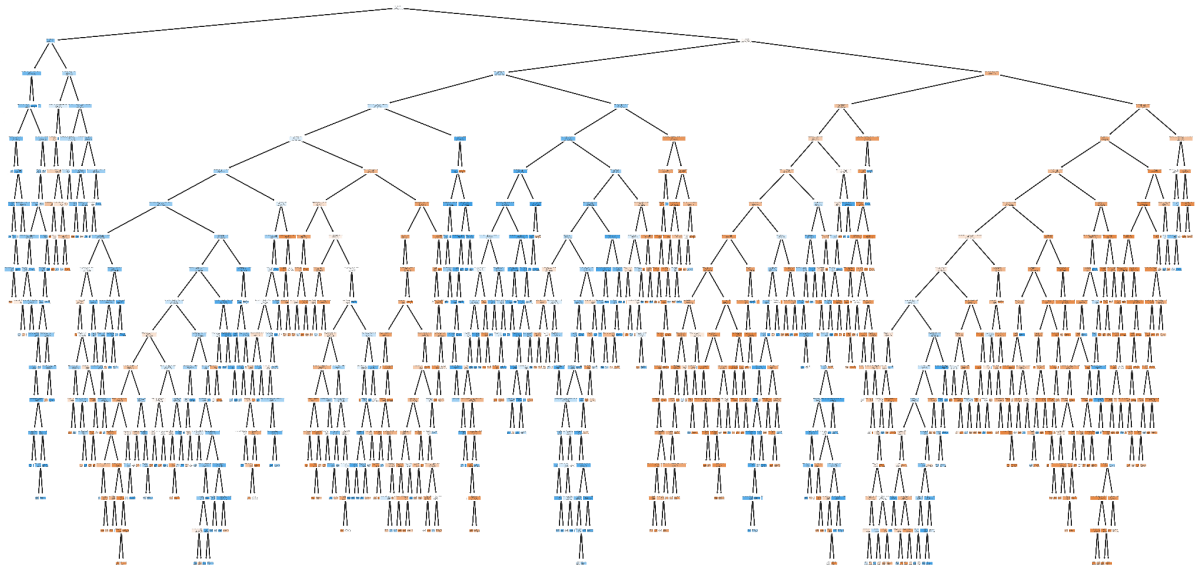


Figure 27. Decision Tree Map

Logistic Regression

Leveraging the coefficients from the logistic regression model provides a nuanced perspective on the relationships between features and the target variable, offering valuable insights to complement those gained from the Random Forest model.

In a logistic regression model, the coefficients represent the change in the log odds of the dependent variable for a one-unit change in the predictor variable, with all other variables held constant. Positive coefficients signify a positive relationship with the target variable, whereas negative coefficients indicate a negative relationship.

	Feature	Coefficient
0	low_card_cat__CryoSleep_True	0.985207
1	low_card_cat__HomePlanet_Europa	0.793656
2	low_card_cat__Deck_C	0.393203
3	low_card_cat__Side_S	0.369577
4	low_card_cat__Deck_G	-0.348791
5	low_card_cat__Deck_B	0.322104
6	low_card_cat__HomePlanet_Mars	0.292200
7	low_card_cat__Deck_E	-0.252825
8	low_card_cat__Destination_TRAPPIST-1e	-0.178719
9	low_card_cat__Destination_PSO J318.5-22	-0.149812
10	Intercept	0.104196
11	low_card_cat__Deck_D	0.079628
12	low_card_cat__Deck_F	-0.061475
13	num__familySize	0.049121
14	low_card_cat__VIP_True	-0.016071
15	num__Age	-0.004217
16	low_card_cat__Deck_T	-0.003223
17	num__Spa	-0.001956
18	num__VRDeck	-0.001716
19	num__RoomService	-0.001608
20	num__FoodCourt	0.000583
21	num__ShoppingMall	0.000385
22	high_card_cat__Num	0.000097

Figure 28. Logistic Regression coefficients (descending based on absoluate value)

Focusing on the most impactful features, `low_card_cat__CryoSleep_True` and `low_card_cat__HomePlanet_Europa` exhibit the highest positive coefficients. This suggests a strong association between being in CryoSleep and originating from Europa with higher odds of achieving the target outcome. This could be indicative of a particular socioeconomic status or access to resources that positively influences the target variable.

On the flip side, features like `low_card_cat__Deck_G`, `low_card_cat__Destination_PSO J318.5-22`, and `low_card_cat__Destination_TRAPPIST-1e` demonstrate negative coefficients, aligning these characteristics with lower odds of the target outcome. This could highlight disparities in conditions or amenities associated with these specific decks or destinations.

Notably, `num__Age` carries a slight negative coefficient, hinting that an increase in age marginally reduces the odds of the target outcome. However, this effect is relatively small, indicating that age may not play a significant role in this context.

The coefficients associated with spending habits aboard (e.g., `num__Spa`, `num__VRDeck`, `num__RoomService`, `num__FoodCourt`, `num__ShoppingMall`) are minimal, suggesting these variables don't heavily influence the target outcome in this logistic regression model. This observation stands in contrast to the Random Forest model, which identified some of these features as influential.

While logistic regression offers a more straightforward interpretation of feature importance due to its linear nature, it’s important to acknowledge its limitations in capturing complex non-linear relationships and interactions between features. Nevertheless, when used alongside ensemble methods like Random Forest, it provides valuable, complementary insights for a more comprehensive understanding of the factors influencing the target variable.

In sum, the logistic regression coefficients have identified several key features with strong associations to the target outcome, delivering clear and interpretable insights. However, it's crucial to consider these findings within the broader analysis context, recognizing logistic regression’s potential limitations and the importance of drawing from multiple models for a holistic understanding.

Conclusion

In conclusion, we found that we could predict with a relatively high accuracy, around 80%, on whether the passengers were transported. We were not able to help our modeling with dimensional reduction. We also were not able to find any valuable insights with our unsupervised learning.

We found that combining multiple models was beneficial in finding important features. The Random Forest model produced slightly better results than Logistic Regression but was less explainable. The two models shared common significant features such as Cryo Sleep, being from Europa, Decks B, C, and G, and starboard side rooms.

We can see in the data that about 3,000 passengers who were transported had opted for CryoSleep. The remaining passengers may have been guests on decks B, C and G, who had habits of enjoying the ship's amenities, being out and about and more vulnerable to a space dust anomaly.

We were able to play around with different variables comparison methods, different models, and different parameters to achieve a relatively successful predictor. Unfortunately, we believe that the dimension reduction and explorability was a bit limited due to the fact it was a synthetic data set and did not contain natural randomness.

Either way, we believe we would need to increase the model score to employ it in any practical use. We learned that in a case like this, where you are attempting to save lives, interpretability can be as significant as predictability. We would want to be able to use this model to help others in similar situations and would want to be able to find any potential faults in a system. Because as we see here, history repeats itself, and we want to be prepared whether it happens with icebergs or space clouds.