AI 511: Machine Learning October 21, 2023

AI 511: Assignment 1

Goutham U R (IMT2021045)
Nimish Gaurish Sinai Khandeparkar (IMT2021077)
Team Name: 3 farthest neighbours Pannaga Bhat (IMT2021080)

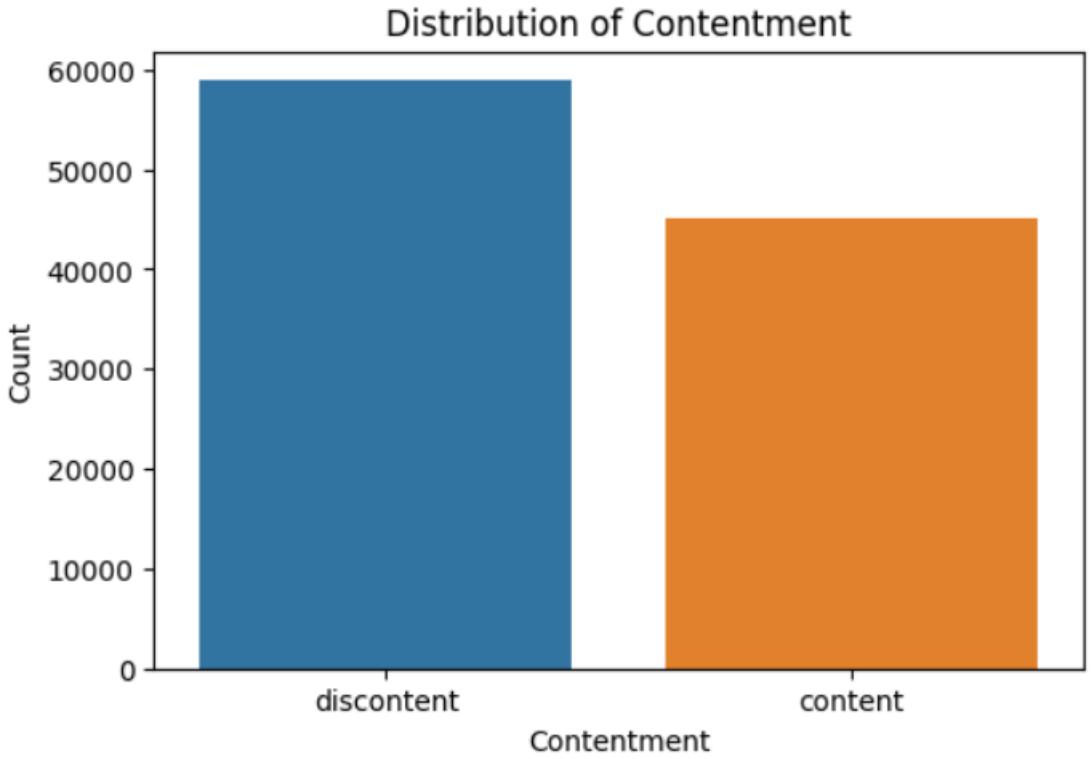
Preprocessing

- The 'Arrival Delay in Minutes' column included 310 NULL values out of 103904 rows. The mean of the rest of the values was calculated and the NULL values were replaced with the mean.
- The 'Gender', 'Traveler Type', 'Type of Travel' and 'Class' columns were categorical string data. The `pandas.get_dummies()` function was used to convert these columns to one Boolean column for each unique string value.

EDA

Contentment

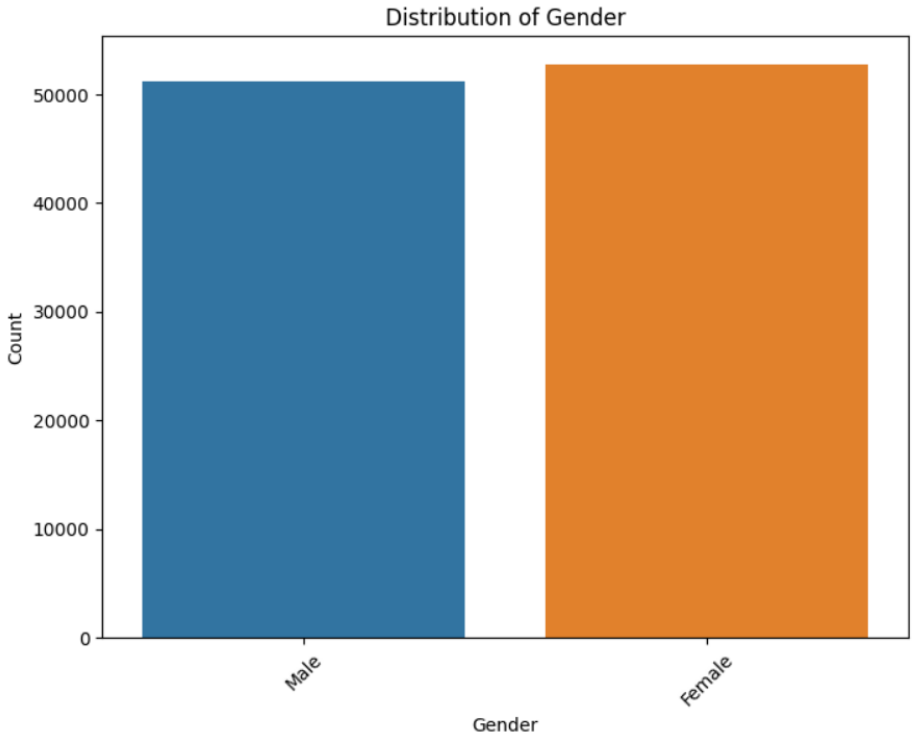
Contentment is the final value being predicted. It has 2 unique values: content and discontent.



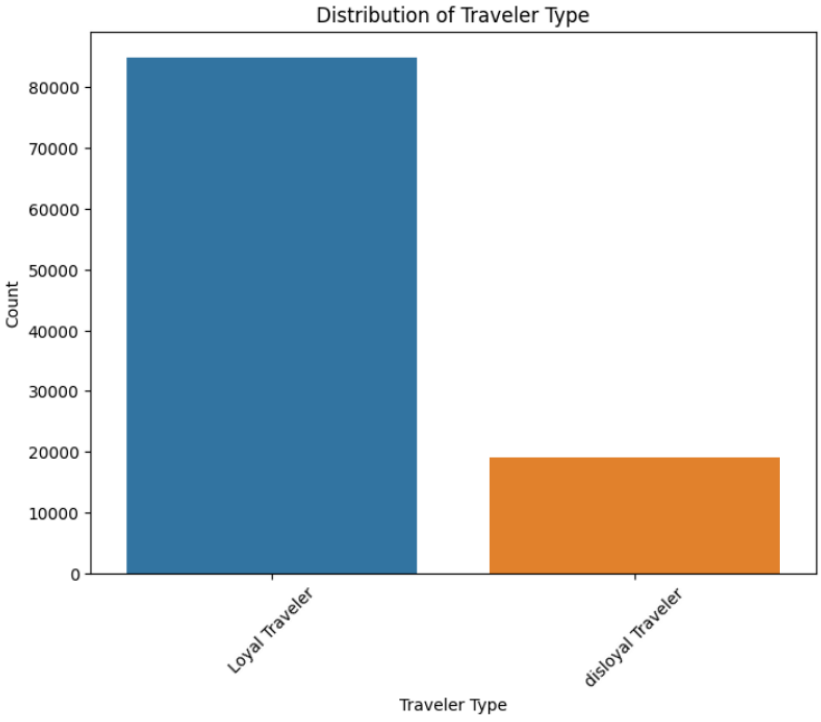
Graph showing distribution of contentment values

Categorical Data

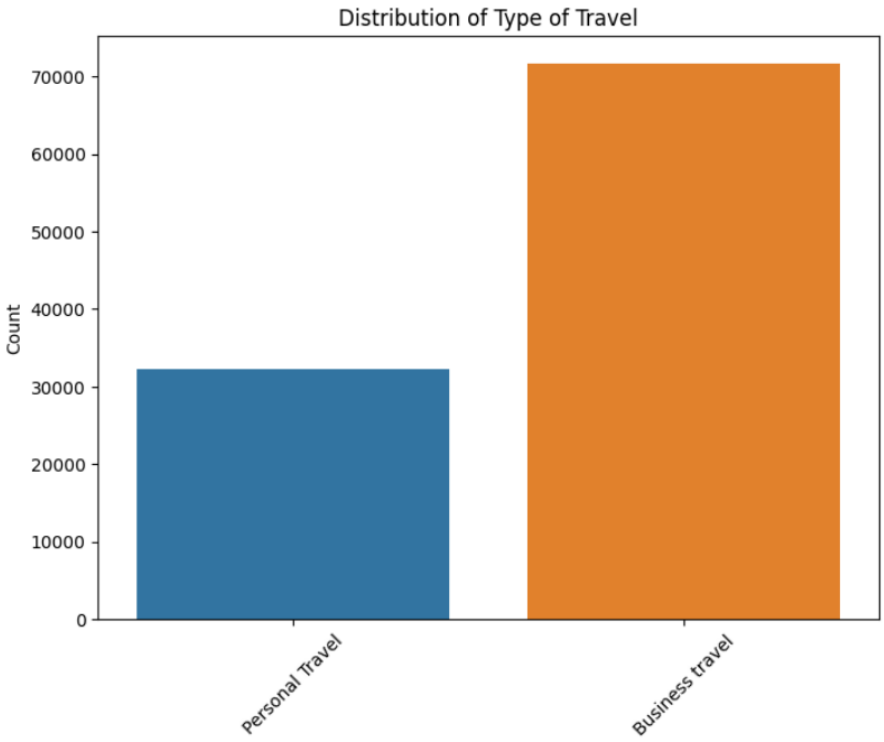
Categorical data includes the columns ‘Gender’, ‘Traveler Type’, ‘Type of Travel’ and ‘Class’.



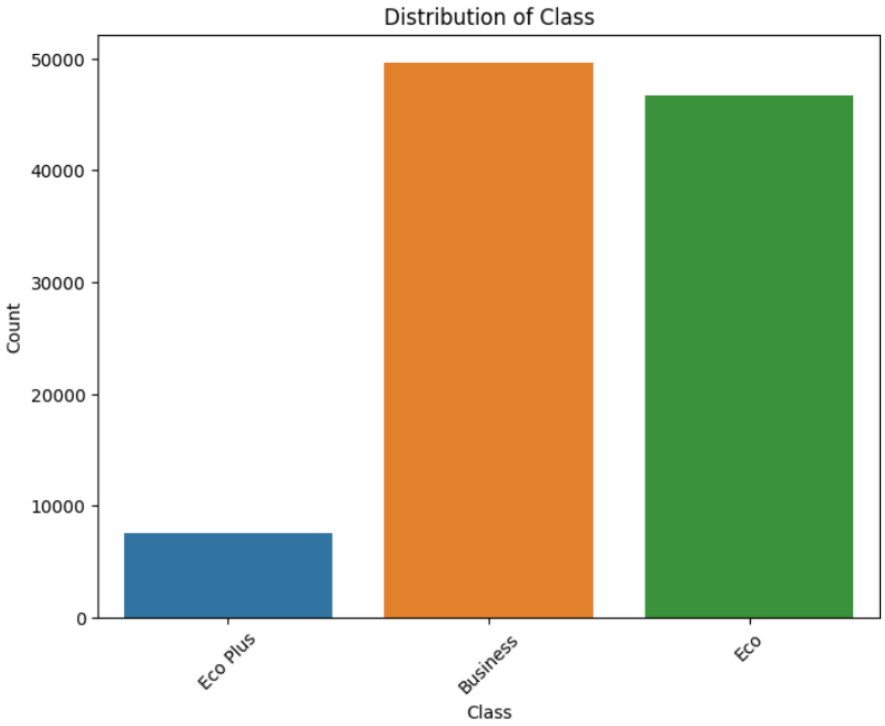
Distribution of values in ‘gender’ column



Distribution of values in ‘traveler type’ column



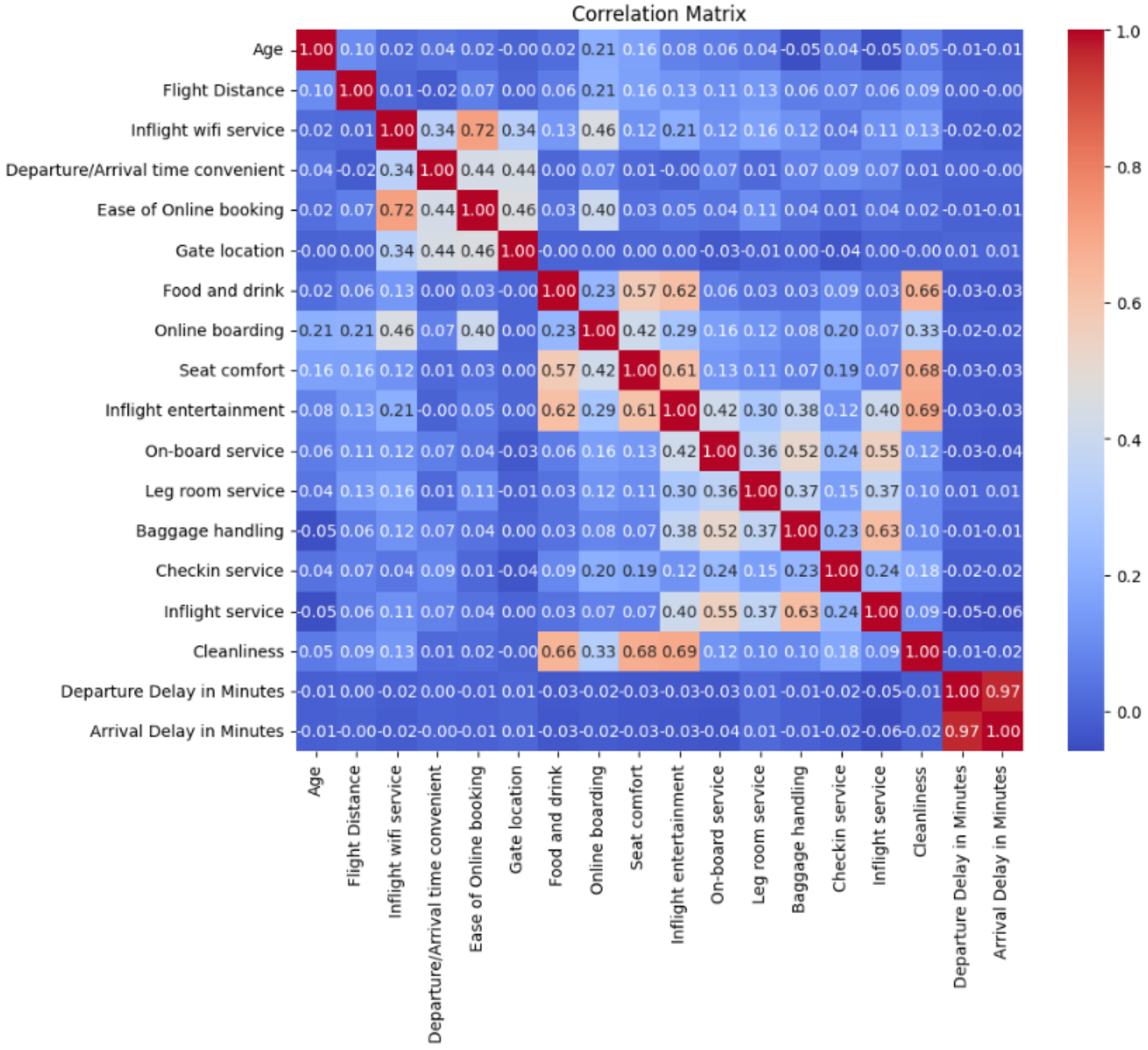
Distribution of values in 'type of travel' column



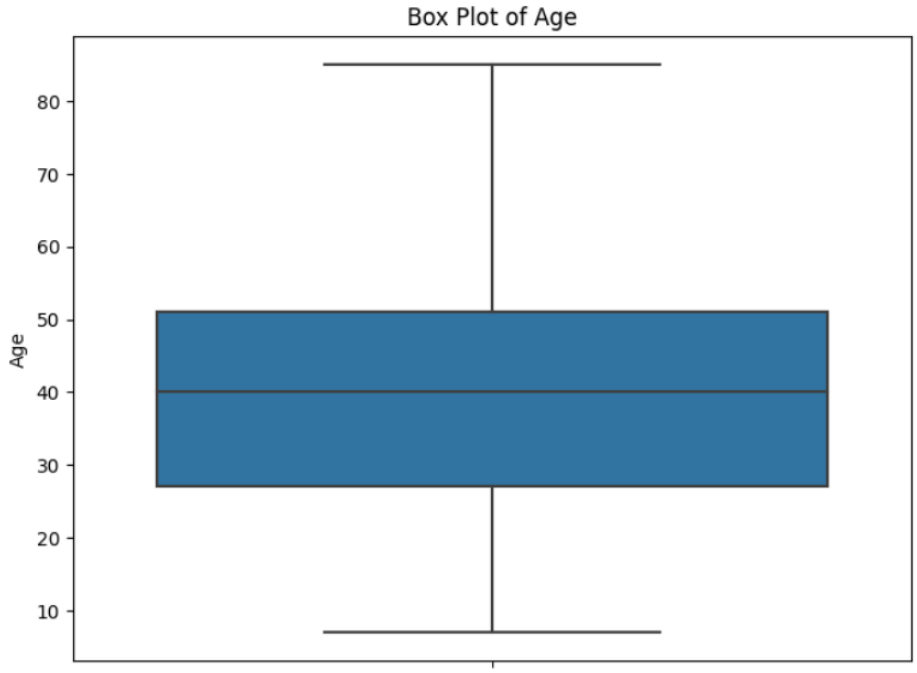
Distribution of values in 'class' column

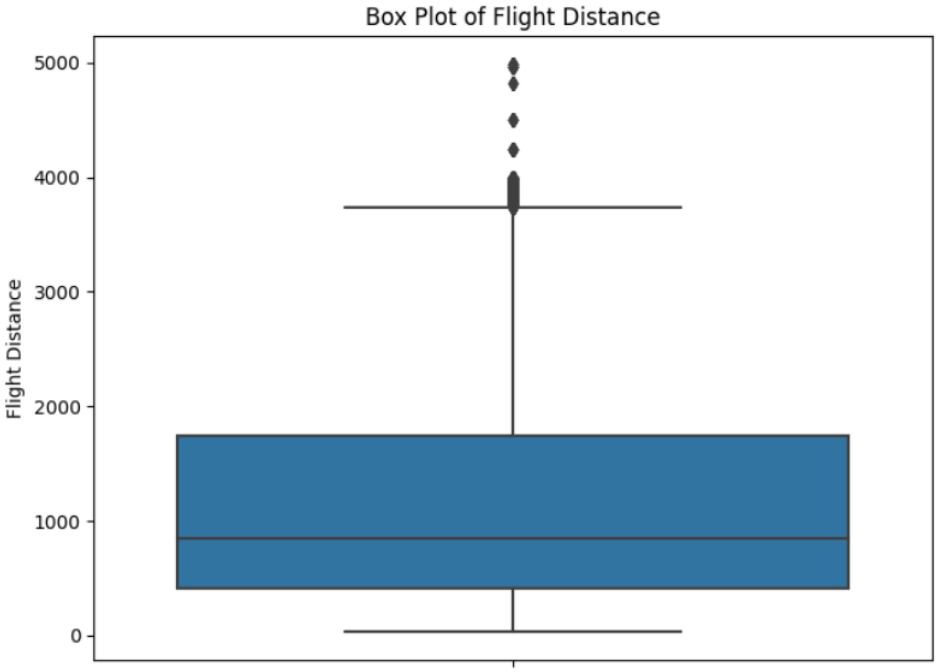
Numerical data

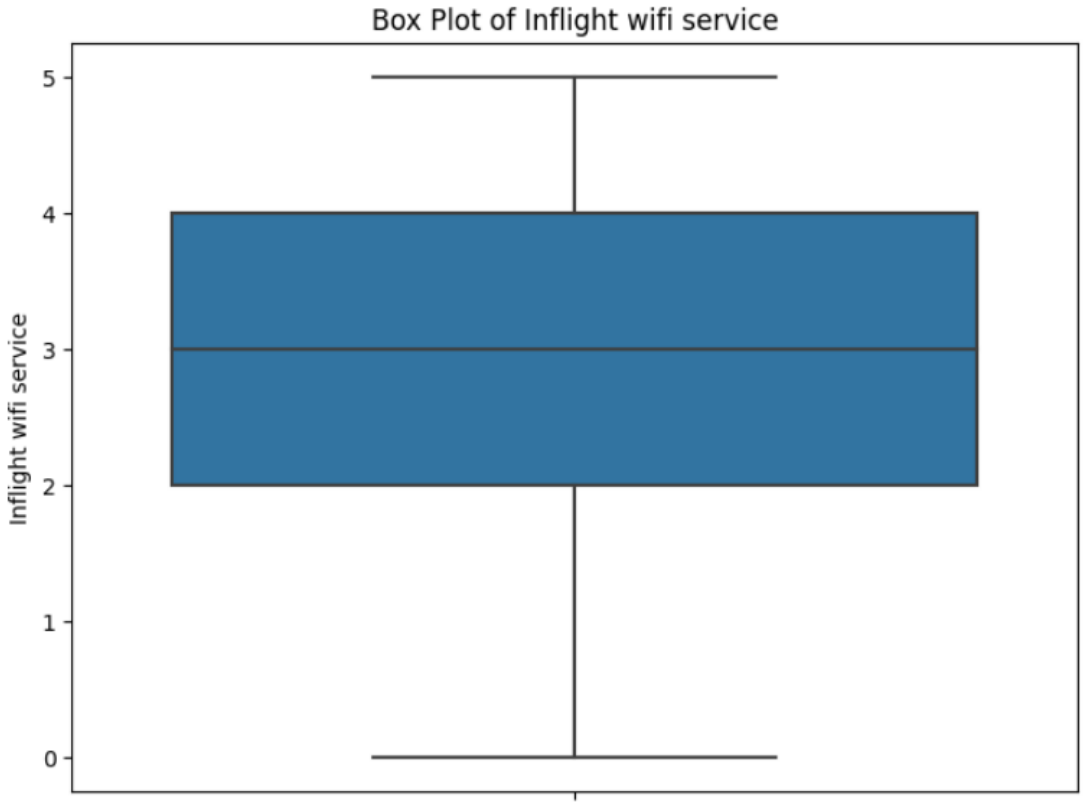
The numerical columns in the data are ‘Age’, ‘Flight distance’, ‘Inflight wifi service’, ‘Departure/Arrival time convenient’, ‘Ease of Online booking’, ‘Gate location’, ‘Food and drink’, ‘Online boarding’, ‘Seat comfort’, ‘Inflight entertainment’, ’On-board service‘, ‘Leg room service’, ‘Baggage handling’, ‘Check-in Service’, ‘Cleanliness’, ‘Arrival Delay in Minutes’ and ‘Departure delay in Minutes’. The final 2 columns are shown with a histogram. The rest are shown with a box plot.

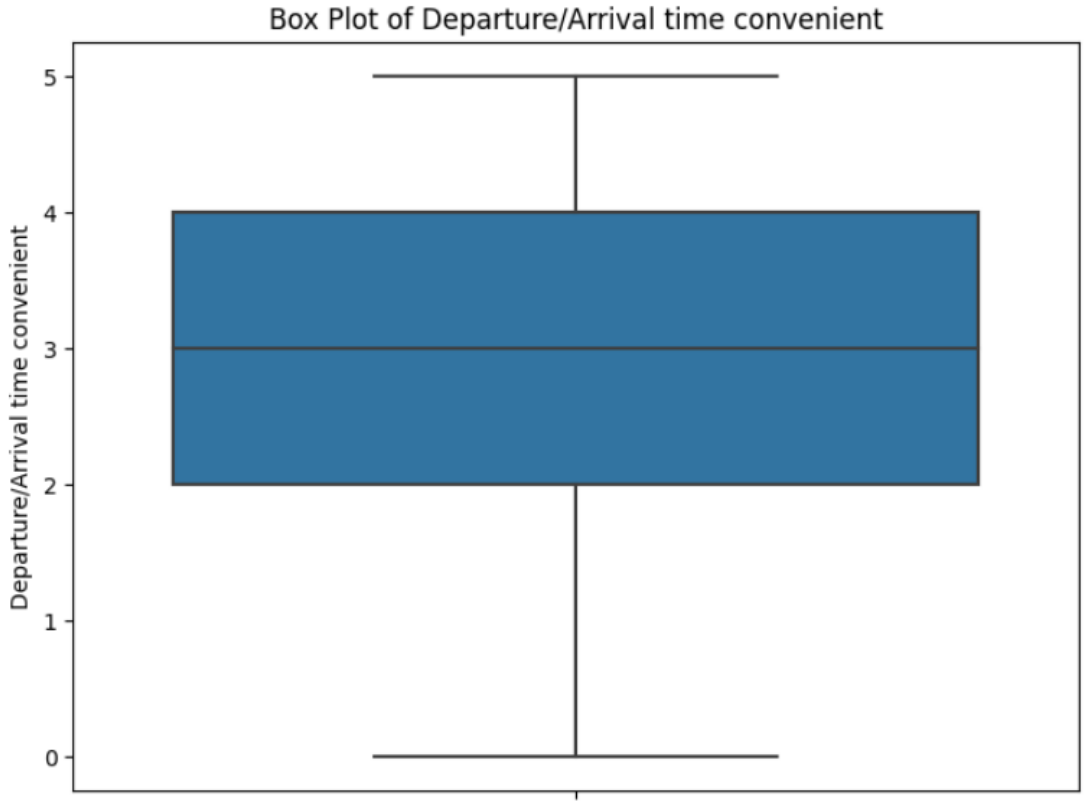


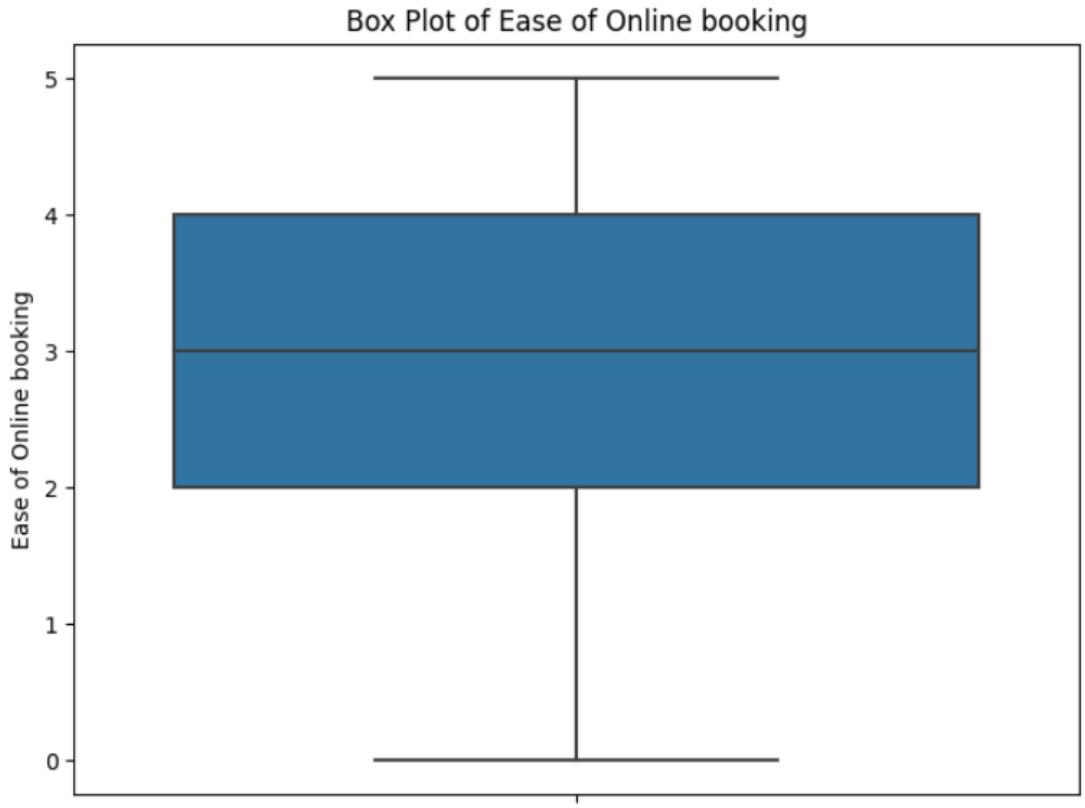
Heatmap of correlation matrix of the data

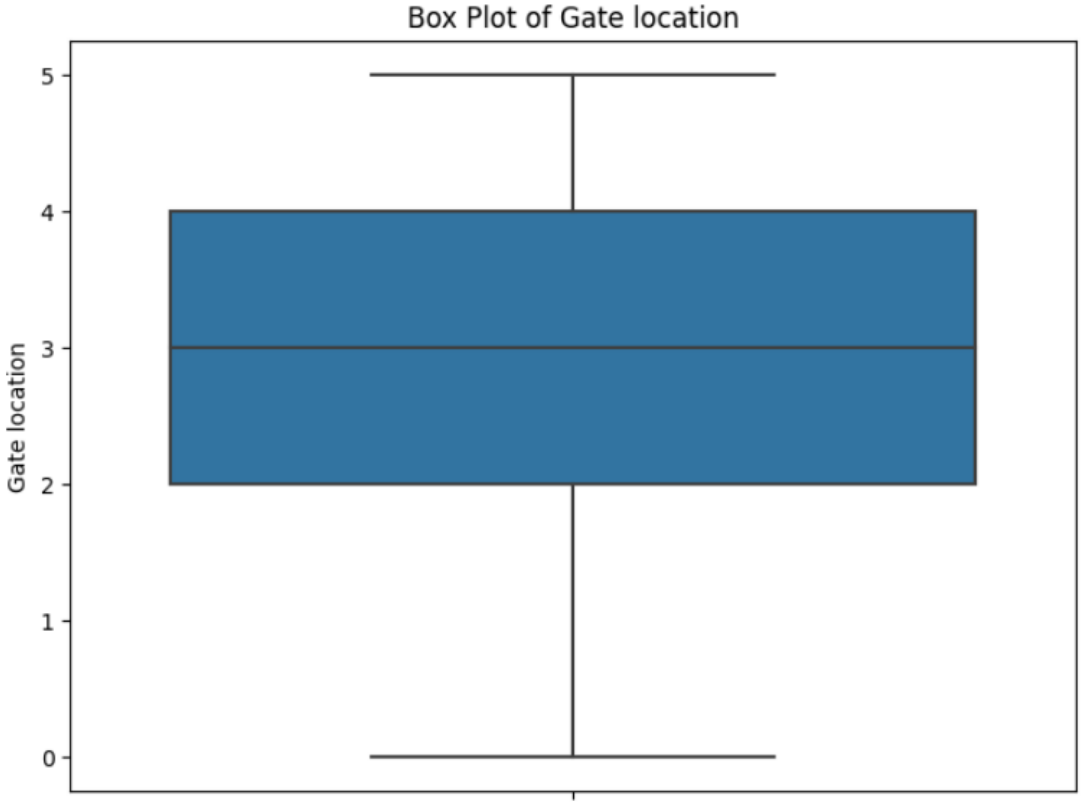

Box Plot of Age
Age
80
70
60
50
40
30
20
10

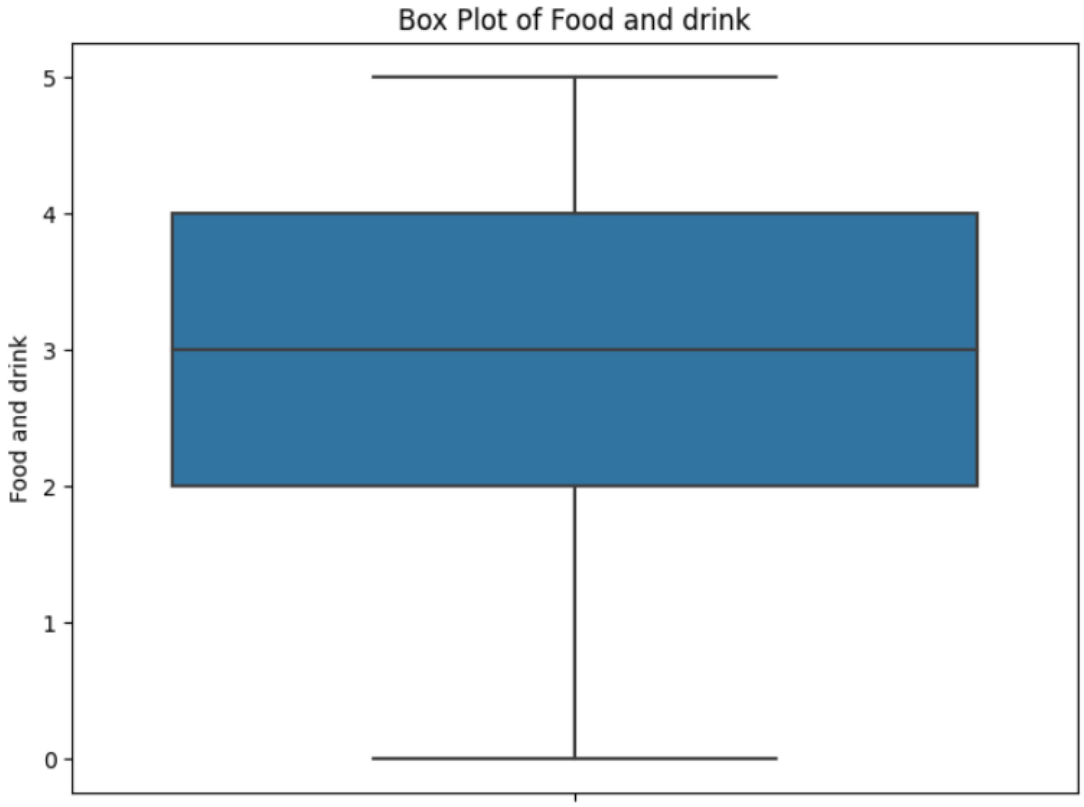

Box Plot of Flight Distance
Flight Distance
5000
4000
3000
2000
1000
0

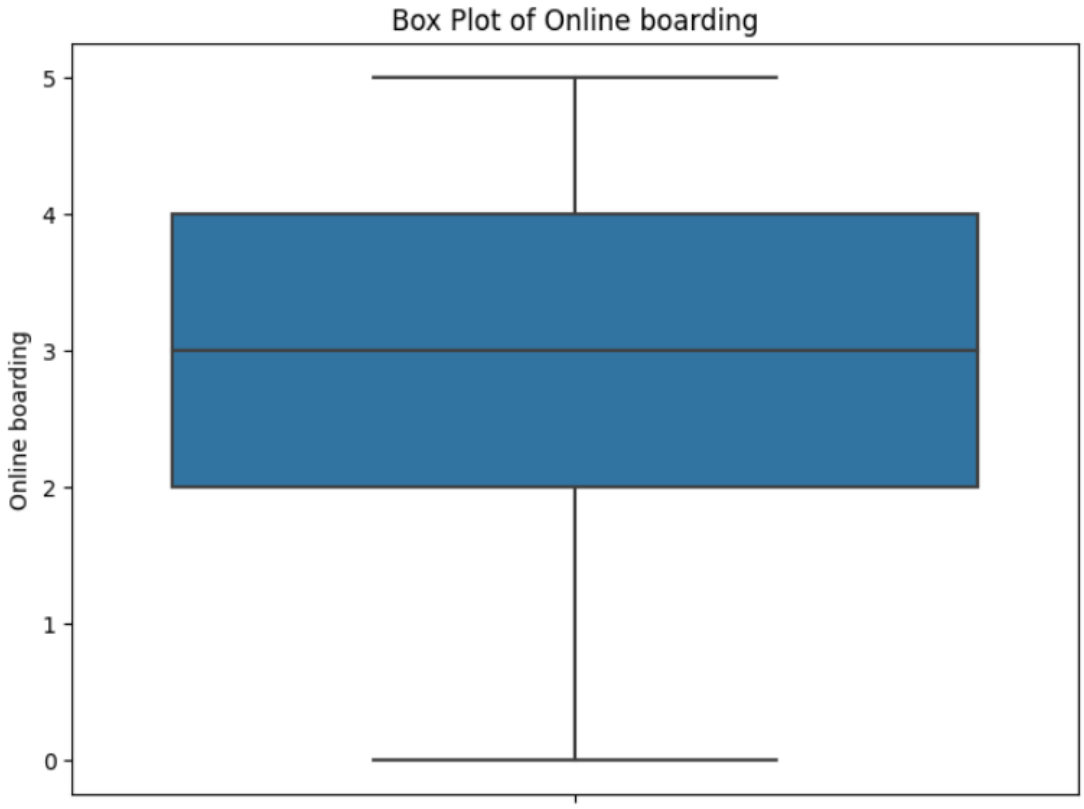

Box Plot of Inflight wifi service
Inflight wifi service
5
4
3
2
1
0

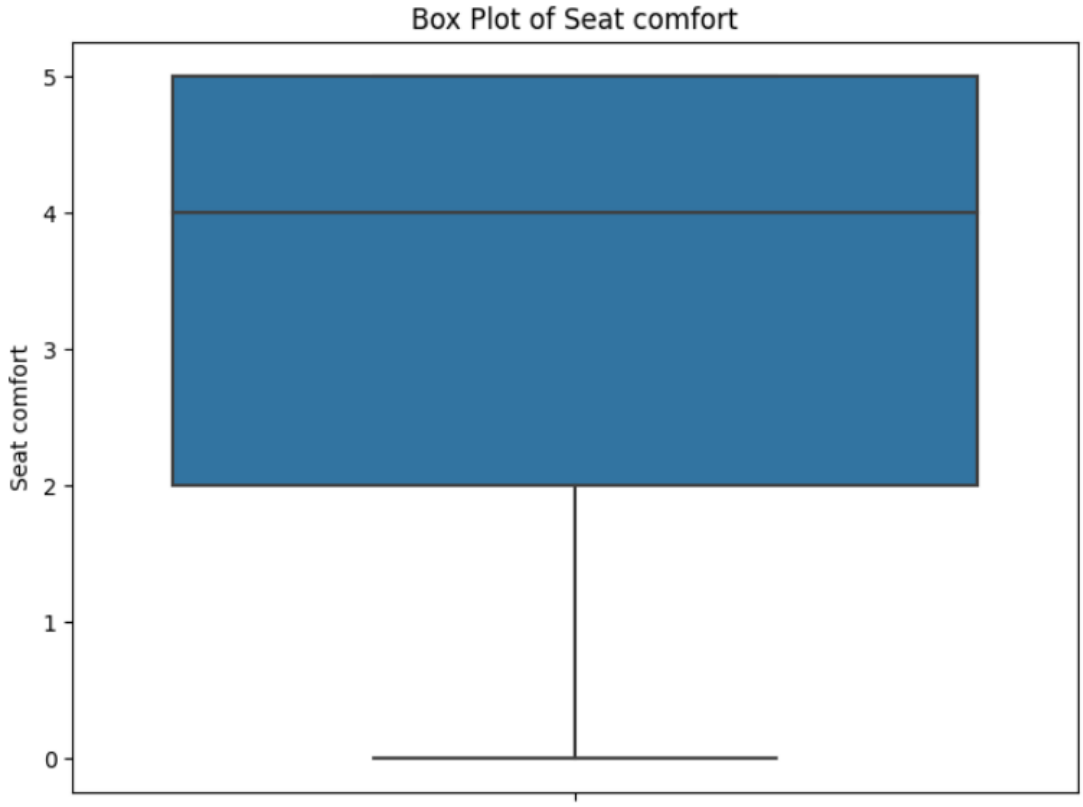

Box Plot of Departure/Arrival time convenient
Departure/Arrival time convenient
5
4
3
2
1
0

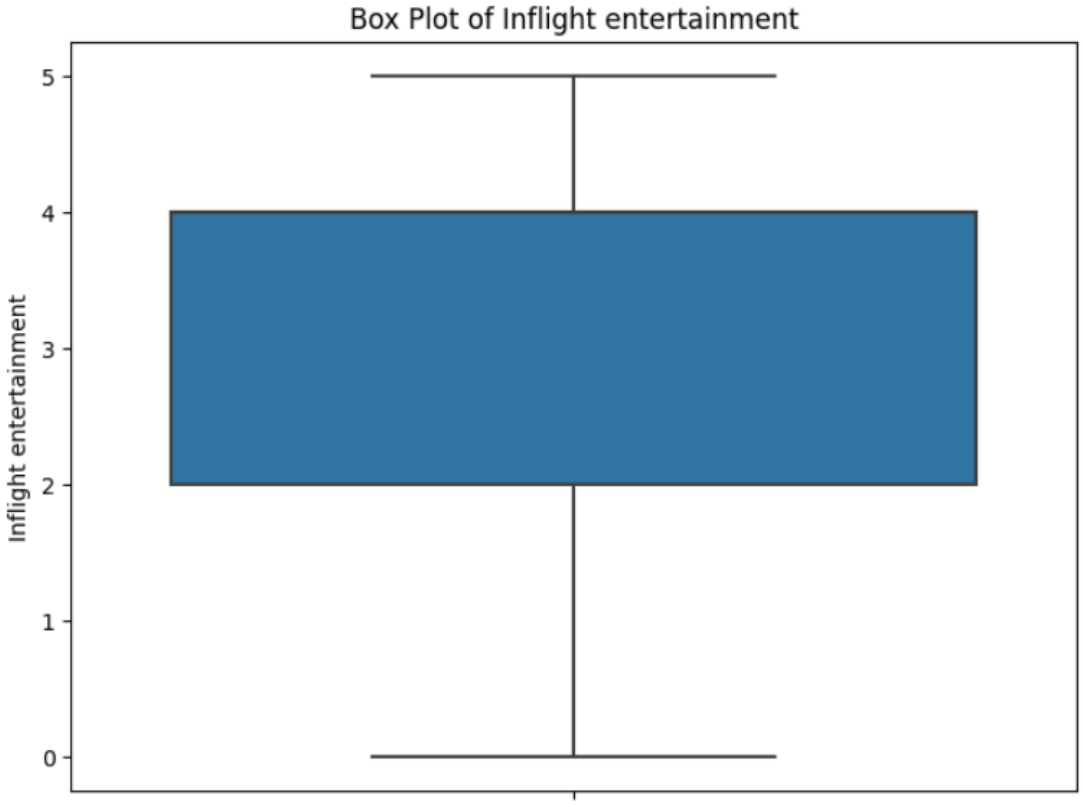

Box Plot of Ease of Online booking
Ease of Online booking
5
4
3
2
1
0

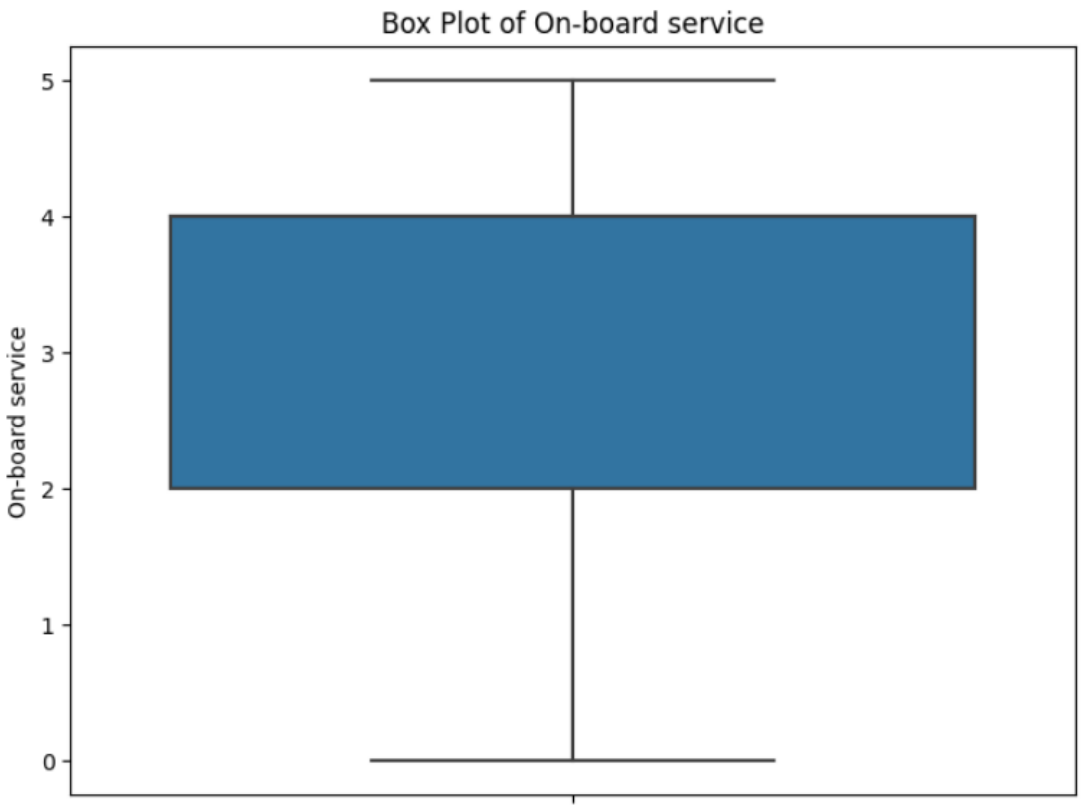

Box Plot of Gate location
Gate location
5
4
3
2
1
0

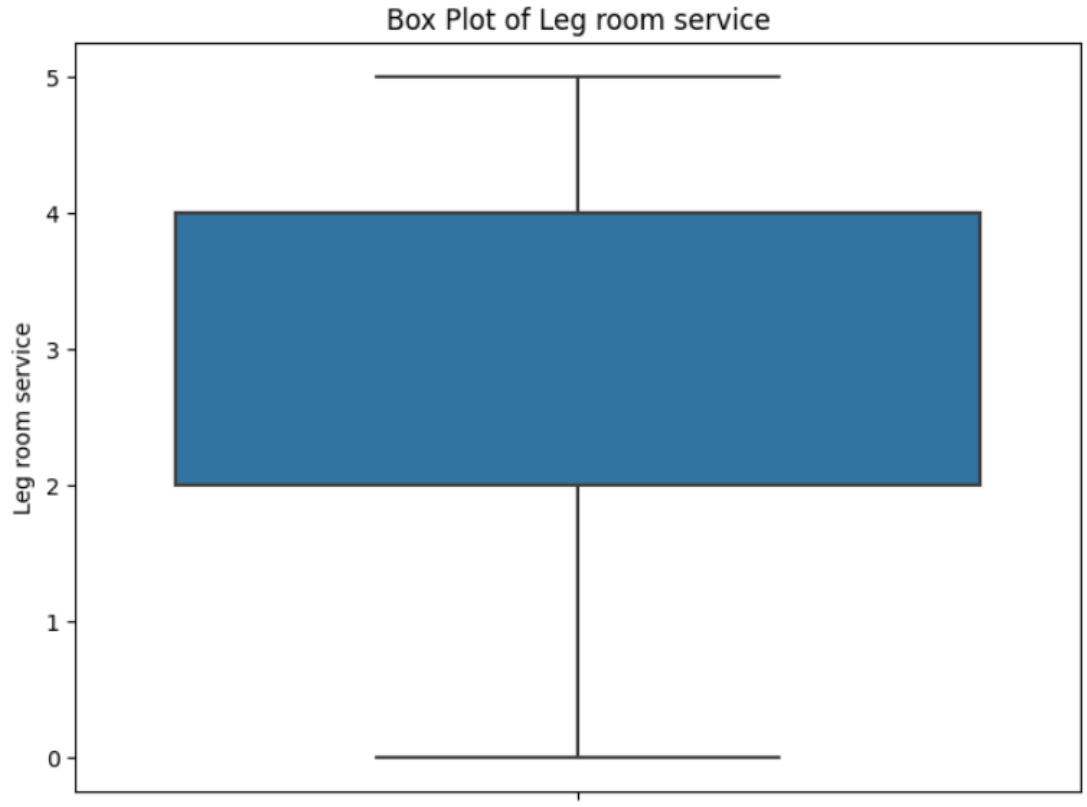

Box Plot of Food and drink
Food and drink
5
4
3
2
1
0

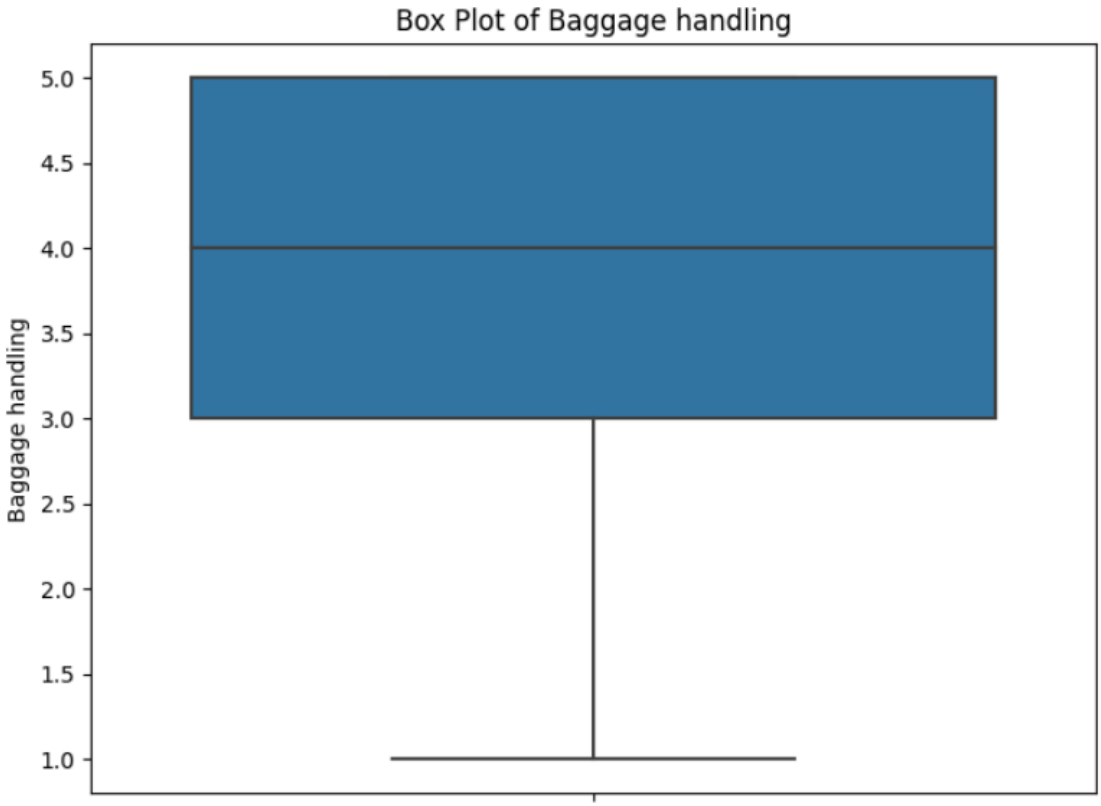

Box Plot of Online boarding
Online boarding
5
4
3
2
1
0

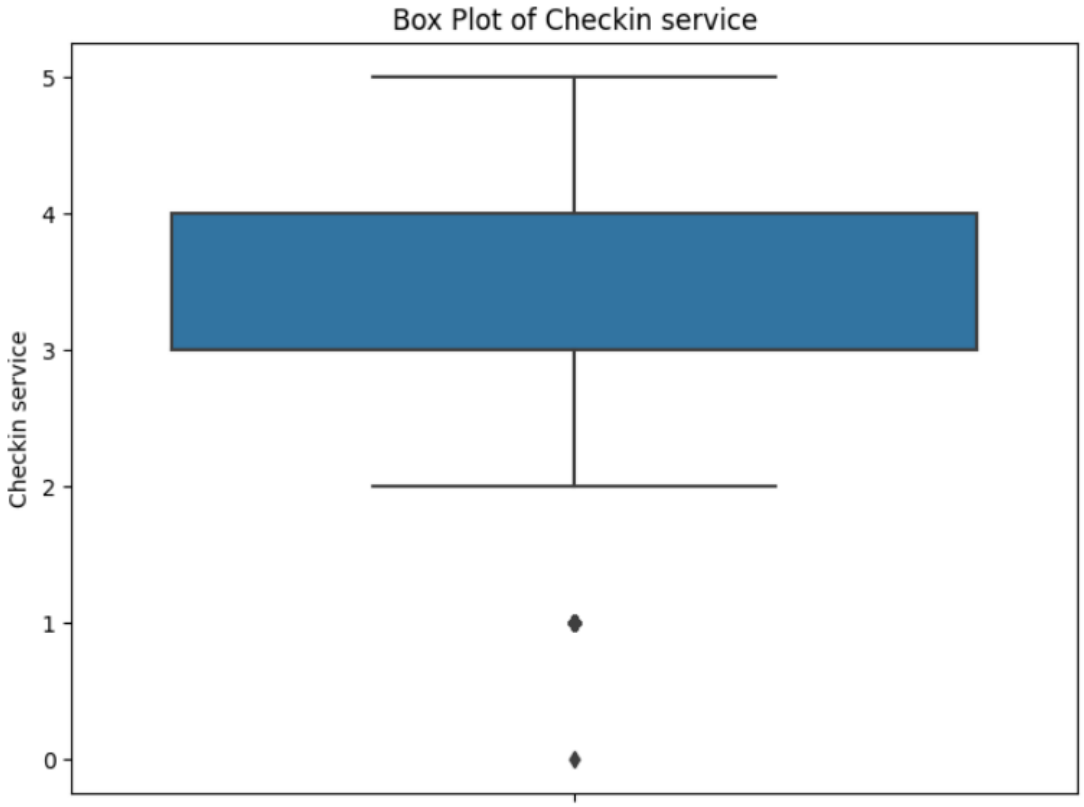

Box Plot of Seat comfort
Seat comfort
5
4
3
2
1
0

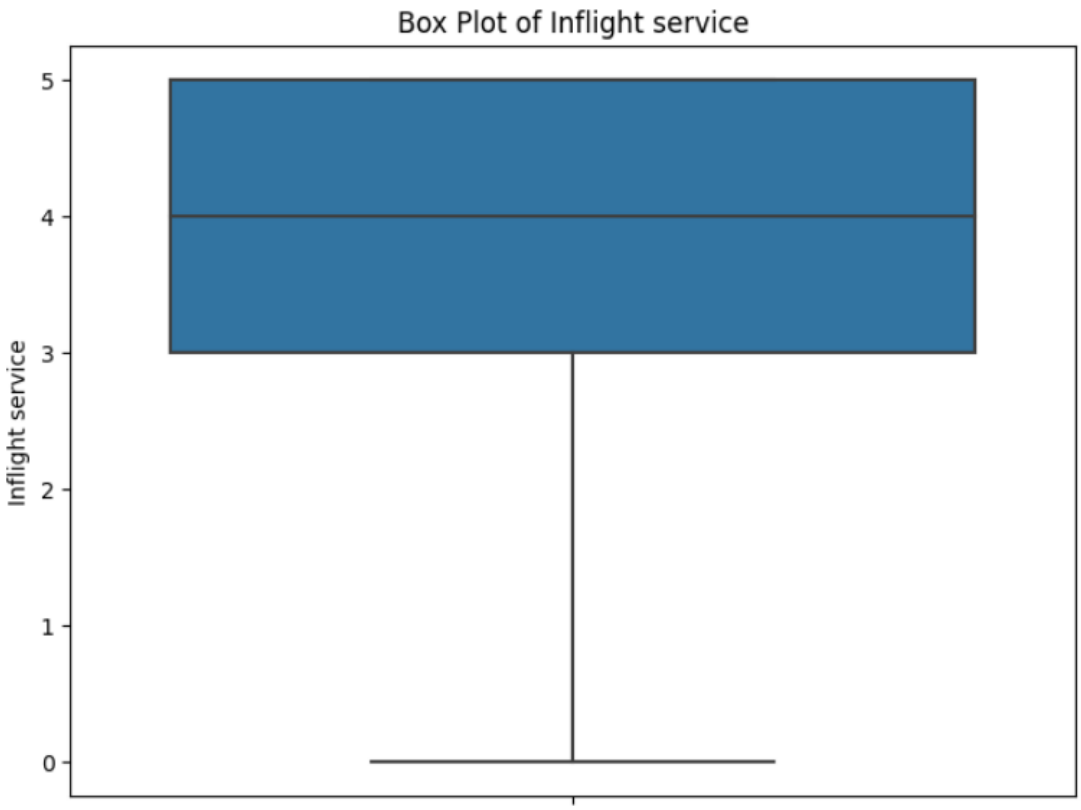

Box Plot of Inflight entertainment
Inflight entertainment
5
4
3
2
1
0

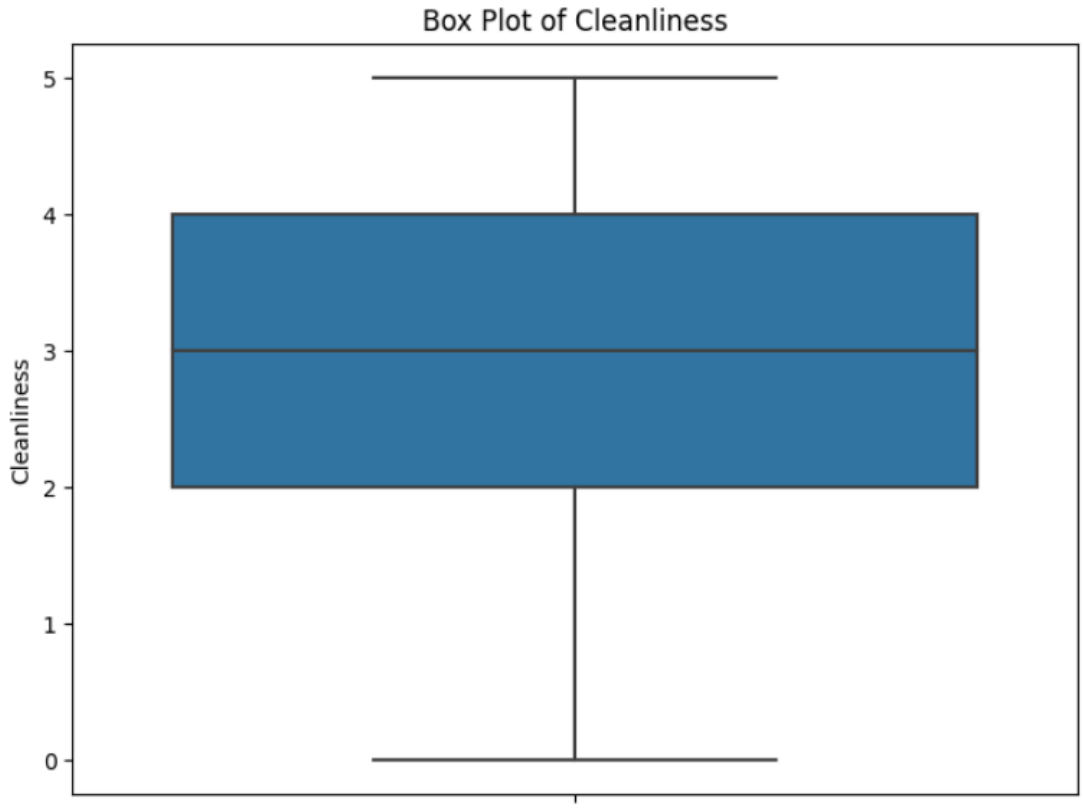

Box Plot of On-board service
On-board service
5
4
3
2
1
0

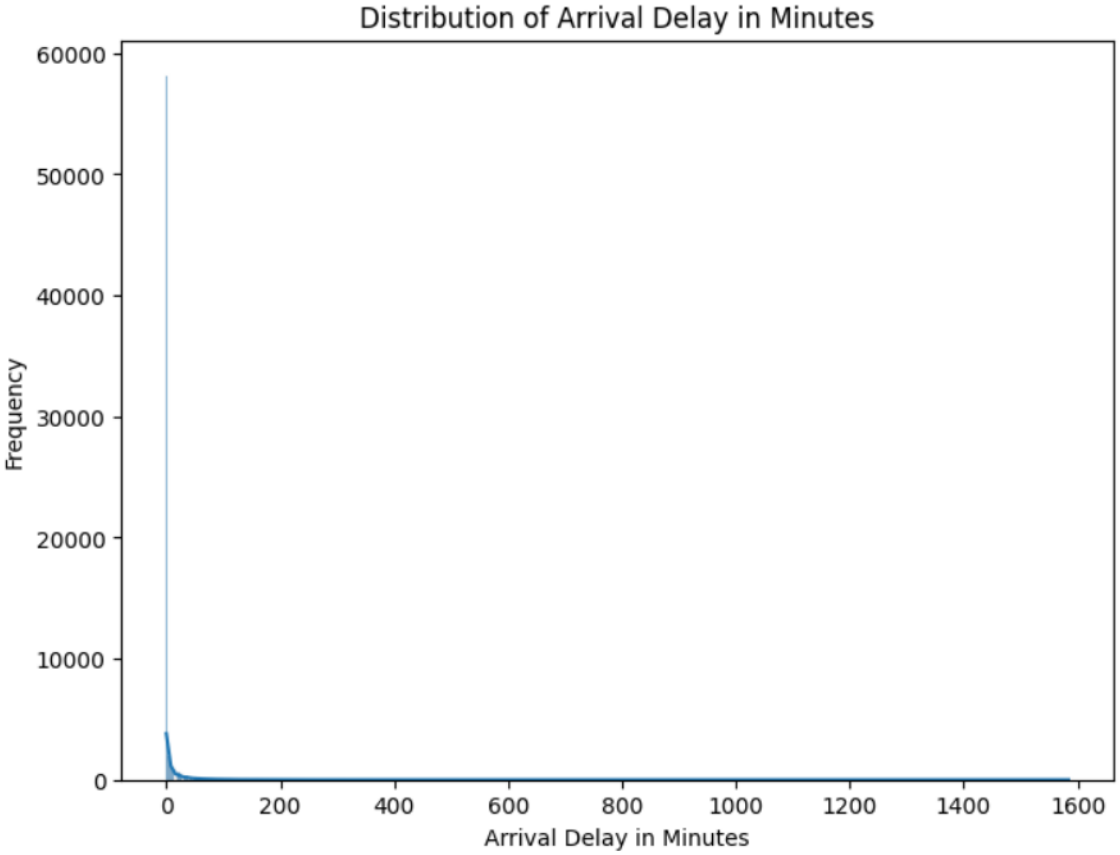

Box Plot of Leg room service
Leg room service
5
4
3
2
1
0


Box Plot of Baggage handling
Baggage handling
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0

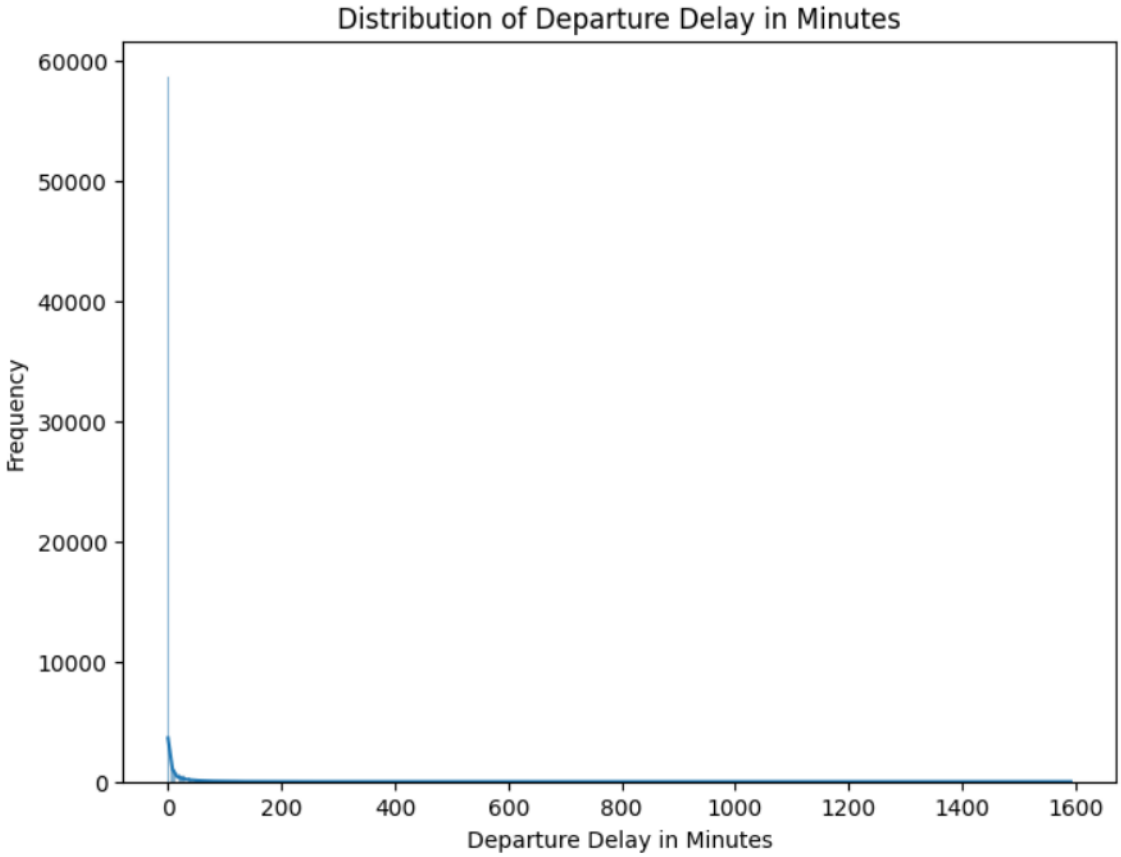

Box Plot of Checkin service
Checkin service
5
4
3
2
1
0


Box Plot of Inflight service
Inflight service
5
4
3
2
1
0


Box Plot of Cleanliness
Cleanliness
5
4
3
2
1
0



Histogram of 'arrival delay in minutes column'



Histogram of 'departure delay in minutes column'

Models

1. **K Closest Neighbours**: Model trained with k value of 3. Provides an accuracy score for test-train split of **0.747**.

```
clf = clf = KNeighborsClassifier(n_neighbors=3)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_val)
accuracy = accuracy_score(y_val, y_pred)
print("Accuracy on the validation set:", accuracy)
```

```
Accuracy on the validation set: 0.7474135027188297
```

2. **Gaussian Naive Bayes**: Provides an accuracy score for test-train split of **0.848**.

```
clf = GaussianNB()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_val)
accuracy = accuracy_score(y_val, y_pred)
print("Accuracy on the validation set:", accuracy)
```

```
Accuracy on the validation set: 0.8476492950291131
```

3. **Logistic Regression**: Provides an accuracy score of **0.876** for a test-train split. Also throws an error after reaching maximum number of iterations.

```
clf = LogisticRegression(random_state=42, max_iter=1000)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_val)
accuracy = accuracy_score(y_val, y_pred)
print("Accuracy on the validation set:", accuracy)
```

```
Accuracy on the validation set: 0.8764255810596218
```

4. **Decision Tree**: Seeded with value of 42. Provides accuracy score for test-train split of **0.947**.

```
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_val)
accuracy = accuracy_score(y_val, y_pred)
print("Accuracy on the validation set:", accuracy)
```

```
Accuracy on the validation set: 0.9466820653481546
```

5. **Extreme Gradient Boost**: Label encoder was used on 'Contentment' column to make sure all data is numerical. Provides accuracy score for test-train split of **0.962**

```
# Encode 'contentment' column to binary labels
label_encoder = LabelEncoder()
y_train_encoded = label_encoder.fit_transform(y_train)
y_val_encoded = label_encoder.transform(y_val)

clf = xgb.XGBClassifier()
clf.fit(X_train, y_train_encoded)
y_pred = clf.predict(X_val)
accuracy = accuracy_score(y_val_encoded, y_pred)
print("Accuracy with XGBoost on the validation set:", accuracy)
```

```
Accuracy with XGBoost on the validation set: 0.9624657138732496
```

6. **Random Forest**: The model is seeded with value of 96. The value of `n_estimators` was checked for 125, 150, 200 and 250. For `n_estimators = 200`, the model gives an accuracy score of **0.963** with the test-train split. **The final submission uses this model.**

```
clf = RandomForestClassifier(n_estimators=200, random_state=96)
clf.fit(X_train, y_train)


# Predict on the validation set
y_pred = clf.predict(X_val)

# Calculate the accuracy score
accuracy = accuracy_score(y_val, y_pred)
print("Accuracy on the validation set:", accuracy)
```

```
Accuracy on the validation set: 0.9631394061883451
```