Analysis of Airline Passenger Satisfaction

Stevens Institute of Technology

Course: CPE 627 - Big Data Analysis

Professor: Yu-Dong Yao

Suyash Bhatt

CWID - 20015443

AGENDA

Introduction

Importance of Services

EDA

Classification Algorithm Used

INTRODUCTION

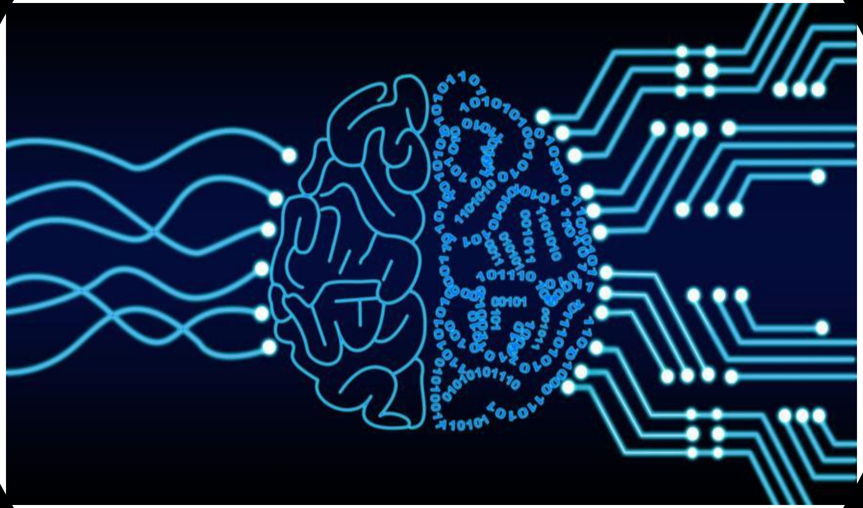
This dataset contains an airline passenger satisfaction survey. What factors are highly correlated to a satisfied (or dissatisfied) passenger?

Kaggle Dataset

GitHub Repository

IMPORTANCE OF CUSTOMER SATISFACTION

1.Loyalty and Repeat Business:
2.Positive Word-of-Mouth:
3.Competitive Advantage:
4.Revenue Generation:
5.Operational Efficiency:
6.Employee Morale and Engagement:



Objective-

- The objective of the project is to perform sentiment analysis on customer satisfaction in commercial aircraft. The goal is to analyze customer feedback and determine their level of satisfaction with airline services. The project aims to build machine learning models that can accurately classify customer satisfaction based on various features and attributes provided in the dataset.

- The project involves the following steps:
- Data Preparation:
- Exploratory Data Analysis:
- Data Preprocessing:
- Model Training and Evaluation:
- Model Comparison:

- The overall objective is to identify the most accurate model that can effectively classify customer satisfaction based on the given dataset. This information can provide valuable insights to airline companies for improving their services and enhancing customer satisfaction.

Train Data

train

File Edit View

```
,id,Gender,Customer Type,Age,Type of Travel,Class,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,Checkin service,Inflight service,Cleanliness,Departure Delay in Minutes,Arrival Delay in Minutes,satisfaction
0,70172,Male,Loyal Customer,13,Personal Travel,Eco Plus,460,3,4,3,1,5,3,5,5,4,3,4,4,5,5,25,18.0,neutral or dissatisfied
1,5047,Male,disloyal Customer,25,Business travel,Business,235,3,2,3,3,1,3,1,1,1,5,3,1,4,1,1,6.0,neutral or dissatisfied
2,110028,Female,Loyal Customer,26,Business travel,Business,1142,2,2,2,2,5,5,5,5,4,3,4,4,4,5,0,0.0,satisfied
3,24026,Female,Loyal Customer,25,Business travel,Business,562,2,5,5,5,2,2,2,2,2,5,3,1,4,2,11,9.0,neutral or dissatisfied
4,119299,Male,Loyal Customer,61,Business travel,Business,214,3,3,3,3,4,5,5,3,3,4,4,3,3,3,0,0.0,satisfied
5,111157,Female,Loyal Customer,26,Personal Travel,Eco,1180,3,4,2,1,1,2,1,1,3,4,4,4,4,1,0,0.0,neutral or dissatisfied
6,82113,Male,Loyal Customer,47,Personal Travel,Eco,1276,2,4,2,3,2,2,2,2,3,3,4,3,5,2,9,23.0,neutral or dissatisfied
7,96462,Female,Loyal Customer,52,Business travel,Business,2035,4,3,4,4,5,5,5,5,5,5,5,4,5,4,4,0.0,satisfied
8,79485,Female,Loyal Customer,41,Business travel,Business,853,1,2,2,2,4,3,3,1,1,2,1,4,1,2,0,0.0,neutral or dissatisfied
9,65725,Male,disloyal Customer,20,Business travel,Eco,1061,3,3,3,4,2,3,3,2,2,3,4,4,3,2,0,0.0,neutral or dissatisfied
10,34991,Female,disloyal Customer,24,Business travel,Eco,1182,4,5,5,4,2,5,2,2,3,3,5,3,5,2,0,0.0,neutral or dissatisfied
11,51412,Female,Loyal Customer,12,Personal Travel,Eco Plus,308,2,4,2,2,1,2,1,1,1,2,5,5,5,1,0,0.0,neutral or dissatisfied
12,98628,Male,Loyal Customer,53,Business travel,Eco,834,1,4,4,4,1,1,1,1,1,1,3,4,4,1,28,8.0,neutral or dissatisfied
13,83502,Male,Loyal Customer,33,Personal Travel,Eco,946,4,2,4,3,4,4,4,4,4,5,2,2,2,4,0,0.0,satisfied
14,95789,Female,Loyal Customer,26,Personal Travel,Eco,453,3,2,3,2,2,3,2,2,4,3,2,2,1,2,43,35.0,neutral or dissatisfied
15,100580,Male,disloyal Customer,13,Business travel,Eco,486,2,1,2,3,4,2,1,4,2,1,4,1,3,4,1,0.0,neutral or dissatisfied
16,71142,Female,Loyal Customer,26,Business travel,Business,2123,3,3,3,3,4,4,4,4,5,3,4,5,4,4,49,51.0,satisfied
17,127461,Male,Loyal Customer,41,Business travel,Business,2075,4,4,2,4,4,4,4,5,5,5,5,3,5,5,0,10.0,satisfied
18,70354,Female,Loyal Customer,45,Business travel,Business,2486,4,4,4,4,3,4,5,5,5,5,5,3,5,4,7,5.0,satisfied
19,66246,Male,Loyal Customer,38,Personal Travel,Eco,460,2,3,3,2,5,3,5,5,1,2,4,3,2,5,17,18.0,neutral or dissatisfied
20,39076,Male,Loyal Customer,9,Business travel,Eco,1174,2,4,2,4,2,2,1,2,1,5,3,4,3,2,0,4.0,neutral or dissatisfied
21,22434,Female,Loyal Customer,17,Personal Travel,Eco,208,3,1,3,3,5,3,5,5,2,5,3,3,4,5,0,0.0,neutral or dissatisfied
22,43510,Female,Loyal Customer,43,Personal Travel,Eco,752,3,5,3,5,5,4,5,3,3,3,5,3,3,4,52,29.0,neutral or dissatisfied
```

Test Data

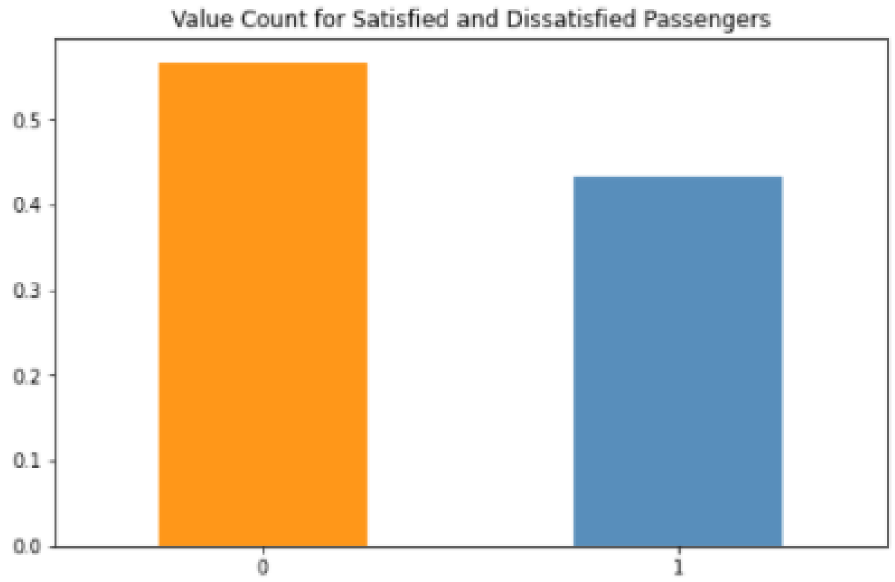
test

File Edit View

```
,id,Gender,Customer Type,Age,Type of Travel,Class,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,Checkin service,Inflight service,Cleanliness,Departure Delay in Minutes,Arrival Delay in Minutes,satisfaction
0,19556,Female,Loyal Customer,52,Business travel,Eco,160,5,4,3,4,3,4,3,5,5,5,5,2,5,5,50,44.0,satisfied
1,90035,Female,Loyal Customer,36,Business travel,Business,2863,1,1,3,1,5,4,5,4,4,4,4,3,4,5,0,0.0,satisfied
2,12360,Male,disloyal Customer,20,Business travel,Eco,192,2,0,2,4,2,2,2,2,4,1,3,2,2,2,0,0.0,neutral or dissatisfied
3,77959,Male,Loyal Customer,44,Business travel,Business,3377,0,0,0,2,3,4,4,1,1,1,1,3,1,4,0,6.0,satisfied
4,36875,Female,Loyal Customer,49,Business travel,Eco,1182,2,3,4,3,4,1,2,2,2,2,2,4,2,4,0,20.0,satisfied
5,39177,Male,Loyal Customer,16,Business travel,Eco,311,3,3,3,3,5,5,3,5,4,3,1,1,2,5,0,0.0,satisfied
6,79433,Female,Loyal Customer,77,Business travel,Business,3987,5,5,5,5,3,5,5,5,5,5,5,4,5,3,0,0.0,satisfied
7,97286,Female,Loyal Customer,43,Business travel,Business,2556,2,2,2,2,4,4,5,4,4,4,4,5,4,3,77,65.0,satisfied
8,27508,Male,Loyal Customer,47,Business travel,Eco,556,5,2,2,2,5,5,5,5,2,2,5,3,3,5,1,0.0,satisfied
9,62482,Female,Loyal Customer,46,Business travel,Business,1744,2,2,2,2,3,4,4,4,4,4,4,5,4,4,28,14.0,satisfied
10,47583,Female,Loyal Customer,47,Business travel,Eco,1235,4,1,1,1,5,1,5,3,3,4,3,1,3,4,29,19.0,satisfied
11,115550,Female,Loyal Customer,33,Business travel,Business,325,2,5,5,5,1,3,4,2,2,2,2,3,2,4,18,7.0,neutral or dissatisfied
12,119987,Female,Loyal Customer,46,Business travel,Business,1009,5,5,5,5,4,5,5,5,5,5,5,5,5,3,0,0.0,satisfied
13,42141,Female,Loyal Customer,60,Business travel,Business,451,1,1,4,1,5,5,4,5,5,5,5,3,5,5,117,113.0,satisfied
14,2274,Female,Loyal Customer,52,Business travel,Business,925,2,2,2,2,5,5,4,4,4,4,4,3,4,5,10,0.0,satisfied
15,22470,Male,Loyal Customer,50,Personal Travel,Eco,83,3,4,0,3,2,0,2,2,4,2,4,4,5,2,5,2.0,neutral or dissatisfied
16,124915,Female,Loyal Customer,31,Business travel,Eco,728,2,5,5,5,2,2,2,2,4,3,3,4,3,2,2,0.0,neutral or dissatisfied
17,17836,Male,Loyal Customer,52,Personal Travel,Eco Plus,1075,5,4,5,3,4,5,4,4,3,5,5,4,5,4,0,0.0,satisfied
18,76872,Female,Loyal Customer,43,Personal Travel,Eco,1927,3,4,3,1,4,4,5,5,5,3,5,4,5,3,0,0.0,neutral or dissatisfied
19,64287,Female,Loyal Customer,50,Business travel,Business,3799,5,5,5,5,4,5,4,4,4,5,4,5,4,5,8,0.0,satisfied
20,63995,Male,Loyal Customer,60,Business travel,Business,612,4,4,4,4,2,4,5,5,5,5,5,5,5,5,21,49.0,satisfied
21,75855,Male,Loyal Customer,43,Personal Travel,Eco,1437,3,4,3,4,2,3,2,2,4,2,4,4,5,2,0,0.0,neutral or dissatisfied
22,106181,Male,Loyal Customer,55,Personal Travel,Eco,302,1,2,4,3,4,4,4,4,1,3,2,4,3,4,0,0.0,neutral or dissatisfied
```

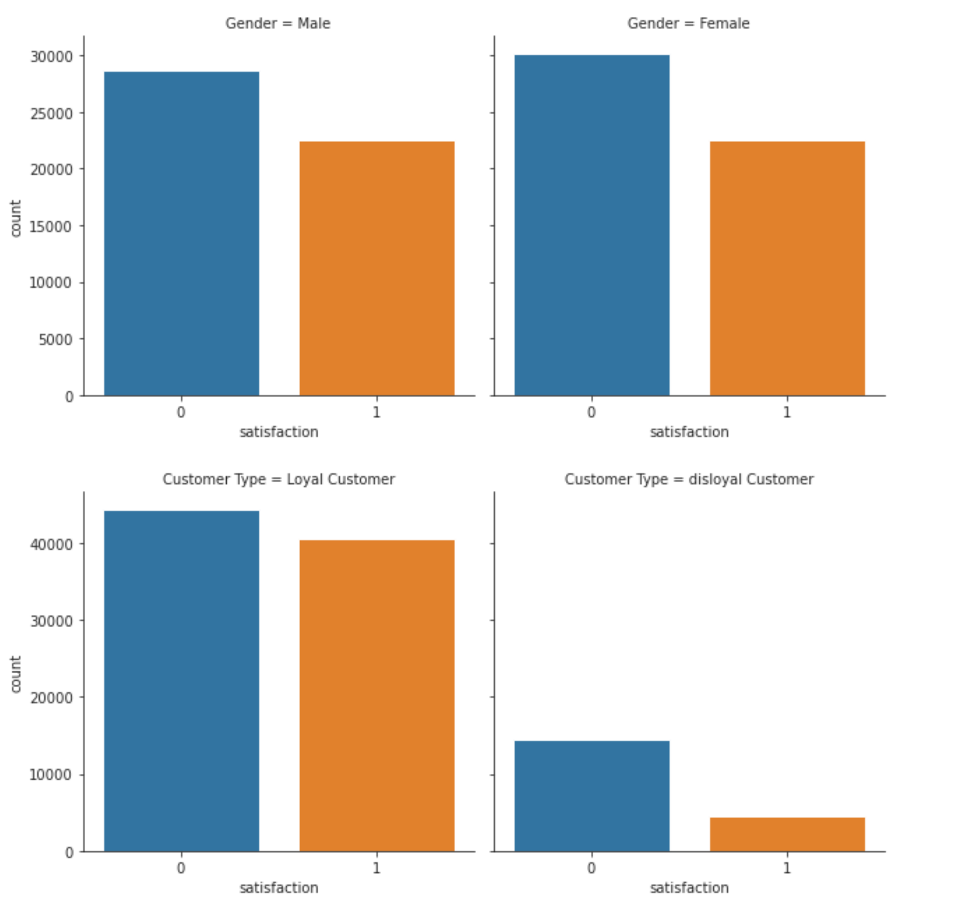
SENTIMENT ANALYSIS OF CUSTOMERS

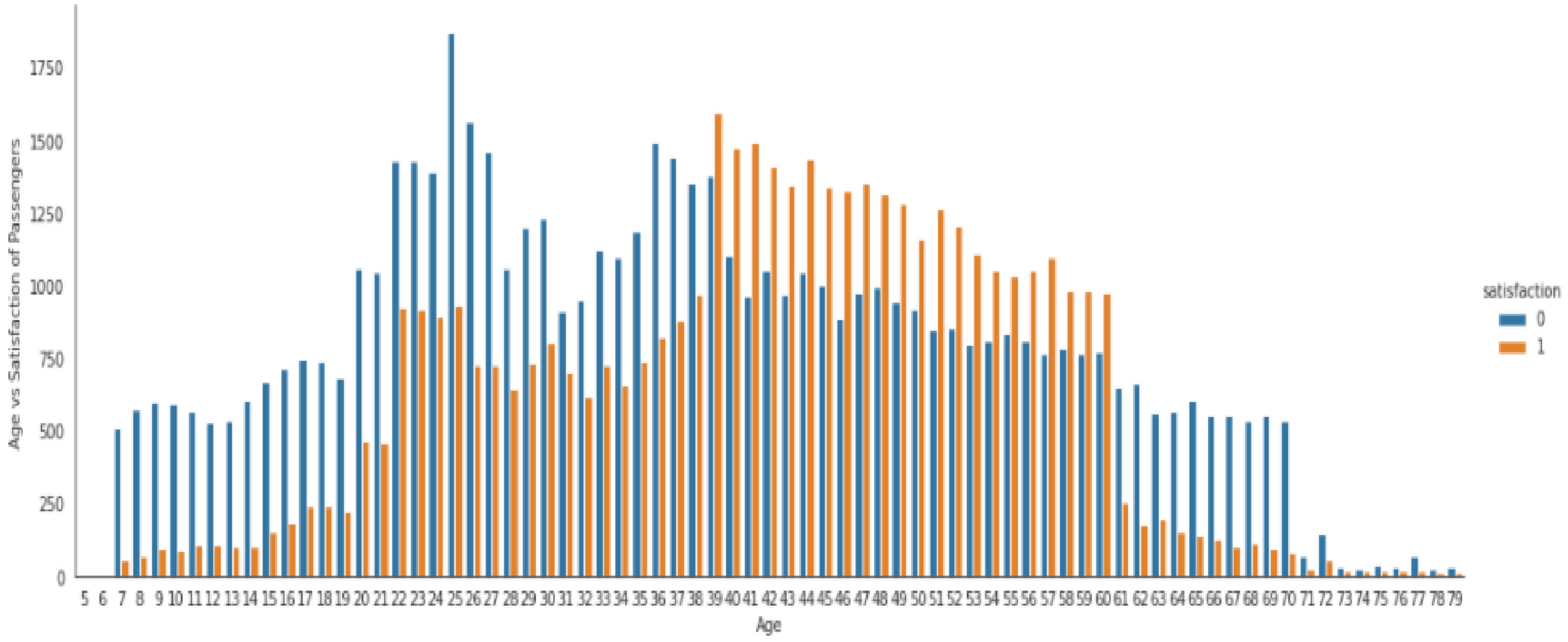
```
import pandas as pd
import numpy as np
import seaborn as sns
df_train = pd.read_csv('train.csv')
df_test = pd.read_csv('test.csv')


fig = plt.figure(figsize = (8,5))

df_train.satisfaction.value_counts(normalize = True).plot(kind='bar',
alpha = 0.9, rot=0, color= ['darkorange','steelblue'])

plt.title(‘Value Count for Satisfied and Dissatisfied Passengers')

plt.show()
```



MALE VS FEMALE
&
LOYAL VS DISLOYAL CUSTOMER SATISFACTION

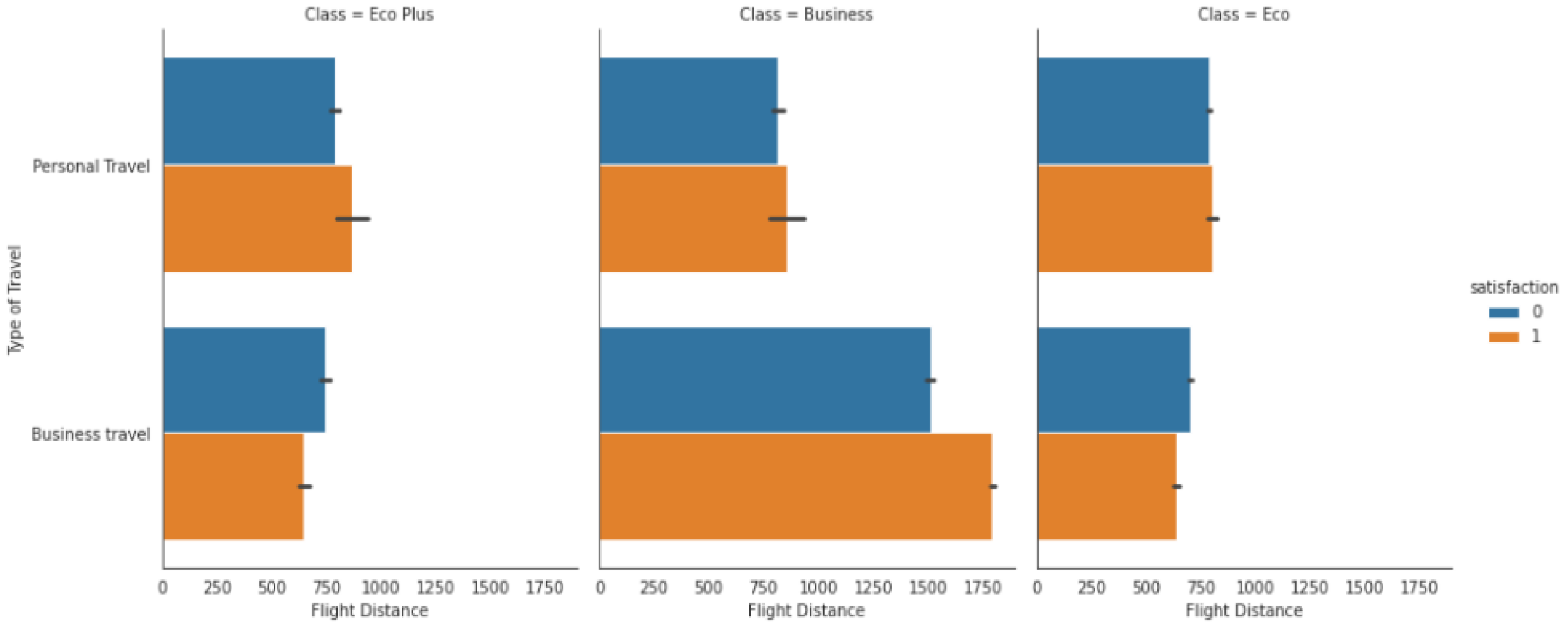
```
df_train['Arrival Delay in Minutes'] = df_train['Arrival Delay in Minutes'].fillna(df_train['Arrival Delay in Minutes'].mean())

df_test['Arrival Delay in Minutes'] = df_test['Arrival Delay in Minutes'].fillna(df_test['Arrival Delay in Minutes'].mean())

with sns.axes_style(style='ticks'):
    g = sns.catplot("satisfaction", col="Gender", col_wrap=2, data=df_train, kind="count", height=4.5, aspect=1.0)
    g = sns.catplot("satisfaction", col="Customer Type", col_wrap=2, data=df_train, kind="count", height=4.5, aspect=1.0)
```





Age - It is observed that from age 7 to 38 and from age 61 to 79 the number of disssatisfied passengers is comparatively higher, which gives us an insight as to which target group should the airline focus to improve the passenger satisfaction ratings.

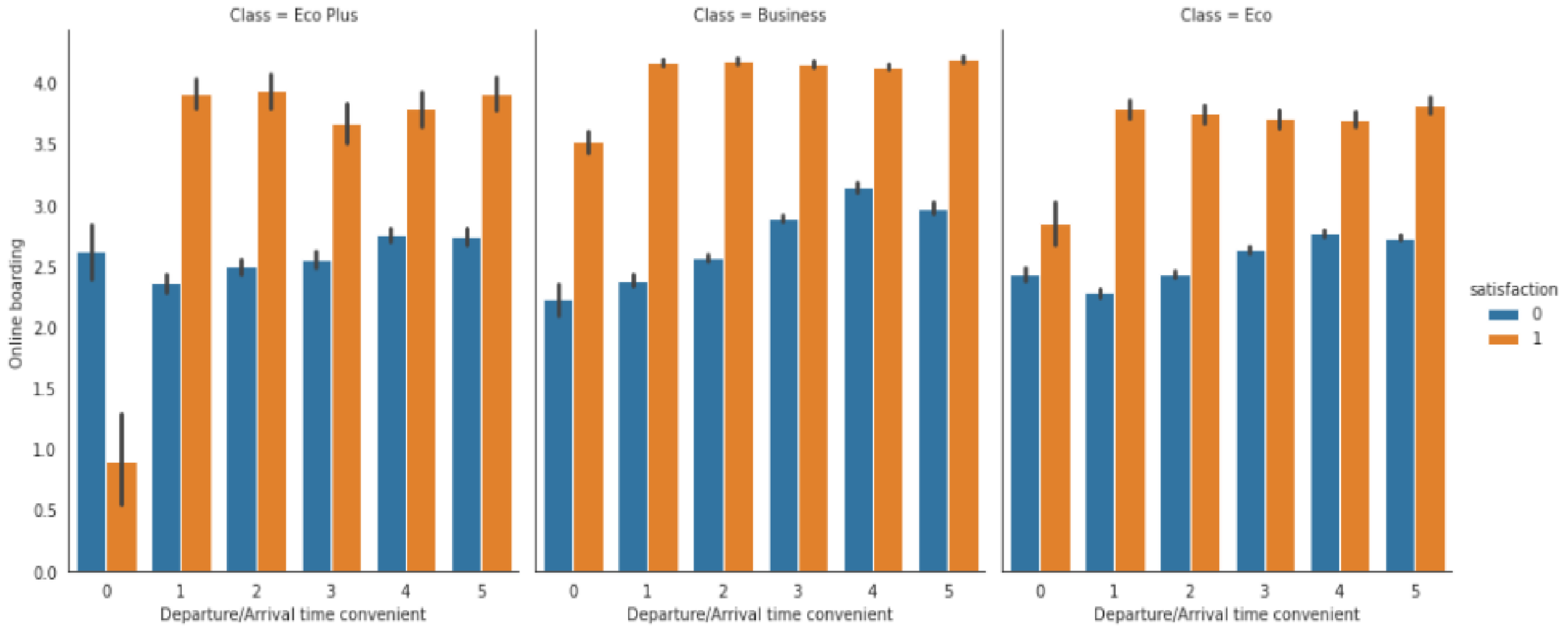
On the contrary, we can see that passenger in the age 39 - 60 are satisfied with their experience.



Class, Flight Distance and Type of Travel

We can see that for Eco, Eco Plus and Business class passengers who are travelling for Personal Reasons the number of Satisfied customers are just a bit more than disssatisfied passengers.

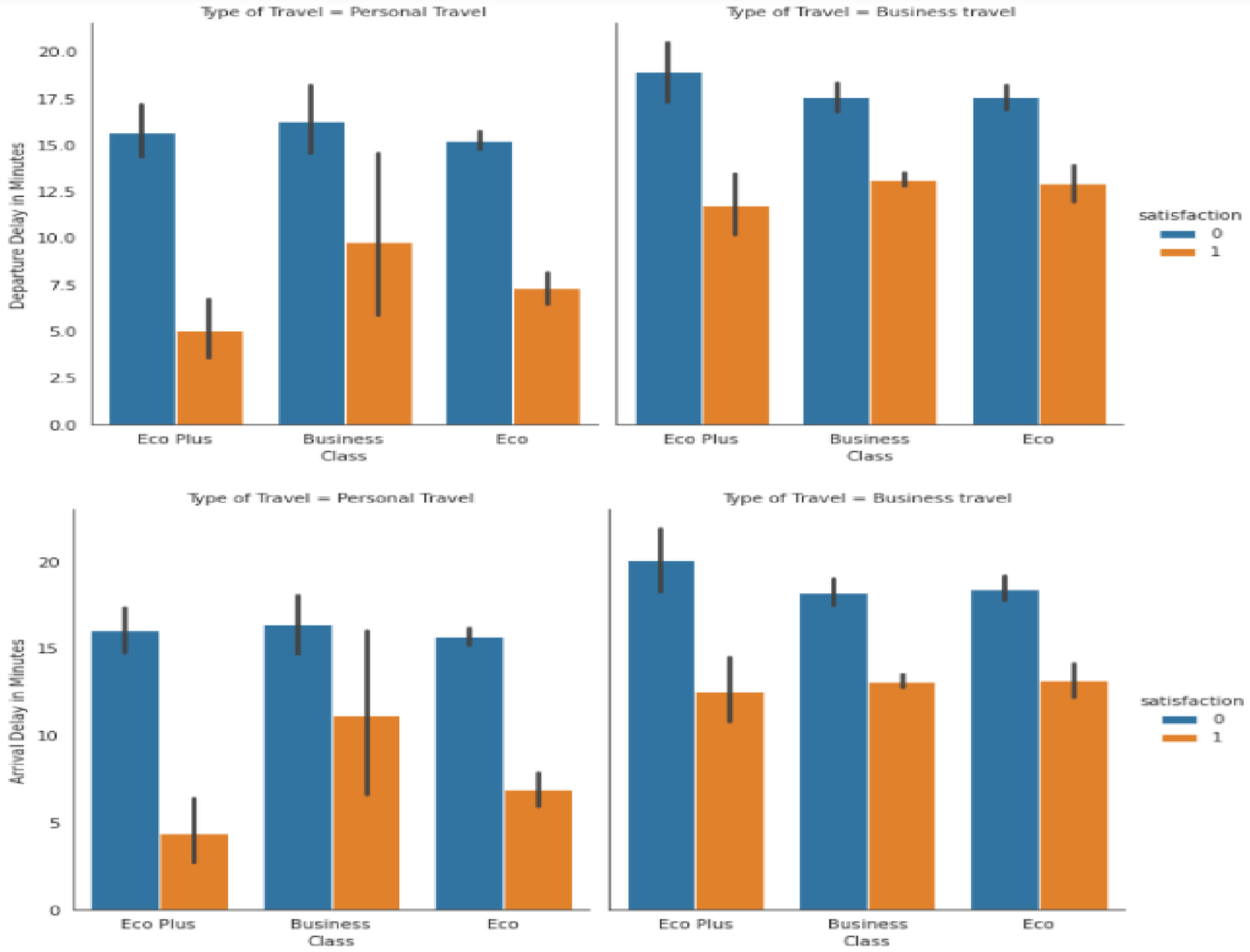
Also we can observe an interesting comparision here, the passengers who are traveling for Business Purpose, but are travelling through Eco and Eco Plus class are more disssatisfied, on the contrary the passengers travelling by Business class for Business Purpose are more satisfied.



Departure/Arrival Time, Online Boarding grouped by Class

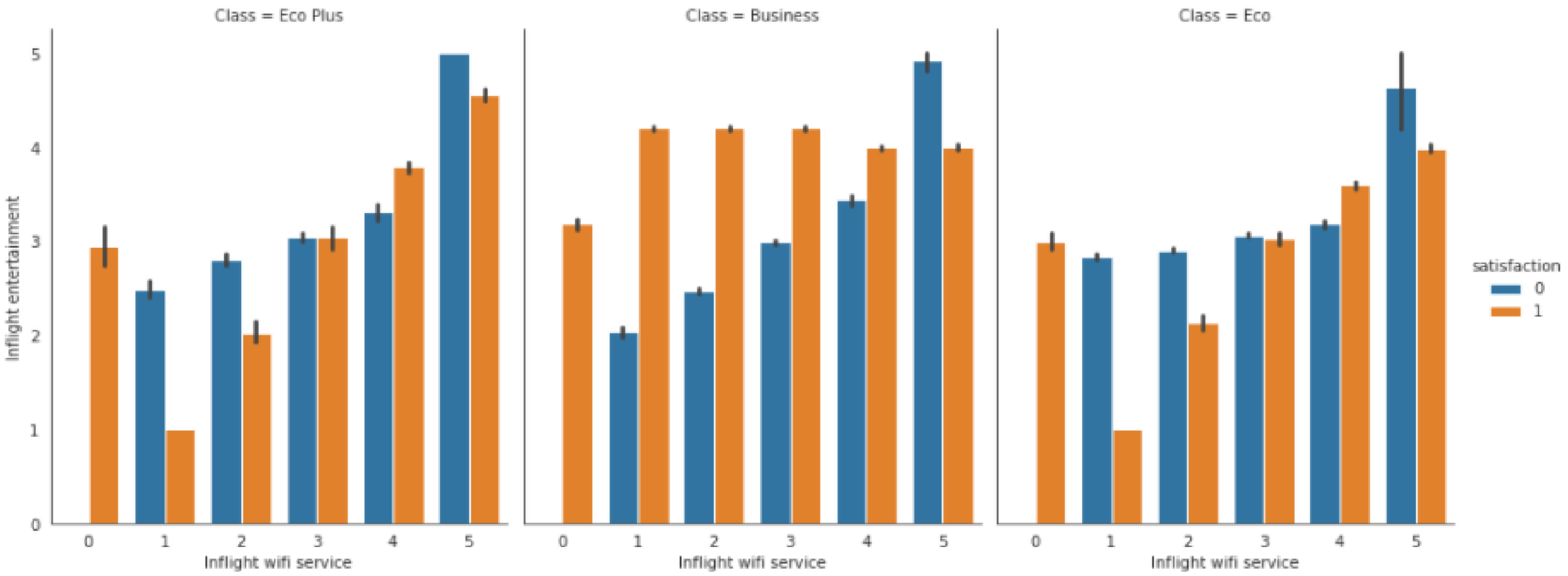
Except for the Eco Plus class which has higher number of disssatisfied passengers, where they have provided 0 rating, there seems to be more number of satisfied passengers across classes.

This analysis proves that passengers need convenient features like Online Boarding to make thier flight experience pleasing.



Arrival and Departure Delay grouped by Type of Travel

From the graphs above it is evident that no passenger likes delays. The number of disssatisfied passengers travelling for Business Purpose are greater than those travelling for Personal Reasons.

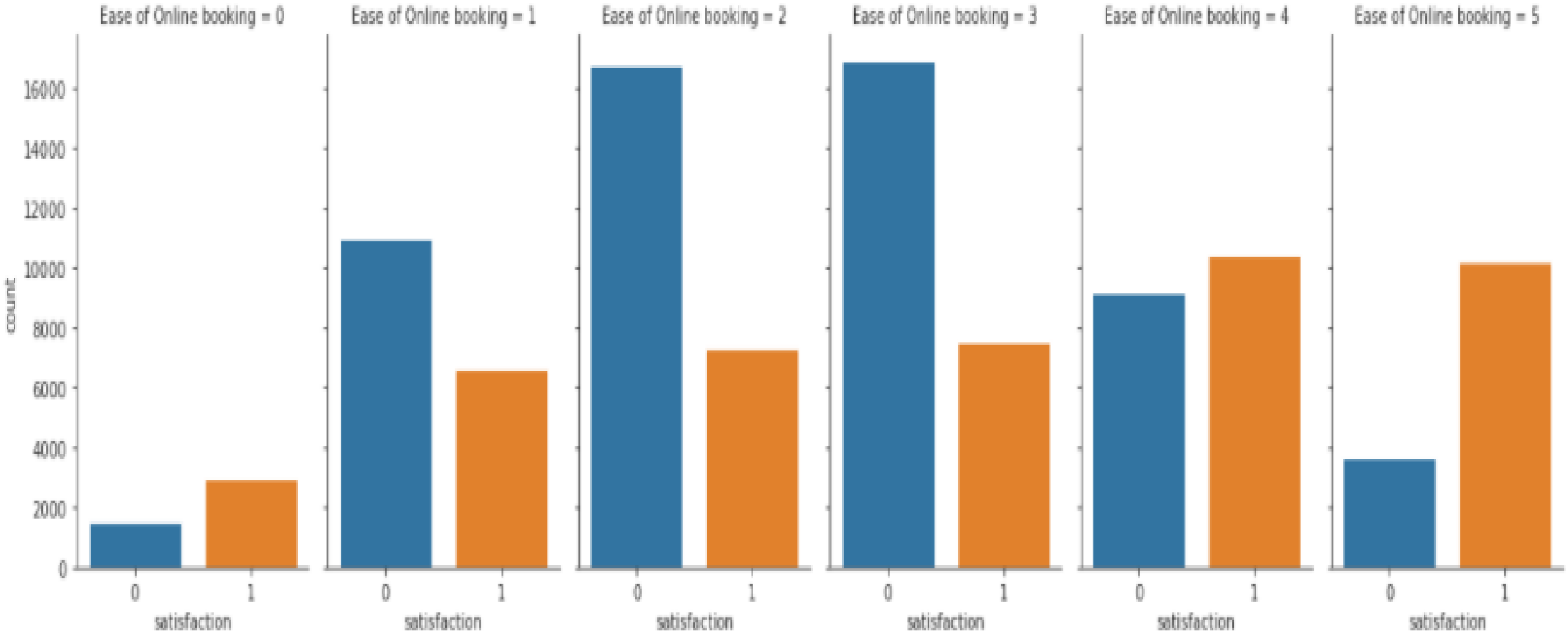


Inflight WiFi and Entertainment grouped by Class

We have a very unusual stat here, where we can see that Eco Plus passengers are satisfied even if they do not have any WiFi services or just Mid Level of Entertainment.

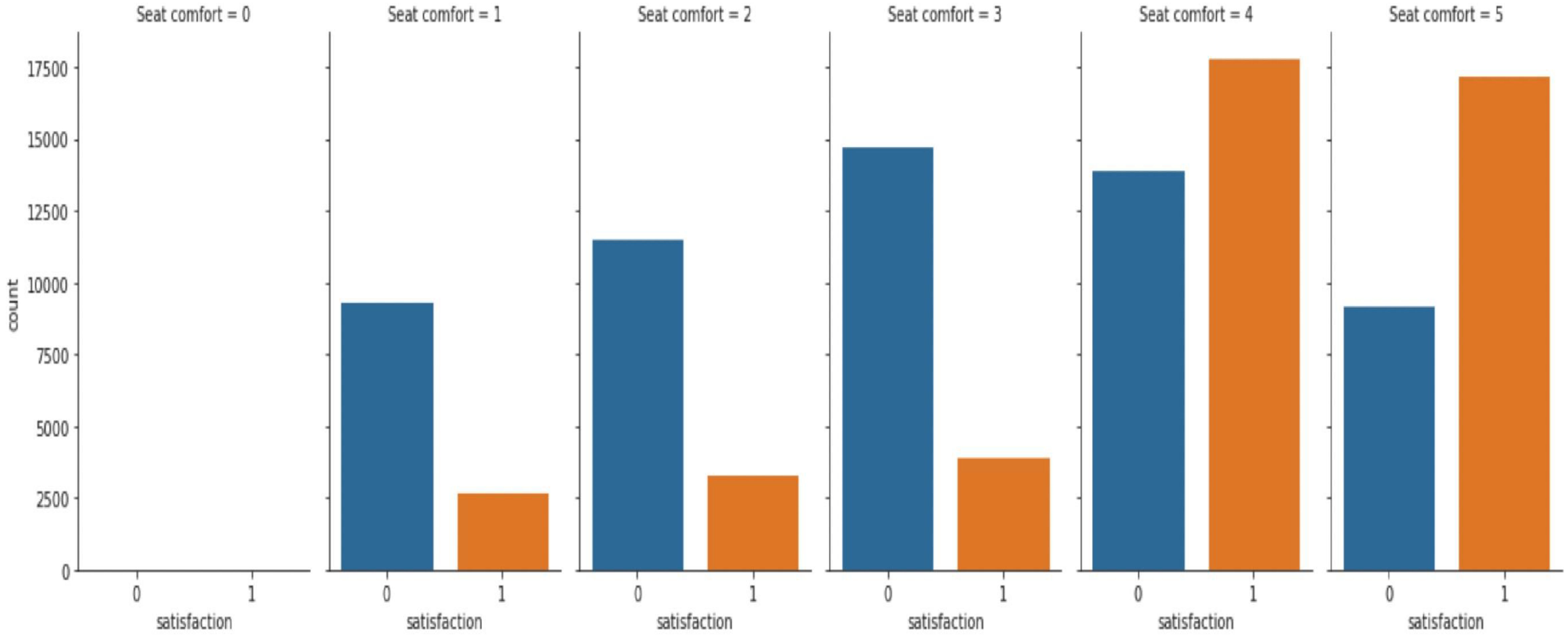
For Business class passengers it is evident that they need the highest levels of WiFi and Entertainment services as they have paid a significantly higher amount of charges per seat.

For Eco passengers, they need high level of Entertainment and WiFi services to be satisfied.



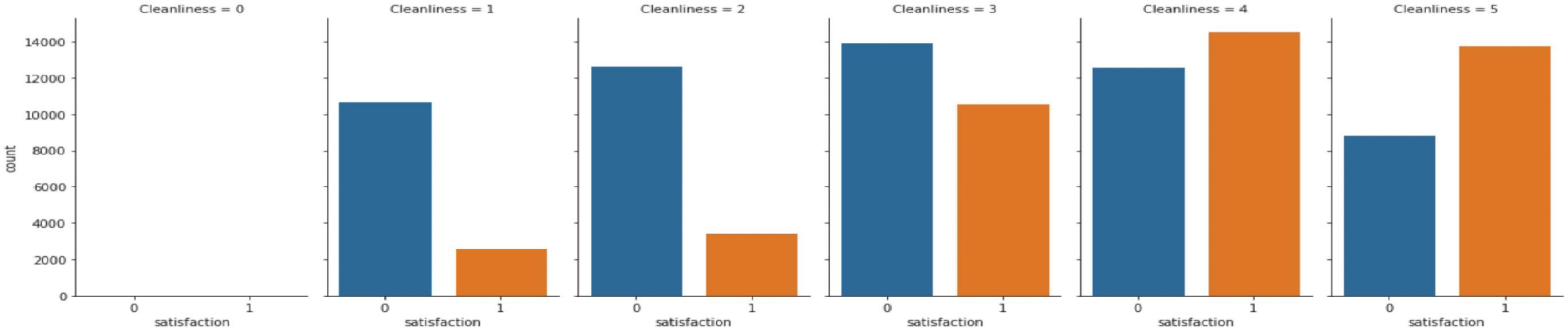
Ease of Online Booking

We can see that passengers are only satisfied with the highest level of convenience of ratings 4 and 5 to be satisfied.



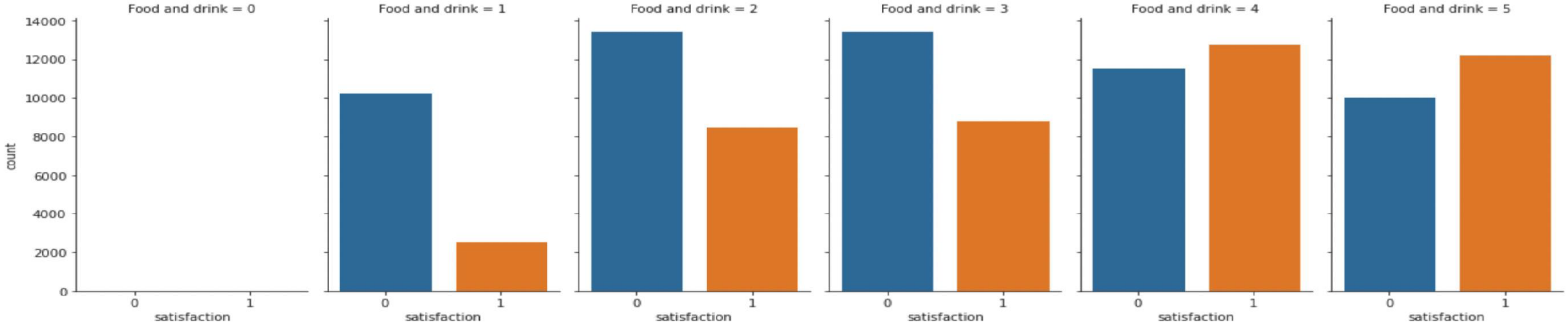
Seat Comfort

We can see that passengers are only satisfied with the highest level of seat comforts, where they are probably getting more leg space or window seats which has ratings 4 and 5 to be satisfied.



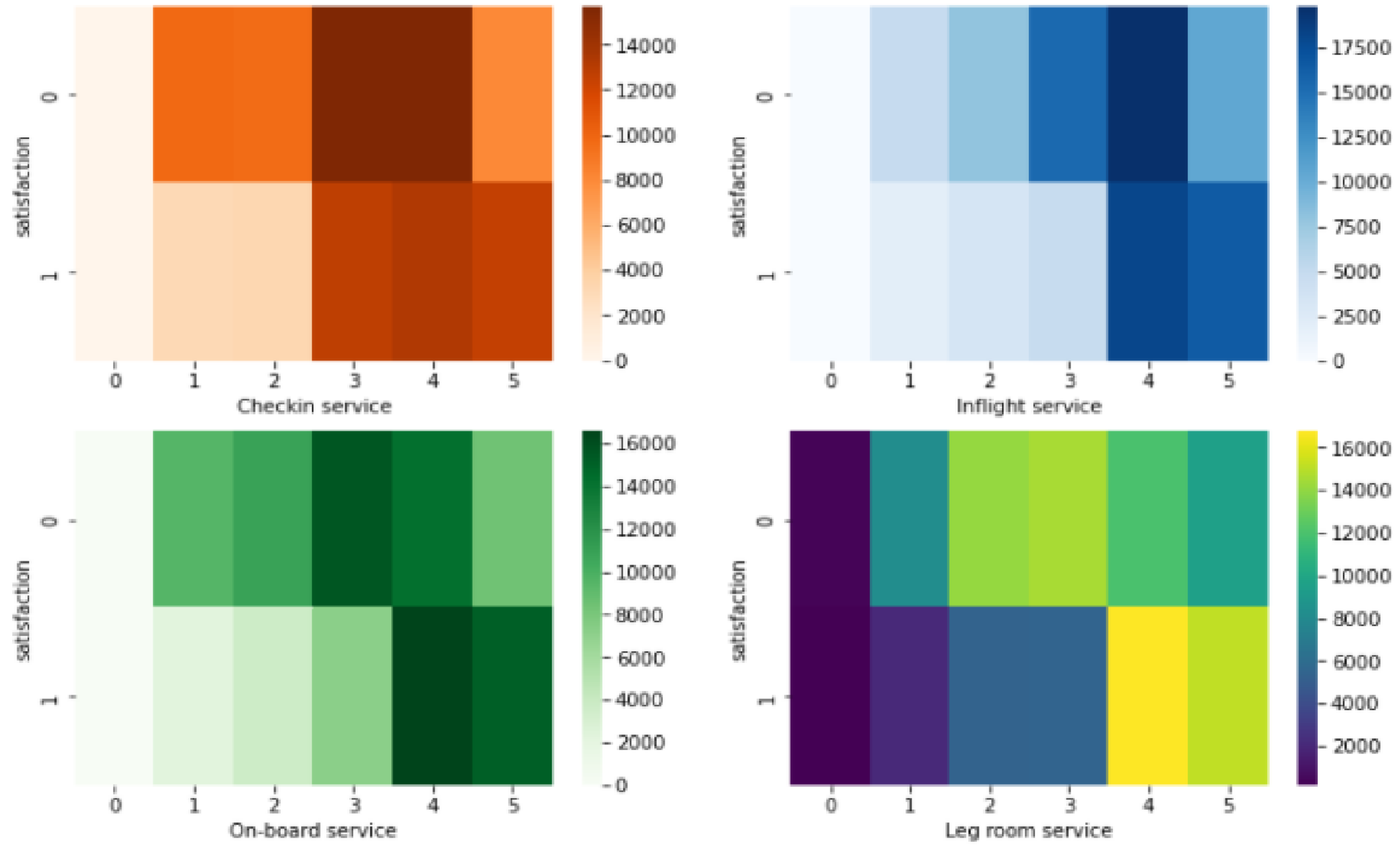
Cleanliness

We can see that passengers are only satisfied with the highest level of cleanliness of ratings 4 and 5 to be satisfied.



Food and Drinks

We can see that passengers are only satisfied with the highest level of Food and Drink where they are probably getting all of the requested food and drinks which has ratings 4 and 5 to be satisfied.

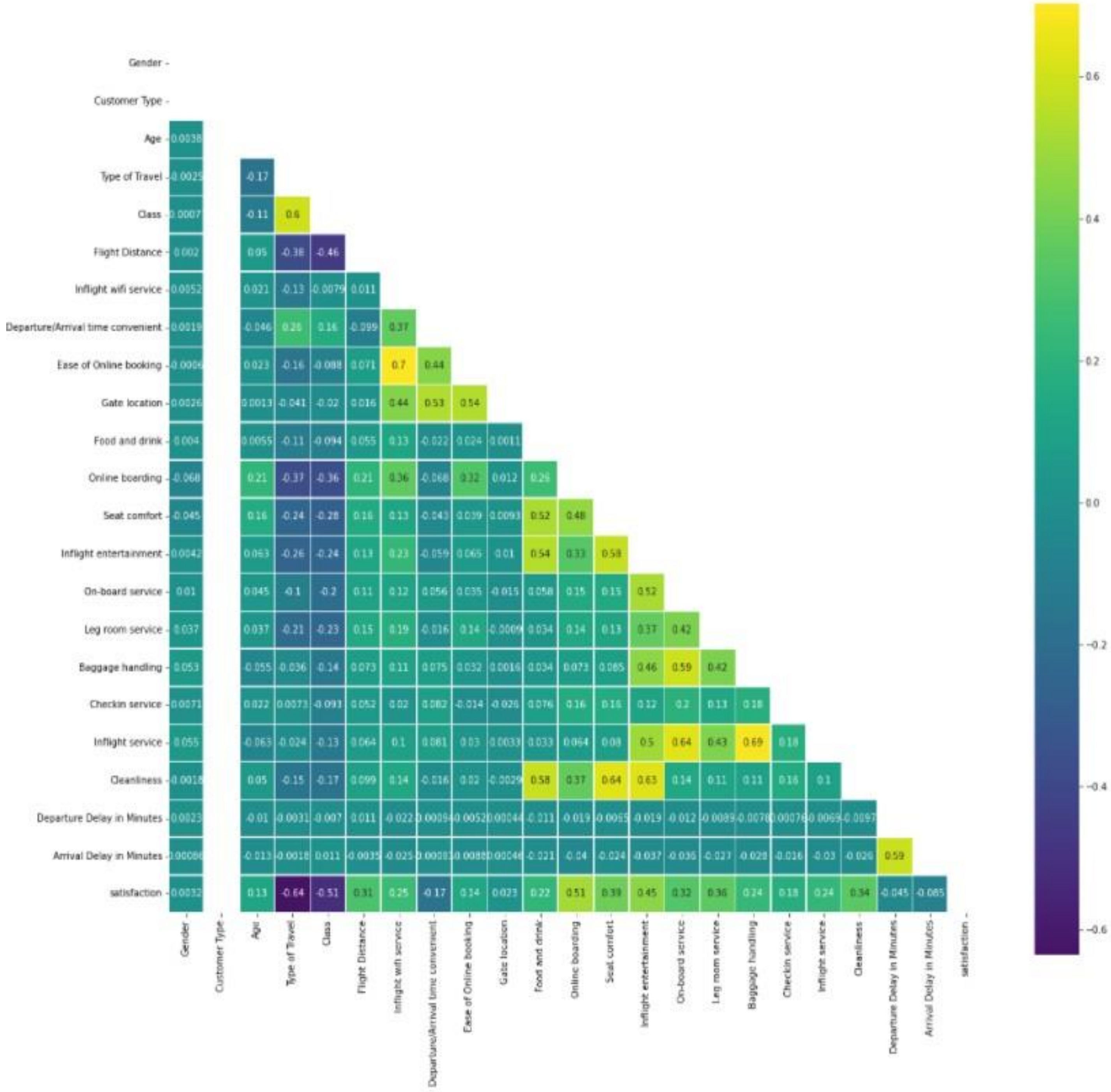


Checkin Service, Inflight Service, On-Board Service, Leg-room service

Except for the checkin service, which has 0 to 2 ratings provided by passengers who look to be the most dissatisfied, for rest of services, passengers who provided 4 and 5 ratings seem to be satisfied.

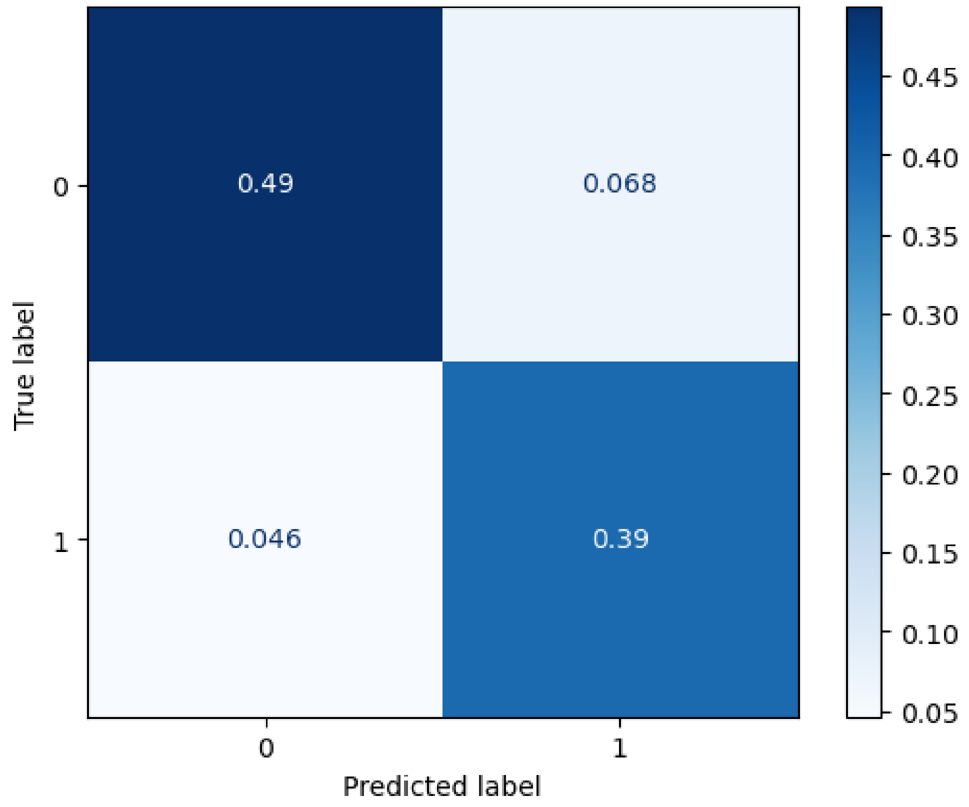
```
df_train = df_train[~((df_train < (q1 - 1.5 * inter_q)) |(df_train >
(q3 + 1.5 * inter_q))).any(axis=1)]
df_train.shape

relation = df_train.corr()
m = np.triu(np.ones_like(relation, dtype=bool))
f, ax = plt.subplots(figsize=(20, 20))
sns.heatmap(relation, mask=m, cmap='viridis', vmax=None,
center=0,square=True, annot=True, linewidths=.5,
        cbar_kws={"shrink": .9})
```

Finding the correlation among the different features to check which ones are important



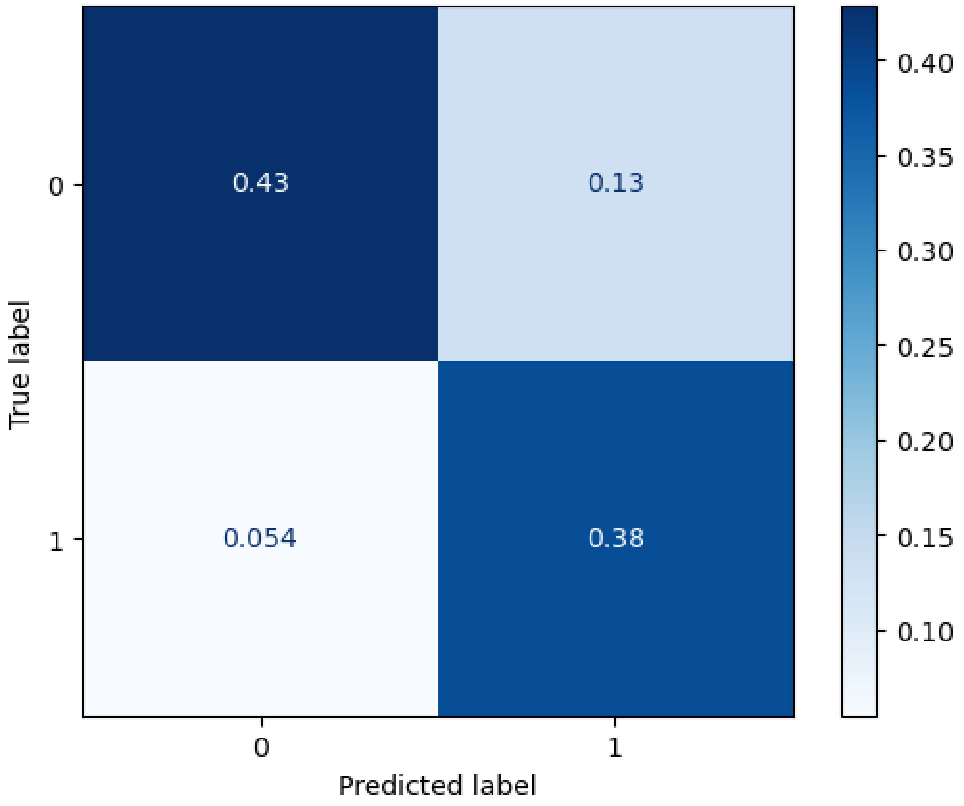
Algorithm -1 :: Logistic Regression

```
from sklearn.linear_model import LogisticRegression

params_lr = {'penalty': 'elasticnet', 'l1_ratio':0.5, 'solver': 'saga'}

model_lr = LogisticRegression(**params_lr)
model_lr, accuracy_lr = classifier(model_lr, X_train, y_train, X_test, y_test)
```

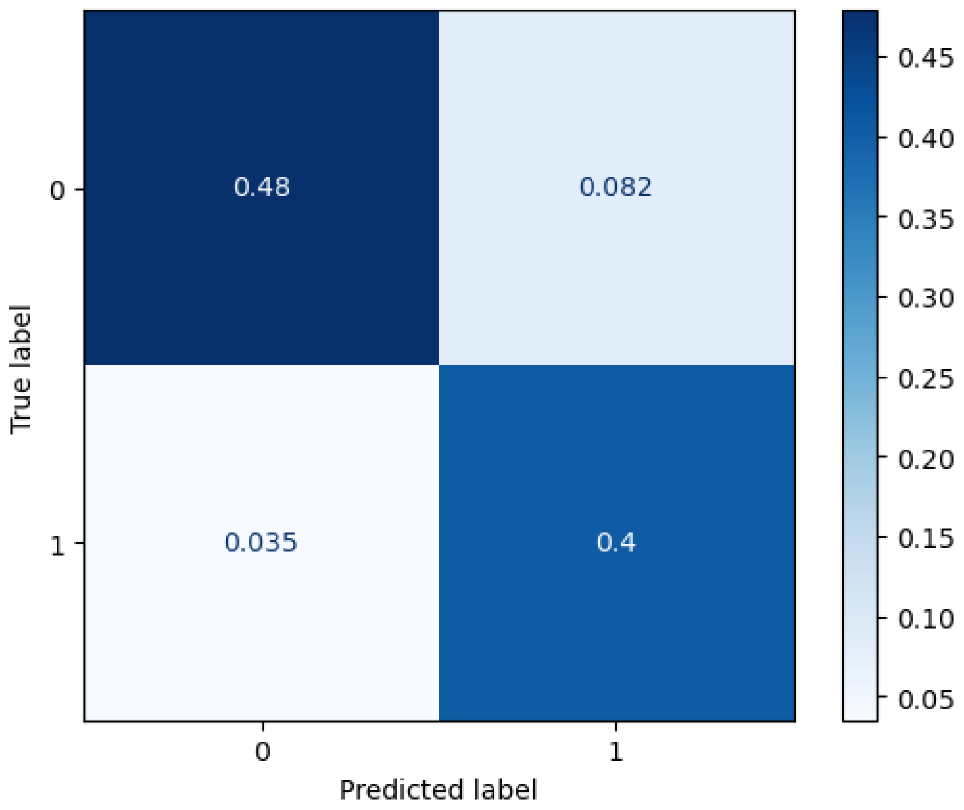


Algorithm -2 :: K- Nearest Neighbor

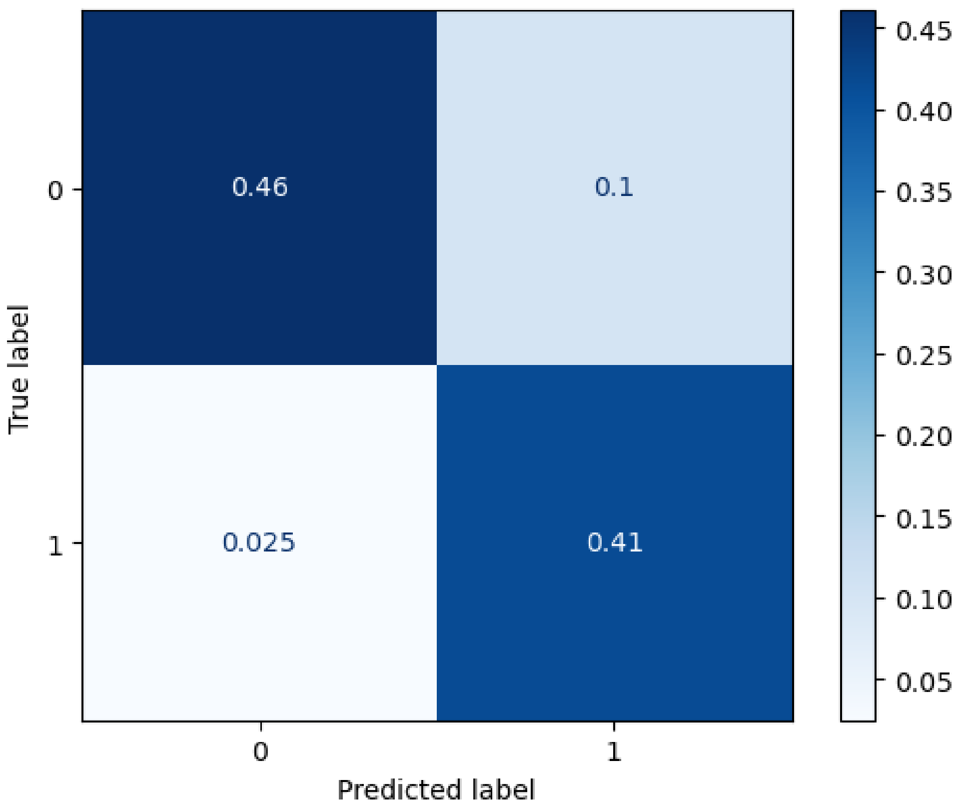
```
from sklearn.neighbors import KNeighborsClassifier

params_kn = {'n_neighbors':10, 'algorithm': 'kd_tree', 'n_jobs':4}

model_kn = KNeighborsClassifier(**params_kn)
model_kn, accuracy_kn = classifier(model_kn, X_train, y_train, X_test,
y_test)
```



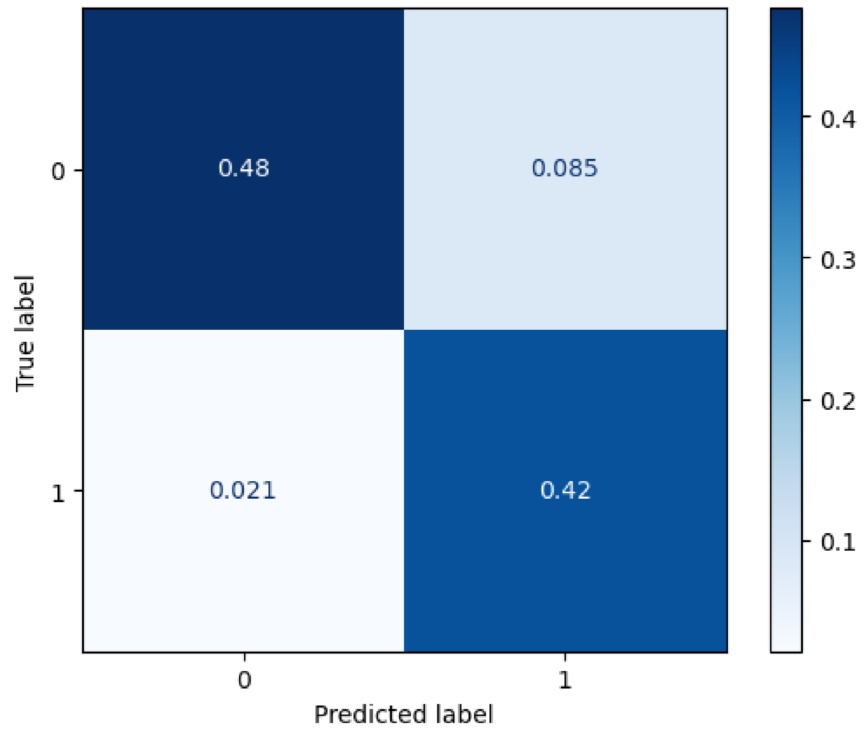
```
# Algorithm -3 :: Decision Tree

from sklearn.tree import DecisionTreeClassifier

params_dt = {'max_depth': 12,
          'max_features': "sqrt"}

model_dt = DecisionTreeClassifier(**params_dt)
model_dt, accuracy_dt = classifier(model_dt, X_train, y_train, X_test,
y_test)
```



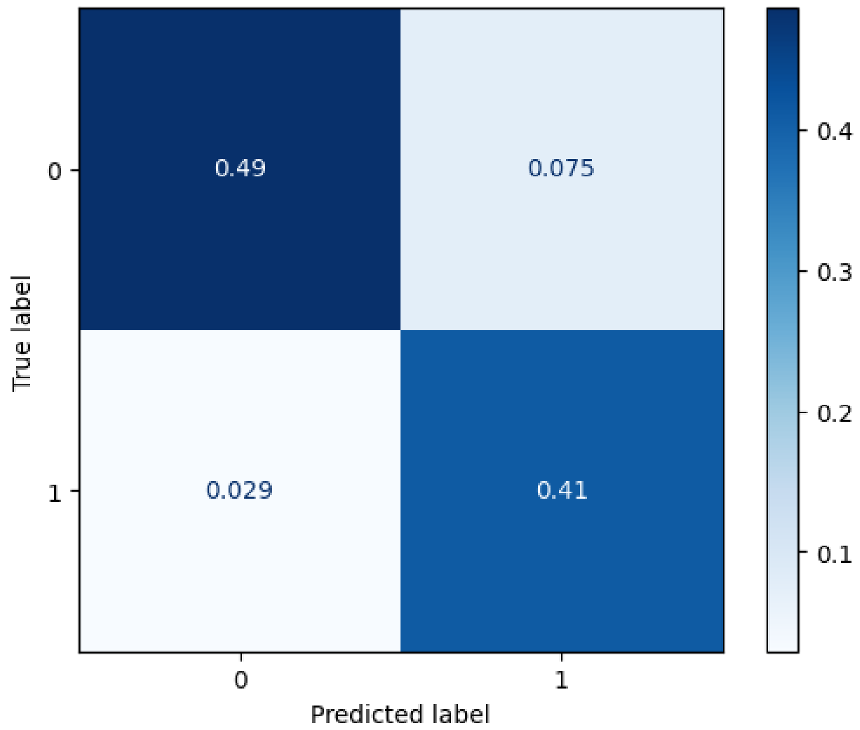

```
# Algorithm -4 :: Artificial Neural Networks
from sklearn.neural_network import MLPClassifier

params_nn = {'hidden_layer_sizes': (30,30,30),
          'activation': 'logistic',
          'solver': 'lbfgs',
          'max_iter': 100}

model_nn = MLPClassifier(**params_nn)
model_nn, accuracy_nn = classifier(model_nn, X_train, y_train, X_test,
y_test)
```

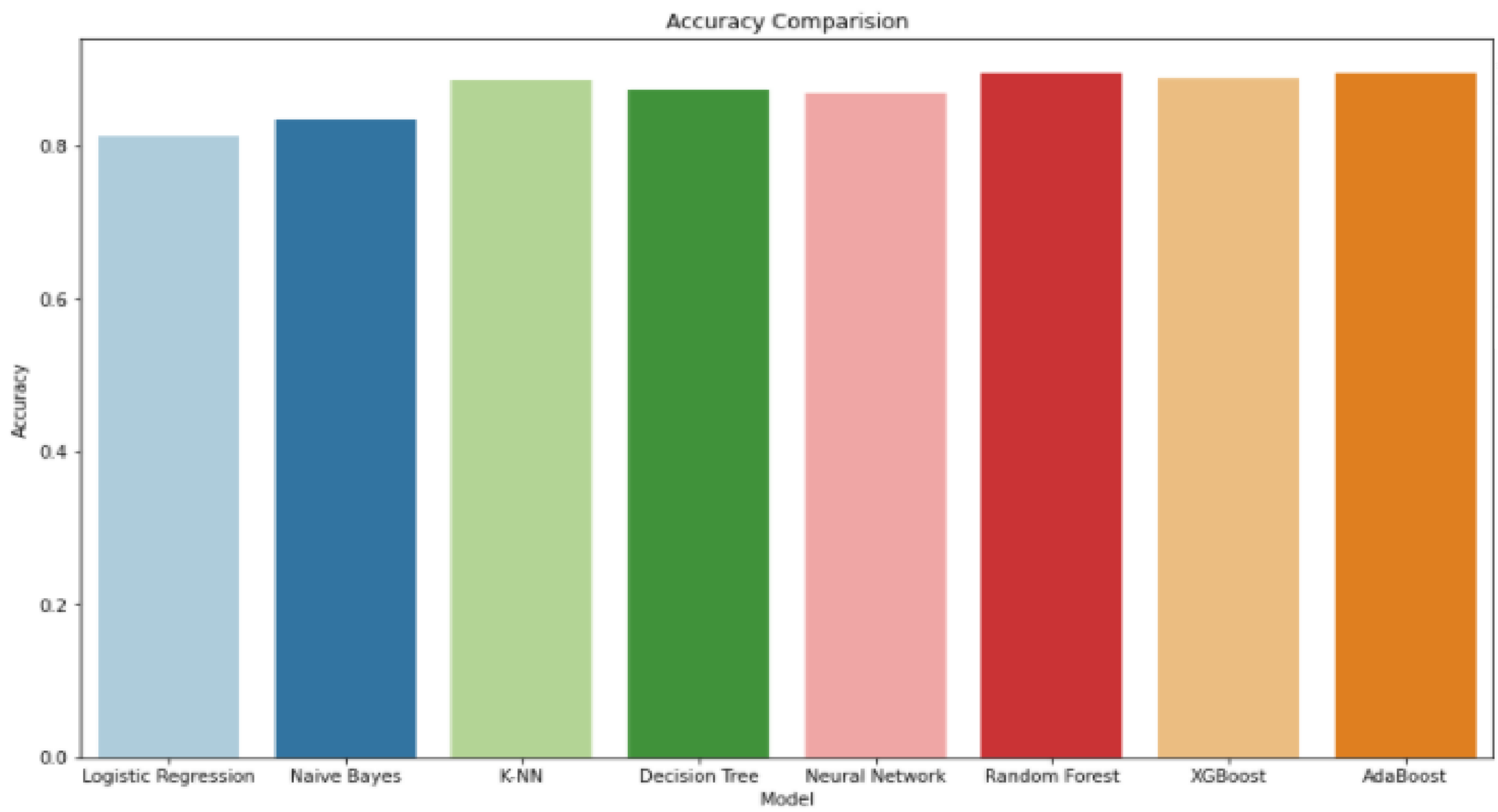


```
# Algorithm -5 :: Random Forest
from sklearn.ensemble import RandomForestClassifier

params_rf = {'max_depth': 16,
          'min_samples_leaf': 1,
          'min_samples_split': 2,
          'n_estimators': 100,
          'random_state': 12345}

model_rf = RandomForestClassifier(**params_rf)
model_rf, accuracy_rf = classifier(model_rf, X_train, y_train, X_test,
y_test)
```




```
# Algorithm -6 :: AdaBoost
from sklearn.ensemble import AdaBoostClassifier as adab
params_adab ={'n_estimators': 500,
          'random_state': 12345}

model_adab = adab(**params_adab)
model_adab, accuracy_adab = classifier(model_adab, X_train,
y_train, X_test, y_test)
```





	Model	Accuracy
0	Logistic Regression	0.813289
1	Naive Bayes	0.833462
2	K-NN	0.886126
3	Decision Tree	0.882430
4	Neural Network	0.873037
5	Random Forest	0.894133
6	XGBoost	0.888820
7	AdaBoost	0.895519

Best Model - From the Accuracies and Bar Plot above, we can see that AdaBoost has the best accuracy of 89.55% followed by Random Forest which is not so behind with an accuracy score of 89.41% and the third best model is XGBoost which has an accuracy score of 88.88%

Convolutional-Neural-Network

Convolutional Neural Network Model-

In [51]:
```
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras import layers

# Load the train and test datasets
train_data = pd.read_csv(r'C:\Users\suyas\OneDrive-stevens.edu\Desktop\Customer_Satisfaction\archive\train.csv')
test_data = pd.read_csv(r'C:\Users\suyas\OneDrive-stevens.edu\Desktop\Customer_Satisfaction\archive\test.csv')

# Prepare the data
X_train = train_data['Cleanliness'].astype(str).values
y_train = train_data['satisfaction'].values
X_test = test_data['Cleanliness'].astype(str).values
y_test = test_data['satisfaction'].values

# Convert target labels to lowercase
y_train = np.array([str(label).lower() for label in y_train])
y_test = np.array([str(label).lower() for label in y_test])

# Encode the target labels
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(y_train)
y_test = label_encoder.transform(y_test)

# Text preprocessing
tokenizer = Tokenizer(num_words=10000)
tokenizer.fit_on_texts(X_train)
X_train = tokenizer.texts_to_sequences(X_train)
X_test = tokenizer.texts_to_sequences(X_test)
X_train = pad_sequences(X_train, maxlen=100, padding='post', truncating='post')
X_test = pad_sequences(X_test, maxlen=100, padding='post', truncating='post')

# Define the CNN architecture
model = tf.keras.Sequential([
    layers.Embedding(input_dim=10000, output_dim=100, input_length=100),
    layers.Conv1D(128, 5, activation='relu'),
    layers.GlobalMaxPooling1D(),
    layers.Dense(64, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

# Train the CNN model
model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10, batch_size=32)
```

Accuracy- 63.53 %

In [50]:
```
from tensorflow.keras.metrics import Accuracy

# Train the CNN model
model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10, batch_size=32)

# Evaluate the model on the test set
y_pred = np.round(model.predict(X_test)).flatten()
accuracy = Accuracy()
accuracy.update_state(y_test, y_pred)
accuracy_value = accuracy.result().numpy()
print(f"Accuracy: {accuracy_value}")
```

```
Epoch 1/10
2598/2598 [==============================] - 75s 29ms/step - loss: 0.6342 - accuracy: 0.6317 - val_loss: 0.6309 - val_accuracy:
0.6343
Epoch 2/10
2598/2598 [==============================] - 85s 33ms/step - loss: 0.6340 - accuracy: 0.6317 - val_loss: 0.6306 - val_accuracy:
0.6343
Epoch 3/10
2598/2598 [==============================] - 103s 40ms/step - loss: 0.6338 - accuracy: 0.6321 - val_loss: 0.6312 - val_accurac
y: 0.6343
Epoch 4/10
2598/2598 [==============================] - 76s 29ms/step - loss: 0.6338 - accuracy: 0.6318 - val_loss: 0.6326 - val_accuracy:
0.6343
Epoch 5/10
2598/2598 [==============================] - 115s 44ms/step - loss: 0.6337 - accuracy: 0.6319 - val_loss: 0.6306 - val_accurac
y: 0.6343
Epoch 6/10
2598/2598 [==============================] - 113s 43ms/step - loss: 0.6337 - accuracy: 0.6322 - val_loss: 0.6309 - val_accurac
y: 0.6343
Epoch 7/10
2598/2598 [==============================] - 79s 30ms/step - loss: 0.6337 - accuracy: 0.6320 - val_loss: 0.6305 - val_accuracy:
0.6343
Epoch 8/10
2598/2598 [==============================] - 71s 27ms/step - loss: 0.6335 - accuracy: 0.6323 - val_loss: 0.6307 - val_accuracy:
0.6343
Epoch 9/10
2598/2598 [==============================] - 71s 27ms/step - loss: 0.6334 - accuracy: 0.6323 - val_loss: 0.6309 - val_accuracy:
0.6343
Epoch 10/10
2598/2598 [==============================] - 72s 28ms/step - loss: 0.6334 - accuracy: 0.6322 - val_loss: 0.6311 - val_accuracy:
0.6343
812/812 [==============================] - 6s 7ms/step
Accuracy: 0.6353557109832764
```

Long-Short-Term-Memory Model

LSTM MODEL-

In [*]:

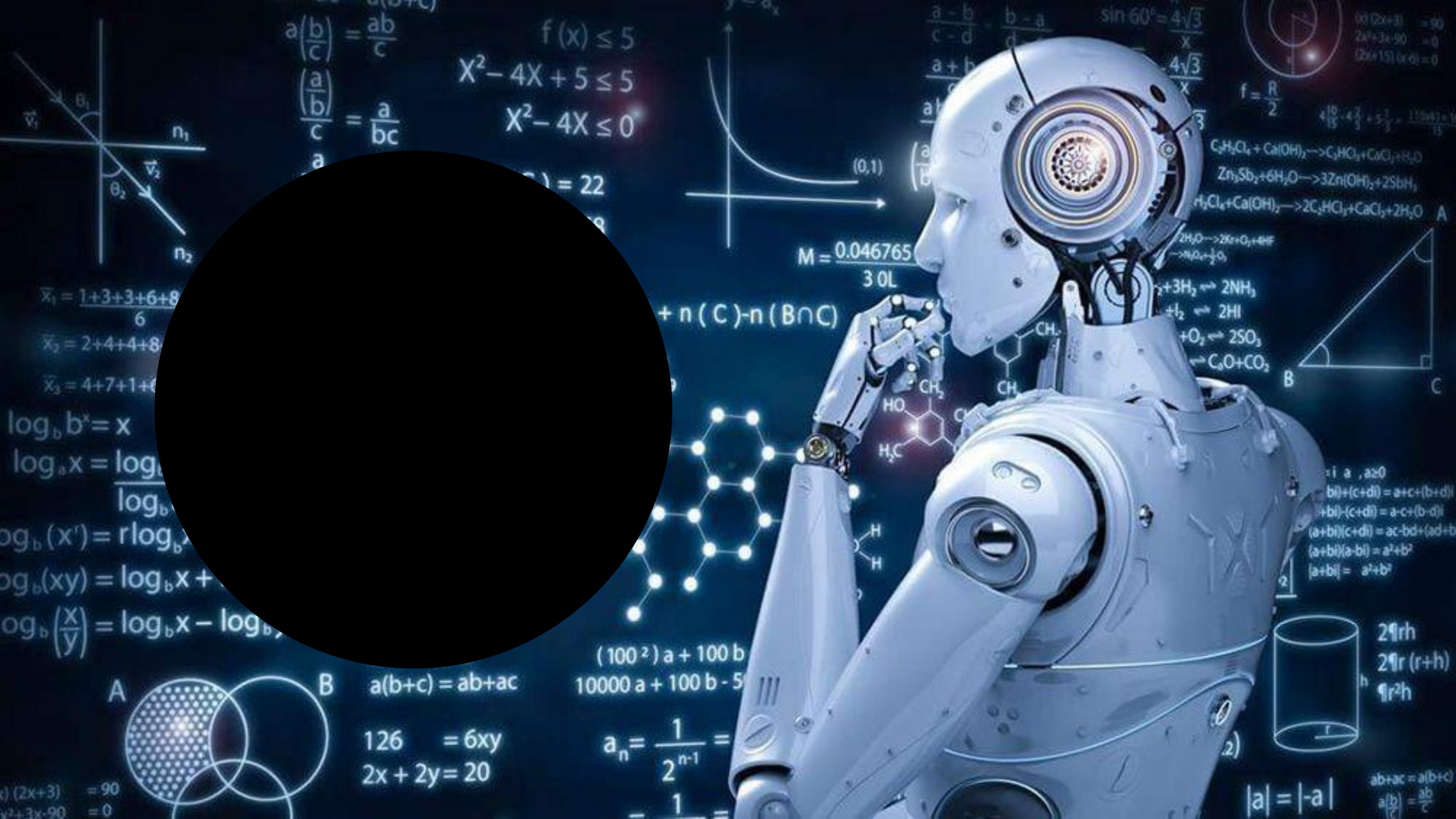
```python
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras import layers

# Load the train and test datasets
train_data = pd.read_csv(r'C:\Users\suyas\OneDrive-stevens.edu\Desktop\Customer_Satisfaction\archive\train.csv')
test_data = pd.read_csv(r'C:\Users\suyas\OneDrive-stevens.edu\Desktop\Customer_Satisfaction\archive\test.csv')

# Prepare the data
X_train = train_data['Cleanliness'].astype(str).values
y_train = train_data['satisfaction'].values
X_test = test_data['Cleanliness'].astype(str).values
y_test = test_data['satisfaction'].values

# Convert target labels to lowercase
y_train = np.array([str(label).lower() for label in y_train])
y_test = np.array([str(label).lower() for label in y_test])

# Encode the target labels
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(y_train)
y_test = label_encoder.transform(y_test)

# Text preprocessing
tokenizer = Tokenizer(num_words=10000)
tokenizer.fit_on_texts(X_train)
X_train = tokenizer.texts_to_sequences(X_train)
X_test = tokenizer.texts_to_sequences(X_test)
X_train = pad_sequences(X_train, maxlen=100, padding='post', truncating='post')
X_test = pad_sequences(X_test, maxlen=100, padding='post', truncating='post')

# Define the LSTM architecture
model = tf.keras.Sequential([
    layers.Embedding(input_dim=10000, output_dim=100, input_length=100),
    layers.LSTM(128, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

# Train the LSTM model
model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10, batch_size=32)
```

Accuracy- 56.10 %

```python
# Train the LSTM model
model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10, batch_size=32)

# Generate predictions on the test set
y_pred = np.round(model.predict(X_test)).flatten()

# Calculate the accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Epoch 1/10
2598/2598 [==============================] - 567s 217ms/step - loss: 0.6844 - accuracy: 0.5674 - val_loss: 0.6850 - val_accurac
y: 0.5636
Epoch 2/10
2598/2598 [==============================] - 549s 212ms/step - loss: 0.6843 - accuracy: 0.5674 - val_loss: 0.6851 - val_accurac
y: 0.5636
Epoch 3/10
2598/2598 [==============================] - 470s 181ms/step - loss: 0.6842 - accuracy: 0.5674 - val_loss: 0.6851 - val_accurac
y: 0.5636
Epoch 4/10
2598/2598 [==============================] - 251s 97ms/step - loss: 0.6842 - accuracy: 0.5674 - val_loss: 0.6851 - val_accurac
y: 0.5636
Epoch 5/10
2598/2598 [==============================] - 438s 169ms/step - loss: 0.6842 - accuracy: 0.5674 - val_loss: 0.6851 - val_accurac
y: 0.5636
Epoch 6/10
2598/2598 [==============================] - 520s 200ms/step - loss: 0.6841 - accuracy: 0.5674 - val_loss: 0.6853 - val_accurac
y: 0.5636
Epoch 7/10
2598/2598 [==============================] - 520s 200ms/step - loss: 0.6841 - accuracy: 0.5674 - val_loss: 0.6850 - val_accurac
y: 0.5636
Epoch 8/10
2598/2598 [==============================] - 973s 375ms/step - loss: 0.6841 - accuracy: 0.5674 - val_loss: 0.6852 - val_accurac
y: 0.5636
Epoch 9/10
2598/2598 [==============================] - 537s 207ms/step - loss: 0.6841 - accuracy: 0.5674 - val_loss: 0.6850 - val_accurac
y: 0.5636
Epoch 10/10
2598/2598 [==============================] - 527s 203ms/step - loss: 0.6841 - accuracy: 0.5674 - val_loss: 0.6855 - val_accurac
y: 0.5636
812/812 [==============================] - 53s 64ms/step
Accuracy: 0.5610178626424391
```

In [ ]:



f(x) ≤ 5
X²– 4X + 5 ≤ 5
X²– 4X ≤ 0
M = 0.046765 / 3 0L
+ n (C)-n (B∩C)
log_b b^x = x
a(b+c) = ab+ac
126 = 6xy
2x + 2y = 20
(100 ²) a + 100 b
10000 a + 100 b
|a| = |-a|
2¶rh
2¶r (r+h)
¶r²h

Classification algorithms

- Logistic Regression: 0.813%
- Naive Bayes: 0.833%
- K-NN: 0.886%
- Decision Tree: 0.873%
- Neural Network: 0.869%
- Random Forest: 0.894%
- XGBoost: 0.888%
- AdaBoost: 0.895%

THANK YOU FOR YOUR PATIENCE