Customer Satisfaction Rating

“Prediction”

Overview

Created by: Md. Rafiquzzaman Rafi

This project develops a machine learning model to predict customer satisfaction in airlines using advanced techniques like CatBoost and Optuna for hyperparameter optimization.

Data Preprocessing

Exploratory Data Analysis

Feature Engineering

Model Training

Key Insights

Data Import and Basic Analysis

```python
import numpy as np
import pandas as pd
df = pd.read_csv('data/train.csv')
df.head()
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103904 entries, 0 to 103903
Data columns (total 25 columns):
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   Unnamed: 0                        103904 non-null  int64
 1   id                                103904 non-null  int64
 2   Gender                            103904 non-null  object
 3   Customer Type                     103904 non-null  object
 4   Age                               103904 non-null  int64
 5   Type of Travel                    103904 non-null  object
 6   Class                             103904 non-null  object
 7   Flight Distance                   103904 non-null  int64
 8   Inflight wifi service             103904 non-null  int64
 9   Departure/Arrival time convenient 103904 non-null  int64
 10  Ease of Online booking            103904 non-null  int64
 11  Gate location                     103904 non-null  int64
 12  Food and drink                    103904 non-null  int64
 13  Online boarding                   103904 non-null  int64
 14  Seat comfort                      103904 non-null  int64
 15  Inflight entertainment            103904 non-null  int64
 16  On-board service                  103904 non-null  int64
 17  Leg room service                  103904 non-null  int64
 18  Baggage handling                  103904 non-null  int64
 19  Checkin service                   103904 non-null  int64
...
 23  Arrival Delay in Minutes          103594 non-null  float64
 24  satisfaction                      103904 non-null  object
dtypes: float64(1), int64(19), object(5)
memory usage: 19.8+ MB
```

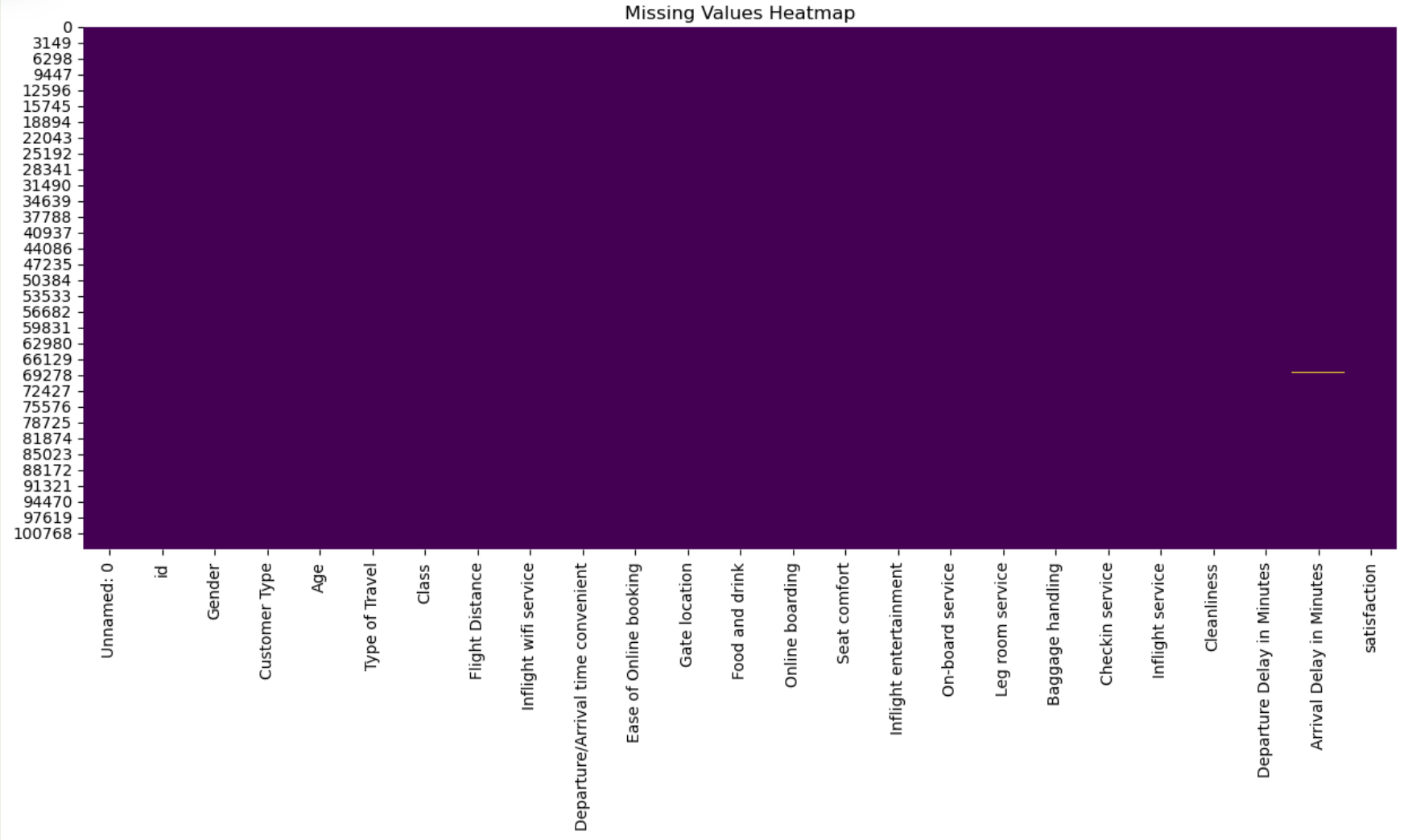
	Unnamed: 0	id	Gender	Customer Type	Age	Type of Travel	Class	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	...	Inflight entertainment	On-board service	Leg room service	Baggage handling	Checkin service	Inflight service	Cleanliness	Departure Delay in Minutes	Arrival Delay in Minutes	satisfaction
0	0	70172	Male	Loyal Customer	13	Personal Travel	Eco Plus	460	3	4	...	5	4	3	4	4	5	5	25	18.0	neutral or dissatisfied
1	1	5047	Male	disloyal Customer	25	Business travel	Business	235	3	2	...	1	1	5	3	1	4	1	1	6.0	neutral or dissatisfied
2	2	110028	Female	Loyal Customer	26	Business travel	Business	1142	2	2	...	5	4	3	4	4	4	5	0	0.0	satisfied
3	3	24026	Female	Loyal Customer	25	Business travel	Business	562	2	5	...	2	2	5	3	1	4	2	11	9.0	neutral or dissatisfied
4	4	119299	Male	Loyal Customer	61	Business travel	Business	214	3	3	...	3	3	4	4	3	3	3	0	0.0	satisfied

5 rows × 25 columns

Data Preprocessing

```python
import matplotlib.pyplot as plt
import seaborn as sns
missing_values = df.isnull().sum()
print(missing_values)
plt.figure(figsize=(15, 6))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.title('Missing Values Heatmap')
plt.show()
```

```
Unnamed: 0                          0
id                                  0
Gender                              0
Customer Type                       0
Age                                 0
Type of Travel                      0
Class                               0
Flight Distance                     0
Inflight wifi service               0
Departure/Arrival time convenient   0
Ease of Online booking              0
Gate location                       0
Food and drink                      0
Online boarding                     0
Seat comfort                        0
Inflight entertainment              0
On-board service                    0
Leg room service                    0
Baggage handling                    0
Checkin service                     0
Inflight service                    0
Cleanliness                         0
Departure Delay in Minutes          0
Arrival Delay in Minutes          310
satisfaction                        0
dtype: int64
```



Data Preprocessing

```python
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

cat_cols = ['Type of Travel', 'Customer Type', 'Class', 'satisfaction']
oh_col = ['Gender']

encoding_dict = {}
le = LabelEncoder()
oh = OneHotEncoder(sparse_output=False, drop='first')

for col in cat_cols:
    df[col] = le.fit_transform(df[col].astype(str))
    encoding_dict[col] = dict(zip(le.classes_, range(len(le.classes_))))

encoded_data = oh.fit_transform(df[oh_col])
encoded_df = pd.DataFrame(encoded_data, columns=oh.get_feature_names_out(oh_col))

df = df.drop(columns=oh_col).reset_index(drop=True)
df = pd.concat([df, encoded_df], axis=1)

print("\nEncoding Mappings:")
for col, mapping in encoding_dict.items():
    print(f"{col}: {mapping}")

df.head()
```

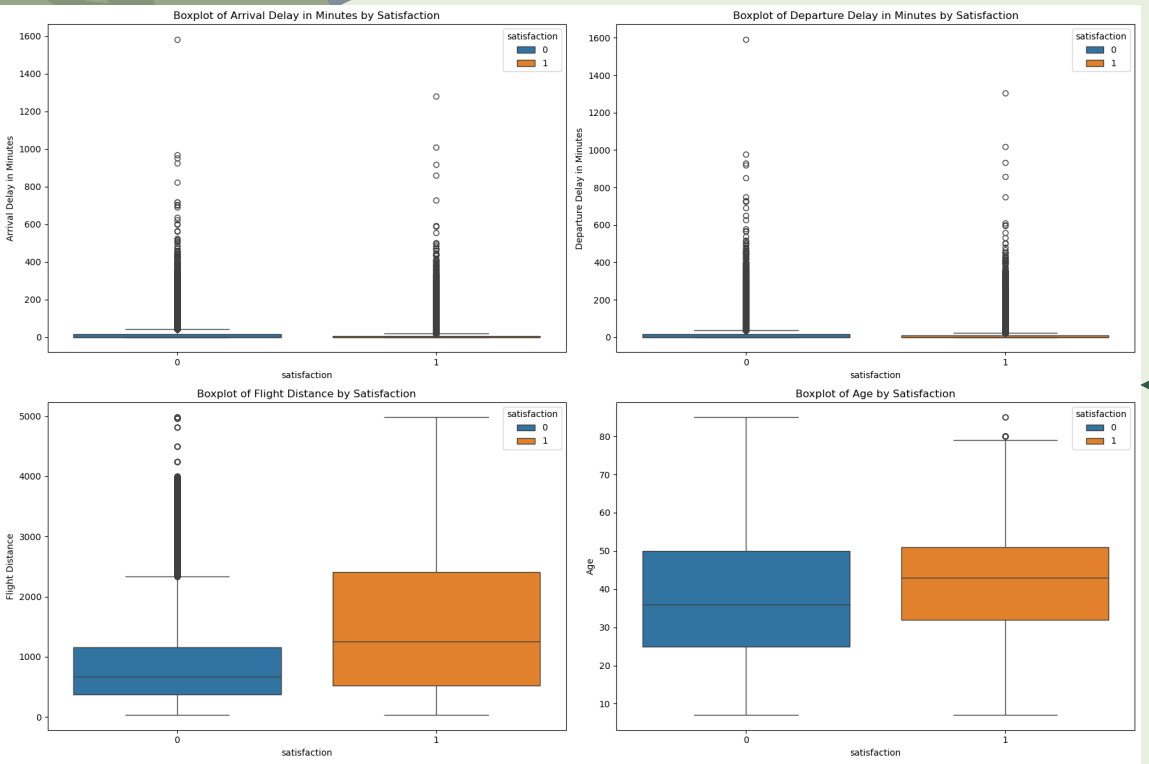
```
Encoding Mappings:
Type of Travel: {'Business travel': 0, 'Personal Travel': 1}
Customer Type: {'Loyal Customer': 0, 'disloyal Customer': 1}
Class: {'Business': 0, 'Eco': 1, 'Eco Plus': 2}
satisfaction: {'neutral or dissatisfied': 0, 'satisfied': 1}
```

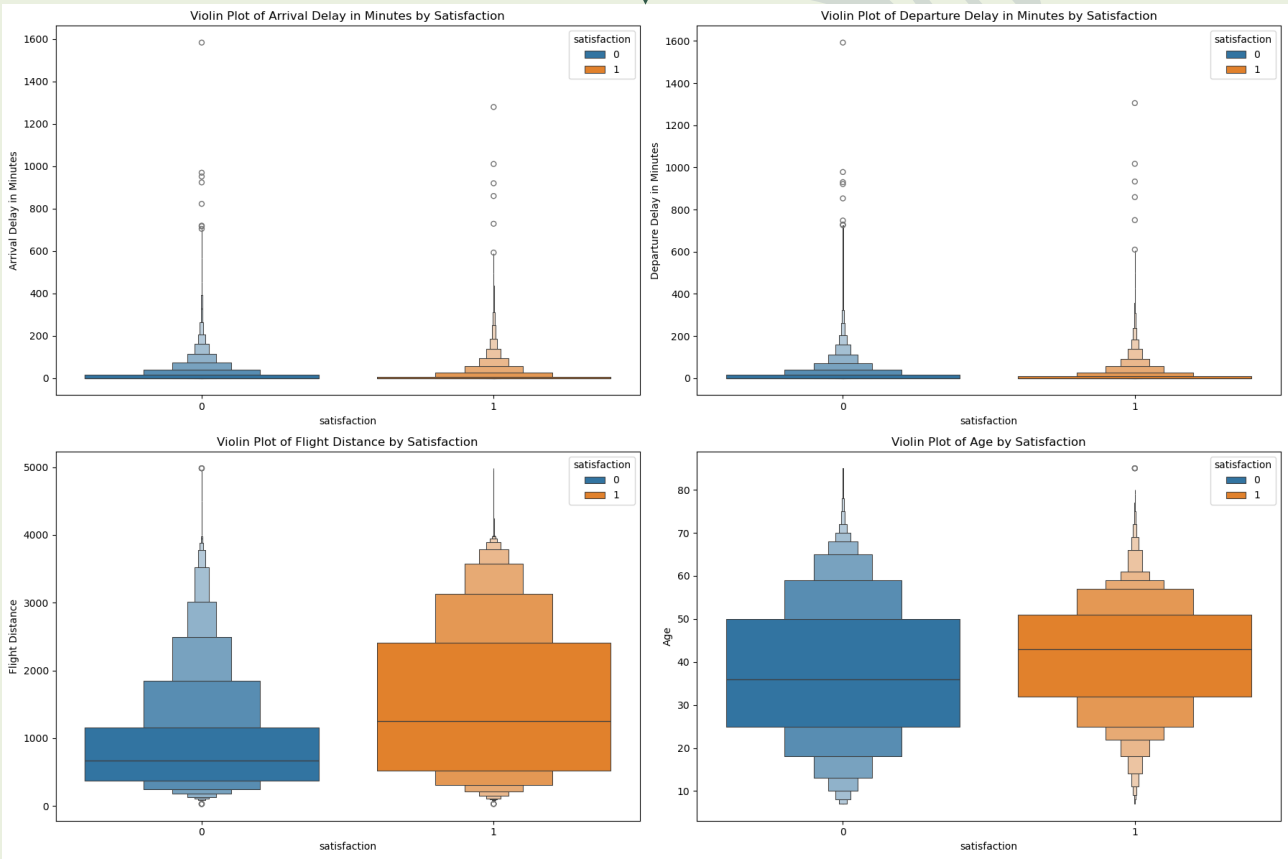
	Customer Type	Age	Type of Travel	Class	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	Food and drink	...	On-board service	Leg room service	Baggage handling	Checkin service	Inflight service	Cleanliness	Departure Delay in Minutes	Arrival Delay in Minutes	satisfaction	Gender_Male
0	0	13	1	2	460	3	4	3	1	5	...	4	3	4	4	5	5	25	18.0	0	1.0
1	1	25	0	0	235	3	2	3	3	1	...	1	5	3	1	4	1	1	6.0	0	1.0
2	0	26	0	0	1142	2	2	2	2	5	...	4	3	4	4	4	5	0	0.0	1	0.0
3	0	25	0	0	562	2	5	5	5	2	...	2	5	3	1	4	2	11	9.0	0	0.0
4	0	61	0	0	214	3	3	3	3	4	...	3	4	4	3	3	3	0	0.0	1	1.0

5 rows × 23 columns

Exploratory Data Analysis

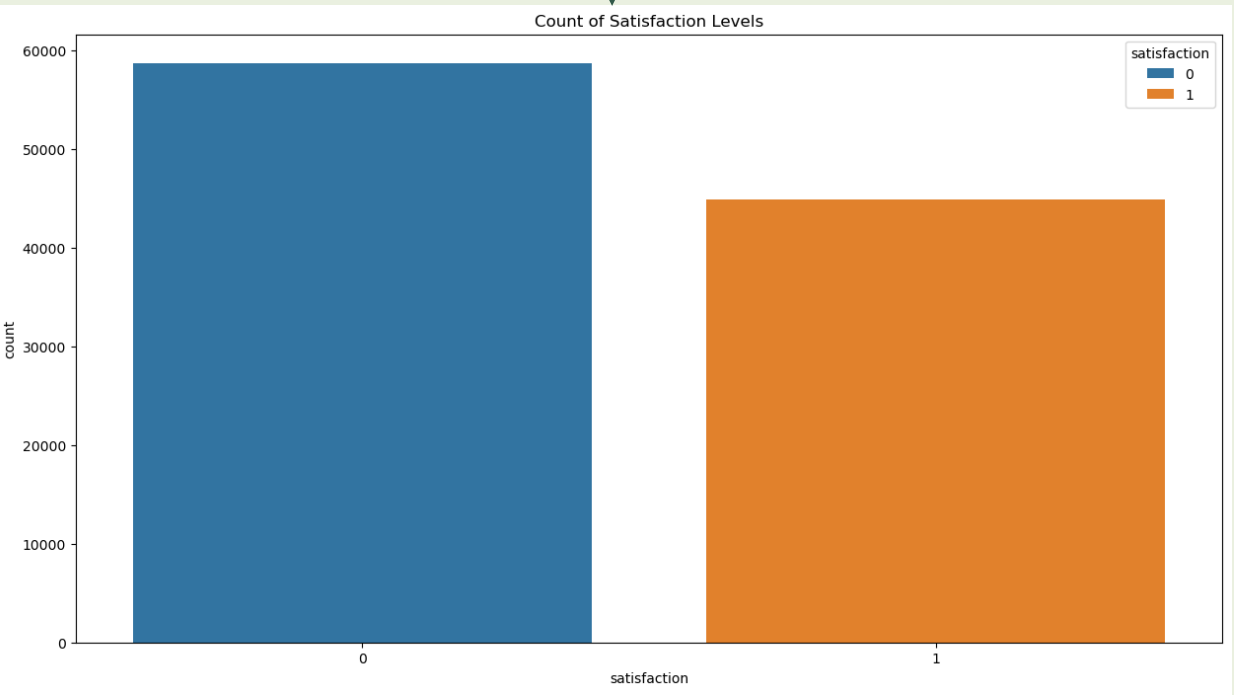
```python
features = df[['Arrival Delay in Minutes', 'Departure Delay in Minutes', 'Flight Distance', 'Age', 'satisfaction']]
plt.figure(figsize=(18, 12))
for i, column in enumerate(features.columns[:-1], 1):
    plt.subplot(2, 2, i)
    sns.boxplot(x='satisfaction', y=column, data=features, hue='satisfaction')
    plt.title(f'Boxplot of {column} by Satisfaction')
plt.tight_layout()
plt.show()

plt.figure(figsize=(18, 12))
for i, column in enumerate(features.columns[:-1], 1):
    plt.subplot(2, 2, i)
    sns.boxenplot(x='satisfaction', y=column, data=features, hue='satisfaction')
    plt.title(f'Violin Plot of {column} by Satisfaction')
plt.tight_layout()
plt.show()
```



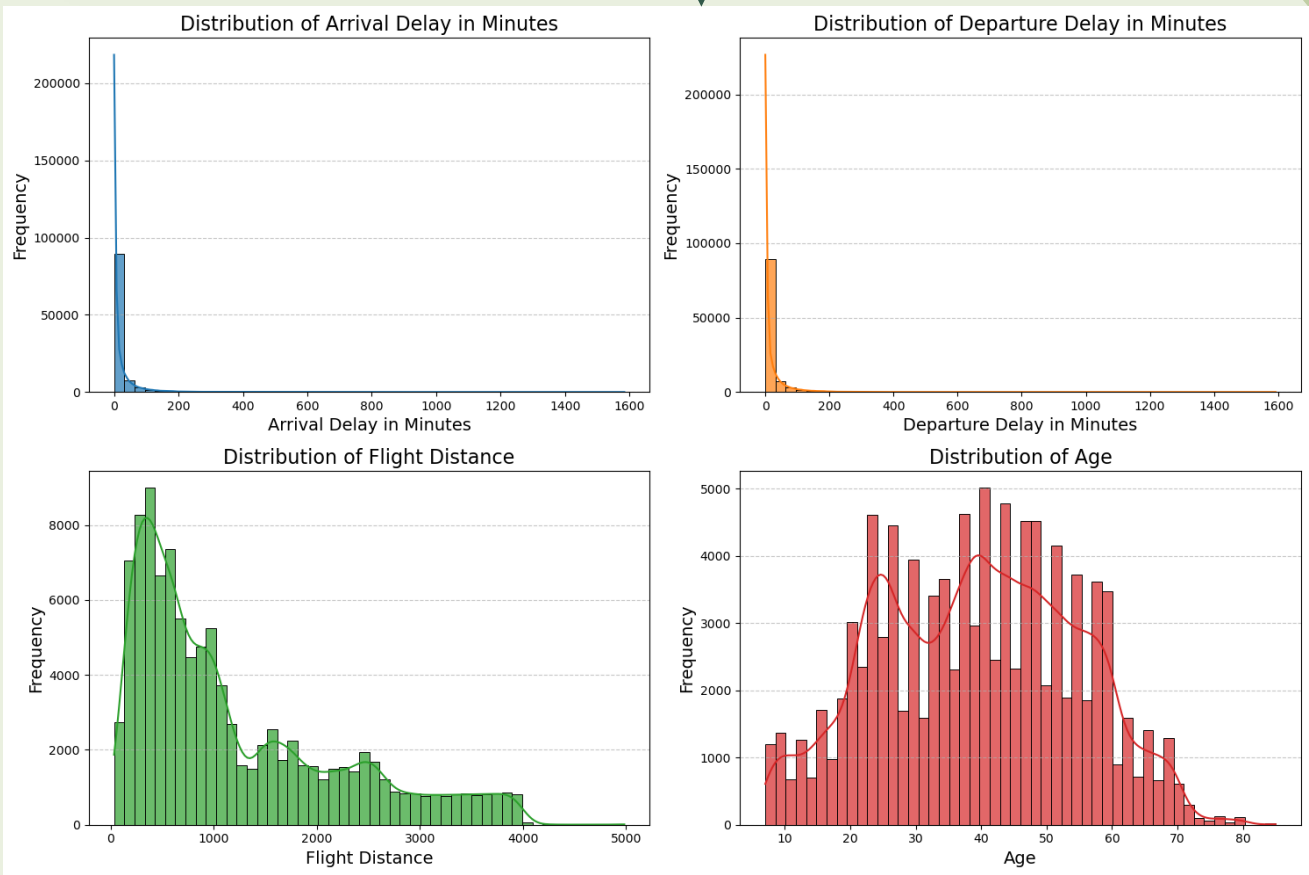


Distribution Analysis

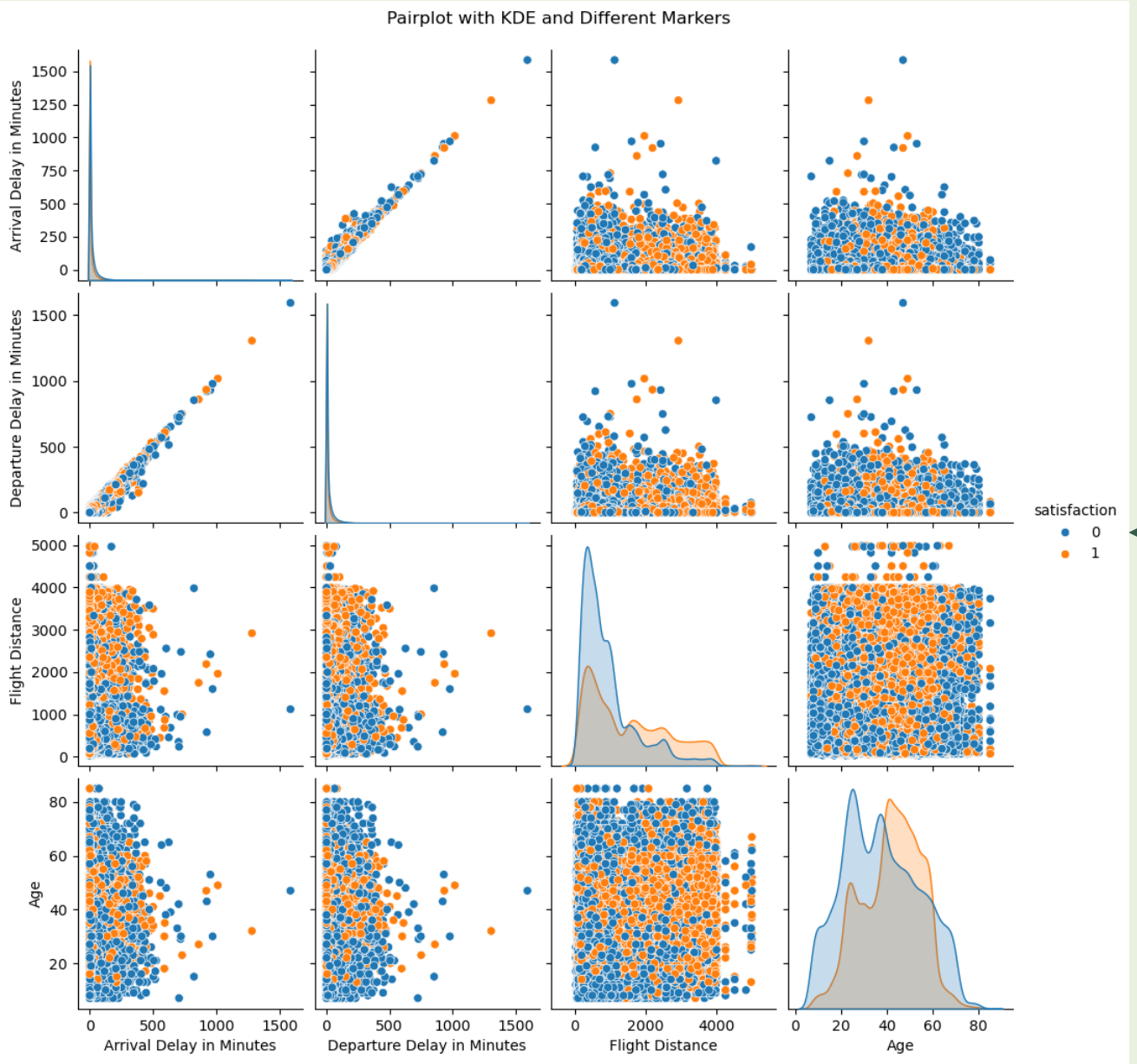
```python
plt.figure(figsize=(15, 8))
sns.countplot(x='satisfaction', data=features, hue='satisfaction')
plt.title('Count of Satisfaction Levels')
plt.show()
```




```python
palette = sns.color_palette(n_colors=len(features.columns)-1)

plt.figure(figsize=(15, 10))
for i, column in enumerate(features.columns[:-1], 1):
    plt.subplot(2, 2, i)
    sns.histplot(features[column], kde=True, bins=50, color=palette[i-1], alpha=0.7)
    plt.title(f'Distribution of {column}', fontsize=16)
    plt.xlabel(column, fontsize=14)
    plt.ylabel('Frequency', fontsize=14)
    plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()
```



Distribution Analysis

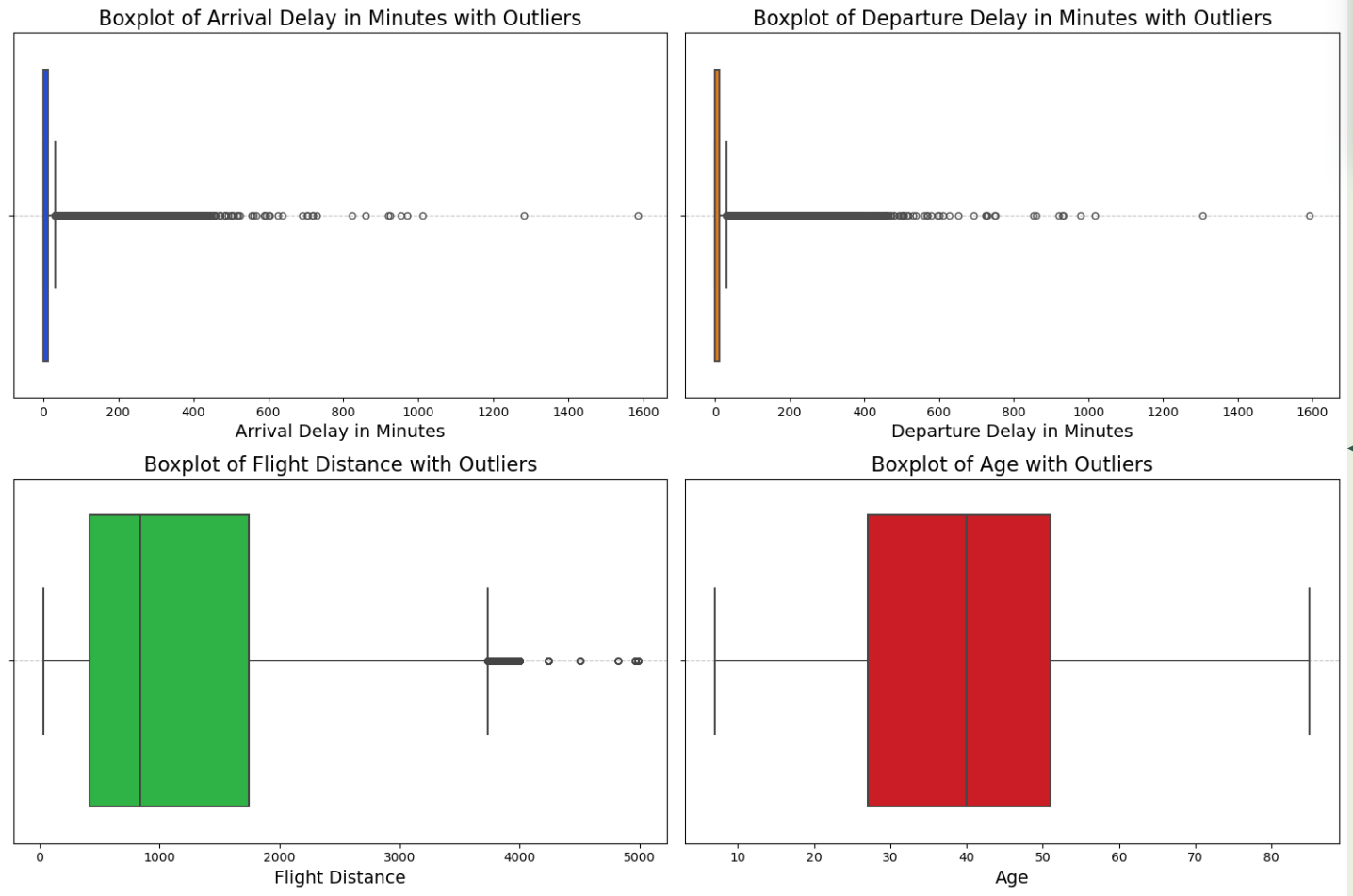



```python
sns.pairplot(features, hue='satisfaction', diag_kind='kde', markers=["o", "o"])
plt.suptitle('Pairplot with KDE and Different Markers', y=1.02)
plt.show()
```

Pair Plot Analysis

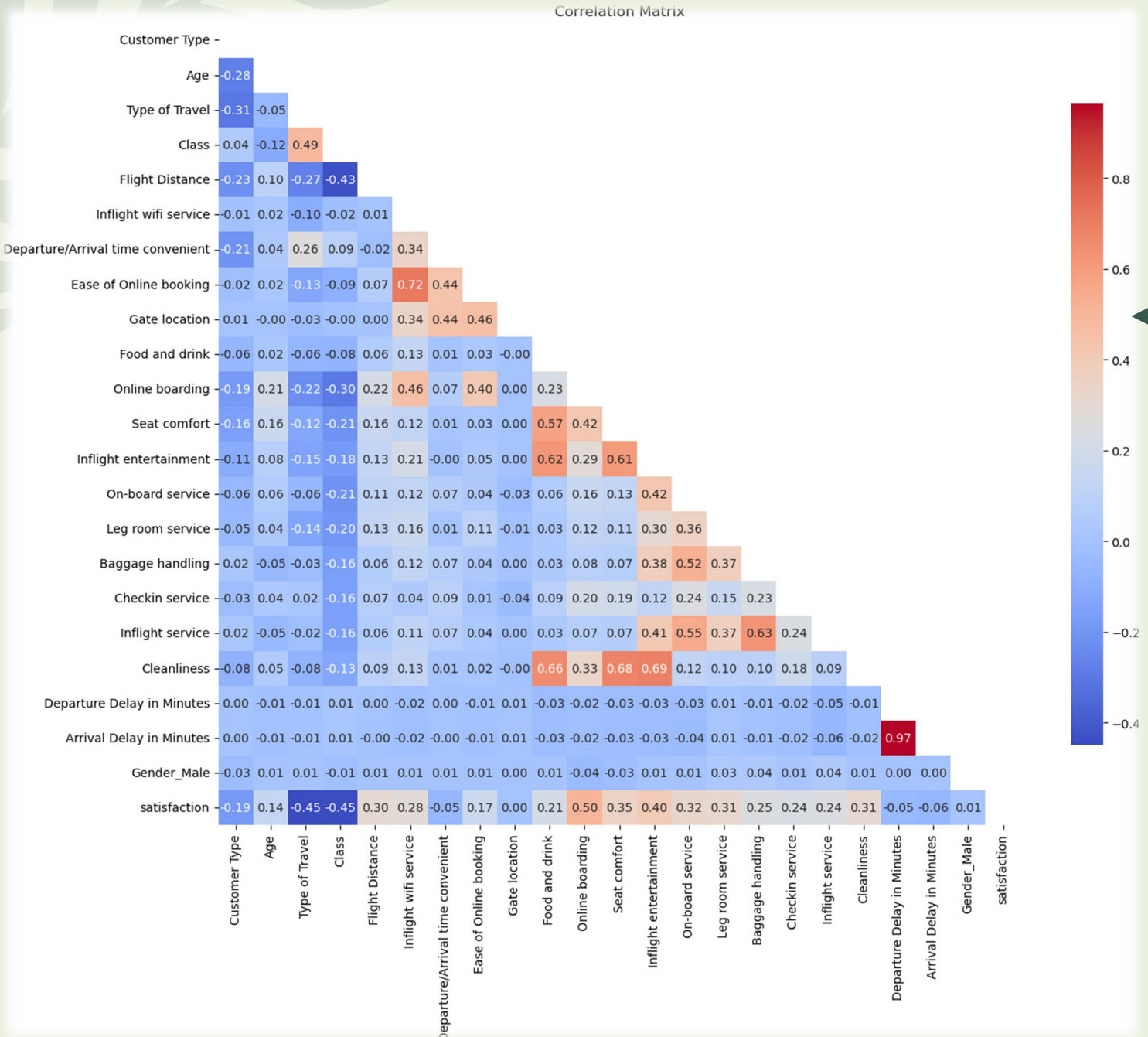
Distribution Analysis

```python
features = df[['Arrival Delay in Minutes', 'Departure Delay in Minutes', 'Flight Distance',
'Age', 'satisfaction']]


palette = sns.color_palette('bright', n_colors=len(features.columns)-1)

Q1 = features.quantile(0.25)
Q3 = features.quantile(0.75)
IQR = Q3 - Q1

outlier_condition = (features < (Q1 - 1.5 * IQR)) | (features > (Q3 + 1.5 * IQR))

plt.figure(figsize=(15, 10))
for i, column in enumerate(features.columns[:-1], 1):
    plt.subplot(2, 2, i)
    sns.boxplot(x=features[column], color=palette[i-1], fliersize=5, linewidth=1.5)
    plt.title(f'Boxplot of {column} with Outliers', fontsize=16)
    plt.xlabel(column, fontsize=14)
    plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()
```



Box Plot Analysis for Outlier Detection

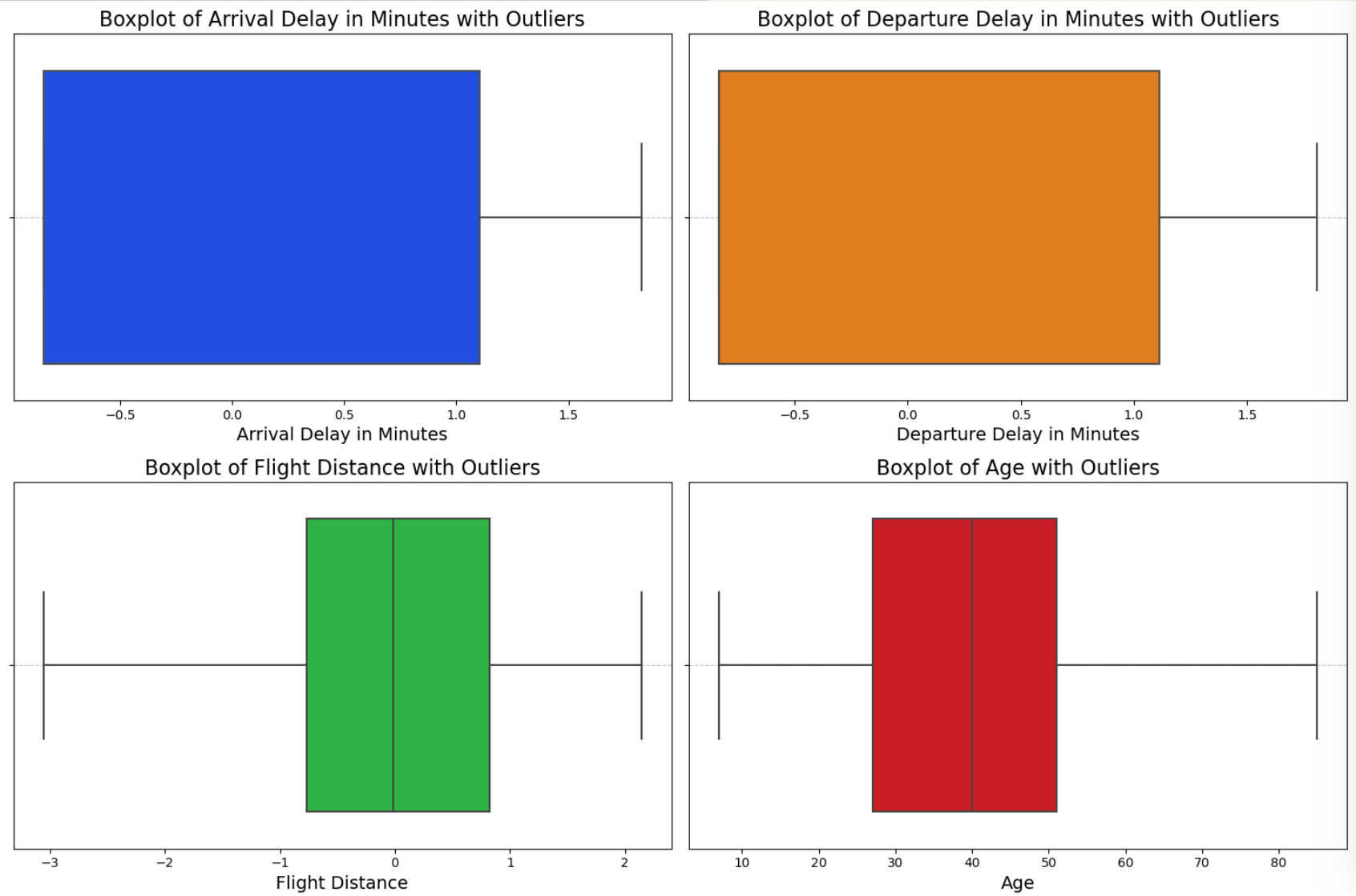
Correlation Analysis




```python
df = df[[col for col in df.columns if col != 'satisfaction'] + ['satisfaction']]

corr = df.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
plt.figure(figsize=(15, 12))
sns.heatmap(corr, mask=mask, cmap='coolwarm', annot=True, fmt=".2f", square=True, cbar_kws={"shrink": .8})
plt.title('Correlation Matrix')
plt.show()
```

Feature Engineering

Yeo-Johnson Transformation for outlier handling




```python
features = df[['Arrival Delay in Minutes', 'Departure Delay in Minutes', 'Flight Distance', 'Age', 'satisfaction']]


palette = sns.color_palette('bright', n_colors=len(features.columns)-1)

Q1 = features.quantile(0.25)
Q3 = features.quantile(0.75)
IQR = Q3 - Q1

outlier_condition = (features < (Q1 - 1.5 * IQR)) | (features > (Q3 + 1.5 * IQR))

plt.figure(figsize=(15, 10))
for i, column in enumerate(features.columns[:-1], 1):
    plt.subplot(2, 2, i)
    sns.boxplot(x=features[column], color=palette[i-1], fliersize=5, linewidth=1.5)
    plt.title(f'Boxplot of {column} with Outliers', fontsize=16)
    plt.xlabel(column, fontsize=14)
    plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()

```

Model Training

Baseline Model Performance

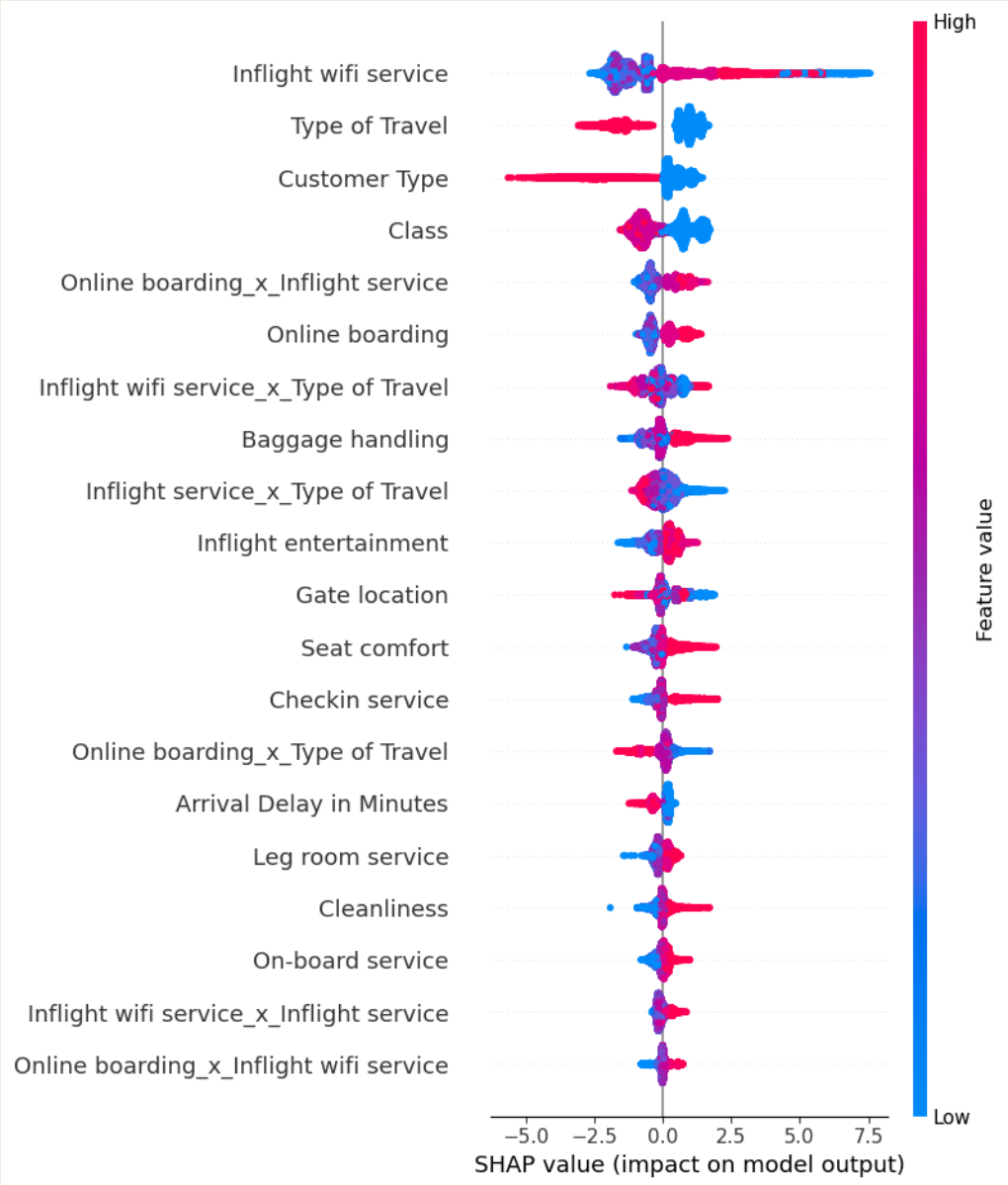
```
Baseline Model Accuracy: 0.8759
Baseline Model ROC AUC: 0.9253

Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.91      0.89     11740
           1       0.87      0.84      0.85      8979

    accuracy                           0.88     20719
   macro avg       0.88      0.87      0.87     20719
weighted avg       0.88      0.88      0.88     20719
```

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, roc_auc_score

X = df.drop(columns=['satisfaction'])
y = df['satisfaction']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

baseline_model = LogisticRegression(random_state=42, max_iter=100000)
baseline_model.fit(X_train, y_train)

y_pred = baseline_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, baseline_model.predict_proba(X_test)[:, 1])
report = classification_report(y_test, y_pred)

print(f"Baseline Model Accuracy: {accuracy:.4f}")
print(f"Baseline Model ROC AUC: {roc_auc:.4f}")
print("\nClassification Report:")
print(report)
```

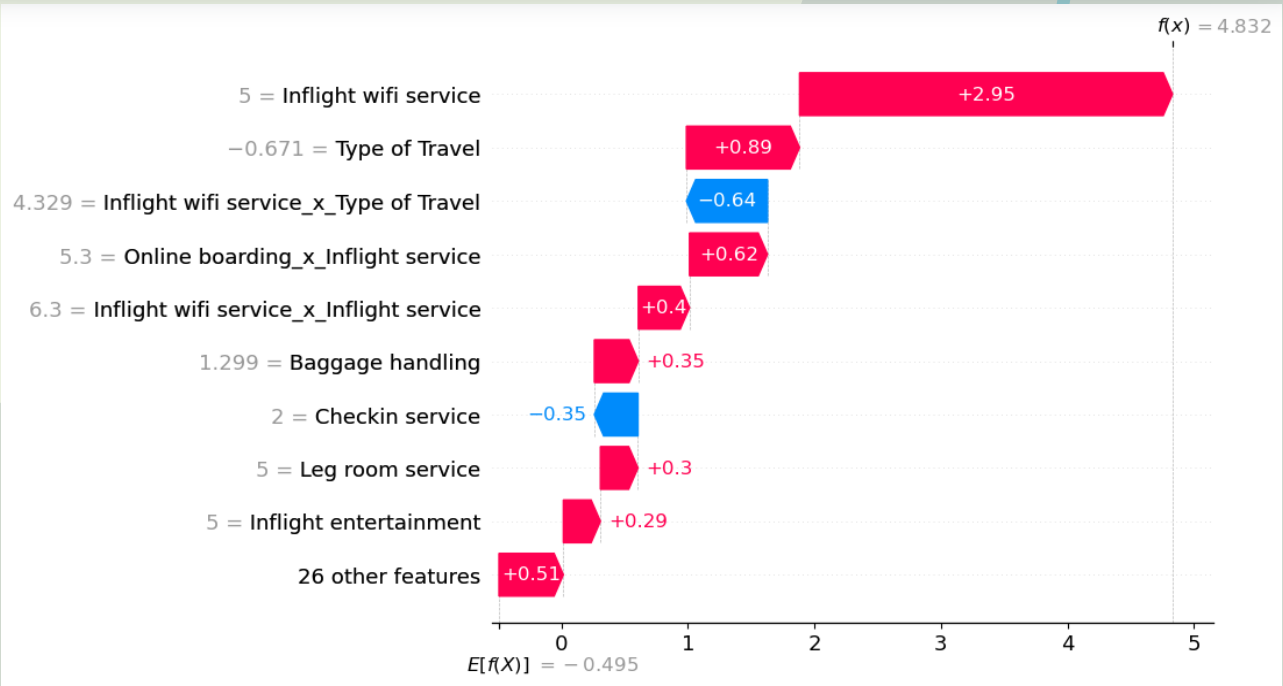
Model Comparison

Model	Accuracy	ROC AUC	R2 Score
CatBoost	0.965346	0.963122	0.858877
XGBoost	0.963801	0.961667	0.852588
LightGBM	0.963077	0.960531	0.849639
Logistic Regression	0.8759	0.9253	0.772895

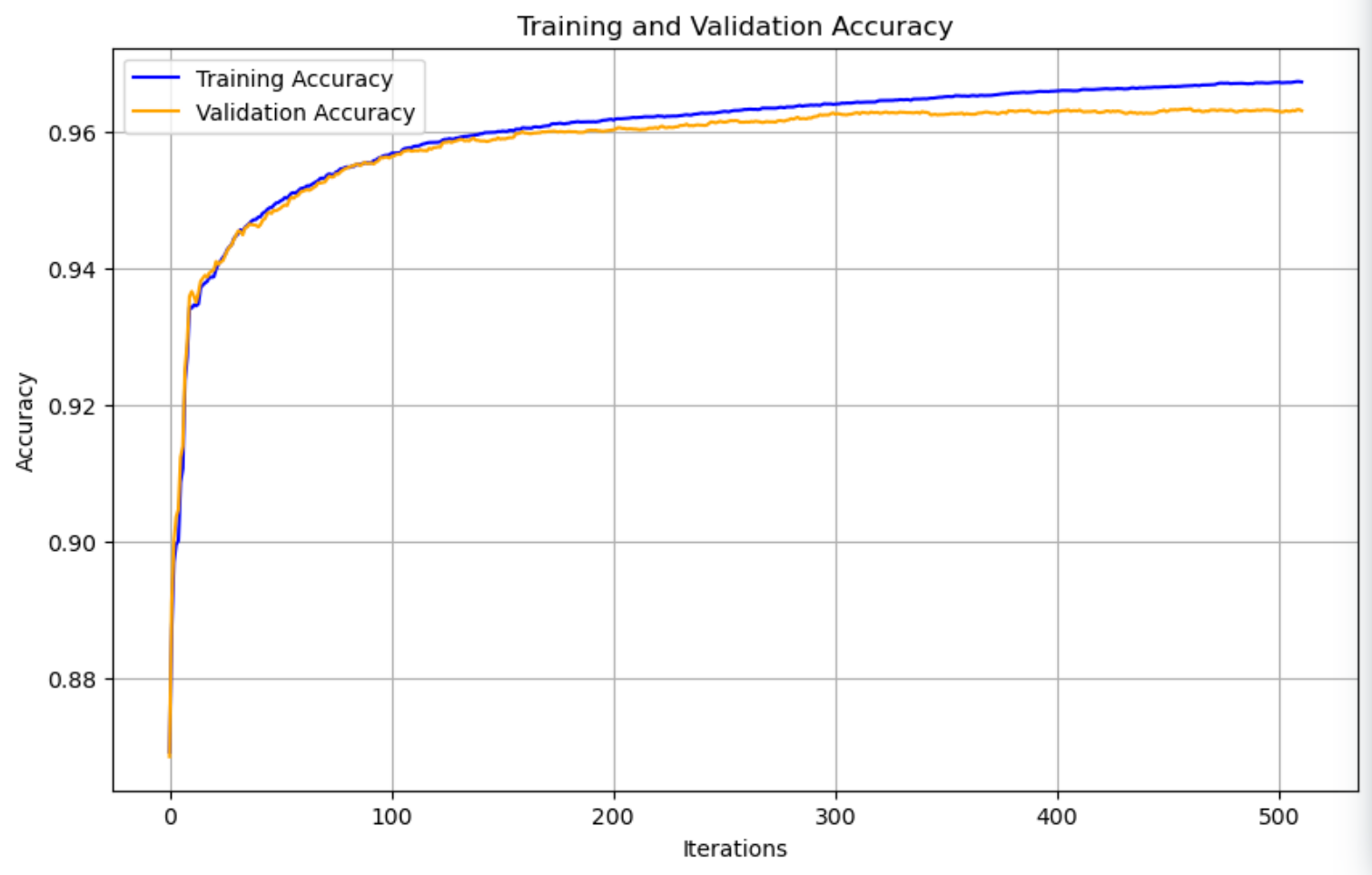
SHAP Explanation




```python
import shap
from catboost import CatBoostClassifier, Pool

categorical_cols = df.select_dtypes(include=["object", "category"]).columns.tolist()

for col in categorical_cols:
    df[col] = df[col].astype(str)

from sklearn.model_selection import train_test_split
X = df.drop(columns=['satisfaction'])
y = df['satisfaction']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

train_pool = Pool(X_train, label=y_train, cat_features=categorical_cols)
test_pool = Pool(X_test, label=y_test, cat_features=categorical_cols)

model = CatBoostClassifier(iterations=500, learning_rate=0.05, depth=8, verbose=200)
model.fit(train_pool)

explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)

shap.summary_plot(shap_values, X_test)

shap.plots.waterfall(shap.Explanation(values=shap_values[0], base_values=explainer.expected_value, data=X_test.iloc[0]))
```



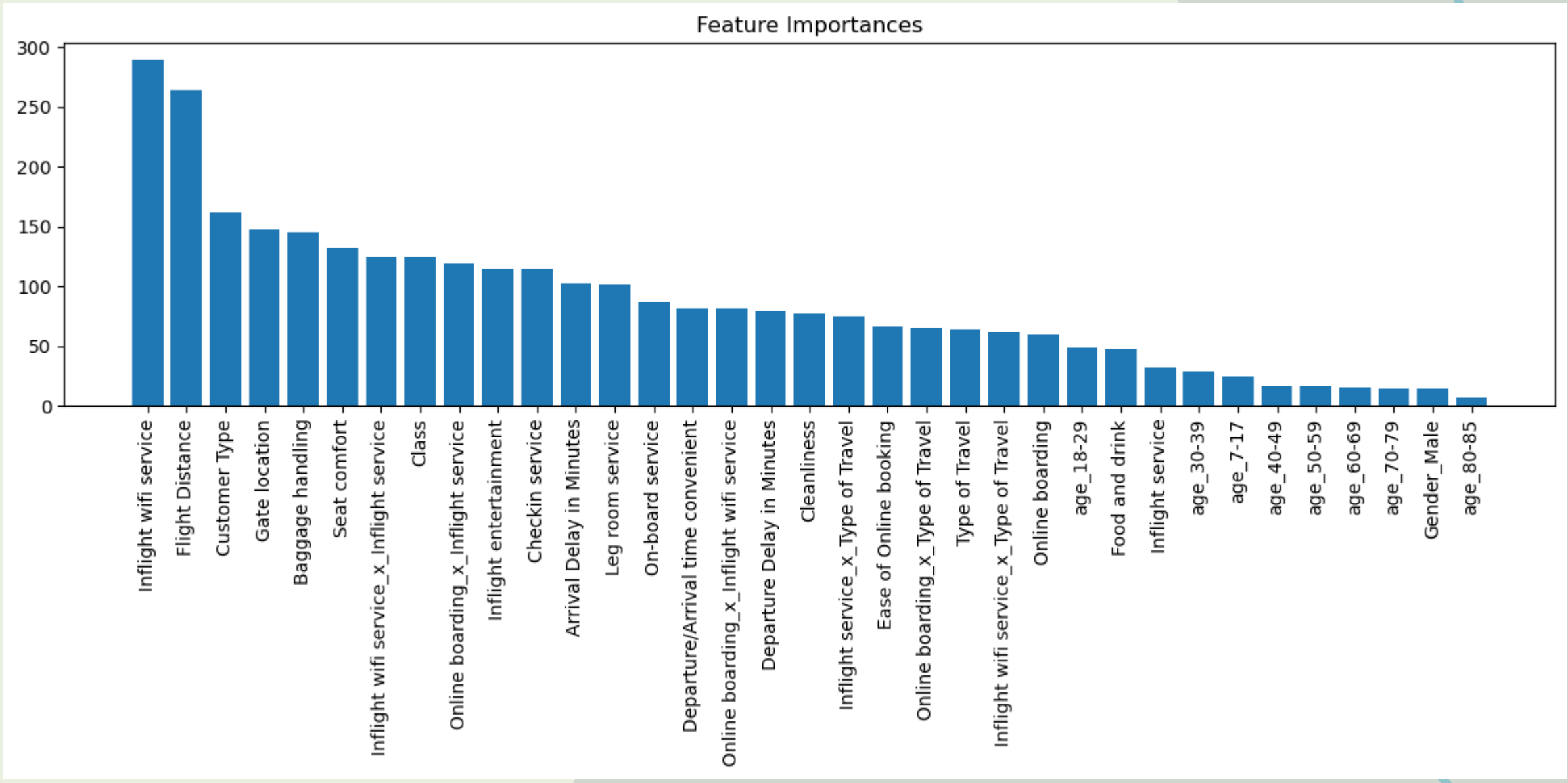
Train & Validation Accuracy




```python
from catboost import CatBoostClassifier
from sklearn.metrics import accuracy_score, classification_report
import joblib

# Parameters found from Optuna tuning
cb_params = {'depth': 4, 'learning_rate': 0.17244137403542156, 'l2_leaf_reg': 3.258820179105591, 'subsample': 0.7977884243064802}

model = CatBoostClassifier(
    **cb_params,
    loss_function="Logloss",
    eval_metric="Accuracy",
    random_seed=42,
    verbose=200
)

model.fit(train_pool, eval_set=test_pool, early_stopping_rounds=50, use_best_model=True)

evals_result = model.get_evals_result()

plt.figure(figsize=(10, 6))
plt.plot(evals_result['learn']['Accuracy'], label='Training Accuracy', color='blue')
plt.plot(evals_result['validation']['Accuracy'], label='Validation Accuracy', color='orange')
plt.title('Training and Validation Accuracy')
plt.xlabel('Iterations')
plt.ylabel('Accuracy')
plt.legend()
plt.grid()
plt.show()

y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
joblib.dump(model, 'models/cat_model.pkl')
```

After optimising using Optuna

Feature Importance



Key Insights

Most Influential Features

- **Inflight wifi service**: Highest overall impact
- **Type of Travel**: Significant variability in satisfaction
- **Customer Type**: Strong influence on rating

Impact Dynamics

Positive Influences:

- ✓ Class
- ✓ Online boarding
- ✓ Inflight entertainment

Negative Influences:

- ✓ Arrival Delay (consistently reduces satisfaction)
- ✓ Inconsistent wifi service

Recommendations

1. **Prioritize Wifi Quality**
    1. Consistent, high-speed internet
    2. Uniform service across different travel types
2. **Minimize Arrival Delays**
    1. Improve scheduling
    2. Enhance operational efficiency

1. **Personalized Customer Experience**
    1. Tailor services to different customer types
    2. Segment-specific satisfaction strategies
2. **Continuous Improvement**
    1. Monitor and enhance secondary features
    2. Holistic approach to service quality

Md. Rafiquzzaman Rafi
rafiquzzamanrafi100@gmail.com

Thanks