MP2-Passenger_Flight

November 16, 2023

1 Instructions:

1. Create an overview of the problem being solved, e.g., what was the story behind the collection of the data, description of the attributes/features used,etc.
2. (Data Preprocessing and Exploratory Analysis) Present descriptive statistics as applicable (e.g., distribution, central tendency, variability) of the data before training the models. Clean the data if there are missing values, etc. You may perform feature engineering (i.e., creating new features out of the given features), but be sure to document your justifications.
3. Split your data into proportions of 70% training set and 30% testing set.
4. Train the following models: (a) logistic regression classifier and (b) naive Bayes classifier on the dataset.
5. Evaluate the performance of the trained model. You may use additional performance measures if you want, but for now I will only require the calculation of the accuracy. The accuracy measures the fraction of correct classifications. With this, you need to generate the confusion matrix. You may read this if you haven't encountered this concept before: https://www.sciencedirect.com/topics/engineering/confusion-matrix#:~:text=A%20confusion%20matrix%20represents%20the,by%20model%20as%20other%20class. Remember to compute this matrix from the test set (not the training set).

[31]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
```

[32]:
```
warnings.filterwarnings("ignore")
```

2 4. Passenger Flight

URL: https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction/data

[33]:

```
dataset = pd.read_csv("passenger_flight.csv")
np.random.seed(123)
dataset = dataset.sample(frac=1).reset_index(drop=True)
display(dataset.head(5))
display(dataset.tail(5))
display(dataset.info())
```

```
   Gender  Customer Type  Age  Type of Travel  Class  Flight Distance  \
0       1              1   53               0      1              946
1       1              1   55               1      1             1620
2       1              1   29               1      1             1400
3       1              1   49               1      1              850
4       1              1   42               1      1             3535

   Inflight wifi service  Departure/Arrival time convenient  \
0                      1                                  1
1                      3                                  3
2                      5                                  2
3                      5                                  5
4                      5                                  5

   Ease of Online booking  Gate location  …  Inflight entertainment  \
0                       1              2  …                       2
1                       3              3  …                       4
2                       2              2  …                       5
3                       5              5  …                       2
4                       1              5  …                       5

   On-board service  Leg room service  Baggage handling  Checkin service  \
0                 2                 5                 2                4
1                 4                 4                 4                3
2                 4                 2                 2                5
3                 2                 3                 2                5
4                 5                 5                 5                3

   Inflight service  Cleanliness  Departure Delay in Minutes  \
0                 3            2                           0
1                 4            4                           0
2                 2            5                           0
3                 2            3                           0
4                 5            3                           5

   Arrival Delay in Minutes  satisfaction
0                       0.0             0
1                       0.0             1
2                       0.0             1
3                       0.0             1
4                       0.0             1
```

```
[5 rows x 23 columns]
       Gender  Customer Type  Age  Type of Travel  Class  Flight Distance  \
25971       0              1   44               1      1             3540
25972       1              0   33               1      0              422
25973       1              1   64               0      0              163
25974       0              1   55               0      1              261
25975       1              1   61               1      1              189

       Inflight wifi service  Departure/Arrival time convenient  \
25971                      1                                  1
25972                      1                                  1
25973                      3                                  5
25974                      3                                  2
25975                      4                                  4

       Ease of Online booking  Gate location  …  Inflight entertainment  \
25971                       1              1  …                       5
25972                       1              3  …                       5
25973                       3              2  …                       2
25974                       0              3  …                       3
25975                       4              4  …                       3

       On-board service  Leg room service  Baggage handling  Checkin service  \
25971                 5                 5                 5                1
25972                 1                 2                 3                3
25973                 4                 4                 4                3
25974                 3                 0                 3                1
25975                 3                 3                 4                2

       Inflight service  Cleanliness  Departure Delay in Minutes  \
25971                 5            4                           0
25972                 4            5                           0
25973                 4            2                           0
25974                 3            1                           0
25975                 3            2                          72

       Arrival Delay in Minutes  satisfaction
25971                       0.0             1
25972                       0.0             0
25973                       9.0             0
25974                       0.0             0
25975                      70.0             0

[5 rows x 23 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25976 entries, 0 to 25975
```

```
Data columns (total 23 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Gender                             25976 non-null  int64
 1   Customer Type                      25976 non-null  int64
 2   Age                                25976 non-null  int64
 3   Type of Travel                     25976 non-null  int64
 4   Class                              25976 non-null  int64
 5   Flight Distance                    25976 non-null  int64
 6   Inflight wifi service              25976 non-null  int64
 7   Departure/Arrival time convenient  25976 non-null  int64
 8   Ease of Online booking             25976 non-null  int64
 9   Gate location                      25976 non-null  int64
 10  Food and drink                     25976 non-null  int64
 11  Online boarding                    25976 non-null  int64
 12  Seat comfort                       25976 non-null  int64
 13  Inflight entertainment             25976 non-null  int64
 14  On-board service                   25976 non-null  int64
 15  Leg room service                   25976 non-null  int64
 16  Baggage handling                   25976 non-null  int64
 17  Checkin service                    25976 non-null  int64
 18  Inflight service                   25976 non-null  int64
 19  Cleanliness                        25976 non-null  int64
 20  Departure Delay in Minutes         25976 non-null  int64
 21  Arrival Delay in Minutes           25893 non-null  float64
 22  satisfaction                       25976 non-null  int64
dtypes: float64(1), int64(22)
memory usage: 4.6 MB
```

```
None
```

2.1 Data Preprocessing

- Features will be uniformed
- Whitespaces in features will be replaced by '_'
- If there are any missing values, it will be replaced by the mean of its corresponding feature.

[34]:
```python
# Column names cleaning
dataset.columns = [c.replace(' ', '_') for c in dataset.columns]

# Find any missing values
def calculate_missing_values(data):
    total_missing = data.isnull().sum()

    missing_data = pd.DataFrame({
        'Total Missing': total_missing,
    })
    return missing_data
```

```
calculate_missing_values(dataset)
```

[34]:

```
                                  Total Missing
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                     83
satisfaction                                  0
```

[55]:

```
# Replace missing values with its column's mean
dataset['Arrival_Delay_in_Minutes'] = dataset['Arrival_Delay_in_Minutes'].
 ↪fillna(dataset['Arrival_Delay_in_Minutes'].mean())
calculate_missing_values(dataset)
```

[55]:

```
                                  Total Missing
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              0
satisfaction                          0
```

2.2 Exploratory Analysis

- Here we try to gain some basic understanding of our dataset

[39]:
```
dataset.describe()
```

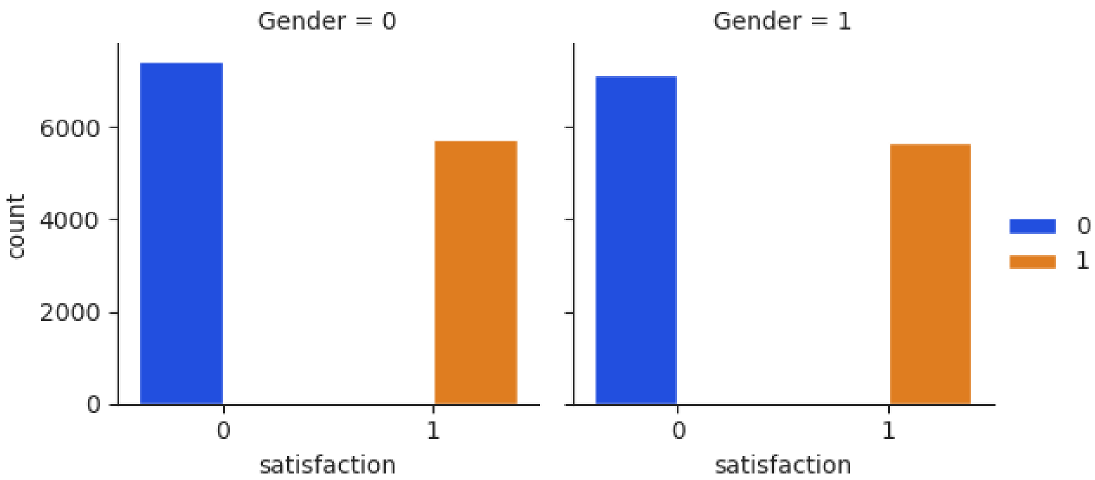
[39]:
```
             Gender  Customer_Type           Age  Type_of_Travel  \
count  25976.000000   25976.000000  25976.000000    25976.000000
mean       0.492917       0.815253     39.620958        0.694410
std        0.499959       0.388100     15.135685        0.460666
min        0.000000       0.000000      7.000000        0.000000
25%        0.000000       1.000000     27.000000        0.000000
50%        0.000000       1.000000     40.000000        1.000000
75%        1.000000       1.000000     51.000000        1.000000
max        1.000000       1.000000     85.000000        1.000000

              Class  Flight_Distance  Inflight_wifi_service  \
count  25976.000000     25976.000000           25976.000000
mean       0.554820      1193.788459               2.724746
std        0.496995       998.683999               1.335384
min        0.000000        31.000000               0.000000
25%        0.000000       414.000000               2.000000
50%        1.000000       849.000000               3.000000
75%        1.000000      1744.000000               4.000000
max        1.000000      4983.000000               5.000000

       Departure/Arrival_time_convenient  Ease_of_Online_booking  \
count                       25976.000000            25976.000000
mean                            3.046812                2.756775
std                             1.533371                1.412951
min                             0.000000                0.000000
25%                             2.000000                2.000000
50%                             3.000000                3.000000
75%                             4.000000                4.000000
max                             5.000000                5.000000
```

```
       Gate_location  …  Inflight_entertainment  On-board_service  \
count   25976.000000  …            25976.000000      25976.000000
mean        2.977094  …                3.357753          3.385664
std         1.282133  …                1.338299          1.282088
min         1.000000  …                0.000000          0.000000
25%         2.000000  …                2.000000          2.000000
50%         3.000000  …                4.000000          4.000000
75%         4.000000  …                4.000000          4.000000
max         5.000000  …                5.000000          5.000000

       Leg_room_service  Baggage_handling  Checkin_service  Inflight_service  \
count      25976.000000      25976.000000     25976.000000      25976.000000
mean           3.350169          3.633238         3.314175          3.649253
std            1.318862          1.176525         1.269332          1.180681
min            0.000000          1.000000         1.000000          0.000000
25%            2.000000          3.000000         3.000000          3.000000
50%            4.000000          4.000000         3.000000          4.000000
75%            4.000000          5.000000         4.000000          5.000000
max            5.000000          5.000000         5.000000          5.000000

        Cleanliness  Departure_Delay_in_Minutes  Arrival_Delay_in_Minutes  \
count  25976.000000                 25976.00000              25893.000000
mean       3.286226                    14.30609                 14.740857
std        1.319330                    37.42316                 37.517539
min        0.000000                     0.00000                  0.000000
25%        2.000000                     0.00000                  0.000000
50%        3.000000                     0.00000                  0.000000
75%        4.000000                    12.00000                 13.000000
max        5.000000                  1128.00000               1115.000000

       satisfaction
count  25976.000000
mean       0.438982
std        0.496272
min        0.000000
25%        0.000000
50%        0.000000
75%        1.000000
max        1.000000

[8 rows x 23 columns]
```

2.2.1 We will do some more visualization of the data to understand each feature

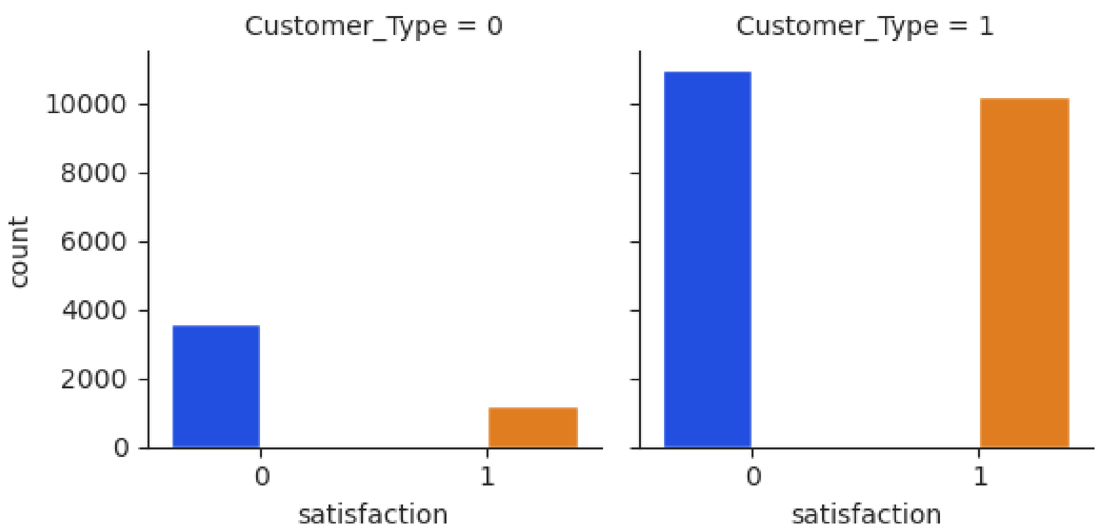
[40]:
```
dataset['satisfaction'] = dataset['satisfaction'].astype(str)
```

[48]:
```
# Gender vs Satisfaction
with sns.axes_style(style='ticks'):
    g = sns.catplot(x="satisfaction", col="Gender", col_wrap=2, data=dataset,␣
 ↪kind="count", height=3, aspect=1.0, palette="bright", hue="satisfaction",␣
 ↪legend=False)
    g.add_legend()
```



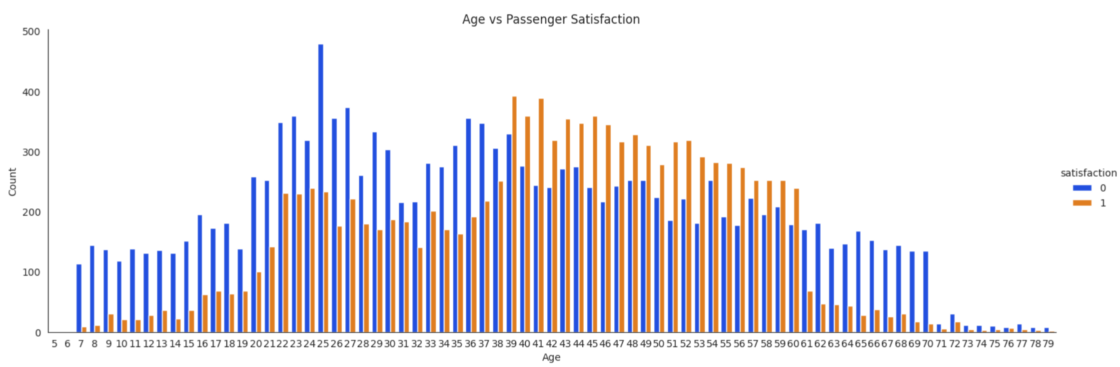
It can be observed that the distribution of satisfied and dissatisfied are quite the same between both genders. Dissatisfied customers are higher in number compared to the satisfied customers.

[49]:
```
# Customer Type vs Satisfaction
with sns.axes_style(style='ticks'):
    g = sns.catplot(x="satisfaction", col="Customer_Type", col_wrap=2,␣
 ↪data=dataset, kind="count", height=3, aspect=1.0, palette="bright",␣
 ↪hue="satisfaction", legend=False)
```



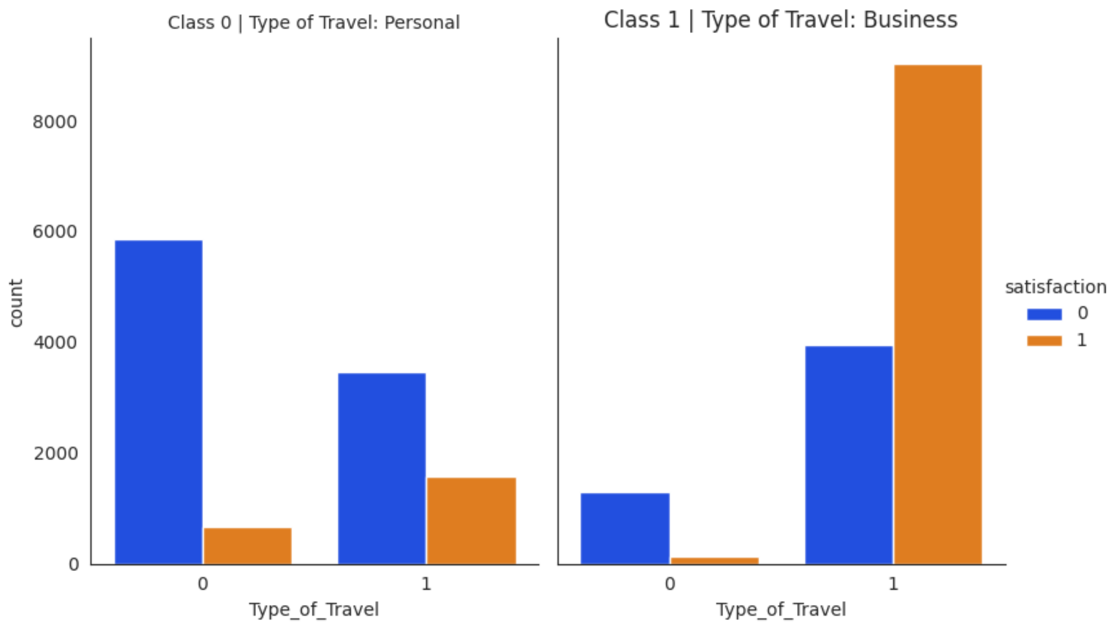
There is more loyal customers compared to the disloyal customers, which gives us the insight that most of the people in the sample are people that have already flown before. We can also see that the proportionality of satisfaction to customer type is significantly high.

[58]:
```
# Age vs Satisfaction
with sns.axes_style('white'):
    g = sns.catplot(x="Age", data=dataset, aspect=3.0, kind='count',␣
 ↪hue='satisfaction', order=range(5, 80), palette="bright")
    g.set_axis_labels('Age', 'Count')
    g.set(title='Age vs Passenger Satisfaction')
```



[59]:
```
# Type of Travel, Class vs Satisfaction
with sns.axes_style('white'):
    g = sns.catplot(x="Type_of_Travel", hue="satisfaction", col="Class",␣
 ↪data=dataset, kind="count", height=5, aspect=.8, palette="bright")
```

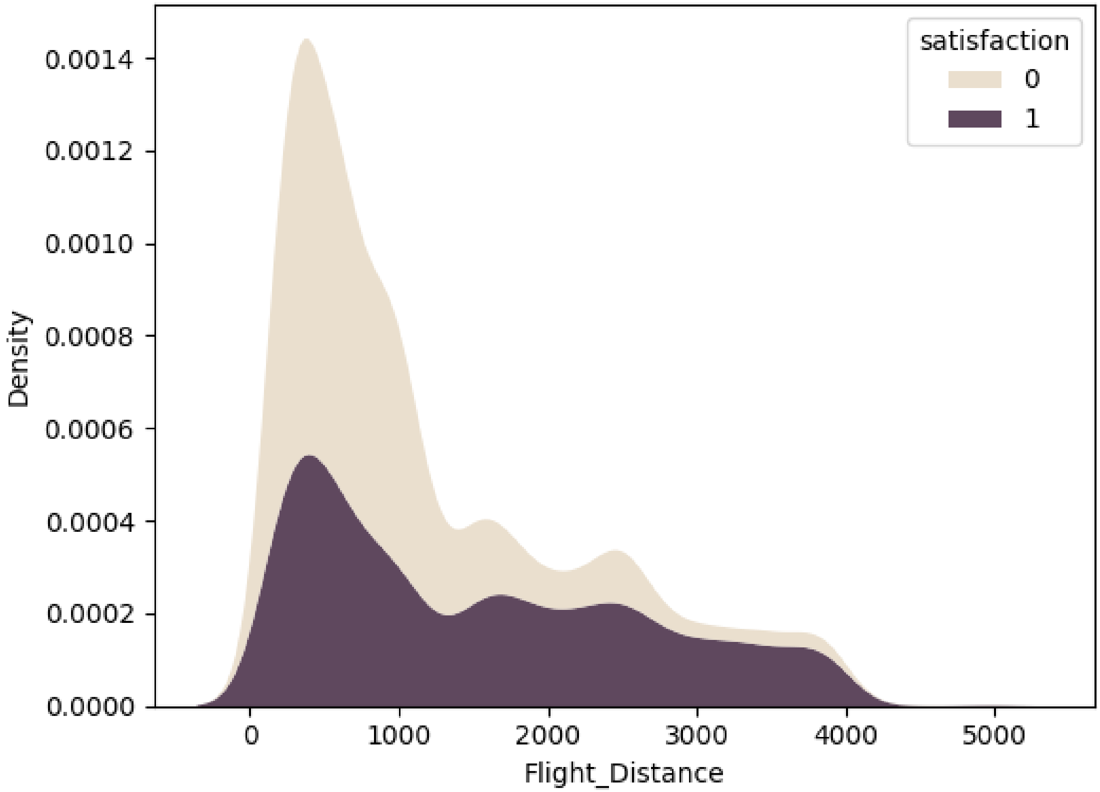
```
    g.set_titles("Class {col_name} | Type of Travel: Personal")
    g.axes[0, 1].set_title("Class 1 | Type of Travel: Business")
```



Personal Travel has more dissatisfied customers compared to business travels. Business travels has more satisfied customers.

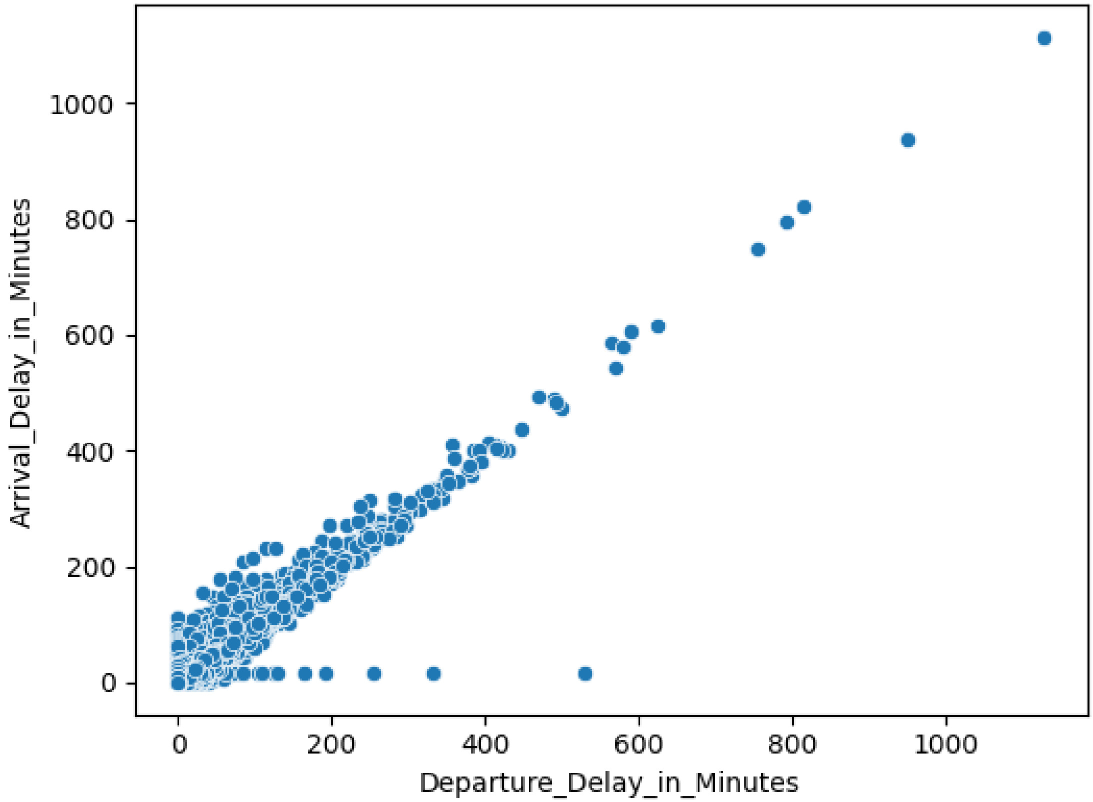
[60]:
```
# Flight Distance vs Satisfaction
sns.kdeplot(data = dataset, x = "Flight_Distance",hue = "satisfaction" , shade␣
 ↪= True,palette="ch:.25",multiple="stack",fill=True, common_norm=False,alpha=.
 ↪8, linewidth=0)
```

[60]: <Axes: xlabel='Flight_Distance', ylabel='Density'>



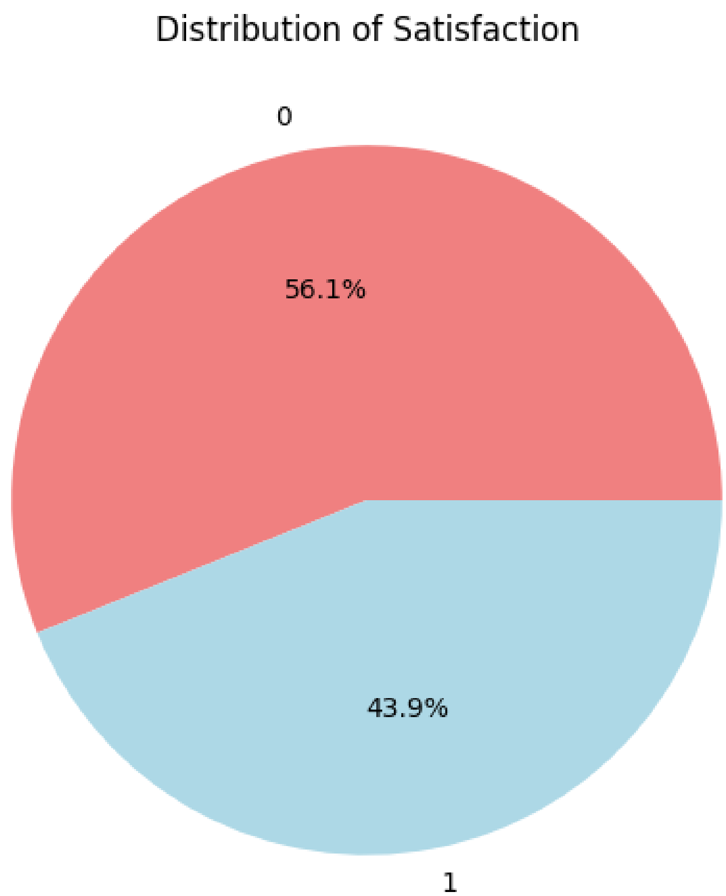
[61]:
```
# Delay
sns.scatterplot(x = 'Departure_Delay_in_Minutes', y =␣
 ↪'Arrival_Delay_in_Minutes', data = dataset,palette="ch:.25")
```

[61]: <Axes: xlabel='Departure_Delay_in_Minutes', ylabel='Arrival_Delay_in_Minutes'>



[62]:
```
# Satisfaction
satisfaction_counts = dataset['satisfaction'].value_counts()

# Plotting the pie chart
plt.figure(figsize=(6, 6))
plt.pie(satisfaction_counts, labels=satisfaction_counts.index, autopct='%1.
 ↪1f%%', colors=['lightcoral', 'lightblue'])
plt.title('Distribution of Satisfaction')
```

[62]: Text(0.5, 1.0, 'Distribution of Satisfaction')



Gender: It can be observed that the distribution of satisfied and dissatisfied are quite the same between both genders. Dissatisfied customers are higher in number compared to the satisfied customers. **Customer Type:** Loyal customers are higher in number compared to the disloyal customers. **Age:** In age groups 39-60, there are more satisfied customers than the dissatisfied customers. **Type of Travel & Class:** Personal Travel has more dissatisfied customers compared to business travels. Business travels has more satisfied customers. **Satisfaction:** There are more dissatisfied customers than satisfied customers. It is worth note taking that more than half are dissatisfied customers.

2.3 Detection of Outliers and Removal

Interquartile Range (IQR) will be used to detect outliers. Outliers will be removed from the dataset.

[11]:
```
Q1 = dataset.quantile(0.25)
Q3 = dataset.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
Gender                               1.0
Customer Type                        0.0
Age                                 24.0
Type of Travel                       1.0
Class                                1.0
Flight Distance                   1330.0
Inflight wifi service                2.0
Departure/Arrival time convenient    2.0
Ease of Online booking               2.0
Gate location                        2.0
Food and drink                       2.0
Online boarding                      2.0
Seat comfort                         3.0
Inflight entertainment               2.0
On-board service                     2.0
Leg room service                     2.0
Baggage handling                     2.0
Checkin service                      1.0
Inflight service                     2.0
Cleanliness                          2.0
Departure Delay in Minutes          12.0
Arrival Delay in Minutes            13.0
satisfaction                         1.0
dtype: float64
```

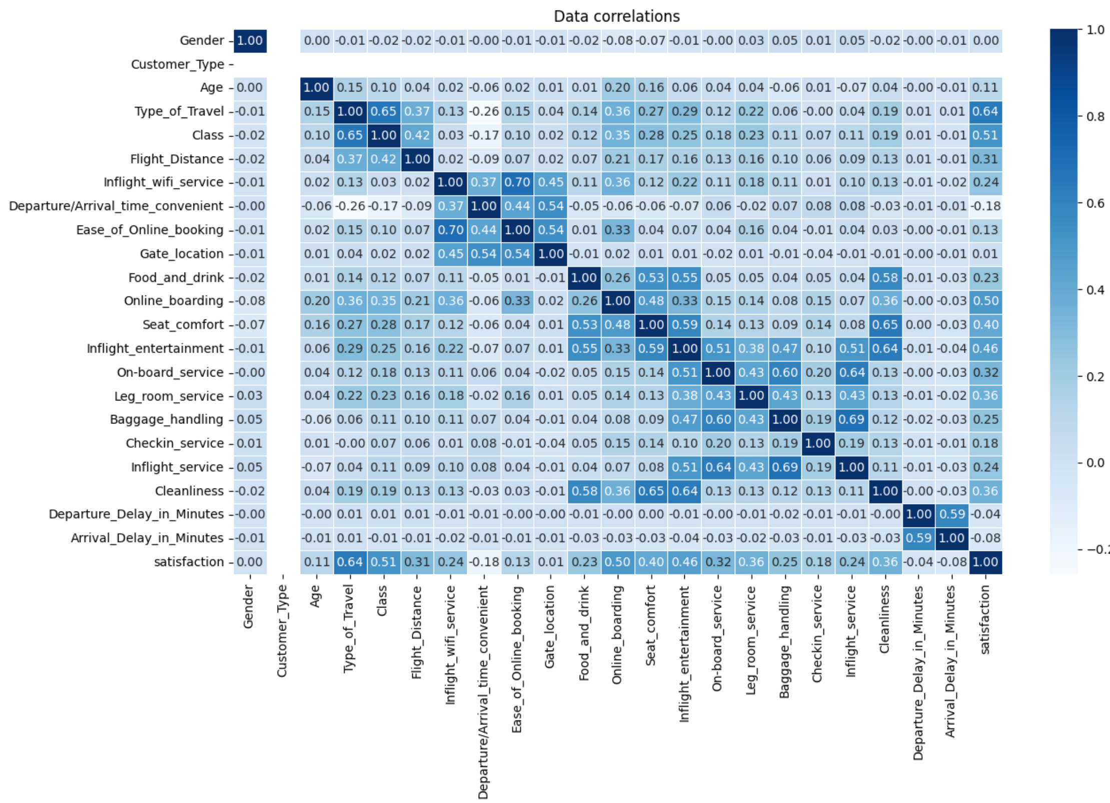
[24]:
```
# Remove Outliers
cleaned_ds = dataset[~((dataset < (Q1 - 1.5 * IQR)) |(dataset > (Q3 + 1.5 *␣
↪IQR))).any(axis=1)]
print(f'Number of rows before removal: {len(dataset)}')
print(f'Number of rows before removal: {len(cleaned_ds)}')
```

```
Number of rows before removal: 25976
Number of rows before removal: 25976
```

2.4 Features Correlation

[78]:
```
correlation=cleaned_ds.corr()
plt.figure(figsize=(14,8))
sns.heatmap(correlation,annot=True,fmt='.2f',annot_kws={'size':␣
↪10},linewidths=0.5,cmap='Blues')
plt.title("Data correlations")
```

[78]:
```
Text(0.5, 1.0, 'Data correlations')
```



2.5 Splitting of Data

[25]:
```
x = cleaned_ds.drop('satisfaction', axis=1)
y = cleaned_ds['satisfaction']

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.3,␣
↪random_state=42)
print("Training set shape:", x_train.shape, y_train.shape)
print("Testing set shape:", x_test.shape, y_test.shape)
```

```
Training set shape: (18183, 22) (18183,)
Testing set shape: (7793, 22) (7793,)
```

2.6 Confusion Matrix

[26]:
```
def make_confusion_matrix(cf,
                          group_names=None,
                          categories='auto',
                          count=True,
                          percent=True,
                          cbar=True,
                          xyticks=True,
```

```
                          xyplotlabels=True,
                          sum_stats=True,
                          figsize=None,
                          cmap='Blues',
                          title=None):

    blanks = ['' for i in range(cf.size)]

    if group_names and len(group_names)==cf.size:
        group_labels = ["{}\n".format(value) for value in group_names]
    else:
        group_labels = blanks

    if count:
        group_counts = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        group_counts = blanks

    if percent:
        group_percentages = ["{0:.2%}".format(value) for value in cf.flatten()/np.sum(cf)]
    else:
        group_percentages = blanks

    box_labels = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(group_labels,group_counts,group_percentages)]
    box_labels = np.asarray(box_labels).reshape(cf.shape[0],cf.shape[1])

    if sum_stats:
        accuracy  = np.trace(cf) / float(np.sum(cf))

        if len(cf)==2:
            #Metrics for Binary Confusion Matrices
            precision = cf[1,1] / sum(cf[:,1])
            recall    = cf[1,1] / sum(cf[1,:])
            f1_score  = 2*precision*recall / (precision + recall)
            stats_text = "\n\nAccuracy={:0.3f}\nPrecision={:0.3f}\nRecall={:0.3f}\nF1 Score={:0.3f}".format(
                accuracy,precision,recall,f1_score)
        else:
            stats_text = "\n\nAccuracy={:0.3f}".format(accuracy)
    else:
        stats_text = ""


    if figsize==None:
        figsize = plt.rcParams.get('figure.figsize')
```

```
    if xyticks==False:
        categories=False


    plt.figure(figsize=figsize)
    sns.
↪heatmap(cf,annot=box_labels,fmt="",cmap=cmap,cbar=cbar,xticklabels=categories,yticklabels=c

    if xyplotlabels:
        plt.ylabel('True label')
        plt.xlabel('Predicted label' + stats_text)
    else:
        plt.xlabel(stats_text)

    if title:
        plt.title(title)
```

2.7 Logistic Regression

[15]:
```
# Feature Scaling
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

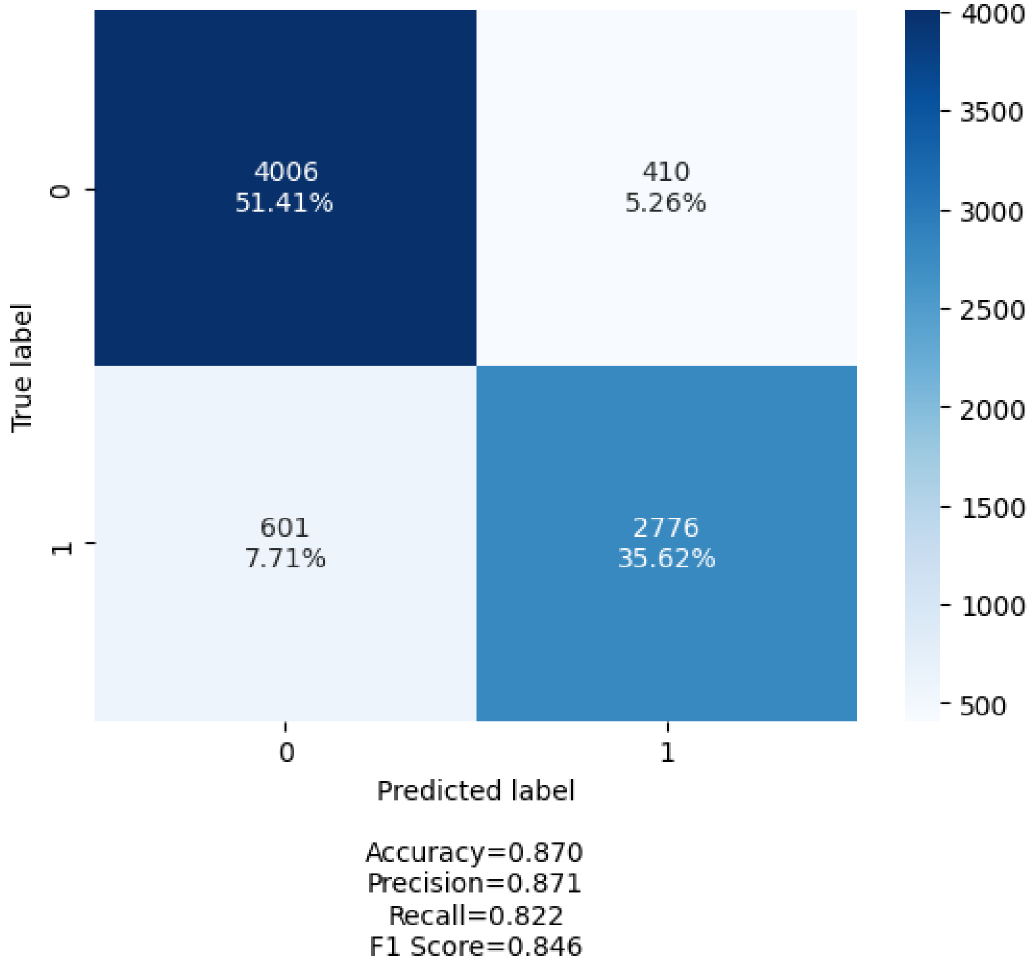
[138]:
```
class LogisticRegressionScratch:
    def __init__(self, learning_rate=0.01, num_iterations=300000,␣
↪fit_intercept=True, verbose=False):
        self.learning_rate = learning_rate
        self.num_iterations = num_iterations
        self.fit_intercept = fit_intercept
        self.verbose = verbose

    def __b_intercept(self, X):
        intercept = np.ones((X.shape[0], 1))
        return np.concatenate((intercept, X), axis=1)

    def __sigmoid_function(self, z):
        return 1 / (1 + np.exp(-z))

    def __loss(self, yp, y):
        return (-y * np.log(yp) - (1 - y) * np.log(1 - yp)).mean()

    def fit(self, X, y):
        if self.fit_intercept:
            X = self.__b_intercept(X)
        self.W = np.zeros(X.shape[1])
```

```
        for i in range(self.num_iterations):

            z = np.dot(X, self.W)
            yp = self.__sigmoid_function(z)
            gradient = np.dot(X.T, (yp - y)) / y.size

            self.W -= self.learning_rate * gradient

            z = np.dot(X, self.W)
            yp = self.__sigmoid_function(z)

            loss = self.__loss(yp, y)

            if(self.verbose ==True and i % 10000 == 0):
                print(f'loss: {loss} \t')

    def predict_prob(self, X):
        if self.fit_intercept:
            X = self.__b_intercept(X)
        return self.__sigmoid_function(np.dot(X, self.W))

    def predict(self, X):
        return self.predict_prob(X).round()
```

[139]:
```
log_res = LogisticRegressionScratch(learning_rate=0.1, num_iterations=300000)
log_res.fit(x_train, y_train)
```

[156]:
```
print(classification_report(y_test, log_res.predict(x_test)))

cm = confusion_matrix(y_test, log_res.predict(x_test))
make_confusion_matrix(cm)
```

```
              precision    recall  f1-score   support

           0       0.87      0.91      0.89      4416
           1       0.87      0.82      0.85      3377

    accuracy                           0.87      7793
   macro avg       0.87      0.86      0.87      7793
weighted avg       0.87      0.87      0.87      7793
```



2.7.1 Using the Library

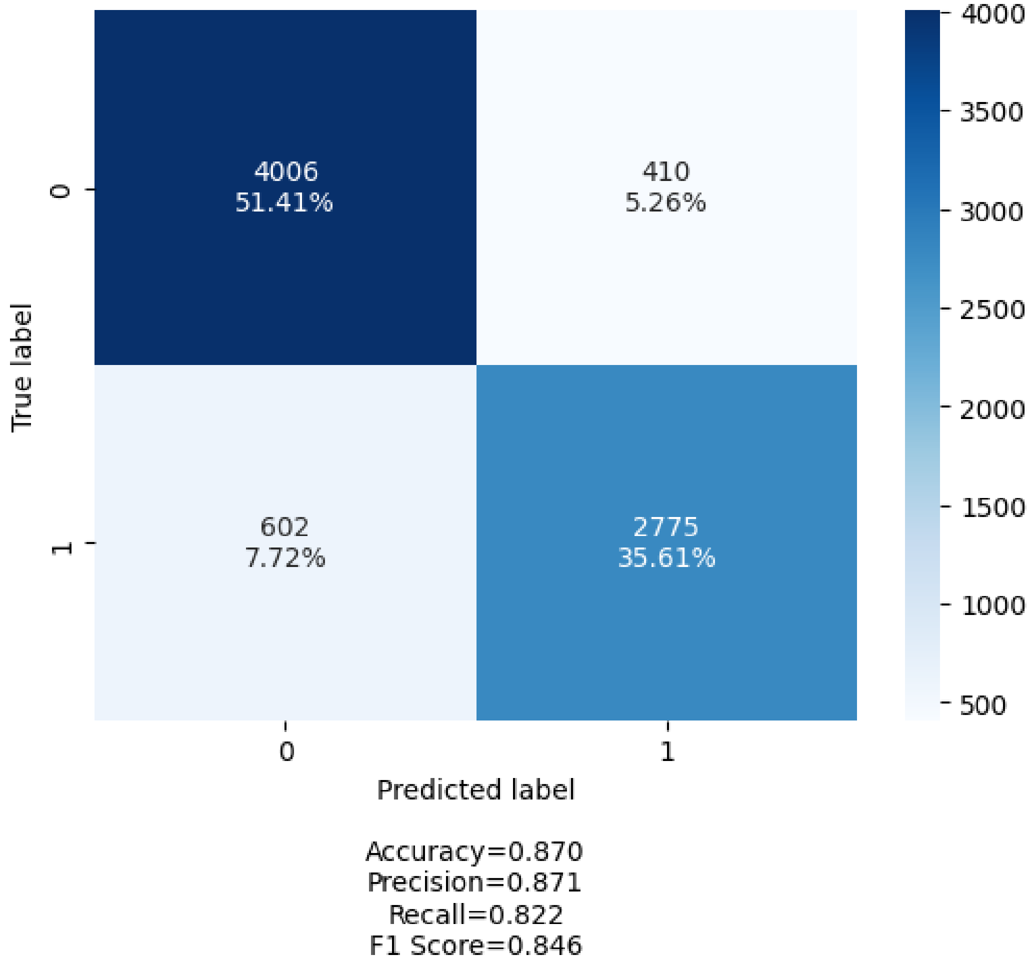
[154]:
```
lr = LogisticRegression()
lr.fit(x_train, y_train)
```

[154]:
```
LogisticRegression()
```

[155]:
```
y_pred = lr.predict(x_test)
print(classification_report(y_test, y_pred))

cm = confusion_matrix(y_test, y_pred)
make_confusion_matrix(cm)
```

```
              precision    recall  f1-score   support

           0       0.87      0.91      0.89      4416
           1       0.87      0.82      0.85      3377
```

```
    accuracy                           0.87      7793
   macro avg       0.87      0.86      0.87      7793
weighted avg       0.87      0.87      0.87      7793
```



2.8 Naive Bayes

2.8.1 Using Builtin Lib

[146]:
```
model = GaussianNB()
model.fit(x_train, y_train)
```

[146]: GaussianNB()

[157]:
```
y_pred = model.predict(x_test)
```

```python
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
make_confusion_matrix(cm)
```

```
              precision    recall  f1-score   support

           0       0.86      0.90      0.88      4416
           1       0.87      0.81      0.84      3377

    accuracy                           0.86      7793
   macro avg       0.86      0.86      0.86      7793
weighted avg       0.86      0.86      0.86      7793
```