# logistic regression

For this activity, you work as a consultant for an airline. The airline is interested in knowing if a better in-flight entertainment experience leads to higher customer satisfaction. They would like you to construct and evaluate a model that predicts whether a future customer would be satisfied with their services given previous customer feedback about their flight experience

In [1]:
```

# Standard operational package imports.
import numpy as np
import pandas as pd

# Important imports for preprocessing, modeling, and evaluation.
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import sklearn.metrics as metrics

# Visualization package imports.
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
#    load dataset

df = pd.read_csv("air.csv")
```

In [3]:
```
df.head(10)
```

Out[3]:

| | satisfaction | Customer Type | Age | Type of Travel | Class | Flight Distance | Seat comfort | Departure/Arrival time convenient | Food and drink | Ic |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | satisfied | Loyal Customer | 65 | Personal Travel | Eco | 265 | 0 | 0 | 0 | |
| **1** | satisfied | Loyal Customer | 47 | Personal Travel | Business | 2464 | 0 | 0 | 0 | |
| **2** | satisfied | Loyal Customer | 15 | Personal Travel | Eco | 2138 | 0 | 0 | 0 | |
| **3** | satisfied | Loyal Customer | 60 | Personal Travel | Eco | 623 | 0 | 0 | 0 | |
| **4** | satisfied | Loyal Customer | 70 | Personal Travel | Eco | 354 | 0 | 0 | 0 | |
| **5** | satisfied | Loyal Customer | 30 | Personal Travel | Eco | 1894 | 0 | 0 | 0 | |
| **6** | satisfied | Loyal Customer | 66 | Personal Travel | Eco | 227 | 0 | 0 | 0 | |
| **7** | satisfied | Loyal Customer | 10 | Personal Travel | Eco | 1812 | 0 | 0 | 0 | |
| **8** | satisfied | Loyal Customer | 56 | Personal Travel | Business | 73 | 0 | 0 | 0 | |
| **9** | satisfied | Loyal Customer | 22 | Personal Travel | Eco | 1556 | 0 | 0 | 0 | |

10 rows × 22 columns

# Explore the data

Check the data type of each column. Note that logistic regression models expect numeric data.

In [4]:
```
df.dtypes
```

Out[4]:
```
satisfaction                          object
Customer Type                         object
Age                                    int64
Type of Travel                        object
Class                                 object
Flight Distance                        int64
Seat comfort                           int64
Departure/Arrival time convenient      int64
Food and drink                         int64
Gate location                          int64
Inflight wifi service                  int64
Inflight entertainment                 int64
Online support                         int64
Ease of Online booking                 int64
On-board service                       int64
Leg room service                       int64
Baggage handling                       int64
Checkin service                        int64
Cleanliness                            int64
Online boarding                        int64
Departure Delay in Minutes             int64
Arrival Delay in Minutes             float64
dtype: object
```

To predict customer satisfaction, check how many customers in the dataset are satisfied before modeling.

In [5]:
```
df['satisfaction'].value_counts(dropna = False)
```

Out[5]:
```
satisfied       71087
dissatisfied    58793
Name: satisfaction, dtype: int64
```

There were 71,087 satisfied customers and 58,793 dissatisfied customers.

54.7 percent (71,087/129,880) of customers were satisfied. While this is a simple calculation, this value can be compared to a logistic regression model's accuracy.

An assumption of logistic regression models is that there are no missing values. Check for missing values in the rows of the data.

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
satisfaction                          0
Customer Type                         0
Age                                   0
Type of Travel                        0
Class                                 0
Flight Distance                       0
Seat comfort                          0
Departure/Arrival time convenient     0
Food and drink                        0
Gate location                         0
Inflight wifi service                 0
Inflight entertainment                0
Online support                        0
Ease of Online booking                0
On-board service                      0
Leg room service                      0
Baggage handling                      0
Checkin service                       0
Cleanliness                           0
Online boarding                       0
Departure Delay in Minutes            0
Arrival Delay in Minutes            393
dtype: int64
```

For this activity, the airline is specifically interested in knowing if a better in-flight entertainment experience leads to higher customer satisfaction. The Arrival Delay in Minutes column won't be included in the binomial logistic regression model; however, the airline might become interested in this column in the future.

In [7]:
```
#  Drop the rows with missing values and save the resulting pandas DataFra


df_subset = df.dropna(axis=0).reset_index(drop = True)
```

If you want to create a plot (sns.regplot) of your model to visualize results later in the notebook, the independent variable Inflight entertainment cannot be "of type int" and the dependent variable satisfaction cannot be "of type object."

Make the Inflight entertainment column "of type float."

In [8]:
```

df_subset = df_subset.astype({"Inflight entertainment": float})
```

Convert the categorical column satisfaction into numeric through one-hot encoding.

In [9]:
```
OneHotEncoder(drop='first').fit_transform(df_subset[['satisfaction']])
```

Out[9]:
```
<129487x1 sparse matrix of type '<class 'numpy.float64'>'
	with 70882 stored elements in Compressed Sparse Row format>
```

In [10]:
```
OneHotEncoder(drop='first').fit_transform(df_subset[['satisfaction']]).toa
```

Out[10]:
```
array([[1.],
       [1.],
       [1.],
       ...,
       [0.],
       [0.],
       [0.]])
```

In [11]:
```
OneHotEncoder().fit_transform(df_subset[['satisfaction']]).toarray()
```

Out[11]:
```
array([[0., 1.],
       [0., 1.],
       [0., 1.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]])
```

In [12]:
```

df_subset['satisfaction'] = OneHotEncoder(drop='first').fit_transform(df_s
```

In [13]:
```
#  To examine what one-hot encoding did to the DataFrame, output the first


df_subset.head(10)
```

Out[13]:

| | satisfaction | Customer Type | Age | Type of Travel | Class | Flight Distance | Seat comfort | Departure/Arrival time convenient | Food and drink | Ic |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | Loyal Customer | 65 | Personal Travel | Eco | 265 | 0 | 0 | 0 | |
| **1** | 1.0 | Loyal Customer | 47 | Personal Travel | Business | 2464 | 0 | 0 | 0 | |
| **2** | 1.0 | Loyal Customer | 15 | Personal Travel | Eco | 2138 | 0 | 0 | 0 | |
| **3** | 1.0 | Loyal Customer | 60 | Personal Travel | Eco | 623 | 0 | 0 | 0 | |
| **4** | 1.0 | Loyal Customer | 70 | Personal Travel | Eco | 354 | 0 | 0 | 0 | |
| **5** | 1.0 | Loyal Customer | 30 | Personal Travel | Eco | 1894 | 0 | 0 | 0 | |
| **6** | 1.0 | Loyal Customer | 66 | Personal Travel | Eco | 227 | 0 | 0 | 0 | |
| **7** | 1.0 | Loyal Customer | 10 | Personal Travel | Eco | 1812 | 0 | 0 | 0 | |
| **8** | 1.0 | Loyal Customer | 56 | Personal Travel | Business | 73 | 0 | 0 | 0 | |
| **9** | 1.0 | Loyal Customer | 22 | Personal Travel | Eco | 1556 | 0 | 0 | 0 | |

10 rows × 22 columns

In [14]:
```
#   you ca see the result from the satisfaction column
```

# Create the training and testing data

Put 70% of the data into a training set and the remaining 30% into a testing set. Create an X and y DataFrame with only the necessary variables.

In [15]:
```
X = df_subset[["Inflight entertainment"]]
y = df_subset["satisfaction"]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, ra
```

In [16]:
```
X.head()
```

Out[16]:

| | Inflight entertainment |
|---|---|
| **0** | 4.0 |
| **1** | 2.0 |
| **2** | 0.0 |
| **3** | 4.0 |
| **4** | 3.0 |

# Model building

In [17]:
```
clf = LogisticRegression().fit(X_train,y_train)
```

Obtain parameter estimates Make sure you output the two parameters from your model.

In [18]:
```
clf.coef_
```

Out[18]: `array([[0.99751462]])`

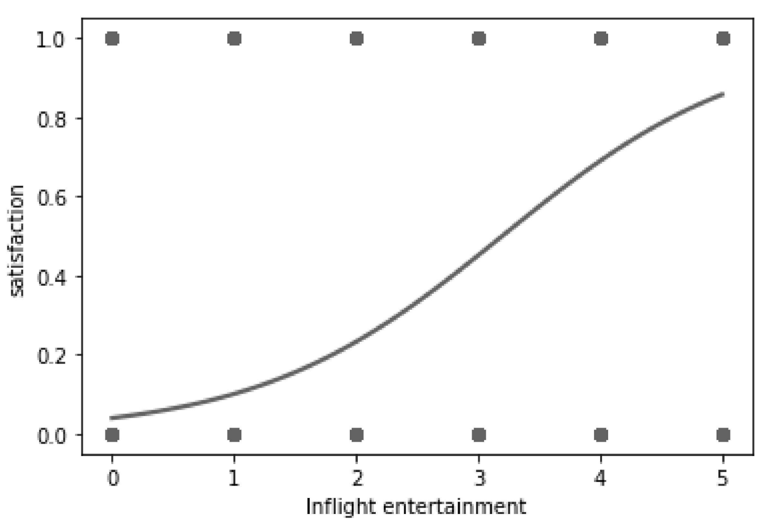
In [19]:
```
clf.intercept_
```

Out[19]: `array([-3.19355406])`

Create a plot of your model to visualize results using the seaborn package.

In [20]:
```
sns.regplot(x="Inflight entertainment", y="satisfaction", data=df_subset,
```

Out[20]: `<AxesSubplot:xlabel='Inflight entertainment', ylabel='satisfaction'>`



The graph seems to indicate that the higher the inflight entertainment value, the higher the customer satisfaction, though this is currently not the most informative plot. The graph currently doesn't provide much insight into the data points, as Inflight entertainment is categorical.

# Results and evaluation

# Predict the outcome for the test dataset

In [25]:
```
# Save predictions.

y_pred = clf.predict(X_test)
```

In [29]:
```
#   In order to examine the predictions, print out y_pred.


print(y_pred)
```

```
[1. 0. 0. ... 0. 0. 0.]
```

In [30]:
```
# Use predict_proba to output a probability.
clf.predict_proba(X_test)
```

Out[30]:
```
array([[0.14258068, 0.85741932],
       [0.55008402, 0.44991598],
       [0.89989329, 0.10010671],
       ...,
       [0.89989329, 0.10010671],
       [0.76826225, 0.23173775],
       [0.55008402, 0.44991598]])
```

In [31]:
```
# Use predict to output 0's and 1's.


clf.predict(X_test)
```

Out[31]:
```
array([1., 0., 0., ..., 0., 0., 0.])
```

# Analyze the results

In [32]:

```
### YOUR CODE HERE ###

print("Accuracy:", "%.6f" % metrics.accuracy_score(y_test, y_pred))
print("Precision:", "%.6f" % metrics.precision_score(y_test, y_pred))
print("Recall:", "%.6f" % metrics.recall_score(y_test, y_pred))
print("F1 Score:", "%.6f" % metrics.f1_score(y_test, y_pred))
```

```
Accuracy: 0.801529
Precision: 0.816142
Recall: 0.821530
F1 Score: 0.818827
```

Precision measures the proportion of data points predicted as True that are actually True. (Ranges from 0 to 1)

Recall measures the proportion of data points that are predicted as True, out of all the data points that are actually True.(Ranges from 0 to 1)

Accuracy measures the proportion of data points that are correctly classified.(Ranges from 0 to 1)

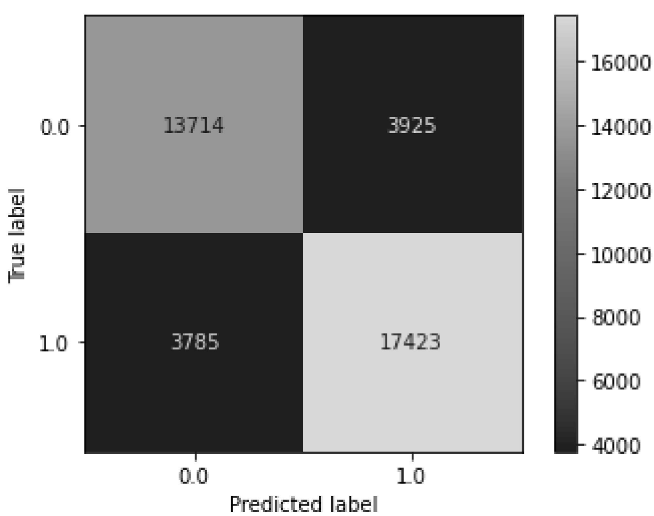
The model performed well since "Accuracy", "Precision", "Recall" and "F1 Score" are up to 80 %

In [35]:

```
#  Let's examine the type of error made by the algorithm

cm = metrics.confusion_matrix(y_test, y_pred, labels = clf.classes_)
disp = metrics.ConfusionMatrixDisplay(confusion_matrix = cm,display_labels
disp.plot()
```

Out[35]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1c2cf594be0>




1.True negatives: 13714

The count of observations that a classifier correctly predicted as False (0)

2. True positives: 17423

The count of observations that a classifier correctly predicted as True (1)

3. False positives: 3925

The count of observations that a classifier incorrectly predicted as True (1)

4. False negatives: 3785

The count of observations that a classifier incorrectly predicted as False (0)

True positve - False nagative = 17423-3925 = 13498

True negative - False negative = 13714-3785 = 9929

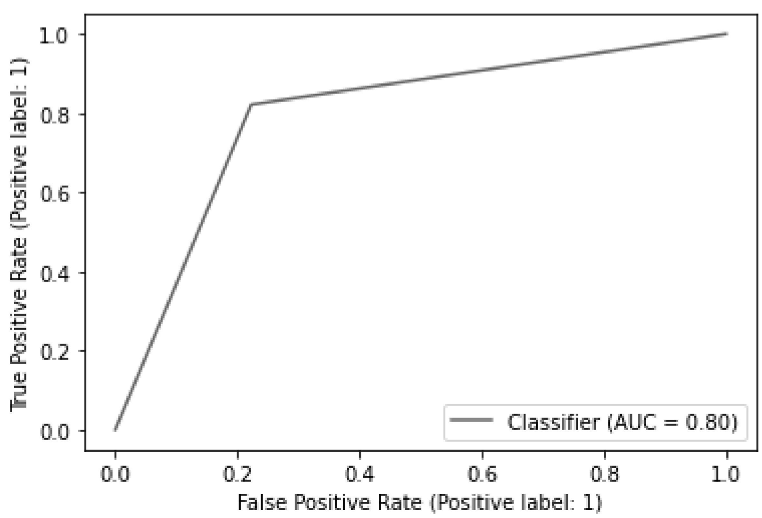
The model performance is not too bad

# ROC curves

An ROC curve helps in visualizing the performance of a logistic regression classifier.

In [41]:
```
from sklearn.metrics import RocCurveDisplay
```

In [42]:
```
RocCurveDisplay.from_predictions(y_test, y_pred)
plt.show()
```




In this graph, the ROC curve indicates that the corresponding classifier performs decently well. AUC = 80 %

# Findings and Recommendations

Customers who rated in-flight entertainment highly were more likely to be satisfied. Improving in-flight entertainment should lead to better customer satisfaction.

The model is 80.2 percent accurate. This is an improvement over the dataset's customer satisfaction rate of 54.7 percent.

The success of the model suggests that the airline should invest more in model developement to examine if adding more independent variables leads to better results. Building this model could not only be useful in predicting whether or not a customer would be satisfied but also lead to a

In [ ]:

```
1
```