Airline_passenger_satisfaction

August 15, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
from sklearn.model_selection import KFold, train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
```

[2]:
```
plane=pd.read_csv('train.csv')
plane
```

[2]:

	Unnamed: 0	id	Gender	Customer Type	Age	Type of Travel	\
0	0	70172	Male	Loyal Customer	13	Personal Travel	
1	1	5047	Male	disloyal Customer	25	Business travel	
2	2	110028	Female	Loyal Customer	26	Business travel	
3	3	24026	Female	Loyal Customer	25	Business travel	
4	4	119299	Male	Loyal Customer	61	Business travel	
…	…	…	…	…	…	…	
103899	103899	94171	Female	disloyal Customer	23	Business travel	
103900	103900	73097	Male	Loyal Customer	49	Business travel	
103901	103901	68825	Male	disloyal Customer	30	Business travel	
103902	103902	54173	Female	disloyal Customer	22	Business travel	
103903	103903	62567	Male	Loyal Customer	27	Business travel	

	Class	Flight Distance	Inflight wifi service	\
0	Eco Plus	460	3	
1	Business	235	3	
2	Business	1142	2	
3	Business	562	2	

```
4       Business              214                      3
…          …                …                     …
103899      Eco              192                      2
103900  Business             2347                     4
103901  Business             1995                     1
103902      Eco              1000                     1
103903  Business             1723                     1

        Departure/Arrival time convenient  …  Inflight entertainment  \
0                                       4  …                       5
1                                       2  …                       1
2                                       2  …                       5
3                                       5  …                       2
4                                       3  …                       3
…                                     …  …                     …
103899                                  1  …                       2
103900                                  4  …                       5
103901                                  1  …                       4
103902                                  1  …                       1
103903                                  3  …                       1

        On-board service  Leg room service  Baggage handling  Checkin service  \
0                      4                 3                 4                4
1                      1                 5                 3                1
2                      4                 3                 4                4
3                      2                 5                 3                1
4                      3                 4                 4                3
…                    …                 …                 …              …
103899                 3                 1                 4                2
103900                 5                 5                 5                5
103901                 3                 2                 4                5
103902                 4                 5                 1                5
103903                 1                 1                 4                4

        Inflight service  Cleanliness  Departure Delay in Minutes  \
0                      5            5                          25
1                      4            1                           1
2                      4            5                           0
3                      4            2                          11
4                      3            3                           0
…                    …          …                         …
103899                 3            2                           3
103900                 5            4                           0
103901                 5            4                           7
103902                 4            1                           0
103903                 3            1                           0
```

```
        Arrival Delay in Minutes             satisfaction
0                          18.0  neutral or dissatisfied
1                           6.0  neutral or dissatisfied
2                           0.0                satisfied
3                           9.0  neutral or dissatisfied
4                           0.0                satisfied
…                           …                        …
103899                      0.0  neutral or dissatisfied
103900                      0.0                satisfied
103901                     14.0  neutral or dissatisfied
103902                      0.0  neutral or dissatisfied
103903                      0.0  neutral or dissatisfied

[103904 rows x 25 columns]
```

[3]:
```
plane.head()
```

[3]:
```
   Unnamed: 0      id  Gender      Customer Type  Age   Type of Travel  \
0           0   70172    Male     Loyal Customer   13  Personal Travel
1           1    5047    Male  disloyal Customer   25  Business travel
2           2  110028  Female     Loyal Customer   26  Business travel
3           3   24026  Female     Loyal Customer   25  Business travel
4           4  119299    Male     Loyal Customer   61  Business travel

      Class  Flight Distance  Inflight wifi service  \
0  Eco Plus              460                      3
1  Business              235                      3
2  Business             1142                      2
3  Business              562                      2
4  Business              214                      3

   Departure/Arrival time convenient  …  Inflight entertainment  \
0                                  4  …                       5
1                                  2  …                       1
2                                  2  …                       5
3                                  5  …                       2
4                                  3  …                       3

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

   Inflight service  Cleanliness  Departure Delay in Minutes  \
0                 5            5                          25
```

```
1                 4           1                          1
2                 4           5                          0
3                 4           2                         11
4                 3           3                          0

   Arrival Delay in Minutes             satisfaction
0                      18.0  neutral or dissatisfied
1                       6.0  neutral or dissatisfied
2                       0.0                satisfied
3                       9.0  neutral or dissatisfied
4                       0.0                satisfied

[5 rows x 25 columns]
```

[4]:
```
plane.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
 20  Inflight service                  103904 non-null  int64
 21  Cleanliness                       103904 non-null  int64
 22  Departure Delay in Minutes        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
```

[5]:
```
plane.drop(['Unnamed: 0', 'id'], axis=1, inplace=True)
plane
```

[5]:

	Gender	Customer Type	Age	Type of Travel	Class	\
0	Male	Loyal Customer	13	Personal Travel	Eco Plus	
1	Male	disloyal Customer	25	Business travel	Business	
2	Female	Loyal Customer	26	Business travel	Business	
3	Female	Loyal Customer	25	Business travel	Business	
4	Male	Loyal Customer	61	Business travel	Business	
…	…	…	…	…	…	
103899	Female	disloyal Customer	23	Business travel	Eco	
103900	Male	Loyal Customer	49	Business travel	Business	
103901	Male	disloyal Customer	30	Business travel	Business	
103902	Female	disloyal Customer	22	Business travel	Eco	
103903	Male	Loyal Customer	27	Business travel	Business	

	Flight Distance	Inflight wifi service	\
0	460	3	
1	235	3	
2	1142	2	
3	562	2	
4	214	3	
…	…	…	
103899	192	2	
103900	2347	4	
103901	1995	1	
103902	1000	1	
103903	1723	1	

	Departure/Arrival time convenient	Ease of Online booking	\
0	4	3	
1	2	3	
2	2	2	
3	5	5	
4	3	3	
…	…	…	
103899	1	2	
103900	4	4	
103901	1	1	
103902	1	1	
103903	3	3	

	Gate location	…	Inflight entertainment	On-board service	\
0	1	…	5	4	
1	3	…	1	1	

```
2                  2  …                    5                 4
3                  5  …                    2                 2
4                  3  …                    3                 3
…                …  …                    …                 …
103899             3  …                    2                 3
103900             4  …                    5                 5
103901             3  …                    4                 3
103902             5  …                    1                 4
103903             3  …                    1                 1

        Leg room service  Baggage handling  Checkin service  Inflight service  \
0                      3                 4                4                 5
1                      5                 3                1                 4
2                      3                 4                4                 4
3                      5                 3                1                 4
4                      4                 4                3                 3
…                    …                 …                …                 …
103899                 1                 4                2                 3
103900                 5                 5                5                 5
103901                 2                 4                5                 5
103902                 5                 1                5                 4
103903                 1                 4                4                 3

        Cleanliness  Departure Delay in Minutes  Arrival Delay in Minutes  \
0                 5                         25                      18.0
1                 1                          1                       6.0
2                 5                          0                       0.0
3                 2                         11                       9.0
4                 3                          0                       0.0
…               …                          …                       …
103899            2                          3                       0.0
103900            4                          0                       0.0
103901            4                          7                      14.0
103902            1                          0                       0.0
103903            1                          0                       0.0

                  satisfaction
0       neutral or dissatisfied
1       neutral or dissatisfied
2                     satisfied
3       neutral or dissatisfied
4                     satisfied
…                           …
103899  neutral or dissatisfied
103900                satisfied
103901  neutral or dissatisfied
103902  neutral or dissatisfied
```

```
103903  neutral or dissatisfied

[103904 rows x 23 columns]
```

[6]:
```
print("Number of Unique values in Gender :",plane['Gender'].nunique())
print("Number of Unique values in Customer Type :",plane['Customer Type'].
 ↪nunique())
print("Number of Unique values in Type of Travel :",plane['Type of Travel'].
 ↪nunique())
print("Number of Unique values in Class Churn :",plane['Class'].nunique())
print("Number of Unique values in satisfaction Churn :",plane['satisfaction'].
 ↪nunique())
```

```
Number of Unique values in Gender : 2
Number of Unique values in Customer Type : 2
Number of Unique values in Type of Travel : 2
Number of Unique values in Class Churn : 3
Number of Unique values in satisfaction Churn : 2
```

[7]:
```
plane.describe()
```

[7]:

	Age	Flight Distance	Inflight wifi service
count	103904.000000	103904.000000	103904.000000
mean	39.379706	1189.448375	2.729683
std	15.114964	997.147281	1.327829
min	7.000000	31.000000	0.000000
25%	27.000000	414.000000	2.000000
50%	40.000000	843.000000	3.000000
75%	51.000000	1743.000000	4.000000
max	85.000000	4983.000000	5.000000

	Departure/Arrival time convenient	Ease of Online booking
count	103904.000000	103904.000000
mean	3.060296	2.756901
std	1.525075	1.398929
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	Food and drink	Online boarding	Seat comfort
count	103904.000000	103904.000000	103904.000000	103904.000000
mean	2.976883	3.202129	3.250375	3.439396
std	1.277621	1.329533	1.349509	1.319088
min	0.000000	0.000000	0.000000	0.000000
25%	2.000000	2.000000	2.000000	2.000000

```
50%         3.000000        3.000000         3.000000       4.000000
75%         4.000000        4.000000         4.000000       5.000000
max         5.000000        5.000000         5.000000       5.000000

       Inflight entertainment  On-board service  Leg room service  \
count           103904.000000     103904.000000     103904.000000
mean                 3.358158          3.382363          3.351055
std                  1.332991          1.288354          1.315605
min                  0.000000          0.000000          0.000000
25%                  2.000000          2.000000          2.000000
50%                  4.000000          4.000000          4.000000
75%                  4.000000          4.000000          4.000000
max                  5.000000          5.000000          5.000000

       Baggage handling  Checkin service  Inflight service    Cleanliness  \
count     103904.000000    103904.000000     103904.000000  103904.000000
mean           3.631833         3.304290          3.640428       3.286351
std            1.180903         1.265396          1.175663       1.312273
min            1.000000         0.000000          0.000000       0.000000
25%            3.000000         3.000000          3.000000       2.000000
50%            4.000000         3.000000          4.000000       3.000000
75%            5.000000         4.000000          5.000000       4.000000
max            5.000000         5.000000          5.000000       5.000000

       Departure Delay in Minutes  Arrival Delay in Minutes
count                103904.000000             103594.000000
mean                     14.815618                 15.178678
std                      38.230901                 38.698682
min                       0.000000                  0.000000
25%                       0.000000                  0.000000
50%                       0.000000                  0.000000
75%                      12.000000                 13.000000
max                    1592.000000               1584.000000
```

[8]:
```
plane.isnull().sum()
```

[8]:
```
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
```

[9]:
```
plane.replace({'Male': 0, 'Female': 1}, inplace=True)
plane.replace({'Loyal Customer':0, 'disloyal Customer':1}, inplace=True)
plane.replace({'Personal Travel':0, 'Business travel':1}, inplace=True)
plane.replace({'Eco Plus':0, 'Business':1, 'Eco':2}, inplace=True)
plane.replace({'neutral or dissatisfied':0, 'satisfied':1}, inplace=True)
```

[10]:
```
plane['Arrival Delay in Minutes'].fillna(plane['Arrival Delay in Minutes'].
↪mean(), inplace=True)
```

[11]:
```
plane.isnull().sum()
```

[11]:
```
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
```

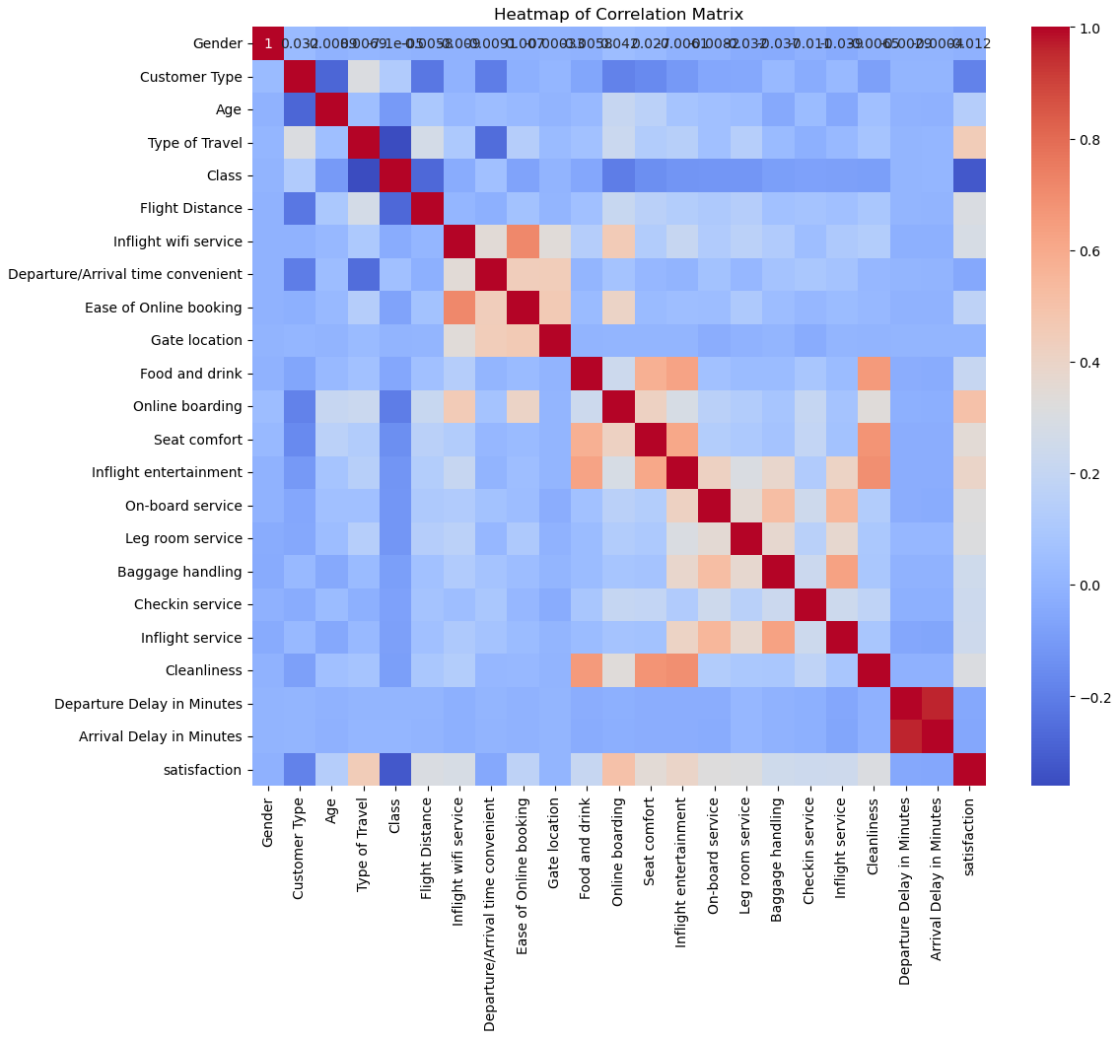
```
satisfaction                        0
dtype: int64
```

[12]:
```
plane["satisfaction"].value_counts().to_frame()
```

[12]:
```
              count
satisfaction
0             58879
1             45025
```

[13]:
```
corr_matrix = plane.corr()

# Plot the heatmap
plt.figure(figsize=(12, 10))  # Adjust the figure size as needed
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm")  # Customize the heatmap␣
 ↪appearance
plt.title('Heatmap of Correlation Matrix')
plt.show()
```



[14]:
```
plane.drop(['Arrival Delay in Minutes'], axis=1, inplace=True)
plane
```

[14]:

	Gender	Customer Type	Age	Type of Travel	Class	Flight Distance	\
0	0	0	13	0	0	460	
1	0	1	25	1	1	235	
2	1	0	26	1	1	1142	
3	1	0	25	1	1	562	
4	0	0	61	1	1	214	
…	…	…	…	…	…	…	
103899	1	1	23	1	2	192	
103900	0	0	49	1	1	2347	
103901	0	1	30	1	1	1995	
103902	1	1	22	1	2	1000	
103903	0	0	27	1	1	1723	

```
        Inflight wifi service  Departure/Arrival time convenient  \
0                           3                                  4
1                           3                                  2
2                           2                                  2
3                           2                                  5
4                           3                                  3
…                         …                                  …
103899                      2                                  1
103900                      4                                  4
103901                      1                                  1
103902                      1                                  1
103903                      1                                  3

        Ease of Online booking  Gate location  …  Seat comfort  \
0                            3              1  …             5
1                            3              3  …             1
2                            2              2  …             5
3                            5              5  …             2
4                            3              3  …             5
…                          …              …  …             …
103899                       2              3  …             2
103900                       4              4  …             5
103901                       1              3  …             5
103902                       1              5  …             1
103903                       3              3  …             1

        Inflight entertainment  On-board service  Leg room service  \
0                            5                 4                 3
1                            1                 1                 5
2                            5                 4                 3
3                            2                 2                 5
4                            3                 3                 4
…                          …                 …                 …
103899                       2                 3                 1
103900                       5                 5                 5
103901                       4                 3                 2
103902                       1                 4                 5
103903                       1                 1                 1

        Baggage handling  Checkin service  Inflight service  Cleanliness  \
0                      4                4                 5            5
1                      3                1                 4            1
2                      4                4                 4            5
3                      3                1                 4            2
4                      4                3                 3            3
…                    …                …                 …          …
```

```
103899                  4                  2                  3                  2
103900                  5                  5                  5                  4
103901                  4                  5                  5                  4
103902                  1                  5                  4                  1
103903                  4                  4                  3                  1

        Departure Delay in Minutes  satisfaction
0                               25             0
1                                1             0
2                                0             1
3                               11             0
4                                0             1
…                              …              …
103899                           3             0
103900                           0             1
103901                           7             0
103902                           0             0
103903                           0             0

[103904 rows x 22 columns]
```

1 Train and Test split

[15]:
```
X = plane.drop('satisfaction', axis=1)  # Features
y = plane['satisfaction']  # Target variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

2 Scaling

[ ]:
```
# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

3 SUPPORT VECTOR MACHINE

[16]:
```
# Initialize the SVM classifier
svm_classifier = SVC(kernel='linear')  # You can change the kernel type as␣
↪needed

# Train the model
```

```
svm_classifier.fit(X_train_scaled, y_train)

y_pred = svm_classifier.predict(X_test_scaled)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.8758481305038256
              precision    recall  f1-score   support

           0       0.87      0.92      0.89     11713
           1       0.88      0.82      0.85      9068

    accuracy                           0.88     20781
   macro avg       0.88      0.87      0.87     20781
weighted avg       0.88      0.88      0.88     20781
```

[25]:
```
# Initialize the SVM classifier
svm_classifier = SVC(kernel='linear')  # You can change the kernel type as␣
 ↪needed

# Perform cross-validation
cv_scores = cross_val_score(svm_classifier, X_train_scaled, y_train, cv=5,␣
 ↪scoring='accuracy')

# Print the average cross-validation score
print("Cross-Validation Accuracy Scores", np.mean(cv_scores))

# Train the model on the entire training set
svm_classifier.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = svm_classifier.predict(X_test_scaled)

# Evaluate the model
print("Test Set Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Cross-Validation Accuracy Scores 0.8739699646131692
Test Set Accuracy: 0.8758481305038256
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.92      0.89     11713
           1       0.88      0.82      0.85      9068
```

```
    accuracy                           0.88     20781
   macro avg       0.88      0.87      0.87     20781
weighted avg       0.88      0.88      0.88     20781
```

4 LOGISTIC REGRESSION

[17]:
```
# Initialize the logistic regression classifier
log_reg = LogisticRegression(max_iter=1000)  # Increase max_iter if convergence␣
 ↪issues

# Train the model
log_reg.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = log_reg.predict(X_test_scaled)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.8744045041143352
              precision    recall  f1-score   support

           0       0.88      0.91      0.89     11713
           1       0.87      0.83      0.85      9068

    accuracy                           0.87     20781
   macro avg       0.87      0.87      0.87     20781
weighted avg       0.87      0.87      0.87     20781
```

[23]:
```
# Initialize the Logistic Regression classifier
log_reg = LogisticRegression(max_iter=1000)  # Increase max_iter if convergence␣
 ↪issues

# Perform cross-validation
cv_scores = cross_val_score(log_reg, X_train_scaled, y_train, cv=5,␣
 ↪scoring='accuracy')

# Print the average cross-validation score
print("Cross-Validation Accuracy Scores", np.mean(cv_scores))

# Train the model on the entire training set
log_reg.fit(X_train_scaled, y_train)
```

```
# Predict on the test set
y_pred = log_reg.predict(X_test_scaled)

# Evaluate the model
print("Test Set Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Cross-Validation Accuracy Scores 0.8727187962688241
Test Set Accuracy: 0.8744045041143352
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.91      0.89     11713
           1       0.87      0.83      0.85      9068

    accuracy                           0.87     20781
   macro avg       0.87      0.87      0.87     20781
weighted avg       0.87      0.87      0.87     20781
```

5 KNN

[18]:
```
# Initialize the k-NN classifier
knn = KNeighborsClassifier(n_neighbors=5)  # You can adjust the number of␣
 ↪neighbors

# Train the model
knn.fit(X_train_scaled, y_train)
# Predict on the test set
y_pred = knn.predict(X_test_scaled)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
  File "/opt/conda/envs/anaconda-2024.02-py310/lib/python3.10/site-
packages/joblib/externals/loky/backend/context.py", line 217, in
_count_physical_cores
    raise ValueError(

Accuracy: 0.92931042779462
              precision    recall  f1-score   support

           0       0.91      0.97      0.94     11713
           1       0.95      0.88      0.92      9068

    accuracy                           0.93     20781
```

```
   macro avg       0.93      0.92      0.93     20781
weighted avg       0.93      0.93      0.93     20781
```

[24]:
```
# Initialize the K-NN classifier
knn = KNeighborsClassifier(n_neighbors=5)  # You can adjust the number of␣
 ↪neighbors

# Perform cross-validation
cv_scores = cross_val_score(knn, X_train_scaled, y_train, cv=5,␣
 ↪scoring='accuracy')

# Print the average cross-validation score
print("Cross-Validation Accuracy Scores", np.mean(cv_scores))

# Train the model on the entire training set
knn.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = knn.predict(X_test_scaled)

# Evaluate the model
print("Test Set Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Cross-Validation Accuracy Scores 0.9256403272377287
Test Set Accuracy: 0.92931042779462
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.97      0.94     11713
           1       0.95      0.88      0.92      9068

    accuracy                           0.93     20781
   macro avg       0.93      0.92      0.93     20781
weighted avg       0.93      0.93      0.93     20781
```

6 GRADIENT BOOSTING CLASSIFIER

[19]:
```
# Initialize the Gradient Boosting classifier
gb_clf = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1,␣
 ↪max_depth=1, random_state=42)

# Train the model
gb_clf.fit(X_train_scaled, y_train)
```

```
# Predict on the test set
y_pred = gb_clf.predict(X_test_scaled)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9109282517684423
              precision    recall  f1-score   support

           0       0.90      0.94      0.92     11713
           1       0.92      0.87      0.89      9068

    accuracy                           0.91     20781
   macro avg       0.91      0.91      0.91     20781
weighted avg       0.91      0.91      0.91     20781
```

[22]:
```
# Initialize the Gradient Boosting classifier
gb_clf = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1,␣
 ↪max_depth=1, random_state=42)

# Perform cross-validation
cv_scores = cross_val_score(gb_clf, X_train_scaled, y_train, cv=5,␣
 ↪scoring='accuracy')

# Print the average cross-validation score
print("Cross-Validation Accuracy Scores", np.mean(cv_scores))

# Train the model on the entire training set
gb_clf.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = gb_clf.predict(X_test_scaled)

# Evaluate the model
print("Test Set Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Cross-Validation Accuracy Scores 0.9112400327092998
Test Set Accuracy: 0.9109282517684423
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.94      0.92     11713
           1       0.92      0.87      0.89      9068
```

```
    accuracy                           0.91     20781
   macro avg       0.91      0.91      0.91     20781
weighted avg       0.91      0.91      0.91     20781
```

7 DECISION TREE CLASSIFIER

[20]:
```
# Initialize the Decision Tree classifier
dt_clf = DecisionTreeClassifier(random_state=42)

# Train the model
dt_clf.fit(X_train_scaled, y_train)
# Predict on the test set
y_pred = dt_clf.predict(X_test_scaled)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9431692411337279
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     11713
           1       0.93      0.94      0.94      9068

    accuracy                           0.94     20781
   macro avg       0.94      0.94      0.94     20781
weighted avg       0.94      0.94      0.94     20781
```

[21]:
```
# Initialize the Decision Tree classifier
dt_clf = DecisionTreeClassifier(random_state=42)

# Perform cross-validation
cv_scores = cross_val_score(dt_clf, X_train_scaled, y_train, cv=5,␣
 ↪scoring='accuracy')

# Print the average cross-validation score
print("Cross-Validation Accuracy Scores", np.mean(cv_scores))

# Train the model on the entire training set
dt_clf.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = dt_clf.predict(X_test_scaled)

# Evaluate the model
```

```
print("Test Set Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Cross-Validation Accuracy Scores 0.9426512732746206
Test Set Accuracy: 0.9431692411337279
Classification Report:
               precision    recall  f1-score   support

           0       0.95      0.95      0.95     11713
           1       0.93      0.94      0.94      9068

    accuracy                           0.94     20781
   macro avg       0.94      0.94      0.94     20781
weighted avg       0.94      0.94      0.94     20781
```