In [28]:
```
#1 Import Libraries & Datasets
# (panda is used analyzing, cleaning, exploring, and manipulating data)
import pandas as pd
# (numpy helps perform operations and work wutg arrays)
import numpy as np
# (seaborn allows the display of statistical graphics and vizualizations of data)
import seaborn as sns
# (matplot creates static, animated, and interactive visualizations like charts and
import matplotlib.pyplot as plt
```

In [11]:
```
#2 Issue: Had to install seaborn library to the notebook before importing
!pip install seaborn
```

```
Requirement already satisfied: seaborn in /opt/conda/lib/python3.10/site-packages
(0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /opt/conda/lib/python3.10/sit
e-packages (from seaborn) (1.26.4)
Requirement already satisfied: pandas>=1.2 in /opt/conda/lib/python3.10/site-package
s (from seaborn) (2.1.4)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /opt/conda/lib/python3.10/
site-packages (from seaborn) (3.8.4)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.10/site-pa
ckages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.10/site-packag
es (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.10/site-p
ackages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/conda/lib/python3.10/site-p
ackages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.10/site-pac
kages (from matplotlib!=3.6.1,>=3.4->seaborn) (23.2)
Requirement already satisfied: pillow>=8 in /opt/conda/lib/python3.10/site-packages
(from matplotlib!=3.6.1,>=3.4->seaborn) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /opt/conda/lib/python3.10/site-pa
ckages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.10/sit
e-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-packag
es (from pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: tzdata>=2022.1 in /opt/conda/lib/python3.10/site-pack
ages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packages
(from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
```

In [13]:
```
#3 Import the cvs file with the dataset of the credit card users and their history
creditcard_df = pd.read_csv('UCI_Credit_Card.csv')
```

In [17]:
```
#4 Display the credit card dataset
# (default.payment.next.month: Will the cusomter default their payment? => 1=yes, 0
# (LIMIT_BAL: amount of credit given in New Taiwanese (NT) dollars)
# (SEX:  1=male & 2=female)
# (EDUCATION: 1=graduated school, 2=university, 3=high school, 4=others, 5=unkown,
# (MARRIAGE: 1=married, 2=single, 3=others)
# (AGE: Age in years)
```

```
# (PAY_0: Repayment statuns in 09 2005, PAY_2: ... in 08 2005, PAY_3: ... in 07 200
# (PAY_#: -1=pay duly, 1=payment delay for 1 month, 2=payment delay for 2 months, .
# (BILL_AMT1: Amount of bill statement in 09 2005 (in NT dollar), BILL_AMT2: ... in
# (PAY_AMT1: Amount of previous payment in 09 2005 (in NT dollar), PAY_AMT2: ... in
creditcard_df
```

Out[17]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_
0	1	20000.0	2	2	1	24	2	2	-1	-
1	2	120000.0	2	2	2	26	-1	2	0	
2	3	90000.0	2	2	2	34	0	0	0	
3	4	50000.0	2	2	1	37	0	0	0	
4	5	50000.0	1	2	1	57	-1	0	-1	
...	...	...	...	...	...	...	...	...	...	
29995	29996	220000.0	1	3	1	39	0	0	0	
29996	29997	150000.0	1	3	2	43	-1	-1	-1	-
29997	29998	30000.0	1	2	2	37	4	3	2	-
29998	29999	80000.0	1	3	1	41	1	-1	0	
29999	30000	50000.0	1	2	1	46	0	0	0	

30000 rows × 25 columns

In [19]:

```
#5 Print the DataFrame, which displays the number of columns, columns label, column
creditcard_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   ID                          30000 non-null  int64  
 1   LIMIT_BAL                   30000 non-null  float64
 2   SEX                         30000 non-null  int64  
 3   EDUCATION                   30000 non-null  int64  
 4   MARRIAGE                    30000 non-null  int64  
 5   AGE                         30000 non-null  int64  
 6   PAY_0                       30000 non-null  int64  
 7   PAY_2                       30000 non-null  int64  
 8   PAY_3                       30000 non-null  int64  
 9   PAY_4                       30000 non-null  int64  
 10  PAY_5                       30000 non-null  int64  
 11  PAY_6                       30000 non-null  int64  
 12  BILL_AMT1                   30000 non-null  float64
 13  BILL_AMT2                   30000 non-null  float64
 14  BILL_AMT3                   30000 non-null  float64
 15  BILL_AMT4                   30000 non-null  float64
 16  BILL_AMT5                   30000 non-null  float64
 17  BILL_AMT6                   30000 non-null  float64
 18  PAY_AMT1                    30000 non-null  float64
 19  PAY_AMT2                    30000 non-null  float64
 20  PAY_AMT3                    30000 non-null  float64
 21  PAY_AMT4                    30000 non-null  float64
 22  PAY_AMT5                    30000 non-null  float64
 23  PAY_AMT6                    30000 non-null  float64
 24  default.payment.next.month  30000 non-null  int64  
dtypes: float64(13), int64(12)
memory usage: 5.7 MB
```

In [20]:
```
#6 Calculate the average, minimum and maximum values for the Limit Balance, and max
creditcard_df.describe()
```

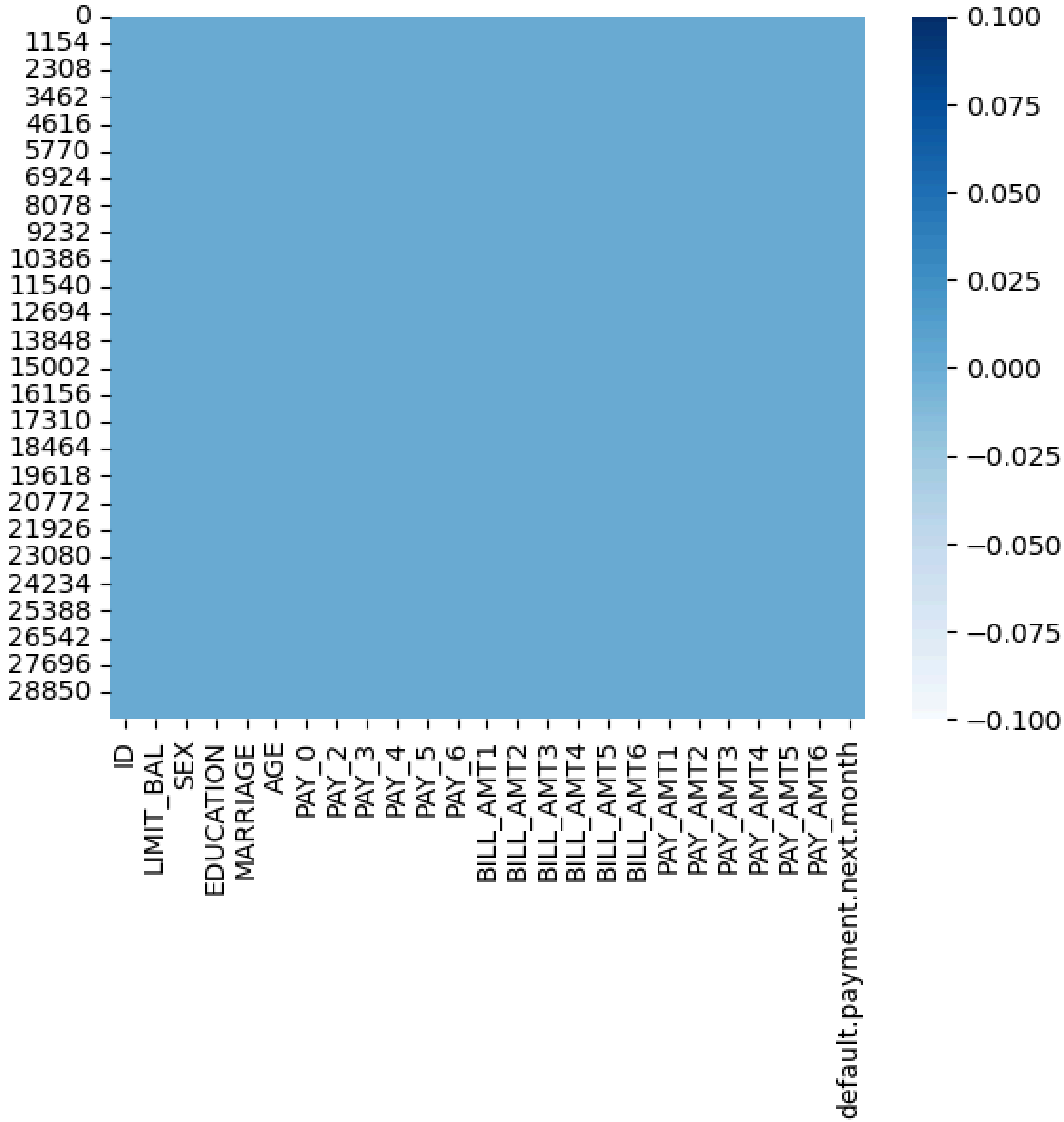
Out[20]:

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	A
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000	30000.0000
mean	15000.500000	167484.322667	1.603733	1.853133	1.551867	35.4855
std	8660.398374	129747.661567	0.489129	0.790349	0.521970	9.2179
min	1.000000	10000.000000	1.000000	0.000000	0.000000	21.0000
25%	7500.750000	50000.000000	1.000000	1.000000	1.000000	28.0000
50%	15000.500000	140000.000000	2.000000	2.000000	2.000000	34.0000
75%	22500.250000	240000.000000	2.000000	2.000000	2.000000	41.0000
max	30000.000000	1000000.000000	2.000000	6.000000	3.000000	79.0000

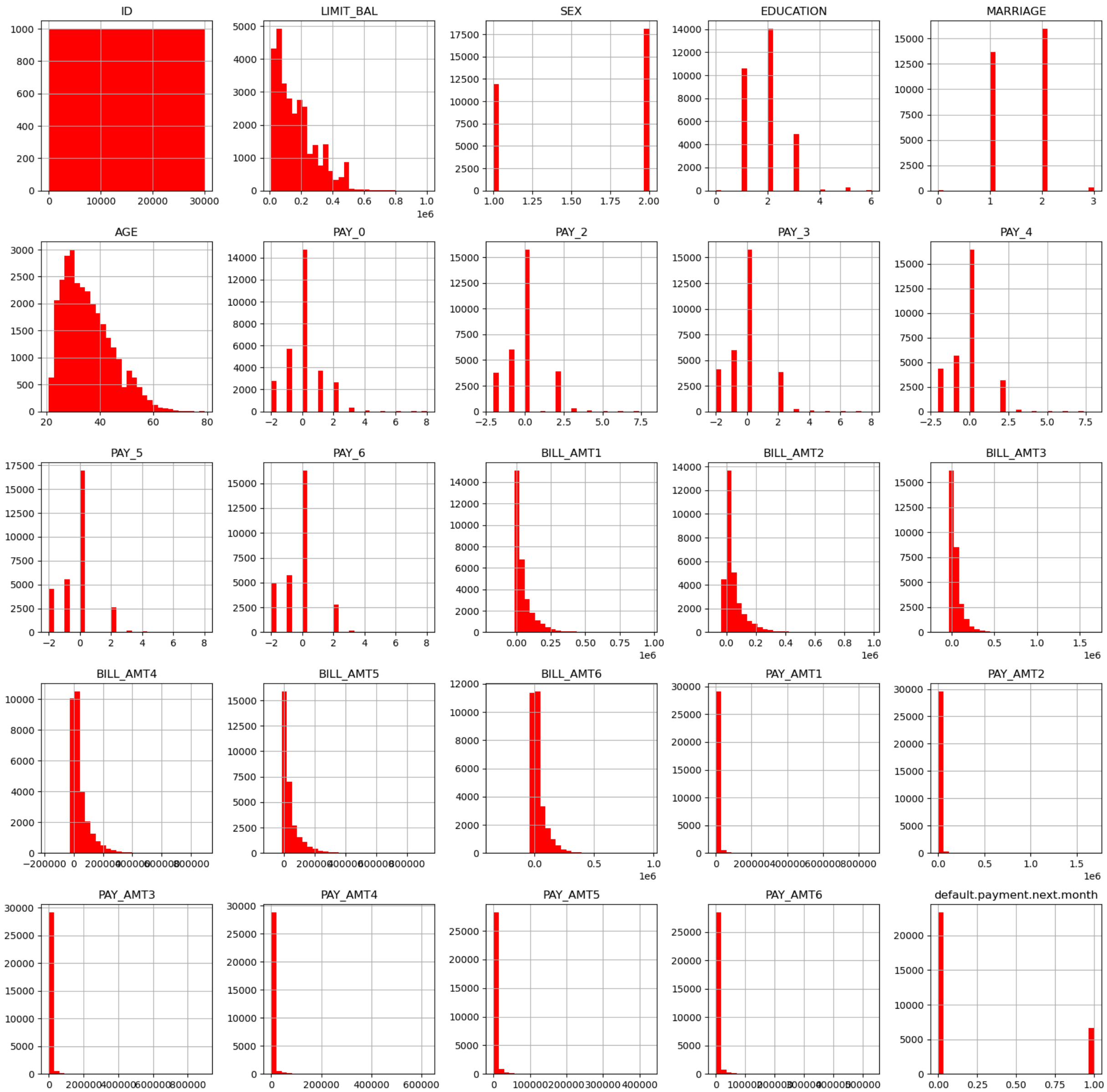
8 rows × 25 columns

In [21]:
```
#7 Verify if the data contains Null elements or not using seaborn and heatmap (ther
sns.heatmap(creditcard_df.isnull(), cmap = 'Blues')
```

Out[21]: <Axes: >



In [29]:
```
#8 Plot all the different data into a histogram to showcase the tendency using matp
creditcard_df.hist(bins = 30, figsize = (20,20), color = 'r')
plt.show()
```



In [31]:
```
#9 Clean the data by removing the ID column (axiz=1 means removing the entire colum
creditcard_df.drop(['ID'], axis=1, inplace=True)
```

In [32]:
```
#10 Print the dataset again to check if the ID column has been dropped
creditcard_df
```

Out[32]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_
0	20000.0	2	2	1	24	2	2	-1	-1	
1	120000.0	2	2	2	26	-1	2	0	0	
2	90000.0	2	2	2	34	0	0	0	0	
3	50000.0	2	2	1	37	0	0	0	0	
4	50000.0	1	2	1	57	-1	0	-1	0	
...	...	...	...	...	...	...	...	...	...	
29995	220000.0	1	3	1	39	0	0	0	0	
29996	150000.0	1	3	2	43	-1	-1	-1	-1	
29997	30000.0	1	2	2	37	4	3	2	-1	
29998	80000.0	1	3	1	41	1	-1	0	0	
29999	50000.0	1	2	1	46	0	0	0	0	

30000 rows × 24 columns

In [34]:

```
#11 Group the customers that default together and the one that did not in another g
cc_default_df    = creditcard_df[creditcard_df['default.payment.next.month'] == 1]
cc_nodefault_df  = creditcard_df[creditcard_df['default.payment.next.month'] == 0]
```

In [35]:

```
#12 Display the total and % of customers that defaults on their credit card payment
#Calculate the total number of customers (len, is the length of the column or # of
print("Total =", len(creditcard_df))
#Calculate the number of customer who defaulted
print("Number of customers who defaulted on their credit card payments =", len(cc_d
#Calculate the % of people who defaulted
print("Percentage of customers who defaulted on their credit card payments =", 1.*l
#Calculate the number of customer who did not default
print("Number of customers who did not default on their credit card payments (paid
#Calculate the % of people who did not default
print("Percentage of customers who did not default on their credit card payments (p
```

```
Total = 30000
Number of customers who defaulted on their credit card payments = 6636
Percentage of customers who defaulted on their credit card payments = 22.12 %
Number of customers who did not default on their credit card payments (paid their ba
lance) = 23364
Percentage of customers who did not default on their credit card payments (paid thei
r balance)= 77.88000000000001 %
```

In [36]:

```
#13 Compare the mean and standard of deviation (std) of the customers who defaulted
cc_default_df.describe()
```

Out[36]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	
count	6636.000000	6636.000000	6636.000000	6636.000000	6636.000000	6636.000000	66
mean	130109.656420	1.567058	1.894665	1.528029	35.725738	0.668174	
std	115378.540571	0.495520	0.728096	0.525433	9.693438	1.383252	
min	10000.000000	1.000000	1.000000	0.000000	21.000000	-2.000000	
25%	50000.000000	1.000000	1.000000	1.000000	28.000000	0.000000	
50%	90000.000000	2.000000	2.000000	2.000000	34.000000	1.000000	
75%	200000.000000	2.000000	2.000000	2.000000	42.000000	2.000000	
max	740000.000000	2.000000	6.000000	3.000000	75.000000	8.000000	

8 rows × 24 columns

In [37]:
```
#14 Compare the mean and standard of deviation (std) of the customers who did not d
cc_nodefault_df.describe()
```

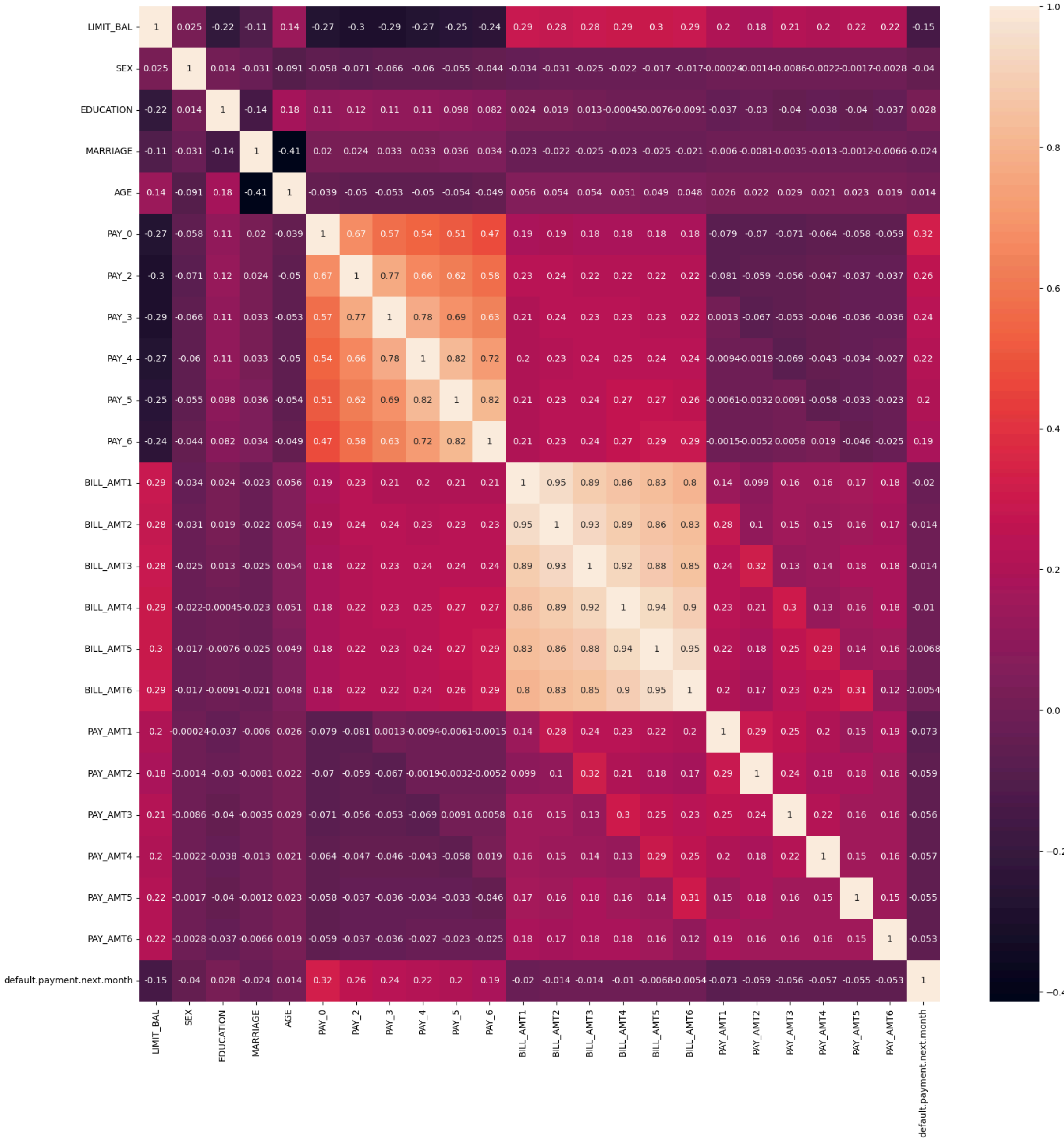
Out[37]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY
count	23364.000000	23364.000000	23364.000000	23364.000000	23364.000000	23364.0000
mean	178099.726074	1.614150	1.841337	1.558637	35.417266	-0.2112
std	131628.359660	0.486806	0.806780	0.520794	9.077355	0.9524
min	10000.000000	1.000000	0.000000	0.000000	21.000000	-2.0000
25%	70000.000000	1.000000	1.000000	1.000000	28.000000	-1.0000
50%	150000.000000	2.000000	2.000000	2.000000	34.000000	0.0000
75%	250000.000000	2.000000	2.000000	2.000000	41.000000	0.0000
max	1000000.000000	2.000000	6.000000	3.000000	79.000000	8.0000

8 rows × 24 columns

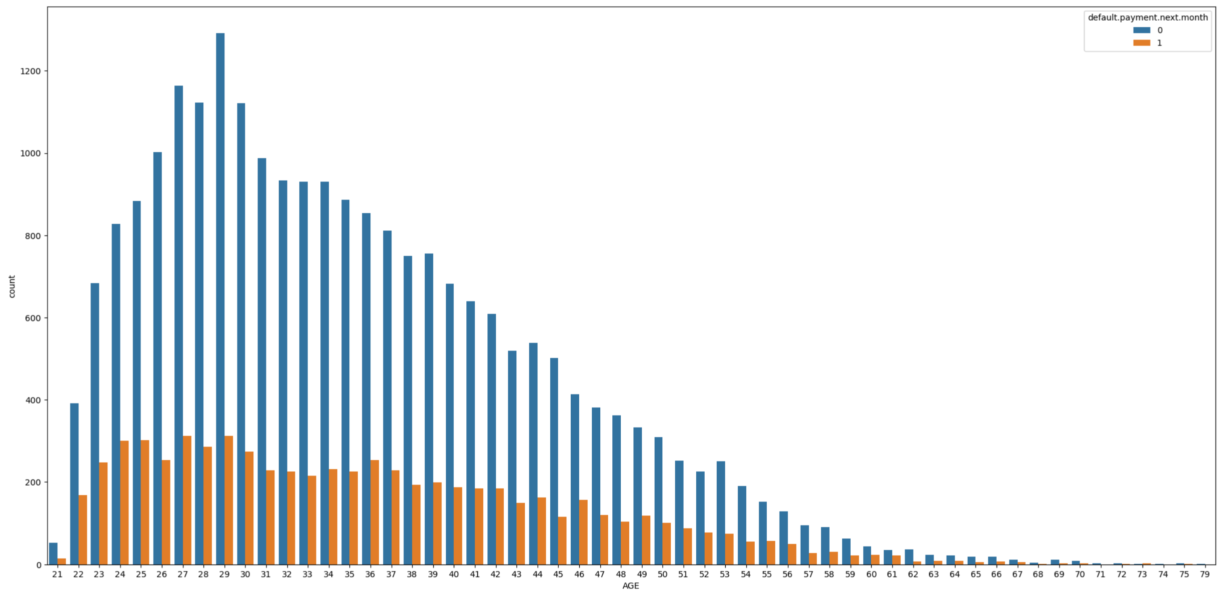
In [39]:
```
#15 Print the correlation of the credit card dataset
correlations = creditcard_df.corr()
f, ax = plt.subplots(figsize = (20,20))
sns.heatmap(correlations, annot = True)
```

Out[39]: <Axes: >



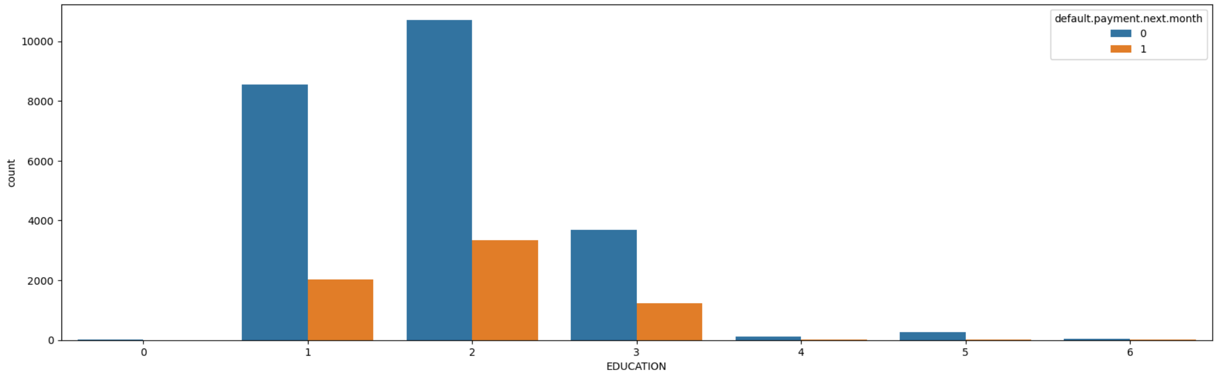
In [40]:
```
#15 Plot the default card payments and nodefault card payments based on the age of
# Observation: The younger they are (21 years old) the more they default, the older
plt.figure(figsize=[25, 12])
sns.countplot(x = 'AGE', hue = 'default.payment.next.month', data = creditcard_df)
```

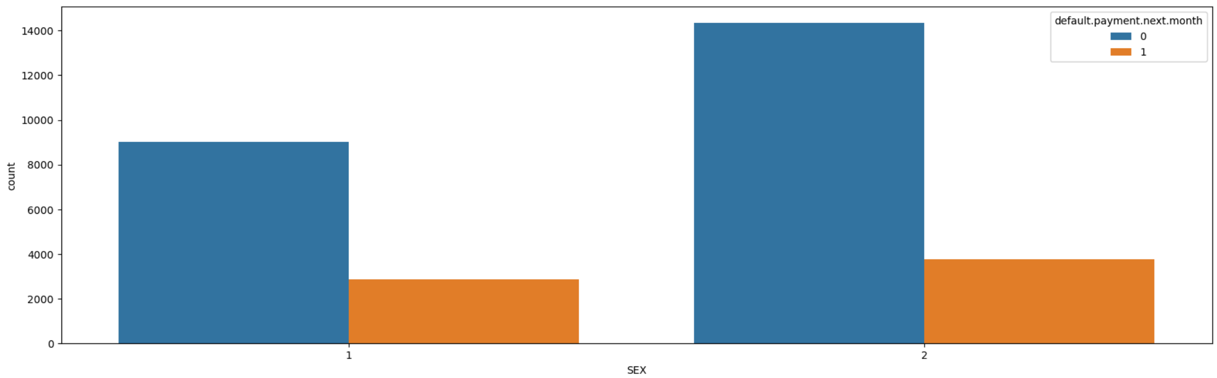
Out[40]: `<Axes: xlabel='AGE', ylabel='count'>`

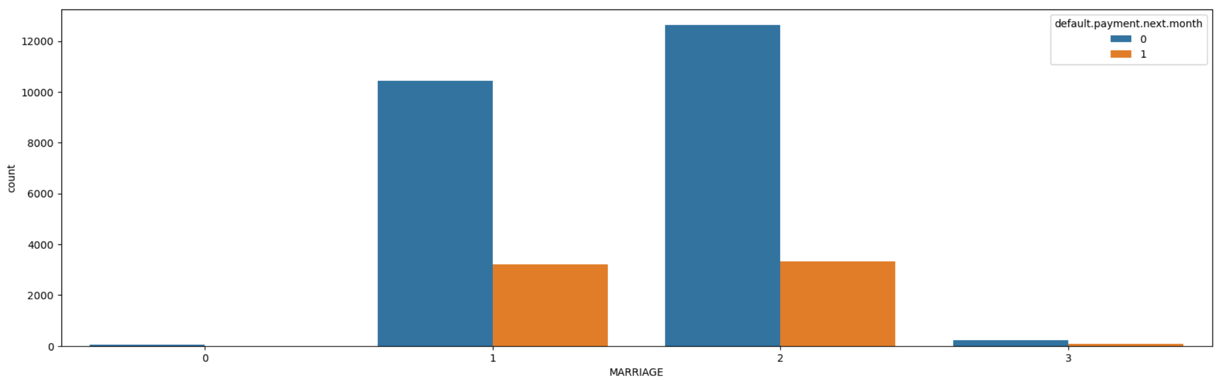


In [42]:
```
#16 Plot the same information (card default & no card default customers) based on t
plt.figure(figsize=[20,20])
plt.subplot(311)
sns.countplot(x = 'EDUCATION', hue = 'default.payment.next.month', data = creditcar
plt.subplot(312)
sns.countplot(x = 'SEX', hue = 'default.payment.next.month', data = creditcard_df)
plt.subplot(313)
sns.countplot(x = 'MARRIAGE', hue = 'default.payment.next.month', data = creditcard
```

Out[42]:
```
<Axes: xlabel='MARRIAGE', ylabel='count'>
```

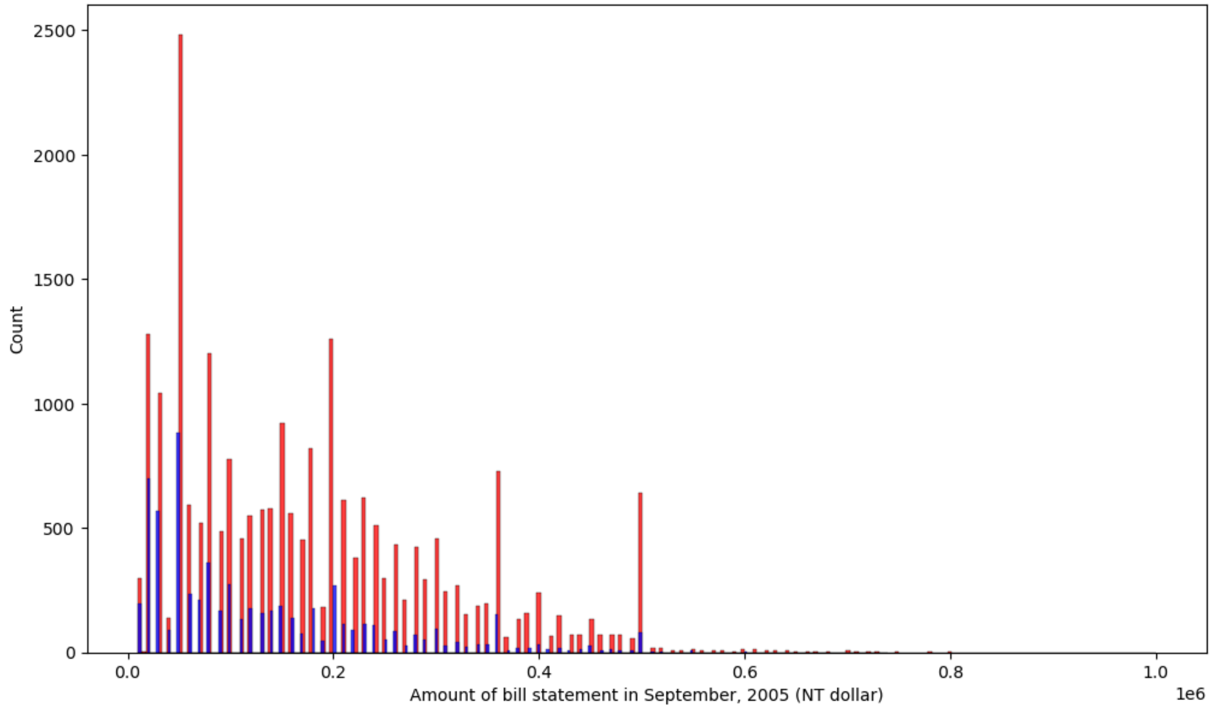






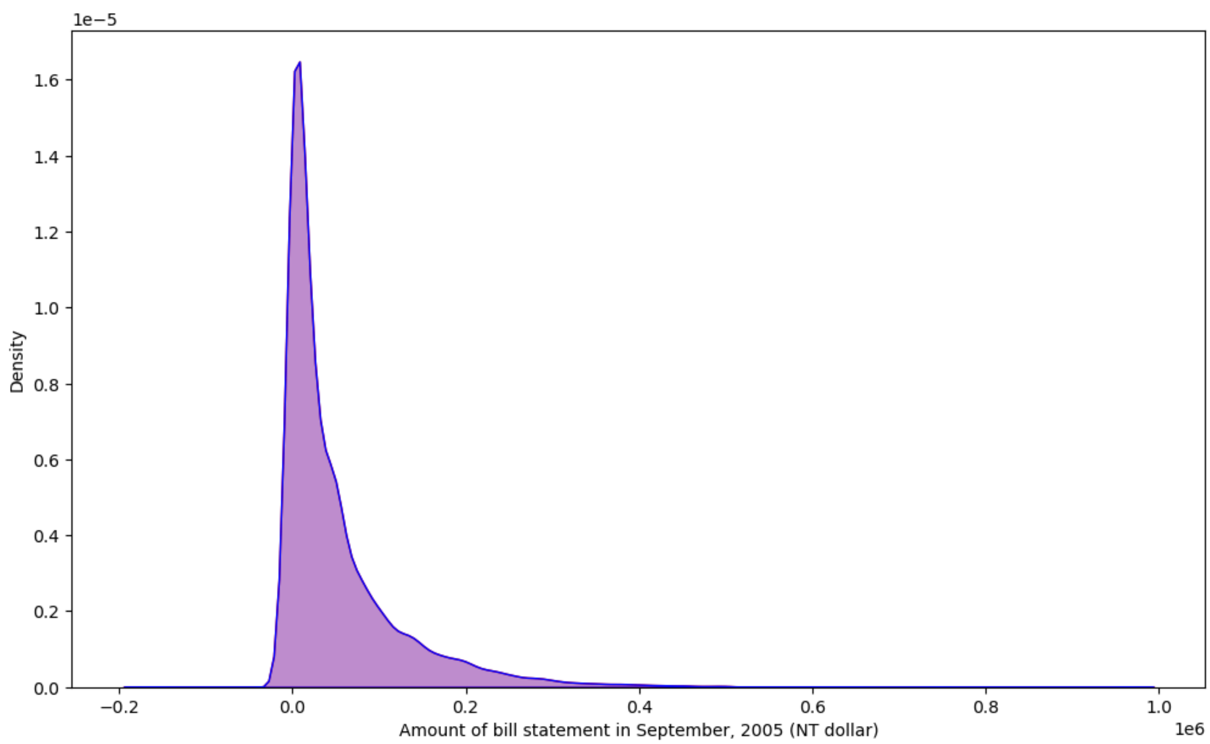
In [49]:
```
#17 Use KDE (Kernel Density Estimate) to visualize the probablity density at differ
# Plot to showcase the density of the Limit Balance
plt.figure(figsize=(12,7))
sns.histplot(cc_nodefault_df['LIMIT_BAL'], bins = 250, color = 'r')
sns.histplot(cc_default_df['LIMIT_BAL'], bins = 250, color = 'b')
plt.xlabel('Amount of bill statement in September, 2005 (NT dollar)')
```

Out[49]:
```
Text(0.5, 0, 'Amount of bill statement in September, 2005 (NT dollar)')
```



In [53]:
```
#17 Use KDE (Kernel Density Estimate) to visualize the density of the bill amount f
plt.figure(figsize=(12,7))
sns.kdeplot(cc_nodefault_df['BILL_AMT1'], label = 'Customers who did not default (p
sns.kdeplot(cc_nodefault_df['BILL_AMT1'], label = 'Customers who defaulted (did not
plt.xlabel('Amount of bill statement in September, 2005 (NT dollar)')
```

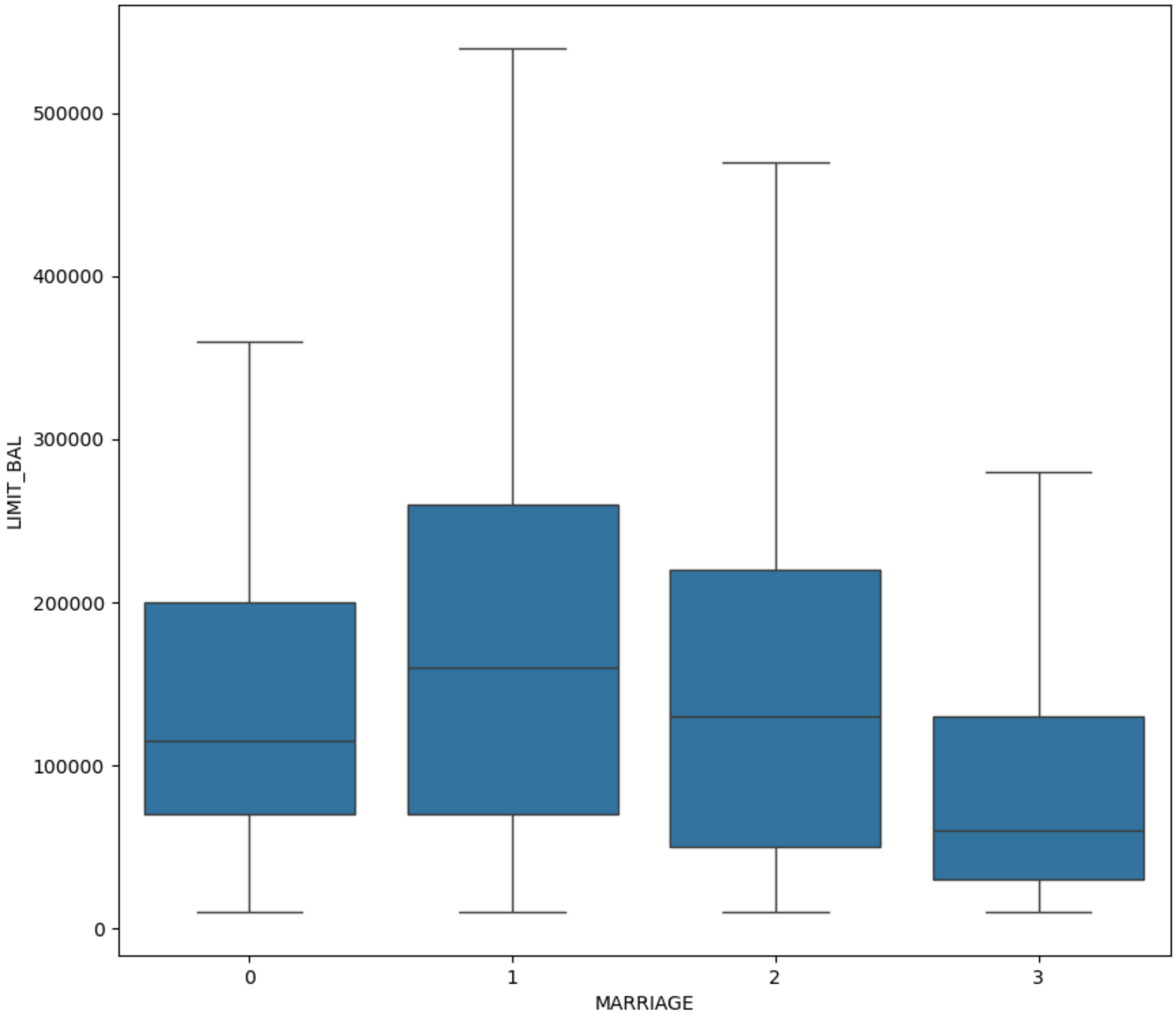
Out[53]: Text(0.5, 0, 'Amount of bill statement in September, 2005 (NT dollar)')

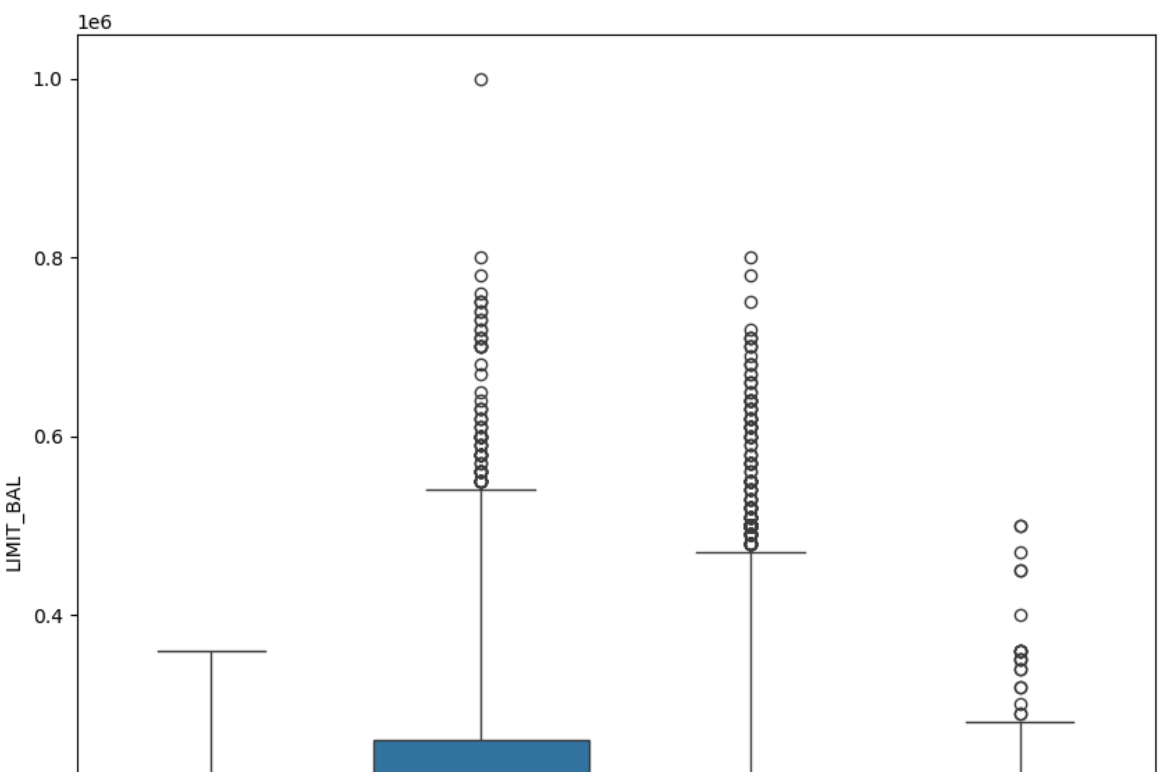


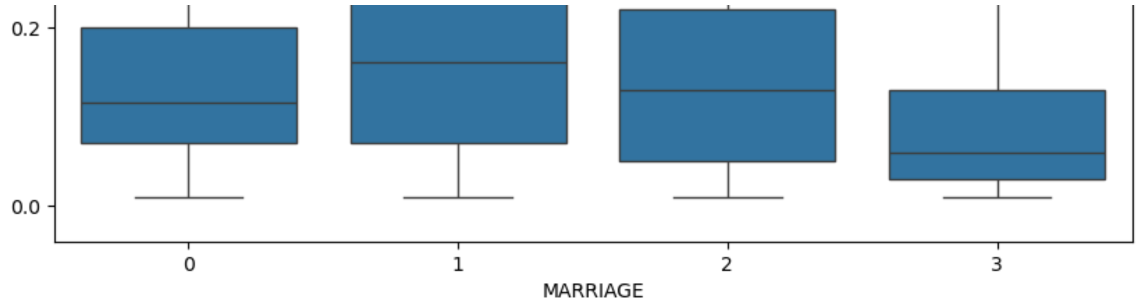
In [56]:
```
#17 Print box plot using Seaborn displaying the correlation between the Marriage an
plt.figure(figsize=[10,20])
#Without outliers
```

```
plt.subplot(211)
sns.boxplot(x = 'MARRIAGE', y = 'LIMIT_BAL', data = creditcard_df, showfliers = Fal
#With outliers
plt.subplot(212)
sns.boxplot(x = 'MARRIAGE', y = 'LIMIT_BAL', data = creditcard_df)
```

Out[56]: <Axes: xlabel='MARRIAGE', ylabel='LIMIT_BAL'>

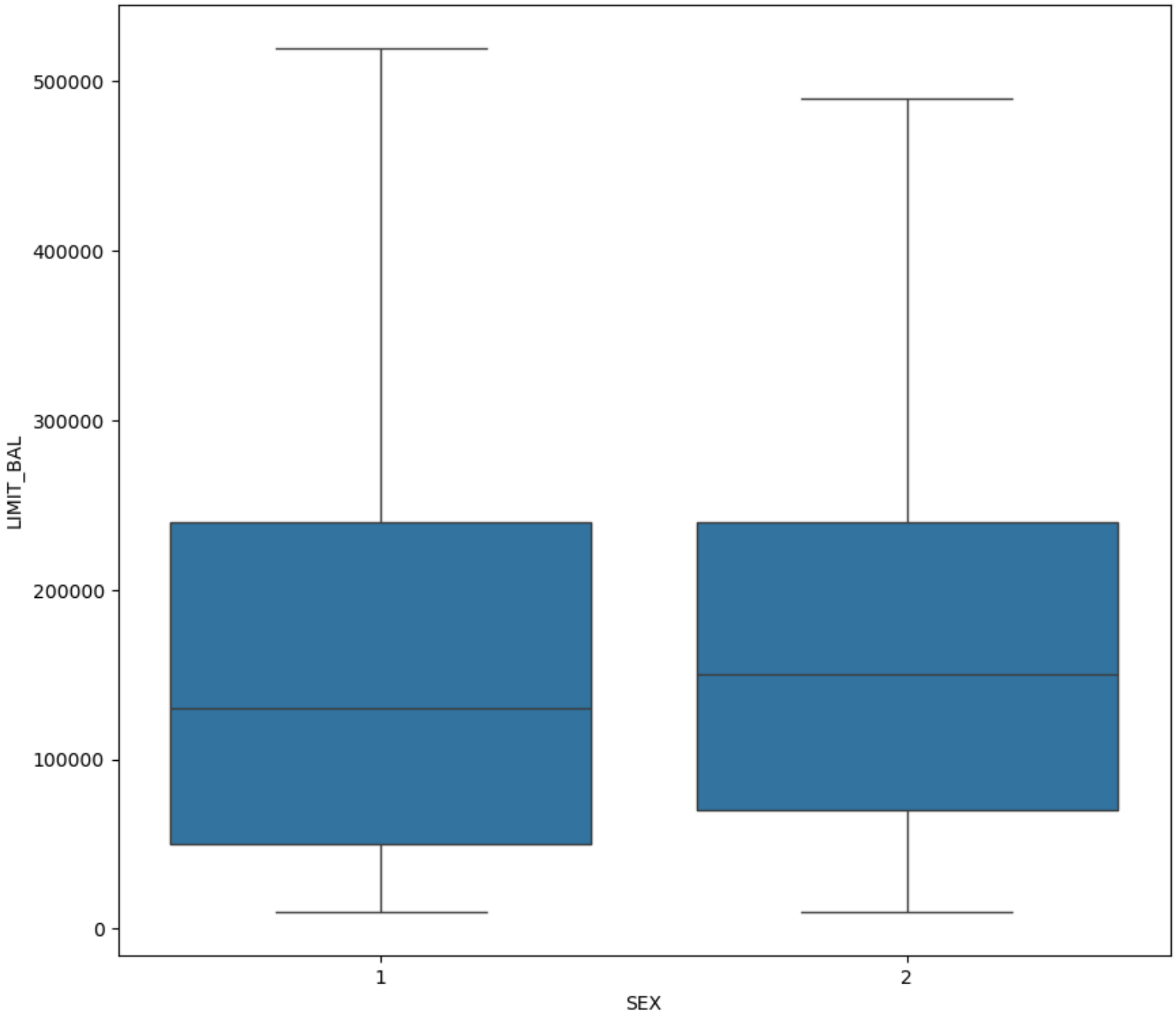

500000
400000
300000
200000
100000
0
LIMIT_BAL
0
1
2
3
MARRIAGE

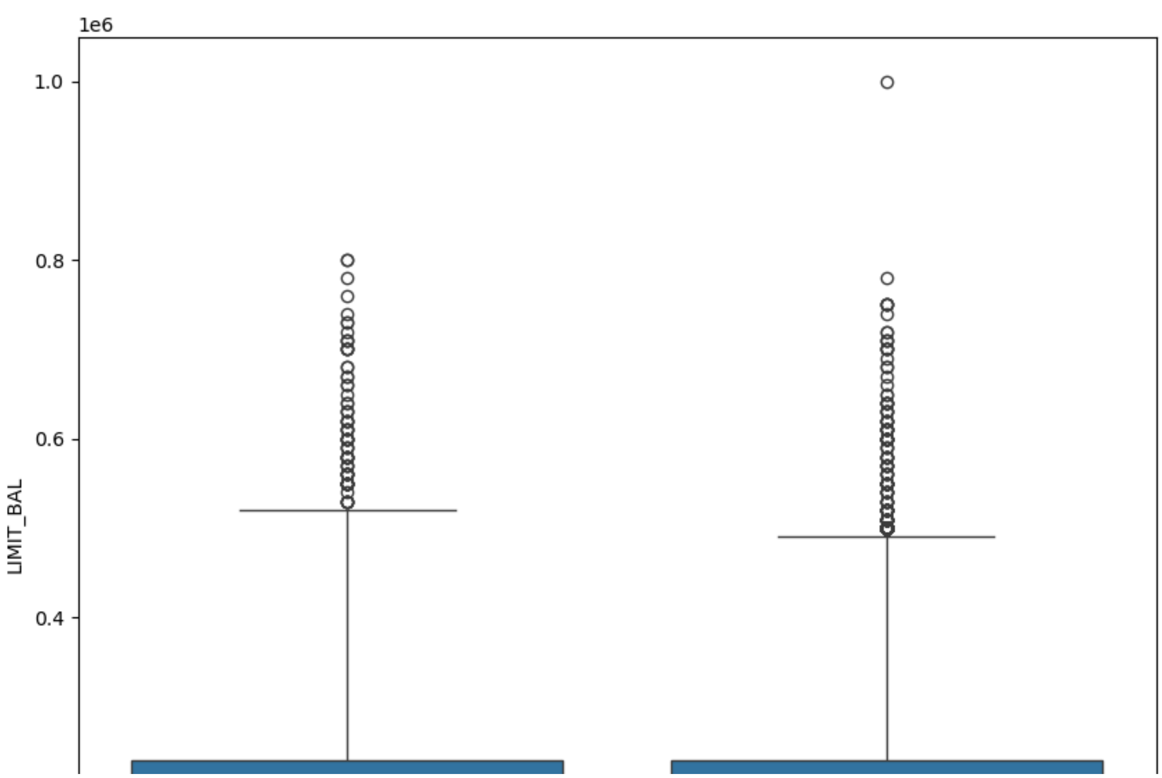

1e6
1.0
0.8
0.6
0.4
LIMIT_BAL

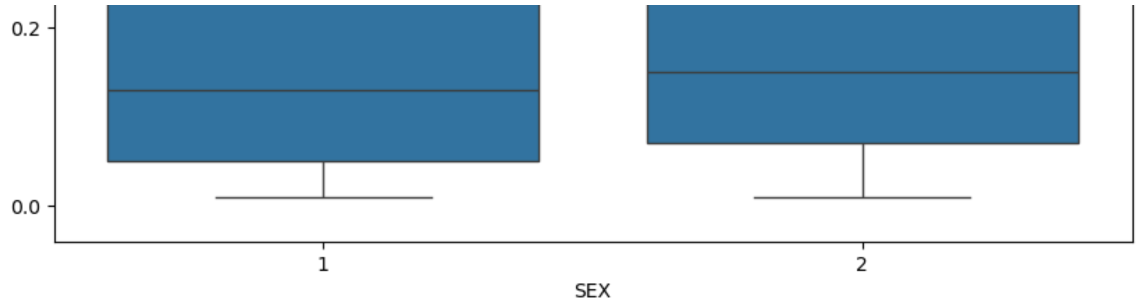


In [57]:
```
#18 Plot the boxplot for the Limit Balance compared to the Sex column & the same bo
plt.figure(figsize=[10,20])
plt.subplot(211)
sns.boxplot(x = 'SEX', y = 'LIMIT_BAL', data = creditcard_df, showfliers = False)
plt.subplot(212)
sns.boxplot(x = 'SEX', y = 'LIMIT_BAL', data = creditcard_df)
```

Out[57]: <Axes: xlabel='SEX', ylabel='LIMIT_BAL'>


500000
400000
300000
200000
100000
0
LIMIT_BAL
1
2
SEX


1e6
1.0
0.8
0.6
0.4
LIMIT_BAL



In [58]:
```
#19 Check the dataset
creditcard_df
```

Out[58]:

	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_
0	20000.0	2	2	1	24	2	2	-1	-1	
1	120000.0	2	2	2	26	-1	2	0	0	
2	90000.0	2	2	2	34	0	0	0	0	
3	50000.0	2	2	1	37	0	0	0	0	
4	50000.0	1	2	1	57	-1	0	-1	0	
...	...	...	...	...	...	...	...	...	...	
29995	220000.0	1	3	1	39	0	0	0	0	
29996	150000.0	1	3	2	43	-1	-1	-1	-1	
29997	30000.0	1	2	2	37	4	3	2	-1	
29998	80000.0	1	3	1	41	1	-1	0	0	
29999	50000.0	1	2	1	46	0	0	0	0	

30000 rows × 24 columns

In [66]:
```
#20 Combine the Education, Sex, and Marriage variables into one group
X_cat = creditcard_df[['SEX','EDUCATION','MARRIAGE']]
X_cat
```

Out[66]:

	SEX	EDUCATION	MARRIAGE
0	2	2	1
1	2	2	2
2	2	2	2
3	2	2	1
4	1	2	1
...	...	...	...
29995	1	3	1
29996	1	3	2
29997	1	2	2
29998	1	3	1
29999	1	2	1

30000 rows × 3 columns

In [67]:

```
#21 Expand the data for the columns to turn into one hote encoder
from sklearn.preprocessing import OneHotEncoder
onehotencoder = OneHotEncoder()
X_cat = onehotencoder.fit_transform(X_cat).toarray()
```

In [68]:

```
#22 Check the modification of the size
X_cat.shape
```

Out[68]: (30000, 13)

In [70]:

```
#23 Convert those into a dataframe
X_cat = pd.DataFrame(x_cat)
```

In [72]:

```
#24 Check the dataframe
X_cat
```

Out[72]:

	0	1	2	3	4	5	6	7	8	9	...	16	17	18	19	20	21	22	2
0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.
1	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.
2	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.
3	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.
4	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
29995	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.
29996	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.
29997	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.
29998	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.
29999	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.

30000 rows × 26 columns

In [73]:

```
#24 Separate the data generated & Print it
X_numerical = creditcard_df[['LIMIT_BAL', 'AGE', 'PAY_0', 'PAY_2', 'PAY_3', 'PAY_4'
                'BILL_AMT1','BILL_AMT2', 'BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BI
                'PAY_AMT1', 'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AM
X_numerical
```

Out[73]:

	LIMIT_BAL	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	PAY_6	BILL_AMT1	BILL_AM
0	20000.0	24	2	2	-1	-1	-2	-2	3913.0	310
1	120000.0	26	-1	2	0	0	0	2	2682.0	172
2	90000.0	34	0	0	0	0	0	0	29239.0	1402
3	50000.0	37	0	0	0	0	0	0	46990.0	4823
4	50000.0	57	-1	0	-1	0	0	0	8617.0	567
...	...	...	...	...	...	...	...	...	...	
29995	220000.0	39	0	0	0	0	0	0	188948.0	19281
29996	150000.0	43	-1	-1	-1	-1	0	0	1683.0	182
29997	30000.0	37	4	3	2	-1	0	0	3565.0	335
29998	80000.0	41	1	-1	0	0	0	-1	-1645.0	7837
29999	50000.0	46	0	0	0	0	0	0	47929.0	4890

30000 rows × 20 columns

In [83]:

```
#25 Concatinate the categorical and numerical data
X_all = pd.concat([X_cat, X_numerical], axis = 1)
#Had to convert all data into string (same data type) for sklearn to process the da
X_all.columns = X_all.columns.astype(str)
X_all
```

Out[83]:

	0	1	2	3	4	5	6	7	8	9	...	BILL_AMT3	BILL_AMT4	BILL_AMT
0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	689.0	0.0	0
1	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	2682.0	3272.0	3455
2	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	13559.0	14331.0	14948
3	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	...	49291.0	28314.0	28959
4	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	35835.0	20940.0	19146
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
29995	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	208365.0	88004.0	31237
29996	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	3502.0	8979.0	5190
29997	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	2758.0	20878.0	20582
29998	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	0.0	...	76304.0	52774.0	11855
29999	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	...	49764.0	36535.0	32428

30000 rows × 46 columns

In [84]:
```
#26 Scaling for XGBOOST
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X = scaler.fit_transform(X_all)
```

In [87]:
```
#27 Define the output as default.payment.next.month
y = creditcard_df['default.payment.next.month']
y
```

Out[87]:
```
0        1
1        1
2        0
3        0
4        0
        ..
29995    0
29996    0
29997    1
29998    1
29999    1
Name: default.payment.next.month, Length: 30000, dtype: int64
```

In [89]:
```
#28 Separate the data for training and testing
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
```

In [90]:
```
#29 Check if the data got splitted into training
X_train.shape
```

Out[90]: (22500, 46)

In [91]:
```
#30 Check if the data got splitted into testing
X_test.shape
```

Out[91]: (7500, 46)

In [92]:
```
#31 Install xgboost
!pip install xgboost
```

```
Requirement already satisfied: xgboost in /opt/conda/lib/python3.10/site-packages
(1.7.6)
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages (fro
m xgboost) (1.26.4)
Requirement already satisfied: scipy in /opt/conda/lib/python3.10/site-packages (fro
m xgboost) (1.11.4)
```

In [96]:
```
#32 Import xgboost & traom the regressor model
import xgboost as xgb
model = xgb.XGBClassifier(objective = 'reg:squarederror', learning_rate = 0.1, max_
model.fit(X_train, y_train)
```

Out[96]:

XGBClassifier

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types
=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_typ
e=None,
              interaction_constraints=None, learning_rate=0.1, max_bin=
None,
```

In [97]:
```
#33 Feed the model with the X_test and get a prediction for the algorithm
from sklearn.metrics import accuracy_score
y_pred = model.predict(X_test)
```

In [99]:
```
#34 Run the prediction
y_pred
```

Out[99]: array([1, 0, 0, ..., 0, 0, 0])

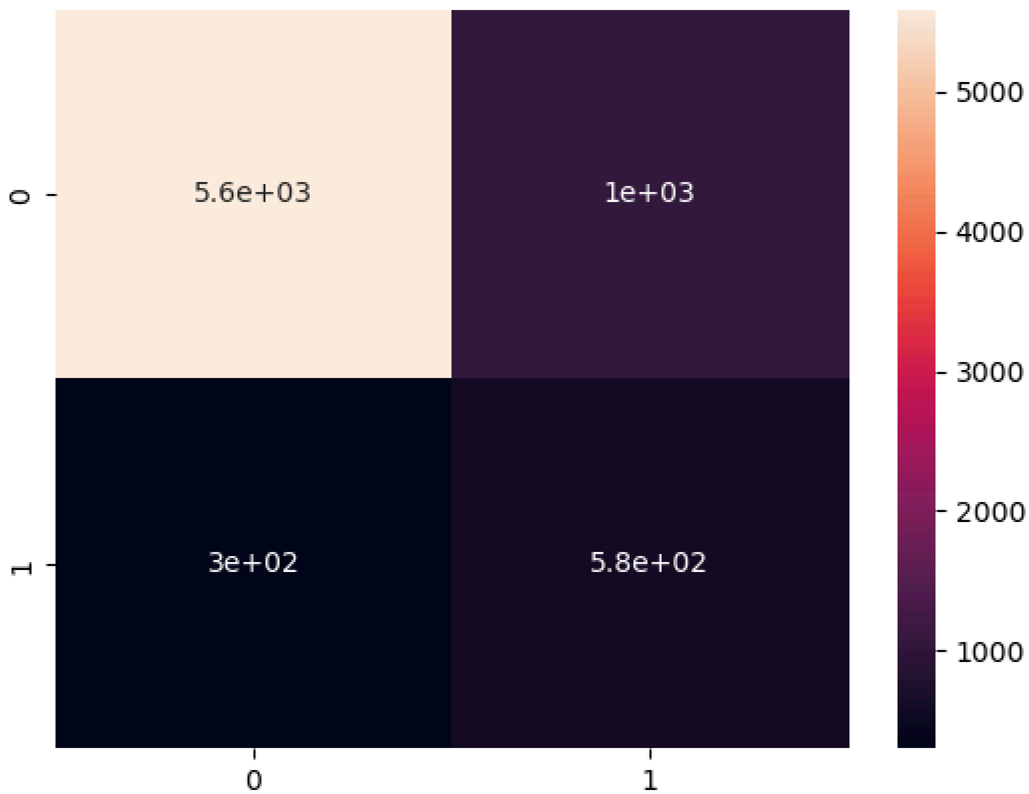
In [104...
```
#35 Print the accuracy score of the model
from sklearn.metrics import confusion_matrix, classification_report
print("Accuracy {} %".format( 100 * accuracy_score(y_pred, y_test)))
```

```
Accuracy 82.17333333333333 %
```

In [105...
```
#36 Print the confusion matrix
cm = confusion_matrix(y_pred, y_test)
```

```python
sns.heatmap(cm, annot=True)
```

Out[105… <Axes: >



In [106…

```python
#37 Print the classification report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.84      0.95      0.89      5887
           1       0.66      0.36      0.46      1613

    accuracy                           0.82      7500
   macro avg       0.75      0.65      0.68      7500
weighted avg       0.80      0.82      0.80      7500
```

In [107…

```python
#38 Train XGBoost with large number of estimators and more depth to improve the acc
import xgboost as xgb
model = xgb.XGBClassifier(objective = 'reg:squarederror', learning_rate = 0.1, max_
model.fit(X_train, y_train)
```

Out[107…

XGBClassifier

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types
=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_typ
e=None,
              interaction_constraints=None, learning_rate=0.1, max_bin=
None,
```

In [115…

```
#39 Feed the model with the X_test and get a prediction for the algorithm
from sklearn.metrics import accuracy_score
y_pred = model.predict(X_test)
```

In [116…

```
#40 Run the prediction
y_pred
```

Out[116…

```
array([1, 0, 0, ..., 0, 0, 0])
```

In [117…

```
#41 Print the accuracy score of the model
from sklearn.metrics import confusion_matrix, classification_report
print("Accuracy {} %".format( 100 * accuracy_score(y_pred, y_test)))
```

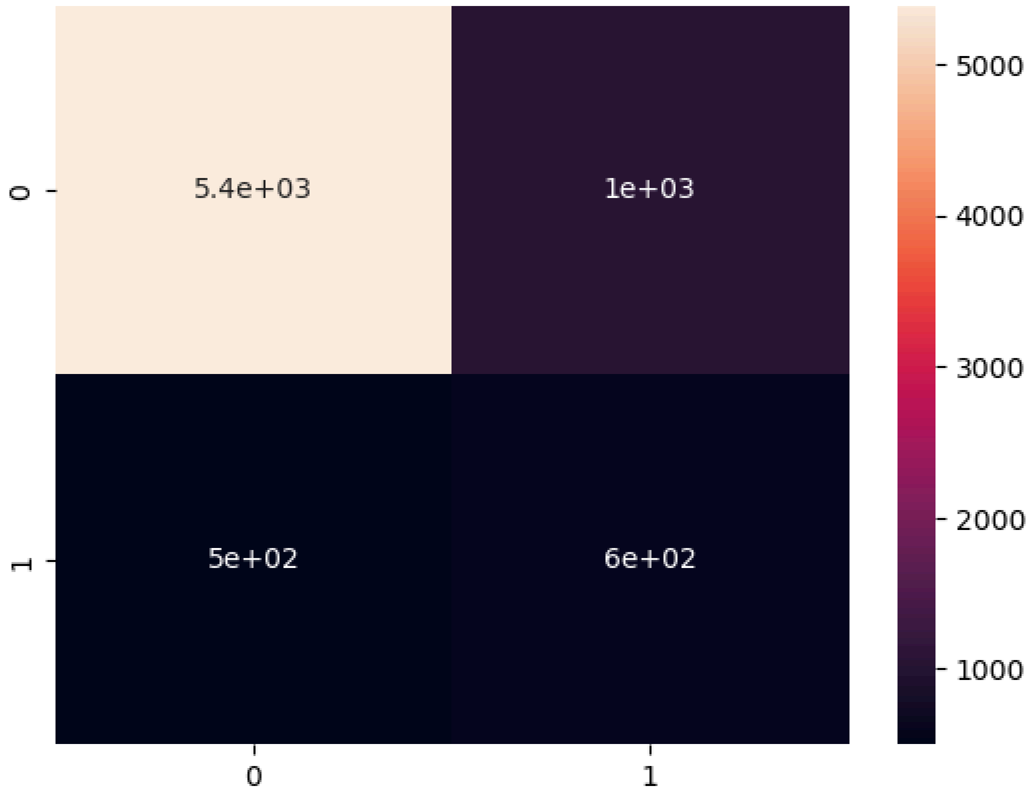
```
Accuracy 79.78666666666666 %
```

In [118…

```
#42 Print the confusion matrix
cm = confusion_matrix(y_pred, y_test)
sns.heatmap(cm, annot=True)
```

Out[118…

```
<Axes: >
```



In [119…

```
#43 Print the classification report
# Observation: The accuracy has decreased
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.84      0.92      0.88      5887
           1       0.54      0.37      0.44      1613

    accuracy                           0.80      7500
   macro avg       0.69      0.64      0.66      7500
weighted avg       0.78      0.80      0.78      7500
```

In [120…

```
#44 Improve XGBoost Parameters by using Grid Search
param_grid = {
# regularization parameter
        'gamma': [0.5, 1, 5],
# % of rows taken to build each tree
        'subsample': [0.6, 0.8, 1.0],
# number of columns used by each tree
        'colsample_bytree': [0.6, 0.8, 1.0],
# depth of each tree
        'max_depth': [3, 4, 5]
        }
```

In [121…

```
#45 Set the XGBoost model to train the data
from xgboost import XGBClassifier
xgb_model = XGBClassifier(learning_rate=0.01, n_estimators=100, objective='binary:l

#46 Use GridSearch to train the model with the different xgboost parameters
```

```
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(xgb_model, param_grid, refit = True, verbose = 4)
grid.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 81 candidates, totalling 405 fits
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
21 total time=   2.2s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
14 total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
21 total time=   1.8s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
22 total time=   1.9s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
24 total time=   2.0s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
22 total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
15 total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
21 total time=   1.9s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
23 total time=   2.0s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
22 total time=   1.8s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
13 total time=   2.1s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
12 total time=   1.9s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
16 total time=   1.7s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
14 total time=   1.7s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
16 total time=   1.8s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
21 total time=   2.7s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
15 total time=   2.5s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
21 total time=   2.1s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
21 total time=   2.1s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
22 total time=   2.1s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
20 total time=   2.1s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
16 total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
19 total time=   2.1s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
22 total time=   2.1s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
24 total time=   2.0s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
11 total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
12 total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
```

```
15 total time=   2.0s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
15 total time=   2.0s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
17 total time=   2.0s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
18 total time=   2.8s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
15 total time=   2.5s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
20 total time=   2.5s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
22 total time=   2.5s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
24 total time=   2.6s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
19 total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
16 total time=   2.6s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
21 total time=   2.6s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
19 total time=   2.6s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
24 total time=   2.5s
[CV 1/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
12 total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
12 total time=   2.5s
[CV 3/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
16 total time=   2.5s
[CV 4/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
17 total time=   2.5s
[CV 5/5] END colsample_bytree=0.6, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
18 total time=   2.5s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.6;, score=0.821
total time=   1.6s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.6;, score=0.814
total time=   1.6s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.6;, score=0.821
total time=   1.6s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.6;, score=0.822
total time=   1.6s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.6;, score=0.824
total time=   1.6s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.8;, score=0.822
total time=   1.6s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.8;, score=0.815
total time=   1.6s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.8;, score=0.821
total time=   1.6s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.8;, score=0.823
total time=   1.6s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=0.8;, score=0.822
total time=   1.6s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=1.0;, score=0.813
```

```
total time=   1.5s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=1.0;, score=0.812
total time=   1.5s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=1.0;, score=0.816
total time=   1.5s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=1.0;, score=0.814
total time=   1.5s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=3, subsample=1.0;, score=0.816
total time=   1.7s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.6;, score=0.821
total time=   2.1s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.6;, score=0.815
total time=   2.1s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.6;, score=0.821
total time=   2.0s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.6;, score=0.821
total time=   2.1s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.6;, score=0.822
total time=   2.3s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.8;, score=0.820
total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.8;, score=0.816
total time=   2.1s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.8;, score=0.819
total time=   2.0s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.8;, score=0.822
total time=   2.1s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=0.8;, score=0.824
total time=   2.1s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=1.0;, score=0.811
total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=1.0;, score=0.812
total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=1.0;, score=0.815
total time=   2.1s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=1.0;, score=0.815
total time=   2.0s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=4, subsample=1.0;, score=0.817
total time=   2.1s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.6;, score=0.818
total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.6;, score=0.815
total time=   2.6s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.6;, score=0.820
total time=   2.6s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.6;, score=0.822
total time=   2.7s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.6;, score=0.823
total time=   2.7s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.8;, score=0.818
total time=   2.6s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.8;, score=0.816
total time=   2.7s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.8;, score=0.822
total time=   3.6s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.8;, score=0.820
```

```
total time=   4.5s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=0.8;, score=0.825
total time=   2.8s
[CV 1/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=1.0;, score=0.812
total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=1.0;, score=0.812
total time=   2.5s
[CV 3/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=1.0;, score=0.816
total time=   2.6s
[CV 4/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=1.0;, score=0.817
total time=   2.7s
[CV 5/5] END colsample_bytree=0.6, gamma=1, max_depth=5, subsample=1.0;, score=0.818
total time=   2.6s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.6;, score=0.820
total time=   2.9s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.6;, score=0.814
total time=   2.9s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.6;, score=0.822
total time=   3.4s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.6;, score=0.822
total time=   2.3s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.6;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.8;, score=0.822
total time=   3.3s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.8;, score=0.815
total time=   3.0s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.8;, score=0.821
total time=   3.1s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.8;, score=0.823
total time=   3.1s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=0.8;, score=0.823
total time=   3.0s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=1.0;, score=0.813
total time=   3.0s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=1.0;, score=0.812
total time=   2.9s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=1.0;, score=0.816
total time=   2.8s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=1.0;, score=0.814
total time=   1.5s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=3, subsample=1.0;, score=0.816
total time=   1.5s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.6;, score=0.821
total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.6;, score=0.815
total time=   2.1s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.6;, score=0.821
total time=   2.1s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.6;, score=0.822
total time=   2.0s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.6;, score=0.823
total time=   2.1s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.8;, score=0.820
total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.8;, score=0.815
```

```
total time=   2.1s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.8;, score=0.820
total time=   2.1s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.8;, score=0.822
total time=   2.1s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=0.8;, score=0.824
total time=   2.1s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=1.0;, score=0.812
total time=   2.0s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=1.0;, score=0.812
total time=   2.0s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=1.0;, score=0.815
total time=   2.0s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=1.0;, score=0.815
total time=   2.0s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=4, subsample=1.0;, score=0.816
total time=   2.0s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.6;, score=0.819
total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.6;, score=0.816
total time=   2.6s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.6;, score=0.820
total time=   2.5s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.6;, score=0.822
total time=   2.6s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.6;, score=0.824
total time=   2.5s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.8;, score=0.819
total time=   2.6s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.8;, score=0.816
total time=   2.6s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.8;, score=0.822
total time=   2.6s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.8;, score=0.819
total time=   2.6s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=0.8;, score=0.824
total time=   2.6s
[CV 1/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=1.0;, score=0.811
total time=   2.5s
[CV 2/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=1.0;, score=0.814
total time=   2.5s
[CV 3/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=1.0;, score=0.816
total time=   2.5s
[CV 4/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=1.0;, score=0.816
total time=   2.6s
[CV 5/5] END colsample_bytree=0.6, gamma=5, max_depth=5, subsample=1.0;, score=0.819
total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
23 total time=   1.9s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
13 total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
21 total time=   2.0s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
23 total time=   2.0s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
```

```
23 total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
24 total time=   2.0s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
15 total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
21 total time=   2.0s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
22 total time=   2.0s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
24 total time=   2.0s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
21 total time=   1.9s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
15 total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
20 total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
22 total time=   1.9s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
24 total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
24 total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
14 total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
22 total time=   2.6s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
22 total time=   2.5s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
23 total time=   2.6s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
24 total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
14 total time=   2.6s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
20 total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
22 total time=   2.6s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
24 total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
22 total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
15 total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
19 total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
22 total time=   2.5s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
24 total time=   2.6s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
23 total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
14 total time=   3.1s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
```

```
22 total time=   3.3s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
22 total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
24 total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
23 total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
16 total time=   3.2s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
20 total time=   3.2s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
19 total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
26 total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
22 total time=   3.1s
[CV 2/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
16 total time=   3.1s
[CV 3/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
19 total time=   3.1s
[CV 4/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
21 total time=   3.1s
[CV 5/5] END colsample_bytree=0.8, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
24 total time=   3.1s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.6;, score=0.823
total time=   2.1s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.6;, score=0.813
total time=   2.0s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.6;, score=0.821
total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.6;, score=0.823
total time=   2.0s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.6;, score=0.823
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.8;, score=0.824
total time=   2.0s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.8;, score=0.815
total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.8;, score=0.821
total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.8;, score=0.822
total time=   1.9s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=0.8;, score=0.824
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=1.0;, score=0.821
total time=   1.9s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=1.0;, score=0.815
total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=1.0;, score=0.820
total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=1.0;, score=0.822
total time=   1.9s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=3, subsample=1.0;, score=0.824
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.6;, score=0.823
```

```
total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.6;, score=0.814
total time=   2.6s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.6;, score=0.822
total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.6;, score=0.822
total time=   2.5s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.6;, score=0.823
total time=   2.6s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.8;, score=0.824
total time=   2.6s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.8;, score=0.814
total time=   2.6s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.8;, score=0.820
total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.8;, score=0.822
total time=   2.6s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=0.8;, score=0.824
total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=1.0;, score=0.822
total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=1.0;, score=0.815
total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=1.0;, score=0.819
total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=1.0;, score=0.822
total time=   2.5s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=4, subsample=1.0;, score=0.824
total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.6;, score=0.823
total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.6;, score=0.814
total time=   3.2s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.6;, score=0.822
total time=   3.2s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.6;, score=0.823
total time=   3.1s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.6;, score=0.824
total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.8;, score=0.823
total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.8;, score=0.815
total time=   3.2s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.8;, score=0.820
total time=   3.2s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.8;, score=0.820
total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=0.8;, score=0.826
total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=1.0;, score=0.822
total time=   3.1s
[CV 2/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=1.0;, score=0.816
total time=   3.1s
[CV 3/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=1.0;, score=0.819
total time=   3.1s
[CV 4/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=1.0;, score=0.821
```

```
total time=   3.1s
[CV 5/5] END colsample_bytree=0.8, gamma=1, max_depth=5, subsample=1.0;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.6;, score=0.823
total time=   2.0s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.6;, score=0.812
total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.6;, score=0.821
total time=   2.0s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.6;, score=0.823
total time=   2.1s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.6;, score=0.823
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.8;, score=0.824
total time=   2.0s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.8;, score=0.815
total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.8;, score=0.821
total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.8;, score=0.822
total time=   2.0s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=0.8;, score=0.824
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=1.0;, score=0.821
total time=   1.9s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=1.0;, score=0.815
total time=   1.9s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=1.0;, score=0.820
total time=   1.9s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=1.0;, score=0.822
total time=   1.9s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=3, subsample=1.0;, score=0.824
total time=   1.9s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.6;, score=0.822
total time=   2.6s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.6;, score=0.814
total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.6;, score=0.821
total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.6;, score=0.822
total time=   2.6s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.6;, score=0.824
total time=   2.6s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.8;, score=0.824
total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.8;, score=0.814
total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.8;, score=0.820
total time=   2.6s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.8;, score=0.822
total time=   2.6s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=0.8;, score=0.824
total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=1.0;, score=0.822
total time=   2.5s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=1.0;, score=0.816
```

```
total time=   2.5s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=1.0;, score=0.819
total time=   2.5s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=1.0;, score=0.822
total time=   2.5s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=4, subsample=1.0;, score=0.824
total time=   2.5s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.6;, score=0.823
total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.6;, score=0.814
total time=   3.1s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.6;, score=0.820
total time=   3.2s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.6;, score=0.822
total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.6;, score=0.825
total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.8;, score=0.823
total time=   3.2s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.8;, score=0.816
total time=   3.2s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.8;, score=0.820
total time=   3.2s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.8;, score=0.821
total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=0.8;, score=0.825
total time=   3.2s
[CV 1/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=1.0;, score=0.822
total time=   3.1s
[CV 2/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=1.0;, score=0.816
total time=   3.1s
[CV 3/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=1.0;, score=0.819
total time=   3.1s
[CV 4/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=1.0;, score=0.821
total time=   3.2s
[CV 5/5] END colsample_bytree=0.8, gamma=5, max_depth=5, subsample=1.0;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
24 total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
13 total time=   2.4s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
21 total time=   2.4s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
22 total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.6;, score=0.8
22 total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
24 total time=   2.5s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
13 total time=   2.4s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
20 total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
22 total time=   2.4s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=0.8;, score=0.8
```

```
23 total time=   2.4s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
24 total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
14 total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
20 total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
21 total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=3, subsample=1.0;, score=0.8
24 total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
24 total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
13 total time=   3.1s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
19 total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
22 total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.6;, score=0.8
24 total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
24 total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
14 total time=   3.1s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
20 total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
22 total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=0.8;, score=0.8
24 total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
23 total time=   3.0s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
14 total time=   3.0s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
18 total time=   3.0s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
22 total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=4, subsample=1.0;, score=0.8
23 total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
24 total time=   3.9s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
13 total time=   3.9s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
20 total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
22 total time=   3.9s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.6;, score=0.8
24 total time=   3.8s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
22 total time=   3.9s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
14 total time=   3.9s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
```

```
20 total time=   3.9s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
20 total time=   3.9s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=0.8;, score=0.8
24 total time=   3.9s
[CV 1/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
22 total time=   3.9s
[CV 2/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
15 total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
18 total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
22 total time=   4.0s
[CV 5/5] END colsample_bytree=1.0, gamma=0.5, max_depth=5, subsample=1.0;, score=0.8
23 total time=   3.8s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.6;, score=0.824
total time=   2.4s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.6;, score=0.813
total time=   2.4s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.6;, score=0.821
total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.6;, score=0.822
total time=   2.4s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.6;, score=0.822
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.8;, score=0.824
total time=   2.4s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.8;, score=0.813
total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.8;, score=0.820
total time=   2.4s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.8;, score=0.822
total time=   2.4s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=0.8;, score=0.823
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=1.0;, score=0.824
total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=1.0;, score=0.814
total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=1.0;, score=0.820
total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=1.0;, score=0.821
total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=3, subsample=1.0;, score=0.824
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.6;, score=0.824
total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.6;, score=0.813
total time=   3.1s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.6;, score=0.819
total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.6;, score=0.822
total time=   3.0s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.6;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.8;, score=0.824
```

```
total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.8;, score=0.814
total time=   3.1s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.8;, score=0.820
total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.8;, score=0.822
total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=0.8;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=1.0;, score=0.823
total time=   3.0s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=1.0;, score=0.814
total time=   3.0s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=1.0;, score=0.818
total time=   3.0s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=1.0;, score=0.822
total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=4, subsample=1.0;, score=0.823
total time=   3.0s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.6;, score=0.824
total time=   3.8s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.6;, score=0.813
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.6;, score=0.820
total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.6;, score=0.822
total time=   3.8s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.6;, score=0.824
total time=   3.8s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.8;, score=0.823
total time=   3.8s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.8;, score=0.814
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.8;, score=0.820
total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.8;, score=0.820
total time=   3.8s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=0.8;, score=0.824
total time=   4.0s
[CV 1/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=1.0;, score=0.822
total time=   3.9s
[CV 2/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=1.0;, score=0.815
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=1.0;, score=0.818
total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=1.0;, score=0.822
total time=   3.8s
[CV 5/5] END colsample_bytree=1.0, gamma=1, max_depth=5, subsample=1.0;, score=0.823
total time=   3.8s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.6;, score=0.824
total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.6;, score=0.813
total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.6;, score=0.821
total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.6;, score=0.822
```

```
total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.6;, score=0.822
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.8;, score=0.824
total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.8;, score=0.813
total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.8;, score=0.820
total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.8;, score=0.822
total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=0.8;, score=0.823
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=1.0;, score=0.824
total time=   2.3s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=1.0;, score=0.814
total time=   2.3s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=1.0;, score=0.820
total time=   2.3s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=1.0;, score=0.821
total time=   2.3s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=3, subsample=1.0;, score=0.824
total time=   2.3s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.6;, score=0.824
total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.6;, score=0.813
total time=   3.0s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.6;, score=0.819
total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.6;, score=0.822
total time=   3.0s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.6;, score=0.824
total time=   3.0s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.8;, score=0.824
total time=   3.1s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.8;, score=0.813
total time=   3.1s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.8;, score=0.820
total time=   3.1s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.8;, score=0.822
total time=   3.1s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=0.8;, score=0.824
total time=   3.1s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=1.0;, score=0.823
total time=   3.0s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=1.0;, score=0.813
total time=   3.0s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=1.0;, score=0.818
total time=   3.0s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=1.0;, score=0.821
total time=   3.0s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=4, subsample=1.0;, score=0.823
total time=   3.0s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.6;, score=0.824
total time=   3.8s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.6;, score=0.812
```

```
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.6;, score=0.819
total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.6;, score=0.822
total time=   3.8s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.6;, score=0.824
total time=   4.0s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.8;, score=0.823
total time=   3.9s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.8;, score=0.814
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.8;, score=0.819
total time=   3.9s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.8;, score=0.821
total time=   3.9s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=0.8;, score=0.824
total time=   3.9s
[CV 1/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=1.0;, score=0.822
total time=   3.8s
[CV 2/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=1.0;, score=0.815
total time=   3.8s
[CV 3/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=1.0;, score=0.818
total time=   3.8s
[CV 4/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=1.0;, score=0.822
total time=   3.8s
[CV 5/5] END colsample_bytree=1.0, gamma=5, max_depth=5, subsample=1.0;, score=0.823
total time=   3.8s
```

Out[121…

```
▸ GridSearchCV
▸ estimator: XGBClassifier
    ▸ XGBClassifier
```

In [124…

```
#47 Apply the predict method using the X_test to provide us with the best model out
y_predict_optim = grid.predict(X_test)
```

In [125…

```
#48 Print the optimal prediction
y_predict_optim
```

Out[125…

```
array([1, 0, 0, ..., 0, 0, 0])
```

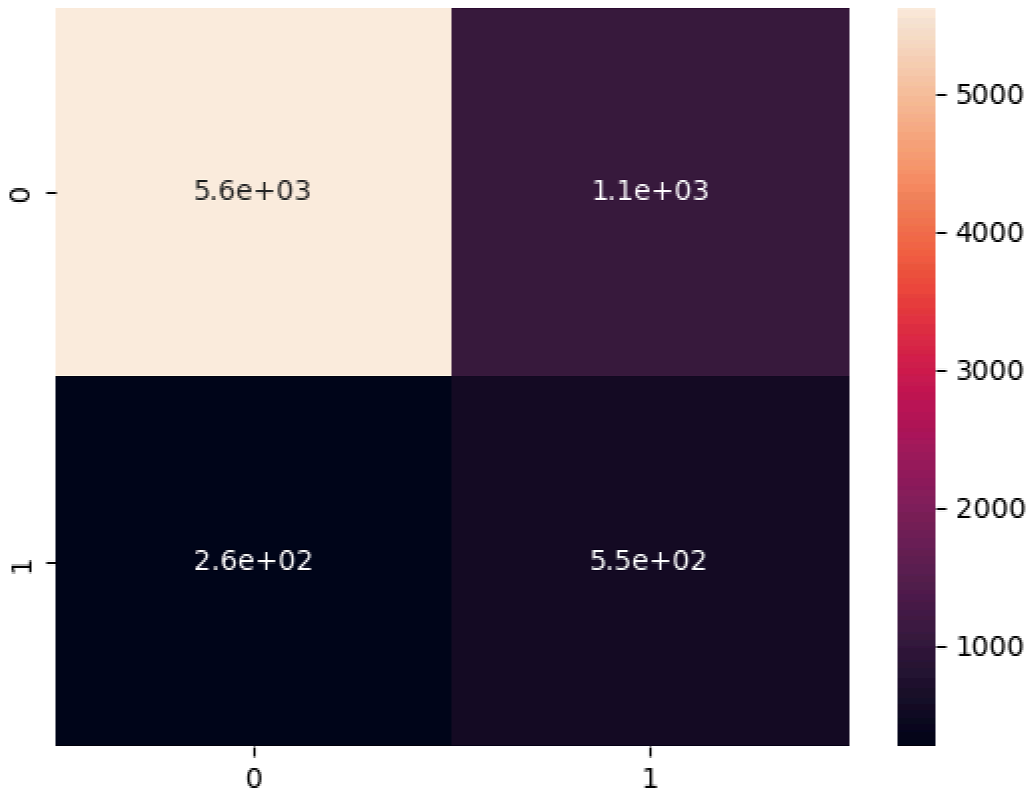
In [126…

```
#49 Checking the accuracy of the model
# Analyzis: Increased performance and accuracy to 0.81
cm = confusion_matrix(y_predict_optim, y_test)
sns.heatmap(cm, annot=True)
print(classification_report(y_test, y_predict_optim))
```

```
              precision    recall  f1-score   support

           0       0.84      0.96      0.89      5887
           1       0.68      0.34      0.45      1613

    accuracy                           0.82      7500
   macro avg       0.76      0.65      0.67      7500
weighted avg       0.81      0.82      0.80      7500
```



In [127…

```
#50 Check the shape of x_train
X_train.shape
```

Out[127… (22500, 46)

In [128…

```
#51 Check the shape of y_train
y_train.shape
```

Out[128… (22500,)

In [134…

```
#52 Convert the data into a format that the XGBoost can process
train_data = pd.DataFrame({'Target':y_train})
#The for loop will concatinate the data
for i in range(X_train.shape[1]):
    train_data[i] = X_train[:,i]
```

In [130…

```
#53 Print the concatinated data
train_data.head()
```

Out[130…

	Target	0	1	2	3	4	5	6	7	8	...	36	37	38
15176	0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.129556	0.232241	0.155373
19168	0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.086409	0.164744	0.080295
29830	1	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	1.0	...	0.086345	0.160138	0.080648
2805	0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	1.0	...	0.086345	0.160138	0.081555
11117	0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	1.0	...	0.086345	0.160138	0.080648

5 rows × 47 columns

In [137…

```
#54 Concatinate the testing data too
val_data = pd.DataFrame({'Target':y_test})
#The for loop will concatinate the data
for i in range(X_test.shape[1]):
    val_data[i] = X_test[:,i]
```

In [138…

```
#53 Print the concatinated test data
val_data.head()
```

Out[138…

	Target	0	1	2	3	4	5	6	7	8	...	36	37	38
18917	1	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.090524	0.168452	0.090676
639	0	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	...	0.118832	0.212784	0.133867
9431	0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.094504	0.173181	0.093986
2523	0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.091636	0.168391	0.089148
15637	0	1.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	...	0.086894	0.160138	0.080648

5 rows × 47 columns

In [145…

```
#54 Check the shape of the test data
val_data.shape
```

Out[145…

```
(7500, 47)
```

In [202…

```
#55 Save the train and validation data as csv files
train_data.to_csv('train.csv', header = False, index = False)
val_data.to_csv('validation.csv', header = False, index = False)
```

In [203…

```
#56 Contain all the data in an Amazon S3 and EC2 instances
import sagemaker
#Boto3 is the Software Development Kit for Python that helps developer write softwa
import boto3
#Create a sagemaker session
sagemaker_session = sagemaker.Session()
```

```
#Specify the bucket, prefix (folder withing the bucket), and key
bucket = 'creditcarddefaultai'
prefix = 'XGBoost-classifier'
key = 'XGBoost-classifier'
#Speficy the role to allow hosting access to the data
role = sagemaker.get_execution_role()
```

In [204…

```
#57 Print the role
print(role)
```

```
arn:aws:iam::339712900161:role/service-role/AmazonSageMaker-ExecutionRole-20240614T2
20193
```

In [205…

```
#58 Upload the training data to S3
import os
with open('train.csv','rb') as f:
          boto3.Session().resource('s3').Bucket(bucket).Object(os.path.join(prefix,
s3_train_data = 's3://{}/{}/train/{}'.format(bucket, prefix, key)
print('uploaded training data location: {}'.format(s3_train_data))
```

```
uploaded training data location: s3://creditcarddefaultai/XGBoost-classifier/train/X
GBoost-classifier
```

In [206…

```
#58 Upload the validation data to S3
with open('validation.csv','rb') as f:
          boto3.Session().resource('s3').Bucket(bucket).Object(os.path.join(prefix,
s3_validation_data = 's3://{}/{}/train/{}'.format(bucket, prefix, key)
print('uploaded training data location: {}'.format(s3_validation_data))
```

```
uploaded training data location: s3://creditcarddefaultai/XGBoost-classifier/train/X
GBoost-classifier
```

In [159…

```
#59 Store the validation data in the S3 bucket
output_location = 's3://{}/{}/output'.format(bucket, prefix)
print('training artifacts will be uploaded to: {}'.format(output_location))
```

```
training artifacts will be uploaded to: s3://creditcarddefaultai/XGBoost-classifier/
output
```

In [208…

```
#60 Get the training data from the S3 container and feed it to XGBoost
from sagemaker import image_uris
container = image_uris.retrieve('xgboost', boto3.Session().region_name, version='0.
```

```
INFO:sagemaker.image_uris:Defaulting to only available Python version: py3
INFO:sagemaker.image_uris:Defaulting to only supported image scope: cpu.
```

In [209…

```
#61 Specify the type of instance we would like to use for training
Xgboost_regressor1 = sagemaker.estimator.Estimator(
    image_uri=container,
    role=sagemaker.get_execution_role(),
    instance_count=1,
    instance_type='ml.m5.2xlarge',
    output_path=output_location,
    sagemaker_session=sagemaker_session
)

#We can tune the hyper-parameters to improve the performance of the model
```

```python
Xgboost_regressor1.set_hyperparameters(
    max_depth=10,
    objective='reg:squarederror',  # Updated from 'reg:linear'
    colsample_bytree=0.3,
    alpha=10,
    eta=0.1,
    num_round=100
)
```

In [210…

```python
#62 Feed the model with the training and validating data
from sagemaker.inputs import TrainingInput
train_input = TrainingInput(s3_data=s3_train_data, content_type='text/csv', s3_data
valid_input = TrainingInput(s3_data=s3_validation_data, content_type='text/csv', s3
data_channels = {'train': train_input,'validation': valid_input}
Xgboost_regressor1.fit(data_channels)
```

```
INFO:sagemaker:Creating training-job with name: sagemaker-xgboost-2024-06-23-20-57-0
6-593
```

```
2024-06-23 20:57:06 Starting - Starting the training job...
2024-06-23 20:57:24 Starting - Preparing the instances for training...
2024-06-23 20:58:02 Downloading - Downloading the training image...
2024-06-23 20:58:33 Training - Training image download completed. Training in progre
ss....
2024-06-23 20:59:03 Uploading - Uploading generated training modelINFO:sagemaker-con
tainers:Imported framework sagemaker_xgboost_container.training
INFO:sagemaker-containers:Failed to parse hyperparameter objective value reg:squared
error to Json.
Returning the value itself
INFO:sagemaker-containers:No GPUs detected (normal if no gpus installed)
INFO:sagemaker_xgboost_container.training:Running XGBoost Sagemaker in algorithm mod
e
INFO:root:Determined delimiter of CSV input is ','
INFO:root:Determined delimiter of CSV input is ','
INFO:root:Determined delimiter of CSV input is ','
[20:58:56] 22500x46 matrix with 1035000 entries loaded from /opt/ml/input/data/trai
n?format=csv&label_column=0&delimiter=,
INFO:root:Determined delimiter of CSV input is ','
[20:58:56] 22500x46 matrix with 1035000 entries loaded from /opt/ml/input/data/valid
ation?format=csv&label_column=0&delimiter=,
INFO:root:Single node training.
[2024-06-23 20:58:56.845 ip-10-0-219-198.ec2.internal:7 INFO json_config.py:90] Crea
ting hook from json_config at /opt/ml/input/config/debughookconfig.json.
[2024-06-23 20:58:56.846 ip-10-0-219-198.ec2.internal:7 INFO hook.py:151] tensorboar
d_dir has not been set for the hook. SMDebug will not be exporting tensorboard summa
ries.
[2024-06-23 20:58:56.846 ip-10-0-219-198.ec2.internal:7 INFO hook.py:196] Saving to
/opt/ml/output/tensors
INFO:root:Debug hook created from config
INFO:root:Train matrix has 22500 rows
INFO:root:Validation matrix has 22500 rows
[0]#011train-rmse:0.481479#011validation-rmse:0.481479
[2024-06-23 20:58:56.934 ip-10-0-219-198.ec2.internal:7 INFO hook.py:325] Monitoring
the collections: metrics
[1]#011train-rmse:0.464812#011validation-rmse:0.464812
[2]#011train-rmse:0.451343#011validation-rmse:0.451343
[3]#011train-rmse:0.440151#011validation-rmse:0.440151
[4]#011train-rmse:0.42749#011validation-rmse:0.42749
[5]#011train-rmse:0.419265#011validation-rmse:0.419265
[6]#011train-rmse:0.412041#011validation-rmse:0.412041
[7]#011train-rmse:0.403913#011validation-rmse:0.403913
[8]#011train-rmse:0.398565#011validation-rmse:0.398565
[9]#011train-rmse:0.394172#011validation-rmse:0.394172
[10]#011train-rmse:0.390614#011validation-rmse:0.390614
[11]#011train-rmse:0.385885#011validation-rmse:0.385885
[12]#011train-rmse:0.381919#011validation-rmse:0.381919
[13]#011train-rmse:0.379746#011validation-rmse:0.379746
[14]#011train-rmse:0.376631#011validation-rmse:0.376631
[15]#011train-rmse:0.374748#011validation-rmse:0.374748
[16]#011train-rmse:0.37234#011validation-rmse:0.37234
[17]#011train-rmse:0.370783#011validation-rmse:0.370783
[18]#011train-rmse:0.368938#011validation-rmse:0.368938
[19]#011train-rmse:0.367572#011validation-rmse:0.367572
[20]#011train-rmse:0.366667#011validation-rmse:0.366667
[21]#011train-rmse:0.36594#011validation-rmse:0.36594
```

```
[22]#011train-rmse:0.364898#011validation-rmse:0.364898
[23]#011train-rmse:0.363955#011validation-rmse:0.363955
[24]#011train-rmse:0.363528#011validation-rmse:0.363528
[25]#011train-rmse:0.362741#011validation-rmse:0.362741
[26]#011train-rmse:0.362317#011validation-rmse:0.362317
[27]#011train-rmse:0.361985#011validation-rmse:0.361985
[28]#011train-rmse:0.361557#011validation-rmse:0.361557
[29]#011train-rmse:0.361091#011validation-rmse:0.361091
[30]#011train-rmse:0.360638#011validation-rmse:0.360638
[31]#011train-rmse:0.360171#011validation-rmse:0.360171
[32]#011train-rmse:0.359479#011validation-rmse:0.359479
[33]#011train-rmse:0.358678#011validation-rmse:0.358678
[34]#011train-rmse:0.358346#011validation-rmse:0.358346
[35]#011train-rmse:0.358116#011validation-rmse:0.358116
[36]#011train-rmse:0.357876#011validation-rmse:0.357876
[37]#011train-rmse:0.357523#011validation-rmse:0.357523
[38]#011train-rmse:0.356744#011validation-rmse:0.356744
[39]#011train-rmse:0.356319#011validation-rmse:0.356319
[40]#011train-rmse:0.356157#011validation-rmse:0.356157
[41]#011train-rmse:0.355693#011validation-rmse:0.355693
[42]#011train-rmse:0.355424#011validation-rmse:0.355424
[43]#011train-rmse:0.355133#011validation-rmse:0.355133
[44]#011train-rmse:0.354927#011validation-rmse:0.354927
[45]#011train-rmse:0.354571#011validation-rmse:0.354572
[46]#011train-rmse:0.35437#011validation-rmse:0.35437
[47]#011train-rmse:0.353711#011validation-rmse:0.353711
[48]#011train-rmse:0.353421#011validation-rmse:0.353421
[49]#011train-rmse:0.353221#011validation-rmse:0.353221
[50]#011train-rmse:0.352902#011validation-rmse:0.352902
[51]#011train-rmse:0.352554#011validation-rmse:0.352554
[52]#011train-rmse:0.352179#011validation-rmse:0.352179
[53]#011train-rmse:0.352011#011validation-rmse:0.352011
[54]#011train-rmse:0.351749#011validation-rmse:0.351749
[55]#011train-rmse:0.351521#011validation-rmse:0.351521
[56]#011train-rmse:0.35114#011validation-rmse:0.35114
[57]#011train-rmse:0.350774#011validation-rmse:0.350774
[58]#011train-rmse:0.350558#011validation-rmse:0.350558
[59]#011train-rmse:0.350352#011validation-rmse:0.350352
[60]#011train-rmse:0.35009#011validation-rmse:0.35009
[61]#011train-rmse:0.349937#011validation-rmse:0.349937
[62]#011train-rmse:0.349546#011validation-rmse:0.349546
[63]#011train-rmse:0.349097#011validation-rmse:0.349097
[64]#011train-rmse:0.348762#011validation-rmse:0.348762
[65]#011train-rmse:0.348519#011validation-rmse:0.348519
[66]#011train-rmse:0.348273#011validation-rmse:0.348273
[67]#011train-rmse:0.348031#011validation-rmse:0.348031
[68]#011train-rmse:0.347731#011validation-rmse:0.347731
[69]#011train-rmse:0.347396#011validation-rmse:0.347396
[70]#011train-rmse:0.34701#011validation-rmse:0.34701
[71]#011train-rmse:0.346893#011validation-rmse:0.346893
[72]#011train-rmse:0.346603#011validation-rmse:0.346603
[73]#011train-rmse:0.346351#011validation-rmse:0.346351
[74]#011train-rmse:0.346143#011validation-rmse:0.346143
[75]#011train-rmse:0.345803#011validation-rmse:0.345803
[76]#011train-rmse:0.345535#011validation-rmse:0.345535
[77]#011train-rmse:0.345325#011validation-rmse:0.345325
```

```
[78]#011train-rmse:0.345047#011validation-rmse:0.345047
[79]#011train-rmse:0.344908#011validation-rmse:0.344908
[80]#011train-rmse:0.344701#011validation-rmse:0.344701
[81]#011train-rmse:0.344524#011validation-rmse:0.344524
[82]#011train-rmse:0.344269#011validation-rmse:0.344269
[83]#011train-rmse:0.344088#011validation-rmse:0.344088
[84]#011train-rmse:0.343771#011validation-rmse:0.343771
[85]#011train-rmse:0.343619#011validation-rmse:0.343619
[86]#011train-rmse:0.343388#011validation-rmse:0.343388
[87]#011train-rmse:0.343129#011validation-rmse:0.343129
[88]#011train-rmse:0.342955#011validation-rmse:0.342955
[89]#011train-rmse:0.342755#011validation-rmse:0.342755
[90]#011train-rmse:0.342582#011validation-rmse:0.342582
[91]#011train-rmse:0.342298#011validation-rmse:0.342298
[92]#011train-rmse:0.342102#011validation-rmse:0.342102
[93]#011train-rmse:0.341762#011validation-rmse:0.341762
[94]#011train-rmse:0.341431#011validation-rmse:0.341431
[95]#011train-rmse:0.341225#011validation-rmse:0.341225
[96]#011train-rmse:0.340897#011validation-rmse:0.340897
[97]#011train-rmse:0.340696#011validation-rmse:0.340696
[98]#011train-rmse:0.340389#011validation-rmse:0.340389
[99]#011train-rmse:0.34018#011validation-rmse:0.34018

2024-06-23 20:59:16 Completed - Training job completed
Training seconds: 89
Billable seconds: 89
```

In [222…

```python
#63 Deploy the model
Xgboost_classifier = Xgboost_regressor1.deploy(initial_instance_count = 1, instance
```

```
INFO:sagemaker:Creating model with name: sagemaker-xgboost-2024-06-23-21-09-01-060
INFO:sagemaker:Creating endpoint-config with name sagemaker-xgboost-2024-06-23-21-09
-01-060
INFO:sagemaker:Creating endpoint with name sagemaker-xgboost-2024-06-23-21-09-01-060
-----!
```

In [242…

```python
#64 Ensure the data receive and exported is in text/csv format
from sagemaker.serializers import CSVSerializer
from sagemaker.deserializers import StringDeserializer
Xgboost_classifier.content_type = 'text/csv'
Xgboost_classifier.serializer = CSVSerializer()
Xgboost_classifier.deserializer = StringDeserializer()
```

In [256…

```python
#65 Make the predictionfrom sagemaker.predictor import csv_serializer
import numpy as np
XGB_prediction = Xgboost_classifier.predict(np.array(X_test))
raw_response = XGB_prediction
```

In [258…

```python
#66 Run the prediction
probabilities = list(map(float, raw_response.split(',')))
binary_predictions = [1 if prob > 0.5 else 0 for prob in probabilities]
print(binary_predictions)
```

```
[1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0,
0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
```

```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
```

```
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0,
0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
```

```
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
```

In [259…

```python
#67 Convert byte to arrays
def bytes_2_array(x):
    # makes entire prediction as string and splits based on ','
    l = str(x).split(',')
    # Since the first element contains unwanted characters like (b,',') we remove t
    l[0] = l[0][2:]
    #same-thing as above remove the unwanted last character (')
    l[-1] = l[-1][:-1]
    # iterating through the list of strings and converting them into float type
    for i in range(len(l)):
        l[i] = float(l[i])
```

```
    # converting the list into array
    l = np.array(l).astype('float32')
    # reshape one-dimensional array to two-dimensional array
    return l.reshape(-1,1)
```

In [268…

```
#68 Call the fuction
binary_predictions = [1 if prob > 0.5 else 0 for prob in probabilities]
predicted_values = np.array(binary_predictions).astype('float32')
```

In [267…

```
#69 Print the predicted value
predicted_values
```

Out[267…

```
array([1., 0., 0., ..., 0., 0., 0.], dtype=float32)
```

In [270…

```
#70 Convert y_test into an array
y_test = np.array(y_test)
y_test = y_test.reshape(-1,1)
```

In [271…

```
y_test
```

Out[271…

```
array([[1],
       [0],
       [0],
       ...,
       [1],
       [0],
       [0]])
```

In [272…

```
#71 Plot the metrics
from sklearn.metrics import precision_score, recall_score, accuracy_score

print("Precision = {}". format(precision_score(y_test, predicted_values, average='m
print("Recall = {}".format(recall_score(y_test, predicted_values, average='macro'))
print("Accuracy = {}".format(accuracy_score(y_test, predicted_values)))
```

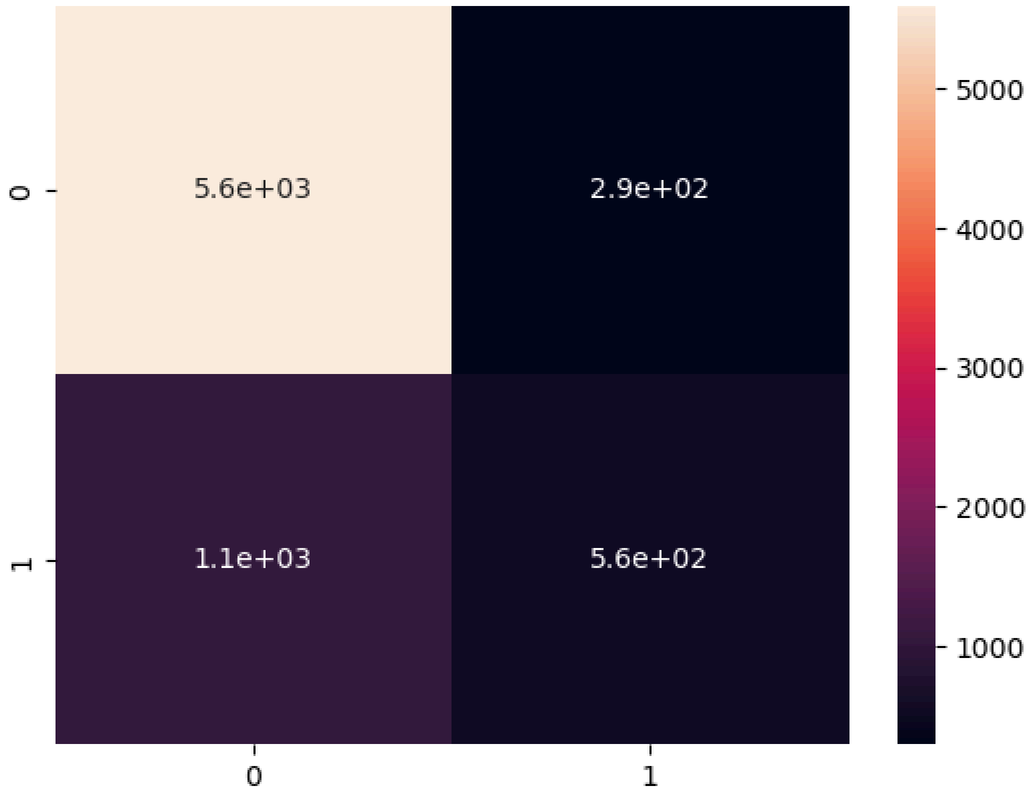
```
Precision = 0.7494417467500551
Recall = 0.648789176946988
Accuracy = 0.8206666666666667
```

In [274…

```
#72 Plot the confusion matrix
#Analysis: 5.6+03 sample has been properly classify and 5.6e+02 sample and the rest
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, predicted_values)
plt.figure()
sns.heatmap(cm, annot=True)
```

Out[274…

```
<Axes: >
```



In [275…

```
#73 Delete the end-point (to prevent being overcharge)
Xgboost_classifier.delete_endpoint()
```

```
INFO:sagemaker:Deleting endpoint configuration with name: sagemaker-xgboost-2024-06-23-21-09-01-060
INFO:sagemaker:Deleting endpoint with name: sagemaker-xgboost-2024-06-23-21-09-01-060
```