Exploratory Data Analysis(EDA) of Financial Data

In [1]:
```
import numpy as np
import matplotlib .pyplot as plt
import pandas as pd
import seaborn as sns
color = sns.color_palette()
import sklearn.metrics as metrics
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
Default = pd.read_csv('Default.csv')
Default.head()
```

Out[2]:

	Unnamed: 0	default	student	balance	income
0	1	No	No	729.526495	44361.625074
1	2	No	Yes	817.180407	12106.134700
2	3	No	No	1073.549164	31767.138947
3	4	No	No	529.250605	35704.493935
4	5	No	No	785.655883	38463.495879

In [3]:
```
Default.shape
```

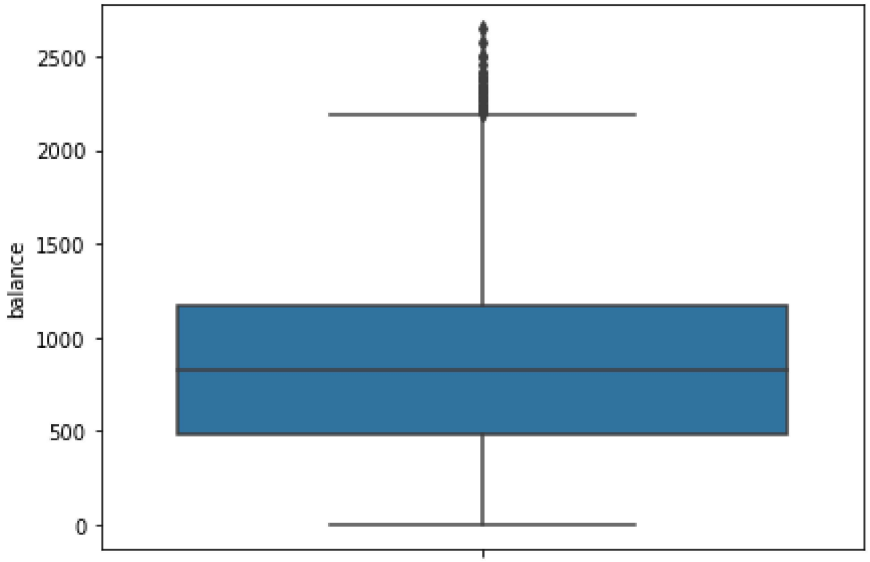
Out[3]:
```
(10000, 5)
```

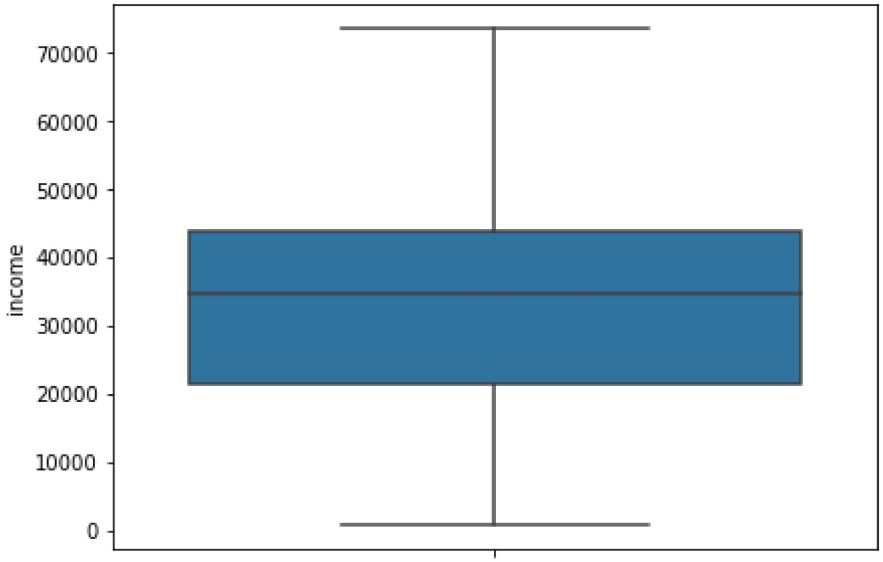
In [4]:
```
Default.describe()
```

Out[4]:

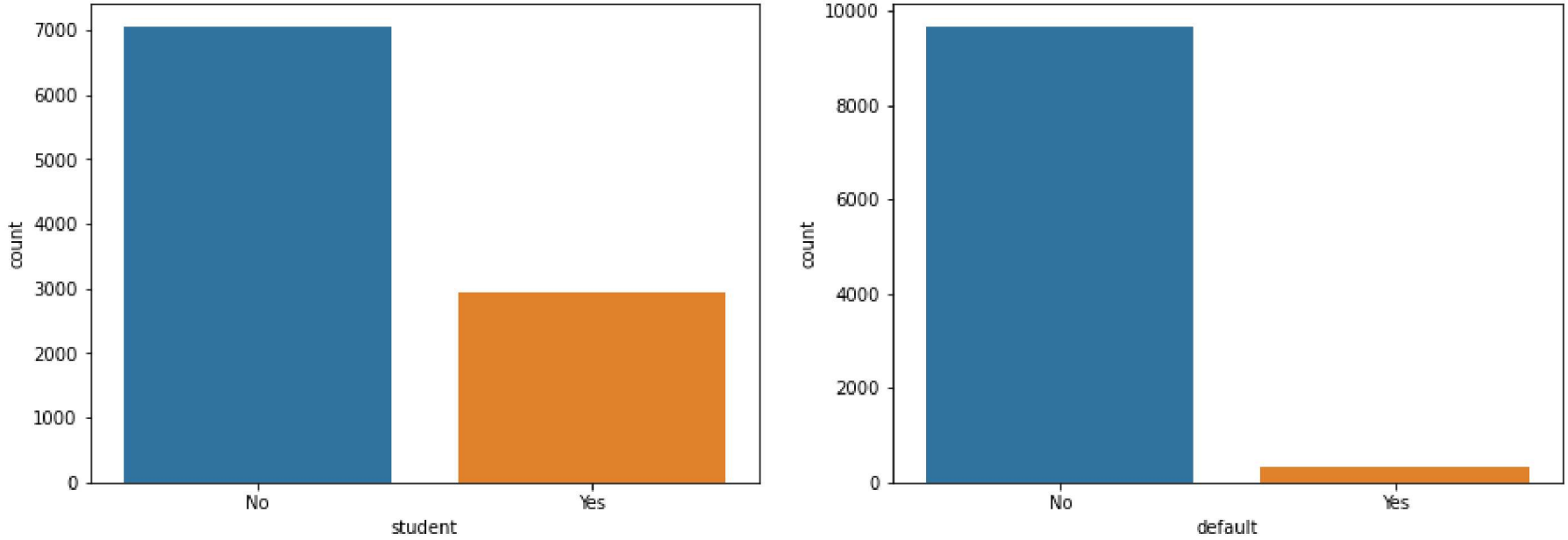
	Unnamed: 0	balance	income
count	10000.00000	10000.000000	10000.000000
mean	5000.50000	835.374886	33516.981876
std	2886.89568	483.714985	13336.639563
min	1.00000	0.000000	771.967729
25%	2500.75000	481.731105	21340.462903
50%	5000.50000	823.636973	34552.644802
75%	7500.25000	1166.308386	43807.729272
max	10000.00000	2654.322576	73554.233495

In [5]:
```
plt.figure(figsize = (15,5))
plt.subplot(1,2,1)
sns.boxplot(y=Default['balance'])

plt.subplot(1,2,2)
sns.boxplot(y=Default['income'])
plt.show()
```





In [6]:
```
plt.figure(figsize = (15,5))
plt.subplot(1,2,1)
sns.countplot(Default['student'])

plt.subplot(1,2,2)
sns.countplot(Default['default'])
plt.show()
```



In [7]:
```
Default['student'].value_counts()
```

Out[7]:
```
No     7056
Yes    2944
Name: student, dtype: int64
```

In [8]:
```
Default['default'].value_counts()
```

Out[8]:
```
No     9667
Yes     333
Name: default, dtype: int64
```

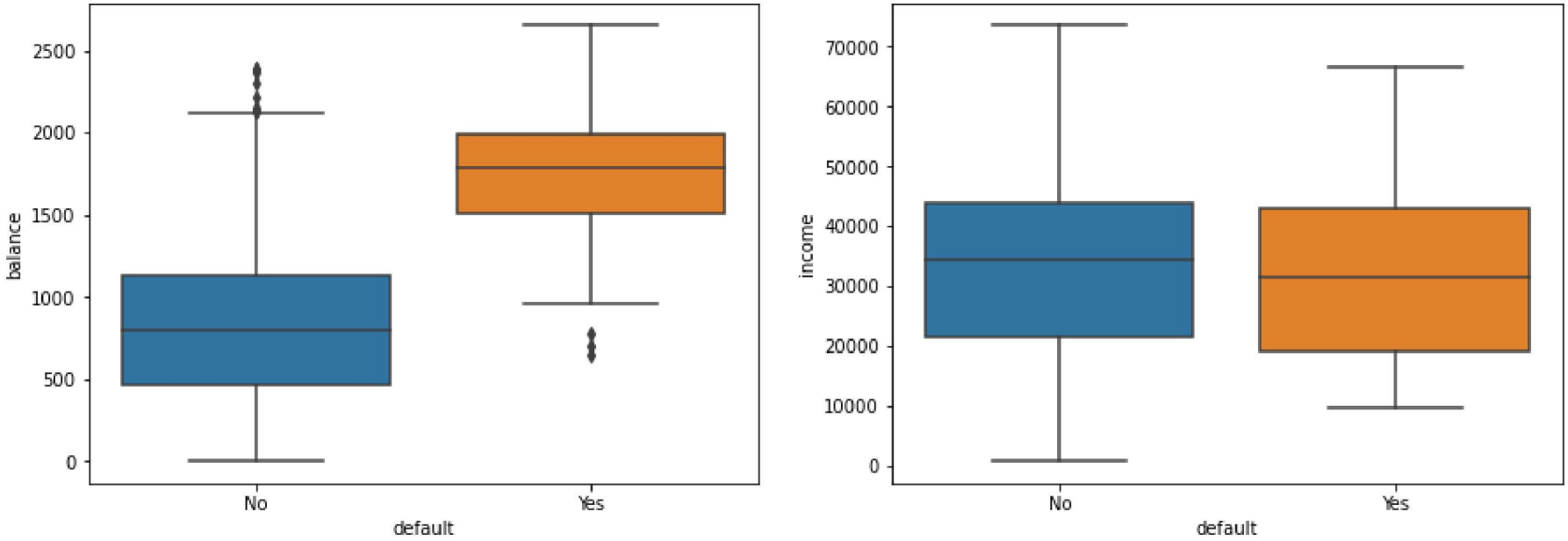
In [9]:
```
Default['student'].value_counts(normalize=True)
```

Out[9]:
```
No     0.7056
Yes    0.2944
Name: student, dtype: float64
```

In [10]:
```
Default['default'].value_counts(normalize=True)
```

Out[10]:
```
No     0.9667
Yes    0.0333
Name: default, dtype: float64
```

In [11]:
```
plt.figure(figsize = (15,5))
plt.subplot(1,2,1)
sns.boxplot(Default['default'],Default['balance'])

plt.subplot(1,2,2)
sns.boxplot(Default['default'],Default['income'])
plt.show()
```



In [12]:
```
pd.crosstab(Default['student'],Default['default'],normalize = 'index').round(2)
```

Out[12]:

student	default	No	Yes
No		0.97	0.03
Yes		0.96	0.04

In [13]:
```
sns.heatmap(Default[['balance','income']].corr(),annot = True)
plt.show()
```

In [14]:
```
Default.isnull().sum()
```

Out[14]:
```
Unnamed: 0    0
default       0
student       0
balance       0
income        0
dtype: int64
```

In [15]:
```
Q1, Q3 = Default['balance'].quantile([.25,.75])
IQR = Q3-Q1
LL = Q1 - 1.5*(IQR)
UL = Q3 + 1.5*(IQR)
```

In [16]:
```
UL
```

Out[16]:
```
2193.174308607817
```

In [17]:
```
df = Default[Default['balance'] > UL]
df
```

Out[17]:

	Unnamed: 0	default	student	balance	income
173	174	Yes	Yes	2205.799521	14271.492253
1136	1137	Yes	No	2499.016750	51504.293960
1160	1161	Yes	Yes	2502.684931	14947.519752
1359	1360	Yes	No	2220.966201	40725.096207
1502	1503	Yes	Yes	2332.878254	11770.234124
1609	1610	Yes	Yes	2269.946966	18021.105948
2096	2097	Yes	Yes	2261.848162	20030.165119
2140	2141	No	Yes	2308.893236	19110.266412
2929	2930	Yes	Yes	2387.314867	28296.914718
3162	3163	Yes	Yes	2415.316994	17429.503375
3189	3190	Yes	No	2228.472283	27438.348988
3702	3703	No	Yes	2370.463612	24251.958722
3855	3856	Yes	Yes	2321.882221	21331.314781
3913	3914	Yes	Yes	2334.123559	19335.889287
3976	3977	No	Yes	2388.174009	7832.135644
4060	4061	Yes	Yes	2216.017669	20911.695635
4231	4232	Yes	Yes	2291.617688	20837.209447
4831	4832	No	Yes	2216.329753	24737.081761
5461	5462	Yes	Yes	2247.421889	17926.723014
6075	6076	Yes	No	2413.319449	38540.572705
6334	6335	Yes	No	2343.797513	51095.293929
6882	6883	Yes	Yes	2287.173842	18692.144311
7437	7438	Yes	Yes	2461.506979	11878.557045
7815	7816	Yes	Yes	2578.469022	25706.647774

	Unnamed: 0	default	student	balance	income
8264	8265	Yes	No	2236.764215	37113.883070
8495	8496	Yes	Yes	2654.322576	21930.388879
8832	8833	Yes	Yes	2207.599054	19780.763519
8992	8993	Yes	Yes	2352.054949	24067.548104
9873	9874	No	No	2391.007739	50302.909557
9893	9894	Yes	No	2288.408082	52043.569052
9978	9979	Yes	No	2202.462395	47287.257108

In [18]:
```
df['default'].count()
```

Out[18]:
```
31
```

In [19]:
```
df['default'].value_counts(normalize=True)
```

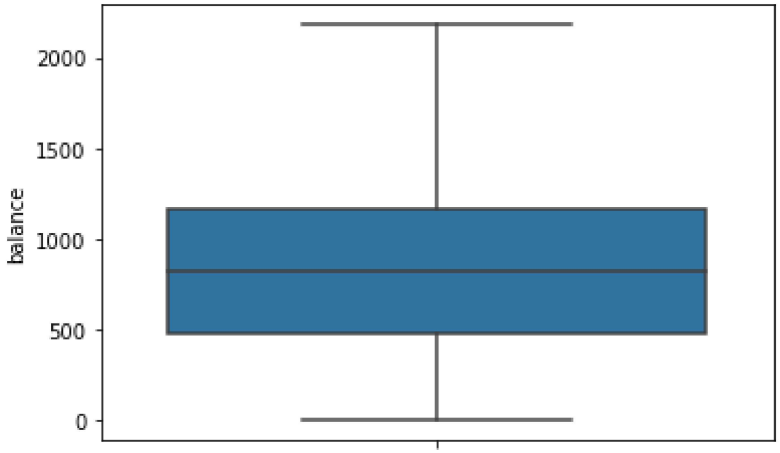
Out[19]:
```
Yes    0.83871
No     0.16129
Name: default, dtype: float64
```

In [20]:
```
df['default'].value_counts()
```

Out[20]:
```
Yes    26
No      5
Name: default, dtype: int64
```

In [21]:
```
Default['balance'] = np.where(Default['balance'] > UL, UL, Default['balance'])
```

In [22]:
```
sns.boxplot(y=Default['balance'])
plt.show()
```



In [23]:
```
Default = pd.get_dummies(Default,drop_first = True)
```

In [24]:
```
Default.head()
```

Out[24]:

	Unnamed: 0	balance	income	default_Yes	student_Yes
0	1	729.526495	44361.625074	0	0
1	2	817.180407	12106.134700	0	1
2	3	1073.549164	31767.138947	0	0
3	4	529.250605	35704.493935	0	0
4	5	785.655883	38463.495879	0	0

In [25]:
```
Default.columns = ['Unnamed','balance','income','default','student']
```

In [26]:
```
Default.head()
```

Out[26]:

	Unnamed	balance	income	default	student
0	1	729.526495	44361.625074	0	0
1	2	817.180407	12106.134700	0	1
2	3	1073.549164	31767.138947	0	0
3	4	529.250605	35704.493935	0	0
4	5	785.655883	38463.495879	0	0

In [27]:
```
from sklearn.model_selection import train_test_split
```

In [28]:
```
x = Default.drop('default',axis=1)
y = Default['default']
```

In [29]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = 21,stratify = y)
```

In [30]:
```
print(x_train.shape)
print(x_test.shape)
```

```
(7000, 4)
(3000, 4)
```

In [31]:
```
print(y_train.value_counts(normalize = True).round(2))
print(' ')
print(y_test.value_counts(normalize = True).round(2))
```

```
0    0.97
1    0.03
Name: default, dtype: float64

0    0.97
1    0.03
Name: default, dtype: float64
```

In [36]:
```
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=33, sampling_strategy=0.75)
x_res, y_res = sm.fit_sample(x_train,y_train)
```

In [37]:
```
from sklearn.linear_model import LogisticRegression
```

In [38]:
```
lr = LogisticRegression()
```

In [39]:
```
lr.fit(x_res,y_res)
```

Out[39]:
```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [40]:
```
y_pred = lr.predict(x_test)
```

In [41]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [42]:
```
confusion_matrix(y_test,y_pred)
```

Out[42]:
```
array([[2226,  674],
       [  21,   79]], dtype=int64)
```

In [43]:
```
(2226+79)/(2226+79+21+674)
```

Out[43]:
```
0.7683333333333333
```

Analyzed by

Md Raiesh, Enrollment number : **19UME116**, Registration number : **1911345**, B Tech,7**th** semester,Section : **A**, Mechanical Engineering Department, National Institute of Technology Agartala, Tripura 799046,

In [ ]: