In [1]:
```python
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

In [2]:
```python
default=pd.read_csv("C:\\Users\\Pranav\\Desktop\\DATA SCIENCE DATA\\CVC file\\attachment_default.csv")
default.head()
```

Out[2]:

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

In [3]:
```python
#shape of dataset
default.shape
```

Out[3]:
```
(10000, 4)
```

In [4]:
```python
#information about dataset
default.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   default  10000 non-null  object
 1   student  10000 non-null  object
 2   balance  10000 non-null  float64
 3   income   10000 non-null  float64
dtypes: float64(2), object(2)
memory usage: 312.6+ KB
```

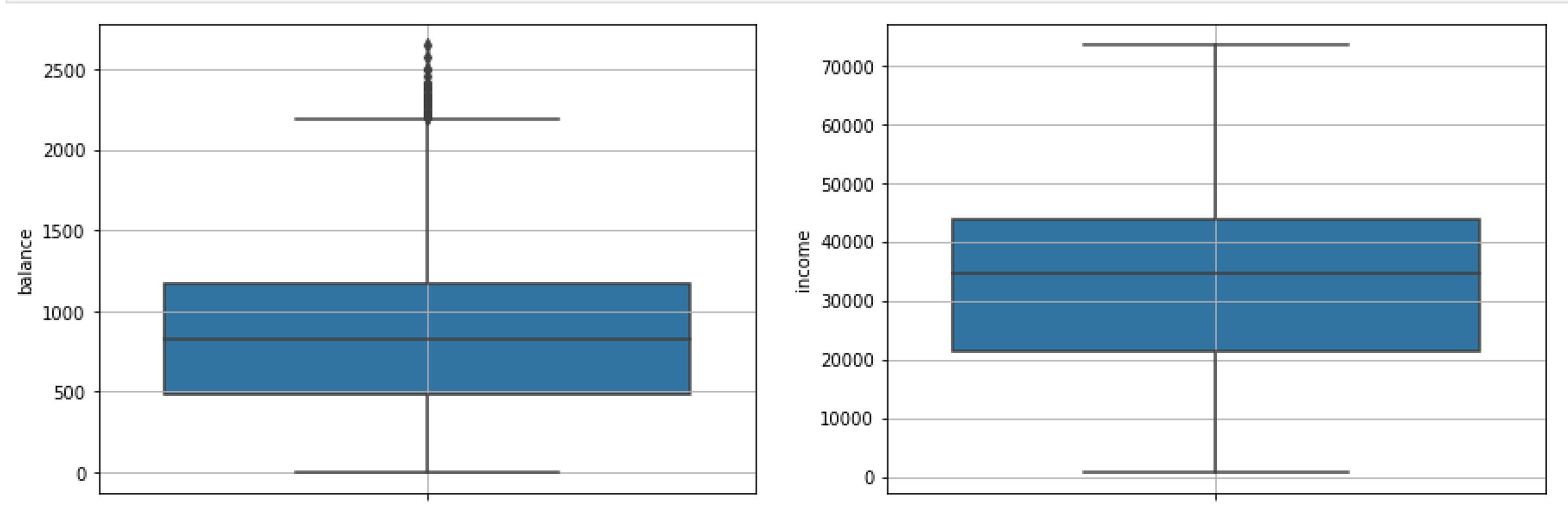
In [5]:
```python
#describe mathamathical information about dataset
default.describe()
```

Out[5]:

	balance	income
count	10000.000000	10000.000000
mean	835.374886	33516.981876
std	483.714985	13336.639563
min	0.000000	771.967729
25%	481.731105	21340.462903
50%	823.636973	34552.644802
75%	1166.308386	43807.729272
max	2654.322576	73554.233495

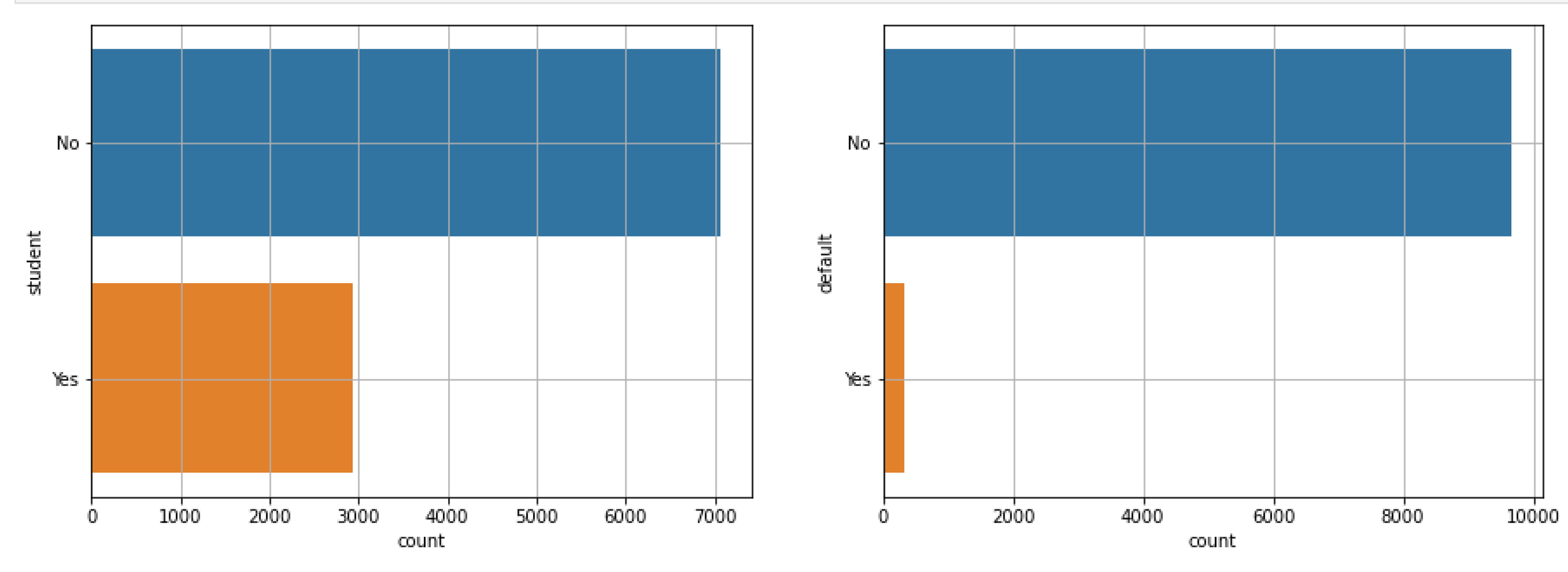
In [6]:
```python
# boxplt
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
sns.boxplot(y=default['balance'])
plt.grid(True)

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



In [7]:
```python
# countplot
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
sns.countplot(y=default['student'])
plt.grid(True)

plt.subplot(1,2,2)
sns.countplot(y=default['default'])
plt.grid(True)
plt.show()
```



In [8]:
```python
# measure the count of default
default['default'].value_counts()
```

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

In [9]:
```python
# measure the count of student
default['student'].value_counts()
```

Out[9]:
```
No     7056
Yes    2944
Name: student, dtype: int64
```

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

Out[10]:
```
No     0.7056
Yes    0.2944
Name: student, dtype: float64
```

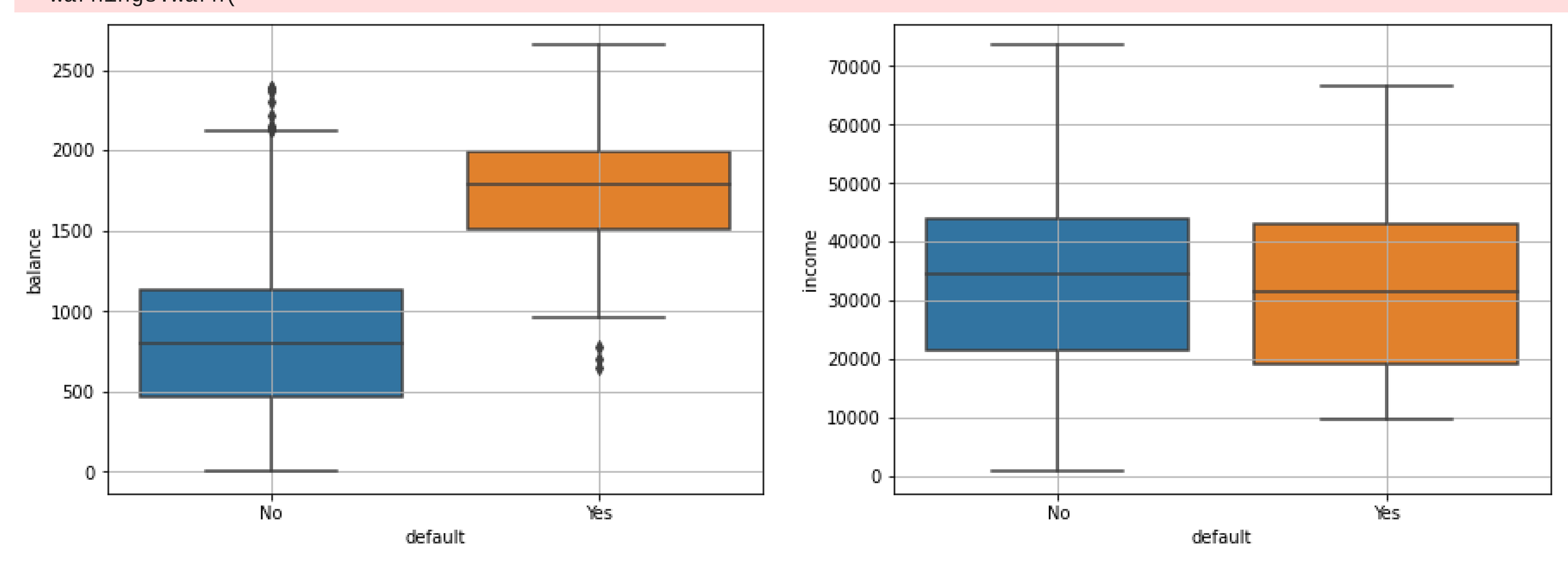
In [11]:
```python
default['default'].value_counts(normalize=True)
```

Out[11]:
```
No     0.9667
Yes    0.0333
Name: default, dtype: float64
```

In [12]:
```python
# boxplt
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
sns.boxplot(default['default'],default['balance'])
plt.grid(True)

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

```
C:\Users\Pranav\.continuum\ppp\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid pos
itional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
C:\Users\Pranav\.continuum\ppp\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid pos
itional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [13]:
```python
pd.crosstab(default['student'],default['default'],normalize='index').round(2)
```

Out[13]:

default	No	Yes
student		
No	0.97	0.03
Yes	0.96	0.04

In [14]:
```python
#heatmap
sns.heatmap(default[['balance','income']].corr(),annot=True)
plt.show()
```

In [15]:
```python
# findout null value in dataset
default.isnull().sum()
```

Out[15]:
```
default    0
student    0
balance    0
income     0
dtype: int64
```

In [16]:
```python
Q1 ,Q3=default['balance'].quantile([.25,.75])
IQR=Q3 -Q1
LL=Q1 -1.5*(IQR)
UL=Q3 +1.5*(IQR)
```

In [17]:
```python
UL
```

Out[17]:
```
2193.174308607817
```

In [18]:
```python
df=default[default['balance']>UL]
```

In [19]:
```python
df
```

Out[19]:

	default	student	balance	income
173	Yes	Yes	2205.799521	14271.492253
1136	Yes	No	2499.016750	51504.293960
1160	Yes	Yes	2502.684931	14947.519752
1359	Yes	No	2220.966201	40725.096207
1502	Yes	Yes	2332.878254	11770.234124
1609	Yes	Yes	2269.946966	18021.105948
2096	Yes	Yes	2261.848162	20030.165119
2140	No	Yes	2308.893236	19110.266412
2929	Yes	Yes	2387.314867	28296.914718
3162	Yes	Yes	2415.316994	17429.503375
3189	Yes	No	2228.472283	27438.348988
3702	No	Yes	2370.463612	24251.958722
3855	Yes	Yes	2321.882221	21331.314781
3913	Yes	Yes	2334.123559	19335.889287
3976	No	Yes	2388.174009	7832.135644
4060	Yes	Yes	2216.017669	20911.695635
4231	Yes	Yes	2291.617688	20837.209447
4831	No	Yes	2216.329753	24737.081761
5461	Yes	Yes	2247.421889	17926.723014
6075	Yes	No	2413.319449	38540.572705
6334	Yes	No	2343.797513	51095.293929
6882	Yes	Yes	2287.173842	18692.144311
7437	Yes	Yes	2461.506979	11878.557045
7815	Yes	Yes	2578.469022	25706.647774
8264	Yes	No	2236.764215	37113.883070
8495	Yes	Yes	2654.322576	21930.388879
8832	Yes	Yes	2207.599054	19780.763519
8992	Yes	Yes	2352.054949	24067.548104
9873	No	No	2391.007739	50302.909557
9893	Yes	No	2288.408082	52043.569052
9978	Yes	No	2202.462395	47287.257108

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

Out[20]:
```
No     9667
Yes     333
Name: default, dtype: int64
```

In [21]:
```python
default['default'].value_counts(normalize=True)
```

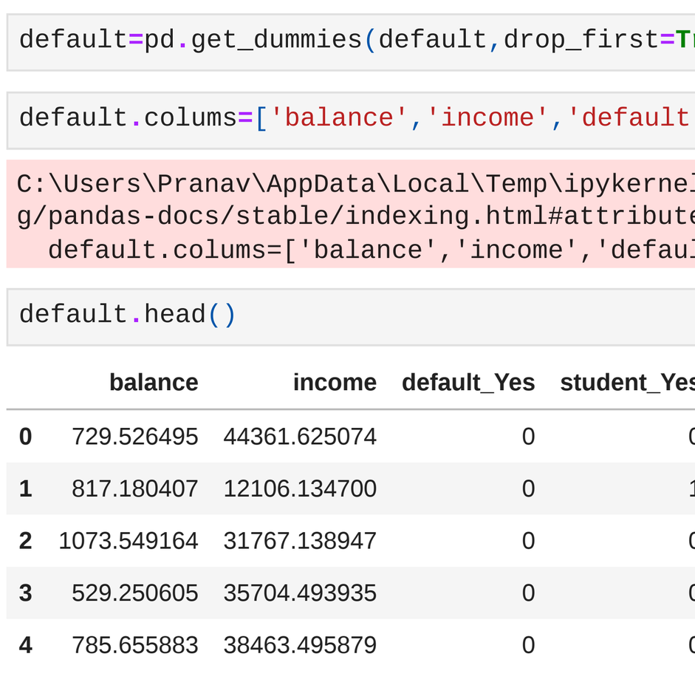
Out[21]:
```
No     0.9667
Yes    0.0333
Name: default, dtype: float64
```

In [22]:
```python
default['balance']=np.where(default['balance']>UL ,UL,default['balance'])
default['balance']
```

Out[22]:
```
0        729.526495
1        817.180407
2       1073.549164
3        529.250605
4        785.655883
           ...
9995     711.555020
9996     757.962918
9997     845.411989
9998    1569.009053
9999     200.922183
Name: balance, Length: 10000, dtype: float64
```

In [23]:
```python
#boxplt
sns.boxplot(default['balance'])
plt.grid(True)
plt.show()
```

```
C:\Users\Pranav\.continuum\ppp\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positi
onal argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [24]:
```python
default=pd.get_dummies(default,drop_first=True)
```

In [25]:
```python
default.colums=['balance','income','default','student']
```

```
C:\Users\Pranav\AppData\Local\Temp\ipykernel_20716\1950501991.py:1: UserWarning: Pandas doesn't allow columns to be created via a new attribute name - see https://pandas.pydata.or
g/pandas-docs/stable/indexing.html#attribute-access
  default.colums=['balance','income','default','student']
```

In [26]:
```python
default.head()
```

Out[26]:

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

In [27]:
```python
x=default.drop('default_Yes',axis=1)
y=default['default_Yes']
```

In [28]:
```python
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.32)
print("shape of x_train= ",x_train.shape)
print("shape of x_test= ",x_test.shape)
print("shape of y_train= ",y_train.shape)
print("shape of y_test= ",y_test.shape)
```

```
shape of x_train=  (6800, 3)
shape of x_test=  (3200, 3)
shape of y_train=  (6800,)
shape of y_test=  (3200,)
```

In [29]:
```python
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_Yes, dtype: float64

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

In [39]:
```python
!pip install SMOTE
```

```
Requirement already satisfied: SMOTE in c:\users\pranav\.continuum\ppp\lib\site-packages (0.1)
Requirement already satisfied: numpy>=1.14.3 in c:\users\pranav\.continuum\ppp\lib\site-packages (from SMOTE) (1.21.5)
Requirement already satisfied: scikit-learn>=0.19.1 in c:\users\pranav\.continuum\ppp\lib\site-packages (from SMOTE) (1.1.2)
Requirement already satisfied: scipy>=1.3.2 in c:\users\pranav\.continuum\ppp\lib\site-packages (from scikit-learn>=0.19.1->SMOTE) (1.7.3)
Requirement already satisfied: joblib>=1.0.0 in c:\users\pranav\.continuum\ppp\lib\site-packages (from scikit-learn>=0.19.1->SMOTE) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\pranav\.continuum\ppp\lib\site-packages (from scikit-learn>=0.19.1->SMOTE) (2.2.0)
```

In [43]:
```python
from imblearn.over_sampling import SMOTE
oversample = SMOTE()
x, y = oversample.fit_resample(X, y)
print(x.shape,y.shape)
```

```
(19334, 3) (19334,)
```

In [44]:
```python
lr=LogisticRegression()
```

In [45]:
```python
lr.fit(x, y)
```

Out[45]:
```
▾ LogisticRegression
LogisticRegression()
```

In [46]:
```python
y_pred=lr.predict(x_test)
```

In [47]:
```python
from sklearn.metrics import confusion_matrix ,classification_report
```

In [48]:
```python
confusion_matrix(y_test,y_pred)
```

Out[48]:
```
array([[2013, 1083],
       [  16,   88]], dtype=int64)
```

In [51]:
```python
#accracy of model
((2013+88)/(2013+88+1083+16))*100
```

Out[51]:
```
65.65625
```

In [ ]: