TASK-1

Data cleaning removing the missing values and outliers

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
```

In [2]:
```
path=r"C:\Users\Sruth\Downloads\Loan_prediction_data.csv"
Loan_df=pd.read_csv(path)
Loan_df
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
0	LP001002	Male	No	0	Graduate	No	58
1	LP001003	Male	Yes	1	Graduate	No	45
2	LP001005	Male	Yes	0	Graduate	Yes	30
3	LP001006	Male	Yes	0	Not Graduate	No	25
4	LP001008	Male	No	0	Graduate	No	60
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	29
610	LP002979	Male	Yes	3+	Graduate	No	41
611	LP002983	Male	Yes	1	Graduate	No	80
612	LP002984	Male	Yes	2	Graduate	No	75
613	LP002990	Female	No	0	Graduate	Yes	45

614 rows × 13 columns

In [3]:
```
Loan_df.head()  # gives first 5 rows
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome
0	LP001002	Male	No	0	Graduate	No	5849
1	LP001003	Male	Yes	1	Graduate	No	4583
2	LP001005	Male	Yes	0	Graduate	Yes	3000
3	LP001006	Male	Yes	0	Not Graduate	No	2583
4	LP001008	Male	No	0	Graduate	No	6000

In [4]:
```
Loan_df.tail() #gives last 5 rows
```

Out[4]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

In [5]:
```
Loan_df.shape
```

Out[5]:
```
(614, 13)
```

In [6]:
```
Loan_df.isnull()
```

Out[6]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncon
0	False	False	False	False	False	False	Fal
1	False	False	False	False	False	False	Fal
2	False	False	False	False	False	False	Fal
3	False	False	False	False	False	False	Fal
4	False	False	False	False	False	False	Fal
...	...	...	...	...	...	...	
609	False	False	False	False	False	False	Fal
610	False	False	False	False	False	False	Fal
611	False	False	False	False	False	False	Fal
612	False	False	False	False	False	False	Fal
613	False	False	False	False	False	False	Fal

614 rows × 13 columns

In [7]:
```
Loan_df.size
```

Out[7]:
```
7982
```

In [8]:
```
Loan_df.shape[0]*Loan_df.shape[1]
```

Out[8]:
```
7982
```

In [9]:
```
Loan_df.columns
```

Out[9]:
```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
      dtype='object')
```

In [10]:
```
len(Loan_df.columns)
```

Out[10]:
```
13
```

In [11]:
```
len(Loan_df)
```

Out[11]:
```
614
```

In [12]:
```
Loan_df.dtypes
```

Out[12]:
```
Loan_ID               object
Gender                object
Married               object
Dependents            object
Education             object
Self_Employed         object
ApplicantIncome        int64
CoapplicantIncome    float64
LoanAmount           float64
Loan_Amount_Term     float64
Credit_History       float64
Property_Area         object
Loan_Status           object
dtype: object
```

In [13]:
```
type(Loan_df.dtypes)
```

Out[13]:
```
pandas.core.series.Series
```

In [14]:
```
cat_cols=Loan_df.select_dtypes(include='object').columns
cat_cols
```

Out[14]:
```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'Property_Area', 'Loan_Status'],
      dtype='object')
```

In [15]:
```
num_cols=Loan_df.select_dtypes(exclude='object').columns
num_cols
```

Out[15]:
```
Index(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History'],
      dtype='object')
```

In [16]:
```
len(cat_cols),len(num_cols)
```

Out[16]:
```
(8, 5)
```

In [17]:
```
Loan_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object
 12  Loan_Status        614 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

STEP-1 Detecting null values

In [18]:
```
Loan_df.isnull().sum()
```

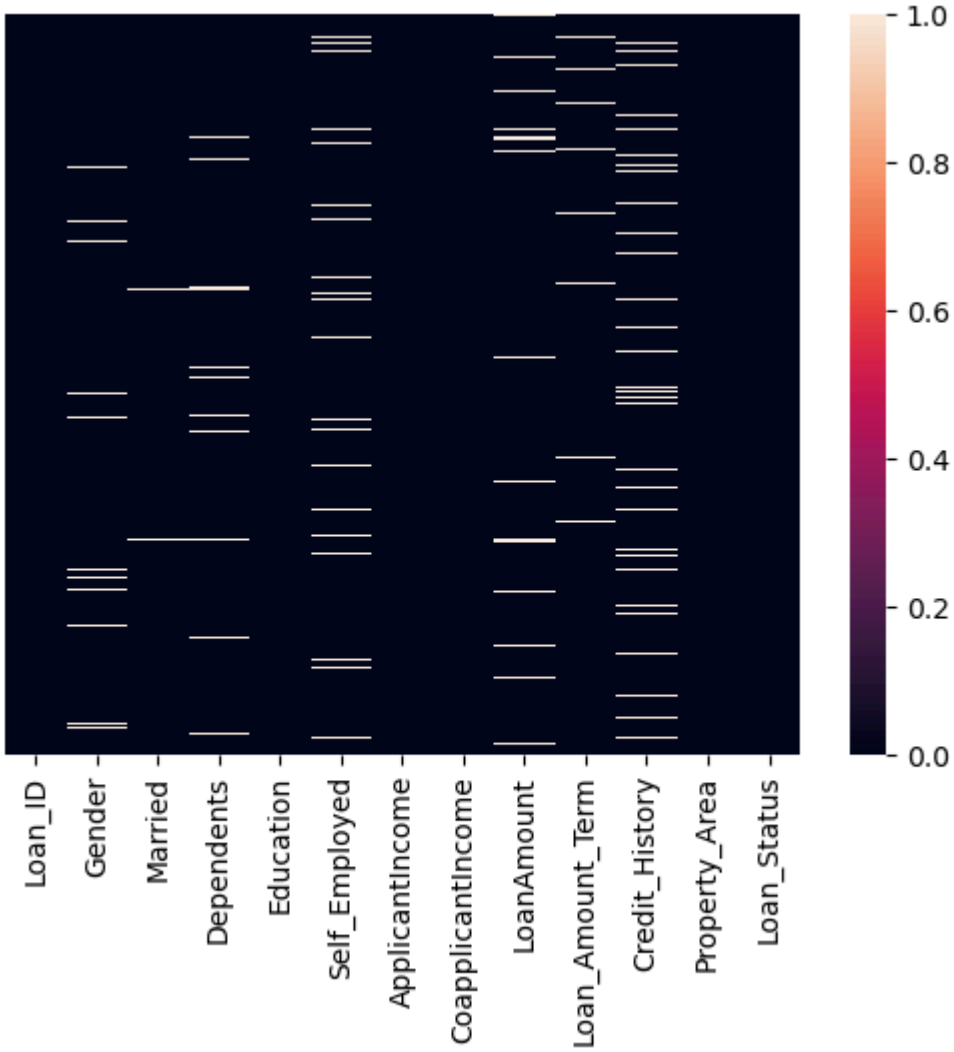
Out[18]:
```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [19]:
```
Loan_df.isnull().any() # missing values so(true means missing value there) , (fa
```

Out[19]:
```
Loan_ID              False
Gender                True
Married               True
Dependents            True
Education            False
Self_Employed         True
ApplicantIncome      False
CoapplicantIncome    False
LoanAmount            True
Loan_Amount_Term      True
Credit_History        True
Property_Area        False
Loan_Status          False
dtype: bool
```

In [20]:
```
sns.heatmap(Loan_df.isnull(),yticklabels=False)  # missing values represented wi
```

Out[20]: <Axes: >



step-2 Removin the missing values

In [21]:
```
Loan_df
```

Out[21]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
0	LP001002	Male	No	0	Graduate	No	58
1	LP001003	Male	Yes	1	Graduate	No	45
2	LP001005	Male	Yes	0	Graduate	Yes	30
3	LP001006	Male	Yes	0	Not Graduate	No	25
4	LP001008	Male	No	0	Graduate	No	60
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	29
610	LP002979	Male	Yes	3+	Graduate	No	41
611	LP002983	Male	Yes	1	Graduate	No	80
612	LP002984	Male	Yes	2	Graduate	No	75
613	LP002990	Female	No	0	Graduate	Yes	45

614 rows × 13 columns

In [27]:

```
Loan_df.dropna(inplace=True)
# missing value row dropped/deleted
```

In [28]:

```
Loan_df
```

Out[28]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
1	LP001003	Male	Yes	1	Graduate	No	45
2	LP001005	Male	Yes	0	Graduate	Yes	30
3	LP001006	Male	Yes	0	Not Graduate	No	25
4	LP001008	Male	No	0	Graduate	No	60
5	LP001011	Male	Yes	2	Graduate	Yes	54
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	29
610	LP002979	Male	Yes	3+	Graduate	No	41
611	LP002983	Male	Yes	1	Graduate	No	80
612	LP002984	Male	Yes	2	Graduate	No	75
613	LP002990	Female	No	0	Graduate	Yes	45

480 rows × 13 columns

In [24]:
```
len(Loan_df)
```

Out[24]:
```
480
```

Outlier analysis using box plot

In [30]:
```
Loan_df.select_dtypes(exclude='object').columns
```

Out[30]:
```
Index(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History'],
      dtype='object')
```

In [31]:
```
Loan_df
```

Out[31]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6
5	LP001011	Male	Yes	2	Graduate	Yes	5
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

480 rows × 13 columns

In [32]:
```
for i in num_cols:
    print(i)
```

```
ApplicantIncome
CoapplicantIncome
LoanAmount
Loan_Amount_Term
Credit_History
```

Rmoving outliers by using IQR

outliers data

In [34]:
```
num_cols
```

Out[34]:
```
Index(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History'],
      dtype='object')
```

Finding the outliers

- we already know that ouliers avaliable less than Q1-1.5*IQR and Q3+1.5*IQR
- step-1: - calculate Q1 = 25p(np.percentile/quartile) - calculate Q2=50p
    - ◦ calculate Q3=75p
- step-2: - calculate IQR= Q3-Q1
- step-3: - Calculate LB= Q1-1.5*IQR - Calculate UB= Q3+1.5*IQR
- step-4:
    - con1 = wage data < LB
    - con2 = wage data > UB
    - con3 = con1 or/and con2
- step-5 : - get the data data[con3]

In [65]:
```
Income_data=Loan_df['ApplicantIncome']
q1=np.percentile(Income_data,25)
q2=np.percentile(Income_data,50)
q3=np.percentile(Income_data,75)

IQR=q3-q1

lb=(q1-(1.5*IQR))
ub=(q3+(1.5*IQR))

con1=Income_data<lb
con2=Income_data>ub
con3=con1|con2
outliers_data=Income_data[con3]
outliers_data=Loan_df[con3]
outliers_data
```

Out[65]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
9	LP001020	Male	Yes	1	Graduate	No	128
34	LP001100	Male	No	3+	Graduate	No	125
54	LP001186	Female	Yes	1	Graduate	Yes	115
67	LP001233	Male	Yes	1	Graduate	No	107
106	LP001369	Male	Yes	2	Graduate	No	114
115	LP001401	Male	Yes	1	Graduate	No	145
119	LP001422	Female	No	0	Graduate	No	104
128	LP001451	Male	Yes	1	Graduate	Yes	105
138	LP001492	Male	No	0	Graduate	No	149
144	LP001508	Male	Yes	2	Graduate	No	117
146	LP001516	Female	Yes	2	Graduate	No	148
155	LP001536	Male	Yes	3+	Graduate	No	399
183	LP001637	Male	Yes	1	Graduate	No	338
185	LP001640	Male	Yes	0	Graduate	Yes	391
191	LP001656	Male	No	0	Graduate	No	120
199	LP001673	Male	No	0	Graduate	Yes	110
254	LP001844	Male	No	0	Graduate	Yes	162
258	LP001859	Male	Yes	0	Graduate	No	146
271	LP001891	Male	Yes	0	Graduate	No	111
278	LP001907	Male	Yes	0	Graduate	No	145
308	LP001996	Male	No	0	Graduate	No	202
324	LP002065	Male	Yes	3+	Graduate	No	150
369	LP002191	Male	Yes	0	Graduate	No	197
370	LP002194	Female	No	0	Graduate	Yes	157
409	LP002317	Male	Yes	3+	Graduate	No	810
424	LP002364	Male	Yes	0	Graduate	No	148
438	LP002403	Male	No	0	Graduate	Yes	104
443	LP002422	Male	No	1	Graduate	No	377
475	LP002527	Male	Yes	2	Graduate	Yes	165
478	LP002531	Male	Yes	1	Graduate	Yes	160
483	LP002541	Male	Yes	0	Graduate	No	108
487	LP002547	Male	Yes	1	Graduate	No	183

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
493	LP002582	Female	No	0	Not Graduate	Yes	17
509	LP002634	Female	No	1	Graduate	No	13
525	LP002699	Male	Yes	2	Graduate	Yes	17
534	LP002731	Female	No	0	Not Graduate	Yes	18
561	LP002813	Female	Yes	1	Graduate	Yes	19
572	LP002855	Male	Yes	2	Graduate	No	16
594	LP002938	Male	Yes	0	Graduate	Yes	16
604	LP002959	Female	Yes	1	Graduate	No	12

In [66]:
```
print(len(outliers_data))
```

```
40
```

Removed outliers in data set

non-outliers data

In [67]:
```
Income_data=Loan_df['ApplicantIncome']
q1=np.percentile(Income_data,25)
q2=np.percentile(Income_data,50)
q3=np.percentile(Income_data,75)

IQR=q3-q1

lb=(q1-(1.5*IQR))
ub=(q3+(1.5*IQR))

con1=Income_data>lb
con2=Income_data<ub
con3=con1&con2
non_outliers_data=Income_data[con3]
non_outliers_data
```

Out[67]:
```
1      4583
2      3000
3      2583
4      6000
5      5417
       ...
609    2900
610    4106
611    8072
612    7583
613    4583
Name: ApplicantIncome, Length: 440, dtype: int64
```

In [89]:
```
non_outliers_df=Loan_df[con3]
non_outliers_df.dropna(inplace=True)
```

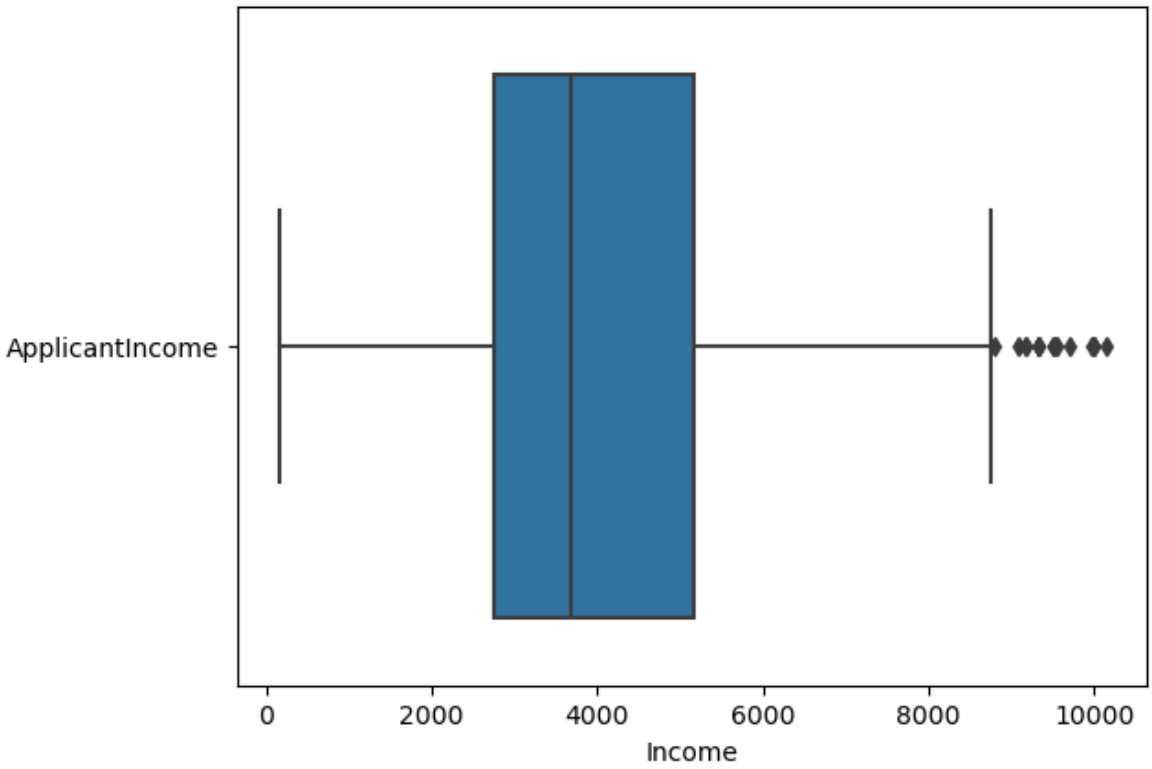
In [91]:
```
non_outliers_df
```

Out[91]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6
5	LP001011	Male	Yes	2	Graduate	Yes	5
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

440 rows × 13 columns

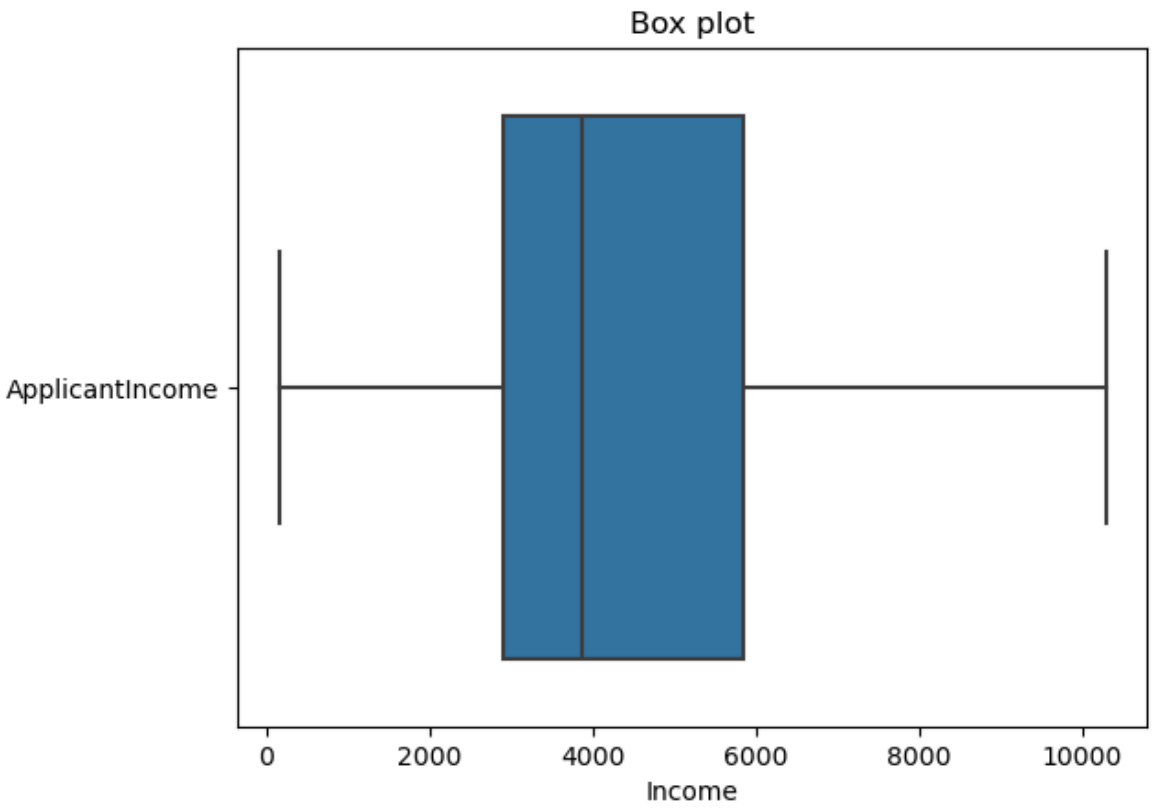
In [43]:
```
Amount=non_outliers_df[['ApplicantIncome']]
sns.boxplot(Amount,orient='h')
plt.xlabel('Income')
plt.show()
```



In [47]:
```
#capping - change the outlier values to upper or lower limit values
import warnings
warnings.filterwarnings('ignore')
non_outliers_df=Loan_df.copy()

non_outliers_df.loc[(non_outliers_df['ApplicantIncome']<lb),'ApplicantIncome']=l
non_outliers_df.loc[(non_outliers_df['ApplicantIncome']>ub),'ApplicantIncome']=u
plt.xlabel("Income")
plt.title("Box plot")
sns.boxplot(non_outliers_df[['ApplicantIncome']],orient='h')
```

Out[47]:
```
<Axes: title={'center': 'Box plot'}, xlabel='Income'>
```



In [93]:
```
print("old_data Before removing outliers:",len(Loan_df))
print("new_data After removing outliers:",len(non_outliers_df))
print(" Number of outliers:",len(Loan_df)-len(non_outliers_df))
```

```
old_data Before removing outliers: 480
new_data After removing outliers: 440
 Number of outliers: 40
```

In [92]:
```
non_outliers_df
```

Out[92]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6
5	LP001011	Male	Yes	2	Graduate	Yes	5
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

440 rows × 13 columns

In [94]:
```
non_outliers_df.isnull().sum()  # no missing values and outliers
```

Out[94]:
```
Loan_ID              0
Gender               0
Married              0
Dependents           0
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
dtype: int64
```

In [96]:
```
non_outliers_df
```

Out[96]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInco
1	LP001003	Male	Yes	1	Graduate	No	4
2	LP001005	Male	Yes	0	Graduate	Yes	3
3	LP001006	Male	Yes	0	Not Graduate	No	2
4	LP001008	Male	No	0	Graduate	No	6
5	LP001011	Male	Yes	2	Graduate	Yes	5
...	...	...	...	...	...	...	
609	LP002978	Female	No	0	Graduate	No	2
610	LP002979	Male	Yes	3+	Graduate	No	4
611	LP002983	Male	Yes	1	Graduate	No	8
612	LP002984	Male	Yes	2	Graduate	No	7
613	LP002990	Female	No	0	Graduate	Yes	4

440 rows × 13 columns

In [ ]: