In [60]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
```

In [61]:
```python
data = pd.read_csv('credit_card_churn.csv')
data.head(2)
```

Out[61]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 768805383 | Existing Customer | 45 | M | 3 | High School | Married | 60K−80K | Blue | 39 | ... |
| **1** | 818770008 | Existing Customer | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 | ... |

2 rows × 21 columns

# Checking nulls and duplicates

In [3]:
```python
data.isnull().sum()
```

Out[3]:
```
CLIENTNUM                   0
Attrition_Flag              0
Customer_Age                0
Gender                      0
Dependent_count             0
Education_Level             0
Marital_Status              0
Income_Category             0
Card_Category               0
Months_on_book              0
Total_Relationship_Count    0
Months_Inactive_12_mon      0
Contacts_Count_12_mon       0
Credit_Limit                0
Total_Revolving_Bal         0
Avg_Open_To_Buy             0
Total_Amt_Chng_Q4_Q1        0
Total_Trans_Amt             0
Total_Trans_Ct              0
Total_Ct_Chng_Q4_Q1         0
Avg_Utilization_Ratio       0
dtype: int64
```

In [4]:
```python
data.duplicated().sum()
```

Out[4]:
```
0
```

In [5]:
```python
data.dtypes
```

Out[5]:
```
CLIENTNUM                     int64
Attrition_Flag               object
Customer_Age                  int64
Gender                       object
Dependent_count               int64
Education_Level              object
Marital_Status               object
Income_Category              object
Card_Category                object
Months_on_book                int64
Total_Relationship_Count      int64
Months_Inactive_12_mon        int64
Contacts_Count_12_mon         int64
Credit_Limit                float64
Total_Revolving_Bal           int64
Avg_Open_To_Buy             float64
Total_Amt_Chng_Q4_Q1        float64
Total_Trans_Amt               int64
Total_Trans_Ct                int64
Total_Ct_Chng_Q4_Q1         float64
Avg_Utilization_Ratio       float64
dtype: object
```

# Summary Statistics

In [6]:
```
data.describe()
```

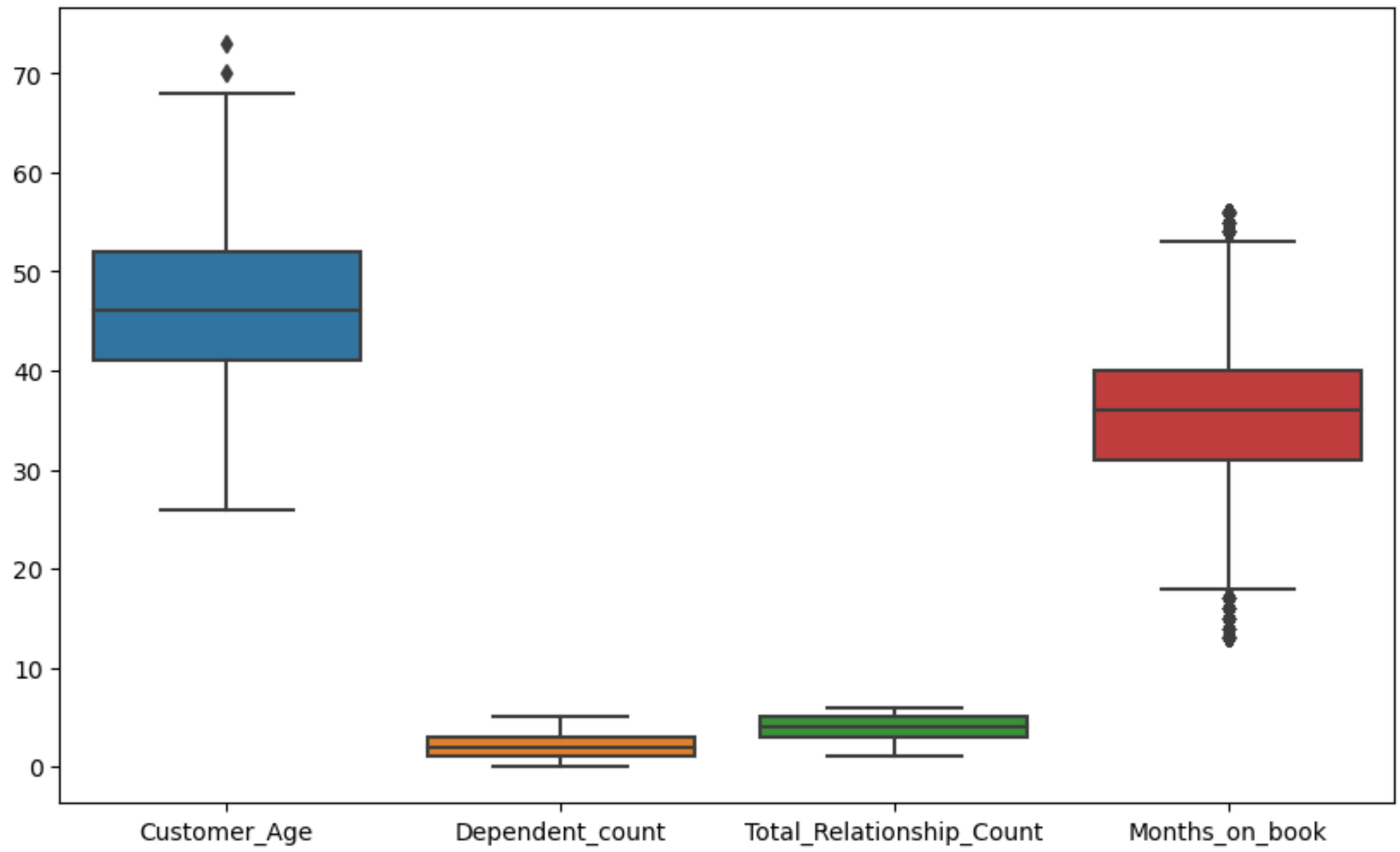
Out[6]:

| | CLIENTNUM | Customer_Age | Dependent_count | Months_on_book | Total_Relationship_Count | Months_Inactive_12_mon | Contacts_Count_12_mon | Credit_Lim |
|---|---|---|---|---|---|---|---|---|
| **count** | 1.012700e+04 | 10127.000000 | 10127.000000 | 10127.000000 | 10127.000000 | 10127.000000 | 10127.000000 | 10127.00000 |
| **mean** | 7.391776e+08 | 46.325960 | 2.346203 | 35.928409 | 3.812580 | 2.341167 | 2.455317 | 8631.95369 |
| **std** | 3.690378e+07 | 8.016814 | 1.298908 | 7.986416 | 1.554408 | 1.010622 | 1.106225 | 9088.77665 |
| **min** | 7.080821e+08 | 26.000000 | 0.000000 | 13.000000 | 1.000000 | 0.000000 | 0.000000 | 1438.30000 |
| **25%** | 7.130368e+08 | 41.000000 | 1.000000 | 31.000000 | 3.000000 | 2.000000 | 2.000000 | 2555.00000 |
| **50%** | 7.179264e+08 | 46.000000 | 2.000000 | 36.000000 | 4.000000 | 2.000000 | 2.000000 | 4549.00000 |
| **75%** | 7.731435e+08 | 52.000000 | 3.000000 | 40.000000 | 5.000000 | 3.000000 | 3.000000 | 11067.50000 |
| **max** | 8.283431e+08 | 73.000000 | 5.000000 | 56.000000 | 6.000000 | 6.000000 | 6.000000 | 34516.00000 |

## Checking Outliers

In [7]:
```
data_number1= data[['Customer_Age', 'Dependent_count', 'Total_Relationship_Count','Months_on_book']]
```

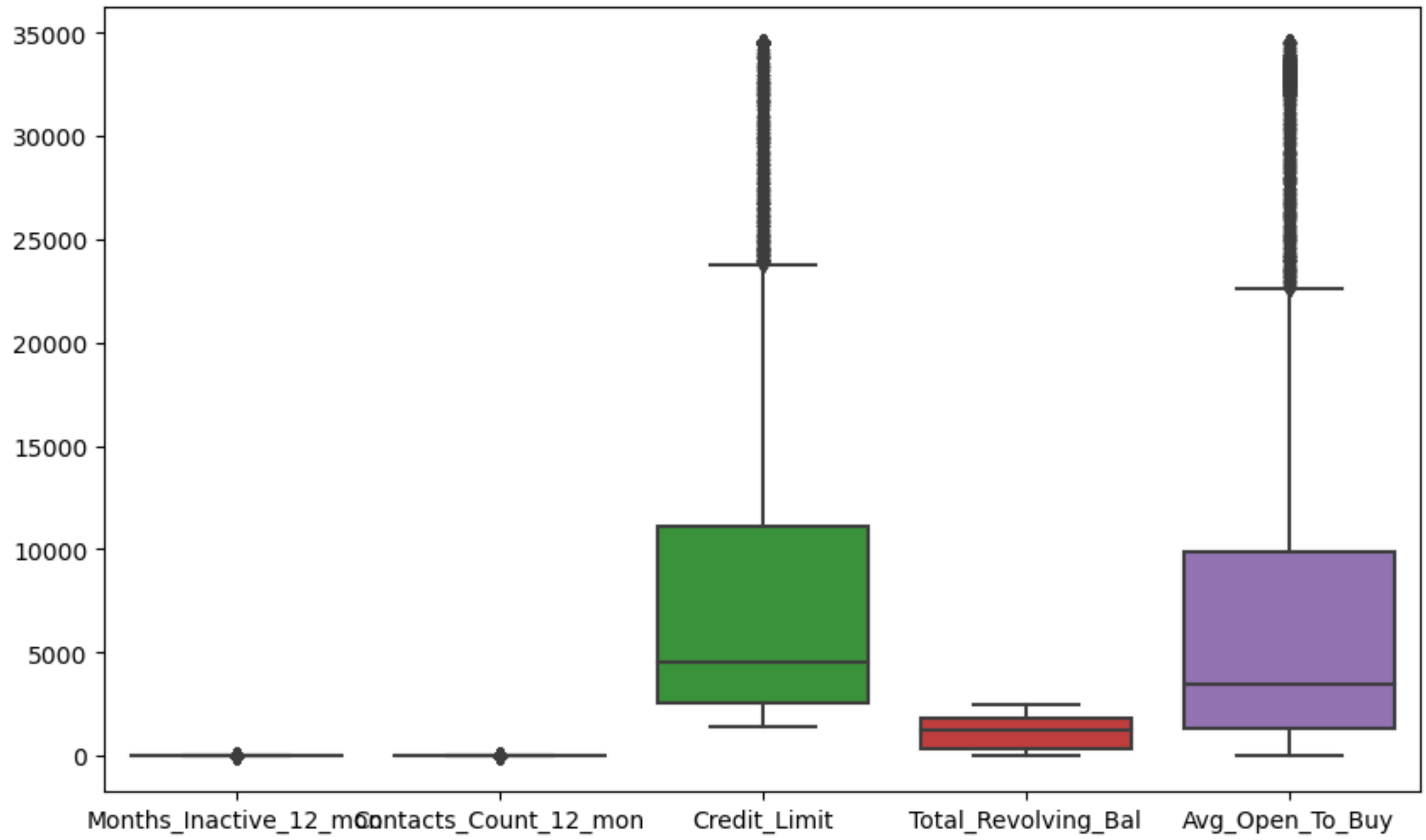
In [8]:
```
plt.figure(figsize=(10,6))
sns.boxplot(data= data_number1)
plt.show()
```




In [9]:
```
data_number2 = data[['Months_Inactive_12_mon','Contacts_Count_12_mon', 'Credit_Limit', 'Total_Revolving_Bal', 'Avg_Open_To_Buy']]
```

In [10]:
```
plt.figure(figsize=(10,6))
sns.boxplot(data=data_number2)
```

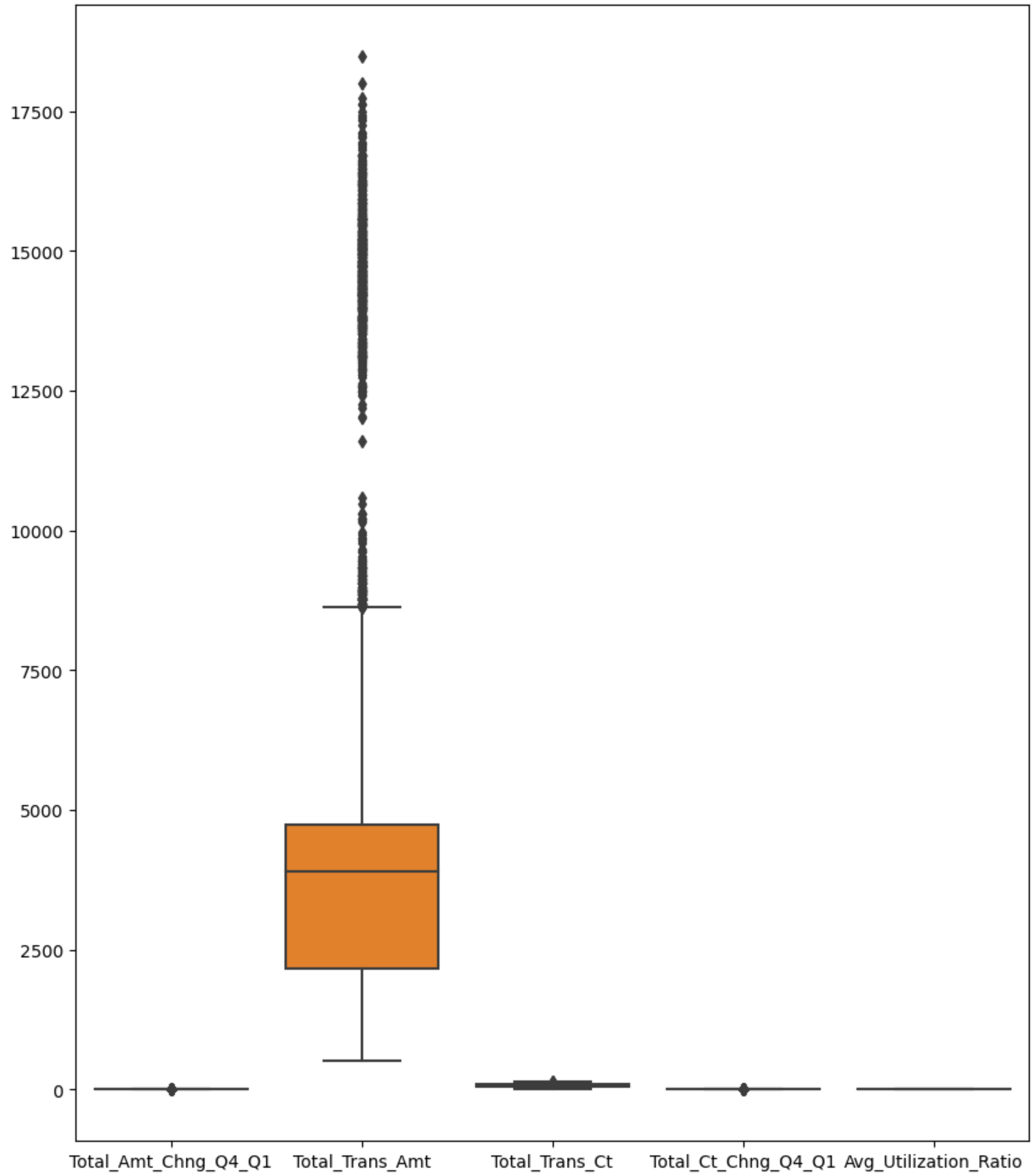
Out[10]: <AxesSubplot:>




In [11]:
```
data_number3 = data[['Total_Amt_Chng_Q4_Q1', 'Total_Trans_Amt','Total_Trans_Ct', 'Total_Ct_Chng_Q4_Q1', 'Avg_Utilization_Ratio']]
```

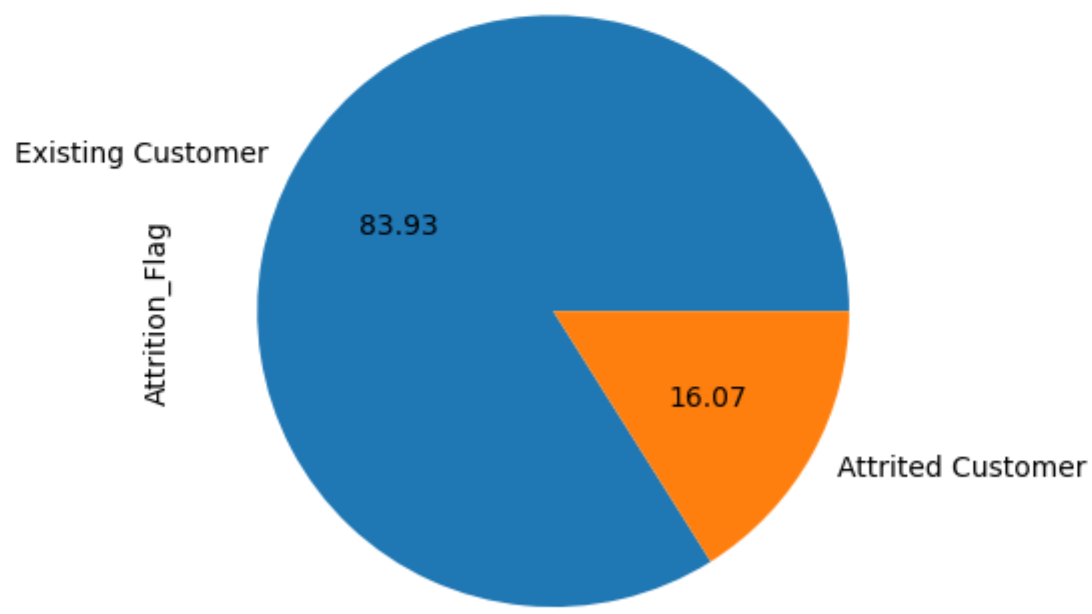
In [12]:
```
plt.figure(figsize=(10,12))

sns.boxplot(data=data_number3)
```

Out[12]: <AxesSubplot:>




In [13]:
```
data['Attrition_Flag'].value_counts().plot.pie(autopct='%.2f')
#The data is not balance
```

Out[13]: <AxesSubplot:ylabel='Attrition_Flag'>



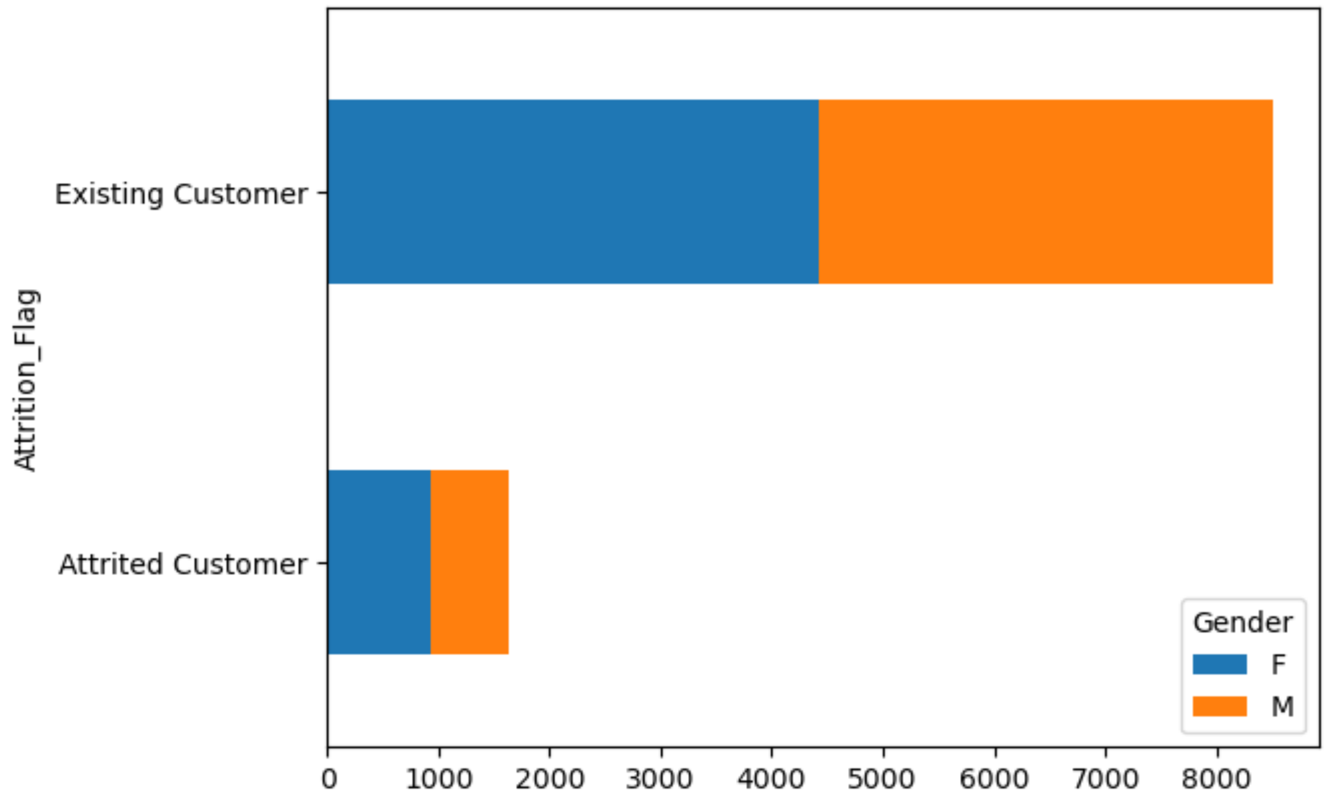
In [112]:
```python
counts = data.groupby(['Attrition_Flag','Gender']).size().unstack(fill_value=0)
counts
```

Out[112]:

| Gender | F | M |
|---|---|---|
| **Attrition_Flag** | | |
| **Attrited Customer** | 930 | 697 |
| **Existing Customer** | 4428 | 4072 |

In [113]:
```python
counts.plot(kind='barh', stacked=True)
```

Out[113]: <AxesSubplot:ylabel='Attrition_Flag'>



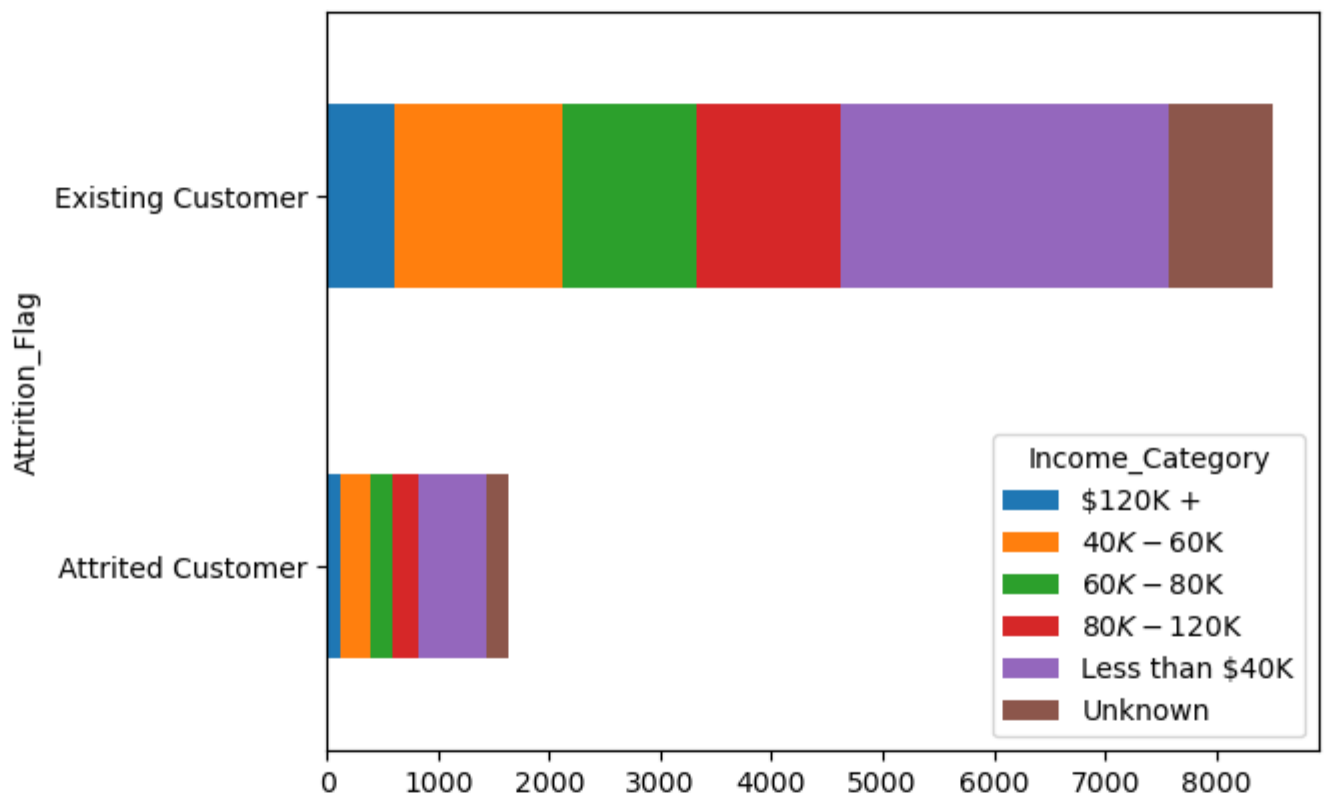

In [114]:
```python
counts_2 = data.groupby(['Attrition_Flag','Income_Category']).size().unstack(fill_value=0)
counts_2
```

Out[114]:

| Income_Category | $120K + | $40K – $60K | $60K – $80K | $80K – $120K | Less than $40K | Unknown |
|---|---|---|---|---|---|---|
| **Attrition_Flag** | | | | | | |
| **Attrited Customer** | 126 | 271 | 189 | 242 | 612 | 187 |
| **Existing Customer** | 601 | 1519 | 1213 | 1293 | 2949 | 925 |

In [116]:
```python
counts_2.plot(kind='barh', stacked=True)
```

Out[116]: <AxesSubplot:ylabel='Attrition_Flag'>




In [14]:
```python
from sklearn.preprocessing import LabelEncoder
```

In [15]:
```python
le = LabelEncoder()
data['Attrition_Flag'] = le.fit_transform(data['Attrition_Flag'])
data.head(2)
```

Out[15]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 768805383 | 1 | 45 | M | 3 | High School | Married | $60K-80K$ | Blue | 39 | ... |
| 1 | 818770008 | 1 | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 | ... |

2 rows × 21 columns

## Checking the correlation

In [16]:
```python
data_corr = data.corr()
```

In [17]:
```python
plt.figure(figsize=(14,12))
sns.heatmap(data_corr, cbar=True, annot=True)
```

Out[17]: <AxesSubplot:>


## Checking the skewness

In [18]:
```python
from scipy.stats import skew
```

In [19]:
```python
Num_data = data[['Customer_Age', 'Dependent_count', 'Months_on_book',
'Total_Relationship_Count',
'Months_Inactive_12_mon',
'Contacts_Count_12_mon',
'Credit_Limit',
'Total_Revolving_Bal',
'Avg_Open_To_Buy',
'Total_Amt_Chng_Q4_Q1',
'Total_Trans_Amt',
'Total_Trans_Ct',
'Total_Ct_Chng_Q4_Q1',
'Avg_Utilization_Ratio']]
```

In [20]:
```python
for col in Num_data:
    print(col)
    print(skew(Num_data[col]))

    plt.figure()
    sns.distplot(Num_data[col])
```

```
Customer_Age
-0.033600038574644426

C:\Users\Romelio Villar Jr\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with sim
ilar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)

Dependent_count
-0.02082245083419453
Months_on_book

C:\Users\Romelio Villar Jr\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with sim
ilar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)

-0.1065495749017217

C:\Users\Romelio Villar Jr\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with sim
```

## Feature Selection

In [ ]:

We will use Point Biserial. The point biserial correlation coefficient, which is a special case of Pearson's correlation coefficient.
It measures the relationship between two variables
One continuous variable and One naturally binary variable.
In our dataset, the "ATTRITION FLAG" is a binary variable.
We will examine the correlation of numerical variables with the "Attrition_flag.

In [21]:
```
data['Attrition_Flag'].dtypes
```

Out[21]:
```
dtype('int32')
```

In [22]:
```
target = data['Attrition_Flag']
feature = data[['Customer_Age', 'Dependent_count', 'Months_on_book',
'Total_Relationship_Count',
'Months_Inactive_12_mon',
'Contacts_Count_12_mon',
'Credit_Limit',
'Total_Revolving_Bal',
'Avg_Open_To_Buy',
'Total_Amt_Chng_Q4_Q1',
'Total_Trans_Amt',
'Total_Trans_Ct',
'Total_Ct_Chng_Q4_Q1',
'Avg_Utilization_Ratio']]
```

In [23]:
```
from sklearn.preprocessing import LabelEncoder
```

In [24]:
```
le = LabelEncoder()
```

In [25]:
```
data['Attrition_Flag'] = le.fit_transform(target)
data['Attrition_Flag']
```

Out[25]:
```
0        1
1        1
2        1
3        1
4        1
        ..
10122    1
10123    0
10124    0
10125    0
10126    0
Name: Attrition_Flag, Length: 10127, dtype: int64
```

In [26]:
```
feature.astype('int32').dtypes
```

Out[26]:
```
Customer_Age                int32
Dependent_count             int32
Months_on_book              int32
Total_Relationship_Count    int32
Months_Inactive_12_mon      int32
Contacts_Count_12_mon       int32
Credit_Limit                int32
Total_Revolving_Bal         int32
Avg_Open_To_Buy             int32
Total_Amt_Chng_Q4_Q1        int32
Total_Trans_Amt             int32
Total_Trans_Ct              int32
Total_Ct_Chng_Q4_Q1         int32
Avg_Utilization_Ratio       int32
dtype: object
```

In [27]:
```
print("Target Shape:", target.shape)
print("Feature Shape:", feature.shape)
```

```
Target Shape: (10127,)
Feature Shape: (10127, 14)
```

In [28]:
```python
for column in feature.columns:
    point_biserial_corr, p_value = stats.pointbiserialr(target, feature[column])
    print(f'Feature: {column}')
    print(f'Point-Biserial Correlation: {point_biserial_corr}')
    print(f'P-value: {p_value}')
```

```
Feature: Customer_Age
Point-Biserial Correlation: -0.01820313853255065
P-value: 0.06698688501759016
Feature: Dependent_count
Point-Biserial Correlation: -0.018990596311193708
P-value: 0.056002392535092434
Feature: Months_on_book
Point-Biserial Correlation: -0.01368685117790971
P-value: 0.1684370287649442
Feature: Total_Relationship_Count
Point-Biserial Correlation: 0.15000522801913754
P-value: 4.829281002183993e-52
Feature: Months_Inactive_12_mon
Point-Biserial Correlation: -0.152448806326925
P-value: 1.0326639995930894e-53
Feature: Contacts_Count_12_mon
Point-Biserial Correlation: -0.2044905099816044
P-value: 4.697489630751521e-96
Feature: Credit_Limit
Point-Biserial Correlation: 0.023872994836161524
P-value: 0.01628535720539447
Feature: Total_Revolving_Bal
Point-Biserial Correlation: 0.2630528831292032
P-value: 6.630148455417239e-160
Feature: Avg_Open_To_Buy
Point-Biserial Correlation: 0.00028507749393779595
P-value: 0.977116089445888
Feature: Total_Amt_Chng_Q4_Q1
Point-Biserial Correlation: 0.13106284781447014
P-value: 4.836642703584966e-40
Feature: Total_Trans_Amt
Point-Biserial Correlation: 0.168598381410079
P-value: 1.857438655661277e-65
Feature: Total_Trans_Ct
Point-Biserial Correlation: 0.37140270118892776
P-value: 0.0
Feature: Total_Ct_Chng_Q4_Q1
Point-Biserial Correlation: 0.29005400688089117
P-value: 1.6477247846937629e-195
Feature: Avg_Utilization_Ratio
Point-Biserial Correlation: 0.178410331561747
P-value: 3.3576893282456845e-73
```

In [29]:
```python
data
```

Out[29]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 768805383 | 1 | 45 | M | 3 | High School | Married | $60K-$80K | Blue | 39 |
| **1** | 818770008 | 1 | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 |
| **2** | 713982108 | 1 | 51 | M | 3 | Graduate | Married | $80K-$120K | Blue | 36 |
| **3** | 769911858 | 1 | 40 | F | 4 | High School | Unknown | Less than $40K | Blue | 34 |
| **4** | 709106358 | 1 | 40 | M | 3 | Uneducated | Married | $60K-$80K | Blue | 21 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **10122** | 772366833 | 1 | 50 | M | 2 | Graduate | Single | $40K-$60K | Blue | 40 |
| **10123** | 710638233 | 0 | 41 | M | 2 | Unknown | Divorced | $40K-$60K | Blue | 25 |
| **10124** | 716506083 | 0 | 44 | F | 1 | High School | Married | Less than $40K | Blue | 36 |
| **10125** | 717406983 | 0 | 30 | M | 2 | Graduate | Unknown | $40K-$60K | Blue | 36 |
| **10126** | 714337233 | 0 | 43 | F | 2 | Graduate | Married | Less than $40K | Silver | 25 |

10127 rows × 21 columns

In [30]:
```python
from sklearn.preprocessing import LabelEncoder as le

le = LabelEncoder()
data['Marital_Status'] = le.fit_transform(data['Marital_Status'])

le = LabelEncoder()
data['Education_Level'] = le.fit_transform(data['Education_Level'])

le = LabelEncoder()
data['Card_Category'] = le.fit_transform(data['Card_Category'])

# Encode 'Gender'
data['Gender'] = le.fit_transform(data['Gender'])

# Encode 'Income_Category'
data['Income_Category'] = le.fit_transform(data['Income_Category'])
data.head(2)
```

Out[30]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 768805383 | 1 | 45 | 1 | 3 | 3 | 1 | 2 | 0 | 39 | ... |
| **1** | 818770008 | 1 | 49 | 0 | 5 | 2 | 2 | 4 | 0 | 44 | ... |

2 rows × 21 columns

In [31]:
```python
data.columns
```

Out[31]:
```
Index(['CLIENTNUM', 'Attrition_Flag', 'Customer_Age', 'Gender',
       'Dependent_count', 'Education_Level', 'Marital_Status',
       'Income_Category', 'Card_Category', 'Months_on_book',
       'Total_Relationship_Count', 'Months_Inactive_12_mon',
       'Contacts_Count_12_mon', 'Credit_Limit', 'Total_Revolving_Bal',
       'Avg_Open_To_Buy', 'Total_Amt_Chng_Q4_Q1', 'Total_Trans_Amt',
       'Total_Trans_Ct', 'Total_Ct_Chng_Q4_Q1', 'Avg_Utilization_Ratio'],
      dtype='object')
```

In [32]:
```python
target.dtypes
```

Out[32]:
```
dtype('int32')
```

In [33]:
```python
categorical_fetaures = data[['Gender', 'Education_Level', 'Marital_Status', 'Income_Category', 'Card_Category']]
```

In [34]:
```python
from sklearn.feature_selection import chi2
```

In [35]:
```python
Chi_scores = chi2(categorical_fetaures, target)
```

In [36]:
```python
Chi_scores[0]
```

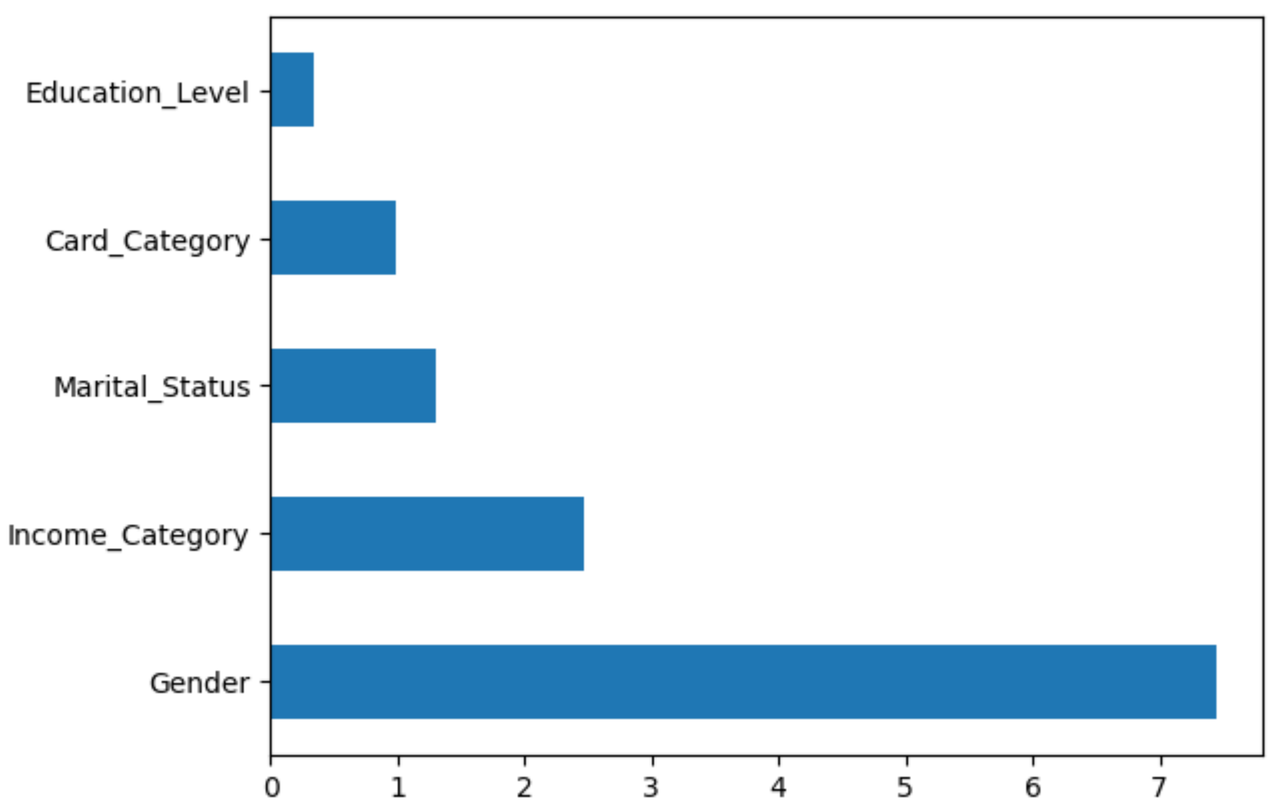
Out[36]:
```
array([7.4432227 , 0.33923111, 1.30275451, 2.47516959, 0.98612022])
```

In [37]:
```python
Chi_scores[1]
```

Out[37]:
```
array([0.00636758, 0.56027338, 0.25371069, 0.11565697, 0.32069248])
```

In [38]:
```python
Chi_values = pd.Series(Chi_scores[0], categorical_fetaures.columns)
Chi_values.sort_values(ascending=False, inplace=True)
Chi_values.plot(kind='barh') #the higher the better
```

Out[38]:
```
&lt;AxesSubplot:&gt;
```

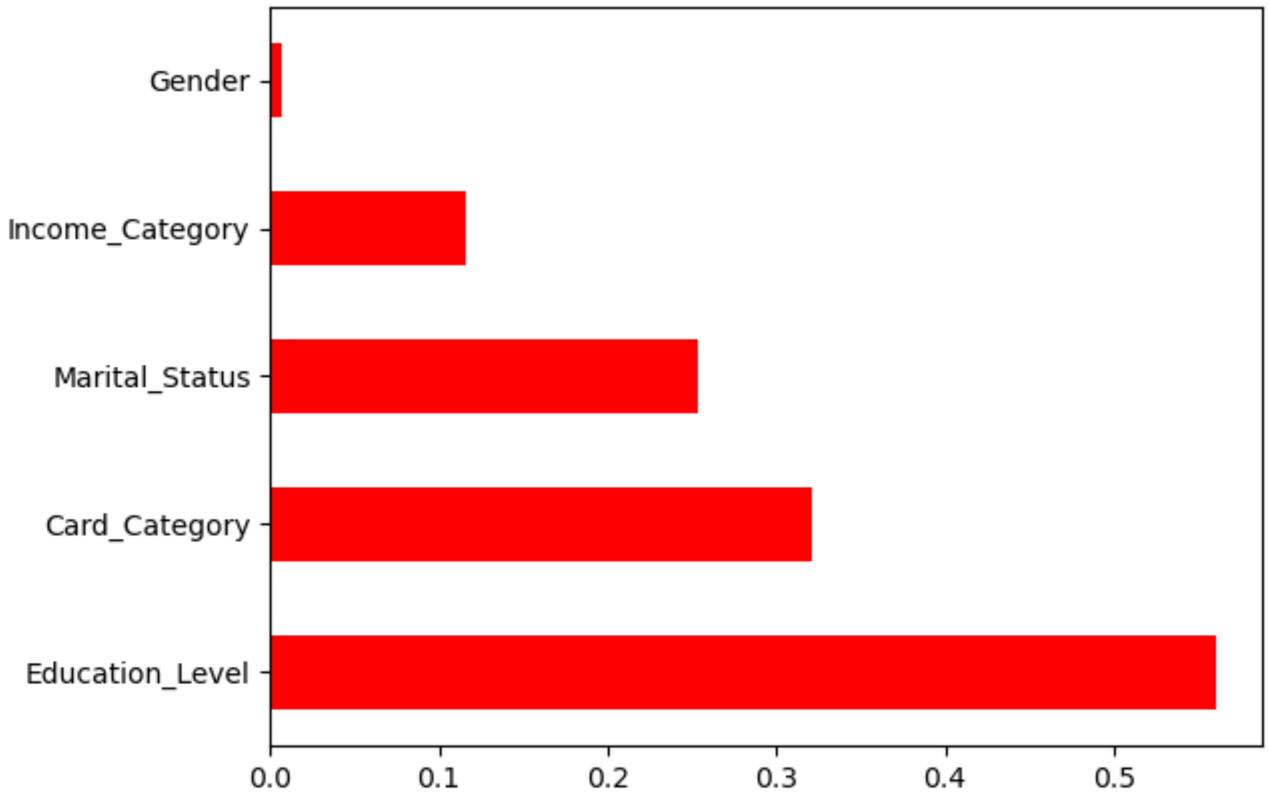


In [39]:
```
P_values = pd.Series(Chi_scores[1], categorical_fetaures.columns)
P_values.sort_values(ascending=False, inplace=True)
P_values
```

Out[39]:
```
Education_Level    0.560273
Card_Category      0.320692
Marital_Status     0.253711
Income_Category    0.115657
Gender             0.006368
dtype: float64
```

In [40]:
```
P_values.plot(kind='barh', color='red') #The lower the better
```

Out[40]: <AxesSubplot:>




In [41]:
```
Finaldata = data[['Attrition_Flag', 'Gender', 'Income_Category', 'Total_Trans_Ct', 'Avg_Utilization_Ratio', 'Total_Revolving_Bal',
```

In [42]:
```
X = Finaldata.drop('Attrition_Flag', axis=1)
y = Finaldata['Attrition_Flag']
```

In [49]:
```
data.head(2)
```

Out[49]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 768805383 | 1 | 45 | 1 | 3 | 3 | 1 | 2 | 0 | 39 | ... |
| **1** | 818770008 | 1 | 49 | 0 | 5 | 2 | 2 | 4 | 0 | 44 | ... |

2 rows × 21 columns