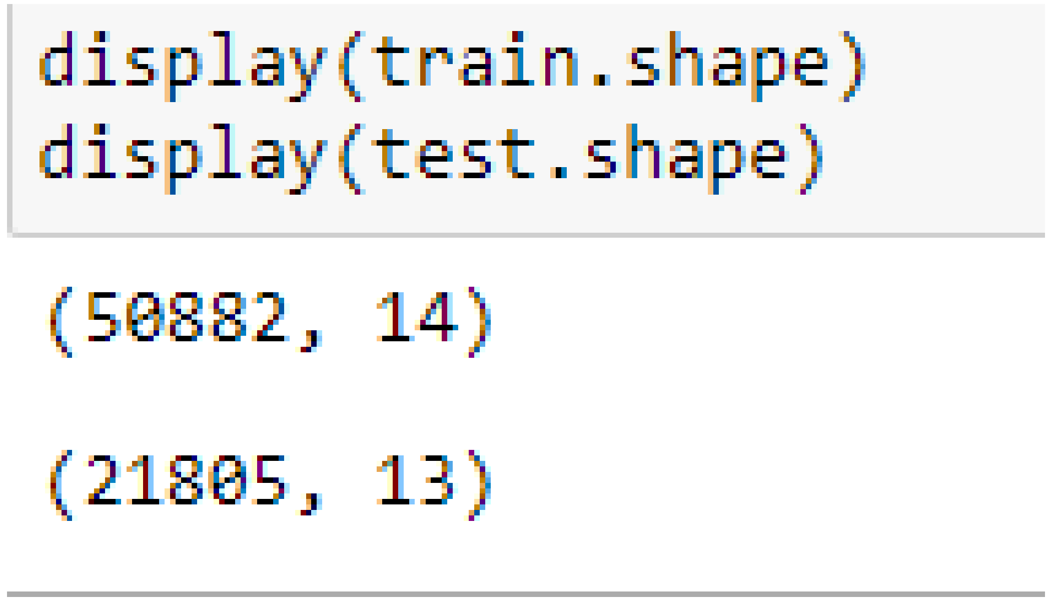
Health Insurance Lead Prediction

Problem Statement

Descriptive Statistics And EDA

Understanding the dataset



```
display(train.shape)
display(test.shape)
```

```
(50882, 14)

(21805, 13)
```

- **Total there are two datasets**

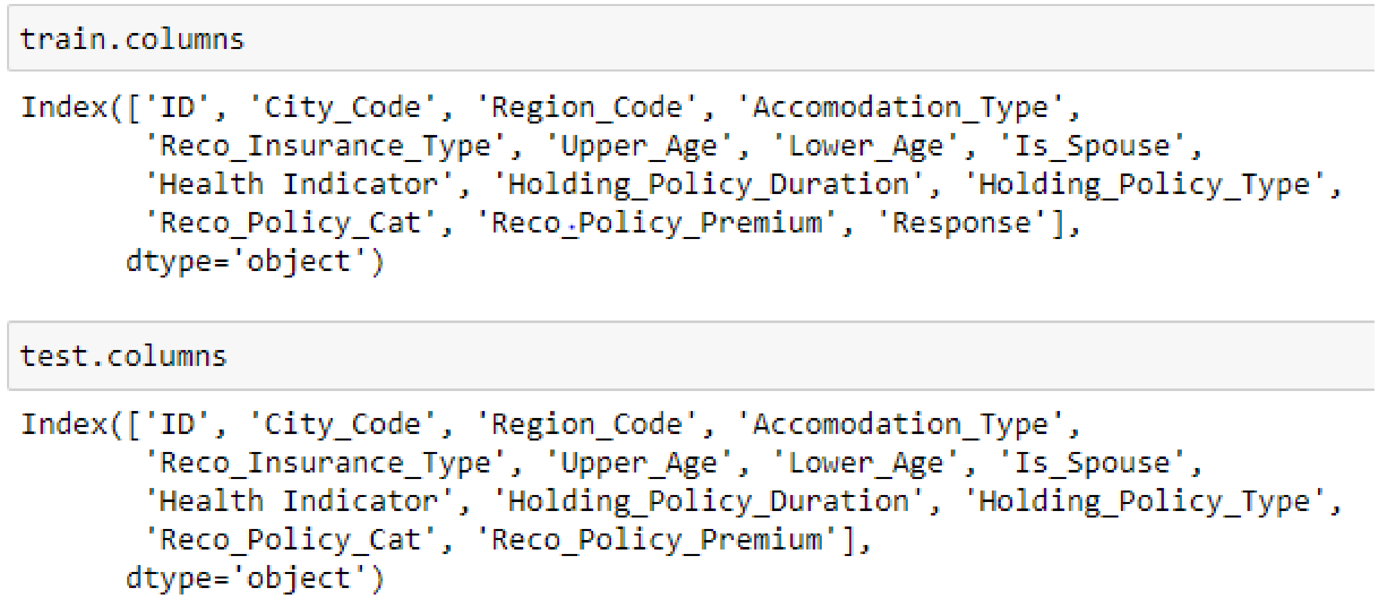
  1) Train.csv

  2) Test.csv

- **Dimension of dataset**

Columns in both datasets

- The only difference between the two datasets is the '**<u>Response</u>**' column as in testing phase we will be predict response of the customers.



```
train.columns
```

```
Index(['ID', 'City_Code', 'Region_Code', 'Accomodation_Type',
       'Reco_Insurance_Type', 'Upper_Age', 'Lower_Age', 'Is_Spouse',
       'Health Indicator', 'Holding_Policy_Duration', 'Holding_Policy_Type',
       'Reco_Policy_Cat', 'Reco_Policy_Premium', 'Response'],
      dtype='object')
```

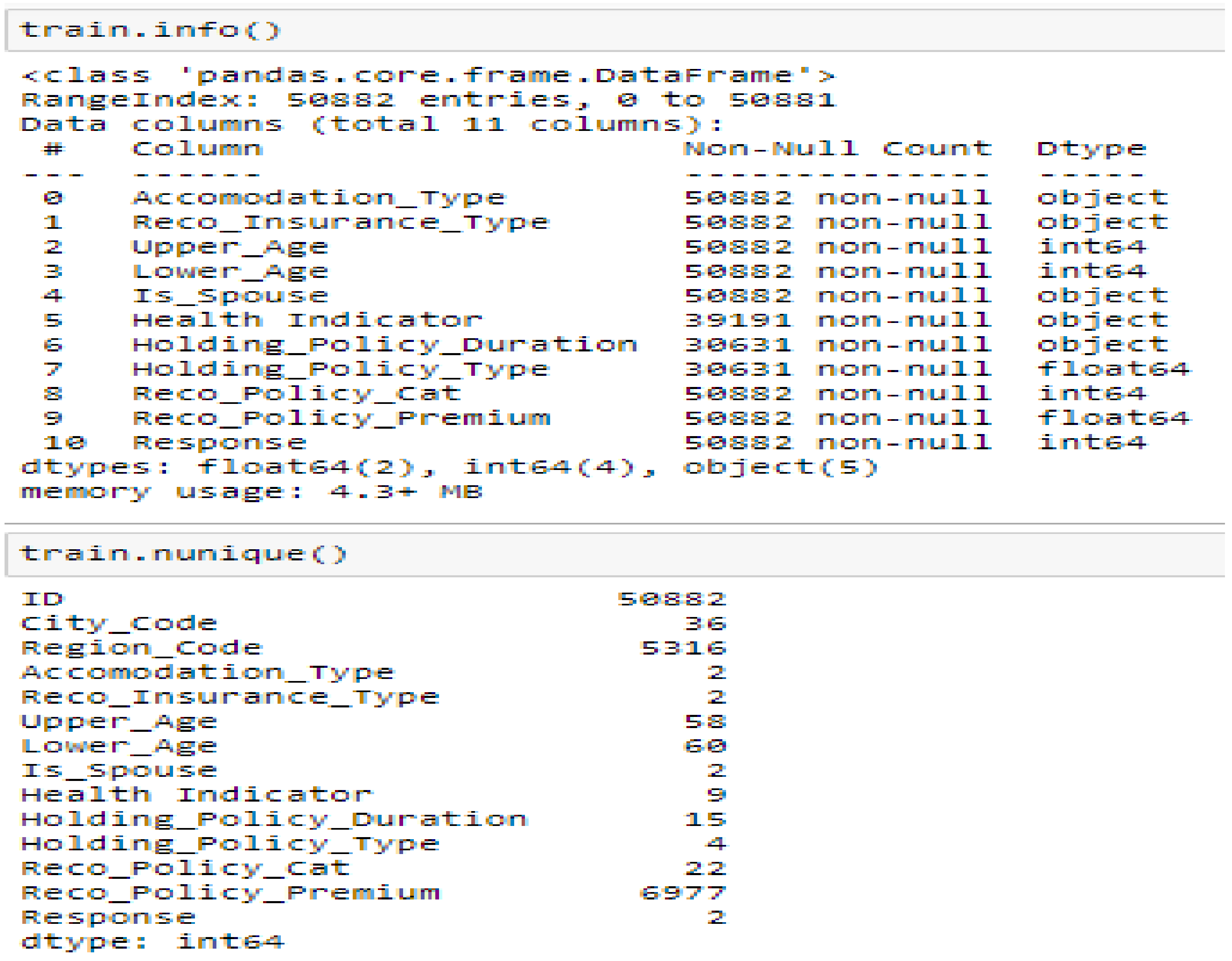
```
test.columns
```

```
Index(['ID', 'City_Code', 'Region_Code', 'Accomodation_Type',
       'Reco_Insurance_Type', 'Upper_Age', 'Lower_Age', 'Is_Spouse',
       'Health Indicator', 'Holding_Policy_Duration', 'Holding_Policy_Type',
       'Reco_Policy_Cat', 'Reco_Policy_Premium'],
      dtype='object')
```

Finding Unique Values

- We will use info() function for datatype, count and other parameters .
- Also we nunique() functions helps us to find out the total unique values as it can be further classifies as numeric, ordinal or categorical.

Train dataset



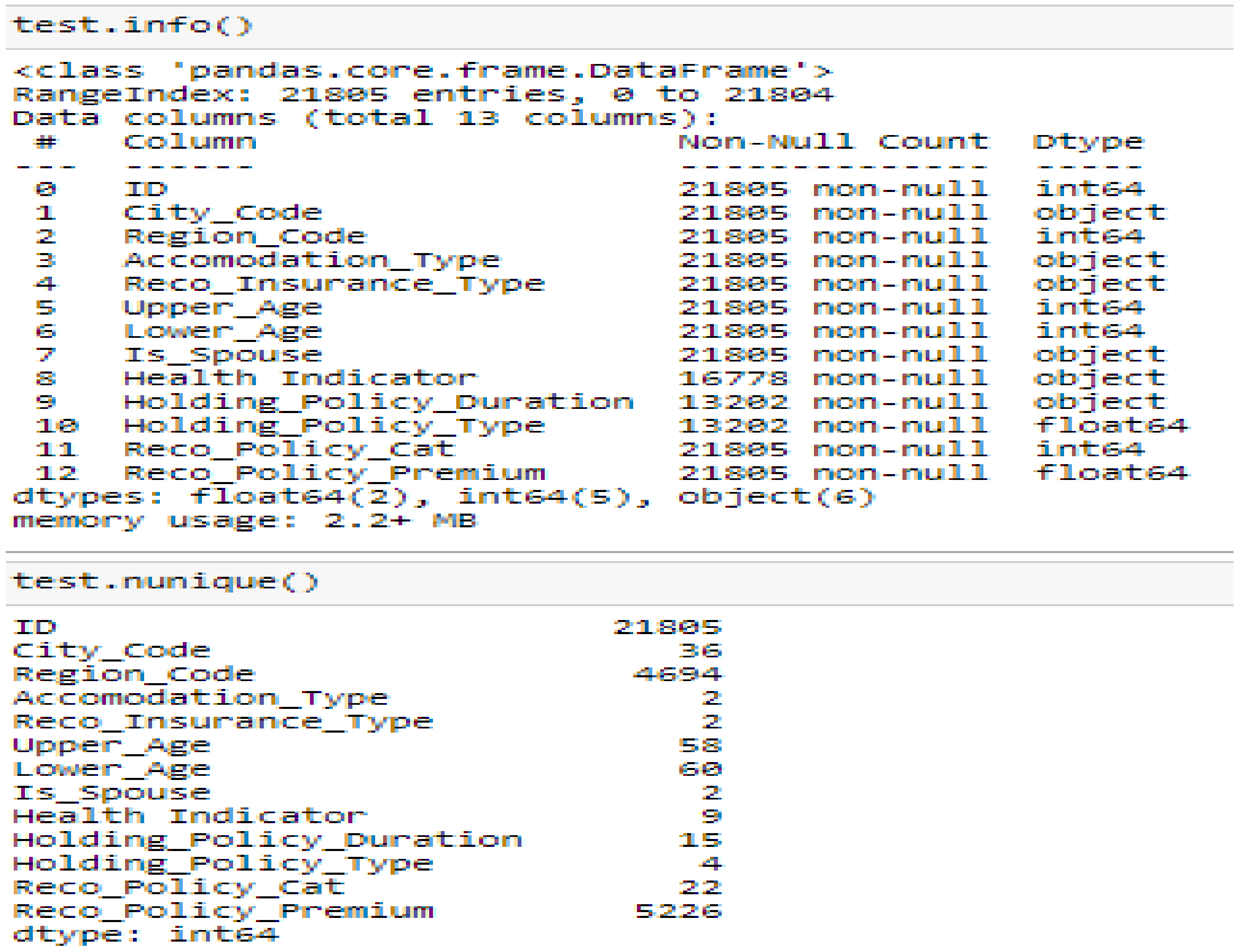
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50882 entries, 0 to 50881
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Accomodation_Type        50882 non-null  object 
 1   Reco_Insurance_Type      50882 non-null  object 
 2   Upper_Age                50882 non-null  int64  
 3   Lower_Age                50882 non-null  int64  
 4   Is_Spouse                50882 non-null  object 
 5   Health Indicator         39191 non-null  object 
 6   Holding_Policy_Duration  30631 non-null  object 
 7   Holding_Policy_Type      30631 non-null  float64
 8   Reco_Policy_Cat          50882 non-null  int64  
 9   Reco_Policy_Premium      50882 non-null  float64
 10  Response                 50882 non-null  int64  
dtypes: float64(2), int64(4), object(5)
memory usage: 4.3+ MB
```

```
train.nunique()
```

```
ID                         50882
City_Code                     36
Region_Code                 5316
Accomodation_Type              2
Reco_Insurance_Type            2
Upper_Age                     58
Lower_Age                     60
Is_Spouse                      2
Health Indicator               9
Holding_Policy_Duration       15
Holding_Policy_Type            4
Reco_Policy_Cat               22
Reco_Policy_Premium         6977
Response                       2
dtype: int64
```

Test dataset



```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21805 entries, 0 to 21804
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   ID                       21805 non-null  int64  
 1   City_Code                21805 non-null  object 
 2   Region_Code              21805 non-null  int64  
 3   Accomodation_Type        21805 non-null  object 
 4   Reco_Insurance_Type      21805 non-null  object 
 5   Upper_Age                21805 non-null  int64  
 6   Lower_Age                21805 non-null  int64  
 7   Is_Spouse                21805 non-null  object 
 8   Health Indicator         16778 non-null  object 
 9   Holding_Policy_Duration  13202 non-null  object 
 10  Holding_Policy_Type      13202 non-null  float64
 11  Reco_Policy_Cat          21805 non-null  int64  
 12  Reco_Policy_Premium      21805 non-null  float64
dtypes: float64(2), int64(5), object(6)
memory usage: 2.2+ MB
```

```
test.nunique()
```

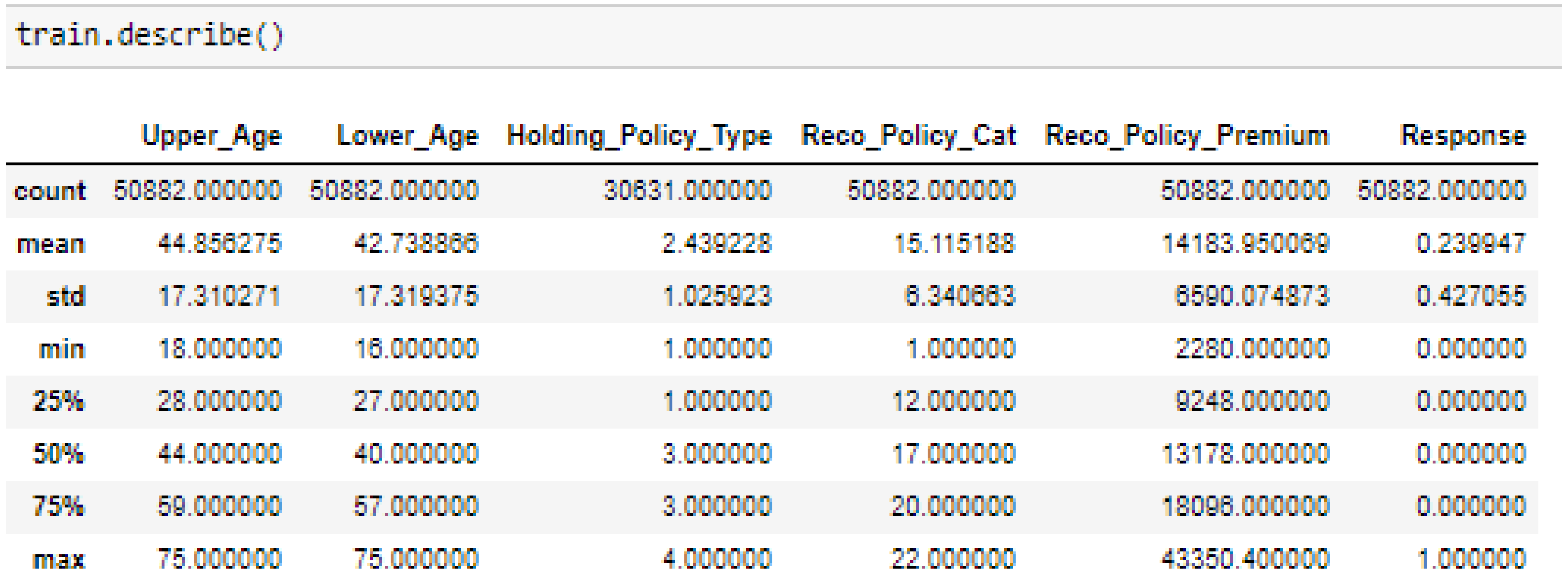
```
ID                         21805
City_Code                     36
Region_Code                 4694
Accomodation_Type              2
Reco_Insurance_Type            2
Upper_Age                     58
Lower_Age                     60
Is_Spouse                      2
Health Indicator               9
Holding_Policy_Duration       15
Holding_Policy_Type            4
Reco_Policy_Cat               22
Reco_Policy_Premium         5226
dtype: int64
```

Breakdown of variables

- As we can see from the output we can easily identify the variables and which category they can be used.

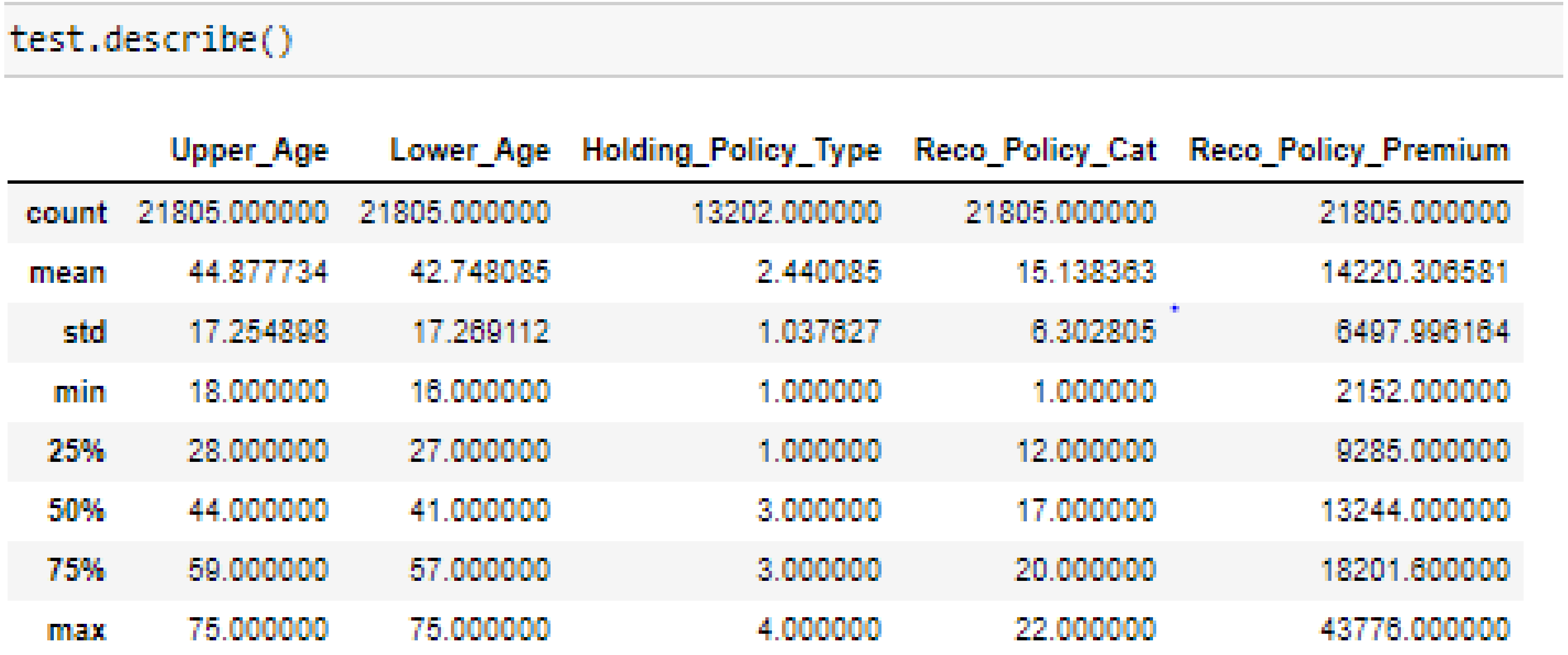
Variable	Type
Accomodation_Type	Categorical
Reco_Insurance_Type	Categorical
Upper_Age	Numeric
Lower_Age	Numeric
Is_Spouse	Categorical
Health Indicator	Categorical
Holding_Policy_Duration	Numeric
Holding_Policy_Type	Categorical
Reco_Policy_Cat	Categorical
Reco_Policy_Premium	Numeric

Summarising Statistics



```
train.describe()
```

	Upper_Age	Lower_Age	Holding_Policy_Type	Reco_Policy_Cat	Reco_Policy_Premium	Response
count	50882.000000	50882.000000	30631.000000	50882.000000	50882.000000	50882.000000
mean	44.856275	42.738866	2.439228	15.115188	14183.950069	0.239947
std	17.310271	17.319375	1.025923	6.340663	6590.074873	0.427055
min	18.000000	16.000000	1.000000	1.000000	2280.000000	0.000000
25%	28.000000	27.000000	1.000000	12.000000	9248.000000	0.000000
50%	44.000000	40.000000	3.000000	17.000000	13178.000000	0.000000
75%	59.000000	57.000000	3.000000	20.000000	18096.000000	0.000000
max	75.000000	75.000000	4.000000	22.000000	43350.400000	1.000000

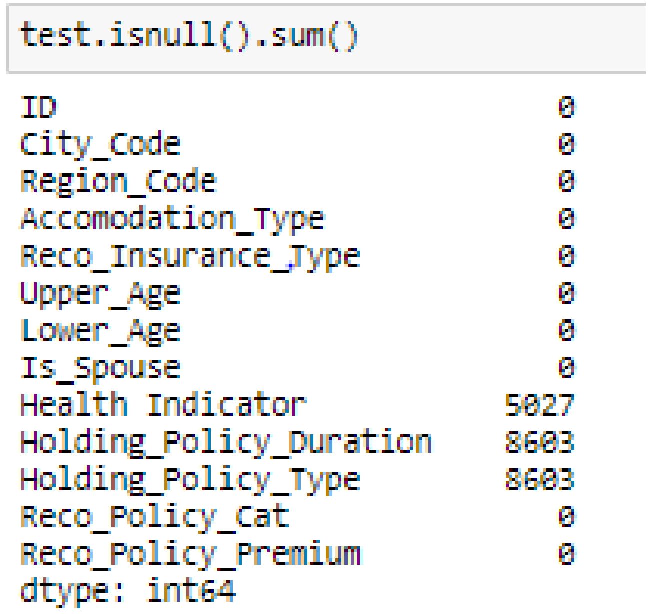


```
test.describe()
```

	Upper_Age	Lower_Age	Holding_Policy_Type	Reco_Policy_Cat	Reco_Policy_Premium
count	21805.000000	21805.000000	13202.000000	21805.000000	21805.000000
mean	44.877734	42.748085	2.440085	15.138363	14220.306581
std	17.254898	17.269112	1.037627	6.302805	6497.996164
min	18.000000	16.000000	1.000000	1.000000	2152.000000
25%	28.000000	27.000000	1.000000	12.000000	9285.000000
50%	44.000000	41.000000	3.000000	17.000000	13244.000000
75%	59.000000	57.000000	3.000000	20.000000	18201.600000
max	75.000000	75.000000	4.000000	22.000000	43776.000000

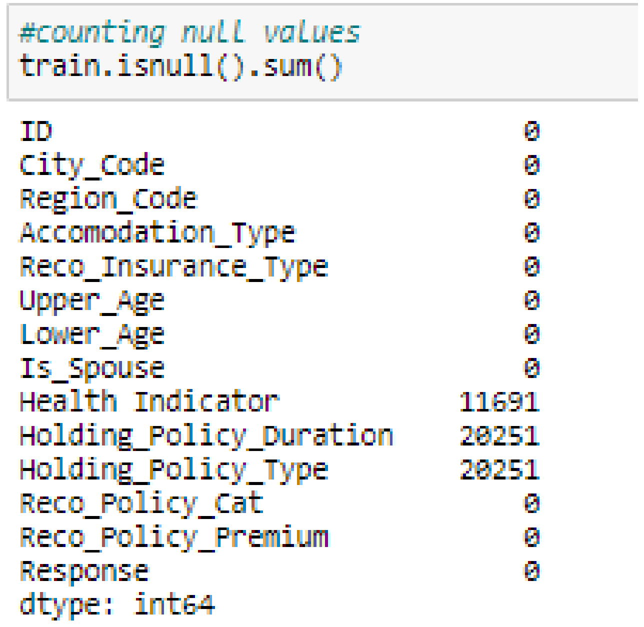
Identifying Null/missing Values

- For Train dataset
- For Test dataset



```
test.isnull().sum()
```

```
ID                          0
City_Code                   0
Region_Code                 0
Accomodation_Type           0
Reco_Insurance_Type         0
Upper_Age                   0
Lower_Age                   0
Is_Spouse                   0
Health Indicator         5027
Holding_Policy_Duration  8603
Holding_Policy_Type      8603
Reco_Policy_Cat             0
Reco_Policy_Premium         0
dtype: int64
```

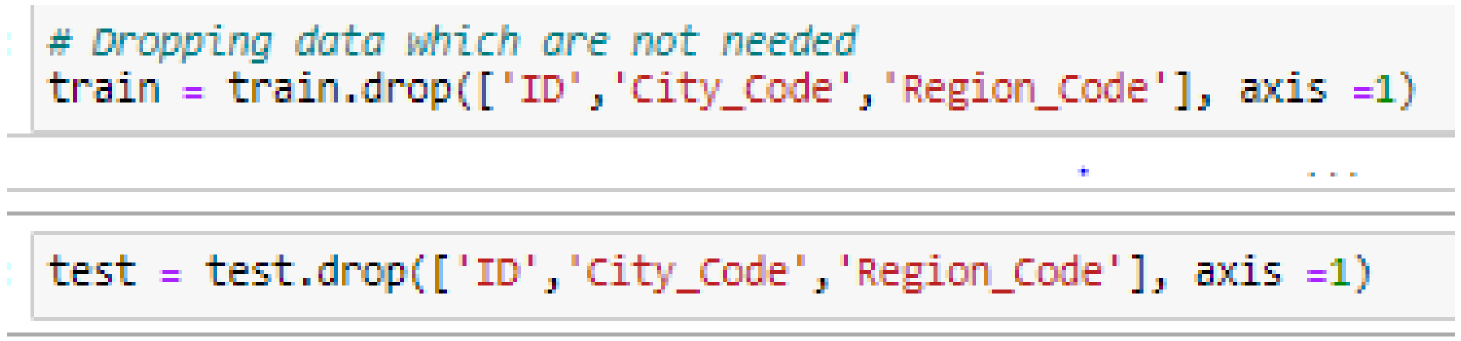


```
#counting null values
train.isnull().sum()
```

```
ID                          0
City_Code                   0
Region_Code                 0
Accomodation_Type           0
Reco_Insurance_Type         0
Upper_Age                   0
Lower_Age                   0
Is_Spouse                   0
Health Indicator        11691
Holding_Policy_Duration 20251
Holding_Policy_Type     20251
Reco_Policy_Cat             0
Reco_Policy_Premium         0
Response                    0
dtype: int64
```

Dropping Unwanted features

- We have dropped ID, City_code and Region_code from both the datasets.

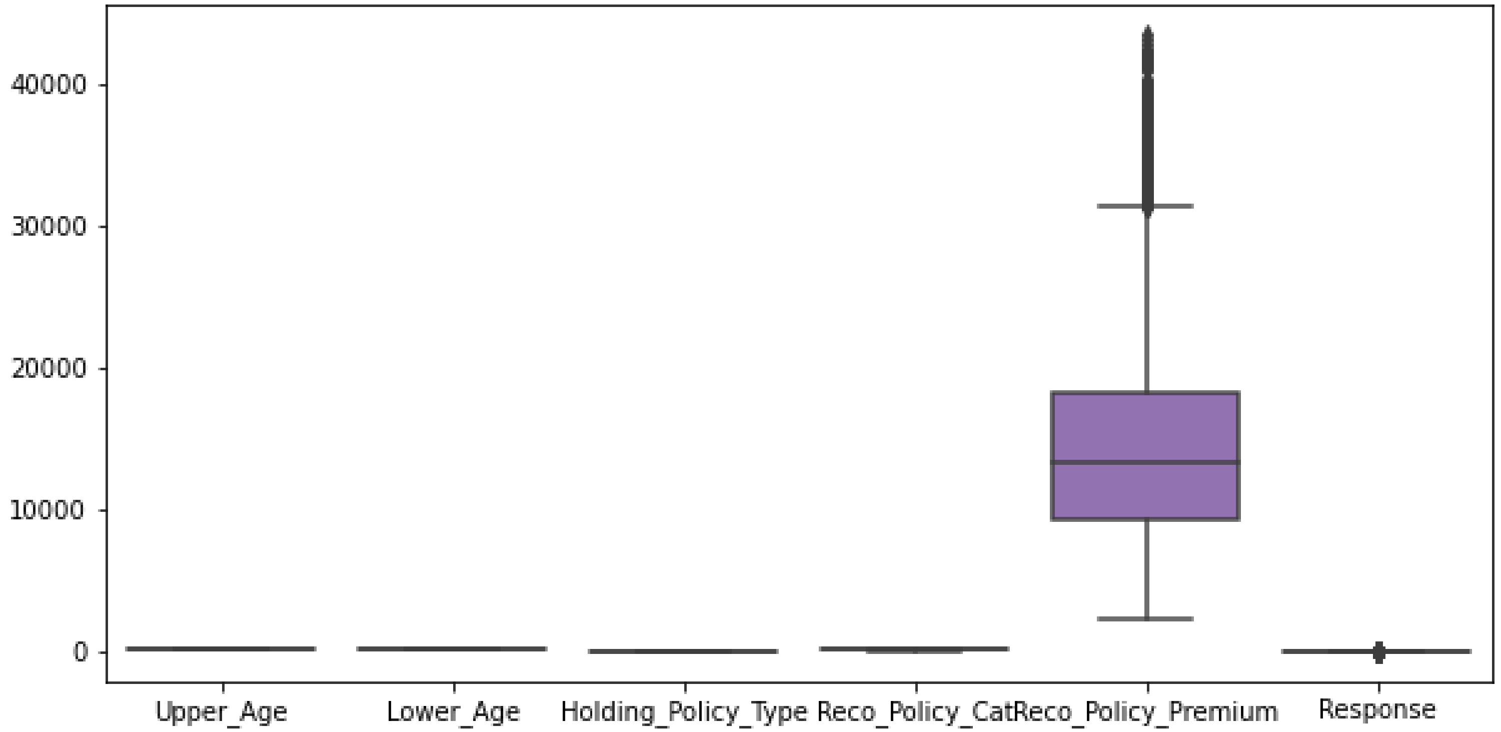


```
# Dropping data which are not needed
train = train.drop(['ID','City_Code','Region_Code'], axis =1)
```

```
test = test.drop(['ID','City_Code','Region_Code'], axis =1)
```

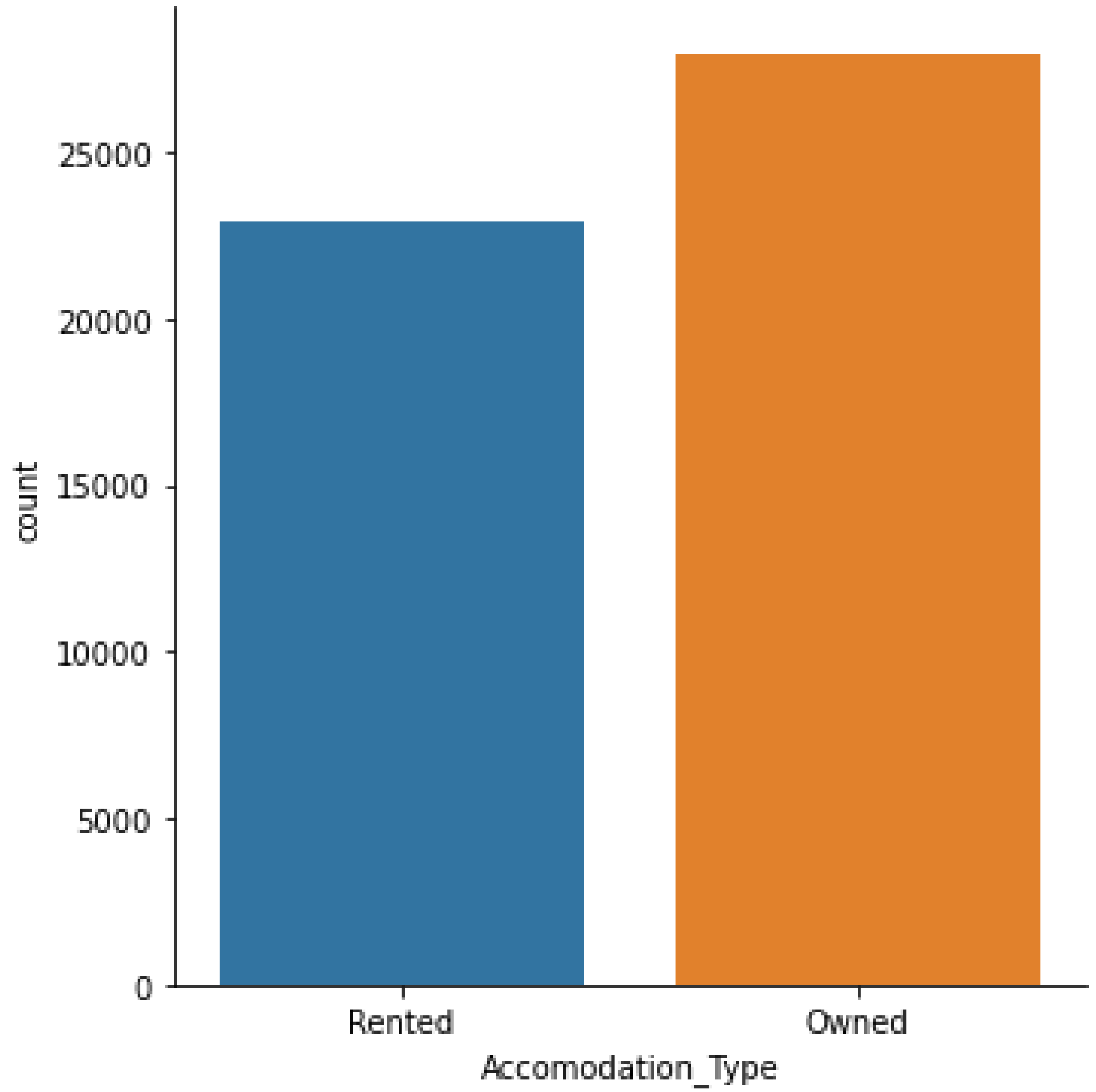
Outliers

- Reco_policy_premium is the only variable having the highest outliers as we can seen from the box plot.

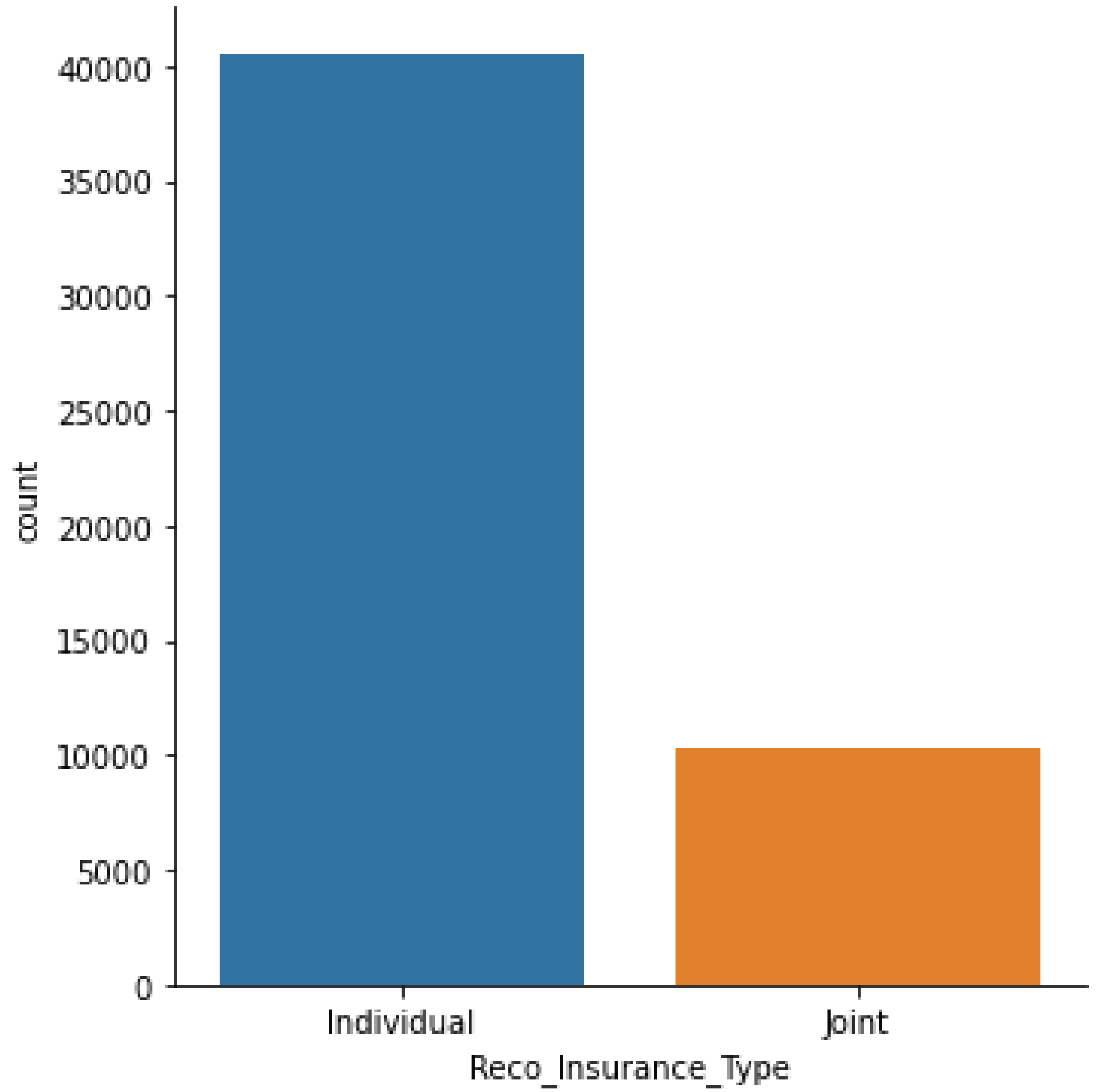


Visualization

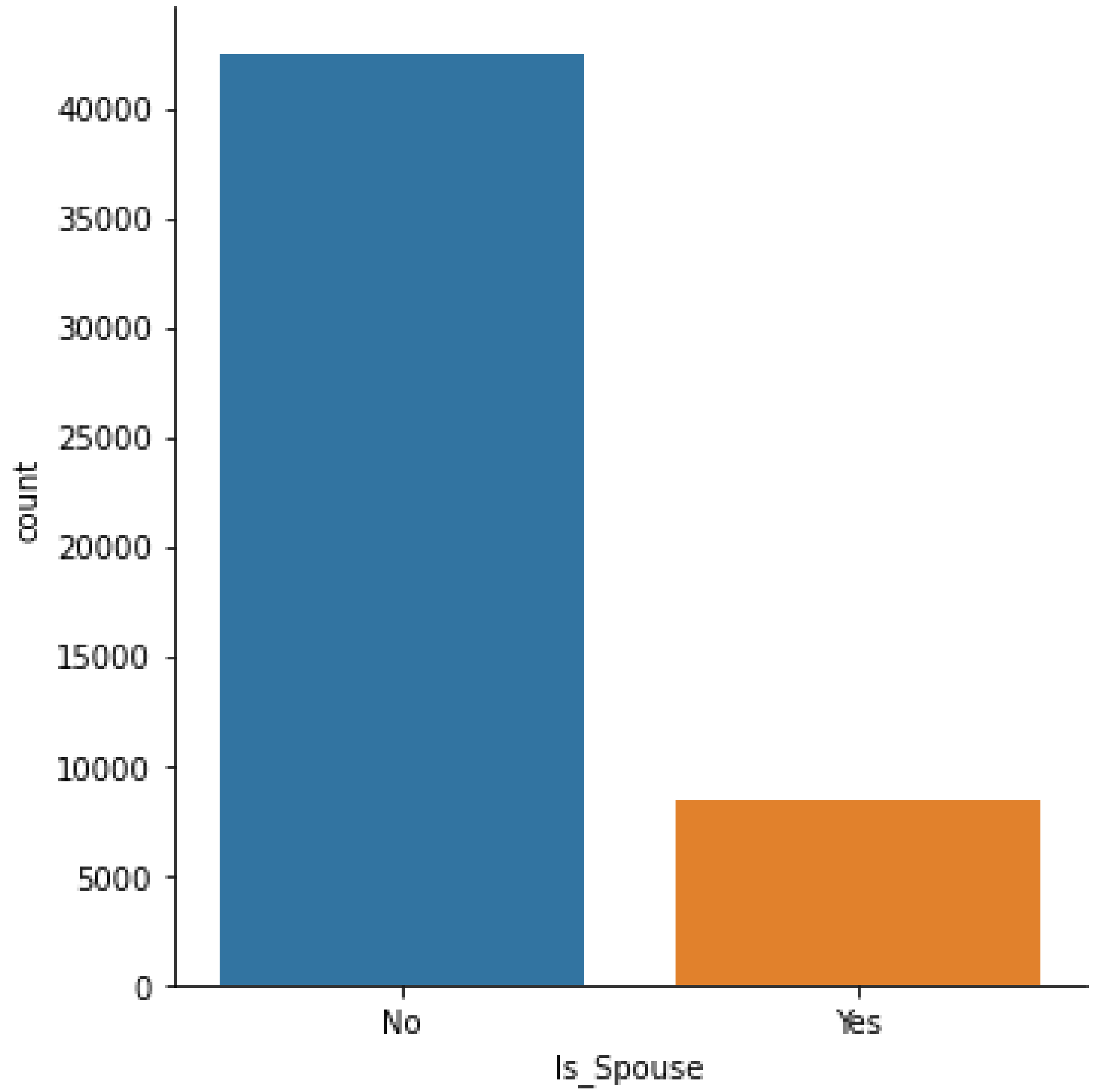
Accomodation_type plot



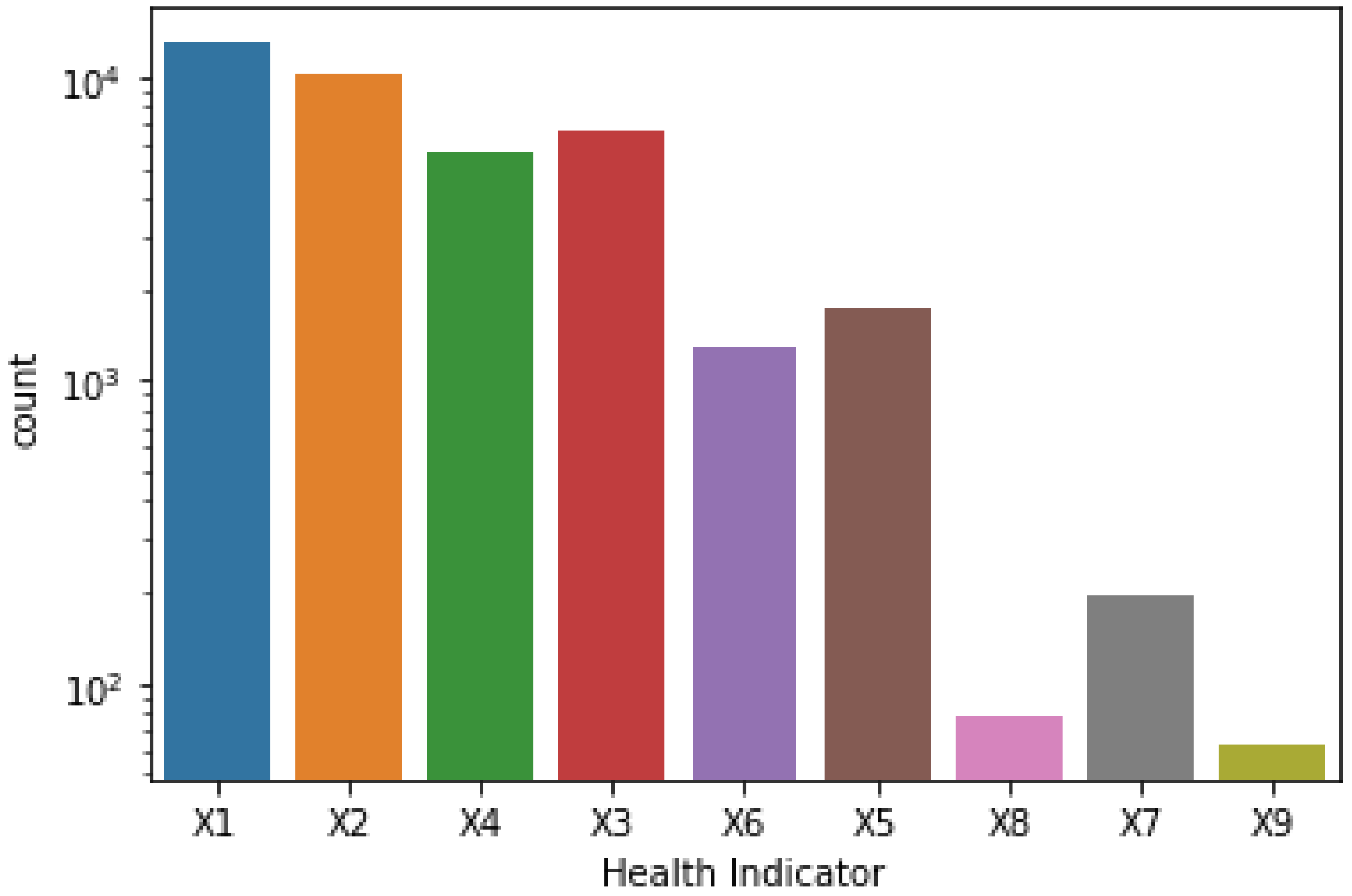
Reco_Insurance_Type plot



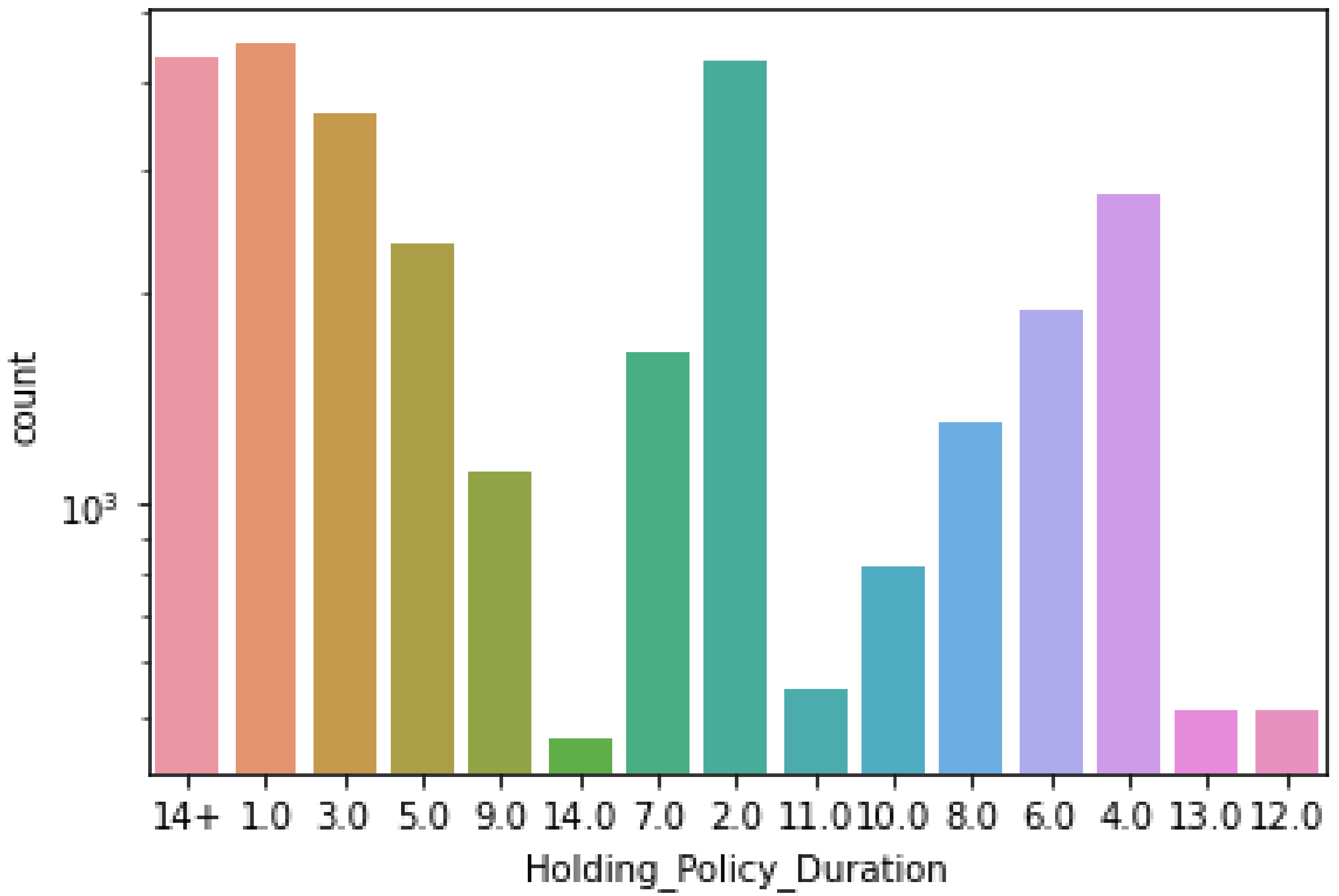
Is_Spouse plot



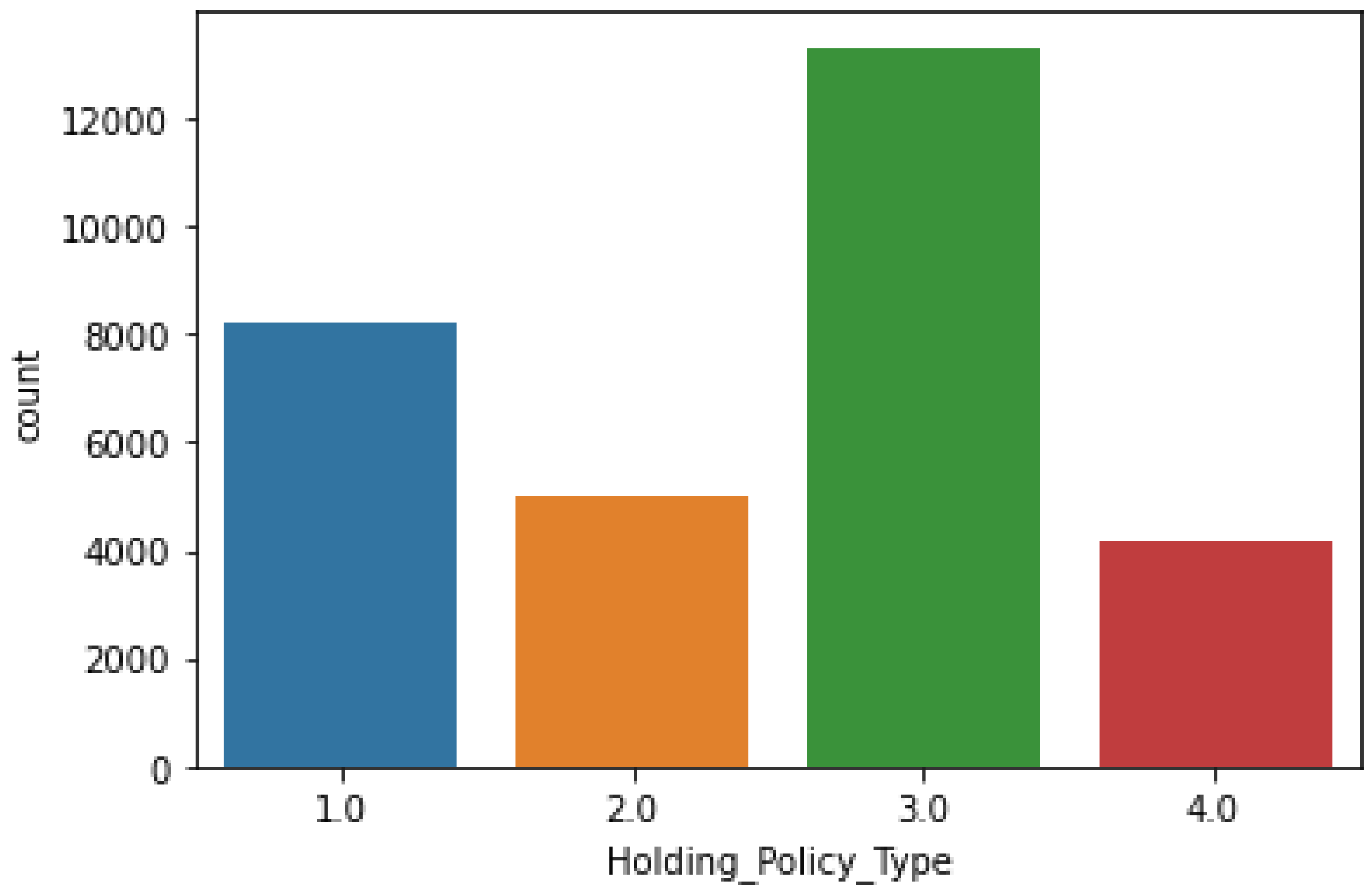
Health Indicator plot



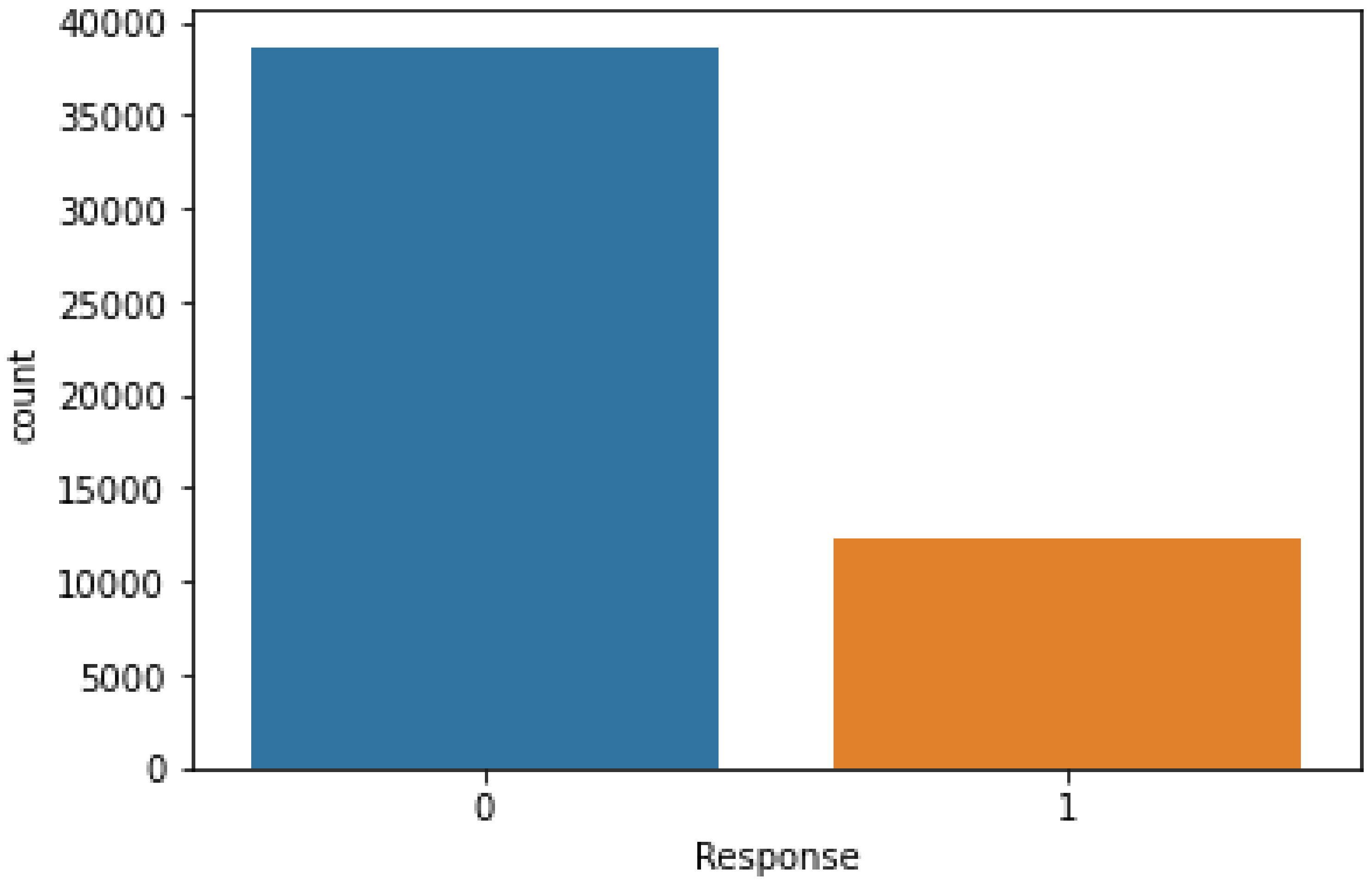
Holding_Policy_Duration plot



Holding_Policy_Type plot



Response plot



GitHub link for Jupyter notebook

- https://github.com/yashbhavsar98/Health-Insurance-Lead-Prediction

Reference links

- https://seaborn.pydata.org/examples/index.html
- https://pandas.pydata.org/docs/index.html
- https://matplotlib.org/stable/gallery/index.html
- https://learn.datacamp.com/courses/exploratory-data-analysis-in-python