Data Manipulation using Pandas and Numpy in Python

The following code is an implementation to manipulate data in Python using Pandas and Numpy Library. Also we will use Seaborn to plot heatmaps.

Importing Libraries

1. Numpy - NumPy is the fundamental package for scientific computing in Python
2. Pandas - Pandas is defined as an open-source library that provides high-performance data manipulation in Python
3. Matplotlib -
4. Seaborn - Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
```

Datasets of sales in a supermarket. There are two csv files

1. sales.csv - contains no null values
2. saltes-data.csv - contains missing at random null random data

Our first step is to load the dataset using read_csv method in python pandas library.

```
df =
pd.read_csv("https://raw.githubusercontent.com/VaibhavUpreti/data-
manipulation-python/main/sales.csv")
miss_df =
pd.read_csv("https://raw.githubusercontent.com/VaibhavUpreti/data-
manipulation-python/main/sales-data.csv")
```

Previewing dataset using head(n) or tail(n) method

These method return the first/last n rows for the object based on position. The default number of rows is set to 5. But, you can change it by writing number of rows that you want to see inside the parentheses.

Indented block

```
df.head()
```

```
    Invoice ID Branch       City Customer type  Gender  \
0  750-67-8428      A     Yangon        Member  Female
1  226-31-3081      C  Naypyitaw        Normal  Female
```

```
2  631-41-3108      A     Yangon        Normal    Male
3  123-19-1176      A     Yangon        Member    Male
4  373-73-7910      A     Yangon        Normal    Male

             Product line  Unit price  Quantity   Tax 5%     Total
Date  \
0       Health and beauty       74.69         7  26.1415  548.9715
1/5/2019
1  Electronic accessories       15.28         5   3.8200   80.2200
3/8/2019
2      Home and lifestyle       46.33         7  16.2155  340.5255
3/3/2019
3       Health and beauty       58.22         8  23.2880  489.0480
1/27/2019
4       Sports and travel       86.31         7  30.2085  634.3785
2/8/2019

    Time      Payment    cogs  gross margin percentage  gross income
Rating
0  13:08      Ewallet  522.83                 4.761905       26.1415
9.1
1  10:29         Cash   76.40                 4.761905        3.8200
9.6
2  13:23  Credit card  324.31                 4.761905       16.2155
7.4
3  20:33      Ewallet  465.76                 4.761905       23.2880
8.4
4  10:37      Ewallet  604.17                 4.761905       30.2085
5.3
```

describe() method

This method is used to get a summary of numeric values in your dataset. It calculates the mean, standard deviation, minimum value, maximum value, 1st percentile, 2nd percentile, 3rd percentile of the columns with numeric values. It also counts the number of variables in the dataset. So, we will be able to see if there are missing values in columns.

```
df.describe()
```

```
        Unit price     Quantity       Tax 5%        Total         cogs
\
count  1000.000000  1000.000000  1000.000000  1000.000000  1000.00000

mean     55.672130     5.510000    15.379369   322.966749   307.58738

std      26.494628     2.923431    11.708825   245.885335   234.17651

min      10.080000     1.000000     0.508500    10.678500    10.17000
```

```
25%      32.875000     3.000000     5.924875   124.422375   118.49750

50%      55.230000     5.000000    12.088000   253.848000   241.76000

75%      77.935000     8.000000    22.445250   471.350250   448.90500

max      99.960000    10.000000    49.650000  1042.650000   993.00000
```

```
       gross margin percentage  gross income      Rating
count              1000.000000   1000.000000  1000.00000
mean                  4.761905     15.379369     6.97270
std                   0.000000     11.708825     1.71858
min                   4.761905      0.508500     4.00000
25%                   4.761905      5.924875     5.50000
50%                   4.761905     12.088000     7.00000
75%                   4.761905     22.445250     8.50000
max                   4.761905     49.650000    10.00000
```

Columns

```
df.columns
```

```
Index(['Invoice ID', 'Branch', 'City', 'Customer type', 'Gender',
       'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Total',
'Date',
       'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross
income',
       'Rating'],
      dtype='object')
```

dtypes() methods

Returns the datastype of each column in the dataset

```
df.dtypes
```

```
Invoice ID                     object
Branch                         object
City                           object
Customer type                  object
Gender                         object
Product line                   object
Unit price                    float64
Quantity                        int64
Tax 5%                        float64
Total                         float64
Date                           object
Time                           object
```

```
Payment                        object
cogs                          float64
gross margin percentage       float64
gross income                  float64
Rating                        float64
dtype: object
```

Number of missing values

isnull().sum() - method It will return the count of null values in each column

```
df.isnull().sum()
```

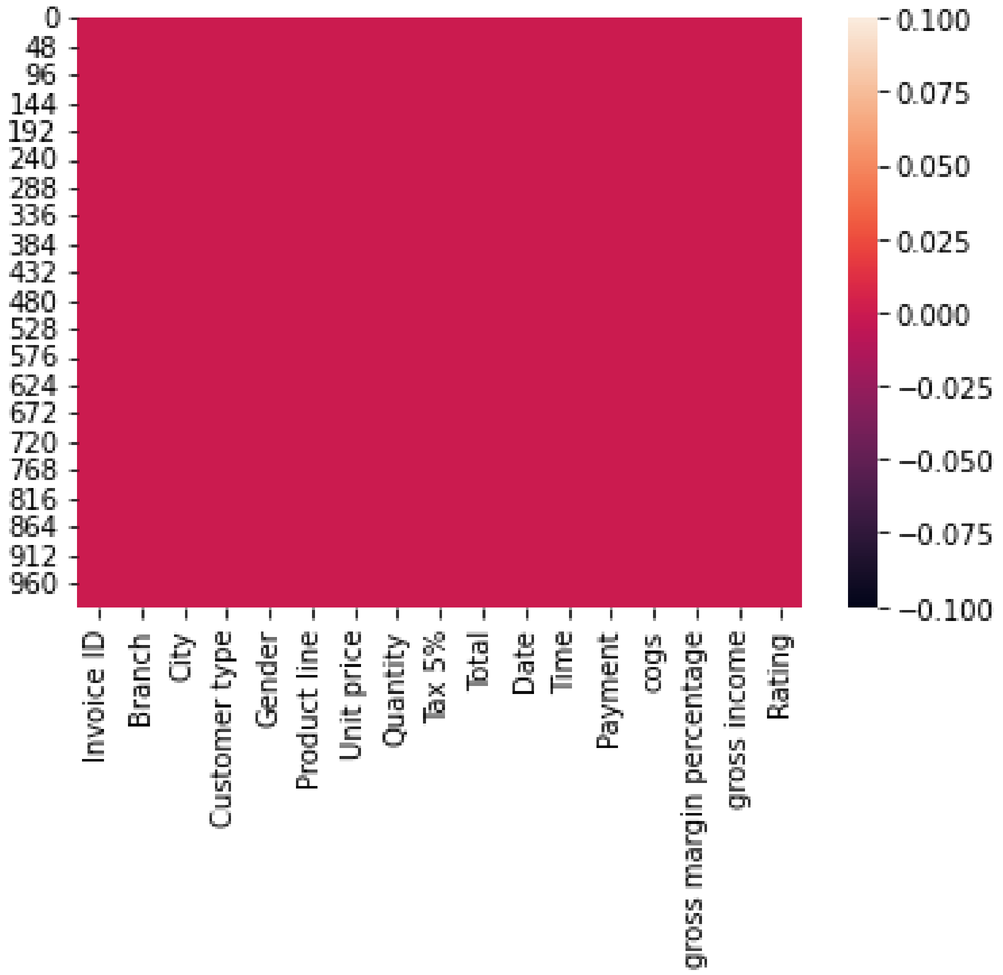
```
Invoice ID                 0
Branch                     0
City                       0
Customer type              0
Gender                     0
Product line               0
Unit price                 0
Quantity                   0
Tax 5%                     0
Total                      0
Date                       0
Time                       0
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                     0
dtype: int64
```

Heat map

The following heatmap does not show any missing value.

```
sns.heatmap(df.isnull())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0d773b5d00>
```



Assigning all the values of Gender column as Female

```
df[df['Gender']=='Female']
```

```
      Invoice ID Branch       City Customer type  Gender  \
0    750-67-8428      A     Yangon        Member  Female
1    226-31-3081      C  Naypyitaw        Normal  Female
6    355-53-5943      A     Yangon        Member  Female
7    315-22-5665      C  Naypyitaw        Normal  Female
8    665-32-9167      A     Yangon        Member  Female
..           ...    ...        ...           ...     ...
990  886-18-2897      A     Yangon        Normal  Female
991  602-16-6955      B   Mandalay        Normal  Female
994  652-49-6720      C  Naypyitaw        Member  Female
996  303-96-2227      B   Mandalay        Normal  Female
999  849-09-3807      A     Yangon        Member  Female

               Product line  Unit price  Quantity   Tax 5%      Total
\
0         Health and beauty       74.69         7  26.1415   548.9715
```

```
1    Electronic accessories      15.28         5   3.8200    80.2200

6    Electronic accessories      68.84         6  20.6520   433.6920

7        Home and lifestyle      73.56        10  36.7800   772.3800

8         Health and beauty      36.26         2   3.6260    76.1460

..                      ...        ...       ...      ...        ...

990      Food and beverages      56.56         5  14.1400   296.9400

991       Sports and travel      76.60        10  38.3000   804.3000

994  Electronic accessories      60.95         1   3.0475    63.9975

996      Home and lifestyle      97.38        10  48.6900  1022.4900

999     Fashion accessories      88.34         7  30.9190   649.2990


          Date   Time      Payment    cogs  gross margin percentage  \
0     1/5/2019  13:08      Ewallet  522.83                 4.761905
1     3/8/2019  10:29         Cash   76.40                 4.761905
6    2/25/2019  14:36      Ewallet  413.04                 4.761905
7    2/24/2019  11:38      Ewallet  735.60                 4.761905
8    1/10/2019  17:15  Credit card   72.52                 4.761905
..         ...    ...          ...     ...                      ...
990  3/22/2019  19:06  Credit card  282.80                 4.761905
991  1/24/2019  18:10      Ewallet  766.00                 4.761905
994  2/18/2019  11:40      Ewallet   60.95                 4.761905
996   3/2/2019  17:16      Ewallet  973.80                 4.761905
999  2/18/2019  13:28         Cash  618.38                 4.761905

     gross income  Rating
0         26.1415     9.1
1          3.8200     9.6
6         20.6520     5.8
7         36.7800     8.0
8          3.6260     7.2
..            ...     ...
990       14.1400     4.5
991       38.3000     6.0
994        3.0475     5.9
996       48.6900     4.4
999       30.9190     6.6

[501 rows x 17 columns]
```

Data Manipulation

1. Creating a copy of the dataframe as "data"
2. Interchanging the following columns -

- "temp" to "Branch"
- "Branch" to "City"
- "City" to "temp"

1. Displaying the data

```
data = df.copy()
data['temp'] = data['Branch']
data['Branch'] = data['City']
data['City'] = data['temp']
data
```

```
      Invoice ID     Branch City Customer type  Gender  \
0    750-67-8428     Yangon    A        Member  Female
1    226-31-3081  Naypyitaw    C        Normal  Female
2    631-41-3108     Yangon    A        Normal    Male
3    123-19-1176     Yangon    A        Member    Male
4    373-73-7910     Yangon    A        Normal    Male
..           ...        ...  ...           ...     ...
995  233-67-5758  Naypyitaw    C        Normal    Male
996  303-96-2227   Mandalay    B        Normal  Female
997  727-02-1313     Yangon    A        Member    Male
998  347-56-2442     Yangon    A        Normal    Male
999  849-09-3807     Yangon    A        Member  Female

                Product line  Unit price  Quantity   Tax 5%      Total
\
0          Health and beauty       74.69         7  26.1415   548.9715

1     Electronic accessories       15.28         5   3.8200    80.2200

2         Home and lifestyle       46.33         7  16.2155   340.5255

3          Health and beauty       58.22         8  23.2880   489.0480

4          Sports and travel       86.31         7  30.2085   634.3785

..                       ...         ...       ...      ...        ...

995        Health and beauty       40.35         1   2.0175    42.3675

996       Home and lifestyle       97.38        10  48.6900  1022.4900

997       Food and beverages       31.84         1   1.5920    33.4320

998       Home and lifestyle       65.82         1   3.2910    69.1110
```

```
999     Fashion accessories       88.34         7  30.9190   649.2990


          Date   Time      Payment    cogs  gross margin percentage  \
0     1/5/2019  13:08      Ewallet  522.83                 4.761905
1     3/8/2019  10:29         Cash   76.40                 4.761905
2     3/3/2019  13:23  Credit card  324.31                 4.761905
3    1/27/2019  20:33      Ewallet  465.76                 4.761905
4     2/8/2019  10:37      Ewallet  604.17                 4.761905
..         ...    ...          ...     ...                      ...
995  1/29/2019  13:46      Ewallet   40.35                 4.761905
996   3/2/2019  17:16      Ewallet  973.80                 4.761905
997   2/9/2019  13:22         Cash   31.84                 4.761905
998  2/22/2019  15:33         Cash   65.82                 4.761905
999  2/18/2019  13:28         Cash  618.38                 4.761905

     gross income  Rating temp
0         26.1415     9.1    A
1          3.8200     9.6    C
2         16.2155     7.4    A
3         23.2880     8.4    A
4         30.2085     5.3    A
..            ...     ...  ...
995        2.0175     6.2    C
996       48.6900     4.4    B
997        1.5920     7.7    A
998        3.2910     4.1    A
999       30.9190     6.6    A

[1000 rows x 18 columns]
```

Dropping the Column "Total"

```
data.drop('Total',axis = 1,inplace = True)
```

Creating a new Column

Creating a new column "total_price" with increased 5% tax on every product

```
data['total_price'] = data['Quantity']*data['Unit price'] + data['Tax
5%'] + data['Tax 5%']
data
```

```
      Invoice ID     Branch City Customer type  Gender  \
0    750-67-8428     Yangon    A        Member  Female
1    226-31-3081  Naypyitaw    C        Normal  Female
2    631-41-3108     Yangon    A        Normal    Male
3    123-19-1176     Yangon    A        Member    Male
```

```
4    373-73-7910     Yangon    A        Normal    Male
..           ...        ...  ...           ...     ...
995  233-67-5758  Naypyitaw    C        Normal    Male
996  303-96-2227   Mandalay    B        Normal  Female
997  727-02-1313     Yangon    A        Member    Male
998  347-56-2442     Yangon    A        Normal    Male
999  849-09-3807     Yangon    A        Member  Female

               Product line  Unit price  Quantity   Tax 5%       Date
Time  \
0         Health and beauty       74.69         7  26.1415   1/5/2019
13:08
1    Electronic accessories       15.28         5   3.8200   3/8/2019
10:29
2        Home and lifestyle       46.33         7  16.2155   3/3/2019
13:23
3         Health and beauty       58.22         8  23.2880  1/27/2019
20:33
4         Sports and travel       86.31         7  30.2085   2/8/2019
10:37
..                      ...         ...       ...      ...        ...
...
995       Health and beauty       40.35         1   2.0175  1/29/2019
13:46
996      Home and lifestyle       97.38        10  48.6900   3/2/2019
17:16
997      Food and beverages       31.84         1   1.5920   2/9/2019
13:22
998      Home and lifestyle       65.82         1   3.2910  2/22/2019
15:33
999     Fashion accessories       88.34         7  30.9190  2/18/2019
13:28

         Payment    cogs  gross margin percentage  gross income
Rating temp  \
0        Ewallet  522.83                 4.761905       26.1415
9.1    A
1           Cash   76.40                 4.761905        3.8200
9.6    C
2    Credit card  324.31                 4.761905       16.2155
7.4    A
3        Ewallet  465.76                 4.761905       23.2880
8.4    A
4        Ewallet  604.17                 4.761905       30.2085
5.3    A
..           ...     ...                      ...           ...     ..
.  ...
995      Ewallet   40.35                 4.761905        2.0175
6.2    C
996      Ewallet  973.80                 4.761905       48.6900
```

```
4.4    B
997         Cash   31.84                 4.761905        1.5920
7.7    A
998         Cash   65.82                 4.761905        3.2910
4.1    A
999         Cash  618.38                 4.761905       30.9190
6.6    A

     total_price
0        575.113
1         84.040
2        356.741
3        512.336
4        664.587
..           ...
995       44.385
996     1071.180
997       35.024
998       72.402
999      680.218

[1000 rows x 18 columns]
```

Checking for missising values in "data"

```
data.isnull().sum()

Invoice ID                 0
Branch                     0
City                       0
Customer type              0
Gender                     0
Product line               0
Unit price                 0
Quantity                   0
Tax 5%                     0
Date                       0
Time                       0
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                     0
temp                       0
total_price                0
dtype: int64
```

Checking for missing data in miss_df

We can see there are three fields that contain missing values and number of values missing.

```
miss_df.isnull().sum()
```

```
Invoice ID                  0
Branch                      0
City                       10
Customer type              10
Gender                     23
Product line                0
Unit price                  0
Quantity                    0
Tax 5%                      0
Total                       0
Date                        0
Time                        0
Payment                     0
cogs                        0
gross margin percentage     0
gross income                0
Rating                     31
dtype: int64
```

```
miss_df.columns
```

```
Index(['Invoice ID', 'Branch', 'City', 'Customer type', 'Gender',
       'Product line', 'Unit price', 'Quantity', 'Tax 5%', 'Total',
'Date',
       'Time', 'Payment', 'cogs', 'gross margin percentage', 'gross
income',
       'Rating'],
      dtype='object')
```

Checking missing values in Gender Column

```
miss_df['Gender'].isnull().sum()
```

```
23
```

There are 23 null values in the column 'Gender'. Let's fill these columns with gender as Male.

Filling all the missing values in Gender Column with Male

```
miss_df['Gender'] = miss_df['Gender'].fillna("Male")
miss_df.head()
```

```
    Invoice ID Branch       City Customer type  Gender  \
0  750-67-8428      A     Yangon        Member    Male
1  226-31-3081      C  Naypyitaw        Normal  Female
2  631-41-3108      A     Yangon        Normal    Male
3  123-19-1176      A     Yangon        Member    Male
4  373-73-7910      A     Yangon        Normal    Male

             Product line  Unit price  Quantity   Tax 5%     Total
Date  \
0       Health and beauty       74.69         7  26.1415  548.9715
1/5/2019
1  Electronic accessories       15.28         5   3.8200   80.2200
3/8/2019
2      Home and lifestyle       46.33         7  16.2155  340.5255
3/3/2019
3       Health and beauty       58.22         8  23.2880  489.0480
1/27/2019
4       Sports and travel       86.31         7  30.2085  634.3785
2/8/2019

    Time      Payment    cogs  gross margin percentage  gross income
Rating
0  13:08      Ewallet  522.83                 4.761905       26.1415
9.1
1  10:29         Cash   76.40                 4.761905        3.8200
9.6
2  13:23  Credit card  324.31                 4.761905       16.2155
7.4
3  20:33      Ewallet  465.76                 4.761905       23.2880
8.4
4  10:37      Ewallet  604.17                 4.761905       30.2085
5.3
```

No empty values in Gender Column

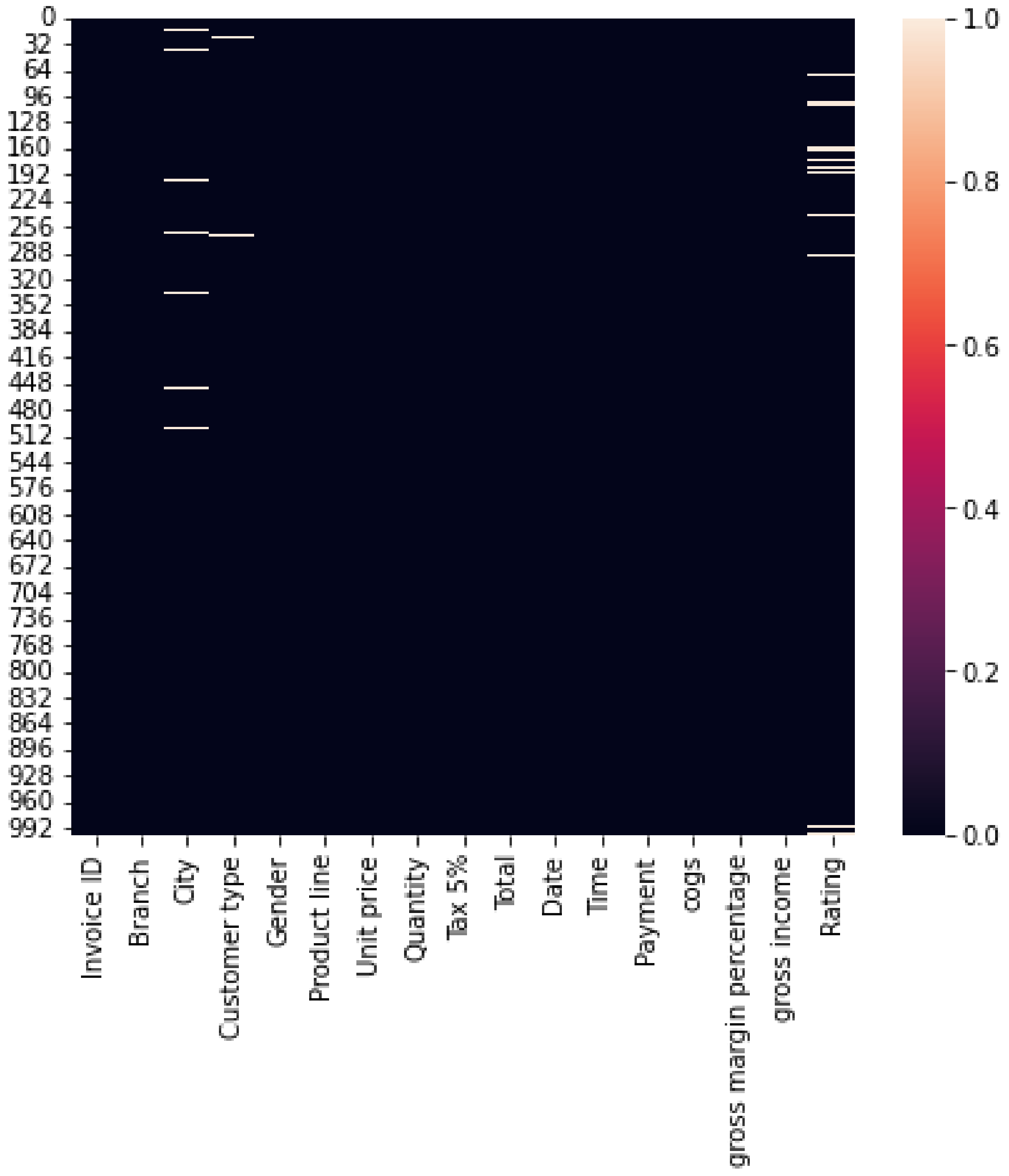
```
miss_df['Gender'].isnull().sum()
```

```
0
```

Plotting heatmap for miss_df

```
corr_matrix = miss_df[['Branch', 'City', 'Gender', 'Unit price',
'Date','Time']].corr()
plt.figure(figsize=(7, 6))
#sns.heatmap(data = corr_matrix,cmap='BrBG',
annot=True,linewidths=0.2)
sns.heatmap(miss_df.isnull())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0d771cda00>
```



```
miss_df.isnull().sum()

Invoice ID                  0
Branch                      0
City                       10
Customer type              10
Gender                      0
Product line                0
Unit price                  0
Quantity                    0
Tax 5%                      0
Total                       0
```

```
Date                       0
Time                       0
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                    31
dtype: int64
```

Numpy

We will use Numpy to fill all the NULL values in rating column replacing them with the median rating.

np.isnan() in numpy

NaN - Not a Number np. isnan() returns a Boolean array. It returns True if an element is NaN . It returns False otherwise.

```
NaN_indexes = miss_df['Rating'].isnull().index
for i in NaN_indexes:
  avg_rating = miss_df['Rating'].median()
  if not np.isnan(avg_rating):
    miss_df['Rating'].iloc[i] = avg_rating
  else:
     miss_df['Rating'].iloc[i] = miss_df['Rating'].median()
```

The following block of code interates all NaN indexes to check if they are missing ... If it a NaN then we fill the NULL value with the median rating of all the ratings.

Result after filling data

We can see now there are no missing entries in ratings column

```
miss_df.isnull().sum()
```

```
Invoice ID                 0
Branch                     0
City                      10
Customer type             10
Gender                     0
Product line               0
Unit price                 0
Quantity                   0
Tax 5%                     0
Total                      0
Date                       0
Time                       0
```

```
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                     0
dtype: int64
```

miss_df

```
      Invoice ID Branch       City Customer type  Gender  \
0    750-67-8428      A     Yangon        Member    Male
1    226-31-3081      C  Naypyitaw        Normal  Female
2    631-41-3108      A     Yangon        Normal    Male
3    123-19-1176      A     Yangon        Member    Male
4    373-73-7910      A     Yangon        Normal    Male
..           ...    ...        ...           ...     ...
995  233-67-5758      C  Naypyitaw        Normal    Male
996  303-96-2227      B   Mandalay        Normal  Female
997  727-02-1313      A     Yangon        Member    Male
998  347-56-2442      A     Yangon        Normal    Male
999  849-09-3807      A     Yangon        Member  Female

               Product line  Unit price  Quantity   Tax 5%      Total
\
0         Health and beauty       74.69         7  26.1415   548.9715

1    Electronic accessories       15.28         5   3.8200    80.2200

2        Home and lifestyle       46.33         7  16.2155   340.5255

3         Health and beauty       58.22         8  23.2880   489.0480

4         Sports and travel       86.31         7  30.2085   634.3785

..                      ...         ...       ...      ...        ...

995       Health and beauty       40.35         1   2.0175    42.3675

996      Home and lifestyle       97.38        10  48.6900  1022.4900

997      Food and beverages       31.84         1   1.5920    33.4320

998      Home and lifestyle       65.82         1   3.2910    69.1110

999     Fashion accessories       88.34         7  30.9190   649.2990


          Date   Time      Payment    cogs  gross margin percentage  \
0     1/5/2019  13:08      Ewallet  522.83                 4.761905
```

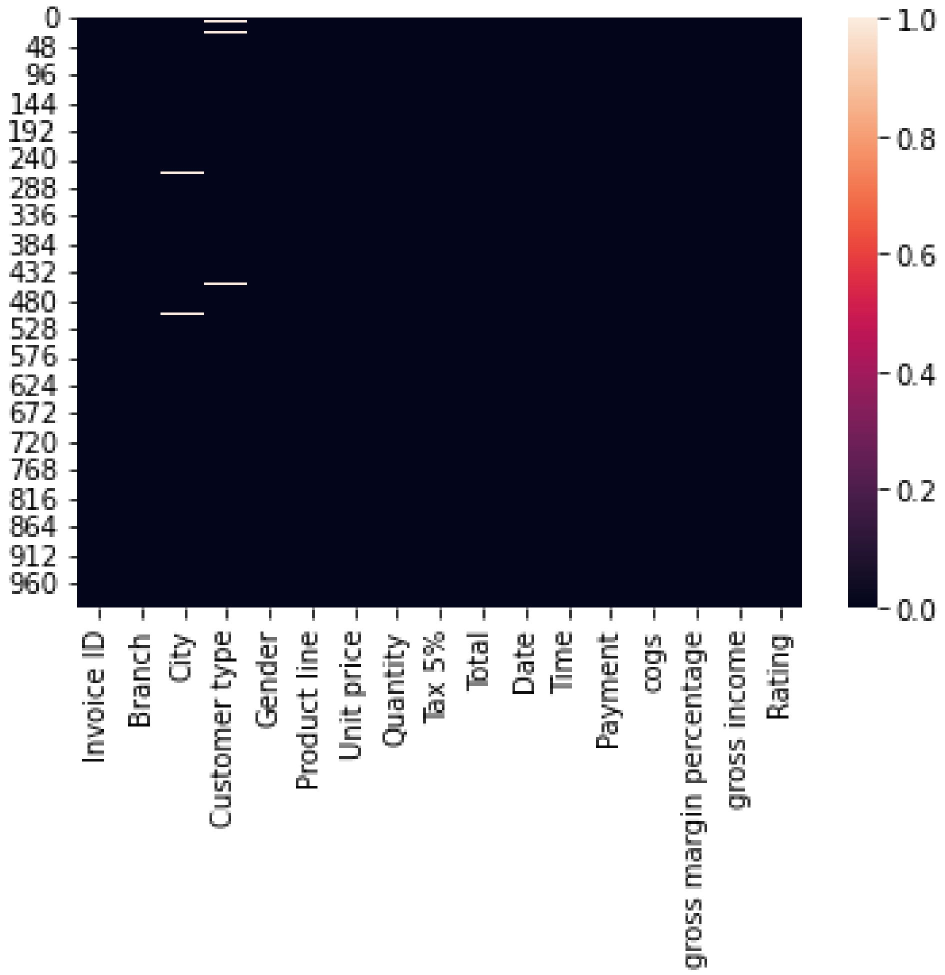
```
1     3/8/2019  10:29         Cash   76.40                 4.761905
2     3/3/2019  13:23  Credit card  324.31                 4.761905
3    1/27/2019  20:33      Ewallet  465.76                 4.761905
4     2/8/2019  10:37      Ewallet  604.17                 4.761905
..         ...    ...          ...     ...                      ...
995  1/29/2019  13:46      Ewallet   40.35                 4.761905
996   3/2/2019  17:16      Ewallet  973.80                 4.761905
997   2/9/2019  13:22         Cash   31.84                 4.761905
998  2/22/2019  15:33         Cash   65.82                 4.761905
999  2/18/2019  13:28         Cash  618.38                 4.761905

     gross income  Rating
0         26.1415     7.0
1          3.8200     7.0
2         16.2155     7.0
3         23.2880     7.0
4         30.2085     7.0
..            ...     ...
995        2.0175     7.0
996       48.6900     7.0
997        1.5920     7.0
998        3.2910     7.0
999       30.9190     7.0

[1000 rows x 17 columns]
```

```
sns.heatmap(miss_df.isnull())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0d7719d3a0>
```


0
48
96
144
192
240
288
336
384
432
480
528
576
624
672
720
768
816
864
912
960
Invoice ID
Branch
City
Customer type
Gender
Product line
Unit price
Quantity
Tax 5%
Total
Date
Time
Payment
cogs
gross margin percentage
gross income
Rating
1.0
0.8
0.6
0.4
0.2
0.0