09/2022

EDA with python

A REPORT BY
VO AI VAN

PART OF PROJECT
EDA PROJECT

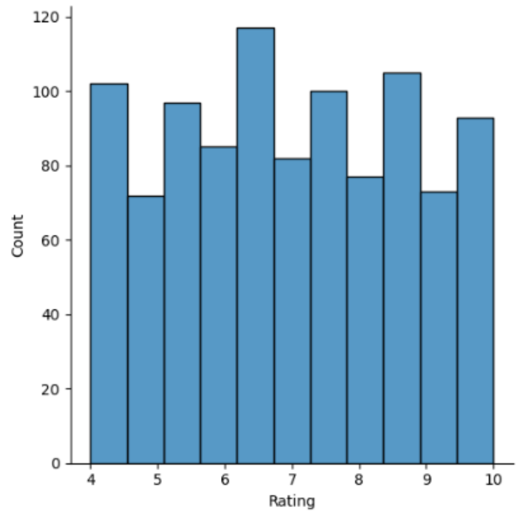


Figure 1: Plot on distribution of customer rating:
sns.distplot(df['Rating'])
plt.show()

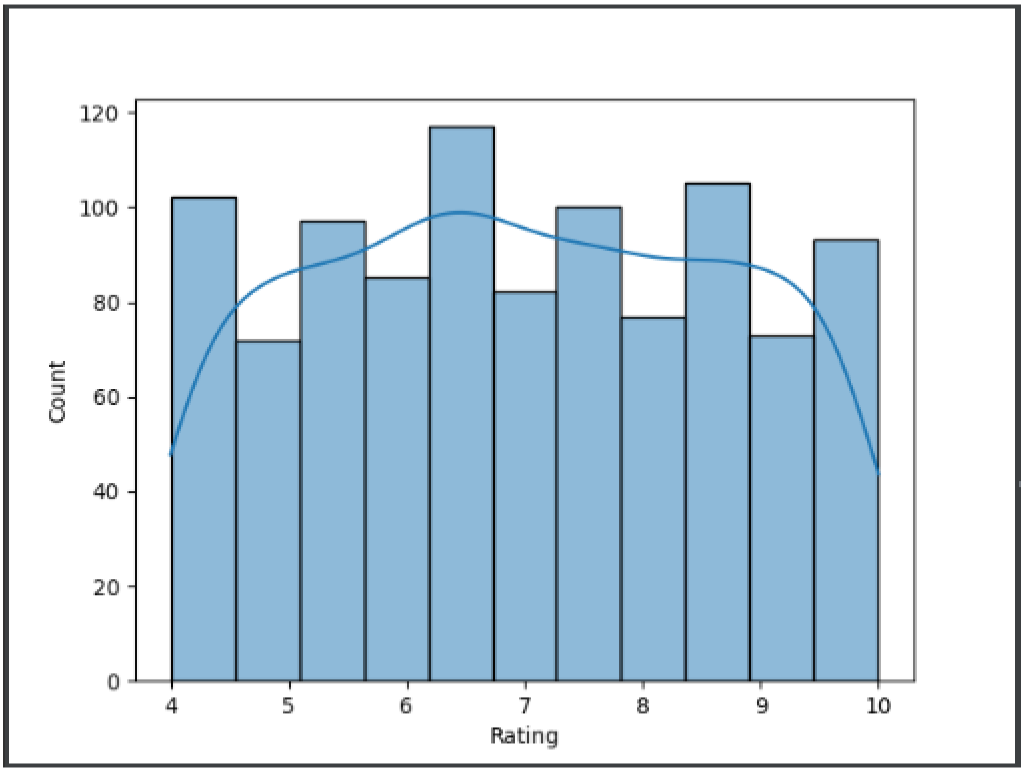


Figure 2: After adding a kernel density estimate to smooth the histogram and provide complementary information about the shape of the distribution. This distribution looks like a uniform distribution which means none of the rating number particularly spike out:
sns.histplot (df['Rating'],kde=True)
plt.show()

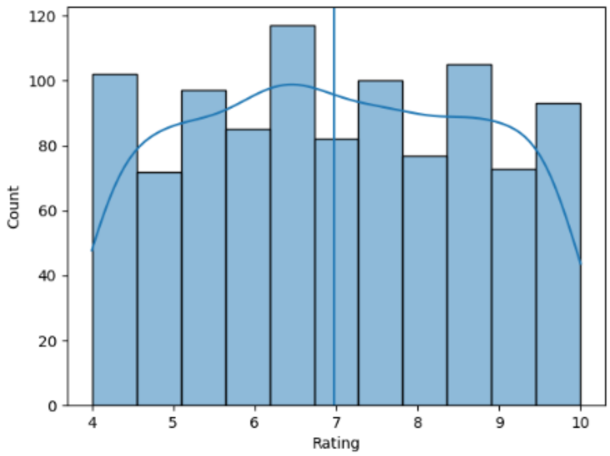


Figure 3: Mean rating in the graph:
sns.histplot(df[‘Rating’], kde= True)
plt.axvline(x=np.mean(df[‘Rating’]))
plt.show()

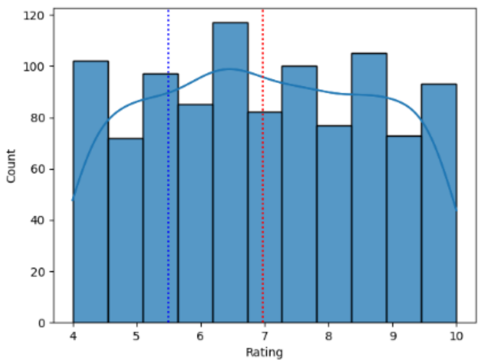


Figure 4: Percentile plot the line can be formatted in different support such as '-', '--', '-.', ':', 'None', ' ', '', 'solid', 'dashed', 'dashdot', 'dotted’ (25 percentile):
sns.histplot(df[‘Rating’],kde= True)
plt.axvline(x=np.mean(df[‘Rating’]),c=’red’,ls=’:’)
plt.axvline(x=np.percentile(df[‘Rating’],25),c=’blue’,ls=’:’)
plt.show()

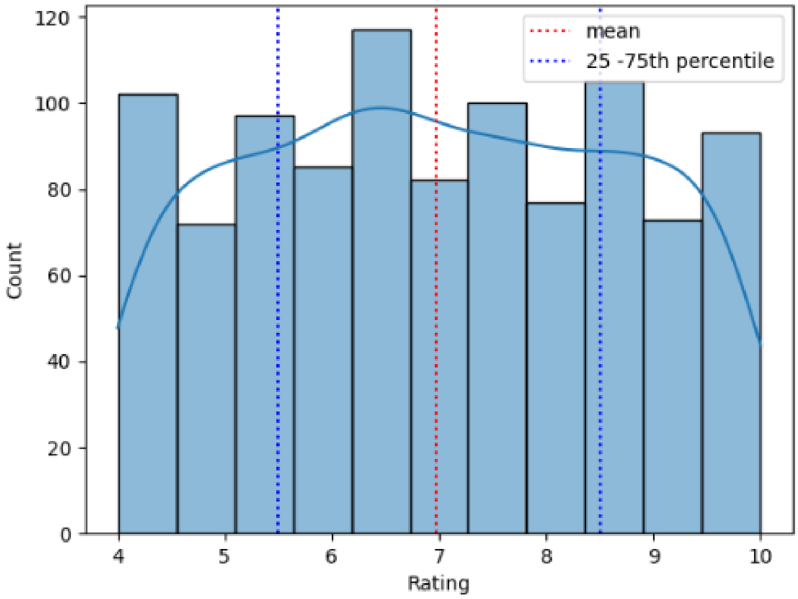


Figure 5: same function as *figure 4* but now we can also add label to the graph to indicate the line better:

```
sns.histplot(df['Rating'], kde= True)
plt.axvline(x=np.mean(df['Rating']),c='red',ls=':',label='mean')
plt.axvline(x=np.percentile(df['Rating'],25),c='blue',ls=':',label='25 -75th percentile')
plt.axvline(x=np.percentile(df['Rating'],75),c='blue',ls=':')
plt.legend()
plt.show()
```

Conclusion: from the graph, we can conclude that there is no skewed in left and right direction

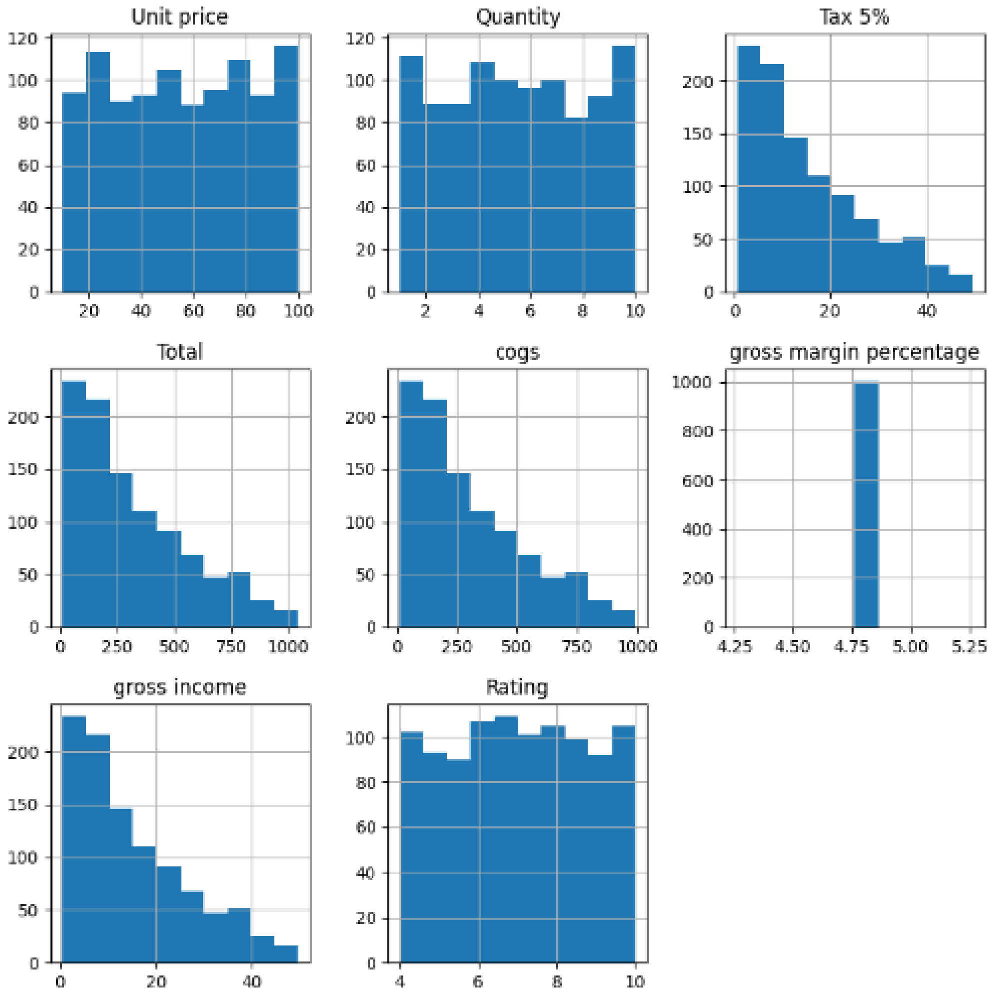


Figure 6: getting all the plot available for the dataset with figure size to avoid messed up plots:
df.hist(figsize=(10,10))
plt.show()

Conclusion:
uniform distributed: unit price, quantity, rating
tax: most of the tax collected fall between 0 and 20, some are at 40
constant value: gross margin percentage
fall precisely the same: total + gross income + cogs

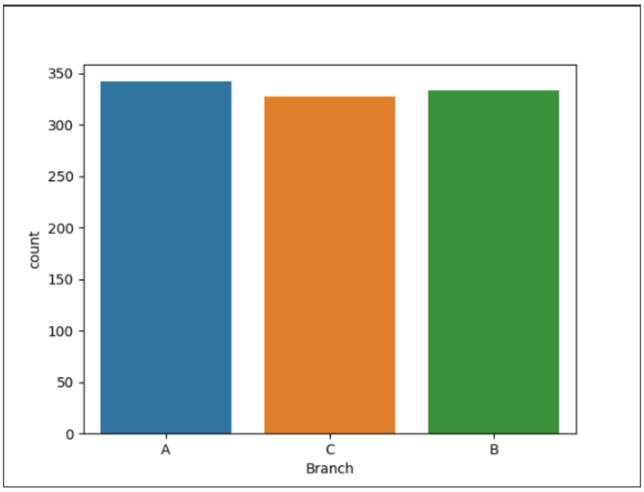


Figure 7: plot to illustrate number of users for each branches:
sns.countplot(df['Branch'])
plt.show()
df['Branch'].value_counts() # show the precise number

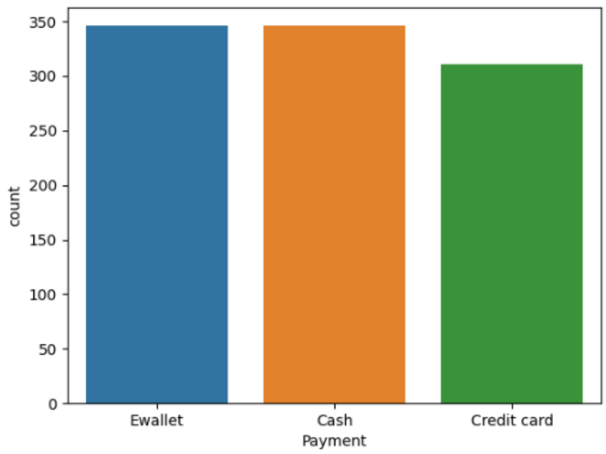


Figure 8: plot number of users for each type of payment:
sns.countplot(df['Payment'])
plt.show()
df['Payment'].value_counts()

```
>>> df['Payment'].value_counts()
Ewallet        346
Cash           346
Credit card    311
Name: Payment, dtype: int64
```

Figure 9: show precise number for number of users on different type of payment

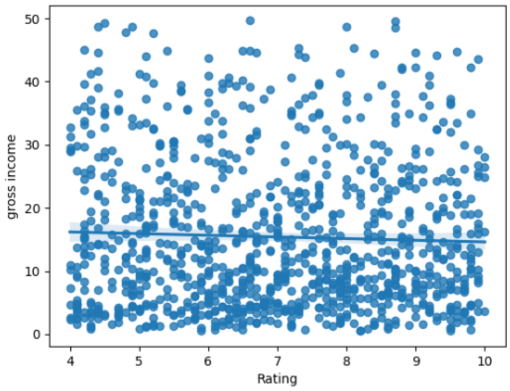
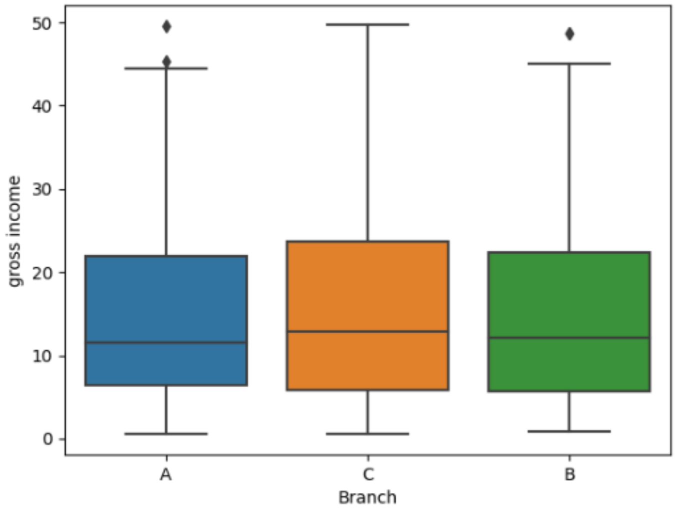


Figure 10: to know the relationship between gross margin and customer rating we use the following code. From that we can see the trend line seems to be very flat → there is no relation between them:

```
sns.scatterplot(df['Rating'],df['gross income'])
plt.show()
# to show the trend line of this relationship:
sns.regplot(df['Rating'],df['gross income'])
sns.relplot
```

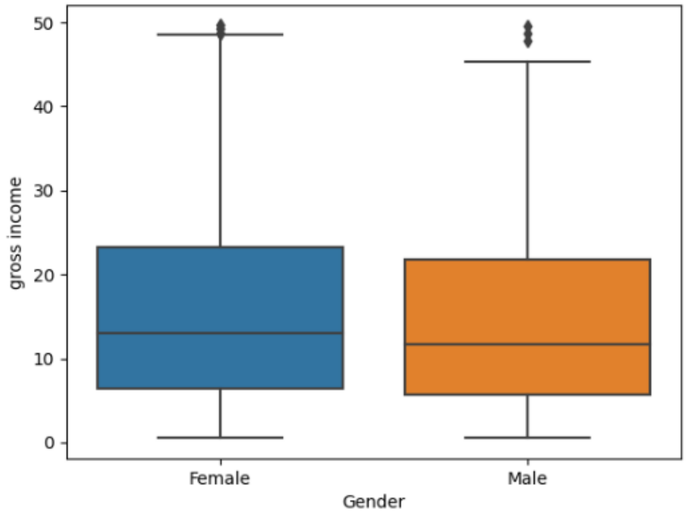
Similarly: we can also learn about the relationship between:
branches and income:



sns.boxplot(x=df['Branch'],y=df['gross income'])
plt.show()

→ Looking at branch A, for instance, gross income is around 10 .And not much of variation between branches and gross income

gender and gross income:



sns.boxplot(x=df['Gender'],y=df['gross income'])
plt.show()

→ ON AVERAGE, both gender spend approximately the same
→ at 50th percentile women spend more than men

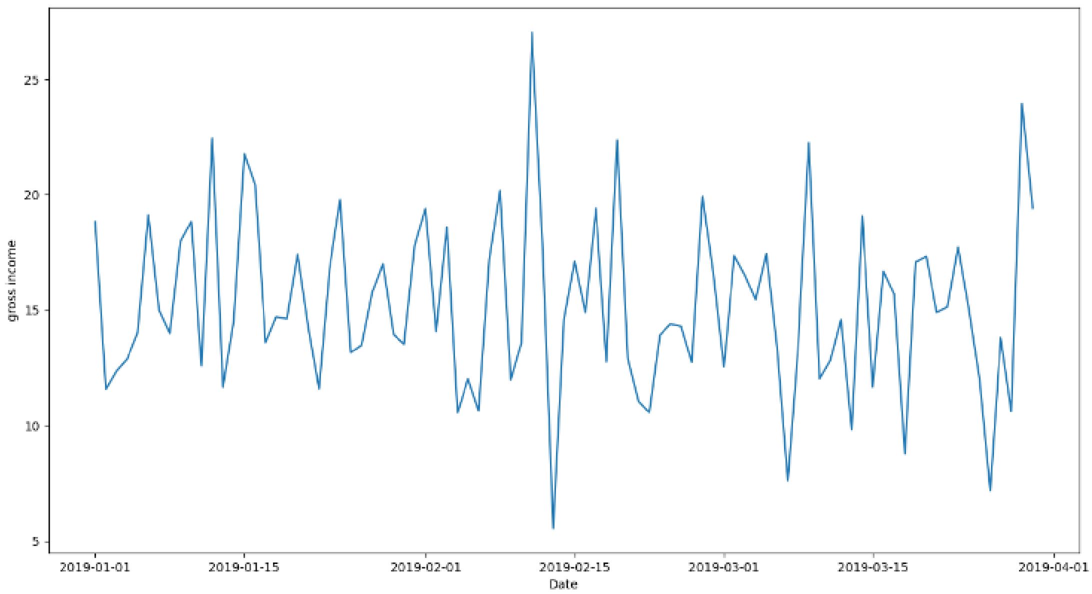


Figure 11: the graph shows lineplot for gross income each day:

```
plt.figure(figsize=(15,8))
sns.lineplot(x=df.groupby(df.index).mean().index,
    y=df.groupby(df.index).mean()['gross income'])
plt.show()
```

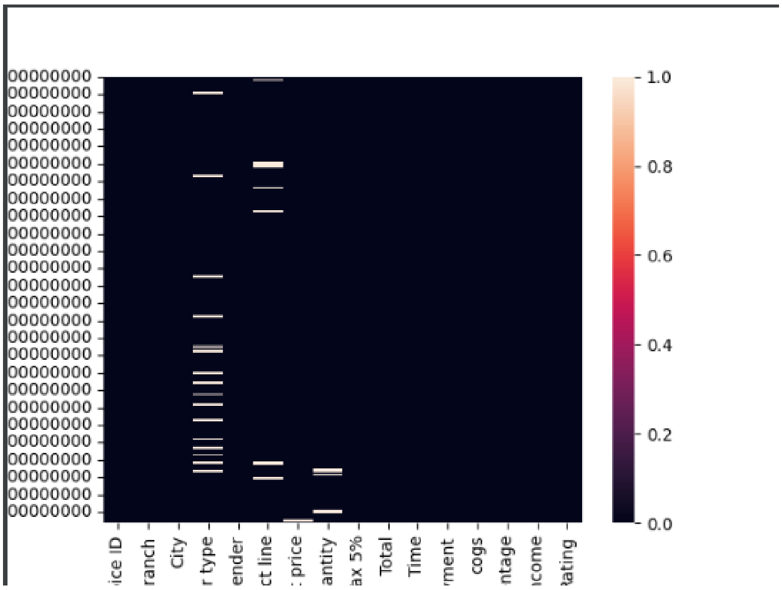


Figure 12: the heatmap indicating missing data:

sns.heatmap(df.isnull(),cbar=False) # cbar is the color indicator

plt.show()

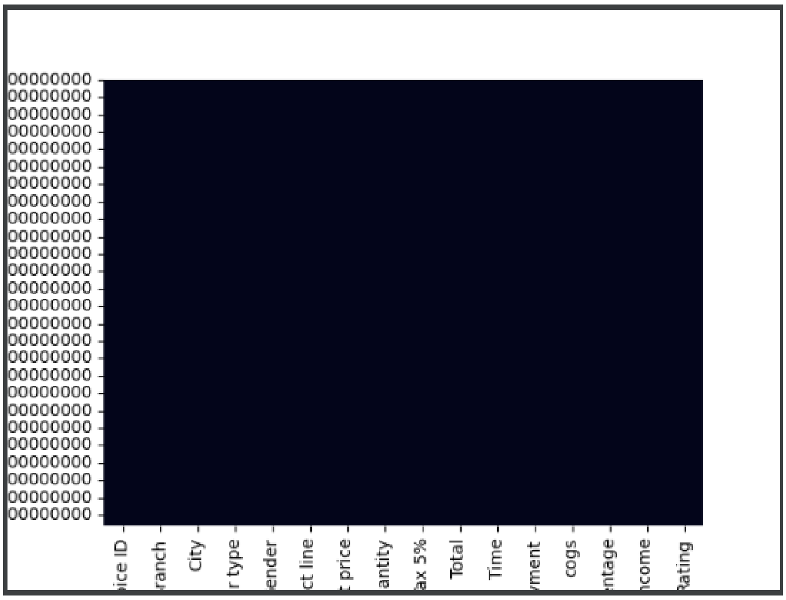


Figure 13: result after transform data to eliminate missing data

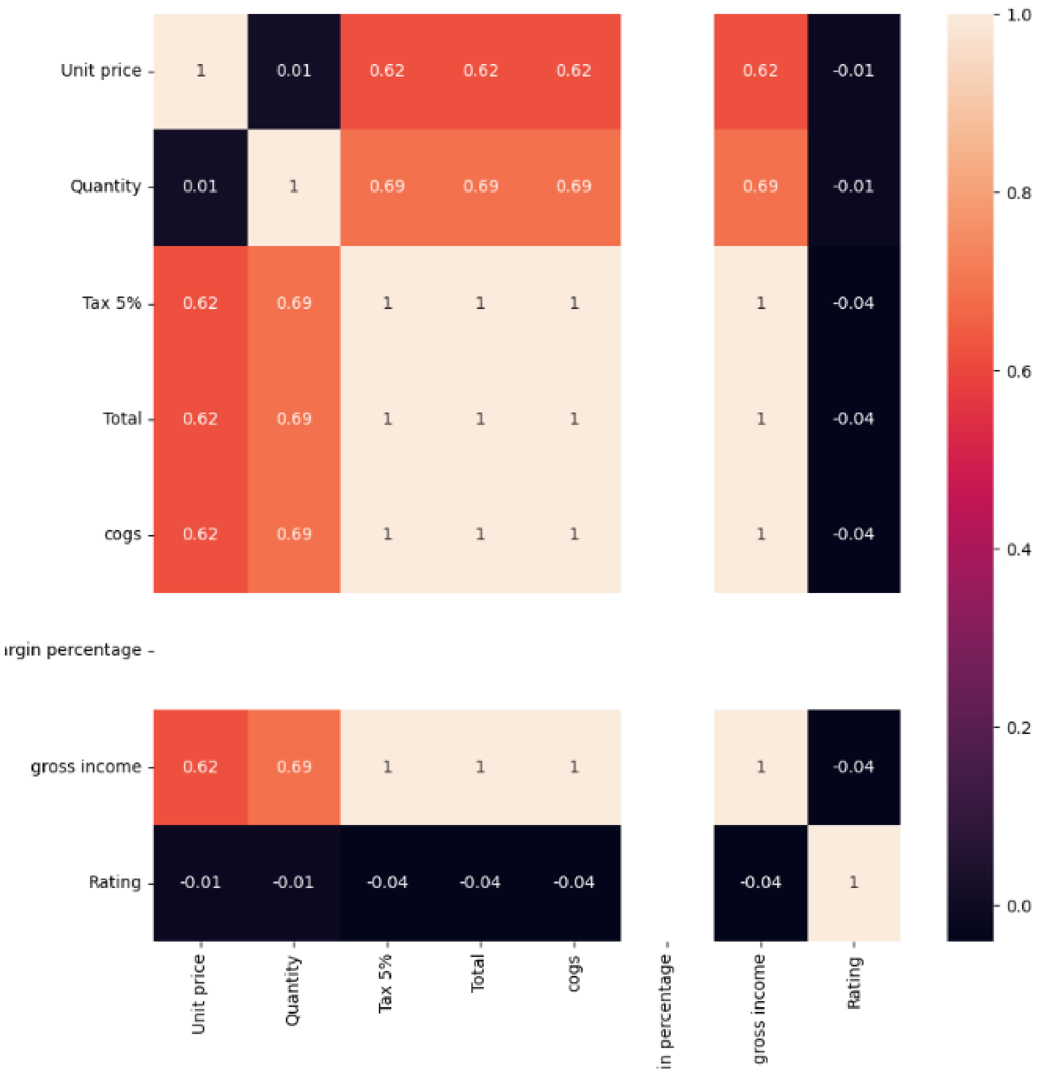


Figure 14: Correlation analysis

```
np.corrcoef(df['gross income'],df['Rating'])
round(np.corrcoef(df['gross income'],df['Rating']) [1][0],2) # pick and rounded to 2nd decimal
# correlation matrix
round(df.corr(),2)

plt.figure(figsize=(10,10)) # resize the figure size in plots
sns.heatmap(round(df.corr(),2),annot=True)
plt.show()
```