Name: Kshitij Hundre

Div: D15C Roll No:18

EXP 1

Aim:

- Load data in Pandas.
- Description of the dataset.
- Drop columns that aren't useful.
- Drop rows with maximum missing values.
- Take care of missing data.
- Create dummy variables.
- Find out outliers (manually)
- standardization and normalization of columns

Data preprocessing

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

Why is Data Preprocessing important?

Preprocessing of data is mainly to check the data quality. The quality can be checked by the following:

- Accuracy: To check whether the data entered is correct or not.
- Completeness: To check whether the data is available or not recorded.
- Consistency: To check whether the same data is kept in all the places that do or do not match.

• Timeliness: The data should be updated correctly.

• Believability: The data should be trustable.

Interpretability: The understandability of the data.

Dataset: <u>SuperMarket Dataset</u>

1) Loading Data in Pandas

0	imp	oort pand a	as as pd														
	df	= pd.read	d_csv('s	sc.csv')													
	df.	head()															
₹		Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gros: income
	0	750-67- 8428	Α	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26.141
	1	226-31- 3081	С	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3.8200
	2	631-41- 3108	Α	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16.215
	3	123-19- 1176	Α	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0480	1/27/2019	20:33	Ewallet	465.76	4.761905	23.2880
	4	373-73- 7910	А	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3785	2/8/2019	10:37	Ewallet	604.17	4.761905	30.208
	4																

2)Description of the dataset.

Attribute/Column	Data	
Name	Туре	Description
Invoice ID	String	Unique identifier for each transaction/invoice.
Branch	String	Branch identifier for the supermarket (A , B , or C).
City	String	City where the supermarket branch is located.
Customer type	String	Type of customer (Member or Normal).
Gender	String	Gender of the customer (Male or Female).
Product line	String	Category of products purchased (Health and beauty , Electronic accessories , etc.).
Unit price	Float	Price per unit of the product.
Quantity	Integer	Number of units purchased.
Tax 5%	Float	5% tax on the total amount for the purchase.
Total	Float	Total bill amount, including tax.
Date	DateTime	Date of the purchase transaction.
Time	String	Time of the purchase transaction.
Payment	String	Payment method used (Cash , Credit card , or Ewallet).
cogs (Cost of Goods Sold)	Float	Total cost of goods sold before tax.
gross margin %	Float	Percentage of gross margin fixed at 4.76%.
gross income	Float	Profit made from the transaction.
Rating	Float	Customer's rating of their experience (range: 1 to 10).

df.info(): Provides an overview of the dataset, including:

- Number of rows and columns.
- Data types of each column (e.g., int, float, object).
- Number of non-null (non-missing) values in each column.

df.describe(): Generates summary statistics for numeric columns, such as:

- count: Number of non-missing values.
- mean: Average value.
- std: Standard deviation.
- min, 25%, 50% (median), 75%, and max: Percentile values.

```
print(df.info())
print(df.describe())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
                                      Non-Null Count Dtype
    Column
     Invoice ID
                                      1000 non-null
                                                           object
      Branch
                                      1000 non-null
                                   1000 non-null
1000 non-null
1000 non-null
1000 non-null
                                                           object
      Customer type
                                                           object
     Gender
                                                           object
      Product line
                                                           object
float64
     Unit price
Quantity
                                     1000 non-null
                                                          int64
      Tax 5%
                                      1000 non-null
                                                           float64
      Total
 10 Date
                                      1000 non-null
                                                           object
      Time
                                      1000 non-null
                                      1000 non-null
1000 non-null
 12
                                                           float64
     cogs
    gross margin percentage 1000 non-null
gross income 1000 non-null
15 gross income
16 Rating
                                                          float64
                                      1000 non-null
                                                          float64
dtypes: float64(7), int64(1), object(9) memory usage: 132.9+ KB
Unit price Quantity Tax 5% Total cogs count 1000.000000 1000.000000 1000.000000 1000.000000 1000.000000
                                                         322.966749
245.885335
mean 55.672130 5.510000 15.379369
std 26.494628 2.923431 11.708825
                                                                            307.58738
234.17651
                                            0.508500
         10.e.
32.87500e
55.230000
77.935000
min
                             1.000000
                                                             10.678500
                                                                              10.17000
                                                          124.422375
253.848000
                                            5.924875
12.088000
25%
                             3.000000
                                                                             118.49750
                                                                             241.76000
                             5.000000
                                            22.445250 471.350250
49.650000 1042.650000
75%
                             8.000000
                                                                             448.90500
                                                                             993.00000
max
                           10.000000
```

3) Drop columns that aren't useful: Columns like Invoice ID may not contribute to analysis (it's often just an identifier). Removing irrelevant columns reduces complexity.



4)Drop rows with maximum missing values.

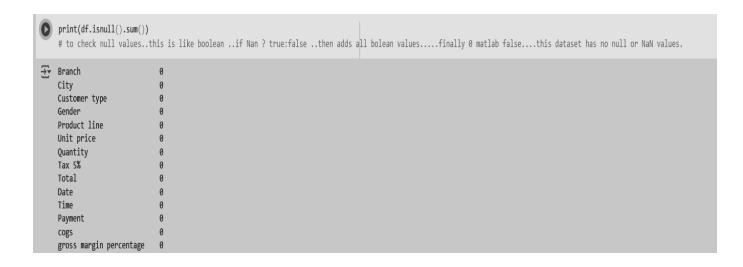
df.dropna(thresh=int(0.5 * len(df.columns))):

- Drops rows where more than half the columns have missing (NaN) values.
- thresh=int(0.5 * len(df.columns)): Ensures that a row must have at least 50% non-null values to remain.

df = ...: Updates the DataFrame after dropping rows.
print(df.info()): Confirms that rows with excessive missing values have been removed.

5)Take care of missing data.

df.fillna(df.mean()): Replaces missing values (NaN) in numeric columns with the mean of that column.

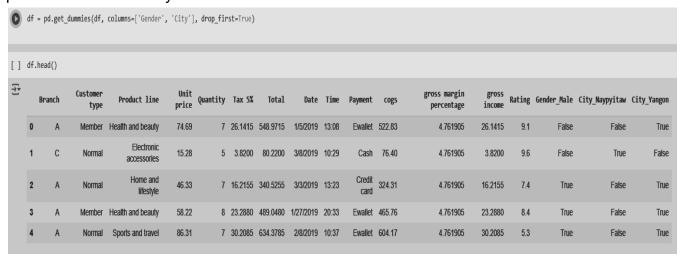


6)Create dummy variables.

pd.get_dummies(): Converts categorical variables into dummy variables (binary indicators: 0 or 1).

• Example: The Gender column becomes Gender_Male (1 if Male, 0 otherwise).

columns=['...']: Specifies which columns to convert. drop_first=True: Avoids the "dummy variable trap" by dropping one dummy variable to prevent multicollinearity.



7) Find out outliers (manually)

```
def detect_outliers(col):
        Q1 = df[col].quantile(0.25) # First quartile (25th percentile)
        Q3 = df[col].quantile(0.75) # Third quartile (75th percentile)
                                  # Interquartile range
        IOR = 03 - 01
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        return df[(df[col] < lower_bound) | (df[col] > upper_bound)]
    outliers = detect_outliers('Total')
    print(outliers)
    if outliers.empty:
       print("No outliers detected.")
       print(f"Outliers detected:\n{outliers}")
    print(f"Number of outliers: {len(outliers)}")
∓
        Branch Customer type
                                   Product line Unit price Quantity Tax 5% \
                                                      95.58
    166
                              Home and lifestyle
                                                                      47.790
                     Normal
                                                                   10
    167
            Α
                     Normal Fashion accessories
                                                      98.98
                                                                   10 49.490
    350
            C
                     Member
                             Fashion accessories
                                                      99.30
                                                                   10 49.650
                                                                      47.720
    357
                                                      95.44
                     Normal
                               Sports and travel
                                                                   10
                             Fashion accessories
                                                      97.21
                                                                      48.605
                     Member
                              Food and beverages
                                                      98.52
                                                                   10
                                                                      49.260
    557
    699
            C
                     Normal
                              Home and lifestyle
                                                      97.50
                                                                   10 48.750
                              Home and lifestyle
                                                      97.37
                                                                   10 48.685
    792
            В
                     Normal
            В
                             Home and lifestyle
                                                      97.38
                                                                  10 48.690
    996
                     Normal
           Total
                       Date
                             Time
                                       Payment cogs gross margin percentage \
    166 1003.590 1/16/2019 13:32
                                          Cash 955.8
                                                                     4.761905
                   2/8/2019 16:20 Credit card 989.8
    167 1039.290
                                                                      4.761905
    350 1042.650 2/15/2019
                             14:53 Credit card 993.0
                                                                      4.761905
                  1/9/2019
    357 1002.120
                             13:45
                                         Cash 954.4
    422 1020.705
                   2/8/2019
                             13:00 Credit card 972.1
                                                                      4.761905
    557 1034.460 1/30/2019 20:23
                                       Ewallet 985.2
                                                                      4.761905
    699 1023.750 1/12/2019 16:18
                                       Ewallet 975.0
                                                                      4.761905
    792 1022.385 1/15/2019 13:48 Credit card 973.7
                                                                      4.761905
    996 1022.490 3/2/2019 17:16
                                       Ewallet 973.8
                                                                      4.761905
         gross income Rating Gender_Male City_Naypyitaw City_Yangon
    166
              47.790
                       4.8
                                     True
                                                    True
                                                               False
                                                   False
              49.490
                                     True
                                                                True
```

```
Total
                  Date
                         Time
                                    Payment
                                             cogs gross margin percentage
166 1003.590 1/16/2019 13:32
                                      Cash 955.8
                                                                  4.761905
    1039.290
              2/8/2019 16:20 Credit card
                                            989.8
                                                                  4.761905
167
350 1042.650 2/15/2019 14:53
                                Credit card 993.0
                                                                  4.761905
357 1002.120
              1/9/2019 13:45
                                     Cash 954.4
                                                                  4.761905
    1020.705
               2/8/2019 13:00 Credit card 972.1
422
                                                                  4.761905
    1034.460 1/30/2019
557
                         20:23
                                   Ewallet 985.2
                                                                  4.761905
699 1023.750 1/12/2019 16:18
                                    Ewallet 975.0
                                                                 4.761905
792 1022.385 1/15/2019 13:48 Credit card 973.7
                                                                  4.761905
    1022.490
              3/2/2019 17:16
                                    Ewallet 973.8
                                                                  4.761905
996
    gross income Rating Gender_Male City_Naypyitaw City_Yangon
          47.790
                    4.8
                                True
166
                                                True
          49.490
                     8.7
                                                False
167
                                 True
                                                             True
          49.650
                               False
350
                     6.6
                                                True
                                                            False
          47.720
357
                     5.2
                                False
                                                True
                                                            False
422
          48.605
                     8.7
                                False
                                                True
                                                            False
557
          49.260
                     4.5
                                False
                                                True
                                                            False
699
          48.750
                     8.0
                                True
                                                True
                                                            False
792
          48.685
                     4.9
                                False
                                               False
                                                            False
996
          48.690
                     4.4
                                False
                                               False
                                                            False
Number of outliers: 9
```

8) standardization and normalization of columns

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

Standardization equation

$$X' = \frac{X - \mu}{\sigma}$$

To standardize your data, we need to import the StandardScalar from the sklearn library and apply it to our dataset.

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

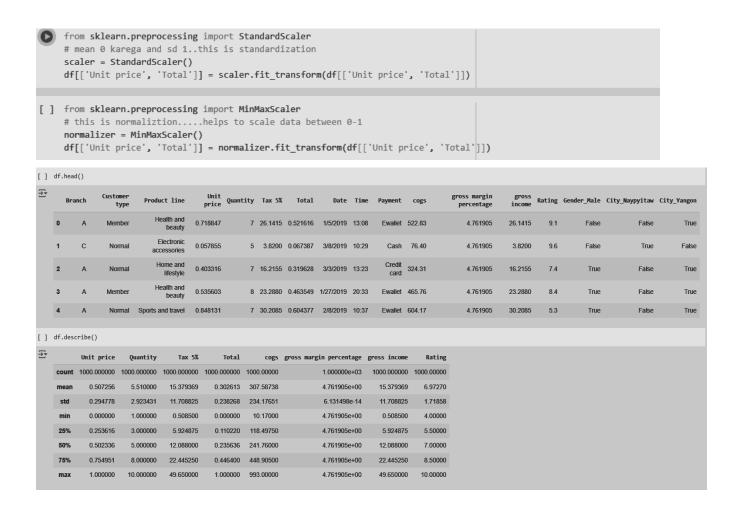
Normalization equation

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, Xmax and Xmin are the maximum and the minimum values of the feature respectively.

- When the value of X is the minimum value in the column, the numerator will be 0, and hence X' is 0
- On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator and thus the value of X' is 1
- If the value of X is between the minimum and the maximum value, then the value of X' is between 0 and 1

To normalize your data, you need to import the MinMaxScalar from the sklearn library and apply it to our dataset.



Conclusion:

Thus we have understood how to perform data preprocessing which can further be taken into exploratory data analysis and further in the Model preparation sequence.