In [5]:

```
!pip install pandas numpy seaborn matplotlib klib dtale scikit-learn joblib pandas-
```

```
Requirement already satisfied: pandas in c:\python3107\lib\site-packages (1.4.4)
Requirement already satisfied: numpy in c:\python3107\lib\site-packages (1.23.3)
Requirement already satisfied: seaborn in c:\python3107\lib\site-packages (0.11.2)
Requirement already satisfied: matplotlib in c:\python3107\lib\site-packages (3.5.
3)
Requirement already satisfied: klib in c:\python3107\lib\site-packages (1.0.5)
Requirement already satisfied: dtale in c:\python3107\lib\site-packages (2.8.1)
Requirement already satisfied: scikit-learn in c:\python3107\lib\site-packages (1.
1.2)
Requirement already satisfied: joblib in c:\python3107\lib\site-packages (1.1.0)
Requirement already satisfied: pandas-profiling in c:\python3107\lib\site-packages
(3.3.0)
Requirement already satisfied: xgboost in c:\python3107\lib\site-packages (1.6.2)
Requirement already satisfied: pytz>=2020.1 in c:\python3107\lib\site-packages (fr
om pandas) (2022.2.1)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\python3107\lib\site-pa
ckages (from pandas) (2.8.2)
Requirement already satisfied: scipy>=1.0 in c:\python3107\lib\site-packages (from
seaborn) (1.9.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\python3107\lib\site-package
s (from matplotlib) (4.37.3)
Requirement already satisfied: cycler>=0.10 in c:\python3107\lib\site-packages (fr
om matplotlib) (0.11.0)
Requirement already satisfied: packaging>=20.0 in c:\python3107\lib\site-packages
(from matplotlib) (21.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:\python3107\lib\site-packages
(from matplotlib) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in c:\python3107\lib\site-packages (f
rom matplotlib) (9.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\python3107\lib\site-package
s (from matplotlib) (1.4.4)
Requirement already satisfied: Jinja2<4.0.0,>=3.0.3 in c:\python3107\lib\site-pack
ages (from klib) (3.1.2)
Requirement already satisfied: requests in c:\python3107\lib\site-packages (from d
tale) (2.28.1)
Requirement already satisfied: openpyxl in c:\python3107\lib\site-packages (from d
tale) (3.0.10)
Requirement already satisfied: dash-bootstrap-components in c:\python3107\lib\site
-packages (from dtale) (1.2.1)
Requirement already satisfied: dash>=2.0.0 in c:\python3107\lib\site-packages (fro
m dtale) (2.6.2)
Requirement already satisfied: et-xmlfile in c:\python3107\lib\site-packages (from
dtale) (1.1.0)
Requirement already satisfied: kaleido in c:\python3107\lib\site-packages (from dt
ale) (0.2.1)
Requirement already satisfied: strsimpy in c:\python3107\lib\site-packages (from d
tale) (0.2.1)
Requirement already satisfied: xarray in c:\python3107\lib\site-packages (from dta
le) (2022.6.0)
Requirement already satisfied: networkx in c:\python3107\lib\site-packages (from d
tale) (2.8.6)
Requirement already satisfied: xlrd in c:\python3107\lib\site-packages (from dtal
e) (2.0.1)
Requirement already satisfied: lz4 in c:\python3107\lib\site-packages (from dtale)
(4.0.2)
Requirement already satisfied: future>=0.14.0 in c:\python3107\lib\site-packages
(from dtale) (0.18.2)
Requirement already satisfied: squarify in c:\python3107\lib\site-packages (from d
tale) (0.4.3)
Requirement already satisfied: dash-daq in c:\python3107\lib\site-packages (from d
tale) (0.5.0)
```

```
Requirement already satisfied: six in c:\python3107\lib\site-packages (from dtale)
(1.16.0)
Requirement already satisfied: flask-ngrok in c:\python3107\lib\site-packages (fro
m dtale) (0.0.25)
Requirement already satisfied: Flask in c:\python3107\lib\site-packages (from dtal
e) (2.2.2)
Requirement already satisfied: dash-colorscales in c:\python3107\lib\site-packages
(from dtale) (0.0.4)
Requirement already satisfied: missingno<=0.4.2 in c:\python3107\lib\site-packages
(from dtale) (0.4.2)
Requirement already satisfied: itsdangerous in c:\python3107\lib\site-packages (fr
om dtale) (2.1.2)
Requirement already satisfied: certifi in c:\python3107\lib\site-packages (from dt
ale) (2022.9.24)
Requirement already satisfied: Flask-Compress in c:\python3107\lib\site-packages
(from dtale) (1.13)
Requirement already satisfied: plotly>=5.0.0 in c:\python3107\lib\site-packages (f
rom dtale) (5.10.0)
Requirement already satisfied: statsmodels in c:\python3107\lib\site-packages (fro
m dtale) (0.13.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\python3107\lib\site-pack
ages (from scikit-learn) (3.1.0)
Requirement already satisfied: phik<0.13,>=0.11.1 in c:\python3107\lib\site-packag
es (from pandas-profiling) (0.12.2)
Requirement already satisfied: tangled-up-in-unicode==0.2.0 in c:\python3107\lib\s
ite-packages (from pandas-profiling) (0.2.0)
Requirement already satisfied: multimethod<1.9,>=1.4 in c:\python3107\lib\site-pac
kages (from pandas-profiling) (1.8)
Requirement already satisfied: visions[type_image_path]==0.7.5 in c:\python3107\li
b\site-packages (from pandas-profiling) (0.7.5)
Requirement already satisfied: htmlmin==0.1.12 in c:\python3107\lib\site-packages
(from pandas-profiling) (0.1.12)
Requirement already satisfied: pydantic<1.10,>=1.8.1 in c:\python3107\lib\site-pac
kages (from pandas-profiling) (1.9.2)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in c:\python3107\lib\site-packag
es (from pandas-profiling) (6.0)
Requirement already satisfied: tqdm<4.65,>=4.48.2 in c:\python3107\lib\site-packag
es (from pandas-profiling) (4.64.1)
Requirement already satisfied: attrs>=19.3.0 in c:\python3107\lib\site-packages (f
rom visions[type_image_path]==0.7.5->pandas-profiling) (22.1.0)
Requirement already satisfied: imagehash in c:\python3107\lib\site-packages (from
visions[type_image_path]==0.7.5->pandas-profiling) (4.3.1)
Requirement already satisfied: dash-html-components==2.0.0 in c:\python3107\lib\si
te-packages (from dash>=2.0.0->dtale) (2.0.0)
Requirement already satisfied: dash-core-components==2.0.0 in c:\python3107\lib\si
te-packages (from dash>=2.0.0->dtale) (2.0.0)
Requirement already satisfied: dash-table==5.0.0 in c:\python3107\lib\site-package
s (from dash>=2.0.0->dtale) (5.0.0)
Requirement already satisfied: Werkzeug>=2.2.2 in c:\python3107\lib\site-packages
(from Flask->dtale) (2.2.2)
Requirement already satisfied: click>=8.0 in c:\python3107\lib\site-packages (from
Flask->dtale) (8.1.3)
Requirement already satisfied: MarkupSafe>=2.0 in c:\python3107\lib\site-packages
(from Jinja2<4.0.0,>=3.0.3->klib) (2.1.1)
Requirement already satisfied: tenacity>=6.2.0 in c:\python3107\lib\site-packages
(from plotly>=5.0.0->dtale) (8.1.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\python3107\lib\sit
e-packages (from pydantic<1.10,>=1.8.1->pandas-profiling) (4.3.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\python3107\lib\site-pac
kages (from requests->dtale) (1.26.12)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\python3107\lib\site-
```

```
packages (from requests->dtale) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in c:\python3107\lib\site-packages (fr
om requests->dtale) (3.4)
Requirement already satisfied: patsy>=0.5.2 in c:\python3107\lib\site-packages (fr
om statsmodels->dtale) (0.5.2)
Requirement already satisfied: colorama in c:\python3107\lib\site-packages (from t
qdm<4.65,>=4.48.2->pandas-profiling) (0.4.5)
Requirement already satisfied: brotli in c:\python3107\lib\site-packages (from Fla
sk-Compress->dtale) (1.0.9)
Requirement already satisfied: PyWavelets in c:\python3107\lib\site-packages (from
imagehash->visions[type_image_path]==0.7.5->pandas-profiling) (1.4.1)
```

In [10]:

```
import pandas as pd
import numpy as np
%matplotlib inline
#magic function in IPython
import matplotlib.pyplot as plt     # is a collection of command style functions th
import seaborn as sns
```

In [11]:

```
df_train= pd.read_csv(r'D:\5th_semester\MiniProject2A\Projectworking\dataset\Train.
df_test= pd.read_csv(r'D:\5th_semester\MiniProject2A\Projectworking\dataset\Test.cs
```

In [12]:

```
df_train.head()  # displays the first five rows of the dataframe by default
```

Out[12]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_I
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	

In [13]:

```
#df_test.head()
```

In [14]:

```
df_train.shape   # a tuple of array dimensions that tells the number of rows and col
```

Out[14]:

```
(8523, 12)
```

In [15]:

```
df_train.isnull().sum()  #seeing the number of null values in the dataset
```

Out[15]:
```
Item_Identifier                 0
Item_Weight                  1463
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                  2410
Outlet_Location_Type            0
Outlet_Type                     0
Item_Outlet_Sales               0
dtype: int64
```

In [16]:
```
df_test.isnull().sum()
```

Out[16]:
```
Item_Identifier                 0
Item_Weight                   976
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                  1606
Outlet_Location_Type            0
Outlet_Type                     0
dtype: int64
```

In [17]:
```
df_train.info()   #seeing the detailed info of the dataset and its types of target
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Item_Identifier            8523 non-null   object 
 1   Item_Weight                7060 non-null   float64
 2   Item_Fat_Content           8523 non-null   object 
 3   Item_Visibility            8523 non-null   float64
 4   Item_Type                  8523 non-null   object 
 5   Item_MRP                   8523 non-null   float64
 6   Outlet_Identifier          8523 non-null   object 
 7   Outlet_Establishment_Year  8523 non-null   int64  
 8   Outlet_Size                6113 non-null   object 
 9   Outlet_Location_Type       8523 non-null   object 
 10  Outlet_Type                8523 non-null   object 
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

In [18]:
```
df_train.describe()  # to generate descriptive statistics that summarize the centra
                     # shape of a dataset's distribution, excluding NaN values.
```

Out[18]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	7060.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.643456	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.773750	0.026989	93.826500	1987.000000	834.247400
50%	12.600000	0.053931	143.012800	1999.000000	1794.331000
75%	16.850000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

Item_Weight is numerical column so we fill it with Mean Imputation

In [19]:

```
df_train['Item_Weight'].describe()  #seeing all the central tendenies of the datase
```

Out[19]:

```
count    7060.000000
mean       12.857645
std         4.643456
min         4.555000
25%         8.773750
50%        12.600000
75%        16.850000
max        21.350000
Name: Item_Weight, dtype: float64
```

In [20]:

```
df_train['Item_Weight'].fillna(df_train['Item_Weight'].mean(),inplace=True)  #repla
df_test['Item_Weight'].fillna(df_train['Item_Weight'].mean(),inplace=True)
```

In [21]:

```
df_train.isnull().sum()  #no null values in item weight
```

Out[21]:

```
Item_Identifier                 0
Item_Weight                     0
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                  2410
Outlet_Location_Type            0
Outlet_Type                     0
Item_Outlet_Sales               0
dtype: int64
```

In [22]:

```
df_train['Item_Weight'].describe()
```

```
Out[22]: count    8523.000000
         mean       12.857645
         std         4.226124
         min         4.555000
         25%         9.310000
         50%        12.857645
         75%        16.000000
         max        21.350000
         Name: Item_Weight, dtype: float64
```

Outlet_Size is catagorical column so we fill it with Mode Imputation

```
In [23]: df_train['Outlet_Size']  #it is a categorical value
```

```
Out[23]: 0       Medium
         1       Medium
         2       Medium
         3          NaN
         4         High
                 ...
         8518      High
         8519       NaN
         8520     Small
         8521    Medium
         8522     Small
         Name: Outlet_Size, Length: 8523, dtype: object
```

```
In [24]: df_train['Outlet_Size'].value_counts()
```

```
Out[24]: Medium    2793
         Small     2388
         High       932
         Name: Outlet_Size, dtype: int64
```

```
In [25]: df_train['Outlet_Size'].mode()
```

```
Out[25]: 0    Medium
         Name: Outlet_Size, dtype: object
```

```
In [26]: df_train['Outlet_Size'].fillna(df_train['Outlet_Size'].mode()[0],inplace=True)
         df_test['Outlet_Size'].fillna(df_test['Outlet_Size'].mode()[0],inplace=True)
```

pandas treats the mode as something special since they can be unimodal , bimodal or multimodal distributions they

had to make sure that 1 value could be returned "Always return series even if only one value is returned"

```
In [27]: df_train.isnull().sum()  #no null value :)
```

Out[27]:
```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content             0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
Outlet_Establishment_Year    0
Outlet_Size                  0
Outlet_Location_Type         0
Outlet_Type                  0
Item_Outlet_Sales            0
dtype: int64
```

In [28]:
```
df_test.isnull().sum()
```

Out[28]:
```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content             0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
Outlet_Establishment_Year    0
Outlet_Size                  0
Outlet_Location_Type         0
Outlet_Type                  0
dtype: int64
```

Dimesnsionality reduction of item identifier and outlet identifier

In [29]:
```
df_train.drop(['Item_Identifier','Outlet_Identifier'],axis=1,inplace=True)
df_test.drop(['Item_Identifier','Outlet_Identifier'],axis=1,inplace=True)
```

In [30]:
```
df_train
```

Out[30]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Establishment_
0	9.300	Low Fat	0.016047	Dairy	249.8092	
1	5.920	Regular	0.019278	Soft Drinks	48.2692	
2	17.500	Low Fat	0.016760	Meat	141.6180	
3	19.200	Regular	0.000000	Fruits and Vegetables	182.0950	
4	8.930	Low Fat	0.000000	Household	53.8614	
...	...	...	...	...	...	
8518	6.865	Low Fat	0.056783	Snack Foods	214.5218	
8519	8.380	Regular	0.046982	Baking Goods	108.1570	
8520	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	
8521	7.210	Regular	0.145221	Snack Foods	103.1332	
8522	14.800	Low Fat	0.044878	Soft Drinks	75.4670	

8523 rows × 10 columns

In [31]:

```
df_test
```

Out[31]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Establishment_
0	20.750000	Low Fat	0.007565	Snack Foods	107.8622	
1	8.300000	reg	0.038428	Dairy	87.3198	
2	14.600000	Low Fat	0.099575	Others	241.7538	
3	7.315000	Low Fat	0.015388	Snack Foods	155.0340	
4	12.857645	Regular	0.118599	Dairy	234.2300	
...	...	...	...	...	...	
5676	10.500000	Regular	0.013496	Snack Foods	141.3154	
5677	7.600000	Regular	0.142991	Starchy Foods	169.1448	
5678	10.000000	Low Fat	0.073529	Health and Hygiene	118.7440	
5679	15.300000	Regular	0.000000	Canned	214.6218	
5680	9.500000	Regular	0.104720	Canned	79.7960	

5681 rows × 9 columns

EDA (Exploratory data analysis) with Dtale library

In [32]:

```
import dtale
```

In [33]:

```
dtale.show(df_train)
```

▶	0
0	

Out[33]:

EDA using Pandas profiling

In [34]:

```
pip install ipywidgets
```

```
Requirement already satisfied: ipywidgets in c:\python3107\lib\site-packages (8.0.
2)
Requirement already satisfied: traitlets>=4.3.1 in c:\python3107\lib\site-packages
(from ipywidgets) (5.4.0)
Requirement already satisfied: jupyterlab-widgets~=3.0 in c:\python3107\lib\site-p
ackages (from ipywidgets) (3.0.3)
Requirement already satisfied: widgetsnbextension~=4.0 in c:\python3107\lib\site-p
ackages (from ipywidgets) (4.0.3)
Requirement already satisfied: ipykernel>=4.5.1 in c:\python3107\lib\site-packages
(from ipywidgets) (6.16.0)
Requirement already satisfied: ipython>=6.1.0 in c:\python3107\lib\site-packages
(from ipywidgets) (8.5.0)
Requirement already satisfied: packaging in c:\python3107\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets) (21.3)
Requirement already satisfied: matplotlib-inline>=0.1 in c:\python3107\lib\site-pa
ckages (from ipykernel>=4.5.1->ipywidgets) (0.1.6)
Requirement already satisfied: nest-asyncio in c:\python3107\lib\site-packages (fr
om ipykernel>=4.5.1->ipywidgets) (1.5.5)
Requirement already satisfied: jupyter-client>=6.1.12 in c:\python3107\lib\site-pa
ckages (from ipykernel>=4.5.1->ipywidgets) (7.3.5)
Requirement already satisfied: tornado>=6.1 in c:\python3107\lib\site-packages (fr
om ipykernel>=4.5.1->ipywidgets) (6.2)
Requirement already satisfied: debugpy>=1.0 in c:\python3107\lib\site-packages (fr
om ipykernel>=4.5.1->ipywidgets) (1.6.3)
Requirement already satisfied: pyzmq>=17 in c:\python3107\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets) (24.0.1)
Requirement already satisfied: psutil in c:\python3107\lib\site-packages (from ipy
kernel>=4.5.1->ipywidgets) (5.9.2)
Requirement already satisfied: stack-data in c:\python3107\lib\site-packages (from
ipython>=6.1.0->ipywidgets) (0.5.1)
Requirement already satisfied: colorama in c:\python3107\lib\site-packages (from i
python>=6.1.0->ipywidgets) (0.4.5)
Requirement already satisfied: jedi>=0.16 in c:\python3107\lib\site-packages (from
ipython>=6.1.0->ipywidgets) (0.18.1)
Requirement already satisfied: decorator in c:\python3107\lib\site-packages (from
ipython>=6.1.0->ipywidgets) (5.1.1)
Requirement already satisfied: pygments>=2.4.0 in c:\python3107\lib\site-packages
(from ipython>=6.1.0->ipywidgets) (2.13.0)
Requirement already satisfied: prompt-toolkit<3.1.0,>3.0.1 in c:\python3107\lib\si
te-packages (from ipython>=6.1.0->ipywidgets) (3.0.31)
Requirement already satisfied: backcall in c:\python3107\lib\site-packages (from i
python>=6.1.0->ipywidgets) (0.2.0)
Requirement already satisfied: pickleshare in c:\python3107\lib\site-packages (fro
m ipython>=6.1.0->ipywidgets) (0.7.5)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in c:\python3107\lib\site-packa
ges (from jedi>=0.16->ipython>=6.1.0->ipywidgets) (0.8.3)
Requirement already satisfied: jupyter-core>=4.9.2 in c:\python3107\lib\site-packa
ges (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (4.11.1)
Requirement already satisfied: entrypoints in c:\python3107\lib\site-packages (fro
m jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (0.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\python3107\lib\site-pa
ckages (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (2.8.2)
Requirement already satisfied: wcwidth in c:\python3107\lib\site-packages (from pr
ompt-toolkit<3.1.0,>3.0.1->ipython>=6.1.0->ipywidgets) (0.2.5)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\python3107\lib\site-
packages (from packaging->ipykernel>=4.5.1->ipywidgets) (3.0.9)
Requirement already satisfied: executing in c:\python3107\lib\site-packages (from
stack-data->ipython>=6.1.0->ipywidgets) (1.1.0)
Requirement already satisfied: asttokens in c:\python3107\lib\site-packages (from
stack-data->ipython>=6.1.0->ipywidgets) (2.0.8)
Requirement already satisfied: pure-eval in c:\python3107\lib\site-packages (from
stack-data->ipython>=6.1.0->ipywidgets) (0.2.2)
Requirement already satisfied: pywin32>=1.0 in c:\python3107\lib\site-packages (fr
om jupyter-core>=4.9.2->jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (30
4)
Requirement already satisfied: six>=1.5 in c:\python3107\lib\site-packages (from p
ython-dateutil>=2.8.2->jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (1.1
6.0)
Note: you may need to restart the kernel to use updated packages.
```

In [37]:

```python
from pandas_profiling import ProfileReport
```

In [38]:

```python
profile = ProfileReport(df_train, title ="Pandas Profiling Report")
```

In [39]:

```python
profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Variables

Item_Weight

Real number ($\mathbb{R}_{\geq 0}$)

HIGH CORRELATION (This variable has a high correlation with 1 fields: Outlet_Type)

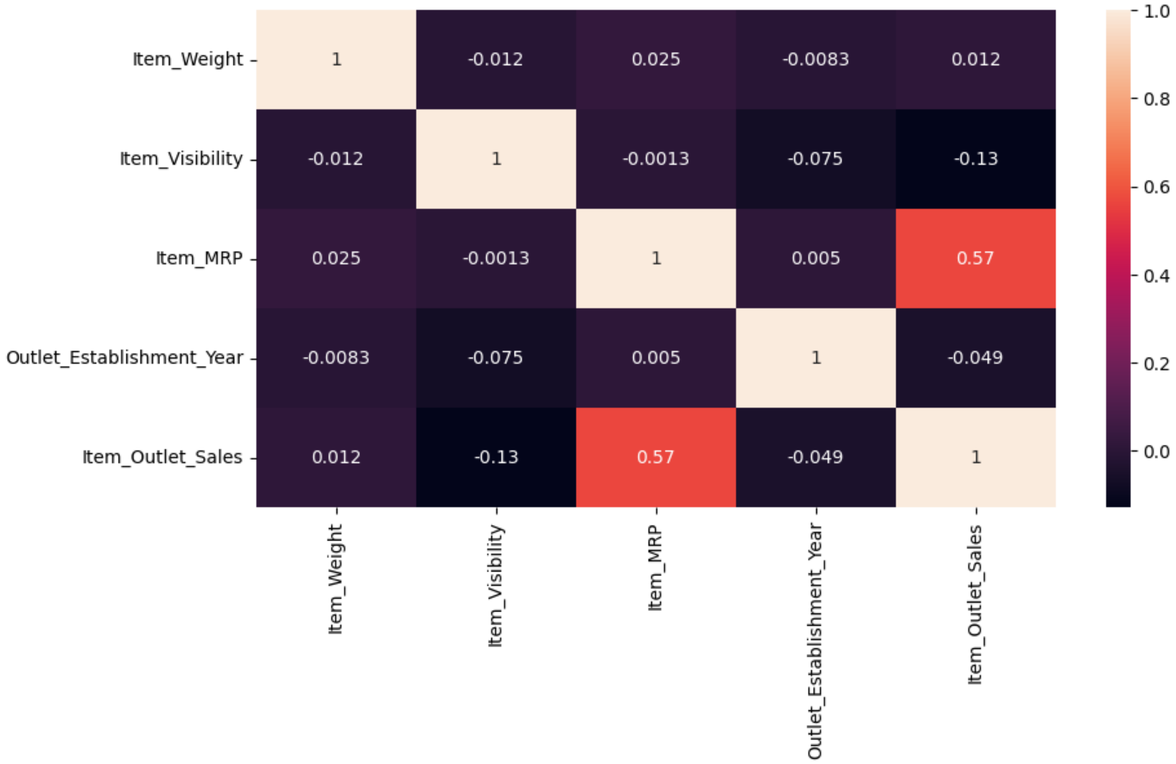
Distinct	416
Distinct (%)	4.9%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Mean	12.85764518

Minimum	4.555
Maximum	21.35
Zeros	0
Zeros (%)	0.0%
Negative	0
Negative (%)	0.0%
Memory size	66.7 KiB

Out[39]:

In [40]:
```
plt.figure(figsize=(10,5))
sns.heatmap(df_train.corr(),annot=True)
plt.show()
```

```
2022-10-07 20:54:08,791 - WARNING  - findfont: Font family ['Heiti TC'] not found.
Falling back to DejaVu Sans.
```

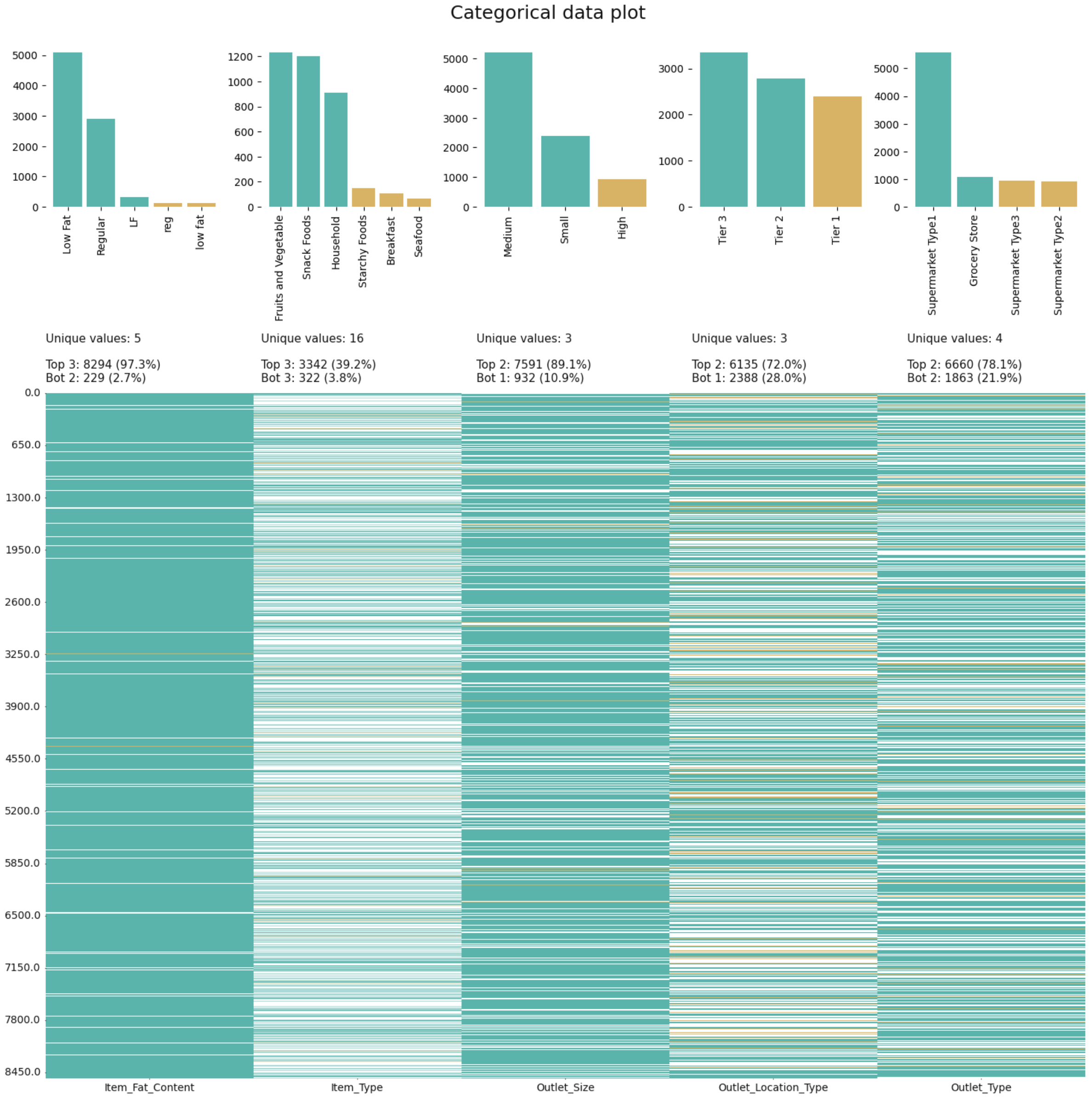


EDA using Klib library

In [41]:
```
import klib
```

In [45]:
```
# klib.describe - functions for visualizing datasets
klib.cat_plot(df_train) # returns a visualization of the number and frequency of ca
```

Out[45]:
```
GridSpec(6, 5)
```



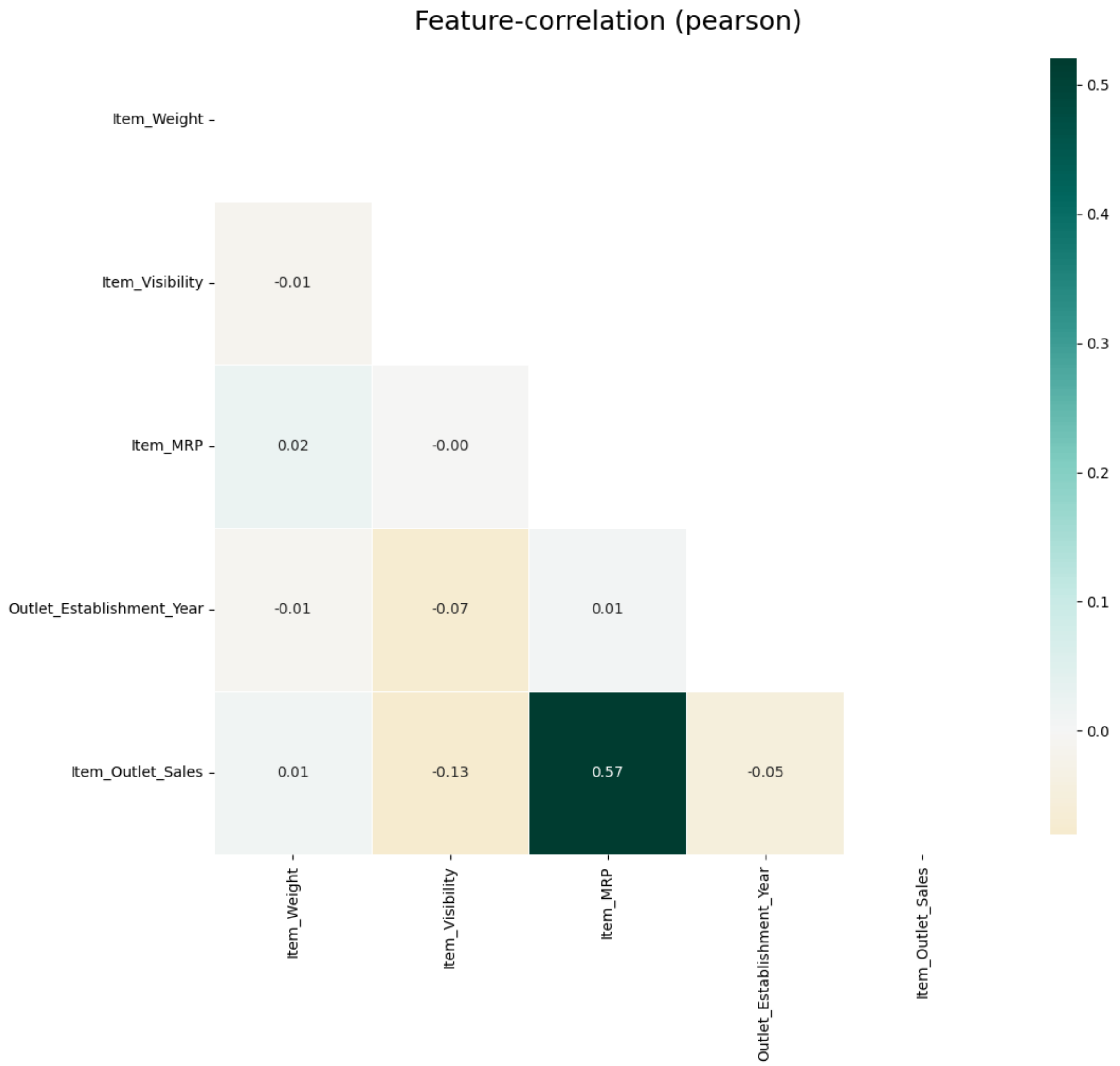
In [46]:
```
klib.corr_mat(df_train) # returns a color-encoded correlation matrix
```

Out[46]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Iten
Item_Weight	1.00	-0.01	0.02	-0.01	
Item_Visibility	-0.01	1.00	-0.00	-0.07	
Item_MRP	0.02	-0.00	1.00	0.01	
Outlet_Establishment_Year	-0.01	-0.07	0.01	1.00	
Item_Outlet_Sales	0.01	-0.13	0.57	-0.05	

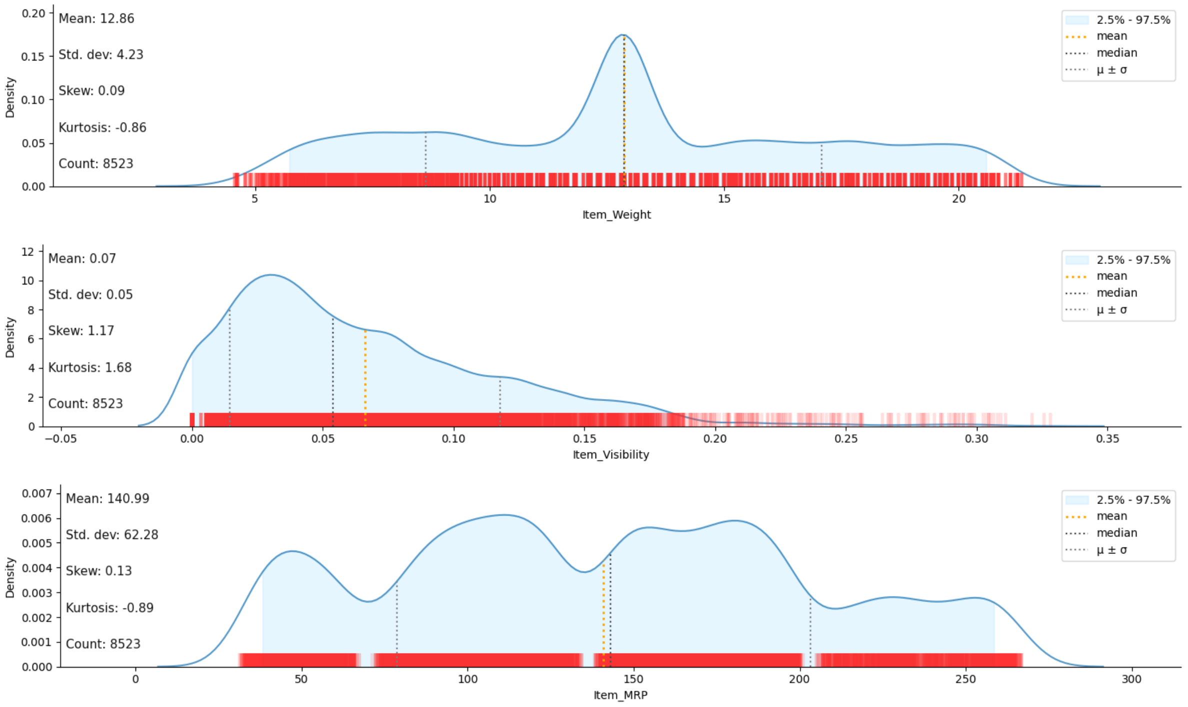
In [47]:
```
klib.corr_plot(df_train) # returns a color-encoded heatmap, ideal for correlations
```

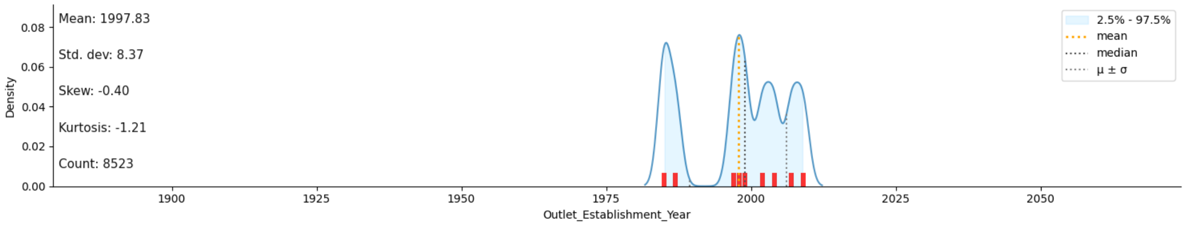
Out[47]:
```
<AxesSubplot:title={'center':'Feature-correlation (pearson)'}>
```

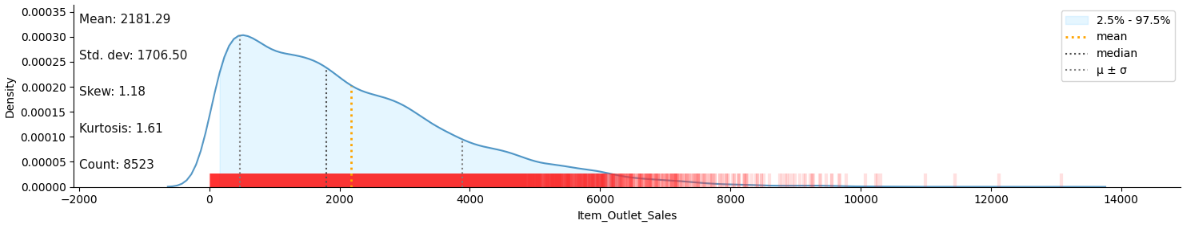


```
In [48]: klib.dist_plot(df_train) # returns a distribution plot for every numeric feature
```

```
Out[48]: <AxesSubplot:xlabel='Item_Outlet_Sales', ylabel='Density'>
```







```
In [49]: klib.missingval_plot(df_train) # returns a figure containing information about miss
```

```
No missing values found in the dataset.
```

Data cleaning Klib

```
In [54]: # klib.clean - functions for cleaning datasets
         klib.data_cleaning(df_train) # performs datacleaning (drop duplicates & empty rows/
```

```
Shape of cleaned data: (8523, 10) - Remaining NAs: 0


Dropped rows: 0
     of which 0 duplicates. (Rows (first 150 shown): [])

Dropped columns: 0
     of which 0 single valued.     Columns: []
Dropped missing values: 0
Reduced memory by at least: 0.46 MB (-70.77%)
```

Out[54]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_y
0	9.300000	Low Fat	0.016047	Dairy	249.809204	1
1	5.920000	Regular	0.019278	Soft Drinks	48.269199	2
2	17.500000	Low Fat	0.016760	Meat	141.617996	1
3	19.200001	Regular	0.000000	Fruits and Vegetables	182.095001	1
4	8.930000	Low Fat	0.000000	Household	53.861401	1
...	...	...	...	...	...	
8518	6.865000	Low Fat	0.056783	Snack Foods	214.521805	1
8519	8.380000	Regular	0.046982	Baking Goods	108.156998	2
8520	10.600000	Low Fat	0.035186	Health and Hygiene	85.122398	2
8521	7.210000	Regular	0.145221	Snack Foods	103.133202	2
8522	14.800000	Low Fat	0.044878	Soft Drinks	75.467003	1

8523 rows × 10 columns

In [55]:

```
klib.clean_column_names(df_train) # cleans and standardizes column names, also call
```

Out[55]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_ye
0	9.300	Low Fat	0.016047	Dairy	249.8092	19
1	5.920	Regular	0.019278	Soft Drinks	48.2692	20
2	17.500	Low Fat	0.016760	Meat	141.6180	19
3	19.200	Regular	0.000000	Fruits and Vegetables	182.0950	19
4	8.930	Low Fat	0.000000	Household	53.8614	19
...	...	...	...	...	...	
8518	6.865	Low Fat	0.056783	Snack Foods	214.5218	19
8519	8.380	Regular	0.046982	Baking Goods	108.1570	20
8520	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	20
8521	7.210	Regular	0.145221	Snack Foods	103.1332	20
8522	14.800	Low Fat	0.044878	Soft Drinks	75.4670	19

8523 rows × 10 columns

In [56]:

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   item_weight                8523 non-null   float64
 1   item_fat_content           8523 non-null   object 
 2   item_visibility            8523 non-null   float64
 3   item_type                  8523 non-null   object 
 4   item_mrp                   8523 non-null   float64
 5   outlet_establishment_year  8523 non-null   int64  
 6   outlet_size                8523 non-null   object 
 7   outlet_location_type       8523 non-null   object 
 8   outlet_type                8523 non-null   object 
 9   item_outlet_sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(5)
memory usage: 666.0+ KB
```

In [57]:

```
df_train=klib.convert_datatypes(df_train) # converts existing to more efficient dty
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype   
---  ------                     --------------  -----   
 0   item_weight                8523 non-null   float32 
 1   item_fat_content           8523 non-null   category
 2   item_visibility            8523 non-null   float32 
 3   item_type                  8523 non-null   category
 4   item_mrp                   8523 non-null   float32 
 5   outlet_establishment_year  8523 non-null   int16   
 6   outlet_size                8523 non-null   category
 7   outlet_location_type       8523 non-null   category
 8   outlet_type                8523 non-null   category
 9   item_outlet_sales          8523 non-null   float32 
dtypes: category(5), float32(4), int16(1)
memory usage: 192.9 KB
```

In [58]:

```
klib.mv_col_handling(df_train)
```

Out[58]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_y
0	9.300000	Low Fat	0.016047	Dairy	249.809204	1
1	5.920000	Regular	0.019278	Soft Drinks	48.269199	2
2	17.500000	Low Fat	0.016760	Meat	141.617996	1
3	19.200001	Regular	0.000000	Fruits and Vegetables	182.095001	1
4	8.930000	Low Fat	0.000000	Household	53.861401	1
...	...	...	...	...	...	
8518	6.865000	Low Fat	0.056783	Snack Foods	214.521805	1
8519	8.380000	Regular	0.046982	Baking Goods	108.156998	2
8520	10.600000	Low Fat	0.035186	Health and Hygiene	85.122398	2
8521	7.210000	Regular	0.145221	Snack Foods	103.133202	2
8522	14.800000	Low Fat	0.044878	Soft Drinks	75.467003	1

8523 rows × 10 columns

Preprocessing Task before Model Building

1) Label encoding

In [59]:

```
from sklearn.preprocessing import LabelEncoder
```

```
le=LabelEncoder()
```

In [60]:

```
df_train['item_fat_content']= le.fit_transform(df_train['item_fat_content'])
df_train['item_type']= le.fit_transform(df_train['item_type'])
df_train['outlet_size']= le.fit_transform(df_train['outlet_size'])
df_train['outlet_location_type']= le.fit_transform(df_train['outlet_location_type']
df_train['outlet_type']= le.fit_transform(df_train['outlet_type'])
```

In [61]:

```
df_train.head(5)
```

Out[61]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_year
0	9.300000	1	0.016047	4	249.809204	1999
1	5.920000	2	0.019278	14	48.269199	2009
2	17.500000	1	0.016760	10	141.617996	1999
3	19.200001	2	0.000000	6	182.095001	1998
4	8.930000	1	0.000000	9	53.861401	1987

2) Splitting our data into train and test files

In [62]:

```
X=df_train.drop('item_outlet_sales',axis=1)
```

In [63]:

```
Y=df_train['item_outlet_sales']
```

In [64]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X,Y, random_state=101, test_si
```

3)Standarization

In [65]:

```
X.describe()
```

Out[65]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishme
count	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000	8523
mean	12.857646	1.369354	0.066132	7.226681	140.992767	1997
std	4.226124	0.644810	0.051598	4.209990	62.275066	8
min	4.555000	0.000000	0.000000	0.000000	31.290001	1985
25%	9.310000	1.000000	0.026989	4.000000	93.826500	1987
50%	12.857645	1.000000	0.053931	6.000000	143.012802	1999
75%	16.000000	2.000000	0.094585	10.000000	185.643700	2004
max	21.350000	4.000000	0.328391	15.000000	266.888397	2009

In [66]:

```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
```

```
In [67]: X_train_std= sc.fit_transform(X_train)
```

```
In [68]: X_test_std= sc.transform(X_test)
```

```
In [69]: X_train_std
```

```
Out[69]: array([[ 1.52290023, -0.57382672,  0.68469731, ..., -1.95699503,
                  1.08786619, -0.25964107],
                [-1.239856  , -0.57382672, -0.09514746, ..., -0.28872895,
                 -0.13870429, -0.25964107],
                [ 1.54667619,  0.97378032, -0.0083859 , ..., -0.28872895,
                 -0.13870429, -0.25964107],
                ...,
                [-0.08197109, -0.57382672, -0.91916229, ...,  1.37953713,
                 -1.36527477, -0.25964107],
                [-0.74888436,  0.97378032,  1.21363045, ..., -0.28872895,
                 -0.13870429, -0.25964107],
                [ 0.67885675, -0.57382672,  1.83915361, ..., -0.28872895,
                  1.08786619,  0.98524841]])
```

```
In [70]: X_test_std
```

```
Out[70]: array([[-0.43860916, -0.57382672, -0.21609253, ..., -0.28872895,
                  1.08786619,  0.98524841],
                [ 1.22570184, -0.57382672, -0.52943464, ..., -1.95699503,
                  1.08786619, -0.25964107],
                [-1.2184578 ,  0.97378032,  0.16277341, ...,  1.37953713,
                 -1.36527477, -0.25964107],
                ...,
                [ 0.65508101, -0.57382672,  0.8782423 , ..., -0.28872895,
                  1.08786619, -1.50453056],
                [ 1.01171909, -0.57382672, -1.28409256, ..., -0.28872895,
                  1.08786619,  0.98524841],
                [-1.56558541,  0.97378032, -1.09265374, ..., -0.28872895,
                 -0.13870429, -0.25964107]])
```

```
In [71]: Y_train
```

```
Out[71]: 3684     163.786804
         1935    1607.241211
         5142    1510.034424
         4978    1784.343994
         2299    3558.035156
                    ...
         599     5502.836914
         5695    1436.796387
         8006    2167.844727
         1361    2700.484863
         1547     829.586792
         Name: item_outlet_sales, Length: 6818, dtype: float32
```

```
In [72]: Y_test
```

Out[72]:
```
8179     904.822205
8355    2795.694092
3411    1947.464966
7089     872.863770
6954    2450.144043
           ...
1317    1721.093018
4996     914.809204
531      370.184814
3891    1358.232056
6629    2418.185547
Name: item_outlet_sales, Length: 1705, dtype: float32
```

In [73]:
```
import joblib
```

In [74]:
```
joblib.dump(sc,r'D:\5th_semester\MiniProject2A\Projectworking\models\sc.sav')
```

Out[74]:
```
['D:\\5th_semester\\MiniProject2A\\Projectworking\\models\\sc.sav']
```

Model building

In [75]:
```
X_test.head()
```

Out[75]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_y
8179	11.000000	1	0.055163	8	100.335800	20
8355	18.000000	1	0.038979	13	148.641800	19
3411	7.720000	2	0.074731	1	77.598602	19
7089	20.700001	1	0.049035	6	39.950600	20
6954	7.550000	1	0.027225	3	152.934006	20

In [76]:
```
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
```

Linear Regression

In [77]:
```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
```

In [78]:
```
lr.fit(X_train_std,Y_train)
```

Out[78]:
```
▾ LinearRegression
LinearRegression()
```

In [79]:
```
Y_pred_lr=lr.predict(X_test_std)
```

In [80]:
```
print(r2_score(Y_test,Y_pred_lr))
print(mean_absolute_error(Y_test,Y_pred_lr))
print(np.sqrt(mean_squared_error(Y_test,Y_pred_lr)))
```

```
0.5041875773270634
880.99990440845
1162.4412631603452
```

In [81]:
```
joblib.dump(lr,r'D:\5th_semester\MiniProject2A\Projectworking\models\lr.sav')
```

Out[81]:
```
['D:\\5th_semester\\MiniProject2A\\Projectworking\\models\\lr.sav']
```

Random Forest Regressor

In [82]:
```
from sklearn.ensemble import RandomForestRegressor
rf= RandomForestRegressor(n_estimators=1000)
```

In [83]:
```
rf.fit(X_train_std,Y_train)
```

Out[83]:
```
▾ RandomForestRegressor
RandomForestRegressor(n_estimators=1000)
```

In [84]:
```
Y_pred_rf= rf.predict(X_test_std)
```

In [85]:
```
print(r2_score(Y_test,Y_pred_rf))
print(mean_absolute_error(Y_test,Y_pred_rf))
print(np.sqrt(mean_squared_error(Y_test,Y_pred_rf)))
```

```
0.5486175811867917
782.141215387397
1109.1355754589515
```

In [86]:
```
joblib.dump(rf,r'D:\5th_semester\MiniProject2A\Projectworking\models\rf.sav')
```

Out[86]:
```
['D:\\5th_semester\\MiniProject2A\\Projectworking\\models\\rf.sav']
```

XG Boost Regressor

In [87]:
```
from xgboost import XGBRegressor
xg= XGBRegressor()
```

In [88]:
```
xg.fit(X_train_std, Y_train)
```

Out[88]:
```
▾ XGBRegressor
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=
1,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwi
se',
             importance_type=None, interaction_constraints='',
             learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=
4,
```

In [89]:
```
Y_pred_xg= xg.predict(X_test_std)
```

In [90]:
```
print(r2_score(Y_test,Y_pred_xg))
print(mean_absolute_error(Y_test,Y_pred_xg))
print(np.sqrt(mean_squared_error(Y_test,Y_pred_xg)))
```

```
0.5313160637898305
800.45557
1130.1923
```

In [91]:
```
joblib.dump(rf,r'D:\5th_semester\MiniProject2A\Projectworking\models\xg.sav')
```

Out[91]:
```
['D:\\5th_semester\\MiniProject2A\\Projectworking\\models\\xg.sav']
```

Hyper parameter tuning

In [92]:
```
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV

# define models and parameters
model = RandomForestRegressor()
n_estimators = [10, 100, 1000]
max_depth=range(1,31)
min_samples_leaf=np.linspace(0.1, 1.0)
max_features=["auto", "sqrt", "log2"]
min_samples_split=np.linspace(0.1, 1.0, 10)

# define grid search
grid = dict(n_estimators=n_estimators)


grid_search_forest = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1,
                                  scoring='r2',error_score=0,verbose=2,cv=2)

grid_search_forest.fit(X_train_std, Y_train)

# summarize results
print(f"Best: {grid_search_forest.best_score_:.3f} using {grid_search_forest.best_
means = grid_search_forest.cv_results_['mean_test_score']
stds = grid_search_forest.cv_results_['std_test_score']
params = grid_search_forest.cv_results_['params']

for mean, stdev, param in zip(means, stds, params):
    print(f"{mean:.3f} ({stdev:.3f}) with: {param}")
```

```
Fitting 2 folds for each of 3 candidates, totalling 6 fits
Best: 0.549 using {'n_estimators': 1000}
0.514 (0.005) with: {'n_estimators': 10}
0.546 (0.004) with: {'n_estimators': 100}
0.549 (0.005) with: {'n_estimators': 1000}
```

In [93]:
```
grid_search_forest.best_params_
```

Out[93]:
```
{'n_estimators': 1000}
```

In [94]:
```
grid_search_forest.best_score_
```

Out[94]:
```
0.5493344344113504
```

In [95]:
```
Y_pred_rf_grid=grid_search_forest.predict(X_test_std)
```

```
In [96]: r2_score(Y_test,Y_pred_rf_grid)

Out[96]: 0.5489701766293793
```

Save the model

```
In [97]: import joblib

In [98]: joblib.dump(grid_search_forest,r'D:\5th_semester\MiniProject2A\Projectworking\rando

Out[98]: ['D:\\5th_semester\\MiniProject2A\\Projectworking\\random_forest_grid.sav']

In [99]: model=joblib.load(r'D:\5th_semester\MiniProject2A\Projectworking\random_forest_grid

In [100…  model.predict(X_test_std)

Out[100]: array([1675.38383487, 3578.52313513, 1277.31334682, ...,  395.41662216,
                 1662.80688269, 2422.13182642])

In [ ]:
```