Super-mart Sales Analytics

Importing The Dependencies

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from xgboost import XGBRegressor
from sklearn import metrics
```

Data Collection & Analysis

In [4]:
```
# loading the dataset from csv file to a pandas dataframe
super_mart_data = pd.read_csv('Test5.csv')
```

In [5]:
```
# first 5 rows of the dataframe
super_mart_data.head()
```

Out[5]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	FDW58	20.750	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Medium	Tier 1	Supermarket Type1	1636.244023
1	FDW14	8.300	reg	0.038428	Dairy	87.3198	OUT017	2007	small	Tier 2	Supermarket Type1	1409.355910
2	NCN55	14.600	Low Fat	0.099575	Others	241.7538	OUT010	1998	small	Tier 3	Grocery Store	710.594286
3	FDQ58	7.315	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	small	Tier 2	Supermarket Type1	2355.184955
4	FDY38	NaN	Regular	0.118599	Dairy	234.2300	OUT027	1985	Medium	Tier 3	Supermarket Type3	5857.916205

In [6]:
```
# number of data points and number of features
super_mart_data.shape
```

Out[6]:
```
(5681, 12)
```

In [7]:
```
# getting some information about the dataset
super_mart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5681 entries, 0 to 5680
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item_Identifier            5681 non-null   object
 1   Item_Weight                4705 non-null   float64
 2   Item_Fat_Content           5681 non-null   object
 3   Item_Visibility            5681 non-null   float64
 4   Item_Type                  5681 non-null   object
 5   Item_MRP                   5681 non-null   float64
 6   Outlet_Identifier          5681 non-null   object
 7   Outlet_Establishment_Year  5681 non-null   int64
 8   Outlet_Size                5681 non-null   object
 9   Outlet_Location_Type       5681 non-null   object
 10  Outlet_Type                5681 non-null   object
 11  Item_Outlet_Sales          5681 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 532.7+ KB
```

Categorical Features

In [8]:
```
# checking for missing values
super_mart_data.isnull().sum()
```

Out[8]:
```
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                    0
Outlet_Location_Type           0
Outlet_Type                    0
Item_Outlet_Sales              0
dtype: int64
```

Handling missing values

In [9]:
```
# mean value of "Item_Weight" column
super_mart_data['Item_Weight'].mean()
```

Out[9]:
```
12.695633368756374
```

In [10]:
```
# filling the missing values in 'Item_Weight' column with 'Mean' value
super_mart_data['Item_Weight'].fillna(super_mart_data['Item_Weight'].mean(), inplace = True)
```

In [11]:
```
#checking for missing values
super_mart_data.isnull().sum()
```

Out[11]:
```
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
```

Data Analysis

In [12]:
```
# statistical measures about the data
super_mart_data.describe()
```

Out[12]:

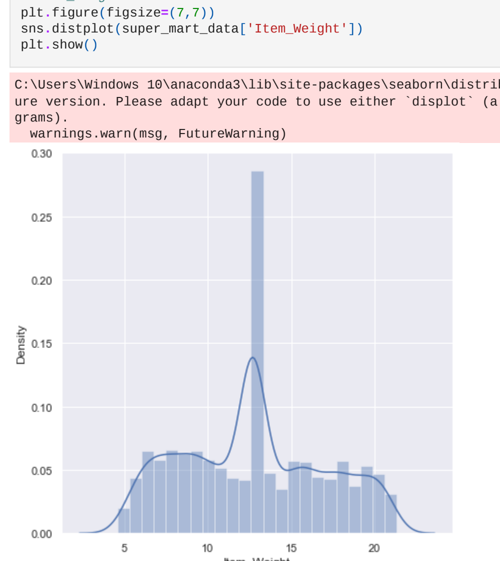
	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	5681.000000	5681.000000	5681.000000	5681.000000	5681.000000
mean	12.695633	0.065684	141.023273	1997.828903	2255.144238
std	4.245189	0.051252	61.809091	8.372256	1525.001339
min	4.555000	0.000000	31.990000	1985.000000	-100.129007
25%	9.195000	0.027047	94.412000	1987.000000	1175.989070
50%	12.695633	0.054154	141.415400	1999.000000	2008.295734
75%	15.850000	0.093463	186.026600	2004.000000	3208.434636
max	21.350000	0.323637	266.588400	2009.000000	46524.870000

Numerical Features

In [13]:
```
sns.set()
```

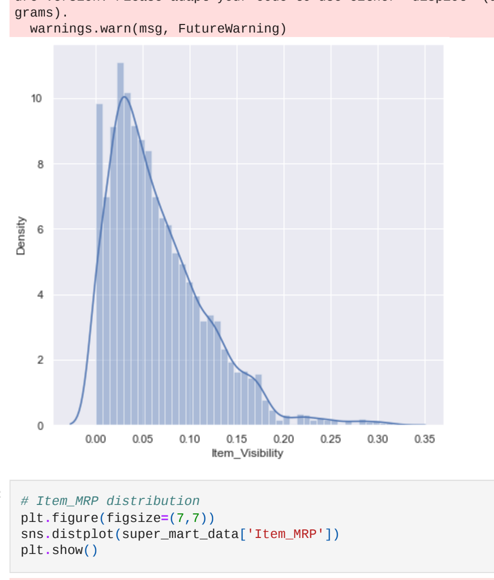
In [14]:
```
# Item_Weight distribution
plt.figure(figsize=(7,7))
sns.distplot(super_mart_data['Item_Weight'])
plt.show()
```

```
C:\Users\Windows 10\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



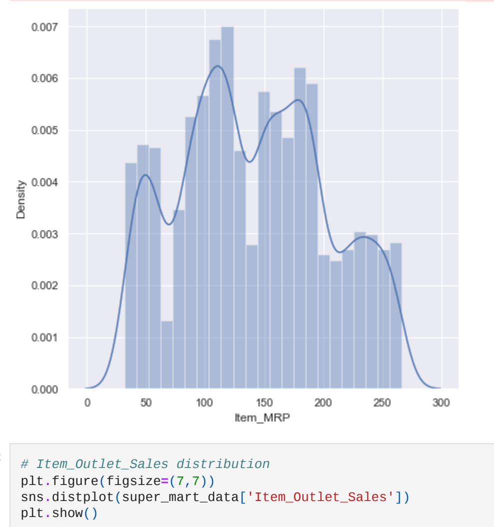
In [15]:
```
# Item_Visibility distribution
plt.figure(figsize=(7,7))
sns.distplot(super_mart_data['Item_Visibility'])
plt.show()
```

```
C:\Users\Windows 10\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



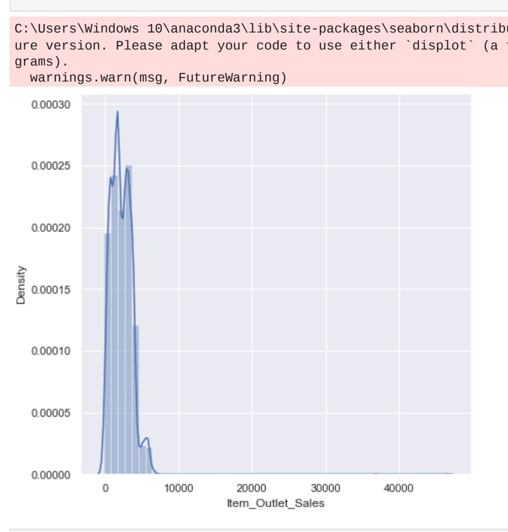
In [16]:
```
# Item_MRP distribution
plt.figure(figsize=(7,7))
sns.distplot(super_mart_data['Item_MRP'])
plt.show()
```

```
C:\Users\Windows 10\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

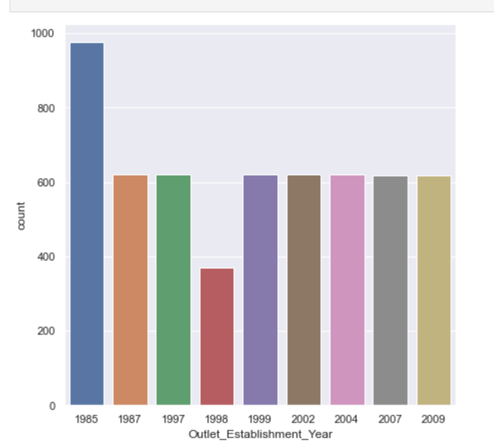


In [17]:
```
# Item_Outlet_Sales distribution
plt.figure(figsize=(7,7))
sns.distplot(super_mart_data['Item_Outlet_Sales'])
plt.show()
```

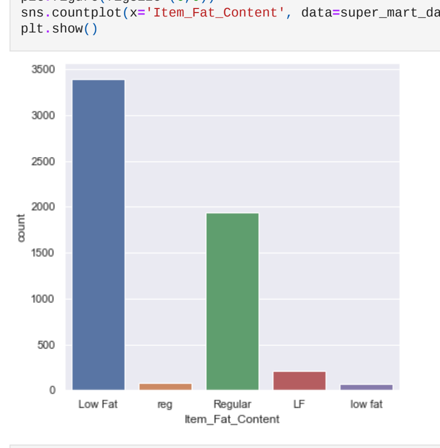
```
C:\Users\Windows 10\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



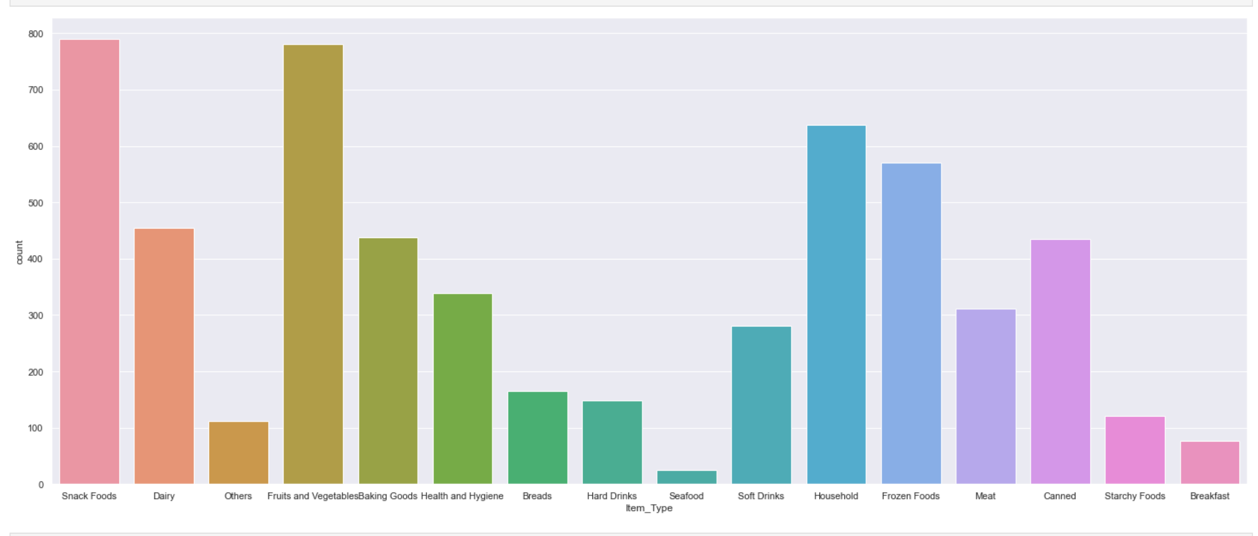
In [18]:
```
# Outlet_Establishment_Year column
plt.figure(figsize=(7,7))
sns.countplot(x='Outlet_Establishment_Year', data=super_mart_data)
plt.show()
```



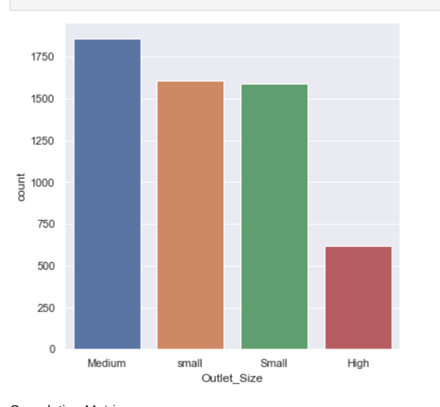
In [19]:
```
# Item_Fat_Content column
plt.figure(figsize=(6,6))
sns.countplot(x='Item_Fat_Content', data=super_mart_data)
plt.show()
```



In [20]:
```
# Item_Type column
plt.figure(figsize=(25,10))
sns.countplot(x='Item_Type', data=super_mart_data)
plt.show()
```



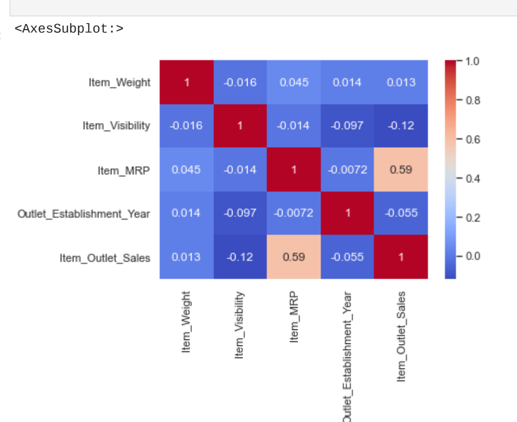
In [21]:
```
# Outlet_Size column
plt.figure(figsize=(6,6))
sns.countplot(x='Outlet_Size', data=super_mart_data)
plt.show()
```



Coorelation Matrix

In [22]:
```
corr = super_mart_data.corr()
sns.heatmap(corr, annot=True, cmap='coolwarm')
```

Out[22]:
```
<AxesSubplot:>
```



In [23]:
```
super_mart_data.head()
```

Out[23]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	FDW58	20.750000	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Medium	Tier 1	Supermarket Type1	1636.244023
1	FDW14	8.300000	reg	0.038428	Dairy	87.3198	OUT017	2007	small	Tier 2	Supermarket Type1	1409.355910
2	NCN55	14.600000	Low Fat	0.099575	Others	241.7538	OUT010	1998	small	Tier 3	Grocery Store	710.594286
3	FDQ58	7.315000	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	small	Tier 2	Supermarket Type1	2355.184955
4	FDY38	12.695633	Regular	0.118599	Dairy	234.2300	OUT027	1985	Medium	Tier 3	Supermarket Type3	5857.916205

In [24]:
```
super_mart_data['Item_Fat_Content'].value_counts()
```

Out[24]:
```
Low Fat    3396
Regular    1935
LF          206
reg          78
low fat      66
Name: Item_Fat_Content, dtype: int64
```

In [25]:
```
super_mart_data.replace({'Item_Fat_Content': {'low fat':'Low Fat','LF':'Low Fat', 'reg':'Regular'}}, inplace=True)
```

In [26]:
```
super_mart_data['Item_Fat_Content'].value_counts()
```

Out[26]:
```
Low Fat    3668
Regular    2013
Name: Item_Fat_Content, dtype: int64
```

Label Encoding

In [27]:
```
encoder = LabelEncoder()
```

In [28]:
```
super_mart_data['Item_Identifier'] = encoder.fit_transform(super_mart_data['Item_Identifier'])

super_mart_data['Item_Fat_Content'] = encoder.fit_transform(super_mart_data['Item_Fat_Content'])

super_mart_data['Item_Type'] = encoder.fit_transform(super_mart_data['Item_Type'])

super_mart_data['Outlet_Identifier'] = encoder.fit_transform(super_mart_data['Outlet_Identifier'])

super_mart_data['Outlet_Location_Type'] = encoder.fit_transform(super_mart_data['Outlet_Location_Type'])

super_mart_data['Outlet_Type'] = encoder.fit_transform(super_mart_data['Outlet_Type'])

super_mart_data['Outlet_Size'] = encoder.fit_transform(super_mart_data['Outlet_Size'])
```

In [29]:
```
super_mart_data.head()
```

Out[29]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
0	1103	20.750000	0	0.007565	13	107.8622	9	1999	1	0	1	1636.244023
1	1067	8.300000	1	0.038428	4	87.3198	2	2007	3	1	1	1409.355910
2	1406	14.600000	0	0.099575	11	241.7538	0	1998	3	2	0	710.594286
3	809	7.315000	0	0.015388	13	155.0340	2	2007	3	1	1	2355.184955
4	1184	12.695633	1	0.118599	4	234.2300	5	1985	1	2	3	5857.916205

Splitting features and Target

In [30]:
```
X = super_mart_data.drop(columns='Item_Outlet_Sales', axis=1)
Y = super_mart_data['Item_Outlet_Sales']
```

In [31]:
```
print(X)
```

```
      Item_Identifier  Item_Weight  Item_Fat_Content  Item_Visibility  \
0                1103    20.750000                 0         0.007565
1                1067     8.300000                 1         0.038428
2                1406    14.600000                 0         0.099575
3                 809     7.315000                 0         0.015388
4                1184    12.695633                 1         0.118599
...               ...          ...               ...              ...
5676              231    10.500000                 1         0.013496
5677              306     7.600000                 1         0.142991
5678             1412    10.000000                 0         0.073529
5679              517    15.300000                 1         0.000000
5680              987     9.500000                 1         0.104720

      Item_Type  Item_MRP  Outlet_Identifier  Outlet_Establishment_Year  \
0            13  107.8622                  9                       1999
1             4   87.3198                  2                       2007
2            11  241.7538                  0                       1998
3            13  155.0340                  2                       2007
4             4  234.2300                  5                       1985
...         ...       ...                ...                        ...
5676         13  141.3154                  8                       1997
5677         15  169.1448                  3                       2009
5678          8  118.7440                  7                       2002
5679          3  214.6218                  2                       2007
5680          3   79.7960                  7                       2002

      Outlet_Size  Outlet_Location_Type  Outlet_Type
0               1                     0            1
1               3                     1            1
2               3                     2            0
3               3                     1            1
4               1                     2            3
...           ...                   ...          ...
5676            2                     0            1
5677            1                     2            2
5678            3                     1            1
5679            3                     1            1
5680            3                     1            1

[5681 rows x 11 columns]
```

In [32]:
```
print(Y)
```

```
0       1636.244023
1       1409.355910
2        710.594286
3       2355.184955
4       5857.916205
           ...
5676    2125.984810
5677    2602.671833
5678    1832.451358
5679    3538.685188
5680    1281.144462
Name: Item_Outlet_Sales, Length: 5681, dtype: float64
```

Splitting the data into Training data anad Testing data

In [33]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

In [34]:
```
print(X.shape, X_train.shape, X_test.shape)
```

```
(5681, 11) (4544, 11) (1137, 11)
```

XGBoost Regressor

In [35]:
```
regressor = XGBRegressor()
```

In [36]:
```
regressor.fit(X_train, Y_train)
```

Out[36]:
```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.300000012,
             max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=100, n_jobs=4,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
```

Evaluation

In [38]:
```
# Prediction on training data
training_data_Prediction = regressor.predict(X_train)
```

In [40]:
```
# R squared Value
r2_train = metrics.r2_score(Y_train, training_data_Prediction)
```

In [41]:
```
print('R Squared value = ',r2_train)
```

```
R Squared value =  0.9932069865155904
```

In [42]:
```
# Prediction on test data
test_data_Prediction = regressor.predict(X_test)
```

In [44]:
```
# R Squared value
r2_test = metrics.r2_score(Y_test, test_data_Prediction)
```

In [45]:
```
print('R Squared value = ',r2_test)
```

```
R Squared value =  0.5848150919749066
```

In [ ]: