In [1]:
```
import xgboost
```

In [2]:
```
import pandas as pd
import numpy as np
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
import warnings
warnings.filterwarnings("ignore")
```

In [3]:
```
data = pd.read_csv("Train.csv")
```

In [4]:
```
data.head()
```

Out[4]:

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

In [5]:
```
data.shape
```

Out[5]:
```
(8523, 12)
```

In [6]:
```
data.info()
```

```
<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 [7]:
```
data.isnull().sum()
```

Out[7]:
```
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 [8]:
```
(data.Item_Weight.isnull().sum()/data.shape[0])*100
```

Out[8]:
```
17.165317376510618
```

In [9]:
```
(data.Outlet_Size.isnull().sum()/data.shape[0])*100
```

Out[9]:
```
28.27642848762173
```

In [10]:
```
data["Item_Weight"].mean()
```

Out[10]:
```
12.857645184136183
```

In [11]:
```
# filling null values
data["Item_Weight"].fillna(data["Item_Weight"].mean(),inplace =True)
```

In [12]:
```
data.isna().sum()
```

Out[12]:
```
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 [13]:
```
outlet_mode = data.pivot_table(values ='Outlet_Size', columns ='Outlet_Type', agg
```

In [14]:
```
print(outlet_mode)
```

```
Outlet_Type Grocery Store Supermarket Type1 Supermarket Type2  \
Outlet_Size         Small             Small            Medium

Outlet_Type Supermarket Type3
Outlet_Size            Medium
```

In [15]:
```
missing_values = data['Outlet_Size'].isnull()
```

In [16]:
```
print(missing_values)
```

```
0       False
1       False
2       False
3        True
4       False
        ...
8518    False
8519     True
8520    False
8521    False
8522    False
Name: Outlet_Size, Length: 8523, dtype: bool
```

In [17]:
```
data.loc[missing_values, 'Outlet_Size']= data.loc[missing_values, 'Outlet_Type'].
```

In [18]:
```
data.isna().sum() # checking again missing values on outlet size
```

Out[18]:
```
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 [19]:
```
data.describe()
```

Out[19]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.226124	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	9.310000	0.026989	93.826500	1987.000000	834.247400
50%	12.857645	0.053931	143.012800	1999.000000	1794.331000
75%	16.000000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

In [20]:
```
# checking categorical attributes
cat_col =[]
for a in data.dtypes.index:
    if data.dtypes[a] =='object':
        cat_col.append(a)
cat_col
```

Out[20]:
```
['Item_Identifier',
 'Item_Fat_Content',
 'Item_Type',
 'Outlet_Identifier',
 'Outlet_Size',
 'Outlet_Location_Type',
 'Outlet_Type']
```

In [21]:
```
# print categorical column
for col in cat_col:
    print(col)
    print(data[col].value_counts())
    print()
```

```
Item_Identifier
FDW13    10
FDG33    10
NCY18     9
FDD38     9
DRE49     9
         ..
FDY43     1
FDQ60     1
FDO33     1
DRF48     1
FDC23     1
Name: Item_Identifier, Length: 1559, dtype: int64

Item_Fat_Content
Low Fat    5089
Regular    2889
LF          316
reg         117
low fat     112
Name: Item_Fat_Content, dtype: int64

Item_Type
Fruits and Vegetables    1232
Snack Foods              1200
Household                 910
Frozen Foods              856
Dairy                     682
Canned                    649
Baking Goods              648
Health and Hygiene        520
Soft Drinks               445
Meat                      425
Breads                    251
Hard Drinks               214
Others                    169
Starchy Foods             148
Breakfast                 110
Seafood                    64
Name: Item_Type, dtype: int64

Outlet_Identifier
OUT027    935
OUT013    932
OUT049    930
OUT046    930
OUT035    930
OUT045    929
OUT018    928
OUT017    926
OUT010    555
```

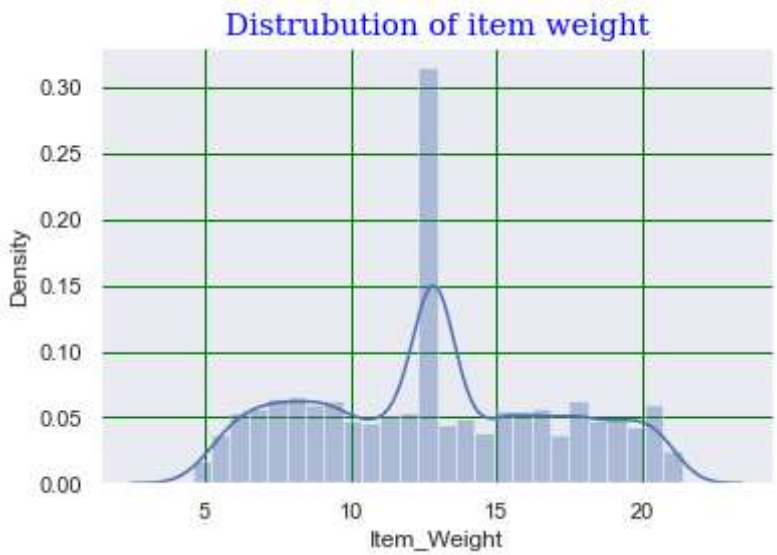
```
OUT019    528
Name: Outlet_Identifier, dtype: int64

Outlet_Size
Medium             2793
[Grocery Store]    2410
Small              2388
High                932
Name: Outlet_Size, dtype: int64

Outlet_Location_Type
Tier 3    3350
Tier 2    2785
Tier 1    2388
Name: Outlet_Location_Type, dtype: int64

Outlet_Type
Supermarket Type1    5577
Grocery Store        1083
Supermarket Type3     935
Supermarket Type2     928
Name: Outlet_Type, dtype: int64
```

In [22]:
```
data['Item_Type'].value_counts()
```

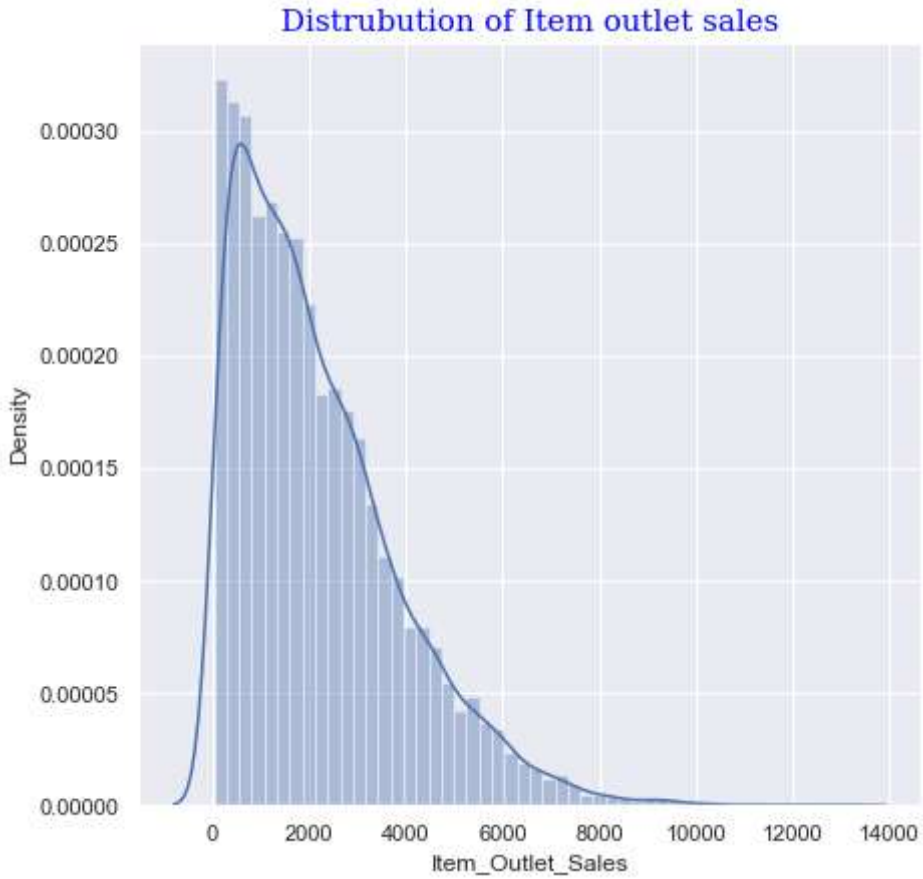
Out[22]:
```
Fruits and Vegetables    1232
Snack Foods              1200
Household                 910
Frozen Foods              856
Dairy                     682
Canned                    649
Baking Goods              648
Health and Hygiene        520
Soft Drinks               445
Meat                      425
Breads                    251
Hard Drinks               214
Others                    169
Starchy Foods             148
Breakfast                 110
Seafood                    64
Name: Item_Type, dtype: int64
```

In [23]:
```
# Data visualization
sns.set() # for some themes for our plots
```

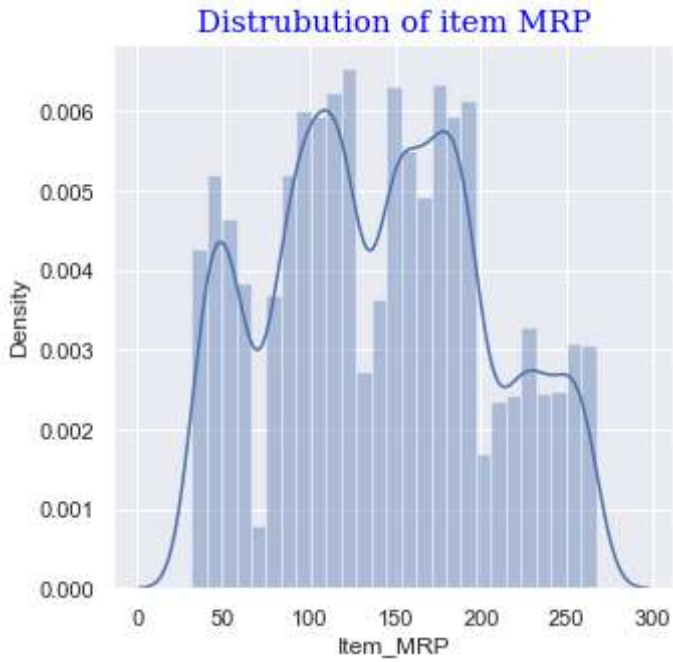
In [24]:
```
sns.distplot(data['Item_Weight'])
font1={'family':'serif', 'color':'blue','size':15}
plt.title('Distrubution of item weight', fontdict =font1)
plt.grid(color='green')
plt.show()
```



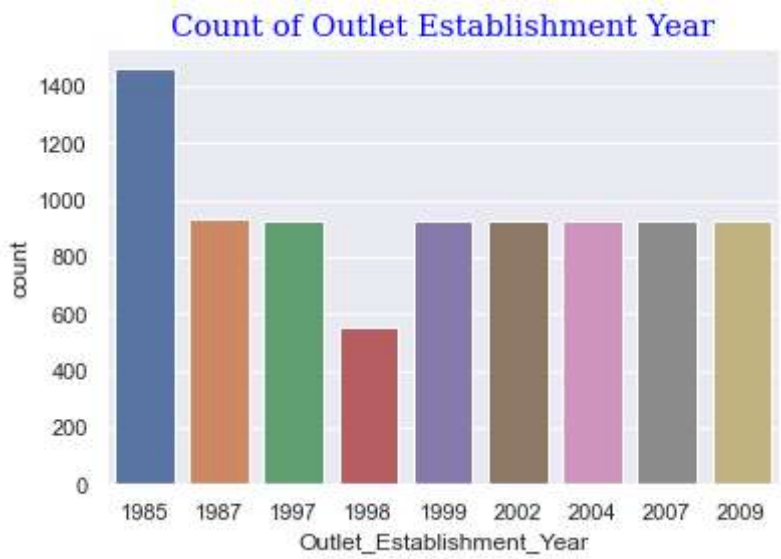
In [25]:
```
plt.figure(figsize =(7,7))
sns.distplot(data['Item_Outlet_Sales'])
font1={'family':'serif', 'color':'blue','size':15}
plt.title('Distrubution of Item outlet sales', fontdict=font1)
plt.show()
```



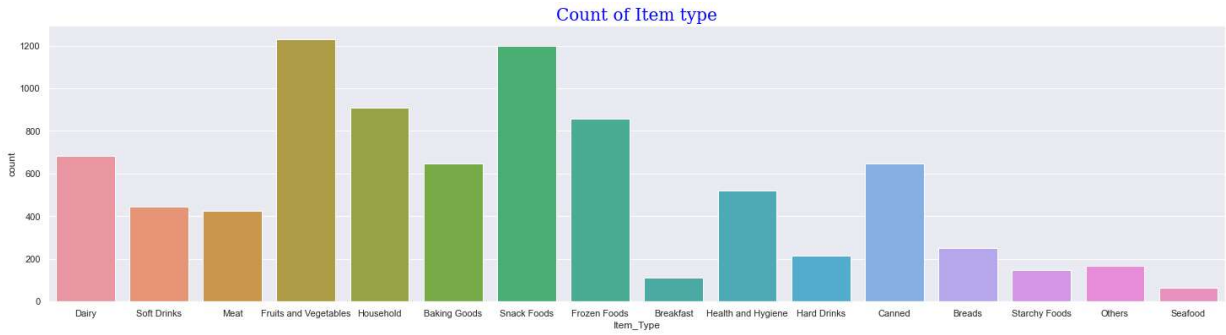
In [26]:
```
plt.figure(figsize =(5,5))
sns.distplot(data['Item_MRP'])
font1={'family':'serif', 'color':'blue','size':15}
plt.title('Distrubution of item MRP', fontdict=font1)
plt.show()
```



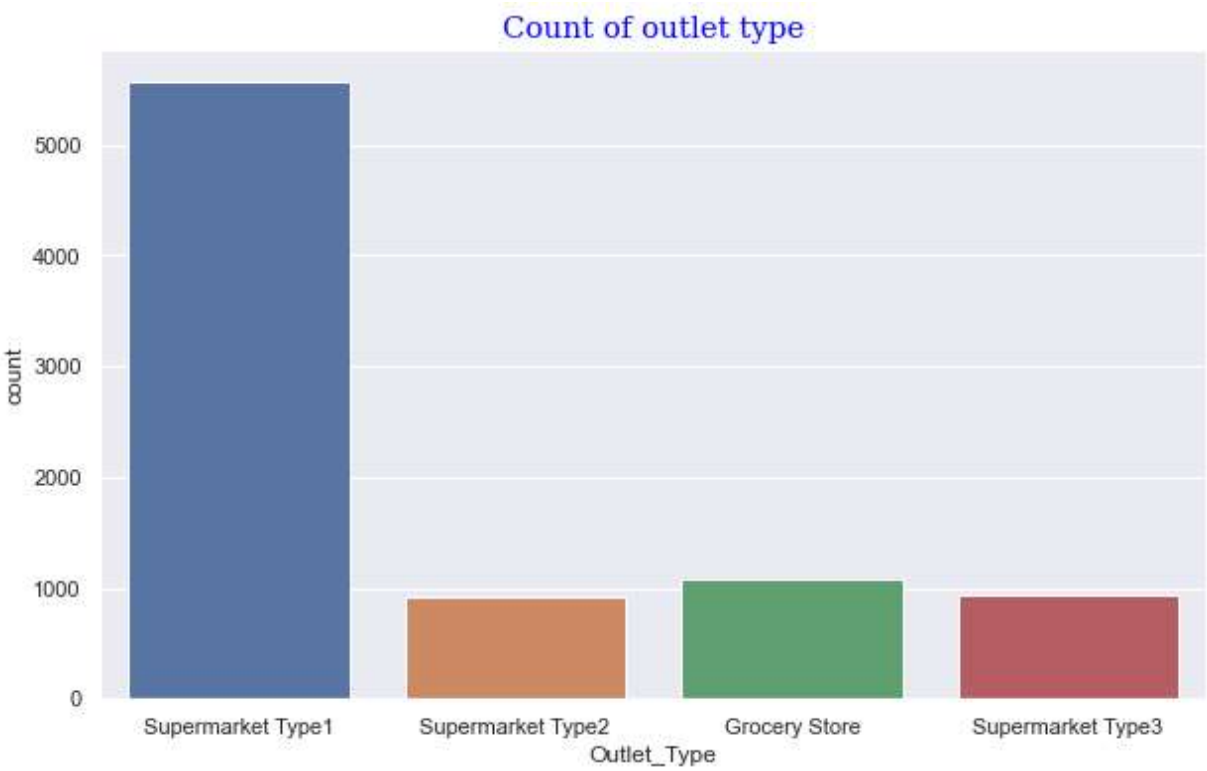
In [27]:
```
sns.countplot(x='Outlet_Establishment_Year', data= data)
font1={'family':'serif', 'color':'blue','size':15}
plt.title('Count of Outlet Establishment Year', fontdict =font1)
plt.show()
```



In [28]:
```
plt.figure(figsize=(25,6))
sns.countplot(x='Item_Type', data= data)
font1={'family':'serif', 'color':'blue','size':20}
plt.title('Count of Item type', fontdict =font1)
plt.show()
```



In [29]:
```
plt.figure(figsize=(10,6))
sns.countplot(x='Outlet_Type', data= data)
font1={'family':'serif', 'color':'blue','size':15}
plt.title('Count of outlet type', fontdict =font1)
plt.show()
```



In [30]:
```
# Data Processing
data.head()
```

Out[30]:

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

In [31]:
```
data['Item_Fat_Content'].value_counts()
```

Out[31]:
```
Low Fat    5089
Regular    2889
LF          316
reg         117
low fat     112
Name: Item_Fat_Content, dtype: int64
```

In [32]:
```
encoder = LabelEncoder()
```

In [33]:
```
data['Item_Identifier'] = encoder.fit_transform(data['Item_Identifier'])
data['Item_Fat_Content'] = encoder.fit_transform(data['Item_Fat_Content'])
data['Item_Type'] = encoder.fit_transform(data['Item_Type'])
data['Outlet_Identifier'] = encoder.fit_transform(data['Outlet_Identifier'])
data['Outlet_Size'] = encoder.fit_transform(data['Outlet_Size'].astype(str))
data['Outlet_Location_Type'] = encoder.fit_transform(data['Outlet_Location_Type']
data['Outlet_Type'] = encoder.fit_transform(data['Outlet_Type'])
```

In [34]:
```
data.head()
```

Out[34]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Iden
0	156	9.30	1	0.016047	4	249.8092	
1	8	5.92	2	0.019278	14	48.2692	
2	662	17.50	1	0.016760	10	141.6180	
3	1121	19.20	2	0.000000	6	182.0950	
4	1297	8.93	1	0.000000	9	53.8614	

In [35]:
```
x= data.drop(columns='Item_Outlet_Sales', axis= 1)
y= data['Item_Outlet_Sales']
print(x)
```

```
      Item_Identifier  Item_Weight  Item_Fat_Content  Item_Visibility  \
0                 156        9.300                 1         0.016047
1                   8        5.920                 2         0.019278
2                 662       17.500                 1         0.016760
3                1121       19.200                 2         0.000000
4                1297        8.930                 1         0.000000
...               ...          ...               ...              ...
8518              370        6.865                 1         0.056783
8519              897        8.380                 2         0.046982
8520             1357       10.600                 1         0.035186
8521              681        7.210                 2         0.145221
8522               50       14.800                 1         0.044878

      Item_Type  Item_MRP  Outlet_Identifier  Outlet_Establishment_Year  \
0             4  249.8092                  9                       1999
1            14   48.2692                  3                       2009
2            10  141.6180                  9                       1999
3             6  182.0950                  0                       1998
4             9   53.8614                  1                       1987
...         ...       ...                ...                        ...
8518         13  214.5218                  1                       1987
8519          0  108.1570                  7                       2002
8520          8   85.1224                  6                       2004
8521         13  103.1332                  3                       2009
8522         14   75.4670                  8                       1997

      Outlet_Size  Outlet_Location_Type  Outlet_Type
0               1                     0            1
1               1                     2            2
2               1                     0            1
3               2                     2            0
4               0                     2            1
...           ...                   ...          ...
8518            0                     2            1
8519            2                     1            1
8520            3                     1            1
8521            1                     2            2
8522            3                     0            1

[8523 rows x 11 columns]
```

In [36]:
```
#splitting data into train and test
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.1, random_state
```

In [37]:
```
print(x.shape, x_train.shape, x_test.shape, y_train.shape, y_test.shape)
```

```
(8523, 11) (7670, 11) (853, 11) (7670,) (853,)
```

In [38]:
```
data.head()
```

Out[38]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Iden
0	156	9.30	1	0.016047	4	249.8092	
1	8	5.92	2	0.019278	14	48.2692	
2	662	17.50	1	0.016760	10	141.6180	
3	1121	19.20	2	0.000000	6	182.0950	
4	1297	8.93	1	0.000000	9	53.8614	

In [39]:
```
# Machine Learning model

reg = XGBRegressor()
```

In [40]:
```
reg.fit(x_train,y_train)
```

Out[40]:
```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             n_estimators=100, n_jobs=None, num_parallel_tree=None,
             predictor=None, random_state=None, ...)
```

In [41]:
```
#predicting on training data
train_data_predict = reg.predict(x_train)
```

In [42]:
```
print(x)
```

```
      Item_Identifier  Item_Weight  Item_Fat_Content  Item_Visibility  \
0                 156        9.300                 1         0.016047
1                   8        5.920                 2         0.019278
2                 662       17.500                 1         0.016760
3                1121       19.200                 2         0.000000
4                1297        8.930                 1         0.000000
...               ...          ...               ...              ...
8518              370        6.865                 1         0.056783
8519              897        8.380                 2         0.046982
8520             1357       10.600                 1         0.035186
8521              681        7.210                 2         0.145221
8522               50       14.800                 1         0.044878

      Item_Type  Item_MRP  Outlet_Identifier  Outlet_Establishment_Year  \
0             4  249.8092                  9                       1999
1            14   48.2692                  3                       2009
2            10  141.6180                  9                       1999
3             6  182.0950                  0                       1998
4             9   53.8614                  1                       1987
...         ...       ...                ...                        ...
8518         13  214.5218                  1                       1987
8519          0  108.1570                  7                       2002
8520          8   85.1224                  6                       2004
8521         13  103.1332                  3                       2009
8522         14   75.4670                  8                       1997

      Outlet_Size  Outlet_Location_Type  Outlet_Type
0               1                     0            1
1               1                     2            2
2               1                     0            1
3               2                     2            0
4               0                     2            1
...           ...                   ...          ...
8518            0                     2            1
8519            2                     1            1
8520            3                     1            1
8521            1                     2            2
8522            3                     0            1

[8523 rows x 11 columns]
```

In [43]:
```
print(y)
```

```
0       3735.1380
1        443.4228
2       2097.2700
3        732.3800
4        994.7052
          ...
8518    2778.3834
8519     549.2850
8520    1193.1136
8521    1845.5976
8522     765.6700
Name: Item_Outlet_Sales, Length: 8523, dtype: float64
```

In [44]:
```
r2_train = metrics.r2_score(y_train, train_data_predict)
print("R squared value", r2_train)
```

```
R squared value 0.8440815663717178
```

In [45]:
```
#predict on test data
test_data_predict = reg.predict(x_test)
```

In [48]:
```
r2_test = metrics.r2_score(y_test, test_data_predict)
print("R squared value", r2_test)
```

```
R squared value 0.5470861782607719
```

In [54]:
```
input_data = (156,9.3000,1,0.016047,4,249.8092,9,1999,1,0,1)
input_data_as_numpy_array = np.asarray(input_data)
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = reg.predict(input_data_reshaped)
print(prediction)
print("initial value is",prediction[0])
print("sales for first product in the dataset is predicted as", prediction[0])
```

```
[4429.0503]
initial value is 4429.0503
sales for first product in the dataset is predicted as 4429.0503
```

In [ ]:

In [ ]: