In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
```

In [2]:
```
df=pd.read_csv("BIg_mart_sales_prediction (1).csv")
```

In [3]:
```
df
```

Out[3]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Locat
0	FDA15	9.300	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.920	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.500	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
3	FDX07	19.200	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	
4	NCD19	8.930	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	
...	...	...	...	...	...	...	...	...	...	
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	
8519	FDS36	8.380	Regular	0.046982	Baking Goods	108.1570	OUT045	2002	NaN	
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	

8523 rows × 12 columns

In [4]:
```
df.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 [5]:
```
df.isnull().sum()
```

Out[5]:
```
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 [6]:
```
df.dropna()
```

Out[6]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Locat
0	FDA15	9.300	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.920	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.500	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
4	NCD19	8.930	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	
5	FDP36	10.395	Regular	0.000000	Baking Goods	51.4008	OUT018	2009	Medium	
...	...	...	...	...	...	...	...	...	...	
8517	FDF53	20.750	reg	0.083607	Frozen Foods	178.8318	OUT046	1997	Small	
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	High	
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	Small	
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Medium	
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	Small	

4650 rows × 12 columns

In [7]:
```
df.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]:
```
nmean=df["Item_Weight"].mean()
df["Item_Weight"].fillna(nmean,inplace=True)
```

In [9]:
```
df.dropna(inplace=True)
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
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 [11]:
```
df.describe()
```

Out[11]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	6113.000000	6113.000000	6113.000000	6113.000000	6113.000000
mean	12.888856	0.064505	141.256859	1995.794373	2322.688445
std	4.073798	0.050092	62.229701	8.842615	1741.592093
min	4.555000	0.000000	31.290000	1985.000000	33.955800
25%	9.800000	0.026681	94.012000	1987.000000	974.731200
50%	12.857645	0.052811	143.178600	1997.000000	1928.156800
75%	15.700000	0.092834	185.892400	2004.000000	3271.075400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

In [12]:
```
print("Outlet_Size:\n", df.value_counts(), "\n\n")
print("Item_Weight:\n", df.value_counts(), "\n\n")
```

```
Outlet_Size:
 Item_Identifier  Item_Weight  Item_Fat_Content  Item_Visibility  Item_Type              Item_MRP  Outlet_Identifier  Outlet_Es
tablishment_Year  Outlet_Size  Outlet_Location_Type  Outlet_Type        Item_Outlet_Sales
DRA12             11.600000    LF                0.000000         Soft Drinks            141.9154  OUT035             2004
Small             Tier 2            Supermarket Type1  992.7078           1
FDV27             12.857645    Regular           0.070017         Meat                   89.3514   OUT019             1985
Small             Tier 1            Grocery Store      177.1028           1
FDV32             7.785000     Low Fat           0.089070         Fruits and Vegetables  62.7510   OUT018             2009
Medium            Tier 3            Supermarket Type2  1707.7770          1
                                                       0.088846         Fruits and Vegetables  61.4510   OUT049             1999
Medium            Tier 1            Supermarket Type1  759.0120           1
                                                       0.088692         Fruits and Vegetables  61.8510   OUT035             2004
Small             Tier 2            Supermarket Type1  1454.7730          1

..
FDJ33             8.895000     Regular           0.088305         Snack Foods            123.4730  OUT035             2004
Small             Tier 2            Supermarket Type1  1478.0760          1
FDJ32             12.857645    Low Fat           0.057512         Fruits and Vegetables  62.5536   OUT027             1985
Medium            Tier 3            Supermarket Type3  1592.5936          1
                  10.695000    Low Fat           0.057744         Fruits and Vegetables  61.2536   OUT013             1987
High              Tier 3            Supermarket Type1  673.7896           1
                               LF                0.057792         Fruits and Vegetables  61.4536   OUT046             1997
Small             Tier 1            Supermarket Type1  428.7752           1
NCZ54             14.650000    Low Fat           0.083699         Household              163.4552  OUT018             2009
Medium            Tier 3            Supermarket Type2  2599.2832          1
Length: 6113, dtype: int64


Item_Weight:
 Item_Identifier  Item_Weight  Item_Fat_Content  Item_Visibility  Item_Type              Item_MRP  Outlet_Identifier  Outlet_Es
tablishment_Year  Outlet_Size  Outlet_Location_Type  Outlet_Type        Item_Outlet_Sales
DRA12             11.600000    LF                0.000000         Soft Drinks            141.9154  OUT035             2004
Small             Tier 2            Supermarket Type1  992.7078           1
FDV27             12.857645    Regular           0.070017         Meat                   89.3514   OUT019             1985
Small             Tier 1            Grocery Store      177.1028           1
FDV32             7.785000     Low Fat           0.089070         Fruits and Vegetables  62.7510   OUT018             2009
Medium            Tier 3            Supermarket Type2  1707.7770          1
                                                       0.088846         Fruits and Vegetables  61.4510   OUT049             1999
Medium            Tier 1            Supermarket Type1  759.0120           1
                                                       0.088692         Fruits and Vegetables  61.8510   OUT035             2004
Small             Tier 2            Supermarket Type1  1454.7730          1

..
FDJ33             8.895000     Regular           0.088305         Snack Foods            123.4730  OUT035             2004
Small             Tier 2            Supermarket Type1  1478.0760          1
FDJ32             12.857645    Low Fat           0.057512         Fruits and Vegetables  62.5536   OUT027             1985
Medium            Tier 3            Supermarket Type3  1592.5936          1
                  10.695000    Low Fat           0.057744         Fruits and Vegetables  61.2536   OUT013             1987
High              Tier 3            Supermarket Type1  673.7896           1
                               LF                0.057792         Fruits and Vegetables  61.4536   OUT046             1997
Small             Tier 1            Supermarket Type1  428.7752           1
NCZ54             14.650000    Low Fat           0.083699         Household              163.4552  OUT018             2009
Medium            Tier 3            Supermarket Type2  2599.2832          1
Length: 6113, dtype: int64
```

In [13]:
```
df['Item_Type']=df['Item_Type'].replace(to_replace =['Dairy','Baking Goods','Meat' ,'Breads',
                                                     'Starchy Foods','Breakfast','Fruits and Vegetables'] ,
                                        value = 'Household')


df['Item_Type']=df['Item_Type'].replace(to_replace =['Seafood' , 'Frozen Foods' , 'Canned'] ,
                                        value = 'Snack Foods')


df['Item_Type']=df['Item_Type'].replace(to_replace =['Soft Drinks' , 'Hard Drinks','Health and Hygiene'] ,
                                        value = 'Others')
```

In [14]:
```
df.Outlet_Type.value_counts()
```

Out[14]:
```
Supermarket Type1    3722
Supermarket Type3     935
Supermarket Type2     928
Grocery Store         528
Name: Outlet_Type, dtype: int64
```

In [15]:
```
df.shape
```

Out[15]: (6113, 12)

In [16]:
```
fat= pd.get_dummies(df['Item_Fat_Content'],drop_first=True)
item= pd.get_dummies(df['Item_Type'],drop_first=True)
loc= pd.get_dummies(df['Outlet_Location_Type'],drop_first=True)
size= pd.get_dummies(df['Outlet_Size'],drop_first=True)
out_type= pd.get_dummies(df['Outlet_Type'],drop_first=True)
```

In [17]:
```
new_df = pd.concat([df,fat , item , loc , size , out_type] ,axis = 1)
new_df.head()
```

Out[17]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_
0	FDA15	9.300	Low Fat	0.016047	Household	249.8092	OUT049	1999	Medium	
1	DRC01	5.920	Regular	0.019278	Others	48.2692	OUT018	2009	Medium	
2	FDN15	17.500	Low Fat	0.016760	Household	141.6180	OUT049	1999	Medium	
4	NCD19	8.930	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	
5	FDP36	10.395	Regular	0.000000	Household	51.4008	OUT018	2009	Medium	

5 rows × 25 columns

In [18]:
```
new_df.drop(['Item_Fat_Content','Item_Type','Outlet_Size','Outlet_Location_Type','Outlet_Type'] , axis = 1 , inplace = True)
```

In [19]:
```
new_df.head()
```

Out[19]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Item_Outlet_Sales	Low Fat	Regular	low fat	reg	Others	Snacl Foods
0	FDA15	9.300	0.016047	249.8092	OUT049	1999	3735.1380	1	0	0	0	0	0
1	DRC01	5.920	0.019278	48.2692	OUT018	2009	443.4228	0	1	0	0	1	0
2	FDN15	17.500	0.016760	141.6180	OUT049	1999	2097.2700	1	0	0	0	0	0
4	NCD19	8.930	0.000000	53.8614	OUT013	1987	994.7052	1	0	0	0	0	0
5	FDP36	10.395	0.000000	51.4008	OUT018	2009	556.6088	0	1	0	0	0	0

In [ ]:
```
```

In [20]:
```
x = df.drop(['Item_Identifier' ,'Outlet_Identifier' , 'Item_Outlet_Sales','Item_Fat_Content','Outlet_Size','Item_Type','Outlet_Lo
```

In [21]:
```
x
```

Out[21]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year
0	9.300	0.016047	249.8092	1999
1	5.920	0.019278	48.2692	2009
2	17.500	0.016760	141.6180	1999
4	8.930	0.000000	53.8614	1987
5	10.395	0.000000	51.4008	2009
...	...	...	...	...
8517	20.750	0.083607	178.8318	1997
8518	6.865	0.056783	214.5218	1987
8520	10.600	0.035186	85.1224	2004
8521	7.210	0.145221	103.1332	2009
8522	14.800	0.044878	75.4670	1997

6113 rows × 4 columns

In [22]:
```
y = df['Item_Outlet_Sales']
```

In [23]:
```
y
```

Out[23]:
```
0       3735.1380
1        443.4228
2       2097.2700
4        994.7052
5        556.6088
           ...
8517    3608.6360
8518    2778.3834
8520    1193.1136
8521    1845.5976
8522     765.6700
Name: Item_Outlet_Sales, Length: 6113, dtype: float64
```

In [24]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
```

In [25]:
```
# Fitting Multiple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[25]:
```
LinearRegression()
```

In [26]:
```
# Predicting the Test set results
y_pred = regressor.predict(x_test)
```

In [27]:
```
#evaluate the model
from sklearn.metrics import r2_score
```

In [28]:
```
r2_score(y_test,y_pred)
```

Out[28]:
```
0.38821398166373433
```

In [29]:
```
#loss function
from sklearn.metrics import mean_absolute_error
```

In [30]:
```
mean_absolute_error(y_test,y_pred)
```

Out[30]:
```
994.1835160595947
```

In [31]:
```
from sklearn.metrics import mean_squared_error
```

In [32]:
```
mean_squared_error(y_test,y_pred)
```

Out[32]:
```
1879407.881292865
```

In [33]:
```
np.sqrt(mean_squared_error(y_test,y_pred))
```

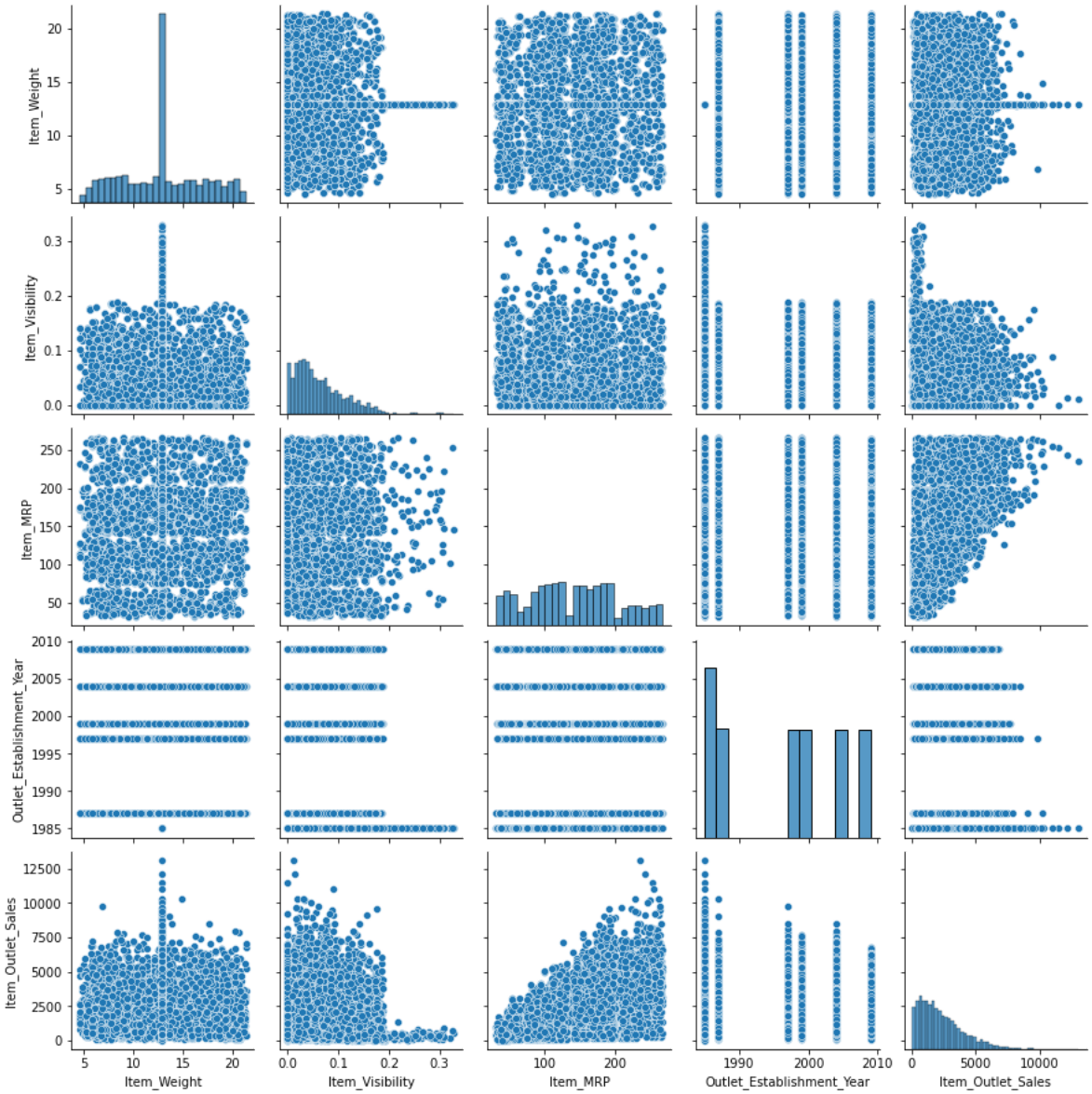
Out[33]: 1370.9149796004365

In [34]:
```
r2_score(y_test,y_pred)
```

Out[34]: 0.38821398166373433

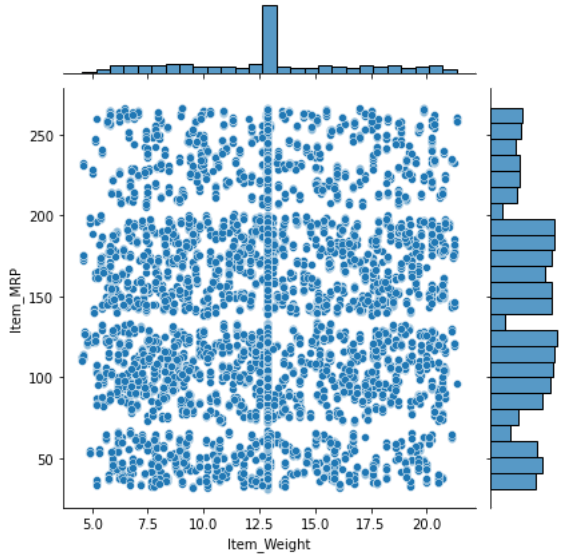
In [35]:
```
sns.pairplot(df)
```

Out[35]: <seaborn.axisgrid.PairGrid at 0x22e44de6ac0>



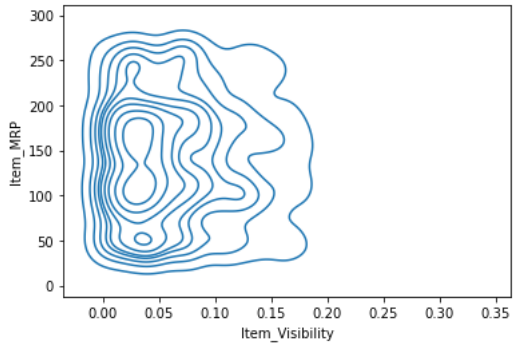
In [36]:
```
# Joint Distribution Plot
sns.jointplot(x='Item_Weight', y='Item_MRP', data=df)
```

Out[36]:
```
<seaborn.axisgrid.JointGrid at 0x22e46d01880>
```



In [37]:
```
sns.kdeplot(df["Item_Visibility"], df["Item_MRP"])
```

Out[37]:
```
<AxesSubplot:xlabel='Item_Visibility', ylabel='Item_MRP'>
```



In [38]:
```
pkmn_type_colors = ['#78C850',  # Grass
                    '#F08030',  # Fire
                    '#6890F0',  # Water
                    '#A8B820',  # Bug
                    '#A8A878',  # Normal
                    '#A040A0',  # Poison
                    '#F8D030',  # Electric
                    '#E0C068',  # Ground
                    '#EE99AC',  # Fairy
                    '#C03028',  # Fighting
                    '#F85888',  # Psychic
                    '#B8A038',  # Rock
                    '#705898',  # Ghost
                    '#98D8D8',  # Ice
                    '#7038F8',  # Dragon
                   ]
```

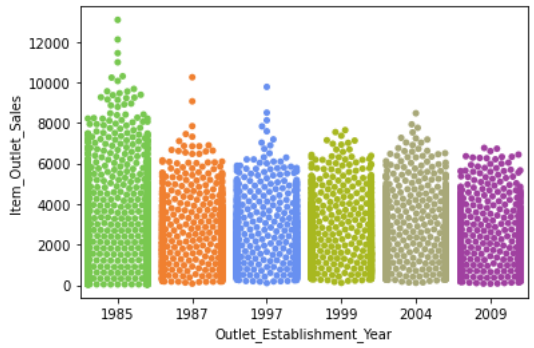
In [39]:
```
# Count Plot (a.k.a. Bar Plot)
sns.countplot(x='Item_Weight', data=df, palette=pkmn_type_colors)

# Rotate x-labels
plt.xticks(rotation=-90)
```

Out[39]:
```
(array([  0,   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,
         13,  14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,
         26,  27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,
         39,  40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,
         52,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,
         65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,
         78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
         91,  92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103,
        104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116,
        117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
        130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142,
        143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155,
        156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168,
        169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181,
        182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194,
        195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207,
        208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220,
        221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233,
        234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246,
```

In [40]:
```
# Swarm plot with Pokemon color palette
sns.swarmplot(x='Outlet_Establishment_Year', y='Item_Outlet_Sales', data=df, palette=pkmn_type_colors)
```

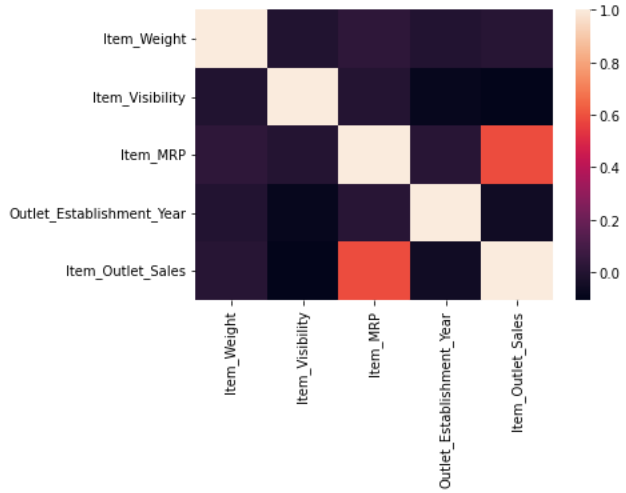
Out[40]:
```
<AxesSubplot:xlabel='Outlet_Establishment_Year', ylabel='Item_Outlet_Sales'>
```



In [41]:
```
# Calculate correlations
corr = df.corr()

# Heatmap
sns.heatmap(corr)
```

Out[41]:
```
<AxesSubplot:>
```



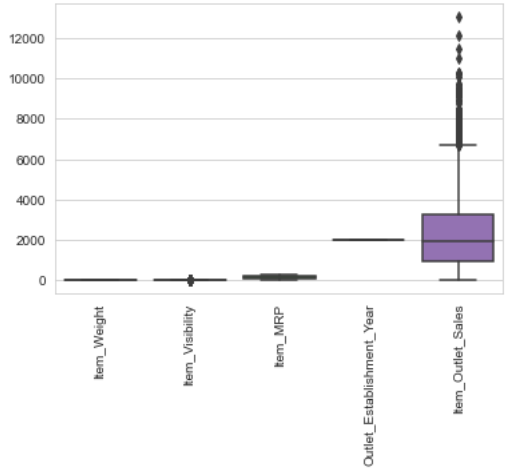
In [42]:
```
# Set theme
sns.set_style('whitegrid')

# Violin plot
sns.violinplot(x='Item_Weight', y='Item_Outlet_Sales', data=df)
plt.xticks(rotation=90)
```

```
  Text(34, 0, '5.51'),
  Text(35, 0, '5.59'),
  Text(36, 0, '5.615'),
  Text(37, 0, '5.63'),
  Text(38, 0, '5.635'),
  Text(39, 0, '5.655'),
  Text(40, 0, '5.675'),
  Text(41, 0, '5.695'),
  Text(42, 0, '5.73'),
  Text(43, 0, '5.735'),
  Text(44, 0, '5.75'),
  Text(45, 0, '5.765'),
  Text(46, 0, '5.78'),
  Text(47, 0, '5.785'),
  Text(48, 0, '5.8'),
  Text(49, 0, '5.82'),
  Text(50, 0, '5.825'),
  Text(51, 0, '5.845'),
  Text(52, 0, '5.86'),
```

In [43]:
```
# Boxplot
sns.boxplot(data=df)
plt.xticks(rotation=90)
```

Out[43]:
```
(array([0, 1, 2, 3, 4]),
 [Text(0, 0, 'Item_Weight'),
  Text(1, 0, 'Item_Visibility'),
  Text(2, 0, 'Item_MRP'),
  Text(3, 0, 'Outlet_Establishment_Year'),
  Text(4, 0, 'Item_Outlet_Sales')])
```



In [44]:
```
import pandas_profiling as pp
```

In [45]:
```
pp.ProfileReport(df)
```

Overview

Dataset statistics

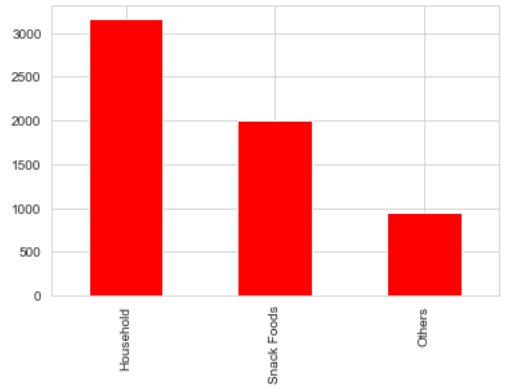
Number of variables	12
Number of observations	6113
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%

Variable types

Categorical	7
Numeric	5

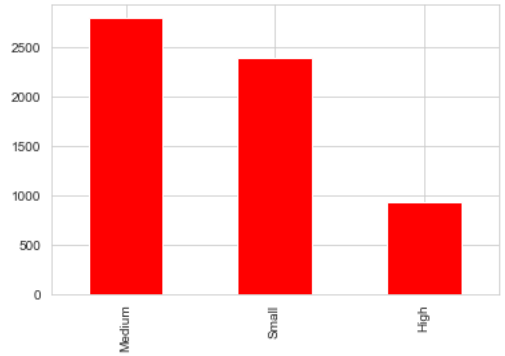
In [46]:
```
df.Item_Type.value_counts().plot(kind='bar',color='red')
```

Out[46]: <AxesSubplot:>



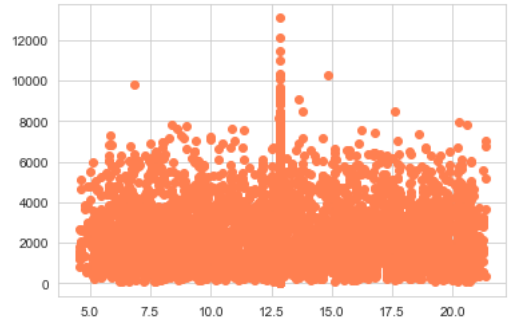
In [47]:
```
df.Outlet_Size.value_counts().plot(kind='bar',color='red')
```

Out[47]: <AxesSubplot:>



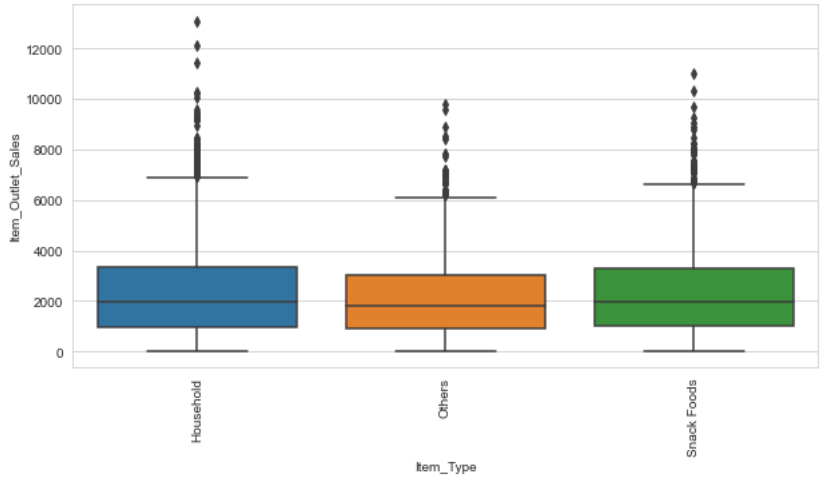
In [48]:
```
#Item wt vs Sales:
plt.scatter(df.Item_Weight,df.Item_Outlet_Sales,color='coral')
#No pattern
```

Out[48]: <matplotlib.collections.PathCollection at 0x22e57336220>



In [49]:
```
# Item type vs Item outlet sales:
plt.figure(figsize=[10,5])
sns.boxplot(x='Item_Type',y='Item_Outlet_Sales',data=df)
plt.xticks(rotation=90)
```

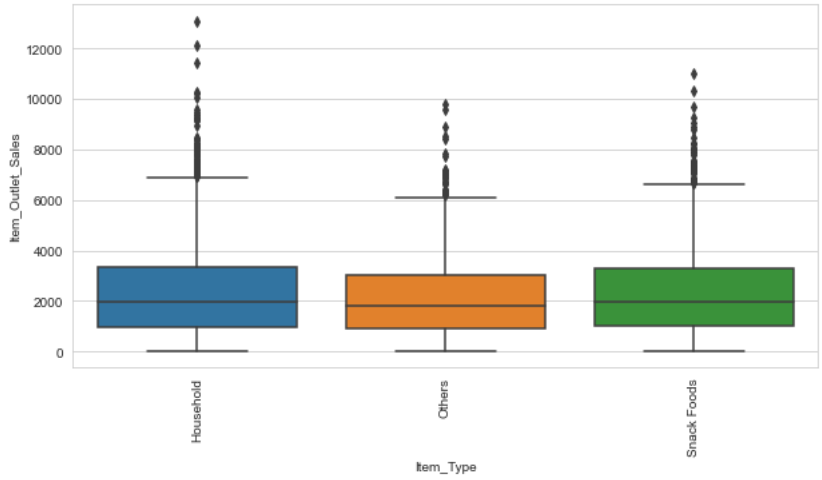
Out[49]:
```
(array([0, 1, 2]),
 [Text(0, 0, 'Household'), Text(1, 0, 'Others'), Text(2, 0, 'Snack Foods')])
```



In [ ]:

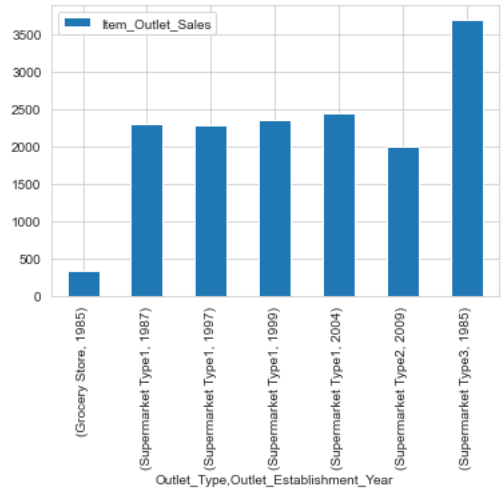
In [50]:
```
# Item type vs Item outlet sales:
plt.figure(figsize=[10,5])
sns.boxplot(x='Item_Type',y='Item_Outlet_Sales',data=df)
plt.xticks(rotation=90)
```

Out[50]:
```
(array([0, 1, 2]),
 [Text(0, 0, 'Household'), Text(1, 0, 'Others'), Text(2, 0, 'Snack Foods')])
```



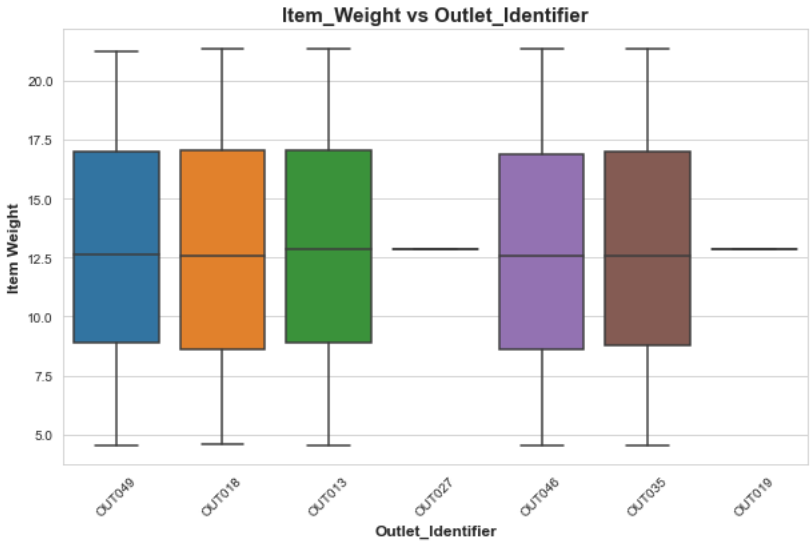
In [51]:
```
df.groupby(['Outlet_Type','Outlet_Establishment_Year']).agg({'Item_Outlet_Sales':np.mean}).plot.bar()
X=plt.gca().xaxis

# Rotating the Xicklabels
for item in X.get_ticklabels():
  item.set_rotation(90)
```



In [52]:
```
plt.figure(figsize = (10,6))
sns.boxplot(x='Outlet_Identifier',y='Item_Weight',data = df)
X=plt.gca().xaxis

# Rotating the Xicklabels
for item in X.get_ticklabels():
  item.set_rotation(45)
plt.xlabel('Outlet_Identifier', fontsize = 12, fontweight = 'bold')
plt.ylabel('Item Weight', fontsize = 12, fontweight = 'bold')
plt.title('Item_Weight vs Outlet_Identifier',fontsize = 15, fontweight = 'bold')
plt.show()
```

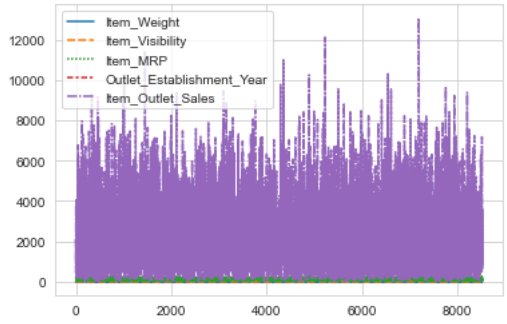


In [53]:
```
df['Outlet_Identifier'].unique()
```

Out[53]:
```
array(['OUT049', 'OUT018', 'OUT013', 'OUT027', 'OUT046', 'OUT035',
       'OUT019'], dtype=object)
```

In [54]:
```
sns.lineplot(data=df)
```

Out[54]: <AxesSubplot:>



In [55]:
```

```

In [56]:
```
df.duplicated().sum()
```

Out[56]: 0

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```