problem defining

The data scientists at BigMart have collected 2013 sales data for 1559 products across 10 stores in different cities. Also, certain attributes of each product and store have been defined. The aim is to build a predictive model and predict the sales of each product at a particular outlet.

Using this model, BigMart will try to understand the properties of products and outlets which play a key role in increasing sales.

Please note that the data may have missing values as some stores might not report all the data due to technical glitches. Hence, it will be required to treat them accordingly.

In [302…
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [303…
```
data1=pd.read_csv('train.csv')
data2=pd.read_csv('test.csv')
```

In [304…
```
data1.head()
```

Out[304]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	

In [305…
```
data2.head()
```

Out[305]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_
0	FDW58	20.750	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Medium	
1	FDW14	8.300	reg	0.038428	Dairy	87.3198	OUT017	2007	NaN	
2	NCN55	14.600	Low Fat	0.099575	Others	241.7538	OUT010	1998	NaN	
3	FDQ58	7.315	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	NaN	
4	FDY38	NaN	Regular	0.118599	Dairy	234.2300	OUT027	1985	Medium	

In [306…
```
data1['source']='train'
data2['source']='test'
```

In [307…
```
data=pd.concat([data1,data2],ignore_index=True)
```

In [308…
```
data.shape,data1.shape,data2.shape
```

Out[308]:
```
((14204, 13), (8523, 13), (5681, 12))
```

In [309…
```
data.dtypes
```

Out[309]:
```
Item_Identifier               object
Item_Weight                  float64
Item_Fat_Content              object
Item_Visibility              float64
Item_Type                     object
Item_MRP                     float64
Outlet_Identifier             object
Outlet_Establishment_Year      int64
Outlet_Size                   object
Outlet_Location_Type          object
Outlet_Type                   object
Item_Outlet_Sales            float64
source                        object
dtype: object
```

In [310…
```
data.isnull().sum()
```

Out[310]:
```
Item_Identifier               0
Item_Weight                2439
Item_Fat_Content              0
Item_Visibility               0
Item_Type                     0
Item_MRP                      0
Outlet_Identifier             0
Outlet_Establishment_Year     0
Outlet_Size                4016
Outlet_Location_Type          0
Outlet_Type                   0
Item_Outlet_Sales          5681
source                        0
dtype: int64
```

In [311…
```
data['Item_Weight'].value_counts()
```

Out[311]:
```
17.600    135
12.150    127
10.500    123
13.650    115
11.800    113
          ...
7.640       7
5.905       7
7.850       6
4.615       6
9.035       6
Name: Item_Weight, Length: 415, dtype: int64
```

In [312…
```
data['Item_Weight'].fillna(data['Item_Weight'].mean(),inplace=True)
```

In [313…
```
data['Item_Weight'].isnull().sum()
```

Out[313]:
```
0
```

In [314…
```
data['Outlet_Size'].value_counts()
```

Out[314]:
```
Medium    4655
Small     3980
High      1553
Name: Outlet_Size, dtype: int64
```

In [315…
```
data['Outlet_Size'].fillna('Medium',inplace=True)
```

In [316…
```
data['Outlet_Size'].isnull().sum()
```

Out[316]:
```
0
```

In [317…
```
data.isnull().sum()
```

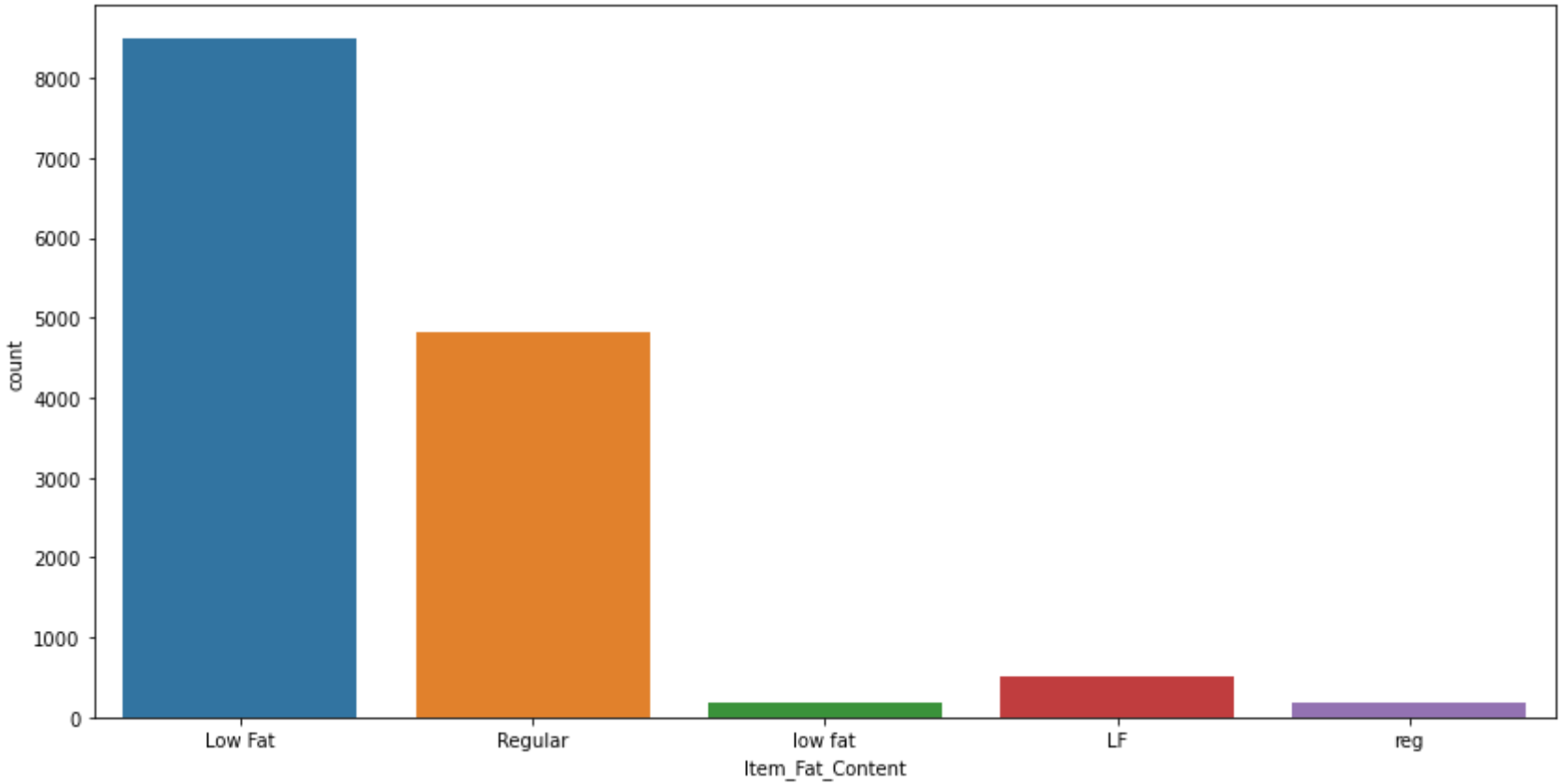
Out[317]:
```
Item_Identifier                 0
Item_Weight                     0
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                     0
Outlet_Location_Type            0
Outlet_Type                     0
Item_Outlet_Sales            5681
source                          0
dtype: int64
```

In [318…
```
data.select_dtypes(include=['object']).columns
```

Out[318]:
```
Index(['Item_Identifier', 'Item_Fat_Content', 'Item_Type', 'Outlet_Identifier',
       'Outlet_Size', 'Outlet_Location_Type', 'Outlet_Type', 'source'],
      dtype='object')
```

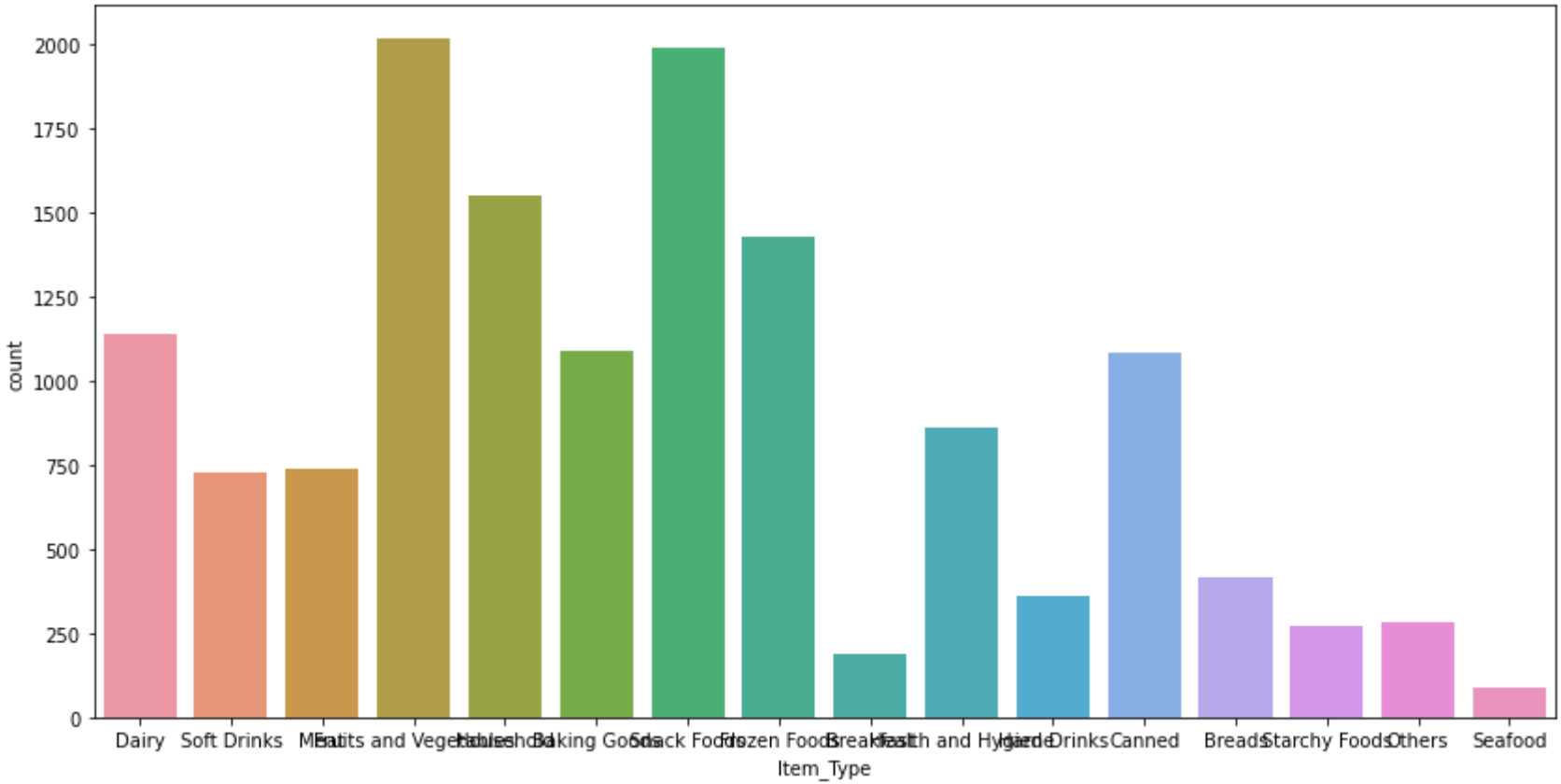
In [319…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Item_Fat_Content'])
```

Out[319]:
```
<AxesSubplot:xlabel='Item_Fat_Content', ylabel='count'>
```



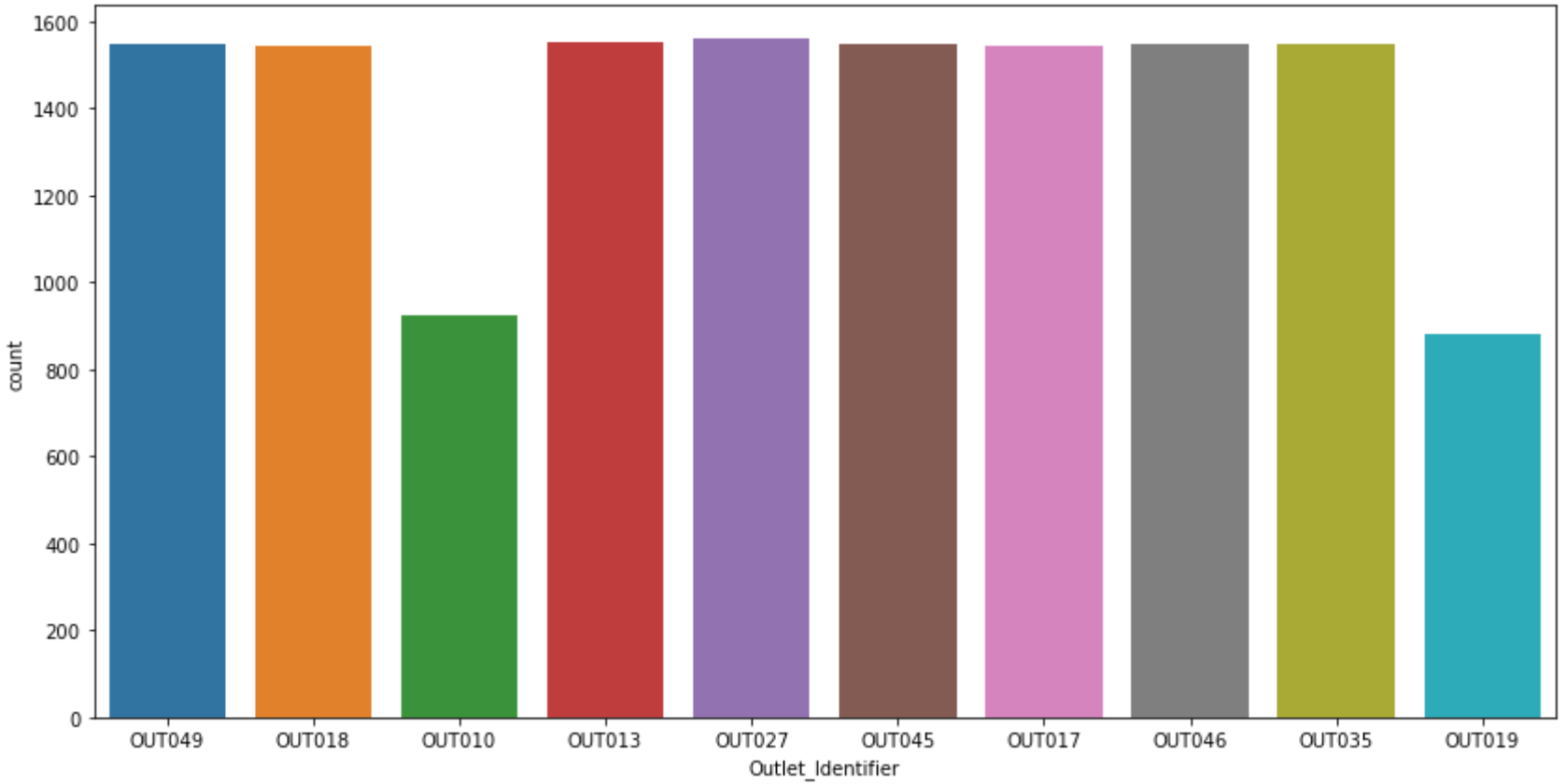
In [320…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Item_Type'])
```

Out[320]:
```
<AxesSubplot:xlabel='Item_Type', ylabel='count'>
```



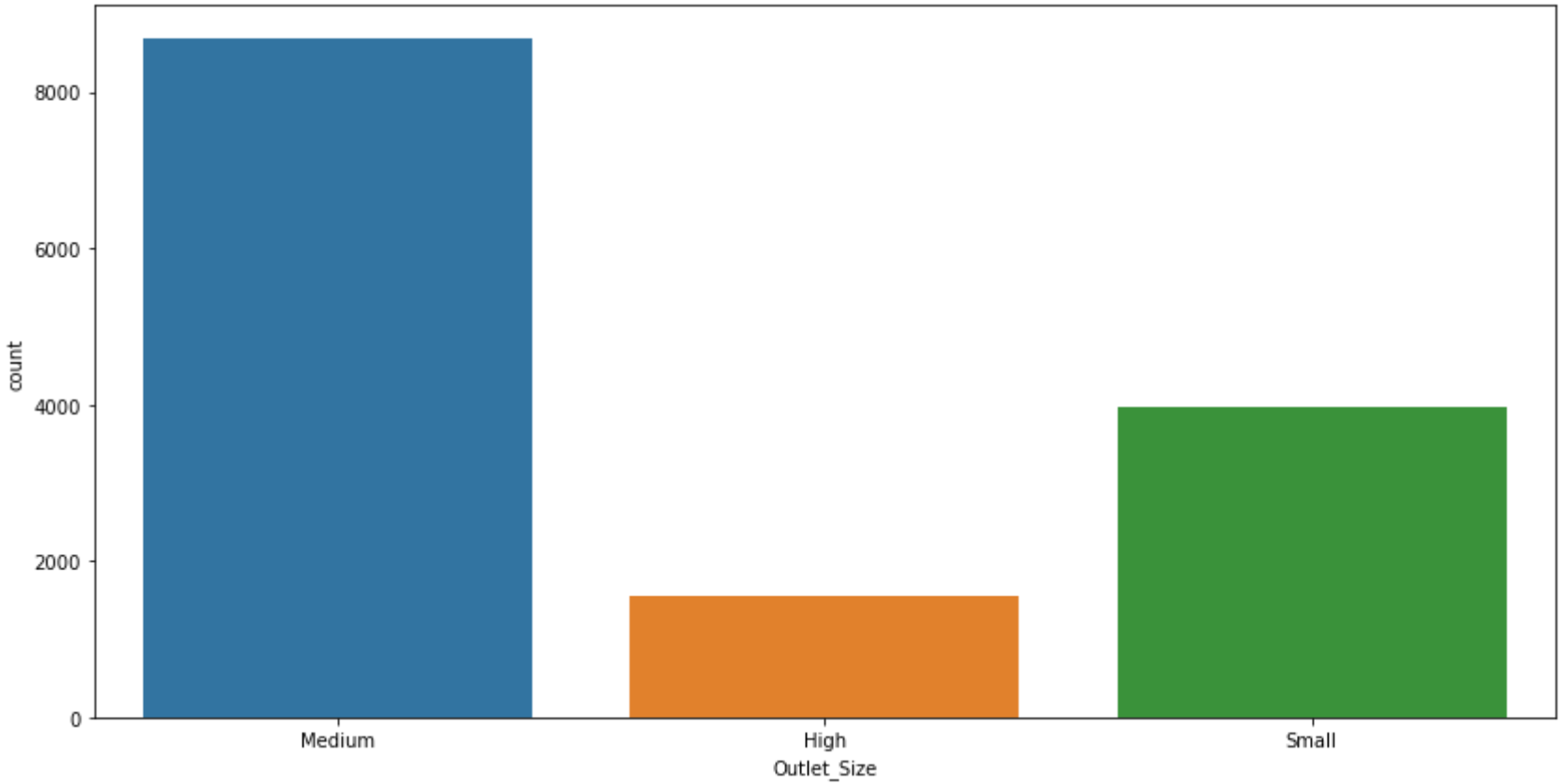
In [321…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Outlet_Identifier'])
```

Out[321]:
```
<AxesSubplot:xlabel='Outlet_Identifier', ylabel='count'>
```



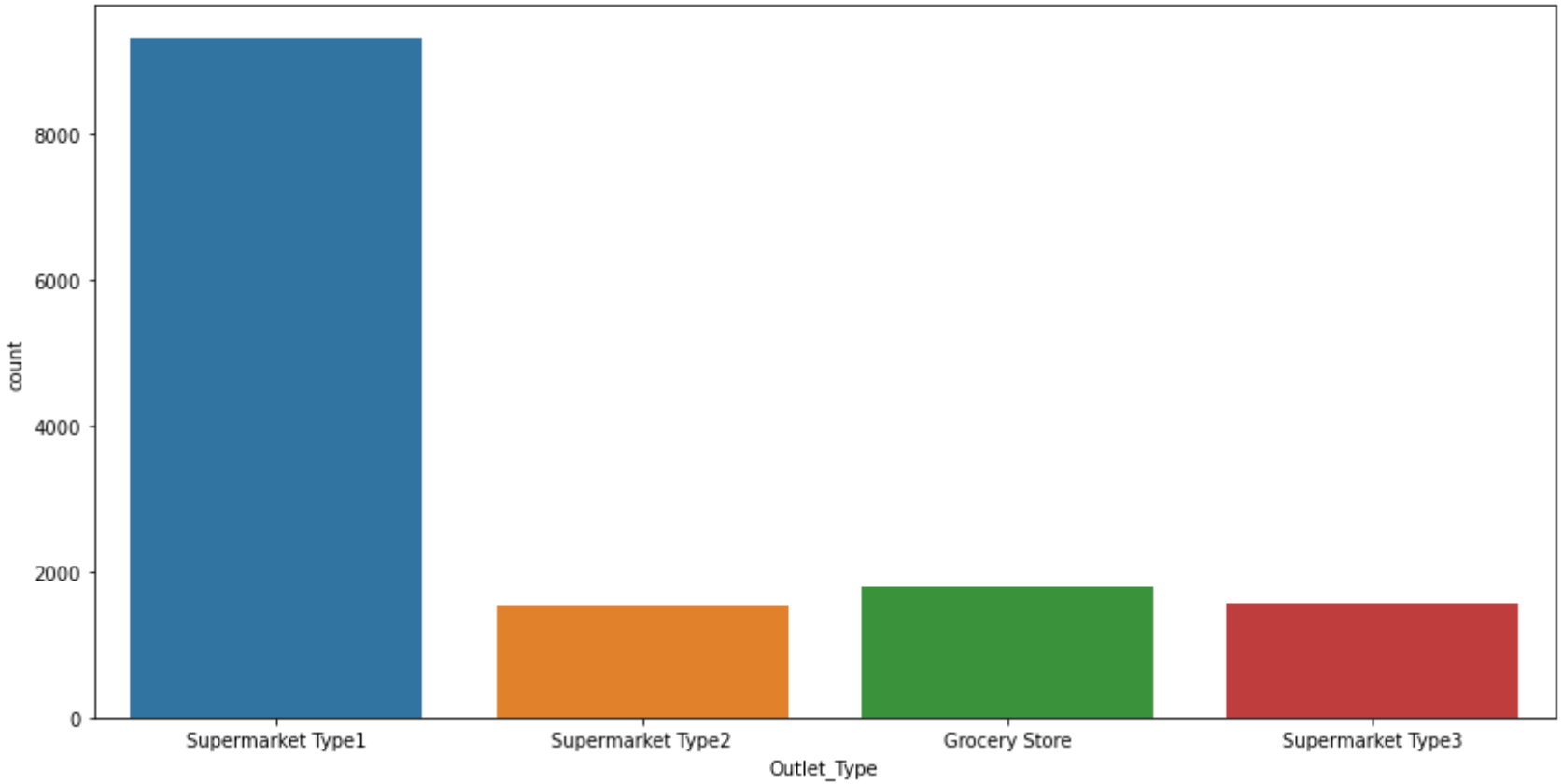
In [322…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Outlet_Size'])
```

Out[322]:
```
<AxesSubplot:xlabel='Outlet_Size', ylabel='count'>
```



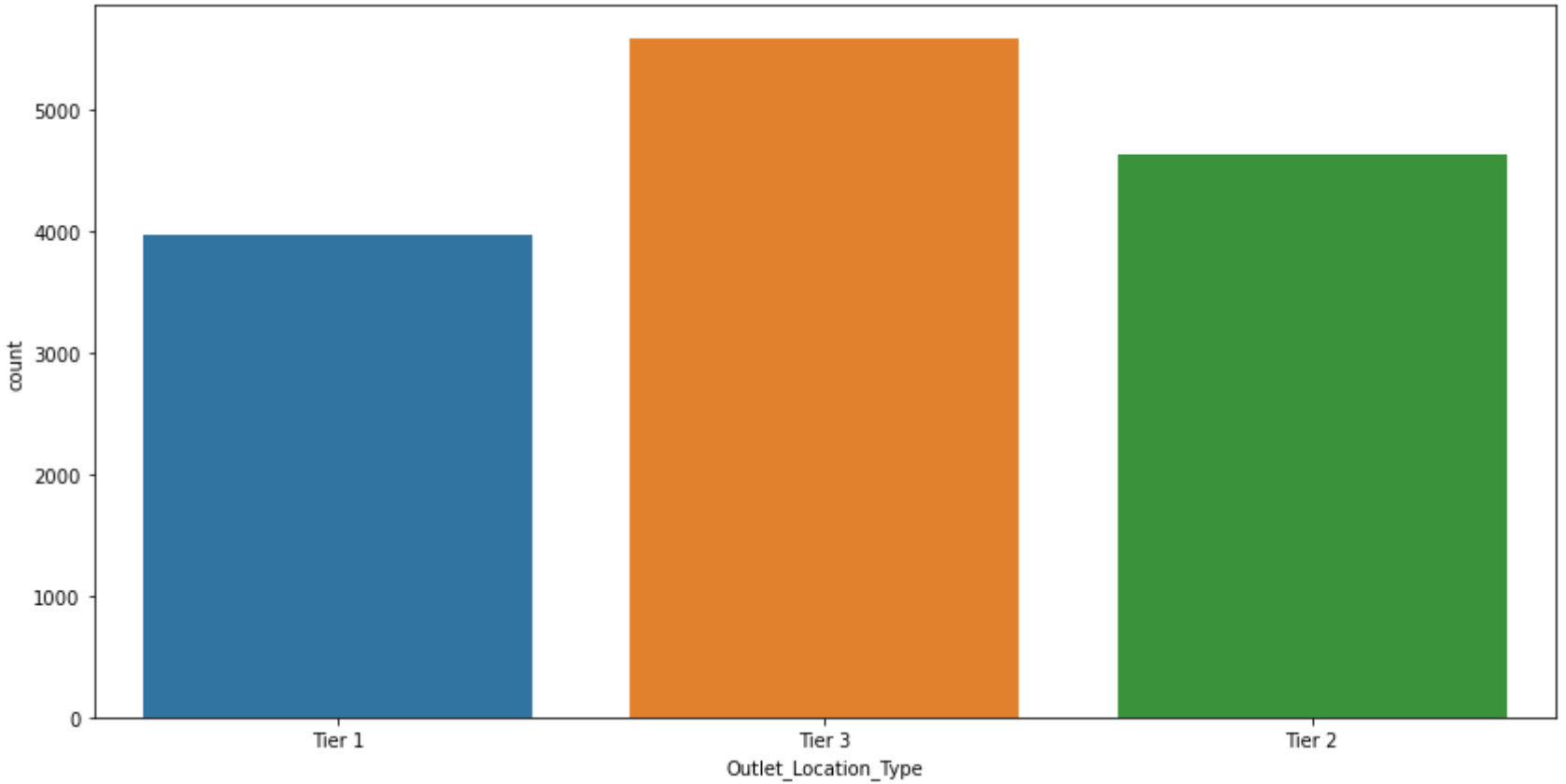
In [323…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Outlet_Type'])
```

Out[323]:
```
<AxesSubplot:xlabel='Outlet_Type', ylabel='count'>
```



In [324…
```
plt.figure(figsize=(14,7))
sns.countplot(data['Outlet_Location_Type'])
```

Out[324]:
```
<AxesSubplot:xlabel='Outlet_Location_Type', ylabel='count'>
```



In [325…
```
data['Item_Fat_Content'].value_counts()
```

Out[325]:
```
Low Fat    8485
Regular    4824
LF          522
reg         195
low fat     178
Name: Item_Fat_Content, dtype: int64
```

In [326…
```
data['Item_Fat_Content']= data['Item_Fat_Content'].map({'Low Fat':'Low Fat',
                                                        'Regular':'Regular',
                                                         'LF':'Low Fat',
                                                         'low fat':'Low Fat',
                                                         'reg':'Regular'})
```

In [327…
```
data['Item_Fat_Content'].value_counts()
```

Out[327]:
```
Low Fat    9185
Regular    5019
Name: Item_Fat_Content, dtype: int64
```

In [328…
```
data.dtypes
```

Out[328]:
```
Item_Identifier               object
Item_Weight                  float64
Item_Fat_Content              object
Item_Visibility              float64
Item_Type                     object
Item_MRP                     float64
Outlet_Identifier             object
Outlet_Establishment_Year      int64
Outlet_Size                   object
Outlet_Location_Type          object
Outlet_Type                   object
Item_Outlet_Sales            float64
source                        object
dtype: object
```

In [329…
```
data['Item_Material']=data['Item_Identifier'].apply(lambda x: x[0:2])
```

In [330…
```
data['Item_Material'].value_counts()
```

Out[330]:
```
FD    10201
NC     2686
DR     1317
Name: Item_Material, dtype: int64
```

In [331…
```
data['Item_Material'] = data['Item_Material'].map({'FD':'Food',
                                                   'NC':'Non-Consumable',
                                                   'DR':'Drinks'})
```

In [332…
```
data['Item_Material'].value_counts()
```

Out[332]:
```
Food              10201
Non-Consumable     2686
Drinks             1317
Name: Item_Material, dtype: int64
```

benchmark model

In [333…
```
data.select_dtypes(include=['int64','float64']).describe()
```

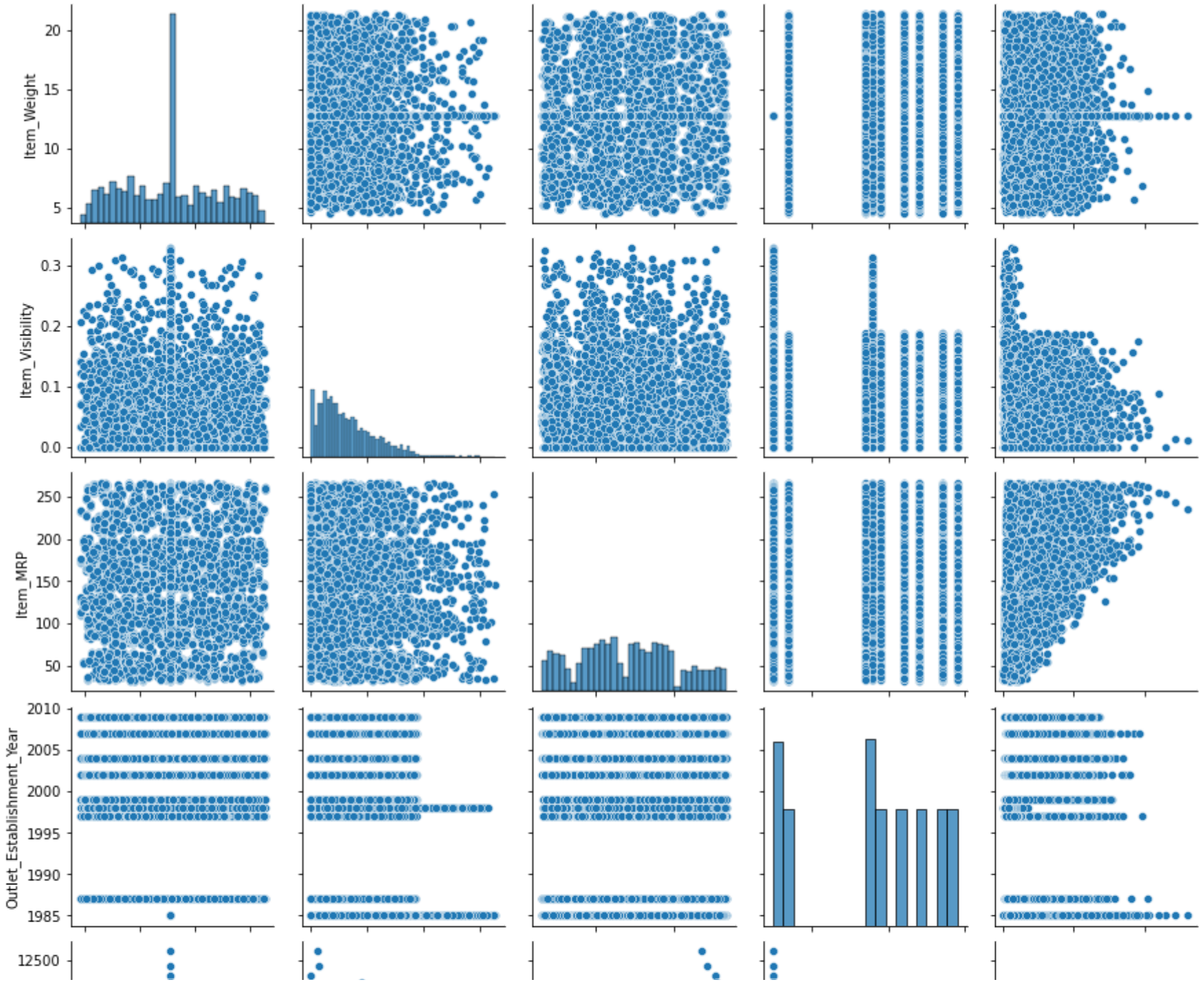
Out[333]:

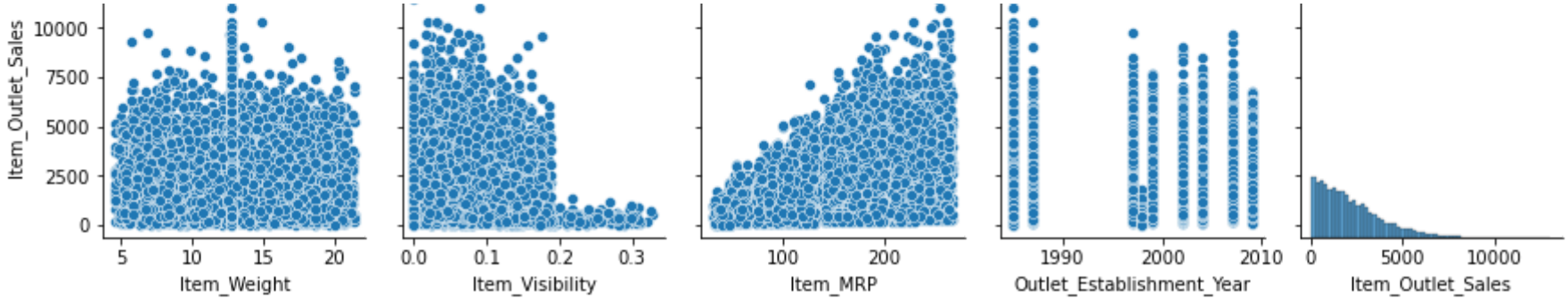
	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	14204.000000	14204.000000	14204.000000	14204.000000	8523.000000
mean	12.792854	0.065953	141.004977	1997.830681	2181.288914
std	4.234226	0.051459	62.086938	8.371664	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	9.300000	0.027036	94.012000	1987.000000	834.247400
50%	12.792854	0.054021	142.247000	1999.000000	1794.331000
75%	16.000000	0.094037	185.855600	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

corelation

In [334…
```
sns.pairplot(data,kind='scatter')
```

Out[334]:
```
<seaborn.axisgrid.PairGrid at 0x20cfede87c0>
```


Item_Weight
Item_Visibility
Item_MRP
Outlet_Establishment_Year
20
15
10
5
0.3
0.2
0.1
0.0
250
200
150
100
50
2010
2005
2000
1995
1990
1985
12500



In [335…
```
data.head()
```

Out[335]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	Medium	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	High	

In [336…
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
data['Outlet'] = le.fit_transform(data['Outlet_Identifier'])
var_mod = ['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Item_Material','Outlet_Type','Outlet']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i])
```

In [337…
```
data.head()
```

Out[337]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_
0	FDA15	9.30	0	0.016047	Dairy	249.8092	OUT049	1999	1	
1	DRC01	5.92	1	0.019278	Soft Drinks	48.2692	OUT018	2009	1	
2	FDN15	17.50	0	0.016760	Meat	141.6180	OUT049	1999	1	
3	FDX07	19.20	1	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	1	
4	NCD19	8.93	0	0.000000	Household	53.8614	OUT013	1987	0	

In [338…
```
data.head()
```

Out[338]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_
0	FDA15	9.30	0	0.016047	Dairy	249.8092	OUT049	1999	1	
1	DRC01	5.92	1	0.019278	Soft Drinks	48.2692	OUT018	2009	1	
2	FDN15	17.50	0	0.016760	Meat	141.6180	OUT049	1999	1	
3	FDX07	19.20	1	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	1	
4	NCD19	8.93	0	0.000000	Household	53.8614	OUT013	1987	0	

In [339…
```
data = pd.get_dummies(data, columns=['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Outlet_Type','Item_Material','Outl
```

In [340…
```
data.dtypes
```

Out[340]:
```
Item_Identifier              object
Item_Weight                 float64
Item_Visibility             float64
Item_Type                    object
Item_MRP                    float64
Outlet_Identifier            object
Outlet_Establishment_Year     int64
Item_Outlet_Sales           float64
source                       object
Item_Fat_Content_0            uint8
Item_Fat_Content_1            uint8
Outlet_Location_Type_0        uint8
Outlet_Location_Type_1        uint8
Outlet_Location_Type_2        uint8
Outlet_Size_0                 uint8
Outlet_Size_1                 uint8
Outlet_Size_2                 uint8
Outlet_Type_0                 uint8
Outlet_Type_1                 uint8
Outlet_Type_2                 uint8
Outlet_Type_3                 uint8
Item_Material_0               uint8
Item_Material_1               uint8
Item_Material_2               uint8
Outlet_0                      uint8
Outlet_1                      uint8
Outlet_2                      uint8
Outlet_3                      uint8
Outlet_4                      uint8
Outlet_5                      uint8
Outlet_6                      uint8
Outlet_7                      uint8
Outlet_8                      uint8
Outlet_9                      uint8
dtype: object
```

In [341…
```
data.head()
```

Out[341]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Item_Outlet_Sales	source	Item_Fat_C
0	FDA15	9.30	0.016047	Dairy	249.8092	OUT049	1999	3735.1380	train	
1	DRC01	5.92	0.019278	Soft Drinks	48.2692	OUT018	2009	443.4228	train	
2	FDN15	17.50	0.016760	Meat	141.6180	OUT049	1999	2097.2700	train	
3	FDX07	19.20	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	732.3800	train	
4	NCD19	8.93	0.000000	Household	53.8614	OUT013	1987	994.7052	train	

5 rows × 34 columns

In [342…

```
import warnings
warnings.filterwarnings('ignore')
#Drop the columns which have been converted to different types:
data.drop(['Item_Type','Outlet_Establishment_Year'],axis=1,inplace=True)
train = data.loc[data['source']=="train"]
test = data.loc[data['source']=="test"]

#Drop unnecessary columns:
test.drop(['Item_Outlet_Sales','source'],axis=1,inplace=True)
train.drop(['source'],axis=1,inplace=True)

#Export files as modified versions:
train.to_csv("train_dummies.csv",index=False)
test.to_csv("test_dummies.csv",index=False)
```

In [343…

```
train = pd.read_csv("train_modified.csv")
test = pd.read_csv("test_modified.csv")
```

In [344…

```
train.head()
```

Out[344]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_MRP	Outlet_Identifier	Item_Outlet_Sales	Item_Fat_Content_0	Item_Fat_Content_1	Item_Fat_Content
0	FDA15	9.30	0.016047	249.8092	OUT049	3735.1380	0	1	
1	DRC01	5.92	0.019278	48.2692	OUT018	443.4228	0	0	
2	FDN15	17.50	0.016760	141.6180	OUT049	2097.2700	0	1	
3	FDX07	19.20	0.000000	182.0950	OUT010	732.3800	0	0	
4	NCD19	8.93	0.000000	53.8614	OUT013	994.7052	0	1	

5 rows × 34 columns

In [345…
```
test.head()
```

Out[345]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_MRP	Outlet_Identifier	Item_Fat_Content_0	Item_Fat_Content_1	Item_Fat_Content_2	Item_Fat_Conte
0	FDW58	20.750000	0.007565	107.8622	OUT049	0	1	0	
1	FDW14	8.300000	0.038428	87.3198	OUT017	0	0	0	
2	NCN55	14.600000	0.099575	241.7538	OUT010	0	1	0	
3	FDQ58	7.315000	0.015388	155.0340	OUT017	0	1	0	
4	FDY38	12.792854	0.118599	234.2300	OUT027	0	0	1	

5 rows × 33 columns

In [346…
```
train.dtypes
```

Out[346]:
```
Item_Identifier           object
Item_Weight              float64
Item_Visibility          float64
Item_MRP                 float64
Outlet_Identifier         object
Item_Outlet_Sales        float64
Item_Fat_Content_0         int64
Item_Fat_Content_1         int64
Item_Fat_Content_2         int64
Item_Fat_Content_3         int64
Item_Fat_Content_4         int64
Outlet_Location_Type_0     int64
Outlet_Location_Type_1     int64
Outlet_Location_Type_2     int64
Outlet_Size_0              int64
Outlet_Size_1              int64
Outlet_Size_2              int64
Outlet_Type_0              int64
Outlet_Type_1              int64
Outlet_Type_2              int64
Outlet_Type_3              int64
Item_Type_Combined_0       int64
Item_Type_Combined_1       int64
Item_Type_Combined_2       int64
Outlet_0                   int64
Outlet_1                   int64
Outlet_2                   int64
Outlet_3                   int64
Outlet_4                   int64
Outlet_5                   int64
Outlet_6                   int64
Outlet_7                   int64
Outlet_8                   int64
Outlet_9                   int64
dtype: object
```

In [347…
```
train.shape,test.shape
```

Out[347]:
```
((8523, 34), (5681, 33))
```

In [348…
```
X_train = train.drop(['Item_Outlet_Sales', 'Outlet_Identifier','Item_Identifier'], axis=1)
y_train = train.Item_Outlet_Sales
```

In [349…
```
y_train.head()
```

Out[349]:
```
0    3735.1380
1     443.4228
2    2097.2700
3     732.3800
4     994.7052
Name: Item_Outlet_Sales, dtype: float64
```

In [353…
```
test = test.drop(['Outlet_Identifier','Item_Identifier'], axis=1)
```

In [354…
```
from sklearn.linear_model import  LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[354]:
```
LinearRegression()
```

In [355…
```
y_pred = regressor.predict(test)
y_pred
```

Out[355]:
```
array([1848.53604783, 1472.81670435, 1875.65285894, ..., 1809.18796433,
       3565.6645235 , 1267.46171871])
```

In [356…
```
import warnings
warnings.filterwarnings('ignore')
# Measuring Accuracy
from sklearn.metrics import accuracy_score, r2_score, mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn import model_selection, metrics
```

In [357…
```
lr_accuracy = round(regressor.score(X_train,y_train) * 100,2)
lr_accuracy
```

Out[357]:
```
56.36
```

In [362…
```
submission = pd.DataFrame({
'Item_Identifier':data2['Item_Identifier'],
'Outlet_Identifier':data2['Outlet_Identifier'],
'Item_Outlet_Sales': y_pred
},columns=['Item_Identifier','Outlet_Identifier','Item_Outlet_Sales'])
```

In [363…
```
submission.to_csv('submission1.csv',index=False)
```

In [ ]:
```

```