##Importing Modules

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib as plt
import warnings
%matplotlib inline
warnings.filterwarnings('ignore')
```

##Loading the dataset

```
df = pd.read_csv('Train.csv')
df.head()

  Item_Identifier  Item_Weight Item_Fat_Content  Item_Visibility  \
0           FDA15         9.30          Low Fat         0.016047
1           DRC01         5.92          Regular         0.019278
2           FDN15        17.50          Low Fat         0.016760
3           FDX07        19.20          Regular         0.000000
4           NCD19         8.93          Low Fat         0.000000

               Item_Type  Item_MRP Outlet_Identifier  \
0                  Dairy  249.8092            OUT049
1            Soft Drinks   48.2692            OUT018
2                   Meat  141.6180            OUT049
3  Fruits and Vegetables  182.0950            OUT010
4              Household   53.8614            OUT013

   Outlet_Establishment_Year Outlet_Size Outlet_Location_Type  \
0                       1999      Medium               Tier 1
1                       2009      Medium               Tier 3
2                       1999      Medium               Tier 1
3                       1998         NaN               Tier 3
4                       1987        High               Tier 3

         Outlet_Type  Item_Outlet_Sales
0  Supermarket Type1          3735.1380
1  Supermarket Type2           443.4228
2  Supermarket Type1          2097.2700
3      Grocery Store           732.3800
4  Supermarket Type1           994.7052

#statical info
df.describe()

       Item_Weight  Item_Visibility     Item_MRP
Outlet_Establishment_Year  \
count  7060.000000      8523.000000  8523.000000
```

```
8523.000000
mean     12.857645         0.066132   140.992782
1997.831867
std       4.643456         0.051598    62.275067
8.371760
min       4.555000         0.000000    31.290000
1985.000000
25%       8.773750         0.026989    93.826500
1987.000000
50%      12.600000         0.053931   143.012800
1999.000000
75%      16.850000         0.094585   185.643700
2004.000000
max      21.350000         0.328391   266.888400
2009.000000

       Item_Outlet_Sales
count        8523.000000
mean         2181.288914
std          1706.499616
min            33.290000
25%           834.247400
50%          1794.331000
75%          3101.296400
max         13086.964800
```

```
#datatype of atributes
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
```

```
#check unique values in dataset
df.apply(lambda x: len(x.unique()))
```

```
Item_Identifier              1559
Item_Weight                   416
Item_Fat_Content                5
Item_Visibility              7880
Item_Type                      16
Item_MRP                     5938
Outlet_Identifier              10
Outlet_Establishment_Year       9
Outlet_Size                     4
Outlet_Location_Type            3
Outlet_Type                     4
Item_Outlet_Sales            3493
dtype: int64
```

##Preprocessing the dataset

```
#checking for null values
df.isnull().sum()
```

```
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
```

```
#checking for categorical values
cat_col = []
for x in df.dtypes.index:
    if df.dtypes[x]== 'object':
        cat_col.append(x)
cat_col
```

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

```
#removing the identifiers
cat_col.remove('Item_Identifier')
cat_col.remove('Outlet_Identifier')
cat_col
```

```
['Item_Fat_Content',
 'Item_Type',
 'Outlet_Size',
 'Outlet_Location_Type',
 'Outlet_Type']
```

```
#print the categorical columns
for col in cat_col:
    print(df[col].value_counts())
    print()
```

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

Outlet_Size
Medium    2793
Small     2388
High       932
Name: count, dtype: int64

Outlet_Location_Type
```

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

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

```
#filling the missing values
item_weight_mean = df.pivot_table(values="Item_Weight", index =
"Item_Identifier")
item_weight_mean
```

```
                 Item_Weight
Item_Identifier
DRA12                 11.600
DRA24                 19.350
DRA59                  8.270
DRB01                  7.390
DRB13                  6.115
...                      ...
NCZ30                  6.590
NCZ41                 19.850
NCZ42                 10.500
NCZ53                  9.600
NCZ54                 14.650

[1555 rows x 1 columns]
```

```
miss_bool = df['Item_Weight'].isnull()
miss_bool
```

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

```
for i, item in enumerate(df['Item_Identifier']):
    if miss_bool[i]:
        if item in item_weight_mean:
            df['Item_Weight'][i] = item_weight_mean.loc[item]
['Item_Weight']
        else :
            df['Item_Weight'][i] = np.mean(df['Item_Weight'])
```

```
df['Item_Weight'].isnull().sum()
```

```
0
```

```
#aggreganting the Outlat type (that has null values) with outlet size
outlet_size_mode = df.pivot_table(values='Outlet_Size',
columns='Outlet_Type', aggfunc=(lambda x: x.mode()[0]))
outlet_size_mode
```

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

Outlet_Type Supermarket Type3
Outlet_Size            Medium
```

```
miss_bool = df['Outlet_Size'].isnull()
df.loc[miss_bool, 'Outlet_Size'] = df.loc[miss_bool,
'Outlet_Type'].apply(lambda x: outlet_size_mode[x])
```

```
df['Outlet_Size'].isnull().sum()
```

```
0
```

```
sum(df['Item_Visibility']==0)
```

```
526
```

```
#replace zeros with mean
df.loc[:, 'Item_Visibility'].replace([0],
[df['Item_Visibility'].mean()], inplace=True)
```

```
sum(df['Item_Visibility']==0)
```

```
0
```

```
#combining item fat content
df['Item_Fat_Content'] = df['Item_Fat_Content'].replace({'LF':'Low
Fat', 'reg':'Regular','low fat':'Low Fat' })
df['Item_Fat_Content'].value_counts()
```

```
Item_Fat_Content
Low Fat    5517
Regular    3006
Name: count, dtype: int64
```

##Creation of New Attributes

```
df['New_Item_Type'] = df['Item_Identifier'].apply(lambda x: x[:2])
df['New_Item_Type']

0       FD
1       DR
2       FD
3       FD
4       NC
        ..
8518    FD
8519    FD
8520    NC
8521    FD
8522    DR
Name: New_Item_Type, Length: 8523, dtype: object

df['New_Item_Type'] =
df['New_Item_Type'].replace({'FD':'Food','NC':'Non-
Consumable','DR':'Drinks'})
df['New_Item_Type'].value_counts()

New_Item_Type
Food              6125
Non-Consumable    1599
Drinks             799
Name: count, dtype: int64

df.loc[df['New_Item_Type']=='Non-Consumable', 'Item_Fat_Content'] =
'Non-Edible'
df['Item_Fat_Content'].value_counts()

Item_Fat_Content
Low Fat       3918
Regular       3006
Non-Edible    1599
Name: count, dtype: int64

#create small values for establishment year (time ago)
df['Outlet_Years'] = 2013 - df['Outlet_Establishment_Year']

df['Outlet_Years']

0       14
1        4
2       14
3       15
4       26
        ..
8518    26
```

```
8519    11
8520     9
8521     4
8522    16
Name: Outlet_Years, Length: 8523, dtype: int64
```

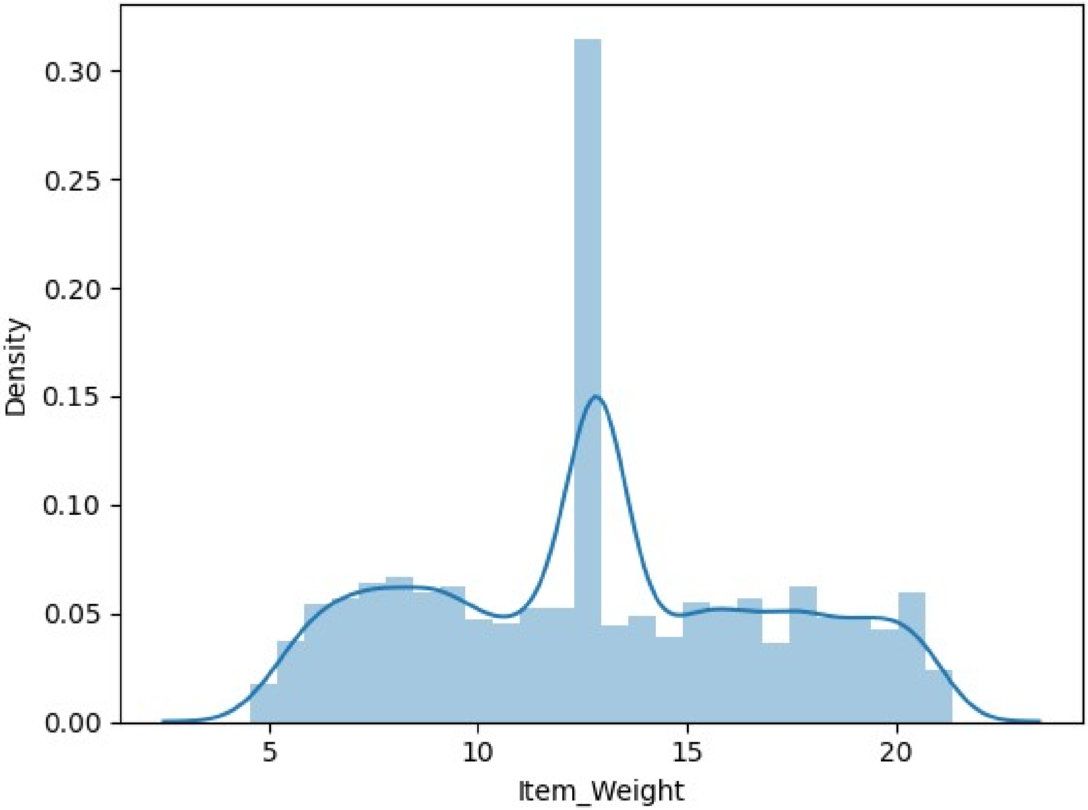
```
df.head()
```

```
  Item_Identifier  Item_Weight Item_Fat_Content  Item_Visibility  \
0           FDA15         9.30          Low Fat         0.016047   
1           DRC01         5.92          Regular         0.019278   
2           FDN15        17.50          Low Fat         0.016760   
3           FDX07        19.20          Regular         0.066132   
4           NCD19         8.93       Non-Edible         0.066132   

               Item_Type  Item_MRP Outlet_Identifier  \
0                  Dairy  249.8092            OUT049   
1            Soft Drinks   48.2692            OUT018   
2                   Meat  141.6180            OUT049   
3  Fruits and Vegetables  182.0950            OUT010   
4              Household   53.8614            OUT013   

   Outlet_Establishment_Year Outlet_Size Outlet_Location_Type  \
0                       1999      Medium               Tier 1   
1                       2009      Medium               Tier 3   
2                       1999      Medium               Tier 1   
3                       1998       Small               Tier 3   
4                       1987        High               Tier 3   

         Outlet_Type  Item_Outlet_Sales   New_Item_Type  Outlet_Years  

0  Supermarket Type1          3735.1380            Food            14  

1  Supermarket Type2           443.4228          Drinks             4  

2  Supermarket Type1          2097.2700            Food            14  

3      Grocery Store           732.3800            Food            15  

4  Supermarket Type1           994.7052  Non-Consumable            26  
```

##Exploratory Data Analysis

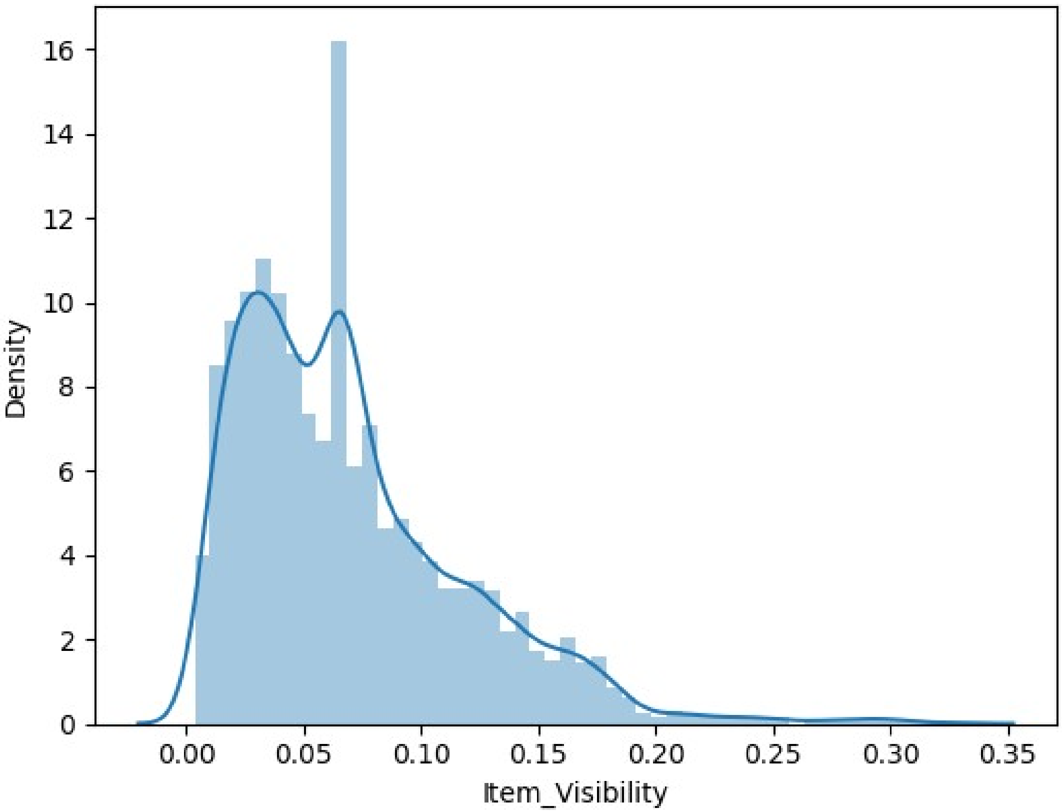
```
sns.distplot(df['Item_Weight'])
```

```
<Axes: xlabel='Item_Weight', ylabel='Density'>
```



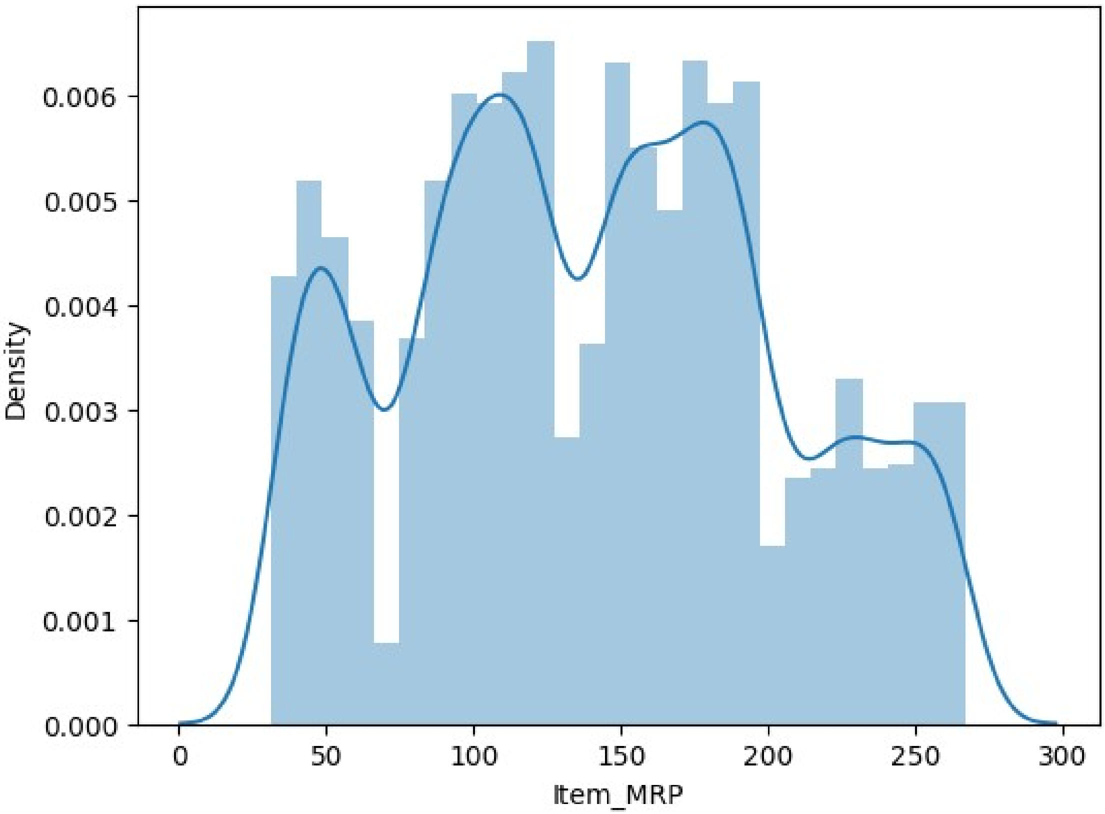
```
sns.distplot(df['Item_Visibility'])
```

```
<Axes: xlabel='Item_Visibility', ylabel='Density'>
```



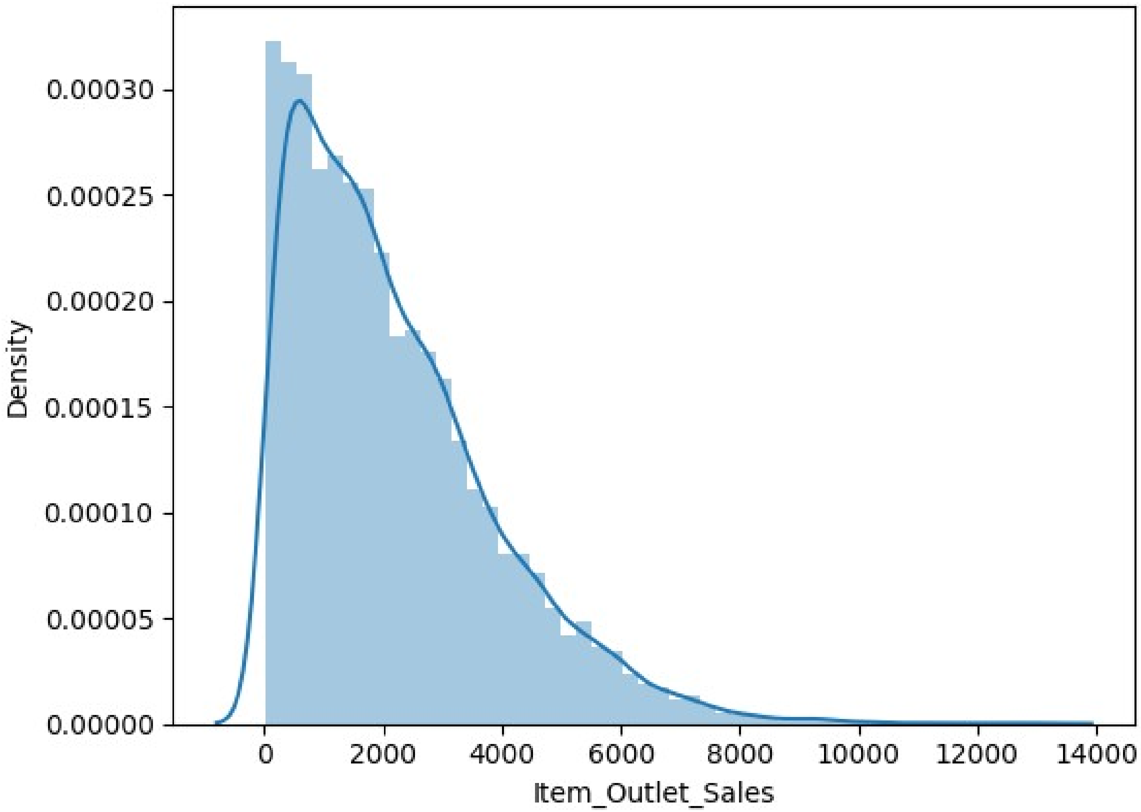
```
sns.distplot(df['Item_MRP'])
```

```
<Axes: xlabel='Item_MRP', ylabel='Density'>
```



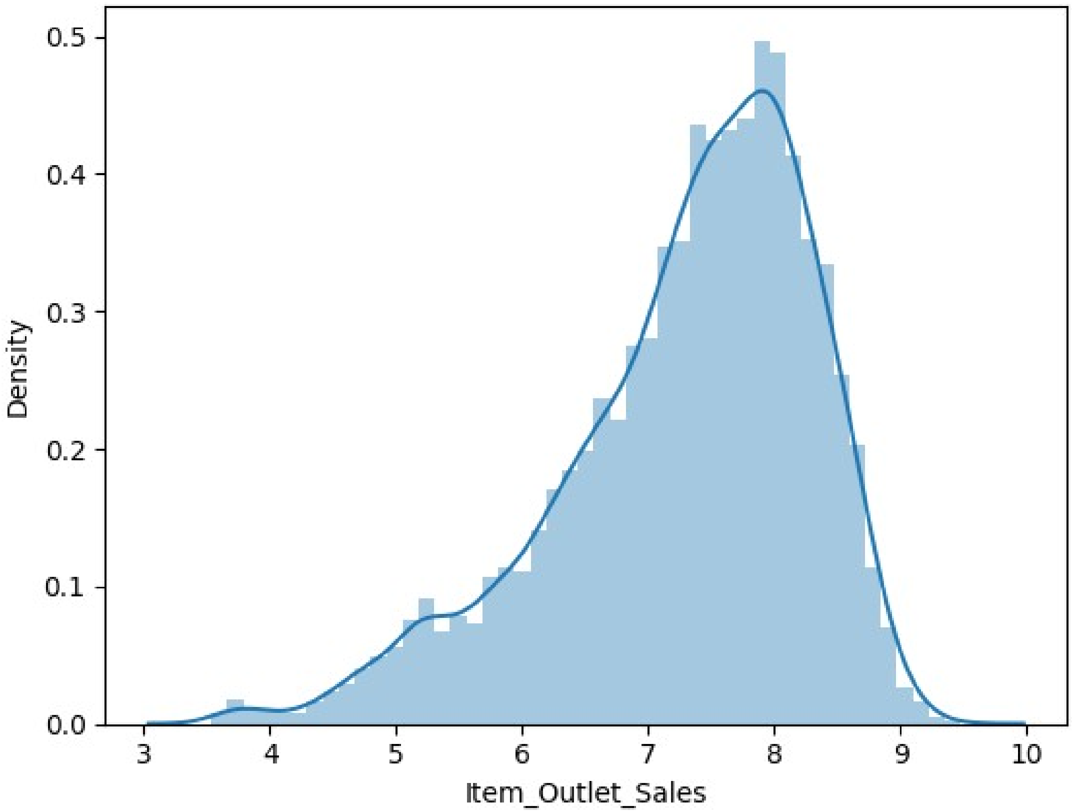
```
sns.distplot(df['Item_Outlet_Sales'])
```

```
<Axes: xlabel='Item_Outlet_Sales', ylabel='Density'>
```



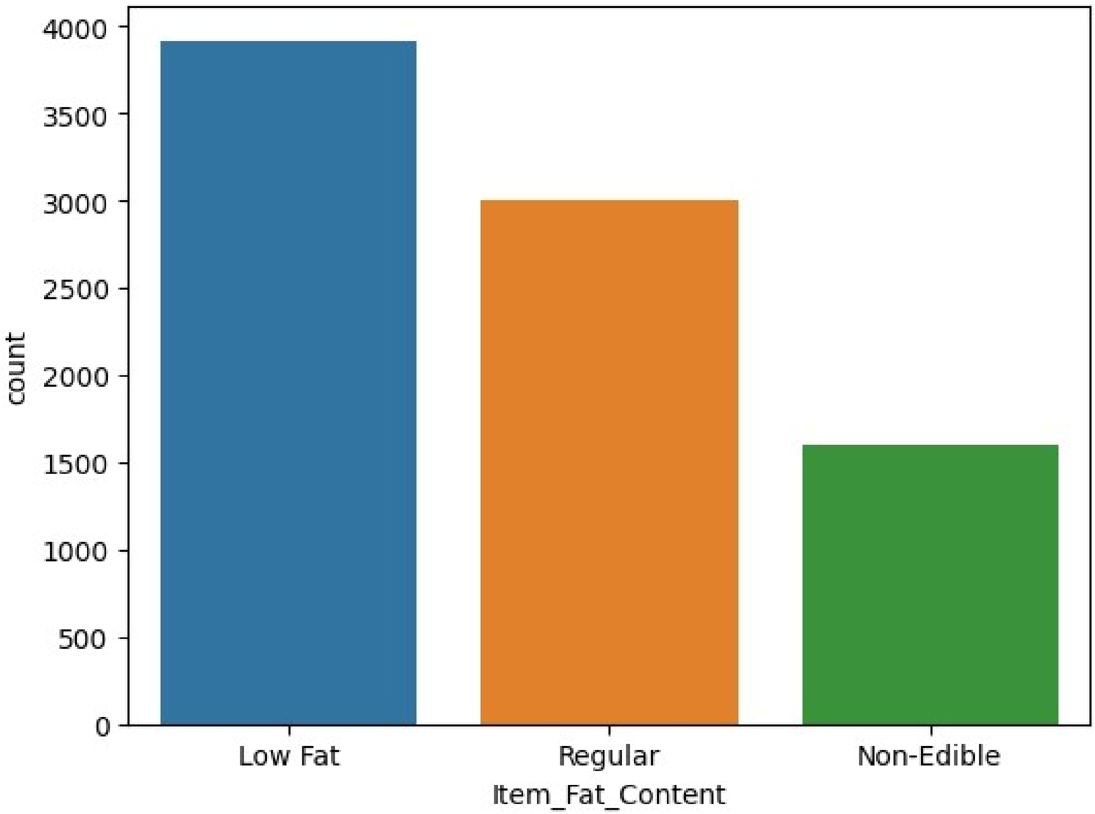
```
#log transformation to normalize item outlet sales
df['Item_Outlet_Sales'] = np.log(1+(df['Item_Outlet_Sales']))

sns.distplot(df['Item_Outlet_Sales'])
```

```
<Axes: xlabel='Item_Outlet_Sales', ylabel='Density'>
```



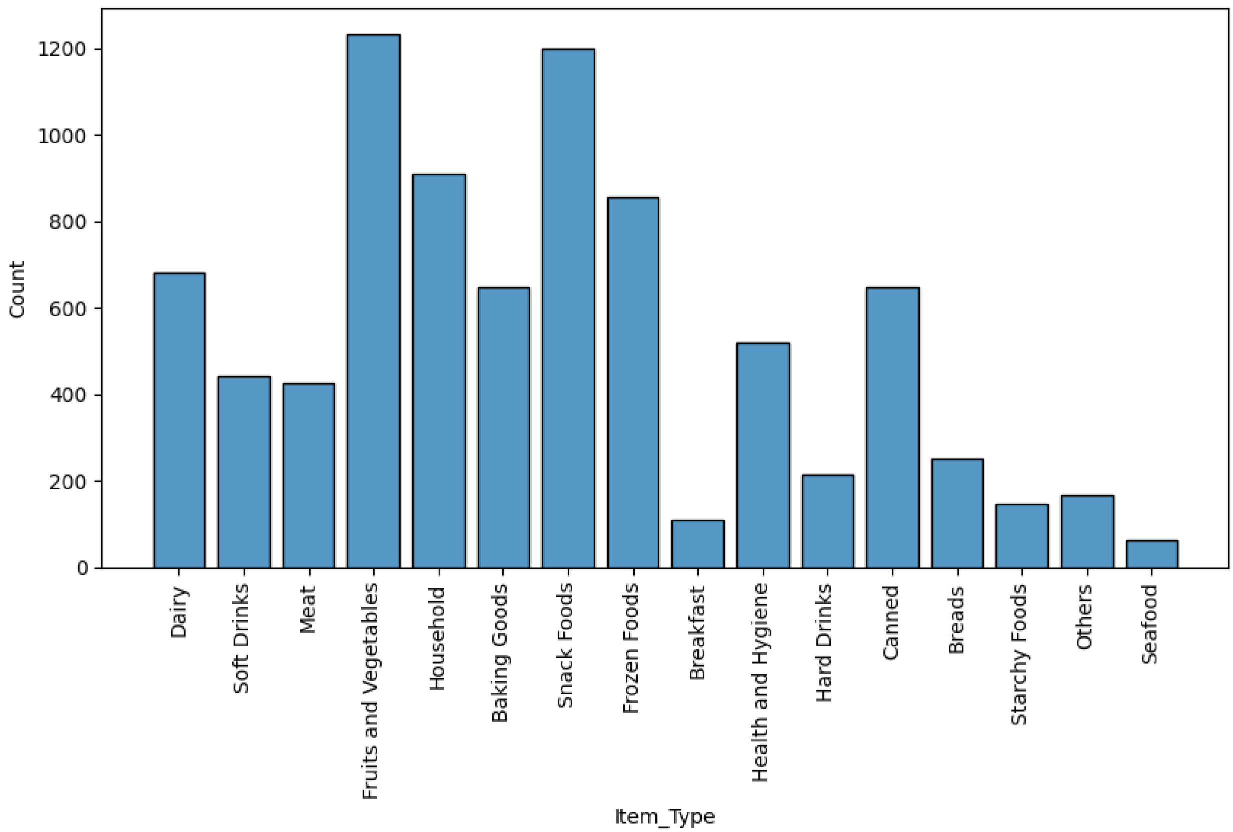
```
#categorical attributes
sns.countplot(x='Item_Fat_Content', data=df)
```

```
<Axes: xlabel='Item_Fat_Content', ylabel='count'>
```



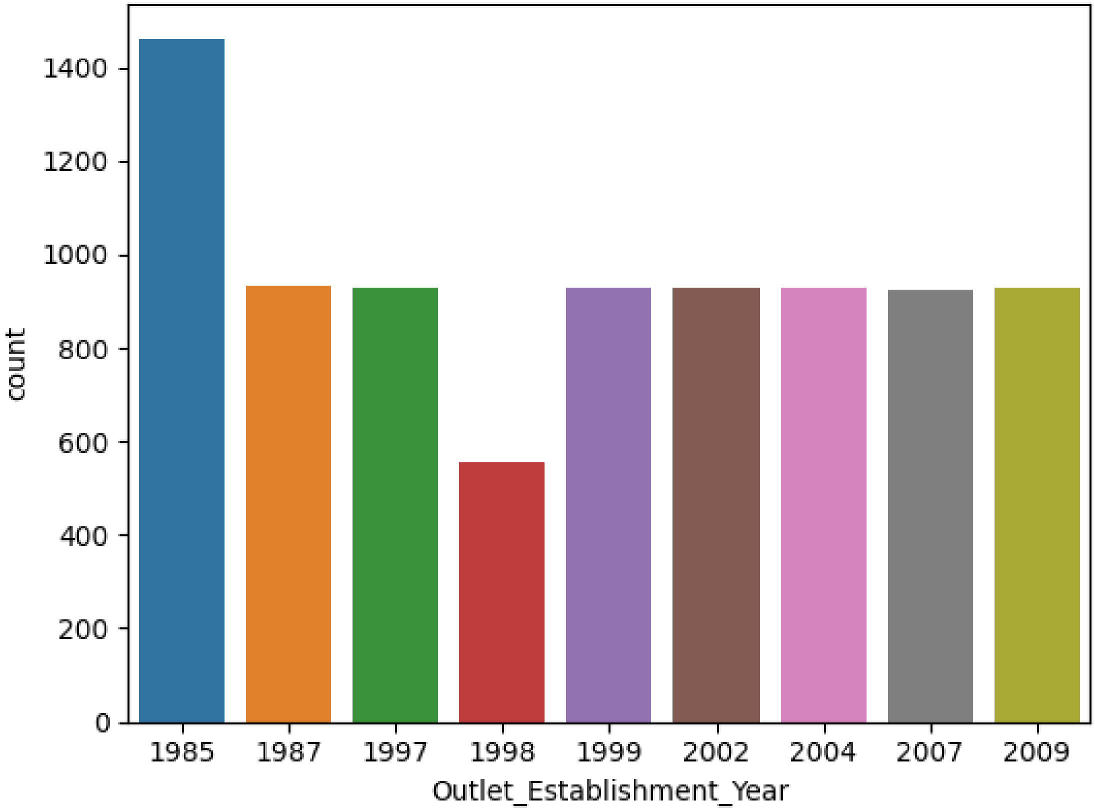
```
plt.pyplot.figure(figsize=(10,5))
l = list(df['Item_Type'].unique())
chart = sns.histplot(df['Item_Type'],discrete="true", binwidth=5,
shrink=.8)
chart.set_xticklabels(labels=l, rotation=90)
```

```
[Text(0, 0, 'Dairy'),
 Text(1, 0, 'Soft Drinks'),
 Text(2, 0, 'Meat'),
 Text(3, 0, 'Fruits and Vegetables'),
 Text(4, 0, 'Household'),
 Text(5, 0, 'Baking Goods'),
 Text(6, 0, 'Snack Foods'),
 Text(7, 0, 'Frozen Foods'),
 Text(8, 0, 'Breakfast'),
 Text(9, 0, 'Health and Hygiene'),
 Text(10, 0, 'Hard Drinks'),
 Text(11, 0, 'Canned'),
 Text(12, 0, 'Breads'),
 Text(13, 0, 'Starchy Foods'),
 Text(14, 0, 'Others'),
 Text(15, 0, 'Seafood')]
```



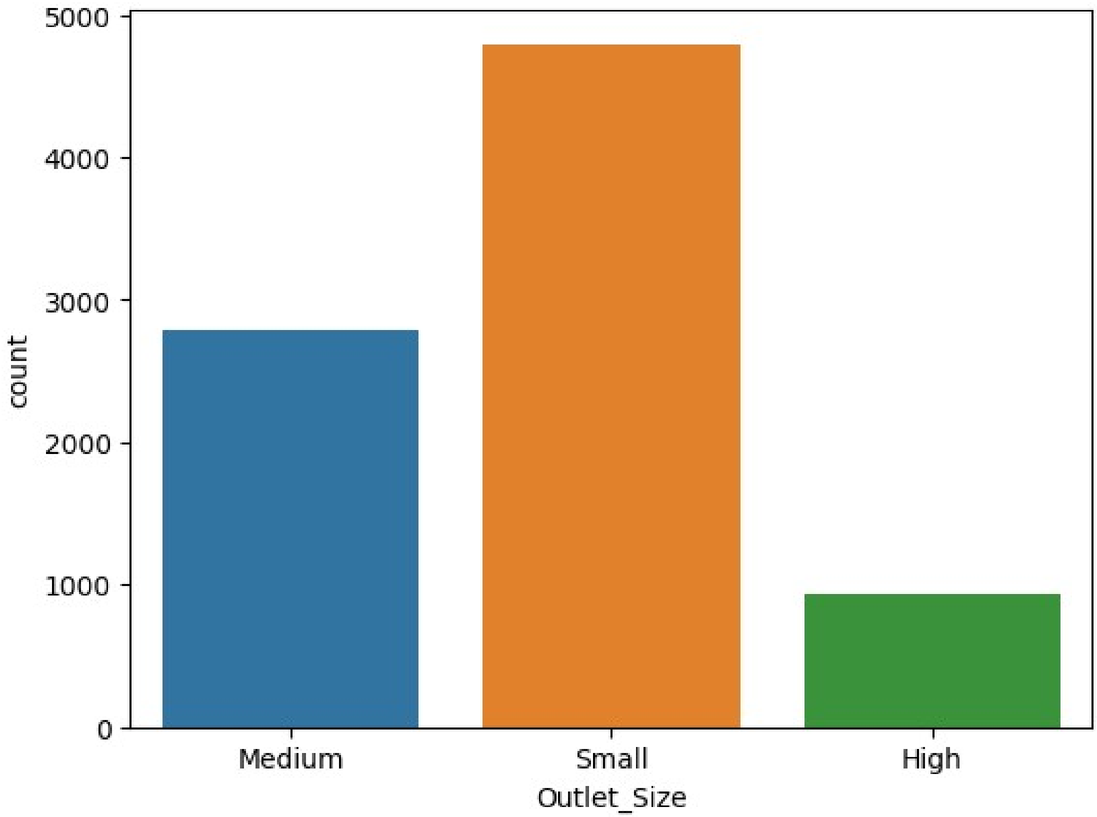
```
sns.countplot(x='Outlet_Establishment_Year', data=df)
```

```
<Axes: xlabel='Outlet_Establishment_Year', ylabel='count'>
```



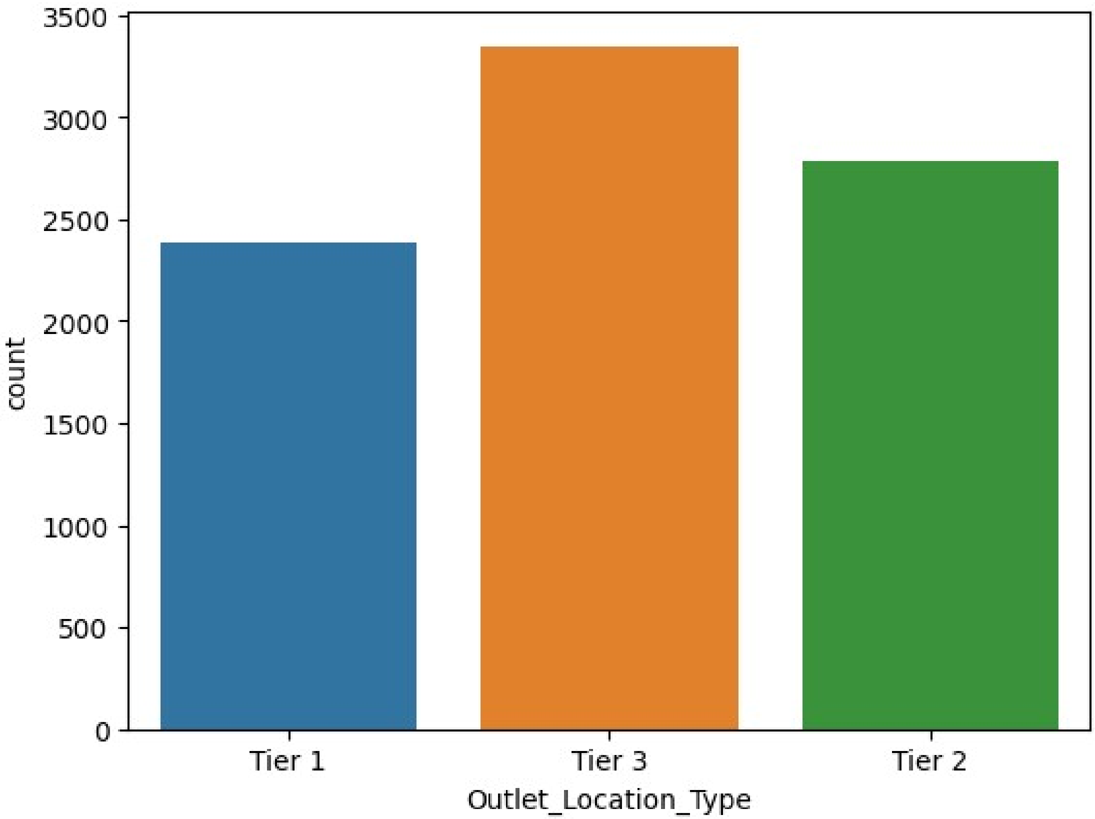
```
sns.countplot(x='Outlet_Size', data=df)
```

```
<Axes: xlabel='Outlet_Size', ylabel='count'>
```



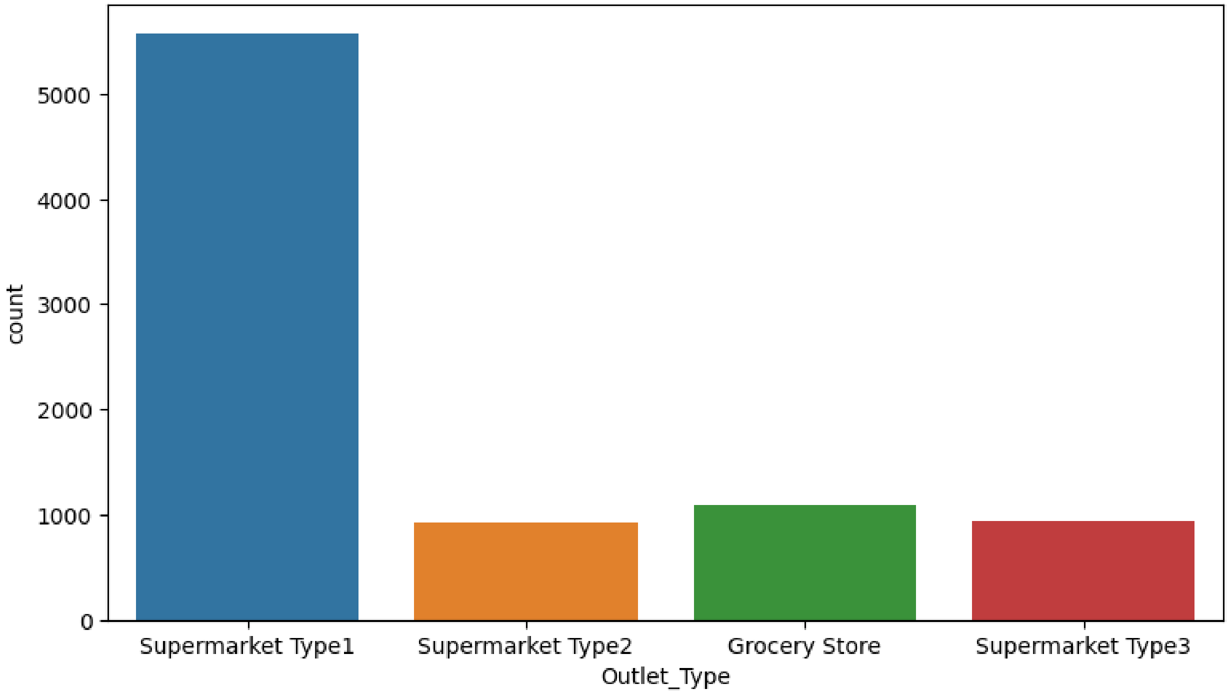
```
sns.countplot(x='Outlet_Location_Type', data=df)
```

```
<Axes: xlabel='Outlet_Location_Type', ylabel='count'>
```



```
plt.pyplot.figure(figsize=(9,5))
sns.countplot(x='Outlet_Type', data=df)
```

```
<Axes: xlabel='Outlet_Type', ylabel='count'>
```



##Label Encoding

```
df.head()

  Item_Identifier  Item_Weight Item_Fat_Content  Item_Visibility  \
0           FDA15         9.30          Low Fat         0.016047
1           DRC01         5.92          Regular         0.019278
2           FDN15        17.50          Low Fat         0.016760
3           FDX07        19.20          Regular         0.066132
4           NCD19         8.93       Non-Edible         0.066132

               Item_Type  Item_MRP Outlet_Identifier  \
0                  Dairy  249.8092            OUT049
1            Soft Drinks   48.2692            OUT018
2                   Meat  141.6180            OUT049
3  Fruits and Vegetables  182.0950            OUT010
4              Household   53.8614            OUT013

   Outlet_Establishment_Year Outlet_Size Outlet_Location_Type  \
0                       1999      Medium               Tier 1
1                       2009      Medium               Tier 3
2                       1999      Medium               Tier 1
3                       1998       Small               Tier 3
4                       1987        High               Tier 3

         Outlet_Type  Item_Outlet_Sales   New_Item_Type  Outlet_Years

0  Supermarket Type1           8.225808            Food            14
```

```
1  Supermarket Type2           6.096776          Drinks             4

2  Supermarket Type1           7.648868            Food            14

3      Grocery Store           6.597664            Food            15

4  Supermarket Type1           6.903451  Non-Consumable            26
```

```python
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['Outlet'] = le.fit_transform(df['Outlet_Identifier'])
cat_col =
['Item_Fat_Content','Item_Type','Outlet_Size','Outlet_Location_Type','
Outlet_Type', 'New_Item_Type']
for col in cat_col:
    df[col] = le.fit_transform(df[col])
```

##Onehot Enconding

```python
df = pd.get_dummies(df,
columns=['Item_Fat_Content','Outlet_Size','Outlet_Location_Type','Outl
et_Type', 'New_Item_Type'])
df.head()
```

```
  Item_Identifier  Item_Weight  Item_Visibility  Item_Type
Item_MRP  \
0           FDA15         9.30         0.016047          4  249.8092

1           DRC01         5.92         0.019278         14   48.2692

2           FDN15        17.50         0.016760         10  141.6180

3           FDX07        19.20         0.066132          6  182.0950

4           NCD19         8.93         0.066132          9   53.8614


  Outlet_Identifier  Outlet_Establishment_Year  Item_Outlet_Sales  \
0            OUT049                       1999           8.225808
1            OUT018                       2009           6.096776
2            OUT049                       1999           7.648868
3            OUT010                       1998           6.597664
4            OUT013                       1987           6.903451

   Outlet_Years  Outlet  ...  Outlet_Location_Type_0
Outlet_Location_Type_1  \
0            14       9  ...                    True
False
1             4       3  ...                   False
```

```
False
2            14        9  ...                         True
False
3            15        0  ...                        False
False
4            26        1  ...                        False
False

   Outlet_Location_Type_2  Outlet_Type_0  Outlet_Type_1  Outlet_Type_2
\
0                   False          False           True          False

1                    True          False          False           True

2                   False          False           True          False

3                    True           True          False          False

4                    True          False           True          False


   Outlet_Type_3  New_Item_Type_0  New_Item_Type_1  New_Item_Type_2
0          False            False             True            False
1          False             True            False            False
2          False            False             True            False
3          False            False             True            False
4          False            False            False             True

[5 rows x 26 columns]
```

##Input Split

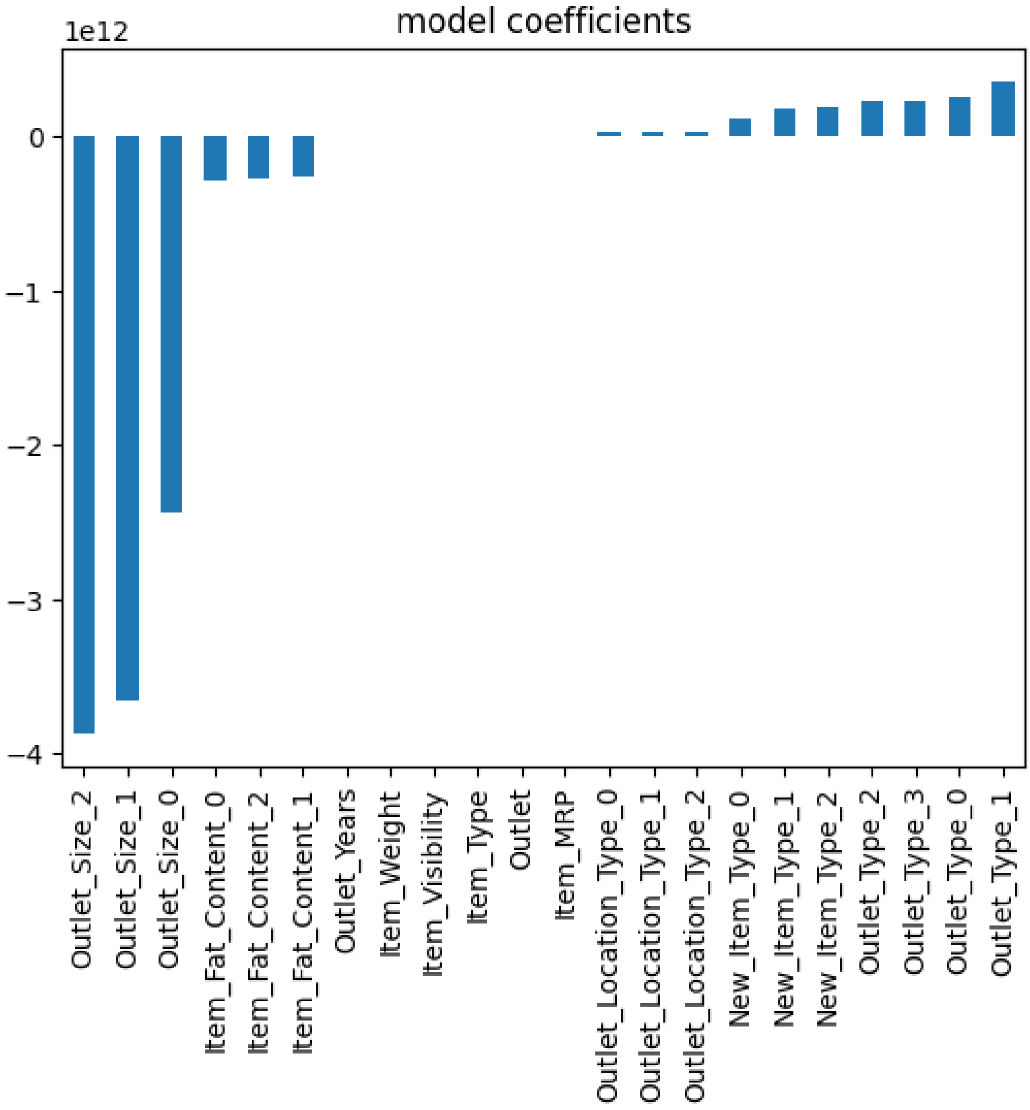
```
X =
df.drop(columns=['Outlet_Establishment_Year','Item_Identifier','Outlet
_Identifier','Item_Outlet_Sales'])
y = df['Item_Outlet_Sales']
```

##Model Training

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error
def train(model, X, y):
    #training the model
    model.fit(X,y)

    #predict the training dataset
    pred = model.predict(X)

    #performing cross validation
    cv_score =
cross_val_score(model,X,y,scoring='neg_mean_squared_error', cv=5)
    cv_score = np.abs(np.mean(cv_score))
    print('Model report')
    print('MSE: ', mean_squared_error(y,pred))
    print('CV score: ', cv_score)

from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
model = LinearRegression()
model.fit(X_scaled, y)
train(model,  X_scaled, y)
coef = pd.Series(model.coef_,X.columns).sort_values()
coef.plot(kind='bar', title = 'model coefficients')
```

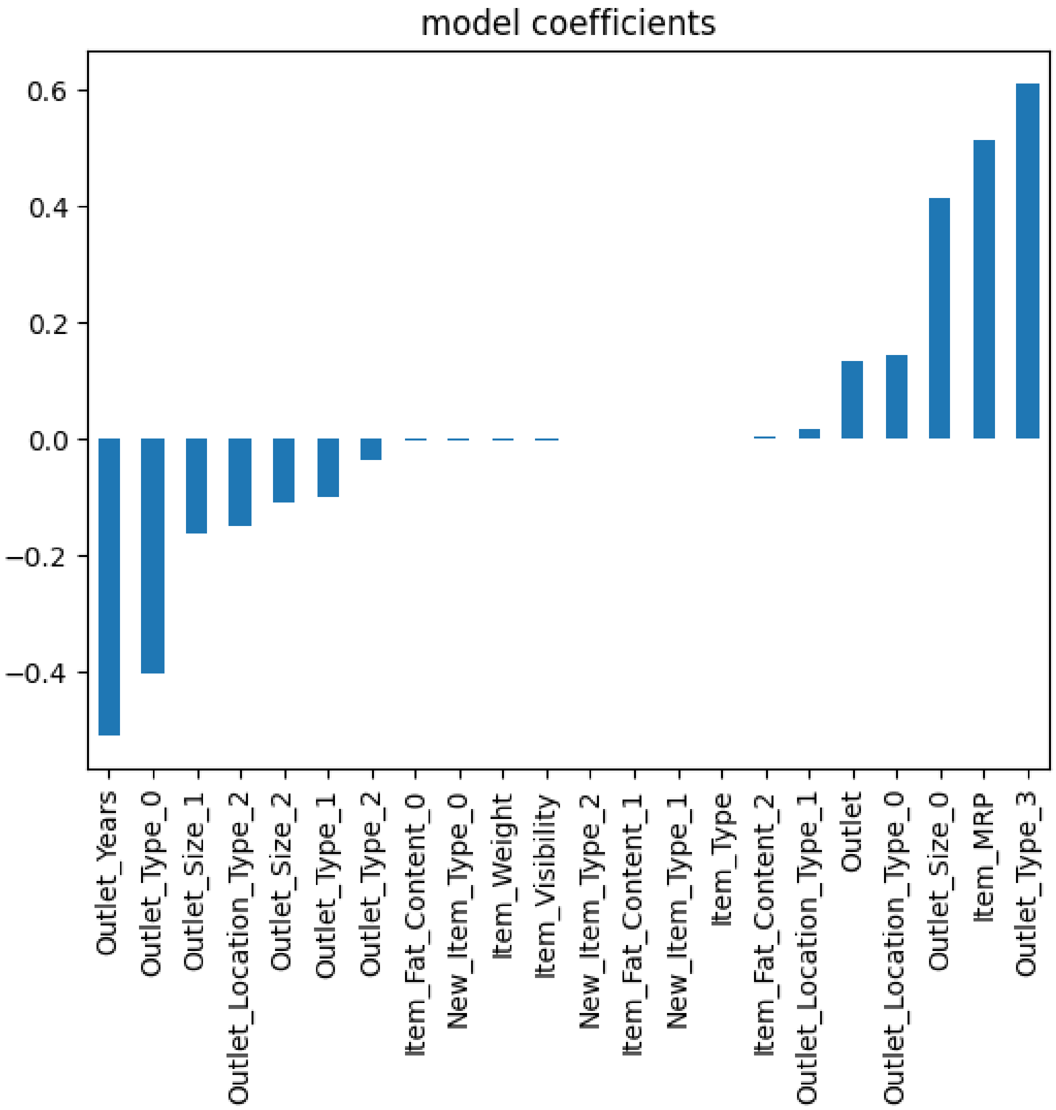
```
Model report
MSE:  0.28801853508326636
CV score:  0.2891617643352646
```

```
<Axes: title={'center': 'model coefficients'}>
```



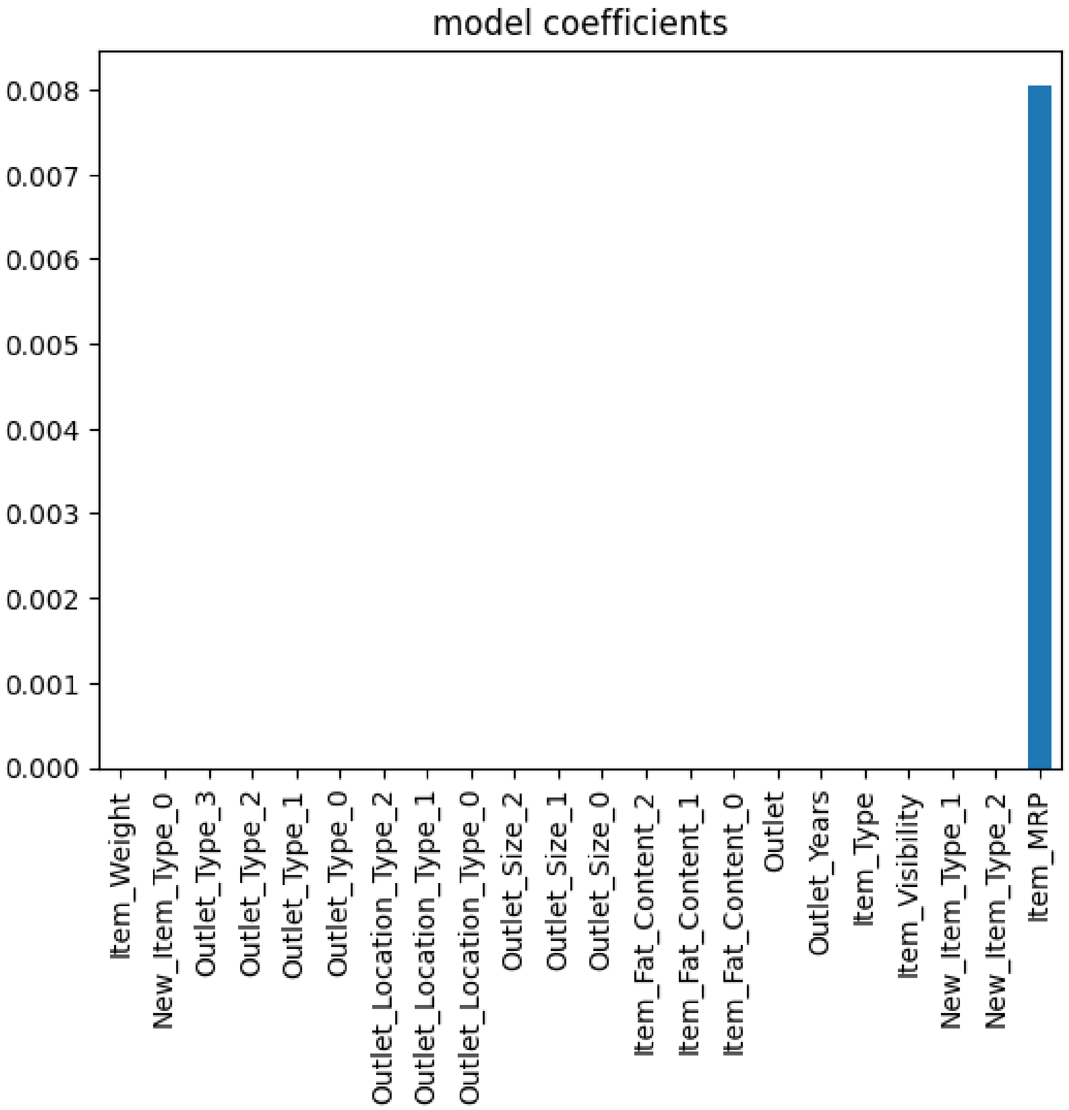
```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
model = Ridge()
model.fit(X_scaled, y)
train(model,  X_scaled, y)
coef = pd.Series(model.coef_,X.columns).sort_values()
coef.plot(kind='bar', title = 'model coefficients')
```

```
Model report
MSE:  0.2880361826180549
CV score:  0.2891442869461051

<Axes: title={'center': 'model coefficients'}>
```



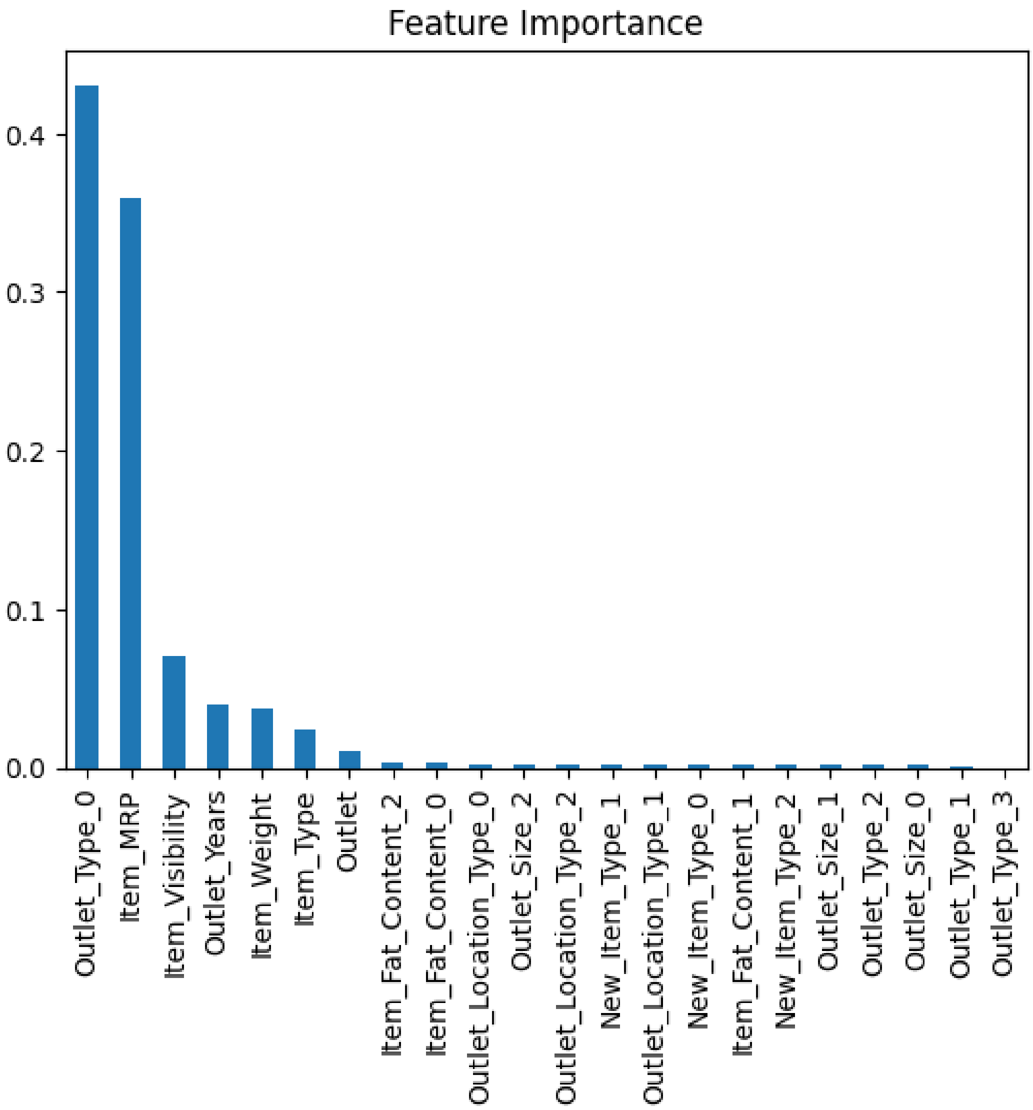
```
model = Lasso()
train(model, X, y)
coef = pd.Series(model.coef_,X.columns).sort_values()
coef.plot(kind='bar', title = 'model coefficients')
```

```
Model report
MSE:  0.7628688679102087
CV score:  0.7630789166281843

<Axes: title={'center': 'model coefficients'}>
```



```
from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
train(model, X, y)
coef =
pd.Series(model.feature_importances_,X.columns).sort_values(ascending=
False)
coef.plot(kind='bar', title = 'Feature Importance')
```

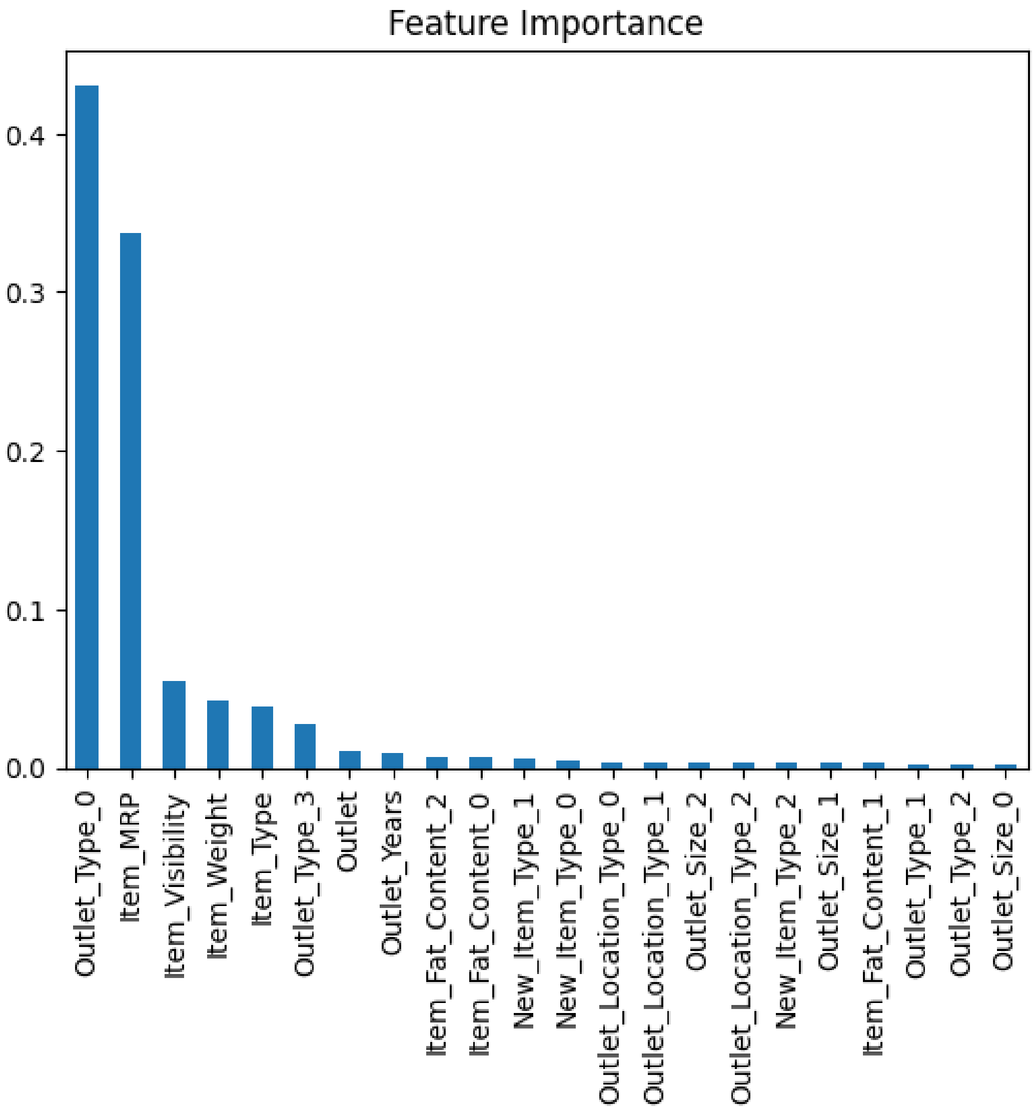
```
Model report
MSE:  5.5534030638578795e-34
CV score:  0.5765825702843724

<Axes: title={'center': 'Feature Importance'}>
```



```
from sklearn.ensemble import ExtraTreesRegressor
model = ExtraTreesRegressor()
train(model, X, y)
coef =
pd.Series(model.feature_importances_,X.columns).sort_values(ascending=
False)
coef.plot(kind='bar', title = 'Feature Importance')
```

```
Model report
MSE:  1.0418489584965893e-28
CV score:  0.3328216342943095
```

```
<Axes: title={'center': 'Feature Importance'}>
```


Feature Importance
0.4
0.3
0.2
0.1
0.0
Outlet_Type_0
Item_MRP
Item_Visibility
Item_Weight
Item_Type
Outlet_Type_3
Outlet
Outlet_Years
Item_Fat_Content_2
Item_Fat_Content_0
New_Item_Type_1
New_Item_Type_0
Outlet_Location_Type_0
Outlet_Location_Type_1
Outlet_Size_2
Outlet_Location_Type_2
New_Item_Type_2
Outlet_Size_1
Item_Fat_Content_1
Outlet_Type_1
Outlet_Type_2
Outlet_Size_0