

```
In [1]: import pandas as pd
import numpy as np
```

```
In [3]: d=pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/Big%20Sales%20D
```

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

Out[4]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Iden
0	FDT36	12.3	Low Fat	0.111448	Baking Goods	33.4874	OU'
1	FDT36	12.3	Low Fat	0.111904	Baking Goods	33.9874	OU'
2	FDT36	12.3	LF	0.111728	Baking Goods	33.9874	OU'
3	FDT36	12.3	Low Fat	0.000000	Baking Goods	34.3874	OU'
4	FDP12	9.8	Regular	0.045523	Baking Goods	35.0874	OU'

```
In [5]: d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14204 entries, 0 to 14203
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Item_Identifier    14204 non-null   object 
 1   Item_Weight        11815 non-null   float64
 2   Item_Fat_Content   14204 non-null   object 
 3   Item_Visibility    14204 non-null   float64
 4   Item_Type          14204 non-null   object 
 5   Item_MRP           14204 non-null   float64
 6   Outlet_Identifier  14204 non-null   object 
 7   Outlet_Establishment_Year  14204 non-null   int64  
 8   Outlet_Size        14204 non-null   object 
 9   Outlet_Location_Type  14204 non-null   object 
 10  Outlet_Type        14204 non-null   object 
 11  Item_Outlet_Sales  14204 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 1.3+ MB
```

```
In [6]: d.columns
```

```
Out[6]: Index(['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'],
  dtype='object')
```

In [7]: `d.describe()`

Out[7]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	11815.000000	14204.000000	14204.000000	14204.000000	14204.000000
mean	12.788355	0.065953	141.004977	1997.830681	2185.836320
std	4.654126	0.051459	62.086938	8.371664	1827.479550
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.710000	0.027036	94.012000	1987.000000	922.135101
50%	12.500000	0.054021	142.247000	1999.000000	1768.287680
75%	16.750000	0.094037	185.855600	2004.000000	2988.110400
max	30.000000	0.328391	266.888400	2009.000000	31224.726950

In [13]: `d['Item_Weight'].fillna(d.groupby(['Item_Type'])['Item_Weight'].transform('mean'))`

In [15]: `d.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14204 entries, 0 to 14203
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Item_Identifier    14204 non-null   object 
 1   Item_Weight         14204 non-null   float64
 2   Item_Fat_Content    14204 non-null   object 
 3   Item_Visibility     14204 non-null   float64
 4   Item_Type           14204 non-null   object 
 5   Item_MRP            14204 non-null   float64
 6   Outlet_Identifier   14204 non-null   object 
 7   Outlet_Establishment_Year 14204 non-null   int64  
 8   Outlet_Size          14204 non-null   object 
 9   Outlet_Location_Type 14204 non-null   object 
 10  Outlet_Type          14204 non-null   object 
 11  Item_Outlet_Sales    14204 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 1.3+ MB
```

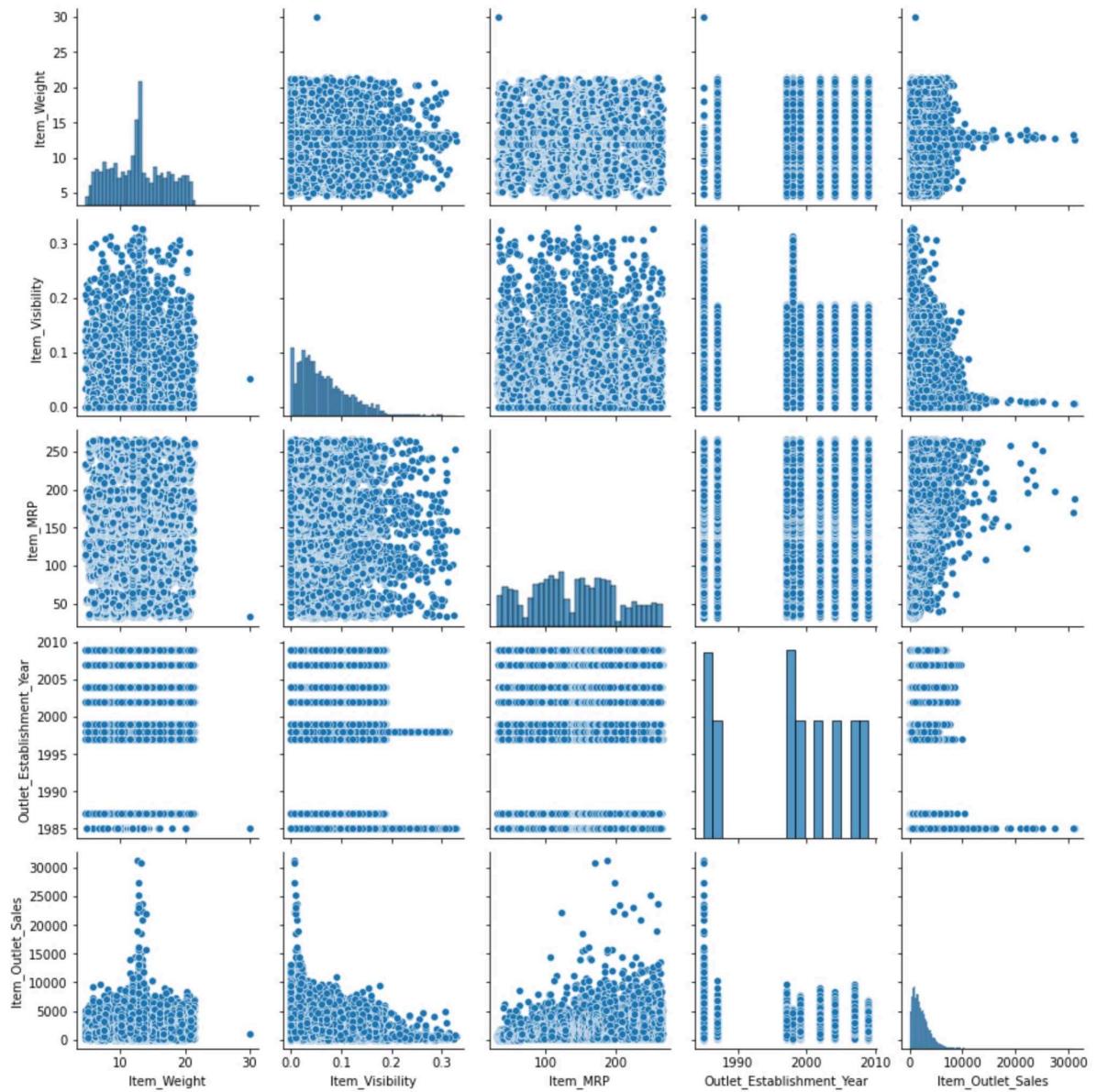
In [16]: `d.describe()`

Out[16]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	14204.000000	14204.000000	14204.000000	14204.000000	14204.000000
mean	12.790642	0.065953	141.004977	1997.830681	2185.836320
std	4.251186	0.051459	62.086938	8.371664	1827.479550
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	9.300000	0.027036	94.012000	1987.000000	922.135101
50%	12.800000	0.054021	142.247000	1999.000000	1768.287680
75%	16.000000	0.094037	185.855600	2004.000000	2988.110400
max	30.000000	0.328391	266.888400	2009.000000	31224.726950

```
In [17]: import seaborn as sns  
sns.pairplot(d)
```

```
Out[17]: <seaborn.axisgrid.PairGrid at 0x1e51e40eee0>
```



Categories and counts of categorical variables

```
In [18]: d[['Item_Identifier']].value_counts()
```

```
Out[18]: Item_Identifier
FDQ08          10
FD024          10
FDQ19          10
FDQ28          10
FDQ31          10
..
FDM52           7
FDM50           7
FDL50           7
FDM10           7
FDR51           7
Length: 1559, dtype: int64
```

```
In [19]: d['Item_Fat_Content'].value_counts()
```

```
Out[19]: Low Fat    8485
Regular     4824
LF          522
reg         195
low fat     178
Name: Item_Fat_Content, dtype: int64
```

```
In [22]: d.replace({'Item_Fat_Content':{'LF':'Low Fat','reg':'Regular','low fat':'Low Fat'}}
```

```
In [23]: d[['Item_Fat_Content']].value_counts()
```

```
Out[23]: Item_Fat_Content
Low Fat          9185
Regular         5019
dtype: int64
```

```
In [24]: d.replace({'Item_Fat_Content':{'Low Fat':0,'Regular':1}},inplace=True)
```

```
In [25]: d['Item_Type'].value_counts()
```

```
Out[25]: Fruits and Vegetables    2013  
Snack Foods                  1989  
Household                     1548  
Frozen Foods                  1426  
Dairy                          1136  
Baking Goods                  1086  
Canned                         1084  
Health and Hygiene            858  
Meat                           736  
Soft Drinks                    726  
Breads                         416  
Hard Drinks                    362  
Others                         280  
Starchy Foods                  269  
Breakfast                      186  
Seafood                        89  
Name: Item_Type, dtype: int64
```

```
In [28]: d.replace({'Item_Type': {'Fruits and Vegetables': 0, 'Snack Foods': 0, 'Household': 1}}, inplace=True)
```

```
In [29]: d['Item_Type'].value_counts()
```

```
Out[29]: 0    11518  
1    2406  
2    280  
Name: Item_Type, dtype: int64
```

```
In [30]: d['Outlet_Identifier'].value_counts()
```

```
Out[30]: OUT027    1559  
OUT013    1553  
OUT049    1550  
OUT046    1550  
OUT035    1550  
OUT045    1548  
OUT018    1546  
OUT017    1543  
OUT010    925  
OUT019    880  
Name: Outlet_Identifier, dtype: int64
```

```
In [34]: ['OUT027':0, 'OUT013':1, 'OUT049':2, 'OUT046':3, 'OUT035':4, 'OUT045':5, 'OUT018':6, 'OUT010':7, 'OUT017':8]
```

```
In [36]: d['Outlet_Identifier'].value_counts()
```

```
Out[36]: 0    1559  
1    1553  
2    1550  
3    1550  
4    1550  
5    1548  
6    1546  
7    1543  
8     925  
9     880  
Name: Outlet_Identifier, dtype: int64
```

```
In [38]: d['Outlet_Size'].value_counts()
```

```
Out[38]: Medium    7122  
Small      5529  
High       1553  
Name: Outlet_Size, dtype: int64
```

```
In [39]: d.replace({'Outlet_Size':{'Small':0,'Medium':1,'High':2}},inplace=True)
```

```
In [41]: d['Outlet_Size'].value_counts()
```

```
Out[41]: 1    7122  
0    5529  
2    1553  
Name: Outlet_Size, dtype: int64
```

```
In [42]: d['Outlet_Location_Type'].value_counts()
```

```
Out[42]: Tier 3    5583  
Tier 2    4641  
Tier 1    3980  
Name: Outlet_Location_Type, dtype: int64
```

```
In [43]: d.replace({'Outlet_Location_Type':{'Tier 1':0,'Tier 2':1,'Tier 3':2}},inplace=True)
```

```
In [44]: d['Outlet_Location_Type'].value_counts()
```

```
Out[44]: 2    5583  
1    4641  
0    3980  
Name: Outlet_Location_Type, dtype: int64
```

In [45]: `d['Outlet_Type'].value_counts()`

Out[45]:

Outlet_Type	Count
Supermarket Type1	9294
Grocery Store	1805
Supermarket Type3	1559
Supermarket Type2	1546

Name: Outlet_Type, dtype: int64

In [99]: `d.replace({'Outlet_Type': {'Grocery Store': 0, 'Supermarket Type1': 1, 'Supermarket Type2': 2, 'Supermarket Type3': 3}})`

In [100]: `d['Outlet_Type'].value_counts()`

Out[100]:

Outlet_Type	Count
1	9294
3	3105
0	1805

Name: Outlet_Type, dtype: int64

In [101]: `d.head()`

Out[101]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Iden
0	FDT36	12.3	0	0.111448	0	33.4874	
1	FDT36	12.3	0	0.111904	0	33.9874	
2	FDT36	12.3	0	0.111728	0	33.9874	
3	FDT36	12.3	0	0.000000	0	34.3874	
4	FDP12	9.8	1	0.045523	0	35.0874	

◀ ▶

In [102]: `d.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14204 entries, 0 to 14203
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Item_Identifier    14204 non-null   object  
 1   Item_Weight        14204 non-null   float64 
 2   Item_Fat_Content   14204 non-null   int64   
 3   Item_Visibility    14204 non-null   float64 
 4   Item_Type          14204 non-null   int64   
 5   Item_MRP           14204 non-null   float64 
 6   Outlet_Identifier  14204 non-null   int64   
 7   Outlet_Establishment_Year 14204 non-null   int64   
 8   Outlet_Size        14204 non-null   int64   
 9   Outlet_Location_Type 14204 non-null   int64   
 10  Outlet_Type        14204 non-null   int64   
 11  Item_Outlet_Sales  14204 non-null   float64 
dtypes: float64(4), int64(7), object(1)
memory usage: 1.3+ MB
```

```
In [103]: d.shape
```

```
Out[103]: (14204, 12)
```

```
In [104]: y=d['Item_Outlet_Sales']
```

```
In [105]: y.shape
```

```
Out[105]: (14204,)
```

```
In [106]: y
```

```
Out[106]: 0      436.608721  
1      443.127721  
2      564.598400  
3      1719.370000  
4      352.874000  
...  
14199    4984.178800  
14200    2885.577200  
14201    2885.577200  
14202    3803.676434  
14203    3644.354765  
Name: Item_Outlet_Sales, Length: 14204, dtype: float64
```

```
In [107]: .Type','Item_MRP','Outlet_Identifier','Outlet_Establishment_Year','Outlet_Size','O
```

```
In [108]: x=d.drop(['Item_Identifier','Item_Outlet_Sales'],axis=1)
```

```
In [109]: x.shape
```

```
Out[109]: (14204, 10)
```

In [110]: `x`

Out[110]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outle
0	12.300000		0	0.111448	0	33.4874	2
1	12.300000		0	0.111904	0	33.9874	7
2	12.300000		0	0.111728	0	33.9874	6
3	12.300000		0	0.000000	0	34.3874	9
4	9.800000		1	0.045523	0	35.0874	7
...
14199	12.800000		0	0.069606	0	261.9252	4
14200	12.800000		0	0.070013	0	262.8252	7
14201	12.800000		0	0.069561	0	263.0252	1
14202	13.659758		0	0.069282	0	263.5252	0
14203	12.800000		0	0.069727	0	263.6252	2

14204 rows × 10 columns

Get X Variables Standardized

In [111]: `from sklearn.preprocessing import StandardScaler`In [112]: `s=StandardScaler()`In [113]: `x_std=d[['Item_Weight','Item_Visibility','Item_MRP','Outlet_Establishment_Year']]`In [114]: `x_std=s.fit_transform(x_std)`In [115]: `x_std`

Out[115]: array([[-0.11541705, 0.88413635, -1.73178716, 0.13968068],
 [-0.11541705, 0.89300616, -1.72373366, 1.09531886],
 [-0.11541705, 0.88958331, -1.72373366, 1.3342284],
,
 [0.00220132, 0.07011952, 1.96538148, -1.29377659],
 [0.20444792, 0.06469366, 1.97343499, -1.53268614],
 [0.00220132, 0.07334891, 1.97504569, 0.13968068]])

In [116]: `lshment_Year']] = pd.DataFrame(x_std, columns=['Item_Weight','Item_Visibility','Item_MRP','Outlet_Establishment_Year'])`

In [117]:

x

Out[117]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outle
0	-0.115417	0	0.884136	0	-1.731787	2	
1	-0.115417	0	0.893006	0	-1.723734	7	
2	-0.115417	0	0.889583	0	-1.723734	6	
3	-0.115417	0	-1.281712	0	-1.717291	9	
4	-0.703509	1	-0.397031	0	-1.706016	7	
...
14199	0.002201	0	0.070990	0	1.947664	4	
14200	0.002201	0	0.078898	0	1.962160	7	
14201	0.002201	0	0.070120	0	1.965381	1	
14202	0.204448	0	0.064694	0	1.973435	0	
14203	0.002201	0	0.073349	0	1.975046	2	

14204 rows × 10 columns



Train Test Split

In [118]:

from sklearn.model_selection import train_test_split

In [119]:

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=252)

In [120]:

x_train.shape,x_test.shape,y_train.shape,y_test.shape

Out[120]:

((12783, 10), (1421, 10), (12783,), (1421,))

Model Train

In [121]:

from sklearn.ensemble import RandomForestRegressor
l=RandomForestRegressor(random_state=2529)

In [122]:

l.fit(x_train,y_train)

Out[122]:

```
RandomForestRegressor
RandomForestRegressor(random_state=2529)
```

Model Prediction

```
In [123]: y_pred=l.predict(x_test)
```

```
In [124]: y_pred.shape
```

```
Out[124]: (1421,)
```

```
In [125]: y_pred
```

```
Out[125]: array([1459.81352734, 726.65859122, 1917.87747221, ..., 2196.2426537 ,  
3286.85491915, 456.48837472])
```

Model Evaluation

```
In [126]: from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

```
In [127]: mean_squared_error(y_test,y_pred)
```

```
Out[127]: 1625874.2334575457
```

```
In [128]: mean_absolute_error(y_test,y_pred)
```

```
Out[128]: 829.99262104014
```

```
In [129]: r2_score(y_test,y_pred)
```

```
Out[129]: 0.5768090767897451
```

Visualisation of Actual and Predicted Results

```
In [130]: import matplotlib.pyplot as p  
p.scatter(y_test,y_pred)  
p.xlabel('Actual Price')  
p.ylabel('Predicted Price')  
p.title("Actual Price Vs Predicted Price")  
p.show()
```



```
In [ ]:
```