Big Sales Prediction using Random Forest Regressor

Get Understanding about Data set

There are 12 variables in dataset.

1. Item_Identifier
2. Item_Weight
3. Item_Fat_Content
4. Item_Visibility
5. Item_Type
6. Item_MRP
7. Outlet_Identifier
8. Outlet_Establishment_Year
9. Outlet_Size
10. Outlet_Location_Type
11. Outlet_Type
12. Item_Outlet_Sales

Import Library

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

```
In [2]:  import numpy as np
```

Import CSV as DataFrame

Use URL of file directly

```
In [3]:  df = pd.read_csv('https://github.com/YBIFoundation/Dataset/raw/main/Big
```

Get the First Five Rows of Dataframe

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

Out[4]:

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

Get Information of DataFrame

In [5]:
```
df.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
```

Get Column Names

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

Out[6]:
```
Index(['Item_Identifier', 'Item_Weight', 'Item_Fat_Content', 'Item_Vis
ibility',
       'Item_Type', 'Item_MRP', 'Outlet_Identifier',
       'Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Ty
pe',
       'Outlet_Type', 'Item_Outlet_Sales'],
      dtype='object')
```

Get the Summary Statistics

In [7]:

```
df.describe()
```

Out[7]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outl
count	11815.000000	14204.000000	14204.000000	14204.000000	14204
mean	12.788355	0.065953	141.004977	1997.830681	2185
std	4.654126	0.051459	62.086938	8.371664	1827
min	4.555000	0.000000	31.290000	1985.000000	33
25%	8.710000	0.027036	94.012000	1987.000000	922
50%	12.500000	0.054021	142.247000	1999.000000	1768
75%	16.750000	0.094037	185.855600	2004.000000	2988
max	30.000000	0.328391	266.888400	2009.000000	31224

Get Missing Values Complete

In [8]:

```
df['Item_Weight'].fillna(df.groupby(['Item_Type'])['Item_Weight'].trans
```

```
/tmp/ipykernel_18/3547494075.py:1: FutureWarning: A value is trying to
be set on a copy of a DataFrame or Series through chained assignment u
sing an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values al
ways behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try usi
ng 'df.method({col: value}, inplace=True)' or df[col] = df[col].method
(value) instead, to perform the operation inplace on the original obje
ct.


  df['Item_Weight'].fillna(df.groupby(['Item_Type'])['Item_Weight'].tr
ansform('median'), inplace=True)
```

In [9]:

```
df.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 [10]:

```
df.describe()
```

Out[10]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outl
count	14204.000000	14204.000000	14204.000000	14204.000000	14204
mean	12.742842	0.065953	141.004977	1997.830681	2185
std	4.257583	0.051459	62.086938	8.371664	1827
min	4.555000	0.000000	31.290000	1985.000000	33
25%	9.300000	0.027036	94.012000	1987.000000	922
50%	12.600000	0.054021	142.247000	1999.000000	1768
75%	16.000000	0.094037	185.855600	2004.000000	2988
max	30.000000	0.328391	266.888400	2009.000000	31224

In [11]:

```
# remove outlier
from scipy import stats
df = df[np.abs(stats.zscore(df['Item_Outlet_Sales'])) < 2]
```

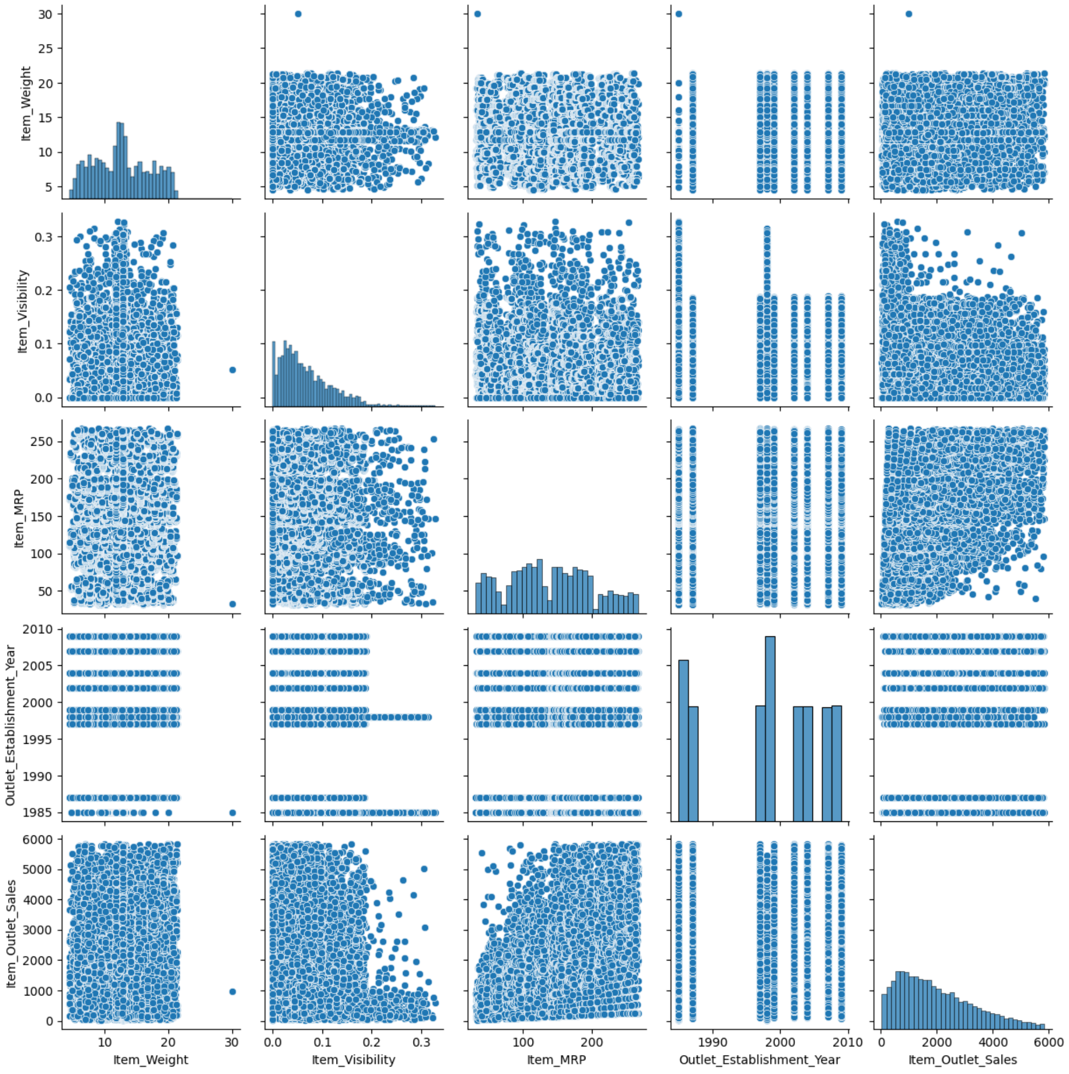
In [12]:

```
# pair plot
import seaborn as sns
sns.pairplot(df)
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: Futu
reWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[12]:

```
<seaborn.axisgrid.PairGrid at 0x7bdec6aef4f0>
```



Get Categories and Counts of Categorical Variables

In [13]:

```
df[['Item_Identifier']].value_counts()
```

Out[13]:

```
Item_Identifier
FDQ08              10
FDD58              10
FDD56              10
FDD53              10
FDY36              10
                   ..
FDC02               5
FDY55               5
NCL42               5
FDT21               5
FDA15               3
Name: count, Length: 1559, dtype: int64
```

In [14]:

```
df[['Item_Fat_Content']].value_counts()
```

Out[14]:

```
Item_Fat_Content
Low Fat             8173
Regular             4665
LF                   512
reg                  190
low fat              170
Name: count, dtype: int64
```

In [15]:

```
df.replace({'Item_Fat_Content': {'LF':'Low Fat','reg':'Regular', 'low f
```

In [16]:

```
df[['Item_Fat_Content']].value_counts()
```

Out[16]:

```
Item_Fat_Content
Low Fat             8855
Regular             4855
Name: count, dtype: int64
```

In [17]:

```
df.replace({'Item_Fat_Content': {'Low Fat': 0,'Regular' : 1}}, inplace=
```

```
/tmp/ipykernel_18/722552587.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Item_Fat_Content': {'Low Fat': 0,'Regular' : 1}}, inpla
ce=True)
```

In [18]:

```
df[['Item_Type']].value_counts()
```

Out[18]:

```
Item_Type
Fruits and Vegetables    1939
Snack Foods              1917
Household                1491
Frozen Foods             1372
Dairy                    1091
Baking Goods             1062
Canned                   1048
Health and Hygiene        828
Meat                      710
Soft Drinks               703
Breads                    406
Hard Drinks               347
Others                    273
Starchy Foods             261
Breakfast                 177
Seafood                    85
Name: count, dtype: int64
```

In [19]:

```
df.replace({'Item_Type':{'Fruits and Vegetables':0,'Snack Foods':0,'Hou
                         'Frozen Foods' : 0, 'Dairy' : 0, 'Baking Goods
                         'Canned' : 0, 'Health and Hygiene' : 1,
                         'Meat' : 0, 'Soft Drinks' : 0, 'Breads' : 0, '
                         'Others' : 2,'Starchy Foods' : 0, 'Breakfast'
                         }},inplace=True)
```

```
/tmp/ipykernel_18/866940864.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Item_Type':{'Fruits and Vegetables':0,'Snack Foods':
0,'Household':1,
```

In [20]:

```
df[['Item_Type']].value_counts()
```

Out[20]:

```
Item_Type
0            11118
1             2319
2              273
Name: count, dtype: int64
```

In [21]:

```
df[['Outlet_Identifier']].value_counts()
```

Out[21]:

```
Outlet_Identifier
OUT018               1529
OUT046               1529
OUT013               1525
OUT045               1523
OUT049               1520
OUT035               1517
OUT017               1511
OUT027               1284
OUT010                925
OUT019                847
Name: count, dtype: int64
```

In [22]:

```
df.replace({'Outlet_Identifier':{'OUT027': 0,'OUT013': 1,
                         'OUT049' : 2, 'OUT046' : 3, 'OUT035' : 4,
                         'OUT045' : 5, 'OUT018' : 6,
                         'OUT017' : 7, 'OUT010' : 8, 'OUT019' : 9,
                         }},inplace=True)
```

```
/tmp/ipykernel_18/2523128275.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Outlet_Identifier':{'OUT027': 0,'OUT013': 1,
```

In [23]:

```
df[['Outlet_Identifier']].value_counts()
```

Out[23]:

```
Outlet_Identifier
3                    1529
6                    1529
1                    1525
5                    1523
2                    1520
4                    1517
7                    1511
0                    1284
8                     925
9                     847
Name: count, dtype: int64
```

In [24]:

```
df[['Outlet_Size']].value_counts()
```

Out[24]:

```
Outlet_Size
Medium         6768
Small          5417
High           1525
Name: count, dtype: int64
```

In [25]:

```
df.replace({'Outlet_Size': {'Small': 0,'Medium' : 1, 'High' : 1}}, inpl
```

```
/tmp/ipykernel_18/171770719.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Outlet_Size': {'Small': 0,'Medium' : 1, 'High' : 1}}, i
nplace=True)
```

In [26]:

```
df[['Outlet_Size']].value_counts()
```

Out[26]:

```
Outlet_Size
1              8293
0              5417
Name: count, dtype: int64
```

In [27]:

```
df[['Outlet_Location_Type']].value_counts()
```

Out[27]:

```
Outlet_Location_Type
Tier 3                  5263
Tier 2                  4551
Tier 1                  3896
Name: count, dtype: int64
```

In [28]:

```
df.replace({'Outlet_Location_Type': {'Tier 1': 0,'Tier 2' : 1, 'Tier 3'
```

```
/tmp/ipykernel_18/941750987.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Outlet_Location_Type': {'Tier 1': 0,'Tier 2' : 1, 'Tier
3' : 2}}, inplace=True)
```

In [29]:

```
df[['Outlet_Location_Type']].value_counts()
```

Out[29]:

```
Outlet_Location_Type
2                       5263
1                       4551
0                       3896
Name: count, dtype: int64
```

In [30]:

```
df[['Outlet_Type']].value_counts()
```

Out[30]:

```
Outlet_Type
Supermarket Type1    9125
Grocery Store        1772
Supermarket Type2    1529
Supermarket Type3    1284
Name: count, dtype: int64
```

In [31]:

```
df.replace({'Outlet_Type': {'Grocery Store': 0,'Supermarket Type1' : 1,
```

```
/tmp/ipykernel_18/1904487509.py:1: FutureWarning: Downcasting behavior
in `replace` is deprecated and will be removed in a future version. To
retain the old behavior, explicitly call `result.infer_objects(copy=Fa
lse)`. To opt-in to the future behavior, set `pd.set_option('future.no
_silent_downcasting', True)`
  df.replace({'Outlet_Type': {'Grocery Store': 0,'Supermarket Type1' :
1, 'Supermarket Type2' : 2, 'Supermarket Type3': 3}}, inplace=True)
```

In [32]:

```
df[['Outlet_Type']].value_counts()
```

Out[32]:

```
Outlet_Type
1              9125
0              1772
2              1529
3              1284
Name: count, dtype: int64
```

Define y (dependent or label or target variable) and X (independent or features or attribute Variable)

In [33]:

```
# check correlation
df.describe().corr()
```

Out[33]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Ite
Item_Weight	1.000000	0.999999	0.999999	0.999999	(
Item_Fat_Content	0.999999	1.000000	1.000000	1.000000	(
Item_Visibility	0.999999	1.000000	1.000000	1.000000	(
Item_Type	0.999999	1.000000	1.000000	1.000000	(
Item_MRP	0.999907	0.999885	0.999884	0.999885	1
Outlet_Identifier	0.999999	1.000000	1.000000	1.000000	(
Outlet_Establishment_Year	0.986922	0.986804	0.986805	0.986804	(
Outlet_Size	0.999999	1.000000	1.000000	1.000000	(
Outlet_Location_Type	0.999999	1.000000	1.000000	1.000000	(
Outlet_Type	0.999999	1.000000	1.000000	1.000000	(
Item_Outlet_Sales	0.922230	0.921632	0.921613	0.921652	(

In [34]:

```
y = df['Item_Outlet_Sales']
```

In [35]:

```
X = df[['Item_Weight', 'Item_Fat_Content', 'Item_Visibility',
        'Item_Type', 'Item_MRP', 'Outlet_Identifier',
        'Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Typ
        'Outlet_Type']]
```

or use .drop function to define X

In [36]:

```
X = df.drop(['Item_Identifier', 'Item_Outlet_Sales'], axis=1)
```

Get Train Test Split

In [37]:

```
from sklearn.model_selection import train_test_split
```

In [38]:

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.
```

In [39]:

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[39]:

```
((10968, 10), (2742, 10), (10968,), (2742,))
```

Get Model Train

In [40]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [41]:

```
rfr = RandomForestRegressor()
```

In [42]:

```
rfr.fit(X_train, y_train)
```

Out[42]:

```
RandomForestRegressor()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Get Model Prediction

In [43]:

```
y_pred = rfr.predict(X_test)
```

Get Model Evaluation

In [44]:

```
from sklearn.metrics import mean_absolute_error
```

In [45]:

```
mean_absolute_error(y_test, y_pred)
```

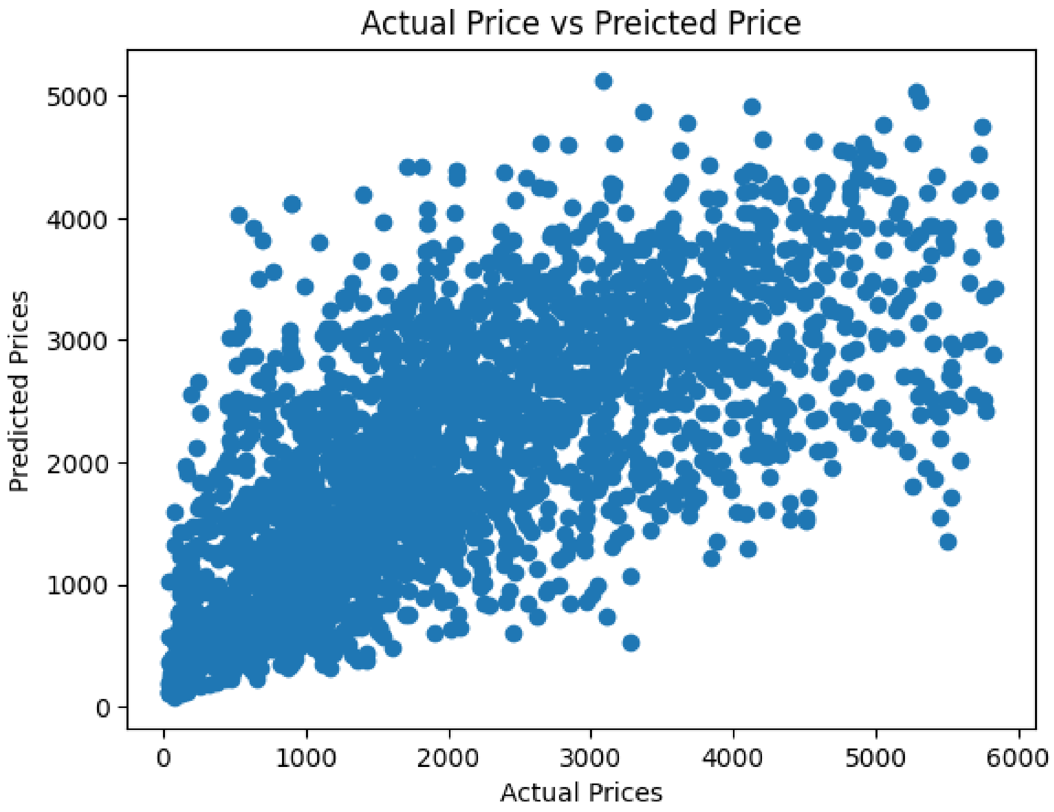
Out[45]:

```
692.0422332454967
```

Get Visualization of Actual Vs Predicted Results

In [46]:

```
import matplotlib.pyplot as plt
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual Price vs Preicted Price")
plt.show()
```


Actual Price vs Preicted Price
Predicted Prices
Actual Prices