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

In [2]:
```python
train = pd.read_csv(r'D:\DS\ML\bigmart_train.csv')
test = pd.read_csv(r'D:\DS\ML\bigmart_test.csv')
```

In [3]:
```python
train
```

Out[3]:

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

8523 rows × 12 columns

In [4]:
```python
test
```

Out[4]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_
0	FDW58	20.750	Low Fat	0.007565	Snack Foods	107.8622	OUT049	1999	Mec
1	FDW14	8.300	reg	0.038428	Dairy	87.3198	OUT017	2007	
2	NCN55	14.600	Low Fat	0.099575	Others	241.7538	OUT010	1998	
3	FDQ58	7.315	Low Fat	0.015388	Snack Foods	155.0340	OUT017	2007	
4	FDY38	NaN	Regular	0.118599	Dairy	234.2300	OUT027	1985	Mec
...	...	...	...	...	...	...	...	...	
5676	FDB58	10.500	Regular	0.013496	Snack Foods	141.3154	OUT046	1997	S
5677	FDD47	7.600	Regular	0.142991	Starchy Foods	169.1448	OUT018	2009	Mec
5678	NCO17	10.000	Low Fat	0.073529	Health and Hygiene	118.7440	OUT045	2002	
5679	FDJ26	15.300	Regular	0.000000	Canned	214.6218	OUT017	2007	
5680	FDU37	9.500	Regular	0.104720	Canned	79.7960	OUT045	2002	

5681 rows × 11 columns

Merging train and test data for preprocessing

In [5]:
```python
trainlen = len(train)
testlen = len(test)
print(trainlen,testlen)
```

```
8523 5681
```

In [6]:
```python
df = pd.concat([train,test],axis=0)
df.reset_index(drop = True, inplace=True)
df
```

Out[6]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outle
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Me
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Me
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Me
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	
...	...	...	...	...	...	...	...	...	
14199	FDB58	10.50	Regular	0.013496	Snack Foods	141.3154	OUT046	1997	
14200	FDD47	7.60	Regular	0.142991	Starchy Foods	169.1448	OUT018	2009	Me
14201	NCO17	10.00	Low Fat	0.073529	Health and Hygiene	118.7440	OUT045	2002	
14202	FDJ26	15.30	Regular	0.000000	Canned	214.6218	OUT017	2007	
14203	FDU37	9.50	Regular	0.104720	Canned	79.7960	OUT045	2002	

14204 rows × 12 columns

Cleaning and preprocessing

In [7]:
```python
df.isna().sum()
```

Out[7]:
```
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
dtype: int64
```

In [8]:
```python
df['Item_Type'].unique()
```

Out[8]:
```
array(['Dairy', 'Soft Drinks', 'Meat', 'Fruits and Vegetables',
       'Household', 'Baking Goods', 'Snack Foods', 'Frozen Foods',
       'Breakfast', 'Health and Hygiene', 'Hard Drinks', 'Canned',
       'Breads', 'Starchy Foods', 'Others', 'Seafood'], dtype=object)
```

In [9]:
```python
median_weights = df.groupby(df['Item_Type'],as_index=False).agg({'Item_Weight':np.median})
median_weights.columns = ['Item_Type','Average_Weight']
median_weights
```

Out[9]:

	Item_Type	Average_Weight
0	Baking Goods	11.650
1	Breads	10.500
2	Breakfast	11.600
3	Canned	12.150
4	Dairy	13.300
5	Frozen Foods	12.300
6	Fruits and Vegetables	13.100
7	Hard Drinks	10.195
8	Health and Hygiene	12.350
9	Household	13.000
10	Meat	12.350
11	Others	14.500
12	Seafood	11.650
13	Snack Foods	12.850
14	Soft Drinks	11.800
15	Starchy Foods	12.850

In [10]:
```python
df = pd.merge(left = df, right = median_weights, on='Item_Type', how = 'left')
df.head()
```

Out[10]:

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

In [11]:
```python
df.shape
```

Out[11]:
```
(14204, 13)
```

In [12]:
```python
df['Item_Weight'] = df.apply(lambda x: x[12] if pd.isna(x[1]) else x[1], axis=1)
```

In [13]:
```python
df['Outlet_Size'].unique()
```

Out[13]:
```
array(['Medium', nan, 'High', 'Small'], dtype=object)
```

In [14]:
```python
df['Outlet_Size'].value_counts()
```

Out[14]:
```
Medium    4655
Small     3980
High      1553
Name: Outlet_Size, dtype: int64
```

In [15]:
```python
df['Outlet_Size'] = df['Outlet_Size'].apply(lambda x: 'Medium' if pd.isna(x) else x)
```

In [16]:
```python
df.isna().sum()
```

Out[16]:
```
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
Average_Weight                  0
dtype: int64
```

In [17]:
```python
df['Item_Fat_Content'].unique()
```

Out[17]:
```
array(['Low Fat', 'Regular', 'low fat', 'LF', 'reg'], dtype=object)
```

In [18]:
```python
df['Item_Fat_Content'] = df['Item_Fat_Content'].map({'Low Fat':'Low Fat','Regular':'Regular','low fat':'Low Fat
```

In [19]:
```python
df['Item_Fat_Content'].unique()
```

Out[19]:
```
array(['Low Fat', 'Regular'], dtype=object)
```

In [20]:
```python
df['Outlet_Location_Type'].unique()
```

Out[20]:
```
array(['Tier 1', 'Tier 3', 'Tier 2'], dtype=object)
```

In [21]:
```python
df['Outlet_Type'].unique()
```

Out[21]:
```
array(['Supermarket Type1', 'Supermarket Type2', 'Grocery Store',
       'Supermarket Type3'], dtype=object)
```

In [22]:
```python
df['New_Type'] = df['Item_Identifier'].apply(lambda x:x[:2])
df['New_Type']
```

Out[22]:
```
0        FD
1        DR
2        FD
3        FD
4        NC
         ..
14199    FD
14200    FD
14201    NC
14202    FD
14203    FD
Name: New_Type, Length: 14204, dtype: object
```

In [23]:
```python
df['New_Type'].unique()
```

Out[23]:
```
array(['FD', 'DR', 'NC'], dtype=object)
```

In [24]:
```python
df['New_Type'] = df['New_Type'].map({'FD':'Food','DR':'Drinks','NC':'Non-consumables'})
df['New_Type'].unique()
```

Out[24]:
```
array(['Food', 'Drinks', 'Non-consumables'], dtype=object)
```

In [25]:
```python
df['Outlet_Age'] = df['Outlet_Establishment_Year'].apply(lambda x: 2021-x)
```

In [26]:
```python
df = pd.get_dummies(columns=['Item_Fat_Content','Item_Type','Outlet_Size','Outlet_Location_Type','Outlet_Type',
```

In [27]:
```python
df.head()
```

Out[27]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Item_Outlet_Sales	Average_Weight	Outl
0	FDA15	9.30	0.016047	249.8092	OUT049	1999	3735.1380	13.30	
1	DRC01	5.92	0.019278	48.2692	OUT018	2009	443.4228	11.80	
2	FDN15	17.50	0.016760	141.6180	OUT049	1999	2097.2700	12.35	
3	FDX07	19.20	0.000000	182.0950	OUT010	1998	732.3800	13.10	
4	NCD19	8.93	0.000000	53.8614	OUT013	1987	994.7052	13.00	

5 rows × 34 columns

In [28]:
```python
(df['Item_Visibility']==0).sum()
```

Out[28]:
```
879
```

In [29]:
```python
df['Item_Visibility'] = df['Item_Visibility'].apply(lambda x:np.mean(df['Item_Visibility']) if x==0 else x)
```

In [30]:
```python
df.head()
```

Out[30]:

	Item_Identifier	Item_Weight	Item_Visibility	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Item_Outlet_Sales	Average_Weight	Outl
0	FDA15	9.30	0.016047	249.8092	OUT049	1999	3735.1380	13.30	
1	DRC01	5.92	0.019278	48.2692	OUT018	2009	443.4228	11.80	
2	FDN15	17.50	0.016760	141.6180	OUT049	1999	2097.2700	12.35	
3	FDX07	19.20	0.065953	182.0950	OUT010	1998	732.3800	13.10	
4	NCD19	8.93	0.065953	53.8614	OUT013	1987	994.7052	13.00	

5 rows × 34 columns

In [31]:
```python
df.drop(['Item_Identifier','Outlet_Identifier','Average_Weight','Outlet_Establishment_Year'],axis=1,inplace=Tru
```

In [32]:
```python
df.head()
```

Out[32]:

	Item_Weight	Item_Visibility	Item_MRP	Item_Outlet_Sales	Outlet_Age	Item_Fat_Content_Regular	Item_Type_Breads	Item_Type_Breakfast	Ite
0	9.30	0.016047	249.8092	3735.1380	22	0	0	0	
1	5.92	0.019278	48.2692	443.4228	12	1	0	0	
2	17.50	0.016760	141.6180	2097.2700	22	0	0	0	
3	19.20	0.065953	182.0950	732.3800	23	1	0	0	
4	8.93	0.065953	53.8614	994.7052	34	0	0	0	

5 rows × 30 columns

In [33]:
```python
train = df.iloc[:trainlen,:]
test = df.iloc[trainlen:,:]
```

In [34]:
```python
train.head()
```

Out[34]:

	Item_Weight	Item_Visibility	Item_MRP	Item_Outlet_Sales	Outlet_Age	Item_Fat_Content_Regular	Item_Type_Breads	Item_Type_Breakfast	Ite
0	9.30	0.016047	249.8092	3735.1380	22	0	0	0	
1	5.92	0.019278	48.2692	443.4228	12	1	0	0	
2	17.50	0.016760	141.6180	2097.2700	22	0	0	0	
3	19.20	0.065953	182.0950	732.3800	23	1	0	0	
4	8.93	0.065953	53.8614	994.7052	34	0	0	0	

5 rows × 30 columns

In [35]:
```python
test.head()
```

Out[35]:

	Item_Weight	Item_Visibility	Item_MRP	Item_Outlet_Sales	Outlet_Age	Item_Fat_Content_Regular	Item_Type_Breads	Item_Type_Breakfast
8523	20.750	0.007565	107.8622	NaN	22	0	0	0
8524	8.300	0.038428	87.3198	NaN	14	1	0	0
8525	14.600	0.099575	241.7538	NaN	23	0	0	0
8526	7.315	0.015388	155.0340	NaN	14	0	0	0
8527	13.300	0.118599	234.2300	NaN	36	1	0	0

5 rows × 30 columns

In [36]:
```python
test = test.drop(columns = ['Item_Outlet_Sales'])
test.head()
```

Out[36]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Age	Item_Fat_Content_Regular	Item_Type_Breads	Item_Type_Breakfast	Item_Type_Canned
8523	20.750	0.007565	107.8622	22	0	0	0	0
8524	8.300	0.038428	87.3198	14	1	0	0	0
8525	14.600	0.099575	241.7538	23	0	0	0	0
8526	7.315	0.015388	155.0340	14	0	0	0	0
8527	13.300	0.118599	234.2300	36	1	0	0	0

5 rows × 29 columns

In [37]:
```python
x = train.drop(columns=['Item_Outlet_Sales'])
y = train['Item_Outlet_Sales']
```

Training the model

In [44]:
```python
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, r2_score

def train(model,x,y):
    model.fit(x,y)
    pred = model.predict(x)

    cv_score = cross_val_score(model,x,y,scoring = 'neg_mean_squared_error')
    cv_score = np.abs(np.mean(cv_score))
    R2_score = r2_score(y,pred)

    print('MSE:',mean_squared_error(y,pred,squared=False))
    print('cv_score:', cv_score)
    print('r2_score:', R2_score)
```

In [45]:
```python
from sklearn.linear_model import LinearRegression
model = LinearRegression(normalize = True)
train(model,x,y)
```

```
MSE: 1127.4628067269703
cv_score: 1283914.2484044794
r2_score: 0.5634409285050267
```

In [48]:
```python
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
train(model,x,y)
```

```
MSE: 428.03505468093437
cv_score: 1327580.4206857528
r2_score: 0.9370787640947891
```

Prediction on test data using Random Forest Regressor

In [50]:
```python
model = RandomForestRegressor()
model.fit(x,y)
predict = model.predict(test)
```

In [51]:
```python
predict
```

Out[51]:
```
array([1657.316018, 1334.722602,  681.566144, ..., 1845.344596,
       4342.107912, 1580.262984])
```