Big Mart Sales prediction

Problem Statement:

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 find out the sales of each product at a particular store.
Using this model, BigMart will try to understand the properties of products and stores 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.

Data Description:

We have train (8523) and test (5681) data set, train data set has both input and output variable(s).We need to predict the sales for test data set.

Variable Description:
Item_Identifier Unique product ID
Item_Weight Weight of product
Item_Fat_Content Whether the product is low fat or not
Item_Visibility The % of total display area of all products in a store allocated to the particular product
Item_Type The category to which the product belongs
Item_MRP Maximum Retail Price (list price) of the product
Outlet_Identifier Unique store ID
Outlet_Establishment_Year The year in which store was established
Outlet_Size The size of the store in terms of ground area covered
Outlet_Location_Type The type of city in which the store is located
Outlet_Type Whether the outlet is just a grocery store or some sort of supermarket
Item_Outlet_Sales Sales of the product in the particular store. This is the outcome variable to be predicted.

To solve this I have followed following steps:

1. *Data Exploration*
2. *Data Cleaning*
3. *Feature Engineering*
4. *Model Building*
5. *Conclusion based on RMSE value*

I have perform these stages with following stages viz

Stage 1: Loading required Libraries
Stage 2: Loading the data files from the local disk
Stage 3: Exploring the data
Stage 4: Visualizing the Data for better understanding
Stage 5A: Imputing NA values for continuous variables by replacing them with Mean
Stage 5B: Imputing NA values for Categorical variables by replacing them with Mode
Stage 6: Feature Engineering
Stage 7: Model Building

Stage 1: Loading required Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
#Import mode function:
from scipy.stats import mode
from sklearn.linear_model import *
import csv as csv
from scipy.stats import mode
from sklearn import cross_validation, metrics
```

Stage 2: Loading the data files from the local disk

Loaded files from the local disk and stored in the variables viz train and test:

```
train = pd.read_csv("F://Python//train.csv")
test = pd.read_csv("F://Python//test.csv")
```

Stage 3: Exploring the data

Dataset	Rows#	Columns#
Train	8523	13
Test	5681	12
Combined test and train as data	14204	13

Here I have combine both train and test data sets into one, will perform feature engineering and then divide them later again. This saves the trouble of performing the same steps twice on test and train.

Checked for NA values with `isnull()`
I have found NA values in Item_Weight and Outlet_Size that I have imputed in step 4.

I have checked basic statistics using `describe()`

	Item_MRP	Item_Outlet_Sales	Item_Visibility	Item_Weight	Outlet_Establishment_Year
count	14204.000000	8523.000000	14204.000000	11765.000000	14204.000000
mean	141.004977	2181.288914	0.065953	12.792854	1997.830681
std	62.086938	1706.499616	0.051459	4.652502	8.371664
min	31.290000	33.290000	0.000000	4.555000	1985.000000
25%	94.012000	834.247400	0.027036	8.710000	1987.000000
50%	142.247000	1794.331000	0.054021	12.600000	1999.000000
75%	185.855600	3101.296400	0.094037	16.750000	2004.000000
max	266.888400	13086.964800	0.328391	21.350000	2009.000000

It is cleared from above information that Outlet_Establishment_Years vary from 1985 to 2009. Item_Visibility has a min value of zero. The Item is not visible cannot be sold so this cannot be zero.

With `unique()` I found that there are there are 1559 products and 10 outlets.

```
Item_Fat_Content                 5
Item_Identifier               1559
Item_MRP                      8052
Item_Outlet_Sales             3494
Item_Type                       16
Item_Visibility              13006
Item_Weight                    416
Outlet_Establishment_Year        9
Outlet_Identifier               10
Outlet_Location_Type             3
Outlet_Size                      4
Outlet_Type                      4
source                           2
dtype: int64
```

I have excluded the ID and source variables for obvious reasons. Then Filtered categorical variables and printed frequency of categories.

The output gives us following observations:

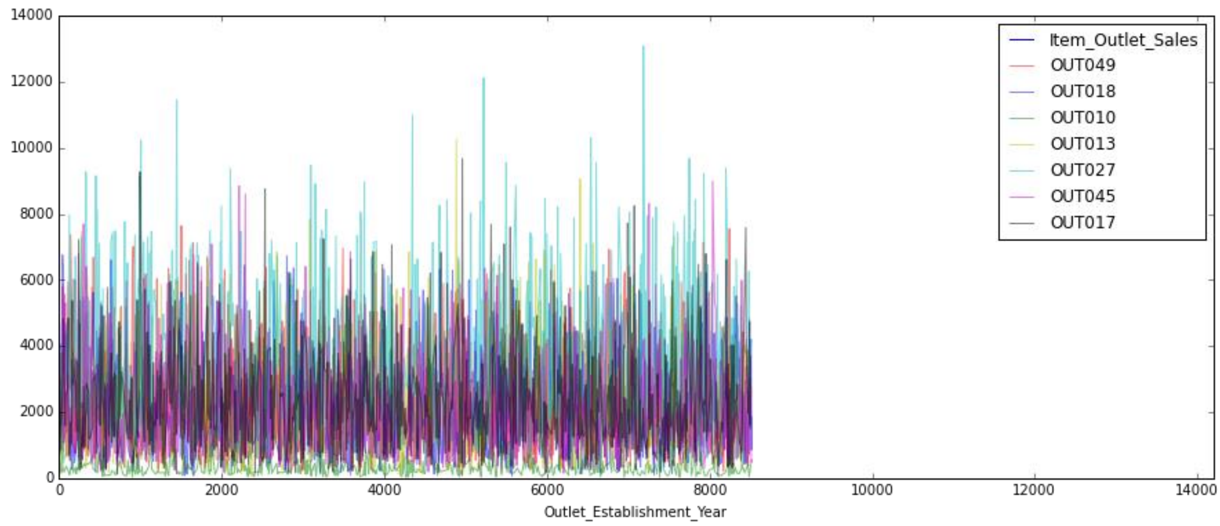
1. **Item_Fat_Content:** Some of ‘Low Fat’ values mis-coded as ‘low fat’ and ‘LF’. Also, some of ‘Regular’ are mentioned as ‘regular’.

2. **Item_Type:** Not all categories have substantial numbers. It looks like combining them can give better results.

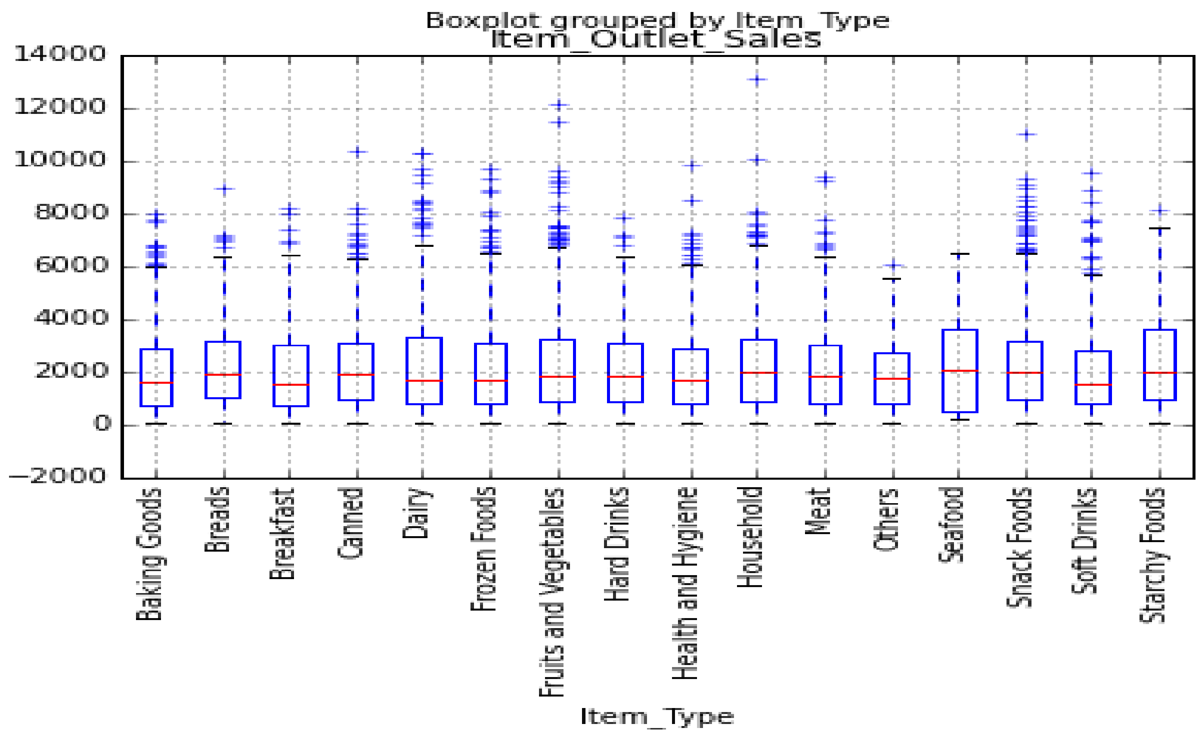
Stage 4: Visualizing the Data for better understanding

I have got following insights.

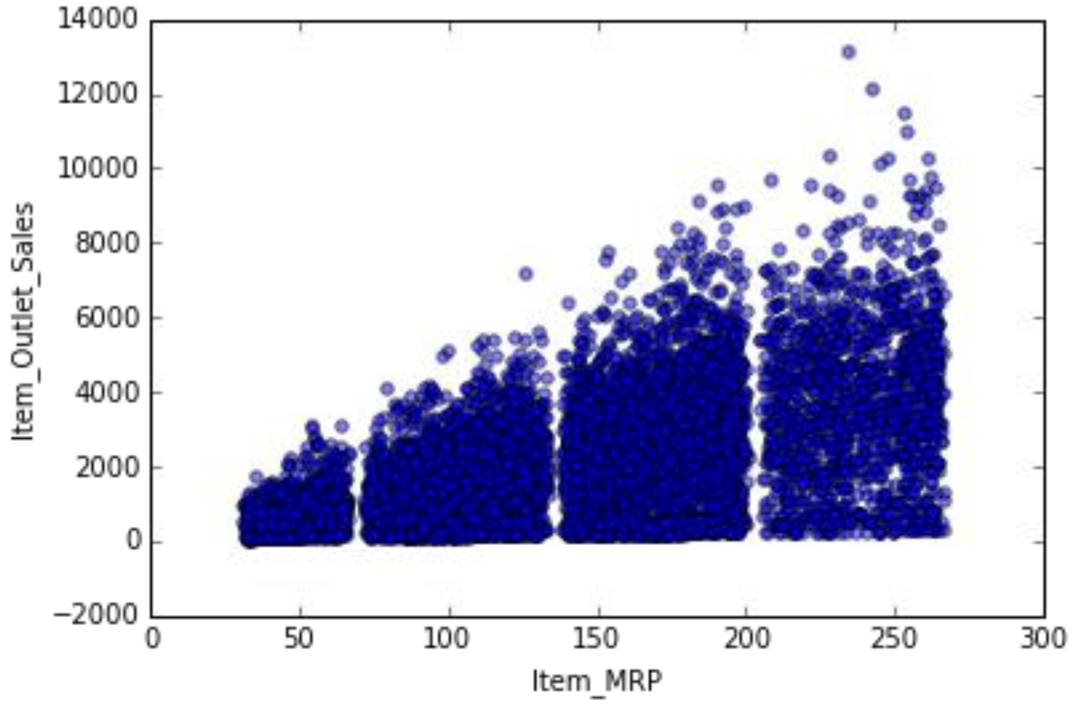
How Establishment year of an outlet year has an impact on sales of that outlet



Correlation between item type and sales for a particular store



Correlation between item prices and item outlet sales



Stage 5A: Imputing NA values for continuous variables by replacing them with Mean

```
Item_Weight
Item_Identifier

Orignal #missing: 2439
Final #missing: 0
```

Stage 5B: Imputing NA values for Categorical variables by replacing them with Mode

```
Mode for each Outlet_Type:
Outlet_Type
Grocery Store            ([Small], [880.0])
Supermarket Type1      ([Small], [3100.0])
Supermarket Type2     ([Medium], [1546.0])
Supermarket Type3     ([Medium], [1559.0])
Name: Outlet_Size, dtype: object

Orignal #missing: 4016
0
```

Stage 6: Feature Engineering

I modified Item_Visibility by considering as NA value.

```
Number of 0 values initially: 879
Number of 0 values after modification: 0
```

Then I created a broad category of Type of Item.

```
Food               10201
Non-Consumable      2686
Drinks              1317
Name: Item_Type_Combined, dtype: int64
```

Changed the categories of low fat and correcting the typos and differences in representation in categories of Item_Fat_Content variable

```
Original Categories:
Low Fat    8485
Regular    4824
LF          522
reg         195
low fat     178
Name: Item_Fat_Content, dtype: int64

Modified Categories:
Low Fat    9185
Regular    5019
Name: Item_Fat_Content, dtype: int64
```

Then marked non-consumables as separate category in low_fat as "Non-Edible

```
Low Fat       6499
Regular       5019
Non-Edible    2686
Name: Item_Fat_Content, dtype: int64
```

I created a new column depicting the years of operation of a store.

```
count    14204.000000
mean        15.169319
std          8.371664
min          4.000000
25%          9.000000
50%         14.000000
75%         26.000000
max         28.000000
Name: Outlet_Years, dtype: float64
```

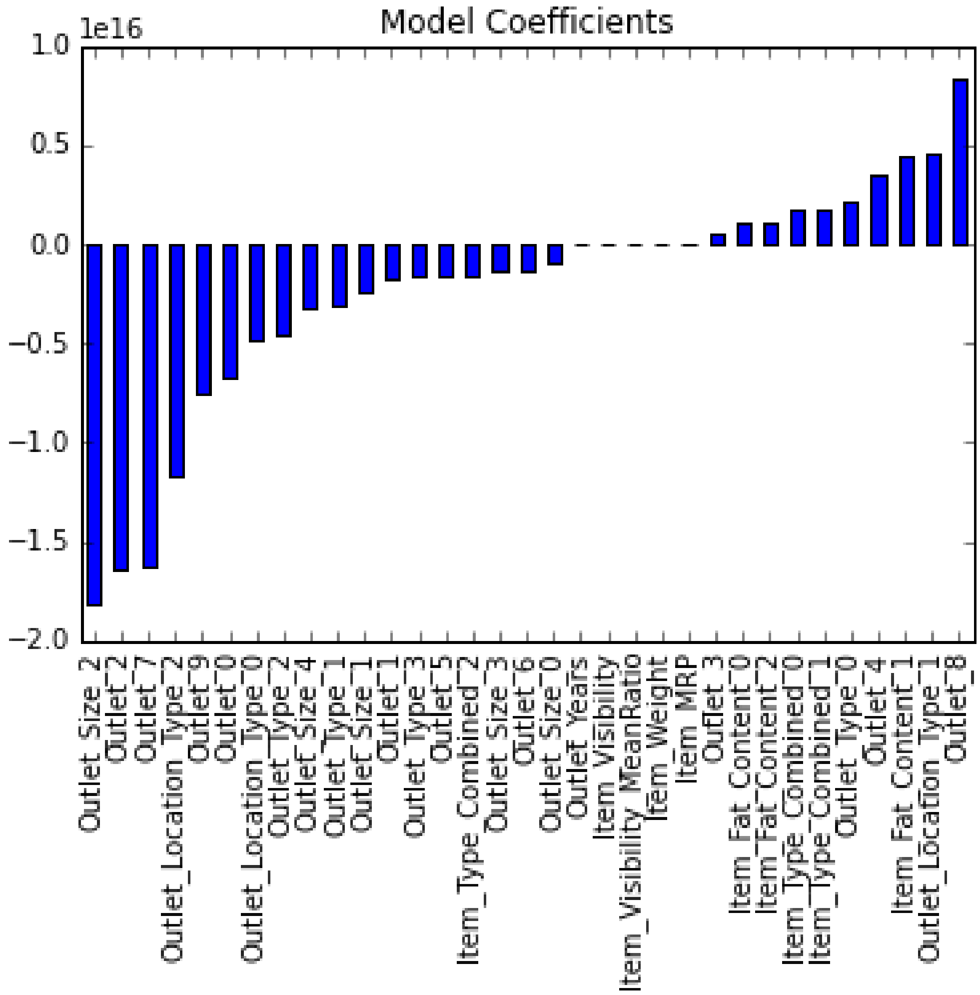
This shows stores which are 4-28 years old.

Exported this clean data into test.csv and train.csv

Stage 7: Model Building

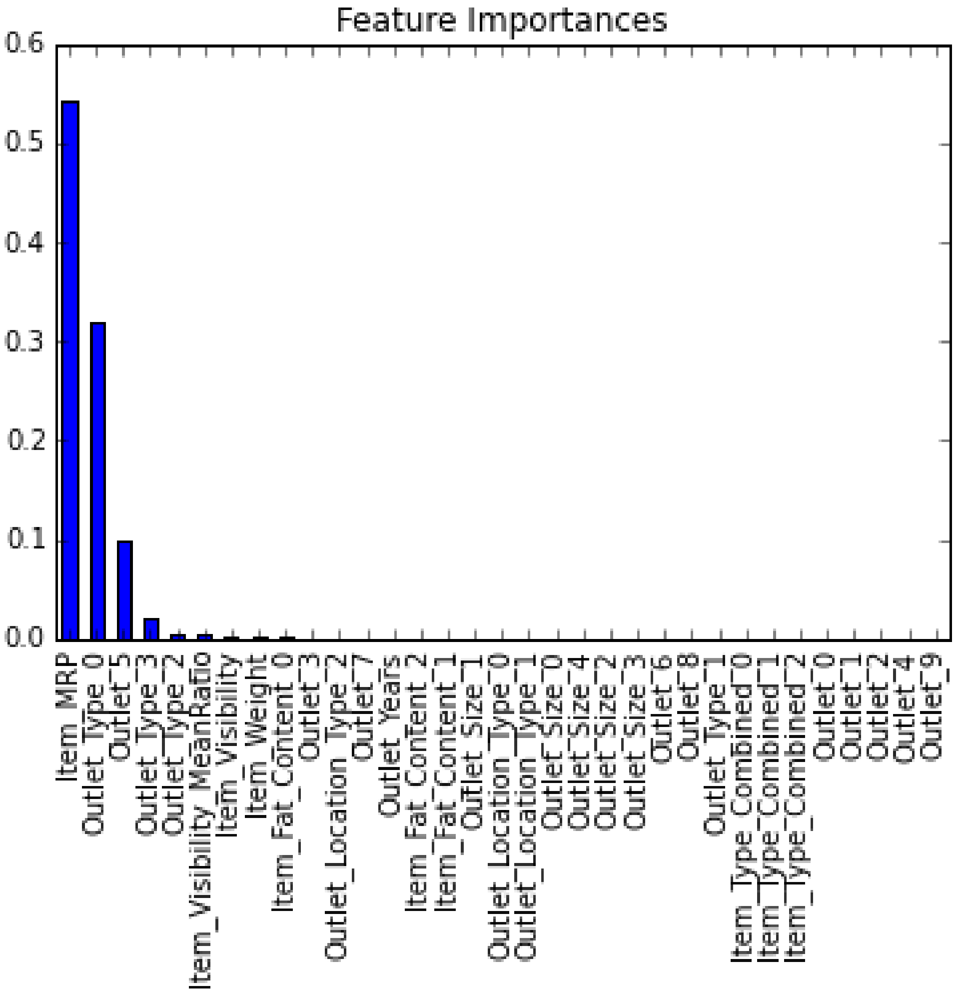
1. **Baseline model:**
   Baseline model is the one which requires no predictive model and it's like an informed guess. I predicted the sales as the overall average sales.

2. **Linear Regression Model:**



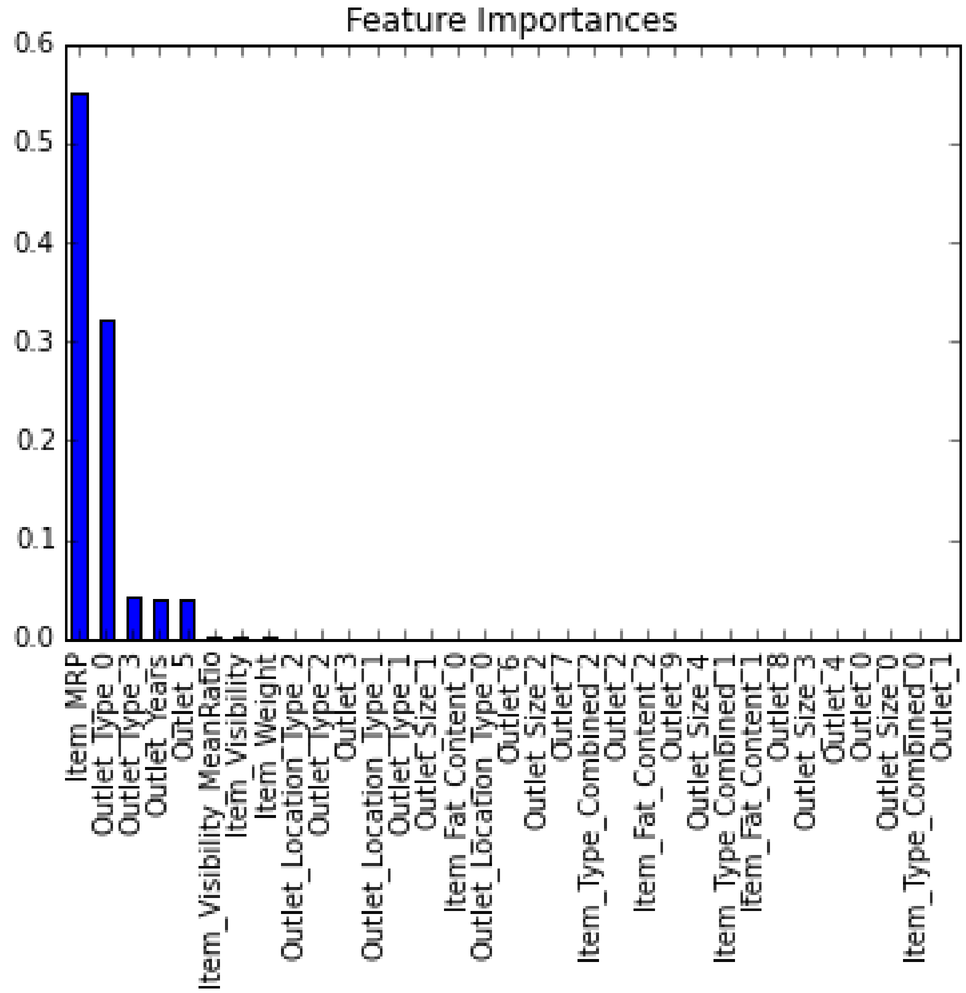

```
Model Report
RMSE : 1127
CV Score : Mean - 1129 | Std - 43.43 | Min - 1075 | Max - 1212
```

3. Decision Tree Model:




```
Model Report
RMSE : 1058
CV Score : Mean - 1093 | Std - 42.18 | Min - 1023 | Max - 1174
```

4. **Random Forest Model**:




```
Model Report
RMSE : 1068
CV Score : Mean - 1082 | Std - 43.05 | Min - 1021 | Max – 1160
```

Conclusion:

We have got RMSE: 1068 with Random forest model with max_depth of 6 and 400 trees. Increasing the number of trees makes the model robust.
So this is the best Model I have found.
The output for sale is saved in CSV file and it is there in the submission.