Project
on
Big Market Sales Prediction

Prepared by :- Jeevan Revaneppa Hirethanad

Abstract:-

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 stores have been defined. The aim of this data science project 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.
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 stores have been defined. The aim of this data science project 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.

Lifecycle of data science projects:-

- Data Analysis
- Feature Engineering
- Feature Selection
- Model Building

Project details:-

1. **Data Analysis:-**
   - In this process we get data from the user and will anaylise the data and create the outline of the the problem.
   - We will understand what all the variables/features are given in the data .
   - We will try try to understand what is missing in the data.

Predicting sales of each product at a particular store.

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

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

Out[1]:

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

8523 rows × 12 columns

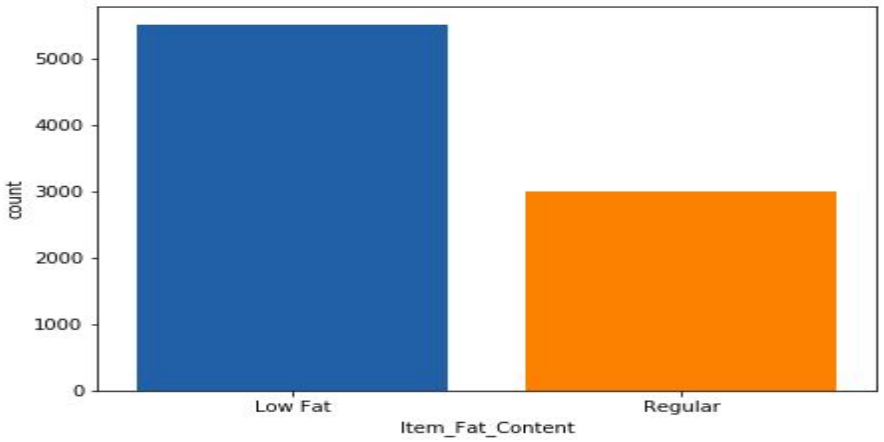
```
In [3]: df.isna().sum()

Out[3]: 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
```

2. <u>Univariate Analysis:-</u>

- In this we try to understand how each variable/feature has the influence on the target variables/feature.
- We visualize the influence in the from of graphs and plots.
- We get an approximation,whether the variables/feature really impact the output/target variables/feature.

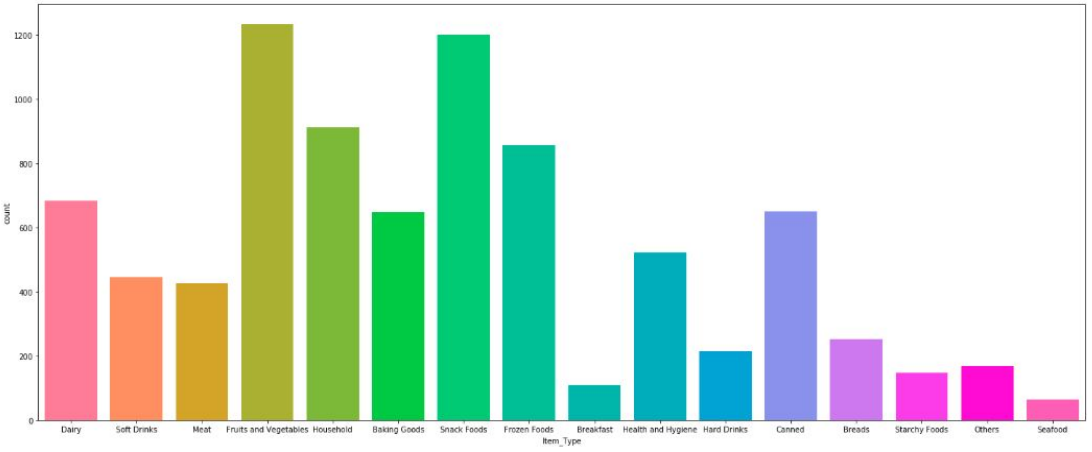
In [15]:
```
plt.figure(figsize=(7,5))
sns.countplot(x="Item_Fat_Content",data=df)
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x1c16db70d48>



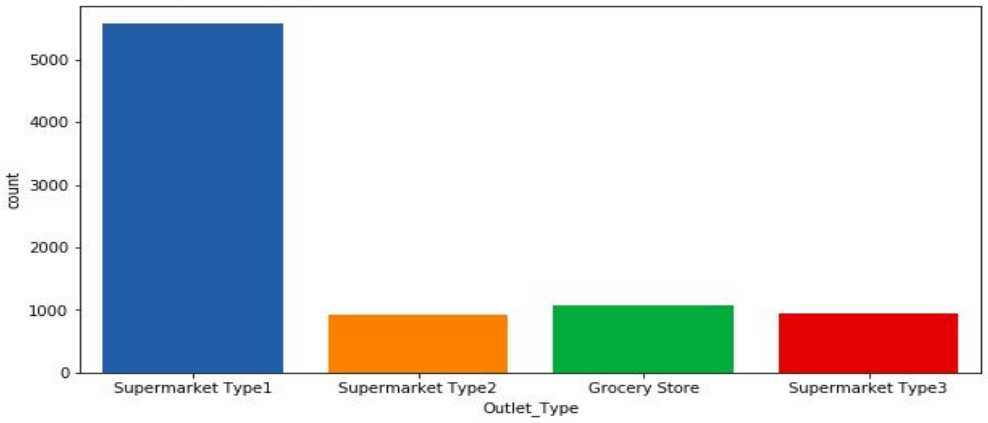
In [18]:
```
plt.figure(figsize=(25,10))
sns.countplot(x="Item_Type",data=df)
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x1c16e34d348>



In [22]:
```
plt.figure(figsize=(10,5))

sns.countplot(x="Outlet_Type",data=df)
print(df['Outlet_Type'].value_counts())
```

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

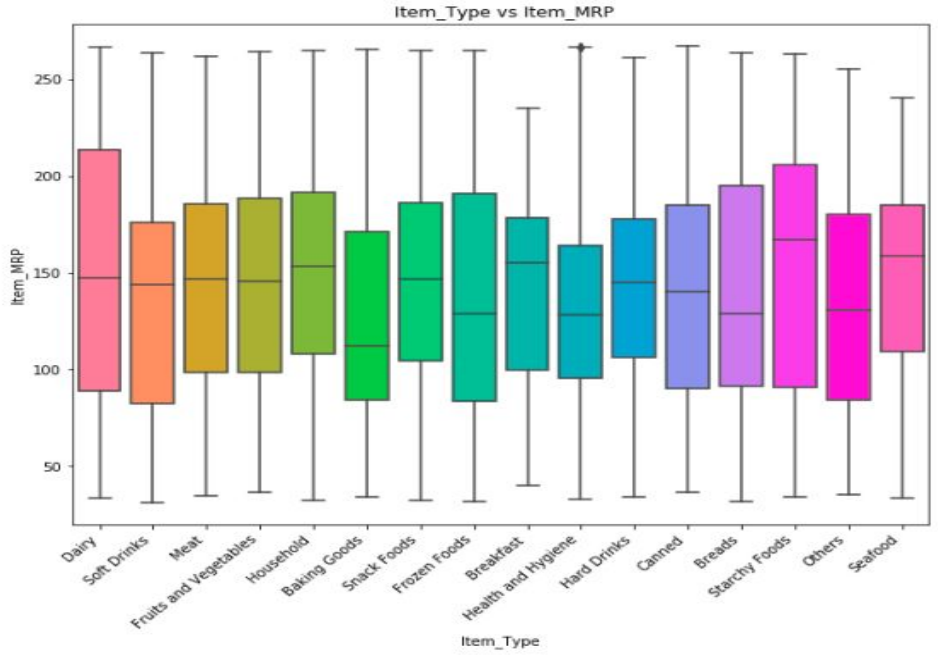


3. Bivariate Analysis:-

- In this we try to understand the how the variable/features are related to each other.
- We visualize the relation through plots.

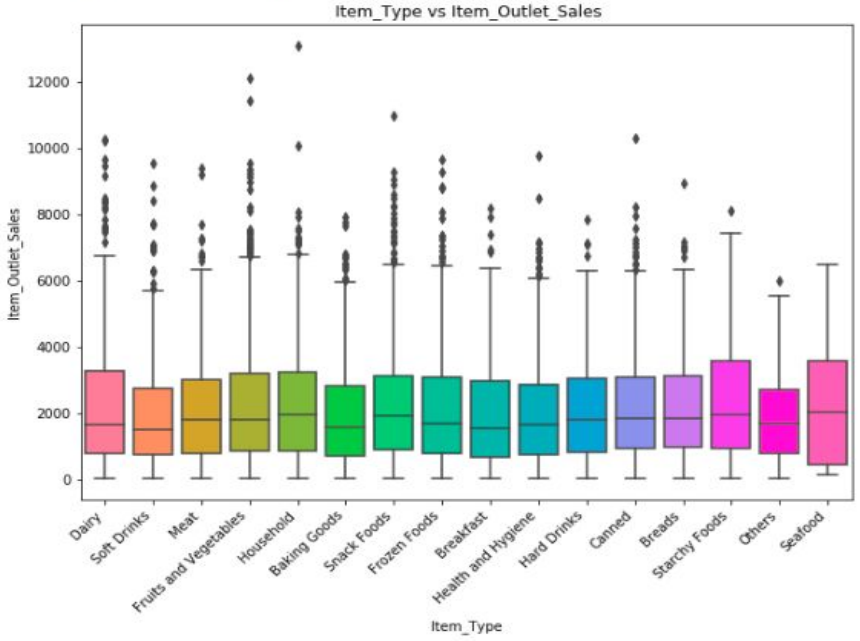
```
In [23]: plt.figure(figsize=(10,8))
         g2=sns.boxplot(x="Item_Type", y="Item_MRP",data=df)
         g2.set_xticklabels(g2.get_xticklabels(),rotation = 45,horizontalalignment='right')

         plt.title(" Item_Type vs Item_MRP")
```

```
Out[23]: Text(0.5, 1.0, ' Item_Type vs Item_MRP')
```




```
In [24]: plt.figure(figsize=(10,7))
         graph1=sns.boxplot(x="Item_Type",y="Item_Outlet_Sales",data=df)
         graph1.set_xticklabels(graph1.get_xticklabels(), rotation = 45,horizontalalignment='right')
         plt.title(" Item_Type vs Item_Outlet_Sales")
```

```
Out[24]: Text(0.5, 1.0, ' Item_Type vs Item_Outlet_Sales')
```



4. Missing Value Treatment:-

- In this we try to fix the missing values either by replacing them with mean values or with specific numerical data like 0 or any or etc...

```
In [7]: 1 df['Item_Weight']=df['Item_Weight'].fillna(df['Item_Weight'].mean())
```

5. Encoding Categorical Variables:-

- In this we try to encode with categorical values with dummy variables or numerical variable so that it will be easy to for mathematical calculations in finding out target variable/feature.

```
In [26]: Item_Fat_Content=pd.get_dummies(df["Item_Fat_Content"])
         Item_Fat_Content
```

Out[26]:

	Low Fat	Regular
0	1	0
1	0	1
2	1	0
3	0	1
4	1	0
...	...	...
8518	1	0
8519	0	1
8520	1	0
8521	0	1
8522	1	0

8523 rows × 2 columns

```
In [28]: Outlet_Type=pd.get_dummies(df["Outlet_Type"])
         Outlet_Type
```

Out[28]:

	Grocery Store	Supermarket Type1	Supermarket Type2	Supermarket Type3
0	0	1	0	0
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	1	0	0
...	...	...	...	...
8518	0	1	0	0
8519	0	1	0	0
8520	0	1	0	0
8521	0	0	1	0
8522	0	1	0	0

8523 rows × 4 columns

6. PreProcessing of Data:-

- This is the final step before modeling.
- In this we concat all the encoded categorical variables with original data.
- We leave out all unrequired columns and non infulence variable/feature.
- We prepare the final cleaned data for modeling.
- We spilt the data into influence/input variable and target variable.
- We will normalize the data so that it will be easy for calculations.
- We spilt the data into testing and training data.

```
In [30]: df1=pd.concat([df,Outlet_Location_Type,Item_Fat_Content,Outlet_Type,Item_Type],axis=1)
         df1
```

Out[30]:

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

8523 rows × 37 columns

In [32]:
```
data=df1[['Item_Weight','Item_Visibility','Item_MRP','Item_Outlet_Sales','Tier 1', 'Tier 2', 'Tier 3',
       'Low Fat', 'Regular', 'Grocery Store', 'Supermarket Type1',
       'Supermarket Type2', 'Supermarket Type3', 'Baking Goods', 'Breads',
       'Breakfast', 'Canned', 'Dairy', 'Frozen Foods', 'Fruits and Vegetables',
       'Hard Drinks', 'Health and Hygiene', 'Household', 'Meat', 'Others',
       'Seafood', 'Snack Foods', 'Soft Drinks', 'Starchy Foods']]
data
```

Out[32]:

	Item_Weight	Item_Visibility	Item_MRP	Item_Outlet_Sales	Tier 1	Tier 2	Tier 3	Low Fat	Regular	Grocery Store	...	Fruits and Vegetables	Hard Drinks	Health and Hygiene	Household	Me
0	9.300	0.016047	249.8092	3735.1380	1	0	0	1	0	0	...	0	0	0	0	0
1	5.920	0.019278	48.2692	443.4228	0	0	1	0	1	0	...	0	0	0	0	0
2	17.500	0.016760	141.6180	2097.2700	1	0	0	1	0	0	...	0	0	0	0	1
3	19.200	0.000000	182.0950	732.3800	0	0	1	0	1	1	...	1	0	0	0	0
4	8.930	0.000000	53.8614	994.7052	0	0	1	1	0	0	...	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8518	6.865	0.056783	214.5218	2778.3834	0	0	1	1	0	0	...	0	0	0	0	0
8519	8.380	0.046982	108.1570	549.2850	0	1	0	0	1	0	...	0	0	0	0	0
8520	10.600	0.035186	85.1224	1193.1136	0	1	0	1	0	0	...	0	0	1	0	0
8521	7.210	0.145221	103.1332	1845.5976	0	0	1	0	1	0	...	0	0	0	0	0
8522	14.800	0.044878	75.4670	765.6700	1	0	0	1	0	0	...	0	0	0	0	0

8523 rows × 29 columns

In [34]:
```
X = data.drop(['Item_Outlet_Sales'], axis = 1)

y=data['Item_Outlet_Sales']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 43)
```

7. Modeling:-

- In this we use various machine learning models to predict the target variables from input variables.
- We use various model like Linear Regression,Regularized Linear Regression, RandomForest, XGBoost
- We cross-validation technique so that the model fits the data better.
- We use various method to find accuracy of the model but root mean squared error is most popular and best method.
- We observe the accuracy of all the models and come to conclusion that which model best fits the given problem.

In [37]:
```
from sklearn.linear_model import *
from sklearn.metrics import accuracy_score,roc_auc_score,mean_squared_error
import numpy as np


classifier = LinearRegression()
classifier.fit(X_train, y_train)


y_pred = classifier.predict(X_test)


from sklearn import metrics

print('Root mean squared error of the Linear Regression Model is :-',np.sqrt(metrics.mean_squared_error(y_test,y_pred)))

list.append(np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

```
Root mean squared error of the Linear Regression Model is :- 1108.9448147490032
```

In [46]:
```
lin_rids=Lasso()

XX=X.values
yy=y.values
lin_rids.fit(XX,yy)

yyy=lin_rids.predict(XX)


print('Root mean squared error of the Lasso Regression Model is :-',np.sqrt(mean_squared_error(yy,yyy)))

list.append(np.sqrt(mean_squared_error(yy,yyy)))
```

```
Root mean squared error of the Lasso Regression Model is :- 1128.7351534160682
```

In [47]:
```
lin_rid=Ridge()

XX=X.values
yy=y.values
lin_rid.fit(XX,yy)

yyy=lin_rid.predict(XX)


print('Root mean squared error of the Ridge Regression Model is :-',np.sqrt(mean_squared_error(yy,yyy)))

list.append(np.sqrt(mean_squared_error(yy,yyy)))
```

```
Root mean squared error of the Ridge Regression Model is :- 1128.5197028520297
```

In [42]:
```
from sklearn.model_selection import cross_val_score

rf=RandomForestRegressor(n_estimators = 100, random_state = 0)
c=11111111111111111111
d=0
for i in range(3,10):
    sc=cross_val_score(rf, X, y, cv=i, scoring='neg_root_mean_squared_error')

    sc=-sc
    print('Root mean squared error of the model for kfolds',i,' is ',sc.mean())
    if sc.mean()<c:
        c=sc.mean()
        d=i
print('\n\n\n\nBest number of kfolds for cross validation is ',d,'\n\n\n\n')
list.append(c)
list
```

```
Root mean squared error of the model for kfolds 3  is  1164.0682118085285
Root mean squared error of the model for kfolds 4  is  1166.3254850863382
Root mean squared error of the model for kfolds 5  is  1159.3311878794182
Root mean squared error of the model for kfolds 6  is  1156.414089572154
Root mean squared error of the model for kfolds 7  is  1159.1187606914782
Root mean squared error of the model for kfolds 8  is  1160.1640207000517
Root mean squared error of the model for kfolds 9  is  1161.5155555302867



Best number of kfolds for cross validation is  6
```

Summary:-

I implemented all the steps successfully for predicting output sales of Bigmart.The uninfluence variables/features were left out, and the Categorical Variables were encoded with dummy variables of 0's and 1's.Finally,the cleaned data was provided for modeling.In modeling I implemented Linear Regression,Regularized Linear Regression, RandomForest, XGBoost Models and also the cross validation technique was used.Finally,the least root mean squared error was observed in the linear regression model,which indicates that the linear regression is the best for this model.