IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

Prepared by : Kaushik Prasad Dey, DBI_0015

ABSTRACT:

Machine Learning is a category of algorithms that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build models and employ algorithms that can receive input data and use statistical analysis to predict an output while updating outputs as new data becomes available. These models can be applied in different areas and trained to match the expectations of management so that accurate steps can be taken to achieve the organization's target.

Nowadays shopping malls and Big Marts keep the track of their sales data of each and every individual item for predicting future demand of the customer and update the inventory management as well. These data stores basically contain a large number of customer data and individual item attributes in a data warehouse. Further, anomalies and frequent patterns are detected by mining the data store from the data warehouse.

The resultant data can be used for predicting future sales volume with the help of different machine learning techniques for the retailers like Big Mart. In this paper, we propose a predictive model using **Linear, lasso, Ridge, Decision-tree, Random-forest, XG boost** Regressor technique for predicting the sales of a company like Big Mart and found that the model produces better performance as compared to existing models.

Keywords: Machine Learning, Sales Prediction, Big Mart, Linear Regression, Lasso regression, Ridge Regression, Decision-tree, Random Forest.

Tools Used

INTRODUCTION:

In today's modern world, huge shopping centers such as big malls and marts are recording data related to sales of items or products with their various dependent or independent factors as an important step to be helpful in prediction of future demands and inventory management.

Big Mart is a big supermarket chain, with stores all around the country and its current board set out a challenge to all Data Scientist out there to help them create a model that can predict the sales, per product, for each store to give accurate results.

The dataset built with various dependent and independent variables is a composite form of item attributes, data gathered by means of customer, and also data related to inventory management in a data warehouse.

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

The data is thereafter refined in order to get accurate predictions and gather new as well as interesting results that shed a new light on our knowledge with respect to the task's data.

PROBLEM STATEMENT:

The Data scientist has collected sales data from **the year 2018**, for **1531** products across **8** 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.

Hypothesis Generation :

Make it a practice to do this before solving any ML problem. Ideally, do it before seeing the data or else you might end up with biased hypotheses.

What could affect the target variable (sales)?

1. Time of week - Weekends usually are more busy.
2. Higher sales according to time of the day (morning and late evening).
3. Higher sales during end of the year.
4. Store size and location.
5. Items with more shelf space sell more.

Methodology:

The steps followed in this work, right from the dataset preparation to obtaining results are represented in Fig.1.

PHASES IN MODEL :

1. Dataset :

Name	Type	Subtype	Description	Segment	Expectation
Item Identifier	numeric	Discrete	Unique product ID	product	Low impact
Item_Weight	numeric	Continuous	Weight of product	product	Medium impact
Item_Fat_Content	Categorical	Ordinal	Checks the Concentration of fat in the product	product	Medium impact

BIG MART SALES PREDICTION USING MACHINE LEARNING

Item_Visibility	numeric	Continuous	The % of total display area of all similar products in a store	product	High Impact
Item_Type	Categorical	Nominal	Product Category	Product	High Impact
Item_MRP	numeric	Discrete	Maximum Retail Price for a Product	Product	Medium impact
Outlet_Identifier	numeric	Discrete	Unique Store ID	Store	Low impact
Outlet_Establishment_Year	numeric	Discrete	The year in which store was established	Store	Low impact
Outlet_Size	Categorical	Ordinal	The size of the store (Area Size Category)	Store	High Impact
Outlet_Location_Type	Categorical	Ordinal	In Terms of city Tiers (Size)	Store	High Impact
Outlet_Type	Categorical	Ordinal	Grocery store or a type of supermarket	Store	High Impact
Item_Outlet_Sales	numeric	Discrete	Sales of the product In the Specific outlet	Product	Target

2. Dataset and its Preprocessing :

Big Mart's data scientists collected sales data of their 8 stores situated at different locations with each store having 1531 different products as per 2018 data collection. Using all the observations it is inferred what role certain properties of an item play and how they affect their sales. The dataset looks like below on using head() function on the dataset variable.

```
df.head()
```

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
df.head()
```

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

The data set consists of various data types from integer to float to object as shown in below.

```
df.dtypes
```

```
Item_Identifier              object
Item_Weight                 float64
Item_Fat_Content             object
Item_Visibility             float64
Item_Type                    object
Item_MRP                    float64
Outlet_Identifier            object
Outlet_Establishment_Year     int64
Outlet_Size                  object
Outlet_Location_Type         object
Outlet_Type                  object
Item_Outlet_Sales           float64
dtype: object
```

In the raw data, there can be various types of underlying patterns which also gives an in-depth knowledge about subject of interest and provides insights about the big Mart Problem. But caution should be observed with respect to data as it may contain null values, or redundant values, or various types of ambiguity, which also demands for pre-processing of data. Dataset should therefore be explored as much as possible.

3. computes a summary of statistics :

Various factors important by statistical means like mean, standard deviation, median, count of values and maximum value etc. It shows five point summary of that big mart Raw data.

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
df.describe()
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	3932.000000	4706.000000	4706.000000	4706.000000	4706.000000
mean	12.955654	0.066001	140.743798	1997.864003	2177.914990
std	4.655408	0.051338	62.387107	8.299149	1700.556703
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.895000	0.026950	93.236350	1987.000000	838.908000
50%	12.800000	0.054367	142.614100	1999.000000	1789.670400
75%	17.100000	0.094276	186.083850	2004.000000	3083.985600
max	21.350000	0.328391	266.688400	2009.000000	11445.102000

4. Checking null Values :

Preprocessing of this dataset includes doing analysis on the independent variables like checking for null values in each column and then replacing or filling them with supported appropriate data types, so that analysis and model fitting is not hindered from its way to accuracy.

```
df.isnull().sum()
```

```
Item_Identifier                 0
Item_Weight                   774
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                  1358
Outlet_Location_Type            0
Outlet_Type                     0
Item_Outlet_Sales               0
dtype: int64
```

5. Data Exploration :

In this phase useful information about the data has been extracted from the dataset. That is trying to identify the information from hypotheses vs available data which shows that the attributes Outlet size and Item weight face the problem of missing values, also the minimum value of Item Visibility is zero which is not actually practically possible. Establishment year of Outlet varies from 1985 to 2009. These values may not be appropriate in this form. So, we need to convert them into how old a particular outlet is. There are 1531 unique products, as well as 10 unique outlets, present in the dataset. The attribute Item type contains 16 unique values. Where as two types of Item Fat Content are there but some of them are misspelled as regular instead of 'Regular' and low fat, LF instead of Low Fat. It was found that the response variable i.e. Item Outlet Sales was positively skewed. So, to remove the skewness of response variable a log operation was performed on Item Outlet Sales

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

6. Exploratory data analysis (EDA) :

We have made our hypotheses and now we are ready to do some data exploration and come up with some inference. The goal for the EDA is to get some insight and if any irregularities are found we will correct that in the next section, **Data Pre-Processing.** This shows that our Item Identifier has some duplicate values. since a product can exist in more than one store it is expected for this repetition.

```
idsTotal = df.shape[0]
idsDupli = df[df['Item_Identifier'].duplicated()]
print(f'There are {len(idsDupli)} duplicate IDs for {idsTotal} total entries')
```

```
There are 6964 duplicate IDs for 8523 total entries
```

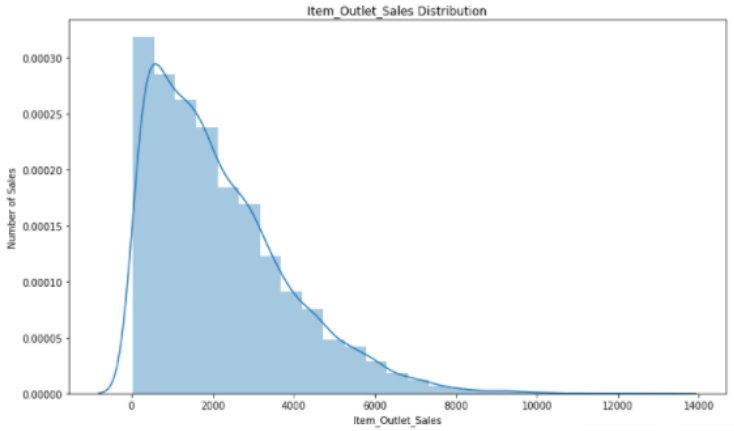
7. Univariate Analysis :

In Univariate analysis we will explore each variable in a dataset.

a. Distribution of the target variable: Item_Outlet_Sales

```
plt.figure(figsize=(12,7))
sns.distplot(df.Item_Outlet_Sales, bins = 25)
plt.xlabel("Item_Outlet_Sales")
plt.ylabel("Number of Sales")
plt.title("Item_Outlet_Sales Distribution")
```

```
Text(0.5, 1.0, 'Item_Outlet_Sales Distribution')
```




```
print ("Skew is:", df.Item_Outlet_Sales.skew())
print("Kurtosis: %f" % df.Item_Outlet_Sales.kurt())
```

```
Skew is: 1.1775306028542796
Kurtosis: 1.615877
```

We can see that our target variable is skewed towards the right. Therefore, we must normalize it.

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

b. Numerical Predictors

Now we will consider our dependent variables. First of all, we will check for the numerical variables in our dataset:

```
num_features = df.select_dtypes(include=[np.number])
num_features.dtypes
```

```
Item_Weight                  float64
Item_Visibility              float64
Item_MRP                     float64
Outlet_Establishment_Year      int64
Item_Outlet_Sales            float64
dtype: object
```

We can see that out of 12 we have only 5 numeric variables.

```
corr=num_features.corr()
corr
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
Item_Weight	1.000000	-0.014048	0.027141	-0.011588	0.014123
Item_Visibility	-0.014048	1.000000	-0.001315	-0.074834	-0.128625
Item_MRP	0.027141	-0.001315	1.000000	0.005020	0.567574
Outlet_Establishment_Year	-0.011588	-0.074834	0.005020	1.000000	-0.049135
Item_Outlet_Sales	0.014123	-0.128625	0.567574	-0.049135	1.000000

```
corr['Item_Outlet_Sales'].sort_values(ascending=False)
```

```
Item_Outlet_Sales            1.000000
Item_MRP                     0.567574
Item_Weight                  0.014123
Outlet_Establishment_Year   -0.049135
Item_Visibility             -0.128625
Name: Item_Outlet_Sales, dtype: float64
```

From the above result, we can see that Item_MRP have the most positive correlation and the Item_Visibility have the lowest correlation with our target variable. It is totally different from our initial hypotheses, this variables was expected to have high impact in the sales increase. Nevertheless, since this is not an expected behaviour and we should investigate.

c. Categorical Predictors

IIM NAGPUR

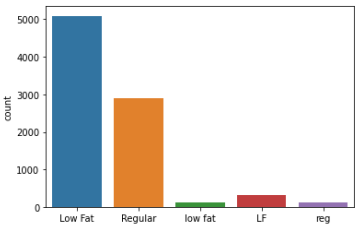
BIG MART SALES PREDICTION USING MACHINE LEARNING

Now let's do some analysis on categorical variable and look at the variables that contain some insight on the hypotheses that we previously made.

C 1. **Distribution of the variable Item_Fat_Content**

```
sns.countplot(df.Item_Fat_Content)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5fca0a4f90>
```

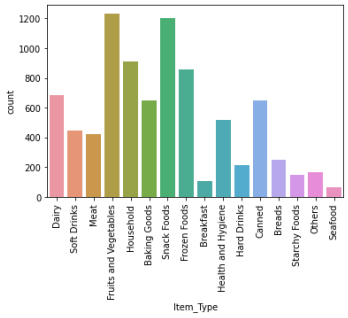


For Item_Fat_Content there are two possible type "Low Fat" or "Regular". However, in our data it is written in different manner. We will Correct this.

C2 **Distribution of the variable Item_Type**

```
sns.countplot(df.Item_Type)
plt.xticks(rotation=90)
```

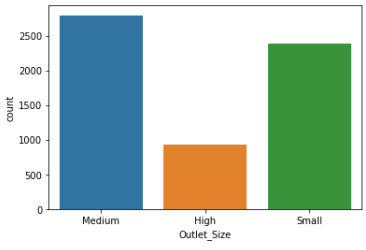
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15]),
 <a list of 16 Text major ticklabel objects>)
```



forItem_Type we have 16 different types of unique values and it is high number for categorical variable. Therefore we must try to reduce it.

```
sns.countplot(df.Outlet_Size)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5fc9e55350>
```



IIM NAGPUR

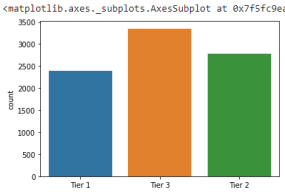
BIG MART SALES PREDICTION USING MACHINE LEARNING

There seems to be less number of stores with size equals to “High”. It will be very interesting to see how this variable relates to our target.

C3. Distribution of the variable Outlet_Location_Type

```
sns.countplot(df.Outlet_Location_Type)
```

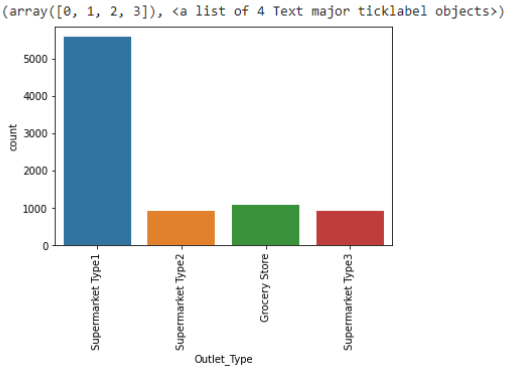
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5fc9ea38d0>
```



From the above graph we can see that Bigmart is a brand of medium and small size city compare to densely populated area.

```
sns.countplot(df.Outlet_Type)
plt.xticks(rotation=90)
```

```
(array([0, 1, 2, 3]), <a list of 4 Text major ticklabel objects>)
```



Distribution of the variable Outlet_Type

There seems like Supermarket Type2 , Grocery Store and Supermarket Type3 all have low numbers of stores, we can create a single category with all of three, but before doing this we must see their impact on target variable.

8. Bivariate Analysis :

Now it time to see the relationship between our target variable and predictors.

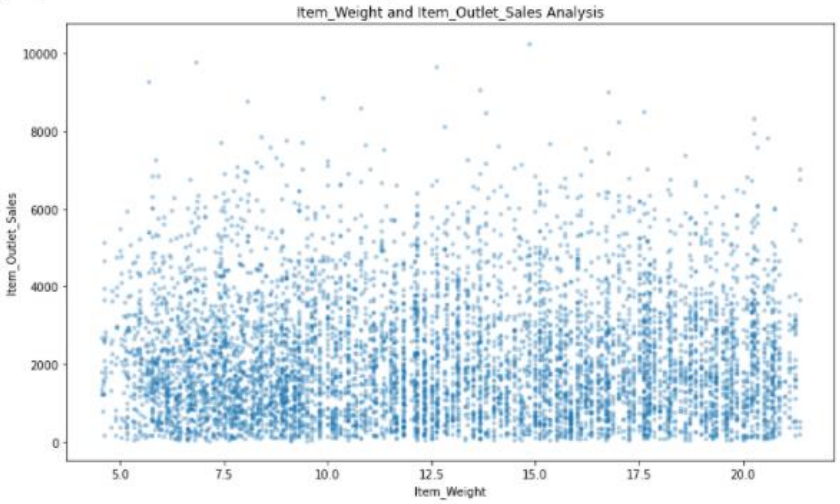
Numerical Variables

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
plt.figure(figsize=(12,7))
plt.xlabel("Item_Weight")
plt.ylabel("Item_Outlet_Sales")
plt.title("Item_Weight and Item_Outlet_Sales Analysis")
plt.plot(df.Item_Weight, df["Item_Outlet_Sales"],'.', alpha = 0.3)
```

```
[<matplotlib.lines.Line2D at 0x7f5fc9c32850>]
```



We saw previously that Item_Weight had a low correlation with our target variable. This plot shows there relation.

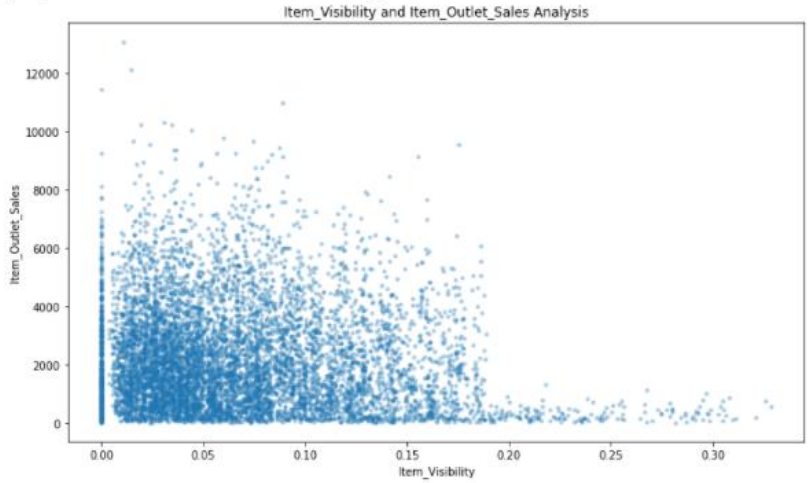
8 a. Item_Visibility and Item_Outlet_Sales analysis

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

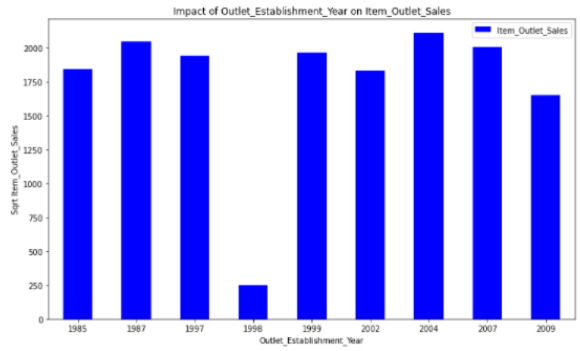
```
plt.figure(figsize=(12,7))
plt.xlabel("Item_Visibility")
plt.ylabel("Item_Outlet_Sales")
plt.title("Item_Visibility and Item_Outlet_Sales Analysis")
plt.plot(df.Item_Visibility, df["Item_Outlet_Sales"],'.', alpha = 0.3)
```

```
[<matplotlib.lines.Line2D at 0x7f5fc9c30590>]
```



8 b. Outlet_Establishment_Year and Item_Outlet_Sales analysis

```
Outlet_Establishment_Year_pivot = df.pivot_table(index='Outlet_Establishment_Year', values="Item_Outlet_Sales", aggfunc=np.median)
Outlet_Establishment_Year_pivot.plot(kind='bar', color='blue',figsize=(12,7))
plt.xlabel("Outlet_Establishment_Year")
plt.ylabel("Sqrt Item_Outlet_Sales")
plt.title("Impact of Outlet_Establishment_Year on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```



There seems to be no appreciable meaning between the year of store establishment and the sales for the items.

8 c. Categorical Variables

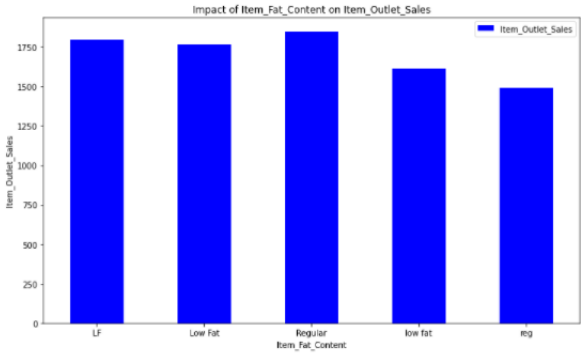
Impact of Item_Fat_Content onItem_Outlet_Sales

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
Item_Fat_Content_pivot = df.pivot_table(index='Item_Fat_Content', values="Item_Outlet_Sales", aggfunc=np.median)
Item_Fat_Content_pivot.plot(kind='bar', color='blue',figsize=(12,7))

plt.xlabel("Item_Fat_Content")
plt.ylabel("Item_Outlet_Sales")
plt.title("Impact of Item_Fat_Content on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```



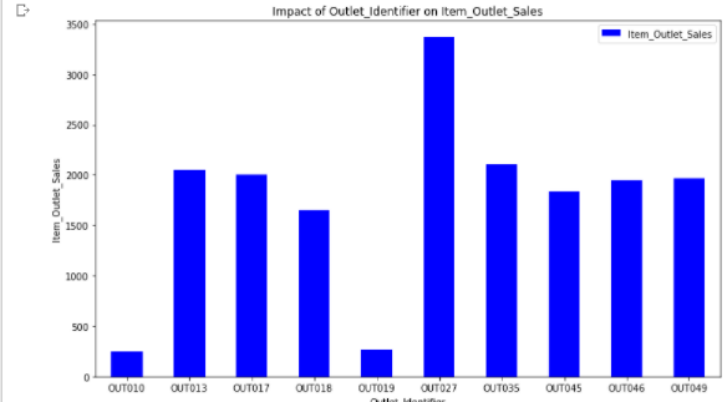
Low Fat products seem to higher sales than the Regular products.

8 d. Impact of Outlet_Identifier on Item_Outlet_Sales

▾ Impact of Outlet_Identifier on Item_Outlet_Sales

```
Outlet_Identifier_pivot = df.pivot_table(index='Outlet_Identifier', values="Item_Outlet_Sales", aggfunc=np.median)
Outlet_Identifier_pivot.plot(kind='bar', color='blue',figsize=(12,7))

plt.xlabel("Outlet_Identifier")
plt.ylabel("Item_Outlet_Sales")
plt.title("Impact of Outlet_Identifier on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```



Out of 10- There are 2 Groceries store, 6 Supermarket Type1, 1Supermarket Type2, and 1 Supermarket Type3. You can see from the below pivot table.

```
df.pivot_table(values='Outlet_Type',
               columns='Outlet_Identifier',
               aggfunc=lambda x:x.mode())
```

Outlet_Identifier	OUT010	OUT013	OUT017	OUT018	OUT019	OUT027	OUT035	OUT045	OUT046
Outlet_Type	Grocery Store	Supermarket Type1	Supermarket Type1	Supermarket Type2	Grocery Store	Supermarket Type3	Supermarket Type1	Supermarket Type1	Supermarket Type1

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
df.pivot_table(values='Outlet_Type',
                   columns='Outlet_Size',
                   aggfunc=lambda x:x.mode())
```

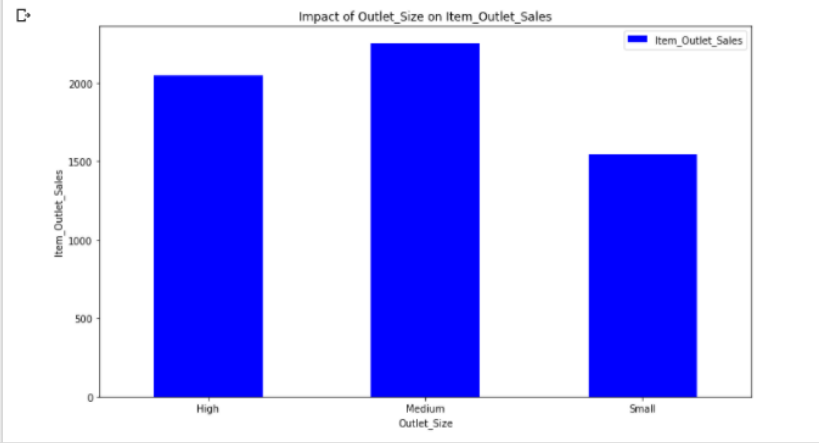
Outlet_Size	High	Medium	Small
Outlet_Type	Supermarket Type1	Supermarket Type3	Supermarket Type1

Most of the stores are of Supermarket Type1 of size High and they do not have best results. whereas Supermarket Type3 (OUT035) is a Medium size store and have best results.

8.e Impact of Outlet_Size on Item_Outlet_Sales

```
Outlet_Size_pivot = df.pivot_table(index='Outlet_Size', values="Item_Outlet_Sales", aggfunc=np.median)
Outlet_Size_pivot.plot(kind='bar', color='blue',figsize=(12,7))

plt.xlabel("Outlet_Size")
plt.ylabel("Item_Outlet_Sales")
plt.title("Impact of Outlet_Size on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```



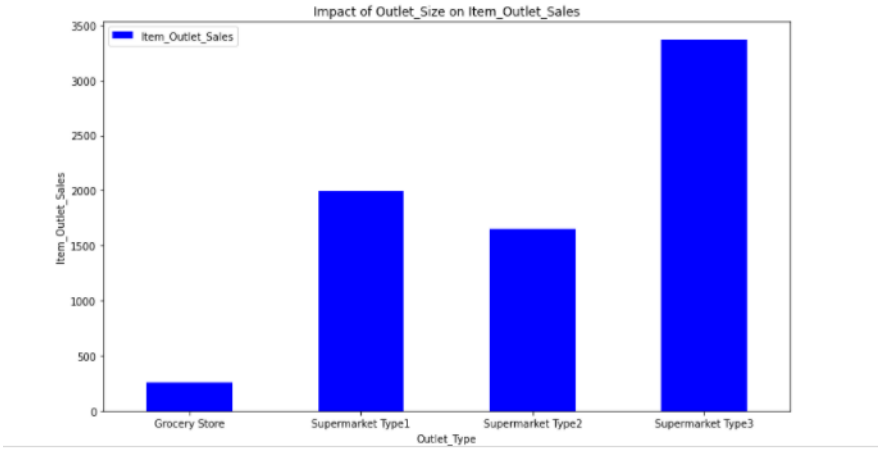
8.f Impact of Outlet_Type on Item_Outlet_Sales

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
Outlet_Type_pivot = df.pivot_table(index='Outlet_Type', values="Item_Outlet_Sales", aggfunc=np.median)
Outlet_Type_pivot.plot(kind='bar', color='blue',figsize=(12,7))

plt.xlabel("Outlet_Type")
plt.ylabel("Item_Outlet_Sales")
plt.title("Impact of Outlet_Size on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```

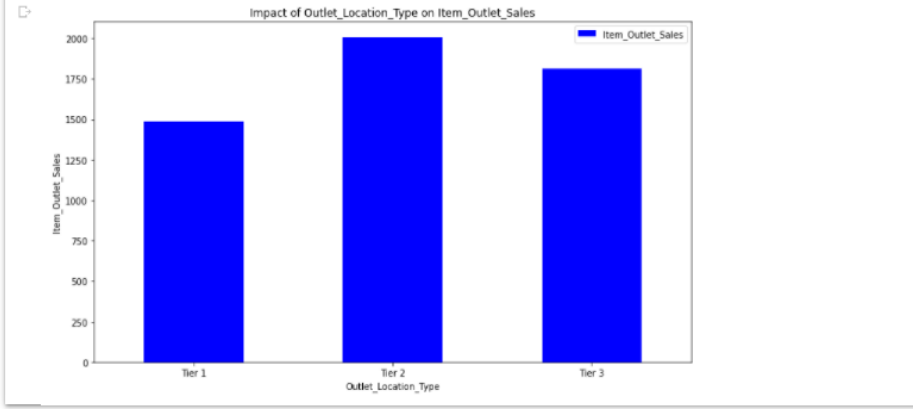


It could be a good idea to create a new feature that shows the sales ratio according to the store size.

Impact of Outlet_Location_Type on Item_Outlet_Sales

```
Outlet_Location_Type_pivot = df.pivot_table(index='Outlet_Location_Type', values="Item_Outlet_Sales", aggfunc=np.median)
Outlet_Location_Type_pivot.plot(kind='bar', color='blue',figsize=(12,7))

plt.xlabel("Outlet_Location_Type")
plt.ylabel("Item_Outlet_Sales")
plt.title("Impact of Outlet_Location_Type on Item_Outlet_Sales")
plt.xticks(rotation=0)
plt.show()
```



This shows that our hypotheses was totaly different from the result that we got from the above plot. Tier 2 cities have the higher sales than the Tier 1 and Tier 2.

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
df.pivot_table(values='Outlet_Location_Type',
               columns='Outlet_Type',
               aggfunc=lambda x:x.mode())
```

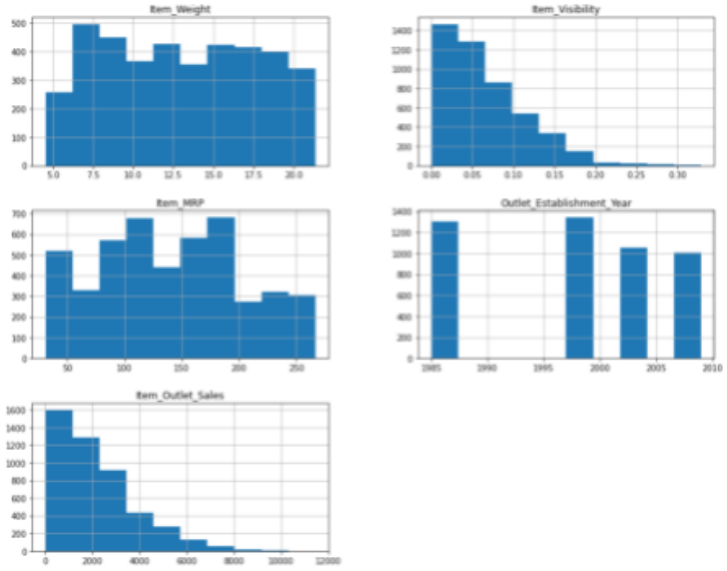
Outlet_Type	Grocery Store	Supermarket Type1	Supermarket Type2	Supermarket Type3
Outlet_Location_Type	Tier 3	Tier 2	Tier 3	Tier 3

9. Data Cleaning:

It was observed from the previous section that the attributes Outlet Size and Item Weight has null values. In our work in case of Outlet Size missing value we replace it by the mode of that attribute and for the Item Weight missing values we replace by mean of that particular attribute. The missing attributes are numerical where the replacement by mean and mode diminishes the correlation among imputed attributes. For our model we are assuming that there is no relationship between the measured attribute and imputed attribute

```
[7] df.hist(figsize=(15,12))

array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7d5c510>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7d225d0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7d2d850>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7c37210>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7c6d810>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f30a7c22e10>]],
      dtype=object)
```



10. Applying Regression Models:

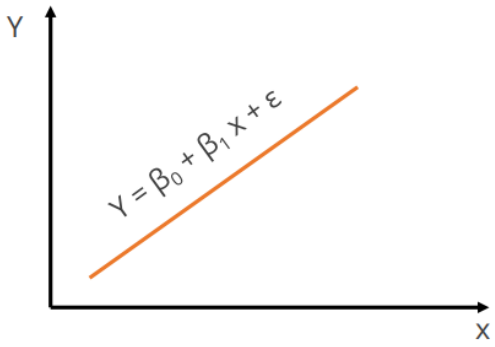
Scikit-Learn can be used to track machine-learning system. Algorithms employed for predicting sales for this dataset are discussed as follows:

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

A. Linear Regression Algorithm

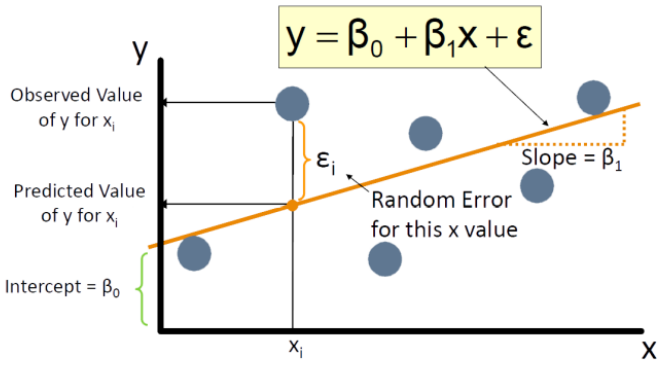
Regression can be termed as a parametric technique which is used to predict a continuous or dependent variable on basis of a provided set of independent variables. This technique is said to be parametric as different assumptions are made on basis of data set.



β1 represents the slope.

- β1 represents the estimated change in the average value of y as a result of a one unit change in x.
- β0 represents the estimated average value of y when the value of x is zero.

The values are labeled as shown



Linear regression Model we have applied in our big Mart dataset with following outputs.

```
[26] X = final_train.drop('Item_Outlet_Sales', axis=1)
     y = final_train['Item_Outlet_Sales']
```

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
[27] X.shape, y.shape
```

```
((8523, 20), (8523,))
```

```
dc = dict()
for i in list(final_train.columns):
    dc[i] = i.replace(' ','_')
```

```
dc
```

```
{'Item_Fat_Content_LF': 'Item_Fat_Content_LF',
 'Item_Fat_Content_Low Fat': 'Item_Fat_Content_Low_Fat',
 'Item_Fat_Content_Regular': 'Item_Fat_Content_Regular',
 'Item_Fat_Content_low fat': 'Item_Fat_Content_low_fat',
 'Item_Fat_Content_reg': 'Item_Fat_Content_reg',
 'Item_MRP': 'Item_MRP',
 'Item_Outlet_Sales': 'Item_Outlet_Sales',
 'Item_Visibility_bins_High Viz': 'Item_Visibility_bins_High_Viz',
 'Item_Visibility_bins_Low Viz': 'Item_Visibility_bins_Low_Viz',
 'Item_Visibility_bins_Viz': 'Item_Visibility_bins_Viz',
 'Item_Weight': 'Item_Weight',
 'Outlet_Location_Type_Tier 1': 'Outlet_Location_Type_Tier_1',
 'Outlet_Location_Type_Tier 2': 'Outlet_Location_Type_Tier_2',
 'Outlet_Location_Type_Tier 3': 'Outlet_Location_Type_Tier_3',
 'Outlet_Size_High': 'Outlet_Size_High',
 'Outlet_Size_Medium': 'Outlet_Size_Medium',
 'Outlet_Size_Small': 'Outlet_Size_Small'
```

```
[31] train_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 21 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Item_Weight                   8523 non-null   float64
 1   Item_MRP                      8523 non-null   float64
 2   Item_Outlet_Sales             8523 non-null   float64
 3   Item_Fat_Content_LF           8523 non-null   uint8
 4   Item_Fat_Content_Low_Fat      8523 non-null   uint8
 5   Item_Fat_Content_Regular      8523 non-null   uint8
 6   Item_Fat_Content_low_fat      8523 non-null   uint8
 7   Item_Fat_Content_reg          8523 non-null   uint8
 8   Outlet_Size_High              8523 non-null   uint8
 9   Outlet_Size_Medium            8523 non-null   uint8
 10  Outlet_Size_Small             8523 non-null   uint8
 11  Outlet_Location_Type_Tier_1   8523 non-null   uint8
 12  Outlet_Location_Type_Tier_2   8523 non-null   uint8
 13  Outlet_Location_Type_Tier_3   8523 non-null   uint8
 14  Outlet_Type_Grocery_Store     8523 non-null   uint8
 15  Outlet_Type_Supermarket_Type1 8523 non-null   uint8
 16  Outlet_Type_Supermarket_Type2 8523 non-null   uint8
```

```
[32] 'Item_Outlet_Sales ~ ' + ' + '.join(list(train_new.drop('Item_Outlet_Sales', axis=1).columns))
```

```
'Item_Outlet_Sales ~ Item_Weight + Item_MRP + Item_Fat_Content_LF + Item_Fat_Content_Low_Fat + Item_Fat_Content_Regular + Item_Fat_Content_low_fat + Item_Fat_Content_reg + Outlet_Size_High + Outlet_Size_Medium + Outlet_Size_Small + Outlet_Location_Type_Tier_1 + Outlet_Location_Type_Tier_2 + Outlet_Location_Type_Tier_3 + Outlet_Type_Grocery_Store + Outlet_Type_Supermarket_Type1 + Outlet_Type_Supermarket_Type2 + Outlet_Type_Supermarket_Type3 + Item_Visibility_bins_Low_Viz + Item_Visibility_bins_Viz + Item_Visibility_bins_High_Viz'
```

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
[33] formula = 'Item_Outlet_Sales ~ ' + ' + '.join(list(train_new.drop('Item_Outlet_Sales', axis=1).columns))
```

```
print(formula)
```

```
Item_Outlet_Sales ~ Item_Weight + Item_MRP + Item_Fat_Content_LF + Item_Fat_Content_Low_Fat + Item_Fat_Content_Regular + It
```

```
[35] import statsmodels.formula.api as smf

     lm = smf.ols(formula=formula, data=train_new).fit()
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated.
  import pandas.util.testing as tm
```

```
[36] lm.summary()
```

OLS Regression Results

Dep. Variable:	Item_Outlet_Sales	**R-squared:**	0.563
Model:	OLS	**Adj. R-squared:**	0.562
Method:	Least Squares	**F-statistic:**	644.4
Date:	Sun, 27 Feb 2022	**Prob (F-statistic):**	0.00
Time:	16:12:34	**Log-Likelihood:**	-71996.
No. Observations:	8523	**AIC:**	1.440e+05
Df Residuals:	8505	**BIC:**	1.442e+05
Df Model:	17		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-166.0730	70.931	-2.341	0.019	-305.114	-27.032
Item_Weight	-0.3559	2.893	-0.123	0.902	-6.027	5.316
Item_MRP	15.5532	0.197	79.089	0.000	15.168	15.939
Item_Fat_Content_LF	-66.9833	60.247	-1.112	0.266	-185.082	51.115
Item_Fat_Content_Low_Fat	-38.7249	34.923	-1.109	0.268	-107.182	29.732
Item_Fat_Content_Regular	16.3128	36.356	0.449	0.654	-54.954	87.579
Item_Fat_Content_low_fat	9.1804	91.448	0.100	0.920	-170.079	188.440
Item_Fat_Content_reg	-85.8580	90.146	-0.952	0.341	-262.566	90.850
Outlet_Size_High	47.1895	86.435	0.546	0.585	-122.244	216.622
Outlet_Size_Medium	224.8751	69.295	3.245	0.001	89.040	360.710
Outlet_Size_Small	126.9344	45.389	2.797	0.005	37.962	215.907
Outlet_Location_Type_Tier_1	-140.3355	50.122	-2.800	0.005	-238.587	-42.084
Outlet_Location_Type_Tier_2	4.0691	43.276	0.094	0.925	-80.762	88.901
Outlet_Location_Type_Tier_3	-29.8065	52.263	-0.570	0.568	-132.255	72.642
Outlet_Type_Grocery_Store	-1651.6967	46.511	-35.512	0.000	-1742.869	-1560.525
Outlet_Type_Supermarket_Type1	242.6167	53.856	4.505	0.000	137.046	348.187
Outlet_Type_Supermarket_Type2	-241.2895	59.052	-4.086	0.000	-357.045	-125.534

```
[37] from sklearn.model_selection import train_test_split
```

```
[38] trainX, testX, trainY, testY = train_test_split(X,y, test_size=0.3)
```

```
[39] trainX.shape, testX.shape, trainY.shape, testY.shape
```

```
((5966, 20), (2557, 20), (5966,), (2557,))
```

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
lm = LinearRegression()
```

```
[41] lm.fit(trainX, trainY)

     LinearRegression()
```

```
[42] lm.intercept_

     -285.4565846464502
```

```
[43] lm.coef_

     array([ 9.64963732e-01,  1.56982526e+01, -3.60572034e+01, -2.80087304e+00,
             4.46007558e+01,  7.44761079e+01, -8.02187873e+01,  7.11155191e+01,
             2.36347056e+02,  1.71035469e+02, -8.08878392e+01,  3.26061442e+01,
             4.82816950e+01, -1.63036528e+03,  3.19242818e+02, -2.16815501e+02,
             1.52793797e+03, -3.18689544e+01, -6.17195173e+01, -3.86554009e+01])
```

```
[45] lm.score(trainX, trainY)

     0.5690553681116713
```

```
[46] lm.score(testX, testY)

     0.5464075571040605
```

```
[47] predicted_y = lm.predict(testX)
```

```
[48] predicted_y

     array([ 609.19877892, 1381.1537209 , -646.96747472, ..., 2955.30476952,
            3780.47940558, 1505.8965968 ])
```

IIM NAGPUR

BIG MART SALES PREDICTION USING MACHINE LEARNING

Now we are ready to apply ML algorithms

```
[ ] from sklearn.model_selection import train_test_split
```

```
[ ] train,test = train_test_split(data_dummy,test_size=0.20,random_state=2019)
```

```
[ ] train.shape , test.shape
```

```
((6816, 34), (1705, 34))
```

```
[ ] train_label=train['Item_Outlet_Sales']
    test_label=test['Item_Outlet_Sales']
    del train['Item_Outlet_Sales']
    del test['Item_Outlet_Sales']
```

Applying Linear Regression

```
[ ] from sklearn.linear_model import LinearRegression
```

```
[ ] lr=LinearRegression()
```

```
[ ] lr.fit(train,train_label)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
[ ] from sklearn.metrics import mean_squared_error
```

```
[ ] predict_lr=lr.predict(test)
```

```
[ ] test
```

	Item_MRP_a	Item_MRP_b	Item_MRP_c	Item_MRP_d	Outlet_Type_Grocery Store	Outlet_Type_Supermarket Type1	Outlet_Type_Supermarket Type2
1539	0	0	0	1	0	0	1
1389	0	1	0	0	1	0	0
3174	0	0	1	0	0	1	0
758	0	0	0	1	0	0	0
6863	0	0	1	0	0	1	0
...	...	...	...	...	...	...	...
3012	0	0	0	1	1	0	0
8025	1	0	0	0	1	0	0
7460	0	0	1	0	0	1	0

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
lr_score=lr.score(test, test_label)
```

```
lr_score
```

```
0.5473128874365277
```

B. RIDGE REGRESSION

It is a way to create a parsimonious model when the number of predictor variables in a set exceeds the number of observations, or when a data set has multicollinearity (correlations between predictor variables).

Ridge Regression

```
from sklearn.linear_model import Ridge
r=Ridge(alpha=0.05,solver='cholesky')
r.fit(train,train_label)
predict_r=r.predict(test)
r_score=r.score(test,test_label)
r_score
```

```
0.5464413494133833
```

C. LASSO

Lasso regression is **a type of linear regression that uses shrinkage**. Shrinkage is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models

```
from sklearn.linear_model import Lasso
l=Lasso(alpha=0.01)
l.fit(train,train_label)
predict_l=l.predict(test)
l_score=l.score(test,test_label)
l_score
```

```
0.5464400112047736
```

D. Elastic Net

Elastic net is a popular type of regularized linear regression that combines two popular penalties, specifically the L1 and L2 penalty functions. ... Elastic Net is an extension of linear regression that adds regularization penalties to the loss function during training.

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
from sklearn.linear_model import ElasticNet
en=ElasticNet(alpha=0.01,l1_ratio=0.5)
en.fit(train,train_label)
predict_r=en.predict(test)
en_score=en.score(test,test_label)
en_score
```

```
0.5455625028839739
```

E. Stochastic gradient

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression.

Stochastic gradient

```
from sklearn.linear_model import SGDRegressor
sgd=SGDRegressor(penalty='l2',n_iter_no_change=100,alpha=0.05)
sgd.fit(train,train_label)
predict_r=sgd.predict(test)
sgd_score=sgd.score(test, test_label)
sgd_score
```

```
0.5294480552760138
```

F. Support Vector Regression (SVR)

Support Vector Regression is a supervised learning algorithm that is **used to predict discrete values**. Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.

```
from sklearn.svm import SVR
svm=SVR(epsilon=15,kernel='linear')
svm.fit(train,train_label)
predict_r=svm.predict(test)
svm_score=svm.score(test, test_label)
svm_score
```

```
0.37648591370848317
```

G. Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label.

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
from sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor()
dtr.fit(train,train_label)
predict_r=dtr.predict(test)
dtr_score=dtr.score(test, test_label)
dtr_score
```

```
0.5826004843412711
```

H. Random forest

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Random Forest

```
[ ] from sklearn.ensemble import RandomForestRegressor
    rf=RandomForestRegressor()
    rf.fit(train,train_label)
    predict_r=rf.predict(test)
    rf_score=rf.score(test, test_label)
    rf_score
```

```
0.5833104764283761
```

I. Bagging Regression

A Bagging regressor is an ensemble meta-estimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction.

Bagging Regressoion

```
[ ] from sklearn.ensemble import BaggingRegressor
```

```
[ ] br=BaggingRegressor(max_samples=70)
```

```
[ ] br.fit(train,train_label)
```

```
BaggingRegressor(base_estimator=None, bootstrap=True, bootstrap_features=False,
                 max_features=1.0, max_samples=70, n_estimators=10, n_jobs=None,
                 oob_score=False, random_state=None, verbose=0,
                 warm_start=False)
```

```
[ ] score=br.predict(test)
```

```
[ ] br_score=br.score(test, test_label)
    br_score
```

```
0.5620187543059627
```

J. Adaptive Boosting

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances.

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
[ ] from sklearn.ensemble import AdaBoostRegressor
    ada=AdaBoostRegressor()
    ada.fit(train,train_label)
    g=ada.predict(test)
    ada_score=ada.score(test, test_label)
    ada_score
```

```
0.5430150476112474
```

K. Gradient Boosting

Gradient boosting is a type of machine learning boosting. It relies on **the intuition** that the best possible next model, when combined with previous models, minimizes the overall prediction error. ... If a small change in the prediction for a case causes no change in error, then next target outcome of the case is zero.

```
[ ] from sklearn.ensemble import GradientBoostingRegressor
    gbr=GradientBoostingRegressor()
    gbr.fit(train,train_label)
    p=gbr.predict(test)
    gb_score=gbr.score(test, test_label)
    gb_score
```

```
0.5829723327412085
```

11.Metrics for Data Modelling

The coefficient of determination R2 (R-squared) is a statistic that measures the goodness of a model's fit i.e. how well the real data points are approximated by the predictions of regression. Higher values of R2 suggest higher model accomplishments in terms of prediction along with accuracy, and the value 1 of R2 is indicative of regression predictions perfectly fitting the real data points. For further better results, the use of adjusted R2 measures works wonders. Taking logarithmic values of the target column in the dataset proves to be significant in the prediction process. So, it can be said that on taking adjustments of columns used in prediction, better results can be deduced. One way of incorporating adjustment could also have included taking square root of the column. It also provides better visualization of the dataset and target variable as the square root of target variable is inclined to be a normal distribution.

The error measurement is an important metric in the estimation period. Root mean squared error (RMSE) and Mean Absolute Error (MAE) are generally used for continuous variable's accuracy measurement. It can be said that the average model prediction error can be expressed in units of the variable of interest by using both MAE and RMSE. MAE is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. The square root of the average of squared differences between prediction and actual observation can be termed as RMSE. RMSE is an absolute measure of fit, whereas R2 is a relative measure of fit. RMSE helps in measuring the variable's

average error and it is also a quadratic scoring rule. Low RMSE values obtained for linear or multiple regression corresponds to better model fitting.

BIG MART SALES PREDICTION USING MACHINE LEARNING

With respect to the results obtained in this work, it can be said that there is no big difference between our train and test sample since the metric RMSE ratio is calculated to be equal to the ratio between train and test sample. The results related to how accurately responses are predicted by our model can be inferred from RMSE as it is a good measure along with measuring precision and other required capabilities. A considerable improvement could be made by further data exploration incorporated with outlier detection and high leverage points.

```
name=['Linear Regression','Ridge Regression','Lasso Regression'
    ,'Elastic Net Regression','SGD Regression',
     'SVM','Decision Tree','Random Forest','Ada Boost',
     'Bagging','Gradient Boost']
```

```
go=pd.DataFrame({'RMSE':[lr_score,r_score,l_score,en_score,
                         sgd_score,svm_score,dtr_score,rf_score,
                         ada_score,br_score,gb_score]},index=name)
```

	RMSE
Linear Regression	0.547313
Ridge Regression	0.546441
Lasso Regression	0.546440
Elastic Net Regression	0.545563
SGD Regression	0.529448
SVM	0.376486
Decision Tree	0.582600
Random Forest	0.583310
Ada Boost	0.543015
Bagging	0.562019
Gradient Boost	0.582972

BIG MART SALES PREDICTION USING MACHINE LEARNING

```
go.RMSE.sort_values()
```

```
SVM                        0.376486
SGD Regression             0.529448
Ada Boost                  0.543015
Elastic Net Regression     0.545563
Lasso Regression           0.546440
Ridge Regression           0.546441
Linear Regression          0.547313
Bagging                    0.562019
Decision Tree              0.582600
Gradient Boost             0.582972
Random Forest              0.583310
Name: RMSE, dtype: float64
```

12 Conclusion & Future Scope

In this paper, basics of machine learning and the associated data processing and modelling algorithms have been described, followed by their application for the task of sales prediction in Big Mart shopping centers at different locations. On implementation, the prediction results show the correlation among different attributes considered and how a particular location of medium size recorded the highest sales, suggesting that other shopping locations should follow similar patterns for improved sales.

Multiple instances parameters and various factors can be used to make this sales prediction more innovative and successful. Accuracy, which plays a key role in prediction-based systems, can be significantly increased as the number of parameters used are increased. Also, a look into how the sub-models work can lead to increase in productivity of system. The project can be further

collaborated in a web-based application or in any device supported with an in-built intelligence by virtue of Internet of Things (IoT), to be more feasible for use. Various stakeholders concerned with sales information can also provide more inputs to help in hypothesis generation and more instances can be taken into consideration such that more precise results that are closer to real world situations are generated.

When combined with effective data mining methods and properties, the traditional means could be seen to make a higher and positive effect on the overall development of corporation's tasks on the whole. One of the main highlights is more expressive regression outputs, which are more understandable bounded with some of accuracy. Moreover, the flexibility of the proposed approach can be increased with variants at a very appropriate stage of regression model-building. There is a further need of experiments for proper measurements of both accuracy and resource efficiency to assess and optimize correctly.