Intel College Excellence Program
Project Synopsis

“Big Mart Sale prediction using Regression”

Team member’s detail			
S.No.	**Participant Name**	**Mobile No.**	**Email ID**
1	Manish Kumar	7827043684	***sahmanish10987@gmail.com***
2	Sakshi Pandey	9821955176	***sakshipandey21022003@gmail.com***
Faculty(college) mentor detail			
S.No.	**Mentor Name**	**Mobile No.**	**Email ID**
1	Anjum Mohd. Aslam	7347058401	***anjum.aslam@galgotiasuniversity.edu.in***
College/University Name			
Galgotias University			

School of Computing Science and Engineering,

Galgotias University

BACKGROUND

Nowadays, shopping malls and Big Marts organizations are expanding their businesses globally, so Sales Prediction is a big matter these days for predicting future customer demand. Sales forecasting can assist a company in working and growing in the appropriate path. We propose a regression-based predictive model for Big Mart sales analysis. The sales volume of Big Mart is forecasted by analyzing the obtained data set using the Regression model.

PROBLEM IDENTIFICATION

As business of big mart, shopping malls are increasing day by day so demand and need of predicting the customer demand and future sale is also increasing. We study the dataset of items sells in supermarket, grocery store, items fat content, type, item size, item visibility, mrp etc and tries to find which factor effect the sale of items.

PROPOSED SOLUTION

We used some of the regression technique like linear regression, decision tree regression and random forest to predict the result with higher accuracy. To perform regression, firstly it is important to clean the data set. Then apply algorithms on it. Linear regression model to predict the value of a dependent variable (y) based on the value of an independent variable (x). As a result of this regression technique, a linear relationship between x i.e., input and y i.e., output is found.

$$y=a+b*x$$

where a = intercept

b = slope of line

In Decision tree is used for regression problems where you are trying to predict something with infinite possible answers such as sale of big mart. Decision trees can be used for either classification or regression problems and are useful for complex datasets. Random forest is a supervised machine learning algorithm that is commonly used to solve classification and regression problems. It creates decision trees from various samples, using the majority vote for classification and the average for regression.

Algorithm

Step 1: Import the dataset
Step 2: Read the dataset
Step 3: Calculate the total missing in each column of dataset
Step 4: Perform Data cleaning
Step 5: Imputing Missing Values
Step 6: Data understanding through visualization (Compare every column with sales to observe which aspect is affecting sale of item)
Step 7: Apply different regression technique and observe the result.

APPROACH TAKEN

"To find out what role certain properties of an item play and how they affect their sales by understanding Big Mart sales." In order to help Big Mart, achieve this goal, a predictive model can be built to find out for every store, the key factors that can increase their sales and what changes could be made to the product or store's characteristics.

Methodology

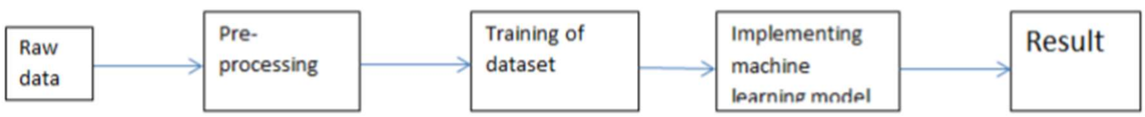


Fig1: Steps followed for obtaining results

Dataset and its processing

Big Mart’s data scientists collected sales data of their 10 stores situated at different locations with each store having 1559 different products as per 2013 data collection. Using all the observations it is inferred what role certain properties of an item play and how they affect its sales.

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

Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermarket Type1	3735.1380
Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Medium	Tier 3	Supermarket Type2	443.4228
Low Fat	0.016760	Meat	141.6180	OUT049	1999	Medium	Tier 1	Supermarket Type1	2097.2700
Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	NaN	Tier 3	Grocery Store	732.3800
Low Fat	0.000000	Household	53.8614	OUT013	1987	High	Tier 3	Supermarket Type1	994.7052

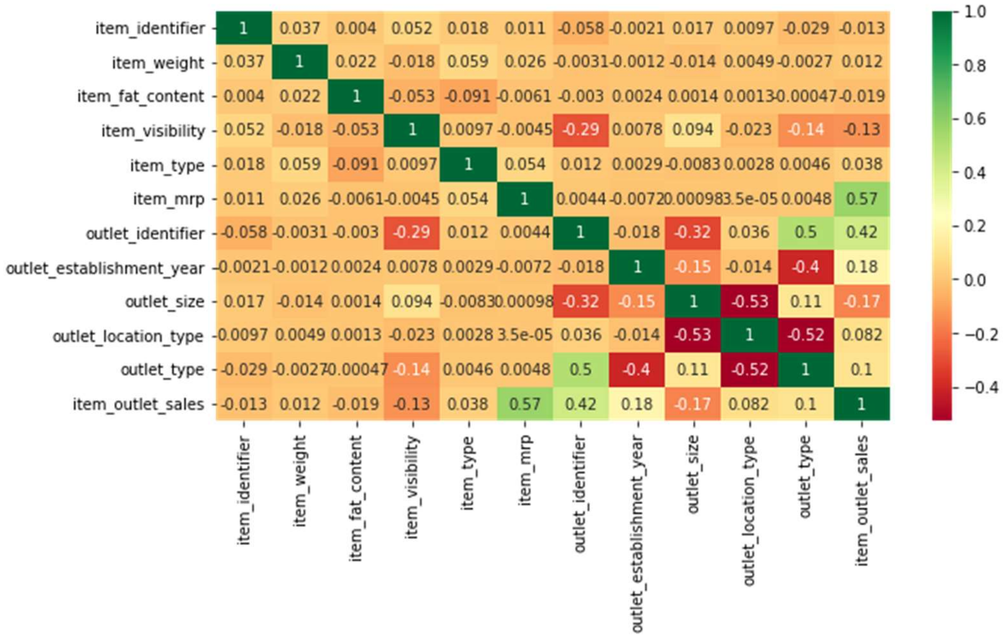
After cleaning of dataset

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. Shown above are some of the representations obtained by using Pandas tools which tells about variable count for numerical columns and modal values for categorical columns. Maximum and minimum values in numerical columns, along with their percentile values for median, plays an important factor in deciding which value to be chosen at priority for further exploration tasks and analysis. Data types of different columns are used further in label processing and one-hot encoding scheme during model building.

	mean
	item_weight
item_type	
Baking Goods	12.277108
Breads	11.346936
Breakfast	12.768202
Canned	12.305705
Dairy	13.426069
Frozen Foods	12.867061
Fruits and Vegetables	13.224769
Hard Drinks	11.400328
Health and Hygiene	13.142314
Household	13.384736
Meat	12.817344
Others	13.853285
Seafood	12.552843
Snack Foods	12.987880
Soft Drinks	11.847460
Starchy Foods	13.690731

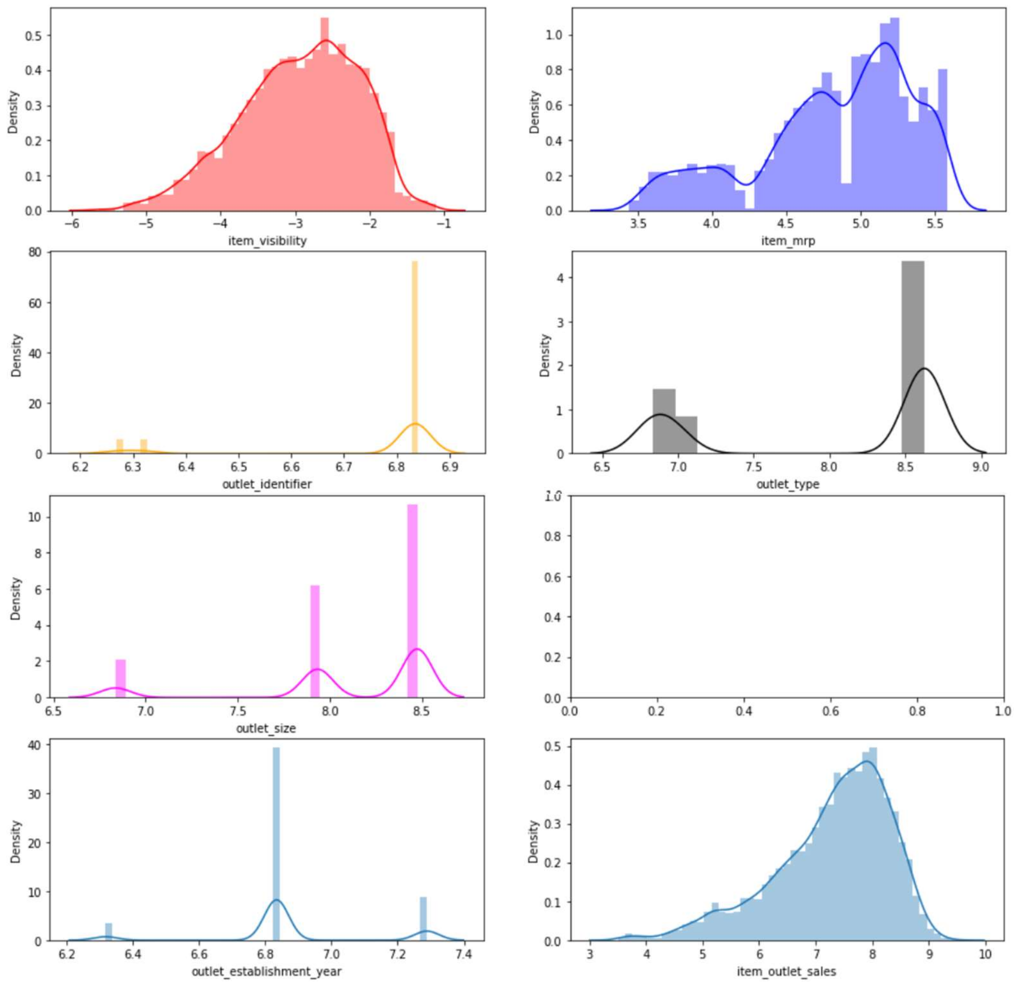
```
data_shopping.describe()
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	7060.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.643456	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.773750	0.026989	93.826500	1987.000000	834.247400
50%	12.600000	0.053931	143.012800	1999.000000	1794.331000
75%	16.850000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800



This Figure shows the correlation among each column of the dataset.

Visualizing the skewness of data



Implementation of Machine Learning Models

```
#Splitting The data  into Train and Test Dataset:
from sklearn.model_selection import train_test_split
x_train,x_test, y_train, y_test = train_test_split(x,y, test_size =0.20, random_state = 13)
```

```
#Applying Linear Regression Model
from sklearn.linear_model import LinearRegression
regressor =LinearRegression()
regressor.fit(x_train, y_train)
```

```
LinearRegression()
```

```
from sklearn.tree import DecisionTreeRegressor
regr = DecisionTreeRegressor()
regr.fit(x,y)
```

```
DecisionTreeRegressor()
```

```
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
lr = LinearRegression(normalize=True)
svr = SVR()
#knr = KNeighborsRegressor()
dt = DecisionTreeRegressor(criterion='mse',max_depth=3)
rf = RandomForestRegressor(n_estimators=10,max_depth=5)
gbr = GradientBoostingRegressor()
```

Result of Machine Learning

```
#Accuracy of Model (Apply R2_score)
from sklearn.metrics import r2_score, mean_squared_error
r2_score(y_test, y_pred)
```

```
0.7079953565588073
```

```
#Checking Root Mean Square error
from math import sqrt
rmse = sqrt(mean_squared_error(y_test,  y_pred))
rmse
```

```
0.5378824974652983
```

```
# Printing Accuracy data
print("Training Accuracy for Decision Tree regressor :", regr.score(x_train, y_train))
```

```
Training Accuracy for Decision Tree regressor : 1.0
```

```
rf.fit(x_train,y_train)
score_reg(rf, x_test, y_test)
```

```
Mean Absolute Error: 0.40804162525898385
Mean Squared Error: 0.27993410438518807
Root Mean Squared Error: 0.529087993045758
Root Mean Squared Log Error 0.06750589885959728
```

```
gbr.fit(x_train,y_train)
score_reg(gbr,x_test, y_test)
```

```
Mean Absolute Error: 0.4055378067932022
Mean Squared Error: 0.27751505150257066
Root Mean Squared Error: 0.5267969737029349
Root Mean Squared Log Error 0.06720080068651171
```

```
lr.fit(x_train,y_train)
score_reg(lr, x_test, y_test)
```

```
Mean Absolute Error: 0.418178247561712
Mean Squared Error: 0.2893175810795063
Root Mean Squared Error: 0.5378824974652979
Root Mean Squared Log Error 0.06877739898512474
```

```
svr.fit(x_train,y_train)
score_reg(svr, x_test, y_test)
```

```
Mean Absolute Error: 0.41437346646624257
Mean Squared Error: 0.2973646795713643
Root Mean Squared Error: 0.545311543588951
Root Mean Squared Log Error 0.07034668102909111
```

```
dt.fit(x_train,y_train)
score_reg(dt, x_test, y_test)
```

```
Mean Absolute Error: 0.4463496488145625
Mean Squared Error: 0.3299725316482588
Root Mean Squared Error: 0.5744323560248489
Root Mean Squared Log Error 0.07280812543015881
```

HARDWARE & SOFWARE REQUIREMENTS

Hardware requirements:

1. PC/Laptop

Software requirements:

1. Anaconda

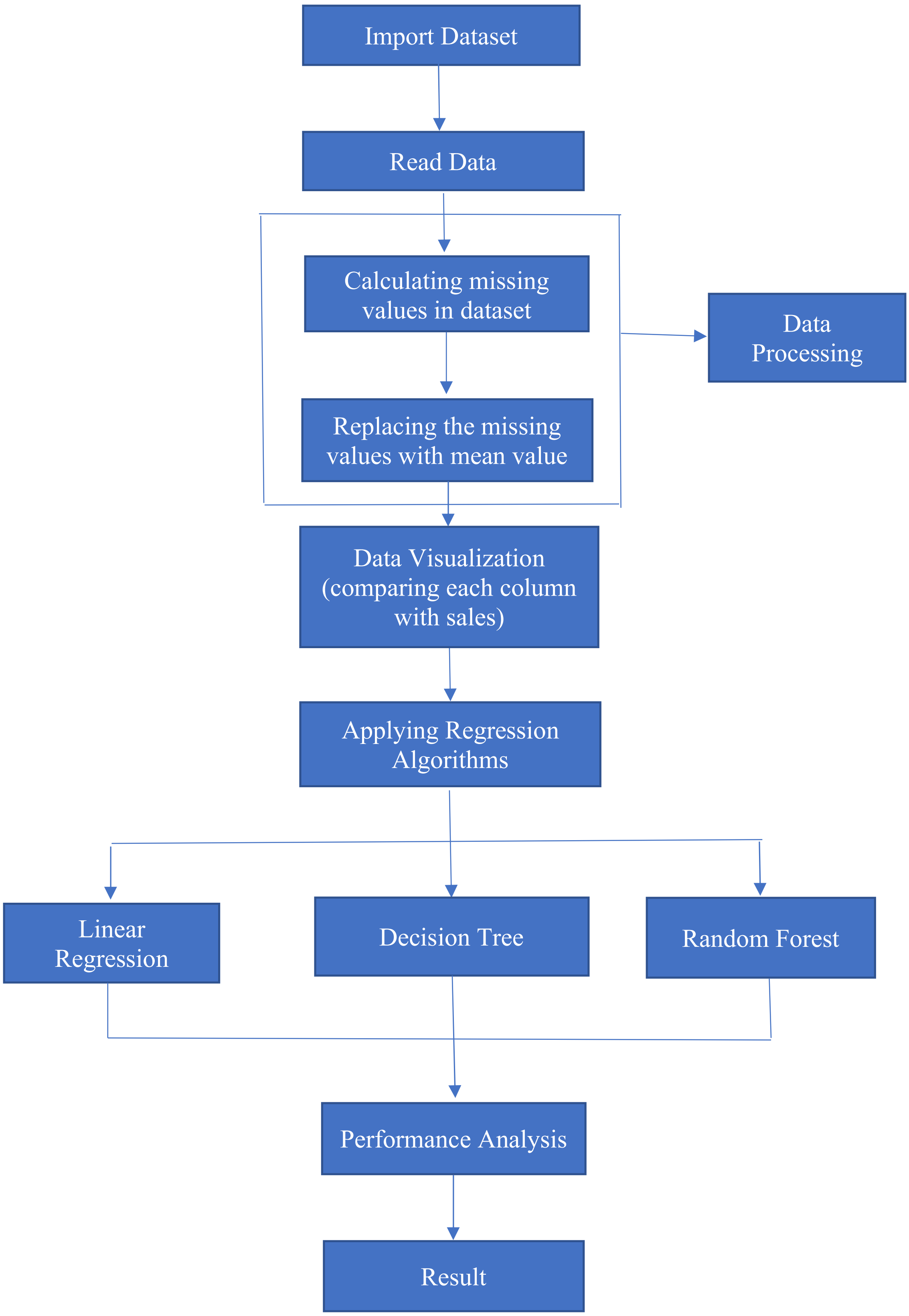
2. Jupyter Notebook

3. Python Libraries installed

    1. Scikit-Learn
    2. Numpy
    3. Pandas
    4. Scipy
    5. Matplotlib (Pyplot)
    6. Seaborn

BLOCK DIAGRAM & DESCRIPTION

We have drawn block diagram for the project. This ER diagram is representing the process of project. Firstly, we have taken the dataset from Kaggle and processed the data. We used anaconda python platform for exploratory data analysis and visualization. Data preprocessing is mandatory for any machine learning or data mining approach, since the performance of a machine learning methodology depends on how well the dataset is prepared and structured



RESULT

Machine Learning algorithms did a great job in the testing phase in this supervised learning environment, and the algorithms used in this will definitely perform well in real Big Mart company to predict the sales. But our best choice would be to work with decision tree regressor algorithm as it was the only one regression algorithm which was able to accomplish 100% accuracy. And the rest of the results does not exceed 70% (approx.) accuracy. So, in conclusion this project would definitely bring accurate results, if worked with decision tree regressor.

It must also be noted that all the accuracy mentioned in Approach Taken is done considering only those columns which have a influence in SALES. And those columns are 'item_visibility', 'item_mrp', 'outlet_identifier', 'outlet_establishment_year', 'outlet_size', and 'outlet_type'.

FUTURE SCOPE

We can work on more dataset and try to apply more algorithm to increase accuracy as predicting the sale is currently on high demand as it helps business to grow positively and its scope will increase in future as well

CONCLUSION

So, in this project we observe that item fat contains, item price and selling sites is influencing the sale of item. By using algorithms and analyzing we tried to get more accurate result. Linear regression gives accuracy of 78% and decision tree algorithm is giving accuracy of 100%. This model can help to predict the future sale of Big Mart.

REFERENCES

https://github.com/Sah-Manish/Intel-COE-ML-Project.git