MUFFAKHAM JAH COLLEGE OF
ENGINEERING AND
TECHNOLOGY(MJCET)

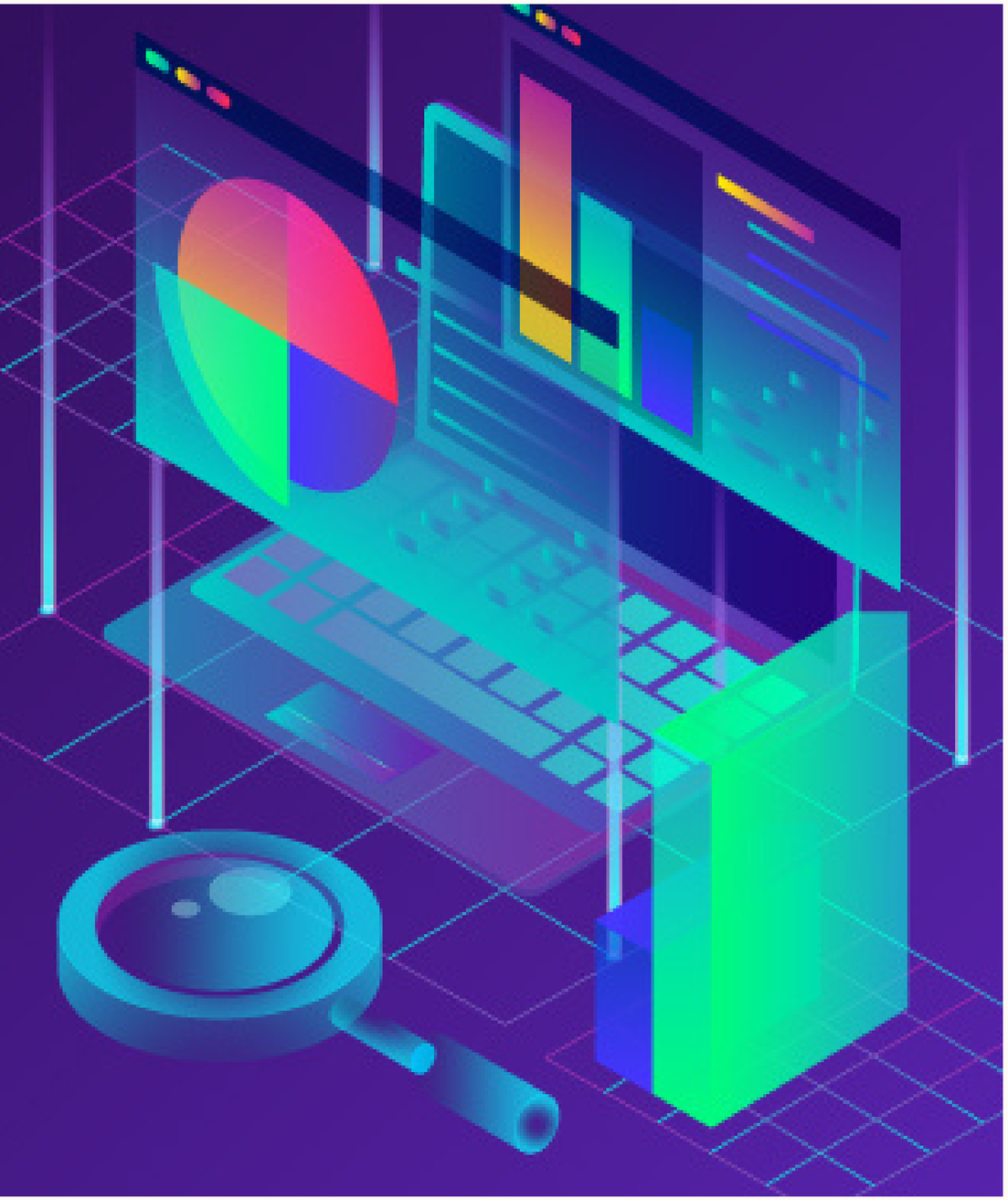
DST:LEVEL_11A

DATA SCIENCE PROJECT

BY: HANIA GHOUSE

Overview

- Introduction
- Problem Statement
- Dataset
- Attribute
- Data Exploaration
- Data Cleaning
- Data Visualization
- Feature Engineering
- Feature Encoding
- Feature Selection
- Feature Scaling
- Model Building
- Result
- Conclusion



Introduction

- BIG MART SALES PREDICTION:

- 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 BigMart 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.

Problem Statement

Objective

- 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..

DATASET

```
Item_Identifier,Item_Weight,Item_Fat_Content,Item_Visibility,Item_Type,Item_MRP,Outlet_Identifier,Outlet_Establishment_Year,Outlet_Size,Outlet_Location_Type,Outlet_Type,Item_Outlet_Sales
FDA15,9.3,Low Fat,0.016047301,Dairy,249.8092,OUT049,1999,Medium,Tier 1,Supermarket Type1,3735.138
DRC01,5.92,Regular,0.019278216,Soft Drinks,48.2692,OUT018,2009,Medium,Tier 3,Supermarket Type2,443.4228
FDN15,17.5,Low Fat,0.016760075,Meat,141.618,OUT049,1999,Medium,Tier 1,Supermarket Type1,2097.27
FDX07,19.2,Regular,0,Fruits and Vegetables,182.095,OUT010,1998,,Tier 3,Grocery Store,732.38
NCD19,8.93,Low Fat,0,Household,53.8614,OUT013,1987,High,Tier 3,Supermarket Type1,994.7052
FDP36,10.395,Regular,0,Baking Goods,51.4008,OUT018,2009,Medium,Tier 3,Supermarket Type2,556.6088
FDO10,13.65,Regular,0.012741089,Snack Foods,57.6588,OUT013,1987,High,Tier 3,Supermarket Type1,343.5528
FDP10,,Low Fat,0.127469857,Snack Foods,107.7622,OUT027,1985,Medium,Tier 3,Supermarket Type3,4022.7636
FDH17,16.2,Regular,0.016687114,Frozen Foods,96.9726,OUT045,2002,,Tier 2,Supermarket Type1,1076.5986
FDU28,19.2,Regular,0.09444959,Frozen Foods,187.8214,OUT017,2007,,Tier 2,Supermarket Type1,4710.535
FDY07,11.8,Low Fat,0,Fruits and Vegetables,45.5402,OUT049,1999,Medium,Tier 1,Supermarket Type1,1516.0266
FDA03,18.5,Regular,0.045463773,Dairy,144.1102,OUT046,1997,Small,Tier 1,Supermarket Type1,2187.153
FDX32,15.1,Regular,0.1000135,Fruits and Vegetables,145.4786,OUT049,1999,Medium,Tier 1,Supermarket Type1,1589.2646
FDS46,17.6,Regular,0.047257328,Snack Foods,119.6782,OUT046,1997,Small,Tier 1,Supermarket Type1,2145.2076
FDF32,16.35,Low Fat,0.0680243,Fruits and Vegetables,196.4426,OUT013,1987,High,Tier 3,Supermarket Type1,1977.426
FDP49,9,Regular,0.069088961,Breakfast,56.3614,OUT046,1997,Small,Tier 1,Supermarket Type1,1547.3192
NCB42,11.8,Low Fat,0.008596051,Health and Hygiene,115.3492,OUT018,2009,Medium,Tier 3,Supermarket Type2,1621.8888
FDP49,9,Regular,0.069196376,Breakfast,54.3614,OUT049,1999,Medium,Tier 1,Supermarket Type1,718.3982
DRI11,,Low Fat,0.034237682,Hard Drinks,113.2834,OUT027,1985,Medium,Tier 3,Supermarket Type3,2303.668
FDU02,13.35,Low Fat,0.10249212,Dairy,230.5352,OUT035,2004,Small,Tier 2,Supermarket Type1,2748.4224
FDN22,18.85,Regular,0.138190277,Snack Foods,250.8724,OUT013,1987,High,Tier 3,Supermarket Type1,3775.086
FDW12,,Regular,0.035399923,Baking Goods,144.5444,OUT027,1985,Medium,Tier 3,Supermarket Type3,4064.0432
NCB30,14.6,Low Fat,0.025698134,Household,196.5084,OUT035,2004,Small,Tier 2,Supermarket Type1,1587.2672
FDC37,,Low Fat,0.057556998,Baking Goods,107.6938,OUT019,1985,Small,Tier 1,Grocery Store,214.3876
FDR28,13.85,Regular,0.025896485,Frozen Foods,165.021,OUT046,1997,Small,Tier 1,Supermarket Type1,4078.025
NCD06,13,Low Fat,0.099887103,Household,45.906,OUT017,2007,,Tier 2,Supermarket Type1,838.908
FDV10,7.645,Regular,0.066693437,Snack Foods,42.3112,OUT035,2004,Small,Tier 2,Supermarket Type1,1065.28
DRJ59,11.65,low fat,0.019356132,Hard Drinks,39.1164,OUT013,1987,High,Tier 3,Supermarket Type1,308.9312
FDE51,5.925,Regular,0.161466534,Dairy,45.5086,OUT010,1998,,Tier 3,Grocery Store,178.4344
FDC14,,Regular,0.072221801,Canned,43.6454,OUT019,1985,Small,Tier 1,Grocery Store,125.8362
FDV38,19.25,Low Fat,0.170348551,Dairy,55.7956,OUT010,1998,,Tier 3,Grocery Store,163.7868
```

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.

ATTRIBUTES

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.

DATA EXPLORATION

```
[ ] df.shape
```

```
(8523, 12)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Item_Identifier            8523 non-null   object 
 1   Item_Weight                7060 non-null   float64
 2   Item_Fat_Content           8523 non-null   object 
 3   Item_Visibility            8523 non-null   float64
 4   Item_Type                  8523 non-null   object 
 5   Item_MRP                   8523 non-null   float64
 6   Outlet_Identifier          8523 non-null   object 
 7   Outlet_Establishment_Year  8523 non-null   int64  
 8   Outlet_Size                6113 non-null   object 
 9   Outlet_Location_Type       8523 non-null   object 
 10  Outlet_Type                8523 non-null   object 
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

```
df.head(3)
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Locatior
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	

```
df.shape
```

```
(8523, 12)
```

```
df.info()
```

Data Cleaning

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

```
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
```

```
#   WE HAVE 2 COLUMNS WITH MISSING VALUES-1) ITEM_WEIGHT 2) OUTLET_SIZE
# NOW WE HAVE TO DO DATA CLEANING
```

```
df["Item_Weight"].fillna(df["Item_Weight"].mean(),inplace=True)
```

```
df["Outlet_Size"].unique()
```

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

```
df['Outlet_Size'].value_counts()
```

```
Medium    2793
Small     2388
High       932
Name: Outlet_Size, dtype: int64
```

```
df['Outlet_Size'].fillna(df['Outlet_Size'].mode()[0],inplace=True)
```

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

```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content             0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
```


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

```
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
```

```
#   WE HAVE 2 COLUMNS WITH MISSING VALUES-1) ITEM_WEIGHT 2) OUTLET_SIZE
# NOW WE HAVE TO DO DATA CLEANING
```

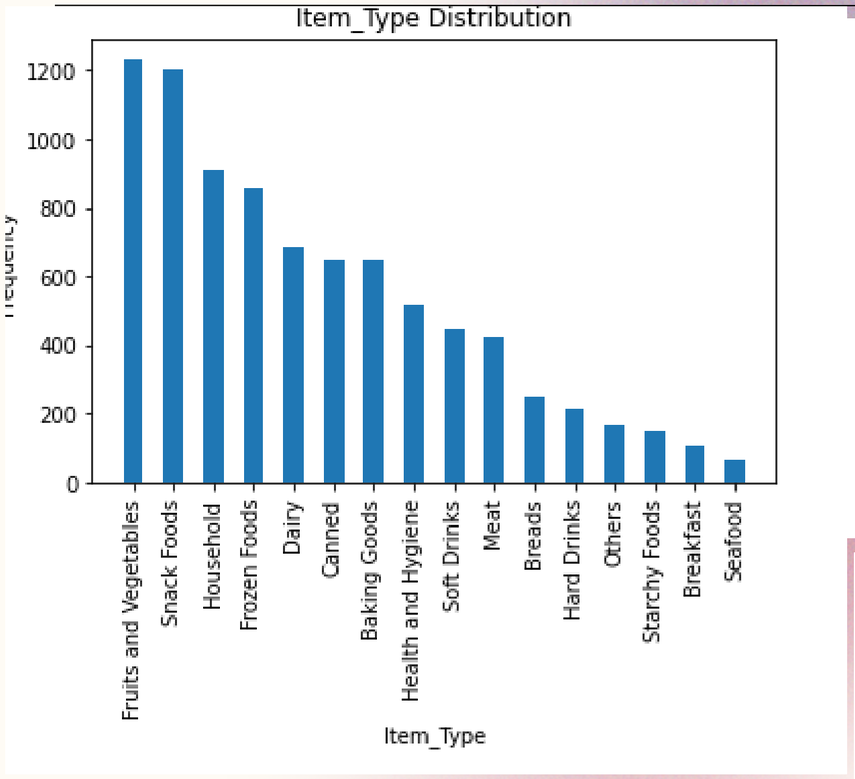

```
 6   Outlet_Identifier          8523 non-null   object
 7   Outlet_Establishment_Year  8523 non-null   int64
 8   Outlet_Size                8523 non-null   object
 9   Outlet_Location_Type       8523 non-null   object
 10  Outlet_Type                8523 non-null   object
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

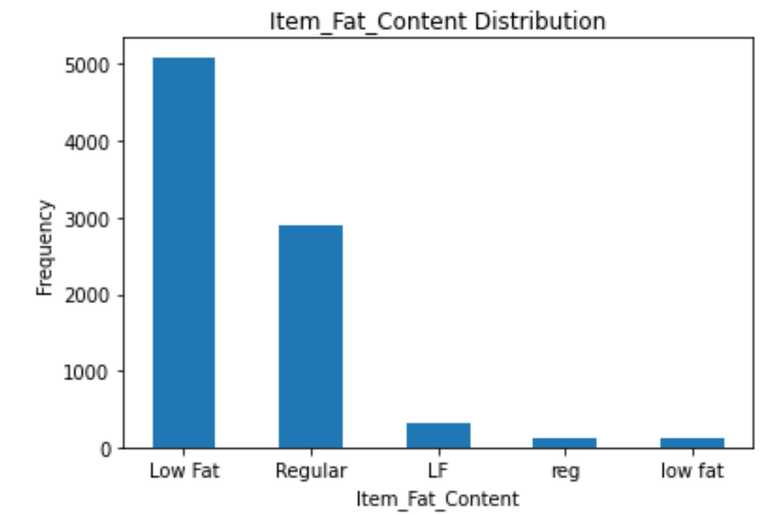
```
df['Item_Fat_Content'].value_counts()
```

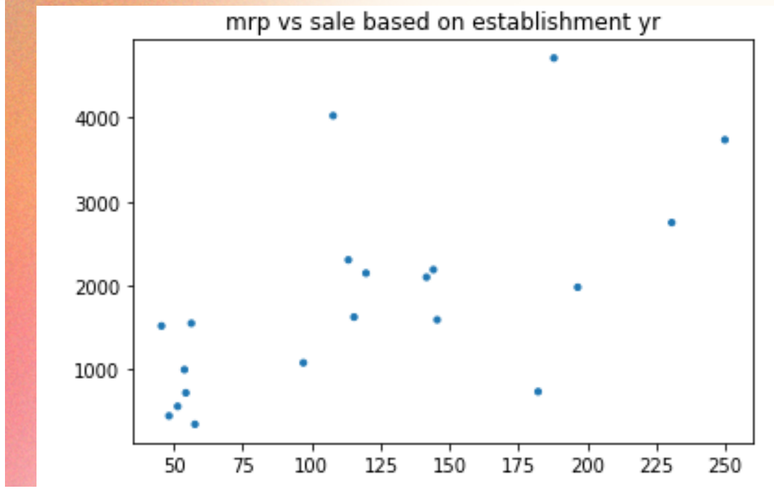
```
Low Fat    5089
Regular    2889
LF          316
reg         117
low fat     112
Name: Item_Fat_Content, dtype: int64
```

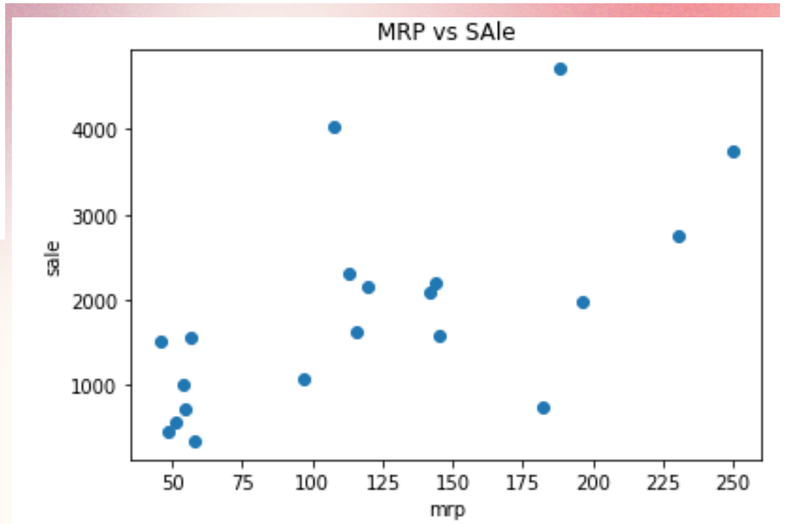
```
df['New_Item_Fat_Content']=df['Item_Fat_Content'].map({'Low Fat' : 'Low Fat','LF': "Low Fat", "low fat":"Low Fat", "reg":"Regular","Regular":"Regular"})
```

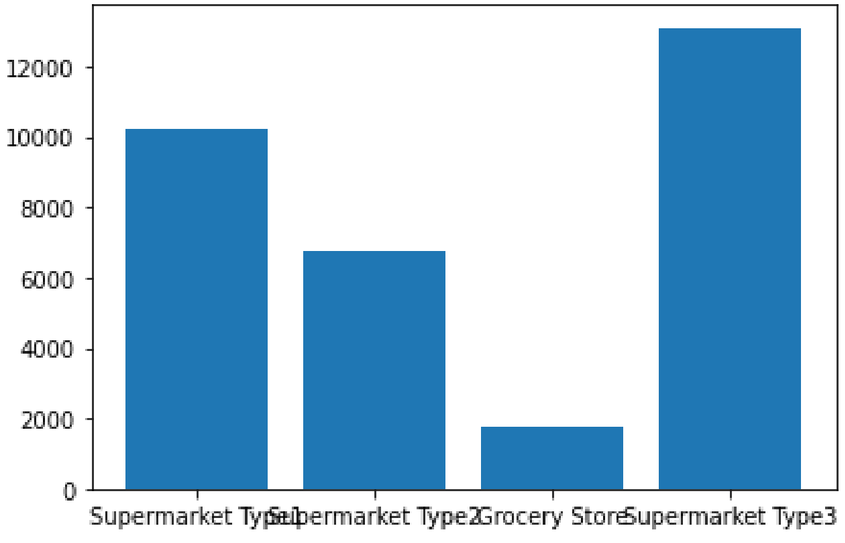
Data Visualization





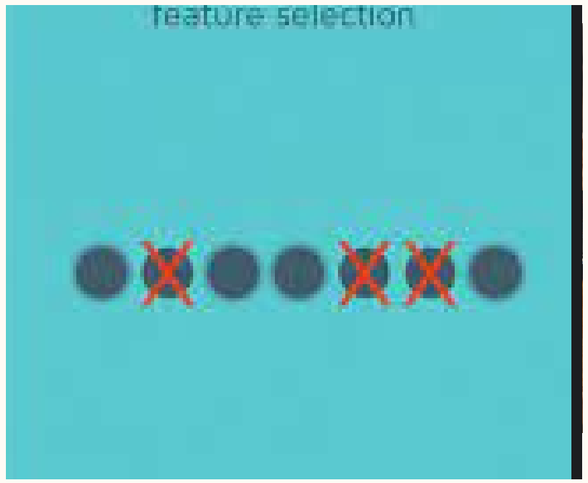






FEATURE ENGINEERING

Back to Overview

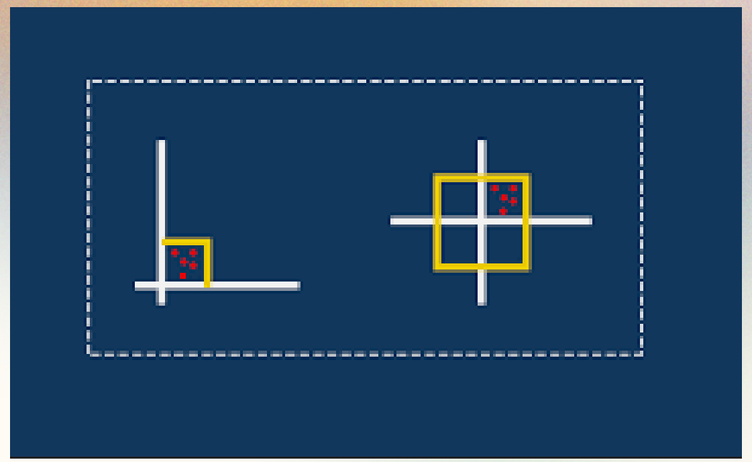


FEATURE SELECTION.

It is the method of reducing input variable to our model by using only relevant data and getting rid of noise in the data

FEATURE ENCODING

ML model can only work on numeric data . for this reason, it is necessary to transform the categorical values to numerical values



FEATURE SCALING

It is a technique to standardize the independent features present in the data into a fixed range

FEATURE ENCODING

```
ce.OrdinalEncoder(cols=df['New_Item_Fat_Content'],return_df=True,
                  mapping=[{'col':'New_Item_Fat_Content',
                            'mapping':{'Low Fat':1,"Regular":2}}])
```

+ Code + Text

```
_transformed_4=encod.fit_transform(df_train_transformed_3)
```

```
_transformed_4.head(5)
```

Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales	New_Item_Fat
Dairy	249.8092	OUT049	1999	2	1	1	3735.1380	
Soft Drinks	48.2692	OUT018	2009	2	3	2	443.4228	
Meat	141.6180	OUT049	1999	2	1	1	2097.2700	
Fruits and Vegetables	182.0950	OUT010	1998	2	3	4	732.3800	
Household	53.8614	OUT013	1987	1	3	1	994.7052	

```
encoder = ce.OrdinalEncoder(cols=df['Outlet_Location_Type'],return_df=True,
                            mapping=[{'col':'Outlet_Location_Type',
                                      'mapping':{'Tier 1':1,"Tier 2":2,"Tier 3":3}}])
```

+ Code + Text

```
[ ] df_train_transformed_2=encoder.fit_transform(df_train_transformed)
```

```
[ ] df_train_transformed_2.head(5)
```

at_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type
Low Fat	0.016047	Dairy	249.8092	OUT049	1999	2	1
Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	2	3
Low Fat	0.016760	Meat	141.6180	OUT049	1999	2	1

```
ncod = ce.OrdinalEncoder(cols=df['Outlet_Type'],return_df=True,
                         mapping=[{'col':'Outlet_Type',
                                   'mapping':{'Supermarket Type1':1,"Supermarket Type2":2,"Supermarket Type3":3,"Grocery Store":4}}])
```

```
f_train_transformed_3=encod.fit_transform(df_train_transformed_2)
```

```
f_train_transformed_3.head(5)
```

ility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sale
16047	Dairy	249.8092	OUT049	1999	2	1	1	3735.138
19278	Soft Drinks	48.2692	OUT018	2009	2	3	2	443.422
16760	Meat	141.6180	OUT049	1999	2	1	1	2097.270
00000	Fruits and Vegetables	182.0950	OUT010	1998	2	3	4	732.380
00000	Household	53.8614	OUT013	1987	1	3	1	994.705

```
[ ] import category_encoders as ce
```

```
encoders = ce.OrdinalEncoder(cols=df['Outlet_Size'],return_df=True,
                             mapping=[{'col':'Outlet_Size',
                                       'mapping':{'High':1,"Medium":2,"Small":3}}])
```

```
[ ] df_train_transformed=encoders.fit_transform(df)
```

```
[ ] df_train_transformed.head(3)
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Size	Outlet_Locatio
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	2	
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	2	
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	2	

Feature Selection

Back to Overview

SPLITTING THE DATA:

```
df_train_transformed_4.drop(['Item_Identifier','Item_Fat_Content','Item_Type','Outlet_Identifier','New_Item_Fat_Content'],axis=1,inplace=True)
```

+ Code + Text

```
[ ] df_train_transformed_4
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales	Breads	Breakfast	...
0	9.300	0.016047	249.8092	1999	2	1	1	3735.1380	0	0	...
1	5.920	0.019278	48.2692	2009	2	3	2	443.4228	0	0	...
2	17.500	0.016760	141.6180	1999	2	1	1	2097.2700	0	0	...
3	19.200	0.000000	182.0950	1998	2	3	4	732.3800	0	0	...
4	8.930	0.000000	53.8614	1987	1	3	1	994.7052	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
8518	6.865	0.056783	214.5218	1987	1	3	1	2778.3834	0	0	...
8519	8.380	0.046982	108.1570	2002	2	2	1	549.2850	0	0	...

```
x=df_train_transformed_4.drop('Item_Outlet_Sales',axis=1)
```

```
]  y = df_train_transformed_4['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=0.33, random_state=10)
```

FEATURE SCALING

Back to Overview

```
from sklearn.preprocessing import MinMaxScaler
```

Loading...

```
norm=MinMaxScaler().fit(x_train)
x_train_norm=norm.transform(x_train)
```

```
norm= MinMaxScaler().fit(x_test)
x_test_norm=norm.transform(x_test)
```

```
x_test_norm
```

```
array([[0.4943522 , 0.29668882, 0.60112569, ..., 0.        , 1.        ,
        0.        ],
       [0.19886871, 0.28797134, 0.31055994, ..., 0.        , 0.        ,
        0.        ],
       [0.48198869, 0.23203687, 0.6798286 , ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.75290265, 0.03678547, 0.56517561, ..., 0.        , 0.        ,
        0.        ],
       [0.8600774 , 0.29693413, 0.0709265 , ..., 0.        , 0.        ,
        0.        ],
       [0.4943522 , 0.20891107, 0.70530582, ..., 0.        , 0.        ,
        0.        ]])
```

```
[ ] scaled_features_df=pd.DataFrame(x_train_norm,index=x_train.index ,columns=x_train.columns)
```

```
scaled_features_df.head(10)
```

Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Breads	Breakfast	Canned	...	Fruits and Vegetables
0.761834	0.513790	0.612828	0.500000	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0
0.270616	0.419592	0.603210	0.541667	0.5	1.0	1.0	1.0	0.0	0.0	...	0.0
0.070259	0.174812	0.622445	0.791667	1.0	0.5	0.0	0.0	0.0	0.0	...	1.0
0.464126	0.118363	0.333478	0.583333	0.5	0.0	0.0	0.0	0.0	0.0	...	1.0
0.964275	0.074066	0.394243	0.083333	0.0	1.0	0.0	0.0	0.0	0.0	...	1.0
0.601072	0.304908	0.085215	0.708333	0.5	0.5	0.0	0.0	0.0	0.0	...	0.0
0.532599	0.223019	0.536939	0.791667	1.0	0.5	0.0	0.0	0.0	0.0	...	0.0
0.532599	0.208338	0.112761	0.500000	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0
0.494352	0.036329	0.382358	0.000000	1.0	0.0	1.0	0.0	0.0	0.0	...	0.0
0.740994	0.049082	0.838483	0.583333	0.5	0.0	0.0	0.0	0.0	0.0	...	0.0

```
scaled_features_df_1=pd.DataFrame(x_test_norm,index=x_test.index ,columns=x_test.columns)
```

```
scaled_features_df_1.head(10)
```

Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Breads	Breakfast	Canned	...	Fruits and Vegetables
0.494352	0.296689	0.601126	0.000000	0.5	1.0	0.666667	0.0	0.0	0.0	...	0.0
0.198869	0.287971	0.310560	0.500000	1.0	0.0	0.000000	0.0	1.0	0.0	...	0.0
0.481989	0.232037	0.679829	0.708333	0.5	0.5	0.000000	0.0	0.0	0.0	...	0.0
0.776719	0.119923	0.282957	1.000000	0.5	1.0	0.333333	0.0	0.0	0.0	...	0.0
0.955344	0.146269	0.666525	0.708333	0.5	0.5	0.000000	0.0	0.0	0.0	...	1.0
0.419470	0.234978	0.599581	0.500000	1.0	0.0	0.000000	0.0	0.0	0.0	...	0.0
0.901756	0.127076	0.330790	0.791667	1.0	0.5	0.000000	0.0	0.0	0.0	...	0.0
0.242632	0.094837	0.648254	0.583333	0.5	0.0	0.000000	0.0	0.0	0.0	...	0.0
0.162548	0.094861	0.602536	0.500000	1.0	0.0	0.000000	0.0	0.0	0.0	...	0.0
0.961298	0.282021	0.190805	0.083333	0.0	1.0	0.000000	0.0	0.0	0.0	...	0.0

MODEL BUILDING

One of the most essential and commonly used regression techniques is linear regression. It's one of the most basic regression techniques. The simplicity with which the results may be interpreted is one of its primary merits. $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_r x_r + \varepsilon$. Where Y - Variable to be Predicted X – Variables used for making a prediction β_0, $\beta 1 \ldots \beta$r - Regression Coeffecients ε - Random Error Regardless of how well the model is trained, tested, and validated, there will always be a variation between observed and predicted, which is irreducible error, so we cannot rely entirely on the learning algorithm's predicted results. Data must meet several conditions for a successful linear regression model.

```
[ ]  from sklearn.linear_model import LinearRegression

[ ]  reg=LinearRegression()

[ ]  reg.fit(x_train,y_train)

     LinearRegression()

[ ]  y_pred=reg.predict(x_test)
```

RESULT

```
print("training acc:",reg.score(x_train,y_train))
print("testing acc:",reg.score(x_test,y_test))
```

```
training acc: 0.4340731940341346
testing acc: 0.4348487323084313 5
```

It can be seen that the training and testing accuracy is nearly the same which indicates that the model is good

CONCLUSION

The goal of this study is to use machine learning techniques to forecast future sales of Big Mart based on
previous year's data. With conventional methods failing to assist businesses in increasing revenue, the
application of Machine Learning methodologies proves to be a significant factor in creating company plans, that
take consumer purchasing habits into account. Predicting sales based on a variety of criteria, including past
year's sales, allows firms to develop effective sales plans and enter the competitive market unafraid.

THANK YOU!