```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from xgboost import XGBClassifier
```

```
df=pd.read_csv("/content/drive/MyDrive/datasets/bigmart Dataset/Train.csv")
df
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	I
0	FDA15	9.300	Low Fat	0.016047	Dairy	
1	DRC01	5.920	Regular	0.019278	Soft Drinks	
2	FDN15	17.500	Low Fat	0.016760	Meat	
3	FDX07	19.200	Regular	0.000000	Fruits and Vegetables	
4	NCD19	8.930	Low Fat	0.000000	Household	
...	...	...	...	...	...	
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	
8519	FDS36	8.380	Regular	0.046982	Baking Goods	
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	

```
df.head()
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item
0	FDA15	9.30	Low Fat	0.016047	Dairy	249
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48
2	FDN15	17.50	Low Fat	0.016760	Meat	141

```
df.tail()
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_
8518	FDF22	6.865	Low Fat	0.056783	
8519	FDS36	8.380	Regular	0.046982	B
8520	NC129	10.600	Low Fat	0.035186	Heal

```
df.shape
```

```
(8523, 12)
```

```
df.isna().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
```

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

```
df.isna().sum()
```

```
Item_Identifier               0
Item_Weight                   0
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
```

```
outlet_mode=df.pivot_table(values='Outlet_Size', columns='Outlet_Type', aggfunc=(lambda
print(outlet_mode)
```

```
Outlet_Type Grocery Store Supermarket Type1 Supermarket Type2  \
Outlet_Size         Small             Small            Medium

Outlet_Type Supermarket Type3
Outlet_Size            Medium
```

```
missing_val=df['Outlet_Size'].isnull()
print(missing_val)
```

```
0       False
1       False
2       False
3        True
4       False
        ...
8518    False
8519     True
8520    False
8521    False
8522    False
Name: Outlet_Size, Length: 8523, dtype: bool
```

```
df.loc[missing_val,'Outlet_Size']= df.loc[missing_val,'Outlet_Type'].apply(lambda x: ou
df.isna().sum()
```

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

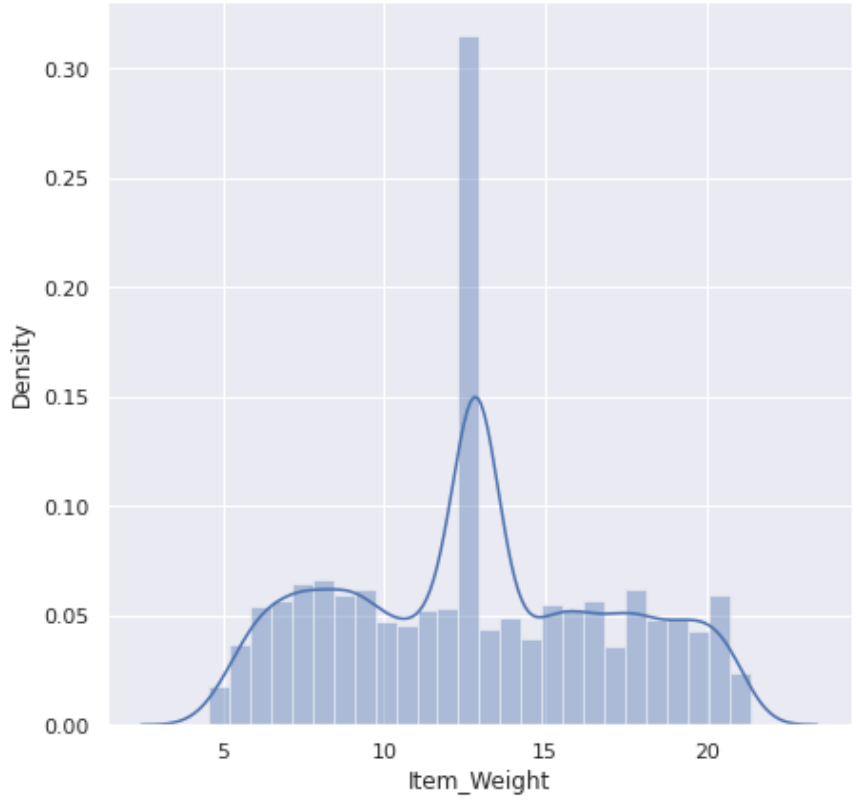
```
df.replace({'Item_Fat_Content':{'low fat':'Low Fat','LF':'Low Fat','reg':'Regular'}}, i
```

```
df.describe()
```

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Ou
count	8523.000000	8523.000000	8523.000000	8523.000000	
mean	12.857645	0.066132	140.992782	1997.831867	
std	4.226124	0.051598	62.275067	8.371760	
min	4.555000	0.000000	31.290000	1985.000000	
25%	9.310000	0.026989	93.826500	1987.000000	
50%	12.857645	0.053931	143.012800	1999.000000	
75%	16.000000	0.094585	185.643700	2004.000000	
max	21.350000	0.328391	266.888400	2009.000000	1

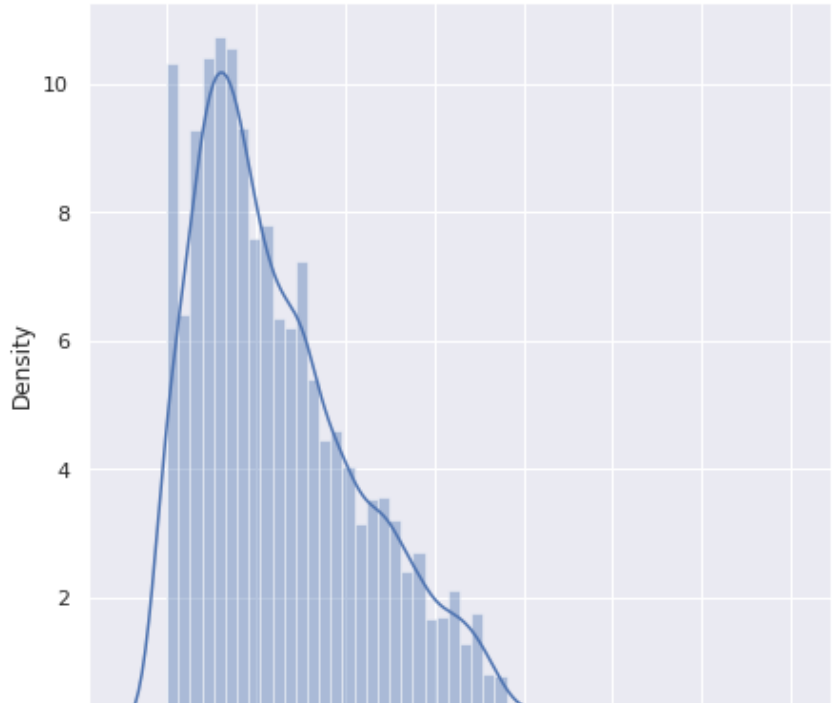
```
sns.set()


plt.figure(figsize=(7,7))
sns.distplot(df['Item_Weight'])
plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarnir
  warnings.warn(msg, FutureWarning)
```



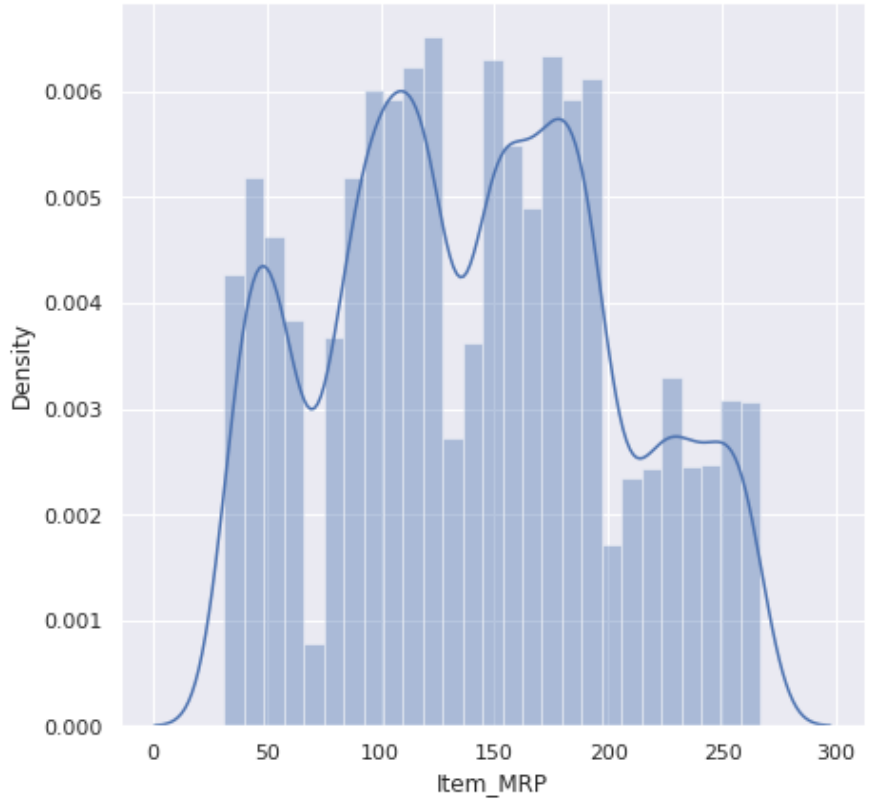

```
plt.figure(figsize=(7,7))
sns.distplot(df['Item_Visibility'])
plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarnir
  warnings.warn(msg, FutureWarning)
```



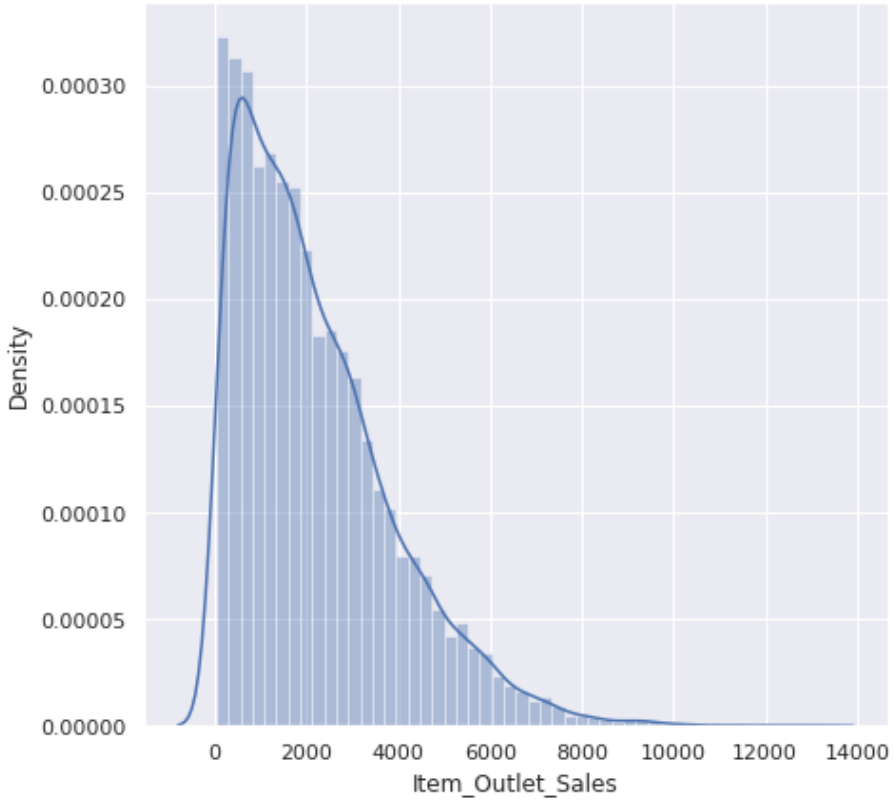

```
plt.figure(figsize=(7,7))
sns.distplot(df['Item_MRP'])
plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: Futur
  warnings.warn(msg, FutureWarning)
```

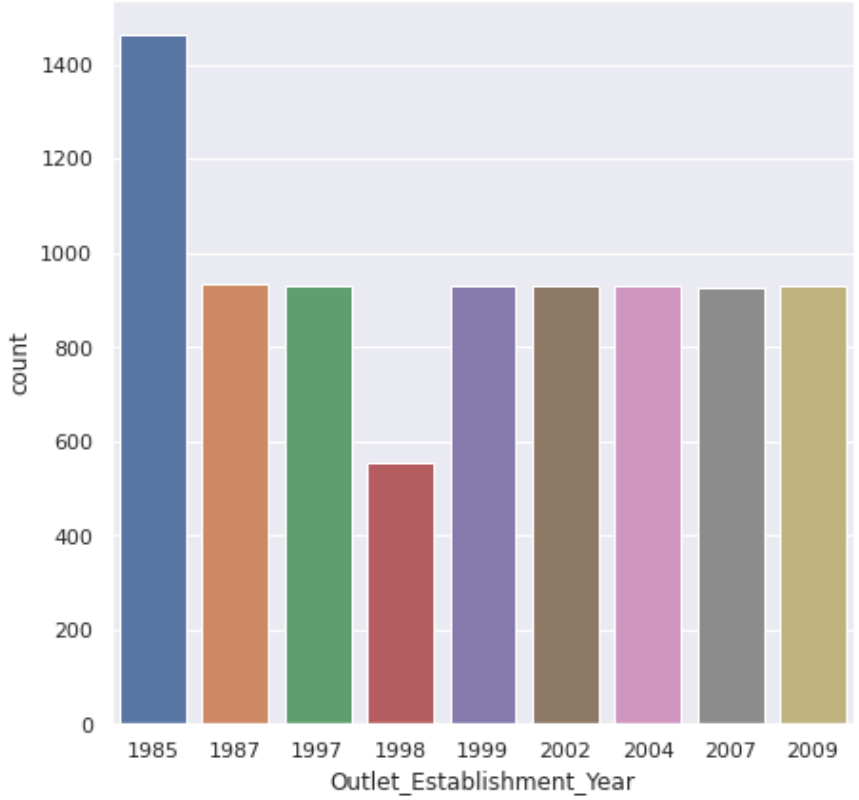



```
plt.figure(figsize=(7,7))
sns.distplot(df['Item_Outlet_Sales'])
plt.show()
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarnir
  warnings.warn(msg, FutureWarning)
```



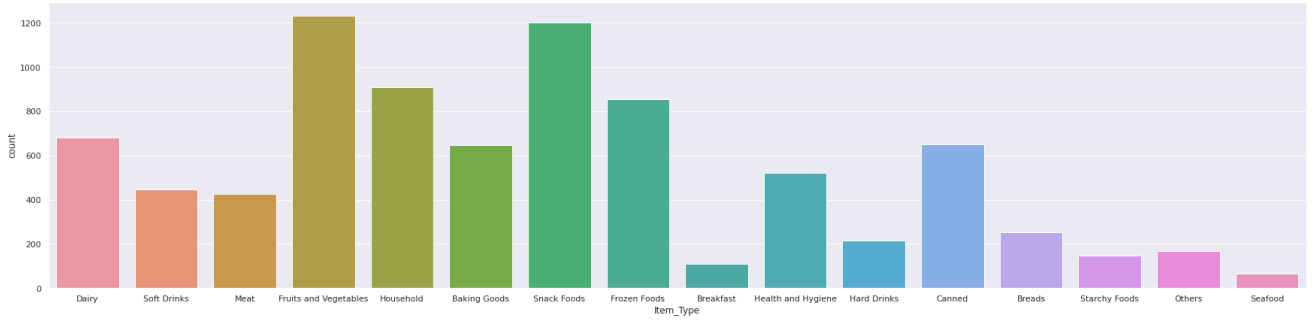

```
plt.figure(figsize=(7,7))
sns.countplot(x='Outlet_Establishment_Year', data=df)
plt.show()
```




```
plt.figure(figsize=(7,7))
sns.countplot(x='Item_Fat_Content', data=df)
```

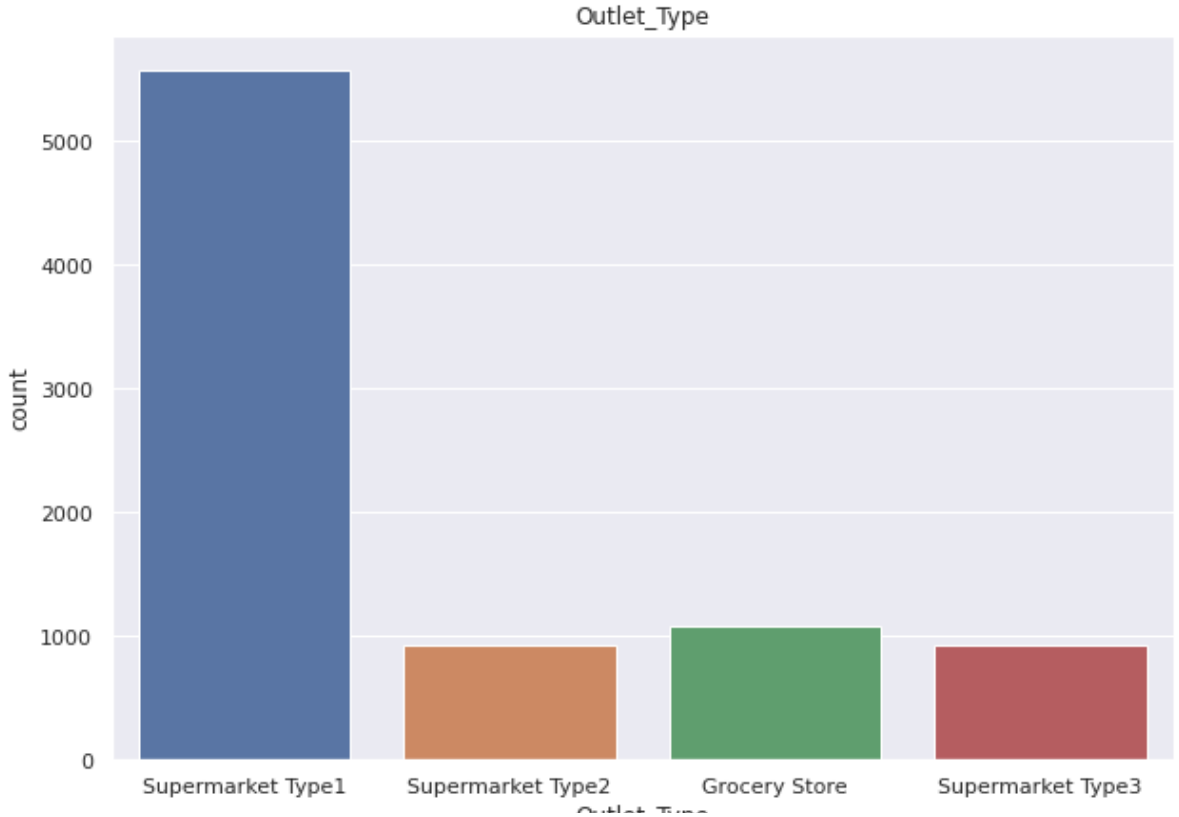
```
plt.show()
```


```
plt.figure(figsize=(30,7))
sns.countplot(x='Item_Type', data=df)
plt.show()
```




```
plt.figure(figsize=(10,7))
sns.countplot(x='Outlet_Type', data=df)
```

```
plt.title("Outlet_Type")
plt.show()
```



LABEL ENCODING

```
df
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	I
0	FDA15	9.300	Low Fat	0.016047	Dairy	
1	DRC01	5.920	Regular	0.019278	Soft Drinks	

```
from sklearn.preprocessing import LabelEncoder
lc = LabelEncoder()
df['Item_Identifier'] = lc.fit_transform(df['Item_Identifier'])
df['Item_Fat_Content'] = lc.fit_transform(df['Item_Fat_Content'])
df['Item_Type'] = lc.fit_transform(df['Item_Type'])
df['Outlet_Identifier'] = lc.fit_transform(df['Outlet_Identifier'])
df['Outlet_Location_Type'] = lc.fit_transform(df['Outlet_Location_Type'])
df['Outlet_Type'] = lc.fit_transform(df['Outlet_Type'])
```

8518	FDF22	6.865	Low Fat	0.056783	Snack	

```
df.drop("Outlet_Size", axis=1, inplace=True)
```

```
df
```

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	I
0	156	9.300	0	0.016047	4	
1	8	5.920	1	0.019278	14	
2	662	17.500	0	0.016760	10	
3	1121	19.200	1	0.000000	6	
4	1297	8.930	0	0.000000	9	
...	...	...	...	...	...	...
8518	370	6.865	0	0.056783	13	
8519	897	8.380	1	0.046982	0	
8520	1357	10.600	0	0.035186	8	
8521	681	7.210	1	0.145221	13	
8522	50	14.800	0	0.044878	14	

8523 rows × 11 columns

```
from sklearn.model_selection import train_test_split
x=df.drop(columns='Item_Outlet_Sales',axis=1)
y=df['Item_Outlet_Sales']
y
```

```
0       3735.1380
1        443.4228
2       2097.2700
3        732.3800
4        994.7052
```

```
          ...
8518    2778.3834
8519     549.2850
8520    1193.1136
8521    1845.5976
8522     765.6700
Name: Item_Outlet_Sales, Length: 8523, dtype: float64
```

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=.30,random_state=2)
```

```
from xgboost.sklearn import XGBRegressor
regressor= XGBRegressor()
```

```
regressor.fit(xtrain,ytrain)
```

```
[13:39:43] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is
XGBRegressor()
```

prediction on Training Data

```
training_data_prediction = regressor.predict(xtrain)
```

```
from sklearn import metrics
r2_train = metrics.r2_score(ytrain,training_data_prediction)
```

```
print(' R squared value =', r2_train)
```

```
 R squared value = 0.634680015092508
```

Prediction On Testing Data

```
test_data_prediction = regressor.predict(xtest)
```

```
r2_test = metrics.r2_score(ytest,test_data_prediction)
```

```
print('·R·squared·value·=',·r2_test)
```

```
 R squared value = 0.6031354712120233
```

✓ 0s completed at 19:23