Industrial Project Report

Submitted in partial fulfillment of the degree of

B.tech in Computer Science and Engineering

By

DRIPTA MAJUMDAR[11900121015]
ROUNAK PRAMANIK[11900121032]
BIPASHA SINHA[11900121047]
CHHATON MODOK[11900121013]
SAIKAT PAUL[11900121010]
ROHAN NAYEK[11900121007]

Second-year student of

SILIGURI INSTITUTE OF TECHNOLOGY

THIS IS SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

AFFILIATED TO

Maulana Abul Kalam Azad University of Technology

Under the supervision of :- Mr. Ripam Kundu

Sikharthy Infotech Pvt. Ltd.

PROJECT ON :- BIGMART SALES PREDICTION USING MACHINE LEARNING

By

DRIPTA MAJUMDAR[11900121015]
ROUNAK PRAMANIK[11900121032]
BIPASHA SINHA[119001210]
CHHATON MODOK[11900121013]
SAIKAT PAUL[11900121010]
ROHAN NAYEK[11900121007]

UNDER THE GUIDANCE OF

Mr. Ripam Kundu

Project Guide

Sikharthy Infotech Pvt. Ltd.

THIS IS SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

B.Tech
IN
COMPUTER SCIENCE AND ENGINEERING

SILIGURI INSTITUTE OF TECHNOLOGY

AFFILIATED TO

Maulana Abdul Kalam Azad University of Technology

<u>Department of Computer Science and Engineering</u>

I hereby forward the documentation prepared under my supervision by **Ripam Kundu Sir** entitled **Siliguri Institute Of Technology** to be accepted as fulfillment of the requirement for the Degree of Bachelor of Technology in Computer Science and Engineering, **Siliguri Institute Of Technology** affiliated to **Maulana Abul Kalam Azad University of Technology** (**MAKAUT**).

Mr.Ripam Kundu
(Software Developer)
Project Guide
Sikharthy Infotech Pvt. Ltd.

HOD

Department Of Electrical Engineering, SIT

Shilpi Ghosal
(Director)
Sikharthy Infotech Pvt. Ltd.

TPO

Certificate of Approval

The foregoing project is hereby approved as a creditable study for the B.Tech in Computer science and Engineering presented in a manner of satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorsed or approved any statement made, opinion expressed or conclusion therein but approve this project only for the purpose for which it is submitted.

Final Examination for
Evaluation of the Project

Signatures of Examiners

ABSTRACT

Nowadays shopping malls and Big Marts keep the track of their sales data for each and every individual item for predicting future demand of the customer and updating 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 stored in 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 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. A retail company wants a model that can predict accurate sales so that it can keep track of customers' future demand and update them in advance of the sale inventory. In this work, we propose a Grid Search Optimization (GSO) technique to

optimize the parameters and select the best tuning hyper parameters, the further ensemble with Xgboost techniques for forecasting the future sales of a retail company such as Big Mart and we found our model produces the better result.

ACKNOWLEDGEMENT

It is a great pleasure for me to acknowledge the assistance and participation of a large number of individuals in this attempt. Our project report has been structured under the valued suggestion, support, and guidance of **Mr. Ripam Kundu**. Under his guidance, we have accomplished the challenging task in a very short time.

Finally, we express our sincere thankfulness to our family members for inspiring me all throughout and always encouraging us.

Group Member's Signature

TABLE OF CONTENTS

INTRODUCTION

Every item is tracked for its shopping centers and BigMarts in order to anticipate a future demand of the customer and also improve the management of its inventory. Big Mart is an immense network of shops virtually all over the world. Trends in Big Mart are very relevant and data scientists evaluate those trends per product and store in order to create potential centres. Using the machine to forecast the transactions of Big Mart helps data scientists to test the various patterns by store and product to achieve the correct results. Many companies rely heavily on the knowledge base and need market patterns to be forecasted. Each shopping center or store endeavors to give the individual and present moment proprietor to draw in more clients relying upon the day, with the goal that the business volume for everything can be evaluated for organization stock administration, logistics and transportation administration, and so forth. To address the issue of deals expectation of things dependent on client's future requests in various BigMarts across different areas diverse Machine Learning algorithms like Linear Regression, Random Forest, Decision Tree, Ridge Regression, XGBoost are utilized for gauging of deals volume. Deals foresee the outcome as deals rely upon the sort of store, populace around the store, a city wherein the store is located. it is possible that it is in an urban zone or country. Population statistics around the store also affect sales, and the capacity of the store and many more things should be considered. Because every business has strong demand, sales forecasts play a significant part in a retail center. A stronger prediction is always helpful in developing and enhancing corporate market strategies, which also help to increase awareness of the market.

WHAT LIBRARIES WE USED

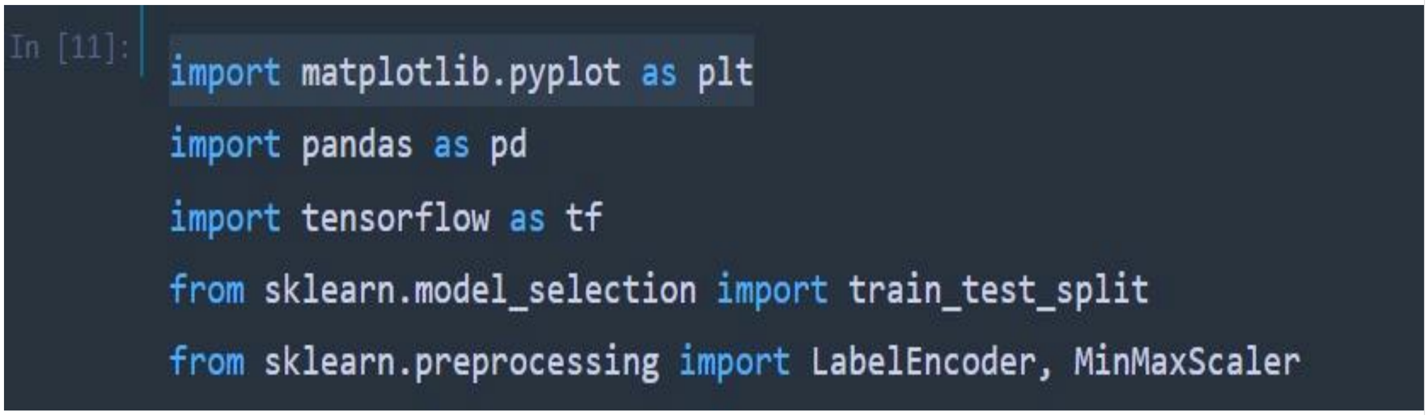
Importing Libraries

The analysis will be done using the following libraries:

- **Pandas**: This library helps to load the data frame in a 2D array format and has multiple functions to perform analysis tasks in one go.
- **Sckitlearn:** Scikit-Learn is a free machine learning library for Python. It supports both supervised and unsupervised machine learning, providing diverse algorithms for classification, regression, clustering, and dimensionality reduction.
- **Sklearn :** Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
- **Tensorflow**: Tensorflow is an open-source end-to-end platform for creating Machine Learning applications. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks.

- **Matplotlib:** Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.
- **Pyplot**: Pyplot is a Matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python and the advantages of being free and and open-source.

To importing all these libraries, we can use the below code:



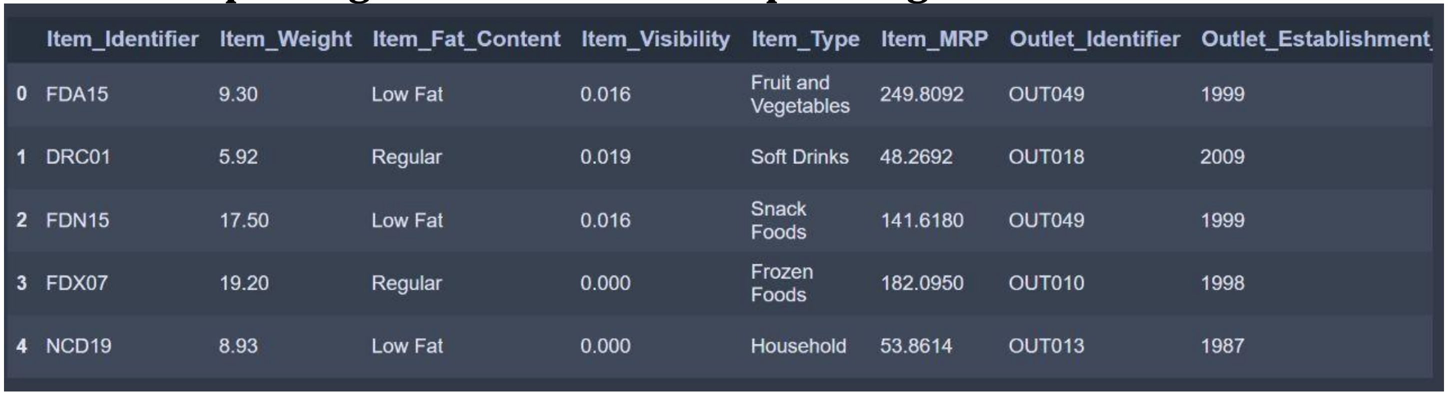
```
In [11]:
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
```

Importing Dataset

After importing all the libraries, you can import the dataset using the pandas library.

```
df = pd.DataFrame({
    'Item_Identifier': ['FDA15', 'DRC01', 'FDN15', 'FDX07', 'NCD19'],
    'Item_Weight': [9.3, 5.92, 17.5, 19.2, 8.93],
    'Item_Fat_Content': ['Low Fat', 'Regular', 'Low Fat', 'Regular', 'Low 
Fat'],
    'Item_Visibility': [0.016, 0.019, 0.016, 0.0, 0.0],
    'Item_Type': ['Fruit and Vegetables', 'Soft Drinks', 'Snack Foods', 'Frozen 
Foods', 'Household'],
    'Item_MRP': [249.8092, 48.2692, 141.618, 182.095, 53.8614],
    'Outlet_Identifier': ['OUT049', 'OUT018', 'OUT049', 'OUT010', 'OUT013'],
    'Outlet_Establishment_Year': [1999, 2009, 1999, 1998, 1987],
    'Outlet_Size': ['Medium', 'Medium', 'Medium', 'Small', 'High'],
    'Outlet_Location_Type': ['Tier 1', 'Tier 3', 'Tier 1', 'Tier 3', 'Tier 3'],
    'Outlet_Type': ['Supermarket Type1', 'Supermarket Type2', 'Supermarket 
Type1', 'Grocery Store', 'Supermarket Type1'],
    'Item_Outlet_Sales': [3735.138, 443.4228, 2097.27, 732.38, 994.7052]
})
```

So after importing the datasets the output we get is : -



	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_
0	FDA15	9.30	Low Fat	0.016	Fruit and Vegetables	249.8092	OUT049	1999
1	DRC01	5.92	Regular	0.019	Soft Drinks	48.2692	OUT018	2009
2	FDN15	17.50	Low Fat	0.016	Snack Foods	141.6180	OUT049	1999
3	FDX07	19.20	Regular	0.000	Frozen Foods	182.0950	OUT010	1998
4	NCD19	8.93	Low Fat	0.000	Household	53.8614	OUT013	1987

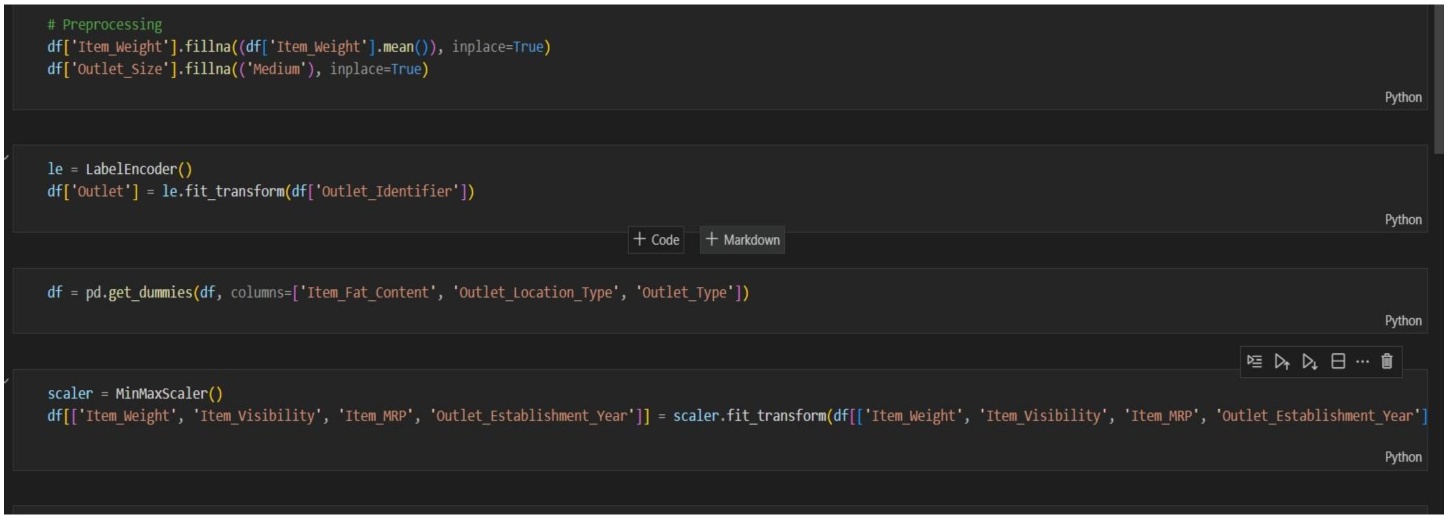
SAVING DATA AS CVV FILE :-

```
# Save the DataFrame as a CSV file

    df.to_csv('bigmart_sales.csv', index=False)
```

DATA PREPROCESSING :-

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data pre-processing task.



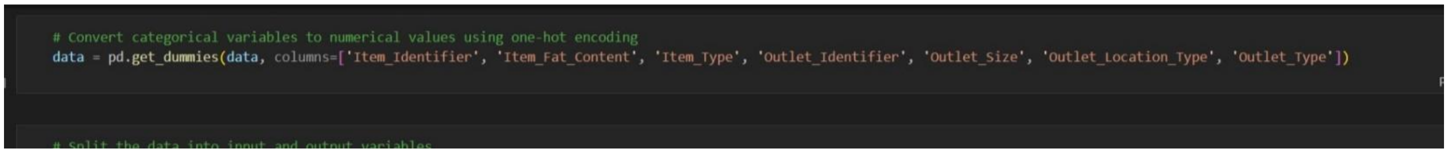
```
# Preprocessing
df['Item_Weight'].fillna((df['Item_Weight'].mean()), inplace=True)
df['Outlet_Size'].fillna(('Medium'), inplace=True)
```

```
le = LabelEncoder()
df['Outlet'] = le.fit_transform(df['Outlet_Identifier'])
```

```
df = pd.get_dummies(df, columns=['Item_Fat_Content', 'Outlet_Location_Type', 'Outlet_Type'])
```

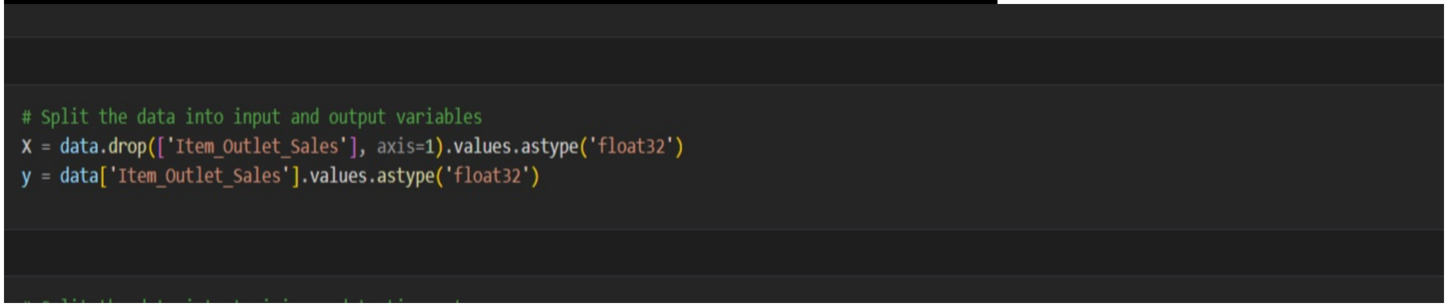
```
scaler = MinMaxScaler()
df[['Item_Weight', 'Item_Visibility', 'Item_MRP', 'Outlet_Establishment_Year']] = scaler.fit_transform(df[['Item_Weight', 'Item_Visibility', 'Item_MRP', 'Outlet_Establishment_Year']
```

Convert categorical variable to numerical values using one hot encoding:-



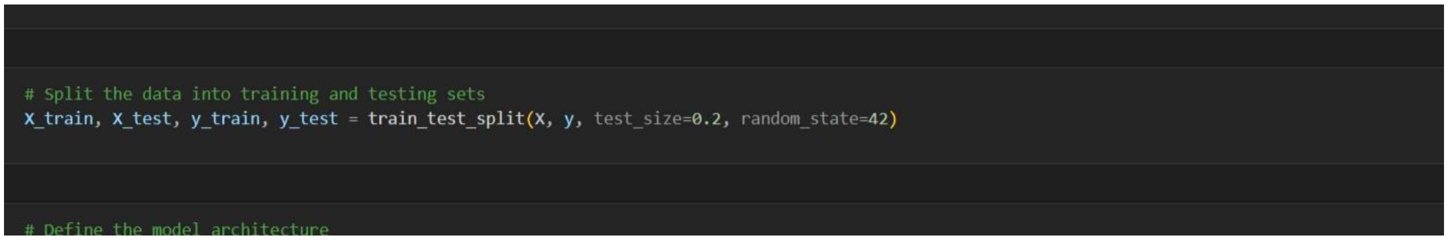
```
# Convert categorical variables to numerical values using one-hot encoding
data = pd.get_dummies(data, columns=['Item_Identifier', 'Item_Fat_Content', 'Item_Type', 'Outlet_Identifier', 'Outlet_Size', 'Outlet_Location_Type', 'Outlet_Type'])
```

Split the data into two input and output variable:-



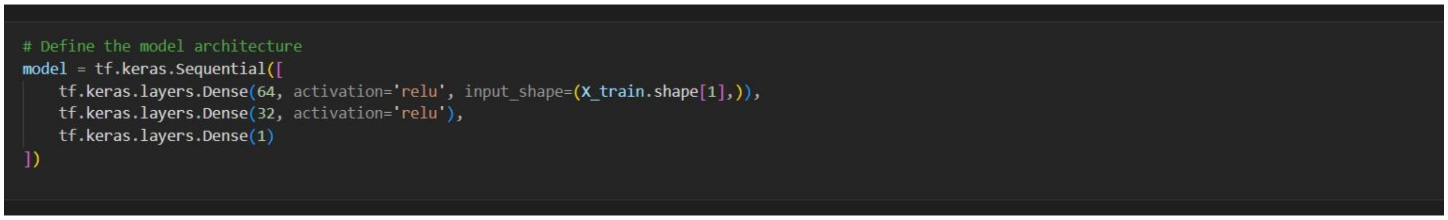
```
# Split the data into input and output variables
X = data.drop(['Item_Outlet_Sales'], axis=1).values.astype('float32')
y = data['Item_Outlet_Sales'].values.astype('float32')
```

Split the data into two input and output variable:-



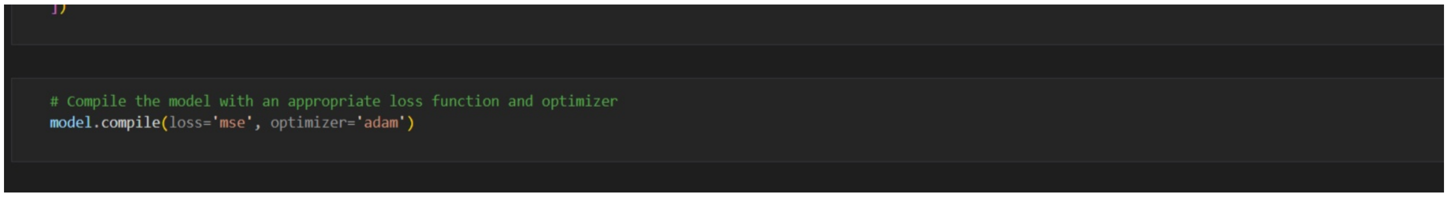
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Define the model architecture :-



```
# Define the model architecture
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(1)
])
```

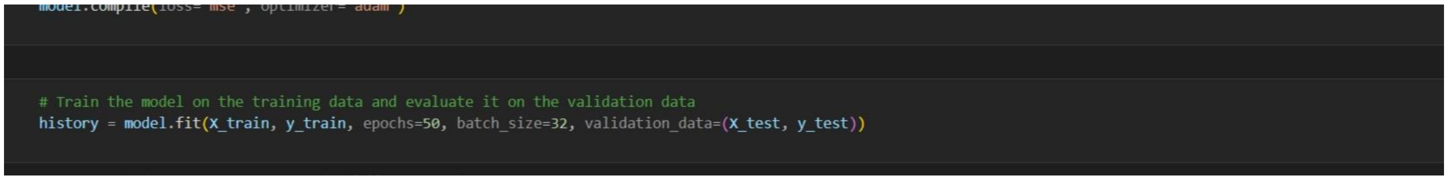
Compile the model with an appropriate loss function and optimizer:-



```
    ])

    # Compile the model with an appropriate loss function and optimizer
    model.compile(loss='mse', optimizer='adam')
```

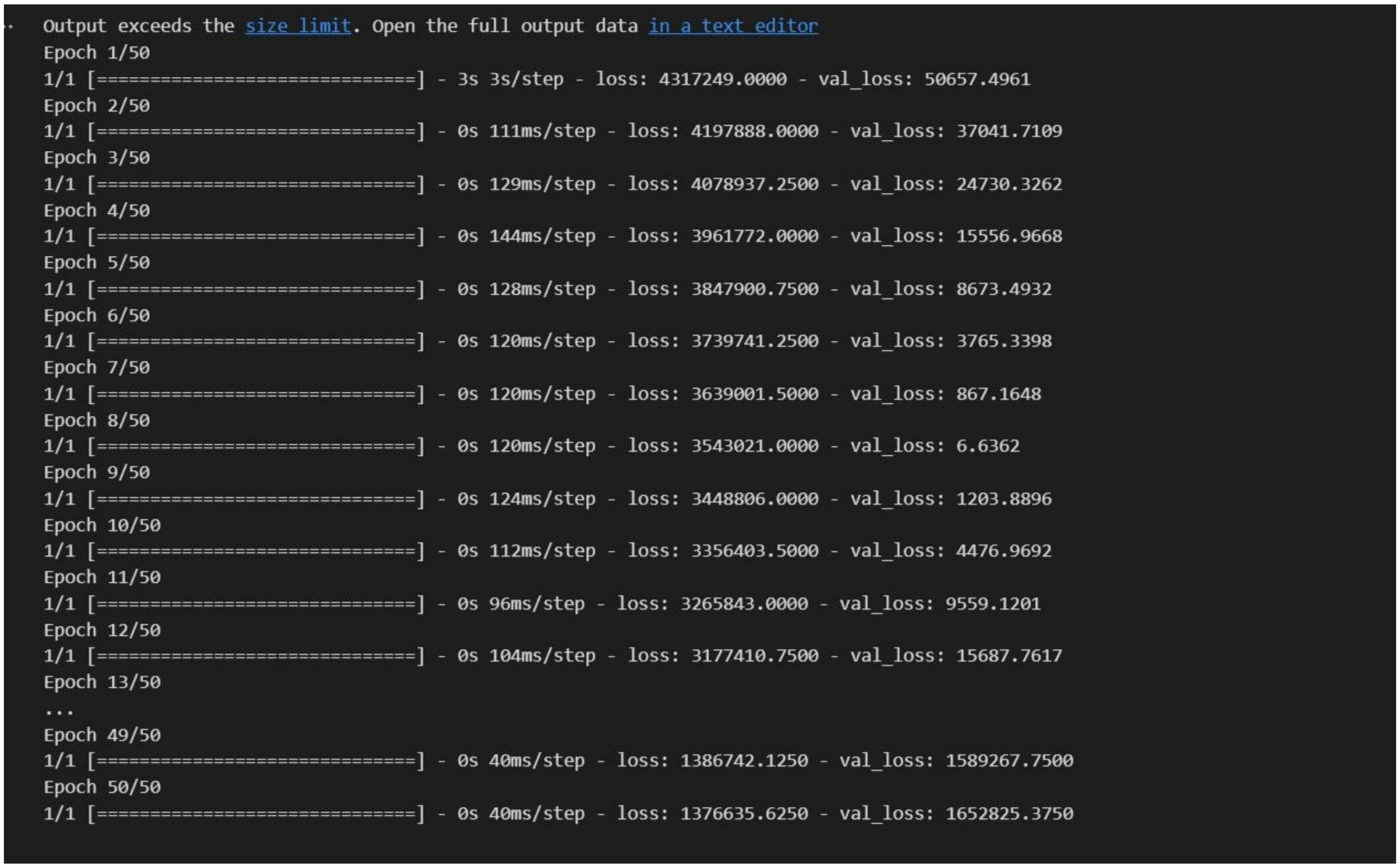
Output :-

Train the model on the training data and evalute it on the validation data:-



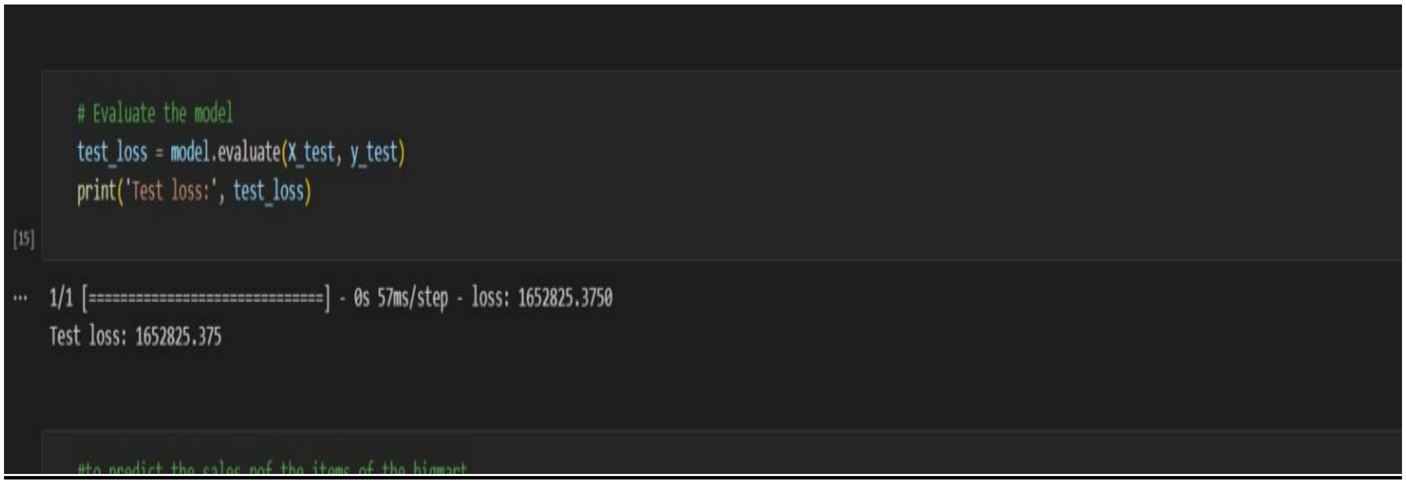
```
  model.compile(loss='mse', optimizer='adam')

  # Train the model on the training data and evaluate it on the validation data
  history = model.fit(X_train, y_train, epochs=50, batch_size=32, validation_data=(X_test, y_test))
```

Output:-



```
Output exceeds the size limit. Open the full output data in a text editor
Epoch 1/50
1/1 [==============================] - 3s 3s/step - loss: 4317249.0000 - val_loss: 50657.4961
Epoch 2/50
1/1 [==============================] - 0s 111ms/step - loss: 4197888.0000 - val_loss: 37041.7109
Epoch 3/50
1/1 [==============================] - 0s 129ms/step - loss: 4078937.2500 - val_loss: 24730.3262
Epoch 4/50
1/1 [==============================] - 0s 144ms/step - loss: 3961772.0000 - val_loss: 15556.9668
Epoch 5/50
1/1 [==============================] - 0s 128ms/step - loss: 3847900.7500 - val_loss: 8673.4932
Epoch 6/50
1/1 [==============================] - 0s 120ms/step - loss: 3739741.2500 - val_loss: 3765.3398
Epoch 7/50
1/1 [==============================] - 0s 120ms/step - loss: 3639001.5000 - val_loss: 867.1648
Epoch 8/50
1/1 [==============================] - 0s 120ms/step - loss: 3543021.0000 - val_loss: 6.6362
Epoch 9/50
1/1 [==============================] - 0s 124ms/step - loss: 3448806.0000 - val_loss: 1203.8896
Epoch 10/50
1/1 [==============================] - 0s 112ms/step - loss: 3356403.5000 - val_loss: 4476.9692
Epoch 11/50
1/1 [==============================] - 0s 96ms/step - loss: 3265843.0000 - val_loss: 9559.1201
Epoch 12/50
1/1 [==============================] - 0s 104ms/step - loss: 3177410.7500 - val_loss: 15687.7617
Epoch 13/50
...
Epoch 49/50
1/1 [==============================] - 0s 40ms/step - loss: 1386742.1250 - val_loss: 1589267.7500
Epoch 50/50
1/1 [==============================] - 0s 40ms/step - loss: 1376635.6250 - val_loss: 1652825.3750
```

Evaluate the Model :-



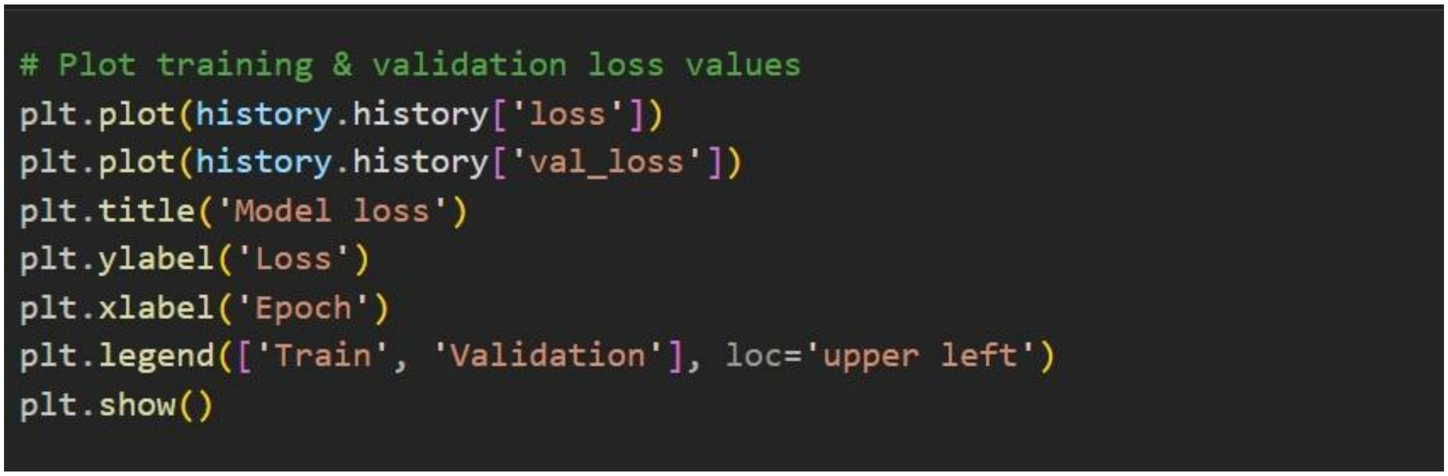
```python
# Evaluate the model
test_loss = model.evaluate(X_test, y_test)
print('Test loss:', test_loss)
```

[15]

```
1/1 [==============================] - 0s 57ms/step - loss: 1652825.3750
Test loss: 1652825.375
```

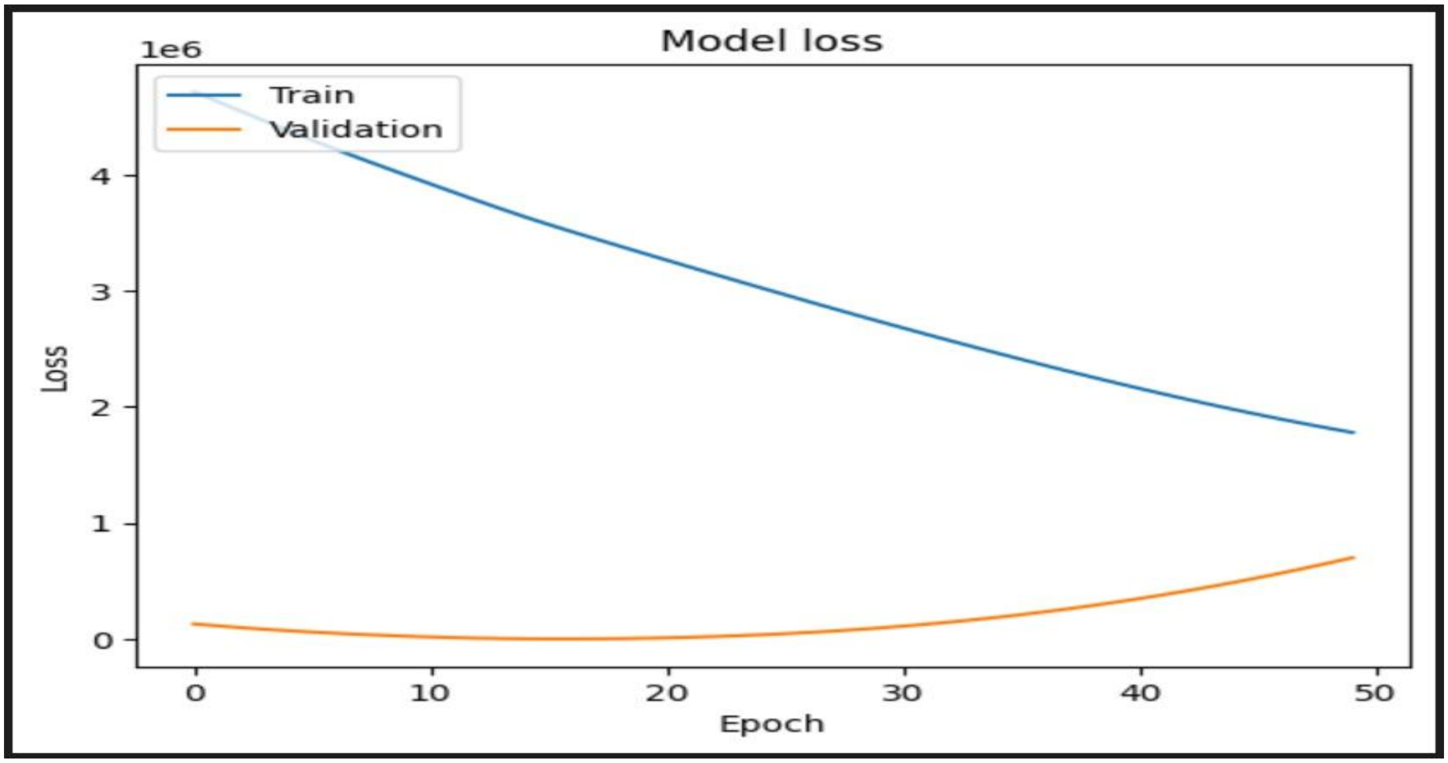
```python
#to predict the sales nof the items of the bigmart
```

Plot training & validation loss values:

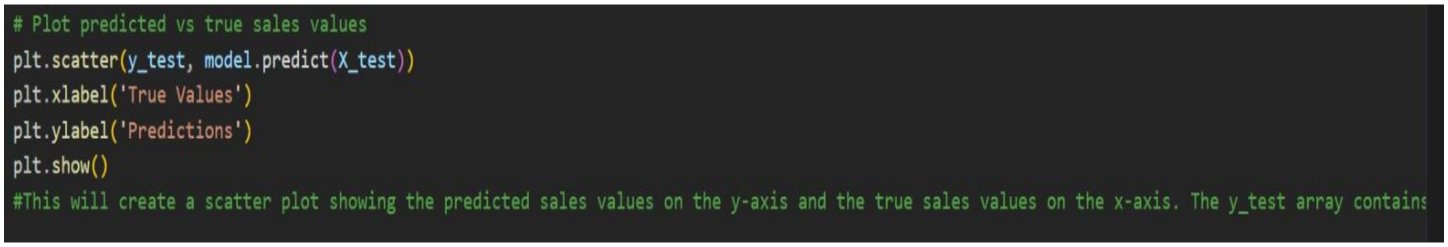


```
# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

Output:

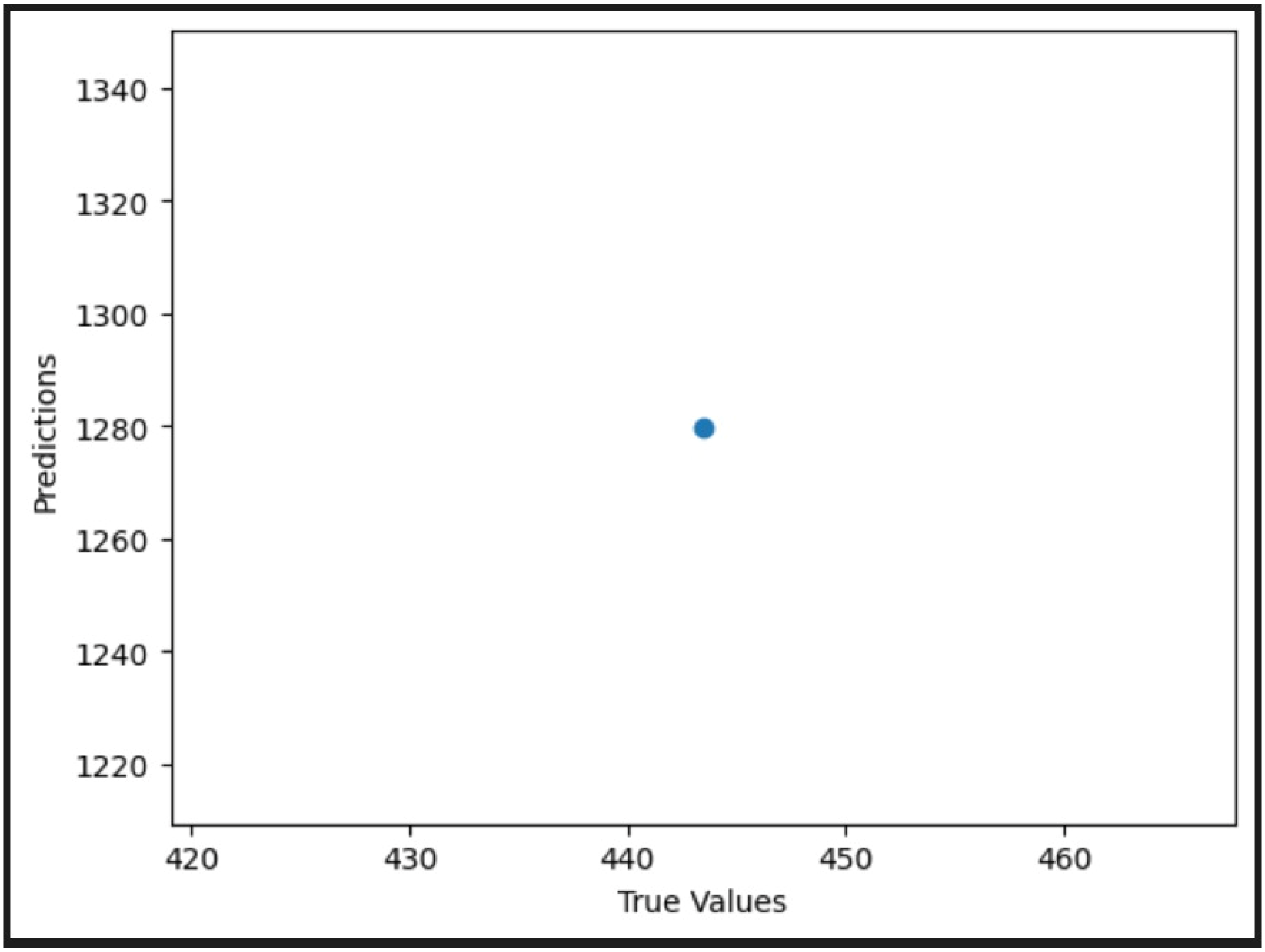


Plot predicted vs true values:

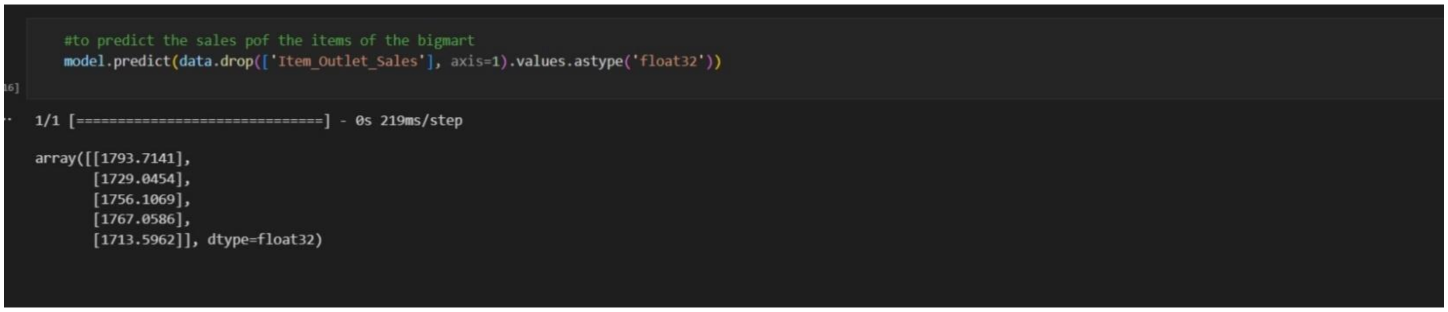


```
# Plot predicted vs true sales values
plt.scatter(y_test, model.predict(X_test))
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.show()
#This will create a scatter plot showing the predicted sales values on the y-axis and the true sales values on the x-axis. The y_test array contains
```

Output:



<u>To predict the sales of the items of the bigmart:-</u>



```
#to predict the sales pof the items of the bigmart
model.predict(data.drop(['Item_Outlet_Sales'], axis=1).values.astype('float32'))
```

```
1/1 [==============================] - 0s 219ms/step

array([[1793.7141],
       [1729.0454],
       [1756.1069],
       [1767.0586],
       [1713.5962]], dtype=float32)
```

<u>FUNCTIONAL REQUIREMENTS OF THE SYSTEM</u>

SOFTWARE:

- *Operating System*
- Windows OS 11

WEB BROWSER:

- Internet Explorer 7
- Google Chrome

CODING LANGUAGE :

- Python

Conclusion :

In this projects, basics of machine learning and the associated data processing and modeling 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 location should follow similar patterns for improved sales.

+

REFERENCE

Analytic Vidya.com

www.geesksforgeesks.com

researchgate.net