

SAVITRIBAI PHULE PUNE UNIVERSITY A MINI

PROJECT REPORT ON

Big Mart Sales Prediction

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

BACHELOR OF ENGINEERING (Computer
Engineering)

BY

Prasann Shimpi	71922822M
Sanket Halake	71922556G
Shivam Dharmshetti	71922510J
Shashank Singh	71922816G

Under The Guidance of

Prof. Amol Dhakne
Academic Year 2020-21



DEPARTMENT OF COMPUTER ENGINEERING Dr. D. Y.
Patil Institute of Engineering, Management & Research
Akurdi, Pune.



Dr. D. Y. Patil Institute of Engineering, Management & Research

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project Entitled
Big Mart Sales Prediction

Submitted by

Prasann Shimpi	71922822M
Sanket Halake	71922556G
Shivam Dharmshetti	71922510J
Shashank Singh	71922816G

is a bonafide work carried out by Students under the supervision of **Prof. Batch Incharge Name** and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering).

Dr. Amol Dhakne

Internal Guide

Dept. of Computer Engg.

Prof. P.P. Shevatekar

H.O.D

Dept. of Computer Engg.

Dr. A. V. Patil

Principal

Dr. D. Y. Patil Institute of Engineering, Management & Research

Sign of Internal Examiner

Sign of External Examiner

PROJECT APPROVAL SHEET

A Project Title

Is successfully completed by

Prasann Shimpi	71922822M
Sanket Halake	71922556G
Shivam Dharmshetti	71922510J
Shashank Singh	71922816G

at

DEPARTMENT OF COMPUTER ENGINEERING

(Dr. D. Y. Patil Institute of Engineering, Management & Research) SAVITRIBAI PHULE

PUNE UNIVERSITY,PUNE ACADEMIC YEAR 2020-2021

Dr. Amol Dhakne

Internal Guide

Dept. of Computer Engg.

Prof. P.P.Shevatekar

H.O.D

Dept. of Computer Engg.

Abstract

Today, malls and major marts track sales data for all items to predict future customer demand and update inventory management. These databases basically contain a large amount of customer data and individual item attributes in the data warehouse. It also detects anomalies and common patterns by mining the data store from the data warehouse. The data obtained can be used to predict future sales volumes using a variety of machine learning techniques for retailers such as Big Mart. In this paper, we proposed a forecasting model that uses the Xgboost method to predict sales for a company like Big Mart, and found that the model performed better than existing models. It also details the comparative analysis of key performance indicators with other models.

Acknowledgments

It gives us great pleasure in presenting the preliminary project report on ‘Project Title’.

I would like to take this opportunity to thank my internal guide **Dr. Amol Dhakne** for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to Prof. P.P. Shevtekar, Head of Computer Engineering Department, Dr. D. Y. Patil Institute of Engineering, Management & Research for his indispensable support, suggestions.

In the end our special thanks to **Dr. Amol Dhakne** for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for Our Project.

Prasann Shimpi	71922822M
Sanket Halake	71922556G
Shivam Dharmshetti	71922510J
Shashank Singh	71922816G
(B.E. Computer Engg.)	

Contents

- 1 1.1 Abstract
- Problem
- 1.2 Goals and
- 2 Technical Keywords
 - 2.1 Area of Project
 - 2.2 Technical Keywords
- 3 Introduction
 - 3.1 Project Idea
 - 3.2 Motivation of the Project
- 4 Problem scope
 - 4.1 Applications:
 - 4.2 Hardware Resources Required
 - 4.3 Software Resources Required
- 5 Project Implementation
 - 5.1 Introduction
 - 5.2 Tools and Technologies Used
 - 5.2.1 Tool
 - 5.2.2 Technology:
 - 5.3 Results
 - 5.3.1 Screen
 - 5.3.2 Outputs
- 6 Conclusion

1.

1.1. Abstract

Today, malls and major marts track sales data for all items to predict future customer demand and update inventory management. These databases basically contain a large amount of customer data and individual item attributes in the data warehouse. It also detects anomalies and common patterns by mining the data store from the data warehouse. The data obtained can be used to predict future sales volumes using a variety of machine learning techniques for retailers such as Big Mart. In this paper, we proposed a forecasting model that uses the Xgboost method to predict sales for a company like Big Mart, and found that the model performed better than existing models. It also details the comparative analysis of key performance indicators with other models.

1.2. Problem Statement:

Use data mining and warehousing, data analysis and machine learning concepts to build a model capable of making accurate predictions related to the sales of a particular item based on various categories.

1.3. Goals and Objectives

To build a model capable of making accurate predictions about the sales of an item based on ten different attributes of the said item.

2. Technical Words

2.1. Area of Project

The project helps to determine the sales of a particular item based on its sales history. The project implements concepts of data mining and warehousing, data analysis and machine learning.

2.2. Technical Keywords

Machine Learning, Sales Forecasting, Random Forest, Regression

3. Introduction

3.1. Project Idea

Due to the rapid growth of global shopping malls and online shopping, competition between various shopping malls and large markets is intensifying every day. All malls or marts seek to offer personalized short-term offers to attract more customers by day. This allows you to predict how much each item will be sold in your organization's inventory management, logistics, and transportation services. The learning algorithm is very sophisticated and provides a method for forecasting or forecasting the future sales demand of a company. It also helps overcome the low availability of computers and storage systems. This article discusses the problem of forecasting or predicting items in terms of future customer demand in different locations and different large markets for products based on previous datasets. Use various machine learning algorithms such as linear regression analysis and random forest to forecast or predict sales volume. Forecasting plays an important role in every shopping complex, as good sales are the life of every business. Always better forecasts help you develop and improve your market business strategy and help you improve your knowledge of the market.

3.2. Motivation of the Project

During these tough times due to the pandemic, the sales of different commodities have varied drastically when compared in a larger time frame. As lockdowns are being lifted and everyone has started to resume their normal routine at a steady pace, the previous sales data is crucial for managing the demand supply requirements of a particular commodity. Thus, a model capable of predicting such sales without human effort whilst saving time is very beneficial for mankind. As Big Mart Sales dataset was available on Kaggle, we have used is to predict sales based on categories like type of item, outlet size, location, weight, visibility, etc. Using this as a initial model, a few more improvements and then some, it will surely help in regulating the demand and supply cycle.

4. Problem Scope

4.1. Applications

In today's digitally connected world, every shopping center wants to know in advance customer requirements to avoid a shortage of items for sale throughout the year. Every day, businesses and shopping malls more accurately forecast product sales demand and user demand. Extensive research in this area is being conducted at the enterprise level to create accurate sales forecasts. Big Marts wants a more accurate forecasting algorithm so that the company doesn't suffer losses because the company's profits are directly proportional to the exact forecast.

4.2. Hardware Resources

8 GB RAM, i5 10th Gen Processor Laptop/PC.

4.3. Software Resources

Python, Anaconda, Jupyter Notebook

5. Project Implementation

5.1. Introduction

Due to the rapid growth of global shopping malls and online shopping, competition between various shopping malls and large markets is intensifying every day. All malls or marts seek to offer personalized short-term offers to attract more customers by day. This allows you to predict how much each item will be sold in your organization's inventory management, logistics, and transportation services. The learning algorithm is very sophisticated and provides a method for forecasting or forecasting the future sales demand of a company. It also helps overcome the low availability of computers and storage systems. This article discusses the problem of forecasting or predicting items in terms of future customer demand in different locations and different large markets for products based on previous datasets. Use various machine learning algorithms such as linear regression analysis and random forest to forecast or predict sales volume. Forecasting plays an important role in every shopping complex, as good sales are the life of every business. Always better forecasts help you develop and improve your market business strategy and help you improve your knowledge of the market. A standard forecasting survey thoroughly analyzes previously encountered situations and situations, draws conclusions about customer acquisition, resource shortages, and strengths, and then sets a budget and marketing plan one year ahead. Useful for. In other words, forecasts are based on past available resources.

5.2. Tools and Technologies

5.2.1. Tool

Python - Anaconda - Jupyter Notebook

5.2.2. Technology

Python Libraries

- Scikit-learn
- Pandas
- numpy
- Matplotlib
- Ddate
- klib

5.3. Results

5.3.1. Screenshots

```
In [4]: df_train.head()
Out[4]:
   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

In [5]: df_train.shape
Out[5]: (8523, 12)

In [6]: df_train.isnull().sum()
Out[6]:
   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
   click to expand output; double click to hide output
   Outlet_Type            0
   Item_Outlet_Sales        0
   Item_Outlet_Area        0
```

Fig 1: Dataframe attributes

Fig 2 and 3: Eliminating Null Values

In [7]: df_train.describe()

Out[7]:

	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

Fill Item_Weight null values with mean of the column

In [8]: df_train['Item_Weight'].fillna(df_train['Item_Weight'].mean(), inplace=True)
df_test['Item_Weight'].fillna(df_test['Item_Weight'].mean(), inplace=True)

In [9]: df_train.isnull().sum()

Out[9]:

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

Fill Outlet_Size null values with mode of the column as it is non-numerical

In [10]: df_train['Outlet_Size'].value_counts()

Out[10]:

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

In [11]: df_train['Outlet_Size'].fillna(df_train['Outlet_Size'].mode()[0], inplace=True)
df_test['Outlet_Size'].fillna(df_test['Outlet_Size'].mode()[0], inplace=True)

In [12]: df_train.isnull().sum()

Out[12]:

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	

In [13]: df_test.isnull().sum()

Out[13]:

Item_Identifier	0
Item_Weight	0
Item_Fat_Content	0
Item_Visibility	0
Item_Type	0

Safari File Edit View History Bookmarks Window Help

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Selecting Features based on general requirements

```
In [14]: df_train.drop(['Item_Identifier','Outlet_Identifier'], axis=1, inplace=True)
df_test.drop(['Item_Identifier','Outlet_Identifier'], axis=1, inplace=True)

In [15]: df_train.head()
```

```
Out[15]:
   Item_Weight Item_Fat_Content Item_Visibility Item_Type Item_MRP Outlet_Establishment_Year Outlet_Size Outlet_Location_Type Outlet_Type Item_Outlet
0      9.30     Low Fat       0.016047    Dairy  249.8092           1999      Medium      Tier 1 Supermarket Type1      375
1      5.92     Regular      0.019278  Soft Drinks  48.2692           2009      Medium      Tier 3 Supermarket Type2       44
2     17.50     Low Fat       0.016760     Meat  141.6180           1999      Medium      Tier 1 Supermarket Type1      205
3     19.20     Regular      0.000000  Fruits and Vegetables  182.0950           1998      Medium      Tier 3 Grocery Store       75
4      8.93     Low Fat       0.000000  Household  53.8614           1987      High       Tier 3 Supermarket Type1      95
```

Exploratory Data Analysis

using Dtale

```
In [16]: import dtale
In [17]: dtale.show(df_train)
```

Safari File Edit View History Bookmarks Window Help

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Exploratory Data Analysis

using Dtale

```
In [15]: import dtale
In [16]: dtale.show(df_train)
```

Index	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet
0	9.30	Low Fat	0.02	Dairy	249.81	1999	Medium	Tier 1	Supermarket Type1	375
1	5.92	Regular	0.02	Soft Drinks	48.27	2009	Medium	Tier 3	Supermarket Type2	44
2	17.50	Low Fat	0.02	Meat	141.62	1999	Medium	Tier 1	Supermarket Type1	205
3	19.20	Regular	0.00	Fruits and Vegetables	182.10	1998	Medium	Tier 3	Grocery Store	75
4	8.93	Low Fat	0.00	Household	53.86	1987	High	Tier 3	Supermarket Type1	95
5	10.40	Regular	0.00	Baking Goods	51.40	2009	Medium			
6	13.65	Regular	0.01	Snack Foods	57.66	1987	High			
7	12.86	Low Fat	0.13	Snack Foods	107.76	1985	Medium			
8	16.20	Regular	0.02	Frozen Foods	96.97	2002	Medium			
9	19.20	Regular	0.09	Frozen Foods	187.82	2007	Medium			
10	11.80	Low Fat	0.00	Fruits and Vegetables	45.54	1999	Medium			
11	18.50	Regular	0.05	Dairy	144.11	1997	Small			
12	15.10	Regular	0.10	Fruits and Vegetables	145.48	1999	Medium			
13	17.60	Regular	0.05	Snack Foods	119.68	1997	Small			
14	16.35	Low Fat	0.07	Fruits and Vegetables	196.44	1987	High			
15	9.00	Regular	0.07	Breakfast	56.36	1997	Small			

Fig 4 and 5: Data exploration and Analysis

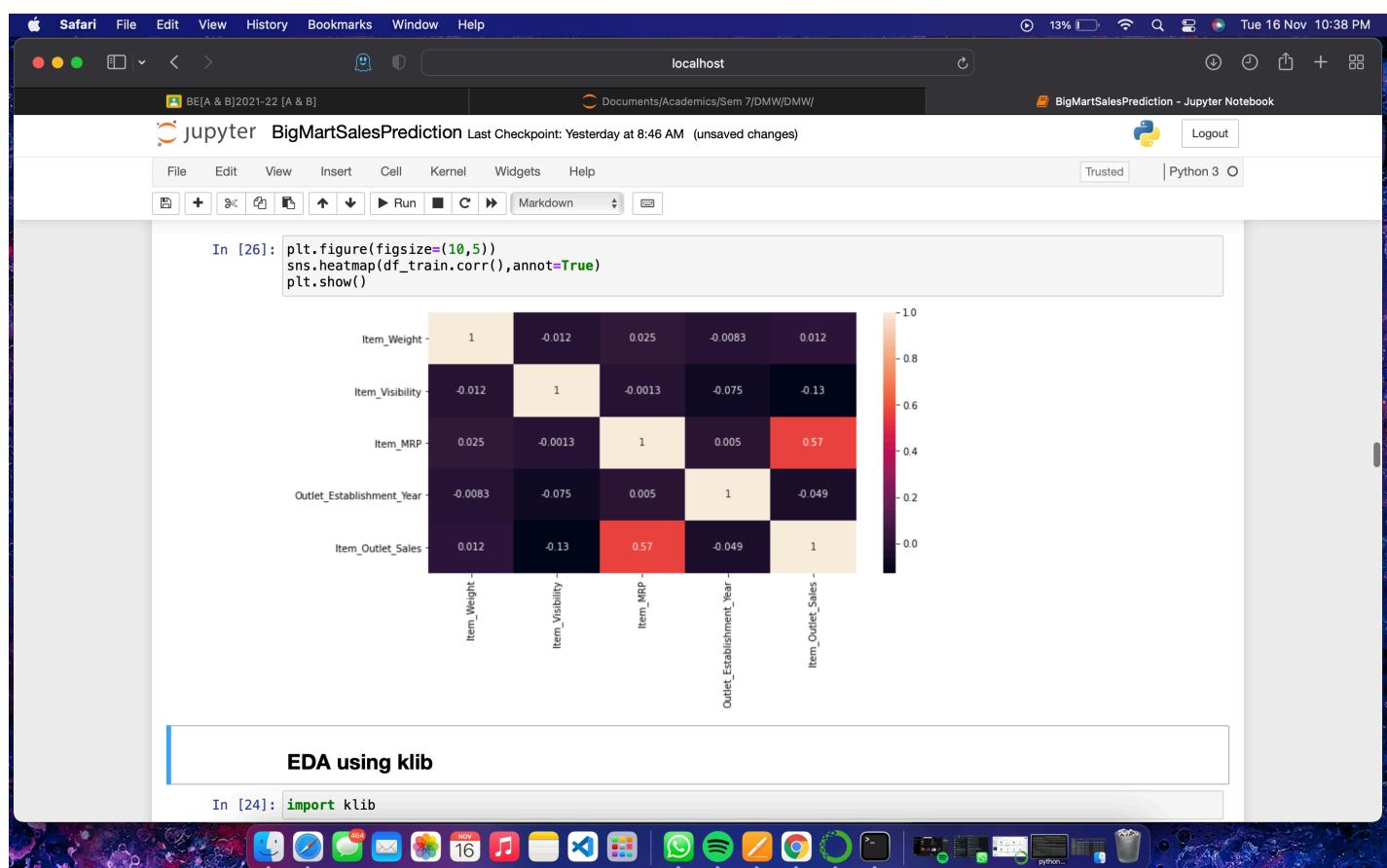
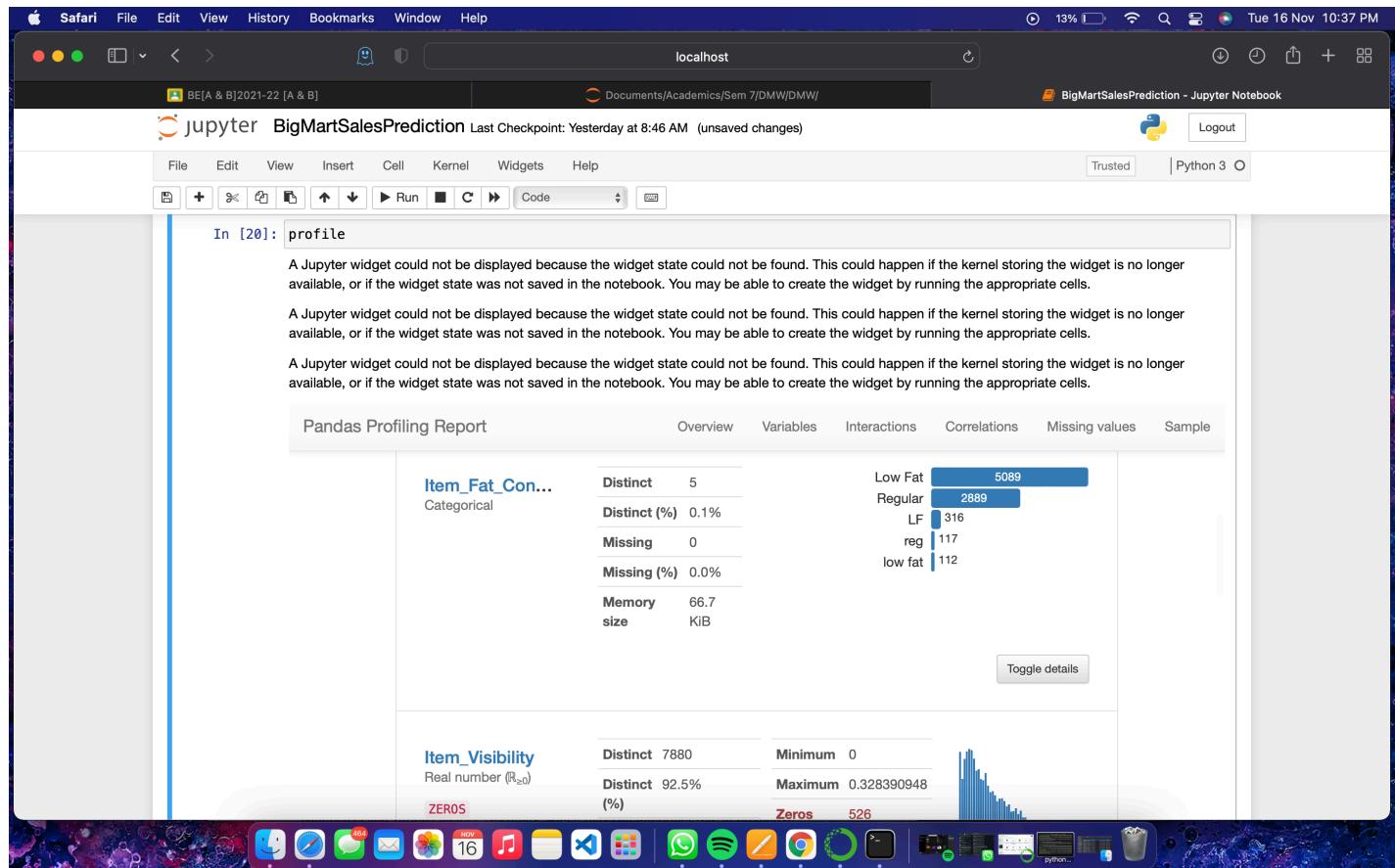


Fig 6 and 7: Data Visualisation

Safari File Edit View History Bookmarks Window Help 13% Tue 16 Nov 10:38 PM

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Data cleaning

In [27]: `klib.data_cleaning(df_train)`

```
Shape of cleaned data: (8523, 10) Remaining NAs: 0

Changes:
Dropped rows: 0
  of which 0 duplicates. (Rows: [])
Dropped columns: 0
  of which 0 single valued. Columns: []
Dropped missing values: 0
Reduced memory by at least: 0.43 MB (-66.15%)
```

Out[27]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_year	outlet_size	outlet_location_type	outlet_type	item_outlet
0	9.3	Low Fat	0.016047	Dairy	249.809204	1999	Medium	Tier 1	Supermarket Type1	3735.1
1	5.92	Regular	0.019278	Soft Drinks	48.269199	2009	Medium	Tier 3	Supermarket Type2	443.4
2	17.5	Low Fat	0.01676	Meat	141.617996	1999	Medium	Tier 1	Supermarket Type1	2097
3	19.200001	Regular	0.0	Fruits and Vegetables	182.095001	1998	Medium	Tier 3	Grocery Store	732.3
4	8.93	Low Fat	0.0	Household	53.861401	1987	High	Tier 3	Supermarket Type1	99
...
8518	6.865	Low Fat	0.056783	Snack Foods	214.521805	1987	High	Tier 3	Supermarket Type1	2778.3
8519	8.38	Regular	0.046982	Baking Goods	108.156998	2002	Medium	Tier 2	Supermarket Type1	549.2

Safari File Edit View History Bookmarks Window Help 13% Tue 16 Nov 10:38 PM

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Preprocessing before Modelling

1. Label Encoding

In [32]: `from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()`

In [33]: `df_train['item_fat_content']= le.fit_transform(df_train['item_fat_content'])
df_train['item_type']= le.fit_transform(df_train['item_type'])
df_train['outlet_size']= le.fit_transform(df_train['outlet_size'])
df_train['outlet_location_type']= le.fit_transform(df_train['outlet_location_type'])
df_train['outlet_type']= le.fit_transform(df_train['outlet_type'])`

In [34]: `df_train`

Out[34]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_year	outlet_size	outlet_location_type	outlet_type	item_outlet
0	9.3	1	0.016047	4	249.809204	1999	1	0	1	3735.13
1	5.92	2	0.019278	14	48.269199	2009	1	2	2	443.42
2	17.5	1	0.01676	10	141.617996	1999	1	0	1	2097.2
3	19.200001	2	0.0	6	182.095001	1998	1	2	0	732.38
4	8.93	1	0.0	9	53.861401	1987	0	2	1	994.
...
8518	6.865	1	0.056783	13	214.521805	1987	0	2	1	2778.38
8519	8.38	2	0.046982	0	108.156998	2002	1	1	1	549.28
8520	10.6	1	0.035186	8	85.122398	2004	2	1	1	1193.11

Fig 8 and 9 : Data Cleaning and Label Encoding

Safari File Edit View History Bookmarks Window Help

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3 Logout

In [35]: `X=df_train.drop('item_outlet_sales',axis=1)`
`Y=df_train['item_outlet_sales']`

In [36]: `from sklearn.model_selection import train_test_split`
`X_train, X_test, Y_train, Y_test = train_test_split(X,Y, random_state=101, test_size=0.2)`

2. Splitting data into train and test

In [37]: `X.describe()`

Out[37]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_year	outlet_size	outlet_location_type	outlet_type
count	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857646	1.369354	0.066132	7.226681	140.992770	1997.831867	1.170832	1.112871	1.201220
std	4.226124	0.644810	0.051598	4.209990	62.275066	8.371760	0.600327	0.812757	0.796459
min	4.555000	0.000000	0.000000	0.000000	31.290001	1985.000000	0.000000	0.000000	0.000000
25%	9.310000	1.000000	0.026989	4.000000	93.826500	1987.000000	1.000000	0.000000	1.000000
50%	12.857645	1.000000	0.053931	6.000000	143.012802	1999.000000	1.000000	1.000000	1.000000
75%	16.000000	2.000000	0.094585	10.000000	185.643707	2004.000000	2.000000	2.000000	1.000000
max	21.350000	4.000000	0.328391	15.000000	266.888397	2009.000000	2.000000	2.000000	3.000000

In [38]: `from sklearn.preprocessing import StandardScaler`
`sc=StandardScaler()`

Safari File Edit View History Bookmarks Window Help

localhost

BE[A & B]2021-22 [A & B] Documents/Academics/Sem 7/DMW/DMW/ BigMartSalesPrediction - Jupyter Notebook

jupyter BigMartSalesPrediction Last Checkpoint: Yesterday at 8:46 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3 Logout

Model Building

In [47]: `from sklearn.linear_model import LinearRegression`
`lr=LinearRegression()`

In [48]: `lr.fit(X_train_std,Y_train)`

Out[48]: `LinearRegression()`

In [49]: `X_test.head()`

Out[49]:

	item_weight	item_fat_content	item_visibility	item_type	item_mrp	outlet_establishment_year	outlet_size	outlet_location_type	outlet_type
8179	11.0	1	0.055163	8	100.3358	2009	1	2	2
8355	18.0	1	0.038979	13	148.6418	1987	0	2	1
3411	7.72	2	0.074731	1	77.598602	1997	2	0	1
7089	20.700001	1	0.049035	6	39.9506	2007	1	1	1
6954	7.55	1	0.027225	3	152.934006	2002	1	1	1

In [50]: `Y_pred_lr=lr.predict(X_test_std)`

In [51]: `from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error`

In [52]: `print(r2_score(Y_test,Y_pred_lr))`
`print(mean_absolute_error(Y_test,Y_pred_lr))`
`print(np.sqrt(mean_squared_error(Y_test,Y_pred_lr)))`

Fig 10 and 11: Data Preparation and Model Building

The screenshot shows a Jupyter Notebook interface running on a Mac OS X desktop. The title bar indicates the notebook is titled "BigMartSalesPrediction" and the last checkpoint was "Yesterday at 8:46 AM (autosaved)". The Python version is listed as "Python 3". The code in cell [58] is for hyperparameter tuning of a Random ForestRegressor. It defines parameters like n_estimators, max_depth, and min_samples_leaf, sets up a GridSearchCV, and prints results. The output shows fitting 2 folds for 3 candidates, totaling 6 fits, with the best score being 0.549 using 1000 estimators.

```

In [58]: from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV

# define models and parameters
model = RandomForestRegressor()
n_estimators = [10, 100, 1000]
max_depth=range(1,31)
min_samples_leaf=np.linspace(0.1, 1.0)
max_features=["auto", "sqrt", "log2"]
min_samples_split=np.linspace(0.1, 1.0, 10)

# define grid search
grid = dict(n_estimators=n_estimators)

#cv = RepeatedStratifiedKFold(n_splits=5, n_repeats=3, random_state=101)
grid_search_forest = GridSearchCV(estimator=model, param_grid=grid, n_jobs=-1,
scoring='r2',error_score=0,verbose=2, cv=2)
grid_search_forest.fit(X_train_std, Y_train)

# summarize results
print("Best: {grid_search_forest.best_score_: .3f} using {grid_search_forest.best_params_}")
means = grid_search_forest.cv_results_['mean_test_score']
stds = grid_search_forest.cv_results_['std_test_score']
params = grid_search_forest.cv_results_['params']

for mean, stdev, param in zip(means, stds, params):
    print(f"mean: {mean:.3f} ({stdev:.3f}) with: {param}")

Fitting 2 folds for each of 3 candidates, totalling 6 fits
Best: 0.549 using {'n_estimators': 1000}
0.512 (0.012) with: {'n_estimators': 10}
0.517 (0.005) with: {'n_estimators': 1000}

```

The screenshot shows a Jupyter Notebook interface running on a Mac OS X desktop. The title bar indicates the notebook is titled "BigMartSalesPrediction" and the last checkpoint was "Yesterday at 8:46 AM (autosaved)". The Python version is listed as "Python 3". The code in cell [59] retrieves the best parameters from the GridSearchCV object. Subsequent cells predict values for the test set and calculate the R-squared score. Cell [63] uses joblib to dump the trained model to a file named "random_forest_grid.sav". Cell [64] loads the model back into memory.

```

In [58]: grid_search_forest.best_params_
Out[58]: {'n_estimators': 1000}

In [59]: grid_search_forest.best_params_
Out[59]: {'n_estimators': 1000}

In [60]: grid_search_forest.best_score_
Out[60]: 0.5491788919542162

In [61]: Y_pred_rf_grid=grid_search_forest.predict(X_test_std)

In [62]: r2_score(Y_test,Y_pred_rf_grid)
Out[62]: 0.548183985978812

Saving the model

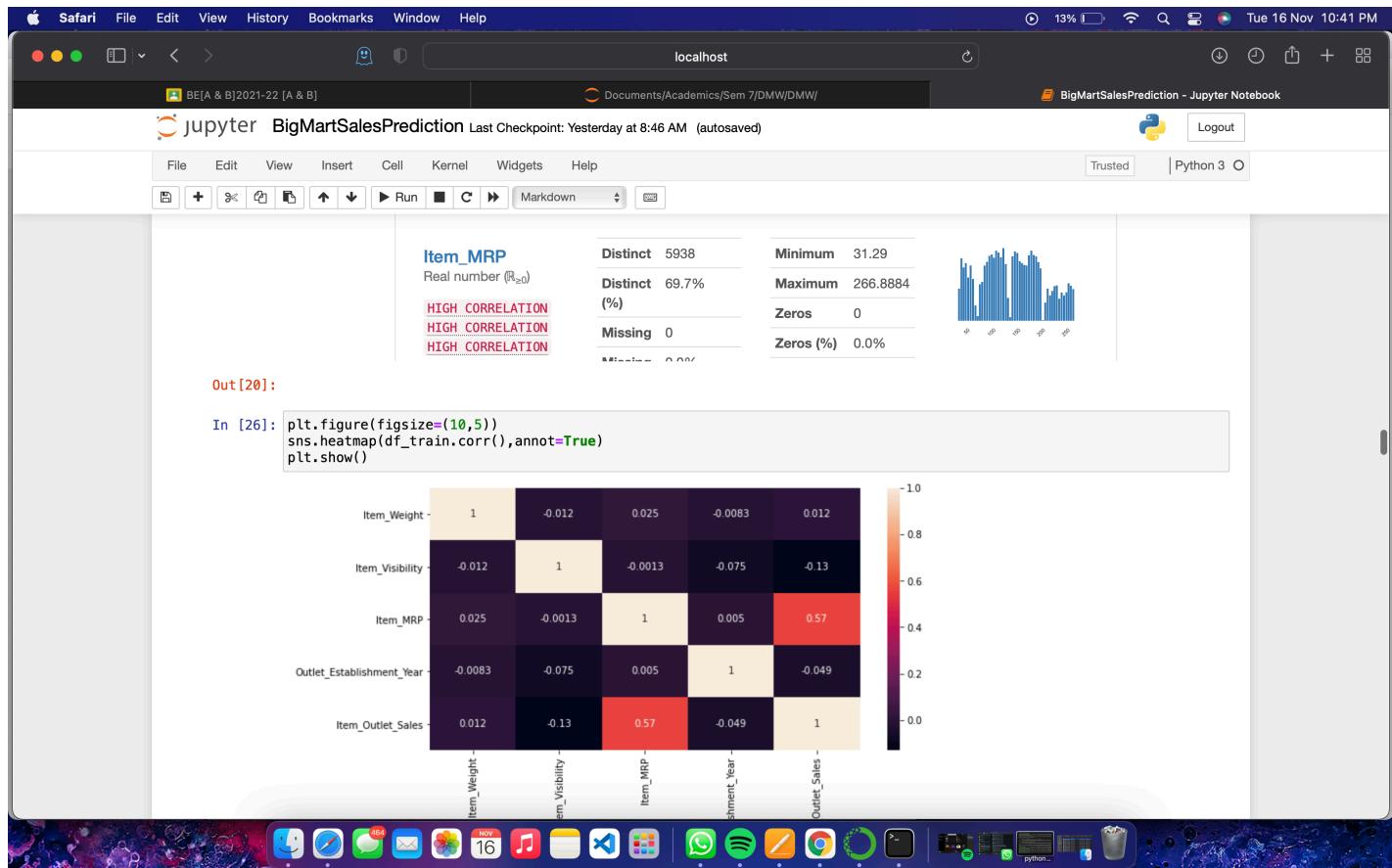
In [63]: joblib.dump(grid_search_forest,r'random_forest_grid.sav')
Out[63]: ['random_forest_grid.sav']

In [64]: model=joblib.load(r'random_forest_grid.sav')

```

Fig 12 and 13: Hyperparameter tuning and Model Dump

5.3.2.Outputs



The screenshot shows a web application titled "Big Mart Sales Prediction" running in a browser window. The URL is "127.0.0.1". The form consists of several input fields:

- Enter Item Weight: 120
- Enter Item Visibility: Regular
- Enter Item MRP: 0.23
- Enter Item MRP: 200
- Outlet Establishment Year (YYYY): 1998
- Enter Outlet Type: Medium
- Enter Outlet Type: Tier 2
- Enter Outlet Type: Supermarket Type2

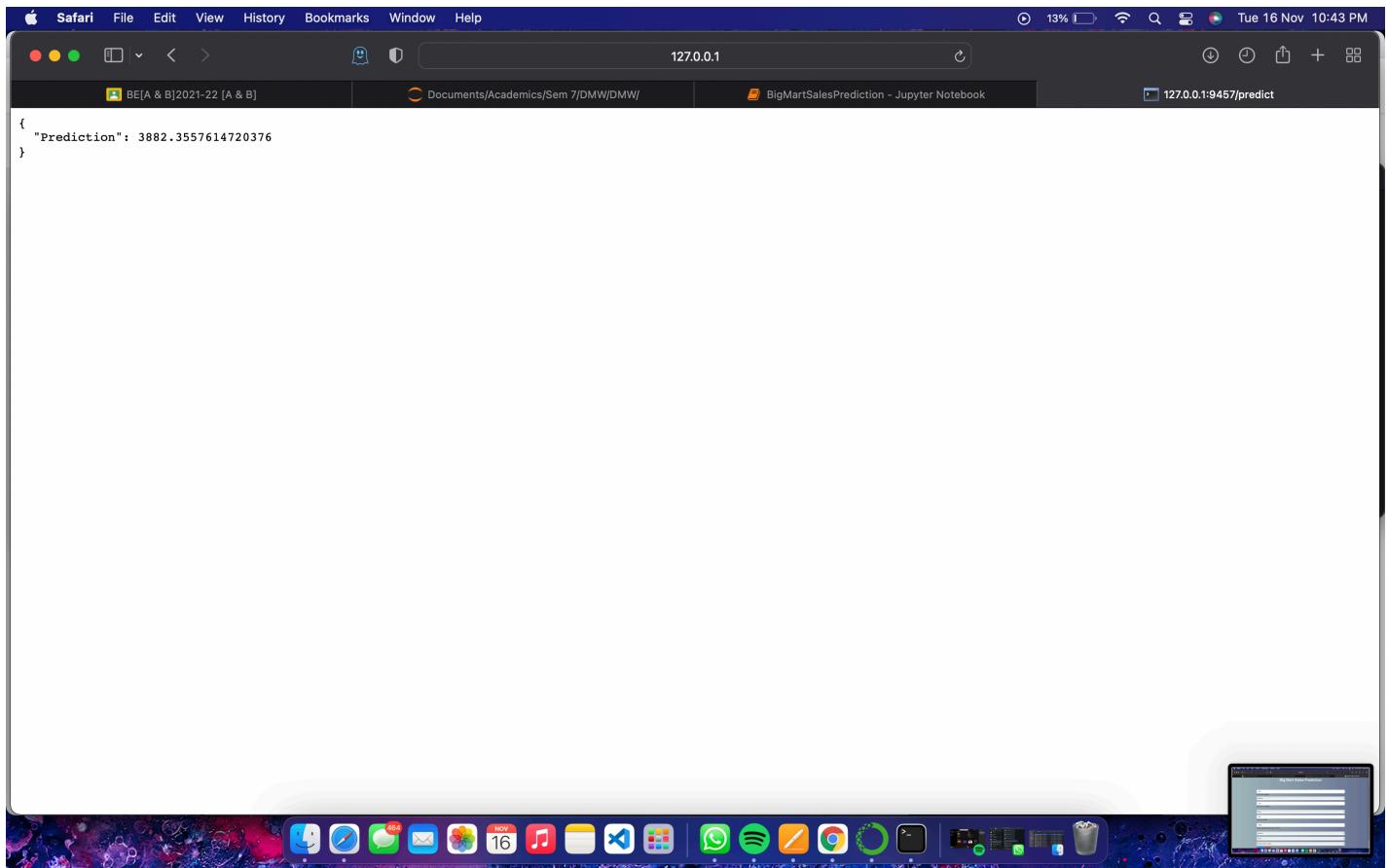


Fig 14, 15 and 16: Correlation Heatmap, Home Page and Sales Prediction

6. Conclusions

In this project, we have explained the basics of machine learning and related data processing and modeling algorithms, and then how to apply them to forecasting tasks at Big Mart Malls in different locations. When implemented, forecast results show the correlation between the various attributes considered and the method with the highest sales for a particular medium-sized location, and other shopping locations need to improve sales according to a similar pattern. Accuracy, which plays an important role in predictive-based systems, can improve significantly as the number of parameters used increases. Checking the functionality of the partial model can also improve the productivity of the system. Projects can be further collaborated using built-in intelligence on web-based applications or any device supported by the Internet of Things (IoT). Various stakeholders working with sales information can also provide more input to help generate hypotheses, consider more cases, and generate more accurate results that are closer to the actual situation. Combined with effective data mining methods and characteristics, traditional tools can have a higher and more positive effect on the overall development of a company's overall tasks. One of the main highlights is the more expressive regression output, which is easier to understand with some accuracy. In addition, the flexibility of the proposed approach can be enhanced by variants at a very appropriate stage of regression modeling. Experiments are still needed to be able to properly evaluate and optimize correct measurements of both accuracy and resource efficiency.