M-SKAM Future Unit Sales Forecasting

A project report submitted to Rashtrasant Tukadoji Maharaj Nagpur University in partial fulfillment for the award of Degree of

Bachelor of Engineering

In

Computer Science and Engineering

By

MANMEET GANDHI

SHREYANSH SAHU

KARTIKEYAN GUPTA

ADNAN HUSAIN

MAHESH JAGANIYA

Guide

PROF. VRUSHALI K BONGIRWAR

Department of Computer Science and Engineering,
Shri Ramdeobaba College of Engineering And Management,
Nagpur-440013

(An Autonomous Institution of RashtrasantTukadojiMaharaj Nagpur University)

November 2019

SHRI RAMDEOBABA COLLEGE OF ENGINEERING & MANAGEMENT, NAGPUR

(An Autonomous Institute affiliated to Rashtrasant Tukdoji Maharaj Nagpur University)

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the Thesis on**"M-SKAM Future Unit Sales Forecasting"** is abonafide work of

MANMEET GANDHI

SHREYANSH SAHU

KARTIKEYAN GUPTA

ADNAN HUSAIN

MAHESH JAGANIYA

submitted to the Rashtrasant Tukdoji Maharaj Nagpur University, Nagpur in partial fulfillment of the award of a Degree of Bachelor of Engineering, in Computer Science and Engineering. It has been carried out at the Department of Computer Science and Engineering, Shri Ramdeobaba College of Engineering and Management, Nagpur during the academic year 2019-20.

Date :16 November, 2019
Place: Nagpur

Prof.V. K. Bongirwar	Dr. M.B. Chandak
Project guide	H.O.D
Department of Computer Science and Engineering	Department of Computer Science and Engineering

DECLARATION

We, hereby declare that the thesis titled **"M-SKAM Future Unit Sales Forecasting"**submitted herein, has been carried out in the Department of Computer Science and Engineering of Shri Ramdeobaba College of Engineering & Management, Nagpur. The work is original and has not been submitted earlier as a whole or part for the award of any degree /diploma at this or any other institution / University.

Date :16 November, 2019
Place: Nagpur

KARTIKEYAN GUPTA
Roll No. 62

MANMEET GANDHI
Roll No. 66

SHREYANSH SAHU
Roll No. 81

ADNAN HUSAIN
Roll No. 43

MAHESH JAGANIYA
Roll No. 64

Approval Sheet

This report entitled
“M-SKAM Future Unit Sales Forecasting”

by

MANMEET GANDHI

SHREYANSH SAHU

KARTIKEYAN GUPTA

ADNAN HUSAIN

MAHESH JAGANIYA

is approved for the degree of Bachelor in Computer Science and Engineering.

Name & signature of Supervisor

Name & Signature of External Examiner(s)

Name & signature of HOD

Date: 16 November 2019

Place: Nagpur

ACKNOWLEDGEMENT

Every project small or big is not possible without the guidance and help of various numbers of people in various fields who have always given their valuable advice and support. We sincerely appreciate the inspiration, guidance, dedication and positivity shown by all those people who in some or the other way have contributed in making this project a success.

We are grateful to our Project Guide Prof. V.K.Bongirwarwho was always there to solve our every small doubt about the project. He not only gave us the right path towards approaching our goal, but was also there to provide moral support and valuable inputs.

We forward our thanks to Prof. A.R. Raipurkar,Project Co-ordinator for providing support and guidance for the project completion.

We are indebted to our Head of Department, Dr. M. B. Chandak who gave us a positive response and provided us various approaches through which the project could be implemented and developed further.

We are thankful to all our panelists and supervisors who were present during the project presentation and submission for their valuable comments and advice which also helped us in improving our presentation and communication skills.

By –

KARTIKEYAN GUPTA62

MANMEET GANDHI66

SHREYANSH SHAHU81

ADNAN HUSAIN43

MAHESH JAGANIYA 64

ABSTRACT

Product sales forecasting is a major aspect of purchasing management. Forecasts are crucial in determining inventory stock levels, and accurately estimating future demand for goods has been an ongoing challenge, especially in the Supermarkets and Grocery Stores industry. If goods are not readily available or goods availability is more than demand overall profit can be compromised. As a result, sales forecasting for goods can be significant to ensure loss is minimized. Additionally, the problem becomes more complex as retailers add new locations with unique needs, new products, ever transitioning seasonal tastes, and unpredictable product marketing.

In this analysis, a forecasting model is developed using machine learning algorithms to improve the accurately forecasts product sales. The proposed model is especially targeted to support the future purchase and more accurate forecasts product sales and is not intended to change current subjective forecasting methods. A model based on a real grocery store's data is developed in order to validate the use of the various machine learning algorithms. In the case study, multiple regression methods are compared. The methods impact on forecast product availability in store to ensure they have just enough products at right time

INDEX

LIST OF FIGURES

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background

The unit sales prediction task, or in other words determining which item or product should be maintained in a minimum quantity in stock on the given date, is a critical maintenance task. Zero Inventory maintenance has been tried to achieve by vendor all around the world to minimize the loss due to unavailability of stock or maintaining excessive (unsold) stock of product. Minimum Inventory Managementis a tedious task faced by vendor on a global scale, even with the knowledge of market sales and conditions such as pricing, festivals,oil pricing, market competitions, etc .The problem becomes more complex as retailers add new locations with unique needs, new products, ever transitioning seasonal tastes, and unpredictable product marketing.

For this project we have considered Corporación Favorita, a large Ecuadorian-based grocery retailer's dataset.We are operatingonover 2,00,000 different products sold by multiple stores spread all over the globe and the data set expands up to 3,00,000 rows of data across stores , oil prices , holiday events and promotions.

.

1.2 Problem Definition

To design and train a model using concepts of supervised learning such that given a dataset of previous sales and transactions and predictthe minimum quantity of the product to be maintained in the inventory such that the real time demand could be satisfied and the vendor could generate maximum profit by having available stock.

The dataset has the following files and properties:

- Train.csv: Consists of train data with unit sales per iter per day.
- Stores.csv: Consists of all the stores, their location and their individual store numbers
- Items.csv: Consists all the items, their family, classes and the item number
- Holidays_Events.csv: Consists of the holidays and events metadata.
- Oils.csv: Consists of Daily oil prices.

We are also forecasting the future transactions of each store and studying the effect of oil prices on thetransactions since Ecuador is an oil dependent country

1.3 Need of the system

Currently, around the globe, supermarkets and local vendors face the problem of Inventory Management. This problem could be divided into two states:

- The product is in excessive quantity hence resulting in wastage of space and cost.
- The product is in inadequate quantity hence the customer demands could not be matched hence resulting in Company’s loss.

Hence we need this system to overcome the current problems faced by the Supermarkets and the local vendors.

1.4 Objectives of the system

The objectives are as follows:

• To design a model that predicts the products minimum quantity on given date..

• To train the model over 300000 of rows of previous sales and transactions avoiding over-fitting and under-fitting problems.

• To make the model “trust-worthy” by reaching reasonable amount of accuracy in detecting the abnormalities correctly.

CHAPTER 2

ANALYSIS AND DESIGN

In this chapter, the procedure for implementation of the project and a brief explanation of why it will be useful for implementing the proposed system is included, and a brief description of the current system development approach is counted.

2.1 Methodology

The project development methodology is as follows:

1. Requirement Analysis

   • Study on Corporación Favorita's dataset required for training model

   • Study the attributes importance and contribution in improving the accuracy prediction algorithm

   • Study of different Supervised Machine Learning Algorithms.

   • Study of different Machine Learning and Python Libraries

2. Coding

   • Python Programing

   • Kaggle IDE

   • Libraries

      1. Panda
      2. Numpy
      3. SKLearn
      4. Xgboost
      5. Seaborn

3. Implementation

      - Pre-processingof the given data-set and anomalies removal.
      - Different Training Model Using Machine Learning and Python Libraries.

- Comparison among the accuracies of the different models.

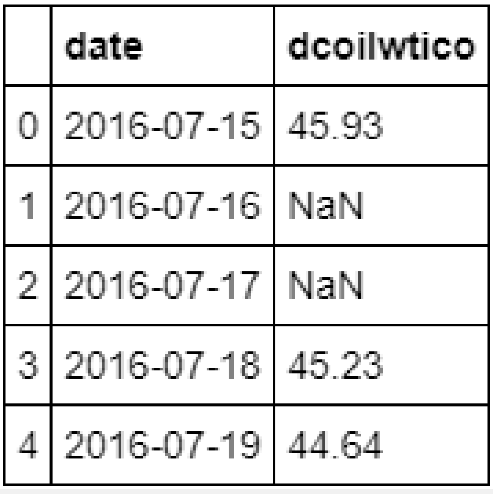
4. Testing

- Testing on proper Cross Validation/Test Dataset
- Feature detecting and categorizing
- Testing dynamic changes in input data

2.2 Flow Control

The main flow of the system comprises of different states which contributes in improving the accuracy of the system.

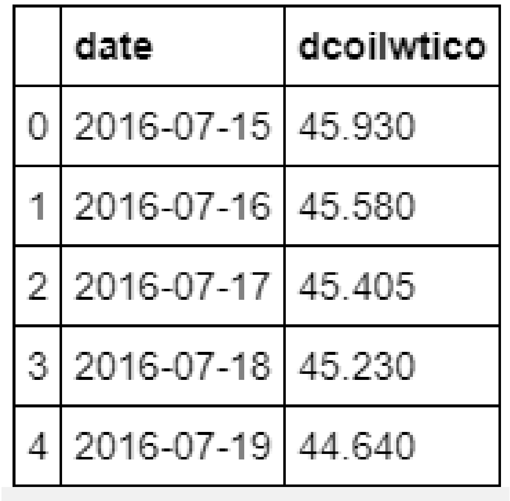
- **Data Loading**
- **Anomaly Detection**



	date	dcoilwtico
0	2016-07-15	45.93
1	2016-07-16	NaN
2	2016-07-17	NaN
3	2016-07-18	45.23
4	2016-07-19	44.64

Table 1: Anomaly Detection

- **Data Preparing and Cleaning**
- **Analyzing Impact of Oil on the sales of other products**



	date	dcoilwtico
0	2016-07-15	45.930
1	2016-07-16	45.580
2	2016-07-17	45.405
3	2016-07-18	45.230
4	2016-07-19	44.640

Table 2: Impact of Oil Prices on Sales

- **Data Pre-Processing**

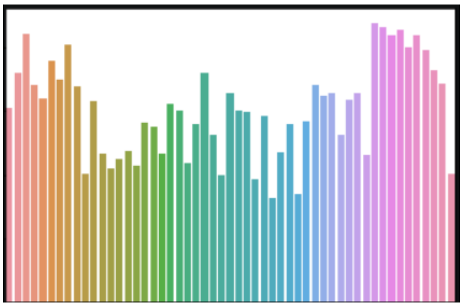


Fig 1: number of stores according to each store type.¶

- **Product Purchase Trend analysis**

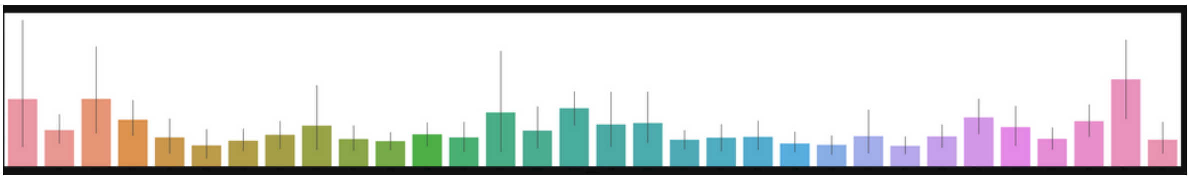


Fig 2: unit sales for different oil prices.

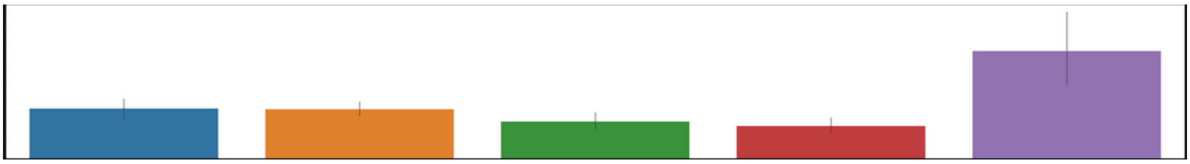


Fig 3: unit sales for different store types.

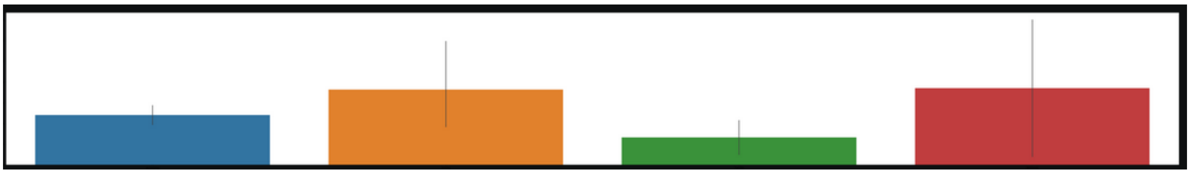


Fig 4:unit sales for different Holiday types.

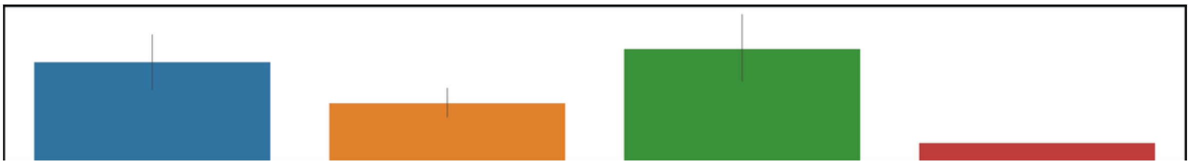


Fig 5: shows unit sales for different family of products.

- ### Exploring and Analyzing Data

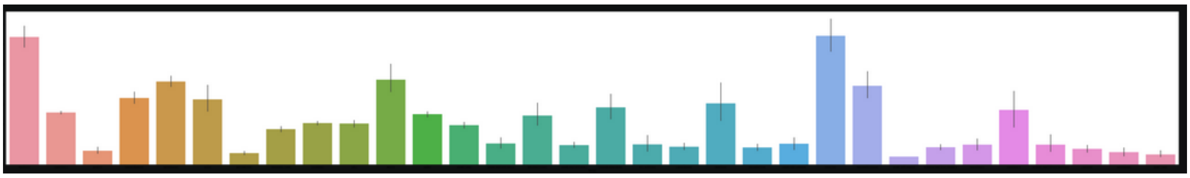


Fig 6: the sales of products per Item family

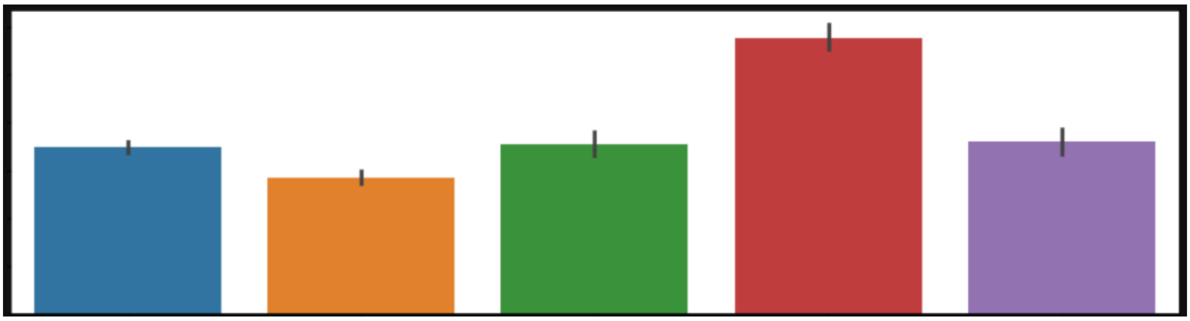
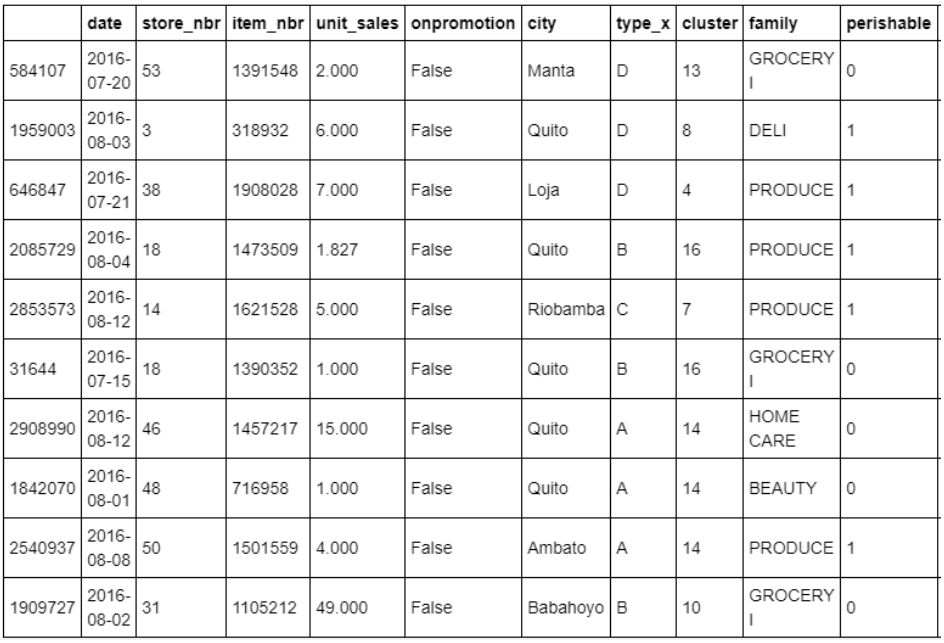


Fig 7: Total Sales per store type



	date	store_nbr	item_nbr	unit_sales	onpromotion	city	type_x	cluster	family	perishable
584107	2016-07-20	53	1391548	2.000	False	Manta	D	13	GROCERY I	0
1959003	2016-08-03	3	318932	6.000	False	Quito	D	8	DELI	1
646847	2016-07-21	38	1908028	7.000	False	Loja	D	4	PRODUCE	1
2085729	2016-08-04	18	1473509	1.827	False	Quito	B	16	PRODUCE	1
2853573	2016-08-12	14	1621528	5.000	False	Riobamba	C	7	PRODUCE	1
31644	2016-07-15	18	1390352	1.000	False	Quito	B	16	GROCERY I	0
2908990	2016-08-12	46	1457217	15.000	False	Quito	A	14	HOME CARE	0
1842070	2016-08-01	48	716958	1.000	False	Quito	A	14	BEAUTY	0
2540937	2016-08-08	50	1501559	4.000	False	Ambato	A	14	PRODUCE	1
1909727	2016-08-02	31	1105212	49.000	False	Babahoyo	B	10	GROCERY I	0

Table 3 : Data Analysis

- ### Applying Machine Learning Models and Accuracy Comparison

	Model Name	Root Mean Square Value
1	Extra Tree Regressor	0.031669213835962835 / 0.0
2	Random Forest Regressor	0.025754442870200338 / 0.0
3	Fradient Boosting	0.0254620285431131 / 0.0
4	XG Boost	0.02505748758471952 / 0.0

Table 4: Model's Accuracy Comparison

2.3Model – XGBOOST

It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. It belongs to a broader collection of tools under the umbrella of the Distributed Machine Learning Community or DMLC who are also the creators of the popular mxnet deep learning library.

XGBoost Features

The library is laser focused on computational speed and model performance, as such there are few frills. Nevertheless, it does offer a number of advanced features.

Model Features

The implementation of the model supports the features of the scikit-learn and R implementations, with new additions like regularization. Three main forms of gradient boosting are supported:

- **Gradient Boosting** algorithm also called gradient boosting machine including the learning rate.
- **Stochastic Gradient Boosting** with sub-sampling at the row, column and column per split levels.
- **Regularized Gradient Boosting** with both L1 and L2 regularization.

System Features

The library provides a system for use in a range of computing environments, not least:

- **Parallelization** of tree construction using all of your CPU cores during training.
- **Distributed Computing** for training very large models using a cluster of machines.
- **Out-of-Core Computing** for very large datasets that don’t fit into memory.
- **Cache Optimization** of data structures and algorithm to make best use of hardware.

Algorithm Features

The implementation of the algorithm was engineered for efficiency of compute time and memory resources. A design goal was to make the best use of available resources to train the model. Some key algorithm implementation features include:

- **Sparse Aware** implementation with automatic handling of missing data values.
- **Block Structure** to support the parallelization of tree construction.
- **Continued Training** so that you can further boost an already fitted model on new data.

CHAPTER 3
SYSTEM DESCRIPTION

3.1 Software tools

1. **Kaggle IDE**

   It is a sophisticated text editor for code, markup and prose. You'll love the slick user interface, extraordinary features and amazing performance. Brackets is a text editor tool that makes it easy to design in the browser, it iss crafted from the ground up for web designers and front-end developers

2. **Python 3.7**

   Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

3. **SK Learn**

   Sklearn is an open source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface.

4. **Pandas**

   pandas (software) In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

5. **NumPy**

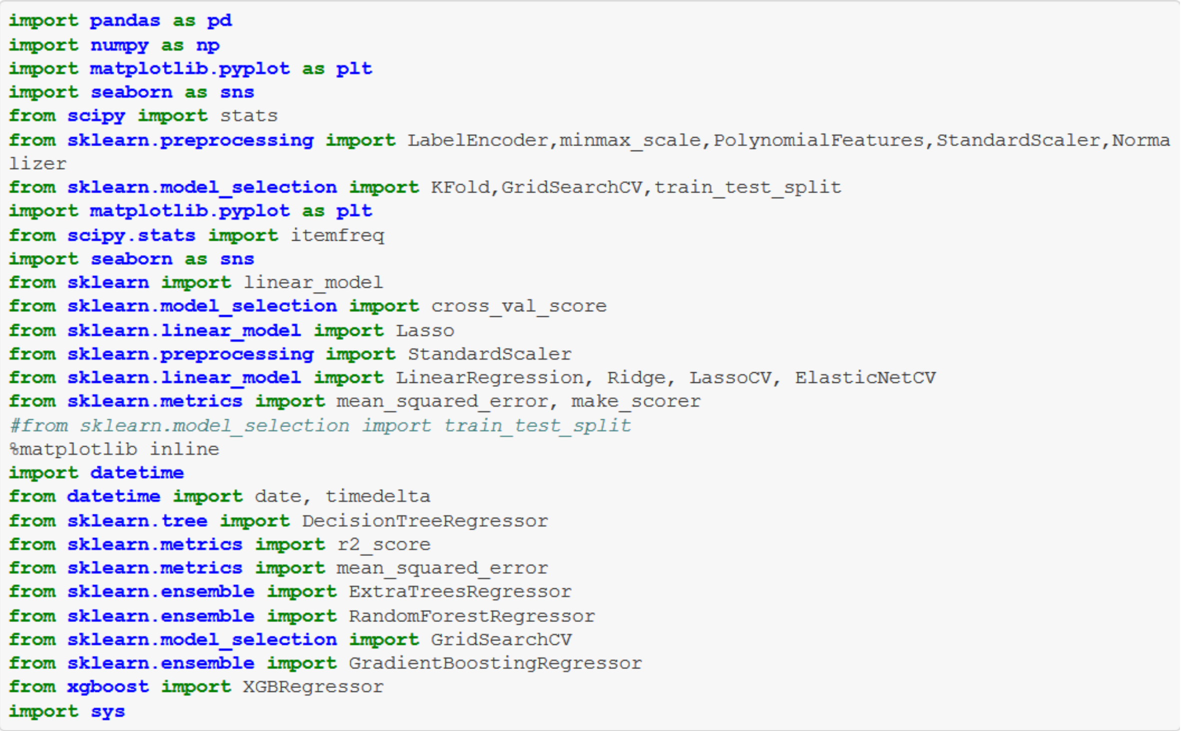
   NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

6. **Seaborn**

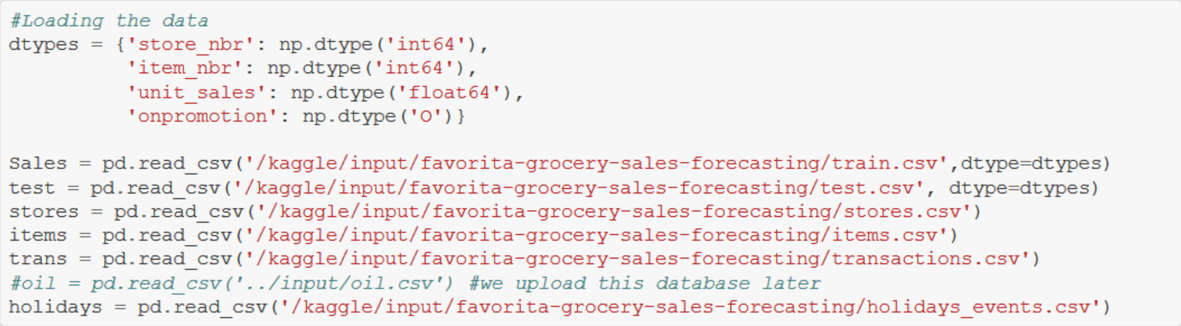
   Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

CHAPTER 4

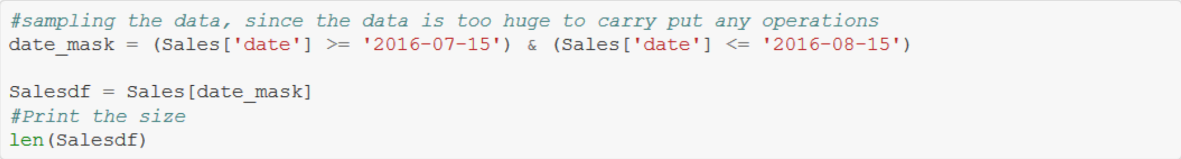
PROJECT IMPLEMENTATION



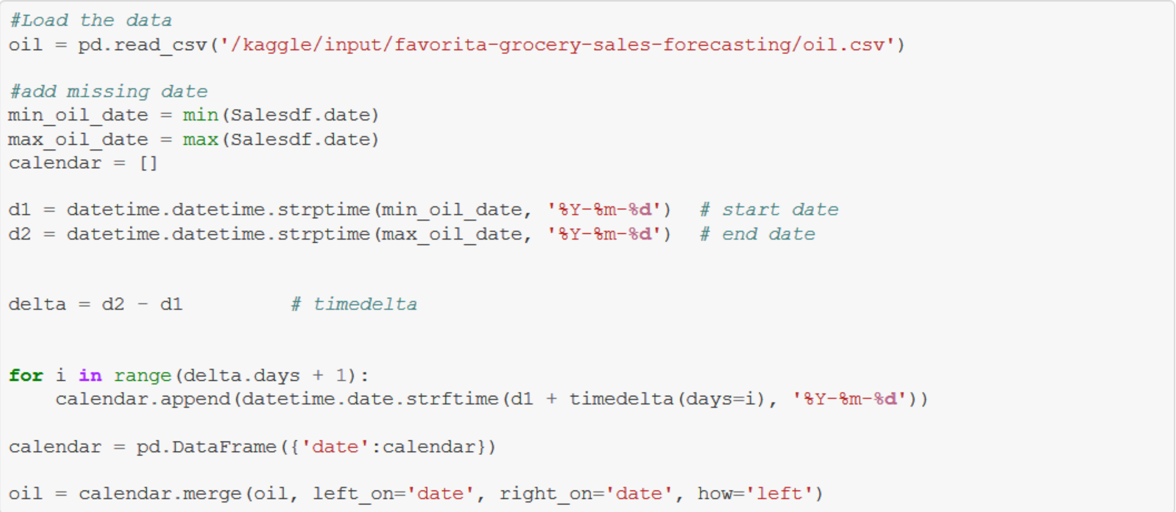
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.preprocessing import LabelEncoder,minmax_scale,PolynomialFeatures,StandardScaler,Norma
lizer
from sklearn.model_selection import KFold,GridSearchCV,train_test_split
import matplotlib.pyplot as plt
from scipy.stats import itemfreq
import seaborn as sns
from sklearn import linear_model
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression, Ridge, LassoCV, ElasticNetCV
from sklearn.metrics import mean_squared_error, make_scorer
#from sklearn.model_selection import train_test_split
%matplotlib inline
import datetime
from datetime import date, timedelta
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingRegressor
from xgboost import XGBRegressor
import sys
```



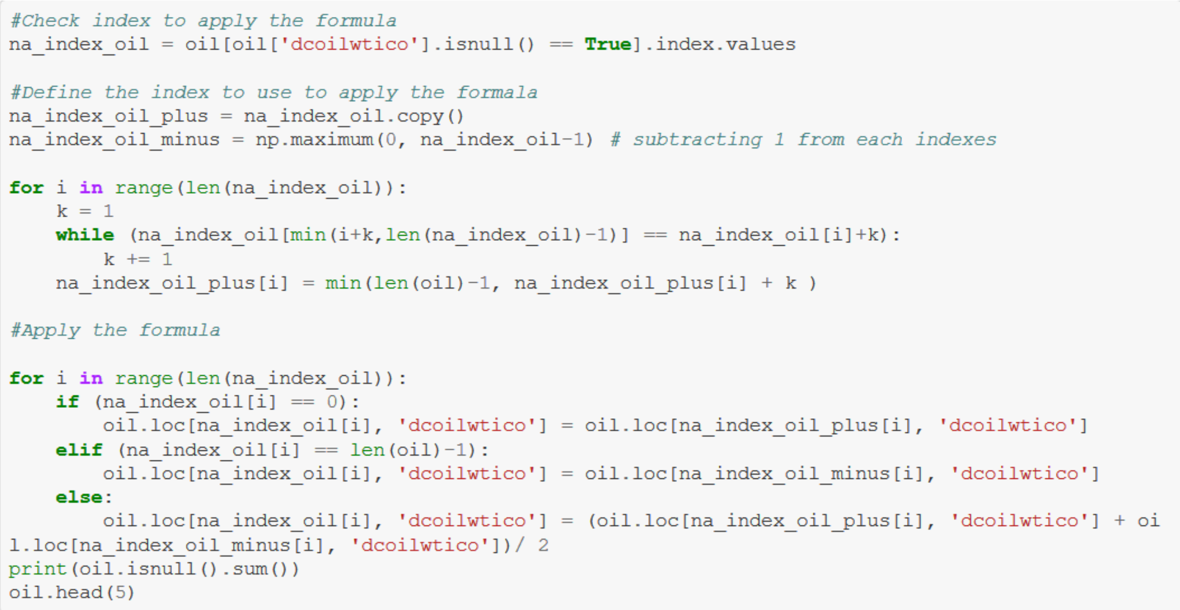
```
#Loading the data
dtypes = {'store_nbr': np.dtype('int64'),
          'item_nbr': np.dtype('int64'),
          'unit_sales': np.dtype('float64'),
          'onpromotion': np.dtype('O')}

Sales = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/train.csv',dtype=dtypes)
test = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/test.csv', dtype=dtypes)
stores = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/stores.csv')
items = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/items.csv')
trans = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/transactions.csv')
#oil = pd.read_csv('../input/oil.csv') #we upload this database later
holidays = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/holidays_events.csv')
```



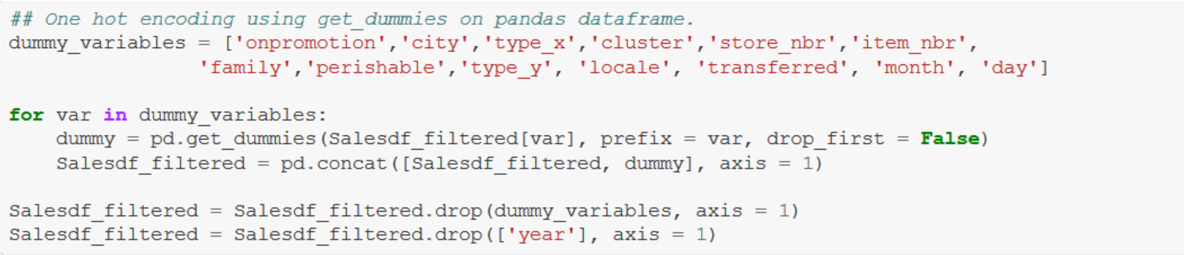
```
#sampling the data, since the data is too huge to carry put any operations
date_mask = (Sales['date'] >= '2016-07-15') & (Sales['date'] <= '2016-08-15')

Salesdf = Sales[date_mask]
#Print the size
len(Salesdf)
```



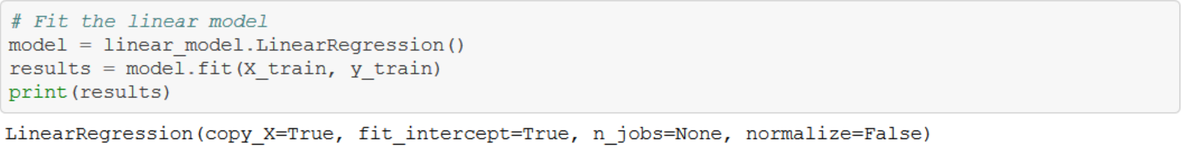
```
#Load the data
oil = pd.read_csv('/kaggle/input/favorita-grocery-sales-forecasting/oil.csv')

#add missing date
min_oil_date = min(Salesdf.date)
max_oil_date = max(Salesdf.date)
calendar = []

d1 = datetime.datetime.strptime(min_oil_date, '%Y-%m-%d')  # start date
d2 = datetime.datetime.strptime(max_oil_date, '%Y-%m-%d')  # end date


delta = d2 - d1         # timedelta


for i in range(delta.days + 1):
    calendar.append(datetime.date.strftime(d1 + timedelta(days=i), '%Y-%m-%d'))

calendar = pd.DataFrame({'date':calendar})

oil = calendar.merge(oil, left_on='date', right_on='date', how='left')
```



```
#Check index to apply the formula
na_index_oil = oil[oil['dcoilwtico'].isnull() == True].index.values

#Define the index to use to apply the formala
na_index_oil_plus = na_index_oil.copy()
na_index_oil_minus = np.maximum(0, na_index_oil-1) # subtracting 1 from each indexes

for i in range(len(na_index_oil)):
    k = 1
    while (na_index_oil[min(i+k,len(na_index_oil)-1)] == na_index_oil[i]+k):
        k += 1
    na_index_oil_plus[i] = min(len(oil)-1, na_index_oil_plus[i] + k )

#Apply the formula

for i in range(len(na_index_oil)):
    if (na_index_oil[i] == 0):
        oil.loc[na_index_oil[i], 'dcoilwtico'] = oil.loc[na_index_oil_plus[i], 'dcoilwtico']
    elif (na_index_oil[i] == len(oil)-1):
        oil.loc[na_index_oil[i], 'dcoilwtico'] = oil.loc[na_index_oil_minus[i], 'dcoilwtico']
    else:
        oil.loc[na_index_oil[i], 'dcoilwtico'] = (oil.loc[na_index_oil_plus[i], 'dcoilwtico'] + oi
l.loc[na_index_oil_minus[i], 'dcoilwtico'])/ 2
print(oil.isnull().sum())
oil.head(5)
```

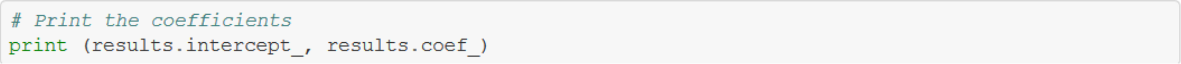


```
## One hot encoding using get_dummies on pandas dataframe.
dummy_variables = ['onpromotion','city','type_x','cluster','store_nbr','item_nbr',
                'family','perishable','type_y', 'locale', 'transferred', 'month', 'day']

for var in dummy_variables:
    dummy = pd.get_dummies(Salesdf_filtered[var], prefix = var, drop_first = False)
    Salesdf_filtered = pd.concat([Salesdf_filtered, dummy], axis = 1)

Salesdf_filtered = Salesdf_filtered.drop(dummy_variables, axis = 1)
Salesdf_filtered = Salesdf_filtered.drop(['year'], axis = 1)
```

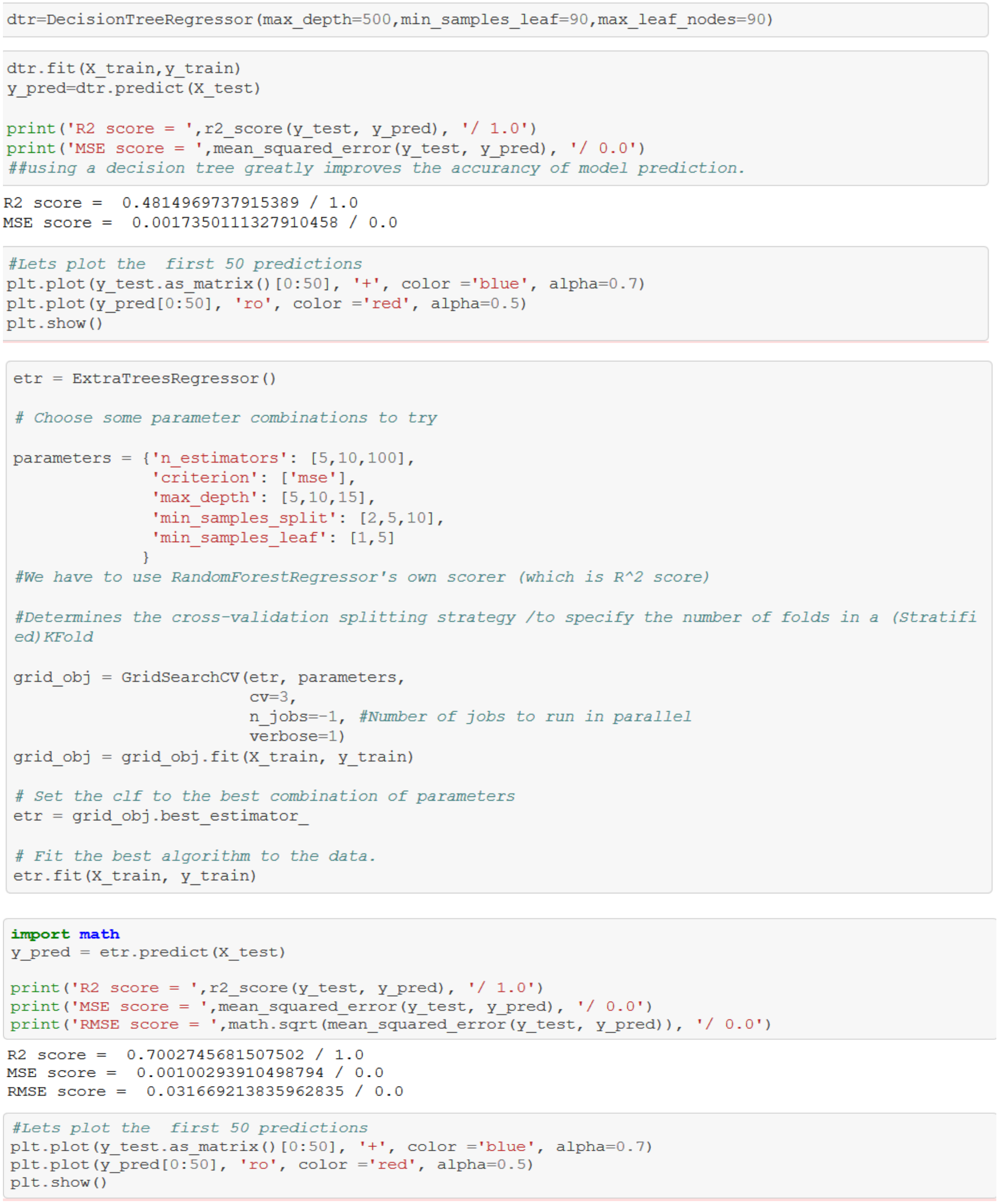


```
# Fit the linear model
model = linear_model.LinearRegression()
results = model.fit(X_train, y_train)
print(results)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```



```
# Print the coefficients
print (results.intercept_, results.coef_)
```



```
dtr=DecisionTreeRegressor(max_depth=500,min_samples_leaf=90,max_leaf_nodes=90)
```

```
dtr.fit(X_train,y_train)
y_pred=dtr.predict(X_test)

print('R2 score = ',r2_score(y_test, y_pred), '/ 1.0')
print('MSE score = ',mean_squared_error(y_test, y_pred), '/ 0.0')
##using a decision tree greatly improves the accurancy of model prediction.
```

```
R2 score =  0.4814969737915389 / 1.0
MSE score =  0.0017350111327910458 / 0.0
```

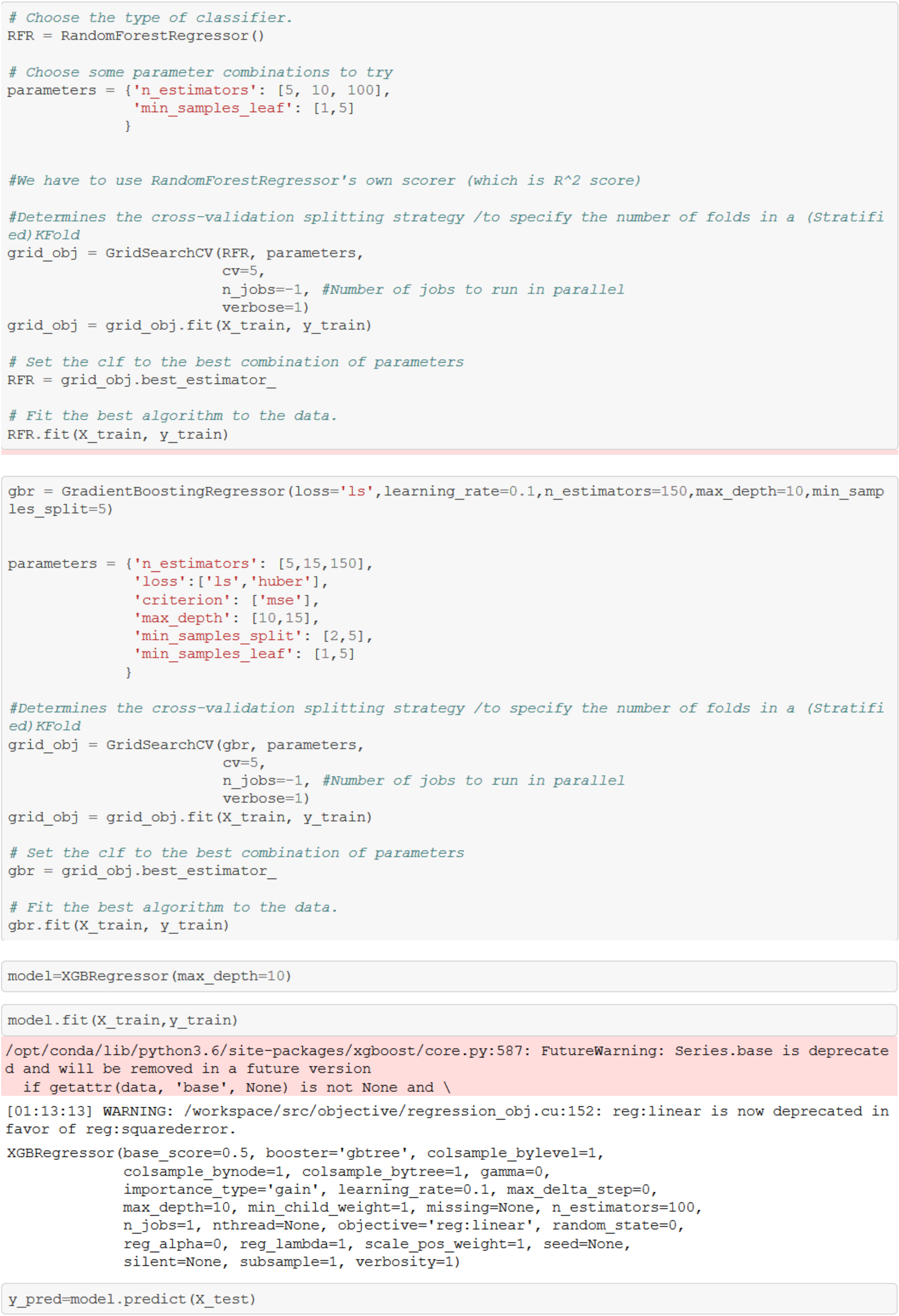
```
#Lets plot the  first 50 predictions
plt.plot(y_test.as_matrix()[0:50], '+', color ='blue', alpha=0.7)
plt.plot(y_pred[0:50], 'ro', color ='red', alpha=0.5)
plt.show()
```

```
etr = ExtraTreesRegressor()

# Choose some parameter combinations to try

parameters = {'n_estimators': [5,10,100],
              'criterion': ['mse'],
              'max_depth': [5,10,15],
              'min_samples_split': [2,5,10],
              'min_samples_leaf': [1,5]
             }
#We have to use RandomForestRegressor's own scorer (which is R^2 score)

#Determines the cross-validation splitting strategy /to specify the number of folds in a (Stratifi
ed)KFold

grid_obj = GridSearchCV(etr, parameters,
                        cv=3,
                        n_jobs=-1, #Number of jobs to run in parallel
                        verbose=1)
grid_obj = grid_obj.fit(X_train, y_train)

# Set the clf to the best combination of parameters
etr = grid_obj.best_estimator_

# Fit the best algorithm to the data.
etr.fit(X_train, y_train)
```

```
import math
y_pred = etr.predict(X_test)

print('R2 score = ',r2_score(y_test, y_pred), '/ 1.0')
print('MSE score = ',mean_squared_error(y_test, y_pred), '/ 0.0')
print('RMSE score = ',math.sqrt(mean_squared_error(y_test, y_pred)), '/ 0.0')
```

```
R2 score =  0.7002745681507502 / 1.0
MSE score =  0.0010029391049879 4 / 0.0
RMSE score =  0.031669213835962835 / 0.0
```

```
#Lets plot the  first 50 predictions
plt.plot(y_test.as_matrix()[0:50], '+', color ='blue', alpha=0.7)
plt.plot(y_pred[0:50], 'ro', color ='red', alpha=0.5)
plt.show()
```



```
# Choose the type of classifier.
RFR = RandomForestRegressor()

# Choose some parameter combinations to try
parameters = {'n_estimators': [5, 10, 100],
              'min_samples_leaf': [1,5]
             }


#We have to use RandomForestRegressor's own scorer (which is R^2 score)

#Determines the cross-validation splitting strategy /to specify the number of folds in a (Stratifi
ed)KFold
grid_obj = GridSearchCV(RFR, parameters,
                        cv=5,
                        n_jobs=-1, #Number of jobs to run in parallel
                        verbose=1)
grid_obj = grid_obj.fit(X_train, y_train)

# Set the clf to the best combination of parameters
RFR = grid_obj.best_estimator_

# Fit the best algorithm to the data.
RFR.fit(X_train, y_train)
```

```
gbr = GradientBoostingRegressor(loss='ls',learning_rate=0.1,n_estimators=150,max_depth=10,min_samp
les_split=5)


parameters = {'n_estimators': [5,15,150],
              'loss':['ls','huber'],
              'criterion': ['mse'],
              'max_depth': [10,15],
              'min_samples_split': [2,5],
              'min_samples_leaf': [1,5]
             }

#Determines the cross-validation splitting strategy /to specify the number of folds in a (Stratifi
ed)KFold
grid_obj = GridSearchCV(gbr, parameters,
                        cv=5,
                        n_jobs=-1, #Number of jobs to run in parallel
                        verbose=1)
grid_obj = grid_obj.fit(X_train, y_train)

# Set the clf to the best combination of parameters
gbr = grid_obj.best_estimator_

# Fit the best algorithm to the data.
gbr.fit(X_train, y_train)
```

```
model=XGBRegressor(max_depth=10)
```

```
model.fit(X_train,y_train)
```

```
/opt/conda/lib/python3.6/site-packages/xgboost/core.py:587: FutureWarning: Series.base is deprecate
d and will be removed in a future version
  if getattr(data, 'base', None) is not None and \
```

```
[01:13:13] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in
favor of reg:squarederror.
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0,
             importance_type='gain', learning_rate=0.1, max_delta_step=0,
             max_depth=10, min_child_weight=1, missing=None, n_estimators=100,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
```

```
y_pred=model.predict(X_test)
```

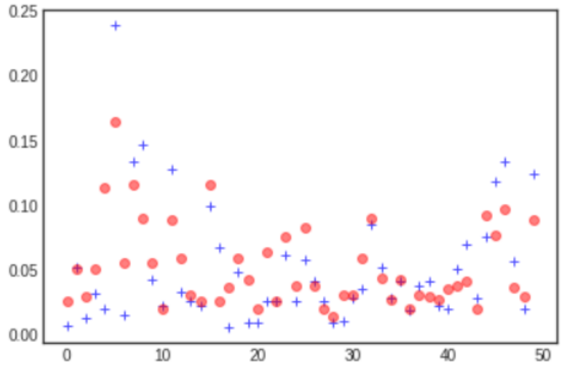


Fig 8:Decision tree plot

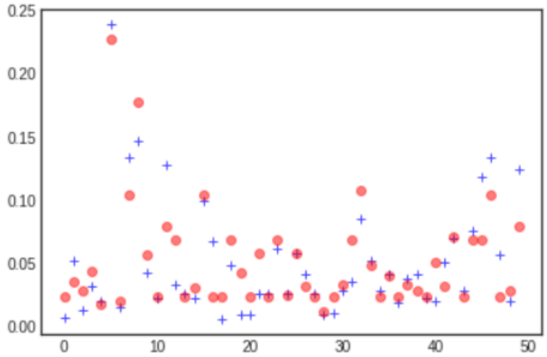


Fig 9:Extra Tree regresser Plot

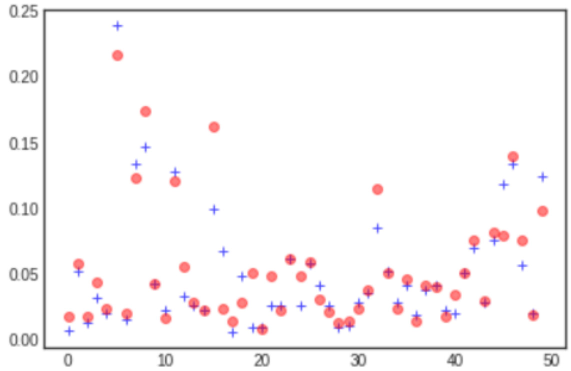


Fig 10: Random Forest plot

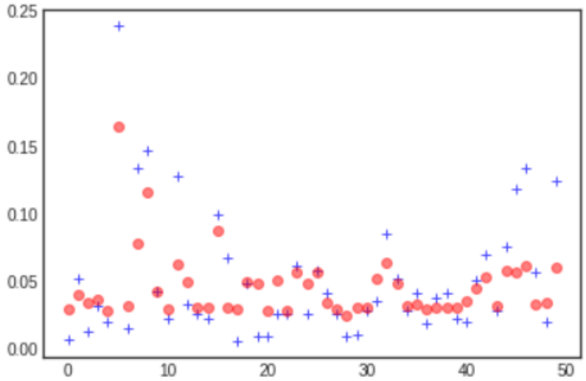


Fig 11: Gradient Boost plot

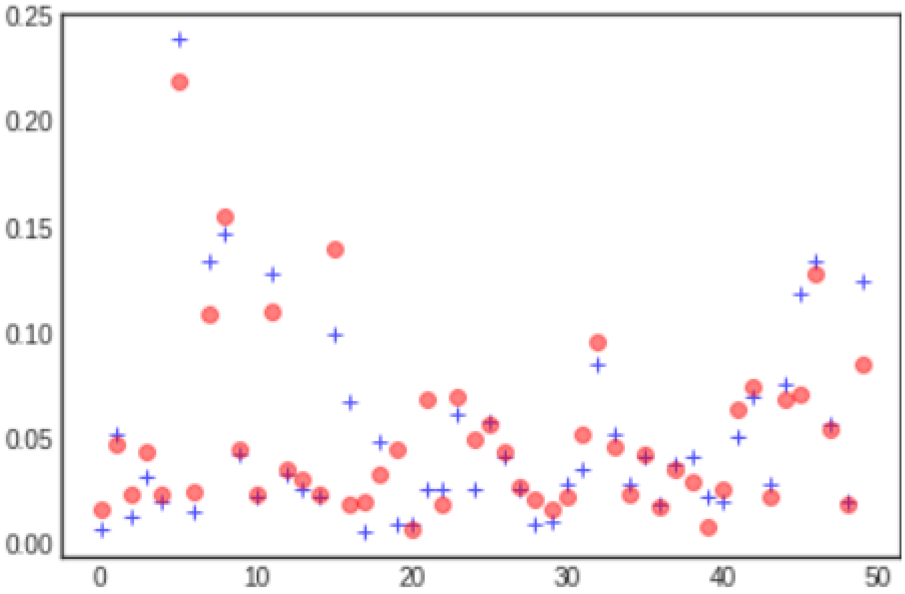
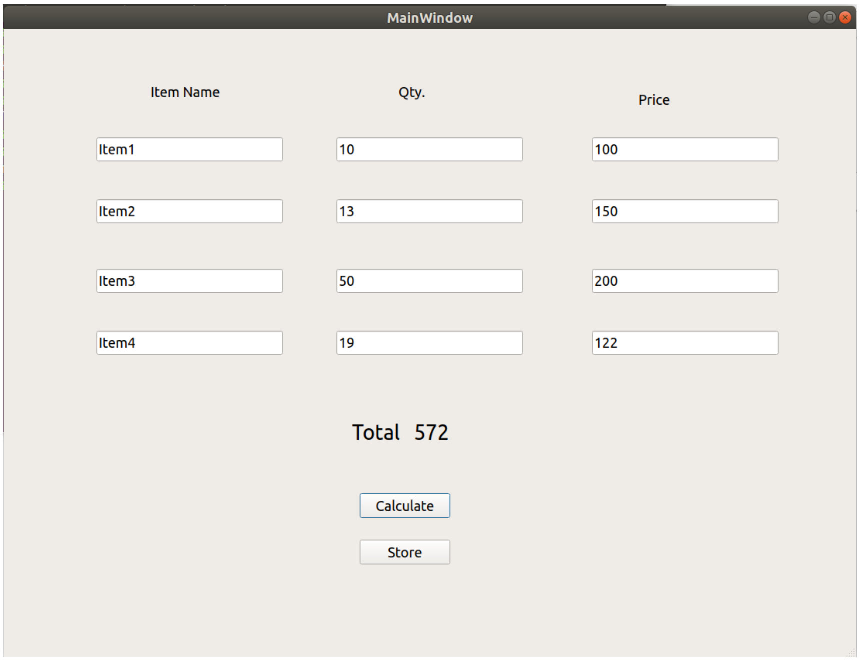


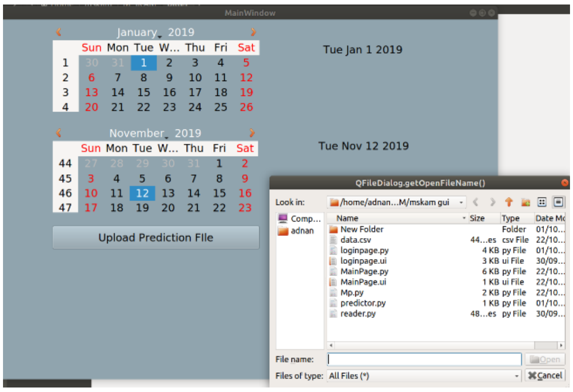
Fig 12: XG BOOST Plot



Item Name	Qty.	Price
Item1	10	100
Item2	13	150
Item3	50	200
Item4	19	122

Total 572

Fig 13:Biller System



January 2019

	Sun	Mon	Tue	W...	Thu	Fri	Sat
1	30	31	1	2	3	4	5
2	6	7	8	9	10	11	12
3	13	14	15	16	17	18	19
4	20	21	22	23	24	25	26

Tue Jan 1 2019

November 2019

	Sun	Mon	Tue	W...	Thu	Fri	Sat
44	27	28	29	30	31	1	2
45	3	4	5	6	7	8	9
46	10	11	12	13	14	15	16
47	17	18	19	20	21	22	23

Tue Nov 12 2019

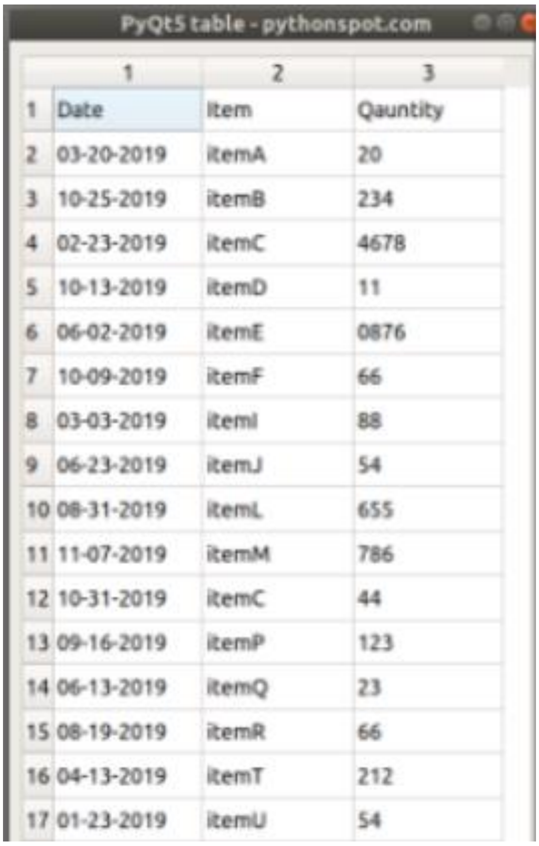
Look in: /home/adnan...M/mskam gui

Name	Size	Type	Date Mc
New Folder		Folder	01/10...
data.csv	44...es	csv File	22/10...
loginpage.py	4 KB	py File	01/10...
loginpage.ui	3 KB	ui File	30/09...
MainPage.py	6 KB	py File	22/10...
MainPage.ui	1 KB	ui File	22/10...
Mp.py	2 KB	py File	22/10...
predictor.py	1 KB	py File	01/10...
reader.py	48...es	py File	30/09...

File name:

Files of type: All Files (*)

Fig 14: Predictor GUI



	1	2	3
1	Date	Item	Qauntity
2	03-20-2019	itemA	20
3	10-25-2019	itemB	234
4	02-23-2019	itemC	4678
5	10-13-2019	itemD	11
6	06-02-2019	itemE	0876
7	10-09-2019	itemF	66
8	03-03-2019	itemI	88
9	06-23-2019	itemJ	54
10	08-31-2019	itemL	655
11	11-07-2019	itemM	786
12	10-31-2019	itemC	44
13	09-16-2019	itemP	123
14	06-13-2019	itemQ	23
15	08-19-2019	itemR	66
16	04-13-2019	itemT	212
17	01-23-2019	itemU	54

Fig 15: Predictor Output

CHAPTER 5
RESULT AND DISCUSSION

We have evaluated models using the Root Means Square Values. The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model's predicted values. Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit. As the square root of a variance, RMSE can be interpreted as the standard deviation of the unexplained variance, and has the useful property of being in the same units as the response variable. Lower values of RMSE indicate better fit. RMSE is a good measure of how accurately the model predicts the response, and it is the most important criterion for fit if the main purpose of the model is prediction.

After evaluating all the models, the XGBoost model gives the best accuracy in terms of RMSE value. Hence for training and prediction algorithm we have finalized the XGBoost model.

Sales Forecasting is the process of using the company's sales records of the past years to predict the short-term or long-term performance in the future. This is one of the pillars of proper financial planning. As with any prediction-related process, risk and uncertainty are unavoidable in Sales Forecasting too. Hence, it's considered good practice for forecasting teams to mention the degree of uncertainties in their forecast.

Accurately forecasting sales and building a sales plan can help to avoid unforeseen cash flow problems and manage production, staff and financing needs more effectively.

CHAPTER 6

CONCLUSION

In this project, the system uses a XGBOOST module to predict future unit sales for the Favorita’s dataset. With reasonably good amount of accuracy advancements of this model could provide a great prediction system to Shopping marts and local vendors to increase their profit margin by meeting the full filling customers demand and minimizing the Inventory Maintenance cost.

CHAPTER 7

FUTURE SCOPE

As of now the system predicts the quantity based on the previous transactions and other affecting attribute. As a future improvement we have discussed and planned on various domains which are:

- Dynamic Graphical User Interface for better understanding.
- Research on algorithms and models for improving the efficiency, hence minimizing the Company's Inventory maintenance cost.
- Online billing system for Shopping Marts and local vendors so that real time data could be stored and be used for further predictions.
- Neural Network implementation.

CHAPTER 8

REFERENCES

[1] Cui, G., Wong, M. L., &Lui, H. K. (2006). Machine learning for direct marketing response models: Bayesian networks with evolutionary programming management Science, 52(4), 597-612

[2] Taylor, E. L. (2014). Predicting Consumer Behaviour. Research World, 2014(46), 67-68

[3] Morwitz, V. G., Steckel, J. H., & Gupta, A. (2007). When do purchase intentions predict sales?. International Journal of Forecasting, 23(3), 347-364

[4] https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data

[5] https://en.wikipedia.org/wiki/Xgboost

[6] https://en.wikipedia.org/wiki/Random_forest

[7] https://en.wikipedia.org/wiki/Decision_tree

[8]https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/

[9] https://www.tutorialspoint.com/sales_forecasting/sales_forecasting_discussion.htm