1. Project description and scope

1.1 Project description

In this project, we focus our study on applying the machine learning algorithms XGBoosting regression and Random Forest to predict future sales for product families sold at Corporacion Favorita stores located in Ecuador. Corporacion Favorita C.A. is a large Ecuadorian retail company that operates supermarkets, hypermarkets and convenience stores across the country. It was founded in 1952 and has since grown to become one of the largest retail companies in Ecuador.

In this study XGBoosting Regression and Random Forest have been used as a machine learning model and applied to the dataset so as to forecast the sales of the company. This study considered comparing the different types of ensemble model to the performance of the predicted sale.

The forecasting sale will help the business grow and predict how much inventory to stock in each type of the product at the given respective stores. If the store has more groceries than the demand by the customers, this will lead to the problem of overstock, waste and perishable goods. On the other hand, if the grocery store has fewer products than what customers need, this will result in dissatisfied customers and reduced revenue.

1.2 Scope

a. The project explores the data Analysis of the store sales patterns from the Corporacion Favorita stores from the selected dataset.
b. The project provides further analysis into the trends and cycles of the store sales within a given period of time.
c. The project gives a deeper insight into how XGBoosting Regression and Random Forest machine learning algorithms work and their application on the selected dataset.
d. Finally, the project conducts an evaluation of the performance of the two models, XGBoosting Regression and Random Forest so as to make a final comparison and conclusion about which algorithm is better.

2. Input dataset explanation

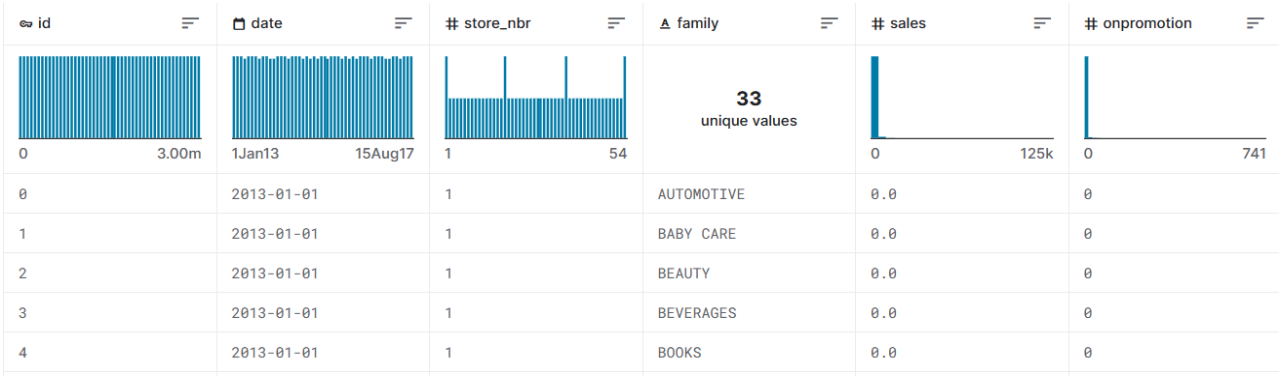
The dataset pertains to sales forecasting for Favorita stores in Ecuador. There are 54 stores located in 16 states of Ecuador, and have 33 product families. Thus, there are 28,512 values that need to be predicted (33 * 54 * 16). The dataset comprises six files, including train.csv, test.csv, stores.csv, oil.csv, holidays_events.csv, and transactions.csv.

2.1 train.csv

In this dataset, it contains 6 attributes. It contains some features and the label to predict sales, the number of sales per day. In total, there are 3,000,888 rows. The start date of the train dataset is 2013-01-01 and the last date of the train dataset is 2017-08-15.

Size 121.8 MB

Attributes	Description	Type
id	The index of the row	numerical
date	Date time	datetime
store_nbr	Store number	numerical
family	Type of product sold	categorical
sales	Number of sales for a product family at a particular store at a given date	numerical
on promotion	Number of products in a product family that were being promoted at a store	numerical



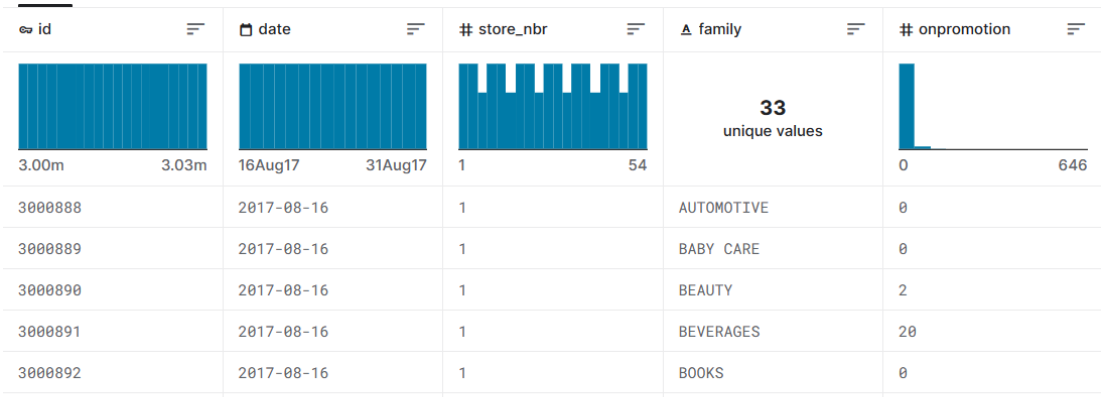
id	date	store_nbr	family	sales	onpromotion
0 – 3.00m	1Jan13 – 15Aug17	1 – 54	33 unique values	0 – 125k	0 – 741
0	2013-01-01	1	AUTOMOTIVE	0.0	0
1	2013-01-01	1	BABY CARE	0.0	0
2	2013-01-01	1	BEAUTY	0.0	0
3	2013-01-01	1	BEVERAGES	0.0	0
4	2013-01-01	1	BOOKS	0.0	0

2.2 test.csv

In this dataset, it contains 5 attributes. The dates in the test data are for the 15 days after the last date in the training data. In total, there are 28,512 rows. The start date of the test dataset is 2017-08-16 and the last date of the test dataset is 2017-08-31.

Size 1.02 MB

Attributes	Description	Type
id	The index of the row	numerical
date	Date time	datetime
store_nbr	Store number	numerical
family	Type of product sold	categorical
on promotion	Number of products in a product family that were being promoted at a store	numerical



id	date	store_nbr	family	onpromotion
3.00m … 3.03m	16Aug17 … 31Aug17	1 … 54	33 unique values	0 … 646
3000888	2017-08-16	1	AUTOMOTIVE	0
3000889	2017-08-16	1	BABY CARE	0
3000890	2017-08-16	1	BEAUTY	2
3000891	2017-08-16	1	BEVERAGES	20
3000892	2017-08-16	1	BOOKS	0

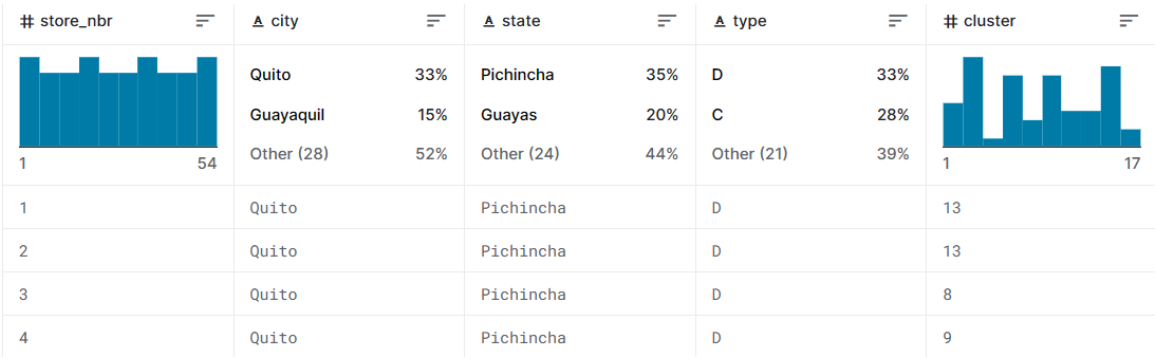
2.3 stores.csv

In this dataset, it contains 5 attributes. It gathers information about the stores. In total, there are 54 rows.

Size 1.39 kB

Attributes	Description	Type
store_nbr	Store number	numerical
city	City store located	categorical

state	State store located	categorical
type	Type of store	categorical
cluster	Grouping of similar stores	numerical



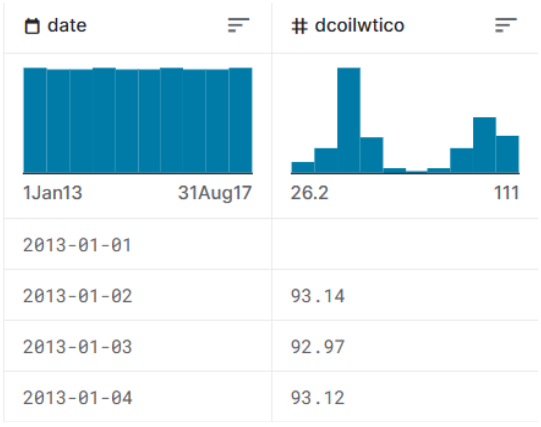
store_nbr	city	state	type	cluster
1 … 54	Quito 33%, Guayaquil 15%, Other (28) 52%	Pichincha 35%, Guayas 20%, Other (24) 44%	D 33%, C 28%, Other (21) 39%	1 … 17
1	Quito	Pichincha	D	13
2	Quito	Pichincha	D	13
3	Quito	Pichincha	D	8
4	Quito	Pichincha	D	9

2.4 oil.csv

In this dataset, it contains 2 attributes. It gathers the daily oil price from January 01, 2013 to August 31, 2017. In total, there are 1,218 rows.

Size 20.58 kB

Attributes	**Description**	**Type**
date	Date time	datetime
dcoilwtico	Oil price in Ecuador in the range 2013-2017.	numerical



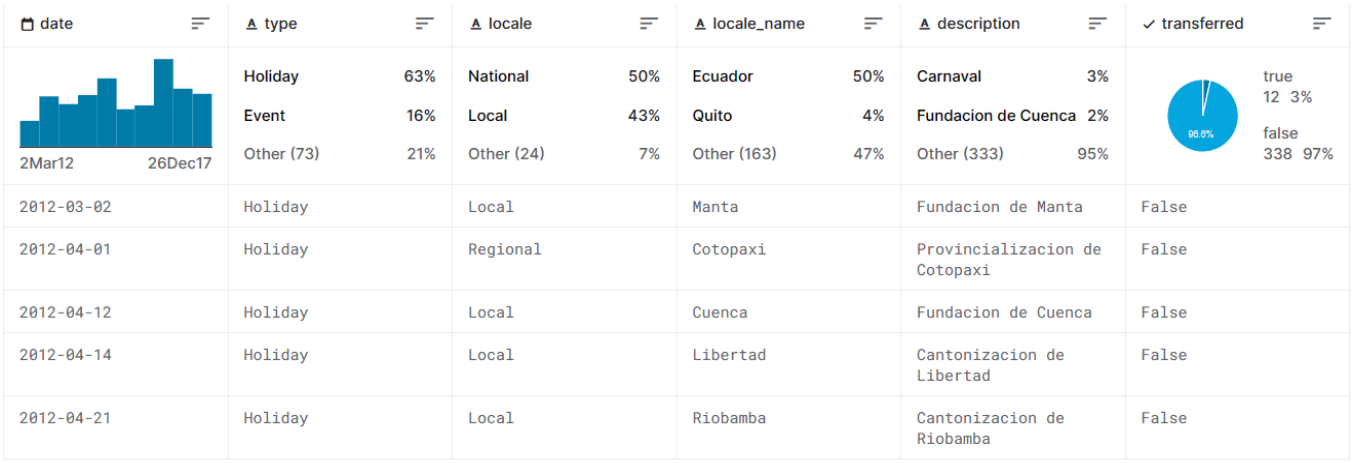
date	dcoilwtico
1Jan13 … 31Aug17	26.2 … 111
2013-01-01	
2013-01-02	93.14
2013-01-03	92.97
2013-01-04	93.12

2.5 holidays_events.csv

This file contains holidays and events in the period of the study. It contains 6 attributes. In total, there are 350 rows.

Size 22.31 kB

Attributes	Description	Type
date	Date time	datetime
type	Type of the holiday	categorical
locale	Scope of the event (Local, Regional, National)	categorical
locale_name	The city where the event takes place	categorical
description	Name of the event	categorical
transferred	Whether the event has been transferred (moved to another day) or not	categorical



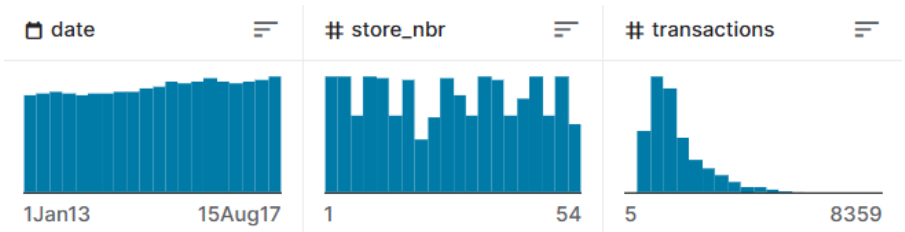
date	type	locale	locale_name	description	transferred
2Mar12 – 26Dec17	Holiday 63% Event 16% Other (73) 21%	National 50% Local 43% Other (24) 7%	Ecuador 50% Quito 4% Other (163) 47%	Carnaval 3% Fundacion de Cuenca 2% Other (333) 95%	true 12 3% false 338 97%
2012-03-02	Holiday	Local	Manta	Fundacion de Manta	False
2012-04-01	Holiday	Regional	Cotopaxi	Provincializacion de Cotopaxi	False
2012-04-12	Holiday	Local	Cuenca	Fundacion de Cuenca	False
2012-04-14	Holiday	Local	Libertad	Cantonizacion de Libertad	False
2012-04-21	Holiday	Local	Riobamba	Cantonizacion de Riobamba	False

2.6 transactions.csv

In this dataset, it contains 3 attributes. It aggregates daily transactions by store. In total, there are 83,488 rows.

Size 1.55 MB

Attributes	Description	Type
date	Date time	datetime
store_nbr	Store number	numerical
transactions	Number of transaction	numerical



date	store_nbr	transactions
2013-01-01	25	770
2013-01-02	1	2111
2013-01-02	2	2358
2013-01-02	3	3487
2013-01-02	4	1922
2013-01-02	5	1903

3. Data preprocessing methods

Data preprocessing refers to the steps and techniques applied to raw data to transform it into a clean, structured, and analyzable format. It involves various tasks to ensure that the data is accurate, complete, consistent, and ready for analysis or machine learning models. The specific preprocessing steps and techniques applied depend on the nature of the data, the analysis objectives, and the algorithms or models being used. Proper data preprocessing can significantly improve the quality and reliability of the results obtained from data analysis or machine learning tasks.

3.1. **Data Cleaning**

Data cleaning refers to the process of identifying and correcting or removing errors, inconsistencies, and inaccuracies in a dataset. It typically involves tasks such as handling missing values. The goal is to ensure that the data is accurate, complete, and suitable for analysis or other purposes. From the dataset, the missing values are found in oil.csv data.

	date	dcoilwtico
0	2013-01-01	NaN
1	2013-01-02	93.14
2	2013-01-03	92.97
3	2013-01-04	93.12
4	2013-01-07	93.20

The Piecewise cubic Hermite interpolation polynomial (PCHIP) technique will be used to handle the missing value for time series dataset. PCHIP is a type of interpolation technique used to smooth the function in some interval. The interpolation is done by using a set of data points and their respective derivatives to smooth the function. It constructs a piecewise polynomial function by interpolating the data points and their derivatives.

Figure 3.1.1.1 Oil price line plot showing the missing value before applying PCHIP

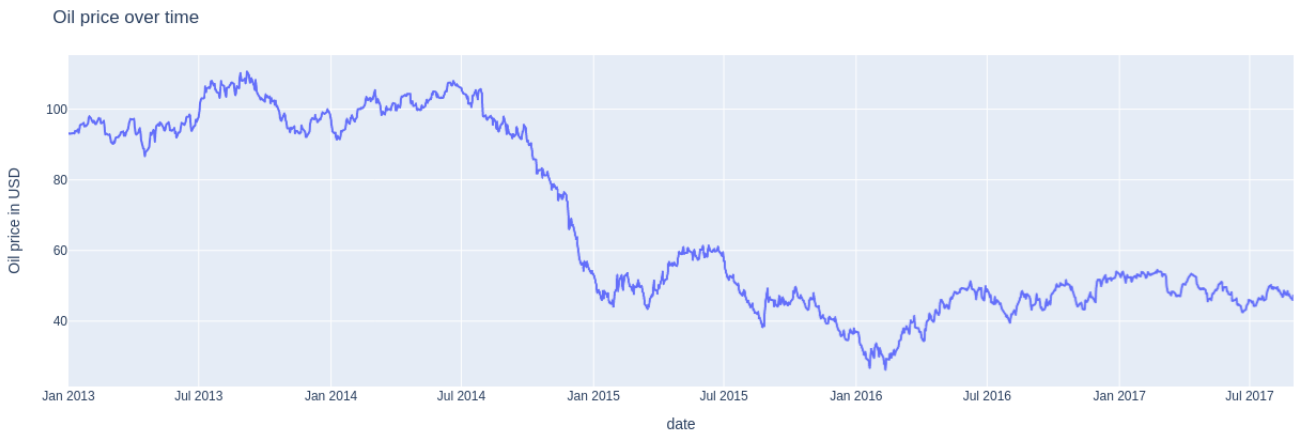


Figure 3.1.1.2 Oil price line plot after applying PCHIP to handle the missing value

3.2 Data Visualization

Data visualization refers to the graphical representation of data to uncover patterns, trends, and insights. It involves creating visual representations, such as charts, graphs, and maps, to effectively communicate complex information. Visualizing data helps identify patterns, outliers, and relationships that may not be evident in raw data. It facilitates data exploration and aids in understanding the underlying information. The project takes advantage of seaborn and matplotlib tools so as to cover the visualization of the data and identify useful trends and cycles of the store sales from the dataset within a given period of time.

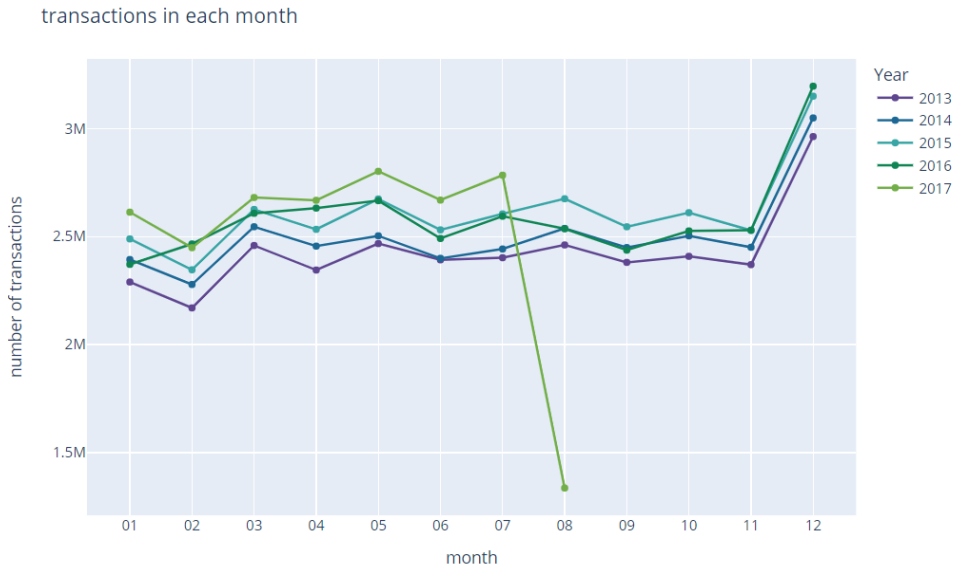


Figure 3.2.1 line plot shows the number of transactions of Corporacion Favorita in each month

Figure 3.2.1 shows the total number of transactions of Corporacion Favorita in each month from 2013 to 2017. It shows that February tends to have the least number of transactions, while December has the highest number of transactions.

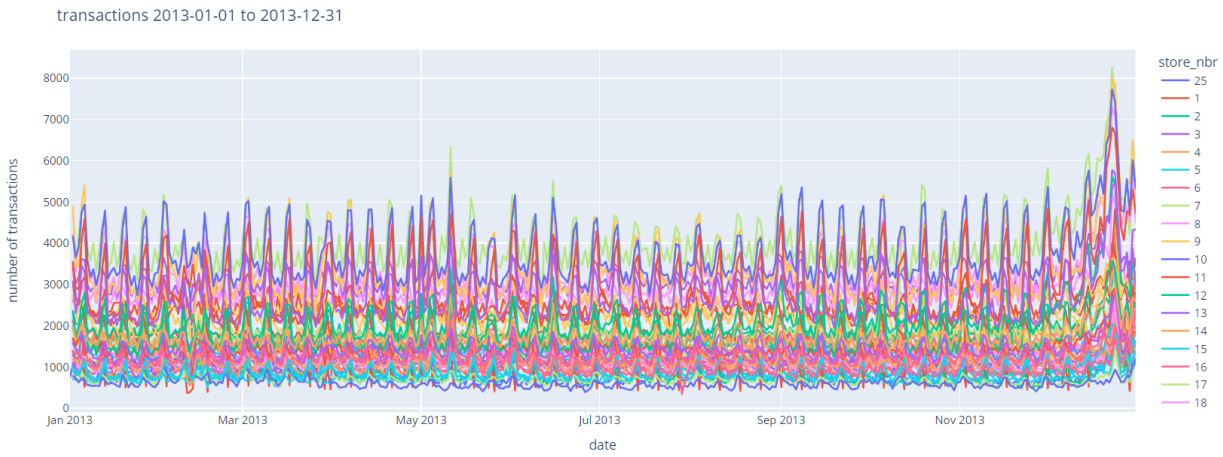
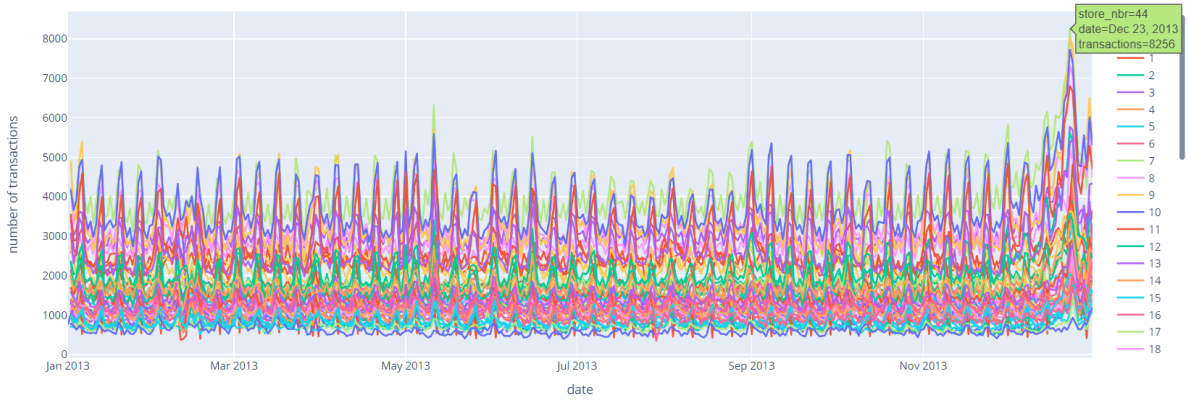


Figure 3.2.2 line plot show the number of transactions of Corporacion Favorita in each store

Figure 3.2.2 shows the total number of transactions of Corporacion Favorita in each store by day from 2013-01-01 to 2013-12-31, each store has its own ID number and different line colors. From the line plot, the distance between the peaks of the two lines is one week or seven days.



The date that has the highest number of transactions in each store is on 23th December before 2 days of Christmas Day. The trends and outcomes are similar in other years.

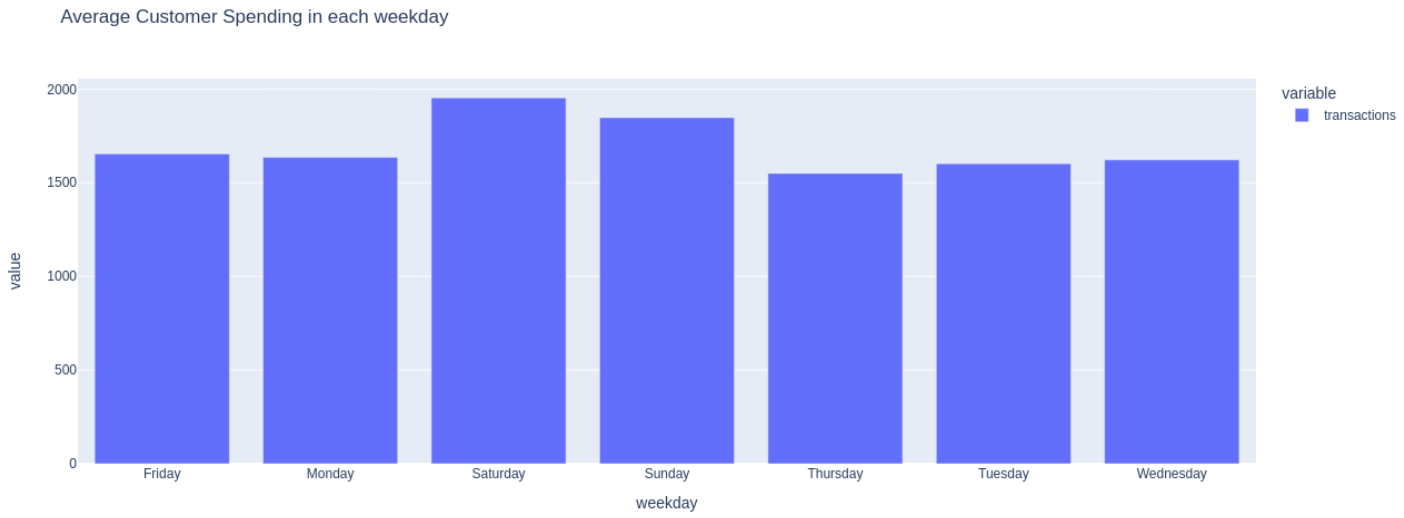


Figure 3.2.3 Barplot of Average customer spending in each weekday

Figure 3.2.3 shows the average spending of each customer during the days of the week. From this graph, Saturday is the day that has the most number of transactions, while Thursday is the day with the least number of transactions spent by customers.

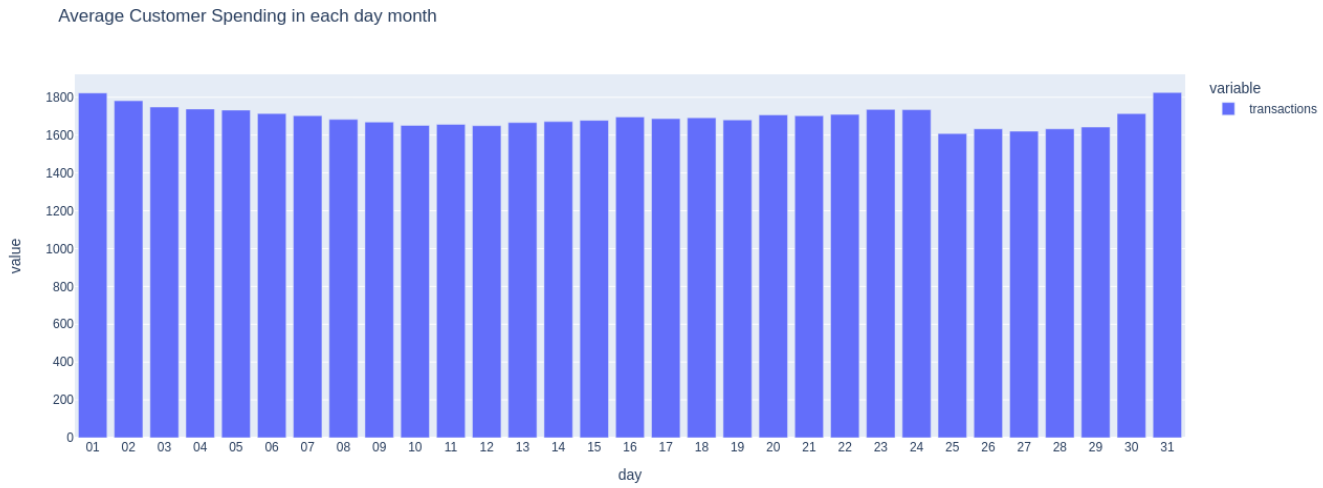
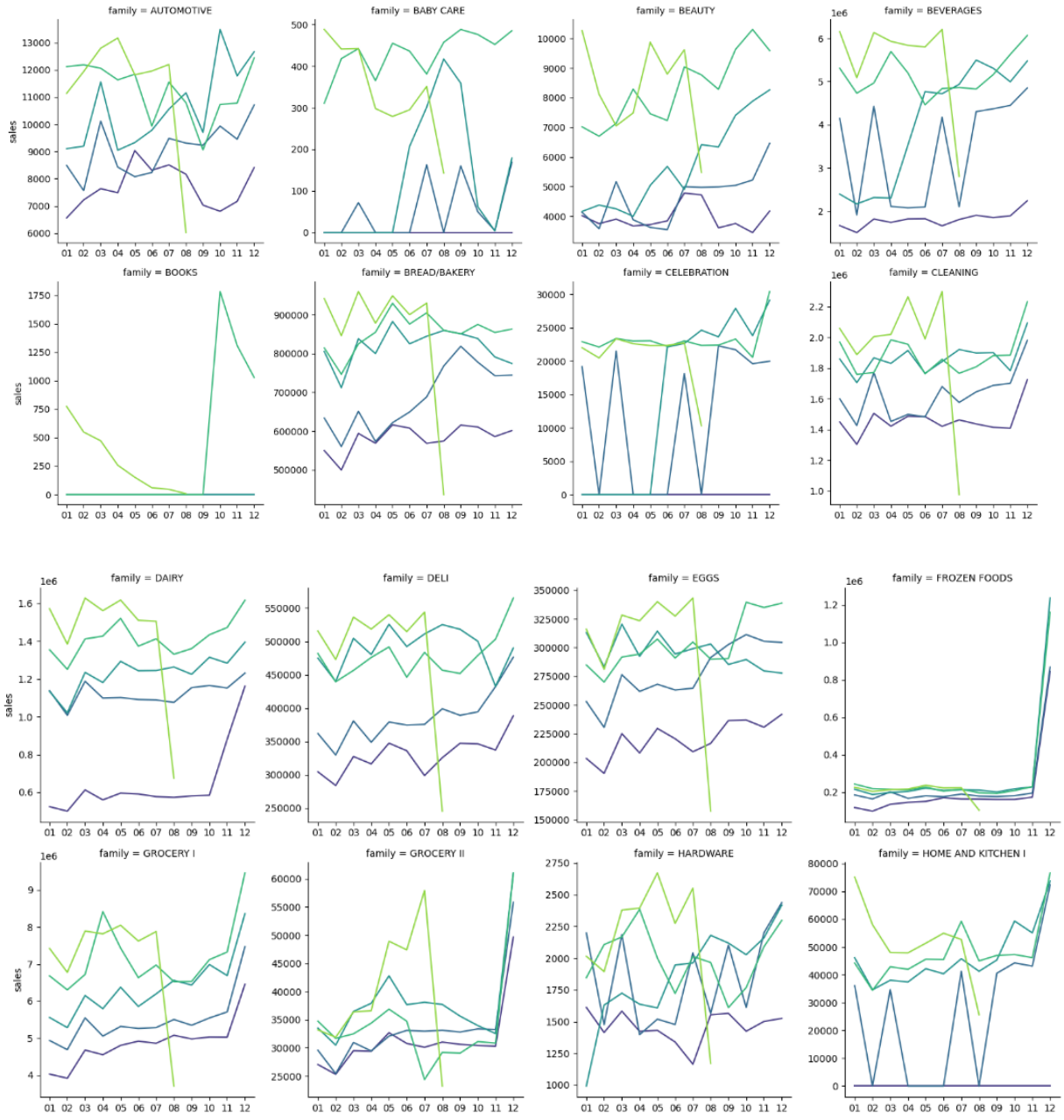


Figure 3.2.4 Barplot of Average customer spending in each day of a month

Figure 3.2.4 shows the average spending of customers according to the 31 days of the month. Key insights from this graph show that the 1st and 31st are the dates that have the most number of daily transactions, while 25th is the date with the least number of transactions. Customer spending is at its peak during the start of the month and at the end of the month.


family = AUTOMOTIVE
family = BABY CARE
family = BEAUTY
family = BEVERAGES
family = BOOKS
family = BREAD/BAKERY
family = CELEBRATION
family = CLEANING
family = DAIRY
family = DELI
family = EGGS
family = FROZEN FOODS
family = GROCERY I
family = GROCERY II
family = HARDWARE
family = HOME AND KITCHEN I
sales
1e6
01 02 03 04 05 06 07 08 09 10 11 12

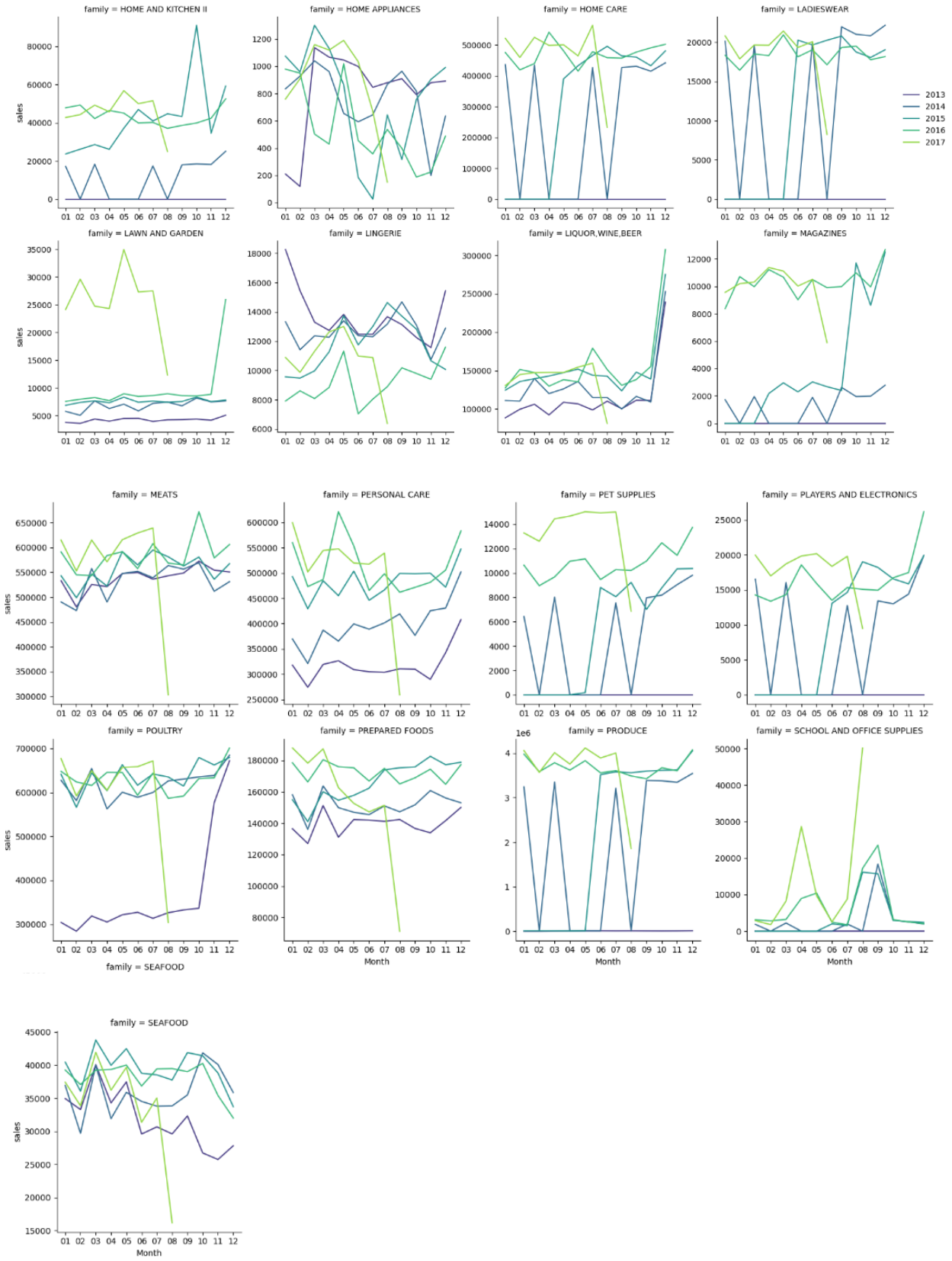

family = HOME AND KITCHEN II
family = HOME APPLIANCES
family = HOME CARE
family = LADIESWEAR
family = LAWN AND GARDEN
family = LINGERIE
family = LIQUOR,WINE,BEER
family = MAGAZINES
family = MEATS
family = PERSONAL CARE
family = PET SUPPLIES
family = PLAYERS AND ELECTRONICS
family = POULTRY
family = PREPARED FOODS
family = PRODUCE
family = SCHOOL AND OFFICE SUPPLIES
family = SEAFOOD
2013
2014
2015
2016
2017
sales
Month
1e6

Figure 3.2.5 Line plot showing the unit sales of each family product by the calendar months of the year.

Figure 3.2.5 shows the relationship between each family product and sales in each month of the year. Each line represents the year from 2013 to 2017.

During December, the sales of frozen food, liquor, wine and beer have increased significantly. July and August are the months when the sales of school and office supplies reach their highest point.

The sales of seafood and school and office supplies have decreased in December compared to the previous month, for the years 2014, 2015, and 2016. On the other hand, during this time of the year, sales of other family products have increased.

During August of 2017, the sales of every family product hit their lowest point, with the exception of School and office supplies, which experienced their highest sales volume.

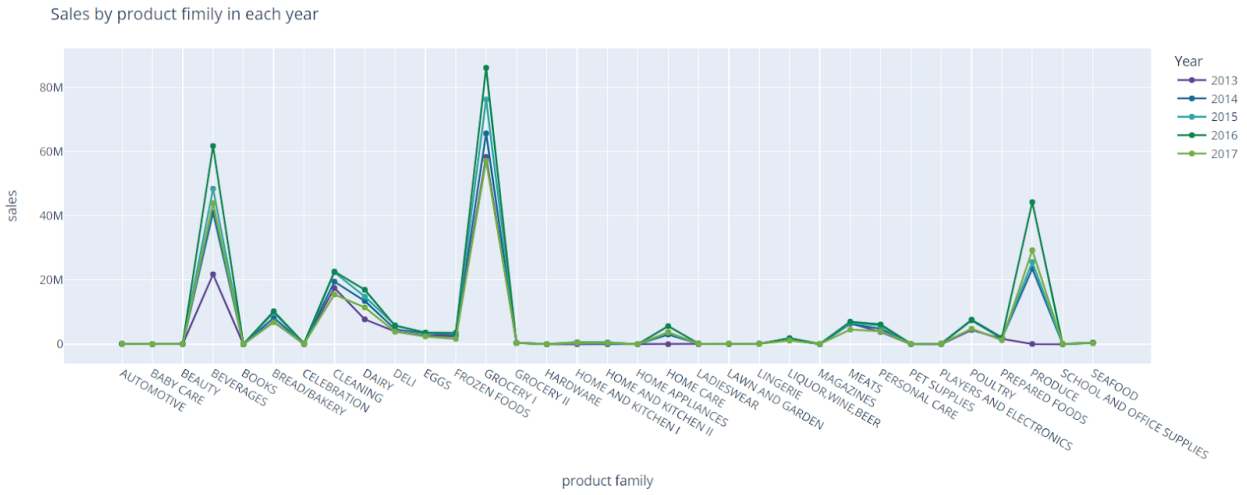
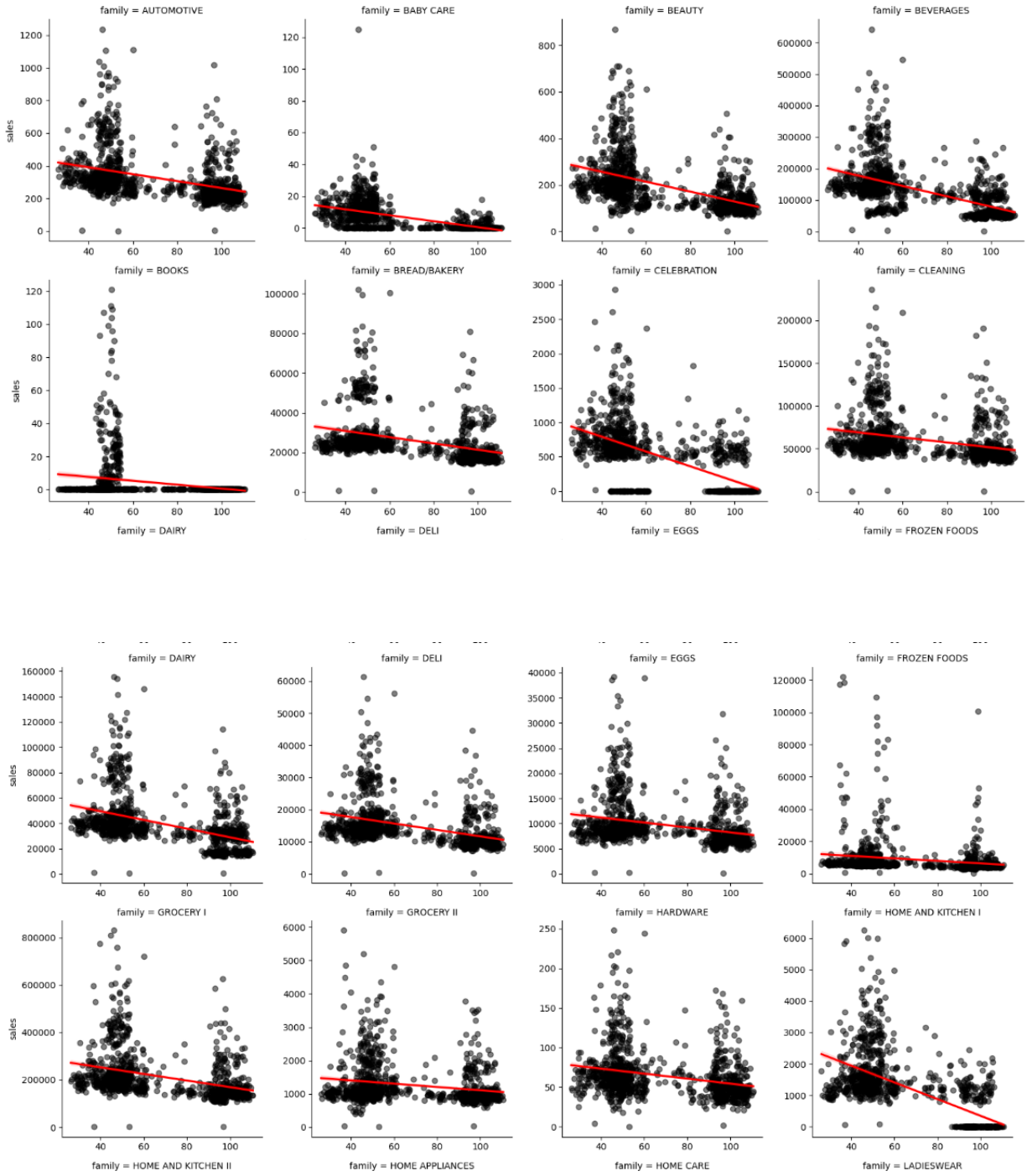


Figure 3.2.6 Line plot showing the overview of the sales of each family product in each year.

Figure 3.2.6 shows that GROCERY 1 is the product family having the highest sales in 2013, as well as in 2014, 2015, 2016, and 2017.


family = AUTOMOTIVE
family = BABY CARE
family = BEAUTY
family = BEVERAGES
family = BOOKS
family = BREAD/BAKERY
family = CELEBRATION
family = CLEANING
family = DAIRY
family = DELI
family = EGGS
family = FROZEN FOODS
family = GROCERY I
family = GROCERY II
family = HARDWARE
family = HOME AND KITCHEN I
family = HOME AND KITCHEN II
family = HOME APPLIANCES
family = HOME CARE
family = LADIESWEAR
sales

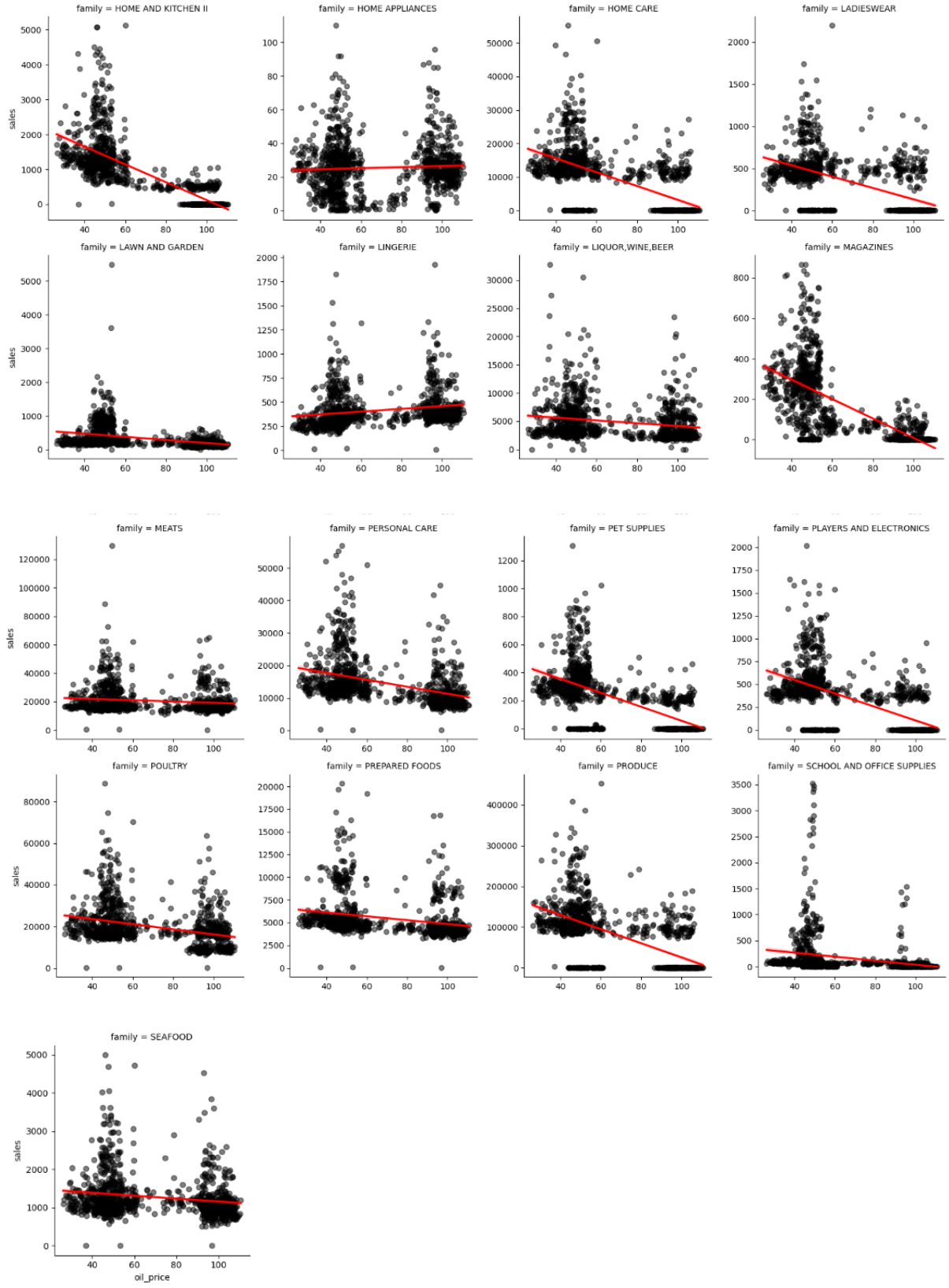

family = HOME AND KITCHEN II
family = HOME APPLIANCES
family = HOME CARE
family = LADIESWEAR
family = LAWN AND GARDEN
family = LINGERIE
family = LIQUOR,WINE,BEER
family = MAGAZINES
family = MEATS
family = PERSONAL CARE
family = PET SUPPLIES
family = PLAYERS AND ELECTRONICS
family = POULTRY
family = PREPARED FOODS
family = PRODUCE
family = SCHOOL AND OFFICE SUPPLIES
family = SEAFOOD
sales
oil_price

Figure 3.2.7 Scatter plot showing the relationship between the oil prices and the sales of each family product.

Figure 3.2.7 shows the relationship between the oil price and the sales in each family product. So we can see that most of the family has a negative correlation with the number of sales, for example, Automotive, Beauty, and Personal Care. However, some of the family also has a positive correlation such as Lingerie. Some of The family has no correlation with the oil price.

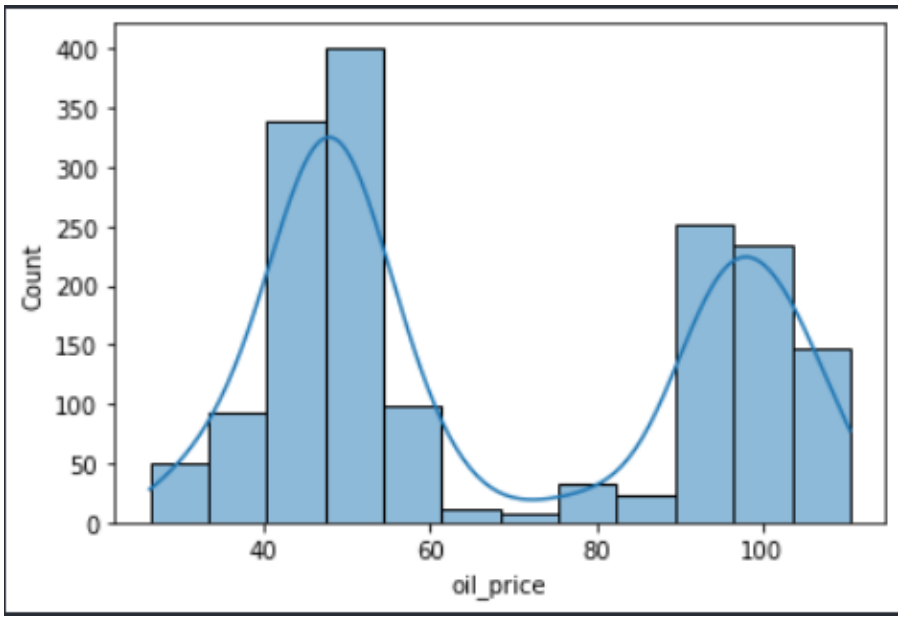


Figure 3.2.7 Distribution of Oil price

Figure 3.2.7 shows the oil price distribution in the period of the dataset, we can see that the oil price is in two ranges.

3.3 Feature Creation

Feature creation is the process of generating new variables, also known as features, from existing data. It involves extracting meaningful information or transforming existing variables to improve the performance of a machine learning model. In this study, we create relevant features to capture the temporal patterns and seasonality in the time series dataset. So we create a time-based feature including day of week, day, month, year and quarter.

	date	store_nbr	family	sales	Year	Month	day	Quarter	weekday	city	state	type	cluster
0	2013-01-01	1	AUTOMOTIVE	0.000	2013	01	01	1	1	Quito	Pichincha	D	13
1	2013-01-01	1	BABY CARE	0.000	2013	01	01	1	1	Quito	Pichincha	D	13
2	2013-01-01	1	BEAUTY	0.000	2013	01	01	1	1	Quito	Pichincha	D	13
3	2013-01-01	1	BEVERAGES	0.000	2013	01	01	1	1	Quito	Pichincha	D	13
4	2013-01-01	1	BOOKS	0.000	2013	01	01	1	1	Quito	Pichincha	D	13
...	...	...	...	...	...	...	...	...	...	...	...	...	...
3000883	2017-08-15	9	POULTRY	438.133	2017	08	15	3	1	Quito	Pichincha	B	6
3000884	2017-08-15	9	PREPARED FOODS	154.553	2017	08	15	3	1	Quito	Pichincha	B	6
3000885	2017-08-15	9	PRODUCE	2419.729	2017	08	15	3	1	Quito	Pichincha	B	6
3000886	2017-08-15	9	SCHOOL AND OFFICE SUPPLIES	121.000	2017	08	15	3	1	Quito	Pichincha	B	6
3000887	2017-08-15	9	SEAFOOD	16.000	2017	08	15	3	1	Quito	Pichincha	B	6

3000888 rows × 13 columns

3.4 Label Encoder

Label encoding is one of data preprocessing techniques that transforms non-numeric variables into numeric values by assigning a distinct integer to each variable category. This facilitates processing of categorical data using machine learning algorithms. In this project, we encoded 5 features including city, state, type, cluster, and holiday_type.

```
encoder = LabelEncoder()
df_train['city']= encoder.fit_transform(df_train['city'])
df_train['state']= encoder.fit_transform(df_train['state'])
df_train['type']= encoder.fit_transform(df_train['type'])
df_train['cluster']= encoder.fit_transform(df_train['cluster'])
df_train['holiday_type']= encoder.fit_transform(df_train['holiday_type'])
```

	date	store_nbr	family	sales	Year	Month	day	Quarter	weekday	city	state	type	cluster	holiday_type	oil_price
1782	2013-01-02	1	AUTOMOTIVE	2.000	2013	01	02	1	2	Quito	Pichincha	D	13	Not_holiday	93.14
1783	2013-01-02	1	BABY CARE	0.000	2013	01	02	1	2	Quito	Pichincha	D	13	Not_holiday	93.14
1784	2013-01-02	1	BEAUTY	2.000	2013	01	02	1	2	Quito	Pichincha	D	13	Not_holiday	93.14
1785	2013-01-02	1	BEVERAGES	1091.000	2013	01	02	1	2	Quito	Pichincha	D	13	Not_holiday	93.14
1786	2013-01-02	1	BOOKS	0.000	2013	01	02	1	2	Quito	Pichincha	D	13	Not_holiday	93.14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3000883	2017-08-15	9	POULTRY	438.133	2017	08	15	3	1	Quito	Pichincha	B	6	Not_holiday	47.57
3000884	2017-08-15	9	PREPARED FOODS	154.553	2017	08	15	3	1	Quito	Pichincha	B	6	Not_holiday	47.57
3000885	2017-08-15	9	PRODUCE	2419.729	2017	08	15	3	1	Quito	Pichincha	B	6	Not_holiday	47.57
3000886	2017-08-15	9	SCHOOL AND OFFICE SUPPLIES	121.000	2017	08	15	3	1	Quito	Pichincha	B	6	Not_holiday	47.57
3000887	2017-08-15	9	SEAFOOD	16.000	2017	08	15	3	1	Quito	Pichincha	B	6	Not_holiday	47.57

Figure 3.4.1 shows the table of train dataset before label encoding

	date	store_nbr	family	sales	Year	Month	day	Quarter	weekday	city	state	type	cluster	holiday_type	oil_price
1782	2013-01-02	1	AUTOMOTIVE	2.000	2013	01	02	1	2	18	12	3	12	4	93.14
1783	2013-01-02	1	BABY CARE	0.000	2013	01	02	1	2	18	12	3	12	4	93.14
1784	2013-01-02	1	BEAUTY	2.000	2013	01	02	1	2	18	12	3	12	4	93.14
1785	2013-01-02	1	BEVERAGES	1091.000	2013	01	02	1	2	18	12	3	12	4	93.14
1786	2013-01-02	1	BOOKS	0.000	2013	01	02	1	2	18	12	3	12	4	93.14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3000883	2017-08-15	9	POULTRY	438.133	2017	08	15	3	1	18	12	1	5	4	47.57
3000884	2017-08-15	9	PREPARED FOODS	154.553	2017	08	15	3	1	18	12	1	5	4	47.57
3000885	2017-08-15	9	PRODUCE	2419.729	2017	08	15	3	1	18	12	1	5	4	47.57
3000886	2017-08-15	9	SCHOOL AND OFFICE SUPPLIES	121.000	2017	08	15	3	1	18	12	1	5	4	47.57
3000887	2017-08-15	9	SEAFOOD	16.000	2017	08	15	3	1	18	12	1	5	4	47.57

Figure 3.4.2 shows the table of train dataset after label encoding

3.5 Data Integration

Data integration is one of the data preprocessing techniques that is used to merge the data present in multiple sources into a single data store. This process commonly involves the collection of data elements from diverse and often unrelated sources, its conversion, and finally its consolidation into a harmonious and consistent dataset.

```
## join stores.csv
df_train = pd.merge(df_train,df_stores,how='left', on='store_nbr')
```

For example, we merged 2 datasets which are train datasets and stored the dataset together into one.

	id	date	store_nbr	family	sales	onpromotion	Year	Month	day	Quarter	weekday
0	0	2013-01-01	1	AUTOMOTIVE	0.000	0	2013	01	01	1	1
1	1	2013-01-01	1	BABY CARE	0.000	0	2013	01	01	1	1
2	2	2013-01-01	1	BEAUTY	0.000	0	2013	01	01	1	1
3	3	2013-01-01	1	BEVERAGES	0.000	0	2013	01	01	1	1
4	4	2013-01-01	1	BOOKS	0.000	0	2013	01	01	1	1
...	...	...	...	...	...	...	...	...	...	...	...
3000883	3000883	2017-08-15	9	POULTRY	438.133	0	2017	08	15	3	1
3000884	3000884	2017-08-15	9	PREPARED FOODS	154.553	1	2017	08	15	3	1
3000885	3000885	2017-08-15	9	PRODUCE	2419.729	148	2017	08	15	3	1
3000886	3000886	2017-08-15	9	SCHOOL AND OFFICE SUPPLIES	121.000	8	2017	08	15	3	1
3000887	3000887	2017-08-15	9	SEAFOOD	16.000	0	2017	08	15	3	1

Figure 3.5.1 shows the table of train dataset before merging

	store_nbr	city	state	type	cluster
0	1	Quito	Pichincha	D	13
1	2	Quito	Pichincha	D	13
2	3	Quito	Pichincha	D	8
3	4	Quito	Pichincha	D	9
4	5	Santo Domingo	Santo Domingo de los Tsachilas	D	4
5	6	Quito	Pichincha	D	13
6	7	Quito	Pichincha	D	8
7	8	Quito	Pichincha	D	8
8	9	Quito	Pichincha	B	6
9	10	Quito	Pichincha	C	15
10	11	Cayambe	Pichincha	B	6

Figure 3.5.2 shows the table of stores dataset before merging

	id	date	store_nbr	family	sales	onpromotion	Year	Month	day	Quarter	weekday	city	state	type	cluster
0	0	2013-01-01	1	AUTOMOTIVE	0.000	0	2013	01	01	1	1	Quito	Pichincha	D	13
1	1	2013-01-01	1	BABY CARE	0.000	0	2013	01	01	1	1	Quito	Pichincha	D	13
2	2	2013-01-01	1	BEAUTY	0.000	0	2013	01	01	1	1	Quito	Pichincha	D	13
3	3	2013-01-01	1	BEVERAGES	0.000	0	2013	01	01	1	1	Quito	Pichincha	D	13
4	4	2013-01-01	1	BOOKS	0.000	0	2013	01	01	1	1	Quito	Pichincha	D	13
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3000883	3000883	2017-08-15	9	POULTRY	438.133	0	2017	08	15	3	1	Quito	Pichincha	B	6
3000884	3000884	2017-08-15	9	PREPARED FOODS	154.553	1	2017	08	15	3	1	Quito	Pichincha	B	6
3000885	3000885	2017-08-15	9	PRODUCE	2419.729	148	2017	08	15	3	1	Quito	Pichincha	B	6
3000886	3000886	2017-08-15	9	SCHOOL AND OFFICE SUPPLIES	121.000	8	2017	08	15	3	1	Quito	Pichincha	B	6
3000887	3000887	2017-08-15	9	SEAFOOD	16.000	0	2017	08	15	3	1	Quito	Pichincha	B	6

Figure 3.5.3 shows the table after merging

3.6 Data Scaling

To scale the value in the oil price feature down, we used MaxMinScaler. To reduce the scale of the data in the column into range[0,1].

The applied formula:

$$x\_scaled = (x-min) / (max-min)$$

3.7 Train and Test data splitting

In this dataset, we splitted the data into 2 parts. The first part is the train dataset. We used data in the range from 2013-01-01 to 2017-07-31 as a train dataset. The second part is the test

dataset, we used the range of data from 2017-08-01 to 2017-08-15. We only attempt a 15-day prediction.

```python
def spilt_train_data(df):

    df_use = df.copy()

    train_data = df_use[df_use.index <= '2017-07-31']
    test_data = df_use[df_use.index > '2017-07-31']

    X_train = train_data.loc[:, df_use.columns != 'sales']
    y_train = train_data['sales']

    X_test = test_data.loc[:, df_use.columns != 'sales']
    y_test = test_data['sales']

    return X_train,y_train,X_test,y_test
```

4. Models

4.1 XGBoost Regression

XGBoosting Regression is an optimized implementation of the gradient boosting algorithm, which is a type of ensemble learning technique that combines multiple weak models to form a strong model. It is a scalable end-to-end tree boosting system. XGBoosting works by combining a number of weak learners to form a strong learner .

In XG Boosting, a series of decision trees are built sequentially, with each tree attempting to correct the errors of the previous tree. The algorithm calculates the gradients of the loss function with respect to the model's predictions, and then uses these gradients to update the model's parameters, in this case the weights assigned to the different features. This process is repeated for a fixed number of iterations or until the loss function converges to a minimum value. XG Boosting has several advantages over other machine learning algorithms. It is highly scalable, can handle both regression and classification problems, and has been shown to produce highly accurate predictions on a wide range of datasets. Additionally, XG Boosting can handle missing data and can automatically learn the importance of each feature in the dataset.

XGBoosting is one of the most popular and widely used machine learning algorithms due to its ability to handle large datasets and its ability to achieve state-of-the-art performance in many machine learning tasks such as classification and regression.

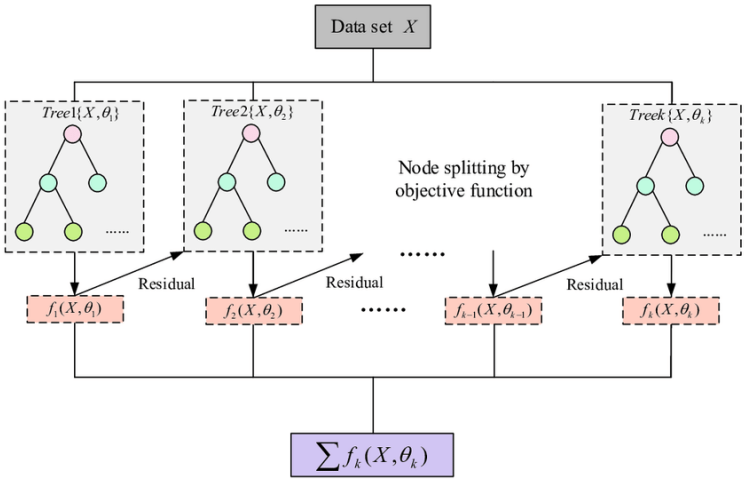


Figure 4.1.1 XGBoosting model

Parameter Setting for XGBOOSTING

Below are the parameter settings for the XGBoosting machine learning algorithm when applied to the project dataset. The base score was set at 0.5, the total number of estimators is 2000, the maximum depth is 5 and finally the learning rate for the algorithm is 0.05.

Base_score = 0.5

n_estimators = 2000

max_depth = 5

learning_rate = 0.05

```
def process_xgboost(df):

    ## create the model
    reg = xgb.XGBRegressor(base_score=0.5, booster='gbtree',
                           n_estimators=2000,
                           max_depth=5,
                           learning_rate=0.05)
    ## spilt the data
    X_train,y_train,X_test,y_test = spilt_train_data(df)


    ## fit the model
    reg.fit(X_train,y_train)


    ## predict test
    y_pred = reg.predict(X_test)

    #if the value is negative convert to 0
    y_pred = list(y_pred)
    y_pred = [max(0, x) for x in y_pred]

    return y_pred , y_test, reg
```

Figure 4.1.2 Parameter settings for the XGBoosting machine learning algorithm

4.2 Random Forest

Random Forest is an ensemble learning technique that combines multiple decision trees to improve the performance of a machine learning model. Random Forest developed by Leo Breiman is a group of un-pruned classification or regression trees made from the random selection of samples of the training data.

In Random Forest, a collection of decision trees are built using a random subset of the features in the dataset. The algorithm then aggregates the predictions of each decision tree to arrive at a final prediction. This approach helps to reduce overfitting and improve the generalization of the model. Each decision tree in a Random Forest is trained using a different subset of the data, which helps to reduce the variance of the model. Additionally, the algorithm

can handle missing data and can automatically learn the importance of each feature in the dataset. Random Forest has several advantages over other machine learning algorithms. It can handle a large number of input variables and can handle non-linear relationships between features and outcomes. Additionally, Random Forest is computationally efficient and can be trained on large datasets.

Random features are selected in the induction process. Prediction is made by aggregating (majority vote for classification or averaging for regression) the predictions of the ensemble.

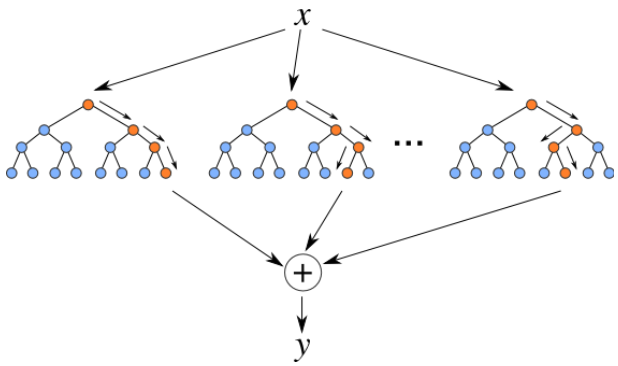


Figure 4.2.1 Random Forest

Parameter Setting for Random Forest

Below are the parameter settings for the random forest machine learning algorithm in application to the project dataset. The total number of estimators are 300 and the maximum depth of 5.

n_estimators = 300

Max_depth = 5

```
def process_random_forest(df):

    ## create the model
    reg = RandomForestRegressor(n_estimators = 300, max_features = 'sqrt', max_depth = 5, random_state = 18)
    ## spilt the data
    X_train,y_train,X_test,y_test = spilt_train_data(df)

    ## fit the model
    reg.fit(X_train,y_train)

    ## predict test
    y_pred = reg.predict(X_test)

    #if the value is negative convert to 0
    y_pred = list(y_pred)
    y_pred = [max(0, x) for x in y_pred]

    return y_pred , y_test , reg
```

Figure 4.2.2 Parameter settings for the random forest machine learning algorithm.

5. Result

5.1 Evaluation Metric

The evaluation metric that was used for this project is the root mean squared logarithmic Error (RMSLE). RMSLE is a measure utilized to assess the efficacy of regression models, specifically in cases where the target variable varies significantly in value. This metric gauges the mean difference between the logarithmic values of the forecasted and actual outcomes. The RMSLE can be calculated using the following equation:

RMSLE = sqrt(1/n*sum((log(y_pred+1) - log(y_true+1))^2))

Where: n = number of observations in the dataset.

Y_pred = the predicted value

Y_true = the actual value

Another metric that was used to measure and evaluate the model is R- squared (R^2). R-squared or coefficient of determination is a statistical measure that assesses the degree to which a linear regression model fits the data points. This metric indicates how much of the dependent variable's variance can be accounted for by the independent variables in the model. The R-squared value ranges between 0 and 1 where 0 indicates that the model does not explain any of the variance in the dependent variable while 1 indicates that the model explains all of the variance in the dependent variable. R-squared can be calculated using the following equation:

R-squared = 1 - (SSR/SST)

Where SSR = sum of the squared differences between the predicted and the actual values

SST = sum of the squared differences between the actual values and the mean of the dependent variable.

store_nbr	family	sales	Year	Month	Quarter	weekday	city	state	type	cluster	oil_price
1	AUTOMOTIVE	2.000	2013	01	1	2	18	12	3	12	0.792965
1	BABY CARE	0.000	2013	01	1	2	18	12	3	12	0.792965
1	BEAUTY	2.000	2013	01	1	2	18	12	3	12	0.792965
1	BEVERAGES	1091.000	2013	01	1	2	18	12	3	12	0.792965
1	BOOKS	0.000	2013	01	1	2	18	12	3	12	0.792965
...	...	...	...	...	...	...	...	...	...	...	...
9	POULTRY	438.133	2017	08	3	1	18	12	1	5	0.253228
9	PREPARED FOODS	154.553	2017	08	3	1	18	12	1	5	0.253228
9	PRODUCE	2419.729	2017	08	3	1	18	12	1	5	0.253228
9	SCHOOL AND OFFICE SUPPLIES	121.000	2017	08	3	1	18	12	1	5	0.253228
9	SEAFOOD	16.000	2017	08	3	1	18	12	1	5	0.253228

Figure 5.1 Final Data was used to train the model

Figure 5.1 shows the final data from data preprocessing that was used in training the model. The data was applied to the XGBoosting model. The data is split to each family type and then the model is trained on each family type.

```
family_test = ['GROCERY I', 'GROCERY II', 'HARDWARE',
       'HOME AND KITCHEN I', 'HOME AND KITCHEN II', 'HOME APPLIANCES']


for family in family_test:

    family_name = family

    df_temp = pd.read_csv('/kaggle/working/'+str(family_name)+'.csv')
    df_temp.set_index('date',inplace=True)
    df_temp.drop(['family'],axis=1,inplace= True)
    y_pred_solution, y_test_solution, reg = process_xgboost(df_temp)

    testing = df_temp[df_temp.index > '2017-07-31'].copy()
    plot_sales_total_predicted(testing.copy(),y_pred_solution)
    evaluation(y_pred_solution,y_test_solution)
```

Figure 5.2 testing

5.2 XGBoosting

Due to time constraints and our scope of work is quite large, so our group have made a decision to focus only on the 7 family types which are PET, GROCERY I, GROCERY II, HARDWARE, HOME AND KITCHEN I, HOME AND KITCHEN II and HOME APPLIANCES.

RMSLE and R square were used to test the performance of the model. The results can be found below. Note that the red line is predicted sales and the blue line is actual sales.

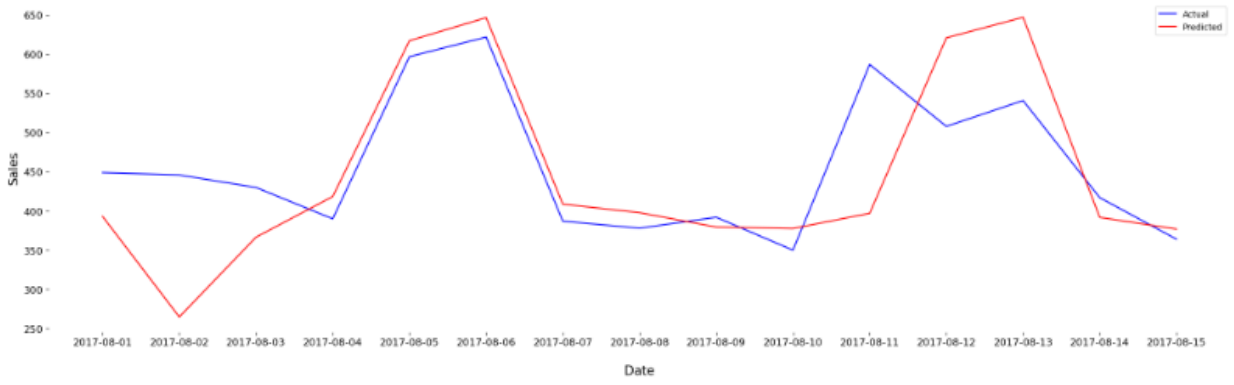


Figure 5.2.1 : Predicted Sale on PET Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting algorithm on PET family type. The accuracy measured by RMSLE is 0.492 and R squared is equal to 0.788.

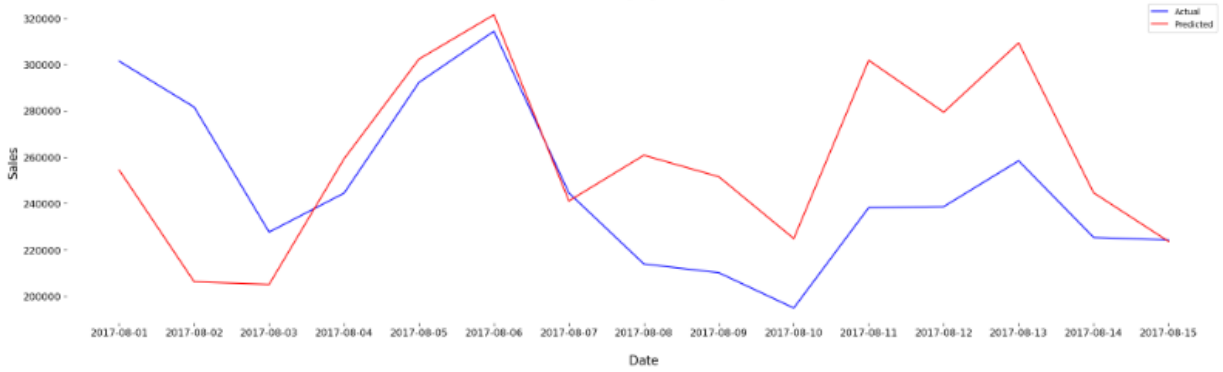


Figure 5.2.2 : Predicted Sale on GROCERY I Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting algorithm on GROCERY I family type. The accuracy measured by RMSLE is 0.235 and R squared is equal to 0.779.

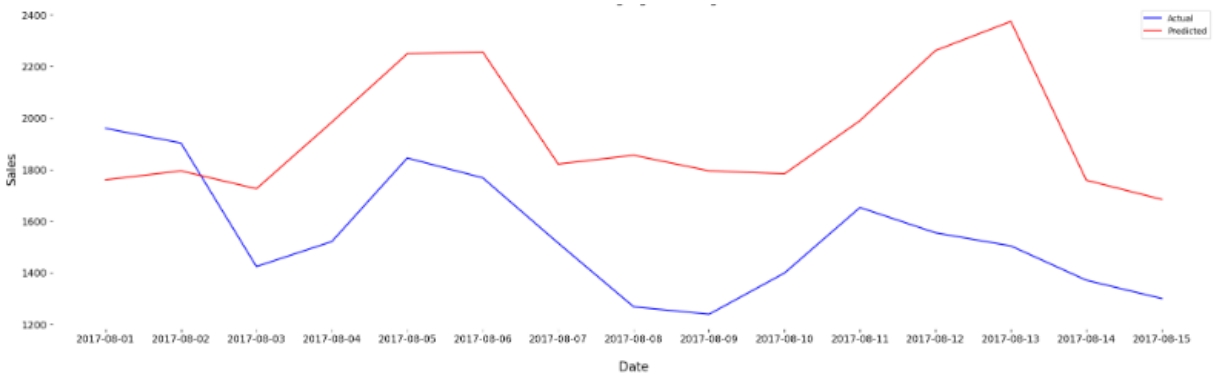


Figure 5.2.3 : Predicted Sale on GROCERY II Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting algorithm on GROCERY II family type. The accuracy measured by RMSLE is 0.730 and R squared is equal to 0.500.

Figure 5.2.4: Predicted Sale on HARDWARE Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting algorithm on Hardware family type. The accuracy measured by RMSLE is 0.565 and R squared is equal to 0.281.

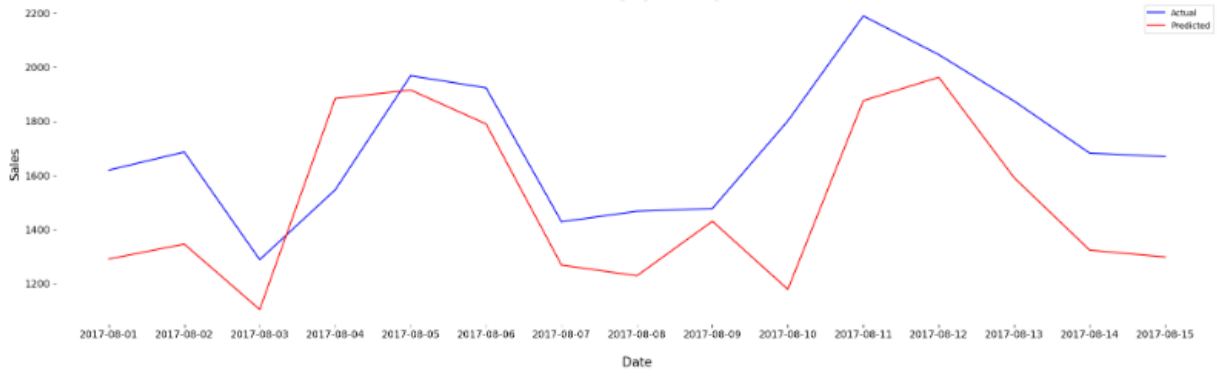


Figure 5.2.5 : Predicted Sale on HOME AND KITCHEN I Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting algorithm on HOME AND KITCHEN family type. The accuracy measured by RMSLE is 0.534 and R squared is equal to 0.505.

Figure 5.2.6: Predicted Sale on HOME AND KITCHEN II Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting on HOME AND KITCHEN II family type. The accuracy measured by RMSLE is 0.556 and R squared is equal to 0.346.

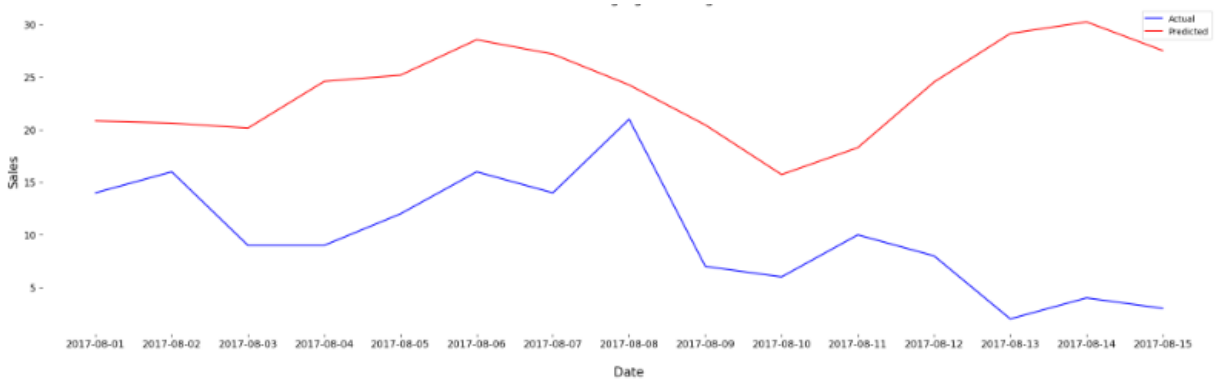


Figure 5.2.7: Predicted Sale on HOME APPLIANCES Family Type by using XGBoosting

This Figure shows the predicted result after applying XGBoosting on HOME AND KITCHEN II family type. The accuracy measured by RMSLE is 0.404 and R squared is equal to -0.556.

Family Type	RMSLE	R square
PET	0.492	0.788
GROCERY I	0.235	0.779
GROCERY II	0.730	0.500
HARDWARE	0.565	0.281
HOME AND KITCHEN I	0.534	0.505
HOME AND KITCHEN II	0.556	0.346
HOME APPLIANCES	0.404	-0.556

5.3 Random Forest

Similar to the XGBoosting model, 7 family types were selected for testing in the Random Forest model.

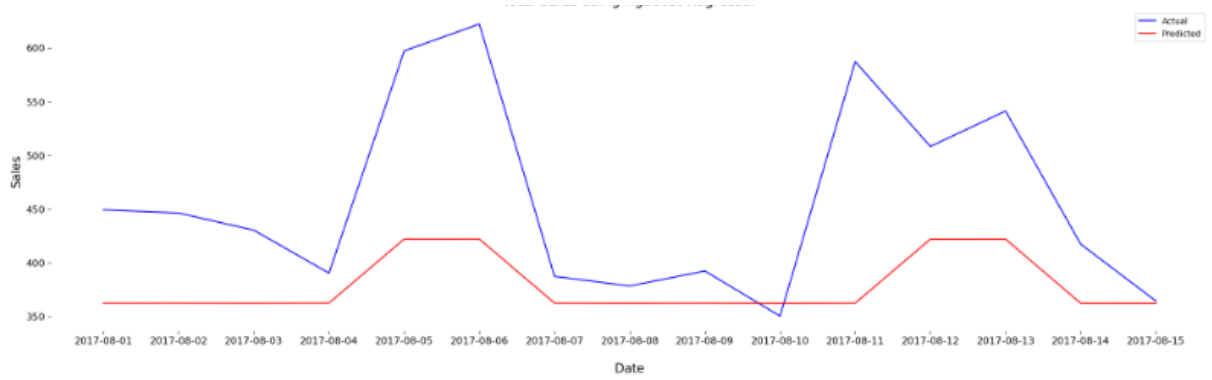


Figure 5.3.8 : Predicted Sale on PET Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on PET family type. The accuracy measured by RMSLE is 0.612 and R squared is equal to 0.660.

Figure 5.3.9: Predicted Sale on GROCERY I Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on GROCERY I family type. The accuracy measured by RMSLE is 0.365 and R squared is equal to 0.609.

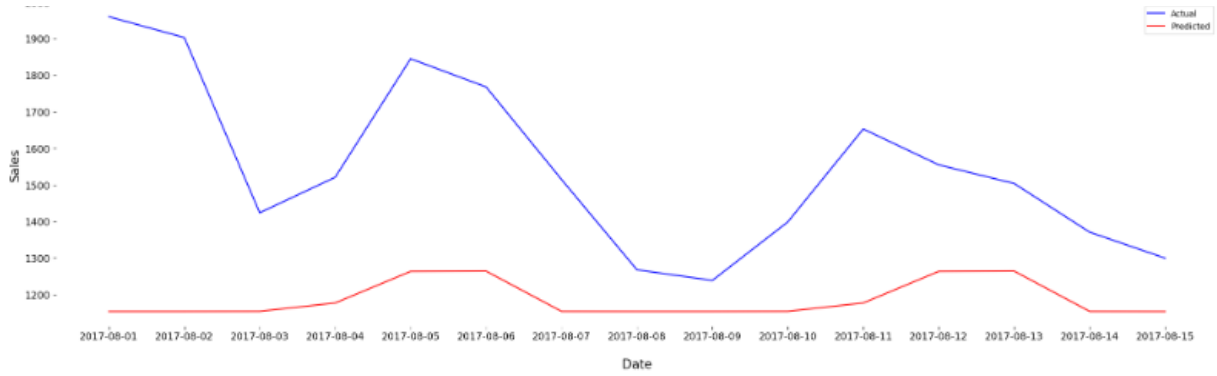


Figure 5.3.10 : Predicted Sale on GROCERY II Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on GROCERY II family type. The accuracy measured by RMSLE is 0.682 and R squared is equal to 0.546.

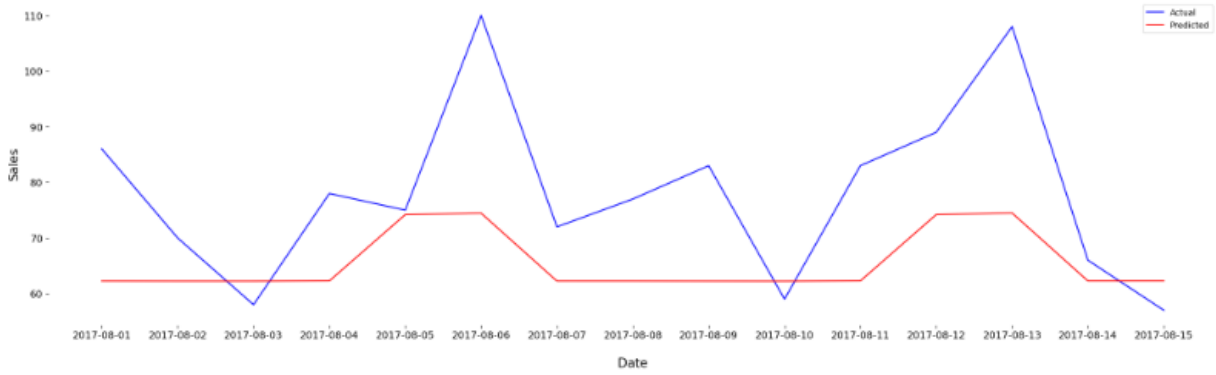


Figure 5.3.11: Predicted Sale on HARDWARE Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on HARDWARE family type. The accuracy measured by RMSLE is 0.585 and R squared is equal to 0.136.

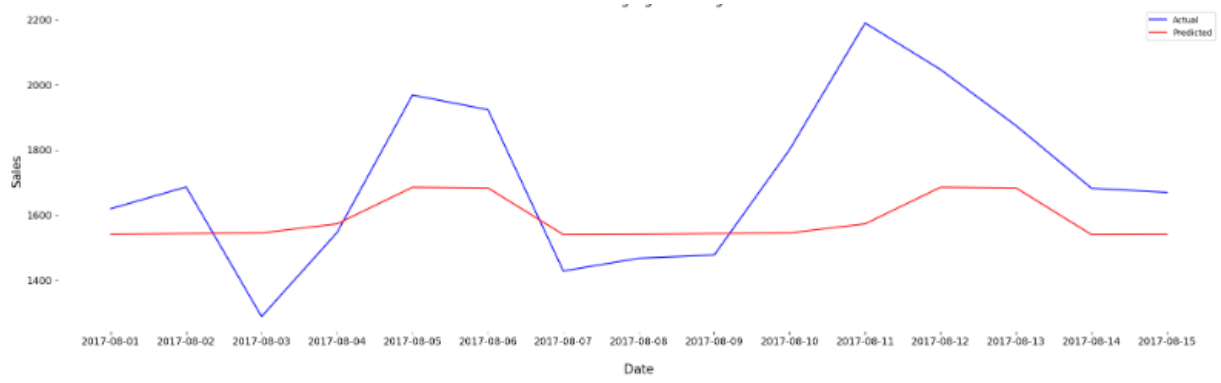


Figure 5.3.12: Predicted Sale on HOME AND KITCHEN I Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on PET family type. The accuracy measured by RMSLE is 0.605 and R squared is equal to 0.434.

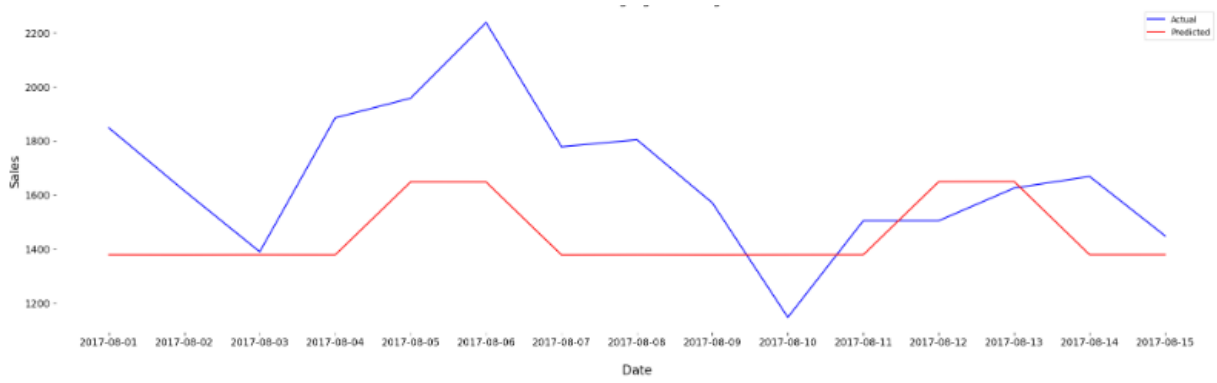


Figure 5.3.13: Predicted Sale on HOME AND KITCHEN II Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on PET family type. The accuracy measured by RMSLE is 0.631 and R squared is equal to 0.269.

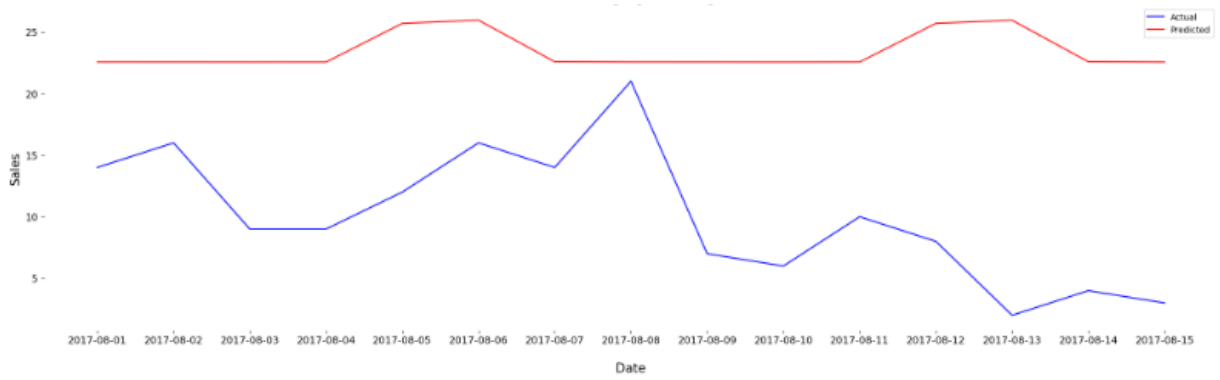


Figure 5.3.14: Predicted Sale on HOME APPLIANCES Family Type by using Random Forest

This Figure shows the predicted result after applying Random Forest on HOME APPLIANCES family type. The accuracy measured by RMSLE is 0.384 and R squared is equal to -0.23.

Family Type	**RMSLE**	**R square**
PET	0.612	0.660
GROCERY I	0.365	0.609
GROCERY II	0.682	0.546
HARDWARE	0.585	0.136
HOME AND KITCHEN I	0.605	0.434
HOME AND KITCHEN II	0.631	0.269
HOME APPLIANCES	0.384	-0.23

6. Comparison and Discussion

6.1 Comparison of the performance of XGBoosting and Random forest machine learning models.

Table 6.1.1 shows comparison accuracy of RMSLE between XGBoosting and Random Forest

Family Type	RMSLE XGboosting	RMSLE Random Forest
PET	0.492	0.612
GROCERY I	0.235	0.365
GROCERY II	0.730	0.682
HARDWARE	0.565	0.585
HOME AND KITCHEN I	0.534	0.605
HOME AND KITCHEN II	0.556	0.631
HOME APPLIANCES	0.404	0.384

From the result of testing based on the RMSLE Perspective, XGboosting has a better performance than Random forest.

For the majority of the family types, the RMSLE score for the XGBoosting model is lower than the score for the Random forest model which means XGBoosting has better performance.

Analyzing the results obtained for both models using the root mean squared logarithmic error (RMSLE) evaluation metric shows that the XGBoosting machine learning model has better accuracy than the Random forest model, and hence better performance.

Table 6.1.2 Show comparison accuracy of R^2 between XGBoosting and Random Forest

Family Type	R^2 XGboosting	R^2 Random Forest
PET	0.788	0.660
GROCERY I	0.779	0.609
GROCERY II	0.500	0.546
HARDWARE	0.281	0.136

HOME AND KITCHEN I	0.505	0.434
HOME AND KITCHEN II	0.346	0.269
HOME APPLIANCES	-0.556	-0.23

From the result of testing based on the R^2 Perspective, XGboosting has a better performance than Random forest.

On the other hand, when R-squared is used as the evaluation metric for both models, since the R-squared score for the XGBoosting model is higher than the R-squared score for the Random forest model when looking at majority of the family types, it is clearly discernible that XGBoosting has better accuracy than Random forest and therefore better performance.

6.3 Discussion

Significant insights from both machine learning models in terms of evaluation show that overall, XGBoosting is better than Random forest in terms of accuracy of the fact that XGBoosting rates higher on the two evaluation metrics that were selected. Nevertheless, when it comes to this project, it was discovered that both models, XGBoosting and Random Forest have low accuracy and performance on the RMSLE and R-squared evaluation metrics. The reason might be that a low number of features was used in the data preprocessing method and this hinders the model in terms of understanding and analyzing the cycle patterns in each weekday and month. Additionally, there might be other factors or unexpected factors which they did not provide any information on dataset that can affect sales for each family type. To enhance the effectiveness and performance of the model, it is necessary to generate new attributes or features like Lag sale or other elements that can help the model discern the pattern more accurately.

7. References

1. XGBoost: A Scalable Tree Boosting System, Tianqi Chen , 10 Jun 2016 https://arxiv.org/pdf/1603.02754.pdf
2. Breiman, L., Random Forests, Machine Learning 45(1), 5-32, 2001.
3. Gradient boosting regression (no date) scikit. Available at: https://scikit-learn.org/stable/auto_examples/ensemble/plot_gradient_boosting_regression.html (Accessed: May 8, 2023).
4. Brownlee, J. (2021, March 6). XGBoost for regression. MachineLearningMastery.com. Retrieved May 8, 2023, from https://machinelearningmastery.com/xgboost-for-regression/
5. XGBoost Documentation — xgboost 1.7.5 documentation. (n.d.). XGBoost Documentation &Mdash; Xgboost 1.7.5 Documentation. https://xgboost.readthedocs.io/en/stable/
6. Verma, N. (2022, September 7). XGBoost Algorithm Explained in Less Than 5 Minutes. Medium. https://medium.com/@techynilesh/xgboost-algorithm-explained-in-less-than-5-minutes-b561dcc1ccee
7. Rao, S. (2023, March 3). XGBoost Regression: Explain It To Me Like I’m 10. Medium. https://towardsdatascience.com/xgboost-regression-explain-it-to-me-like-im-10-2cf324b0bbdb
8. Citation for the dataset: https://www.kaggle.com/competitions/store-sales-time-series-forecasting/data