Datastorm 5.0

Report Submission
by

Beast Trio

Team Members

- Usitha Jayaweera
- Savindu Rajapaksha
- Sandaruth Siriwardana

Github Repository Link : https://github.com/UsithaDJay/Datastorm5.0.git

Content

1. Introduction
   - Task
2. Methodology
   - Data Exploration
   - Data Preprocessing
   - Feature Engineering
   - Model Selection and Training
3. Enhancing Marketing Strategies
   - Strategic Implication of Customer Segmentation
   - Marketing Communication and Product Placement
   - Leveraging Data for Future Campaigns
   - Conclusion

Introduction

The "Data Storm 5.0" competition, organized by the UOM and Faculty of Science UOC, challenges participants to develop sophisticated analytical methods to enhance marketing strategies through personalized customer segmentation. The primary objective of this competition is to leverage historical sales data provided by KJ Marketing to identify distinct groups of similar customers, thus enabling the creation of tailored marketing strategies that are more effective than traditional, one-size-fits-all approaches.

KJ Marketing is a leading retail Supermarket chain in Sri Lanka with 22 outlets throughout the country in both urban and suburban areas. The Company offers a wide range of products for customers,including dry goods, Fresh Items and Luxury Items. They target the market for the end customers.

Over the past few years they have identified that usage of conventional and standard marketing strategies has not been much effective to their current prevailing market. Therefore they have decided to approach using a different marketing strategy to increase their sales. In this attempt they have decided to adopt a personalized marketing strategy after taking individual customer preferences into consideration.

Task

In this problem we are tasked to develop a comprehensive analytic method to identify small groups of customers with similar preferences when it comes to buying goods from KJ outlets. Also we need to provide a CSV file classifying the new customers into relevant segments. Through an initial analysis we were provided with 6 customer segments and our task is to classify the new customers into relevant segments

Methodology

Data Exploration

Using ‘Pandas’ library we have extracted the relevant dataset into Google Colab notebook. In the training dataset there are 774155 records of each customer and they have been classified into 6 categories. In the testing dataset there are 40749 records.

Before proceeding to the testing dataset we checked the null values in the training dataset as the initial step. Here are listed our initial findings and the abnormalities in the dataset

1. Null values
    - Customer_ID : 2
    - outlet_city: 2
    - luxury_sales: 35
    - fresh_sales: 41
    - dry_sales: 30
    - cluster_catgeory: 1

In the testing dataset there were no null values

2. Duplicate values
    - There were no duplicate Customer_ID's except for null values.

3. Intersections in Testing and Training Dataset
    - There were no intersections here. It was confirmed by absence of common Customer_ID's in both training and testing dataset.

4. Invalid Values
    - Customer_ID: None
    - outlet_city: None (Inconsistent Category Names)
    - Luxury_sales: All the values are None negative (Inconsistent Data types)
    - Fresh Sales: All the values are None negative (Inconsistent Data types)
    - Dry_sales: All the values are None negative (Inconsistent Data types)
    - Cluster_category: There were invalid values. Also data types of these values were not consistent.

Data Preprocessing

Data Cleaning

1. Sales Columns (Fresh, Dry and Luxury)

    There were data type inconsistencies in the sales columns. First we converted the convertible values from String to Integer.

    Invalid Values (which raised ValueError) were later handled individually, as the example stated below.

```
# Mapping dictionary for textual representations of numbers to their numeric equivalents
text_to_numeric_fresh = {
    'nul': 0,
    'Six hundread and five ruppes': 605,
    'Three thousana and five hundread': 3500,
    'thirteen thousand ruppes': 13000,
    'Five thousand ruppes': 500,
    'Two thousand seven hundread ruppess': 2700
}

# Convert textual representations of numbers into numeric values
train_df['fresh_sales'] = train_df['fresh_sales'].replace(text_to_numeric_fresh)

# Filter the fresh_sales column for string type values
str_values_fresh_sales_train = train_df['fresh_sales'][train_df['fresh_sales'].apply(lambda x: isinstance(x, str))]

# Print unique string type values
print("Unique string type values in fresh_sales column:", str_values_fresh_sales_train.unique())
✓ 0.5s
```

    Ultimately we managed to get a Fresh_sales column with consistent data types and valid values. Both other sales columns (Dry and Luxury items sales columns went through the same process).

2. Outlet City Column

   We checked the values in both the Testing and Training dataset for cleaning. From that we identified there were abnormalities in the values in the Test Dataset.

   Unique values in outlet_city column in Train Dataset:

   ['Kelaniya', 'Moratuwa', 'Wattala', 'Homagama', 'Dehiwala-Mount Lavinia', 'Panadura', 'Kaduwela', 'Peliyagoda', 'Kotte', 'Nuwara Eliya', 'Batticaloa', 'Colombo', 'Jaffna', 'Gampaha', 'Kalmunai', 'Galle', 'Katunayake', 'Negombo', 'Trincomalee', 'Kandy', nan]

   Unique values in outlet_city column in Test Dataset:

   ['batticaloa', 'Batticaloa', 'Colombo', 'Dehiwala-Mount Lavinia', 'Anuradhapura', 'Galle', 'Gampaha', 'Homagama', 'Jaffna', 'Kaduwela', 'Kalmunai', 'kalmunai', 'Kandy', 'Katunayake', 'Kelaniya', 'Madawachiya', 'Kotte', 'Moratuwa', 'MoraTuwa', 'Negombo', 'Nuwara Eliya', 'Panadura', 'Peliyagoda', 'PeliyagodA', 'Trincomale', 'Trincomalee', 'Wattala']

   After checking both unique values we checked the difference between each of the above sets and used it to fix the typos in outlet_city column.

3. Cluster Category Column

   In the Category Column there were type inconsistencies as well as invalid values.

```
Unique values in cluster_category column: ['4' '1' '99' '2' '5' '3' '6' '6\\' 4 2 1 95 3 98 5 6 nan 100.0 89.0]
```

   To address these type inconsistencies we changed the type of the values from string to Int. Then the string values which gave ValueError were handled explicitly using replace.

```
# Fix typo in the cluster_category column
train_df['cluster_category'] = train_df['cluster_category'].replace({"6\\": 6})
```

   In the problem statement there were 6 categories. So other invalid values were kept as null for the time being. So after this step there were 6 null values in the category column (Transformations of invalid values as null values to category column included)

Handling Missing values

Steps and Reasoning

1. Filling MIssing Sales Data with zero

   For sales columns current missing values which were kept as ‘null’ or ‘NaN’ were replaced using 0. We took that decision because there were no 0 values in the records of these 3 sales columns in the original dataset, so we decided that the values which are indicated as null are actually there to indicate that there are no sales. So we imputed null values with 0.

We did this because,

- Sales columns would be very crucial in the final decision making process.
- It allows us us to maintain a consistent data size for analysis
- It reduces the possibility of being bias due to removal of these data especially if data is not missing completely randomly

2. Handling missing Customer Ids

   Since using a completely random value is okay if it is unique for Customer_ID we've decided to use a range from existing customer_ID's to missing sum of customer_ID's.

```
# Find rows with missing Customer_ID
missing_customer_id = train_df['Customer_ID'].isnull()

# Generate unique IDs for missing values
new_ids = range(10774155, 10774155 + missing_customer_id.sum())

# Assign new IDs to missing values
train_df.loc[missing_customer_id, 'Customer_ID'] = new_ids
```

   This approach gives a unique identifier for each rows with missing customer_ID's

3. Dropping rows with missing Cluster Category data.

   There were 6 missing values in the "cluster_category" column. We decided to drop the rows with missing values for this column, because we have no use in training data where target variable data is missing. Also 6 out of 774155 values is not going to make much difference in the decision making process.

Outlier Handling

Presence of outliers in the dataset can significantly affect the performance of the predictive models, especially in extreme cases. So we have identified the outliers in Sales columns and handled them accordingly.

Outlier Detection

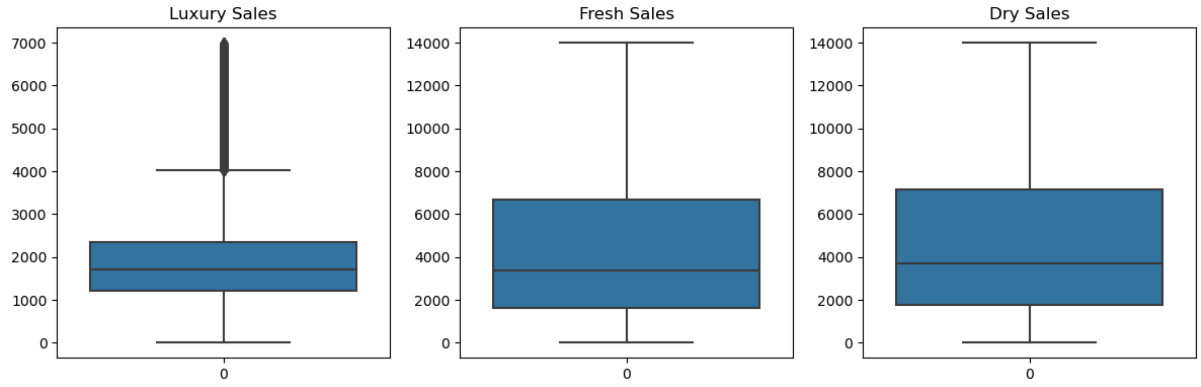
To detect outliers, we employed the ***Interquartile Range (IQR)*** method, a statistical technique that measures the statistical spread of the data. The IQR is the difference between the 75th percentile (Q3) and the 25th percentile (Q1) of the data.

Outliers were defined as observations that,

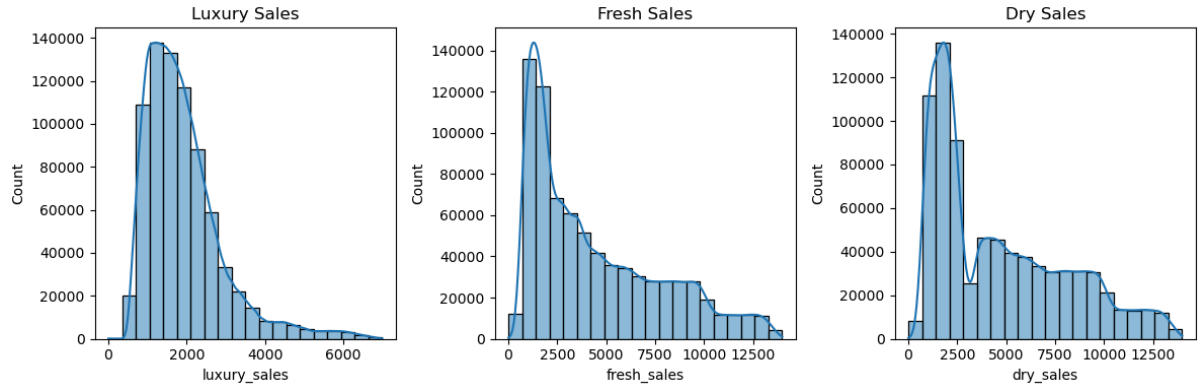
- fall below **Q1 - 1.5 * IQR** or above **Q3 + 1.5 * IQR**.

This method is particularly robust for datasets with skewed distributions, as it does not rely on mean and standard deviation, which can be unduly influenced by outliers.

From this we identified that there were outliers only in the luxury Items columns. Check the following figures.



Long tails in the histogram of Luxury Sales indicate the outliers in that column.



So this discrepancy highlights the different purchasing patterns across product categories and underscores the importance of tailored analytical approaches for each category.

Handling Strategy

There were no outliers with regard to “Fresh Sales” and “Dry Sales” data.

Also there were considerably large numbers of outliers in “Luxury Sales” data. So we decided that outliers which show in the “Luxury Sales” column are not actually outliers and considered them as real data.

So we continued our decision making process without handling outliers.

Feature Engineering

Feature engineering is a critical process in data science that involves creating new variables or modifying existing variables to improve the predictive power of the models. For our dataset, several feature engineering steps were undertaken to better capture the nuances of our data and enhance model performance.

Adding new features

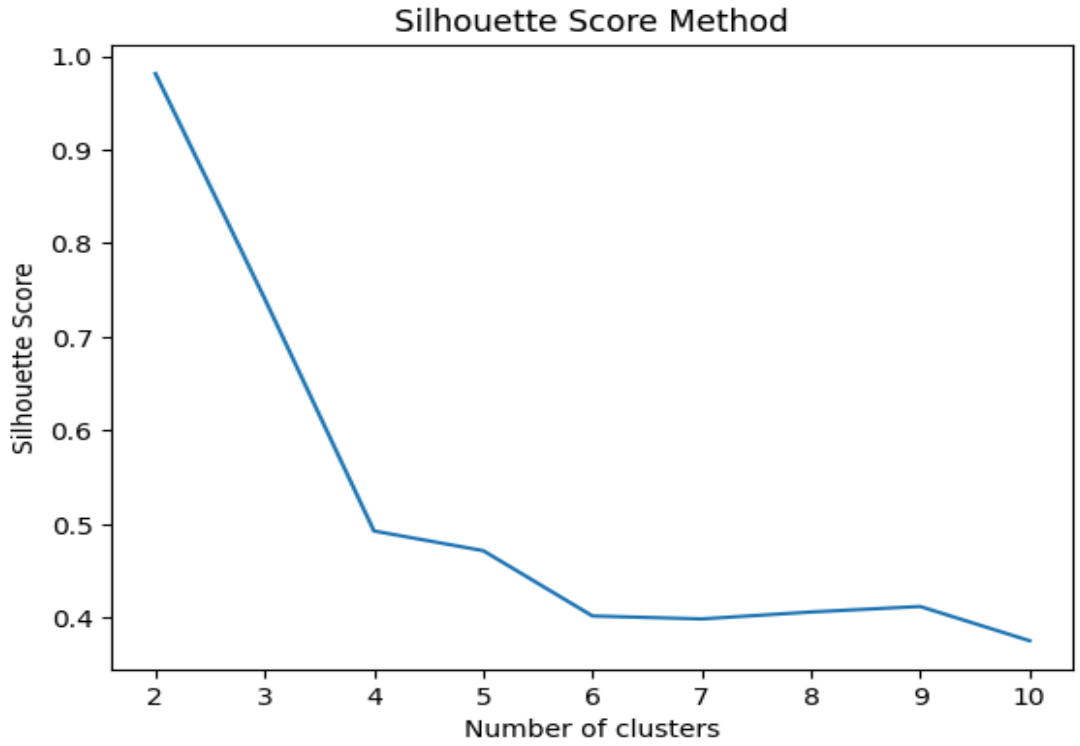
Adding a City Cluster Column

We have decided to go through this approach because Madawachchiya and Anuradhapura were present in the testing dataset but didn't exist in the training dataset.

```
Different outlet_citys between train and test DataFrames: {'Anuradhapura', 'Madawachchiya',}
```

To create a connection for this in the dataset we've decided to create clusters based on the city. We expect to have 2-3 clusters because of the existence of urban, suburban and rural areas in the outlet_city.

To check this properly we used the Silhouette Score Method(SSM) to get a confirmed number of clusters.



In the above diagram SS for 2 clusters is much higher than other cluster numbers. Gradually, when increasing the number of clusters SS decreases. Therefore we've decided to go with **two clusters.**

Then using K-means clustering by keeping the number of clusters = 2, We created the following two clusters for cities.

```
Unique cities under Cluster 0: ['Dehiwala-Mount Lavinia' 'Homagama' 'Kaduwela' 'Kelaniya' 'Kotte'
 'Madawachchiya' 'Moratuwa' 'Nuwara Eliya' 'Panadura' 'Peliyagoda'
 'Wattala']
Unique cities under Cluster 1: ['Anuradhapura' 'Batticaloa' 'Colombo' 'Galle' 'Gampaha' 'Jaffna'
 'Kalmunai' 'Kandy' 'Katunayake' 'Negombo' 'Trincomalee']
```

After adding the new feature, **city_clusters** we've handled the inconsistencies in data types.

One-Hot Encoding

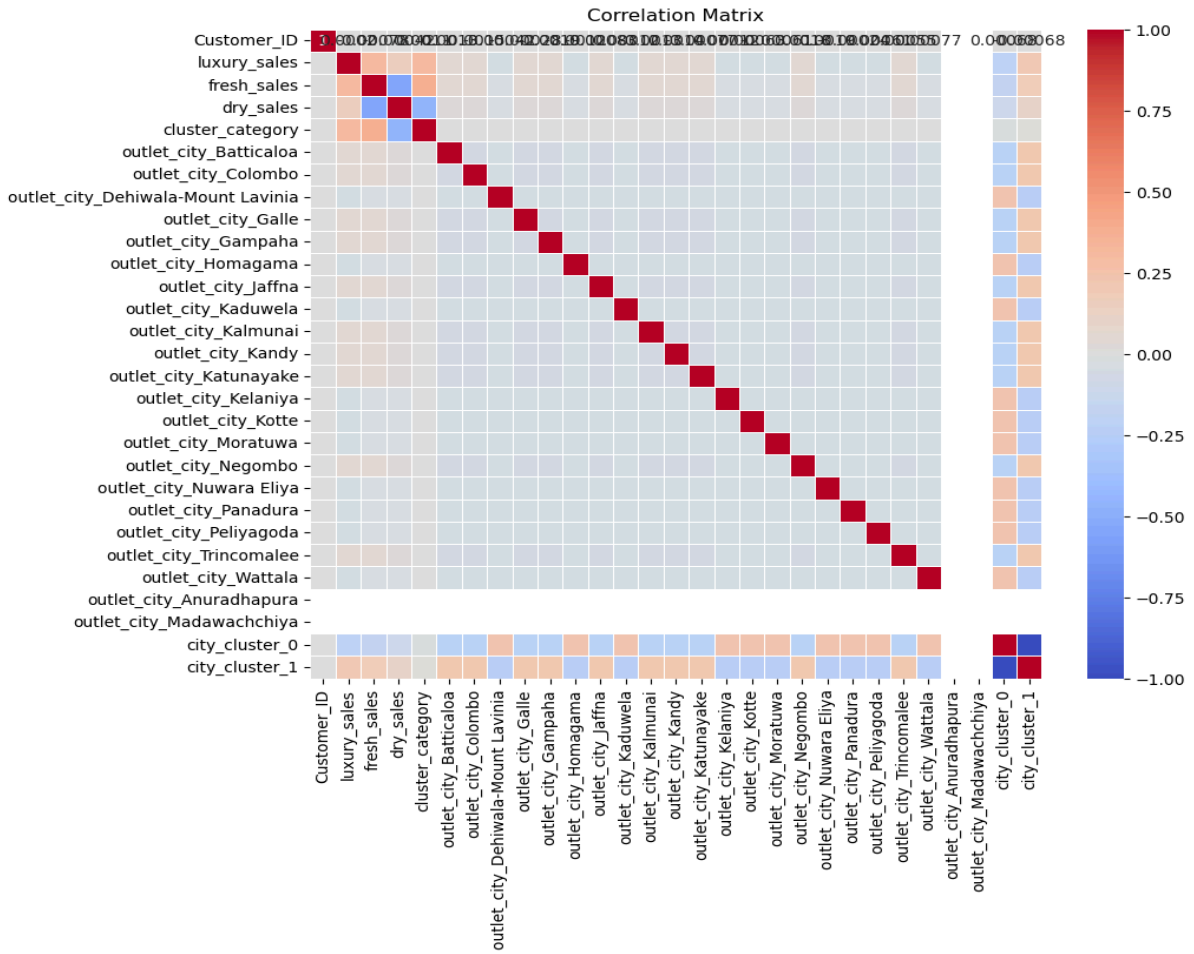
One-hot encoding was applied to the **outlet_city** and newly created **city_cluster** columns to transform categorical data into a format that could be better utilized by machine learning algorithms.

This step is vital for handling categorical variables, as it allows models to interpret them correctly without assuming any ordinal relationship between categories.

Additionally, since there were no records related to Anuradhapura and Madawachiya, in the “outlet_city” column in the Training Data set, we added two new columns for those two cities after the One-hot encoding.

Correlation Analysis

A correlation analysis was done to see how the features are correlating with each other and with the target variable. We generated the correlation matrix and visualized it using a heatmap.



According to the Heatmap, there were no features which highly correlated with each other.

The **Customer_ID** column has no meaningful correlation with other features, which is expected because it's likely an identifier rather than a numeric feature, So we decided to drop this feature.

Model Selection and Training

Initially the dataset was split into training and validation subsets to enable effective model training and performance validation.

Using 80% of the data for training and 20% of the data for validation from the training dataset. This step ensures that the model can learn from a significant amount of data while being unbiased.

Standardization of the Dataset

In our dataset, we used the **StandardScaler** from Scikit-learn to standardize the **luxury_sales**, **fresh_sales**, and **dry_sales** features, which are crucial for clustering cities based on their sales patterns.

Standardization was used to maintain consistency in data handling across different stages of training the dataset.

Modeling Approach: Random Forest Algorithm

The Random Forest Algorithm was chosen as our primary modeling approach for its robustness in handling high-dimensional data and its effectiveness in classification tasks. Random Forest, an ensemble method, builds multiple decision trees and merges them together to get a more accurate and stable prediction.

Parameters

- Num_of_estimators = 100
- Random_state = 42

Accuracy Scores

	precision	recall	f1-score	support
1	1.00	1.00	1.00	38087
2	1.00	1.00	1.00	30887
3	1.00	1.00	1.00	9817
4	1.00	1.00	1.00	34406
5	1.00	1.00	1.00	7871
6	1.00	1.00	1.00	33762
accuracy			1.00	154830
macro avg	1.00	1.00	1.00	154830
weighted avg	1.00	1.00	1.00	154830

Also we've checked using the accuracy score given by SKlearn library.

```
Validation Accuracy (sklearn): 0.9998
```

Enhancing Marketing Strategies

The analytical solution we developed through the "Data Storm 5.0" competition, particularly the segmentation of KJ Marketing's customer base into distinct clusters, provides a strategic advantage in tailoring marketing strategies. This section elaborates on how the identified customer segments can be leveraged to enhance the effectiveness of the company's marketing initiatives.

Strategic Implications of Customer Clusters

The classification of customers into two main clusters based on their purchasing patterns and regional characteristics offers a nuanced understanding of the market dynamics. These clusters represent unique segments of the market with distinct preferences and shopping behaviors.

Cluster 0 (Major Urban Centers):

Characteristics

This cluster includes cities like Colombo, Galle, and Jaffna, which are major urban centers. Customers in these areas tend to have higher disposable incomes and a preference for luxury items.

Marketing Strategy

For this cluster, KJ Marketing can implement premium pricing strategies and stock a higher proportion of luxury goods. Marketing campaigns can focus on exclusivity and high-quality service, using targeted advertisements through online platforms that resonate with an urban demographic.

Cluster 1 (Suburban and Emerging Markets):

Characteristics

Comprising cities such as Homagama and Moratuwa, this cluster includes areas with growing economic activities but traditionally less access to a wide range of products.

Marketing Strategy

For these regions, KJ Marketing could focus on promotions and discounts, especially on essential goods and affordable luxury items, to attract a cost-conscious consumer base. Community engagement through local events and loyalty programs can enhance brand presence and customer loyalty.

Marketing Communications and Product Placement

With a clear understanding of these clusters, KJ Marketing can customize its communication strategies:

- *Personalized Promotions*

  Tailor promotions based on the regional sales data and customer preferences identified in each cluster. For example, sending personalized emails with offers on products that align with historical purchasing trends in each cluster.

- *Optimized Product Placement*

Adjust product placement in stores according to the cluster-specific preferences to increase visibility of the products more likely to be purchased by the regional customers.

Leveraging Data for Future Campaigns

The insights gained from the customer segmentation can also inform future marketing campaigns:

- *Data-Driven Decision Making*

Use the sales data and cluster information to make informed decisions about inventory management, store layout designs, and promotional activities.

- *Feedback Loop*

Regularly update the clustering model with new sales data to refine customer segments and adapt marketing strategies in real-time, responding to changing market conditions and customer preferences.

Conclusion

By integrating the segmentation results into its marketing strategies, KJ Marketing can more effectively reach different segments of the market, increasing customer satisfaction and loyalty, while optimizing marketing expenditures. The strategic use of data-driven customer insights will enable KJ Marketing to maintain a competitive edge in a rapidly changing retail environment, ultimately driving higher sales and market share.