**Project Report**

**Classification Analysis on Store Profile Predictor**

**Data Storm 4.0 – Storming Round**

* Team Name: **3\_Amigos**
* Kaggle Username: DataStorm037

* Kaggle Display Name: DataStorm037
* Team Members:

Sangaran Thevarasa

Pairavi Thanancheyan

Nithursika Kalanantharasa

* GitHub Repository Link: https://github.com/Pairavi/DataStorm-4.0/
* Highest Total F1 score achieved: **66.66%(0.66)**

**Data**

Datastorm 4.0 offers us three data source files to develop a data analytics solution.

Historical Transaction Data Set - 488,788 records total, collected from each customer purchase, make up the data set. The data set's level of scale is at the date-customer-invoice-item level.

Store Info Data Set which includes 124 stores with shops space and store profile.

The Data Dictionary consists of information about data attributes described in both above datasets.

**Tools used.**

Python programming language was used for the entire analysis. The following Python libraries were used:

Data manipulation and analysis: NumPy, pandas

Modeling: Scikit-learn

Data visualization – Matplotlib, Seaborn

**Step 01: Descriptive Analysis**

1. The dataset was sorted for each shop.

2. Since item\_discription and invoice\_id have missing values, we filled them using the correct procedures.

3. Next, we updated the dataset with new features and changed some existing features into new features.

4. Next, we used the RandomForestClassifier to fit the data.

5. The shop profiles for the next 24 shops are predicted.

Transaction date: Start: 2021-10-15

 End: 2021-12-15

**Step 02: Data Cleaning**

1. ***Missing Values Handling***

Item\_discription and invoice\_id attributes both had missing values. By substituting a random item\_name for the rows with missing values, we take care of this in item\_discription.

We handled missing values in invoice\_id by replacing them with random invoice ids.

1. ***Removing duplicates (We dealt with this using the right code.)***
2. ***Removing redundant attributes***
3. ***Outliers handling (We handled this after feature engineering)***

**Step 03: Exploratory Analysis**

We found a correlation between quantity\_sold and item\_price. And it was 0.604. Other than this, we cannot come to a better conclusion.

**Step 04: Feature Engineering**

***Feature Transformation***

In the training dataset, we converted item\_discription into item\_name and item\_size. We converted invoice\_id into count\_invoice (invoice\_id count for each shop). We converted transaction\_date into date and weekper\_month. We converted customer\_id, shop\_id into customer\_countfor\_shop.

***Feature Creation***

The following new features we have created:

total\_amount: income of shop

total\_quantity: sum of quantity for each item\_name

shop\_income: income for each shop

weekly\_sales: number of unique quantities sold per week

weekly\_income

customer\_count\_per\_day\_per\_shop: the number of unique customers per day per shop

price\_per\_sq\_ft: shop income per square feet

bev\_countfor\_shop: beverage count for each shop regarding their size

itemdic\_countfor\_shop: beverage count for each shop

Ave\_quantitysold\_perinvoice

Total number of unique items

Ave\_inc\_perinvoice: average income for each invoice

***Feature encoding***

We encoded the features such as item\_name, item\_description, customer\_id, shop\_profile by using the *Label Encoding* technique.

**Outliers handling**

In the given data, there are some outlier values that exist; we replaced those values. If that value was greater than the upper bound, then we replaced it with the maximum value of that feature; otherwise, we replaced it with the minimum value of that feature.

**Correlation analysis**

We created some new features because we believed they would be useful and meaningful features of the given domain. Then we checked whether those values were correlated or not. So we use correlation analysis to remove the most correlated features.

Chart, bar chart, waterfall chart

Description automatically generated

**Feature scaling**

We scaled the features because those given and created were of a different scale, so we used the Min-Max scaling technique.

**Step 05: Best Model Selection**

We split our dataframe into train dataframe and test dataframe by using test size = 0.2. Initially selected models for this project are *Linear Regression,*

*SVM, GBM, RandomForestRegressor*, *Neural Networks, Bagging, Naive Bayes, KNeighborsRegressor and DecisionTreeRegressor* because these are pretty good for Multiclass classification problems, simplest models, scalable and non-linearity.

The Random Forest Regressor was chosen as our best model.

**Step 06: Tuning for The RandomForestRegressor**

We tried to tune the *n\_estimators* and *random\_state* of *RandomForestRegressor* using *GridSearchCV*.

As a result, we had discovered our best model and optimal hyperparameters. That was the RandomForestRegressor with the parameters random\_state=10 and n\_estimators=90.

**Step 07: In the test dataset, predict the values for the shop profile**

We used that model to predict the values.

**Additional attributes that are developed to perform store profiling**

* The population rate of the shop-located area  
  (Typically, the selling rate is high in highly populated areas.)
* The economic stage details the population in the shop-located area.  
  (Usually, high-income people buy the beverage frequently.)
* What was the weather condition during the recorded days?  
  (Usually in the hottest areas, the beverages sell well.)
* How many years has the shop been running  
  (Typically, customers prefer to buy from shops of all ages.)
* Were there any offers/discounts during the recorded weeks/days?

**List down interventions that the Beverages Company XYZ can take to enhance their decision-making process using the above store profiling mechanism.**

**Chart, bar chart

Description automatically generated**

**According to the above graph, most sales happened on Saturday and Sunday in every week.**