

# **A synopsis on**

**CUSTOMER SEGMENTATION ML PROJECT**

**Under the guidance of mentor:**

**AISHWARYA SINGH**

* **Name of Internee:** Rajveer Singh
* **Technology:** Python with ml
* **College:** Lucknow University F.O.E

|  |  |  |
| --- | --- | --- |
| **N0.** | **TOPIC** | **PAGE NO.** |
| **1** | **INTRODUCTION** | **3** |
| **2** | **OBJECTIVE** | **4** |
| **3** | **BACKGROUND** | **5** |
| **4** | **HARDWARE AND SOFTWARE REQUIREMENTS** | **9** |
| **5** | **FUTURE SCOPE** | **10** |
| **6** | **CONCLUSION** | **11** |
| **7** | **BIBLIOGRAPHY AND REFRENCES** | **12** |

**INTRODUCTION**

**Customer segmentation**

Customer segmentation is the practice of dividing a customer base into groups of individuals that are similar in specific ways relevant to marketing, such as age, gender, interests and spending habits**.**

When you perform customer segmentation, you find similar characteristics in each customer’s behavior and needs. Then, those are generalized into groups to satisfy demands with various strategies. Moreover, those strategies can be an input of the

* Targeted marketing activities to specific groups
* Launch of features aligning with the customer demand
* Development of the product roadmap

Photo source: Costumer purchasing groceries



**OBJECTIVE**

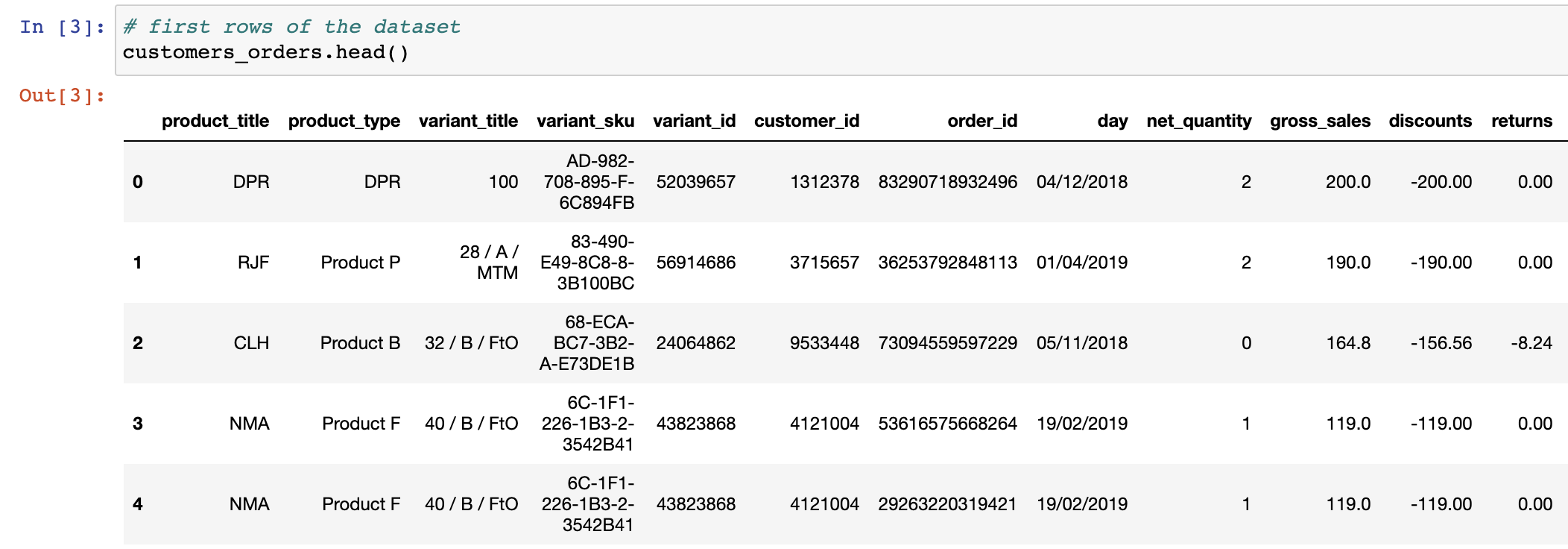
* I will apply an [unsupervised machine learning algorithm](https://en.wikipedia.org/wiki/Unsupervised_learning) with Python.
* Load dataset from November 2018 — April 2019 is actual sales data courtesy of an e-commerce company for case study.
* I will apply [K-Means clustering](https://en.wikipedia.org/wiki/K-means_clustering) to the dataset with the following steps.
* Business Case
* Data Preparation
* Segmentation with K-means Clustering
* Hyper parameter Tuning
* Visualization and Interpretation of the Results
* I will explain how K-means clustering works. Eventually, I will provide specific strategies for the segments formed.

**BACKGROUND**

**PROCEDURE:**

* I visualized the customer behavior and characteristics from diverse aspects. Taking it one step further, I will form the business case around the question: **Can the customer base be grouped to develop customized relationships?**
* I will approach this question from a behavioral aspect (alternatives can be geographical or demographical perspectives) to better understand customers’ spending and ordering habits with the following features: **Number of products ordered, average return rate and total spending.**
* DATA PREPARATION:

There are approximately 25000 unique customers combined with their order information in the raw dataset:



Dataset is well-formatted and had no NA values. So, we can start by forming the features. 3 features will be calculated per customer\_id and they will help us with the visualization (using [Plotly](https://plot.ly/python/) library) and algorithm explainability in the latter steps. Data preparation will be done with [pandas](https://pandas.pydata.org/) and [numpy](https://numpy.org/).

* **Number of products ordered:** It is calculated by counting the product\_type ordered by a customer with the below function:

|  |
| --- |
| def encode\_column(column): |
|  | if column > 0: |
|  | return 1 |
|  | if column <= 0: |
|  | return 0 |
|  |  |
|  |  |
|  | def aggregate\_by\_ordered\_quantity(dataframe, column\_list): |
|  | '''this function: |
|  | 1. aggregates a given dataframe by column list, |
|  | as a result creates a aggregated dataframe by counting the ordered item quantities |
|  |  |
|  | 2. adds number\_of\_X ordered where X is the second element in the column\_list |
|  | to the aggregated dataframe by encoding ordered items into 1 |
|  |  |
|  | 3. creates final dataframe containing information about |
|  | how many of X are ordered, based on the first element passed in the column list''' |
|  |  |
|  | aggregated\_dataframe = (dataframe |
|  | .groupby(column\_list) |
|  | .ordered\_item\_quantity.count() |
|  | .reset\_index()) |
|  |  |
|  | aggregated\_dataframe["products\_ordered"] = (aggregated\_dataframe |
|  | .ordered\_item\_quantity |
|  | .apply(encode\_column)) |
|  |  |
|  | final\_dataframe = (aggregated\_dataframe |
|  | .groupby(column\_list[0]) |
|  | .products\_ordered.sum() # aligned with the added column name |
|  | .reset\_index()) |
|  |  |
|  | return final\_dataframe |

* **Average return rate:**It is the ratio of returned\_item\_quantity to the ordered\_item\_quantity averaged for all orders of a customer.
* **Total spending:**It istheaggregated sum of total sales, which is the final amount after taxes and returns.
* **Let’s have a look at the individual distribution of the features:**



All 3 distributions are positively [skewed distributions](https://en.wikipedia.org/wiki/Skewness). Products ordered shows a power-law distribution and average return rate of 99% of the customers are 0.

3 features have different ranges varying between [1, 13], [0, 1] and [0, 1000] which is an important observation showing that features need scaling!

* **Scaling:**

K-means algorithm interprets each row in the customers data frame as a point in a 3-dimensional space. When grouping them, it uses the [euclidian distance](https://en.wikipedia.org/wiki/Euclidean_distance) between the data points and the center of the group. With highly varying ranges, algorithm may perform poorly and not be able to form the groups as expected.

For K-means to perform effectively, we are going to scale the data using logarithmic transformation which is a suitable transformation for skewed data. This will scale down proportionally the 3D space which our data is spread, yet preserving the proximity between the points.

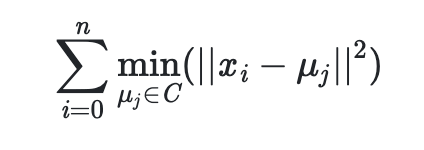
* **After applying the above function,**customers**data frame is ready to be fed into K-means clustering.**
* **Segmentation with K-means Clustering**
* We are going to use [K-means algorithm from scikit-learn](https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html). Let’s first understand how the algorithm will form customer groups:
* Initialize *k*=*n centroids*=*number-of-clusters* randomly or smartly

Assign each data point to the closest centroid based on euclidian distance, thus forming the groups

Move centers to the average of all points in the cluster

Repeat steps 2 and 3 until [convergence](https://www.mathsisfun.com/definitions/converge.html).

* While running the steps through, the algorithm checks the sum of squared distances between clustered-point and center for each cluster. Mathematically speaking, it tries to minimize — optimize the ***within-cluster sum-of-squared-distances*** or ***inertia***of each cluster***.***

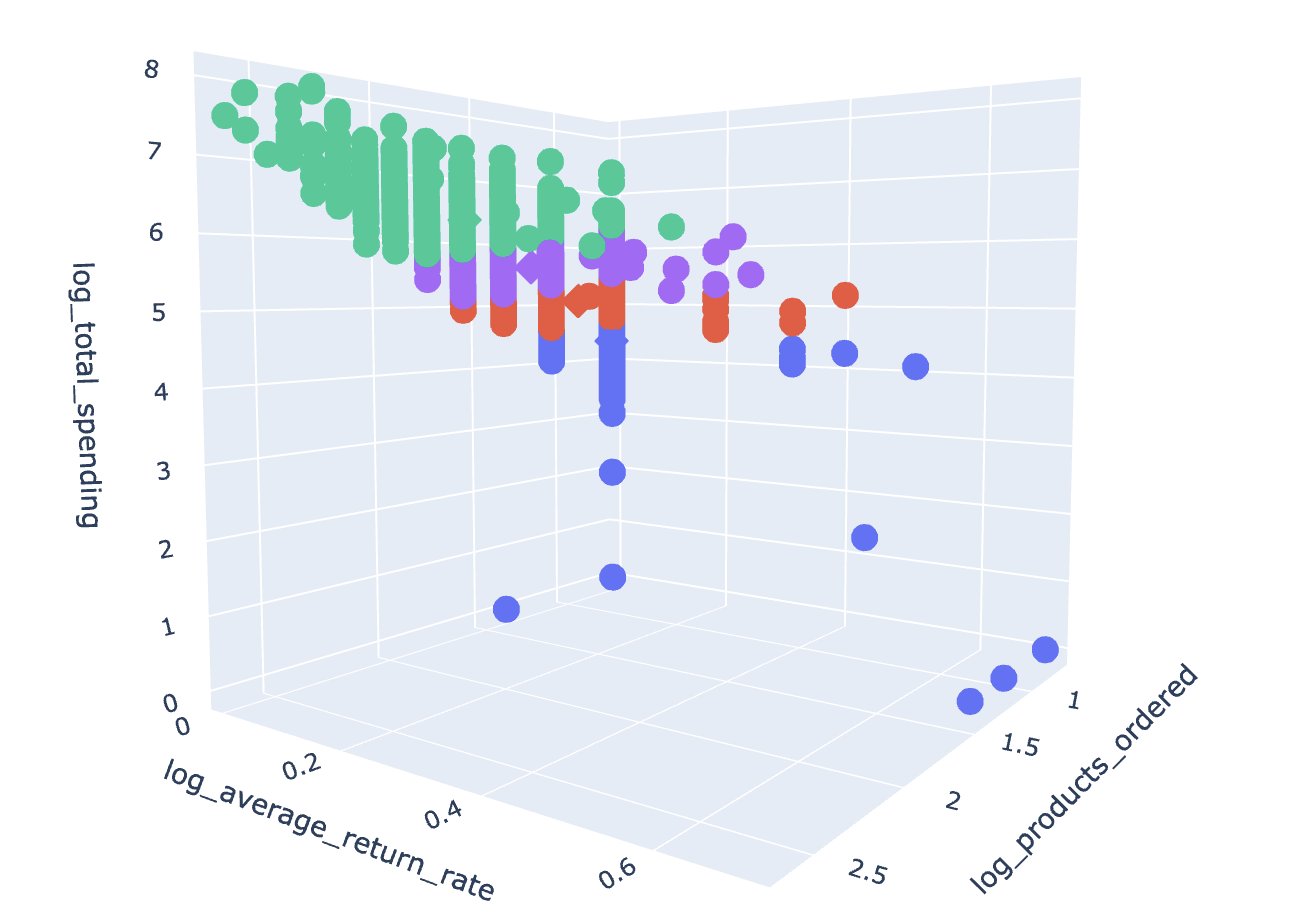


When ***inertia*** value does not minimize further, algorithm converges. Thus, iteration stops.

* **Hyperparameter Tuning**

While selecting *k,*we are going to decide against the optimization criteria of the K-means, inertia, using [elbow method](https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/). We are going to build different K-means models with *k* values 1 to 15, and save the corresponding *inertia* values.

* **Visualizing and interpretation of results:**



Data points are shown in spheres and centroids of each group are shown with cubes. 4 customer groups are as follows:

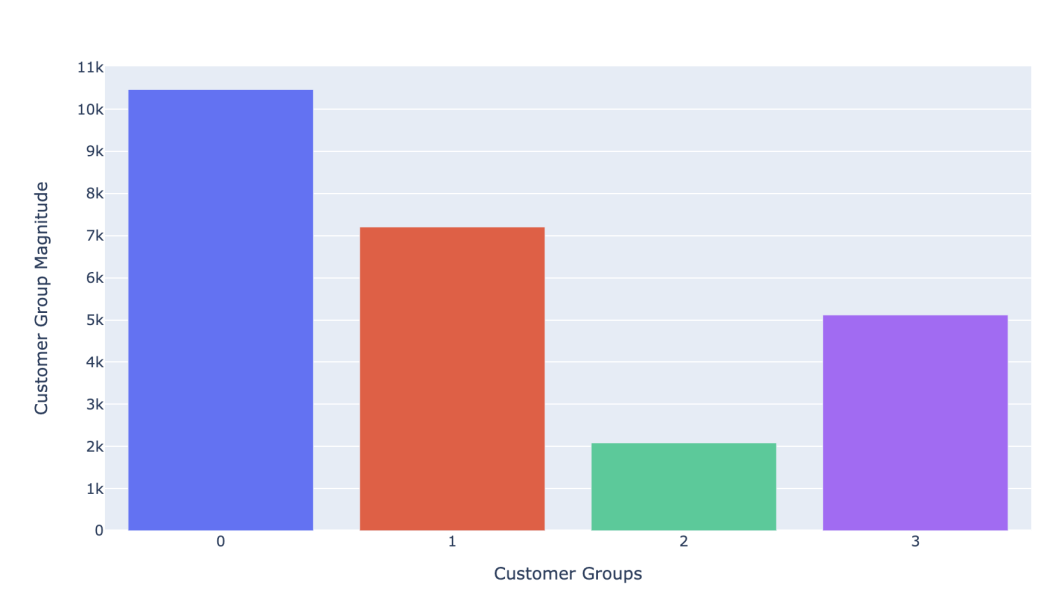
**Blue:** Customers who ordered at least one product, with maximum total spending of 100 and having the highest average return rate. They might be the newcomers of the e-commerce website.

**Red:** Customers who ordered 1 to 4 products, with average total spending of 150 and a maximum return rate of 0.5.

**Purple:** Customers who ordered 1 to 4 products, with average total spending of 300 and a maximum return rate of 0.5.

**Green:** Customers who ordered 1 to 13 products, with average total spending of 600 and average return rate as 0. It makes the most favourable customer group for the company.

* **Let’s look at how many customers are there in each group — known as cluster magnitudes:**



**HARDWARE AND SOFTWARE REQUIREMENTS**

**HARDWARE REQUIREMENTS:**

|  |  |
| --- | --- |
| Hardware tools | Minimum requirements |
| Processor | i5 or above |
| RAM | 4gb |
| Monitor | 17”colored |
| Mouse | Optical |
| Keyboard | 122keys |

**SOFTWARE REQUIREMENTS:**

|  |  |
| --- | --- |
| Software tools | Minimum requirements |
| Platform | Windows, linux or macos |
| Operating System | Windows, linux or macos |
| Technology | Machine learning-Python |
| Scripting Language | Python |
| IDE | Pycharm and Jupyter Notebook |

**FUTURE SCOPE**

Companies can use [marketing automation](https://searchcustomerexperience.techtarget.com/definition/marketing-automation) software to define and create customer segments. The customer segments can be based on demographic data, psychographic data and activity-based data such as actions that users took on a website. Companies use marketing automation software to configure, schedule and execute campaigns for particular customer segments.

**CONCLUSION**

* We approached customer segmentation problem from a behavioural aspect with the number of products ordered, average return rate and total spending for each customer. Use of 3 features helped us with the understandability and visualization of the model.
* All in all, the dataset was apt to perform an unsupervised machine learning problem. At first, we only had customers data with order information and did not know if they belonged to any group. With the K-means clustering, patterns in the data were found and extended further into groups. We carved out strategies for the formed groups, making meaning out of a dataset that is a dust cloud initially.

**REFRENCES AND BIBLIOGRAPHY**

* Datasets(<https://www.kaggle.com/datasets>)
* Sklearn libraries(<https://scikitlearn.org/stable/modules/generated/sklearn.cluster.KMeans.html>)
* Algorithm refrences(<https://nbviewer.jupyter.org/github/cereniyim/Customer-Segmentation-Unsupervised-ML-Model/blob/3c4374dd16861ea365cdf468bd9b2c28a964f4e3/Customer_Segmentation_Kmeans_Clustering.ipynb>)