Unveiling Customer Segmentation: A K-means Clustering Approach using Python, PySpark and Map-Reduce

Simranpal Singh (SXX210010)  
*Eric Jonsson School of Engineering and Computer Science*  
*University of Texas at Dallas*Richardson, Texas  
[sxx210010@utdallas.edu](mailto:sxx210010@utdallas.edu)

Dhruv Mittal (DXM220015)  
*Eric Jonsson School of Engineering and Computer Science*  
*University of Texas at Dallas*Richardson, Texas  
dxm220015@utdallas.edu

Tarun Teja Obbina (TXO220011)  
*Eric Jonsson School of Engineering and Computer Science*  
*University of Texas at Dallas*Richardson, Texas  
txo220011@utdallas.edu

Datta Vageesha   
*Eric Jonsson School of Engineering and Computer Science*  
*University of Texas at Dallas*Richardson, Texas  
@utdallas.edu

*Abstract*—*This study explores the implementation of K-means clustering algorithm from scratch using Python, PySpark and MapReduce technologies to segment customers data based on their purchasing behavior. In an era of data-driven decision-making, understanding customer behaviour is crucial for businesses to tailor their marketing strategies and enhance customer satisfaction. Leveraging a dataset and capturing customer shopping behaviors, including factors such as the hour of the day, day of the week, and department IDs and names, we aim to uncover distinct groups of customers exhibiting similar purchasing patterns. By employing K-means clustering, a popular unsupervised learning technique, we delineate customer segments, enabling businesses to gain valuable insights into customer preferences, optimize resource allocation, and devise targeted marketing campaigns. Through this research endeavor, we demonstrate the practical applicability of K-means clustering in customer segmentation and offer insights into its implementation using Python, PySpark and MapReduce technologies paving the way for informed decision-making and enhanced customer-centric strategies in retail and beyond.*

Keywords — K-means, customer segmentation, Exploratory data analysis (EDA), WCSS, Machine learning, Big data

# Introduction

In today’s competitive surroundings, understanding customer behaviour in businesses is the one of the most important aspects that help businesses strive and stay ahead of the curve. As businesses are growing, huge amount of data is being generated on various aspects and here comes the challenge of extracting useful information and gain valuable insights from the data that can help to drive proper decision making. So we have utilized the K-Means Clustering machine learning technique to do customer segmentation in a better way.

K-Means Clustering is a popular unsupervised machine learning technique which is used for clustering data points into K clusters based on similarity of the data point and the cluster. Each cluster is represented by a centroid, which is basically the co-ordinates of that point in n-dimensions where n represents the number of features we are using for the k-means clustering. By iteratively optimizing cluster assignments and centroid positions, K-Means help to identify natural groupings within the data. This helps businesses to extract helpful insights from the customer data and therefore have better marketing strategies tailored toward those particular segments of audience.

In this report we have dig deep into the application of K-Means clustering algorithm from scratch using PySpark Framework, MapReduce and Python programming language. We demonstrate the scalability and efficiency of K-Means clustering when coupled with the parallel processing capabilities of Map Reduce using PySpark.

# Problem definition

We have a shopping dataset, which has details like the order hour of the day, order day of the week, to which department did the order belong to, along with the name of the department for all the purchases that happened. Leveraging a real world situation, using customer interactions like departmental preferences, transactional histories, timing of the purchase of items during the day and week.

We showcase the practical usage of K-Means clustering in finding useful insights around which day of the week had the most and the least number of order, in a similar way which hour of the day had the maximum and minimum number of orders. On a similar note we also try to find which department receives a lesser number of orders on which days of the week. The aim is to develop a scalable and efficient approach to customer segmentation using machine learning algorithm like K-means clustering . The primary objective is to leverage the power of data analytics to uncover hidden patterns and segment customers based on their purchasing behavior, preferences, and interactions with the business.

# Implementation

# Kmeans Algorithm

In our implementation of the K-means algorithm using PySpark's map-reduce fashion, we first transformed our DataFrame into a plain RDD to leverage its distributed processing capabilities. This conversion allowed us to work with the data in a distributed manner, essential for handling large datasets efficiently. By converting each DataFrame row into a simple list, we simplified the data structure, making it easier to manipulate and process during the algorithm's execution.

Following the conversion, we introduced an index to each instance in the RDD. This index served as a unique identifier for each data point and enabled us to maintain consistency and order throughout the various stages of the algorithm. Having indexed instances facilitated tracking and referencing individual data points during the clustering process.

Next, we fetched the starting centroids required to initiate the K-means clustering process. These initial centroids act as the starting points for cluster formation and play a pivotal role in determining the final clusters. By obtaining the initial centroids, we established the foundation for the subsequent iterations of the algorithm.

During each iteration of the algorithm, we performed several key steps. Firstly, we paired each instance with the list of centroids, forming key-value pairs. Subsequently, we utilized a custom function to calculate the distance between each instance and the centroids, assigning the closest centroid to each instance. This step was crucial for determining the initial cluster assignments for each data point.

After assigning initial clusters, we counted the number of instances in each cluster by removing the index from the paired instances and adding a unit value to each instance. This step enabled us to track the number of instances belonging to each cluster, providing insights into cluster sizes and distributions.

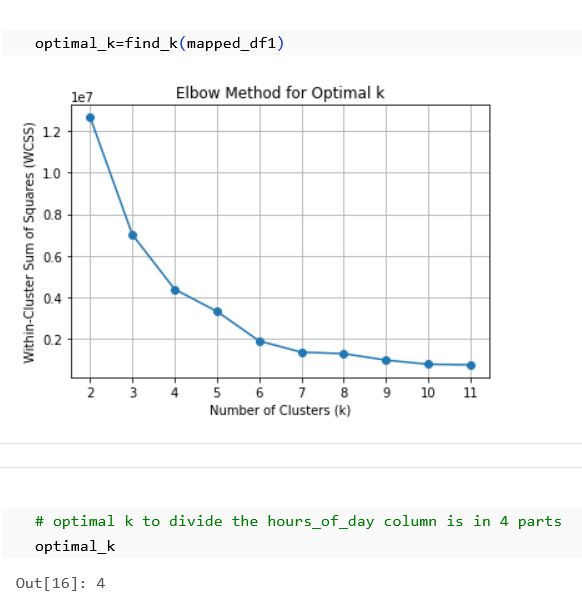
We then reduced the key-value pairs based on the cluster index, aggregating the points and unit values for each cluster. This aggregation step allowed us to compute the sum of the points data and the count of instances in each cluster, which formed the basis for calculating the mean values.

Using the sums of the points data and the count of instances in each cluster, we computed the mean values, which served as the new centroids for the subsequent iteration. If the new centroids matched the previous centroids, indicating convergence, we terminated the loop, as further iterations would not yield significant changes in cluster assignments.

Throughout the implementation, we leveraged several helper functions, such as `is clusters same()`, `initialize centroids 1()`, and `calculate\_distance\_and\_assign\_centroid()`, to streamline various aspects of the algorithm, from centroid initialization to cluster assignment and convergence checking. These helper functions enhanced the modularity and readability of our implementation, enabling efficient and scalable K-means clustering in a distributed environment.

(B) WCSS (Within Cluster Sum of Squares)

The WCSS method in KMeans Clustering is used to determine how well the data points are clustered. WCSS is the sum of square distance between each data point and the centroid of the cluster which they belong to. Then with each value of K, we try to plot the WCSS values with the respective K value and determine the optimal value of K for the current feature which is known as The Elbow Method.



# Result and Discussions

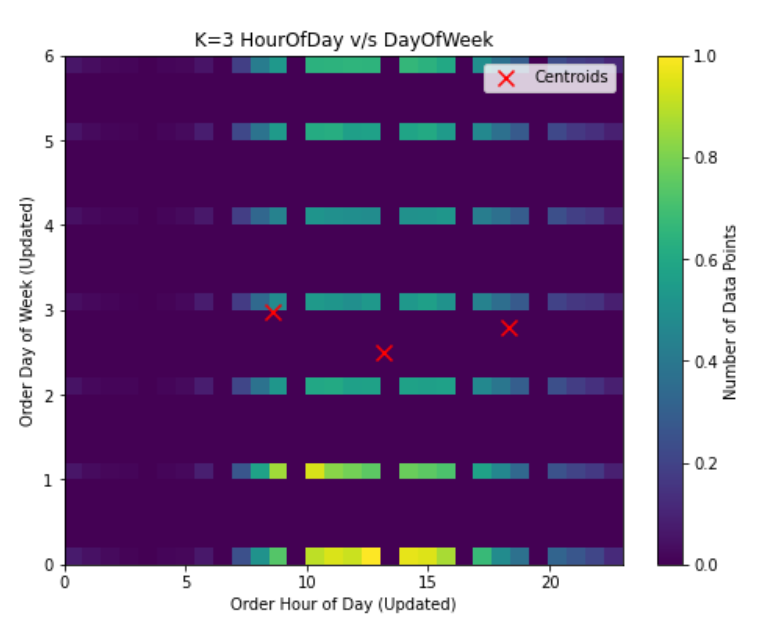
The aim was to use K-Means clustering in finding useful insights around which day of the week had the most and the least number of order, in a similar way which hour of the day had the maximum and minimum number of orders. Gathering insight on all these metrics will help us run targeted marketing campaigns during the time and day when we receive the least number of order, and also giving promotional offers on the department which is less popular will help us boost sales of that department also. We got K=4 as the optimal value of K using the WCSS method.

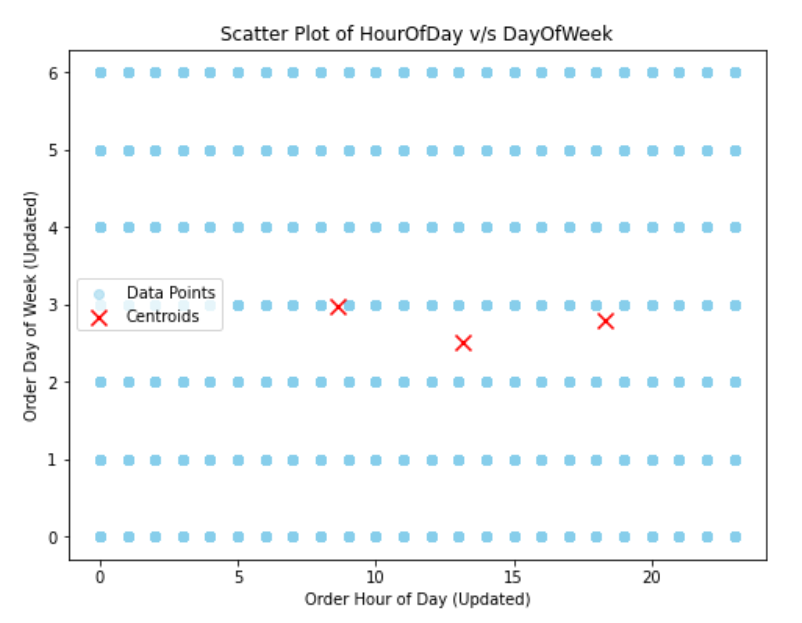
We have tried K=3 and 5 as well just for comparison with our optimal K value i.e. K=4.

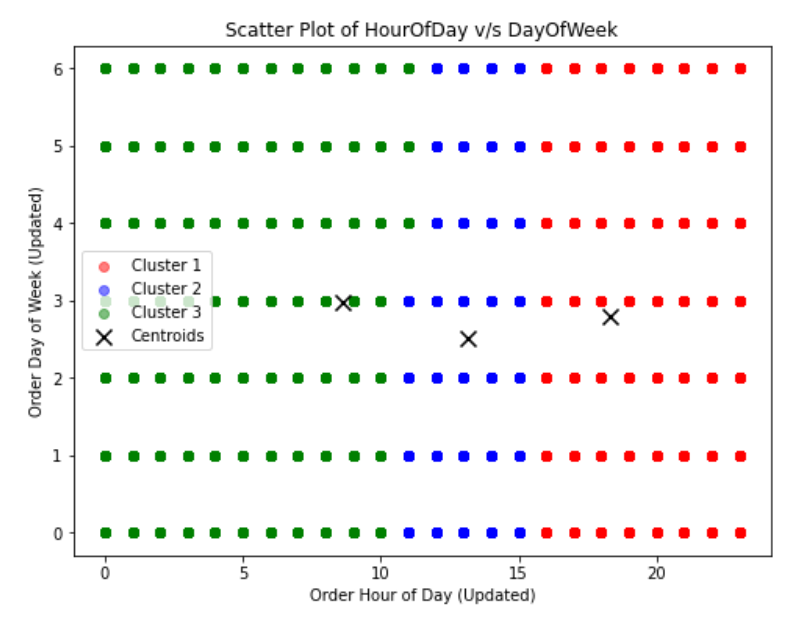
**K=3**

**Day of the order/ Hour of the order**

From the below plot we can infer that centroids happen to be near the middle of the week which is Wednesdays and Thursdays and timing to place the order is also during afternoon time.

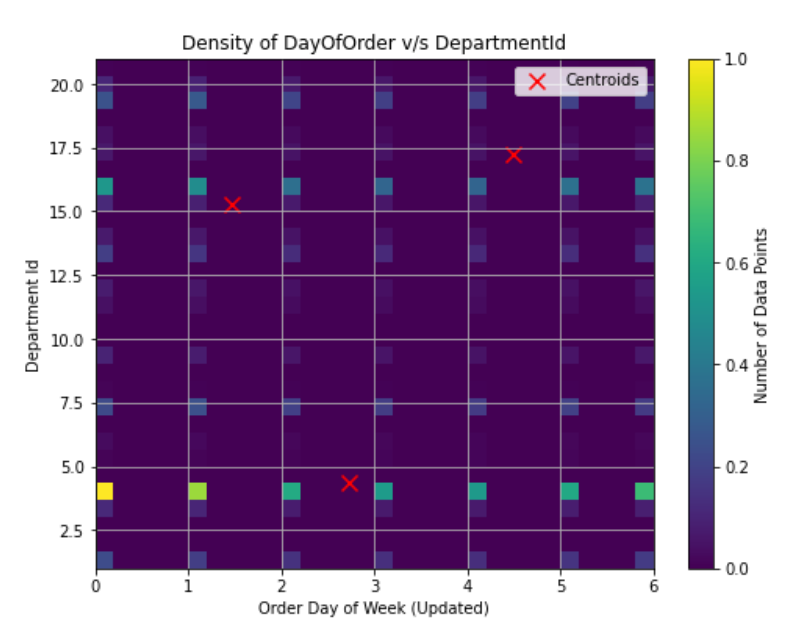
****

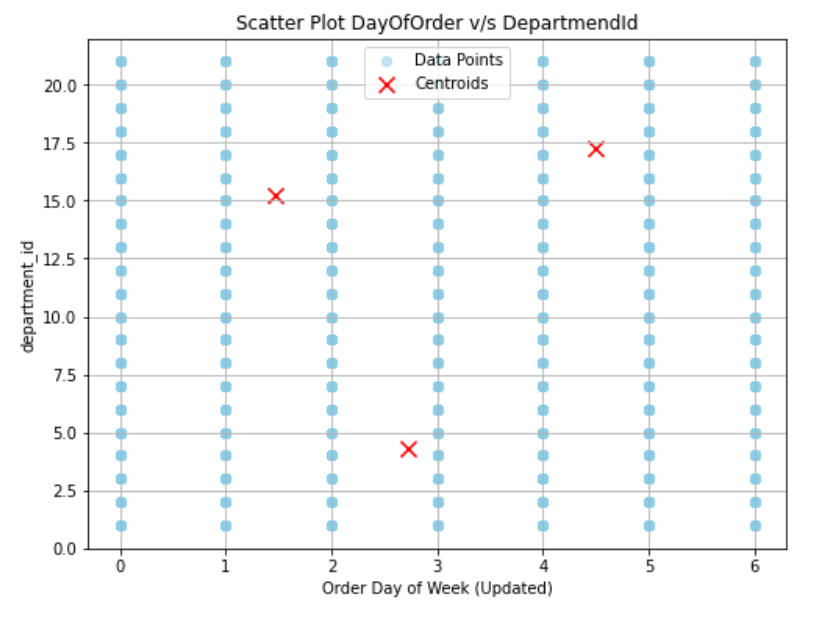
****

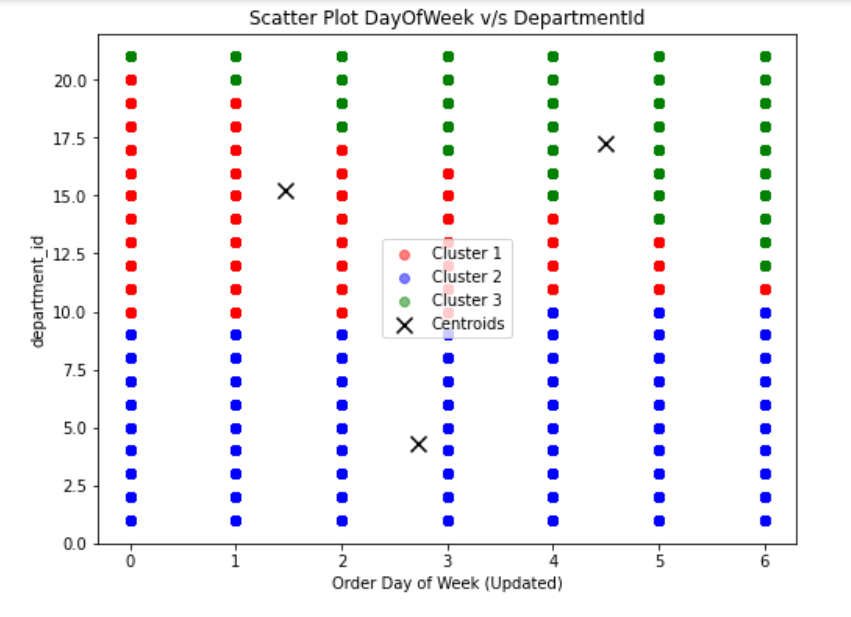
****

**Day of the order/ Department**

From the plot below we can infer that department 4, 15 and 16 are the most popular which shows people order most from these departments and during Mondays, and Wednesdays.

****

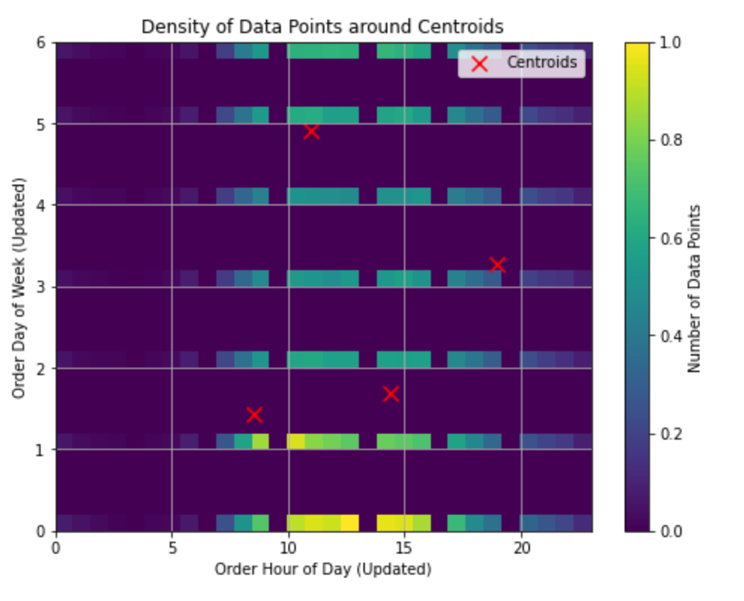
****

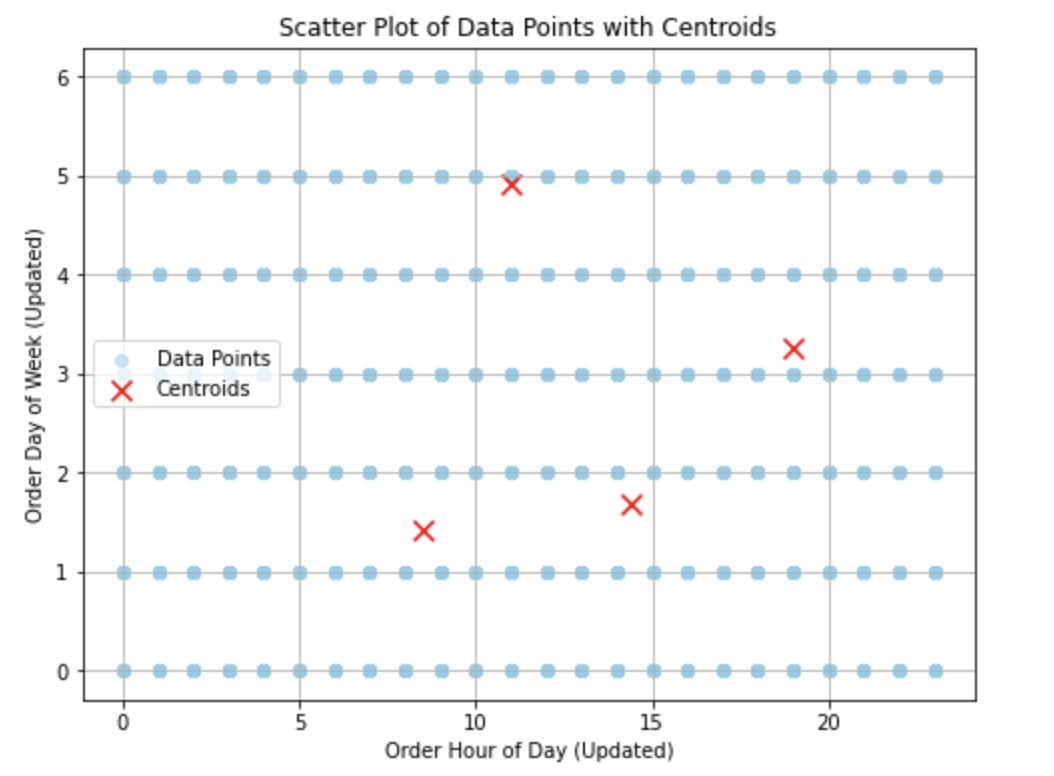
****

**K=4**

**Day of the order/ Hour of the order**

From the below given heat map, we can infer that majority orders are being placed from 10 am to 4 pm, and also most orders are placed during the start of the week i.e. Monday and Tuesday, and during the weekends. We can infer that on every day of the week, from midnight till morning 8 am, we encounter negligible orders which shows that least amount of orders are being placed during this time frame.



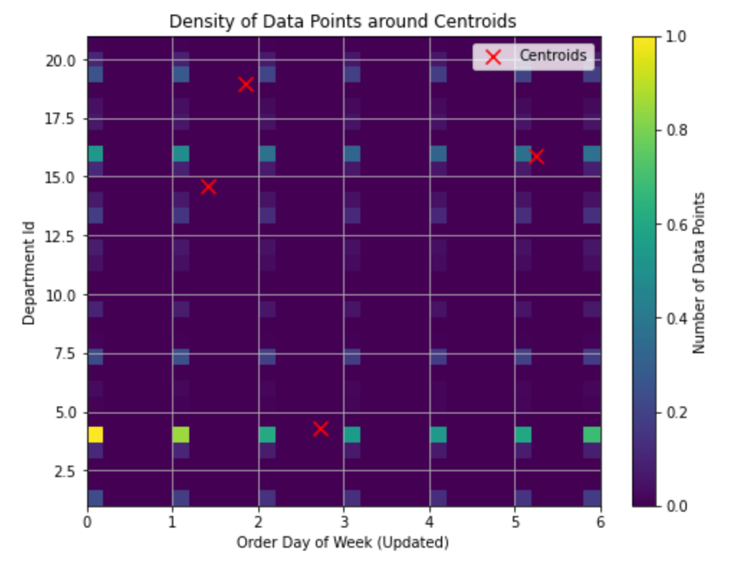


A chart with colored dots

Description automatically generated

**Day of the order/ Department**

From the plot below we can infer that, most popular departments are **16 - dairy and eggs, 7- beverages, 4-produce, 19 – personal care.** Although there are orders from other departments as well, but order quantity is less as compared to these 4 departments during the week.



A graph with red and blue dots

Description automatically generated

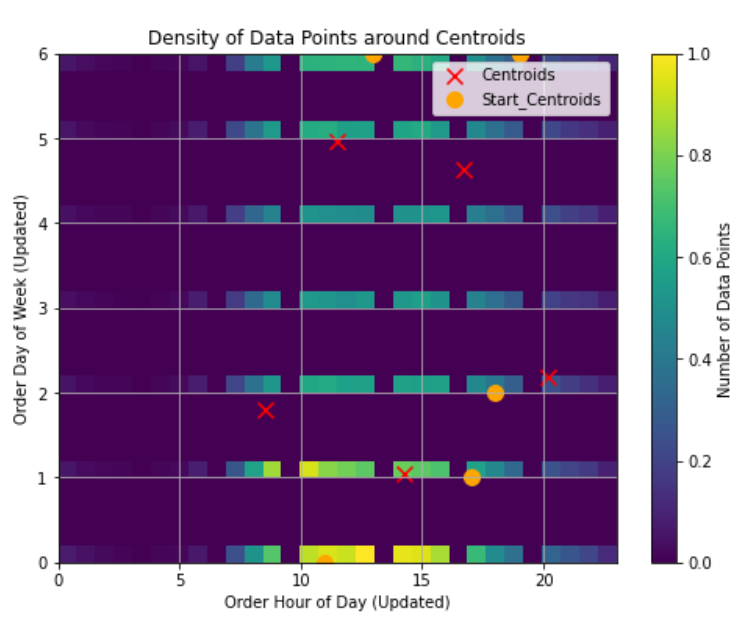
A graph of multiple colored dots

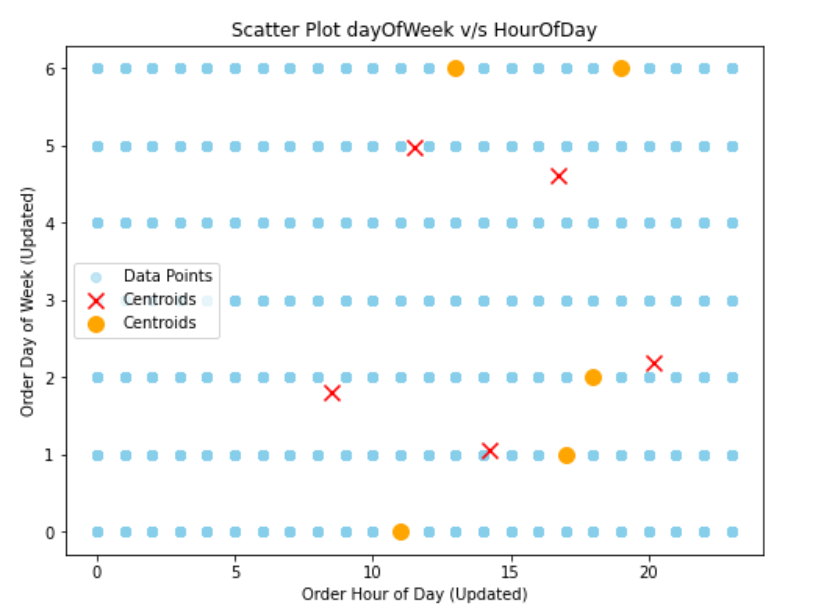
Description automatically generated with medium confidence

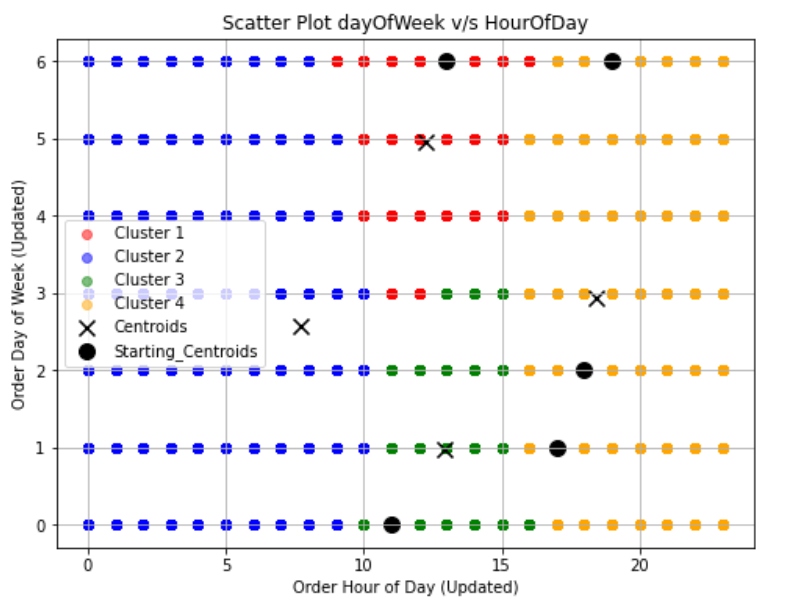
**K=5**

**Day of the order/ Hour of the order**

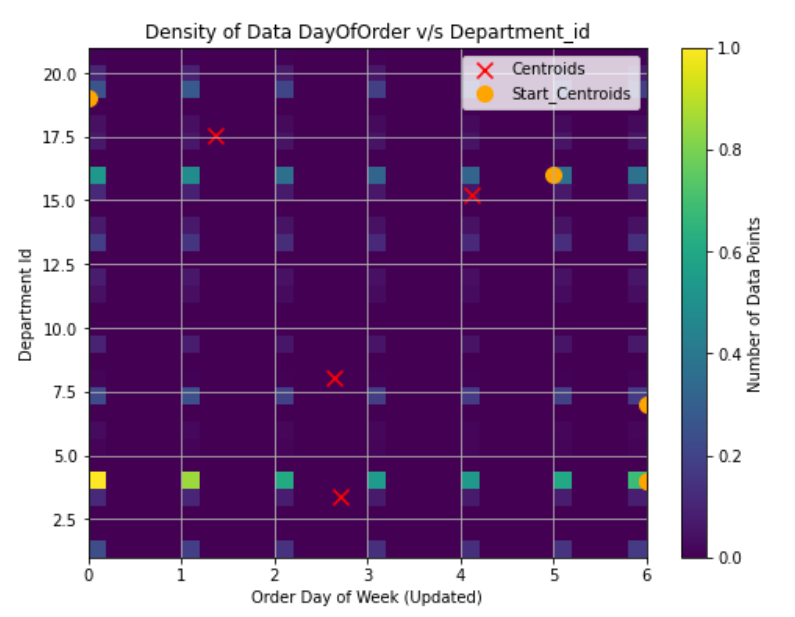
From the plot below we can infer that majority orders are being placed between 10am to 5pm but during Mondays, Tuesdays and Fridays.

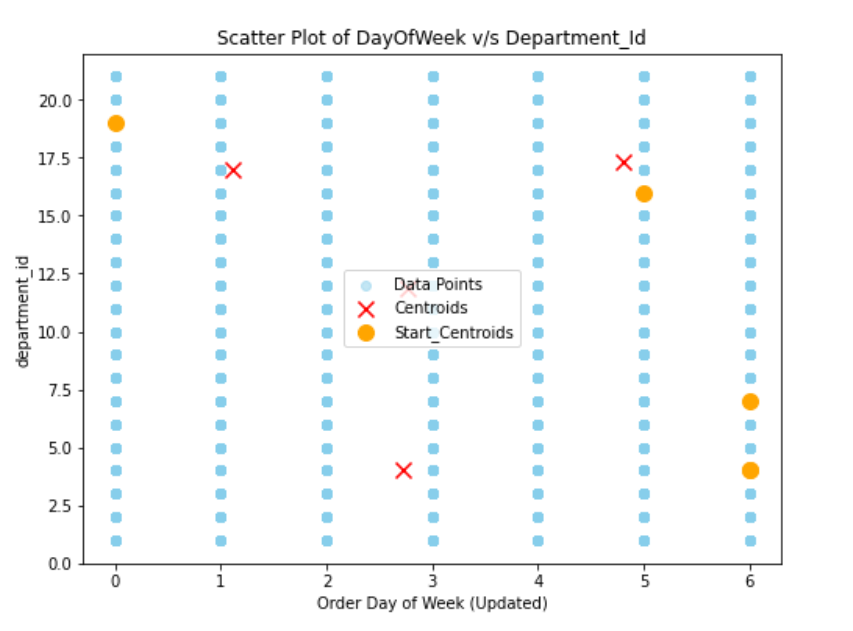
****

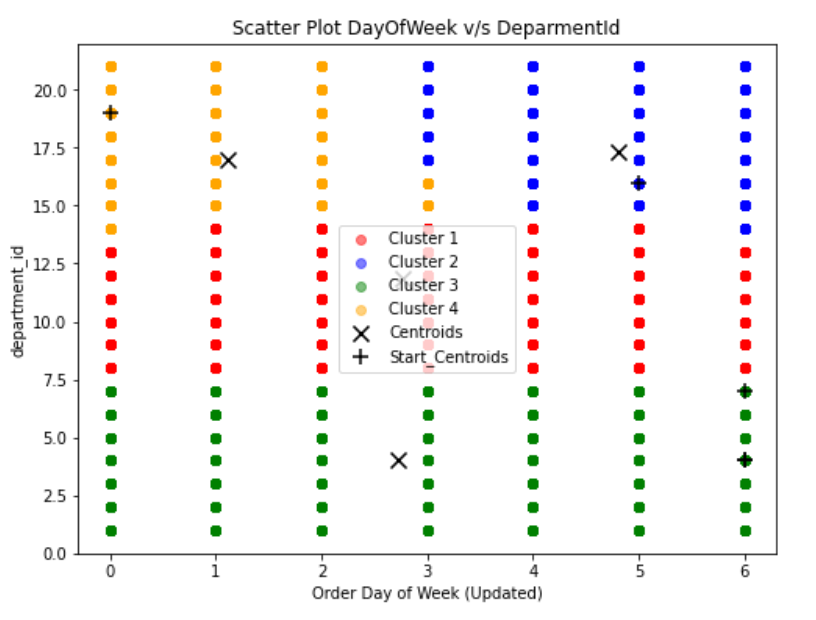
****

****

**Day of the order/ Department**

****From the plot below we can infer that, most popular departments are **16 - dairy and eggs, 7- beverages, 4-produce, 19 - personal care,** during middle of the week**.** Although there are orders from other departments as well, but order quantity is less as compared to these departments during the week.

****

****

# Conclusion

|  |  |  |
| --- | --- | --- |
| Experiment No. | Parameters | Results |
| 1 | **K=3**  (A) Day of the order, Hour of the Day  Initial Centroids-(17, 0), (13, 0), (8, 3)  (B) Day of the order, Department Id  Initial Centroid- (0, 16), (4, 14), (6, 19) | (A) final centroid-  (18.29, 2.78), (13.16, 2.49), (8.61, 2.98)  (B) final centroid-  (1.46, 15.25), (2.72, 4.33), (4.50, 17.26) |
| 2 | **K=5**  (A) Order of the Day, Hour of the Day  Initial Centroids-  (11, 0), (18, 2), (19, 6), (13, 6), (17, 1)  (B) Day of the order, Department Id  Initial Centroid-  (6, 4), (5, 16),  (6, 4), (6, 7),  (0, 19) | (A) final centroid-  (8.52, 1.79), (20.17, 2.18), (16.72, 4.62), (11.50, 4.97), (14.24, 1.05)  (B) final centroid-  (2.71, 3.40), (4.12, 15.22),  (5.23, 3.45), (2.64, 8.03), (1.36, 17.52) |
| 3 | **K=4** (Optimal K)  (A) Day of the order, Hour of the Day  Initial Centroids-  (9, 6), (7, 1), (9, 3), (16, 3)  (B) Day of the order, Department Id  Initial Centroid-  (0, 14), (0, 19), (3, 13), (0, 16) | (A) final centroid-  (18.96, 3.266), (8.52, 1.42), (10.97, 4.91), (14.41, 1.68)  (B) final centroid-  (1.42, 14.57), (1.84,18.96), (2.72, 4.32), (5.24, 15.871) |

# References

1. "Learning Spark: Lightning-Fast Big Data Analysis" by Holden Karau, Andy Konwinski, Patrick Wendell, and Matei Zaharia - This book provides a comprehensive guide to Apache Spark, including PySpark, which you utilized for parallel processing in your implementation..
2. "Python Machine Learning" by Sebastian Raschka and Vahid Mirjalili - This book covers various machine learning algorithms implemented in Python, including K-means clustering, with practical examples and explanations.
3. "Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis" by Mohammed Guller - This book focuses on practical applications of Spark for big data analytics, including clustering techniques like K-means.
4. https://sampsonipiankama.medium.com/customer-segmentation-using-k-means-clustering-ae73e3d82934
5. https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/
6. https://www.geeksforgeeks.org/k-means-clustering-introduction/