Applying Machine Learning on ABC-XYZ Inventory Model Using Multivariate and Hierarchical Clustering

Irvan Prama Defindal[1] , Nopriadi Saputra[2]

[1]Master of Accounting, University of Indonesia; irvan.prama@ui.ac.id
[2]Management Department, Bina Nusantara University; nopriadi.saputra@binus.ac.id
defindal@gmail.com

Abstract. Most of ventures need to manage many kinds of product urging them to prioritize one product on top of another. The prioritization criteria will depend on each department, making one product possibly get different priority level by different department. Unsupervised Machine Learning discipline could provide solution for this problem by using Hierarchical Clustering either Agglomerative or Divisive Clustering. This research uses a skincare company as case study which have 300 kinds of products to be managed and stored in 8 distributed warehouses. Each department recommend their own variables to be considered in classifying products from business aspect to nature of the products. Factors such as revenue contribution ranging from high-mid-low (ABC class) and demand volatility ranging from seasonal-linear-constant (XYZ class) making the clustering become multivariate. Using 4-years data, the algorithms have successfully classified product into multiple categories and mapped as matrix, where each product on the same box of matrix would be treated equally. Having the clustering established, each department would run business based on this classification, such as inventory placement in warehouse, prioritized product when purchasing or conducting promotion program. The implementation of this initiative on SME could be an inspire for other SMEs or even bigger company to implement the same methodology, concept, or model. This research presents how Machine Learning can be implemented in the Industry 5.0 with minimum effort and investment but has an impact on the companywide level. This research also promotes the use of data as the basis in decision-making and strategic planning processes, not using instinct, convention, or any form of irrational process.

Keywords: Inventory Management, Multivariate Clustering, Machine Learning.

1. Introduction

In fast paced environment nowadays, companies are required to take decisions and act as fast as possible. During the execution of strategies, programs, and portfolios, companies need to put priority of one task to another. Or in certain cases, need to put one product on higher priority than the rest of the products. Each product has its importance

D. V. Ferezagia et al. (eds.), *Proceedings of the International Conference on Vocational Education Applied Science and Technology (ICVEAST 2023)*, Advances in Social Science, Education and Humanities Research 783,
https://doi.org/10.2991/978-2-38476-132-6_30

level different in each department. Each department defines their own criteria and categorization mechanism which makes some products not fall into the same priority when it comes to different departments. Each department treats the same product differently because they have different perspectives and factors to consider. Even a few departments might use irrational ways to categorize products. Companies need a uniform categorization based on multiple criteria considering multiple departments' needs.

The organization as the object of study is a Small Medium Enterprise (SME) in beauty industry which has been established for 12 years. The company has multiple sales channel: 5 branches located in Central Java and 3 platforms (Shopee, WhatsApp and Android). The company already implements an Information System and has a centralized database which keeps all patients' medical records as well as sales data in all branches. The use of Data Analytics in organizations proven can help a company in running business.

2. Literature Review

2.1 ABC Inventory

In inventory management, the concept of ABC inventory is carried out to group items based on their importance and value. ABC inventory is a method of classifying products in inventory into different groups to prioritize management efforts and allocate resources effectively[1]. The classification is basically implementing Pareto principle, which states that a small portion of products typically account for a large proportion of the total value or impact.

In ABC inventory technique, products are categorized into 3(three) classes: A, B, and C. Each class represents a different level of importance and requires a distinct management approach:

- Class A: This class contains the products with the highest value or impact. This class usually represents a small percentage of the total number of products but contributes to a huge percentage of the total inventory value. Managing Class-A items is crucial to the overall success of the business. They often include high-value products, critical components, or items in high demand. These items require close monitoring, frequent replenishment, and tighter control to ensure their availability and minimize out-of-stock.
- Class B: These products are of medium importance. They have a moderate value or impact compared to Class A products. They usually represent a moderate percentage of the total number of products and contribute to a relatively lower portion of the total inventory value. Managing Class B products requires a balanced approach. They may include products with moderate demand or value. These items need regular monitoring and adequate stock levels to meet customer needs without excessive inventory holding costs.

- Class C: These are the products with the lowest value or impact. They take up the big percentages of the total number of products but contribute to only the small portions of the total inventory value. Category C items often have lower demand, lower value, or longer shelf life compared to the A and B Class. They require less attention and can be managed with less frequent replenishment or more relaxed inventory control. Categorizing them as such helps focus resources on more critical items.

The goal of ABC inventory analysis is to prioritize resources and efforts by allocating more attention, monitoring, and control to Class A products, while adopting a more flexible approach for Class C items. By classifying inventory products based on their value and impact, businesses can optimize their inventory management strategies, improve customer service levels, and minimize costs associated with inventory carrying and out-of-stock problems.

2.2 XYZ Inventory

An XYZ stock analysis is a complement to an ABC analysis and adds a layer of statistical review that shows the standard deviation of usage. Other names for this analysis include Runners, Repeaters, or Strangers (RRS). The goal of this analysis is to understand which parts have steady usage and which parts have unpredictable demand so companies can make the best inventory decisions, successfully manage their shortages, and accurately determine order policies.

Using variations in demand, the XYZ model classifies goods as one of three categories [2]:

- X – Very little variation: X items are characterized by steady turnover over time. Future demand can be reliably forecast.
- Y – Some variation: Although demand for Y items is not steady, variability in demand can be predicted to an extent. This is usually because demand fluctuations are caused by known factors, such as seasonality, product lifecycles, competitor action or economic factors. It's more difficult to forecast demand accurately.
- Z – The most variation: Demand for Z items can fluctuate strongly or occur sporadically. There is no trend or predictable causal factors, making reliable demand forecasting impossible.

Based on these classifications, companies can use the demand forecast and XYZ material classification to determine optimal order schedules. X items should be ordered the most often—with low demand variation, buyers should be able to forecast demand accurately and place orders as often as daily. Y items should be ordered less frequently, keeping seasonal and other expected variations in mind. Lastly, Z items should be ordered the least frequently, as their demand levels are irregular and often unpredictable.

2.3 Hierarchical clustering

Hierarchical clustering is a clustering method that builds a hierarchy of clusters by recursively dividing or merging clusters based on the similarity or dissimilarity between data points. It does not require specifying the number of clusters in advance, and the resulting hierarchy can be visualized as a dendrogram, which represents the nested structure of the data [3]. Hierarchical clustering offers several advantages [4] that make it a popular approach in clustering analysis such as:

1. Hierarchy of Clusters
   Hierarchical clustering produces a hierarchy of clusters, often represented as a dendrogram. This provides a visual representation of the relationships between clusters at different levels of granularity. It allows for a more detailed understanding of the data structure and facilitates interpretation and decision-making.
2. Flexibility in Cluster Exploration
   Hierarchical clustering allows for exploring clusters at different levels of the hierarchy. By cutting the dendrogram at different heights, one can obtain clusters of varying sizes and granularity. This flexibility enables the identification of meaningful subgroups within the data and supports diverse analysis purposes.
3. No Need to Specify the Number of Clusters
   Hierarchical clustering does not require the prior specification of the number of clusters. It automatically generates a hierarchy of clusters based on the data's inherent structure and similarity measures. This is particularly useful when the number of clusters is unknown or when exploring different possible cluster solutions.
4. Handling Non-Linearly Separable Data
   Hierarchical clustering can handle data with non-linearly separable patterns. It can identify clusters of arbitrary shapes and sizes without assuming specific distributional assumptions. This makes it suitable for a wide range of datasets with complex structures and mixed characteristics.

2.4 Agglomerative Clustering

Agglomerative clustering is a hierarchical clustering algorithm that starts with each data point as an individual cluster and iteratively merges the most similar clusters until a single cluster containing all data points is formed. It belongs to the bottom-up approach, where clusters are successively combined based on their similarity or proximity [5]. The steps in agglomerative clustering are:

1. Initialization
   Begin by assigning each data point as an individual cluster, treating them as singletons.
2. Compute Pairwise Distances or Similarities
   Calculate the distance or similarity between each pair of clusters. The choice of distance or similarity metric depends on the nature of the data and problem domain. Common metrics include Euclidean distance for numeric data, Manhattan distance, or cosine similarity for text data.
3. Merge the Most Similar Clusters

Identify the two most similar clusters based on the computed distances or similarities. The definition of "similar" can vary, such as the smallest distance, largest similarity, or other linkage criteria like average or complete linkage.

4. Update the Distance Matrix
   Recalculate the pairwise distances or similarities between the newly formed cluster and the remaining clusters. This step is necessary to reflect the change in distances caused by the cluster merger.
5. Repeat Steps 3 and 4
   Repeat the merging and distance matrix update steps until all data points belong to a single cluster. The stopping criterion can be a predefined number of clusters or a specific threshold distance/similarity value.
6. Build the Dendrogram
   Construct a dendrogram, which is a tree-like diagram representing the hierarchy of clusters. Each node in the dendrogram represents a cluster, and the height of the node corresponds to the distance or similarity at which the clusters are merged.
7. Cut the dendrogram.
   Based on the desired number of clusters or the distance threshold, cut the dendrogram to obtain the final clusters. This cutting point determines the level at which the dendrogram is divided into distinct clusters.

2.5 Divisive Clustering

Divisive clustering is a hierarchical clustering algorithm that starts with a single cluster containing all data points and recursively divides the clusters into smaller subclusters until each data point is assigned to its own cluster [6]. It follows a top-down approach, where clusters are successively split based on dissimilarity. Here are the general steps involved in divisive clustering:

1. Initialization
   Start with a single cluster containing all data points.
2. Compute Dissimilarity
   Calculate the dissimilarity or distance between each pair of data points within the current cluster. The choice of dissimilarity metric depends on the nature of the data and problem domain. Common metrics include Euclidean distance, Manhattan distance, or cosine similarity.
3. Select a Cluster for Splitting
   Identify the cluster within the current hierarchy that exhibits the highest dissimilarity or greatest heterogeneity. Various techniques can be used to determine the most dissimilar cluster, such as comparing cluster centroids, cluster densities, or other dissimilarity measures.
4. Split the Cluster
   Divide the selected cluster into two or more subclusters. The specific splitting method depends on the algorithm being used. Common approaches include partitioning the cluster using a clustering algorithm (e.g., k-means) or recursively applying divisive clustering.

5. Repeat Steps 2-4
   Continue the process recursively on the newly formed subclusters. Compute dissimilarity, select the most dissimilar cluster, and split it until each data point is assigned to its own cluster.
6. Build the Dendrogram
   Construct a dendrogram, which represents the hierarchy of clusters. Each node in the dendrogram corresponds to a cluster, and the height of the node reflects the dissimilarity or heterogeneity at which the clusters are split.
7. Cut the Dendrogram
   The resulting dendrogram can be cut to a specific height to obtain a desired number of clusters or explore clusters at different levels of granularity.

2.6 Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is a hierarchical clustering algorithm designed for efficiently clustering large-scale datasets. It utilizes a tree-based data structure called the CF (Clustering Feature) tree to represent the data hierarchy. BIRCH combines clustering and data summarization techniques to build the CF tree, enabling fast and scalable clustering [7].

1. Initialization:
   a. Set the maximum number of clusters, branching factor, and threshold values for radius and diameter.
   b. Create an empty CF tree.
2. Insertion of Data Points:
   a. Process each data point one by one.
   b. Update the CF tree to incorporate the new data point.
   c. Calculate the distance between the data point and existing clusters in the tree.
   d. Insert the data point into the appropriate leaf node or create a new leaf node.
3. Clustering Feature Update:
   a. Update the clustering features (sums, sums of squares, and count) of the CF tree nodes affected by the insertion.
   b. Propagate the updates towards the root of the tree.
4. Clustering Feature Compression:
   a. Periodically check the clustering features of the CF tree nodes.
   b. If a node exceeds the specified threshold values for radius or diameter, perform local clustering on that node.
   c. Replace the node with a new higher-level node representing the cluster's summary.
5. Merge Clusters:
   a. If the number of clusters in the tree exceeds the maximum allowed number of clusters, merge the clusters.
   b. Determine the closest pair of clusters based on distance or similarity measures.
   c. Merge the clusters into a new higher-level cluster.

6. Construct Dendrogram:
   a. Traverse the CF tree to generate a dendrogram that represents the hierarchy of clusters.
   b. Each node in the dendrogram corresponds to a cluster, and the height of the node indicates the level of the hierarchy.

The resulting dendrogram can be cut to a specific height to obtain a desired number of clusters or explore clusters at different levels of granularity. BIRCH offers the advantage of reduced memory requirements and efficient processing for large datasets, making it suitable for scalable clustering tasks.

2.7 Clustering Using Representatives (CURE)

Clustering Using Representatives (CURE) is a hierarchical clustering algorithm that combines elements of partitional and hierarchical clustering approaches. It aims to efficiently and effectively cluster data points by using representative points and a multi-resolution clustering process [8]. The CURE algorithm can be described in the following steps:

1. Selection of Representative Points:
   a. Randomly select a set of representative points from the dataset.
   b. The number of representative points to be selected is typically determined in advance.
2. Data Point Movement:
   a. Move each data point towards the nearest representative point by a specific fraction.
   b. This movement ensures that the data points get closer to representative points, making them more representative of their local neighborhoods.
3. Hierarchical Clustering:
   a. Apply a hierarchical clustering algorithm, such as agglomerative clustering, to the set of representative points.
   b. The hierarchical clustering is performed based on the modified positions of the representative points after the data point movement step.
4. Cutting the Dendrogram:
   Cut the resulting dendrogram at a specific height to obtain the desired number of clusters.

The height at which the dendrogram is cut determines the granularity of the clusters. CURE offers several advantages, such as the ability to handle non-linearly separable data, scalability to large datasets, and the flexibility to handle clusters of varying sizes and shapes. By using representative points and the multi-resolution clustering approach, CURE can effectively handle noise and outliers in the data.

3. Research Methodology

This research methodology is using CRISP-ML[9] as a basis which consists of 6 stages: a) Business and Data Understanding, b) Data Preparation, c) Modelling, d) Evaluation, e) Deployment, and f) Monitoring and Maintenance. Since this is just initial research, we just take 4 phases only:

1. Business and Data Understanding

Before any other step takes place, we need to make sure we understand the business process of the organization. We conduct interview with all department in organization that has involved in product management; a) Vice of President, b) Purchasing Department, c) Marketing Department and d) Warehouse Department. We ask them what factors they have in mind when it comes to product categorization.

Data is acquired from the existing Information System the company used and owned. There are 378.647 sales data from February 2019 to March 2023 recorded in the system. The data is retrieved directly from MySQL Database using the following query:

```
SELECT
    `od`.`product_id `
    SUM(`od_`.`quantity`)
    SUM(`od_`.`subtotal`)
FROM
    ((`order_details` `od_`
LEFT JOIN `order` `or` ON ((`or`.`order_id ` =
```

The result of the above query is a set of 296 rows of sales data that are already grouped by product. Each row has its own corresponding number of products sold (quantity) and revenue generated (subtotal). Fig 1 shows 8 samples data out of all 296 products data.

	A	B	C
1	product_id	quantity	subtotal
290	389	4	Rp740.000
291	392	1	Rp187.000
292	398	3	Rp576.000
293	403	61	Rp4.270.000
294	404	2	Rp110.000
295	405	10	Rp150.000
296	406	2	Rp120.000
297	407	5	Rp350.000

2. Data Preparation

To make sure the data is valid, we do the following checks regarding to data quality:

- Void invoice
  We need to make sure all invoices are valid; invoices that have been voided or altered need to be eliminated from population.
- Test/Dummy data
  During the use of information, some data is not real sales data. Some data stored in the system are dummy data and generated during the testing phase of the system update.
- Inconsistency data
  Few products are registered under the different IDs, these data should be merged so 1 product only registered under the same product ID.

3. Modelling

During the modeling process, we are using Scikit Learn[10] to implement Hierarchical Clustering and following Classes are used: a) **AgglomerativeClustering** to implement Agglomerative Clustering algorithm, b) **KMeans** to implement Divisive Clustering algorithm, c) **Birch** to implement BIRCH algorithm, and d) **FeatureAgglomeration** to implement CURE.

4. Evaluation

There are 2 evaluation metrics we use in this research:

- Silhouette coefficient: This metric measures how well each data point fits into its assigned cluster and ranges from -1 to 1. A high silhouette coefficient indicates that the data points are well-clustered, while a low coefficient indicates that the data points may be assigned to the wrong cluster.
- Jaccard coefficient: This metric measures the similarity between the clustering results and the ground truth, considering the number of data points in each cluster.

4. Result & Discussion

After running the steps explained in Research Methodology, the following results are gained:

Table 1. ABC Inventory Classification

	Revenue		Product Number	
Class	Total	Ratio	Total	Ratio
A	Rp40,785,160	79.77%	23	7.88%
B	Rp4,971,696	9.72%	17	5.82%
C	Rp5,371,628	10.51%	252	86.30%
	Rp51,128,484		292	

From table 1 we can see that 292 items are clustered into 3 categories:

- Class A consists of only 7.88% of total products but contributes to the majority (79.77%) of total revenue.
- Class B consists of 5.82% of total products and inhibits 9.72% of total revenue.
- Class C consists of the majority (86.3%) of products but only contributes 10.51% of total revenue.

The following pie chart visualize the portion of each class in terms of product numbers and revenue:

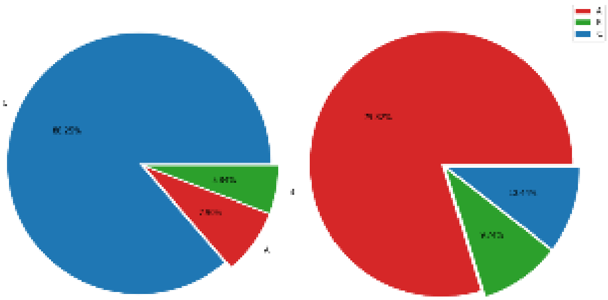

Fig. 1. ABC Inventory by Product Numbers (Left) and Revenue(right) using Agglomerative Clustering

When in come to XYZ classification, we acquired the results as displayed on Table 2:

- Class X consists of only 4 products and contributes to 0.23% of total revenue.
- Class Y consists of 9.25% of total products but inhibits the majority (72.99%) of total revenue.
- Class Z consists of the majority (69.52%) of products but only contributes 26.7% of total revenue.

Table 2. XYZ Inventory Classification

	Revenue		Product Number	
Class	Total	Ratio	Total	Ratio
X	Rp118,150	0.23%	4	1.37%
Y	Rp37,320,570	72.99%	27	9.25%
Z	Rp13,649,190	26.70%	203	69.52%
X0	Rp35,418	0.07%	30	10.27%
O	Rp5,153	0.01%	28	9.59%
	Rp51,128,482		292	

Running cross operation to ABC and XYZ classification we get the following two tables: Table 3 in terms of Product Numbers and Table 4 in terms of Revenue.

Table 3. ABC-XYZ Inventory Classification by Product Number

Class	X	Y	Z	X0	O
A		17	6		
B		2	15		
C	4	8	182	30	28

Table 4. ABC-XYZ Inventory Classification by Revenue

Class	X	Y	Z	X0	O
A		Rp36,553,400	Rp4,231,752		
B		Rp615,250	Rp4,356,446		
C	Rp118,150	Rp151,911	Rp5,060,994	Rp35,418	Rp5,153

The result is presented to the stakeholders across the company and used as base for prioritizing in any levels: operational, managerial, and operational. The following matrix summarize the ABC-XYZ product clustering:

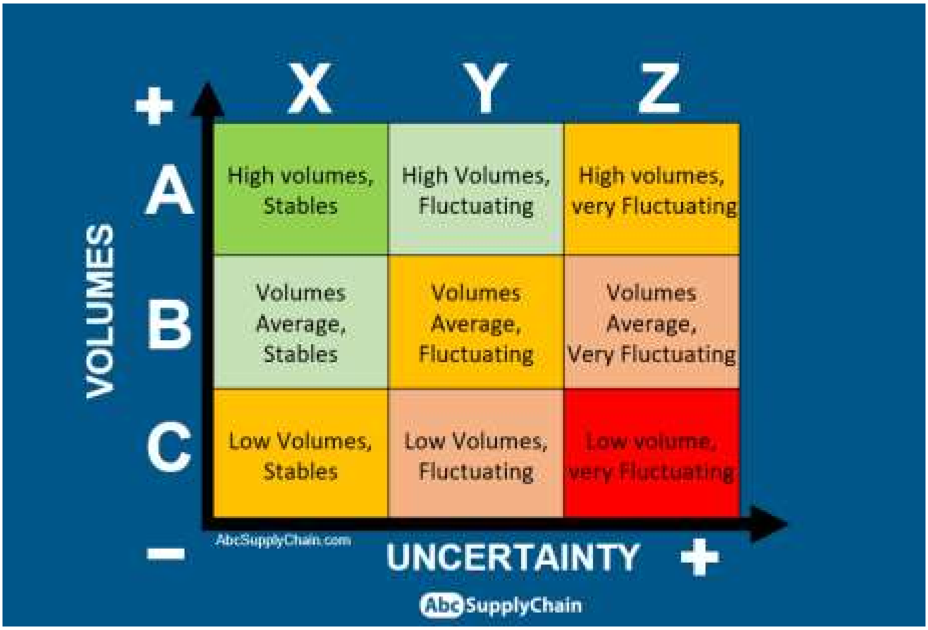

Fig. 2. ABC-XYZ Analysis Matrix[11]

5. Conclusion

The output of these research provides the company with an optimized level of inventory; not too much working capital or carrying costs are tied up in the X parts at once since they are used consistently, while enough is invested in the Y and Z parts so that stocking out will be minimized.

The implementation of this initiative on SME could be an inspiration for other SMEs or even bigger companies to implement the same methodology, concept, or model. This research shows how Machine Learning can be used in the company with minimum effort and investment but has an impact on the companywide level.

References

1. T. Wild, *Best practice in inventory management*. Routledge, 2017.
2. "Cost Transformation - XYZ inventory management," Aug. 03, 2018. https://www.aicpa-cima.com/resources/article/cost-transformation-xyz-inventory-management (accessed May 19, 2023).
3. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. USA: Prentice-Hall, Inc., 1988.
4. B. S. Everitt, S. Landau, M. Leese, and D. Stahl, *Cluster Analysis*. Wiley, 2011. doi: 10.1002/9780470977811.
5. R. Sibson, "SLINK: An optimally efficient algorithm for the single-link cluster method," *Comput J*, vol. 16, no. 1, pp. 30–34, Jan. 1973, doi: 10.1093/comjnl/16.1.30.
6. J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Appl Stat*, vol. 28, no. 1, p. 100, 1979, doi: 10.2307/2346830.
7. T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," *ACM SIGMOD Record*, vol. 25, no. 2, pp. 103–114, Jun. 1996, doi: 10.1145/235968.233324.
8. S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," *ACM Sigmod record*, vol. 27, no. 2, pp. 73–84, 1998.
9. S. Studer *et al.*, "Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology," *Mach Learn Knowl Extr*, vol. 3, no. 2, pp. 392–413, Apr. 2021, doi: 10.3390/make3020020.
10. F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
11. E. Thieuleux, "ABC XYZ Analysis in Inventory Management: example in Excel." https://abcsupplychain.com/abc-xyz-analysis/ (accessed May 19, 2023).