TecnoCampus
Escola Superior
Politècnica

Centre adscrit a la
upf. Universitat Pompeu Fabra Barcelona

Industrial Organization Engineer's Degree

Optimizing Pricing Strategies in Manufacturing through Big Data, Machine Learning, and Artificial Intelligence

An Analysis of Customer Segmentation using Machine Learning Clustering Techniques

Project Final Report

Jordi Ruiz Gratacós

SPRING 2023

TecnoCampus
Mataró-Maresme

Abstract

This project focuses on the development of a machine learning (ML) model to cluster a manufacturing company's clients according to their preferences, behaviours, and feedback. The final deliverables include the ML model and a comprehensive report explaining the methodology and outcomes. To substantiate the model's technical robustness, economic viability, and environmental feasibility, the study delves into existing technologies and applications. The financial analysis reveals that the project is estimated to incur a cost of approximately €3,921.91. The model is anticipated to quickly turn profitable owing to enhanced customer satisfaction and retention stemming from tailored services.

Resum

Aquest projecte se centra en el desenvolupament d'un model d'aprenentatge automàtic (ML) per agrupar els clients d'una empresa manufacturera d'acord amb les seves preferències, comportaments i hàbits de compra. Els materials del lliurament final inclouen el model ML i un informe exhaustiu que explica la metodologia i els resultats. Per justificar la robustesa tècnica del model, la viabilitat econòmica i la viabilitat mediambiental, l'estudi s'aprofundeix en les tecnologies i aplicacions existents. L'anàlisi financera revela que el projecte s'estima que té un cost d'aproximadament 3.291€. Es preveu que el model giri ràpidament rendible a causa de la satisfacció del client i la retenció derivada dels serveis personalitzats.

Resumen

Este proyecto se centra en el desarrollo de un modelo de aprendizaje automático (ML) para agrupar los clientes de una empresa manufacturera de acuerdo con sus preferencias, comportamientos y hábitos de compra. Los materiales de la entrega final incluyen el modelo ML y un informe exhaustivo que explica la metodología y los resultados. Para justificar la robustez técnica del modelo, la viabilidad económica y la viabilidad medioambiental, el estudio se profundiza en las tecnologías y aplicaciones existentes. El análisis financiero revela que el proyecto se estima que tiene un coste de aproximadamente 3.291€. Se prevé que el modelo gire rápidamente rentable a causa de la satisfacción del cliente y la retención derivada de los servicios personalizados.

Index.

List of figures.

List of tables.

Glossary of terms.

MA	Machine Learning
AI	Artificial Intelligence
DL	Deep Learning
DNNs	Deep Neural Networks
NNs	Neural Networks
ISO	International Organization for Standardization
UOM	Unit of Measure
GMM	Gaussian Mixture Model
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
PDM	Product Data Management
EMEA	Europe, Middle East, and Africa.
APAC	Asia-Pacific

1. Objectives.

1.1. Purpose.

The main objective of this project is to design a state-of-the-art ML model capable of segmenting the company's clients and providing valuable information for the pricing division. The model must demonstrate a high level of precision and to have the ability to offer solid suggestions for improving pricing policies and making smart pricing choices. Upon completion, the deliverables will include the fully built model and a comprehensive report which explains the approach used and the results of the project.

1.2. Object.

The objective of this project is to develop a ML model that can analyse the company's investment portfolio and provide pertinent insights for the pricing department. The model should have the ability to efficiently process a large volume of data, detect significant patterns and trends, and suggest optimal clustering groups in order to maximise the prising strategy. This project will entail a thorough investigation of the company's portfolio data, a detailed review of relevant ML methods and algorithms, and a selection of the most appropriate approach for this specific use case.

1.3. Scope.

The aim of this project is to create an ML model that can cluster customers based on their pricing preferences and behaviour. The project will involve the gathering and cleaning of the customer data, the design and implementation of the ML model, and a comprehensive assessment of the model's effectiveness and accuracy. The project will also involve documenting the approach and outcomes, as well as providing suggestions for future work. The scope of the project is restricted to the creation of the ML model and its application to the customer data. Any wider implications or suggestions for the general use of ML in pricing optimisation will be beyond the scope of the project.

1.4. Context in the lines of research and knowledge transfer of Tecnocampus.

The forthcoming student-led initiative to develop machine learning models for portfolio analysis and pricing optimization is closely aligned with the principles and objectives of an engineering degree in manufacturing and smart factories. This project will demonstrate the students' ability to apply theoretical engineering concepts to practical challenges, generating innovative and impactful solutions relevant to the manufacturing industry.

Participating in this project will significantly contribute to the students' development as future engineers in the field of manufacturing and smart factories. By utilizing machine learning technology in a real-world context, they will gain a deeper understanding of the critical role of data analysis and optimization in this field.

Moreover, the project will underscore the importance of the interplay between data analysis, model development, and optimization - all crucial components of the smart factory paradigm. It will serve as evidence of the students' potential as future contributors to the manufacturing industry, capable of adding value to organizations through their expertise in data analysis and optimization.

1.5. Gender perspective.

Our project is focused on developing a machine learning model to cluster our company's clients. As we develop this model, we must ensure that it does not perpetuate any biases or discrimination based on gender. Given that this segmentation project is targeting a B2B market we anticipate that the project will have minimal impact on gender equity.

In addition to minimizing the gender impact of the model itself, we are also dedicated to ensuring that the project has minimal impact on our current employees' organization. We plan to work closely with them to make sure that any changes resulting from the implementation of the model are carried out in a fair and equitable manner. By involving our employees in the process and being transparent about our goals and methods, we believe we can minimize any potential negative impacts on our organization knowing how important the customer management personnel is.

In summary, we are confident that this project will have a positive impact on both our company and our clients while minimizing any potential negative impacts on gender equity and our current employees’ organization.

2. Background.

2.1. Artificial intelligence.

AI is a broad field that encompasses various areas of computer science and engineering. The origins of AI can be traced back to the mid-20th century when scientists and engineers began exploring the idea of creating intelligent machines.[1] The primary goal of AI is to create systems that can perform tasks that normally require human intelligence, such as recognizing speech, making decisions, and solving problems.

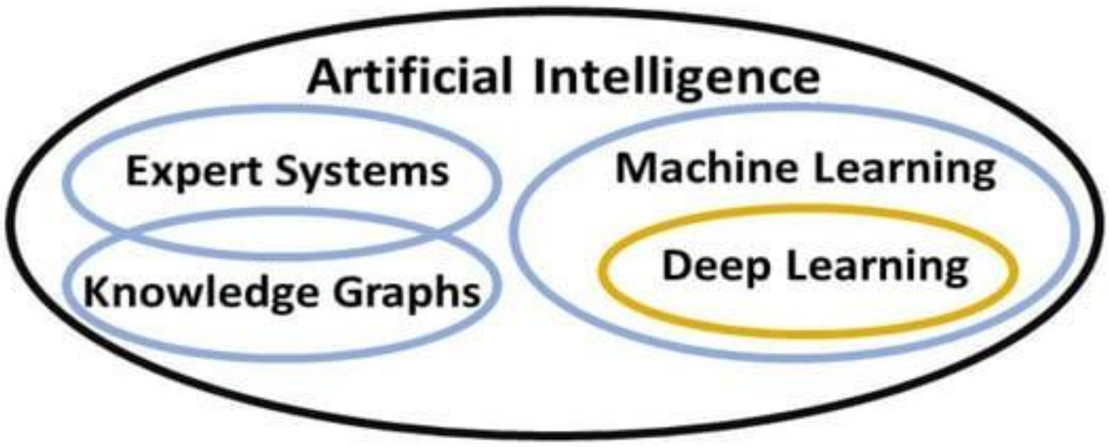


Fig 2.1. AI containing ML, DL, Expert Systems, and Knowledge Graphs.[2]

AI systems work by processing large amounts of data and using algorithms to identify patterns and relationships. There are several types of AI algorithms, including rule-based systems, decision trees, neural networks, and DL algorithms. Each of these algorithms has different strengths and weaknesses and is suitable for different types of tasks. For example, DL algorithms are particularly well-suited for tasks such as image and speech recognition, while decision trees are often used in predictive analytics.[3]

AI is rapidly advancing and is already having a significant impact on many industries, including healthcare, finance, and transportation. Soon, it is expected that AI will play an even greater role in shaping our world. For example, it is predicted that AI will be used to improve the accuracy of medical diagnoses, increase the efficiency of business processes, and enhance the quality of life for people with disabilities. As AI continues to evolve, it will be interesting to see how it continues to shape the world and improve our lives.

2.2. Machine learning.

ML is a field of AI that focuses on the design of algorithms that can learn from data and make predictions or decisions without being explicitly programmed. The origins of ML can be traced b ack to the 1950s and 1960s, when researchers first started exploring the idea of teaching computers to learn from data.[4]

ML algorithms use mathematical models to find patterns in data and use this information to make predictions or decisions. These algorithms are trained on large amounts of data, allowing them to identify relationships and insights that are not immediately apparent to humans.[5] There are various types of ML algorithms, including supervised learning, unsupervised learning, and reinforcement learning.

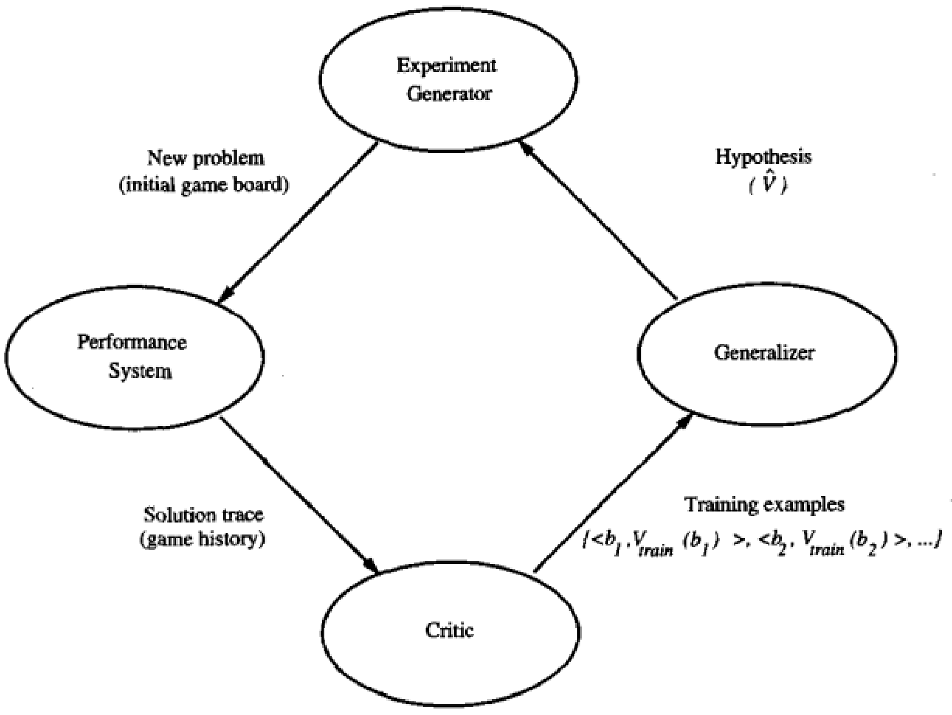

Experiment Generator
New problem (initial game board)
Hypothesis ($\hat{V}$)
Performance System
Generalizer
Solution trace (game history)
Training examples
$\{<b_1, V_{train}(b_1)>, <b_2, V_{train}(b_2)>, ...\}$
Critic

Fig 2.2. ML structure diagram.[6]

Supervised learning algorithms are trained on a labelled dataset, where the correct answer is known. These algorithms are used in tasks such as image classification, where the algorithm must identify the objects in an image, or speech recognition, where the algorithm must transcribe spoken words. Unsupervised learning algorithms are used when the correct answer is not known, and the algorithm must find patterns in the data on its own. These algorithms are often used for clustering, where the goal is to group similar data points together.

Reinforcement learning algorithms are used in decision-making tasks, where the algorithm must make a series of decisions based on rewards and punishments.[7]

The technology of ML has advanced rapidly in recent years, driven by the availability of large amounts of data and the increasing power of computers. ML algorithms are now used in a wide range of applications, including image and speech recognition, natural language processing, and self-driving cars.[8] As the technology continues to advance, it is likely that ML will play an even larger role in our lives in the future, with applications in areas such as healthcare, finance, and education.

2.3. Deep learning.

DL is a subfield of machine learning that is concerned with creating artificial neural networks that can learn and make decisions or predictions on their own. It uses algorithms inspired by the structure and function of the human brain, known as artificial neural networks, to process and analyse large amounts of complex data.

DNNs, or Deep Neural Networks, are a type of neural network that have multiple layers between the input and output layers, allowing for more sophisticated and complex analysis of data. They are derived from the standard Neural Networks (NNs), which consist of interconnected artificial neurons that operate through mathematical functions.

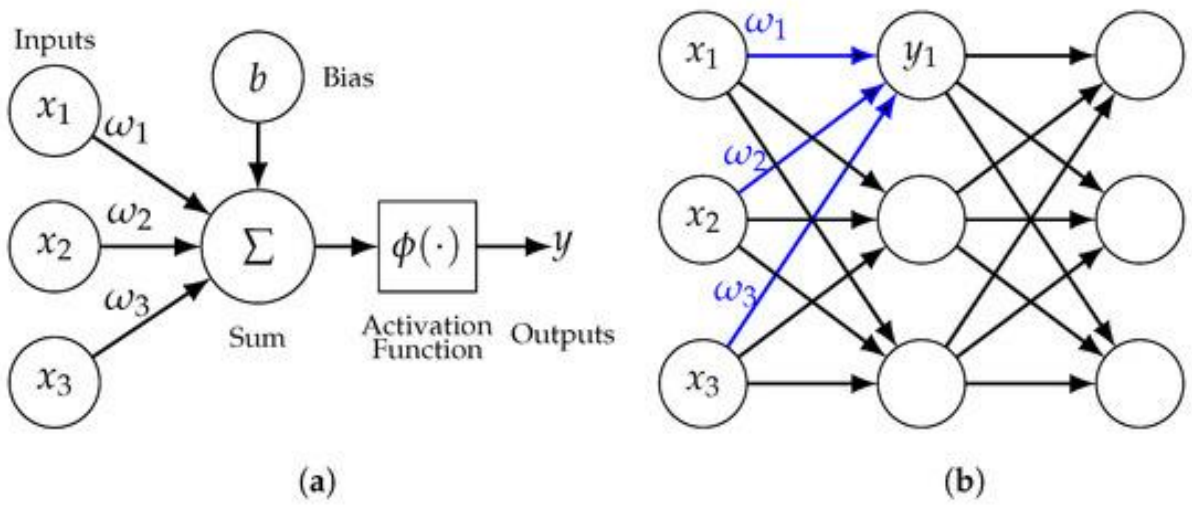


Fig 2.3. The output of a neuron in a neural network is calculated as the sum of all input elements, bias, and a non-linear function applied to the weighted sum.[2]

In DL, these artificial neural networks are trained on massive amounts of data, allowing them to identify patterns and relationships in the data, and then make predictions or decisions based on those patterns. This makes deep learning particularly useful for tasks such as image and speech recognition, natural language processing, and autonomous systems.

2.4. Data processing.

2.4.1 Data acquisition.

ML has been greatly impacted by the growth of data in recent years. ML algorithms are designed to learn from data, and the more data that is available, the better the algorithms can perform.[9] The volume of data generated by sources such as sensors, the internet of things, and e-commerce platforms has increased dramatically, providing an abundance of data for ML algorithms to learn from.

Fig 2.4. Visual representation of a data export from the ERP with a laptop.[10]

Data acquisition involves collecting data from various sources, such as databases, APIs, and flat files. The data must then be transformed into a format suitable for use by ML algorithms. This process can be time-consuming and requires specialized skills, but it is essential for the effective use of ML in a variety of applications.

In the pricing and manufacturing fields, the growth of data has enabled ML algorithms to be applied more effectively. For example, in the manufacturing industry, data can be acquired from sensors and production logs, which can then be transformed into a format that is suitable for ML algorithms to learn from. This data can be used to optimize manufacturing processes and reduce costs. In the pricing domain, data can be acquired from historical sales data, competitor pricing information, and market trends, which can then be transformed into a format that ML algorithms can use to dynamically set prices that maximize revenue and profits.

In conclusion, the growth of data has been a major factor in the development of ML technologies. Data acquisition stages are critical components of the process of using ML algorithms, and advances in these areas will continue to drive the growth of ML and its use in various industries.

2.4.2 Data preparation.

Data preparation is a crucial step in developing ML models, as the quality and relevance of the data directly affects the accuracy of the model. The process involves cleaning the data to remove missing values, outliers, or inconsistencies, transforming it into a suitable format, and splitting it into a training and test set.

Cleaning the data involves techniques such as imputation for missing values, where missing values can be estimated based on the mean, median or mode of the data, or based on more advanced techniques such as multiple imputation. Outliers can be detected and removed through statistical methods, such as Z-scores, or through visualization techniques, such as box plots. Inconsistencies in the data can be resolved by transforming the data into a consistent format, such as converting all the dates into a common format.[11]-[13]

Data transformation involves converting categorical data into numerical data, normalizing the data, and creating new features through feature engineering. This can be accomplished through techniques such as one-hot encoding for categorical data, normalizing the data to bring it to a common scale, and creating new features based on domain knowledge or by combining existing features. [14][15]

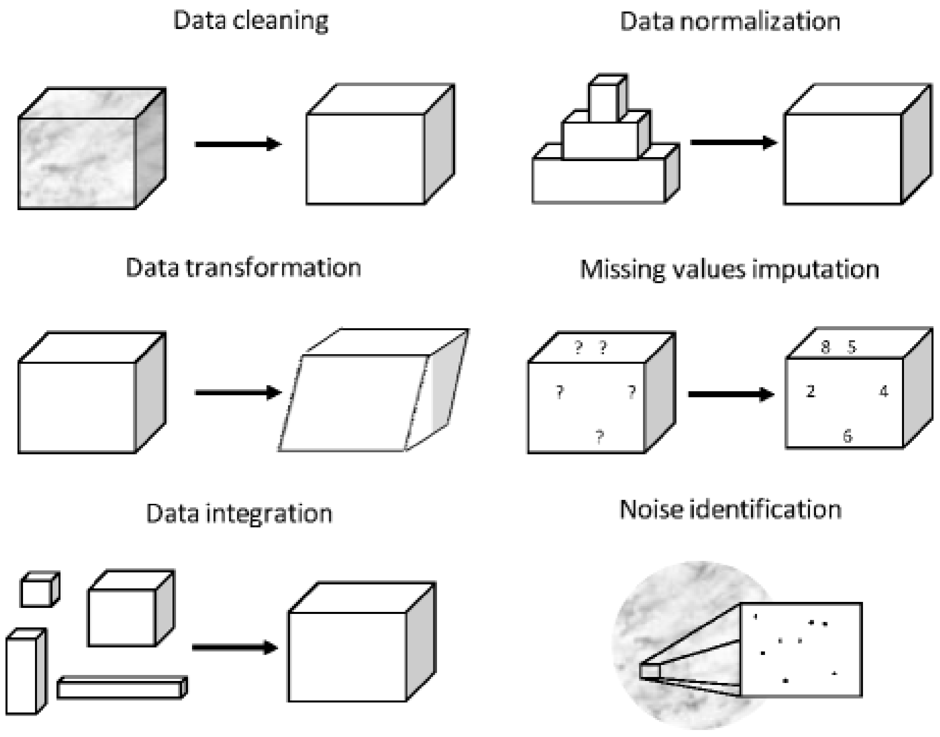


Fig 2.5. Data preparation techniques.[16]

The data is split into a training and test set, with the training set used to train the model and the test set used to evaluate its performance. This split helps to prevent overfitting, where the model is too complex and fits the training data too well but does not generalize well to unseen data.

2.5. Model types.

Modelling techniques in ML play a crucial role in the development of autonomous tools for clustering in manufacturing. There are several techniques that are commonly used in this field, including: [17-20]

Make this text a couple of sentences longer in each paragraph adding valuables trusted and verified information:

1. K-means clustering: This is a simple and popular technique used to partition a set of data points into k clusters. It is often used to identify patterns and group similar data points together. K-means clustering can be used in various domains, such as banking, recommendation engines, cyber security, document clustering and image segmentation. It is typically applied to data that has a smaller number of dimensions,

is numeric and is continuous. K-means clustering works by minimizing the sum of squared distances between data points and their assigned cluster centroids.

2. Hierarchical clustering: This technique involves building a hierarchy of clusters by either merging smaller clusters into larger ones or dividing larger clusters into smaller ones. In the context of manufacturing, it could be used to group similar products or processes together. Hierarchical clustering can also be used to analyse genetic or biological data, such as creating dendrograms to represent mutation or evolution levels. Hierarchical clustering can use any valid distance measure between data points and any linkage criterion to determine the dissimilarity between clusters.
3. Density-Based Spatial Clustering of Applications with Noise: This technique involves identifying clusters of data points based on their density. It is particularly useful for identifying clusters of arbitrary shapes and for handling noise and outliers in the data. DBSCAN can be applied to spatial data analysis, such as detecting regions of high traffic or finding earthquake epicentres. DBSCAN can also be used for image segmentation, anomaly detection and social network analysis. DBSCAN works by finding core points that have at least a minimum number of neighbours within a given radius and forming clusters around them.
4. Gaussian Mixture Models (GMM): This technique involves fitting a mixture of Gaussian distributions to the data to model the underlying clusters. It can be used to model complex relationships between variables and to identify clusters with different shapes and sizes. GMMs can be used for density estimation, classification, feature extraction and image segmentation. GMMs can also handle mixed data types, such as continuous and categorical variables. GMMs work by using the expectation-maximization algorithm to estimate the parameters of the Gaussian components.
5. Additionally, a DL model is tested. This is not the typical problem to approach with neural networks yet there is research on how this branch of AI could increase the capabilities of ML segmentation. The model chosen is called SpectralNet, building on top of spectral clustering it aims to overcome some of the limitations of the base model with the use of DL. [17-20]

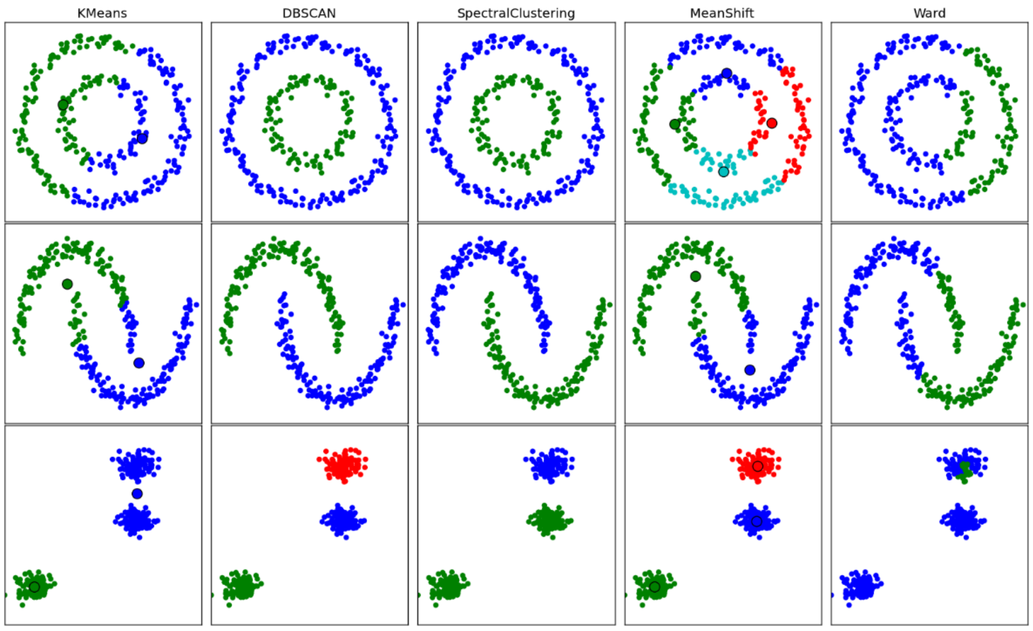


Fig 2.6. Clustering models comparison.

To sum up, clustering methods are among the most widely used unsupervised learning techniques in ML, and they offer a strong set of tools for building ML solutions for pricing in manufacturing. The selection of the best method will depend on the specific needs of the problem at hand, but any of these methods can deliver useful insights and forecasts in the pricing scenario.

2.6. Real world examples.

Clustering or unsupervised learning can be used for various purposes, such as data exploration, customer segmentation, recommender systems, target marketing campaigns, and data preparation and visualization. In this text, we will present five real-world applications of their use.

1. Identifying Fake News: Fake news is a serious problem that can misinform and manipulate the public opinion. To combat this issue, some researchers have proposed using clustering algorithms to identify fake news based on the content of the articles. The algorithm works by examining the words used in the articles and clustering them based on their frequency and co-occurrence. Then, it compares the clusters with those of genuine news articles and assigns a probability score for each article being fake or not. This method can help filter out sensationalized, click-bait, or phishing articles from credible sources. [21]

2. Spam Filtering: Spam emails are not only annoying but also potentially dangerous, as they may contain malware or phishing links. To protect users from spam emails, many email services use clustering algorithms to classify emails as spam or not based on their content and metadata. The algorithm works by extracting features from the emails, such as words, phrases, sender address, subject line, etc., and clustering them based on their similarity or dissimilarity. Then, it assigns a label to each cluster based on the proportion of known spam emails in that cluster. This way, it can automatically detect and filter out new spam emails that belong to the same cluster as previous ones. [22]
3. Customer Segmentation: Customer segmentation is a marketing strategy that divides customers into groups based on their characteristics, behaviours, preferences, or needs. It can help businesses tailor their products, services, prices, promotions, and communications to different customer segments and increase their customer satisfaction and loyalty. To perform customer segmentation, many businesses use clustering algorithms to analyse customer data, such as demographics, purchase history, browsing behaviour, feedback, etc., and group customers into clusters based on their similarity or dissimilarity. Then, they can assign a profile to each cluster and target them accordingly. [23]
4. Recommender Systems: Recommender systems are systems that suggest items or services to users based on their preferences or needs. They can be found in various domains, such as e-commerce, entertainment, education, travel, etc. To build recommender systems, many companies use clustering algorithms to analyse user data, such as ratings, reviews, preferences, browsing history, etc., and item data, such as features, descriptions, categories, etc., and group users and items into clusters based on their similarity or dissimilarity. Then, they can recommend items from the same cluster as the user's previous choices or preferences or items that are popular among similar users. [24]
5. Data Preparation and Visualization: Data preparation and visualization are essential steps in any data analysis project. They involve cleaning, transforming, summarizing, and presenting data in a meaningful way. To facilitate data preparation and visualization, many analysts use clustering algorithms to explore and understand data better. The algorithm works by grouping data points into clusters based on their similarity or dissimilarity and highlighting the patterns or outliers in the data. Then, it can generate summary statistics or visual representations for each cluster or for the

whole data set. This can help analysts gain insights into the data structure and distribution and make informed decisions about further analysis or modelling. [25]

2.7. Legislation.

The legislation surrounding the use of AI in the manufacturing sector is still in its infancy and varies greatly between different countries and regions. Despite this, there are several international organizations and governmental bodies that are working towards creating a comprehensive set of guidelines and regulations for the safe and ethical use of these technologies.

One such organization is the European Union's (EU) General Data Protection Regulation (GDPR), which sets strict standards for the collection and use of personal data by businesses operating within the EU. Additionally, the EU has established the European AI Alliance, which aims to create a comprehensive framework for the development and use of AI in the region.[26]

Another important player in the development of AI legislation is the International Organization for Standardization (ISO), which has established a working group dedicated to the creation of international standards for AI. The goal of this working group is to ensure that AI systems are designed, developed, and deployed in a safe, transparent, and trustworthy manner.[27][28]

Finally, the Organization for Economic Cooperation and Development (OECD) has created a set of principles for the responsible development and use of AI, which provide guidelines for the ethical and safe deployment of these technologies. These principles focus on issues such as privacy, accountability, and transparency, and aim to ensure that AI systems are developed and used in a way that is consistent with societal values and norms.[29]

Overall, while there is still much work to be done in terms of creating comprehensive legislation for the use of ML in manufacturing, there are several organizations and initiatives that are working towards this goal, and it is likely that we will see continued progress in this area in the coming years.

2.8. Environment and pricing.

The world economy and manufacturing industry are currently marked by both expansion and difficulties. Technological advancements, automation and globalization have led to increased production efficiency and lower costs, driving overall economic growth. However, challenges have arisen due to intense competition in the global marketplace, fluctuations in exchange rates and raw material costs, and rising labour costs. These challenges result in price volatility and inflation, which can negatively impact purchasing power and economic growth. To mitigate these effects, monetary policies aimed at controlling inflation are implemented, but they can also have unintended consequences.

The International Monetary Fund (IMF) World Economic Outlook Update report from January 2023 projects that the global economy is on track for a moderate recovery, with a growth rate of 5.5% in 2022 and 4.2% in 2023. The report cites the accelerated pace of vaccinations and the implementation of fiscal and monetary support measures as factors contributing to the recovery, but also notes that the ongoing impact of the COVID-19 pandemic and ongoing fiscal and monetary support measures will weigh on future growth. The report also highlights significant disparities in the recovery across countries and suggests a need for further policy support, particularly in low-income countries, to ensure a more equitable recovery.[30]

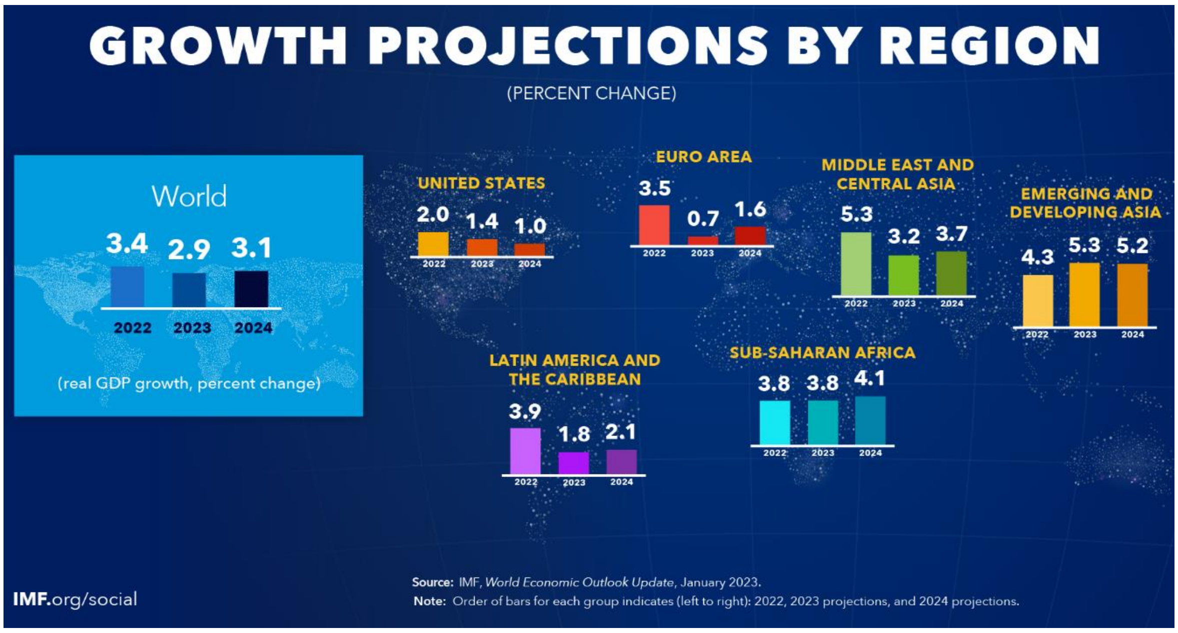


Fig 2.7. IMF growth projections by region.[30]

Therefore, managing selling prices correctly remains a crucial issue in maintaining profitability in the world economy and manufacturing industry. One potential solution to these challenges is to implement pricing actions. These can include strategies such as cost-plus pricing, value-based pricing, and market-oriented pricing. Cost-plus pricing involves adding a markup to the cost of production to arrive at the final price, while value-based pricing considers the perceived value of the product or service to the customer. Market-oriented pricing involves analysing the prices of competitor products to determine the optimal price for a company's offering. Additionally, companies can also implement dynamic pricing, where prices are adjusted in real-time based on changes in supply and demand. By implementing these pricing actions, companies can ensure that their prices are competitive, reflective of their costs, and aligned with market conditions. This can help to stabilize prices and reduce the impact of inflation on the economy.

2.9. The company.

Velcro Brand is a global company that not only prioritizes customers and quality but also brings ORIGINAL THINKING™ to everything they do. More than just the maker of the original hook and loop fastener, their story begins with its invention over 60 years ago. This invention has transformed the world one strip at a time and continues to inspire them today.

Fig 2.8. Velcro Brand corporate logo

I have been working as a pricing trainee in Velcro for the last year and a half mainly offering support to the department when tackling day-to-day requests. Additional tasks handled have been providing the commercial team of updated prices reports for customers while verifying the correct price alignment with the company's goals, tackling analytic projects on past data such as batch cleaning price lines in coordination with PDM, ensuring enhanced functionality developments from Oracle or Salesforce teams related to pricing requests are accomplished.

As part of my internship, I purposely asked for an opportunity to focus my final thesis on some aspect related to the company and making use of my ML, big data, analytics, and coding skills. At the start of the project aimed to design and develop an ML model to analyse the company's portfolio and provide recommendations for pricing optimization. The model would have had to processes large amounts of data to identify key trends and patterns. However, after reviewing the source data and an increased interest on reviewing the customer classification system at Velcro it was deemed more suitable to reassess the project.

The company is open about testing new technologies in house first before looking at the market for more robust solutions, thus acknowledging that the results of this project will have a limited outcome on the current organisation. Nevertheless, it is always good practice to verify the efficiency of current systems and being open to change and making use of new tools at their disposal.

2.10. Data confidentiality.

The results of the clustering can be presented to an external individual in a way that does not disclose any confidential information by using visualizations and summary statistics that provide an overview of the segmentation without revealing specific details about individual customers.

For example, the company could create a visualization that shows the distribution of customers across different segments, without providing any information about the specific characteristics of the customers in each segment. This could be done using a bar chart or pie chart that shows the relative size of each segment.

Additionally, the company could provide summary statistics for each segment, such as the average transaction value or the average customer lifetime value. These statistics provide an overview of the characteristics of each segment without revealing any confidential information about individual customers.

3. Project structure.

Organizing ML projects involves defining the problem, planning the project structure, creating a version control system, documenting work, monitoring model performance, using appropriate evaluation metrics, and regularly updating the project plan.[31]

3.1. Data collection and refinement.

When collecting and doing a first overview of the data for the project, a systematic and methodical approach will be followed. In this project the data must be exported from the enterprise’s resource planer, from the order line organiser precisely. The scope of data to get a representative outcome has been set from 2021 to 2022, therefore a two-year period to account for temporality in the purchasing behaviour of the customer. This means that our data set contains more than 250.000 observations, perfectly suitable for the models.

The raw data comes from an export of orders from the 2021 and 2022, each line contains different valuable fields, in this project this includes customer`s name and id, item code number, selling price, UOM, currency, quantity UOM adjusted, invoice date, customer segment, customer category… in total we have more than 20 different variables to work with. Therefore, the data set must be reviewed to avoid any unwanted deviation due to blank or outlier observations, that can introduce noise and fussiness to the results.

The set is then analysed and processed to make it suitable for use in the models. It is required to standardize or normalise it before feeding the models which also means that all variables must be encoded since only integers can go through that process. Once all data is prepared to

3.2. Dimensionality reduction.

Dimensionality reduction is a powerful technique used in machine learning and data analysis to simplify complex datasets and reveal underlying patterns and relationships. By reducing the number of variables or features in a dataset while retaining as much of the original information as possible, it is possible to gain a deeper understanding of the data and make more informed decisions.

One common method for dimensionality reduction is Principal Component Analysis (PCA). PCA works by identifying the directions in which the data varies the most and projecting it onto a lower-dimensional space defined by these directions. This can help to remove noise and redundancy from the data and reveal the most important features.

Another popular method is UMAP, a dimension reduction technique that can be used for visualization similarly to t-SNE. It seeks to preserve the global structure of the data, making it useful for tasks such as clustering. This makes it particularly effective at preserving both local and global structure in the data.

T-SNE (t-Distributed Stochastic Neighbour Embedding) is a non-linear dimensionality reduction technique that is particularly effective at preserving the local structure of the data. It constructs a probability distribution over pairs of high-dimensional objects and then maps them to a lower-dimensional space in such a way that similar objects remain close together while dissimilar objects are pushed apart.

In conclusion, dimensionality reduction techniques such as PCA, UMAP and t-SNE can be powerful tools for simplifying complex datasets and revealing underlying patterns and relationships. By using these techniques in conjunction with machine learning algorithms, it is possible to gain a deeper understanding of the data and make more informed decisions.

3.3. Model selection.

In this project, we will employ multiple models to analyse the data and generate the new segmentation based on the order history. The selection of these models is of utmost importance for ensuring a proper grouping for better economic results, and it is crucial to understand the strengths and weaknesses of each model to make an informed decision.

1. K-means is a popular clustering algorithm that partitions data into k clusters based on their similarity. One strength of K-means is its simplicity and ease of implementation. It is also computationally efficient, making it suitable for large datasets. However, one weakness of K-means is that it assumes clusters are spherical and equally sized, which may not always be the case in real-world data. Additionally, the number of clusters k must be specified in advance, which can be challenging to determine.
2. A Gaussian Mixture Model (GMM) is a probabilistic model that assumes data is generated from a mixture of several Gaussian distributions. One strength of GMM is its flexibility in modelling complex data distributions. It can also provide a measure of

uncertainty in cluster assignments. However, one weakness of GMM is that it can be sensitive to initialization and may converge to a local optimum.

3. DBSCAN is a density-based clustering algorithm that groups together points that are closely packed together and marks points that lie alone in low-density regions as outliers. One strength of DBSCAN is that it does not require the number of clusters to be specified in advance and can handle clusters of different shapes and sizes. However, one weakness of DBSCAN is that it may not perform well on data with varying densities or high-dimensional data.
4. Spectral Net is a deep learning-based clustering algorithm that uses spectral clustering to partition data into clusters. One strength of Spectral Net is its ability to handle non-linearly separable data and clusters of different shapes and sizes. It also does not require the number of clusters to be specified in advance. However, one weakness of Spectral Net is that it can be computationally expensive and may require a large amount of training data.

The use of a machine learning model for customer segmentation provides a comprehensive understanding of our customer base. It will help us identify the most crucial characteristics that define different customer segments and provide accurate predictions of customer behaviour, making it easier for the marketing and pricing team to make informed decisions. It is essential to consider the strengths and limitations of the model and choose the appropriate one for the specific needs of the project to guarantee the accuracy and reliability of the results.

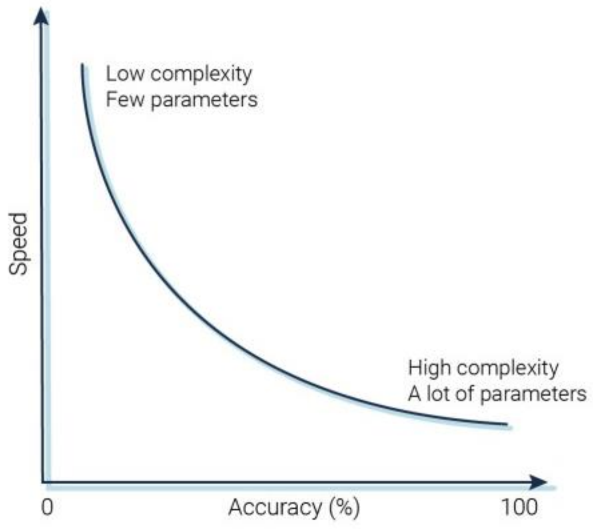


Fig 3.1. Complexity of the model has correlation between the speed of the model against its accuracy.[31]

3.4. Clustering evaluation.

One approach to cluster evaluation is to calculate cluster validation metrics, such as the silhouette score or the Davies-Bouldin index. These metrics provide a quantitative measure of how well-separated the clusters are and how cohesive the data points within each cluster are. Additionally, the characteristics of each cluster are analysed by calculating summary statistics for each variable within each cluster and comparing them to identify any meaningful differences between the clusters. Visualizations such as box plots or scatter plots can also be useful in gaining a deeper understanding of the characteristics of each cluster.

Furthermore, collaboration with domain experts can provide valuable insights into whether the resulting clusters align with business objectives and make sense from a practical perspective. This helps ensure that the clustering analysis produces results that are not only statistically sound but also meaningful and actionable in a real-world context.

3.5. Deployment.

The deployment of the ML model in this project consists of providing insights for the pricing team of the current customer portfolio and its different characteristics, being able to maximise the offering to the different needs on each client while maintaining a consistent pricing strategy. The model will be used to analyse the data and predict customer trends based on buying behaviour.

Once the best model has been selected through the evaluation process, it can be deployed in a production environment. This may involve integrating the model into existing systems, such as a database or a web application, or creating a standalone application specifically designed to use the model.

The deployment of the ML model results will provide the pricing team with valuable insights into pricing. By utilizing the model's predictions in real-world applications, the team will be able to optimize their pricing strategies, increase competitiveness in the market, and make informed decisions about customer optimal pricing and offering of products. The model's ability to analyse market trends and provide data-driven insights will allow the team to stay ahead of the curve and drive positive outcomes for their organization.

4. Feasibility.

4.1. Technical feasibility.

4.1.1. Technical roadmap.

For a project aimed at creating an AI tool for pricing in a manufacturing company, utilizing a data set containing less than 5,000 observations, the necessary hardware and software components have been identified. It is recommended that a laptop or desktop computer with a moderate to high-performance CPU and at least 8 GB of RAM is used to ensure sufficient processing power. Additionally, to store the data set and intermediate results, an external hard drive or cloud storage service can be utilized.

With regards to software, the programming language of choice for this project will be Python. Python is a popular language in the AI and machine learning community due to its ease of use, robust libraries, and clear syntax. Additionally, it has a strong community that provides support, documentation, and tutorials, making it ideal for projects of this type.

For the AI component of the project, several libraries and frameworks will be necessary, including scikit-learn, TensorFlow, and PyTorch. These libraries provide a vast array of functions and algorithms for machine learning and deep learning, allowing the user to train, evaluate, and test the AI tool. Furthermore, a development environment such as Jupyter Notebook or Spyder will also be required to support the coding and testing process.

Data pre-processing and cleaning are critical stages in the development of an AI tool. Without proper preparation of the data set, the results of the AI tool may be unreliable or biased. To this end, tools such as pandas or OpenRefine can be utilized to clean and prepare the data set for analysis. Additionally, data visualization tools, such as Matplotlib, Seaborn, or Plotly, will be necessary to effectively analyse the data and interpret the results. These tools allow the user to create charts, graphs, and plots to visualize the data, making it easier to identify patterns and trends.

Finally, statistical analysis and modelling tools, such as scikit-learn, statsmodels, or caret, will be necessary to build and evaluate the AI tool. These tools allow the user to create models that can be trained and tested on the data set. Model evaluation and selection is a crucial step in the development process, and cross-validation techniques or metrics such as accuracy, precision, recall, and F1 score can be used to evaluate the model's performance. These metrics provide an objective measurement of the model's performance, allowing the user to determine the best model for the specific requirements of the project.

In conclusion, this list of hardware and software components assumes that the data set is small, and the complexity of the project is limited. If the project increases in size and complexity, it may be necessary to scale up the hardware and software components. In such cases, consulting with a data science or IT professional is highly recommended to ensure that the appropriate tools are being used for the specific requirements of the project.

4.1.2. Available resources.

The Lenovo ThinkPad T460s laptop is a suitable choice for training small ML models due to several key features. Firstly, the laptop is equipped with a powerful 6th Generation Intel Core i5 processor, which provides the processing power necessary for running complex ML algorithms and training models. With 8GB of DDR4 RAM, the laptop has sufficient memory to handle the demands of training small data sets.[32]-[34]

Additionally, the laptop comes with a 256GB SSD for storage, which provides enough space to store the data used for training, as well as any resulting models. The Full HD IPS display with a resolution of 1920 x 1080 pixels provides clear and detailed visual output, making it easier to monitor the training process and assess the performance of the models.

In addition, Windows is a technically feasible operating system for carrying out an AI-powered pricing tool project. This is due to its wide availability and well-established ecosystem of hardware and software vendors, which make it easily accessible for organizations and individuals who want to implement AI-powered solutions. Additionally, Windows integrates well with Microsoft tools, such as Excel. Furthermore, Windows supports popular programming languages, including Python, which will be used in this machine learning project, making it easier to develop and implement ML algorithms and models. Additionally, Windows has robust security features, such as Windows Defender, which protect against malware and other cyber threats, which is critical for ML projects as they often involve sensitive data that must be protected from unauthorized access. These factors contribute to the technical feasibility of Windows as an operating system for carrying out an AI-powered pricing tool project.

In conclusion, the Lenovo ThinkPad T460s is capable of handling small machine learning and artificial intelligence projects. Its processor, memory, and storage provide the necessary resources for training small models.

4.1.3. WinPython.

WinPython is a free open-source distribution of the Python programming language for Windows operating systems. It includes a suite of tools and libraries for scientific computing, data analysis, and machine learning, such as NumPy, SciPy, pandas, and scikit-learn. Additionally, WinPython includes a pre-configured development environment, IPython, and the Spyder integrated development environment (IDE), making it a comprehensive and convenient package for Windows users who want to work with Python for scientific computing, data analysis, and machine learning.[35]

Fig. 4.1. Contents included in the WinPython package.[35]

WinPython is designed to be portable and can be run from a USB drive without the need for installation. This makes it easy for IT departments to manage the installation and lock its connectivity capabilities to ensure cyber security within the work environment.

Overall, WinPython is a useful tool for Windows users who want to use Python for scientific computing, data analysis, and machine learning, and provides a convenient and easy-to-use package that includes all the necessary components.

4.2. Economic feasibility.

The economic feasibility of the AI-powered pricing tool project has been analysed in the economic study has drawn the following conclusions. First, the estimated investment cost is 8,927.68€ detailed in the budget section which provides a comprehensive overview of the financial resources required to successfully complete the project. The performance analysis of the project is not fully determinable currently, but it is expected to be profitable due to increased efficiency, better decision-making, and reduction in workload for the pricing manager. However, the accuracy of the performance analysis will depend on the definition of the lost opportunity revenue. Overall, the implementation of the AI-powered pricing tool in the pricing department is expected to bring financial benefits and improve the decision-making process.

4.3. Environmental feasibility.

This analysis focuses on evaluating the environmental impact of implementing an AI-powered pricing tool in a pricing department. The environmental impact of the project will be assessed by considering the energy consumption, carbon footprint, and waste reduction potential of the project.

The training of machine learning models can be computationally intensive and consume a significant amount of energy. The energy consumption of the computing resources used for the project will be estimated and compared to industry standards to determine the environmental impact of the project. If the energy consumption is deemed excessive, alternative solutions, such as cloud computing or energy-efficient hardware,

The carbon footprint of the project will be estimated. This will consider not only the energy consumption of the computing resources but also the transportation costs associated with the project. This will provide a comprehensive assessment of the project's impact on the environment and help determine if there are any steps that can be taken to reduce the carbon footprint of the project.

The AI-powered pricing tool has the potential to reduce waste in the pricing department by reducing the need for paper and other materials. The waste reduction potential of the tool will be evaluated and compared to the status quo to determine the overall environmental impact of the project.

The results of the environmental feasibility analysis have shown that the impact of the AI-powered pricing tool project on the environment is acceptable. Energy consumption, carbon footprint, and waste reduction have all been evaluated and found to be within acceptable levels. As a result, the project will proceed with a focus on reducing the environmental impact where possible and ensuring that all regulations and standards are met.

5. Project resource planning.

In this section, the activities and durations that will be included in the Gantt chart for the project are listed and defined. The project will be executed by a student working as an apprentice in the company, and therefore, it is crucial to effectively plan and allocate time for each task to ensure a successful outcome. The Gantt chart will serve as a visual representation of the project timeline, providing a clear overview of the tasks to be completed, their interdependencies, and the expected duration of each task. In the following table, we will present a detailed list of activities and their estimated durations, serving as a roadmap for the successful implementation of the ML-powered segmentation tool.

Task Description	Total Days
Project designer hours dedicated to the search for background information and various previous information.	24
Project designer's hours devoted to the definition and planning of the solution.	14
Project designer's hours devoted to the development and design of the solution.	19
Project designer's hours devoted to the realisation and implementation of the solution.	7
Project designer's hours devoted to the written elaboration and editing of the project documents.	11

Tab. 5.1. Main Tasks groups and total hours to complete.

The first task of the project is to gather and analyse background information and previous data, which will take 96 hours. The next step is to define and plan the solution (56 hours) and identify patterns and trends in the data. The third task is the development and design of the solution (76 hours) using ML. The fourth task is the realization and implementation of the solution (28 hours) with testing and fine-tuning. Finally, the last task is the written elaboration and editing of project documents (44 hours) to ensure proper documentation.

In summary, this project is aimed at training and exploiting insights from the pricing data of an actual portfolio and will require a total of 300 hours of the project designer's time, spread across five key tasks. There’s an additional 16h as buffer at the end of the planification.

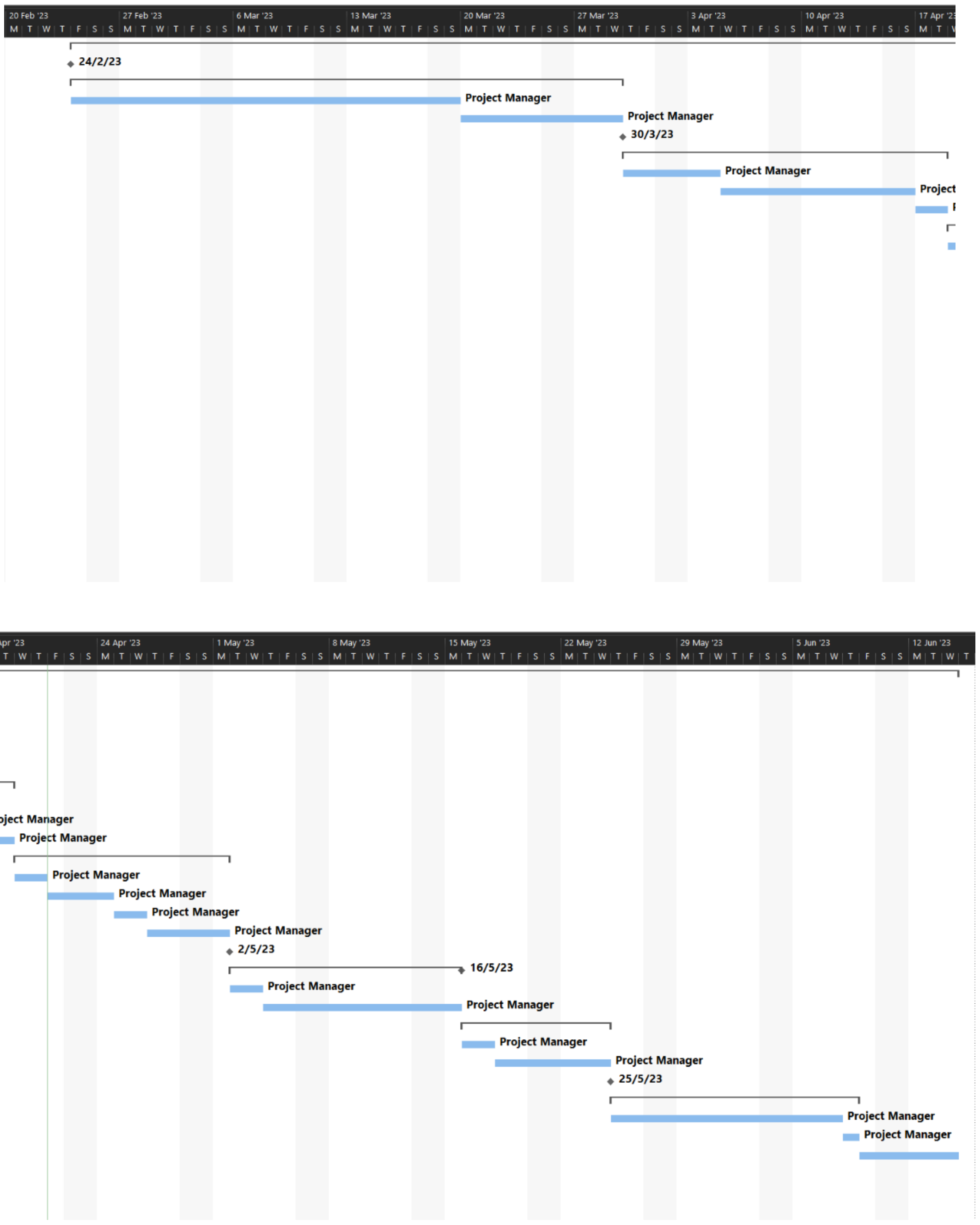


Fig. 5.1. Gantt diagram of the project.

ID	Task Name	Duration (days)	Start	Finish	Predecessors
1	**ML&AI Pricing App - Clustering**	**79**	**Fri 24/2/23**	**Thu 15/6/23**	
2	Kick Off	0	Fri 24/2/23	Fri 24/2/23	
3	**A:Research**	**24**	**Fri 24/2/23**	**Thu 30/3/23**	**2**
4	A1 Proposal review	16	Fri 24/2/23	Mon 20/3/23	2
5	A2 New Proposal Verification	8	Mon 20/3/23	Thu 30/3/23	4
6	Aproval of review	0	Thu 30/3/23	Thu 30/3/23	5
7	**B:Definition**	**14**	**Thu 30/3/23**	**Wed 19/4/23**	**6**
8	B1 Model Selection	4	Thu 30/3/23	Wed 5/4/23	6
9	B2 Data Preparation	8	Wed 5/4/23	Mon 17/4/23	8
10	B3 Dimensinality reduction	2	Mon 17/4/23	Wed 19/4/23	9
11	**C:Model Application**	**9**	**Wed 19/4/23**	**Tue 2/5/23**	**10**
12	C1 K-means	2	Wed 19/4/23	Fri 21/4/23	10
13	C2 DBSCAN	2	Fri 21/4/23	Tue 25/4/23	12
14	C3 GMM	2	Tue 25/4/23	Thu 27/4/23	13
15	C4 SpectralNet	3	Thu 27/4/23	Tue 2/5/23	14
16	Meeting Management	0	Tue 2/5/23	Tue 2/5/23	15
17	**D:Cluster Evaluation**	**10**	**Tue 2/5/23**	**Tue 16/5/23**	**16**
18	D1 Cluster Evaluation	2	Tue 2/5/23	Thu 4/5/23	15
19	D2 Iterative Model Tweaking	8	Thu 4/5/23	Tue 16/5/23	18
20	**E:Implementation**	**7**	**Tue 16/5/23**	**Thu 25/5/23**	**19**
21	E1 Result export	2	Tue 16/5/23	Thu 18/5/23	19
22	E2 Results Presentation	5	Thu 18/5/23	Thu 25/5/23	21
23	Results Meeting	0	Thu 25/5/23	Thu 25/5/23	22
24	**F:Final Documentation**	**11**	**Thu 25/5/23**	**Fri 9/6/23**	**23**
25	F1 Final Documentation	10	Thu 25/5/23	Thu 8/6/23	23
26	F2 Final Review	1	Thu 8/6/23	Fri 9/6/23	25
27	**Buffer**	**4**	**Fri 9/6/23**	**Thu 15/6/23**	**26**

Tab. 5.2. Extended task table.

5.1. Final evaluation of the planning.

As the project is now finished the planification has been completed with no major deviations. The buffer at the end has been utilised due to the increase of time consumed in the iterative tweaking part of the project but it has not translated to a delay of the closure of the project documentation presented in this document.

6. Budget.

This section of the project documentation outlines the estimated cost of implementing an AI-powered pricing tool in a pricing department. The budget section provides a comprehensive overview of the financial resources required to successfully complete the project. The budget is broken down into various components, including hardware and software costs, personnel expenses, and operational costs. This section also highlights any potential cost-saving measures that can be implemented to reduce the overall cost of the project. The budget section is a crucial component of the project documentation as it provides an understanding of the financial resources required to complete the project and allows for better planning and decision-making.

Budget		
Chapter	**Description**	**€**
1	Project development	2.625,00
2	Materials	316,25
3	Amortizations	300,00
	Total	3.241,25
	IVA (21 %)	680,66
	FINAL Total	3.921,91

Tab. 6.1. Budget summary of the project.

7. Contingency plan.

A contingency plan is an essential component of any project, including this one about a clustering project. It provides a framework for dealing with potential challenges and setbacks, helping to ensure the project's success.

One important aspect of a contingency plan for a clustering analysis project is to have a backup plan in case the initial clustering algorithm or the chosen number of clusters does not produce satisfactory results. This could involve trying different algorithms or adjusting the number of clusters until the desired outcome is achieved. For example, as already explained the initial algorithm used is k-means, alternative algorithms such as GMM clustering or DBSCAN are also considered. Similarly, if the initial number of clusters does not produce meaningful results, the number could be increased or decreased to see if this improves the outcome.

Another important element of a contingency plan is to have a robust data validation and cleaning process in place. This can help ensure that the data used in the analysis is accurate and free of errors, which can significantly impact the results. The data validation process could involve checking for outliers, inconsistencies, and duplicate entries, while the data cleaning process could involve removing or correcting any errors identified during the validation process.

It's also important to have a plan for dealing with missing or incomplete data. This could involve imputing missing values using techniques such as mean imputation or regression imputation, or excluding certain data points from the analysis if they are deemed to be unreliable.

Finally, involving domain experts and stakeholders in the project can help ensure that the results are meaningful and actionable from a business perspective. Having regular check-ins and feedback sessions can help identify any issues early on and allow for timely adjustments to the project plan. For example, if the initial results are not considered useful by the domain experts, their feedback could be used to adjust the analysis parameters or try a different approach.

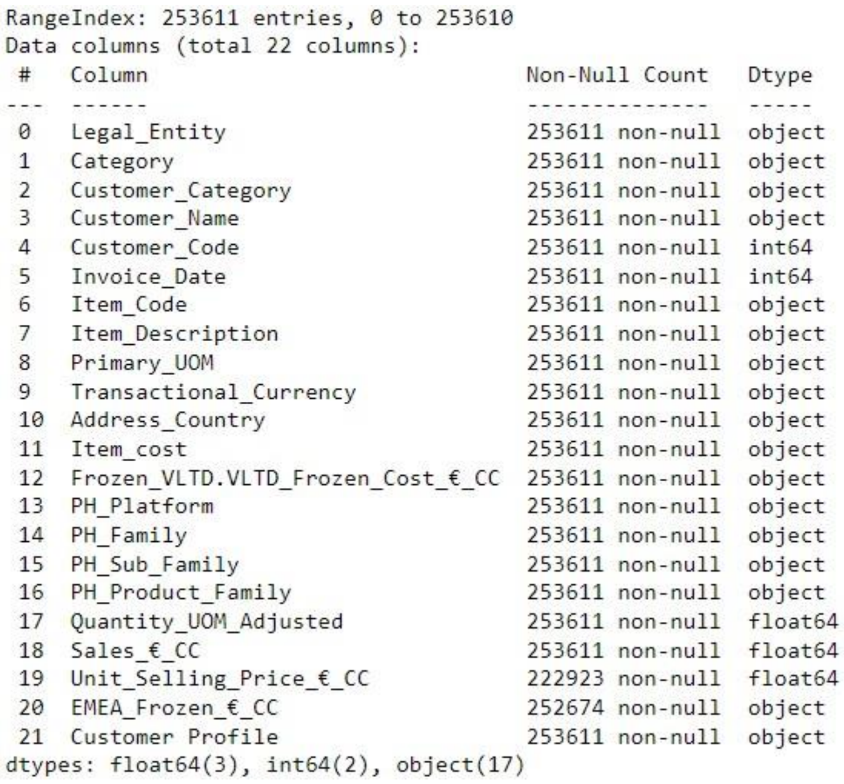
8. Implementation.

8.1. Data collection and refinement.

The data collected for this project has been supplied by the – from the financial department at the company. This colleague is responsible of preparing the main financial reports to keep track of the operation of the company in Europe.

The data provided contains a registry of all financial operations with the company's customers during 2021 and 2022. This time frame has been chosen to deal with any possible temporality in purchase behaviour by the customers, which range from a various of industries that can present this temporality in consumer demand.

The raw data is no fit for the models we want to train, therefore we must summarise, transform and refine it into something the models can process. This file contains more than 250.000 lines of transactions and contains a range of variables per line as described in the following table. This data serves both as a direct source for our data set with minor calculation whilst also adding other variables that will be defined with more complex combinations.



```
RangeIndex: 253611 entries, 0 to 253610
Data columns (total 22 columns):
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   Legal_Entity                      253611 non-null  object
 1   Category                          253611 non-null  object
 2   Customer_Category                 253611 non-null  object
 3   Customer_Name                     253611 non-null  object
 4   Customer_Code                     253611 non-null  int64
 5   Invoice_Date                      253611 non-null  int64
 6   Item_Code                         253611 non-null  object
 7   Item_Description                  253611 non-null  object
 8   Primary_UOM                       253611 non-null  object
 9   Transactional_Currency            253611 non-null  object
 10  Address_Country                   253611 non-null  object
 11  Item_cost                         253611 non-null  object
 12  Frozen_VLTD.VLTD_Frozen_Cost_€_CC  253611 non-null  object
 13  PH_Platform                       253611 non-null  object
 14  PH_Family                         253611 non-null  object
 15  PH_Sub_Family                     253611 non-null  object
 16  PH_Product_Family                 253611 non-null  object
 17  Quantity_UOM_Adjusted             253611 non-null  float64
 18  Sales_€_CC                        253611 non-null  float64
 19  Unit_Selling_Price_€_CC           222923 non-null  float64
 20  EMEA_Frozen_€_CC                  252674 non-null  object
 21  Customer Profile                  253611 non-null  object
dtypes: float64(3), int64(2), object(17)
```

Fig. 8.1. File contents summary in the original data

This file rendered a new file after careful refinement and evaluation that could be fed to the clustering models and made sense in the decision-making process of segmenting the customers in a coherent manner. The features or columns in the refined file can be categorized as direct and calculated variables depending on the amount of processing required to achieve them based on the original data.

8.1.1. Direct variables.

As explained previously, some of the variables are retrieved in somewhat of a direct manner from the export provided. The ones that fit under this umbrella are the category, customer category, customer code, item code, address country, sales adjusted to single currency and customer profile.

8.1.2. Calculated variables.

The calculated variables incorporated are dependent on the fields invoice date, item code, EMEA frozen cost, unit selling price, product hierarchy platform and product family. From these variables the new set of relevant data has been created:

1. Percentage of sales per product platform.
2. Percentage of sales per month.
3. Percentage of sales per product family.
4. Margin per product family.

All this set of variables generate several columns directly related to the distinct number of values they contain. The only limitation applied has been working with the top 80% of sales values in product family, to minimize the compute power required whilst maximising the results.

8.1.3. Standardization.

In terms of data standardization there has been two defined steps. Firstly, running a one hot encoding process to those categorical values that need to be transformed into numerical sets to be compatible with the segmentation models. Secondly, using an encoder all data has been standardised, so all features have equal weight when applying the segmentation models.

8.1.4. Weights of the variables.

Since all features have the same level of importance when fed to the models, the criteria to group the data has the high chance of rendering distant results to what is expected or logical by the knowledgeable peers at the company. One way to combat this is scaling the features of most relevance to the company so the results are closer to real situations. At the end of the day not all features have the same impact for example on the value of the customer or the special needs that might have.

The first run however is done without altering the weights, and once the results are in a cyclic process of tunning is started were progressively the tweaks on the weights lead to a more accurate result.

8.2. Dimensionality reduction.

Once the data file is prepared, a process of dimensionality reduction is undertaken to improve runtimes of the clustering models, reduce the compute strain and finally to have the ability to represent the results in a visual manner.

As mentioned in the project structure part of this document, there are numerous methods to apply dimensionality reduction and the fitment of it depends highly on the data distribution we find. Therefore, all the mentioned have been tested and this have been the results.

8.2.1 PCA.

Definition

Principal Component Analysis is a statistical technique for reducing the dimensionality of a dataset. This is accomplished by linearly transforming the data into a new coordinate system where (most of) the variation in the data can be described with fewer dimensions than the initial data. Formally, PCA finds the eigenvectors of a covariance matrix with the highest eigenvalues and projects the data onto these eigenvectors to obtain lower-dimensional data while preserving as much of the data’s variation as possible.

The mathematical formula for PCA involves calculating the covariance matrix of the data, finding its eigenvectors and eigenvalues, and projecting the data onto the eigenvectors corresponding to the largest eigenvalues.

Let X be a d-dimensional dataset with n samples. The covariance matrix C is calculated as:

$$C = \frac{1}{n-1} X^T X$$

The eigenvectors v and eigenvalues λ of C are calculated by solving the equation:

$$Cv = \lambda v$$

The eigenvectors corresponding to the largest eigenvalues are then used to project the data onto a lower-dimensional space.

Application

PCA has not been the best when analysing the output results, as seen in Fig. 8.2 and Fig. 8.3, when plotting both in two and three dimensions there are no visible group separation but rather few distant observations and central high-density group with most of the customers. This renders unsuitable for the objectives of the project and the working process of the clustering models.

The following figures depict in a visual manner the outputs of this method in 2D and 3D. This shows no possibility of applying the clustering methods without a deep modification of the fed data, altering the intended scope on the decision.

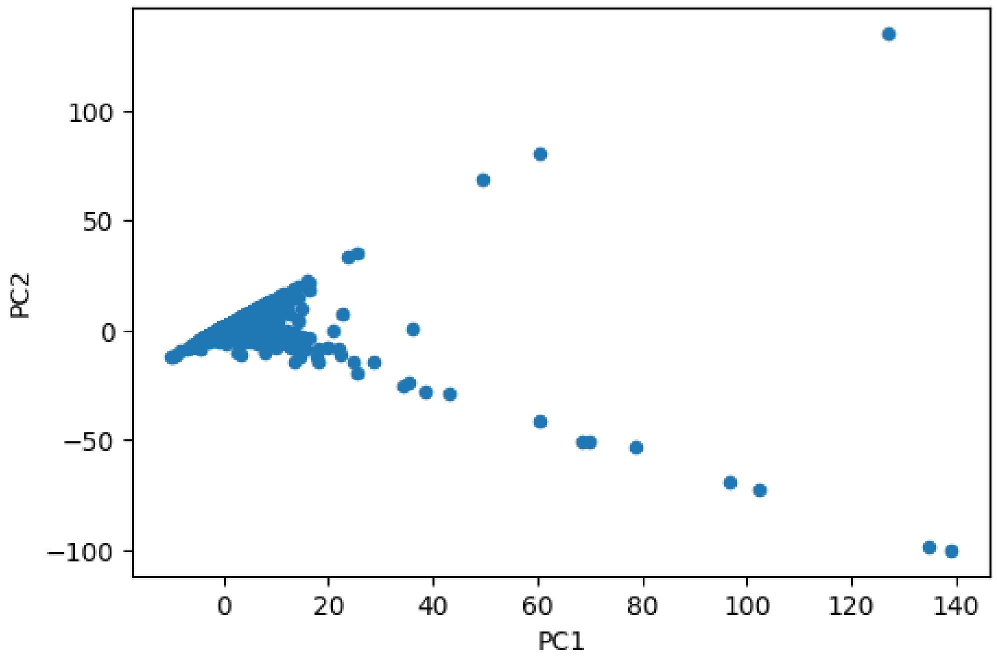


Fig. 8.2. 2D representation of the PCA output.

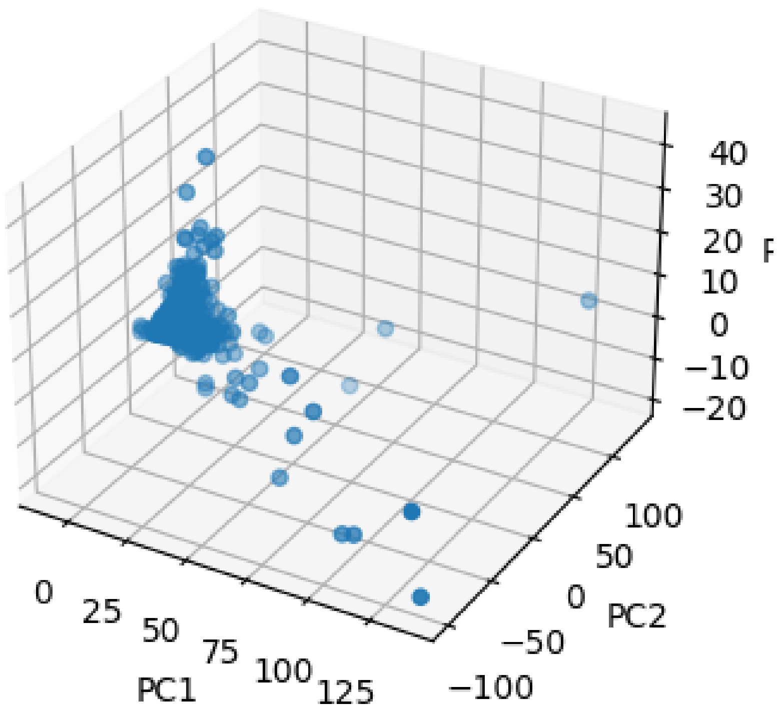


Fig. 8.3. 3D representation of the PCA output.

8.2.2. T-SNE.

Definition

T-SNE is a statistical method that computes probabilities p_{ij} proportional to the similarity of objects $\mathbf{x}_i$ and $\mathbf{x}_j$. For i≠j, p_{j|i} is defined as: $\mathbf{x}_j$

$$p_{j|i} = \frac{\exp\left(-\parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\parallel \mathbf{x}_i - \mathbf{x}_k \parallel^2 / 2\sigma_i^2\right)}$$

and $p_{i|i} = 0$. The similarity of datapoint $\mathbf{x}_j$ to datapoint $\mathbf{x}_i$ is the conditional probability, $p_{j|i}$, that $\mathbf{x}_i$ would pick $\mathbf{x}_j$ as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centred at $\mathbf{x}_i$. Now define:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

The bandwidth of the Gaussian kernels σ_i is set in such a way that the entropy of the conditional distribution equals a predefined entropy using the bisection method. As a result, the bandwidth is adapted to the density of the data: smaller values of σ_i are used in denser parts of the data space. Since the Gaussian kernel uses the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|$, it is affected by the curse of dimensionality, and in high dimensional data when distances lose the ability to discriminate, the p_{ij} become too similar.

t-SNE aims to learn a d-dimensional map y_1,…, y_N (with $y_i \in R^d$ and d typically chosen as 2 or 3) that reflects the similarities p_{ij} as well as possible. To this end, it measures similarities q_{ij} between two points in the map y_i and y_j, using a very similar approach. Specifically, for i≠j, define q_{ij} as:

$$\mathrm{KL}(P \mid\mid Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

and set q_{ij}=0. Herein a heavy-tailed Student t-distribution (with one-degree of freedom, which is the same as a Cauchy distribution) is used to measure similarities between low-dimensional points to allow dissimilar objects to be modelled far apart in the map.

Application

T-SNE has shown far more success preserving the distances between the observations once reviewing the plots of the results. When watching Fig. 8.4 and Fig. 8.5, both 2D and 3D are highly usable and presents as one of the dimensionality reduction methods to be used in the project. The resulting figure show the clear tendencies within the customer data that will surely perform with the clustering methods.

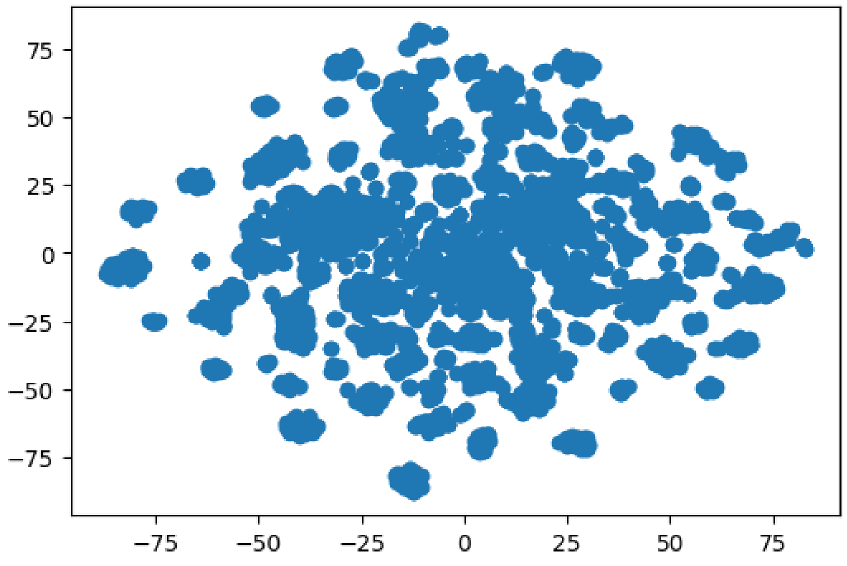


Fig. 8.4. 2D representation of the T-SNE output.

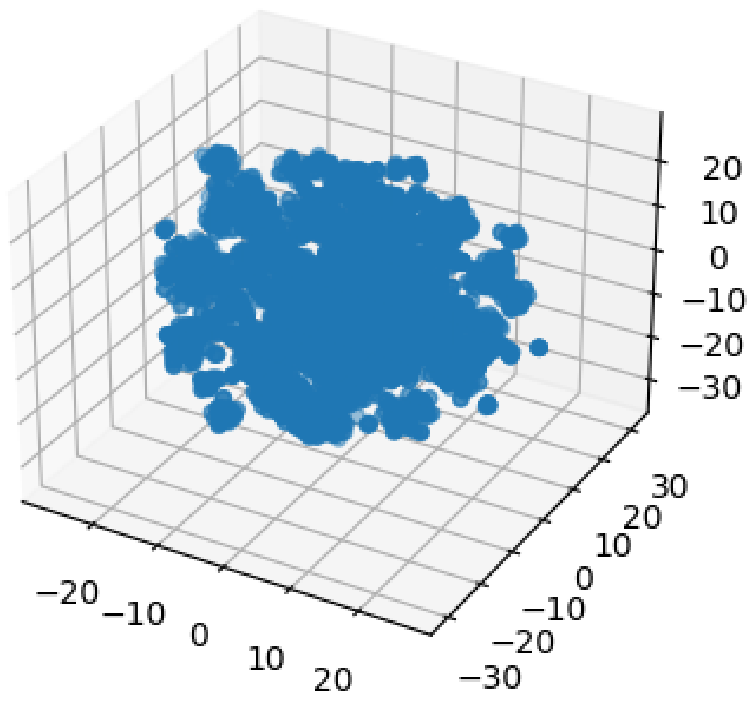


Fig. 8.5. 3D representation of the T-SNE output.

8.2.3. UMAP.

Definition

Uniform Manifold Approximation and Projection is a dimension reduction technique that can be used for visualization similarly to t-SNE, but also for general non-linear dimension reduction. The algorithm is founded on three assumptions about the data: The data is uniformly distributed on a Riemannian manifold; the Riemannian metric is locally constant or approximately locally constant; and the data has a locally connected sparse topological structure.

UMAP starts by constructing a weighted k-nearest neighbour graph on the high-dimensional data. The graph is then embedded in low-dimensional space via optimization of a cost function that balances preserving the global structure of the data against preserving local structure.

Application

UMAP has worked up to a certain point, when looking at Fig. 8.6 and Fig. 8.7 it can be said it is far better than PCA but not as clear definition as T-SNE. This means that its use might be considered but does not stand as the main method to be applied in the project. Therefore, if the first chosen method is successful it is discarded.

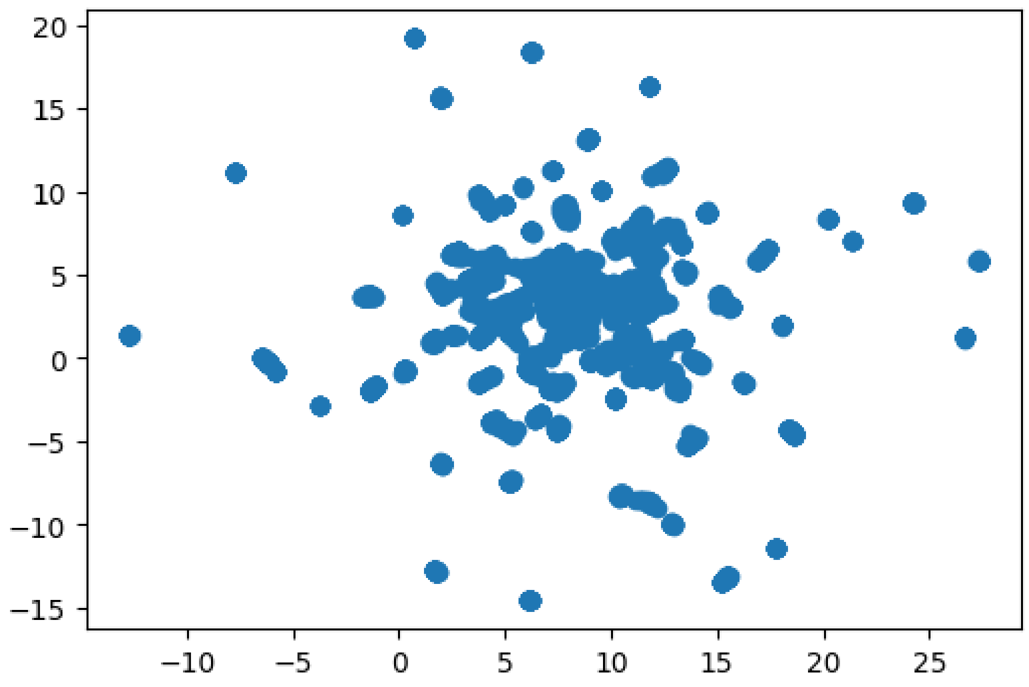


Fig. 8.6. 2D representation of the UMAP output.

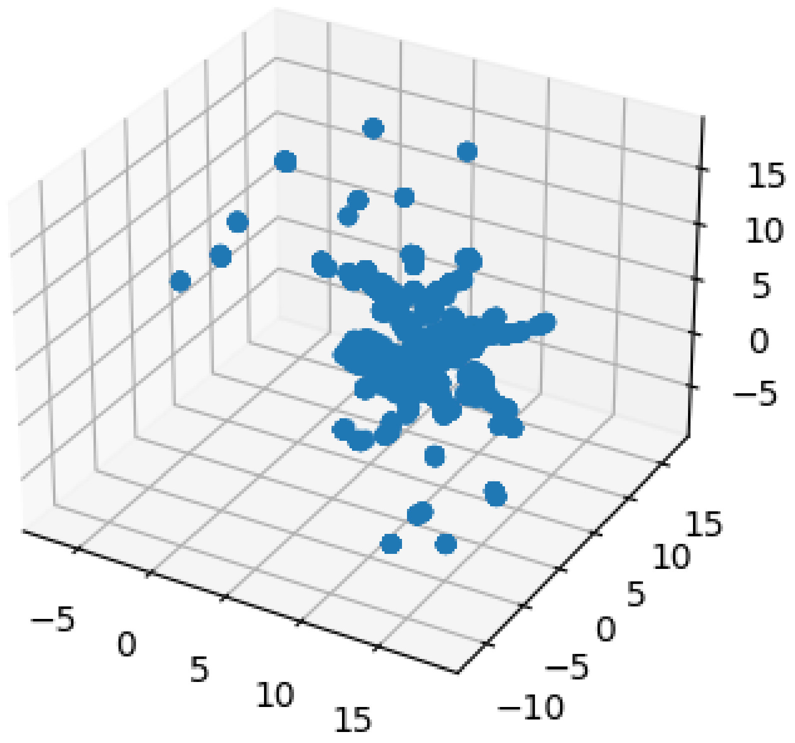


Fig. 8.7. 3D representation of the UMAP output.

8.2.4. Dimensionality reduction method conclusion.

When analysing the performance of the different methods listed above for dimensionality reduction it has been found that T-SNE is the most suitable to be applied the segmentation models. Additionally, the following models will be fed with the two-dimensional data by design and only explore higher dimensionality if required by the results. T-SNE has had the best separation whilst maintaining a coherent group structure, leading to some kind of tendencies to be exploited by the segmentation models.

8.3. Model selection.

In the planning phase of this project a series of clustering methods were explored to obtain the best clustering results possible. There is not an absolute best in all scenarios method but rather some fit better with the data distribution or simply are easier to run on limited compute processing power. That is why all methods have been tested in the following order of appearances. The performance of the methods will be dictated by the ability of the model to keep relevant customers from defined business segments by de company, the type of relation they have with the end customer and the type of products their portfolios have in common.

8.3.1. K-means.

Definition

K-means is a widely utilized clustering algorithm that partitions a set of n observations into k clusters, where each observation belongs to the cluster with the nearest mean. The objective function of K-means is to minimize the within-cluster sum of squares (WCSS), which is defined as the sum of squared Euclidean distances between each observation and the centroid of its assigned cluster. Mathematically, this can be expressed as:

$$\sum_{i=1}^{k} \sum_{x \in S_i} || \, x - \mu_i \, ||^2$$

where S_i represents the set of observations assigned to cluster i, and μ_i represents the centroid of cluster i. The algorithm proceeds by iteratively assigning each observation to the cluster with the closest centroid and then recomputing the centroid of each cluster as the mean of its assigned observations. This process is repeated until convergence, i.e., until the assignments no longer change.

In addition to its use in clustering, K-means can also be employed for dimensionality reduction by mapping high-dimensional data to a lower-dimensional space while preserving the structure of the data. This is achieved by computing the principal components of the data and projecting it onto a lower-dimensional subspace. The principal components are computed by performing singular value decomposition (SVD) on the data matrix and retaining only the top k singular vectors, where k is the desired dimensionality of the reduced data. The reduced data is then obtained by projecting the original data onto the subspace spanned by these singular vectors.

The SVD of a matrix $X \in R^{n \times d}$ can be written as:

$$X = U \Sigma V^T$$

where $U \in Rn \times n$ and $V \in Rd \times d$ are orthogonal matrices containing the left and right singular vectors of X, respectively, and $\Sigma \in \mathbb{R}^{n \times d}$ is a diagonal matrix containing the singular values of X. The top k singular vectors corresponding to the largest k singular values can be used to define a projection matrix $P = U_k$, where $U_k$contains only the first k columns of U. The reduced data can then be obtained by computing $X_k = XP$.

Application

This method is the easiest to run in a portable laptop, as is the case, and takes limited amount of time to achieve the results. Given the reduced dimensionality best performance is obtained with 18 groups. With the data fed the following plot shows the following distribution of data.

The separation is as expected given the how the K-means method creates the clusters, all of them being round in form and cutting what could be seen as possible data groups in different segments due to this same reason. But the coherence on the groups is what matters, this first segmentation showed no relevant customer segments, given the data that the model gave more importance to was not aligned with the vision of the company.

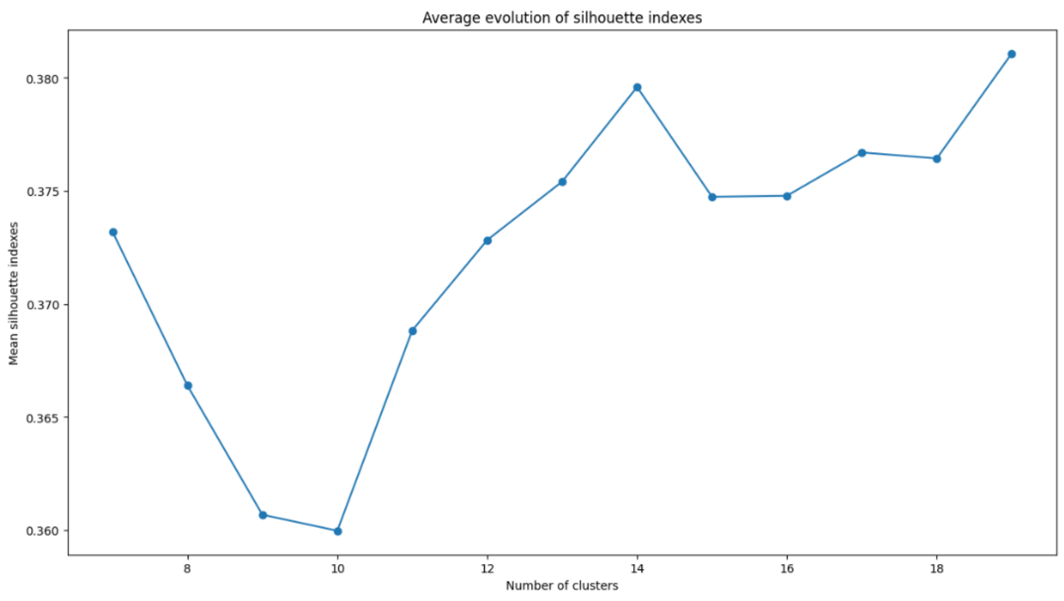


Fig. 8.8. Silhouette method indicates best performance k = 14.

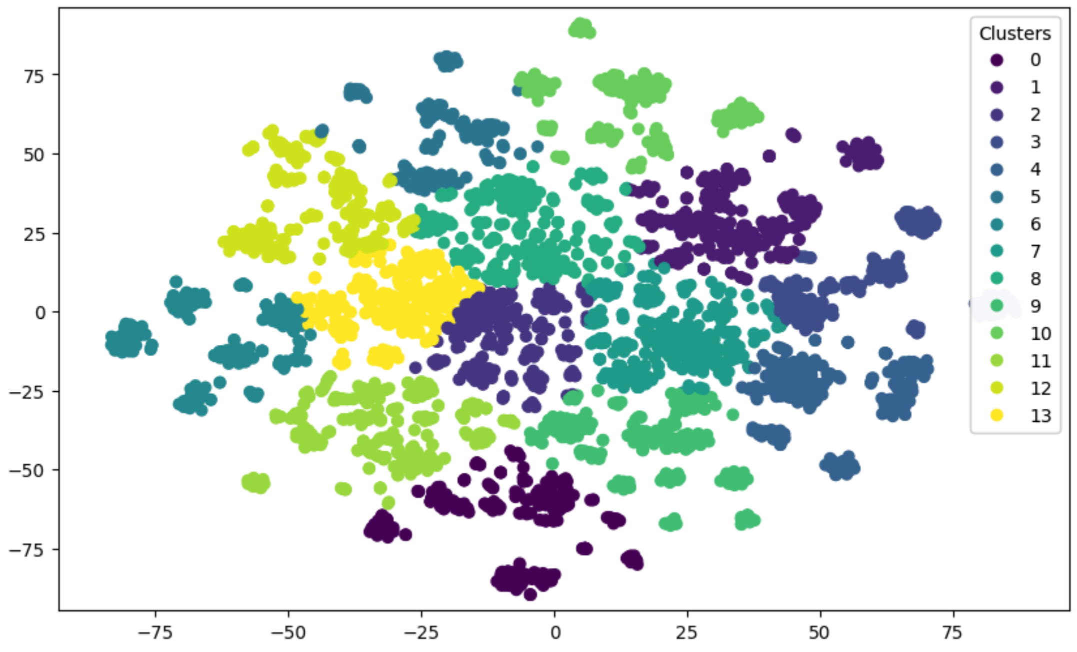


Fig. 8.9. 2D representation of the K-means output.

Therefore, a new iteration of the process was generated with adjusted weight to the features that should play a major role on the decision. The features affected were the sales, the margin, the number of distinct items sold, the customer category and the customer profile in decreasing order of weight added. These adjustments gave the data the following form.

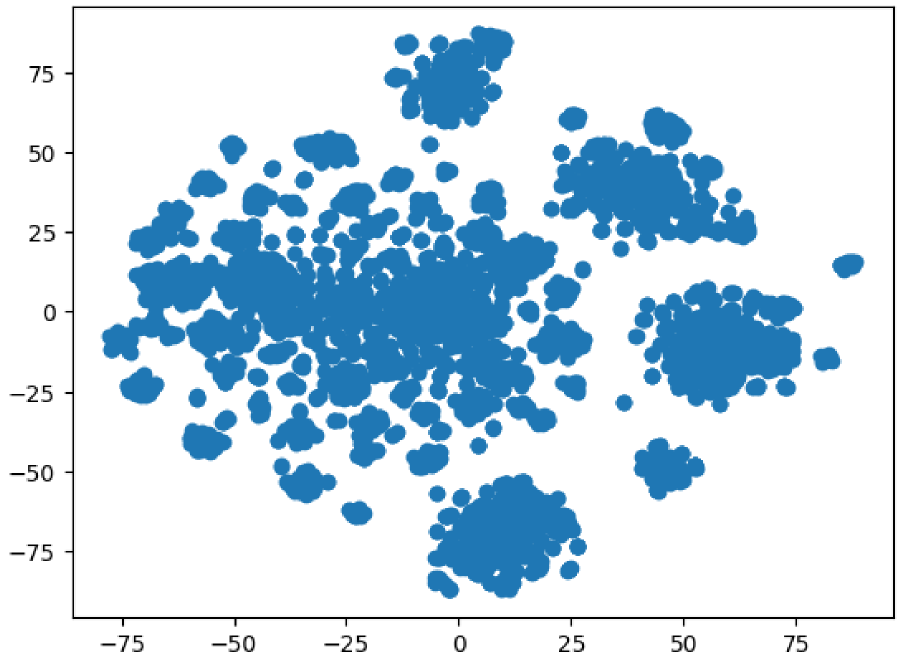


Fig. 8.10. 2D representation of the weighted T-SNE output.

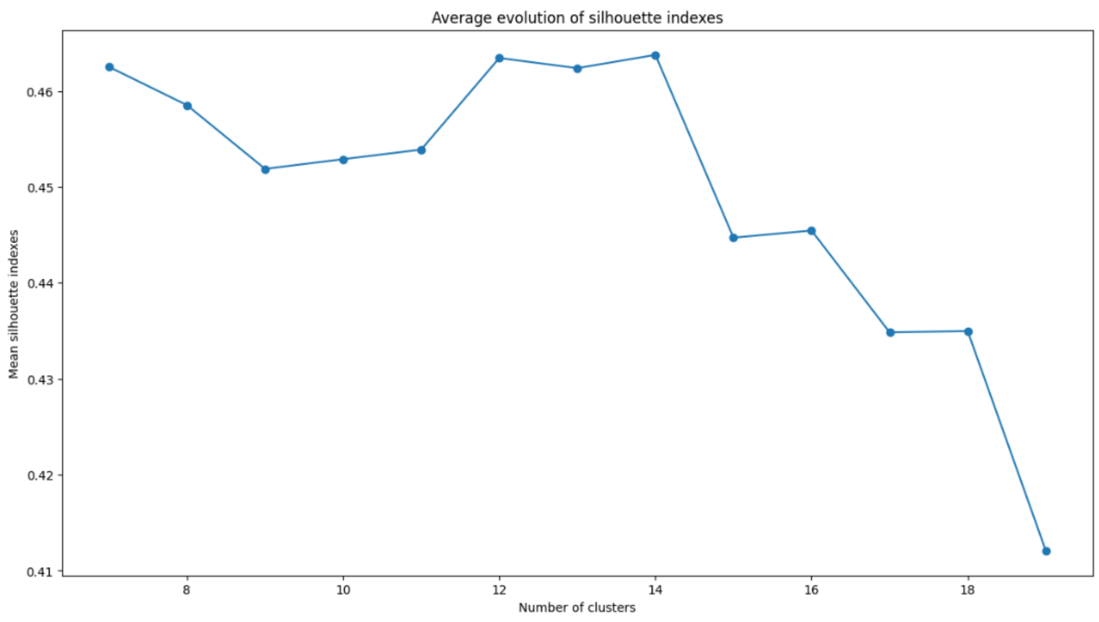


Fig. 8.11. Silhouette method adjusted indicates best performance k = 14.

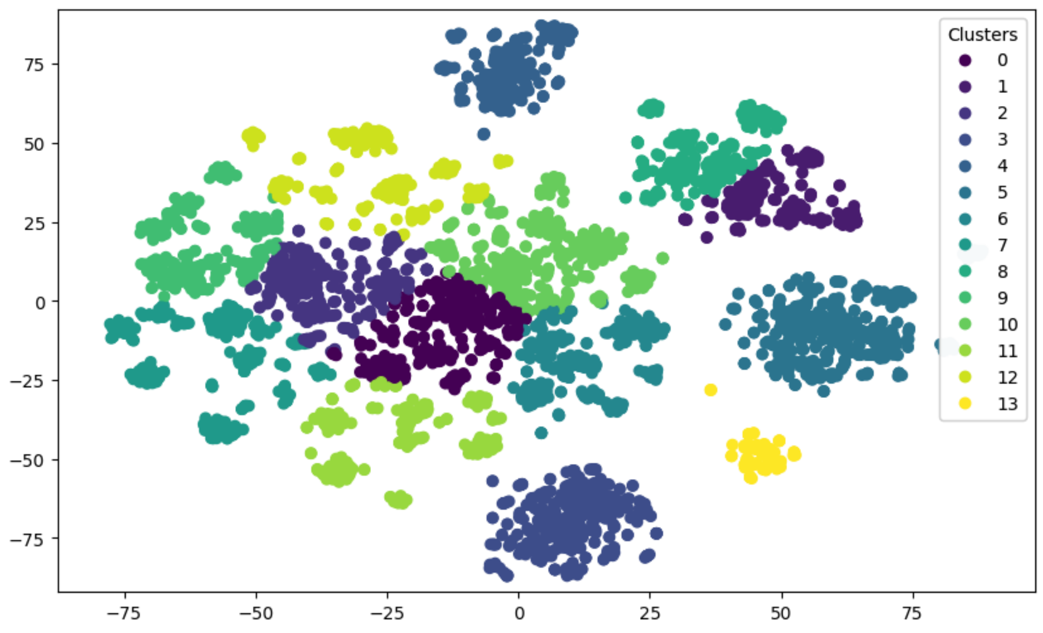


Fig. 8.12. 2D representation of the adjusted K-means output.

Given this adjusted data set shown in Fig. 8.12, the results are much closer to the company's objectives of the tool. The clusters are now far more realistic in terms of coherence, having the observations more in common on the important features. This was not only verified by reviewing a set of customers in the context of the cluster were allocated, as in the previous run, but since this first threshold was surpassed the following bar plots were crafted as verification of the results.

The scale of the results is not related to the real values they represent, they just work as visual representation of the comparative differences between clusters.

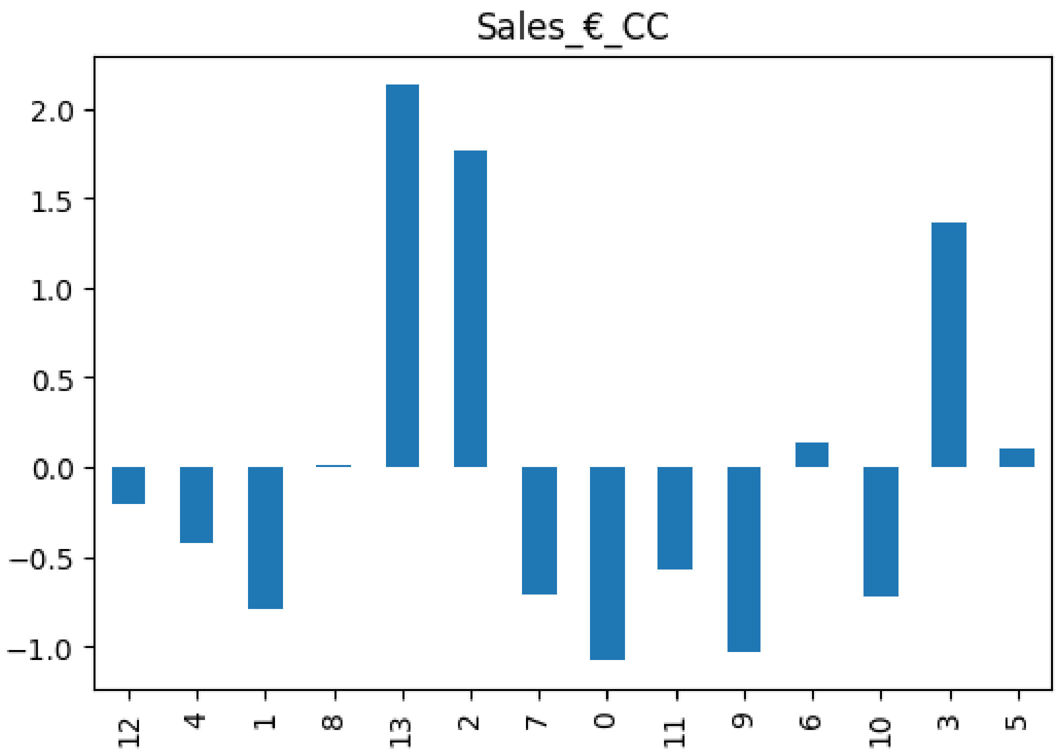


Fig. 8.13. Cluster comparison of the normalized sales averages.

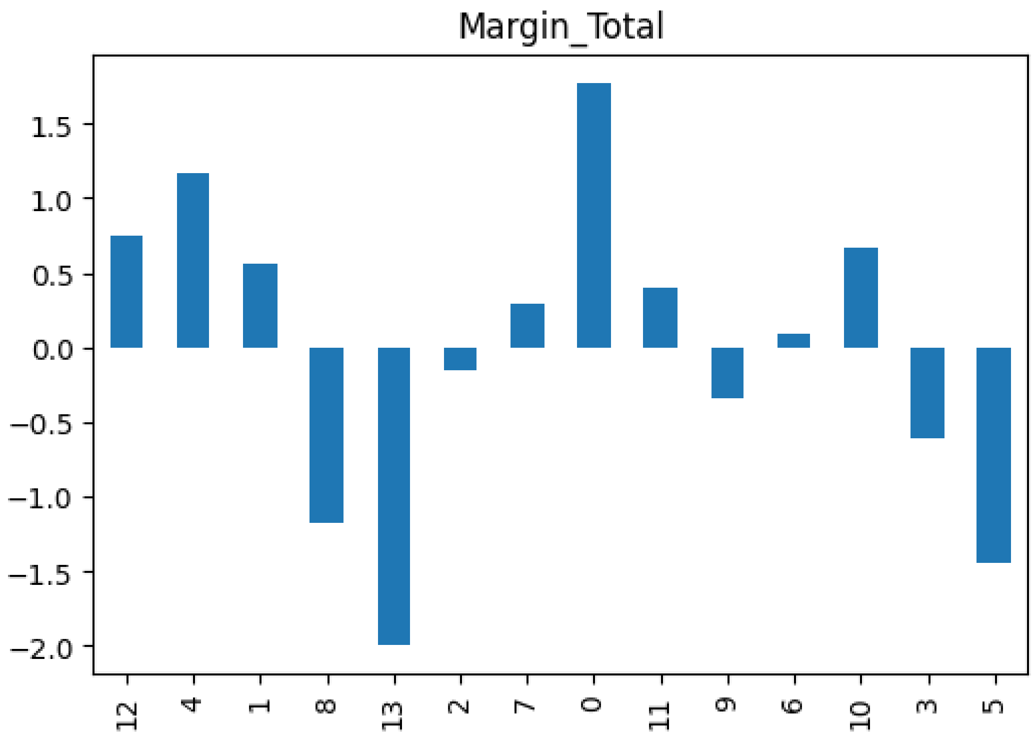


Fig. 8.14. Cluster comparison of the normalized margin averages.

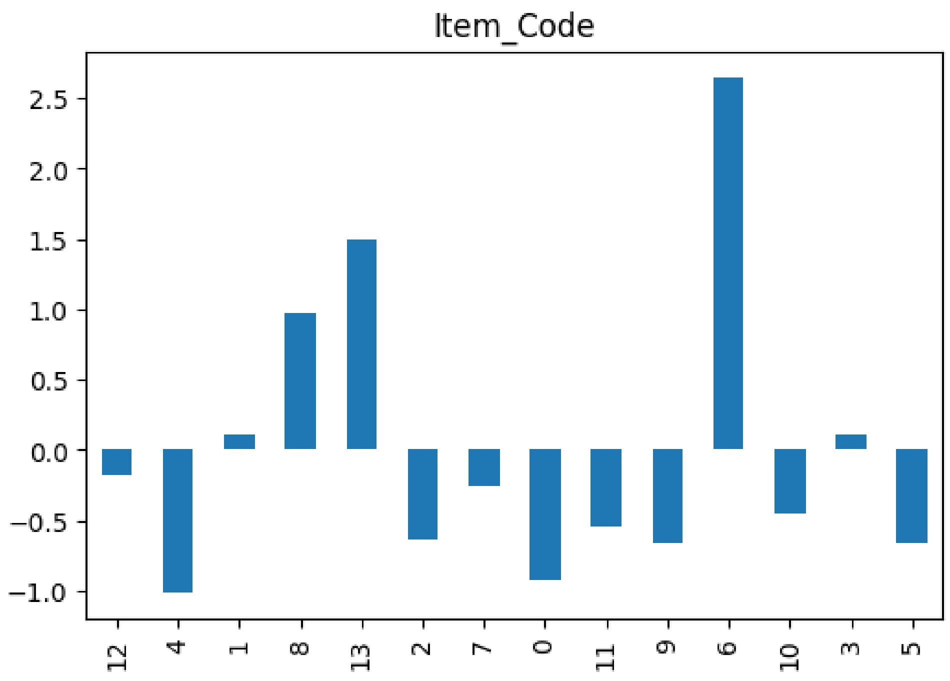


Fig. 8.15. Cluster comparison of the normalized sold item number averages.

And once we see how the current segmentation stacks with the k-means results the suitability of them is even clearer. The category that shows more variability displayed in Fig. 8.16, in purple, is the largest and broader which means there will be room for analysis of the customers in it to be move to another category or a brand new one.

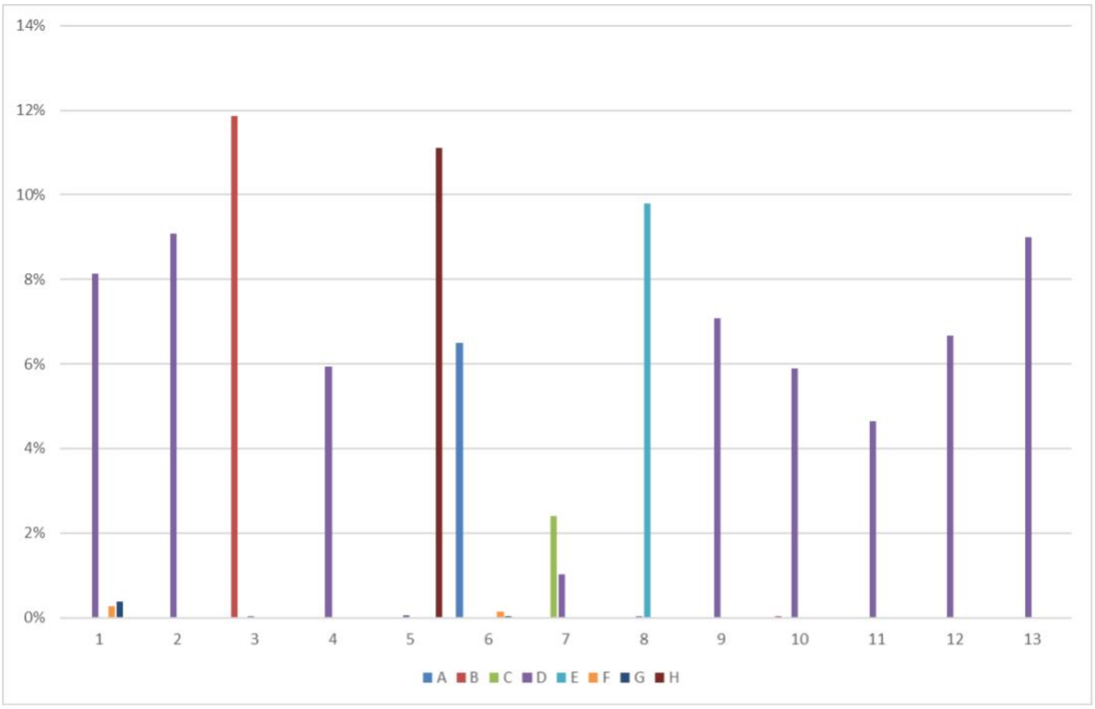


Fig. 8.16. Customer distribution per cluster based on current segmentation.

8.3.2. DBSCAN.

Definition

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that can identify clusters of arbitrary shape in a dataset. Given a set of n observations, DBSCAN groups together observations that are closely packed together, marking as outlier observations that lie alone in low-density regions.

The algorithm requires two parameters: ϵ, which specifies the radius of the neighbourhood around a data point, and MinPts, which specifies the minimum number of points required to form a dense region. A point p is considered a core point if at least MinPts points are within distance ϵ from it. A point q is directly reachable from p if it is within distance ϵ from p and p is a core point. A point q is density-reachable from p if there is a chain of points $p1, \ldots, pn,$ $where\ p1 = p\ and\ pn = q$, such that each point p_{i+1} is directly reachable from pi. A point q is density-connected to a point p if there exists a point o such that both p and q are density-reachable from o. A cluster is defined as a non-empty subset of the dataset such that any two points in the cluster are density-connected, and any point not in the cluster is not density-reachable from any point in the cluster.

The DBSCAN algorithm proceeds by arbitrarily selecting an unvisited point and retrieving its ϵ-neighbourhood. If this neighbourhood contains at least MinPts points, a new cluster is started. The algorithm then iteratively expands the cluster by adding all density-reachable points to it. Once no more points can be added to the cluster, the algorithm selects another unvisited point and repeats the process until all points have been visited.

Application

Given the weights applied in the K-means section prior and the resulting T-SNE dimensionally reduced data, the DBSCAN model was tested following the steps required to determine the optimal parameters for the data set. These parameters are the minimum observations per cluster and the epsilon, the density threshold determined by the knee point method.

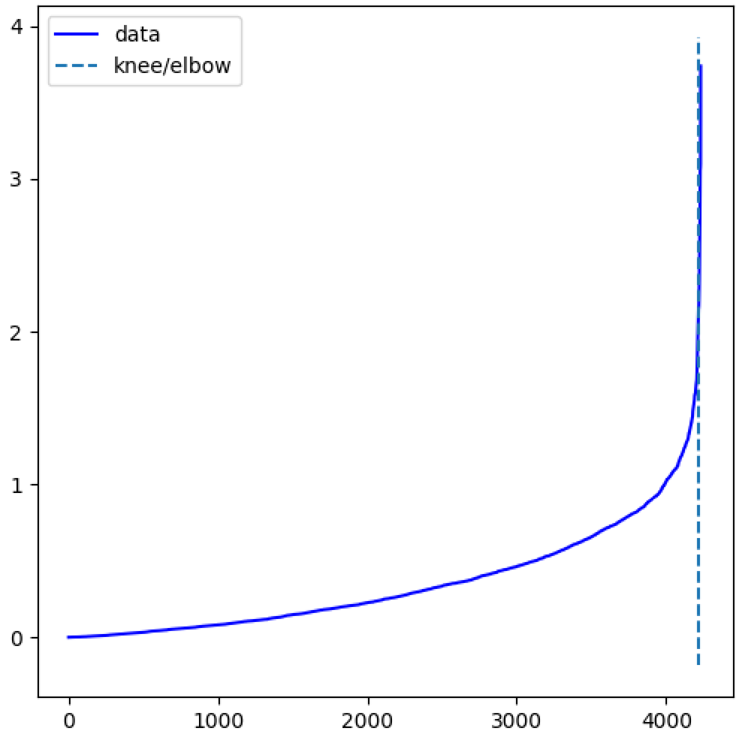


Fig. 8.17. Knee point sets the optimal epsilon = 1,78

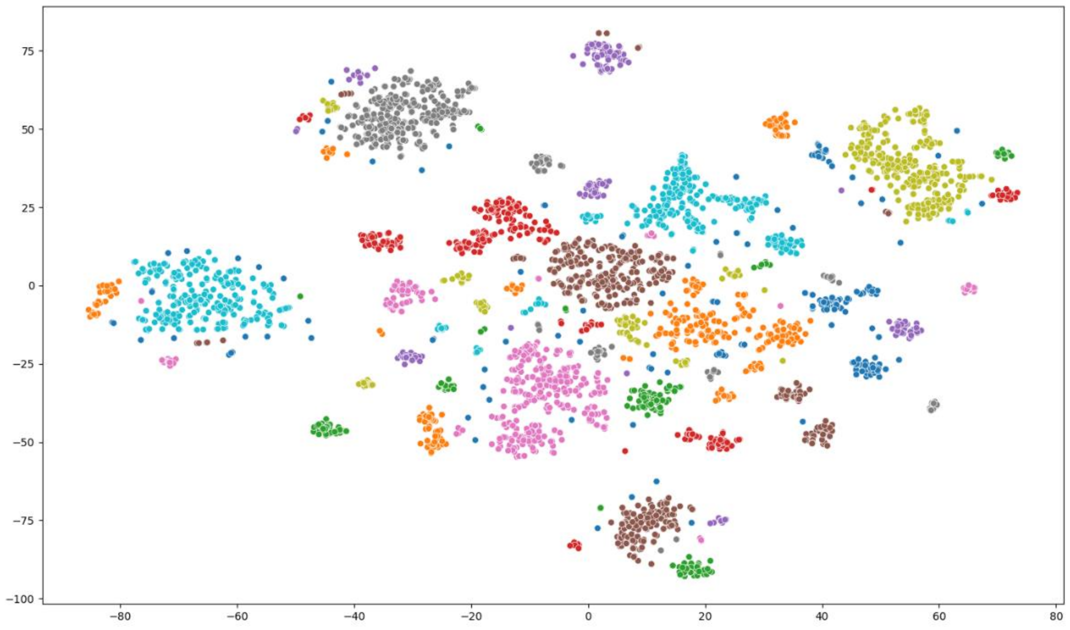


Fig. 8.18. DBSCAN output defines 101 clusters.

As seen in figure 8.18 the results of this model are not what expected. The goal of the project is to set several clusters but always less than 20. The data is too fragmented for the model to consider as a valid output. As a final test the model was tested with an increased value of epsilon than the optimal at 5,5.

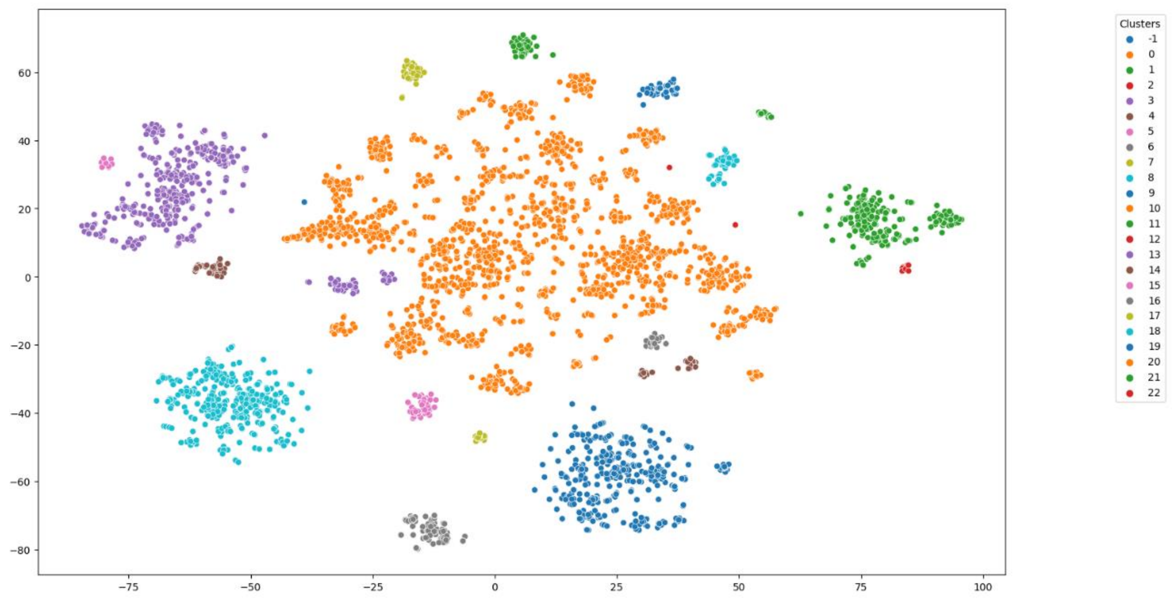


Fig. 8.19. DBSCAN output with epsilon=5,5

8.3.3. Gaussian Mixture.

Definition

The Gaussian Mixture Model is a probabilistic model that assumes that the data is generated by a mixture of several Gaussian distributions. Given a set of n observations, the goal of GMM is to estimate the parameters of the mixture model, including the means, covariances, and mixing proportions of the component Gaussian distributions.

The Expectation-Maximization algorithm is commonly used to estimate the parameters of a GMM. The EM algorithm iteratively estimates the posterior probabilities of each observation belonging to each component (the E-step) and then updates the parameters of the mixture model to maximize the likelihood of the data given the current posterior probabilities (the M-step).

In the E-step, the posterior probability that observation x_i belongs to component j is computed as:

$$\gamma_{ij} = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)}$$

where π_j is the mixing proportion of component $j, N(xi \mid \mu j, \Sigma j)$ is the probability density function of a multivariate Gaussian distribution with mean μ_j and covariance matrix Σ_j, and K is the number of components in the mixture model.

In the M-step, the parameters of the mixture model are updated as follows:

$$\mu_j^{new} = \frac{\sum_{i=1}^{n} \gamma_{ij} x_i}{\sum_{i=1}^{n} \gamma_{ij}}$$

$$\Sigma_j^{new} = \frac{\sum_{i=1}^{n} \gamma_{ij}(x_i - \mu_j^{new})(x_i - \mu_j^{new})^T}{\sum_{i=1}^{n} \gamma_{ij}}$$

$$\pi_j^{new} = \frac{\sum_{i=1}^{n} \gamma_{ij}}{n}$$

The EM algorithm iterates between the E-step and M-step until convergence.

GMMs have several advantages over other clustering algorithms such as K-means. For instance, they can model clusters with different shapes and sizes and can provide a measure of uncertainty for each cluster assignment.

Application

GMM models are an advanced clustering machine learning model based on K-means, and as such it is expected to provide similar results to that model with slight variation due to its properties. The main goal of this test is to provide a comparison point to the first model validated results and to observe any possible improvements when using this method.

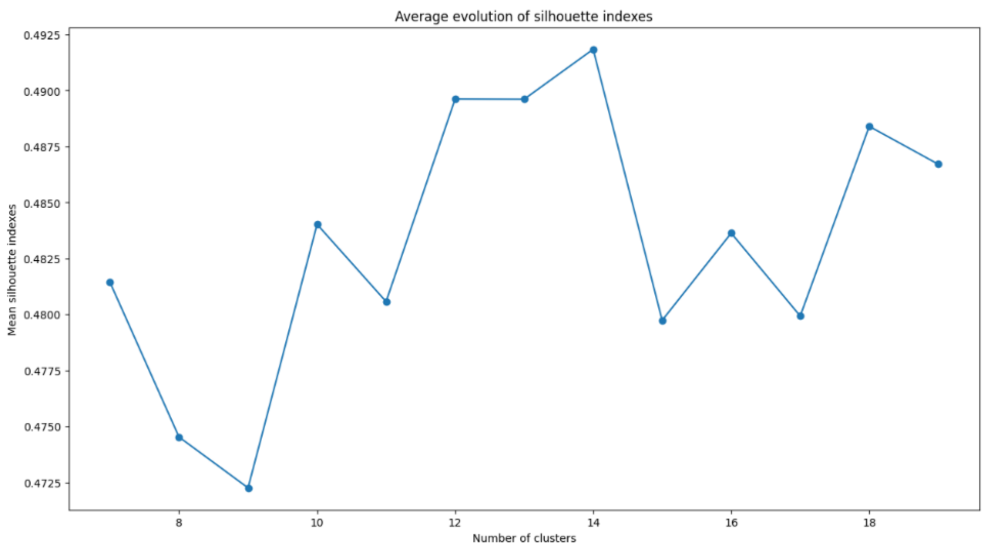


Fig. 8.20. Silhouette method indicates best performance k = 14.

The data fed to the model has the same weight adjustment to the features as the optimal run in K-means and the only test in DBSCAN, and based on the silhouette method the optimal number of clusters has been set to 14.

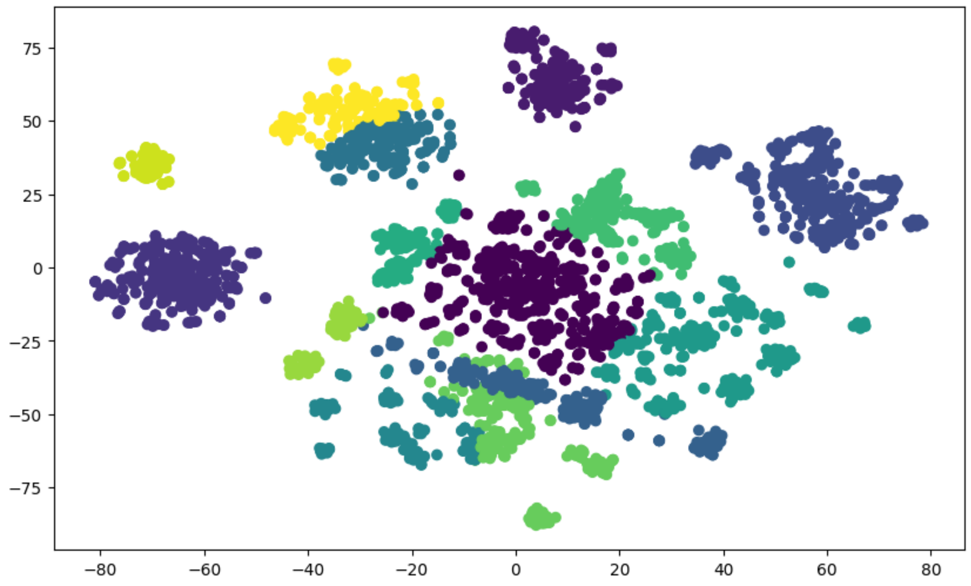


Fig. 8.21. 2D representation of the GMM output.

Similar to the final K-means output, the results are closer to the companies' goals. The output is then analysed, and the following information is gathered.

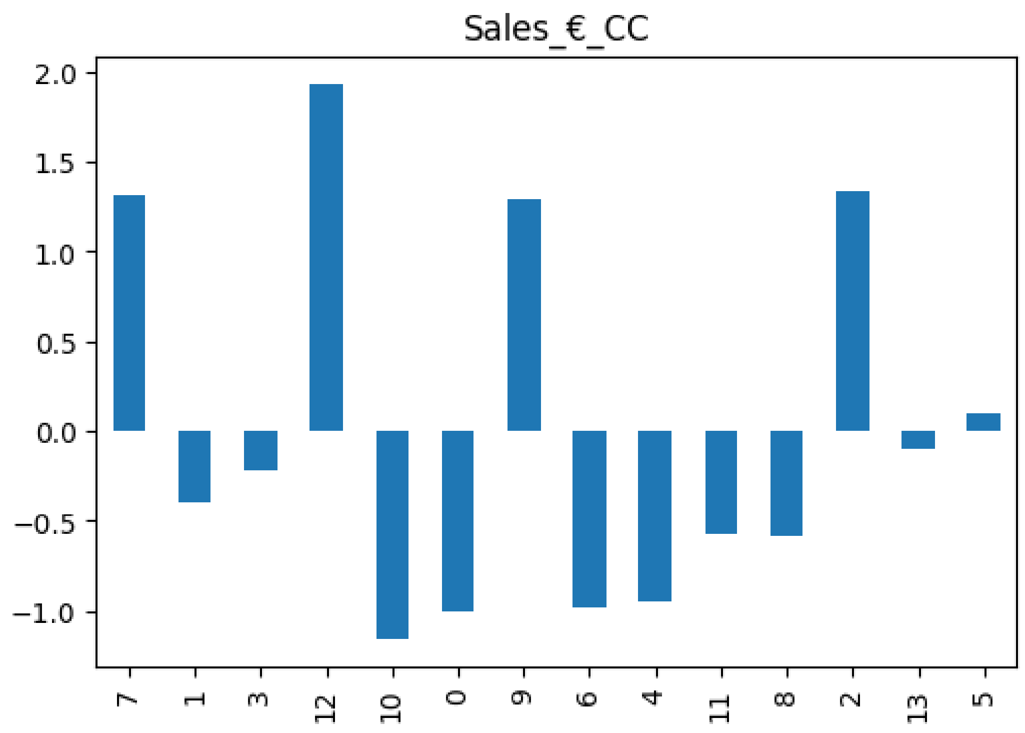


Fig. 8.22. Cluster comparison of the normalized sales averages.

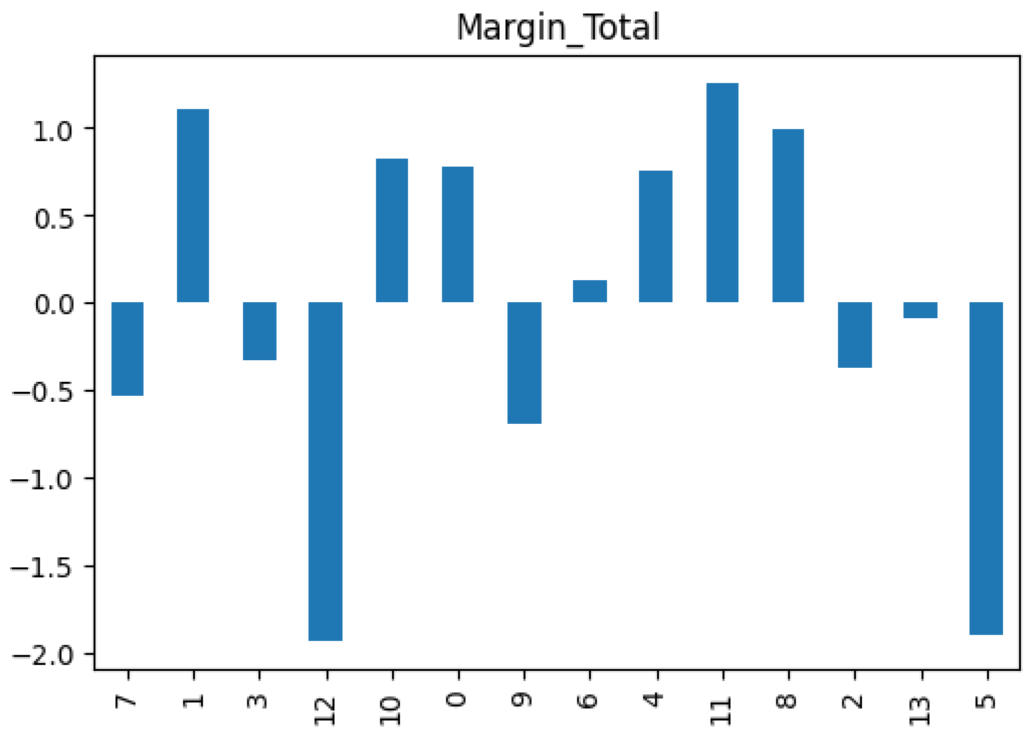


Fig. 8.23. Cluster comparison of the normalized margin averages.

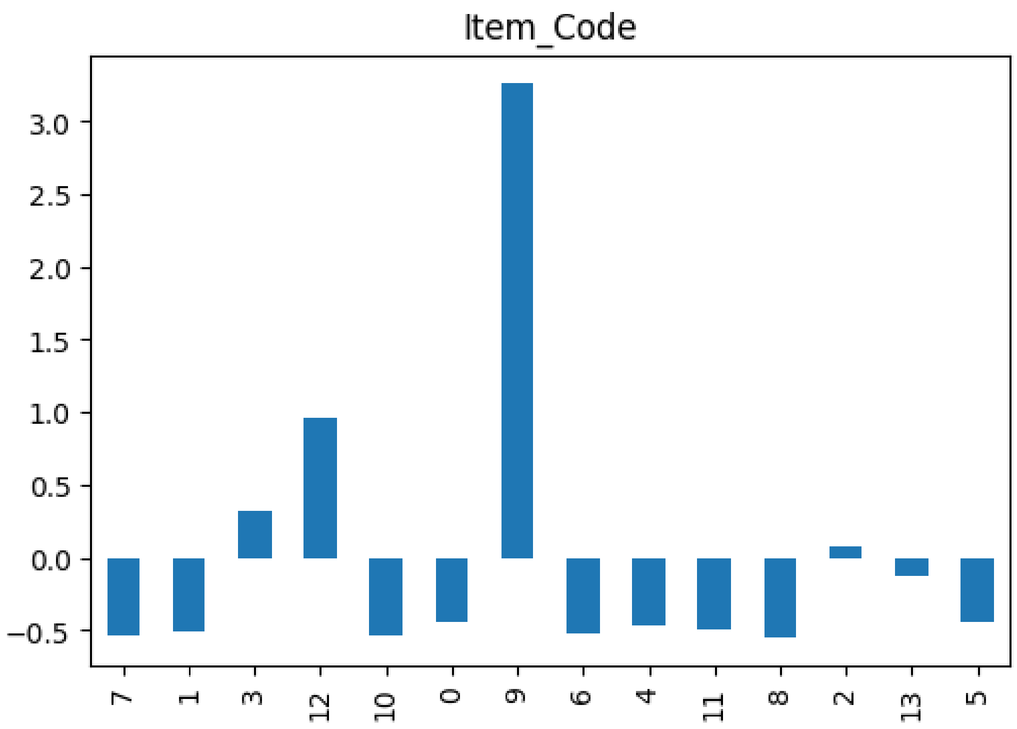


Fig. 8.24. Cluster comparison of the normalized sold item number averages.

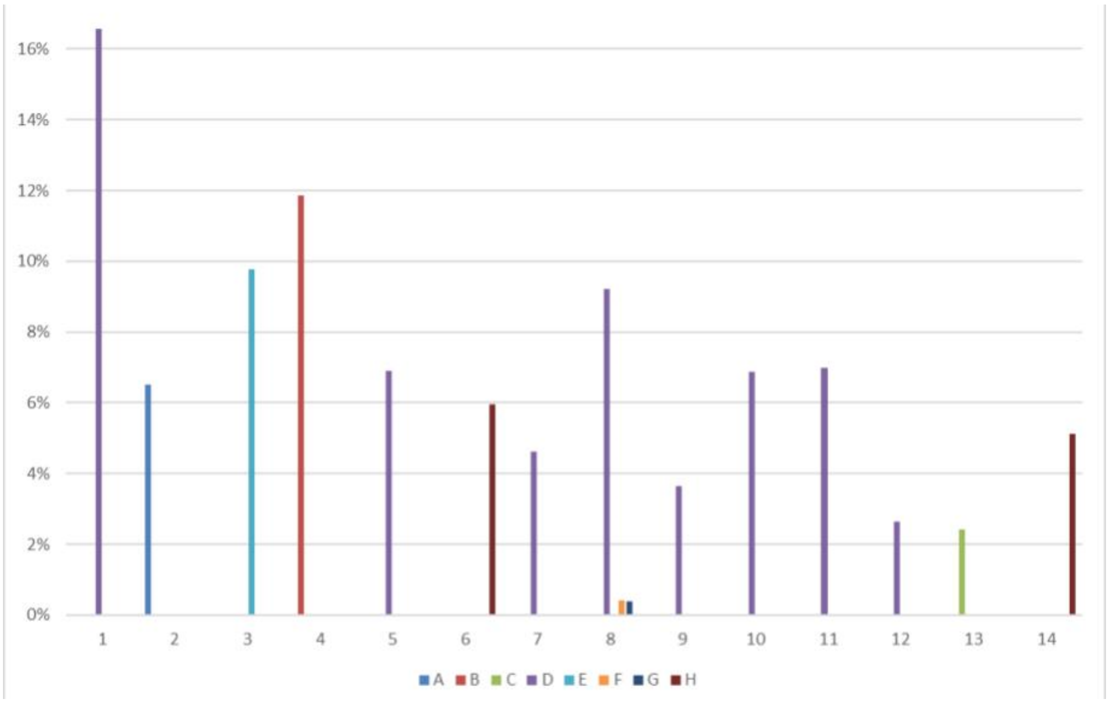


Fig. 8.25. Customer distribution per cluster based on current segmentation.

8.3.4. Conclusion.

Once all the results have been collected and analysed individually a comparison of the models’ outputs took place. Mainly K-means against GMM due to the found limitations with DBSCAN and the already acceptable results of the first mentioned.

Based on the detailed analysis of certain key customers and the homogeneity of certain clients in the dedicated to the same final industries the project developer and the manager at the company have decided the GMM model to be more adequate for the use case. It has been found that GMM has provided correct aggrupation of key players in the portfolio on a more consistent basis than K-means, while also arising new tendencies based on the rearrangement of the current most broad customer segment. This will lead to more cohesive pricing offering to certain customers that at the current date are not fitted to their most suitable segment.

8.4. Deployment.

The contents of this tool consist of the notebooks in python that tackle data preparation, dimensionality reduction, K-means, DBSCAN and GMM. This material will be left to the company with a proper manual of use to repeat the process that led to the results detailed in this document. This means that in case of revision or application in another region the company will already have the code in working condition.

9. Conclusion.

The aim of this project was to develop a machine learning (ML) model to segment the customer base of an industrial supplier. By extracting insights and optimizing engagement strategies, the model will provide a valuable tool for the company.

Once the development closes and the results are analysed, it is confirmed that the artificial intelligence models have the capability of resolving segmentation problems based on large amounts of data but not by themselves. The importance of a human input on the results analysis and the correct adjustment of the features weight has been crucial on this project. Thanks to my manager and his extensive experience with the product and the customer portfolio the results are inline with the company's vision and needs, bringing a valuable tool to verify the current segmentation and justify changes on the customer distribution based on factual and standardized data.

The final segmentation not only provides the ability to justify the customer belonging more easily to a current segment based on factual data but also defining those middle ground cases that can usually be source of frustration to the commercial team since they are managed in a way that might not be the most suitable for their characteristics.

The code developed during this project will be properly documented for its use in future revisions of the segmentation with data collected the following years. More importantly, the current project has focused on the EMEA and APAC regions, which means it can easily be exported to other operating units of the company and will serve to verify their segmentations as well.

The project has also proven the importance of experience when working with python, a tool that has had limited use during the degree but a really valuable asset to understand and exploit on a professional level. Spreadsheets are very common on the workplace, and they are a great tool for the right job but fell short of options and compute capability for some of the tasks that they end up supporting. The abilities acquired during the realisation of this project will surely carry during my professional career.

The estimated cost of around 3.921,91€ of the project is expected to generate profitability based on the better management of the customers. This will be achieved through improved customer engagement and reduced failed interactions that result in miscommunication, devolution of products or not offering the best items for their purpose. This cumulative savings on the first months will cover up the expense of the project while on the long run will generate profits for the company.

In conclusion, ML models are not about definitive results, but grounded ones and the outcome of the project has shown potential to the current portfolio with data from 2021 and 2022. This means that a revision of the segmentation can be stablished so it remains relevant and fits within the mindset of the continuous improvement. Additionally, the run cost will be limited since the code is already developed.

10. Future research.

This project has proven to me the hard work of the programmers, data scientists and data analytics pour into their projects and the importance of a cycle management of developments of this sort. The code can always be improved, the outputs can be refined, more data can be added, and this quickly translates to time. ML projects of this sort are never-ending, it is important to stick to the scope whilst being realistic with the resources. Therefore, the following elements could be explored in the future as the next step of the project.

Given the resources available at the time of this project, K-means was the most used clustering model out of the three used. This means no harm to the validity of the actual results but rather the opportunity to improve the definition of the outcome. As explained previously DBSCAN focuses highly on density and due to the nature of the data the number of clusters was too high for a relevant use. Additionally Spectral Net could not be tested due to its low level of development at the time of the project. Therefore, a deeper study on how to adjust the data to fit the model and the parameters used can be appointed.

Another source of improvement could be a better definition of the output analysis than the one presented in the project; the resources available during this project about similar cases of application and how to validate the cluster coherence have been limited to none and the best solution applied has been the norm applied to any case of segmentation. This means that without the experience of my manager and the visualisation of the output based on case defining customers the result's precision would have been less valuable than it is now. This could also be explored in a future revision of the project.

And finally, as stated previously, the project data only consists of some of the regions the company is operating. Therefore, it is only natural that the project could grow into the rest of the regions with their corresponding data, validating and proposing adjustments to the current segmentation of the customers.

11. References.

[1] Minsky, M. (2006). The emotion machine: Common-sense thinking, AI, and the future of the human mind. https://web.media.mit.edu/~minsky/Introduction.html

[2] Xu J, Kovatsch M, Mattern D, Mazza F, Harasic M, Paschke A, Lucia S. A Review on AI for Smart Manufacturing: Deep Learning Challenges and Solutions. Applied Sciences. 2022; 12(16):8239. https://doi.org/10.3390/app12168239

[3] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. ACM Computing Surveys, 31(3), 264-323. https://doi.org/10.1145/331499.331504

[4] Dressel, J., & Farid, H. (2018). The accuracy, fairness, and limits of predicting recidivism. Science Advances, 4(1), eaao5580. https://doi.org/10.1126/sciadv.aao5580

[5] Goodfellow, I., Bengio, Y., & Courville, A. (2016). DL. Cambridge, MA: MIT Press. http://www.deeplearningbook.org

[6] Brownlee, J. (2021). How to Handle Missing Data with Python. ML Mastery. https://machinelearningmastery.com/handle-missing-data-with-python/

[7] Mitchell, T. M. (1997). ML (1st ed.). McGraw-Hill, Inc. USA. ISBN 0070428077. http://www.cs.cmu.edu/~tom/mlbook.html

[8] Russell, S. J., & Norvig, P. (2019). AI: a modern approach (Vol. 3). Pearson.

[9] Keith D Foote. (2022). Machine Learning Algorithms.

https://www.dataversity.net/machine-learning-algorithms/

[10] Delighterp. ERP Features: Export Data in Excel and PDF.

https://www.delighterp.com/erp-features/export-data-in-excel-and-pdf/

[11] Ntu. (n.d.). Outlier Detection.

https://www3.ntu.edu.sg/home/elpwang/pdf_web/08_AnomalyDetection.pdf

[12] Kdnuggets. (2021). Outlier Detection with Python

https://www.kdnuggets.com/2021/05/outlier-detection-python.html

[13] Kelleher, J. D., Mac Namee, B., & D'Arcy, A. (2015). Fundamentals of ML for predictive data analytics: algorithms, worked examples, and case studies. MIT press.

[14] DataCamp. (2021). Encoding Categorical Variables in Python. https://www.datacamp.com/community/tutorials/encoding-categorical-variables

[15] DataCamp. (2020). Feature Scaling and Normalization.

https://www.datacamp.com/community/tutorials/feature-scaling-and-normalization

[16] Garousi, V., Lohmann, S., & Winter, A. (2016). Big Data Analytics in Healthcare: Opportunities and Challenges. BMC Medical Informatics and Decision Making, 16(1), 1-8. https://doi.org/10.1186/s41044-016-0014-0

[17] Dataaspirant. (n.d.). Five most popular unsupervised learning algorithms. Retrieved from https://dataaspirant.com/unsupervised-learning-algorithms/

[18] IBM. (n.d.). Unsupervised learning.

https://www.ibm.com/topics/unsupervised-learning

[19] IBM. (2021, March 12). Supervised vs. Unsupervised Learning: What’s the Difference? Retrieved from https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

[20] Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., & Kluger, Y. (2018). SpectralNet: Spectral Clustering using Deep Neural Networks. arXiv preprint arXiv:1801.01587. Retrieved from https://arxiv.org/abs/1801.01587

[21] Uppal, A., Sachdeva, V., & Sharma, S. (2020). Fake news detection using discourse segment structure analysis. In 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 751-756). Noida, India: IEEE. https://doi.org/10.1109/Confluence47617.2020.9058106

[22] IBM. (n.d.). Unsupervised learning. Retrieved from https://www.ibm.com/topics/unsupervised-learning

[23] AltexSoft. (n.d.). Unsupervised machine learning: What it is and why it matters. https://www.altexsoft.com/blog/unsupervised-machine-learning/

[24] AltexSoft. (n.d.). Unsupervised machine learning: What it is and why it matters. Retrieved from https://www.altexsoft.com/blog/unsupervised-machine-learning/

[25] IBM. (n.d.). Unsupervised learning.

https://www.ibm.com/topics/unsupervised-learning

[26] European Union. (2018). General Data Protection Regulation (GDPR).

https://gdpr-info.eu/

[27] European AI Alliance. (2021).

https://ec.europa.eu/digital-single-market/en/policies/artificial-intelligence

[28] International Organization for Standardization. (2021). ISO/IEC JTC 1/SC 42: AI. https://www.iso.org/committee/6266604.html

[29] Organization for Economic Cooperation and Development. (2020). OECD Recommendation on AI. https://www.oecd.org/going-digital/ai/oecd-recommendation-on-artificial-intelligence/

[30] International Monetary Fund. (2023, January 31). World Economic Outlook Update, January 2023. https://www.imf.org/en/Publications/WEO/Issues/2023/01/31/world-economic-outlook-update-january-2023

[31] Barla, N. (2023, January 27). How to organize DL projects: Best practices. Neptune AI. https://neptune.ai/blog/how-to-organize-deep-learning-projects-best-practices

[32] Gadgets360 (n.d.). Lenovo ThinkPad T460s (6109) [Laptop]. Retrieved February 9, 2023, from https://www.gadgets360.com/lenovo-thinkpad-t460s-6109

[33] Lenovo (n.d.). ThinkPad T460s | Laptops | Lenovo NG. Retrieved February 9, 2023, from https://www.lenovo.com/ng/en/laptops/thinkpad/t-series/ThinkPad-T460s/

[34] LaptopMag (n.d.). Lenovo ThinkPad T460s Review. Retrieved February 9, 2023, from https://www.laptopmag.com/reviews/laptops/lenovo-thinkpad-t460s

[35] WinPython: Python for Windows.

https://winpython.github.io/

Industrial Organization Engineer's Degree

Big data, ML and AI in pricing application for a manufacturing company.

Economic study

Jordi Ruiz Gratacós

SPRING 2023

Summary.

1. Budget.

1.1. Rates.

This chapter collects the corresponding rates for engineering (design and development of the solution) and the materials used in the creation of the prototype.

Chapter I: Project Development		
Code	**Description**[1]	**Unitary**
1.1	Project designer hours dedicated to the search for background information and various previous information.	7
1.1	Project designer's hours devoted to the definition and planning of the solution.	7
1.3	Project designer's hours devoted to the development and design of the solution.	7
1.4	Project designer's hours devoted to the realisation and implementation of the solution.	7
1.5	Project designer's hours devoted to the written elaboration and editing of the project documents.	7

Chapter II: Material		
Code	**Description**	**Unitary**
2.1	Trained models	5

[1] The indicated concepts correspond to groupings of the activities considered in the project planning.

1.2. Table of prices.

Chapter I: Elaboració del projecte		
Code	**Units**	**Unit price (€)**
1.1	Hours	7
1.2	Hours	7
1.3	Hours	7
1.4	Hours	7
1.5	Hours	7

Chapter II: Material		
Code	**Units**	**Unit price (€)**
2.1	CI	50

1.3 Partial budget.

Chapter I: Project Development				
ENGINIERING COST[2]				
Code	**Description**	**Total Units**	**Unit price (€)**	**Total (€)**
1.1	Project designer hours dedicated to the search for background information and various previous information.	96	7	672
1.1	Project designer's hours devoted to the definition and planning of the solution.	56	7	392
1.3	Project designer's hours devoted to the development and design of the solution.	76	7	532
1.4	Project designer's hours devoted to the realisation and implementation of the solution.	28	7	196
1.5	Project designer's hours devoted to the written elaboration and editing of the project documents.	44	7	308
COSTOS INDIRECTES				
1.6	Indirect costs			525

TOTAL Chapter I (25 % margin) **2.625,00 €**

[2] Results of planning and resource allocation.

Chapter II: Material				
COSTOS MATERIAL PROTOTIP				
Code	**Description**	**Total Units**	**Unit price (€)**	**Total (€)**
2.1	Trained models	5	50	250
COSTOS INDIRECTES				
2.2	Indirect costs			25

TOTAL CAPÍTOL II (15 % unplanned) **316,25 €**

Chapter III: Amortizations[3]				
EQUIPS INFORMÀTICS I SOFTWARE				
Codi	**Description**	**Cos Inv.**	**N (Years)**	**€/any**
3.1	Computer	500	3	166,67
3.2	Office 365	300	3	100
3.3	Software MS-Project	200	3	33,33

TOTAL CAPÍTOL III **300,00 €**

[3] The amortizations are calculated on the assumption of the completion of 2 projects per year.

1.4. Final budget.

Chapter I Total	2.625,00 €
Chapter II Total	316,25 €
Chapter III Total	300,00 €
TOTAL	3.241,25 €
IVA 21 %	680,66 €
TOTAL BUDGET	**3.921,91 €**

The project will have a total cost of eight thousand nine hundred twenty-seven euros and sixty-eight cents.

2. Economic Feasibility Analysis.

2.1. Budget of the project.

This section of the project documentation outlines the estimated cost of implementing an AI-powered pricing tool in a pricing department. The budget section provides a comprehensive overview of the financial resources required to successfully complete the project. The budget is broken down into various components, including hardware and software costs, personnel expenses, and operational costs. This section also highlights any potential cost-saving measures that can be implemented to reduce the overall cost of the project. The budget section is a crucial component of the project documentation as it provides an understanding of the financial resources required to complete the project and allows for better planning and decision-making.

Budget		
Chapter	**Description**	**€**
1	Project development	2.625,00
2	Materials	316,25
3	Amortizations	300,00
	Total	3.241,25
	IVA (21 %)	680,66
	FINAL Total	3.921,91

Taula 2.1. Project budget

2.2. Investment cost.

The investment of the project is a critical factor in determining the feasibility of the project and its overall success. This cost is estimated to be 3.921,91€ and covers the necessary resources required to implement the project, including hardware, software, and personnel expenses. This investment cost is a comprehensive representation of the financial resources required to successfully complete the project and is subject to change based on various factors, such as market conditions and project requirements. It is important to have a clear understanding of the investment cost to effectively plan and make informed decisions regarding the project.

2.3. Performance analysis.

The performance analysis of the AI project in the pricing department is not determinable at this time due to the variability of the lost opportunity revenue, which has not been defined. Despite this uncertainty, it is expected that the implementation of the AI-powered pricing tool will be profitable. The opportunity lost revenue is a crucial factor in determining the performance of the project, as it provides insight into the potential revenue that could have been generated without the implementation of the AI-powered pricing tool. Until this variable is defined, it is not possible to accurately assess the performance of the project. However, the integration of AI technology in a pricing department is likely to result in increased efficiency and better decision-making, which in turn, is expected to drive profitability.

The profitability of the project can also be justified by the reduction in workload it provides for the pricing manager. The man-hours dedicated to aligning the portfolio far surpass the cost of the project, and the implementation of the results will significantly relieve this workload.

2.4. Conclusion.

The AI-powered pricing tool project is economically feasible with an estimated investment cost of 8,927.68€. It is expected to be profitable due to increased efficiency and better decision-making, but the performance analysis is not fully determinable until the lost opportunity revenue is defined. The project is expected to bring financial benefits and improve the decision-making process in the pricing department.

Industrial Organization Engineer's Degree

Big data, ML and AI use in a pricing department of a manufacturing company.

ML clustering customers based on historical data.

Annex

Jordi Ruiz Gratacós

SPRING 2023

Summary.

Annex I. Data processing python code.

1. Import libraries.

In [ ]:
```
%reset -f
```

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from mpl_toolkits.mplot3d import Axes3D
import hashlib
from sklearn import preprocessing
import joblib

#from matplotlib.animation import FuncAnimation, PillowWriter
```

1. Import data set as is from data base. Run an innitial exploration of the data: column types, blanks, transform certain values to more workable assets (specially date)...

In [ ]:
```
data = pd.read_excel('Export Gross Margin Data v001 - All.xlsx')
```

In [ ]:
```
print(data.columns.tolist())
data.head()
```

In [ ]:
```
summary = data.info()
print(summary)
```

In [ ]:
```
print(data.dtypes)
print(data.isnull().sum())
```

In [ ]:
```
data['Unit_Selling_Price_€_CC'] = pd.to_numeric(data['Unit_Selling_Price_€_CC'], errors='coerce')
data['EMEA_Frozen_€_CC'] = pd.to_numeric(data['EMEA_Frozen_€_CC'], errors='coerce')

# assuming data is the data frame containing the export of all the order lines
epsilon = 1e-6
data['Margin'] = ((1 - (data['EMEA_Frozen_€_CC']) / data['Unit_Selling_Price_€_CC'])) * (data['Sales_€_CC'])
data.loc[data['Unit_Selling_Price_€_CC'] == 0, 'Margin'] = 0
```

In [ ]:
```
#data = data.dropna(subset=['EMEA_Frozen_€_CC', 'Unit_Selling_Price_€_CC'])
#data['Unit_Selling_Price_€_CC'] = data['Unit_Selling_Price_€_CC'].astype('object')
#data['EMEA_Frozen_€_CC'] = data['EMEA_Frozen_€_CC'].astype('object')
#data['Sales_€_CC'] = data['Sales_€_CC'].astype('object')

data['date'] = pd.to_datetime(data['Invoice_Date'])

distinct_countv2 = data['date'].nunique()
print(distinct_countv2)

data['day'] = data['date'].dt.day
data['month'] = data['date'].dt.month
data['year'] = data['date'].dt.year

data.tail()
```

1. Create the working data frame that will be encoded, standaridzed and fed to the clustering models.

In [ ]:
```
# Drop duplicates & Keep only the Customer_Code column
grouped_df = pd.DataFrame()
grouped_df = data.drop_duplicates(subset='Customer_Code')
grouped_df = grouped_df[['Customer_Code']]

grouped_df.shape
```

Adding Sales sum to the working data frame

In [ ]:
```
# Calculate the total sales quantity for each customer
total_sales = data.groupby('Customer_Code')['Sales_€_CC'].sum().reset_index()

# Merge the total sales quantity with the new data set
grouped_df = pd.merge(grouped_df, total_sales, on='Customer_Code', how='left')

# Drop rows where the value in column 'A' is equal to zero
#grouped_df = grouped_df[grouped_df['Sales_€_CC'] != 0]

grouped_df.head()
```

Adding Company Address Country

In [ ]:
```
# Create a new column for the most repeated Address_Country per customer
```

```
grouped_df['Address_Country'] = grouped_df['Customer_Code'].apply(lambda x: data[data['Customer_Code'] == x]['A

grouped_df.head()
```

```
In [ ]: # Create a new column for the most repeated Address_Country per customer
        grouped_df['Customer_Profile'] = grouped_df['Customer_Code'].apply(lambda x: data[data['Customer_Code'] == x]['

        grouped_df.head()
```

```
In [ ]: # Create a new column for the most repeated Address_Country per customer
        grouped_df['Category'] = grouped_df['Customer_Code'].apply(lambda x: data[data['Customer_Code'] == x]['Category

        grouped_df.head()
```

```
In [ ]: # Create a new column for the most repeated Address_Country per customer
        grouped_df['Customer_Category'] = grouped_df['Customer_Code'].apply(lambda x: data[data['Customer_Code'] == x][

        grouped_df.head()
```

Adding weighted margin based on sale volume to the workinf data frame

```
In [ ]: # group by Customer_Code and sum the Margin values
        grouped_df_Margin = data.groupby('Customer_Code')['Margin'].sum().reset_index()

        # merge the grouped data with grouped_df
        merged_df_Margin = pd.merge(grouped_df_Margin, grouped_df, on='Customer_Code', how='left', suffixes=('', '_Tota

        # calculate the total margin per customer
        merged_df_Margin['Margin_Total'] = np.where(merged_df_Margin['Sales_€_CC'] == 0, 0, merged_df_Margin['Margin']
        merged_df_Margin['Margin_Total'] = merged_df_Margin['Margin_Total'].round(5)

        merged_df_Margin = merged_df_Margin.groupby('Customer_Code')['Margin_Total'].max().reset_index()

        # Add to working pd
        if 'Margin_Total' in grouped_df.columns:
            del grouped_df['Margin_Total']

        grouped_df = grouped_df.merge(merged_df_Margin[['Customer_Code','Margin_Total']],on='Customer_Code',how='left')
        grouped_df.head()
        grouped_df.shape
```

```
In [ ]: grouped_df.tail(15)
```

Find the most common value per client in Item_Code and Primary_UOM Not really usefull, product family

```
In [ ]: DistinctCount = data.groupby('Customer_Code')['Item_Code'].nunique().reset_index()

        data = data.rename(columns={'Distinct_Item_Codes': 'Count_Diff_Item_Code'})

        grouped_df = pd.merge(grouped_df, DistinctCount, on='Customer_Code', how='left')

        grouped_df.shape
```

```
In [ ]: distinct_values_ProdFamily = list(data['PH_Product_Family'].unique())
        # print(distinct_values_ProdFamily)

        distinct_values_Platfotm = list(data['PH_Platform'].unique())
        print(distinct_values_Platfotm)
```

```
In [ ]: columns_to_add  = ['TEXTILES', 'PLASTICS', 'TRADE/DISTRIBUTION', 'FABRICATION']

        df_summaryv1 = data.groupby(['Customer_Code', 'PH_Platform']).agg({'Sales_€_CC': 'sum'}).reset_index()

        total_sales = df_summaryv1.groupby('Customer_Code')['Sales_€_CC'].transform('sum')
        df_summaryv1['Sales_Percentage'] = df_summaryv1['Sales_€_CC'] / total_sales

        df_summaryv2 = df_summaryv1.pivot(index='Customer_Code', columns='PH_Platform', values='Sales_Percentage')
        df_summaryv2 = df_summaryv2[columns_to_add]
        df_summaryv2.columns = [f'{col}' for col in df_summaryv2.columns]
        df_summaryv2 = df_summaryv2.reset_index()

        grouped_df = pd.merge(grouped_df, df_summaryv2, on='Customer_Code')
        grouped_df.shape
```

```
In [ ]: grouped_df.head(10)
```

Month Sales % of total added to the output dataframe

```
In [ ]: df_summary = data.groupby(['Customer_Code', 'month']).agg({'Sales_€_CC': 'sum'}).reset_index()
        df_summary = df_summary.pivot(index='Customer_Code', columns='month', values='Sales_€_CC')
        df_summary.columns = ['Month_' + str(col) for col in df_summary.columns]
        df_summary = df_summary.reset_index()
```

```
# Calculate the total sales for each customer
df_summary['Total_Sales'] = df_summary.iloc[:, 1:].sum(axis=1)

# Calculate the percentage of month sales vs total sales
for col in df_summary.columns[1:-1]:
    df_summary[col] = (df_summary[col] / df_summary['Total_Sales'])

# Drop the Total_Sales column
df_summary = df_summary.drop('Total_Sales', axis=1)

import calendar
month_dict = {i: calendar.month_name[i] for i in range(1,13)}
df_summary = df_summary.rename(columns=lambda x: month_dict[int(x.split('_')[1])] if x.startswith('Month_') els

# create a list of month names
month_names = [calendar.month_name[i] for i in range(1, 13)]

grouped_df = pd.merge(grouped_df, df_summary, on='Customer_Code')

grouped_df.tail()
```

```
In [ ]: grouped_df.shape
```

Adding Sales % of Product Family per Customer per customer total sales

```
In [ ]: # Calculate the total sales for each PH_Product_Family
total_sales = data.groupby('PH_Product_Family').agg({'Sales_€_CC': 'sum'})

# Sort the families from highest to lowest sales
total_sales = total_sales.sort_values(by='Sales_€_CC', ascending=False)

# Calculate the 88% threshold
sales_threshold = total_sales['Sales_€_CC'].sum() * 0.88

# Filter PH_Product_Family values that account for 80% of the sales
top_families = total_sales[(total_sales['Sales_€_CC'].cumsum() <= sales_threshold) & (total_sales.index != "TO

print(top_families)
```

```
In [ ]: value_threshold = 9999999 # Set a threshold for the maximum value that can be filled in the new columns

# Create new columns for each distinct value in PH_Product_Family
new_columns = pd.DataFrame(0, index=grouped_df.index, columns=data['PH_Product_Family'].unique())
grouped_df = pd.concat([grouped_df, new_columns], axis=1)

# Fill new columns with sales per customer of each PH_Product_Family
for index, row in grouped_df.iterrows():
    customer_code = row['Customer_Code']
    customer_sales = data[data['Customer_Code'] == customer_code].groupby('PH_Product_Family').agg({'Sales_€_CC
    for family, sales in customer_sales.iterrows():
        if sales['Sales_€_CC'] <= value_threshold:
            grouped_df.at[index, family] = sales['Sales_€_CC']
        else:
            grouped_df.at[index, family] = 0 # Fill with 0 or another appropriate value

# Divide by Sales_€_CC column
for family in data['PH_Product_Family'].unique():
    grouped_df[family] = grouped_df[family] / grouped_df['Sales_€_CC']

# Rename columns
grouped_df.rename(columns={family: "Sales_%_" + family for family in data['PH_Product_Family'].unique()}, inpla
```

```
In [ ]: # Get the columns to drop
columns_to_drop = ["Sales_%_" + family for family in data['PH_Product_Family'].unique() if family not in top_fa

# Drop the columns
grouped_df = grouped_df.drop(columns=columns_to_drop)
```

```
In [ ]: grouped_df.shape
```

```
In [ ]: grouped_df.head(10)
```

Adding Margin per Product Family per Customer

```
In [ ]: # Create new columns for each distinct value in PH_Product_Family
new_columns = pd.DataFrame(0, index=grouped_df.index, columns=data['PH_Product_Family'].unique())
grouped_df = pd.concat([grouped_df, new_columns], axis=1)

value_threshold = 100 # Set a threshold for the maximum value that can be filled in the new columns

# Fill new columns with margin per customer of each PH_Product_Family
for index, row in grouped_df.iterrows():
    customer_code = row['Customer_Code']
    customer_margin = data[data['Customer_Code'] == customer_code].groupby('PH_Product_Family').agg({'Margin':
```

```
    for family, margin in customer_margin.iterrows():
        if margin['Sales_€_CC'] != 0:
            new_value = margin['Margin'] / margin['Sales_€_CC']
            if new_value <= value_threshold:
                grouped_df.at[index, family] = new_value
            else:
                grouped_df.at[index, family] = 0 # Fill with 0 or another appropriate value
        else:
            grouped_df.at[index, family] = 0

# Round the filled columns to the third decimal
grouped_df = grouped_df.round(8)

# Check for and remove any infinite or very large values in numeric columns
# numeric_columns = grouped_df.select_dtypes(include=[np.number]).columns
# grouped_df = grouped_df[np.isfinite(grouped_df[numeric_columns]).all(axis=1)]

# Rename columns
grouped_df.rename(columns={family: "Margin_%_" + family for family in data['PH_Product_Family'].unique()}, inpl
```

In [ ]:
```
# Get the columns to drop
columns_to_drop = ["Margin_%_" + family for family in data['PH_Product_Family'].unique() if family not in top_f

# Drop the columns
grouped_df = grouped_df.drop(columns=columns_to_drop)
```

In [ ]:
```
grouped_df.head(10)
grouped_df.shape
```

2. Adjust missing values & outliers.

Outlier droping

In [ ]:
```
'''
# Assuming you have a DataFrame called 'grouped_df' with multiple columns that contain outliers
# Loop over each column in the DataFrame
for col in grouped_df.columns:
    # Check if the column is numerical
    if np.issubdtype(grouped_df[col].dtype, np.number):
        # Calculate the IQR for the current column
        Q1 = grouped_df[col].quantile(0.01)
        Q3 = grouped_df[col].quantile(0.99)
        IQR = Q3 - Q1

        # Define the upper and lower bounds for outliers
        lower_bound = Q1 - 500 * IQR
        upper_bound = Q3 + 500 * IQR

        # Remove the outliers from the DataFrame
        grouped_df = grouped_df[(grouped_df[col] >= lower_bound) & (grouped_df[col] <= upper_bound)]

grouped_df.shape
'''
```

In [ ]:
```
df = grouped_df
df.isnull().sum()
```

In [ ]:
```
print(df.dtypes)
```

1. Encoding.

In [ ]:
```
dffind = pd.DataFrame()
dffind = data[['Customer_Code', 'Customer_Name']].drop_duplicates()

# Perform the merge operation
dfuncoded = pd.merge(df, dffind[['Customer_Code', 'Customer_Name']], on='Customer_Code', how='left')

# Reorder the columns to move the 'Customer_Name' column to the second position
cols = dfuncoded.columns.tolist()
cols.insert(1, cols.pop(cols.index('Customer_Name')))
dfuncoded = dfuncoded[cols]

dfuncoded.to_excel('00X_FeededDataUncoded.xlsx', index=False)
dfuncoded.shape
```

In [ ]:
```
dfuncoded.head()
```

In [ ]:
```
# print(df.columns.tolist())
# df.head()
df.shape
```

In [ ]:
```
from IPython.display import display
```

```
# Create a list of pairs containing the column headers and their data types
header_type_pairs = [(col, dtype) for col, dtype in zip(df.columns, df.dtypes) if '%' not in col]

# Create a new DataFrame from the list of pairs
header_type_df = pd.DataFrame(header_type_pairs, columns=['Column', 'Dtype'])

# Display the DataFrame as a table
display(header_type_df)
```

In [ ]:
```
dfH = pd.DataFrame()

dfH = df

dfH.fillna(0, inplace=True)

# Perform one-hot encoding on the column_name column
one_hot_Address_Country = pd.get_dummies(dfH['Address_Country'], prefix='Address_Country')
one_hot_Customer_Profile = pd.get_dummies(dfH['Customer_Profile'], prefix='Customer_Profile')
one_hot_Customer_Category = pd.get_dummies(dfH['Customer_Category'], prefix='Customer_Category')
one_hot_Category = pd.get_dummies(dfH['Category'], prefix='Category')

# Concatenate the one-hot encoded columns with the original DataFrame
dfH = pd.concat([dfH, one_hot_Address_Country, one_hot_Customer_Profile, one_hot_Customer_Category, one_hot_Cat

# Drop the original column_name column
dfH = dfH.drop('Address_Country', axis=1)
dfH = dfH.drop('Customer_Profile', axis=1)
dfH = dfH.drop('Customer_Category', axis=1)
dfH = dfH.drop('Category', axis=1)

# Drop columns whose headers contain the word "other"
dfH = dfH.loc[:, ~dfH.columns.str.contains('other', case=False)]

dfH.head(15)
```

In [ ]:
```
dfH.shape
```

In [ ]:
```
# Set the display options to show all rows and columns
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

# Print the full output
print(str(dfH.dtypes))
```

In [ ]:
```
inf_rows = df.isin([np.inf, -np.inf])
print(inf_rows[inf_rows.any(axis=1)])
```

1. Standardise all data for model feeding and discuss weight ajustment.

In [ ]:
```
df = dfH
scaler = StandardScaler()
df = df.set_index('Customer_Code')
```

In [ ]:
```
'''
# Get a list of the one-hot encoded column names
one_hot_cols = list(one_hot_Address_Country.columns) + list(one_hot_Customer_Profile.columns) + list(one_hot_Cu

# Get a list of all column names in the DataFrame
all_cols = list(df.columns)

# Get a list of columns to scale by removing the one-hot encoded columns from the list of all columns
cols_to_scale = [col for col in all_cols if col not in one_hot_cols]

# Select only the columns you want to scale
df_to_scale = df[cols_to_scale]

# Fit the scaler to the data
scaler.fit(df_to_scale)

# Transform the data
df_scaled = scaler.transform(df_to_scale)
'''
```

In [ ]:
```
# Set the display options to show all rows and columns
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

df_lowest = df.nsmallest(5, 'Sales_€_CC')
print(df_lowest)
```

In [ ]:
```
df = df[df['Sales_€_CC'] > 0]
df.shape
```

In [ ]:
```
columns = df.columns
df_standardized = pd.DataFrame(scaler.fit_transform(df), columns=columns, index=df.index)

df_standardized.to_excel('df_standardized.xlsx', index=True)
df_standardized.head()
```

Annex II. Dimensionality reduction testing and K-means python code.

In [ ]:
```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from mpl_toolkits.mplot3d import Axes3D

from scipy import stats
import numpy as np

from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.manifold import TSNE
from sklearn.manifold import smacof
import umap

from keras.layers import Input, Dense
from keras.models import Model

from sklearn import preprocessing
from sklearn.metrics import silhouette_score

#from matplotlib.animation import FuncAnimation, PillowWriter
```

In [ ]:
```python
dfUncodedOG = pd.read_excel('00X_FeededDataUncoded.xlsx')
```

In [ ]:
```python
df_standardized_plus = pd.read_excel('df_standardized.xlsx')
```

In [ ]:
```python
df_standardized = df_standardized_plus#.drop('Category', axis=1)
dfUncoded = dfUncodedOG
df_standardized_plus.shape
```

In [ ]:
```python
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to set the column named 'column_name' as the index
df_standardized = df_standardized.set_index('Customer_Code')
index = df_standardized.index
```

In [ ]:
```python
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to increase the weight of the column named ['']
# Scale the columns of the data:
df_standardized['Sales_€_CC'] = MinMaxScaler(feature_range=(0, 5)).fit_transform(df_standardized[['Sales_€_CC']
df_standardized['Item_Code'] = MinMaxScaler(feature_range=(0, 150)).fit_transform(df_standardized[['Item_Code']
df_standardized['Margin_Total'] = MinMaxScaler(feature_range=(0, 55)).fit_transform(df_standardized[['Margin_To

# Select only the columns that contain the string "Category_" in their header & Scale the selected columns of t
scalerCategory = MinMaxScaler(feature_range=(0, 25))
category_columns = [col for col in df_standardized.columns if 'Category_' in col and 'Customer_Category' not in
df_standardized[category_columns] = scalerCategory.fit_transform(df_standardized[category_columns])

# Select only the columns that contain the string "Customer_Profile" in their header & Scale the selected colum
Customer_ProfileCategory = MinMaxScaler(feature_range=(0, 2))
Customer_Profile_columns = [col for col in df_standardized.columns if 'Customer_Profile' in col]
df_standardized[Customer_Profile_columns] = Customer_ProfileCategory.fit_transform(df_standardized[Customer_Pro

df_standardized.head()
```

In [ ]:
```python
# df_standardized = df_standardized.drop('Category_Industrial', axis=1)
```

In [ ]:
```python
df_standardized.shape
```

T-SNE

2D Approach

In [ ]:
```python
# assuming your standardized data is stored in a variable called 'df_standardized'
tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
X_tsne = tsne.fit_transform(df_standardized)

# plot the results
plt.scatter(X_tsne[:, 0], X_tsne[:, 1])
plt.show()
```

In [ ]:
```python
range_n_clusters = range(7,20)

meanValues_silhouette = []

for n_clusters in range_n_clusters:
    modelo_kmeans = KMeans(
                         n_clusters   = n_clusters,
                         n_init       = 20,
                         random_state = 123
                     )
    cluster_labels = modelo_kmeans.fit_predict(X_tsne)
    silhouette_avg = silhouette_score(X_tsne, cluster_labels)
    meanValues_silhouette.append(silhouette_avg)

fig, ax = plt.subplots(1, 1, figsize=(15, 8))
ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
ax.set_title("Average evolution of silhouette indexes")
ax.set_xlabel('Number of clusters')
ax.set_ylabel('Mean silhouette indexes');
```

```
In [ ]: # assuming your t-SNE transformed data is stored in a variable called 'X_tsne'
        kmeans = KMeans(n_clusters=13)
        kmeans.fit(X_tsne)

        # Create a DataFrame from the t-SNE transformed data
        df = pd.DataFrame(X_tsne, columns=['PC1', 'PC2'])

        # Add the cluster labels as a new column to the DataFrame
        df['Cluster'] = kmeans.labels_
        df['Index'] = index

        # Plot the clusters in 2D
        fig = plt.figure(figsize=(10, 6))
        ax = fig.add_subplot(111)
        scatter = ax.scatter(df['PC1'], df['PC2'], c=df['Cluster'])

        '''
        # Add a legend to the plot
        legend1 = ax.legend(*scatter.legend_elements(),
                            loc="upper right", title="Clusters")
        ax.add_artist(legend1)
        '''

        # Adjust the axis limits to zoom out
        ax.set_xlim([df['PC1'].min() - 10, df['PC1'].max() + 10])
        ax.set_ylim([df['PC2'].min() - 5, df['PC2'].max() + 5])

        plt.show()
```

```
In [ ]: Final = pd.DataFrame()
        Final ['Customer_Code'] = df['Index']
        Final ['Cluster'] = df['Cluster']

        Final = pd.merge(Final, df_standardized, on='Customer_Code', how='left')

        Final.head()
```

```
In [ ]: # delete the dataframe 'dfexport' if it exists

        # create a new dataframe 'dfexport'
        dfexport = pd.DataFrame()

        # add columns 'Customer_Code' and 'Cluster' to 'dfexport' from the dataframe 'df'
        dfexport['Customer_Code'] = df['Index']
        dfexport['Cluster'] = df['Cluster']

        # check if the column 'Cluster' exists in the dataframe 'dfUncoded'
        if 'Cluster' in dfUncoded.columns:
            # remove the column 'Cluster' from the dataframe 'df'
            dfUncoded = dfUncoded.drop('Cluster', axis=1)
            # merge 'dfexport' with 'dfUncoded' on the column 'Customer_Code'
            dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')
        else:
            # merge 'dfexport' with 'dfUncoded' on the column 'Customer_Code'
            dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')

        dfexport.head(8)
```

```
In [ ]: dfexport.to_excel('00X_dfexportCluster_Kmeans2D.xlsx', index=False)
```

3D Approach

```
In [ ]: # assuming your standardized data is stored in a variable called 'X'
        tsne = TSNE(n_components=3, perplexity=30, learning_rate=200)
        X_tsne = tsne.fit_transform(df_standardized)

        # plot the results
        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        ax.scatter(X_tsne[:, 0], X_tsne[:, 1], X_tsne[:, 2])
        plt.show()
```

In [ ]:
```
range_n_clusters = range(4,20)

meanValues_silhouette = []

for n_clusters in range_n_clusters:
    modelo_kmeans = KMeans(
                        n_clusters   = n_clusters,
                        n_init       = 20,
                        random_state = 123
                    )
    cluster_labels = modelo_kmeans.fit_predict(X_tsne)
    silhouette_avg = silhouette_score(X_tsne, cluster_labels)
    meanValues_silhouette.append(silhouette_avg)

fig, ax = plt.subplots(1, 1, figsize=(15, 8))
ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
ax.set_title("Average evolution of silhouette indexes")
ax.set_xlabel('Number of clusters')
ax.set_ylabel('Mean silhouette indexes');
```

In [ ]:
```
# assuming your t-SNE transformed data is stored in a variable called 'X_tsne'
kmeans = KMeans(n_clusters=11)
kmeans.fit(X_tsne)

# Create a DataFrame from the t-SNE transformed data
df = pd.DataFrame(X_tsne, columns=['PC1', 'PC2', 'PC3'])

# Add the cluster labels as a new column to the DataFrame
df['Cluster'] = kmeans.labels_
df['Index'] = index

# Plot the clusters in 3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df['PC1'], df['PC2'], df['PC3'], c=df['Cluster'])
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.show()
```

In [ ]:
```
Final = pd.DataFrame()
Final ['Customer_Code'] = df['Index']
Final ['Cluster'] = df['Cluster']

Final = pd.merge(Final, df_standardized, on='Customer_Code', how='left')

Final.head()
```

In [ ]:
```
dfexport = pd.DataFrame()
dfexport ['Customer_Code'] = df['Index']
dfexport ['Cluster'] = df['Cluster']

dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')

dfexport.to_excel('00X_dfexportCluster_Kmeans3D.xlsx', index=False)

dfexport.head()
```

In [ ]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df['PC1'], df['PC2'], df['PC3'],c=df['Cluster'])
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.show()

def update(i):
    ax.view_init(elev=10., azim=i)

anim = FuncAnimation(fig, update, frames=np.arange(0, 360, 1), interval=25)
anim.save('my_animation.gif', writer='imagemagick')
plt.show()
```

PCA

2D Approach

In [ ]:
```
pca = PCA(n_components=2)
df_reduced = pd.DataFrame(pca.fit_transform(df_standardized), columns=['PC1', 'PC2'],index=df_standardized.inde

df_reduced.plot.scatter(x='PC1', y='PC2')

#ax.scatter(df_reduced['PC1'], df_reduced['PC2'])
```

```
plt.show()
```

```
In [ ]: from scipy.spatial import distance

        # Calculate the mean and covariance matrix of the PCA dimensionality-reduced dataset
        mean = np.mean(df_reduced, axis=0)
        cov = np.cov(df_reduced.T)

        # Calculate the Mahalanobis distance for each observation
        mahalanobis_distance = pd.Series([distance.mahalanobis(observation, mean, np.linalg.inv(cov)) for observation i

        # Set a threshold for the maximum Mahalanobis distance
        distance_threshold = 5

        # Identify and remove outliers
        outliers = df_reduced[mahalanobis_distance > distance_threshold]
        cleaned_dataset = df_reduced[mahalanobis_distance <= distance_threshold]

        # Plot the cleaned dataset
        cleaned_dataset.plot.scatter(x='PC1', y='PC2')
        plt.show()
```

```
In [ ]: range_n_clusters = range(4,20)

        meanValues_silhouette = []

        for n_clusters in range_n_clusters:
            modelo_kmeans = KMeans(
                                n_clusters   = n_clusters,
                                n_init       = 20,
                                random_state = 123
                            )
            cluster_labels = modelo_kmeans.fit_predict(df_reduced)
            silhouette_avg = silhouette_score(df_reduced, cluster_labels)
            meanValues_silhouette.append(silhouette_avg)

        fig, ax = plt.subplots(1, 1, figsize=(15, 8))
        ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
        ax.set_title("Average evolution of silhouette indexes")
        ax.set_xlabel('Number of clusters')
        ax.set_ylabel('Mean silhouette indexes');
```

```
In [ ]: # assuming your t-SNE transformed data is stored in a variable called 'X_tsne'
        kmeans = KMeans(n_clusters=7)
        kmeans.fit(df_reduced)

        # Create a DataFrame from the t-SNE transformed data
        df = pd.DataFrame(df_reduced, columns=['PC1', 'PC2'])

        # Add the cluster labels as a new column to the DataFrame
        df['Cluster'] = kmeans.labels_
        df['Index'] = index

        # Plot the clusters in 2D
        fig = plt.figure(figsize=(10, 7))
        ax = fig.add_subplot(111)
        scatter = ax.scatter(df['PC1'], df['PC2'], c=df['Cluster'])

        # Add a legend to the plot
        legend1 = ax.legend(*scatter.legend_elements(),
                            loc="upper right", title="Clusters")
        ax.add_artist(legend1)

        # Adjust the axis limits to zoom out
        ax.set_xlim([df['PC1'].min() - 3, df['PC1'].max() + 3])
        ax.set_ylim([df['PC2'].min() - 2, df['PC2'].max() + 2])

        ax.set_xlabel('PC1')
        ax.set_ylabel('PC2')
        plt.show()
```

```
In [ ]: dfexport = pd.DataFrame()
        dfexport ['Customer_Code'] = df['Index']
        dfexport ['Cluster'] = df['Cluster']

        Input_Info = pd.DataFrame()

        Input_Info['Sales_€_CC'] = data.groupby('Customer_Code')['Sales_€_CC'].sum()
        Input_Info['Category'] = data.groupby('Customer_Code')['Category'].apply(lambda x: x.value_counts().index[0])
        Input_Info['Customer_Name'] = data.groupby('Customer_Code')['Customer_Name'].apply(lambda x: x.value_counts().i

        dfexport = pd.merge(dfexport, Input_Info, on='Customer_Code', how='left')

        dfexport.to_excel('00X_dfexportCluster_Kmeans2D.xlsx', index=False)

        dfexport.head()
```

3D approach

In [ ]:
```python
pca = PCA(n_components=3)
df_reduced = pd.DataFrame(pca.fit_transform(df_standardized), columns=['PC1', 'PC2', 'PC3'],index=df_standardiz

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df_reduced['PC1'], df_reduced['PC2'], df_reduced['PC3'])
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.show()
```

In [ ]:
```python
range_n_clusters = range(1, 11)
inertias = []

for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters)
    kmeans.fit(df_reduced)
    inertias.append(kmeans.inertia_)

plt.plot(range_n_clusters, inertias, 'o-')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Sum of squared distances')
plt.show()
```

In [ ]:
```python
range_n_clusters = range(3,15)

meanValues_silhouette = []

for n_clusters in range_n_clusters:
    modelo_kmeans = KMeans(
                        n_clusters   = n_clusters,
                        n_init       = 20,
                        random_state = 123
                    )
    cluster_labels = modelo_kmeans.fit_predict(df_reduced)
    silhouette_avg = silhouette_score(df_reduced, cluster_labels)
    meanValues_silhouette.append(silhouette_avg)

fig, ax = plt.subplots(1, 1, figsize=(15, 8))
ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
ax.set_title("Average evolution of silhouette indexes")
ax.set_xlabel('Number of clusters')
ax.set_ylabel('Mean silhouette indexes');
```

In [ ]:
```python
kmeans = KMeans(n_clusters=4)
kmeans.fit(df_reduced)

# Add the cluster labels as a new column to the reduced data
df_reduced['Cluster'] = kmeans.labels_

# Plot the clusters in 3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df_reduced['PC1'], df_reduced['PC2'], df_reduced['PC3'], c=df_reduced['Cluster'])
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.show()
```

In [ ]:
```python
# Define the centroids variable
centroids = kmeans.cluster_centers_

# Calculate the variance of the centroids for every dimension
variance = np.var(centroids, axis=0)

# Rank the PCA components according to their importance
component_importance = np.argsort(variance)[::-1]

# Calculate the feature importance scores for the original features
feature_importance = pca.components_[component_importance] ** 2

# Rank the original features according to their importance
feature_ranking = np.argsort(feature_importance)[::-1]

# Assume you have a DataFrame called 'df_standardized'
# Get the feature names from the column headings
feature_names = df_standardized.columns.tolist()

# Flatten the feature_ranking array
feature_ranking_flat = feature_ranking.flatten()

# Print the ranked feature names using the flattened feature_ranking array
print('Ranked feature names:', [feature_names[i] for i in feature_ranking_flat])
```

In [ ]:
```python
dfexport = pd.DataFrame()
```

```
In [ ]: dfexport = pd.DataFrame()
        df_reduced = df_reduced.reset_index()
        dfexport ['Customer_Code'] = df_reduced['Customer_Code']
        dfexport ['Cluster'] = df_reduced['Cluster']

        dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')

        dfexport.to_excel('00X_dfexportCluster_Kmeans3D.xlsx', index=False)

        dfexport.head(8)
```

```
In [ ]: Final = pd.DataFrame()
        Final ['Customer_Code'] = df_reduced['Customer_Code']
        Final ['Cluster'] = df_reduced['Cluster']

        Final = pd.merge(Final, df_standardized, on='Customer_Code', how='left')

        Final.head()
```

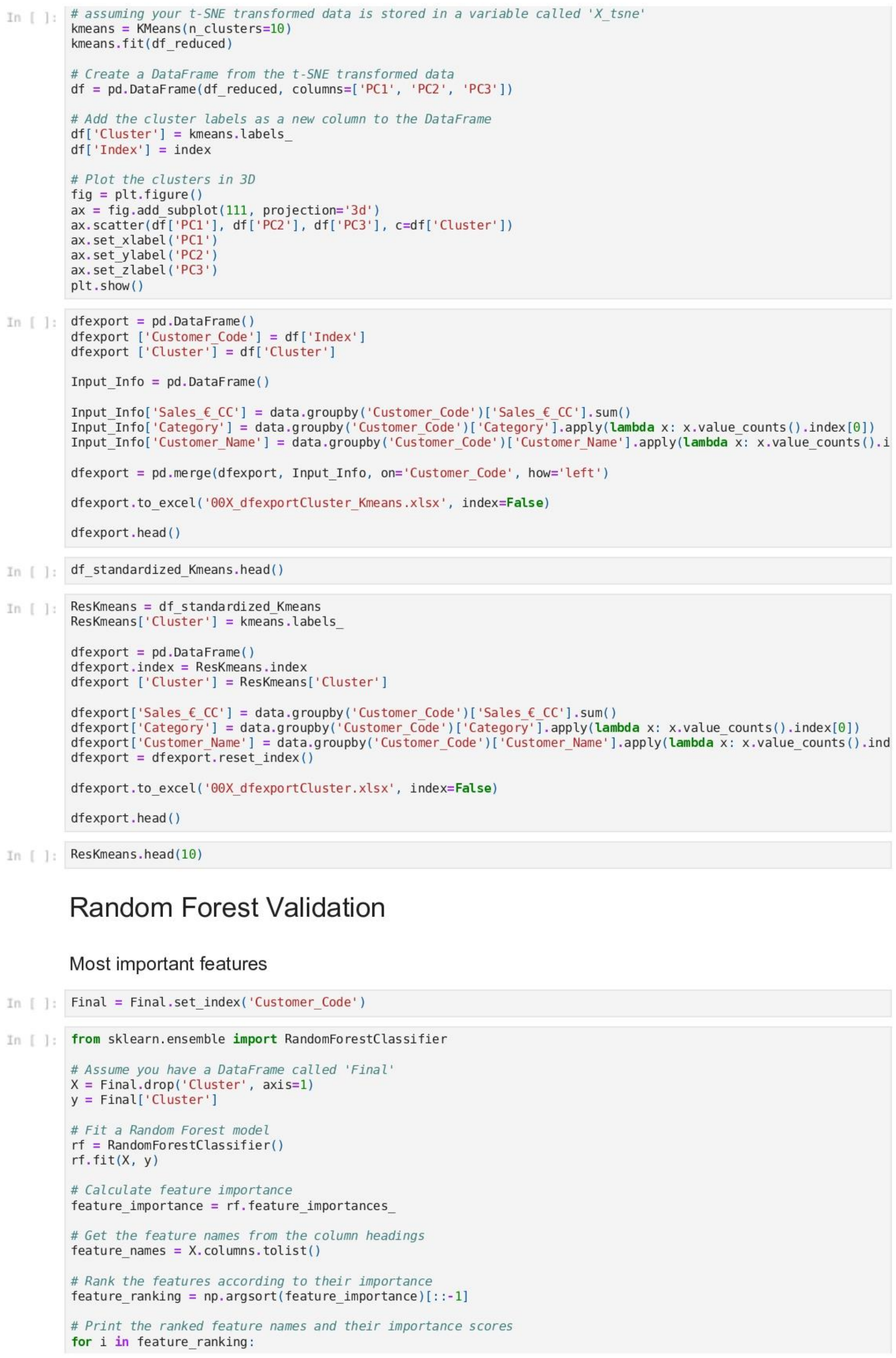
UMAP

```
In [ ]: # Create an instance of UMAP
        reducer = umap.UMAP(n_components=2)

        # Fit and transform your data
        df_reduced = reducer.fit_transform(df_standardized)

        # Plot the results
        plt.scatter(df_reduced[:, 0], df_reduced[:, 1])
        plt.show()
```

```
In [ ]: # Create an instance of UMAP
        reducer = umap.UMAP(n_components=3)

        # Fit and transform your data
        df_reduced = reducer.fit_transform(df_standardized)

        # plot the results
        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        ax.scatter(df_reduced[:, 0], df_reduced[:, 1], df_reduced[:, 2])
        plt.show()
```

```
In [ ]: from scipy import stats
        df_reduced = df_reduced[(np.abs(stats.zscore(df_reduced)) < 0.6).all(axis=1)]

        # plot the results
        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        ax.scatter(df_reduced[:, 0], df_reduced[:, 1], df_reduced[:, 2])
        plt.show()
```

```
In [ ]: range_n_clusters = range(1, 11)
        inertias = []

        for n_clusters in range_n_clusters:
            kmeans = KMeans(n_clusters=n_clusters)
            kmeans.fit(df_reduced)
            inertias.append(kmeans.inertia_)

        plt.plot(range_n_clusters, inertias, 'o-')
        plt.xlabel('Number of clusters (k)')
        plt.ylabel('Sum of squared distances')
        plt.show()

        range_n_clusters = range(3,11)

        meanValues_silhouette = []

        for n_clusters in range_n_clusters:
            modelo_kmeans = KMeans(
                                n_clusters   = n_clusters,
                                n_init       = 20,
                                random_state = 123
                            )
            cluster_labels = modelo_kmeans.fit_predict(df_reduced)
            silhouette_avg = silhouette_score(df_reduced, cluster_labels)
            meanValues_silhouette.append(silhouette_avg)

        fig, ax = plt.subplots(1, 1, figsize=(15, 8))
        ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
        ax.set_title("Average evolution of silhouette indexes")
        ax.set_xlabel('Number of clusters')
        ax.set_ylabel('Mean silhouette indexes');
```



```
In [ ]: # assuming your t-SNE transformed data is stored in a variable called 'X_tsne'
        kmeans = KMeans(n_clusters=10)
        kmeans.fit(df_reduced)

        # Create a DataFrame from the t-SNE transformed data
        df = pd.DataFrame(df_reduced, columns=['PC1', 'PC2', 'PC3'])

        # Add the cluster labels as a new column to the DataFrame
        df['Cluster'] = kmeans.labels_
        df['Index'] = index

        # Plot the clusters in 3D
        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')
        ax.scatter(df['PC1'], df['PC2'], df['PC3'], c=df['Cluster'])
        ax.set_xlabel('PC1')
        ax.set_ylabel('PC2')
        ax.set_zlabel('PC3')
        plt.show()
```

```
In [ ]: dfexport = pd.DataFrame()
        dfexport ['Customer_Code'] = df['Index']
        dfexport ['Cluster'] = df['Cluster']

        Input_Info = pd.DataFrame()

        Input_Info['Sales_€_CC'] = data.groupby('Customer_Code')['Sales_€_CC'].sum()
        Input_Info['Category'] = data.groupby('Customer_Code')['Category'].apply(lambda x: x.value_counts().index[0])
        Input_Info['Customer_Name'] = data.groupby('Customer_Code')['Customer_Name'].apply(lambda x: x.value_counts().i

        dfexport = pd.merge(dfexport, Input_Info, on='Customer_Code', how='left')

        dfexport.to_excel('00X_dfexportCluster_Kmeans.xlsx', index=False)

        dfexport.head()
```

```
In [ ]: df_standardized_Kmeans.head()
```

```
In [ ]: ResKmeans = df_standardized_Kmeans
        ResKmeans['Cluster'] = kmeans.labels_

        dfexport = pd.DataFrame()
        dfexport.index = ResKmeans.index
        dfexport ['Cluster'] = ResKmeans['Cluster']

        dfexport['Sales_€_CC'] = data.groupby('Customer_Code')['Sales_€_CC'].sum()
        dfexport['Category'] = data.groupby('Customer_Code')['Category'].apply(lambda x: x.value_counts().index[0])
        dfexport['Customer_Name'] = data.groupby('Customer_Code')['Customer_Name'].apply(lambda x: x.value_counts().ind
        dfexport = dfexport.reset_index()

        dfexport.to_excel('00X_dfexportCluster.xlsx', index=False)

        dfexport.head()
```

```
In [ ]: ResKmeans.head(10)
```

Random Forest Validation

Most important features

```
In [ ]: Final = Final.set_index('Customer_Code')
```

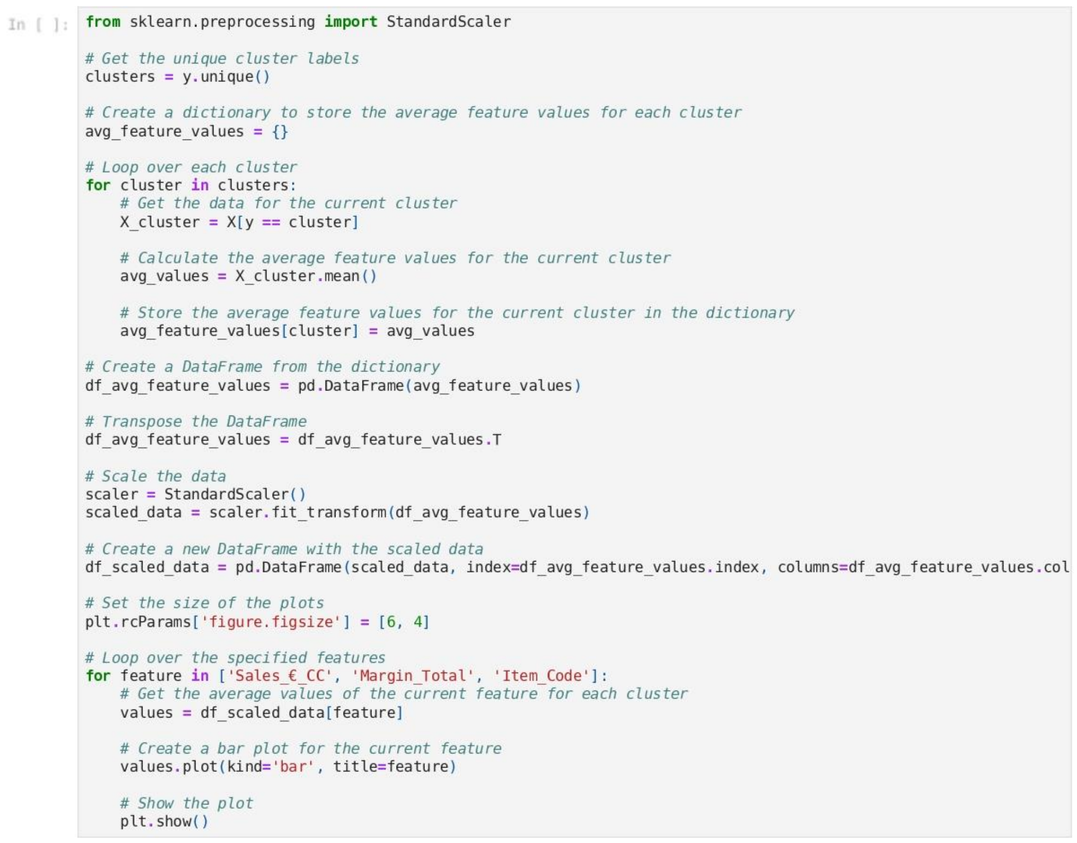
```
In [ ]: from sklearn.ensemble import RandomForestClassifier

        # Assume you have a DataFrame called 'Final'
        X = Final.drop('Cluster', axis=1)
        y = Final['Cluster']

        # Fit a Random Forest model
        rf = RandomForestClassifier()
        rf.fit(X, y)

        # Calculate feature importance
        feature_importance = rf.feature_importances_

        # Get the feature names from the column headings
        feature_names = X.columns.tolist()

        # Rank the features according to their importance
        feature_ranking = np.argsort(feature_importance)[::-1]

        # Print the ranked feature names and their importance scores
        for i in feature_ranking:
```

```
    print(f'{feature_names[i]}: {feature_importance[i]:.4f}')
```

In [ ]:
```
# Get the unique cluster labels
clusters = y.unique()

# Create a dictionary to store the average feature values for each cluster
avg_feature_values = {}

# Loop over each cluster
for cluster in clusters:
    # Get the data for the current cluster
    X_cluster = X[y == cluster]

    # Calculate the average feature values for the current cluster
    avg_values = X_cluster.mean()

    # Store the average feature values for the current cluster in the dictionary
    avg_feature_values[cluster] = avg_values

# Create a DataFrame from the dictionary
df_avg_feature_values = pd.DataFrame(avg_feature_values)

# Transpose the DataFrame
df_avg_feature_values = df_avg_feature_values.T

# Calculate the variance of each feature for each cluster
variance = df_avg_feature_values.var()

# Calculate the inverse of the variance for each feature
inverse_variance = variance.rdiv(1)

# Sort the features by their variance in descending order
inverse_variance = inverse_variance.sort_values(ascending=False)

# Get the number of clusters
n_clusters = len(clusters)

# Get the top features with the highest variance
top_features = variance[:n_clusters].index.tolist()

# Set the size of the plots
plt.rcParams['figure.figsize'] = [6, 4]

# Loop over the top features
for feature in top_features:
    # Get the average values of the current feature for each cluster
    values = df_avg_feature_values[feature]

    # Create a bar plot for the current feature
    values.plot(kind='bar', title=feature)

    # Show the plot
    plt.show()
```

In [ ]:
```
# Create an empty DataFrame to store the results
diff = pd.DataFrame(columns=df_avg_feature_values.columns)

# Calculate the median values of each feature for all the data
median_values_all = df_avg_feature_values.median()

# Loop over all clusters
for cluster in sorted(clusters):
    # Get the median values of each feature for the current cluster
    median_values = df_avg_feature_values.loc[cluster].median()

    # Calculate the absolute difference in median values between the cluster and the average of all the data
    diff.loc[f'Cluster {cluster}'] = abs(median_values - median_values_all)

# Display the results
print('Absolute difference in median values between each cluster and the average of all the data:')
display(diff)
```

In [ ]:
```
# Calculate the variance of each feature for each cluster
variance = df_avg_feature_values.var()

# Calculate the inverse of the variance for each feature
inverse_variance = variance.rdiv(1)

# Sort the features by their variance in descending order
inverse_variance = inverse_variance.sort_values(ascending=False)

# Get the number of clusters
n_clusters = len(clusters)

# Get the top features with the highest variance
top_features = inverse_variance[:n_clusters].index.tolist()
```



In [ ]:

```python
from sklearn.preprocessing import StandardScaler

# Get the unique cluster labels
clusters = y.unique()

# Create a dictionary to store the average feature values for each cluster
avg_feature_values = {}

# Loop over each cluster
for cluster in clusters:
    # Get the data for the current cluster
    X_cluster = X[y == cluster]

    # Calculate the average feature values for the current cluster
    avg_values = X_cluster.mean()

    # Store the average feature values for the current cluster in the dictionary
    avg_feature_values[cluster] = avg_values

# Create a DataFrame from the dictionary
df_avg_feature_values = pd.DataFrame(avg_feature_values)

# Transpose the DataFrame
df_avg_feature_values = df_avg_feature_values.T

# Scale the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df_avg_feature_values)

# Create a new DataFrame with the scaled data
df_scaled_data = pd.DataFrame(scaled_data, index=df_avg_feature_values.index, columns=df_avg_feature_values.col

# Set the size of the plots
plt.rcParams['figure.figsize'] = [6, 4]

# Loop over the specified features
for feature in ['Sales_€_CC', 'Margin_Total', 'Item_Code']:
    # Get the average values of the current feature for each cluster
    values = df_scaled_data[feature]

    # Create a bar plot for the current feature
    values.plot(kind='bar', title=feature)

    # Show the plot
    plt.show()
```

Annex III. DBSCAN python code

In [ ]:
```
%matplotlib inline

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import normalize
from sklearn.cluster import KMeans, DBSCAN
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from scipy.spatial import distance
from kneed import KneeLocator
from sklearn.neighbors import NearestNeighbors
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
from collections import Counter
from scipy import stats
from matplotlib.animation import FuncAnimation
```

In [ ]:
```
dfUncoded = pd.read_excel('00X_FeededDataUncoded.xlsx')

df_standardized_plus = pd.read_excel('df_standardized.xlsx')
```

In [ ]:
```
df_standardized = df_standardized_plus#.drop('Category', axis=1)
df_standardized_plus.shape
```

In [ ]:
```
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to set the column named 'column_name' as the index
df_standardized = df_standardized.set_index('Customer_Code')
index = df_standardized.index
```

In [ ]:
```
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to increase the weight of the column named ['']
# Scale the columns of the data:
df_standardized['Sales_€_CC'] = MinMaxScaler(feature_range=(0, 5)).fit_transform(df_standardized[['Sales_€_CC']
df_standardized['Item_Code'] = MinMaxScaler(feature_range=(0, 150)).fit_transform(df_standardized[['Item_Code']
df_standardized['Margin_Total'] = MinMaxScaler(feature_range=(0, 55)).fit_transform(df_standardized[['Margin_To

# Select only the columns that contain the string "Category_" in their header & Scale the selected columns of t
scalerCategory = MinMaxScaler(feature_range=(0, 25))
category_columns = [col for col in df_standardized.columns if 'Category_' in col and 'Customer_Category' not in
df_standardized[category_columns] = scalerCategory.fit_transform(df_standardized[category_columns])

# Select only the columns that contain the string "Customer_Profile" in their header & Scale the selected colum
Customer_ProfileCategory = MinMaxScaler(feature_range=(0, 2))
Customer_Profile_columns = [col for col in df_standardized.columns if 'Customer_Profile' in col]
df_standardized[Customer_Profile_columns] = Customer_ProfileCategory.fit_transform(df_standardized[Customer_Pro

df_standardized.head()
```

In [ ]:
```
df_standardized.shape
```

DBSCAN 2D

We use the k-NN (k-nearest neighbors) algorithm to find the nearest neighbors of each point in a data set that has been reduced to two components by PCA. It then plots the distance of the k-nearest neighbor to help select the appropriate value of k for the k-means based clustering algorithm:

In [ ]:
```
# assuming your standardized data is stored in a variable called 'df_standardized'
tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
df_reduced_TSNE = tsne.fit_transform(df_standardized)

# plot the results
plt.scatter(df_reduced_TSNE[:, 0], df_reduced_TSNE[:, 1])
plt.show()
```

In [ ]:
```
df_reduced = df_reduced_TSNE

# n_neighbors = 5 as kneighbors function returns distance of point to itself (i.e. first column will be zeros)
nbrs = NearestNeighbors(n_neighbors = 2).fit(df_reduced)

# Find the k-neighbors of a point
neigh_dist, neigh_ind = nbrs.kneighbors(df_reduced)

# Sort the neighbor distances (lengths to points) in ascending order
# axis = 0 represents sort along first axis i.e. sort along row
sort_neigh_dist = np.sort(neigh_dist, axis = 0)
```

```
plt.subplots(figsize=(17,10))
k_dist = sort_neigh_dist[:, 1]
plt.plot(k_dist)
plt.ylabel("k-NN distance")
plt.xlabel("Sorted observations (4th NN)")
plt.show()
```

We use the "KneeLocator" library to find the "elbow point" on a graph of distance and number of clusters. Then, we plot the result and display the value of the estimated elbow point. The estimated elbow point value is used as the optimal number of clusters for a k-means based clustering algorithm:

In [ ]:
```
kneedle = KneeLocator(x = range(1, len(neigh_dist)+1), y = k_dist, S = 1.0,
                      curve = "concave", direction = "increasing", online=True)
kneedle.plot_knee()
plt.show()

# Get the estimate of knee point
print('The Knee point (eps) is:', kneedle.knee_y)
```

This code uses the estimated value of the "elbow point" as the optimal value for the epsilon parameter in the DBSCAN clustering algorithm. It then applies the DBSCAN algorithm to the data set reduced to two components using PCA and displays the number of clusters found:

In [ ]:
```
# Find the optimum value of epsilon from the graph
epsilon = kneedle.knee_y+1

# Use the DBSCAN algorithm to perform the data clustering.
dbscan = DBSCAN(eps=epsilon, min_samples=4)
clusters = dbscan.fit_predict(df_reduced)
print('Number of clusters:',clusters,epsilon)
```

In [ ]:
```
print(epsilon)
```

Using the DBSCAN clustering algorithm with the optimal values of the input parameters and obtain the corresponding cluster labels for each point in the data set. Then, it counts the number of points in each cluster and a scatter plot is generated showing each point colored according to its assigned cluster:

In [ ]:
```
# Get cluster labels
labels = clusters

# Count the number of points in each cluster
cluster_counts = Counter(labels)

# Plot the t-SNE reduced data with cluster labels as hue
plt.subplots(figsize=(17,10))
custom_palette = sns.color_palette("tab10")
p = sns.scatterplot(x = df_reduced[:, 0], y = df_reduced[:, 1], hue = labels, legend = "full", palette = custom
sns.move_legend(p, "upper right", bbox_to_anchor = (1.17, 1.), title = 'Clusters')
plt.show()
```

DBSCAN 3D

We use the k-NN (k-nearest neighbors) algorithm to find the nearest neighbors of each point in a data set that has been reduced to two components by PCA. It then plots the distance of the k-nearest neighbor to help select the appropriate value of k for the k-means based clustering algorithm:

In [ ]:
```
pca = PCA(n_components=3)
df_reduced = pd.DataFrame(pca.fit_transform(df_standardized), columns=['PC1', 'PC2', 'PC3'],index=df_standardiz

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df_reduced['PC1'], df_reduced['PC2'], df_reduced['PC3'])
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.show()
```

In [ ]:
```
# n_neighbors = 5 as kneighbors function returns distance of point to itself (i.e. first column will be zeros)
nbrs = NearestNeighbors(n_neighbors = 2).fit(df_reduced)
# Find the k-neighbors of a point
neigh_dist, neigh_ind = nbrs.kneighbors(df_reduced)
# Sort the neighbor distances (lengths to points) in ascending order
# axis = 0 represents sort along first axis i.e. sort along row
sort_neigh_dist = np.sort(neigh_dist, axis = 0)

plt.subplots(figsize=(17,10))
k_dist = sort_neigh_dist[:, 1]
plt.plot(k_dist)
plt.ylabel("k-NN distance")
plt.xlabel("Sorted observations (4th NN)")
plt.show()
```

We use the "KneeLocator" library to find the "elbow point" on a graph of distance and number of clusters. Then, we plot the result and display the value of the estimated elbow point. The estimated elbow point value is used as the optimal number of clusters for a k-means based clustering algorithm:

In [ ]:
```
kneedle = KneeLocator(x = range(1, len(neigh_dist)+1), y = k_dist, S = 1.0,
                      curve = "concave", direction = "increasing", online=True)
kneedle.plot_knee()
plt.show()

# Get the estimate of knee point
print('The Knee point (eps) is:', kneedle.knee_y)
```

This code uses the estimated value of the "elbow point" as the optimal value for the epsilon parameter in the DBSCAN clustering algorithm. It then applies the DBSCAN algorithm to the data set reduced to two components using PCA and displays the number of clusters found:

In [ ]:
```
# Find the optimum value of epsilon from the graph
epsilon = kneedle.knee_y

# Use the DBSCAN algorithm to perform the data clustering.
dbscan = DBSCAN(eps=epsilon, min_samples=4)
clusters = dbscan.fit_predict(df_reduced)
clusters = DBSCAN(eps = epsilon, min_samples = 4).fit(df_reduced)
print(clusters)
```

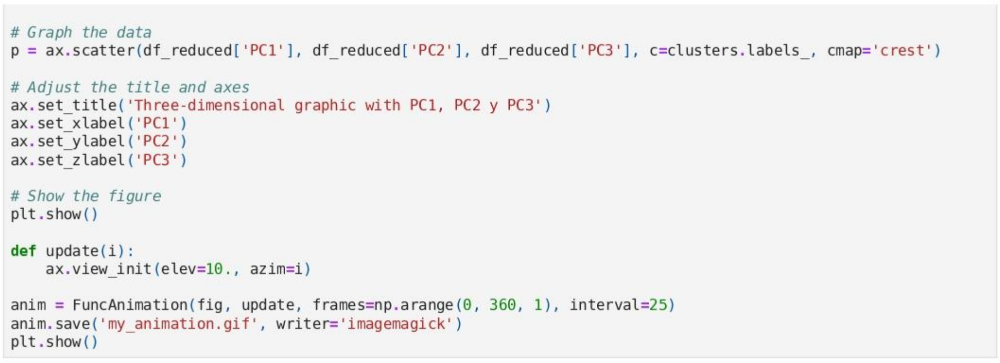
The code shows how to obtain cluster labels, create a 3-D scatter plot of data (PC1, PC2 and PC3) using different combinations of axes, and add a color legend to represent the different clusters:

In [ ]:
```
# get cluster labels
clusters.labels_
Counter(clusters.labels_)

plt.subplots(figsize=(17,10))
custom_palette = sns.color_palette("crest", as_cmap=True)
p = sns.scatterplot(data = df_reduced, x = "PC1", y = "PC2", hue = clusters.labels_, legend = "full", palette =
sns.move_legend(p, "upper right", bbox_to_anchor = (1.17, 1.), title = 'Clusters')
plt.show()

plt.subplots(figsize=(17,10))
custom_palette = sns.color_palette("crest", as_cmap=True)
p = sns.scatterplot(data = df_reduced, x = "PC1", y = "PC3", hue = clusters.labels_, legend = "full", palette =
sns.move_legend(p, "upper right", bbox_to_anchor = (1.17, 1.), title = 'Clusters')
plt.show()

plt.subplots(figsize=(17,10))
custom_palette = sns.color_palette("crest", as_cmap=True)
p = sns.scatterplot(data = df_reduced, x = "PC2", y = "PC3", hue = clusters.labels_, legend = "full", palette =
sns.move_legend(p, "upper right", bbox_to_anchor = (1.17, 1.), title = 'Clusters')
plt.show()
```

Finally, this code creates a 3-D scatter plot (PC1, PC2 and PC3) of the data using different colors to represent the different clusters previously identified, and adjusts the axis labels and the title of the plot:

In [ ]:
```
# Get cluster labels
clusters.labels_
Counter(clusters.labels_)

# Create figure and 3D axis
fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')

# Graph the data
p = ax.scatter(df_reduced['PC1'], df_reduced['PC2'], df_reduced['PC3'], c=clusters.labels_, cmap='crest')

# Adjust the title and axes
ax.set_title('Three-dimensional graphic with PC1, PC2 y PC3')
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')

# Show the figure
plt.show()
```

In [ ]:
```
# Get cluster labels
clusters.labels_
Counter(clusters.labels_)

# Create figure and 3D axis
fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')
```



```
# Graph the data
p = ax.scatter(df_reduced['PC1'], df_reduced['PC2'], df_reduced['PC3'], c=clusters.labels_, cmap='crest')

# Adjust the title and axes
ax.set_title('Three-dimensional graphic with PC1, PC2 y PC3')
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')

# Show the figure
plt.show()

def update(i):
    ax.view_init(elev=10., azim=i)

anim = FuncAnimation(fig, update, frames=np.arange(0, 360, 1), interval=25)
anim.save('my_animation.gif', writer='imagemagick')
plt.show()
```

Annex IV. GMM python code.

In [ ]:
```
%matplotlib inline

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import normalize
from sklearn.cluster import KMeans, DBSCAN
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from scipy.spatial import distance
from kneed import KneeLocator
from sklearn.neighbors import NearestNeighbors
from sklearn.neighbors import KNeighborsClassifier
from sklearn import preprocessing
from sklearn.metrics import silhouette_score
from sklearn import metrics
from collections import Counter
from scipy import stats
from matplotlib.animation import FuncAnimation
```

In [ ]:
```
dfUncoded = pd.read_excel('00X_FeededDataUncoded.xlsx')

df_standardized_plus = pd.read_excel('df_standardized.xlsx')
```

In [ ]:
```
df_standardized = df_standardized_plus#.drop('Category', axis=1)
df_standardized_plus.shape
```

In [ ]:
```
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to set the column named 'column_name' as the index
df_standardized = df_standardized.set_index('Customer_Code')
index = df_standardized.index
```

In [ ]:
```
# assuming your standardized data is stored in a DataFrame called 'df_standardized'
# and you want to increase the weight of the column named ['']
# Scale the columns of the data:
df_standardized['Sales_€_CC'] = MinMaxScaler(feature_range=(0, 5)).fit_transform(df_standardized[['Sales_€_CC']
df_standardized['Item_Code'] = MinMaxScaler(feature_range=(0, 150)).fit_transform(df_standardized[['Item_Code']
df_standardized['Margin_Total'] = MinMaxScaler(feature_range=(0, 55)).fit_transform(df_standardized[['Margin_To

# Select only the columns that contain the string "Category_" in their header & Scale the selected columns of t
scalerCategory = MinMaxScaler(feature_range=(0, 25))
category_columns = [col for col in df_standardized.columns if 'Category_' in col and 'Customer_Category' not in
df_standardized[category_columns] = scalerCategory.fit_transform(df_standardized[category_columns])

# Select only the columns that contain the string "Customer_Profile" in their header & Scale the selected colum
Customer_ProfileCategory = MinMaxScaler(feature_range=(0, 2))
Customer_Profile_columns = [col for col in df_standardized.columns if 'Customer_Profile' in col]
df_standardized[Customer_Profile_columns] = Customer_ProfileCategory.fit_transform(df_standardized[Customer_Pro

df_standardized.head()
```

In [ ]:
```
# assuming your standardized data is stored in a variable called 'df_standardized'
tsne = TSNE(n_components=2, perplexity=30, learning_rate=200)
df_reduced_TSNE = tsne.fit_transform(df_standardized)

# plot the results
plt.scatter(df_reduced_TSNE[:, 0], df_reduced_TSNE[:, 1])
plt.show()
```

In [ ]:
```
range_n_clusters = range(7,20)

meanValues_silhouette = []

for n_clusters in range_n_clusters:
    modelo_kmeans = KMeans(
                    n_clusters   = n_clusters,
                    n_init       = 20,
                    random_state = 123
                )
    cluster_labels = modelo_kmeans.fit_predict(df_reduced_TSNE)
    silhouette_avg = silhouette_score(df_reduced_TSNE, cluster_labels)
    meanValues_silhouette.append(silhouette_avg)

fig, ax = plt.subplots(1, 1, figsize=(15, 8))
ax.plot(range_n_clusters, meanValues_silhouette, marker='o')
ax.set_title("Average evolution of silhouette indexes")
ax.set_xlabel('Number of clusters')
ax.set_ylabel('Mean silhouette indexes');
```

In [ ]:
```
from sklearn.mixture import GaussianMixture
```

```
# Assuming df_reduced_TSNE is the 2D dataset obtained after applying t-SNE on df_standardized
gmm = GaussianMixture(n_components=14) # Change n_components to the desired number of clusters
gmm.fit(df_reduced_TSNE)
labels = gmm.predict(df_reduced_TSNE)

# Create a figure object with a width of 10 inches and a height of 5 inches
fig = plt.figure(figsize=(10, 6))

# Assuming labels is the array of cluster labels obtained from the previous step
plt.scatter(df_reduced_TSNE[:, 0], df_reduced_TSNE[:, 1], c=labels, cmap='viridis')
plt.show()
```

```
In [ ]: Final = pd.DataFrame()
        Final['Customer_Code'] = df_standardized.index
        Final['Cluster'] = labels

        Final = pd.merge(Final, df_standardized, left_on='Customer_Code', right_index=True, how='left')
        Final.head()
```

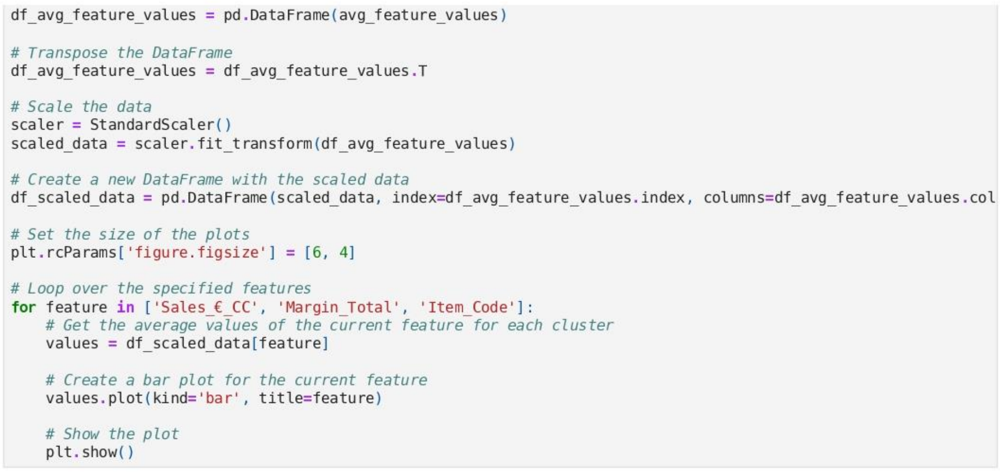
```
In [ ]: # create a new dataframe 'dfexport'
        dfexport = pd.DataFrame()

        # add columns 'Customer_Code' and 'Cluster' to 'dfexport' from the dataframe 'df'
        dfexport['Customer_Code'] = df_standardized.index
        dfexport['Cluster'] = labels

        # check if the column 'Cluster' exists in the dataframe 'dfUncoded'
        if 'Cluster' in dfUncoded.columns:
            # remove the column 'Cluster' from the dataframe 'df'
            dfUncoded = dfUncoded.drop('Cluster', axis=1)
            # merge 'dfexport' with 'dfUncoded' on the column 'Customer_Code'
            dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')
        else:
            # merge 'dfexport' with 'dfUncoded' on the column 'Customer_Code'
            dfexport = pd.merge(dfexport, dfUncoded, on='Customer_Code', how='left')

        dfexport.head(8)
```

```
In [ ]: dfexport.to_excel('00X_dfexportCluster_GaussMix2D.xlsx', index=False)
```

```
In [ ]: Final = Final.set_index('Customer_Code')
```

```
In [ ]: from sklearn.ensemble import RandomForestClassifier

        # Assume you have a DataFrame called 'Final'
        X = Final.drop('Cluster', axis=1)
        y = Final['Cluster']

        # Fit a Random Forest model
        rf = RandomForestClassifier()
        rf.fit(X, y)

        # Calculate feature importance
        feature_importance = rf.feature_importances_

        # Get the feature names from the column headings
        feature_names = X.columns.tolist()

        # Rank the features according to their importance
        feature_ranking = np.argsort(feature_importance)[::-1]

        # Print the ranked feature names and their importance scores
        for i in feature_ranking:
            print(f'{feature_names[i]}: {feature_importance[i]:.4f}')
```

```
In [ ]: from sklearn.preprocessing import StandardScaler

        # Get the unique cluster labels
        clusters = y.unique()

        # Create a dictionary to store the average feature values for each cluster
        avg_feature_values = {}

        # Loop over each cluster
        for cluster in clusters:
            # Get the data for the current cluster
            X_cluster = X[y == cluster]

            # Calculate the average feature values for the current cluster
            avg_values = X_cluster.mean()

            # Store the average feature values for the current cluster in the dictionary
            avg_feature_values[cluster] = avg_values

        # Create a DataFrame from the dictionary
```



```python
df_avg_feature_values = pd.DataFrame(avg_feature_values)

# Transpose the DataFrame
df_avg_feature_values = df_avg_feature_values.T

# Scale the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df_avg_feature_values)

# Create a new DataFrame with the scaled data
df_scaled_data = pd.DataFrame(scaled_data, index=df_avg_feature_values.index, columns=df_avg_feature_values.col

# Set the size of the plots
plt.rcParams['figure.figsize'] = [6, 4]

# Loop over the specified features
for feature in ['Sales_€_CC', 'Margin_Total', 'Item_Code']:
    # Get the average values of the current feature for each cluster
    values = df_scaled_data[feature]

    # Create a bar plot for the current feature
    values.plot(kind='bar', title=feature)

    # Show the plot
    plt.show()
```

Annex V. Environmental report.

summary[1]

1. Identification of the basic elements of the project

1.1 Raw materials

1.2 Assimilative capacity of the location

1.3 Process design phase

1.4 Construction phase

1.5 Phase of operation

1.6 Social and cultural aspects

1.7 Health aspects

1.8 Final waste

1.9 Future expansions

2. Pre-assessment of Environmental Impact

2.1 Factors related to the project

2.2 Factors related to location

2.3 Factors related to the environmental impact

2.4 General considerations

3. Summary and conclusions

Bibliography

[1]If appropriate, given the sensitivity of the environment in which the project must be developed, a detailed environmental inventory will be included as chapter 1.

1. IDENTIFICATION OF THE BASIC ELEMENTS OF THE PROJECT

1.1 Raw materials

question	YES	NO	May be	OBSERVATIONS
1. What raw materials will be used?		X		
2. How will these raw materials be obtained?		X		
3. In the system of sending (transport) the raw materials to the planned location, have the possible environmental impacts been taken into consideration?		X		
4. Is there a plan that links the project to the environmental aspects of extraction, transport and storage of raw materials?		X		

1.2 Assimilative capacity of the location

question	YES	NO	May be	Observations
1. Have alternative sites or locations been considered in an effort to avoid or mitigate environmental degradation?		X		
2. Are there hydrological, geological and meteorological studies of the location to anticipate and minimize possible damage to humans, flora and fauna?		X		
3. Will the wastewater be discharged directly or indirectly outside?		X		
4. What will be the receiving medium?		X		
5. Have studies been made of the physical, chemical and biological properties of the receiving aquatic environment, such as temperature, flow regime, dissolved oxygen, chemical oxygen demand?		X		
6. Will waste be generated? Is its characterization planned? Where do you plan to treat it, if it is generated?		X		

1.3 Design phase of the process

question	YES	NO	May be	Observations
1. What elements must be incorporated into the design of the plant from an environmental point of view?		X		It consists of the software development of the product.
2. Has the possibility of using a clean technology been considered, for the whole process, or for some of the operations involved?			X	Use renewable energy.

1.4. Construction phase

question	YES	NO	May be	Observations
1. Has the construction plan taken ecological factors into consideration?		X		No construction will take place.
2. Have actions been planned to minimize environmental damage, for the construction of roads, excavations, fillings, etc.?		X		They will not be realized

1.5. Operation Phase

question	YES	NO	May be	Observations
1. Are safety mechanisms in place for the handling of hazardous materials, if any?		X		Controlled atmosphere for the cooling gases of the refrigerator
2. Are there risks of explosion or accidental spills?		X		
3. Is an internal security plan planned, with the incorporation of all the necessary operational mechanisms?		X		
4. Have special measures been taken in hazardous material storage systems?		X		
5. Have appropriate precautions been taken to prevent losses from storage tanks?		X		
6. What types and amounts of waste streams will be produced?		X		
7. What pollution control systems are planned?	X			The use of clean energy supplied by solar panels onsite.
8. Are the planned discharges, if any, into aquatic systems (rivers, lakes, coastal waters) compatible with their present and future uses, particularly during dry periods?		X		
9. Can waste streams have synergistic effects with other materials?		X		
10. Do the waste streams contain potentially toxic materials?		X		
11. Are effects of wastewater discharges to the receiving environment to be expected, such as algal growth, fish kills, etc.?		X		
12. Is it planned to be monitored? Through specific, periodic or real-time measures?		X		
13. What systems are in place to remove toxic materials?		X		
14. If waste is produced, what treatment system do you intend to use?		X		
15. Has the recycling of this waste been considered?		X		No waste created by the project.

16. What forecasts are there to train the plant's staff in the environmental aspects of its management?		X		
17. How will odors be controlled?		X		

1.6 Social and cultural aspects

question	YES	NO	May be	Observations
1. How and to what degree does the presence and operation of the project result alter the environment of its location, and affect economic and social activities?	X			Improve effectiveness can result in creation of jobs.
2. Will urbanization problems be created or accentuated?		X		
3. Will there be an increase in traffic?		X		

1.7 Health aspect

question	YES	NO	May be	Observations
1. Will there be emissions that directly or indirectly affect health?		X		
2. What new health problems can arise?		X		
3. Can atmospheric or aquifer transport of pollutants affect health, at a local or regional level?		X		
4. What measures have been taken to ensure workers a safety and hygiene program?		X		

1.8 Final waste

question	YES	NO	May be	Observations
1. What management is planned to be done with the final waste?		X		

1.9 Future expansions

question	YES	NO	May be	Observations
1. How will future projects affect the environment?		X		

2. PRE-ASSESSMENT OF ENVIRONMENTAL IMPACT

2.1. Factors related to the project

Generalitats

question	YES	NO	May be	Observations
1. Will the project cause particularly complex effects on the environment?		X		
2. Will the project involve widespread soil disturbance, land clearing or clearing, leveling or large-scale underground works?		X		
3. Will the project mean significant alterations to the current or planned use of the land or urban planning?		X		
4. Will the project require the construction of auxiliary structures for the supply of water, energy and fuel?		X		
5. Can the project cause alterations to the water pipes?		X		
6. Can the project cause the need to modify the sewer network?		X		
7. Can the project cause changes to the drains in cases of intense rain?		X		
8. Can the project cause changes in the electricity transmission networks?		X		
9. Will the project require the construction of new roads or off-road tracks?		X		
10. Will the construction or operation of the project cause large volumes of traffic?		X		
11. Will the project mean explosives, or similar activities?		X		
12. Can the project cause an increase in demand for existing energy sources or a requirement for new energy sources?			X	There is a minimal use of power.
13. Will the project be closed or shut down after a limited lifetime?			X	Digital assets only affected.

atmospheric environment

question	YES	NO	May be	Observations
1. Will the project cause air emissions from fuel use, production processes, material handling, construction activities, or other sources?			X	Possible emissions depending on electricity source to power the computers
2. Will the project require the destruction of waste through open burning (for example, logging or construction waste)?		X		

aquatic environment

question	YES	NO	May be	Observations
1. Will the project require large quantities of water or the production of large volumes of wastewater or industrial effluents?		X		
2. Will the project mean a degradation of existing drainage patterns (including building dams or diverting watercourses or increasing flood risks?		X		
3. Will the project require the dredging of canals or the rectification of waterway crossings?		X		
4. Will the project require the construction of docks or levees?		X		
5. Will the project require the construction of offshore structures (breakwaters, oil platforms, etc.)?		X		

Waste production

question	YES	NO	May be	Observations
1. Can the project cause a large amount of inert waste?		X		
2. Can the project cause a large amount of toxic or special waste?		X		
3. Will the project require the evacuation of slag or waste from the mining process?		X		
4. Will the project require the evacuation of urban or industrial waste?		X		
5. Will the project facilitate the possibility of an increase in pollutants?		X		
6. Will the project contaminate the soil and groundwater?		X		

Noises, etc.

question	YES	NO	May be	Observations
1. Will the project cause sound emissions, vibrations, light, heat or other forms of radiation in the environment?			X	There'll be limited emission of heat from the computer.

risks

question	YES	NO	May be	Observations
1. Will the project violate toxic effluent standards?		X		
2. Will the implementation of the project require the storage, handling, use, production or transport of hazardous substances (flammable, explosive, toxic, radioactive, carcinogenic or mutagenic)?		X		
3. Will the operation of the project require the production of electromagnetic radiation or others that could affect human health or electronic equipment?			X	Computers always have some kind of electromagnetic emission.
4. Will the project require the regular use of pest and weed control chemicals?		X		
5. Will the project register an operational failure that renders the normal environmental protection measures insufficient?		X		
6. Can the project cause risks of exploitation or emission of dangerous substances (pesticides, chemical substances, radiation) because of an accident or anomaly?		X		
7. Can the project cause possible interference with an emergency or evacuation plan?		X		
8. Can the project cause possible decreases in job security?		X		

Social aspects

question	YES	NO	May be	Observations
1. Can the project cause a substantial reduction in the quality of the environment?		X		
2. Can the project cause the elimination of a singular element due to religion?		X		

.

3. Can the project cause any substantial adverse effect on human assets?		X		
4. Will the project involve jobs for many workers?		X		
5. Will the workforce have appropriate access to accommodation and other structures?		X		
6. Will the project involve significant costs in the local economy?		X		
7. Will the project cause changes in health conditions?		X		
8. Can the project cause changes in the location, distribution, density or growth rate of the population in the area?		X		
9. Will the project involve significant requirements in terms of installation of services?.	X			Minimal requirement of software to code, open source.
10. Can the project cause housing needs by generating new demand?		X		
11. Can the project cause any incidence or generation of new needs for public services in the area of protection against fire (firemen, ...)?		X		
12. Can the project cause any incidence or generation of new needs for public services in the area of the police?		X		
13. Can the project cause any incidence or generation of new needs for public services in the area of schools?		X		
14. Can the project cause any incidence or generation of new needs for public services in the area of parks or other recreational facilities?		X		
15. Can the project cause any incidence or generation of new needs for public services in the area of maintenance of public facilities including roads and streets?		X		
16. Can the project cause any incidence or generation of new needs for public services in the area of other government services?		X		

2.2 Factors related to location

Legal Protection

question	YES	NO	May be	Observations
1. Is the project located in areas designated or protected by the legislation of the Member State or close to them?		X		
2. Is the project located in an area where the environmental quality standards established by the legislation of the Member State are violated?		X		

General characteristics

question	YES	NO	May be	Observations
1. Is the project located in an area with unique natural features?		X		
2. Will the regeneration capacity of natural areas, such as coastal, mountainous and forest areas, be negatively affected by the project?		X		
3. Does the project area experience high levels of pollution or other environmental damage?			X	Industrialised area hosts the offices were the project takes place. Compliance with regulations is guaranteed.
4. Is the project located in an area whose soils and/or groundwater may have already been contaminated by previous uses?		X		

Hydrological data

question	YES	NO	May be	Observations
1. Is the project located on or near wetlands, watercourses or bodies of water?		X		
2. Is the project located near important sources of groundwater?		X		

Landscape and aesthetic characteristics

question	YES	NO	May be	Observations
1. Is the project located in an area of high quality and/or landscape sensitivity?		X		
2. Is the project located in an area visible to a significant number of people?		X		

Atmospheric conditions

question	YES	NO	May be	Observations
1. Is the project located in an area subject to adverse atmospheric conditions (temperature inversions, dense fog, strong wind)?		X		The office has temperature control.

Historical and cultural characteristics

question	YES	NO	May be	Observations
1. Is the project located in the vicinity of particularly important or valuable historical or cultural heritage?		X		

Stability

question	YES	NO	May be	Observations
1. Is the project located in an area prone to natural disasters or accidents caused by natural or man-made causes?		X		
2. The project is located in an area of steep topography that may be prone to landslides, erosion, etc. ?		X		
3. Is the project located in a coastal area, or close to it, prone to erosion?		X		
4. Is the project located in an area prone to earthquakes or seismic faults?		X		

Ecology

question	YES	NO	May be	Observations
1. Is the project located in the vicinity of particularly important or valuable habitats?		X		
2. Are there rare or endangered species in the area?		X		
3. Could the site prove resistant to natural or programmed re-vegetation?		X		

Land use

question	YES	NO	May be	Observations
1. Will the project conflict with existing urban planning or land use policy?		X		
2. Will the proposed land use conflict with neighboring land use (existing or proposed)?		X		

3. Is the project located in an area of high population density or in the vicinity of residential areas or other sensitive land use areas (eg: hospitals, schools, places of worship, public services)?		X		
4. Is the project located on land of high agricultural value?		X		
5. Is the project located in an area of recreational / tourist importance?		X		

2.3. factors related to environmental impact

Land and Properties

question	YES	NO	May be	Observations
1. Will the project cause significant land degradation or loss of use?		X		
2. Can the project cause changes in unstable soil conditions or geological substructures?		X		
3. Can the project cause breaks, displacements, compaction or discovery of the soil?		X		
4. Can the project cause changes in the topography or relief characteristics of the land surface?		X		
5. Can the project cause destruction, modification or covering of any geological singularity or physical feature?		X		
6. Will the project cause a general degradation of the land?		X		
7. Can the project cause soil pollution?		X		
8. Is there a risk of impact on the supporting infrastructure required by the project (wastewater disposal facility, roads, supply of electricity and water systems, schools)?		X		
9. Is there a risk of the project impacting the use of neighboring land?		X		
10. Is there a risk of impact of the surface facilities supporting the project on neighboring land uses?		X		
11. Is there a risk that underground works could cause disasters or accidents?		X		
12. Will the project cause the demolition of structures or the occupation of properties (houses, gardens, commercial establishments)?		X		

erosion

question	YES	NO	May be	Observations
1. Is the project likely to cause erosion?		X		
2. Will the adoption of erosion control measures lead to other adverse effects?		X		
3. Could the project cause any increase in soil erosion by wind or water both inside and outside the facility?		X		
4. Will the project cause dune erosion, or coastal encroachment or adverse changes in coastal systems?		X		
5. Can the project cause changes in the disposition of beach sands, modification of river and lake beds due to deposition, sedimentation or erosion and changes to the seabed and the coast?		X		

aquatic environment

question	YES	NO	May be	Observations
1. Will the project cause impacts on the quantity and/or quality of private or municipal water supplies?		X		
2. Will the use of water affect the availability of existing local supplies?		X		
3. Will the project negatively affect the quality, direction, flow or volume of surface or underground water due to sedimentation, hydrological alterations or discharges?		X		
4. Can the project lead to discharge into groundwater or surface water, or any alteration of surface or groundwater quality including temperature, dissolved oxygen, turbidity and all the usual parameters?		X		
5. Can the project cause changes in currents, in the course and direction of water movements, both fresh and marine?		X		
6. Will the project cause an increase in particulate matter?		X		
7. Can the project cause changes in the absorption rates, drainage models or in the rates of evacuation and surface emptying?		X		
8. Can the project cause alterations in the course or flow of floods and avenues?		X		
9. Will the project cause fluctuating water levels?		X		
10. Will the project cause changes in salinity gradients?		X		
11. Can the project cause changes in the amount of groundwater, either through direct additions or withdrawals, or through the interruption of any aquifers through cuttings or excavations?		X		

12. Will the natural alteration of the water course have a negative effect on natural habitats (eg, water flow velocity and fish farming) or other water uses (fishing, navigation, bathing)?		X		
13. Will the project cause an impact on the sustainability of both commercial and recreational fish farms?		X		
14. Will the project have an impact on all water-related recreational activities?		X		
15. Will the project cause significant alterations to wave action models, sediment movement or increased water circulation?		X		
16. Will the project limit the use of water for recreational, sport fishing, fishing, boating, research, conservation or scientific purposes?		X		
17. Will the project cause the possibility of an impact on the water according to the results of physical, chemical and biological tests?		X		
18. Will the project cause the possibility of impacts on the sediments according to the results of physical, chemical and biological tests?		X		
19. Will the project cause the possibility of impacts on downstream streams?		X		
20. Will the project cause an impact on wetland production values?		X		
21. Will the project cause an impact on the values for the protection of wetlands from natural disasters (floods, big storms...)?		X		
22. Will the project cause impact as a result of obstructive sedimentation?		X		
23. Will the project cause an impact on the separation and recycling of inorganic nutrients by the tides?		X		
24. Will the project cause an impact on the waters of the estuaries?		X		
25. Will the project cause an impact on the presence of unique wetlands or with unique geological characteristics?		X		
26. Can the project expose people or property to water risks such as floods, storms or underwater earthquakes?		X		

Air quality

question	YES	NO	May be	Observations
1. Can the project cause significant atmospheric emissions or deterioration of air quality?		X		
2. Can the emissions caused by the project negatively affect human health or well-being, fauna or flora, material or other resources?		X		
3. Can the emissions caused by the project negatively affect human health or well-being, fauna or flora, material or other resources?		X		
4. Can the project cause unpleasant odors?		X		
5. Can the project cause dust generation?		X		

Atmospheric conditions

question	YES	NO	May be	Observations
1. Can the project cause alteration of air movements, humidity or temperature or changes in both local and regional climate?		X		
2. Will the project cause alterations to the physical environment that could affect microclimatic conditions (turbulence, ice areas, increased humidity, etc.)?		X		
3. Can the project expose people or property to geological risks, such as earthquakes, landslides, mudslides, etc.?		X		

Noise, etc.

question	YES	NO	May be	Observations
1. Can the project cause an increase in existing noise levels?			X	Computer sound.
2. Can the project expose people to excessive noise?		X		
3. Can the project cause a considerable increase in light radiation or glare?		X		Minimal screen time when in use. Does not apply.
4. Will the project have repercussions on people, structures or other sensitive receptors / elements or noise, vibrations, light, heat or other forms of radiation?		X		

ecology

question	YES	NO	Maybe	Observations
1. Will the project cause a reduction in genetic diversity?		X		
2. Will the project cause physical loss of the substrate and its habitat?		X		
3. Will the project cause the loss or degradation of particularly valuable habitats, ecosystems or habitats of rare or endangered species (both flora and fauna)?		X		
4. Will the project cause impacts on the presence of rare or unique plants or animals at the site?		X		
5. Will the project cause impacts on the presence of plants or animals in the near boundaries of the territory?		X		
6. Can the project cause a decline in the fish or fauna population below the limits of self-sufficiency?		X		
7. Can the project lead to the introduction of new plant species in the area or barriers to the normal		X		

development of existing species?				
8. Can the project reduce the yield of any agricultural plantation?		X		
9. Will the project cause changes in the diversity of plant species, or the number of any plant species (including trees, shrubs, grasses, plantations or underwater plants)?		X		
10. Will the project cause impacts on the components of the aquatic food chain?		X		
11. Will the project cause the deterioration of reproduction and / or nutrition of aquatic species?		X		
12. Will the project cause impacts on mammals associated with aquatic ecosystems?		X		
13. Will the project cause impacts on fish associated with aquatic ecosystems?		X		
14. Will the project cause impacts on birds associated with aquatic ecosystems?		X		
15. Will the project cause impacts on reptiles associated with aquatic ecosystems?		X		
16. Will the project cause impacts on special aquatic locations (marines, in refuges or in marine sanctuaries)?		X		
17. Will the project cause impact on / or removal of wetlands?		X		
18. Will the project cause an impact on / or removal of sludge?		X		
19. Will the project cause impact on/or removal of vegetation in shallow waters?		X		
20. Will the project cause impact on / or removal of pond complexes and surface streams?		X		
21. Will the project cause the possibility of impacts on the benthos (flora and fauna found at the bottom of the lake or sea)?		X		
22. Will the project cause any degree of stress on biological community structures?		X		
23. Can the project cause changes in the diversity of animal species, or the number of some animal species (birds, mammals, reptiles, amphibians, fish, insects, crustaceans, molluscs or any other higher organisms)?		X		
24. Can the project cause the introduction of new species of animals in the area or barriers to the movement of migratory species?		X		
25. Will the project disturb or impair the ability of species to reproduce or negatively affect migration or feeding, rearing, breeding or resting areas or lead to significant obstacles to migration?		X		
26. Will the impacts in terms of noise, vibrations, light or heat caused by the project disturb the birds or other animals?		X		
27. Will the project disrupt ecological processes essential to biotic systems?		X		
28. Will the project lead to the introduction of noxious weeds, parasites or diseases, or aid the spread of known pathogens, harmful/exotic organisms or problem species?		X		
29. Will the project involve the large-scale use of pesticides, fertilizers or other chemical products that could generate waste in the terrestrial or aquatic environment?		X		
30. Will the project significantly increase fire risks?		X		

.

31. Will sedimentation resulting from the project cause adverse effects on aquatic life due to a decrease in available light?		X		

Landscape and aesthetic characteristics

question	YES	NO	May be	Observations
1. Will the project significantly affect a scenically attractive or historically or culturally important area?		X		
2. Will the project affect the landscape of the site, being in view of a significant number of people?		X		
3. Will the project cause an impact on aesthetics-presence of plants or animals with high visual quality?		X		
4. Will the project cause an impact on the aesthetics-presence of an associated body of water?		X		
5. Will the project cause an impact on the aesthetics-type of wetlands or topographic diversity?		X		
6. Can the project cause an obstruction due to the visibility of the landscape or will it be an unsightly view of the public?		X		

Traffic-related impacts

question	YES	NO	May be	Observations
1. Will the project lead to significant changes to traffic (road or otherwise), with the consequent effects for other users in terms of noise, air quality, comfort, etc., and impacts on other receptors?		X		
2. Will the accessibility changes resulting from the project lead to an increase in the development potential of the area?		X		
3. Can the project cause the generation of a substantial increase in the movement of vehicles?		X		
4. Can the project cause an increase in the number of parking spaces?		X		
5. Can the project cause a substantial impact on existing transport systems?		X		
6. Can the project cause an alteration to existing circulation patterns or movements of people and/or goods?		X		
7. Can the project cause changes in sea, air or rail traffic?		X		
8. Can the project cause an increase in traffic risks for motor vehicles, cyclists or passers-by?		X		

Social and health impacts

question	YES	NO	May be	Observations
1. Will the project significantly affect the labor or real estate market in the area?		X		
2. Will the project cause the physical division of an existing population?		X		
3. Will the project lead to a shortage of social infrastructure by having to deal with a temporary or permanent increase in population or economic activity?		X		
4. Will the project significantly affect the demographic characteristics of the area?		X		
5. Will the project cause an impact on educational or scientific qualities?		X		
6. Can the project expose the population to potential health risks?		X		
7. Can the project cause a decrease in the quality and/or quantity of possible recreational activities?		X		
8. Can the project cause an alteration or destruction of archaeological assets?		X		
9. Can the project cause physical or aesthetic discomfort for existing architectural monuments?		X		
10. Can the project cause a potential change to the physical environment that could affect ethnic cultural values?		X		
11. Can the project cause restrictions on religious and folkloric uses in its area of influence?		X		

others

question	YES	NO	May be	Observations
1. Will the effects be irreversible?		X		
2. Are the effects cumulative with those of other projects?		X		
3. Will the effects be synergistic?		X		
4. Is there a possibility of adverse side effects?		X		

2.4 General considerations

question	YES	NO	May be	Observations
1. Will the project cause public controversy? Can the project raise major concerns?		X		
2. Are there cross-border effects that need to be taken into account?		X		
3. Will the project lead to irreversible or unavoidable effects on future generations?			X	Use of AI tools.
4. Will the project conflict with existing international, national or local policy or legislation?		X		
5. Will the project require an alteration of the environmental policy in force?		X		
6. Is there legislation on pollution control, which guarantees due attention to the environmental impacts of the project?		X		
7. Will the project have an importance that exceeds the local level?	X			Might be applied to other regions.
8. Will the project involve eventual uncertain effects or that involve unique or unknown risks?			X	Can led to mis categorization of certain customers. Low risk.
9. Can the project cause any rejection by popular associations or organizations regarding the environmental effects of the project?		X		
10. Will the project provide structures that succeed in incentivizing further (induced) development, for example through the provision of service infrastructure (urbanisation, industrial development, transport requirements)?		X		
11. Will the project significantly need any resource whose supply may become scarce?		X		
12. Will the project have an impact on increased expenditures or revenues of the state, country, or local government (increased expenditures for support facilities or increased tax revenues)?	X			Improved efficiency might result in higher revenue.
13. Will the project have an economic impact - value of wetlands as a source of nutrients and/or habitat for aquatic life?		X		
14. Will the project have economic impacts - value as a recreational area?		X		
15. Will the project have economic impacts - value for flood control / flood prevention?		X		
16. Will the project have economic impacts - port maintenance costs?		X		
17. Will the project have an economic impact on the public (both public and private) of the facilities supporting the project?		X		
18. Will the project have an economic impact (both public and private) on the use of neighboring land?		X		
19. Are there one or more reasonably practicable project alternatives that meet the project's objectives with less adverse environmental impact?		X		

3. SUMMARY AND CONCLUSIONS[2]

The project is about the improvement of the performance of a consumer product, of a refrigerator, and of its manufacturing process and the assessment of the impact on business results.

In the detailed Environmental Impact Study, different aspects will have to be considered that have to do with the environmental factors impacted and the impactful actions, in our case, the energy source to power the computer, the use of machine learning within company's data and higher efficiency by the commercial team when managing customers.

The main actions and factors that will have to be taken into consideration in the detailed study are listed below, in table form.

Shocking actions[3]

	Impressive Actions	Observations
Construction or Execution Phase	Coding the program will require a computer.	The use of solar panels reduces the emissions when using electricity, the computer is on a renewal plan I order to maximise its lifespan and minimise the waste when it finishes, and finally a work from home is in place for 60% of the week to minimise commute emissions.
	Energy consumption when coding and computing test runs of the data.	
	Atmospheric pollution by vehicles when going to the office.	
Phase of Operation or Exploitation	Energy consumption when computing the runs of the data.	The use of solar panels reduces the emissions when using electricity

[2]The Environmental Report does not include the corrective actions necessary to minimize the negative effects of the project on the environment. In any case, if any effect is of great impact, a brief explanation of the possible prevention and/or correction actions must be included in this chapter of the report.

[3]List and include comments that are deemed appropriate.

	Atmospheric pollution by vehicles when going to the office.	and a work from home is in place for 60% of the week to minimise commute emissions.
Phase of Use[4]	Energy consumption when computing the runs of the data.	During the use of the product, it may cause discomfort to users in the environment

Impacted environmental factors

	Environmental factor	Impact on...
Natural environment	atmosphere	Apparently, there are no consideration impacts
	Ground	Office space might be required.
	water	Apparently, there are no consideration impacts
	flora	Apparently, there are no consideration impacts
	Fauna	Apparently, there are no consideration impacts
	perceptual environment	Views and landscape
Socioeconomic environment	Uses of the territory	AI legislation is being discussed and might affect the project. Very low risk.
	cultural	Apparently, there are no consideration impacts
	Infrastructure	Apparently, there are no consideration impacts
	humans	Apparently, there are no consideration impacts

[4]The checklist does not include direct questions regarding the use of the product or service resulting from the operation of the project. However, if applicable, the main impacting actions related, and the environmental factors impacted should be collected.

	Economy and population	There can be a difference in the purchase options of the customers, most probably for the good.

Bibliography

Internet addresses of interest

State agency, State Official Gazette. Royal Decree 552/2019 of September 27.

https://www.boe.es/diario_boe/txt.php?id=BOE-A-2019-15228

State agency, State Official Gazette. Royal Decree 115/2017 of February 17.

https://www.boe.es/buscar/doc.php?id=BOE-A-2017-1679

Generalitat de Catalunya (Environment Department). nd "Environment"

https://web.gencat.cat/ca/temes/mediambient/