ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Department of Computer Science and Engineering - DISI

Second Cycle Degree in Artificial Intelligence

A User-Centric Recommendation System for Anomaly Exploration in the Healthcare Domain

Dissertation in Artificial Intelligence in Industry

Supervisior:
Professor
Michele Lombardi

Co-Supervisor:
Dr.
Giorgia Morelli

Presented by:
Alessio Conti

Graduation Session March 2025

Academic Year 2023/2024

Abstract

In the era of data-driven decision-making, the detection and analysis of anomalies have become increasingly critical, particularly within the healthcare sector. In this context, enhancing efficiency, accuracy, and the early identification of irregularities is essential. Given the complexity, and growing volume of healthcare data, there is a pressing need for intelligent systems that not only identify anomalous patterns, but also support users in effectively navigating vast amounts of information.

This thesis investigates the integration of recommendation system frameworks within an outlier analysis platform on healthcare data, with the objective of improving user experience and streamlining investigative processes.

Three distinct recommendation approaches were developed and evaluated: (i) Rank Aggregation, designed to improve the prioritization of elements on the platform's homepage by refining ranking mechanisms; (ii) Similar Product Recommendation, which assists users in discovering related outliers through a clustering-based approach and a nearest-neighbor ranking algorithm; and (iii) User-Based Personalized Recommendation, leveraging user interaction history to generate tailored suggestions via an autoencoder-based embedding model and a binary classifier.

This work holds significant implications for the healthcare domain, where identifying anomalies in medical and administrative data can lead to better resource allocation, fraud detection, and improved patient outcomes. By integrating recommendation techniques, the system empowers users with more intuitive tools to explore and understand critical insights. The emphasis is placed on a user-centric approach, ensuring that recommendations align with individual search behaviors, preferences, and investigative needs.

Future work will focus on expanding the dataset with real user interactions, opti-

mizing candidate generation, and incorporating explicit user preferences to enhance recommendation quality. Additionally, A/B testing will be implemented to systematically assess user satisfaction and engagement, ensuring that recommendations align with real-world needs.

This research lays the foundation for intelligent, user-centered outlier detection in healthcare data, offering a valuable tool for professionals navigating complex datasets and improving decision-making efficiency.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The healthcare sector is undergoing rapid digitalization, leading to an exponential increase in available data. Data management in the healthcare sector represents a particularly complex challenge. This is consequent to the high volume of data in combination with their heterogeneity. Healthcare data span into a large plethora: critical patient information, operative clinical insights and administrative data; each with its own peculiarity and specific requisites for elaboration. The increase in volume of data and their heterogeneous characteristics delve the analysis process extremely challenging. Thus the adoption of specific software and tools is indeed an obligation to manage efficiently and effectively those information.

1.1.1 Enhancing Healthcare Data Management with AI-Driven Anomaly Detection

Traditional data management processes in the healthcare sector rely on manual workflows and fragmented systems. This have been proven inadequate in addressing the growing demands of the modern landscape. Manual data handling, in particular, increases the risk of human er-

ror, delays in processing and transmitting information, and ultimately compromises the timeliness of critical decision-making. The *Outliers Project (OP)* was conceived to tackle one of the key challenges in managing complex data: the accurate identification and interpretation of anomalies. In data-intensive environments such as healthcare, anomalies are not merely potential errors within operational processes; they can also indicate significant shifts in system behavior or emerging trends that require attention. The project takes advantage of Artificial Intelligence algorithms capable of intelligently detecting these anomalies, enabling proactive analysis and management. Besides identifying errors that could mine decision quality, OP also support users interpreting anomalies as valuable indicators for monitoring deviations from standard operational or clinical patterns. By transforming outliers into actionable insights, the *Outliers Project* enhances both the efficiency and accuracy of data-driven decision-making in healthcare.

1.1.2 The Outliers Project and Its Intentions

Anomalies in data can signal critical operational changes that require attention. For instance, a sudden drop in activity might indicate a data collection issue or an organizational shift. Without proper analysis, such changes risk being dismissed as mere recording errors rather than potential signs of necessary human intervention or resource reallocation The project focuses on three main objectives:

- Optimizing organizational workflows
- Enhancing efficiency and responsiveness in data management
- Improving decision quality

By promptly identifying irregularities in operational processes, managers gain a clearer, more comprehensive view of organizational processes. Detecting unexpected data shifts provides insights into the impact of strategic decisions, such as technology adoption or departmental

restructuring. For example, if a department experiences a sudden fluctuation in activity, the system alerts managers, enabling data-driven decisions to address the situation effectively. This proactive strategy reduces downtime, optimizes resource utilization, and minimizes manual workload. As a result, productivity improves while ensuring smoother and more efficient operations.

1.2 Problem Statement

This software is the result of a collaboration between the *Onit Group* and *USL Toscana Sud Est (TSE)*, designed to support the management of multiple healthcare facilities within the USL network. The primary objective is to develop a system capable of effectively assisting managers in overseeing diverse operational units simultaneously.

A key challenge that emerged in this context is the large number of outliers detected, even after applying selection and filtering processes. This issue stems from the system's broad field of view, which encompasses a wide range of facilities and operational parameters. Consequently, a critical aspect of system design is determining how to present and prioritize outliers within the software in a meaningful and efficient manner. Establishing a general ranking mechanism for outliers is inherently complex. The ranking must balance the organization's specific requirements - prioritizing elements deemed relevant - while avoiding excessive focus on narrow categories. The goal is to develop an ordering system that aligns with institutional priorities while preserving the system's ability to highlight diverse and novel insights.

Additionally, the ranking methodology must adhere to long-term design principles that ensure scalability and adaptability. The system is envisioned to extend beyond healthcare applications, potentially serving

industries such as agriculture and the agrifood sector. To facilitate this expansion, the ranking framework must be modular and composable, allowing for seamless integration of new criteria without restricting flexibility.

Beyond static ranking strategies, the system could greatly benefit from personalized recommendation mechanisms. By leveraging user interaction data, the system could intelligently guide users in exploring outliers based on their past engagements, identifying relevant anomalies with greater precision. In this advanced scenario, the system should not only surface outliers similar to those previously examined but also introduce novel elements with shared characteristics, fostering a level of serendipity in the discovery process. This approach aims to enhance analysis by revealing potential issues or irregularities that users may not have otherwise encountered, thereby improving decision-making and operational oversight.

1.3 Objectives and Contributions

The primary objective of this study is to develop a suite of tools designed to enhance user-driven data exploration within the software. By addressing existing limitations and introducing intelligent recommendation mechanisms, the system aims to improve the identification and prioritization of relevant outliers.

The key contributions of this work are as follows:

- Enhancing the Ranking System: The study seeks to overcome the constraints of the current ranking approach, which relies on Euclidean distances, by implementing a more flexible and adaptive ranking mechanism.
- Introducing Advanced Recommendation Systems: Two distinct rec-

ommendation systems are integrated to assist users in discovering novel and potentially significant outliers. These systems serve as navigational aids, guiding users toward unexplored yet relevant anomalies.

To achieve these objectives, the following aspects were investigated and developed, each one tailored to address specific user needs within the software:

Ranking system The ranking approach employs metasearch rank aggregation techniques, combining multiple ranking signals to generate a unified and optimized ordering of outliers. This method integrates diverse ranking criteria, ensuring a balanced trade-off between organizational priorities and the need to discover novel insights.

Similar Product Recommender Also referred to as the correlated product recommender, this system suggests outliers that share characteristics with the one currently under analysis. Its function is analogous to recommendation features found in e-commerce platforms, where similar products are displayed during the checkout process. Soft clustering techniques on top of UMAP dimensionality reduction are introduced to discover similarities among outliers' characteristics.

Personalized Recommender This more advanced system generates recommendations based on a user's past interactions with the platform. By analyzing prior engagements, it identifies and suggests outliers that align with the user's interests, much like the "Recommended for You" section in streaming services. A deep learning based method has been crafted to specifically embed interaction histories into low dimensional vectors, which are then fed on to a classifier for final recommendations.

Figures 1.1 and 1.2 respectively, show mock-ups of the "Outlier Project" software with the additional contributions. Figure 1.1 shows the main

outlier dashboard that presents items given the computed ranking, on the left, while adding a section on the right for personalized outliers' recommendations based on user interest and past interactions. Figure 1.2 refers to the inspector page: selecting an item from the dashboard list, it can be further analyzed through graphs and detailed descriptions. On the right side of the page, an additional section proposes similar items that could be of interest given the current analysis.

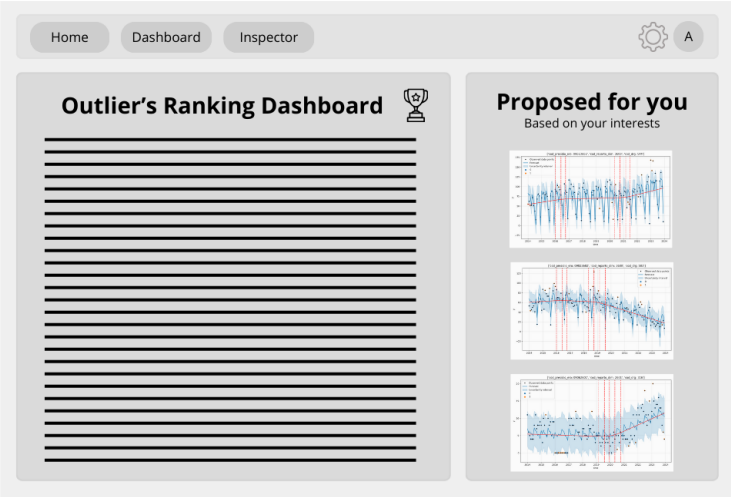

Figure 1.1: Dashboard mock-up of the software, focusing on the Ranking (left), and Personalized Recommender (right)

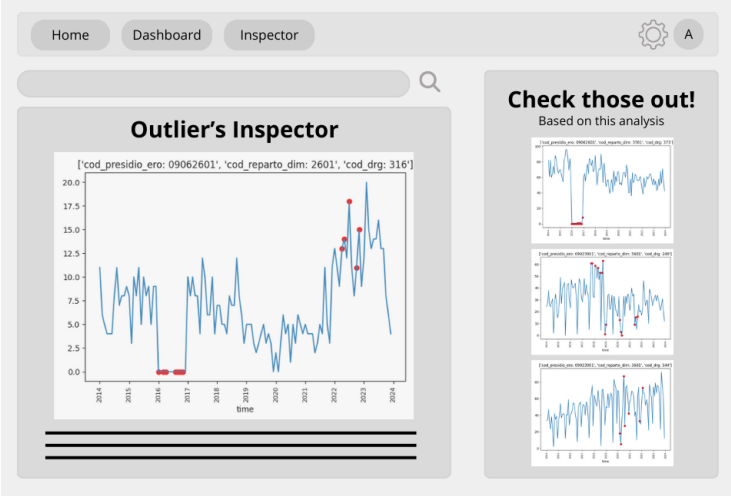

Figure 1.2: Inspector page mock-up of the software, focusing on the Similar Product Recommender (right)

1.4 Thesis Outline

The structure of this thesis follows a division based on the key arguments explored, reflecting the distinct conceptual areas under investigation.
Chapter 2 provides a comprehensive review of existing methodologies in recommendation systems, along with a technical overview of metasearch approaches relevant to ranking.
Chapters 3, 4 and *5* detail the research conducted in three primary areas: ranking systems, similar product recommendation, and personalized recommendation. Each chapter focuses on the methodology, model design, experimental setup, and results obtained.
Chapter 5 also includes a dedicated section on the creation of synthetic datasets.
Chapter 6 discusses the key contributions of this work, addressing its limitations and the expected impact on user experience. The thesis concludes with proposals for future directions, emphasizing strategies to explicitly evaluate user satisfaction and system effectiveness.

Chapter 2

Background

2.1 Outlier Project

The *Outlier Project* employs advanced anomaly detection algorithms to identify and classify irregularities in healthcare data. These anomalies fall into three main categories:

Point Anomalies A point anomaly occurs when a single data point significantly deviates from the expected range within a dataset. These anomalies are relatively easy to detect as they stand out distinctly from the normal data distribution.
Example: A hospital ward that typically registers 2,000 patient admissions per day suddenly reports 20,000 admissions on a single day. This could indicate a data entry error, a system malfunction, or an exceptional event such as a large-scale emergency, necessitating further investigation.

Contextual Anomalies A contextual anomaly arises when a data point appears unusual within a specific temporal, spatial, or categorical context but may seem normal otherwise. These anomalies often occur in time-series data, where seasonality and trends influence expected values.
Example: Flu vaccination rates generally peak during autumn and win-

ter, with lower numbers recorded in summer. If an unexpectedly high number of flu vaccinations is reported in August, this anomaly could indicate a data entry error or an unusual public health trend requiring further analysis.

Collective Anomalies A collective anomaly occurs when a group of data points collectively exhibits an unexpected pattern that deviates from the norm, even though individual data points may not appear anomalous on their own. *Example*: In medical billing, there is usually a correlation between gross service charges and patient co-payments. If a subset of procedures displays inconsistent pricing - such as identical treatments being billed at significantly different rates without justification - this may indicate incorrect pricing, billing fraud, or administrative errors, all requiring further investigation.

To effectively detect these anomalies, the *Outlier Project* employs a combination of specialized algorithms tailored to different anomaly types:

- Point Anomalies: *Isolation Forest* [2] isolates anomalies by recursively partitioning the dataset. This approach is particularly well-suited for high-dimensional healthcare data, where outliers exhibit distinct separability.
- Contextual Anomalies: *Prophet* [3], *STL decomposition* [4], analyze seasonal trends and fluctuations. These methods effectively identify deviations from expected time-series patterns, ensuring the detection of anomalies influenced by temporal or contextual variations.
- Collective Anomalies: *Correlation Distance* [5] detects inconsistencies within groups of correlated data points. This technique is particularly valuable for identifying irregularities in structured data, such as medical billing discrepancies.

2.2 Metasearch

Metasearch, or metasearch aggregation, refers to the process of combining results from multiple search engines or ranking systems into a single, unified output. This technique is widely applied in domains where heterogeneous sources provide overlapping but distinct rankings or scores. Common applications include web search engines, travel fare aggregators and job search platforms. In cases where no ground truth ranking is available, metasearch techniques provide a structured way to integrate multiple independent rankings into a coherent decision-making framework. [6][7]
Metasearch is rooted in the idea that no single ranking function or source can fully capture the complexity of relevance, importance, or user preference. Instead, different ranking sources contribute complementary information. The challenge lies in determining how to aggregate these rankings to produce a meaningful final ordering.

Metasearch techniques can be broadly categorized into two main approaches: *rank-based fusion* and *score-based fusion.*

Rank-based fusion methods determine the position of an item in the final ranking based on how many independent rankers "vote" for that item to appear at a specific position. This approach draws an analogy to social voting systems, where different rankers contribute to a collective decision. Algorithms such as *Borda Count*, *Bayes Fuse* and *Condorcet* fall into this particular category [7][8].

On the other hand, score-based fusion methods aggregate rankings based on numerical similarity scores assigned to each item. Instead of relying on ordinal positions, these methods combine the similarity values of items retrieved from different ranking models. Belong to this category *CombMIN*, *CombMAX* and *CombSUM* [9].

Regardless of the chosen method, metasearch must address several challenges, including rank bias, information redundancy, and conflicts between different ranking sources. The presence of incoherent rank-

ings, where one criterion ranks an element drastically differently from others, further complicates aggregation. [6] By aggregating rankings systematically, metasearch ensures that an element's final position reflects a compromise between different sources, overcoming the problem of skewed rankings.

2.3 Recommendation systems

A recommendation system is a software tool designed to help users discover relevant content within a large dataset. Typically functioning alongside search or filtering mechanisms. Its primary role is to surface items that users may not have explicitly searched for but could still find valuable.

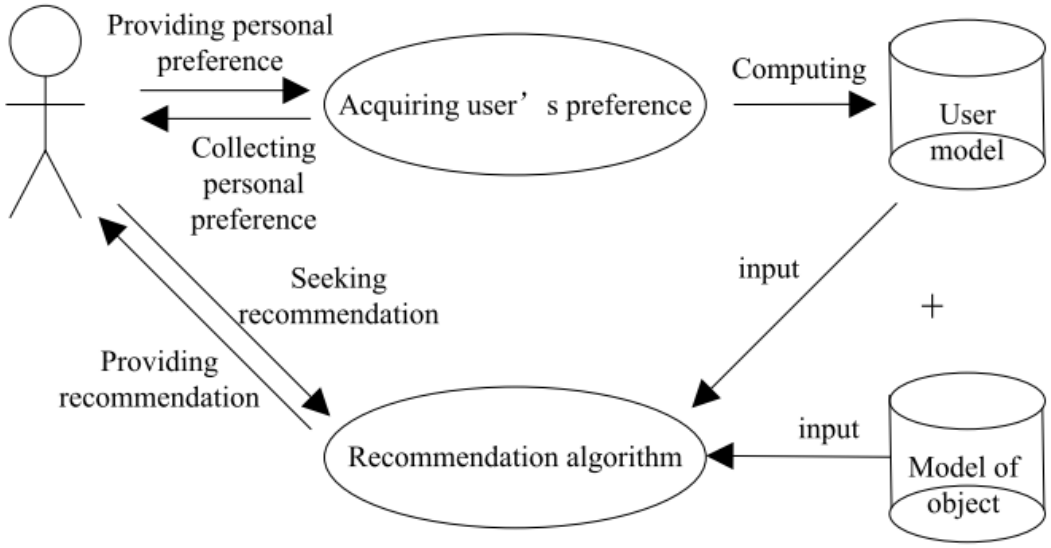

Figure 2.1: A generalized model for recommendation systems. Source: [1].

Two main types of recommendation systems are widely used: *collaborative filtering (CF)* and *content-based filtering (CB)*. These approaches differ in how they identify relevant items.
Collaborative filtering relies on identifying users with similar interests to the target user and inferring recommendations based on their pref-

erences. The underlying assumption is that if two users share similar tastes, items preferred by one may also be of interest to the other. [1] Content-based filtering, on the other hand, focuses on item similarities. Instead of analyzing user behavior, it recommends items with characteristics similar to those the user has already shown interest in. [10][11] Hybrid approaches then leverage aspects of both methods. Typically deep learning approaches uses to embed user interests or MLP models to predict interests toward specific items. [12][13]

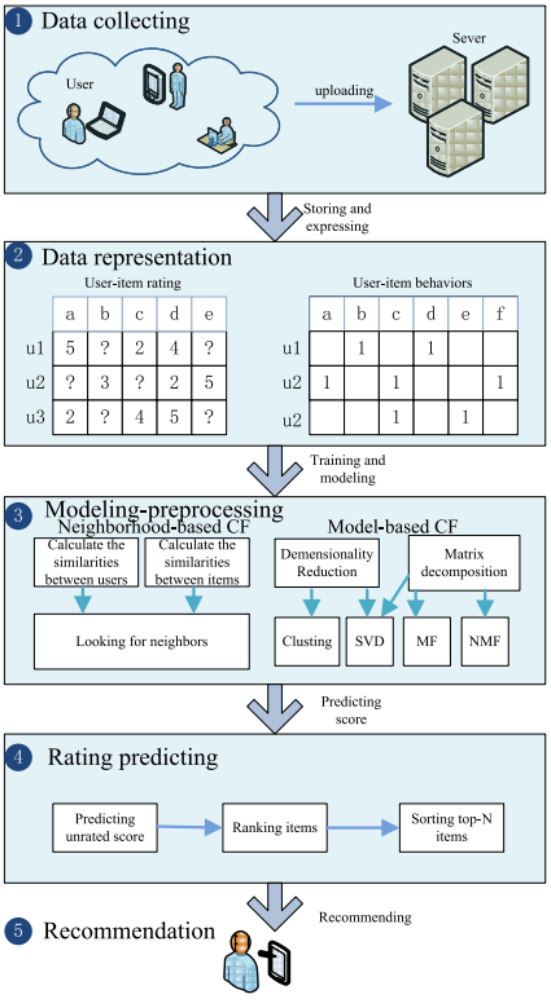

Figure 2.2: Framework for content recommender, explicating differences between content based filtering and collaborative filtering. Source: [1].

A recommendation system typically follows a structured pipeline composed of three main stages: candidate generation, scoring, and re-ranking. These steps work together to identify and prioritize the most relevant items for the user.

The process begins with the candidate generation phase. The system starts with a vast dataset and reduces it to a much smaller subset of potential recommendations. This step is crucial because evaluating every item in a large-scale corpus would be computationally infeasible. Instead, the system quickly filters out irrelevant items, retaining only a manageable set of promising candidates. [1]

Once this initial selection is made, the scoring phase refines the ranking by assigning relevance scores to the candidates. Unlike the previous step, which focuses on

reducing the dataset size, scoring is concerned with identifying the best options from the remaining candidates. Since this phase deals with far fewer items, more sophisticated models can be employed, leveraging a richer set of features such as user preferences, historical interactions, and contextual information. The goal is to assign meaningful scores that reflect the likelihood of user interest in each item.

Finally, the re-ranking phase further optimizes the selection by incorporating additional constraints and business logic. This step tailors the ranking to specific needs by considering factors beyond basic relevance, such as content freshness, diversity, and user preferences. For instance, the model might demote items the user has explicitly disliked, boost newer content to prevent outdated recommendations, or remove inappropriate suggestions. Thus the final ranked list is the consequence of multiple refinement stages, ensuring that the recommendations are not only relevant but also contextually appropriate and engaging for the user.

2.4 Dataset

The dataset used for this exploration consists of actual outliers extracted using the Outlier Project software. Specifically, it comprises ten months of extractions, spanning from January to October 2024, and includes a total of *25300 items*.
These alerts have already undergone preliminary filtering, ensuring that only the most relevant anomalies requiring client attention are retained. Each outlier is characterized by approximately thirty distinct features that define the domain of interest and provide contextual information. Among these features, some are specifically designed to assess the severity of the anomaly, including numerical scores that quantify its gravity. To gain a deeper understanding of the dataset, a brief investigation of the main features involved in the analysis is presented.

The healthcare domain is broadly categorized into two main **service types**: *SPA (outpatient services)* and *SDO (hospital services)*. SPA events, depending on the services provider, are further classified into *SSN public national healthcare services* and *PAG private healthcare services*.

Each alert in the system is characterized by one or more **dimensions**, which define the attributes of the analysis and provide essential contextual information. Dimensions play a crucial role in interpreting **measures**, representing the specific quantitative values under analysis. A measure alone, without its corresponding dimensions, lacks context and does not allow meaningful interpretability.

For instance, a common measure might be the number of patient accesses within a given time frame, such as a month. However, this value alone lacks meaning unless paired with relevant dimensions. By introducing *facility code* as a dimension, the measure can be contextualized: e.g., the monthly number of accesses at *Hospital N-001*. Moreover, dimensions can be combined to create a more granular field of analysis. For example, using both facility code and department code allows for a more refined comparison. This could results in evaluating monthly accesses to the pneumology department across multiple hospitals within a region.
Some examples of measures and dimensions can be found in Table 2.1:

measures	num_access, num_hospitalisations, gross_amount, net_amount, postsurgery_day, presurgery_day, waiting_time
dimensions	cod_facility, cod_hospital, cod_department, doc, departure_modality, prescription, cod_analysis, exemption, patient, cod_citizenship

Table 2.1: Examples of measures and dimensions analyzed by the software.

Each outlier record includes not only the observed measure value but also the **expected value** predicted by the anomaly detection algorithm. This comparison is crucial in assessing the severity of the alert.

Additional metadata is recorded, including the **timestamp of the analysis** and the **time span of the data** used. Each record also specifies key details about the **type of analysis** performed: whether it is point-based, collective, or contextual, along with the specific **algorithm** applied. Furthermore, any algorithm-specific parameters are stored, such as time-series **window granularity**, **aggregation type**, **seasonality**, and the **number of analyzed elements**. To quantify the severity of an outlier, two computed values are provided: **normalized difference** and **relevance score**. Normalized difference represents the deviation between the actual and expected values. To ensure comparability, it undergoes a two-step normalization process: first, within the batch of alerts detected for the same measure and dimension; then, across different types of outliers. Relevance score further refines this assessment incorporating both the severity of the deviation (as captured by the normalized difference) and the seasonal impact of the period under consideration, ensuring a more context-aware evaluation of outliers.

	analysis_type	event_type	event_subset	dimensions_keys	dimensions_vals	measures	values	reference_values	outlier_score	num_events	relevance_score	diff_norm	time
1	contestuali	SPA	SSN	cod_fonte	9LIS	num_accessi	1575.0	177656.0	40.23	1575.0	100.00	100.0	2024-10-01
2	contestuali	SPA	SSN	cod_struttura_ero | cod_azienda_res	34111A | 190203	imp_lordo	1826.0	7.45	89.16	1.0	100.00	100.0	2024-10-01
3	contestuali	SPA	PAG	cod_prestazione | cod_uop_ero	89.01 | 10805680310L008	tempo_attesa_1disp	341.0	0.0	73.33	1.0	100.00	100.0	2024-10-01
4	contestuali	SPA	PAG	cod_tipo_contatto	0	num_accessi	4299.0	13119.5	40.26	4299.0	100.00	100.0	2024-10-01
5	contestuali	SPA	SSN	cod_struttura_ero | cod_priorita	41261A | D	num_accessi	36052.0	804.98	39.83	36052.0	17.14	95.43	2024-10-01
6	contestuali	SPA	PAG	cod_modalita_acc | cod_tipo_contatto	11 | 2	num_accessi	13584.0	4706.37	34.08	13584.0	13.51	94.03	2024-10-01
7	contestuali	SPA	SSN	cod_priorita	D	num_accessi	575544.0	412853.78	47.03	575544.0	10.74	92.39	2024-10-01
8	contestuali	SPA	PAG	cod_branca | cod_priorita	F | P	tempo_attesa_1disp	344.0	57.73	97.29	1.0	4.927	83.94	2024-10-01
9	contestuali	SDO	-	cod_presidio_ero	09023701	num_ricovero	0.0	246.16	52.45	0.0	4.452	78.26	2024-10-01
10	puntuali	SPA	SSN	cod_ambito | cod_esenzione | cod_speci...	090107 | 000000 | 519	tempo_attesa_1disp	3648.0	1.11	28.57	NaN	0.819	45.37	2024-10-23
11	puntuali	SPA	SSN	cod_ambito | cod_branca | cod_esenzione	090107 | K | 000000	tempo_attesa_1disp	3648.0	1.12	28.54	NaN	0.819	45.37	2024-10-23
12	puntuali	SDO	-	cod_modalita_dim | cod_regime_ric	02 | 2	gg_postoperatorie	217.0	0.45	21.99	NaN	1.665	59.32	2024-10-04
13	contestuali	SPA	SSN	cod_struttura_ero | cod_azienda_res	34111A | 070102	imp_lordo	1316.0	26.65	88.98	3.0	2.359	70.89	2024-10-01
14	contestuali	SPA	PAG	cod_prestazione | cod_uop_ero	89.01 | 10803220406L008	tempo_attesa_1disp	238.0	0.0	73.33	2.0	2.240	69.79	2024-10-01
15	contestuali	SPA	SSN	cod_struttura_ero	41241A	num_accessi	155986.0	276926.98	32.88	155986.0	2.129	68.68	2024-10-01
16	contestuali	SDO	-	cod_reparto_dim | cod_presidio_ero	3601 | 09023701	num_ricovero	0.0	214.12	54.64	0.0	2.494	68.06	2024-10-01
17	contestuali	SPA	SSN	cod_fonte	3CUP	num_accessi	473658.0	354729	24.38	473658.0	2.022	67.54	2024-10-01
18	puntuali	SDO	-	cod_regime_ric | cod_tipo_ric	2 | 5	gg_postoperatorie	171.0	0.46	16.54	NaN	0.979	46.66	2024-10-10
19	puntuali	SDO	-	cod_azienda_res | cod_regime_ric	090203 | 2	gg_postoperatorie	171.0	0.46	17.08	NaN	0.979	46.66	2024-10-10
20	puntuali	SDO	-	cod_azienda_res | cod_modalita_ero | c...	090203 | 0 | 2	gg_postoperatorie	171.0	0.46	17.06	NaN	0.979	46.66	2024-10-10

Figure 2.3: Dataset example, only the most relevant features are shown.

Chapter 3

Ranking System

This chapter aims to describe the proposed solution to enhance the ranking methodology used in the Outlier Project by addressing the limitations of the current system.

3.1 Distance Based Ranking

At present, ranking is based on distance metrics, specifically *Euclidean distance*, calculated between each detected outlier and a prototyped reference element, using only a predefined subset of features. This method was initially implemented to align with client preferences, allowing rankings to prioritize specific aspects of an outlier.

The current process involves generating a *mock outlier* that serves as a reference point. This prototype is constructed with values reflecting the client's priorities. For example, if the client is primarily interested in recent outliers with a significant deviation between the actual and expected values, the mock element would be assigned a recent timestamp (e.g., the current date) and a maximized difference value (e.g., +inf). By computing distances from this reference, the system effectively ranks outliers based on their similarity to the client's ideal anomaly.

Euclidean distance is a distance metric typically used to measure the distance between two points in an n-dimensional space.

Given two points $A = (a_1, a_2, ..., a_n)$ and $B = (b_1, b_2, ..., b_n)$ is defined as:

$$d(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

The choice of this metric is consequent to its computational efficiency and straightforward geometric interpretation. In this context of ranking, Euclidean distance is leveraged to quantify the similarity between detected anomalies and a reference prototype.
By calculating the Euclidean distance between each detected outlier and this reference point, the system ranks anomalies based on their proximity to the prototype. The closer an outlier is to the mock reference, the higher it is ranked in relevance to the client's priorities.

This approach assumes all selected features contribute equally to the ranking. To mitigate this a weighted version of Euclidean distance is introduced allowing for differential feature importance.
This allows to shift the focus of ranking toward the characteristics that are considered to have greater impact. Thus the formulation shifts to:

$$dw(A, B) = \sqrt{\sum_{i=1}^{n} w_i (a_i - b_i)^2}$$

where w_i is the weight associated to feature i.

While this method yields meaningful rankings in straightforward cases, it exhibits significant limitations when applied to more complex ranking criteria. As the number and complexity of relevant features increase, manually constructing an appropriate mock outlier becomes increasingly challenging. Certain features may require non-trivial computations to generate representative values. Inaccurately hypothesizing these values could introduce bias or lead to misleading rankings. Consequently, emerged the need for a more flexible and scalable ranking mechanism that can dynamically adapt to diverse client preferences without relying

on predefined reference points.

3.2 Rank Aggregation

This section explores the core principles of metasearch and its application to multi-criteria decision-making, specifically in the context of ordering elements where multiple competing factors must be balanced. A key challenge in this task is the absence of user feedback or labelled data to validate the rankings, making it necessary to design a system that performs effectively without explicit supervision.

The requirements for the development of such a ranking system aligns with those previously established for distance-based ranking. It must be modular, allowing customization to accommodate specific client preferences. It should ensure a coherent ordering that respects the relative importance of different characteristics. Finally, it must overcome the limitations of the prior approach, particularly by eliminating the need for a prototyped reference outlier and improving the overall consistency of results.

Metasearch techniques have been investigated as a solution to address these challenges. Metasearch provides a structured method for integrating multiple rankings without relying on user feedback or predefined ground truth labels. The goal is to establish an optimal ordering of elements, ensuring that rankings reflect an effective balance of all relevant attributes, even when those attributes may be in conflict. [7]

Metasearch techniques are typically employed in contexts where a unified ranking must be derived from multiple independent sources, such as aggregating search engine results or consolidating rankings in a democratic voting system. [7][14] However, in this scenario, there are no inherently separate rankers, but only a single ranking process that can be systematically decomposed. The key distinction lies in the fact that, unlike subjective settings where rankings emerge from indistinguishable

combinations of individual preferences (such as in democratic consensus voting), here the ranking criteria are explicitly defined and can be independently assessed. This allows the single ranking mechanism to be decomposed into multiple independent rankers, each corresponding to a distinct feature of interest (e.g., relevance, popularity or recency). Each of these features functions as a dedicated ranker, producing an individual ranking based solely on that criterion. The process of generating these rankings is therefore reduced to a straightforward sorting operation for each relevant feature, directly aligning with client-defined priorities.
For instance, if the ranking must prioritize both *recency* and *high discrepancy*, two rankers are independently evaluated: one orders elements based on detection date, ensuring more recent outliers are prioritized, while the other ranks them by descending discrepancy values. This results in two distinct rankings, each emphasizing a specific characteristic relevant to the user. The challenge then becomes effectively merging these rankings into a unified ordering that balances all prioritized factors, ensuring that no single criterion disproportionately dominates the final ranking.

3.2.1 Methodology

The proposed methodology combines **rank-based** concepts and **score-based** aggregation techniques to refine the ranking process. The result of k singular ranking is therefore k ordered list of items indicating the position each ranker assigns to each outlier.

Since only the *rank positions* of the items are available, a purely rank-based aggregation approach might seem like an intuitive choice. However some criticalities arises. Majority voting works under the assumption that if most rankers agree on the relative ranking of two items, their aggregated ranking should reflect this consensus. This principle is effective when all rankers evaluate items based on the same underly-

ing criteria, leading to natural alignment in their preferences. However, when rankers prioritize different factors, such as recency, discrepancy, or severity, their rankings may diverge significantly. This could introduce inconsistencies that majority voting algorithms fail to resolve. In such cases, the absence of a shared objective introduces noise into the ranking process, making it difficult to derive a coherent final ordering using purely rank-based methods. [15][16]

To address this, a score-based approach is adopted. Instead of directly merging rank positions, the rank of each item in the individual orderings is translated into a score. This trick is inspired by the work of Lee [17], who leveraged ranks instead of similarity scores in combining multiple sources of evidence for document retrieval. By applying this transformation, the model can utilize score-based aggregation techniques, even in the absence of explicit scores at fusion time.

To achieve this, each ranking position is mapped to a normalized score within the range $[0, 1]$ using the following function:

$$S(rank_i) = 1 - \frac{rank_i - 1}{number\_of\_items}$$

In any ranking method, two primary types of errors can occur: assigning a high rank to a non-relevant element or assigning a low rank to a relevant one [9]. However, in our particular scenario those two aspects assume minority importance due to the simplified nature of the rankers; each of which assigns ordering over only one single criteria. It has been shown that different ranking paradigms will perform differently on the same set of data, often with little overlap in the set of results [17]. This phenomenon is particularly evident in our setting, where one ranker might assign a high rank to an item while another assigns it a much lower rank due to the difference of objctives among rankers. This discrepancy, commonly observed in information retrieval, suggests that a fusion method incorporating multiple ranking perspectives can help mitigate inconsistencies and reduce the likelihood of suboptimal rank-

ings.

3.2.2 Combination Algorithms

Combination methods [17][9][15] work on explicit relevance scores provided by experts and include various linear combinations like *CombSUM*, *CombMNZ* and *CombANZ*. Among non-linear methods, *CombMIN* and *CombMAX* respectively rank the items on the minimum and maximum scores received across the lists provided by experts. A comparative analysis is therefore carried on between CombSUM, CombMIN and CombMAX.

The rationale behind the CombMIN combination method is to minimize the probability of ranking a non-relevant element too highly.

$$CombMIN = min(score_i)$$

It contrasts with its counterpart, CombMAX, which aims to prevent relevant elements from being ranked too low.

$$CombMAX = max(score_i)$$

However, both methods have inherent flaws: each is specialized to handle a particular type of ranking error while neglecting the opposing issue. For instance, CombMIN prioritizes caution by ranking an element conservatively based on its lowest score, but this can lead to relevant items being unfairly demoted. Conversely, CombMAX promotes items based on their highest score, which risks elevating non-relevant elements [9].

To address this duality, the CombSUM combination method takes a more balanced approach. Instead of selecting a single score from each ranking, it aggregates all available scores by simply summing them [9]. This straightforward technique ensures that an element's final ranking reflects contributions from all individual rankers, reducing the likelihood

of extreme misclassifications and providing a more stable overall ranking.

$$CombSUM = \sum_{i=0}^{k} score_i$$

Figure 3.1 shows the three tested algorithms, explaining how they address the aggregation process with a sample dataset. As the exam-

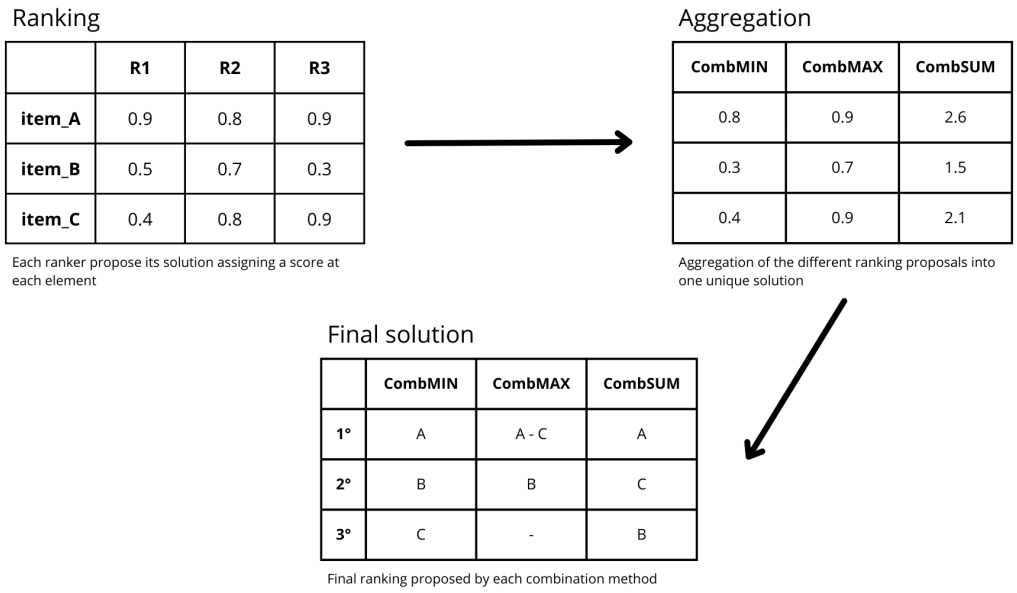

Figure 3.1: Exemplification of different aggregation methods

ple indicates the three methodologies act very differently and the obtained results are clearly diverging. Despite the existence of elements exhibiting complete correspondence between the three algorithms *(A)*, or partial correspondence *(B)*, there are instances *(C)* where the models demonstrate a complete divergence placing the elements in all different positions.

3.2.3 Metrics

The challenge in defining meaningful evaluation metrics for this ranking problem stems from the absence of ground truth labels. Without a correct ordering to compare against, it becomes infeasible to assess the

absolute correctness of the ranking beyond qualitative domain knowledge. In such unsupervised scenarios, a common approach is to establish key performance indicators (KPIs) that reflect the expected behaviour of the model. For this particular case, an essential requirement is that the final ranking adequately integrates all individual criteria. One useful KPI to measure this is the coverage of elements. This describes how well the final aggregated ranking preserves the presence of items that were ranked highly by each individual criterion. A practical way to quantify this is through **recall@k**, which measures how many relevant items from the original individual rankings appear within the top-k positions of the final aggregated ranking. This ensures that no single ranking criterion dominates at the expense of others and that the final list remains representative of all relevant aspects.

$$recall@k = \frac{Number\_of\_relevant\_items}{Total\_number\_of\_relevant\_items}$$

In this particular setting, identifying relevant items requires careful consideration. Each individual ranker prioritizes different aspects of the data, meaning that the top-ranked items will vary significantly across rankers. Given this variability, a useful approach is to evaluate the recall of the aggregated ranking with respect to each individual ranker's proposed ordering. Instead of selecting a fixed k (i.e., a predefined number of top items), *percentile-based thresholds* are used to ensure a more adaptive and comparable evaluation across different rankers. Specifically, the top percentiles (75, 80, 85, 90, and 95) are considered in order to analyze how well the aggregated ranking retains elements from each individual ranking.

For a given ranking criterion, the *relevant items* are those that fall within the top percentile of interest (e.g., the top 80%). The recall is then computed by determining the overlap between these top-ranked items and those that appear within the same range in the final aggregated ranking.

Thus the formulation becomes:

$$recall@\% = \frac{criteria\_top\% \cap aggregator\_top\%}{criteria\_top\%}$$

This approach provides valuable insights by assessing the influence of each individual criterion on the final ranking. By evaluating recall separately for each criterion, it becomes possible to determine how strongly each feature contributes in shaping the top-ranked elements. A straightforward way to quantify the overall representativeness of the aggregated ranking is to compute the *average recall* across all criteria, obtaining a single comprehensive index of alignment.

One limitation of this method is that it focuses exclusively on the top-ranked portion of the list, without providing information about the entire ranking distribution. However, this is a minor concern considering its practical applications. In real-world scenarios, users primarily interact with the top elements, making their correct positioning the most critical factor. Since the ranking is designed to surface the most relevant outliers, ensuring that the top-ranked items accurately reflect the various criteria is the key priority.

3.2.4 Obtained Results

The tests were conducted using the same criteria as the Euclidean distance method currently used in production. The selected criteria include: *maximized difference value* ($C1$), measuring the deviation between the actual and expected outlier value, and *temporal recency* ($C2$), prioritizing recent outliers. This alignment enables comparability between the proposed ranking aggregation methods and the existing approach.

Table 3.1 presents key results for different percentile ranges. Across all tested approaches (CombMIN, CombMAX, and CombSUM), $C1$ consistently maintains strong representation, with recall values exceed-

	75%			80%			85%			90%			95%		
	Min	**Max**	**Sum**	**Min**	**Max**	**Sum**	**Min**	**Max**	**Sum**	**Min**	**Max**	**Sum**	**Min**	**Max**	**Sum**
C1	**0.763**	0.571	0.701	**0.885**	0.538	0.680	**0.872**	0.501	0.647	**0.872**	0.501	0.647	**0.872**	0.501	0.647
C2	0.567	0.565	**0.589**	0.387	0.485	**0.510**	0.301	0.439	**0.456**	0.276	0.372	**0.411**	0.245	0.254	**0.257**
AVG	**0.665**	0.568	0.644	**0.636**	0.512	0.595	**0.587**	0.470	0.552	**0.574**	0.436	0.529	**0.558**	0.378	0.452

Table 3.1: Comparison between recall@% of different combination methods.

ing 0.75 and reaching peaks of 0.87 for percentile thresholds above 80. Among the three methods, CombMIN best preserves $C1$ representation across all percentiles. Conversely, $C2$ is less well represented, with *recall@75th* values around 0.5, decreasing to 0.25 at *recall@95th*. The CombSUM method demonstrates the strongest ability to retain $C2$ relevance, outperforming the other methods in maintaining recency representation.

When considering the average representativeness of both criteria, CombMIN emerges as the most effective aggregation method across all percentiles, consistently preserving the influence of both $C1$ and $C2$. CombSUM follows closely behind, while CombMAX ranks third, trailing the top method by $10 - 20$ percentage points, depending on the percentile range. These results suggest that CombMIN provides the best overall balance between the two ranking criteria.

Rank Solutions Table 3.2 presents the top 20 outliers ranked by each combination method, highlighting the key differences in the proposed results. CombMAX heavily favours $C1$ (difference value) over $C2$ (recency), leading to a ranking where all top-20 outliers have the maximum possible difference value of 100, but the corresponding dates are inconsistent, with some outliers being more than 10 months old. This behaviour stems from the fundamental design of CombMAX, which assigns each element the highest score across the rankers. As a result, all outliers with $C1 = 100$ receive the same score, causing numerous ties in ranking and reducing the model's ability to differentiate among them.

CombMIN, on the other hand, is more conservative in elevating non-relevant elements, as it assigns each outlier the lowest score among the

	CombMIN		CombMAX		CombSUM	
	C1	**C2**	**C1**	**C2**	**C1**	**C2**
1	59.32	04/10/24	100	19/01/24	100	01/10/24
2	59.32	04/10/24	100	24/01/24	100	01/10/24
3	59.31	04/10/24	100	19/01/24	100	01/10/24
4	59.25	04/10/24	100	19/01/24	100	01/10/24
5	59.20	04/10/24	100	19/01/24	95.43	01/10/24
6	59.00	04/10/24	100	10/06/24	94.04	01/10/24
7	58.90	04/10/24	100	10/06/24	92.39	01/10/24
8	58.87	04/10/24	100	29/02/24	83.94	01/10/24
9	58.73	04/10/24	100	18/01/24	78.26	01/10/24
10	55.55	01/10/24	100	16/05/24	45.37	23/10/24
11	55.02	01/10/24	100	19/01/24	45.37	23/10/24
12	48.66	10/10/24	100	18/01/24	59.32	04/10/24
13	46.67	10/10/24	100	18/01/24	59.32	04/10/24
14	46.67	10/10/24	100	18/01/24	59.32	04/10/24
15	46.62	10/10/24	100	27/05/24	59.32	04/10/24
16	45.95	09/10/24	100	19/01/24	59.31	04/10/24
17	45.94	09/10/24	100	19/01/24	58.97	04/10/24
18	45.93	09/10/24	100	19/01/24	58.96	04/10/24
19	45.88	09/10/24	100	19/01/24	58.92	04/10/24
20	45.80	10/10/24	100	27/05/24	58.90	04/10/24

Table 3.2: Top 20 ranked results for Combination methods.

proposed values. This approach produces a more coherent ranking, with dates skewed towards recent observations. Therefore the values of $C1$ in the top ranks are not maximum. While this trade-off may be acceptable, it is not fully aligned with the client's requirements, which prioritize both high discrepancy and recency.

Finally, CombSUM offers the most balanced ranking between $C1$ and $C2$. The values of $C1$ span a broad range, rather than being dominated by the maximum score, while the recency factor $C2$ ensures that recent outliers remain well represented. This approach reflects the core principle of CombSUM, balancing the different criteria rather than favoring one over the other. Despite not achieving the highest recall performance (0.529 for $recall@90th$, compared to 0.574 for CombMIN), CombSUM provides the most interpretable ranking. The fluctuations in $C1$ values are clearly linked to corresponding variations in $C2$, making it easier for non-technical users to understand the ranking logic.

In conclusion, while CombSUM does not deliver the absolute best recall, it remains a strong candidate for ranking due to its balance, transparency, and usability. Its ability to account for all criteria without overemphasizing any single aspect makes it a practical choice for this application, especially considering scenarios where more than two criteria must be considered.

3.3 Comparative Analysis

Figure 3.2 compares the top-15 ranked outliers produced by the current Euclidean distance-based ranking and the best-performing aggregation method, CombSUM. Both approaches yield coherent rankings but with notable differences in prioritization.

CombSUM favours recency while still considering high difference values ($C1$). This ensures that the top-ranked elements are both recent and significantly different from expectations, aligning better with real-world usability where fresh insights are typically more actionable. In contrast, the Euclidean distance method leans more heavily toward maximizing $C1$, sometimes at the cost of recency. As a result, it introduces older outliers into the top 15, which may not always be relevant depending on the client's needs.

Figure 3.3 illustrates how four different outliers are ranked under Euclidean Distance (ED) and CombSUM (CS). The first example, item 22929, is ranked first by ED and second by CS, demonstrating a high level of agreement between the two methods. This suggests that the item has both a strong difference value and reasonable recency, making it highly relevant regardless of the ranking approach. However, item 23701 sees a significant boost from position 55 under ED to position 4 under CS. This shift occurs because ED has a tendency to prioritize difference values (C1), rather than balancing the criteria, causing a

Euclidean distance (a)

	dimensions_keys	dimensions_vals	measures	values	reference_values	diff_norm C1	time C2
1	cod_struttura_ero | cod_azienda_res	34111a | 190203	imp_lordo	1826.0	7.45	100.0	2024-10-01
2	cod_tipo_contatto	0	num_accessi	4299.0	13119.5	100.0	2024-10-01
3	cod_prestazione | cod_uop_ero	89.01 | 10805680310l008	tempo_attesa_1disp	341.0	0.0	100.0	2024-10-01
4	cod_cittadinanza | cod_tipo_ric	000 | 1	imp_regionale	86480.0	653.88	95.62	2024-09-01
5	cod_fonte	9lis	num_accessi	1575.0	177656.0	100.0	2024-10-01
6	cod_cittadinanza | cod_branca	523 | b	tempo_attesa_1disp	511.0	0.04	100.0	2024-09-01
7	cod_scelta_utente | cod_tipo_contatto | cod_tipo_prestazione	1 | 1 | 1	num_accessi	8.0	1.0	100.0	2024-06-10
8	cod_cittadinanza | cod_compensa | cod_tipo_contatto	100 | 1 | 1	num_accessi	8.0	1.0	100.0	2024-06-10
9	cod_azienda_res | cod_modalita_dim | cod_modalita_ero	090203 | 02 | 0	gg_preoperatorie	197.0	0.99	99.51	2024-07-24
10	cod_modalita_dim | cod_modalita_ero	02 | 0	gg_preoperatorie	197.0	0.94	99.54	2024-07-24
11	cod_cittadinanza | cod_modalita_dim | cod_modalita_ero	100 | 02 | 0	gg_preoperatorie	197.0	0.94	99.54	2024-07-24
12	cod_cittadinanza | cod_modalita_dim	100 | 02	gg_preoperatorie	197.0	0.93	99.54	2024-07-24
13	cod_classe_pri	b	num_ricovero	257.0	571.47	100.0	2024-08-01
14	cod_medico_base | cod_tipo_ric	CFXXXX| 1	imp_regionale	95880.0	6127.97	100.0	2024-08-01
15	cod_cittadinanza | cod_classe_pri	100 | a	gg_preoperatorie	197.0	0.73	99.64	2024-07-24

CombSUM (b)

	dimensions_keys	dimensions_vals	measures	values	reference_values	diff_norm C1	time C2
1	cod_fonte	9LIS	num_accessi	1575.0	177656.0	100.0	2024-10-01
2	cod_struttura_ero | cod_azienda_res	34111A | 190203	imp_lordo	1826.0	7.45	100.0	2024-10-01
3	cod_prestazione | cod_uop_ero	89.01 | 10805680310L008	tempo_attesa_1disp	341.0	0.0	100.0	2024-10-01
4	cod_tipo_contatto	0	num_accessi	4299.0	13119.5	100.0	2024-10-01
5	cod_struttura_ero | cod_priorita	41261A | D	num_accessi	36052.0	804.98	95.43	2024-10-01
6	cod_modalita_acc | cod_tipo_contatto	11 | 2	num_accessi	13584.0	4706.37	94.03	2024-10-01
7	cod_priorita	D	num_accessi	575544.0	412853.78	92.39	2024-10-01
8	cod_branca | cod_priorita	F | P	tempo_attesa_1disp	344.0	57.73	83.94	2024-10-01
9	cod_presidio_ero	09023701	num_ricovero	0.0	246.16	78.26	2024-10-01
10	cod_ambito | cod_esenzione | cod_specialita	090107 | 000000 | 519	tempo_attesa_1disp	3648.0	1.11	45.37	2024-10-23
11	cod_ambito | cod_branca | cod_esenzione	090107 | K | 000000	tempo_attesa_1disp	3648.0	1.12	45.37	2024-10-23
12	cod_modalita_dim | cod_regime_ric	02 | 2	gg_postoperatorie	217.0	0.45	59.32	2024-10-04
13	cod_modalita_dim | cod_modalita_ero | cod_regime_ric	02 | 0 | 2	gg_postoperatorie	217.0	0.45	59.32	2024-10-04
14	cod_modalita_ero | cod_regime_ric	0 | 2	gg_postoperatorie	217.0	0.45	59.32	2024-10-04
15	cod_azienda_res | cod_modalita_ero | cod_regime_ric	090203 | 0 | 2	gg_postoperatorie	217.0	0.46	59.32	2024-10-04

Figure 3.2: Comparison between ranking methods results:
(a) Euclidean Distance (b) CombSUM

Euclidean distance (a)

idx	dimensions_keys	measures	values	reference_values	diff_norm C1	time C2	ranking_pos
22929	cod_struttura_ero | cod_azienda_res	imp_lordo	1826.0	7.45	100.0	2024-10-01	0
23701	cod_struttura_ero | cod_priorita	num_accessi	36052.0	804.98	95.43	2024-10-01	55
2365	cod_azienda_res | cod_classe_pri	gg_preoperatorie	16.0	0.58	7.443	2024-01-26	7366
5074	cod_branca | cod_cittadinanza	imp_lordo | imp_ticket	78.01 | 0.0	40.53 | 38.55	0.000	2024-02-22	24206

CombSUM (b)

idx	dimensions_keys	measures	values	reference_values	diff_norm C1	time C2	ranking_pos
22929	cod_struttura_ero | cod_azienda_res	imp_lordo	1826.0	7.45	100.0	2024-10-01	1
23701	cod_struttura_ero | cod_priorita	num_accessi	36052.0	804.98	95.43	2024-10-01	4
2365	cod_azienda_res | cod_classe_pri	gg_preoperatorie	16.0	0.58	7.443	2024-01-26	16609
5074	cod_branca | cod_cittadinanza	imp_lordo | imp_ticket	78.01 | 0.0	40.53 | 38.55	0.000	2024-02-22	23783

Figure 3.3: Analysis on ranking positions for different algorithms:
(a) Euclidean Distance (b) CombSUM

lower rank. Conversely, item 2365 experiences a sharp drop of $10,000$ positions under CS, primarily due to its early timestamp (January). While its difference value is still consistent, the older date reduces its importance under the CS approach. Finally, item 5074 remains largely unchanged at the bottom of both rankings, lacking both in significant difference and recent occurrence.

Figure 3.4 shows the distribution of rank displacement among the two algorithms. High peaks can be found for jumps of less than 400 elements, yet many records experience significant displacement. This effect is primarily attributed to the greater emphasis on balancing the time feature, which shifts rankings toward more recent outliers.

Overall, this example highlights how CombSUM effectively incorporates recency, leading to more timely rankings compared to the difference-heavy Euclidean Distance approach. By balancing both recency and discrepancy, CombSUM provides a more interpretable ranking, making

it a better fit for scenarios where both factors are important.

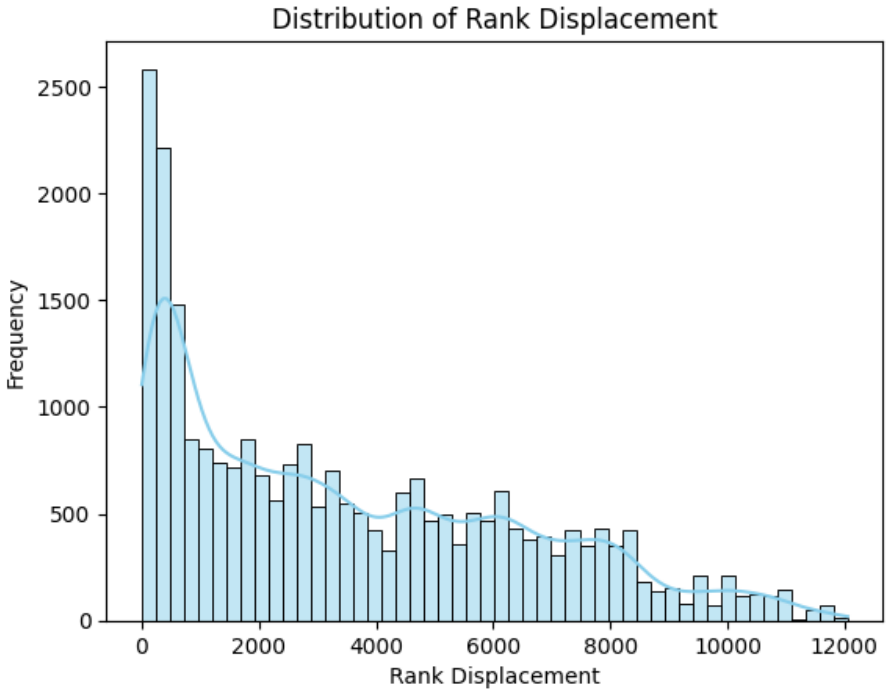

Figure 3.4: Rank displacement distribution between CombSUM and Euclidean distance approach

Chapter 4

Similar Products Recommendations

This section aims to integrate a system that proposes similar elements to the one currently under analysis. These recommendations should consider both domain-specific characteristics (e.g., type of event, measure, dimension, observation time) and anomaly-specific attributes (e.g., detection algorithm, severity, value discrepancy).

By incorporating both perspectives, this approach ensures that recommendations are not just similar in content but also in structure, promoting diversity. By considering multiple aspects, the system moves beyond direct similarity and incorporates serendipity, allowing users to uncover unexpected yet relevant patterns.

A traditional recommendation system would primarily focus on direct feature similarity. For example, when analyzing an outlier related to unpaid amounts at the hospital level, a standard model would likely suggest other cases with similarly high unpaid amounts. While this method ensures consistency, it can overlook alternative but structurally related insights.

However, this model takes a broader approach, identifying similarity not only in content but also in structural characteristics. Instead of restricting recommendations to unpaid amounts at hospitals, the system might surface cases from different facilities (e.g., wards or private clinic) or identify structurally similar anomalies in different domains,

such as sharp fluctuations in hospital access rates. Even though access rate variations are not directly related to unpaid amounts, they may follow a comparable reporting pattern, making them informative from an analytical standpoint.

By expanding the definition of similarity, this methodology helps users move beyond expected correlations and instead discover patterns that might have otherwise gone unnoticed. This approach broadens the analytical scope, ensuring that the most relevant and insightful outliers are surfaced—not just the most obvious ones.

Ultimately, this strategy enriches the recommendation process by blending relevance with diversity, allowing users to gain a more comprehensive understanding of the dataset.

4.1 Methodology

The proposed system is designed as a two-stage pipeline Figure 4.1, ensuring an efficient and structured approach to outlier recommendation. These two stages consist of:

(I) *Candidate Generation* focusing on identifying a pool of potential similar elements to the input query outlier. The process utilizes soft clustering methodologies on data to identify a pool of elements with similar characteristics to the input query element. The concept is as follows: elements belonging to the same cluster as the query element are presumed to be correlated or at least share some common characteristics with it. Therefore, elements of this group can be considered a valid candidate pool. The process utilizes Gaussian Mixture Model (GMM) clustering over UMAP embedded data.

(II) *Item Scoring* stage involves ordering the candidates based on their relevance to the query element. A distance-based approach, similar to K-Nearest Neighbors (KNN), is used to rank the candidates leveraging Euclidean distance to the query item. This method ensures that the

most similar and contextually relevant outliers are prioritized in the recommendation process.

Many recommendation systems include a third re-ranking step, which refines the ordered list by filtering out less relevant elements, such as already seen outliers or redundant recommendations. However, in this system, re-ranking is not currently included due to the absence of user-specific interaction data (e.g., user feedback, historical preferences). Since this type of user-aware filtering requires additional contextual information, it is planned as a future enhancement of the system.

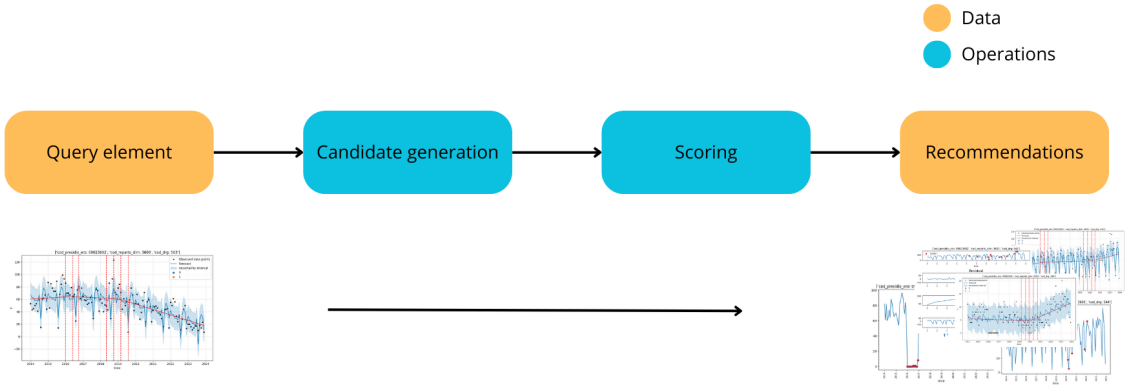

Figure 4.1: Similar Product Recommendation operative pipeline

4.1.1 Feature Selection

Given the variegated nature of the domain, the feature selection process was designed to strike a balance between maximizing informational value and maintaining a broad contextual perspective. The key challenge in achieving this balance was handling *measures* and *dimensions*, as they both carry essential contextual knowledge necessary for outlier analysis.

Encoding Measures Measures are at the core of the anomaly detection process, as they define the type of data being analyzed. To ensure that the model captures both the scope of the analysis and the numerical significance of each measure, a tailored encoding approach was required.

The dataset originally stored measure-related information in two separate columns: 1. Measure Key – Representing the name or type of the measure. 2. Measure Value – Containing the corresponding numerical value.

This structure necessitated a refactoring step to ensure the data was in a format that differentiates measures. The encoding strategy involved transforming each measure into a separate feature, where each unique measure key became a column. The corresponding measure values were then assigned to their respective positions in this new format. This ensures that the model captures and preserves the range and scale of different measures separately.

Encoding Dimensions Dimensions define the scope and context in which measures exist. However, including all dimension values could overwhelm the model with excessive granularity, leading to unnecessary complexity. To mitigate this, only dimension keys were retained as contextual anchors, without including their specific values. A one-hot encoding technique was applied to represent these categorical attributes efficiently.

For example, an outlier related to *monthly hospital access numbers* would retain its measure value (e.g., 10,000 accesses) while keeping only the dimension type (e.g., hospital code) rather than the specific hospital it relates to.

This simplified encoding enables the model to generalize anomaly categories rather than overfitting to specific details, ensuring that the system captures broader patterns rather than isolated cases.

Other features included in the model are: *reference value*, *value difference*, *time*, *granularity*, *analysis type*, *event type*, *event subset*, *algorithm*, *outlier score*, *seasonality*, *aggregation*, *number of events*, *avg number of events*.

The pre-processing phase focused on transforming the features into

a structured and meaningful format for analysis: numerical features were normalized to ensure consistency across different scales. Categorical features were one-hot encoded. At the end of the process, results approximately 55 features for the model.

Initial clustering analysis did not reveal well-defined groupings, as indicated by a silhouette score of 0.132, suggesting weak cluster separation. To explore potential improvements, dimensionality reduction techniques were applied: PCA and t-SNE were tested but did not yield significant benefits for clustering, resulting in a silhouette respectively of *0.187* and *0.193*. Given these limitations, a nonlinear transformation approach, UMAP, was explored as an alternative to better capture the underlying structure of the data.

4.1.2 Uniform Manifold Approximation and Projection

UMAP (Uniform Manifold Approximation and Projection) is a dimension reduction technique based on Riemannian geometry. The algorithm operates by constructing a high-dimensional topological representation of the data as a weighted k-neighbor graph, with edge weights determined by the distance between points, and then finds a low-dimensional embedding that preserves this topological structure. [18]

The algorithm wants to create a dimensional reduction that maintains both local and global relationships found in the high dimensional space. To do so, the following assumptions about the data are made:

- There exists a manifold on which the data would be uniformly distributed.
- The underlying manifold of interest is locally connected.
- Preserving the topological structure of this manifold is the primary goal.

High Dimensional Space The approach is based on the creation of a *weighted k-nearest neighbor graph* to approximate the manifold structure. For each point x_i, it identifies its k nearest neighbors and computes a local connectivity measure ρ_i as the distance to its nearest neighbor.

UMAP then establishes a set of probabilities $p_{i,j}$ that represent the likelihood of a connection between points x_i and x_j in the high-dimensional space. These probabilities are calculated using a smooth kernel function:

$$p_{i,j} = exp(\frac{-max(0, d(x_i, x_j) - \rho_i)}{\sigma_i})$$

where $d(x_i, x_j)$ is the distance between points, and σ_i is a normalization factor chosen to ensure that the sum of probabilities from point x_i to its neighbors equals a predefined parameter called "local connectivity." This creates what the authors call a "fuzzy topological representation", where the elements have degrees of membership valued in the real unit interval $[0, 1]$, effectively capturing the manifold structure locally [18][19][20].

Fuzzy topological set composition UMAP has the ability to combine multiple models integrating different fuzzy topological sets into one cohesive space.

As said, high-dimensional data are represented through fuzzy simplicial sets, which are combinatorial structures that capture the topological and geometric relationships within the data. Each simplex (a generalization of triangles to higher dimensions) in these sets has an associated membership strength between 0 and 1, indicating the degree to which data points form a cluster or neighborhood. [18]

Multiple UMAP models can be trained considering different feature subsets or different parameter configurations; each model yields a distinct fuzzy simplicial set reflecting the manifold structure as perceived through those characteristics. To integrate these models into a single representation, UMAP employs operations such as *intersections* or

unions on the fuzzy simplicial sets.

With intersection the membership strength of a simplex in the combined set is taken as the minimum membership strength from the corresponding simplices in the individual sets. Union instead utilizes as membership strength the maximum membership strength from the individual sets [21].

These operations result in a new fuzzy simplicial set that encapsulates the shared or combined topological features across the different perspectives [18][22]. This combined set is then used to construct a low-dimensional embedding that reflects the integrated manifold structure of the data.

Low Dimensional Space In the low-dimensional space, UMAP defines a similar probability distribution $q_{i,j}$ for points y_i and y_j based on their distances:

$$q_{i,j} = (1 + a||y_i - y_j||^{2b})^{-1}$$

where a and b are parameters that control the strength of attraction and repulsion between points, and $||y_i, y_j||$ is the distance (by default Euclidean distance) in the low-dimensional embedding space [18][20].

Optimization The loss function optimized by UMAP is the cross-entropy to measure how well the low dimensional similarities $q_{i,j}$ represent the high dimensional similarities $p_{i,j}$. This can be expressed as:

$$C = \Sigma_{i,j}(p_{i,j}log(\frac{p_{i,j}}{q_{i,j}}) + (1 - p_{i,j})log(\frac{1 - p_{i,j}}{1 - q_{i,j}}))$$

The optimization process employs a specialized version of the stochastic gradient descent (SGD) with negative sampling techniques. The gradient update rule adjusts the positions of points in the low-dimensional space based on a set of attractive forces applied along edges and a set of repulsive forces applied among vertices. The algorithm proceeds by

iteratively applying attractive and repulsive forces at each edge or vertex. This amounts to a non-convex optimization problem. Convergence to a local minima is guaranteed by slowly decreasing the attractive and repulsive forces in a similar fashion to that used in simulated annealing [18][20].

Hyperparameters The parameter settings in UMAP significantly influence its behavior: the *n_neighbors* parameter controls the size of the local neighborhood and thereby affects the balance between preserving local versus global structure; larger values emphasize global structure at the expense of local details. The *min_dist* parameter influences how close the points in the low dimensional embedding can be packed together. It influences, together with the *spread* parameter, the calculation of the a and b values. Influencing the computation of the low dimensional similarities $q_{i,j}$. Low values will result in compact embeddings, which can be useful for clustering [18].

4.1.3 Gaussian Mixture Models Clustering

Gaussian Mixture Models (GMM) [23] are selected as the preferred clustering algorithm. GMM are probabilistic models that assumes all datapoints are generated from a mixture of several Gaussian distributions with unknown parameters. In particular each cluster is defined by a Gaussian distribution, and the datapoints are assigned probabilities of belonging to different clusters based on their distance from each Gaussian. In the context of clustering, GMM provide a flexible approach. It can be used to perform either *hard clustering* or *soft clustering* on query data. To perform hard clustering, the GMM assigns query data points to the multivariate normal components that maximize the component posterior probability, given the data. Hard clustering assigns a data point to exactly one cluster. Additionally, GMM can be used to perform more flexible clustering on data, soft clustering, assigning a

score to a data point for each cluster. The value of the score indicates the association strength of the data point to the cluster. As opposed to hard clustering methods, soft clustering methods are flexible because they can assign a data point to multiple clusters. This characteristic is particularly suitable within the recommendation domain. An outlier may be included in more than one group given its characteristics, thus boundaries are not exclusive [24][23].

Gaussian Mixture Models (GMM) The model assumes each datapoint as a random variable independently and identically distributed from an unknown distribution with a probability density function expressed as:

$$p(x) = \sum_{j=1}^{k} \pi_j \phi(x; \mu_j, \Sigma_j)$$

This indicates a mixture of k component multivariate Gaussian distributions where $\phi(x; \mu_j, \Sigma_j)$ is a multivariate Gaussian density with unknown parameters (μ_j, Σ_j). Thus

$$\mathcal{N}(x|\mu_j, \Sigma_j) = \frac{1}{|2\pi\Sigma_j|^{1/2}} exp(-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j))$$

Factor π_j is the unknown probability of selecting the component j, satisfying $\sum_{j=1}^{k} \pi_j = 1$. Solving clustering task resolves in inferring the latent component z_i responsible for each x_i.

$$p(z_i = j|x_i) = \frac{p(z_i = j)p(x_i|z_i = j)}{p(x_i)} = \frac{\pi_j \phi(x_i; \mu_j, \Sigma_j)}{\sum_{l=1}^{k} \pi_l \phi(x_i; \mu_l, \Sigma_l)}$$

To achieve this, a subtle problem must be addressed: estimating the parameters $(\pi, \mu_{1:k}, \Sigma_{1:k})$ through statistical inference with the Expectation-Minimization (EM) algorithm. [23]

Expectation-Minimization EM EM is an iterative optimization technique that seeks to find maximum likelihood estimates in the presence of latent variables, which in this case are the actual cluster assignments. The algorithm is composed of two main steps: expectation step (E) and minimization step (M). Expectation computes the posterior probability that each data point x_i belongs to each cluster j.

$$\tau_{ij} = P(z_i = j | x_{ij}, \pi, (\mu_l, \Sigma_l))$$

Maximization updates the parameters π_k, μ_k and Σ_k by inserting the factor τ_{ij} to maximize the expected log-likelihood.

$$\pi_j = \frac{1}{n}\sum_{i=1}^{n}\tau_{ij}, \quad \mu_j = \frac{\sum_{i=1}^{n}\tau_{ij}x_i}{\sum_{i=1}^{n}\tau_{ij}}, \quad \Sigma_j = \frac{\sum_{i=1}^{n}\tau_{ij}(x_i-\mu_j)(x_i-\mu_j)^T}{\sum_{i=1}^{n}\tau_{ij}}$$

The EM algorithm iterates between these two steps until convergence, typically when the change in the log-likelihood function falls below a predetermined threshold. This iterative process effectively handles the uncertainty of cluster assignments.[23][24]

Soft clustering technique A key advantage of Gaussian Mixture Models is the ability to provide soft clustering by expressing the probability density of an element belonging to each cluster. This property becomes particularly valuable in this setting where clustering is used for candidate generation. In some cases, elements do not have a clear and exclusive cluster assignment, particularly when they are positioned near cluster boundaries. In these uncertain situations, soft clustering proves highly useful, as it allows for a more flexible candidate selection.

To account for this uncertainty, an agglomerative clustering technique is applied to refine the candidate selection process. When determining an element's belonging cluster to generate recommendations, a threshold-based merging strategy is introduced. If an element's probability of belonging to multiple clusters is within a certain percentage of

its highest probability score, the corresponding clusters are merged into a single candidate group. This approach ensures that items with similar probabilities across multiple clusters are not arbitrarily assigned to one, but instead considered within a broader yet structured context, leading to more accurate and meaningful recommendations.

For this dataset, the merging lower bound threshold was set at 30%, ensuring that elements with significant uncertainty are properly accounted for. This threshold was determined based on the observation that most items exhibit a strong tendency to belong to a single dominant cluster, meaning they remain unaffected by the merging process. However, a small subset of cases displays high uncertainty, where an element has relatively balanced probabilities across multiple clusters. By setting the threshold at 30%, the model captures these ambiguous cases effectively.

Considering a scenario where an element's cluster membership probabilities are $0.3, 0.3, 0.2, 0.1$, with all other clusters summing to 0.1. In this case, it is reasonable to assume that all four clusters contain potentially relevant similar elements. However, in a scenario where an element has a strong primary cluster membership (e.g., $0.8, 0.1$ with all others summing to 0.1), merging multiple clusters would be less justified.

4.1.4 Nearest Neighbors Selection

Given a query element x, the task consists in determining the k most similar elements from the restricted candidate pool identified by GMM. This is done using the KNN algorithm, involving the following steps: first, the distance between x and each element in the candidate pool is computed, then those distances are sorted in ascending order, and finally the top k elements, with smallest distances, are selected. The Euclidean distance is used as the similarity metric of choice; ensuring consistency with UMAP, which was used in the dimensionality reduction step. Since UMAP's low-dimensional space is defined using Euclidean geometry,

using the same metric in the ranking phase maintains coherence between the embedding process and the retrieval mechanism.

4.2 Experimental Setup and Results

4.2.1 Training Strategy

The model underwent extensive parameter tuning to optimize clustering performance.

The training strategy was designed to enhance the model's ability to cluster elements effectively. A key component of this approach was the embedding method. In this case, UMAP was not just used for dimensionality reduction but also for creating a meaningful Euclidean space where all feature types could be represented coherently. To evaluate the final clustering performance, a comprehensive pipeline was developed. Since the problem is unsupervised, the Silhouette Score was chosen as the guiding metric for hyperparameter tuning. The pipeline optimized all stages of the clustering process, from UMAP embedding creation to final clustering execution, ensuring that each component contributed to a more coherent and effective grouping structure.

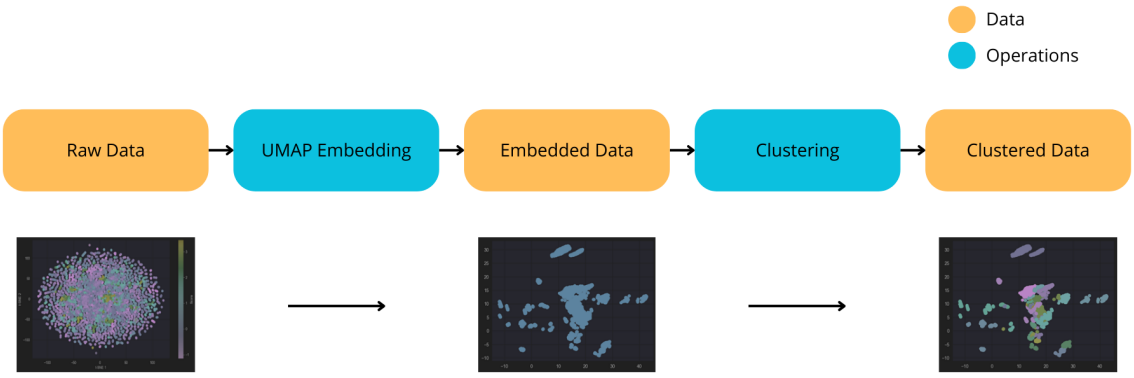

Figure 4.2: Similar product Recommendation: embedding and clustering

To better capture the underlying structure of the dataset, two separate UMAP models were trained, one dedicated to numerical features and the other focused solely on categorical features. This distinction was

introduced to preserve the integrity of the data structure and to mitigate potential topological distortions that could arise from differences in scales and distributions. Since numerical and categorical features exist in fundamentally different metric spaces, combining them directly in a single model could lead to imbalanced feature representation. In such a scenario, certain features, particularly those with high variance, might dominate the learned space, reducing expressivity of others and distorting the overall vector space structure. The decision to learn separate embeddings ensures that each data type is processed using the most appropriate similarity metric, leading to a more meaningful and interpretable representation of the data. Additionally, training distinct models allows for independent hyperparameter tuning, optimizing the representation of each feature type and improving the clustering quality.

Once the two UMAP embeddings are learned, they are fused into a single fuzzy topological representation, which is then embedded into a low-dimensional Euclidean space.

4.2.2 Hyperparameters Optimization

A random search approach was employed to optimize the various embedding parameters. The best hyperparameters identified through this process are summarized in Table 4.1. One of the key findings from this search was the determination of the optimal latent space size, which was established at *14* dimensions.

	Numerical UMAP	***Categorical UMAP***
n-components	14	14
n-neighbors	250	100
min-dist	0.231	0.282
num-epochs	411	490
metric	manhattan	rogerstanimoto
combination-method	union	
k-clustering	20	

Table 4.1: UMAP best hyperparameters - silhouette 0.592

This optimized configuration significantly enhanced clustering performance. The silhouette score improved substantially. Without UMAP embeddings, clustering yielded a silhouette score of *0.193*. However, after incorporating the optimized UMAP transformation, the score increased to *0.592*, indicating a far more cohesive grouping of data points.

Following this, a secondary parameter search was conducted to refine the settings for the Gaussian Mixture Model (GMM) clustering algorithm. The best configurations for GMM, presented in Table 4.2, further enhanced clustering effectiveness. With these optimized settings, the silhouette score reached its peak at *0.653*. The optimal number of clusters for the dataset was determined to be *22*.

n-components	22
covariance-type	full
tol	0.001
init-params	k-means++

Table 4.2: GMM best hyperparameters - silhouette 0.653

For the last step of nearest neighbors scoring, no training is necessary. Having at hand the pool of generated candidates, the only operation needed is to measure the Euclidean distance between the query element and each candidate. The k elements having shorter distance to the query element are selected as proposals.

4.2.3 Results

Follow some examples of query and their recommended similar elements.

The results presented in Table 4.3 illustrate the system's effectiveness in detecting and contextualizing outliers related to access numbers. In this particular case, the query outlier corresponds to a 50% increase in patient visits to a specific doctor within a particular ASL (local health authority) during May 2024. This deviation is flagged as anomalous due to its significant divergence from historical patterns. The recommendations maintain a high level of contextual relevance, as they all refer to the

Query element									
analysis	*event*	*dimensions_keys*	*dimensions*	*measures_keys*	*measure*	*ref_val*	*outlier_score*	*diff_norm*	*time*
contextual	SDO	cod_azienda | cod_medico_base	902 | XXX	num_ricovero	19	13.66	26.94	6.67	2024-06-01
Recommendations									
analysis	*event*	*dimensions_keys*	*dimensions*	*measures_keys*	*measure*	*ref_val*	*outlier_score*	*diff_norm*	*time*
contextual	SDO	cod_modalita_ero | cod_tipo_drg	6 | C	num_ricovero	12	5.44	37.66	8.21	2024-05-01
contextual	SDO	cod_modalita_ero | cod_ambito	6 | 901	num_ricovero	11	4.3	39.30	8.32	2024-05-01
contextual	SDO	cod_cittadinanza | cod_medico_base	100 | YYY	num_ricovero	17	10.71	34.11	7.86	2024-07-01
contextual	SDO	cod_azienda | cod_medico_base	902 | XXX	num_ricovero	17	11.64	27.03	6.69	2024-08-01
contextual	SDO	cod_diagnosi_pri | cod_modalita_dim	653 | 02	num_ricovero	11	3.62	36.67	9.24	2024-06-01

Table 4.3: Top-5 Recommendations for outlier concerning number of access

number of accesses while remaining within the same analysis type (contextual) and event type (SDO). However, rather than providing identical cases, the system expands the scope by suggesting similar anomalies in slightly different domains. Each recommended case also exhibits a comparable increase in access (around 50%) and falls within a similar time frame.

This suggests an accurate analysis of the phenomena: investigating the causes of this access variation can help technicians to understand the presence of ongoing anomalous trends in the area, allowing for preventive measures.

Query element									
analysis	*event*	*dimensions_keys*	*dimensions*	*measures_keys*	*measure*	*ref_val*	*outlier_score*	*diff_norm*	*time*
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	3.78	38.20	5.50	2024-06-01
Recommendations									
analysis	*event*	*dimensions_keys*	*dimensions*	*measures_keys*	*measure*	*ref_val*	*outlier_score*	*diff_norm*	*time*
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	3.50	38.31	5.51	2024-05-01
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	3.49	38.28	5.52	2024-04-01
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	2.70	38.58	5.56	2024-03-01
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	2.78	38.66	5.55	2024-02-01
contextual	SPA	cod_prestazione | cod_tipo_contatto	88992 | 1	imp_lordo	103.0	2.57	37.50	5.57	2024-01-01

Table 4.4: Top-5 Recommendations for outlier concerning ticket price

The results in Table 4.4 highlight the system's ability to detect and contextualize pricing discrepancies in healthcare services. The query outlier pertains to a significant mismatch in gross ticket price, where a visit estimated at *€3.0* was instead charged at *€103.0*. Such discrepancies are not uncommon in healthcare settings. The most likely causes

include nationwide service price variations or human errors in price entry. Regardless of the underlying reason, such inconsistencies require careful investigation to ensure financial and procedural coherence.

The recommendation system assists in this process by identifying five additional cases that closely resemble the query scenario. Notably, all proposed cases exhibit the same type of anomaly: a significant overcharge for the same type of visit (n.88992), but occurring in different past months. This pattern suggests that the issue has persisted over time, remaining undetected and unaddressed due to the rarity of the visit itself, which occurs only a few times per month.

By surfacing these past occurrences, the recommendation system provides users with immediate awareness of a long-standing issue, making it easier to assess whether the pricing error is systematic or isolated. This historical perspective is particularly valuable, as it enables healthcare administrators to trace the origin of the anomaly, implement corrective actions, and prevent further instances of incorrect billing.

Query element									
analysis	***event***	***dimensions_keys***	***dimensions***	***measures_keys***	***measure***	***ref_val***	***outlier_score***	***diff_norm***	***time***
puntuali	SDO	cod_modalità_ero | cdo_classe_pri | cod_modalita_dim	88992 | A | 02	gg_preoperatorie	17	1	4.47	7.93	2024-10-25
Recommendations									
analysis	***event***	***dimensions_keys***	***dimensions***	***measures_keys***	***measure***	***ref_val***	***outlier_score***	***diff_norm***	***time***
puntuali	SDO	cod_azienda_res | cdo_classe_pri | cod_modalita_dim	90203 | A | 02	gg_preoperatorie	16	1	3.53	7.44	2024-10-09
puntuali	SDO	cdo_classe_pri | cod_modalita_dim	A | 02	gg_preoperatorie	17	1	4.52	7.93	2024-10-25
puntuali	SDO	cod_citta | cdo_classe_pri | cod_modalita_dim	100 | A | 02	gg_preoperatorie	16	1	4.16	7.42	2024-10-09
puntuali	SDO	cod_citta | cdo_classe_pri | cod_modalita_dim	100 | A | 02	gg_preoperatorie	17	1	4.45	7.93	2024-10-25
puntuali	SDO	cod_azienda_res | cdo_classe_pri | cod_modalita_dim	90203 | A | 02	gg_preoperatorie	17	1	3.77	7.95	2024-10-25

Table 4.5: Top-5 Recommendations for outlier concerning preoperative delay time (days)

Table 4.5 presents the system's recommendations for an outlier related to surgical waiting times for patients classified under priority class *'A'*. Normally, patients in this category are scheduled for surgery within one day, yet the detected outlier indicates a delay exceeding two weeks. Such a significant deviation raises concerns about potential inefficiencies,

bottlenecks, or administrative issues in the scheduling process.

The recommendation system suggests five additional cases with similar characteristics: same priority class and time frame. By analyzing these recommendations, healthcare administrators can determine whether the delays are isolated to specific hospitals or widespread across multiple facilities. Notably, two out of the five recommendations point to a particular facility (n.90203), suggesting that the issue may be concentrated in that specific location.

This insight enables hospital administrators to conduct targeted investigations to identify the root cause of the delays and implement corrective actions accordingly. By leveraging these recommendations, decision-makers can take a proactive approach to reducing wait times and ensuring that urgent surgical cases are handled promptly.

4.2.4 Expert Assessment

The model revealed valuable insights over the data, demonstrating its ability to generate meaningful recommendations. To assess how well the model's proposals aligned with domain knowledge, a validation process was conducted with an expert. Since labeled data indicating the most appropriate outliers recommendations for a given query were unavailable, conducting an extensive evaluation was impractical. Instead, a preliminary validation phase was designed with the active participation of healthcare experts involved in the project.

A single expert initially participated in the evaluation, reviewing ten randomly selected cases. For each case, the expert was presented with one query element and five recommended items generated by the model. The task required the expert to classify each recommendation as either coherent or not coherent with the query element, resulting in a binary evaluation.

The results were promising: in most cases, the expert deemed three out of five recommendations as coherent, with one case where all five

recommendations were validated. The average validation rate was 3.4 out of 5, indicating that the majority of the recommendations aligned with expert expectations.

To gain further insights, the test was also conducted with three additional participants who were not from the healthcare domain. Their assessments resulted in a slightly higher validation rate of 3.6 out of 5.

4.3 Methodology Evaluation

Recognizing the limitations of the initial validation approach: the small sample size and the limited number of expert reviewers, a more extensive evaluation method was adopted. Given the lack of labeled data and the impracticality of manually verifying recommendations at scale, the testing instead focused on validating the internal components of the model, particularly the clustering process.

Since clustering directly influences candidate selection for recommendations, it is crucial to determine whether the clusters are coherent with domain knowledge. If the clusters fail to accurately group similar outliers, then the recommendations generated from them will also lack reliability.

To address this, a synthetic data generation approach was explored. The idea was to use Gaussian Mixture Model (GMM) generative properties to create synthetic samples that follow the same distribution as the real data. These labeled synthetic samples would then serve as a benchmark for evaluating clustering coherence.

However, this approach led to an issue of self-fulfilling prophecy. Since the synthetic data inherently conformed to the GMM-generated distributions, any clustering performed on it would naturally align with the original clustering structure. As a result, testing with this method became meaningless, as it merely reconfirmed the model's assumptions rather than providing an independent evaluation of clustering validity.

4.3.1 Validation Process

The focus moved from the direct evaluation of the clustering output towards validating the methodology itself. The guiding intuition behind this approach was that if the framework is sound, then the resulting clusters could be trusted as well.

To test the soundness of the methodology, the validation process draw inspiration from knowledge-driven validation and semi-supervised clustering validation [25]. This approach ensures that the clustering process adheres to meaningful domain rules, making the results more interpretable and aligned with expert insights.
The validation methodology is structured around the following key steps:

1. Rule Creation: A set of domain-specific rules is defined to explore different aspects of the dataset. These rules are crafted based on expert knowledge, ensuring they reflect real-world relationships and respect patterns within the data.

2. Cluster Consistency Check: The existing clusters are evaluated against these rules to determine whether they respect the predefined domain structure. The ideal scenario would be that elements within a given cluster consistently adhere to a single rule, indicating that the clustering process effectively captures meaningful data structures.

3. Re-Clustering for Rule Enhancement: The subset of elements that matches a specific rule set are isolated from the whole dataset and re-clustered to assess whether the new clusters improved rule representativeness. This step provides insights into whether the clustering methodology is capable of refining the groupings based on the embedding characteristics of the data.

This framework, designed to validate the methodology, can also serve as a tool for domain experts to explore and interpret the dataset. By systematically investigating how different aspects of the data align with

domain knowledge, the process enables both generalized insights and the identification of highly representative subsets.

This validation approach not only helps in understanding which domain specific aspects are naturally identified by clustering, but also goes further by assessing whether the underlying embedding is truly representative of the data structure. Step three in the validation process is specifically designed to test this aspect. In some cases, after applying a re-clustering procedure on a subset of elements aligned with a given rule, the newly formed groupings may better align with domain expectations. If this occurs, it suggests that the UMAP embedding successfully captures the rule-based subdivision, meaning that the data inherently supports coherent clustering that aligns with expert-defined domain rules.

The methodology should be considered reliable under two key properties:

(I) *Cluster coherence with domain rules*: The clusters should naturally align with known domain rules (Step 2).

(II) *Embedding reliability*: The underlying representation should allow for meaningful rule-based subdivisions (Step 3).

These two aspects can be independently validated by experts, ensuring that the model's decision-making aligns with real-world patterns. Furthermore, the framework's flexibility allows experts to test reliability at different levels of detail, from broad generalizations to highly specific use cases.

4.3.2 Purity Metric for Clustering Validation

The concept of purity as a clustering validation metric is inspired by existing measures that evaluate the alignment between clustering partitions and external constraints, such as class labels or domain rules. Traditional purity metrics [26] assess how well clusters correspond to ground-truth categories, while homogeneity scores [27] measure whether

clusters contain only elements of a single class. Our approach extends these ideas by defining purity with respect to rule-based constraints, similar to validation techniques in constrained clustering [28][25]. Specifically, our metric quantifies the consistency of a clustering solution with respect to predefined rule set, measuring their alignment.

Purity takes values in the range $[0, 1]$, where:

$P(r_i) = 0$ indicates that the rule is entirely misaligned with the clustering, meaning no cluster predominantly adheres to the rule.

$P(r_i) = 1$ indicates perfect alignment, meaning one or more clusters are fully contained within the rule.

Figure 4.3 provides a graphical representation of this concept.

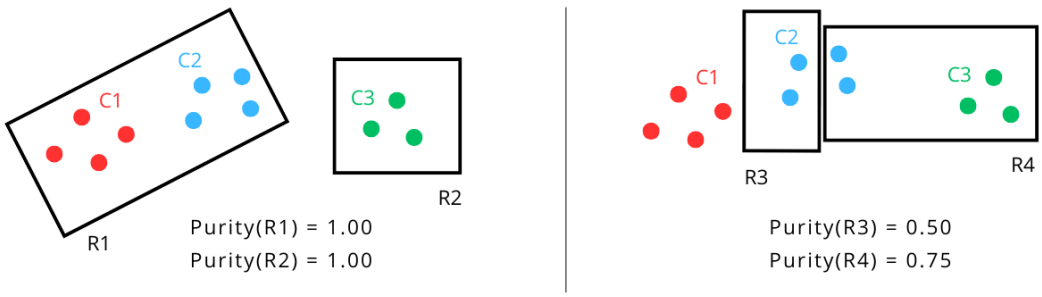

Figure 4.3: Purity metric for clustering validation

Purity of a rule r_i can be formally defined as:

$$P(r_i) = \frac{1}{|C(r_i)|} \sum_{j \in C(r_i)} \frac{|R_i \cap C_j|}{|C_j|}$$

where:

$P(r_i)$ is the purity score of rule r_i.

$C(r_i)$ is the set of clusters that contain at least one element satisfying rule r_i.

R_i is the set of elements satisfying rule r_i.

C_j is the set of elements belonging to cluster j.

This metric provides an interpretable measure of clustering consistency with respect to domain rules. Higher purity values indicate a stronger agreement between clusters and predefined rules, thereby validating the clustering structure against expected domain constraints. Additionally, purity serves as a performance measure for evaluating the overall effectiveness of the clustering model. By analyzing purity scores across multiple rules, we can gain insights into which aspects of the data structure are effectively captured and where improvements may be necessary.

4.3.3 Evaluation Results and Considerations

This section provides a series of rule sets tested and considerations from the validation procedure. To test the overall methodology coherence of the clustering procedure, different levels of details are investigated: from broad generalizations to highly specific use cases.

Broad Investigation Subdivision by measures is a broad investigation of clustering. It's expected that not all rules have completely separated coverage because the elements that define the rule are not specific.

```
rules = {
    '0': measures == 'tempo_attesa_1disp',
    '1': measures == 'gg_postoperatorie',
    '2': measures == 'imp_lordo'
}
```

Each rule includes respectively $[4656, 3048, 2583]$ elements.

	0	**1**	**2**
Clustering	0.80	1.00	0.58
Re-Clustering	0.68	1.00	1.00

Table 4.6: Purity score: subdivision by measures

Analyzing the purity score, we observe that the original clustering successfully separates *gg_postoperatoria* perfectly from the other measures. Additionally, *tempo_attesa_1disp* also shows a fairly distinct division, with a purity score of 0.80, indicating that this rule includes many clusters that are not shared with other rules. Specifically, clusters containing elements with *tempo_attesa_1disp* do not contain elements related to *gg_postoperatoria* or *imp_lordo*, although there is a small overlap with elements from clusters shared among multiple measures.

This effect becomes even more pronounced when considering *imp_lordo*. In this case, the purity score is 0.58, suggesting that nearly half of the clusters involved in this rule are shared with at least one other rule.

For *gg_postoperatoria*, the division remains intact even after performing a re-clustering, confirming the correct distribution of data within the UMAP subspace, as supported by a purity score of 1. For *imp_lordo*, despite not being perfectly separated in the original clustering, it achieves a purity score of 1, indicating that the model has successfully abstracted information consistent with the subdivision by measures. Overall, the purity score reaches 1 for two out of the three rules, while the third rule still has a relatively high score (0.70). This suggests that the proposed methodology effectively captures the intended separation based on the predefined rule set.

Mixed Rule Set Investigation In this case, the rules include both generic criteria for measures and more specific conditions based on time periods.

```
rules = {
    '0':  outlier_score > 30 and measures == 'num_accessi'
          and analysis_type == 'puntuali',
    '1':  measures == 'num_ricovero'
          and analysis_type == 'contestuali'
          and (time_dimensions_vals.str.contains('-04-') or
          time_dimensions_vals.str.contains('-05-') or
```

```
            time_dimensions_vals.str.contains('-06-'))",
    '2':  measures == 'num_accessi'
}
```

Each rule includes respectively $[624, 277, 5856]$ elements.

	0	**1**	**2**
Clustering	0.45	1.00	0.70
Re-Clustering	1.00	1.00	0.70

Table 4.7: Purity score: mixed subdivision by measures and time period

Rule 1 *num_ricovero* is perfectly respected, with a purity score of 1 indicating the model ability to distinguish distinct clusters for this specific measure.

Rule 2 again, on a measure *num_accessi*, is partially respected. A purity score of 0.70 suggests that the model can distinguish some clear clusters for this rule, but not exclusively.

Rule 0 is computed over measure *num_accessi* and other filtering criteria. Here the re-clustering evaluates a purity score of 1, in contrast to the performance of the original model assessing a purity score of 0.45. Even though this rule is not perfectly recognized by the initial clustering, the underlying data characteristics that define it are preserved within the embedding. This is confirmed by the re-clustering process, which achieves a score of 1.

Specific Investigation Only two rules are evaluated, investigating the same rule with a slight variation in the *outlier_score*. Rule 0 considers elements with *outlier_score* < 50. Rule 1 considers elements with *outlier_score* in the range $[50, 70]$.

```
rules = {
    '0': measures == 'gg_postoperatorie'
         and analysis_type == 'puntuali'
         and dimensions.contains('cod_azienda_res')
         and outlier_score < 50,
```

```
    '1': measures == 'gg_postoperatorie'
         and analysis_type == 'puntuali'
         and dimensions.contains('cod_azienda_res')
         and outlier_score > 50,
}
```

Each rule includes respectively $[529, 7]$ elements. Rule 0 achieves a high

	0	**1**
Clustering	0.99	0.02
Re-Clustering	0.99	1.00

Table 4.8: Purity score: specific investigation exploiting one parameter

purity score of 0.99, indicating that the model effectively groups elements within this rule, with minimal overlap with other rules. Therefore only a small number of elements fall within Rule 1, having a size of only 2% of Rule 0).

A purity score of only 0.02 indicates the elevated complexity of the model to identify this rule. The primary reason for this low score is the small size of Rule 1, which contains only 7 elements. The original clustering algorithm struggles to isolate such a small subset into a distinct cluster, indicating its tendency to find global patterns rather than highly specific ones. In this case, no single cluster (or a combination of clusters) is uniquely representative of Rule 1, demonstrating that the rule is not "pure" in terms of clustering.

Therefore, Rule 1 benefits significantly from the re-clustering process, showing a much sharper distinction between clusters. This suggests that the data embedding inherently supports meaningful segmentation, even in this specific scenario, as indicated by the purity score raised to 1. The improved ability of the re-clustered model to capture subtle differences is largely due to the reduced dataset size, which simplifies intersections between elements and makes rule-driven distinctions more apparent.

4.4 Conclusive Remarks

Considering the obtained results and expert feedback, it is evident that the proposed recommendation system effectively fulfills its intended objectives. The findings demonstrate that the UMAP-based method successfully captures the spatial distribution of data, enabling an intuitive and structured clustering of similar elements within a low-dimensional space. This allows for a meaningful identification of correlated outliers, surfacing patterns and anomalies that might otherwise remain undetected.

A key advantage of this approach lies in its ability to address the limitations of traditional ranking-based dashboards, which were discussed in the previous chapter. Given the high volume of data processed, the ranking system may struggle to highlight each relevant outlier, especially those that are neither recent nor highly ranked. As a result, significant but less obvious anomalies may be overlooked.

While homepage-ranked data provide a useful starting point for detecting anomalies, deeper investigation is required. The proposed similarity-based recommendation system enhances this analytical process by allowing users to detect, compare, and investigate related elements, facilitating their understanding. This ensures that important insights are not lost due to ranking biases and empowers decision-makers with a broader, more contextualized view of anomalies within the dataset.

Chapter 5

Personalized Recommendations

This section explores a different approach to recommendations, shifting the focus from finding similar outliers, as discussed in the previous chapter, to personalized suggestions tailored to the user's interests. This method aims to function as a "Recommended for You" feature, similar to those seen on streaming platforms, where suggested items are based on past interactions with the system.

Given that the system serves a diverse user base, each with specific business needs and analytical goals, this tool is designed to adapt to individual preferences. It will be seamlessly integrated into the homepage as a side panel, complementing the ranking displayed on the main dashboard. With this addition, users can discover targeted items that align with their interests while still benefiting from the broader ranking mechanism.

By incorporating personalized recommendations, this tool enhances user engagement and efficiency, helping analysts quickly identify relevant outliers without needing to manually sift through large datasets. This not only streamlines the investigative process but also provides a sense of personalization, making the platform more intuitive and user-friendly.

Since the system does not currently store chronological data of user interactions or preferences, a synthetic dataset is used to train the model. This phase serves as an exploratory stage to assess the feasibility and

the expected effects of implementing personalized recommendations in the actual system.

The primary goal is to validate the methodology before committing to a full-scale implementation. If successful, the next step will involve developing an infrastructure for data collection, enabling real-time tracking of user interactions.

In Chapter 2, we outlined the typical stages of a recommendation system: candidate generation, scoring, and re-ranking. For this experiment, the initial candidate generation phase is deliberately bypassed to focus on the effectiveness of the scoring and ranking processes in proposing relevant elements. This decision allows for a more focused evaluation of the model's ability to assess and rank items based on user interaction history. However, the candidate generation step remains a modular component that can be seamlessly integrated at a later stage, enhancing the system's overall pipeline without requiring fundamental changes to the existing framework.

5.1 System Description

Given the sparsity of the dataset, which consists of many outliers but relatively few interactions, user interests are modeled based on viewed (clicked) elements. This approach is commonly used in recommendation systems, particularly when explicit feedback are limited. Engagement signals serve as implicit indicators of preference [29].

At this stage, user interests are simply defined by past interactions. From these interactions, additional informative features can be extracted, such as topics of the viewed items and distribution of interactions over time.

Therefore, the task objective is to estimate the probability of a user engaging with a particular item based on their specific interaction history. This enables the system to recommend relevant outliers that align

with individual user preferences, enhancing their discovery process. The model treats the problem as a binary classification task, estimating the compatibility likelihood between a user history and one outlier item. Figure 5.1 explains graphically the architecture.

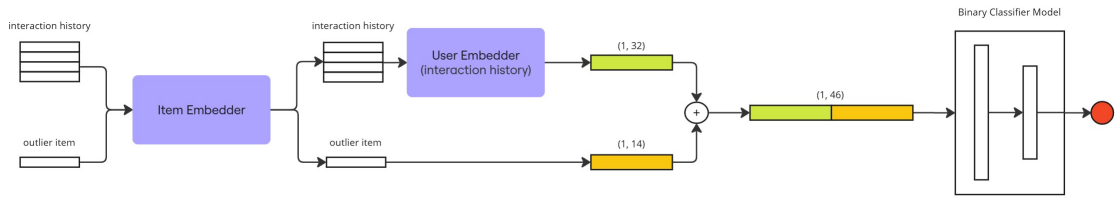

Figure 5.1: Personalized Recommendation System - model architecture

The model has three main components: *item embedder*, *user-history embedder* and the final *binary classifier* combining user history and items.

5.1.1 Item Embedding

This component is essential for embedding input outliers into a suitable vector space. The natural choice was to leverage the UMAP-based encoding, which has already demonstrated strong performance in the clustering scenario (as discussed in Chapter 4).

Each outlier processed by this pipeline is embedded using UMAP, mapping it into a 14-dimensional Euclidean space. This step is indeed useful for mapping a multivariate mixed dataset with both numerical and categorical features into a cohesive numerical representation. For a detailed explanation of the UMAP encoding process, refer to the previous chapter 4.1.2.

5.1.2 User History Embedding

This embedder is the most critical component among the two, as it is responsible for creating a meaningful representation of the user's history.

The model used for this task is an autoencoder, enhanced with specific mechanisms to explicitly handle recurrent interests and outlier recency 5.2.

The autoencoder was chosen among the plethora of different embedding models due to its ability to learn compact and robust embeddings, ensuring that user interaction patterns are captured effectively even in an unsupervised scenario.

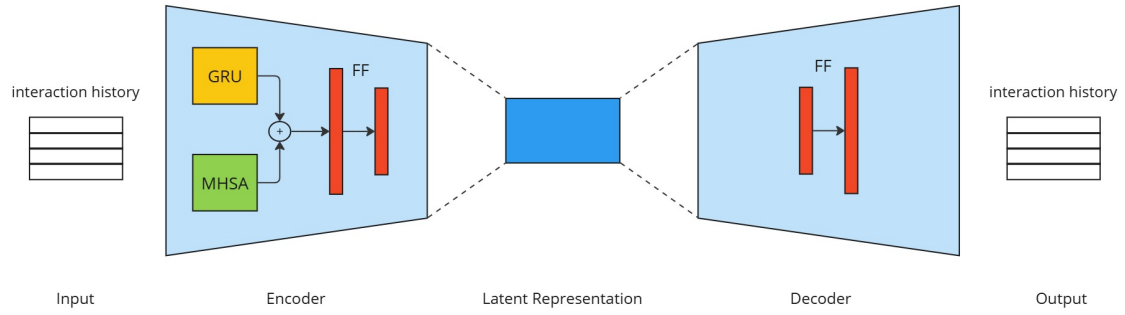

Figure 5.2: User History embedding model - autoencoder

Autoencoder An autoencoder architecture provides an efficient framework for generating low-dimensional embeddings through unsupervised dimensionality reduction by projecting high-dimensional input data into a latent vector space. The model is divided into a two-stage process: encoding and decoding.

The encoder component performs a non-linear transformation $\varphi : \mathcal{R}^D \to \mathcal{R}^d$ (where $d \ll D$) that maps the input to a compressed representation z in the latent space. This mapping optimizes the preservation of essential structural and semantic information while discarding redundant features.

The decoder subsequently attempts to reconstruct the original input from this compressed representation through an inverse mapping $\psi : \mathcal{R}^d \to \mathcal{R}^D$.

Training is driven by minimizing a reconstruction loss $L(x, \psi(\varphi(x)))$, implemented as mean squared error, which ensures that z preserves critical information necessary for accurate reconstruction.

The resulting embeddings z inhabit a continuous, structured manifold that captures disentangled and data variation, making them suitable for downstream tasks such as clustering, visualization, or as input to supervised models.

To make the model embed also information regarding the ordering of the items within the chronological history and the importance of some elements over others, two additional aspects are added to the embedding module.

Gated Recurrent Unit - GRU The Gated Recurrent Unit (GRU) is a recurrent neural network (RNN) variant specifically designed to address the challenges of modeling temporal dependencies in sequential data. In this scenario, the network is used as a component inside the encoder pipeline to model temporal dependencies.

Unlike traditional RNNs, which struggle with vanishing gradients and limited memory over long sequences, GRUs introduce gating mechanisms to dynamically regulate information flow across time steps.

The update gate z_t and the reset gate r_t enable the GRU to selectively retain or discard information from previous states, allowing it to capture both short-term and long-term dependencies effectively.

The update gate z_t determines the balance between preserving the previous hidden state h_{t-1} and incorporating new information from the current input x_t, while the reset gate r_t controls the influence of h_{t-1} on the candidate hidden state.

GRU has been chosen over Long Short-Term Memory (LSTM), another popular gated architecture, for its simpler structure with fewer gates, making it easier to train and faster to run compared to LSTMs [30].

Multi Head Self Attention Self-attention initially introduced by Vaswani in "Attention is all you need" [31], is commonly used in various AI models, one over all: transformers. This mechanism allows each element in

a sequence to attend to all other elements, capturing dependencies and relationships regardless of their distance in the sequence. Unlike GRUs, which process information sequentially, self-attention operates on the entire sequence simultaneously.

Given an input sequence represented as a matrix $X \in \mathcal{R}^{n \times d}$ (where n is the sequence length and d is the feature dimension), the model first linearly transforms X into three different representations: Query matrix (Q), Key matrix (K) and Value matrix (V), computed using learnable weight matrices.

The attention scores between tokens are computed using scaled dot-product attention:

$$Attention(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Here, QK^T calculates the similarity between each token's query and all keys, and softmax normalizes these scores to determine how much attention each token should pay to others. The scaling factor $\sqrt{d_k}$ prevents extreme values, improving stability.

Instead of relying on a single attention operation, Multi-Head Self-Attention splits the input into multiple subspaces and applies multiple self-attention layers in parallel. If there are h attention heads, each computes independent attention using different projection matrices, resulting in multiple attention outputs:

$$\text{head}_i = \text{Attention}(Q_i, K_i, V_i)$$

where i represents different heads. The independent attention outputs are then concatenated to produce the final representation.

In the context of outlier's interaction histories, self-attention allows the model to directly compare any inspected item with any other within the sequence, regardless of when they occurred. This creates rich representations that capture complex relationships between items.

By combining these approaches, the embedder wants to capture both: how items relate to each other and how preferences evolve over time. The combination of these factors then undergoes standard linear layer performing dimensionality reduction. Figure 5.2.

5.1.3 Binary Classifier

The recommendation task is formulated as a binary classification problem [32][33].
Given a user's interaction history H and a candidate outlier I, the model predicts the likelihood that the user would find I relevant.

The model gets as input a concatenated vector $[H; I]$, where H represents the user history embedding, dynamically updated as the user interacts with the system, and I is the outlier embedding obtained through UMAP.

A neural network processes this input, with a final sigmoid activation function $S(x) = \frac{1}{1+e^{-x}}$ responsible for converting the network output into a probability score between 0 and 1. High values indicate strong relevance, while lower values suggest the item is unlikely to be of interest.

The architecture consists of four fully connected layers that progressively reduce the input dimensionality, allowing the model to extract meaningful patterns while preventing overfitting. These layers are interleaved with normalization and dropout layers, ensuring stable training and improving generalization.

The model parameters are optimized to minimize a Binary Cross-Entropy loss function, which minimizes the difference between the predicted probability and the actual user engagement labels.

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)]$$

Where n is the number of observations, y_i is the actual binary label

$(0, 1)$ of the $i - th$ observation and p_i is the probability of the $i - th$ observation to be in class 1.

Since the user history embedding evolves over time, this approach enables the model to adapt its predictions to shifting interests, and ensures that recommendations remain relevant as new interactions occur.

5.2 Dataset

As stated in the introduction of this chapter, no real user interaction data are available for this task, necessitating the creation of a synthetic dataset to simulate a realistic scenario of users engaging with the system.

5.2.1 User-History Generation

The most critical aspect of this pipeline is ensuring that user interaction histories are coherent and reflect actual usage patterns. Each user history is defined as the last n user's interactions, representing the most recently viewed items. A naive approach would involve selecting items at random, but this method fails to capture the structured nature of real interactions.

In real scenarios, users tend to focus on a restricted set of topics based on their specific expertise or administrative responsibilities, making a completely random selection unrealistic. To address this, user histories are generated using insights from the previous clustering techniques 4. The process begins by selecting a random item i from the dataset, identifying its corresponding cluster c_i, and performing a k-nearest neighbors search starting from i. A set of k elements is selected from its closest neighbors, and the process is repeated iteratively for these new items. One important aspect is ensuring that all selected elements remain within the cluster boundaries. If a cluster lacks sufficient items, a new random element is chosen from another cluster.

This method preserves history coherence, following a similarity-driven logic that aligns with real user behavior. The number of covered topics remains limited through controlled cluster membership.

5.2.2 Users Definition and Generation

To ensure effective performance tracking in this synthetic and unsupervised setting, the binary classifier has been trained on a restricted number of users. This allows for better monitoring of model behavior and evaluation of its capability to generalize across different user profiles. Three distinct user types, or personas, are defined to reflect real-world interactions with the platform: *administrative-oriented*, *medical-oriented*, and *economic-oriented* users. Each encapsulates a unique set of interests.

For each persona type, three users are generated and carefully supervised to avoid inconsistencies.

Administrative-oriented users primarily focus on the number of accesses as a key measure, and dimensions such as facility codes, priority access codes, service codes, and discharge modalities. User may focus on a specific hospital, or maintain a broader field of view.

Economic-oriented users, on the other hand, refer to financial metrics, such as gross and net amounts, regional taxations, and variations across specific time periods, often aligning with budgetary and regulatory constraints.

Medical-oriented users exhibit a keen interest in data related to hospitals, doctors, services, and medical examinations, ensuring that insights are tailored to healthcare-specific analytical needs.

Once these personas are defined, user generation followed a structured approach. Each user is defined by its interaction history, meaning that generating a new user equals to constructing an interaction sequence constrained by predefined topics and content categories. This method ensures that the synthetic dataset accurately models real-world patterns.

5.2.3 Recommendation Dataset

For each of the nine users defined in Section 5.2.2, a set of 140 relevant items is generated as described in 5.2.1. These items serve as the basis for constructing the training, validation, and test datasets for the binary classification task. Specifically, the first 100 interactions are allocated to the training set, while the most recent 40 interactions are divided between the validation set (20 items) and the test set (20 items).

To define each user, a history of past interactions is required. In this framework, the user history is fixed at a length of 30 items. Consequently, within the 100 training set items, the first 30 interactions establish the initial user history, while the remaining 70 serve as positive item candidates. A sliding window approach is employed, where the dynamic user history consists of the most recent 30 interactions, and the subsequent item in the sequence is designated as the candidate outlier.

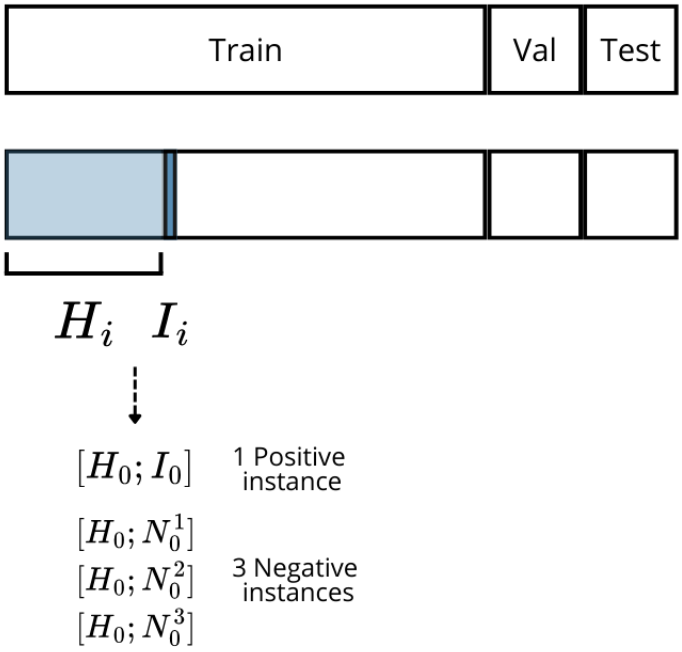

Figure 5.3: Dataset creation for Personalized Recommendations

Given the current user-history H_0 and the candidate item I_0 the coupling $[H_0; I_0]$ represents a positive instance. Since the recommendation task is framed as a classification problem, the model also requires neg-

ative samples for training, representing unrelated items.
Negative instances are generated by randomly selecting items from the pool of non-relevant items (i.e., the dataset excluding the 140 initially selected items). Consequently, negative samples $[H_0; N_0^1]$, $[H_0; N_0^2]$, $[H_0; N_0^3]$ are generated to balance the training process.
The class distribution follows a 1:3 ratio, with negative samples outweighing positive ones. This weighting reflects the natural user interaction flow within the platform, where only a fraction of the displayed items receive actual engagement. The ratio remains balanced enough to prevent severe class imbalance, avoiding exaggerated discrepancies that could lead to overfitting.

At this stage, the history window shifts forward by one position, including the newly observed item I_0. This process is iteratively repeated until all the elements in the training set have been processed. As a result, the final training dataset consists of 630 positive and 1890 negative instances.

The last 40 selected items are divided into two sets of 20 each, forming the validation and test sets. These sets are constructed similarly to the training set, with one key difference: the user history is not generated from scratch but instead evolves from the history established during training.

This approach ensures that the test instances share partially the history with the training instances, mirroring real-world scenarios where user interactions continue to evolve between consecutive model re-trainings in a production environment.

5.3 Experimental Setup

5.3.1 User-History Embedder

The experiment focuses on training the autoencoder model responsible for user-history embedding. The objective is to develop a model capable

of learning a latent representation of a user's interaction history, compressing a variable-length sequence into a single 32-dimensional vector. The training dataset consists of 250 batches, each containing 32 user histories, with each history comprising 30 outlier items. This results in a dataset with a final shape of $[250, 32, 30]$. Each user-history is generated as indicated in the previous section 5.2.1, but without any further supervision.

The model was trained until the loss function, measured by mean squared error (MSE), converged. The training process stabilized after approximately 120 epochs with a learning rate of 0.001.

For what concerns the internal layers, the hyperparameter setup is described in Table 5.1.

MHSA	
Number of heads	2
Dropout	0.05
GRU	
Hidden dimension	64
Number of layers	4
Dropout	0.05
FF	
Latent dimension	(1,32)

Table 5.1: Hyperparameters for User-History embedding model - autoencoder

Figure 5.4 illustrates the loss improvement throughout training, reaching a minimum value of 6.62. This result represents the best configuration obtained after a random search over different hyperparameter settings. The model is trained on a diverse set of user histories, enabling it to effectively learn how to compress interaction sequences into meaningful latent representations.

The nine predefined personas are used as a reference to assess the quality of these embeddings. Ideally, the best embedding configuration should ensure that similar personas produce embeddings that are closer to each other, while those with different interests remain more distinct. This evaluation is conducted using cosine similarity, a widely

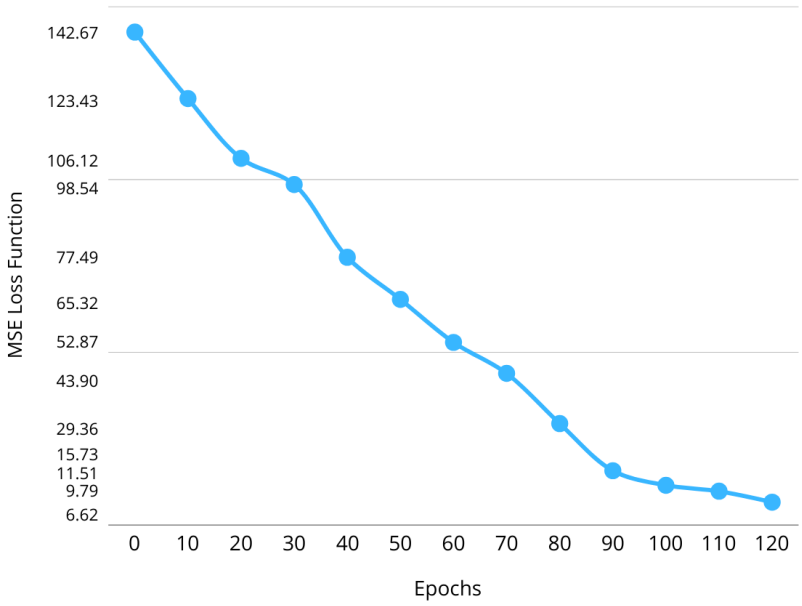

Figure 5.4: MSE loss for training User-History Embedder

used metric for comparing high-dimensional vectors. Cosine similarity measures the cosine of the angle between two vectors. A value of 1 indicates perfect alignment (high similarity) and a value of 0 indicates complete divergence. Analyzing these similarities, it is possible to determine whether the model successfully captures the underlying patterns within user interaction histories.

The upper triangular matrix in Figure 5.5 summarizes these distances. It is evident that users with similar histories, particularly those focused on administrative and economic topics, exhibit strong similarities in their history embeddings. For most users within these categories, the learned embeddings correctly cluster together, reflecting their shared interests.

However, for users categorized under medical personas, the expected similarity pattern is less clearly defined. In several cases, embeddings for medical-oriented users are distant from each other. Sometimes those share even higher similarity with administrative or economic personas than with medical ones. A notable example is *"medical_03"*, whose embedding appears more aligned with users from the administrative

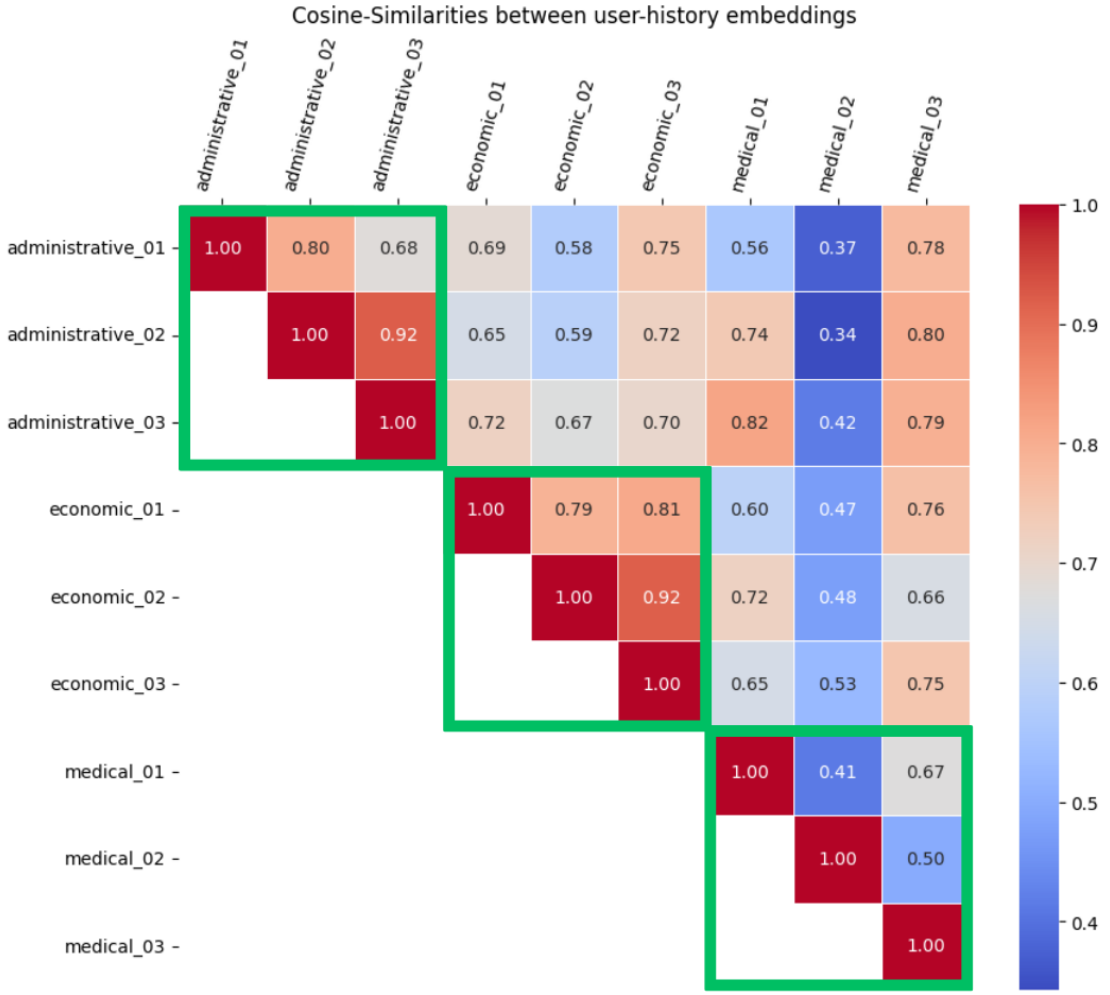

Figure 5.5: Cosine similarity among computed User-History Embeddings

and economic categories rather than with other medical-focused users.

A possible explanation for this discrepancy is that medical-oriented outliers belong to a broader and more diverse underlying clustering structure, leading to greater dispersion in the vector space. Unlike administrative or economic-oriented outliers, which tend to form more compact and well-defined groups because of their restricted features variability (limited dimensions and restricted range of possible measures), medical-related cases may exhibit a far wider spectrum of feature values. As a result, different users with medical interests may assume embeddings that are not necessarily close to each other, despite sharing a general thematic focus.

5.3.2 Binary Classifier Recommender

The binary classifier model serves as the core of the recommendation system. To better assess its performance in this synthetic setting, it is trained on a limited set of users, allowing for a controlled evaluation of its behavior. Specifically, the training set is composed of three users per persona type, resulting in a total of nine users. 2520 pairs of user-history, item $[H; I]$ are utilized. This setup ensures a balanced representation of different interaction patterns across personas. The finalized hyperparameter configuration is expressed in Table 5.2.

Binary Classifier	
Number of layers	4
Dropout	0.1
Batch size	64
Learning rate	0.001

Table 5.2: Hyperparameters for final Binary Recommender Classifier model

To mitigate overfitting, the training process is guided by the validation loss forcing the training to stop after *10 epochs*, to prevent excessive memorization of the training data. The loss function showed a noticeable decrease in the first epochs but exhibited minimal improvement in the latest iterations, Figure 5.6.

To evaluate the model's performance, standard classification metrics are computed: accuracy, precision, recall, and F1 score (Table 5.3). These metrics are calculated using a macro-average approach, which provides an equal-weighted comparison between classes without being influenced by class imbalances. This approach results in a stricter evaluation, ensuring that both positive and negative predictions are considered with equal importance.

The model demonstrated strong performance during testing, Table 5.3, successfully distinguishing between relevant and irrelevant items for all three user personas, Table 5.4. Overall, the model achieved an F1 score of 0.875 on the test set.

The primary challenge during evaluation was correctly identifying

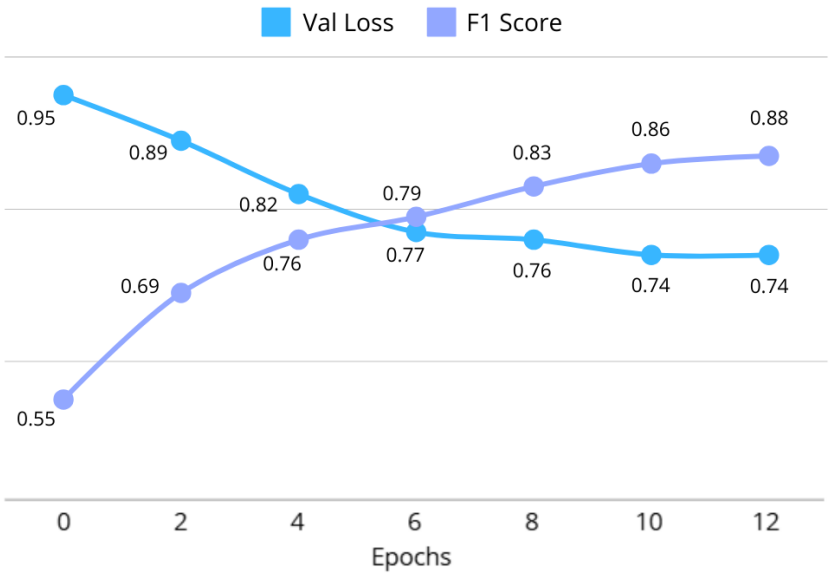

Figure 5.6: Validation Loss and F1 Score over epochs

Metric	Overall	Positive	Negative
Accuracy	0.905	0.905	0.905
Precision	0.871	0.800	0.942
Recall	0.879	0.828	0.930
F1 Score	0.875	0.814	0.936

Table 5.3: Binary Classifier performance metrics over test set: overall (average macro), and per class

true positive labels. A detailed analysis of per-label performance revealed a notable disparity: the F1 score for positive-labeled items was significantly lower at 0.814, while the negative class achieved a significantly higher score of 0.936. This slight unbalance was expected due to the different ratio in the training set, weighting more the negative class.

However, this limitation is of minor concern given the underlying business objective. The primary aim is to prevent the proposal of incoherent outliers, rather than strictly identifying the most optimal one. Determining what constitutes a relevant proposal is inherently subjective, as user interests and priorities can vary significantly. Conversely, identifying irrelevant elements is a more straightforward task, as these are typically characterized by a clear lack of alignment with the user's past interactions and preferences.

	Administrative		**Economic**		**Medical**	
	Positive	***Negative***	***Positive***	***Negative***	***Positive***	***Negative***
Accuracy	0.933	0.933	0.908	0.908	0.895	0.895
Precision	0.797	0.993	0.880	0.915	0.753	0.953
Recall	0.983	0.916	0.733	0.966	0.866	0.905
F1 Score	0.880	0.953	0.800	0.940	0.806	0.928

Table 5.4: Metrics for each persona in the test set, computed per class.

Table 5.4 analyzes the performance across different personas, revealing the overall model strength. The user with administrative interests is the best performing one, reflecting the strong embedding quality for this user type as seen in Figure 5.5. This result aligns with the well-clustered nature of administrative user histories, where interactions tend to be tightly grouped. All users in fact focus on elements related to the same measure: "number of access." Despite variations in dimensions and time periods, the model successfully identifies the underlying similarities between these outliers, contributing to its strong performance.

In contrast, the economic and medical-oriented users exhibit slightly less consistent performance, especially for recall and F1 score. Since these personas share by design fewer similarities among each other, their embeddings result more sparse compared to those of administrative users, Figure 5.5. As a consequence, the model faces greater difficulties in establishing clear classification boundaries. Despite these variations, the overall metrics remain encouraging, demonstrating the potential for a coherent recommendation system. In this setting, the business objective is not to maximize metrics, but to propose coherent elements: thus avoid proposing unrelated outliers. Thanks to the high performance obtained for the negative instances, it can be affirmed that only relevant and compliant outliers would be proposed at recommendation time.

5.3.3 Recommendation Results

Since no restricted candidate pool is available, the recommender system evaluates all items in the dataset (except for those the user has already

interacted with). While this approach is not ideal for a fully deployed system, it serves well for these initial research stages.

When generating the top k recommendations for a user, the model assigns to each candidate item a likelihood score reflecting the relevance of the pairing between the user-history embedding and the candidate item.

The results are then ranked in descending order, with the top five highest-scoring items selected as personalized recommendations, ensuring that only the most relevant suggestions are presented to the user. Follow a set of examples to visually assess the model potentials.

Figure 5.7 and 5.8 present the proposed recommendations for two test users. At the top of each example, an extract of the user's actual history is displayed, while the bottom section contains the five recommendations generated by the model.

Administrative user-history

	analysis_type	dimensions_keys	dimensions_vals	measures	values	reference_values	outlier_score	time
1	contestuali	cod_struttura_ero	30212D	num_accessi	0	212	38.97	2024-08-01
2	contestuali	cod_struttura_ero	21011A	num_accessi	186	4647	33.30	2024-08-01
3	contestuali	cod_struttura_ero	24911A	num_accessi	3	77	38.15	2024-08-01
4	contestuali	cod_struttura_ero	24821A	num_accessi	111	205	37.70	2024-08-01
5	contestuali	cod_struttura_ero	41251A	num_accessi	863	3713	43.93	2024-08-01
...	...	...	...	...	...	...	...	...
18	contestuali	cod_struttura_ero	11162D	num_accessi	156	14	11.05	2024-08-01
19	contestuali	cod_struttura_ero	22821B	num_accessi	339	1204	38.72	2024-08-01
20	contestuali	cod_struttura_ero	21012D	num_accessi	4717	21	78.26	2024-08-01
...	...	...	...	...	...	...	...	...
28	contestuali	cod_struttura_ero	11081A	num_accessi	238.0	0.0	73.33	2024-07-01
29	contestuali	cod_struttura_ero	41241A	num_accessi	822	431	12.82	2024-07-01
30	contestuali	cod_struttura_ero	41211A	num_accessi	0.0	214.12	23.71	2024-07-01

Recommendation Proposals

	analysis_type	dimensions_keys	dimensions_vals	measures	values	reference_values	outlier_score	time
1	contestuali	cod_struttura_ero	21311A	num_accessi	1263	358	57.83	2024-08-01
2	contestuali	cod_struttura_ero	02321A	num_accessi	98	761	33.30	2024-08-01
3	contestuali	cod_struttura_ero	41061A	num_accessi	2118	722	56.85	2024-08-01
4	contestuali	cod_struttura_ero	G1011A	num_accessi	800	390	37.70	2024-07-01
5	contestuali	cod_struttura_ero	21242D	num_accessi	0	402	25.83	2024-07-01

Figure 5.7: Administrative-oriented user: personalized recommendations output proposal

For the administrative user (Figure 5.7), the history consists of out-

liers all related to the same topic: the number of accesses to a specific facility within a one-month period. Notably, the user inspects these elements in chronological order, with the most recent data appearing last. The outlier scores of these elements are relatively high, suggesting that the user's focus is directed toward the most critical cases flagged by the system.

The generated recommendations align well with the user's past interactions, as the proposed elements follow the same structure, reflecting monthly access data for a specific facility. Additionally, they pertain to the same time periods as previously viewed items, revealing the model's ability to identify temporal patterns coherently.

Medical user-history

	analysis_type	dimensions_keys	dimensions_vals	measures	values	reference_values	outlier_score	time
1	contestuali	cod_medico_base | cod_modalita_dim	XXX | 02	num_ricovero	22	6	74.54	2024-07-24
2	puntuali	cod_modalita_dim | cod_tipo_drg	02 | C	gg_postoperatorie	366	2	74.38	2024-03-01
3	puntuali	cod_azienda_res | cod_prestazione	24911A | G239	gg_postoperatorie	197	0	74.23	2024-07-24
4	contestuali	cod_reparto_dim | cod_presidio_ero	4351 | 09064101	num_ricovero	73	47	73.90	2024-07-24
5	contestuali	cod_diagnosi_pri | cod_drg_reg	4870__SEP__089	num_ricovero	21	0	73.79	2024-05-01
...	...	...	...	...	...	...	...	...
18	contestuali	cod_azienda_res | cod_prestazione	11162D | A142E	num_ricovero	270	142	72.22	2024-07-24
19	contestuali	cod_cittadinanza | cod_medico_base	100 | XXX	num_ricovero	21	0	72.19	2024-07-24
20	contestuali	cod_diagnosi_pri | cod_ambito	4870 | 090109	num_ricovero	17	0	72.17	2024-01-24
...	...	...	...	...	...	...	...	...
28	contestuali	cod_medico_base | cod_regime_ric	XXX | 1	num_ricovero	18	2	68.10	2024-01-24
29	puntuali	cod_ambito | cod_tipo_ric	090108 | 5	gg_postoperatorie	366	5	68.06	2024-07-01
30	contestuali	cod_medico_base | cod_tipo_prestazione	XXX | F	num_ricovero	31	3	67.53	2024-01-24

Recommendation Proposals

	analysis_type	dimensions_keys	dimensions_vals	measures	values	reference_values	outlier_score	time
1	contestuali	cod_medico_base | cod_tipo_ricetta	XXX | A	num_ricovero	17	2	73.96	2024-07-24
2	contestuali	cod_medico_base | cod_tipo_prestazione	XXX | G45	num_ricovero	2	12	74.40	2024-03-01
3	puntuali	cod_struttura_ero | cod_tipo_servizio	45042A | A59	gg_postoperatorie	12	0	72.84	2024-08-01
4	contestuali	cod_intervento_pri | cod_tipo_ric	6952 | 5	imp_regionale	14343.0	4500.0	79.12	2024-07-01
5	puntuali	cod_classe_pri | cod_dimissione	A | XXX	gg_preoperatorie	197	0	86.90	2024-07-01

Figure 5.8: Medical-oriented user: personalized recommendations output proposal

Figure 5.8, on the other hand, presents an example for a medical user, whose interests, based on its history, are more diverse and less sharply defined than in the previous case. This user focuses on different aspects of the medical field, particularly the monthly number of hospitalizations

and the number of hospitalization days before and after surgery. Unlike the administrative user, this one does not prioritize recent outliers but rather the most severe ones. The chronological ordering of inspected items follows the decreasing value of the feature outlier score, indicating a strong interest in cases with high variance compared to usual patterns.

Generating relevant recommendations in this scenario is inherently more complex due to the greater variability in item topics. Despite this challenge, the model successfully provides a coherent set of outliers, with only minor biases. Specifically, the recommended items align with the investigated measures, except for one instance where the model suggests an item related to the amount owed by the region for a specific priority service (never seen during past interaction). The dimensions covered in the recommendations are more sparse due to the broader scope of analysis, yet they remain generally aligned with the user's areas of interest. Looking at the outlier score in more detail, it can be noticed that it closely follows the user's previous exploration patterns. Notably, the first proposed items relate to hospitalization numbers: the most frequently occurring outliers in the user history. This highlights the model's ability to recognize recurrent patterns, beyond merely temporal trends, revealing its strength in identifying meaningful recommendations even in highly variable contexts.

The model performs in line with initial expectations, effectively identifying and suggesting items that could be of interest to a particular user. As mentioned at the beginning of the chapter, this section serves as a support tool during the exploration phase, assisting users in their research and facilitating the discovery of new items to inspect. Rather than replacing traditional search bars or filtering systems, this recommender system functions as an extension of these tools, offering personalized suggestions that complement existing search methods and help users navigate large datasets more efficiently.

Chapter 6

Conclusion and Future Work

This thesis explored how different recommendation frameworks can be integrated into a software system for outlier analysis on healthcare data. The objective was to enhance user experience, facilitate research and analysis processes, and improve the accessibility of critical insights. The overall results are positive, with each proposed solution addressing a specific aspect of usability and interaction within the system.

6.1 Key Contributions

Rank Aggregation The Rank Aggregation phase was introduced to improve the ordering of elements displayed on the homepage. Compared to the baseline approach, which relied on Euclidean distances, this method provided a more structured and meaningful ranking that aligned better with the generalized guidelines established by stakeholders. The new approach successfully overcame technical challenges encountered in the first iteration, particularly the need to define a target element, which previously complicated ranking consistency. By leveraging rank aggregation, the system produces a more interpretable and user-friendly ordering of items, ensuring that critical information is prioritized effectively.

Similar Products Recommendations The Similar Product Recommender was developed to assist users during the outlier's analysis phase by suggesting elements related to the one under examination. The main goal of this feature is to support users in discovering relevant insights, identifying patterns, and uncovering new data points that could enrich their ongoing investigation. This system is built on a two-stage pipeline:

Clustering Phase: this step groups outliers based on their defining features, such as the type of anomaly, the nature of the analysis, and the characteristics of the data signal (e.g., intensity and deviation from standard values). This ensures that similar anomalies are categorized together, forming coherent clusters that aid in candidate generation.

Ranking Phase: this step is meant to perform a ranking of the relevant set of candidates identified for a given query outlier. This is addressed with a nearest neighbor algorithm approach that utilizes Euclidean distance in the embedded space.

The results demonstrated the effectiveness of this method, as it consistently produced coherent and useful recommendations. The success of this approach also validated the use of UMAP embeddings, which proved to be a strong choice for representing outlier characteristics in a lower-dimensional space while preserving meaningful relationships between data points.

Personalized Recommendations The final and most complex task involved developing a personalized recommendation system based on user interactions within the platform. The goal was to implement a feature similar to the "Recommended for You" section found in streaming services, where recommendations are tailored to individual users based on their past interactions. Given the diverse range of users engaging with the software, each with different interests and needs, this functionality is designed to complement traditional content search, assisting users in discovering new, yet relevant data points.

One of the main challenges of this phase was the absence of real user

interaction data. To address this, a synthetic dataset was generated to simulate realistic user behaviors.

The recommendation model was designed as a binary classifier, distinguishing between relevant and non-relevant items based on historical user interactions. A key component of this system was the autoencoder-based user-history embedder, which compresses a user's chronological interaction history into a lower-dimensional representation. Instead of treating users as static entities, the model focuses on their evolving history, allowing it to generalize across different users with similar exploration patterns.

The evaluation of this system showed promising results, demonstrating that the model was able to generate relevant recommendations aligned with past user interactions. While some challenges remain, particularly regarding generalization due to the limited training data, the approach successfully captured patterns in user behavior, proving to be a viable direction for further refinement.

6.2 Final Thoughts and Future Work

This research has demonstrated that integrating recommendation techniques into healthcare outlier analysis software can significantly enhance usability and decision-making. Each of the three proposed solutions: rank aggregation, similar product recommendation, and user-based recommendations, plays a distinct role in making the system more intuitive and user-friendly.

To ensure these improvements translate into meaningful user engagement and improved satisfaction, A/B testing should be employed as a critical evaluation method. Direct testing with real users is indeed essential to measure the impact of each approach on user interactions, task efficiency, and overall system satisfaction. Key performance indicators (KPIs) such as click-through rate (CTR), time spent on recommended

items, and user engagement serve as valuable indicators of user appreciation and can help refine the system to better meet user needs.

Beyond evaluation, future enhancements could focus on:

- Optimizing candidate generation in the similar product recommendation task to improve computational efficiency and scalability.
- Enhancing outlier embeddings by experimenting with alternative techniques to refine clustering and similarity calculations.
- Expanding the training dataset in the personalized recommendation task by incorporating real user interactions, allowing the model to be refined and validated in a real-world setting.
- Improving user embeddings by integrating alongside implicit chronological interactions, also explicit user feedback, such as: likes, saved items, and stated topic preferences. This would create a more nuanced representation of user interests, helping the model distinguish between exploratory interactions and genuinely valuable items.
- Enhancing recommendation diversity to ensure users receive not only similar recommendations but also complementary insights aligned with their explicit interests.

This research highlights the potential of integrating recommendation systems into healthcare outlier analysis, offering a powerful tool to support data-driven decision-making. By enhancing usability and facilitating more intuitive exploration of complex datasets, the proposed system can assist healthcare professionals in identifying critical patterns, anomalies, and trends more efficiently. The ability to personalize recommendations based on user interactions and preferences ensures that analysts receive relevant insights tailored to their specific areas of interest. By bridging advanced recommendation techniques with healthcare analytics, this work lays the foundation for a smarter, more adaptive system capable of enhancing both efficiency and accuracy in medical data analysis.

Bibliography

[1] Rui Chen, Qingyi Hua, Yan-Shuo Chang, Bo Wang, Lei Zhang, and Xiangjie Kong. A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE Access*, 6:64301–64320, 2018.

[2] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008.

[3] Sean J. Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[4] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. Stl: A seasonal-trend decomposition procedure based on loess (with discussion). *Journal of Official Statistics*, 6:3–73, 1990.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009.

[6] Muhammad Ali Norozi and Paavo Arvola. Selection fusion in semi-structured retrieval. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, CIKM '13, page 1291–1300, New York, NY, USA, 2013. Association for Computing Machinery.

[7] Javed A. Aslam and Mark Montague. Models for metasearch. In *Proceedings of the 24th Annual International ACM SIGIR Confer-*

ence on Research and Development in Information Retrieval, SIGIR '01, page 276–284, New York, NY, USA, 2001. Association for Computing Machinery.

[8] Mark Montague and Javed A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, page 538–548, New York, NY, USA, 2002. Association for Computing Machinery.

[9] Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *Text Retrieval Conference*, 1993.

[10] Michael J. Pazzani and Daniel Billsus. *Content-Based Recommendation Systems*, pages 325–341. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[11] Mingsheng Fu, Hong Qu, Zhang Yi, Li Lu, and Yongsheng Liu. A novel deep learning-based collaborative filtering model for recommendation system. *IEEE Transactions on Cybernetics*, 49(3):1084–1096, 2019.

[12] Xin Dong, Lei Yu, Zhonghuo Wu, Yuxia Sun, Lingfeng Yuan, and Fangxi Zhang. A hybrid collaborative filtering model with deep structure for recommender systems. In *Proceedings of the AAAI Conference on artificial intelligence*, volume 31-1, 2017.

[13] Amit Kumar Jaiswal. Towards a theoretical understanding of two-stage recommender systems, 2024.

[14] Avradeep Bhowmik and Joydeep Ghosh. Letor methods for unsupervised rank aggregation. In *Proceedings of the 26th International Conference on World Wide Web*, page 1331–1340, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee.

[15] Sikri A., Singh N.P., and Dalal S. Analysis of rank aggregation techniques for rank based on the feature selection technique, 2023.

[16] Niclas Boehmer, Robert Bredereck, and Dominik Peters. Rank aggregation using scoring rules, 2022.

[17] Joon Ho Lee. Analyses of multiple evidence combination. *SIGIR Forum*, 31(SI):267–276, July 1997.

[18] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[19] Ishrat Zahan and Rehena Nasrin. An introduction to fuzzy topological spaces. *Advances in Pure Mathematics*, 11:483–501, 05 2021.

[20] Jan Niklas Böhm, Philipp Berens, and Dmitry Kobak. Attraction-repulsion spectrum in neighbor embeddings, 2022.

[21] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[22] Adele Jackson. The mathematics of umap, 2019.

[23] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[24] Nikita A. Andriyanov, Alexander Tashlinsky, and Vitaly Dementiev. Detailed clustering based on gaussian mixture models. In *Intelligent Systems with Applications*, 2020.

[25] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM)*, pages 333–344, 2004.

[26] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[27] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[28] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 577–584, 2001.

[29] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. Recommending what video to watch next: a multitask ranking system. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 43–51, New York, NY, USA, 2019. Association for Computing Machinery.

[30] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[32] Tian Wang, Yuri M. Brovman, and Sriganesh Madhvanath. Personalized embedding-based e-commerce recommendations at ebay, 2021.

[33] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th*

ACM Conference on Recommender Systems, RecSys '16, page 191–198, New York, NY, USA, 2016. Association for Computing Machinery.