D5.3 - Trusted and Explainable AI Techniques. II

Title	D5.3 - Trusted and Explainable AI Techniques. II
Revision Number	3.0
Task reference	T5.1 T.52
Lead Beneficiary	IBM
Responsible	Fabiana Fournier, Lior Limonad
Partners	ENG, ATOS, MOH, UNP
Deliverable Type	DEM
Dissemination Level	PU
Due Date	2025-09-30 [Month 33]
Delivered Date	2025-09-12
Internal Reviewers	UBI INNNEUROPE
Quality Assurance	UPRC
Acceptance	Coordinator Accepted
Project Title	FAME - Federated decentralized trusted dAta Marketplace for Embedded finance
Grant Agreement No.	101092639
EC Project Officer	Stefano Bertolo
Programme	HORIZON-CL4-2022-DATA-01-04

This project has received funding from the European Union's Horizon research and innovation programme under Grant Agreement no 101092639

Revision History

Version	Date	Partners	Description
0.1	2025-06-30	IBM	TOC
0.2	2025-07-28	IBM UPRC ATOS	Contribution and updates
0.3	2025-08-01	IBM UPRC ATOS	Updates
0.4	2025-08-15	JSI	Contribution
0.5	2025-08-15	IBM JSI	Updates
1.0	2025-09-10	IBM INNEUROPE	Review from INNEUROPE (BL)
1.1	2025-09-10	IBM UBI	Review from UBI (KP)
2.0	2025-09-10	IBM	Version for QA
3.0	2025-09-12	IBM	Version for submission

Definitions

Acronym	Definition
AAAI	American Association for Artificial Intelligence
AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under the Curve
BPM	Business Process Management
CD	Continuous Development
CDTI	Technological Development and Innovation Centre (from Spanish “Centro para el Desarrollo Tecnológico y la Innovación)
CID	Content IDentifier
CNN	Convolutional Neural Networks
CSV	Comma Separated Value files
DOI	Digital object identifier
DSTI	Debt-to-Service-Income ratio
FAME	Federated decentralized trusted dAta Marketplace for Embedded finance
FML	Federated Machine Learning
IBM	International Business Machines
ID	Identity
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LIME	Local Interpretable Model-agnostic Explanations
LLM	Large language model
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MOH	Motor Oil (Hellas) Diilistiria Korinthou A.E.
PDF	Portable Description Format
PDP	Partial Dependence Plots
PFI	Permutation Feature Importance
PM	Person Month
REST	Representational State Transfer
RF	Radio Frequency
RFM	Recency, Frequency, and Monetary
RMSE	Root Mean Squared Error
SAX	Situation Aware eXplainability
SAX4B	Situation Aware eXplainability for Business Process Management

SHAP	SHapley Additive exPlanations
SME	Small Medium Enterprise
SMOTE	Synthetic Minority Over-sampling Technique
UI	User Interface
XAI	Explainable Artificial Intelligence

Other acronyms and abbreviations not present in the table, are introduced in the text along with their definitions.

Executive Summary

FAME's objective is to develop a marketplace for Embedded Finance (EmFi). A central pillar of this marketplace is the provision of trustworthy AI/ML models with explainability capabilities. Within this context, *Task 5.1 – Catalogue of AI/ML Techniques for EmFi* and *Task 5.2 – Explainable and Trustworthy Artificial Intelligence*, under *WP5 – Trusted and Energy Efficient Analytics*, are responsible for delivering a catalogue of AI/ML models and techniques for Embedded Finance. These serve as the foundation for data analytics and for the specification and implementation of novel eXplainable AI (XAI) techniques.

Deliverable 5.3 – Trusted and Explainable AI Techniques II is the second and final deliverable in this series, covering the work carried out during Months 16–33 of the project, i.e., the advancements achieved since the submission of *D5.1 – Trusted and Explainable AI Techniques I*.

This document details the main functionalities developed under Tasks 5.1 and 5.2, namely: *ML/AI analytics, SAX techniques, the XAI Scoring Framework, and the StreamStory XAI dashboard*. For each functionality, it provides a description of progress, technical specifications, and applicability to the project pilots.

Main achievements include:

- *AI/ML Analytics*: Implementation and deployment of models across six categories: customer segmentation, risk assessment, information extraction, sentiment analysis, anomaly detection, and time-series forecasting. Multiple models were developed and evaluated within each category, tailored to pilot needs while ensuring applicability and reusability.
- *SAX Techniques*: Development and open-source release of the SAX4BPM library for business process explainability, together with a dedicated scale for assessing the quality of explanations by Large Language Models (LLMs) and a benchmark dataset for causal business process reasoning. This foundational research has also resulted in 10 publications in leading conferences, workshops, and journals.
- *XAI Scoring Framework*: Design of a framework for tabular datasets, enabling users to upload datasets and retrieve multidimensional XAI scores.
- *StreamStory XAI Dashboard*: Development of an advanced system that transforms complex industrial sensor data into actionable, LLM-generated insights for predictive maintenance. This includes the StreamStoryPyClient for simplified programmatic interaction and the StreamStory Causal pipeline for preparing time-series data for causal discovery.

The performance metrics achieved in these two tasks exceeded expectations: 20 assets were indexed in the FAME platform for broad exploitation, 21 AI/ML models were developed (against an expected ≥10), and 12 XAI techniques/models were delivered (against an expected ≥8).

Table of Contents

List of Figures

List of Tables

1 Introduction

One of the main pillars of the developed FAME data marketplace is the integration of trusted analytics based on novel Machine Learning (ML) models and technologies such as eXplainable AI (XAI) and Situation Aware eXplainability (SAX). As a result, end users of the platform will be able to deploy and execute specific AI/ML models and enhance them with explainability capabilities. This has been made possible through the work carried out under Task T5.1 – *Catalogue of AI/ML Techniques for EmFi* and Task T5.2 – *Explainable and Trustworthy Artificial Intelligence*.

It should be noted that, as this is a second version of a previous deliverable (*D5.1 – Trusted and Explainable AI Techniques I* 1]), there are some text that has been maintained from that version to ease the readability of the document.

1.1 Objective of the Deliverable

The objective of this deliverable is to document the advancements made to the FAME platform components related to the AI/ML catalogue and the XAI techniques since the submission of the first iteration, *D5.1 – Trusted and Explainable AI Techniques I* [1], at month 15 of the project (M15). More specifically, it describes the models developed within FAME under the AI/ML catalogue (T5.1), the XAI techniques (T5.2), and their mapping to applications in the FAME pilots.

1.2 Insights from other Tasks and Deliverables

As aforementioned, the objective of this deliverable is to document the advancements of Task 5.1 and Task 5.2 since M16 of the project. These tasks provide a catalogue of trained ML models together with XAI and SAX techniques. Although closely intertwined, they also depend on the other tasks of WP5: Task 5.3 – *Incremental Energy Efficient Analytics*, which provides the necessary incremental data management; *Task 5.4 – Edge Data Management and Edge AI Optimization*; and *Task 5.5 – FML for Privacy-Friendly and Energy-Efficient Data Markets*, which enables the smart deployment of models and tools from Tasks 5.1 and 5.2 in an energy-efficient manner. In addition, Tasks 5.1 and 5.2 are connected to the overall FAME platform architecture, as specified in WP2 – *FAME Platform: Specifications, Architecture and Integration* (D2.1–D2.6), and to the pilot use cases, as described in WP6 – *Integration, Validation and Evaluation of EmFi Use Cases* (D6.1, D6.2, D6.4).

1.3 Structure

In general, the structure of this deliverable reflects the advancements achieved since M16 in the main developed functionalities under T5.1 and T5.2, namely ML/AI analytics, SAX techniques, the XAI Scoring Framework, and the StreamStory XAI dashboard. For each functionality, a description of the progress and the technical specifications is provided, along with their applicability to the project pilots. More specifically:

- **Section 1** introduces the deliverable, including its objective, insights from other tasks and deliverables in the project, and a summary of the main achievements since D5.1.
- **Section 2** presents the performance metrics of T5.1 and T5.2 in relation to the indexed assets and key performance indicators specified in the Description of Work (DoA).
- **Section 3** details each of the four developed functionalities, describing the models and capabilities developed along with their technical specifications.
- **Section 4** outlines the applications made available to the pilots for each functionality
- **Section 5** provides the main conclusions.

1.4 Summary of changes from D5.1 - Trusted and Explainable AI Techniques I

Advancements from D5.1 [1] under Tasks T5.1 and T5.2 are summarized in the following high-level bullet points by topic or functionality. Further details are provided in the corresponding sections for each topic.

1.4.1 Machine Leaning (ML) models

Since the publication of Deliverable 5.1 (D5.1) [1], the following new ML model implementations have been developed under Task 5.1, to address the specific needs of the pilots.

- Clustering segmentation using KMeans on Recency, Frequency, and Monetary (RFM) metrics, enabling the grouping of customers into distinct segments to better understand behavioural profiles.
- Risk classification with TabNet[1] for detecting clients at risk of default, incorporating several data processing techniques to address severe class imbalance.
- Information extraction via question-to-value pipeline using FLAN-T5[2] with LoRA[3] adaptation to extract numerical data from complex PDF documents.
- Anomaly detector for temperature sensor data using a convolutional autoencoder, enabling daily abnormal pattern recognition in industrial machinery.
- Forecasting with Long Short-Term Memory (LSTM) - Convolutional Neural Networks (CNN) multivariate models to predict one-week temperature sensor trends from two weeks of historical data.

In addition, dedicated data processing pipelines were implemented, and datasets were generated or synthetically augmented to support model training. All these implementations have been carried out within the scope of Task 5.1.

1.4.2 Situation-Aware eXplainability (SAX)

Since the publication of Deliverable 5.1 (D5.1) [1], the following new SAX techniques implementations have been developed under Task 5.2.

- Release of the SAX4BPM library to the open-source community.
- Scale development and conduction of two surveys to assess the quality of LLM generated explanations.
- Development and release of a benchmark to assess the causal business process reasoning.
- Development of models for explanation of clustering ML models.

1.4.3 StreamStory XAI Dashboard

Since the publication of Deliverable 5.1 (D5.1) [1], the following new capabilities have been added to StreamStory under Task 5.2.

- An LLM-powered analytical pipeline which helps to transform complex industrial sensor data into clear, actionable insights for predictive maintenance and process optimization.
- A redesigned user interface for the StreamStory XAI Dashboard, enabling interactive visualization of extracted patterns and their interpretations.

[1] https://www.kaggle.com/code/enigmak/tabnet-deep-neural-network-for-tabular-data
[2] https://huggingface.co/docs/transformers/en/model_doc/flan-t5
[3] https://www.cloudflare.com/learning/ai/what-is-lora/

- Extension of the StreamStoryPyClient with new functionalities, including practical usage examples
- Designed and implemented a StreamStory pipeline for generating transactions for causal discovery, which converts multivariate sensor data into Markov Chains and then into transaction formats compatible with causal discovery methods.

1.4.4 Explainable Artificial Intelligence (XAI) Scoring Framework

Since the initial version described in D5.1 [1], the XAI Scoring Framework has undergone enhancements in both methodology and implementation.

- Firstly, the integration of qualitative assessment through virtual personas generated by Large Language Models (LLMs). This approach addresses key challenges related to user study scalability and bias, enabling consistent, diverse, and reproducible user-centric evaluation of XAI methods. The quantitative benchmarking suite has expanded to incorporate additional metrics—such as performance (accuracy, recall) — alongside fidelity, stability, and simplicity.
- In addition, the framework now includes an ML estimator that leverages dataset features and historical benchmarks to instantly estimate XAI scores without requiring repeated full benchmarking.
- The system architecture is now a modular, containerized microservices for seamless deployment, maintenance, and integration. Furthermore, Docker-based deployment and a separation between web dashboard and Representational State Transfer (REST) Application Programming Interface (API) afford users flexibility in both interactive and programmatic use. These advancements collectively mark a transition from a fixed, manual assessment tool to a scalable, automated, and user-aware solution for XAI benchmarking.

2 FAME Platform

2.1 Indexed Assets

Table 1 presents the assets indexed in the FAME platform in the scope of T5.1 and T5.2, categorized by asset name and description, relation to pilot, type, and license. Assets 1-4 were indexed under SAX techniques, assets 5-16 under ML/AI analytics, assets 17-19 under the StreamStory XAI dashboard, and asset 20 under the XAI Scoring Framework.

The detailed descriptions of the assets are provided in Sections 3.1 to 3.4.

Table 1: Indexed assets to the FAME platform.

#	Data asset	Pilot	Type	Description	License
1	SAX open-source library	N/A	Library	A set of services for Situation-Aware eXplainability, that is, process aware explanations https://marketplace.fame-horizon.eu/asset/6qpJLyuBYKCHFXMX6	Open source
2	Survey for explanation quality	N/A	Dataset	A dedicated scale created to assess the quality of explanations as perceived by users https://marketplace.fame-horizon.eu/asset/sa2jiSm5Y5M9qKBKQ	Open source
3	BP^C: A benchmark dataset for causal business process reasoning	N/A	Dataset	A benchmark consisting of question-answering data https://marketplace.fame-horizon.eu/asset/odcw3B2MWgWinpnPL	Open source
4	Explanation of clustering	1+2	Code	Code that explains the decision trees generated by a clustering algorithm https://marketplace.fame-horizon.eu/asset/9xYwmMjy9sujrsMT9	Open source
5	K-Means	1	Model	Segmentation of customers based on a profiler application. https://marketplace.fame-horizon.eu/asset/WB7k776Thk7nQjDpA	Proprietary
6	Tabnet Classifier	1	Model	Risk assessment model https://marketplace.fame-horizon.eu/asset/M3XEnWZSKiAgfkt5X	Proprietary
7	K-Means	2	Model	Segmentation of customers based on parking ticket purchases in Athens https://marketplace.fame-horizon.eu/asset/G8xFMwzFihmkw8WS5	Proprietary

8	LLM	4	Model	Structured information extraction from CDTI documents implemented training 5 LLMs. https://marketplace.fame-horizon.eu/asset/2rcRkWFZRJ76Y5MGT	Proprietary
9	Autoencoder	7	Model	Anomaly detection for industrial machinery implemented training 5 autoencoders. https://marketplace.fame-horizon.eu/asset/wNKri5uhnre5ofJ86	Proprietary
10	LSTM	7	Model	Anomalies forecasting for industrial machinery implemented training 4 LSTM. https://marketplace.fame-horizon.eu/asset/WBnFkxEvbxRQjyiR6	Proprietary
11	RFM Processing	1 & 2	Pipeline	Data preprocessing pipeline implementing the RFM (Recency, Frequency and Monetary) technique based on recent customer activity. https://marketplace.fame-horizon.eu/asset/pKJPQ4SPWHaguEYbp	Proprietary
12	RFM Dataset	1	Dataset	Processed dataset from Pilot 1 using the RFM technique. https://marketplace.fame-horizon.eu/asset/RebhxwpKnePs2q5FQ	Proprietary
13	RFM Dataset	2	Dataset	Processed dataset from Pilot 2 using the RFM technique. https://marketplace.fame-horizon.eu/asset/E74qzX23sp3Wq8x5M	Proprietary
14	Frequent parking zones per cluster	2	Dataset	Dataset of the most frequented parking zones, identified via K-Means-based customer segmentation. https://marketplace.fame-horizon.eu/asset/LpRcQbZ85WDeCGhDj	Proprietary
15	PDF Data Extraction	4	Pipeline	Content extraction of specific sections from a PDF. https://marketplace.fame-horizon.eu/asset/wJK2b7k7RoeyqR9nr	Proprietary

16	Synthetic Dataset	4	Dataset	Synthetic dataset reflecting the structure and content of CDTI documents, designed for training and evaluating document extraction models. https://marketplace.fame-horizon.eu/asset/eY9eCfgjg3CYttTBn	Proprietary
17	StreamStory PyClient	7	Library	Python library which simplifies creation of StreamStory models and enables fetching and deleting those models https://marketplace.fame-horizon.eu/asset/LghoFyPEwp524MxPi	Open source
18	StreamStory XAI Dashboard	7	Application	It is an advanced system that uses StreamStory, clustering, pattern mining, and explainable AI to transform industrial data into complex findings, which LLMs then translate into actionable insights delivered through an interactive UI. https://marketplace.fame-horizon.eu/asset/sKq4z2NbXKjHKYiqx	Open source
19	StreamStory Causal	7	Library	The pipeline generates a StreamStory model and then prepares the data for causal analysis by converting it into a transaction dataset https://marketplace.fame-horizon.eu/asset/MEAzFQsHXtC3vh8af	Open source
20	XAI Scoring Framework	N/A	Service	A microservice that quantifies explainability in a way that is both rigorous and user centred and produces a multidimensional XAI score for a new dataset. https://github.com/GeorgeMakridis/xai-scoring-framework	Open source

2.2 Key Performance Indicators

Table 2 shows the measured Key Performance Indicators (KPIs) relative to the expected values set for the end of the project for *Objective 5: ML/AI/XAI Models*. As can be seen, both KPIs exceed expectations.

Table 2: Key Performance Indicators values in T5.1 and T5.2.

Objective category	Objective	Task	Measured KPI	Expected (DoA)	Achieved
ML/AI/XAI Models	O5	T5.1	Number of AI/ML models and algorithms to use incremental analytics	≥ 10	21
		T5.2	Number of XAI techniques/models to be supported	≥ 8	12

We list below these KPIs. The models are detailed in Section 3.1 (for T5.1) and Sections 3.23.33.4 (for T5.2).

Number of AI/ML models (T5.1) - Table 3 in Section 3.1 indicates the number of models developed for each specified functional group by pilot:

- *ML Profiler*: Models for customer segmentation and profiling, 2 models.
- *ML Risk Assessment*: Models assessing customer or transaction risk, 1 model.
- *ML Forecasting*: Predictive models for estimating future outcomes, 5 models.
- *ML Sentiment Analysis*: Models analysing sentiment in textual data, 3 models.
- *ML Anomaly Detector*: Models for detecting abnormal patterns in temporal data, 5 models.
- *Information Extraction*: Models for extracting information from unstructured documents, 5 models.

Number of XAI techniques/models (T5.2) – We distinguish between SAX, XAI Scoring Framework, and StreamStory XAI Dashboard models.

SAX models (8 models):

- *Causal discovery model*: The causal discovery algorithm based on LiNGAM.
- *Explanation synthesis*: The input of knowledge ingredients to an LLM along with a query of interest. The LLM synthesizes these knowledge ingredients to produce an explanation for the query of interest.
- *Process and causal discovery from series data*: The application of process and causal discovery algorithms to series data.
- *Causal discovery model unification*: The extended causal discovery algorithm that enables the unification of multiple process variants.
- *Scale development for the assessment of the quality of the explanations*: The dimensions and respective questions to assess the quality of explanations generated by an LLM as perceived by users.
- *Benchmarking for LLM ability to reason*: Creation of a causal process benchmark.
- *K sufficient terms to explain sentiments model*: The k terms sufficient to give an explanation in a sentiment model.

- *Explanation of clustering*: The decision tree generated for the K-means algorithm.

XAI Scoring Framework models (1 model):

The updated XAI Scoring Framework directly contributes to the KPI by expanding the platform's catalogue of explainability services with the Partial Dependence Plot (PDP) model.

- In D5.1 only two techniques were utilized; the new release supports four XAI methods—SHAP, LIME, Permutation Feature Importance and Partial Dependence Plots.
- The architecture makes it trivial to add further methods and data types (e.g. Grad-CAM), ensuring the KPI can grow steadily as new techniques are onboarded.

StreamStory XAI Dashboard (3 models):

- *StreamStory XAI Dashboard*: An advanced system that transforms complex industrial sensor data into actionable, LLM-generated insights for predictive maintenance.
- *StreamStoryPyClient:* A Python library that simplifies the creation and management of StreamStory models
- *StreamStory Causal*: A pipeline that prepares time-series data for causal discovery by converting it into a transaction dataset.

3 Components Specification

3.1 ML/AI Analytics

This section presents a summary of the ML models developed across the FAME pilots, organized by use case category and pilot. Table 3 provides an aggregated view of the ML models implemented in Task 5.1, categorized into six main functional groups:

- **ML Profiler**: Models for customer segmentation and profiling.
- **ML Risk Assessment**: Models assessing customer or transaction risk.
- **ML Forecasting**: Predictive models for estimating future outcomes.
- **ML Sentiment Analysis**: Models analysing sentiment in textual data.
- **ML Anomaly Detector**: Models for detecting abnormal patterns in temporal data.
- **Information Extraction**: Models for extracting information from unstructured documents.

Each column represents a specific FAME pilot, while the rows indicate the number of models developed for each functional group within the corresponding pilot. This table highlights the distribution and coverage of ML development across the project. Each model will be explained in detail in Section 3.1.1. As can been seen a total of 21 models have been developed.

Table 3: Summary of types of implemented models across the seven FAME pilots.

	Pilot 1	Pilot 2	Pilot 3	Pilot 4	Pilot 5	Pilot 6	Pilot 7
ML Profiler	**1**	**1**	-	-	-	-	-
ML Risk Assessment	**1**	-	-	-	-	-	-
ML Forecasting	-	-	-	-	**1**	-	**4**
ML Sentiment Analysis	-	-	-	-	**3**	-	-
ML Anomaly Detector	-	-	-	-	-	-	**5**
Information Extraction	-	-	-	**5**	-	-	-
Total per pilot	**2**	**1**	**0**	**5**	**4**	**0**	**9**

In Deliverable D5.1, the Sentiment Analysis and Forecasting models developed for Pilot 5 were already described in detail. As their implementation was fully completed at the time of writing that deliverable, and no further development has been carried out since then, these models are not covered again in this document.

No ML/AI analytics were implemented for Pilot 3, as no data was available to support the development process. For Pilot 6, no implementation was undertaken within Task 5.1 because the pilot team already possessed the necessary technical expertise and infrastructure to carry out the required analytics independently.

3.1.1 Description

This section provides a detailed description of the Machine Learning models implemented. For each model, the relevance to the specific use case will be explained, along with the technical rationale

behind the selected approach, the model specifications, and the results obtained. Since the datasets collected from each pilot required preprocessing before training and evaluation, the applied data cleaning and transformation pipelines are also described.

3.1.1.1 ML Profiler

Pilots 1 and 2 addressed distinct use cases but shared a common objective: gaining a better understanding of customer behaviour to enable personalized recommendations.

Pilot 1, under Use Case 1, focused on developing a financial recommendation engine targeted at families. The objective was to analyse how customers interacted with the financial platform, with the aim of identifying behavioural patterns and offering suitable financial products to specific user segments more likely to need them.

Pilot 2, based in Athens, aimed to identify the most frequently used parking areas among customers. By analysing customer activity, the goal was to develop individual customer profiles to enable tailored offers, such as discounts on parking ticket purchases.

Although the pilots focused on different application areas, both relied on historical, time-ordered transaction datasets in which each record was associated with a specific customer.

3.1.1.1.1 Available datasets

The dataset from Pilot 1 spanned approximately two years. However, due to data privacy constraints, all numerical variables were transformed into categorical values by grouping them into 10 intervals (from 0 to 9), based on the minimum and maximum range of each variable. To reduce dataset size, transactions were aggregated into two windows: the last 12 months and the last 3 months. As a result, the dataset lacked continuity both in terms of the original numerical values and in the temporal sequence of individual transactions. Table 4 summarises the dataset attributes, grouped into thematic categories.

Table 4: Summary of variable groups in the dataset provided by Pilot 1. All numerical variables are grouped into bins and computed over different time windows (the last 12 or 3 months).

Group	Included Variables (Summary)	Description
Identification	CustomerID, DATE	Unique customer ID and timestamp of the record.
Demographics	client_age, client_number_dependents, profession_group, marital_status, nationality, generation, residence_status, urbanization	Personal and socio-demographic characteristics of the client.
Income & Credit Profile	buying_power, monthly_net_salary, contract_age, credit_limit, credit_limit_amount, min_credit_limit_amount, revolving_percentage, credit_limit_amount_pct_increase	Financial attributes including income and credit-related indicators.
App Usage	money_transfers, service_payments, app_logins, app_pages_valuable_actions, app_distinct_events	Frequency of interaction with specific app features over recent time windows.
Digital Profile	digital_product, digital_user, client_email_statement, light_plan	Indicators of digital engagement and preferences.
Transaction Timing	months_since_last_trx, months_since_first_trx, months_between_first_last_trx	Temporal metrics summarizing customer transaction history.

Transaction Volume	trxs_count, avg_trx_value, total_trx_value	Summary statistics on the number and value of transactions.
Transaction Breakdown	withdrawals_value, transfers_value, online_value, max_purchase_amount, instalments_value, instalments_count, appliances_value, household_value, travel_value	Value and frequency of transactions by category.
Credit Statement	revolving_due_capital_amount_last_statement, instalments_due_capital_amount_last_statement	Outstanding capital amounts from the latest credit statement.
Communication Activity	communications_sent, communications_received, communications_open, communications_click	Customer engagement with communications across multiple time windows.
Contract & Onboarding	employment_contract_date, card_life_cycle, address_district, onboarding_store, cardholders	Contract start date, account lifecycle indicators, and onboarding metadata.
Institutional Info	Institution_Type, bank	Institution type and affiliated bank for the client.

In contrast, Pilot 2 provided 6.5 months of data, with each record corresponding to a customer transaction for booking or purchasing parking services within a defined area. Each transaction included a timestamp and a customer identifier, allowing for the reconstruction of individual user activity over time. Table 5 presents the variables included in this transaction-level dataset.

Table 5: Description of the dataset used in Pilot 2, including variable names, definitions, and example values from a typical parking transaction record.

Name	Description	Example
Creation Date	Timestamp when the transaction was created in the system.	2021-04-07 10:26:23.000
Type	Type of transaction performed.	“purchase” or “reversal”
Service Type	Type of service involved in the transaction.	parking
Amount	Amount paid for the parking service (in euros).	20
Success Status	Indicates whether the transaction was successfully completed.	Yes
Start Parking At	Start time of the parking session.	07/04/2021 10:26
End Parking At	End time of the parking session.	07/04/2021 12:26
Parking Duration	Total duration of the parking session in minutes.	120
Cancel Parking At	Timestamp when the parking was cancelled, if applicable.	07/04/2021 22:14:00
Parking Location Address	Full address of the parking location.	Ermou 45, Athina 105 63, Greece
Parking Location Longitude	Longitude coordinate of the parking location.	23.729
Parking Location Latitude	Latitude coordinate of the parking location.	3.797
Parking Zone	Zone identifier assigned to the parking location.	zone_7

Extension Status	Indicates whether the session was extended after initial booking.	"No" or "Yes"

3.1.1.1.2 Data Processing and clustering approach

Due to limitations in both datasets, the most suitable strategy for customer segmentation was to apply a clustering model using interpretable behavioural indicators.

In Pilot 1, the main limitation was the loss of continuity and granularity. All numerical variables were pre-binned, and behavioural features were aggregated into fixed time windows (12 and 3 months). This format removed the possibility of using time-aware modelling approaches.

In Pilot 2, the dataset preserved individual transaction records. However, the available variables were transactional and offered little diversity in behavioural information. This limited the capacity of direct clustering on raw data to produce meaningful results.

Given these constraints, the best approach was to obtain the Recency-Frequency-Monetary (RFM) metrics per customer and using the KMeans clustering model to define the customers segmentation.

Clustering methodology

RFM processing is a behavioural profiling method that assigns three metrics to each customer:

- Recency: How recently the customer interacted with the service.
- Frequency: How often the customer interacted.
- Monetary: How much the customer spent.

These metrics help describe the interaction pattern of each customer with the service. Using the datasets available from both pilots, the RFM metrics were calculated per user, resulting in a new dataset per pilot where each customer was assigned a Recency, Frequency, and Monetary value. This dataset was used as input for the clustering model.

KMeans was selected as the clustering algorithm. It is an unsupervised machine learning algorithm that divides a dataset into a fixed number of clusters. Each data point is assigned to the cluster with the closest centroid. The algorithm iteratively refines these centroids to minimize the variance within clusters and maximize the separation between them. It is well suited for segmentation tasks involving structured numeric input, which is the case for the RFM metrics 11].

Pilot 1: RFM dataset construction

In this pilot, customers were identified using *Customer ID*, and samples were sorted by the *DATE* column. RFM features were assigned as follows:

- Recency was computed using a weighted combination of:
  - The number of months since the last transaction (for the past 12 month).
  - How tight and concentrated the user's activity was during that period.

  Both variables were derived from the 12-month aggregation window. The formula used was:

  $$recency = 0.85 \cdot month\_since\_last\_trx + 0.15 \cdot (12 - \text{month\_between\_first\_last\_trx})$$

  Figure 1 shows how the recency score changes as a function of the number of months since the last transaction and the span between the first and last transaction in the 12-month window. As intended by the formula, customers with more recent and concentrated activity have higher

recency scores, while those whose activity is more spread out over the year tend to have lower values.

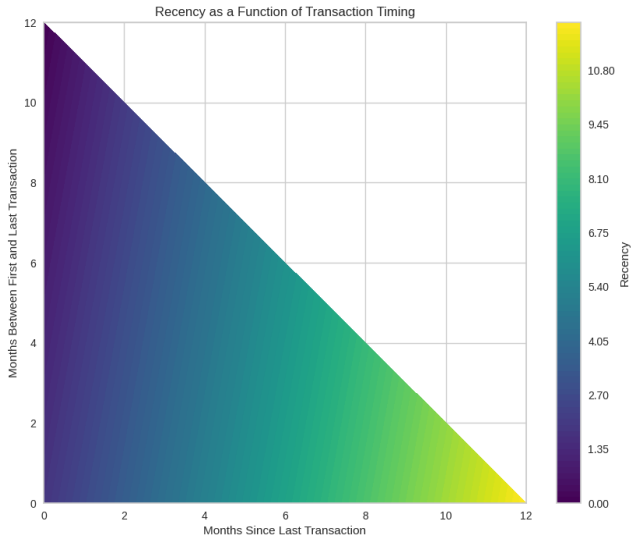


Figure 1: Heatmap showing how the Recency metric varies based on months since the last transaction and the span between the first and last transaction in the 12-month window.

- Monetary was taken from the total amount spent in all transactions from the past 12 months (*total_trx_value_l12m*).
- Frequency was based on the number of transactions made in the last 12 months (*trxs_count_l12m*).

All RFM values were normalized using StandardScaler to ensure equal contribution to the clustering algorithm.

Pilot 2: RFM dataset construction

In this case, users were identified using *User ID*, and transactions were sorted by *Creation Date*. Prior to computing RFM features, the dataset was filtered to retain only valid and relevant records:

- Only transactions implying a purchase were included.
- Transactions marked as unsuccessful were excluded.
- Transactions that were extensions of previous parking sessions were not counted in the frequency metric, although their monetary value was included.

The Pilot 2 dataset did not include any variables that directly represented the RFM metrics. Therefore, these features had to be derived from existing variables by analysing transaction history and computing metrics per customer. The RFM metrics were computed as follows:

- Recency: Number of days since the last transaction per user, calculated as the difference between their most recent *Creation Date* and the latest date in the dataset.
- Monetary: Total spending per user, adjusted by a decay function to reduce the impact of older transactions. The decay weight was computed as:

$$weight = e^{-0.005 \cdot recency\_days}$$

Each transaction's amount was multiplied by its weight and then aggregated per user.

- Frequency: Number of valid parking sessions per user, excluding extensions. One was subtracted from the total count to ensure a minimum of zero.

In this pilot, the RFM metrics were not normalized, as using the original value scales resulted in better clustering performance than after applying different scaling techniques.

3.1.1.1.3 Training process

After computing the RFM metrics for each customer for both pilots, 70% of the dataset was used for training and the remaining 30% for testing and validation. Hyperparameter tuning was performed to optimise the KMeans configuration, as described in Section 3.1.2.1.

The optimal number of clusters was determined using the silhouette score, which measures how well each data point fits within its assigned cluster compared to others. In the silhouette score plot, higher values indicate more compact and better-separated clusters. The most suitable configuration is therefore the one that achieves a high score while also maintaining an appropriate number of clusters, ensuring a balance between segmentation granularity, interpretability, and analytical value.

Results for Pilot 1

The first step in analysing the results was to determine the optimal number of clusters. The silhouette score analysis (Figure 2) indicated that the most suitable configuration was achieved with K = 4 clusters, with a score of approximately 0.43. Although the highest silhouette score was observed at K = 2, this configuration was not selected, as it would have produced only two customer groups. Such a coarse segmentation would not have offered sufficient granularity to capture the diversity of customer behaviours and needs. The elbow method also pointed to K = 4 as the most appropriate number of clusters and is included in Annex A, for reference.

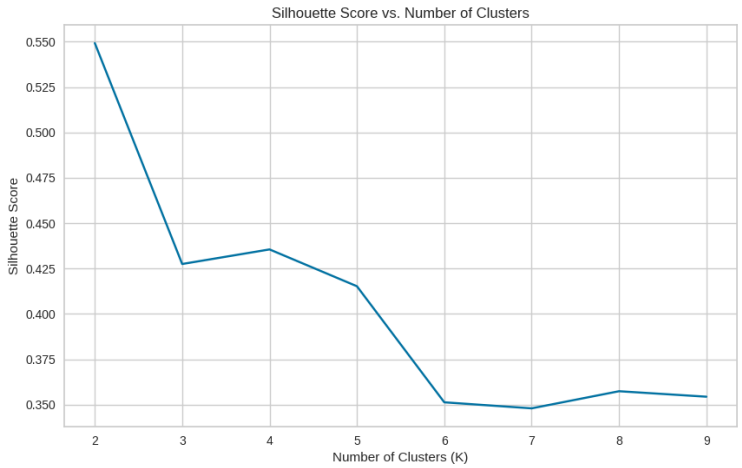


Figure 2: The analysis shows the highest score at K = 2, but a good balance between cohesion and separation is achieved at K = 4, selected as the optimal number of clusters.

Results for Pilot 2

The clustering process for Pilot 2 began by determining the most suitable number of segments. The silhouette analysis (Figure 3) also indicated that a configuration of K = 4 clusters provided a balanced trade-off between similarity within clusters and separation from other clusters, with a silhouette score of approximately 0.54. While K = 2 achieved the highest score, it would have reduced the segmentation to only two categories, offering little practical value for differentiated customer profiling. For completeness, the elbow method analysis is included in Annex B and it points to the same optimal K.

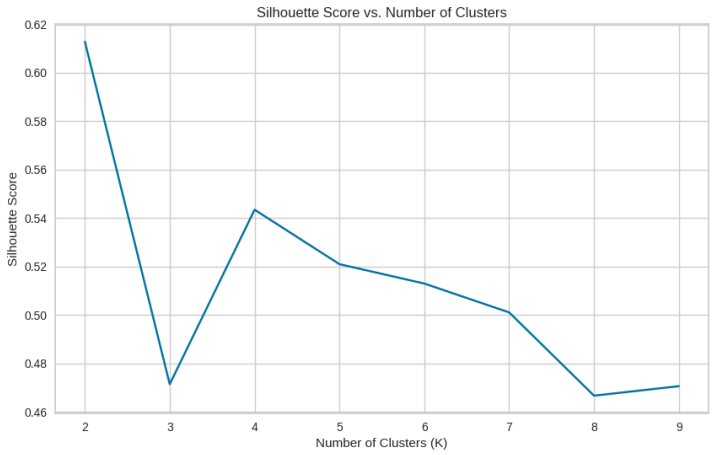


Figure 3: Silhouette score for different values of K, showing that K = 4 achieves a balanced separation of clusters.

In summary, the clustering models implemented for Pilots 1 and 2 successfully generated meaningful customer segments using the RFM methodology and the KMeans algorithm, despite the initial constraints of each dataset. These segments provided a solid foundation for targeted behavioural analysis and the development of pilot-specific applications. The results highlight that the effectiveness of a machine learning model depends more on the quality and representativeness of the data than on the quantity available. Although Pilot 2 relied on only six months of historical data, the dataset was sufficient to train a high-performance model, in comparison to Pilot 1, whose model also achieved strong performance but was trained on a full year of data.

The subsequent sections, 4.1.1.1(for Pilot 1 UC1) and 4.1.2 (for Pilot 2), present the detailed results of these models and illustrate how the identified clusters were applied to address each pilot's objectives, from designing a financial recommendation engine to identifying high-value parking zones. The technical specifications of the implemented models are described in Section 3.1.2.1.

3.1.1.2 ML Risk assessment

This model was developed for Pilot 1, Use Case 2 (UC2), which aimed to implement a predictive scoring system for the *Universo StorePay* product. The main objective was to identify customers at risk of default based on onboarding and transaction-related attributes. The use case targeted three levels of delinquency:

- General Default: Customers who defaulted at any point.
- Default +30 Days: Customers with payment delays exceeding 30 days.
- Early Fraud: Customers who defaulted within the initial months of onboarding.

Due to data sparsity in the last two categories, the model focused exclusively on identifying the general default, defined by the binary variable *flag_atraso*.

The dataset consisted of 10,075 customer records and 23 variables, including demographic, financial, and behavioural information. Table 6 presents the variables available in the dataset, each variable is described with its name, type, and whether it was used during model development. The meaning of each status label is further explained in the feature engineering section.

Table 6: Variables in the Pilot 1 UC2 dataset used for the risk assessment model.

Name	Description	Type	State
Number of dependent	The number of dependents the customer has	Numerical	Used
age	Customer's age	Numerical	Used
Gender	Customer's gender	Categorical	Not used (Low importance)
Marital Status	Customer's marital status	Categorical	Not used (Low importance)
Nationality	Customer's nationality	Categorical	Used
ProfessionMonthly Wage	Average wage for the customer's profession	Numerical	Used
RequestedAmount	Amount of credit requested	Numerical	Used
Profession	Profession category	Categorical	Used
work_experience	Customer's work experience in years		Not used (Low importance)
Housing Type	Customer's housing status (e.g., owner, renter)	Categorical	Used
Net_Monthly_ Income	Declared net monthly income	Numerical	Used
Fixed_Monthly_ Charges	Fixed monthly charges reported by the customer	Numerical	Used
Postal_Code	Postal code of residence	Categorical	Used
DSTI	Debt-to-Service-Income ratio	Numerical	Not used (Low importance)
Missed_Payments_ CRC	Number of missed payments reported to the CRC	Numerical	Not used (Low importance)
Number_of_Financial _Commitments	Total number of financial commitments	Numerical	Used
Universo_Monthly_ Payment	Expected monthly payment in the Universo StorePay product	Numerical	Used
UpdatedOn	Date of the latest data update	Date	Used
flag_atraso	Target variable indicating customer default	Categorical	Used
flag_30_day	Flag for delayed payments over 30 days	Categorical	Not used (Irrelevant)
flag_fraude	Flag for early fraud detection	Categorical	Traget

flag_cliente_existente	Indicates if the customer was previously registered	Categorical	Not used (Irrelevant)
NIF_anonymized	Anonymized national ID	Identifier	Not used (Irrelevant)

The target variable *flag_atraso* was highly imbalanced, with only 2.63% of customers labelled as positive (defaulting at any point). The remaining default flags (*flag_30_day* and *flag_fraude*) had even fewer positive cases and were excluded from modelling. Table 7 presents the distribution of target values in the dataset.

Table 7: Distribution of flag variables in the Pilot 1 UC2 dataset.

flag_atraso	flag_30_day	flag_fraude	Sample count
0	0	0	9810
1	0	0	150
1	1	0	70
1	1	1	45

A correlation analysis was conducted to evaluate the predictive power of the available features. As shown in Figure 4 the overall correlation between the numerical input variables and the target was minimal, with the highest value reaching only 0.07. This represented a limitation on the predictive capacity of standard models.

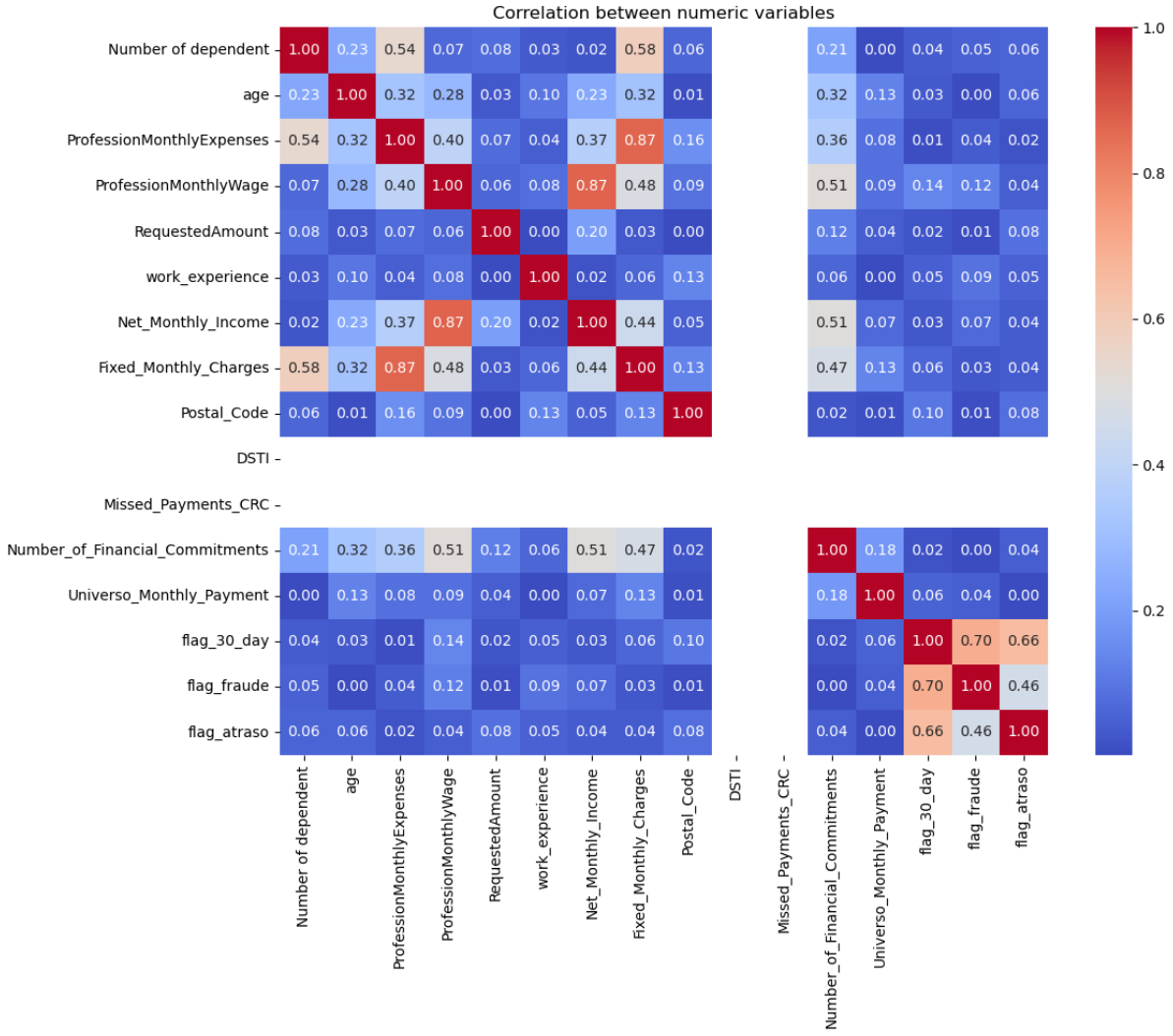


Figure 4: Correlation heatmap between numeric variables and target (flag_atraso).

Two variables, DSTI and Missed_Payments_CRC, appear in the axis of the correlation matrix but display empty cells in the heatmap. This occurs because both variables exhibit no variance in the dataset, each having only a single unique value across all non-missing records. Given that these two variables do not apport any information, they were removed from the dataset to not include noise and unnecessary data into the training process.

Data preparation and feature engineering

Prior to model training, several data preparation and feature engineering processes were carried out to optimize model results. Initially, variables considered redundant or irrelevant for the prediction task, all marked with a “Not used (Irrelevant)” status on Table 6, were removed. Several numerical features were originally stored in the dataset as strings using commas as decimal separators, these were converted to proper float values to enable numerical processing. Missing values were addressed using imputation strategies: median imputation for numerical variables and most frequent value imputation for categorical ones.

Feature engineering was performed to enhance the dataset’s predictive capacity. This included the creation of new variables by combining existing features, such as the product between *RequestedAmount* and the *Number of dependent*, as well as the extraction of temporal components from date fields by splitting the year, month, and day into separate variables. Frequency encoding, a technique that replaces each category with its frequency of occurrence in the dataset, was applied to

selected categorical variables to introduce ordinal information. However, none of these feature engineering techniques significantly increased the correlation between features and the target *flag_fraude* variable.

Given the highly imbalanced nature of the target variable, class distribution was corrected through the application of SMOTE (Synthetic Minority Over-sampling Technique), which creates synthetic examples of the minority class by interpolating between existing samples. To avoid data leakage, SMOTE was applied within each cross-validation fold. Several SMOTE variants were evaluated, including ADASYN (which generates new samples adaptively based on density), SMOTE-Tomek (combining SMOTE with Tomek Links under sampling), Borderline-SMOTE, and SMOTE-ENN (Edited Nearest Neighbours). Ultimately, the traditional SMOTE was selected as it yielded the best overall results. Additionally, custom class weighting was used to further reinforce the contribution of the minority class during model training.

Modelling approach

An experimentation phase was undertaken to identify the most suitable model for predicting the risk of default (*flag_atraso*). Given the high-class imbalance, only 2.63% of customers were flagged as defaulters, and the low correlation between features and the target variable, several modelling strategies were explored.

Initial attempts focused on classical machine learning models, including Random Forest [12], Gradient Boosting [13], and ensemble methods such as Voting Classifiers [14]. These models, however, consistently failed to capture the minority class, yielding zero recall for positive cases. Although the Balanced Random Forest offered some improvement in recall, it introduced a considerable number of false positives, reducing overall model reliability.

To address these challenges, a deep learning approach using the TabNet architecture was adopted. TabNet is particularly well-suited to structured tabular data, as it integrates attention-based feature selection mechanisms and native interpretability [15]. This model demonstrated the highest potential, especially when coupled with specific strategies to address class imbalance, including SMOTE-based oversampling and custom class weighting.

Hyperparameter optimization was conducted using Optuna, targeting architectural parameters such as the number of decision steps, learning rate, and sparsity regularization. Additionally, precision-recall curves were used to identify optimal classification thresholds, prioritizing recall to align with business objectives.

The evolution of the experimentation process is summarized in Table 8, which outlines the key models tested, their rationale, and their impact on performance metrics. Ultimately, the selected TabNet model achieved a recall of 0.62 on the minority class, significantly outperforming classical alternatives and maximizing positive case identification.

Table 8: Summary of models tested for the Pilot 2 UC2 risk assessment task, focusing on handling class imbalance and improving minority class detection.

Model Type	Rationale	Outcome Summary
Random Forest	Baseline model with ensemble capacity	Failed to detect any positive cases (recall = 0)
Gradient Boosting	Tested for incremental improvements over Random Forest	Performance close to random (recall = 0)

Voting Classifier (Hard)	Ensemble of classical models to improve robustness	Slightly improved AUC, recall remained weak
Voting Classifier (Soft)	Averaged probabilities for smoother predictions	Marginal gain in ROC-AUC, poor precision-recall balance
Balanced Random Forest	Introduced sampling to address class imbalance	Improved recall (0.53), but with substantial false positives
TabNet (Initial)	Neural network tailored for tabular data with built-in feature learning	Significant recall improvement. Moderate precision
TabNet + SMOTE Variants	Tested multiple resampling methods	Traditional SMOTE gave best trade-off in recall and FP
Final TabNet (Optimised)	Tuned with Optuna and custom thresholds, recall-focused thresholding	Selected as final model Recall = 0.623

Detailed evaluation metrics for each model, including confusion matrices and derived scores, are provided in Annex C to properly see the results obtained of each experiment.

Feature Importance

A feature importance score was generated based on TabNet's internal attention-based mechanism using the final model, as shown in Figure 5. Variables such as *UpdatedOn*, *Universo_Monthly_Payment*, and *Number of dependent* ranked among the most influential in the prediction, whereas *Nationality*, *RequestedAmount* and *Fixed_Monthly_Charges* were identified as having minimal impact on fraud detection.

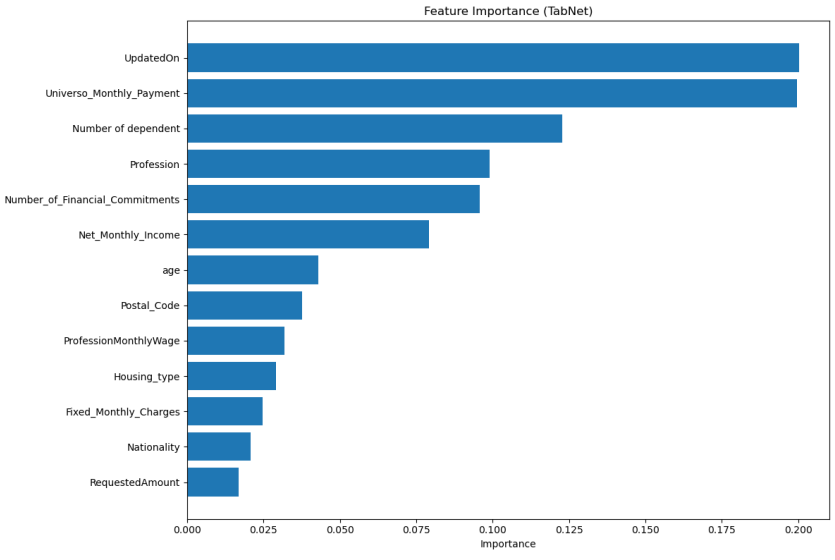


Figure 5: TabNet feature importance bar plot.

All variables marked as “Not used (Low importance)” in Table 6 were excluded from the model’s implementation dataset, as their feature importance scores were among the lowest across all features. These variables were therefore removed to reduce noise and improve model efficiency.

3.1.1.3 Information extraction

Pilot 4 needed to automate the extraction of specific information from the Technological Development and Innovation Centre (CDTI, from Spanish "Centro para el Desarrollo Tecnológico y la Innovación") documents published on the official website. These documents specify the percentage of public funding granted to companies undertaking research, development and innovation projects. The applicable percentage varies according to several eligibility conditions and is disclosed only in the publicly available PDFs on CDTI's site. The three instruments in scope are:

1. ID: Proyectos de I+D (R&D Projects)
2. LIC-A: Línea Directa de Expansión (Direct Line Expansion)
3. LIC: Línea Directa de Innovación (Direct Line of Innovation)

Pilot 4 created an Excel file to record these percentages in a format that allows them to easily identify the funding percentage applicable to each company. The most important columns for this model's development, together with an illustrative example of each type of row, are shown in Table 9.

Table 9: Example rows from CDTI funding tables showing eligibility conditions and funding values to be extracted.

ID. TIPOLOGIA	TIPOLOGIA	TIPO AYUDA	PYME	COMUNIDAD	FONDOS PROPIOS	Tramo no reembolsable_ fondos CDTi (%)	Tramo no reembolsable_ fondos europeos (%)	Duración_ min (meses)	Duración_ max (meses)	Desembolso _min (hito, mes)	Desembolso_ max (hito, mes)	Potencial anticipo max sin aval (%)	Potencial anticipo max con aval (%)	Anticipo max (€)
CID	INVESTIGACIÓN Y DESARROLLO COOPERACION	Ayudas Parcialmente Reembolsables de I+D	S	ANDALUCIA	PRESUPUESTO-APORTACIÓN CDTI	20%	33%	12	36	9	18	50%	75%	300.000,00 €
CID	INVESTIGACIÓN Y DESARROLLO COOPERACION	Ayudas Parcialmente Reembolsables de I+D	N	ANDALUCIA	PRESUPUESTO-APORTACIÓN CDTI	15%	30%	12	36	9	18	50%	75%	300.000,00 €
ID	INVESTIGACIÓN Y DESARROLLO	Ayudas Parcialmente Reembolsables de I+D	S	LA RIOJA	PRESUPUESTO-APORTACIÓN CDTI	17%	20%	12	36	9	18	50%	75%	300.000,00 €
ID	INVESTIGACIÓN Y DESARROLLO	Ayudas Parcialmente Reembolsables de I+D	N	ANDALUCIA	PRESUPUESTO-APORTACIÓN CDTI	10%	10%	12	36	9	18	50%	75%	300.000,00 €
LIC	LÍNEA DIRECTA DE INNOVACIÓN	Ayudas Parcialmente Reembolsables de Innovación	N	ANDALUCIA	PRESUPUESTO-APORTACIÓN CDTI	7%	10%	9	24	9	24	50%	75%	300.000,00 €
LICa	LÍNEA DIRECTA DE EXPANSIÓN	Ayudas Parcialmente Reembolsables de Innovación	N	Comunidad Foral de Navarra	PRESUPUESTO-APORTACIÓN CDTI	10%	20%	9	24	9	24	50%	75%	8.000.000,00 €

Columns in white indicate the condition specifications for each type of company:

- Typology ID (*Id. tipología*): refers to the type of PDF, which may be ID, CID, LIC, or LICa. The ID and CID are both included in the "Proyectos de I+D" PDF.
- Typology (*Tipología*): refers to the complete name of the Typology ID.
- Aid type (*Tipo de ayuda*)
- SME (*PYME*)
- Community (*Comunidad*): refers to the Spanish Autonomous Community where the company is located.
- Own funds (*Fondos propios*)

The columns coloured in yellow in Table 9 represent the percentages needed to be automatically extracted based on these values and the information provided in the CDTI documentation. The English translation of each column is:

- Tramo no reembolsable fondos CDTI: Non-repayable CDTI funds tranche
- Tramo no reembolsable fondos europeos: Non-repayable European funds tranche
- Duración mín. (meses): Minimum duration (months)
- Duración máx. (meses): Maximum duration (months)
- Desembolso mín. (meses): Minimum disbursement (months)

- Desembolso máx. (meses): Maximum disbursement (months)
- Potencial anticipo máx. sin aval: Maximum advance without guarantee
- Potencial anticipo máx. con aval: Maximum advance with guarantee
- Anticipo máximo: Maximum advance

Hence, the purpose of the ML analysis for Pilot 4 is to implement a model capable of extracting the numerical values required to populate the yellow-highlighted columns in Table 9. This table contains 152 rows, each corresponding to a unique combination of condition specifications. The "Typology ID" column indicates the PDF from which the parameters must be extracted, and these parameters appear in fixed sections for all types and versions of the PDFs. All variables are located in three specific sections of the documents.

Figure 6 depicts excerpts from these relevant sections, with selected areas highlighting examples of the variables to be extracted. As shown, some appear in purely numeric form (e.g., *30%*), others in textual form (e.g., *80 millones de euros*, meaning "80 million euros"), and some combine both (e.g., *Hasta 20%*, meaning "Up to 20%").

Proyectos realizados por una única empresa (categorías a, c, d, e, f, g)

CATEGORÍAS DE PROYECTOS DE I+D [18]	TIPO DE EMPRESA[17] PYME	GRAN EMPRESA
1. Proyectos de I+D Cofinanciados con fondos FEDER en Andalucía, Canarias, Castilla La Mancha, Ceuta, Extremadura y Melilla.	hasta 30%	No aplica (*) (**)
2. Proyectos de I+D Cofinanciados con fondos FEDER en el resto de las CCAA.	Hasta 20%	No aplica (*)
3. Proyectos de I+D Capacitación Tecnológica Internacional. 4. Proyectos de I+D de Cooperación Tecnológica Internacional. 5. Proyectos de I+D de Cooperación Tecnológica Europea. 6. Proyectos de I+D de Tecnologías Duales.	Hasta 30%	Hasta 25%
7. Proyectos de I+D Orientados (PO)	Hasta 33%	
8. Proyectos de I+D no incluidos en las categorías anteriores.	hasta 17%	hasta 10%

(*) A las pequeñas empresas de mediana capitalización (499 o menos empleados) se les podrá aplicar el TNR correspondiente a una PYME (hasta el 30% o el 20%, según la región de desarrollo) siempre que cumplan con el resto de las condiciones FEDER.
(**) Las grandes empresas que desarrollen el proyecto en la CCAA de Canarias podrán recibir un TNR de hasta el 30% siempre que cumplan con el resto de las condiciones FEDER, en virtud de su elegibilidad como región ultraperiférica.

8. Duración de los proyectos de I+D.

Los proyectos de I+D tendrán una duración de 12 a 36 meses, a excepción de los Proyectos de I+D de Cooperación Nacional (tipología b), que tendrán una duración de 12 a 48 meses, sin perjuicio de la aprobación de posibles prórrogas de ejecución.

Todos los proyectos podrán tener uno o varios hitos técnicos, cuya duración deberá ser de 9 a 18 meses.

16. Anticipos de la ayuda.

Esta ayuda ofrece las siguientes alternativas de anticipo, el cual será abonado una vez formalizado el contrato de préstamo:

a) La empresa dispondrá de un anticipo de hasta el 50% de la ayuda concedida, sin exigencia de garantías adicionales a las condiciones financieras acordadas por el Consejo de Administración en la aprobación del proyecto.

b) La empresa podrá solicitar aumentar el anticipo hasta el 75% de la ayuda concedida, siempre que garantice dicho aumento, aportando avales de entidades de crédito (Bancos, Cajas de Ahorro o Cooperativas de Crédito), Establecimientos Financieros de Crédito o de Sociedades de Garantía Recíproca (SGR), inscritas todas ellas en el Registro de Entidades del Banco de España. En cualquier caso, han de contar con solvencia suficiente a juicio del CDTI para garantizar el aumento de anticipo y no suponer ayuda pública.

Este primer desembolso no podrá superar, en ningún caso, los 8 millones de euros.

Figure 6: Examples of relevant PDF sections showing numeric, textual, and mixed-format variables to be extracted.

3.1.1.3.1 Pipeline overview

A high-level view of the end-to-end pipeline, PDF parsing, dataset construction, synthetic data generation, and LLM fine-tuning, is shown in Figure 7.

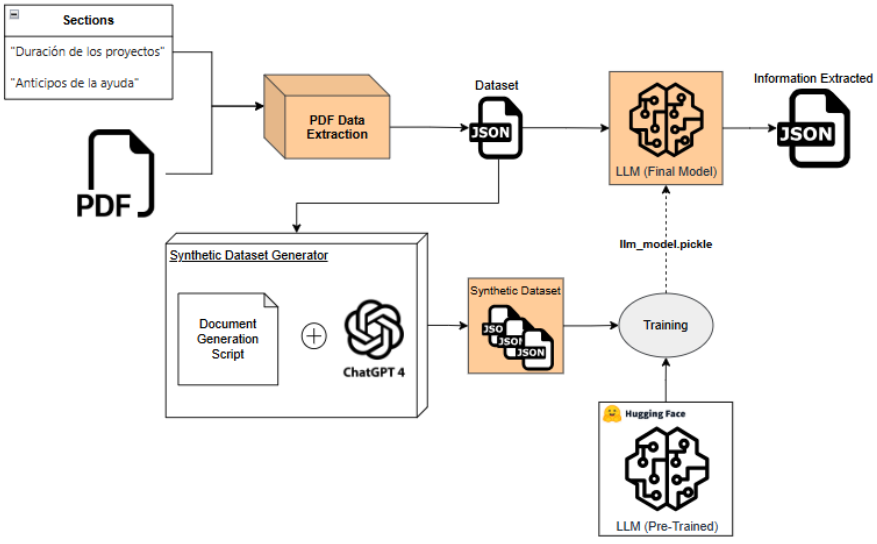


Figure 7: High-level pipeline for PDF sectioning, dataset construction, synthetic data generation, and LLM fine-tuning.

The information extraction process for Pilot 4 was implemented in four main stages, from PDF parsing to the deployment of fine-tuned extractors.

Step 1: PDF data extraction

The process began by locating the relevant sections within each PDF, such as "*Duración de los proyectos"* and "*Anticipos de la ayuda*". Section boundaries were identified from the index page and extracted sequentially on a page-by-page basis. Plain text content was extracted using *pdfplumber*[4], explicitly excluding table regions to avoid structural loss. Tables were extracted separately using *Camelot*[5] in lattice mode, preserving the original row and column structure. The extraction output was compiled in JSON format, containing for each input PDF the cleaned text and any extracted tables for every section. Figure 8 provides an example of the output generated containing the extracted text from each section of the three sections of the three PDFs.

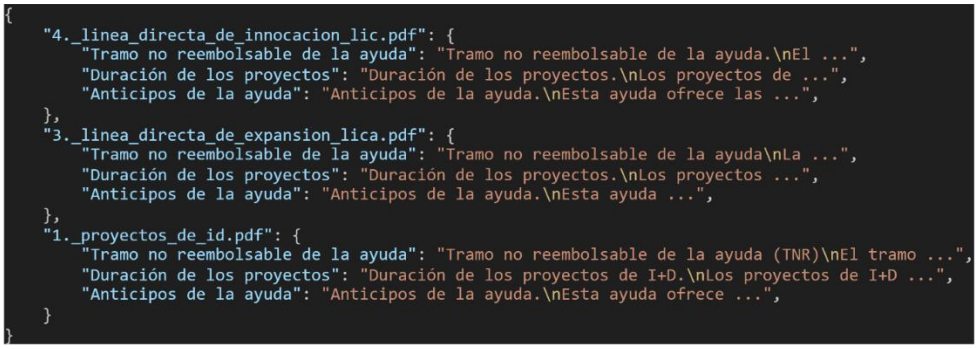


Figure 8: Example of the PDF data extraction output in a JSON format. Each PDF contains three already specified sections from which the text was extracted.

Step 2: Dataset structuring and synthetic dataset generation

For this pilot, the only available data source were the three PDFs published by the CDTI. In order to implement the information extraction model, it was necessary to construct a dedicated dataset. This dataset was created by combining:

1. The extracted text from each relevant section of the PDFs (Figure 8).
2. The complete set of condition specifications and variables to be filled, as defined in the Excel template (Table 9).

A JSON file was generated containing one sample for each row of the Excel file. Each row corresponded to a specific set of conditions (white columns in Table 9) and the variable to be extracted from the text. Two prompt formats were defined, depending on whether the variable to be filled was expressed as a percentage or as a numeric value:

A JSON file was generated with a sample for each row of the Excel file. Each row could contain two types of formats, depending on if the excel variable to fill was supposed to be a percentage or a numeric value:

- Percentage variable prompt format:
  `"Extract decimal percentage of '{variable}' when {condition}: '{text}'"`
- Numeric variable prompt format:
  `"Extract numeric value of '{variable}' when {condition}: '{text}'"`

[4] https://pypi.org/project/pdfplumber/
[5] https://pypi.org/project/camelot-py/

Where:

- {variable} is the name of the Excel column to be completed.
- {condition} is the set of specifications describing the row's context. For example:

```
ID. TIPOLOGIA='ID'; TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO'; TIPO DE AYUDA='Ayudas
Parcialmente Reembolsables de I+D'; PYME='S'; COMUNIDAD='ANDALUCIA'; FONDOS
PROPIOS='PRESUPUESTO-APORTACIÓN CDTI'
```

- {text} is the extracted content of the PDF section where the variable's value is located, as obtained during the text extraction stage.

Figure 9 presents an excerpt of the generated input dataset in its original language (Spanish), showing the structure of the prompts provided to the model for variable extraction.

```
[
    "Extrae porcentaje decimal de 'Tramo no reembolsable_fondos CDTi' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y ...",
    "Extrae porcentaje decimal de 'Tramo no reembolsable_fondos europeos' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN ...",
    "Extrae valor numerico de 'Duración_min (meses)' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO ...",
    "Extrae valor numerico de 'Duración_max (meses)' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO COOPERACION'...",
    "Extrae valor numerico de 'Desembolso_min (hito, mes)' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO ...",
    "Extrae valor numerico de 'Desembolso_max (hito, mes)' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO ...",
    "Extrae porcentaje decimal de 'Potencial anticipo max sin aval' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y ...",
    "Extrae porcentaje decimal de 'Potencial anticipo max con aval' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y ...",
    "Extrae valor numerico de 'Anticipo max (€)' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y DESARROLLO COOPERACION'; ...",
    "Extrae porcentaje decimal de 'Tramo no reembolsable_fondos CDTi' cuando ID. TIPOLOGIA='CID';TIPOLOGIA='INVESTIGACIÓN Y ...",
    "Extrae porcentaje decimal de 'Tramo no reembolsable_fondos europeos' cuando ID. TIPOLOGIA='CID';TIPOLOGIA..."
    ...
]
```

Figure 9: Dataset created, containing a sample for each row of the Excel file with the prompts to input the model in order to obtain the numeric variable from the text.

The corresponding output dataset contained the numeric values for each variable in the same order as the JSON input file. An example output sequence would be: `{0.2, 0.33, 12, 36, 9, …}`.

Given that the original dataset consisted of only 152 samples, additional synthetic data was generated to improve coverage and model robustness. Numeric values were systematically perturbed while preserving internal consistency between inputs and outputs. Perturbations were constrained to maintain contextual plausibility, for example, if the original value was 8 million euros, the generated values were kept within the same order of magnitude.

Linguistic diversity was introduced by paraphrasing each sample's text using ChatGPT, ensuring semantic equivalence while varying sentence structure and wording.

The resulting augmented dataset contained 1,368 samples, representing nine synthetic variations for each of the 152 original condition, variable combinations. The original dataset was reserved for testing to ensure unbiased evaluation of the model's performance.

Step 3: Text processing

Some sections in the source PDFs were substantially long, resulting in large character counts for certain dataset samples. For example, samples related to the "Non-repayable CDTI funds tranche" variable reached up to 4,673 characters. Since a pre-trained Large Language Model (LLM) was to be used for information extraction, it was necessary to account for the input length limitations of the selected models, which typically is 512 or 1,024 tokens.

To ensure compatibility while preserving the relevant information, two preprocessing strategies were applied:

- **Lemmatisation:** Lemmatisation is the process of reducing words to their base or dictionary form (lemma), considering their grammatical role and meaning. In this pilot, lemmatisation

was applied in either full mode (all words) or soft mode (only selected word types), depending on the impact on extraction accuracy.

- **Truncation:** When the section text still exceeded the model's maximum token limit, it was truncated in the middle portion, as the most relevant information was generally located at the beginning or end of the section.

Step 4: Model training

The information extraction solution was based on a question-to-value approach, where the model receives a question about a specific variable and a relevant excerpt from the PDF, and returns the corresponding value. The synthetic dataset was divided into training and validation subsets using a stratified split to ensure that the associated context was consistent across both sets, allowing for a reliable evaluation of the models' performance.

The base model used was google/flan-t5-small[6], a lightweight sequence-to-sequence language model well-suited for text generation and comprehension tasks. Fine-tuning was performed using the Low-Rank Adaptation (LoRA) [16] method, which enables efficient adaptation of large pre-trained models by updating only a small subset of parameters, reducing computational requirements while preserving performance. This configuration was selected for its balance between accuracy, efficiency, and adaptability to the domain-specific nature of the task.

Several dataset processing variations (Step 3) were evaluated during training to determine the optimal settings for accuracy and robustness. To maintain clarity and scalability, independent extraction models were trained, each dedicated to predicting one of the target variables. This modular structure allows the addition of new variables in the future without affecting the performance of existing models.

The testing phase confirmed that the chosen configuration could accurately extract the required values from the PDFs while operating within the input length limitations of the underlying language model. Figure 10 illustrates the loss curve for the model trained to extract the "Maximum duration" variable, showing a consistent decrease in both training and validation loss across epochs.

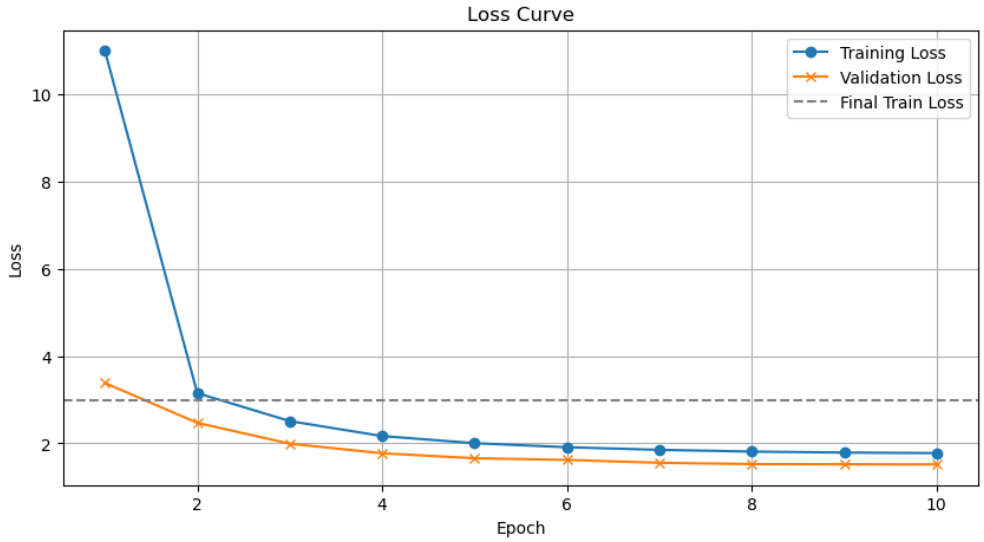


Figure 10: Loss curve of the model trained to extract the maximum duration variable.

[6] https://huggingface.co/google/flan-t5-small

3.1.1.4 ML Anomaly detector

The anomaly detection solution developed in Pilot 7 constitutes the first phase of the equipment refinery's monitoring pipeline, preceding the forecasting models. Its objective is to identify abnormal behaviour in machinery sensors, enabling the triggering of alarms for unexpected operational patterns.

The approach is based on CNN autoencoders trained separately for each sensor type, leveraging horizontal correlations to capture similar behaviour of the same sensor type across different machines. Each model learns the normal patterns for its sensor type and attempts to reproduce incoming signals. When the reconstructed signal deviates significantly from the actual reading, this difference is treated as an indicator of abnormal behaviour. The implemented architecture contains two output branches:

- A regression output for signal reconstruction.
- A classification output to identify anomalies.

Dataset characteristics

The model was trained using the Pilot 7 dataset, which included raw sensor readings collected at regular intervals from multiple machines. The machinery is organised into compressor groups, with each machine equipped with a variable number and type of sensors. This organisation is illustrated in Figure 11. The main sensor types are temperature, pressure, axial displacement, and vibration.

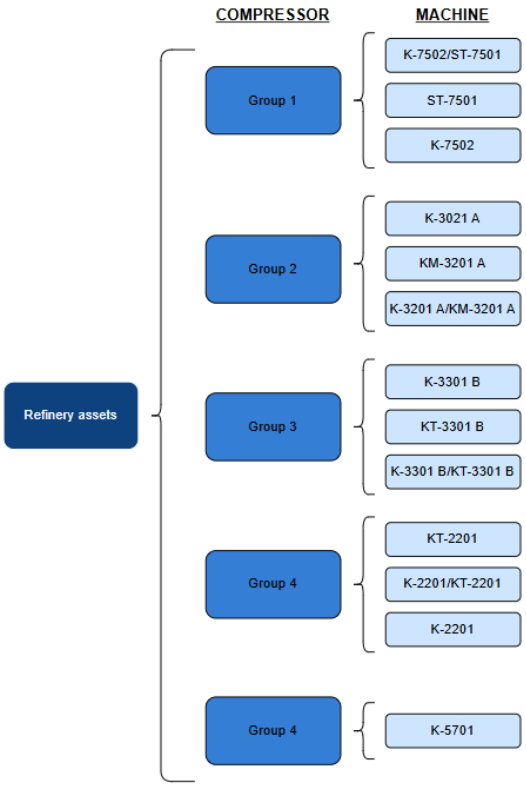


Figure 11: Diagram of the organisation of equipment into compressor groups.

An individual autoencoder was implemented for each sensor type, using data from all machines equipped with that category of sensor. This configuration enabled the exploitation of horizontal correlations, capturing normal operational patterns across machines of the same type.

The dataset originally included labelled anomalies, although their number was very limited, making it difficult to extract relevant patterns for training. To overcome this limitation and ensure the

development of the model, a new definition of anomaly was agreed with the pilot, considering as anomalous those events that deviated from the mean by more than three standard deviations. Using this criterion, the dataset was modified preserving the original sensor records but replacing the anomaly labels. This adaptation enabled the models to be trained and tested in a consistent way and demonstrated that the approach can be scaled to more realistic datasets containing a larger number of validated anomalies.

The implementation focused on temperature sensors as a representative example, selected to demonstrate the approach and assess its feasibility for other sensor types. The anomaly detector was deployed for the temperature sensors of the following machines:

- K-2201
- K-3201
- K-3301
- K-5701
- K-7502

Data processing

The data processing pipeline contained the following steps:

1. **Window segmentation**: time series data were divided into fixed-length windows of 24 hours, sampled at five-minute intervals (12 samples per hour), resulting in 288 timesteps per window (input size of 24x12).
2. **Normalisation**: each sensor's readings were scaled individually to the range [0, 1] using Min-Max scaling. This ensured comparability of magnitudes across machines and promoted stable model convergence.
3. **Dataset splitting**: the dataset was split into training, validation, and test subsets, preserving temporal order to maintain the integrity of time-dependent patterns.
4. **Target preparation**: two distinct targets were generated:
    - Reconstruction target: the original sensor window, used for signal reconstruction.
    - Classification target: A binary anomaly indicator marking anomalous events within the dataset.

Model architecture and training

The anomaly detection [17] model implemented in Pilot 7 was a dual-output convolutional autoencoder, designed to both reconstruct the input sensor signal and classify anomalies. The architecture combines:

- Convolutional layers (encoder) to extract short-range temporal features and progressively reduce sequence length, enabling a compact latent representation of the signal.
- Transposed convolutional layers (regression decoder) to reconstruct the original time series from the latent space, preserving temporal coherence.
- Dense layers (classification decoder) to map the latent features to a binary anomaly score, enabling direct identification of abnormal patterns.

The network was implemented using TensorFlow/Keras[7]. The internal structure of the model is summarised in Table 10. The model receives an input window of 288 timesteps (24 hours sampled at

[7] https://www.tensorflow.org/guide/keras

5-minute intervals). The convolutional encoder uses causal padding to ensure that each timestep is only influenced by current and past values, preserving the temporal order.

Table 10: Internal architecture of the anomaly detection network.

Component	Layer (type)	Output shape
Encoder	Input (window=288 timesteps, 1 feature)	(288, 1)
	Conv1D (32 filters, k=6, stride=2, causal)	(142, 32)
	ReLU	(142, 32)
	Conv1D (64 filters, k=4, stride=2, causal)	(70, 64)
	ReLU	(70, 64)
Regression decoder	Conv1DTranspose (32 filters, k=7, stride=2)	(140, 32)
	ReLU	(140, 32)
	Conv1DTranspose (1 filter, k=7, stride=2)	(288, 1)
	Linear activation (reconstruction)	(288, 1)
Classification decoder	Flatten	(4480)
	Dense	(288)
	Dense	(24)
	Dense (sigmoid)	(1)

Training configuration

The anomaly detection model was trained using the following settings:

- **Loss function**: A combination of *Mean Squared Error* (for the reconstruction branch) and *Binary Focal Crossentropy* (for the classification branch). The first penalises large deviations in the reconstructed signal, encouraging accurate reproduction of normal patterns, while the second addresses class imbalance by focusing learning on harder-to-classify anomaly cases.
- **Optimiser**: *Adam* optimiser with the default learning rate, which adapts the learning rate for each parameter based on historical gradients, improving stability and convergence speed.
- **Batch size**: 500 samples per batch, balancing computational efficiency with gradient stability.
- **Early stopping**: Training was terminated if validation loss did not improve for 30 consecutive epochs, preventing overfitting and unnecessary computation.

3.1.1.5 ML Forecasting

One of the objectives of Pilot 7 was to design, implement, and deploy a forecasting system capable of predicting sensor values for the following week using the most recent two weeks of historical data. Pilot 7 operates with multiple industrial machines, each equipped with a diverse set of sensors. The implemented models focused exclusively on temperature sensors, as these presented anomalies within the available datasets and ensured consistency with the sensor type used in the anomaly detection implementation.

The datasets comprised high-frequency measurements collected from November 2019 to January 2021 at a five-minute sampling rate. Each machine's dataset was stored in an independent file, containing a Timestamp column and one column per sensor.

The first stage consisted of evaluating the suitability of each sensor for forecasting. This was determined through an analysis of both variability and temporal dependency, using two key parameters. The standard deviation quantified the variability of a sensor's data over time, with higher values indicating greater fluctuations. The autocorrelation measured the degree of similarity between most recent data and those taken at earlier time steps, indicating how strongly past values influenced future observations.

Sensors with a one-week autocorrelation below 0.5 were considered to have insufficient temporal dependency for reliable forecasting, while those with a standard deviation below 3.0 were thought too stable or constant to provide meaningful predictive modelling. These thresholds were selected after examining the distribution of autocorrelation and variability values across all available sensors.

In addition, the entire historical record of each sensor was examined for anomalies, defined as events with more than three standard deviations away from the mean. This ensured that the models would be trained on realistic operational data, only sensors with more than one anomaly over the full recording period were included. The final list of machines and sensors selected for modelling was as follows:

- **K-2201**: sensors 22TI118 and 22TI42
- **K-3201**: sensors 32TI444 and 32TI439
- **K-3301**: sensor 33TI610
- **K-5701**: sensors 57TI030 and 57TI003

Data preparation and processing

Data preparation followed a consistent, automated pipeline. The datasets were split into 70% for training and the remaining 30% for testing, maintaining temporal order to preserve the integrity of time-dependent patterns. For each machine, the selected sensors were normalised individually using Min-Max scaling to ensure comparability of magnitudes and facilitate model convergence. Sliding time windows were then constructed to match the input and output requirements of the forecasting model. Specifically, the model used a lookback window of two weeks (4,032 timesteps at five-minute intervals) to predict the next seven days (2,016 timesteps). The input data thus had the shape (lookback, number_of_sensors) and the output had the shape (forecast_horizon, number_of_sensors). A model was implemented for each machine, resulting in five models in total, four of which were trained with two input sensors, and one (K-3301) with a single input sensor.

Model architecture and training

The forecasting model implemented was an LSTM-CNN multivariate time series predictor [18], capable of simultaneously processing multiple sensor inputs to generate parallel forecasts. The architecture combines:

- Convolutional layers (CNN) for short-range temporal feature extraction.

- Bidirectional LSTM layers to capture dependencies in both forward and backward temporal directions, enhancing long-term pattern recognition.
- Dense and reshape layers to produce parallel forecasts for all sensors over the one-week horizon.

The network was implemented as a sequential Keras[8] model with the Nadam optimiser (learning rate 0.0005) and Huber loss to mitigate the effect of outliers. Training used early stopping based on validation loss, with a patience of 20 epochs to prevent overfitting.

The internal structure of the model is summarised in Table 11. The model receives an input window of 4,032 timesteps (two weeks at a five-minute interval). After passing through the first Conv1D layer with a kernel size of 3 and padding set to "valid", the sequence length is reduced by two timesteps, resulting in 4,030 timesteps. This explains the difference between the raw input length described in the data preparation section and the post-convolution sequence length shown in the architecture table.

Table 11: Internal architecture of the forecasting network.

Layer (type)	Output shape (batch, …)	Params
Conv1D (16 filters, k=3, ReLU, valid)	(4030, 16)	112
MaxPooling1D (pool=2)	(2015, 16)	0
Bidirectional LSTM (128, return_sequences=True)	(2015, 256)	148,480
Dropout (rate=0.3)	(2015, 256)	0
LSTM (64)	(64)	82,176
Dense (2016×2 = 4,032 units)	(4,032)	262,080
Reshape	(2016, 2)	0

Figure 12 contains the training and validation loss curve of the K-5701 model, the convergence behaviour indicates a rapid reduction in loss during the initial epochs, followed by stable generalisation.

[8] https://keras.io/

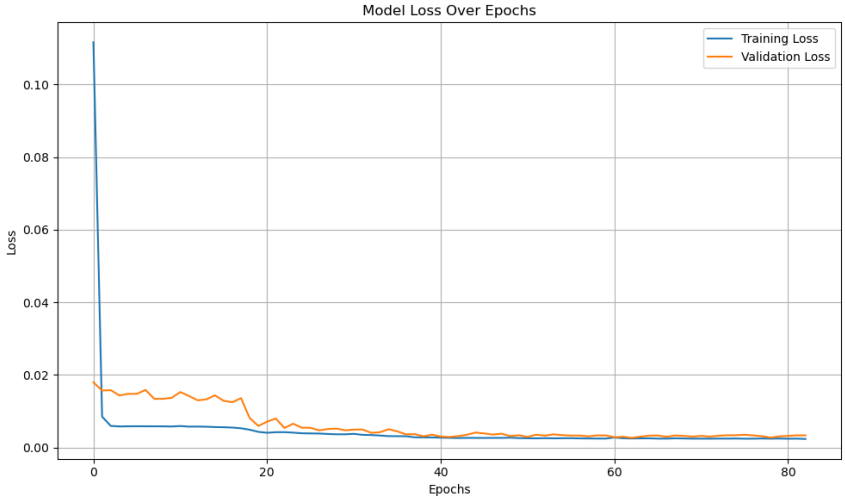


Figure 12: Training and validation loss curves for the LSTM model predicting K-5701 sensors.

Evaluation

Performance was assessed quantitatively using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) for each sensor. Additionally, the error distribution was visualised using histograms to inspect bias and dispersion, and time series overlays were generated to compare predicted values against actual measurements across the entire test period.

Figure 13 shows the performance of the K-5701 model for its two temperature sensors, 57TI003 (left) and 57TI030 (right). The upper panels present the comparison between the predicted and actual values, while the lower panels display the corresponding prediction error distributions.

For sensor 57TI003, the model's predictions closely follow the observed values, particularly during stable operating conditions, with the majority of errors concentrated near zero. This indicates that the model effectively captured the sensor's temporal dynamics. Sensor 57TI030 also demonstrates good alignment between predicted and actual values for much of the period. However, during certain intervals characterised by abrupt operational changes, the predictions diverged more noticeably from the ground truth. This is reflected in the wider spread of its error distribution compared to that of 57TI003.

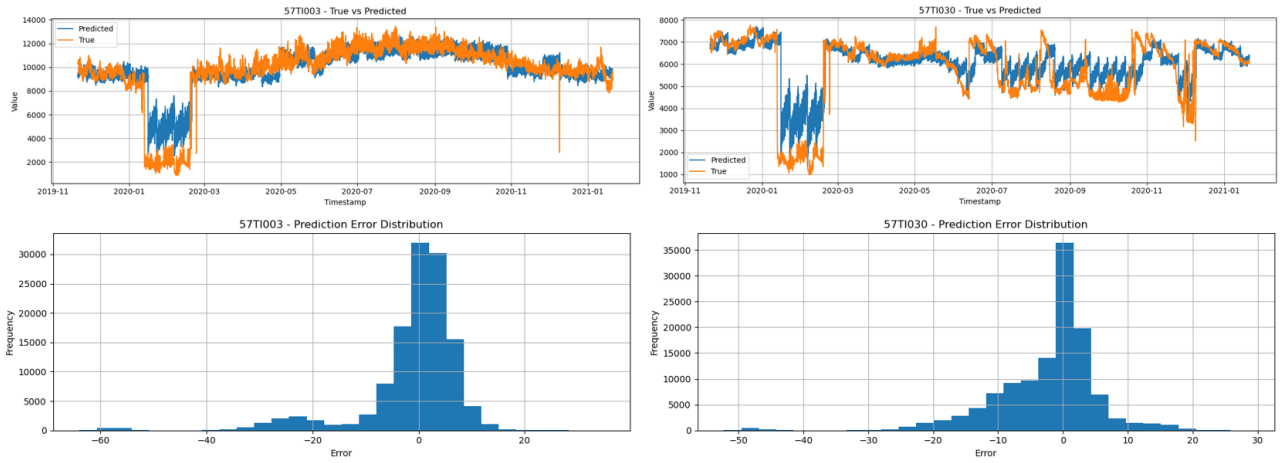


Figure 13: True versus predicted time series and prediction error distributions for K-5701 sensors 57TI003 (left) and 57TI030 (right).

The detailed performance plots for the remaining forecasting models developed in Pilot 7 are provided in Annex F, which includes the true versus predicted time series and prediction error distributions for all other machines and sensors.

The implemented forecasting system represents an improvement over the deployed approach developed by Pilot 7. The previous system often experienced significant deviations from the real sensor behaviour, as it was generating so many false positives that the pilot reported having to disable notifications altogether. In contrast the new developed models demonstrated a better performance with a more consistent alignment, even being able to anticipate anomalies. This supports more accurate operational planning and earlier detection of potential deviations in machine performance.

3.1.2 Technical Specification

3.1.2.1 ML Profiler

Both clustering models for Pilots 1 and 2 were implemented using the KMeans algorithm from the Sklearn[9] library. They were configured to produce four groups of customer segments. Table 12 summarises the main hyperparameters used in each model in the training process, with a brief description of each parameter.

Table 12: Hyperparameter configuration for the KMeans models implemented in Pilots 1 and 2.

Name	Parameter	Pilot 1	Pilot 2	Description / Justification
Number of clusters	n_clusters	4	4	Set based on silhouette and elbow analysis during model selection.
Initialization method	init	k-means++	k-means++	Provides a more stable starting point by spreading initial centroids.
Number of initializations	n_init	10	auto	Values were assigned based on a hyperparameter tuning process.
Maximum iterations	max_iter	1000	1000	Allows the algorithm sufficient iterations to converge.
Tolerance	tol	$1.0 \cdot e^{-8}$	$1.0 \cdot e^{-8}$	Specifies the minimum centroid movement required to stop iterations.
Random state	random_state	42	42	Ensures reproducibility of results.

For both pilots, model evaluation was performed using a 5-fold cross-validation strategy with shuffled splits. Cross-validation was applied to verify the stability of the clustering structure and to confirm that segmentation patterns were consistent across different subsets of the data.

3.1.2.2 ML Risk assessment

The risk assessment model implemented in Pilot 1 UC2 was based on the TabNet architecture. The model was designed to predict the binary target variable *flag_atraso*, identifying customers at risk of payment delay. Given than only 2.6% of the samples were positives, model training incorporated SMOTE resampling to address the significant class imbalance to further balance the learning process.

Hyperparameter tuning was performed using Optuna, optimizing jointly for ROC-AUC and F2-score to favour recall while maintaining reasonable precision. The search space included architecture

[9] https://scikit-learn.org/stable/

parameters, regularization strength, learning rate, and number of decision steps. The configuration yielding the best validation performance is summarized in Table 13.

Table 13: Hyperparameter configuration for the TabNet risk assessment model for Pilot 1 UC2.

Name	Parameter	Value
Number of decision steps	n_steps	6
Decision layer size	n_d	60
Attention layer size	n_a	41
Sparsity regularization	lambda_sparse	0.05218
Feature reusage coefficient	gamma	1.7364
Optimizer function	optimizer_fn	Adam
Learning rate	lr	0.00279
Mask type	mask_type	sparsemax
Learning rate scheduler	scheduler_fn	StepLR (step_size=10, gamma=0.9)
Random state	seed	42

The optimal probability threshold for classification was determined using threshold tuning based on the precision-recall curve, applied to the testing set by maximising the F1-score, which balances precision and recall. This resulted in a threshold of 0.53, meaning that customers with a predicted probability ≥ 0.53 were classified as “at risk”.

At this threshold, the model achieved a recall of 0.62 for the at-risk class, correctly identifying the majority of truly at-risk customers, while the precision for the safe class remained high (98%), providing strong confidence in negative predictions. Although the precision for the at-risk class was low (4%), this configuration was intentionally selected to align with the modelling objective of minimising missed high-risk cases, even at the cost of a higher number of false positives. The model’s detailed performance is documented on Section 4.1.1.2.

3.1.2.3 Information extraction

To facilitate the operational use of the information extraction models in Pilot 4, the solution was deployed as a containerized service using Docker, exposing a REST API implemented with FastAPI. This architecture enables straightforward integration with other systems and provides a reproducible execution environment. The API offers two main endpoints:

Health check

- **Method:** `GET /helloworld`
- **Purpose:** Verifies that the service is running and accessible.
- **Response:** Returns a JSON confirmation message.

Information extraction

- **Method:** `POST /extract_information`

- **Description:** Processes an uploaded PDF and extracts the relevant numerical values based on the specified document type and target sections.
- **Request body parameters:**
  - file: PDF document to process (uploaded file).
  - type: Identifier of the document type, with accepted values: ID, CID, LIC, LICA.
  - sections: List of section indices to process, corresponding to:
    - 0: *Tramo no reembolsable de la ayuda* (non-repayable aid tranche).
    - 1: *Duración de los proyectos* (Project duration).
    - 2: *Anticipos de la ayuda* (Aid advances).
- **Response:** Returns a JSON array of one or more records, depending on the sections specified. Each record contains:
  - The context fields copied from the document row (e.g., "ID. TIPOLOGIA", "TIPOLOGIA", "TIPO AYUDA", "PYME", "COMUNIDAD", "FONDOS PROPIOS"), and
  - The extracted variables for the requested sections (e.g., "Tramo no reembolsable_fondos CDTi (%)", "Duración_min (meses)", "Potencial anticipo max sin aval (%)", etc.).

  A full example of the JSON response is provided in Annex D.

Interactive documentation is available at `/docs` once the container is running.

3.1.2.4 ML Anomaly detection

The anomaly detection models for Pilot 7 were packaged into a Docker container exposing a REST API for real-time anomaly scoring. Once running, the API is accessible through an interactive documentation interface, allowing users to submit sensor data windows and receive anomaly scores without requiring specialised technical skills.

The API accepts numerical sensor readings as comma-separated values for a single fixed-length window of 24 hours at a five-minute interval (288 samples). A valid request must also specify the corresponding sensor type. The output includes an anomaly score and a binary indicator of whether the reading window is considered anomalous. The API consists of three available endpoints:

Service information

- **Method:** `GET /`
- **Purpose:** Verifies that the anomaly detection service is running.
- **Response:** Returns a JSON message with basic service information.

Anomaly prediction

- **Method:** `POST /predict_anomaly`
- **Description:** Generates an anomaly score for the provided time window of sensor readings.
- **Request body parameters:**

  - *sensor_data*: A comma-separated string of 288 numeric readings representing the sensor window.
  - *sensor_type*: Type of sensor to evaluate. Accepted values currently include: "temperature".

- **Response:**
  - *score*: A floating-point value in the range [0, 1] representing the model's estimated probability that the input window is anomalous.
  - *is_anomaly*: Boolean string ("True" or "False") indicating whether the score exceeds the 0.5 anomaly threshold.
- **Error handling:** Returns a JSON error message if the input size is incorrect or the sensor type is invalid.

Model retrieval

- **Method:** `POST /get_model`
- **Description:** Retrieves the architecture of the trained anomaly detection model for the specified sensor type.
- **Request body parameters:**
  - sensor_type: Type of sensor for which the model architecture should be returned.
- **Response:** A JSON string containing the serialised model architecture in *Keras* JSON format.

Interactive documentation is available at `/docs` when the container is running, allowing users to test requests and review parameter formats directly in the browser.

3.1.2.5 ML Forecasting model

The forecasting models were packaged into a Docker container exposing a REST API for prediction requests. Once running, the API is accessible through an interactive documentation interface, enabling users to submit data files and obtain forecasts without specialised technical skills.

The API accepts .csv or .xlsx files containing at least the Timestamp column and the relevant sensor columns for the selected machine. The files must contain a minimum of two weeks of historical data at five-minute frequency, although additional historical rows can be present. Users can select their preferred output format, either JSON for integration into automated pipelines or CSV for direct inspection and reporting. The API provides three main endpoints:

Health check

- **Method:** `GET /helloworld`
- **Purpose:** Verifies that the service is running and accessible.
- **Response:** Returns a JSON confirmation message.

Retrieve available models

- **Method:** `GET /details`

- **Description:** Returns metadata on all available forecasting models, including the machine identifiers and their corresponding sensors.
- **Response:** A JSON object where the keys are machine IDs (e.g., "K-2201", "K-3201") and the values are arrays containing the associated sensor names.

Forecasting

- **Method:** `POST /predict`
- **Description:** Generates a one-week forecast for the selected machine's sensors based on the previous two weeks of historical data.
- **Request body parameters:**
  - *file*: Historical sensor data file (Excel .xlsx or CSV .csv) containing at least:
    - a Timestamp column,
    - one column per required sensor for the selected machine,
    - at least two weeks of data at a five-minute sampling rate.
  - *machine*: Machine ID, with accepted values matching those returned by the /details endpoint.
  - *response_format*: Output format for the prediction results. Accepted values:
    - "file": returns a CSV file for download,
    - "json": returns the predictions as a JSON array.
- **Response:** Returns the forecasted values for each sensor over the next week, in a CSV or JSON format, depending on the response format specified in the parameters. A full example of the JSON response is provided in E.

Interactive documentation is available at `/docs` once the container is running.

3.2 SAX Techniques

The work on SAX techniques from M16 until M33 revolved around four major assets as shown in Table 1 and **briefly** presented below. For further information the reader is referred to the publications as stated in the following sub-sections.

3.2.1 Description

3.2.1.1 SAX open-source library

Causal reasoning is essential for business process interventions and improvement, as it requires a clear understanding of the causal relationships among activity execution times in an event log. The core functionality of the services developed in the SAX4BPM library was presented in D5.1. Since then, the major advancement has been a fundamental extension to the core causal algorithm, expanding the discovery of causal graphs from a single variant to a set of variants. Unifying different variants into a single causal graph introduces challenges, such as handling missing values and expressing the alternating conditions that arise from merging traces with varying activities. Our method preserves the correctness of the original causal models while explicitly representing their causal-flow alternations.

Figure 14 illustrates the output of our unification algorithm using a simple example. By applying the core causal discovery algorithm to variants 1 and 2, we obtain two causal process models. These are then unified into a single model, as shown in the figure. It is important to note that the unification algorithm introduces a new semantic notation to express logical gateways (AND, XOR, and OR).

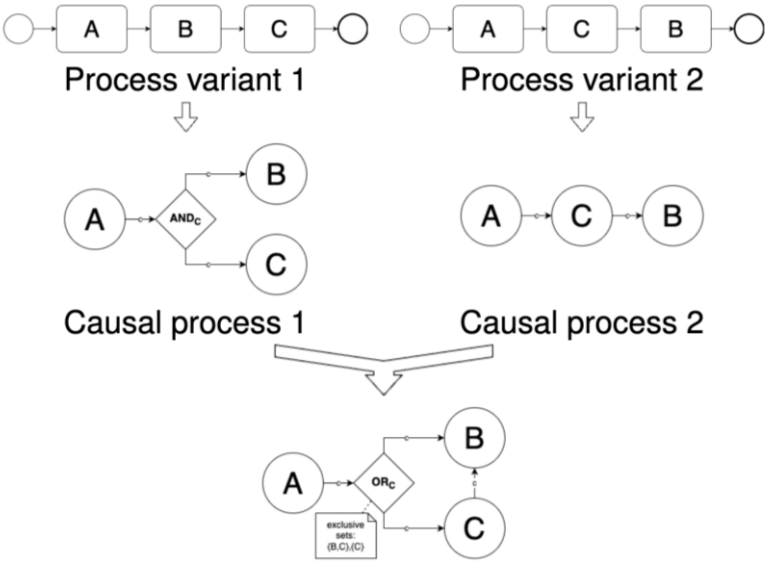


Figure 14: Unification of causal models.

For further details on the core algorithm the reader is referred to [2], and for the extension, to [6, 7].

3.2.1.2 Survey for explanation quality

The SAX4BPM framework developed during the FAME project, takes event logs from business process executions and generates three views about the factors affecting process execution outcomes (see D5.1 in [1]):

- Process models arising from process discovery (PD),
- Causal execution models arising from causal process discovery (CD),
- Feature importance schemes arising from explainable AI (XAI).

These views or knowledge elements are then transformed into textual form and presented as part of a prompt to an LLM, which then generates an explanation in response to a specific user inquiry.

The underlying premise was that the synthesis performed by the LLM yields more sound and interpretable explanations. To test this hypothesis, we carried out a survey on the quality of the explanations generated as perceived by the users.

3.2.1.2.1 First survey

The first step was to conduct a literature survey on the quality of explanations. Based on this survey, we adopted **fidelity** and **interpretability** as the two main latent constructs and subsequently defined six dimensions and three statements under each construct. Figure 15 presents the three statements formulated for each construct. Each statement was rated on a 1 to 7 scale, and together, they were designed to reveal users' underlying perceptions related to the fidelity and interpretability of the explanations.

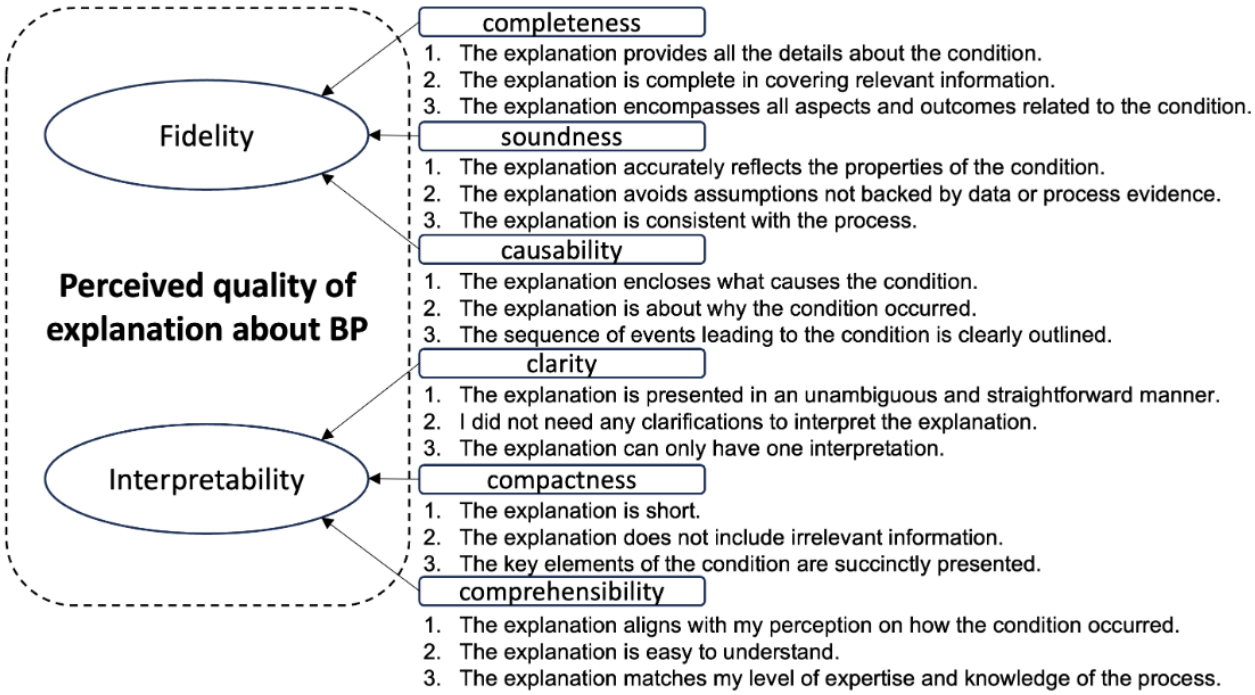


Figure 15: The 18 questions populated for the measurement dimensions, corresponding to each of the two high-level (latent) constructs.

Given the nature of our controlled experiment with voluntary participation, we later also incorporated **trust** and **curiosity** as potential covariates in our analysis. These are treated alongside the two main dimensions, fidelity and interpretability, to provide a more nuanced understanding of how users experience the explanations. Figure 16 presents the six statements formulated to this end.

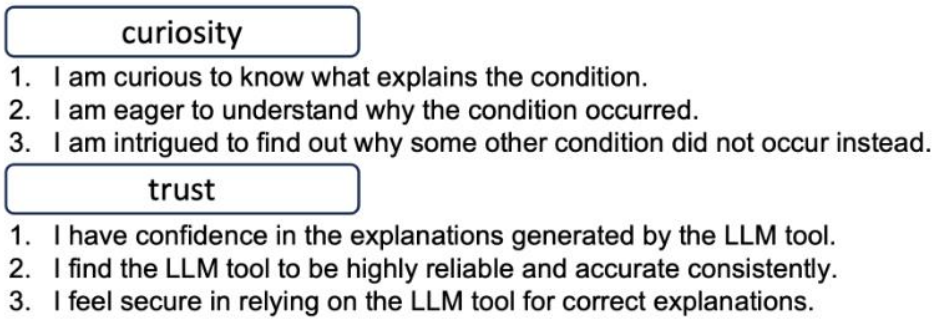


Figure 16: Additional 6 questions populated for the moderating factors (covariates).

Our study involved 50 participants and covered three domains: pizza delivery, parking fines, and loan approval. The first two domains were based on synthetic data, while the loan approval use case used real, publicly available data. We performed several manipulations of the knowledge inputs provided to the LLM, then asked participants to rate a series of explanations using the Likert scale. Our goal was to assess both *fidelity* and *interpretability*, while also considering the roles of *trust* and *curiosity* as experienced by the users. Our results showed that adding knowledge ingredients does, in fact, improve the *perceived fidelity* of the generated explanations. However, contrary to our initial hypothesis, the improved fidelity can come at the cost of perceived interpretability. In other words, while the explanations may seem more accurate, they may also become harder to understand. Moreover, this positive effect on fidelity is not guaranteed. It may disappear entirely when users have

low *trust* in the LLM, and it tends to weaken when users have low *curiosity* about the question or outcome being explained. Still, our developed scale proved to be a valuable tool for assessing how much users appreciate and engage with explanations about different process outcomes.

For a full description of the survey the reader is referred to 3].

3.2.1.2.2 Second survey

As is well known, different LLMs can produce significantly different explanations, even when given the exact same input. With the rapid proliferation of LLMs in the market, selecting the most suitable one for automating process explainability has become a real challenge. It is essential to embed a model into the system that keeps users both engaged and satisfied. To address this challenge, our second survey focused on evaluating how our scale could be used to quantify user perceptions and enable meaningful comparisons between different LLMs.

We launched this second survey to explore how users perceive explanations provided by various LLMs in the context of a tax refund process. A total of 128 participants were asked to rate explanations for different tax refund decisions using our evaluation scale (see Figure 15 and Figure 16).

In this case, our hypothesis was that the perceived quality of explanations generated by different LLMs could support the ranking and comparison of alternative models, ultimately guiding the selection of the most effective one.

For the experiment, we manipulated three variations of different inquiries related to the tax refund process, and employed 4 different LLM models (Granite3, Llama3, GPT 4.o, and FLAN-UL2) to generate the explanations. We selected these LLMs to represent a diverse cross-section of the current LLM landscape in terms of scale (ranging from 8B to 70B parameters), architecture (e.g., encoder-decoder in FLAN-UL2 vs. decoder-only in Llama3), licensing (open-source models like Granite and FLAN vs. proprietary GPT-4.o), and origin (community-driven vs. big tech labs). This selection enables meaningful comparisons that reflect the breadth of LLM design and deployment strategies. For explanation generation, each LLM prompt consisted of blended process descriptions along with an instruction to generate an explanation for a specific user's inquiry, such as**:** "Why was I notified to correct my form and resubmit?"

Figure 17 shows the distribution of average perceived interpretability and fidelity ratings across the four LLM models, with trust and curiosity included as moderating factors.

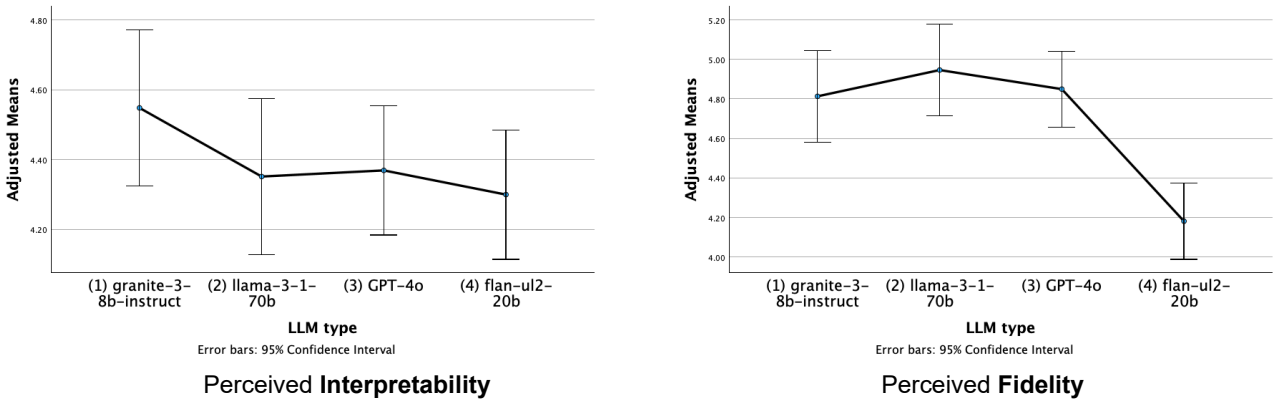


Figure 17: Perceived interpretability and fidelity by users of the explanation quality given by the tested LLMs.

For further information the reader is referred to [5].

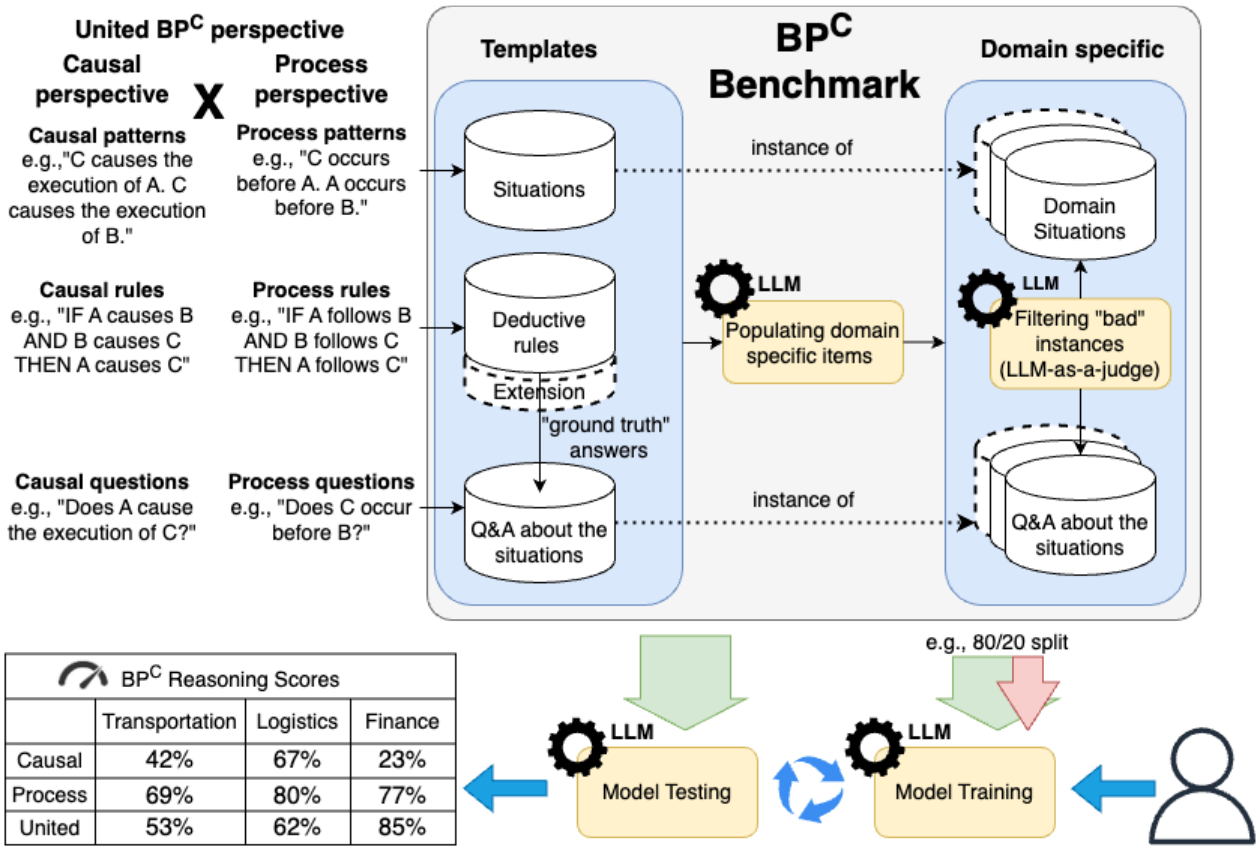
3.2.1.3 BP^C: A benchmark dataset for causal business process reasoning

The question addressed was whether LLMs can reason about Causal Business Processes (BP^Cs) discovered by applying our causal techniques to process event logs. The goal was to develop a benchmark to evaluate the ability of an LLM to reason about BP^Cs. The purpose of the developed novel benchmark is twofold

- Testing dataset that can be employed to quantify the ability of an LLM to reason about BP^Cs.
- Training dataset that can be used to adapt the ability of an LLM for this task.

Textual descriptions of BP^Cs differ from "conventional process descriptions" by also including statements about the causal execution dependencies among the activities as first-class citizens. Our approach relied on defining a core set of template questions and situations for basic reasoning about BP^Cs narratives. These descriptions combine statements about process activities, time precedence relations, and causal execution dependencies.

The overall approach is depicted in Figure 18 and detailed in [4].



BP^C Reasoning Scores

	Transportation	Logistics	Finance
Causal	42%	67%	23%
Process	69%	80%	77%
United	53%	62%	85%

Figure 18: Benchmark generation pipeline.

We picked up the following domains: transportation, manufacturing, logistics, retail, finance, insurance, and medical. Results are shown in Table 14. This table shows the proportion of questions answered correctly by each LLM. The results are split between the template questions and domain-specific questions, and are also partitioned by the various perspectives, considering the causal perspective both without and with the addition of the extending questions.

When training an LLM to improve its reasoning about BP^C, question answers serve as the "ground truth" to determine model accuracy, which can be analysed by perspective and domain. For training,

the questions and answers can be randomly split into training and testing subsets (e.g., 80/20). The testing subset is used before and after training to measure improvement.

Table 14: LLM accuracy results using the prototype.

		Template questions					Domain-specific questions	
	Num of questions	Merlinite 7b	Mixtral instruct 8x 7b	GPT 4o	GPT 4	GPT 3.5	Merlinite 7b	Mixtral instruct 8x 7b
Process	7	58%	71%	100%	85%	65%	100%	100%
Causal	6	66%	100%	100%	100%	56%	98%	100%
Process + Causal	2	100%	100%	100%	80%	15%	55%	95%
Extension	4	50%	50%	70%	65%	55%	73%	50%
Causal+Extension	10	60%	80%	88%	86%	56%	88%	80%
Total weighted avg		**63%**	**79%**	**94%**	**85%**	**55%**	**89%**	**89%**

The benchmark and corresponding prompts are available here: https://github.com/IBM/SAX/tree/main/NLP4BPM2024. Data set open source is available here: https://huggingface.co/datasets/ibm/BPC.

For further information the reader is referred to [4].

3.2.1.4 Explanation of clustering

We have trained XAI models for the k-means classification models developed for Pilots 1 and 2. Concretely, a decision tree, as the one illustrated in the following figure, was trained using orthogonal clusters as an ante-hoc method to generate simple rules that explain specific observations. For instance, the classifications of the two points in Figure 19 corresponding to Pilot 2 can be interpreted through feature thresholding of their recency and monetary values, as derived from the decision tree trained to approximate the k-means machine learning model.

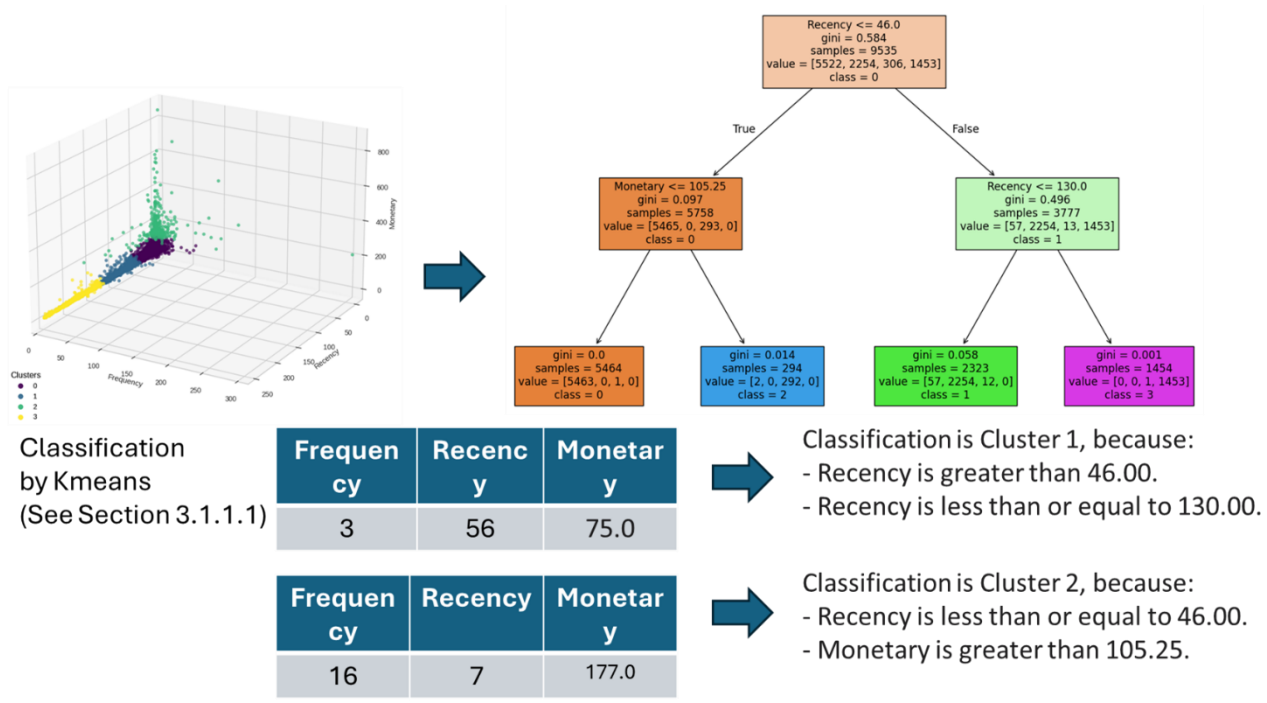


Figure 19: Decision tree explaining the classification model in Pilot 2.

3.2.2 Technical Specification

The SAX4BPM library and related assets as shown is Table 1 were released to the open source. Technical specifications of these assets can be found in D5.1 [1] and in the different publications [23, 4, 5, 6, and 7].

3.2.2.1 Explanation of clustering

In an approach to generating post-hoc explainability to the k-mean models discovered as described in section 3.1, we followed the approach presented in [10] that handles the trade-off between explainability and accuracy. The partitioning is bound by k clusters with maximization of the separation among the cluster yielding an ante-hoc decision tree as the instrumentation for interpretability of the clusters. Such separation among the clusters accommodates tree depth in the population of simplified explanations corresponding to each cluster while retaining classification accuracy. An explaining rule is derived from each cluster corresponding to the path from the tree's root to the respective cluster's tree leaf (i.e., ante-hoc explainability).

With regards to the XAI model developed for the ML models created in T5.1, Figure 20 shows a snippet of the developed code for the XAI models. The code can be found at: https://zenodo.org/records/15622721.

```
import pickle
import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier, export_text, plot_tree
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
def train_and_save_decision_tree(X_train, train_labels):
    """Train a Decision Tree on KMeans labels and save the model and plot."""
    clf = DecisionTreeClassifier(max_depth=2, random_state=42)
    clf.fit(X_train, train_labels)
    with open("decision_tree_model_pilot2.pickle", "wb") as file:
        pickle.dump(clf, file)
    plt.figure(figsize=(15, 10))
    plot_tree(clf, feature_names=X_train.columns, class_names=[str(i) for i in
np.unique(train_labels)], filled=True)
    plt.savefig("decision_tree_plot_pilot2.png")
    plt.close()
    return clf
def load_decision_tree():
    """Load the saved Decision Tree model."""
    with open("decision_tree_model_pilot2.pickle", "rb") as file:
        return pickle.load(file)
def explain_sample(sample, model, feature_names):
    """Generate a clear explanation for why a sample belongs to a cluster."""
    sample = sample.to_numpy().reshape(1, -1)  # Ensure correct shape and retain feature names
    node_index = model.apply(sample)[0]
    decision_path = model.decision_path(sample).toarray()[0]
    explanation = ["This sample was classified into a specific cluster based on the following
conditions:"]
    for i, active in enumerate(decision_path):
        if active and i < len(model.tree_.threshold):
            threshold = model.tree_.threshold[i]
            feature_idx = model.tree_.feature[i]
```

```
        if feature_idx == -2:  # Ignore leaf nodes
            continue
        feature = feature_names[feature_idx]
        if sample[0. feature_idx] <= threshold:
            explanation.append(f"- {feature} is less than or equal to {threshold:.2f}.")
        else:
            explanation.append(f"- {feature} is greater than {threshold:.2f}.")
    explanation.append(f"Final classification: Cluster {model.predict(sample)[0]}.")
    return "\n".join(explanation)
def main():
    # Load the KMeans model
    with open("pilot2_results/model/model.pkl", "rb") as file:
        kmeans = pickle.load(file)
    # Load the datasets
    train_df = pd.read_excel("pilot2_results/customer_segmentation/customer_clustering_assignation_training.xlsx")
    test_df = pd.read_excel("pilot2_results/customer_segmentation/customer_clustering_assignation_testing.xlsx")
```

Figure 20: Snippet of the XAI code.

3.3 XAI Scoring Framework

In this section, we present the advancements of the XAI Scoring Framework during the reporting period.

Our goals from M23 until M34 were:

(i) Extend the metric set with stability and accuracy/precision.
(ii) Introduce qualitative evaluation through GPT-generated virtual personas.
(iii) Benchmark public tabular datasets sourced from the UCL Machine-Learning Repository to re-train and validate the scoring estimator.
(iv) Deliver a dashboard and full REST API for friction-free integration with external pipelines.

3.3.1 Description

The XAI Scoring Framework has undergone evolution since its initial conception in D5.1, transforming from a primarily quantitative evaluation tool into a comprehensive assessment platform that integrates both technical metrics and human-centered evaluation approaches. This enhanced framework addresses the critical gap in XAI evaluation methodologies by combining objective performance measurements with simulated user feedback through innovative virtual persona technology.

The framework's core innovation lies in its dual-assessment methodology. While maintaining rigorous quantitative evaluation of explainability methods through metrics such as fidelity, simplicity, stability, and accuracy, it now incorporates a groundbreaking virtual persona system powered by Large Language Models (LLMs). This system generates diverse user profiles that simulate real-world perspectives on AI explanations, providing scalable qualitative insights without the traditional constraints of human user studies.

The framework evaluates four primary XAI techniques—SHAP (SHapley Additive exPlanations), LIME (Local Interpretable Model-agnostic Explanations), PFI (Permutation Feature Importance),

and PDP (Partial Dependence Plots)—across diverse tabular datasets. Each technique is assessed through multiple dimensions:

- **Fidelity**: Quantifies how accurately the explanation reflects the underlying model's decision-making process.
- **Simplicity**: Measures the cognitive load required to understand the explanation.
- **Stability**: Evaluates consistency of explanations under minor input perturbations.
- **Accuracy**: Assesses the trade-off between model performance and explanation quality.
- **User Satisfaction**: Captures perceived interpretability through virtual persona assessments.

The approach is based on the XAI evaluation strategy presented in D5.1 and further enhanced with the qualitative assessment lever generating virtual personal LLM based on an anthology of backstories and insights from three surveys. The system architecture comprises three primary components: (1) a quantitative evaluation module, (2) a qualitative assessment using virtual personas, and (3) a ML system to generate a general XAI score.

3.3.1.1 System Architecture Overview

- Quantitative Evaluation: Benchmark XAI methods by measuring fidelity, stability, simplicity, and accuracy/precision. A high-level pipeline for this process is shown in Figure 22.
- Qualitative User Assessment: Use GPT-4o-mini to generate virtual personas based on an LLM-generated Anthology of backstories. These virtual personas respond to a tailored questionnaire to capture user preferences and interpretability requirements. This approach, illustrated in Figure 23, generates virtual personas and aggregates their feedback.
- ML estimator: Leverages the characteristics of the dataset to estimate an XAI score based on LightGBM model.

3.3.1.1.1 Quantitative Metrics for Benchmarking XAI Methods

The quantitative evaluation focuses on benchmarking 4 widely used XAI techniques: Permutation Feature Importance (PFI) that quantifies feature relevance through random permutation, Partial Dependence Plots (PDP), which visualizes marginal feature effects, (SHAP) and (LIME)—against tabular datasets. Following the methodology described in D5.1, the key metrics for evaluation include:

- Fidelity: Measures the degree to which an explanation accurately reflects the behavior of the underlying model.
- Simplicity: Assesses the interpretability of the explanation by quantifying its complexity.
- Stability: Evaluates the consistency of explanations when minor perturbations are applied to the input data.
- Accuracy and Precision: Quantifies the trade-off between the performance of the model and the quality of its explanations.

The process for the quantitative assessment and mainly the benchmarking process is presented in Figure 21.

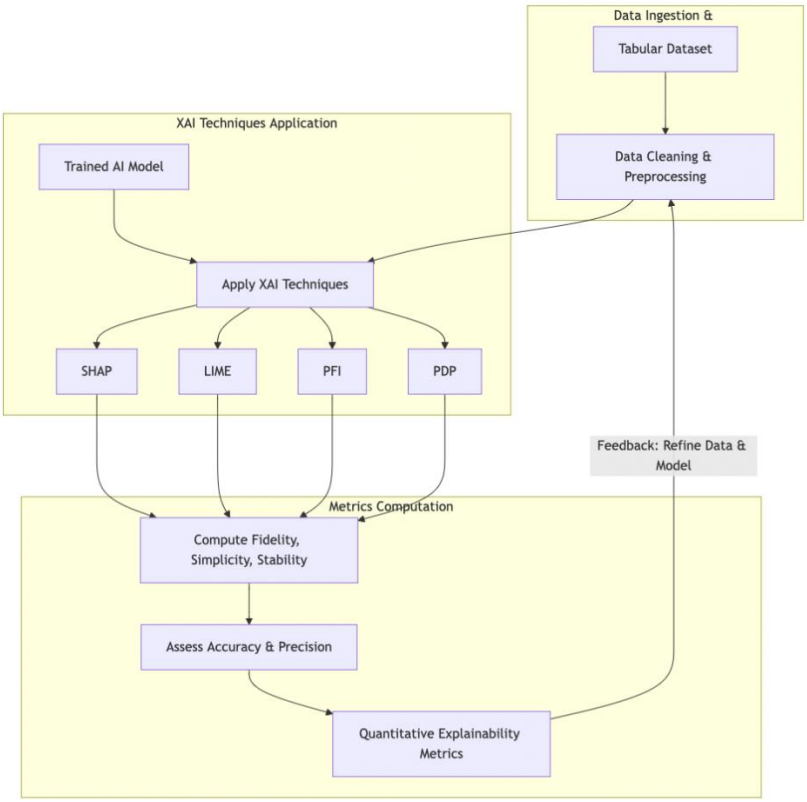


Figure 21: High-level pipeline for the quantitative evaluation of XAI methods. Data is ingested and preprocessed, then a trained AI model is explained using SHAP, LIME, PFI, or PDP.

3.3.1.1.2 Qualitative Assessment

Qualitative assessment through virtual personas addresses several challenges inherent in traditional user satisfaction studies. The motivation came from [8]. Recruiting real end users can be time-consuming, costly, and prone to bias due to inconsistent participation. By generating virtual personas with carefully designed demographic and professional profiles—as outlined in recent research—our process simulates a broad range of real-world perspectives in a consistent and scalable manner. Figure 23 illustrates the process that involves:

- Virtual Personas Generation: Using GPT-4o-mini (via OpenAI API) we generated 1000 backstories by applying open questions. Then we chose in a balanced way 100 out of 1000 backstories (Figure 22 shows a representative subset) and based on them the virtual personas are generated using LLMs, each with a unique demographic and professional profile, including factors such as age, profession, level of AI expertise, and specific explainability preferences.
- Survey Conditioning: Each persona's profile conditions a structured questionnaire aimed at evaluating XAI methods.
- Feedback Analysis: The responses of these virtual personas are aggregated and analyzed according to user satisfaction and preferences, providing qualitative insights into the interpretability of the model explanations.

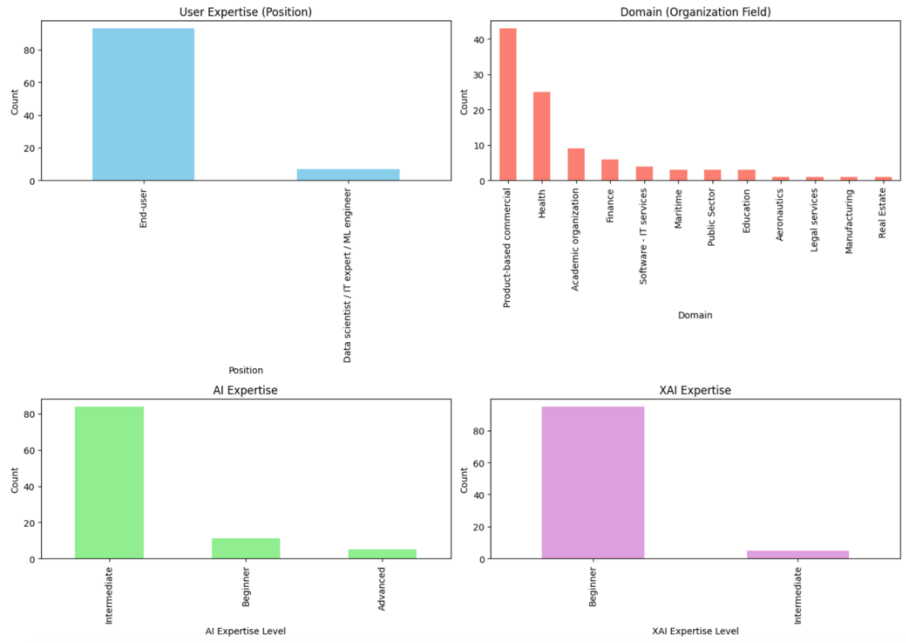


Figure 22: Distribution of various demographics of virtual personas.

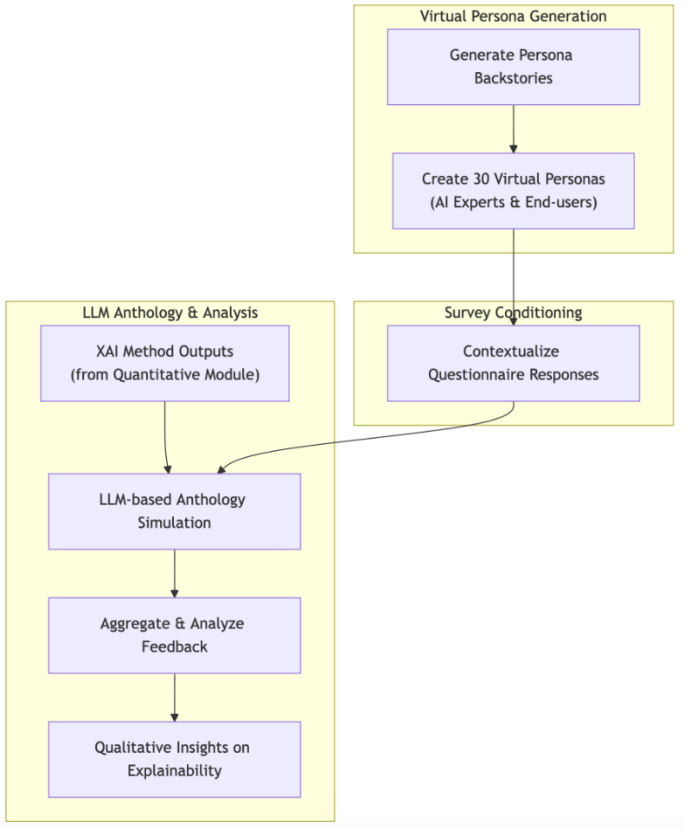


Figure 23: Overview of the qualitative assessment approach. Virtual personas are generated, and their feedback on the XAI outputs is collected through structured surveys.

3.3.2 Technical Specification

The framework is delivered as a containerized microservice and it can be found here: https://github.com/GeorgeMakridis/xai-scoring-framework.

Users can interact with it visually through a web dashboard, or programmatically via a full-featured REST API.

This section provides a step-by-step guide to:

- Cloning and building the XAI Scoring Framework using Docker.
- Running the system as either a web-based dashboard or an API service.
- Understanding the project's internal structure.
- Grasping its core capabilities and modes of operation.
- Reviewing all supported API endpoints in a clear, example-driven table.

3.3.2.1 Cloning and building the Docker Image

First, clone the official repository by executing the following commands, order to clone the repository and navigate into the main folder:

```
git clone https://github.com/GeorgeMakridis/xai-scoring-framework.git

cd xai-scoring-framework
```

The framework includes a `Dockerfile` and `docker-compose.yml` for easy deployment. So, to build and launch both the Web UI and REST API execute the following command that build and starts all the containers:

```
docker-compose up -d
```

As a next step you can visit the framework as follows:

- **Web Interface:** http://localhost:8501
- **API:** http://localhost:8000 (Swagger/OpenAPI docs: http://localhost:8000/docs)

To run services individually:

```
docker run -p 8501:8501 xai-scoring-framework python3 web_app.py

docker run -p 8000:8000 xai-scoring-framework python3 -m uvicorn api.main:app --
host 0.0.0.0 --port 8000
```

As we have already described XAI scoring framework can be used either as (a) a dashboard or via (b) a REST API.

(a) As a Dashboard (Web UI) a user should follow the following steps.

- **Start the UI:** Launch via Docker or locally (`python3 web_app.py`)
- **Access:** http://localhost:8501
- **Workflow:**
  - Upload Benchmark/Rating files (`.xlsx`, `.csv`)
  - Upload a dataset for scoring
  - Select domain and adjust metric weights (fidelity, stability, etc.)
  - View results, recommendations, and metrics in the dashboard

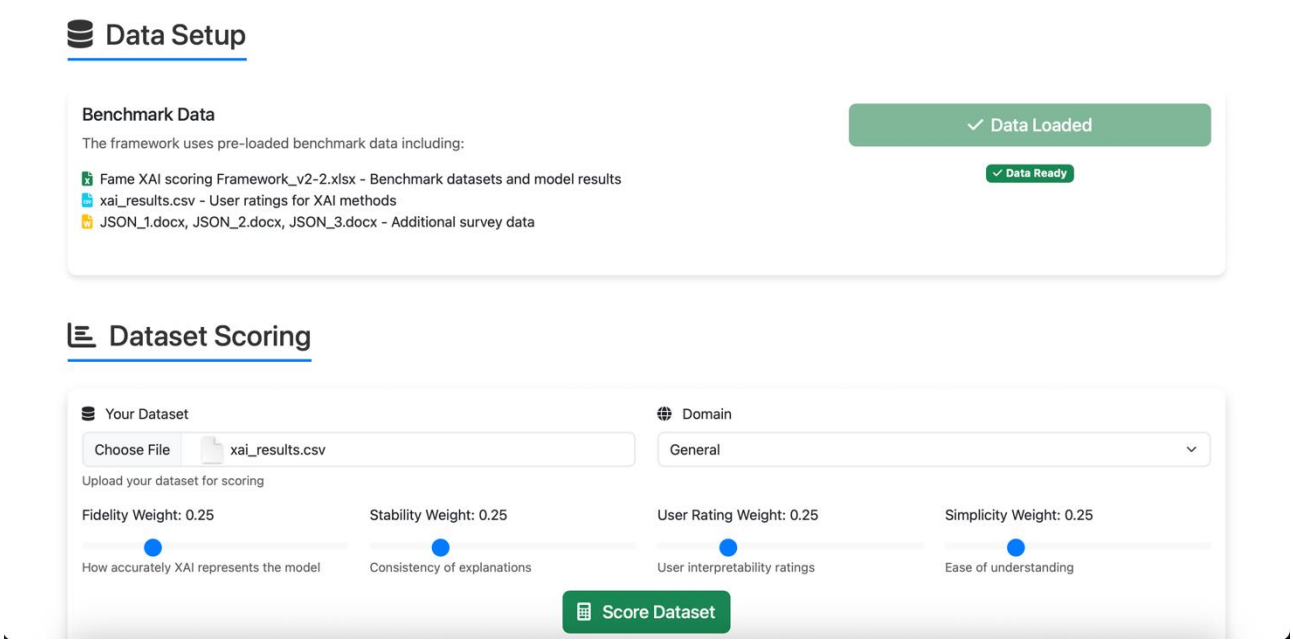


Figure 24: Main UI of XAI scoring framework where the user can upload a dataset and get the XAI score estimation.

(b) As a REST API

- **Start the API:** Launch via Docker or locally (`python3 -m uvicorn api.main:app --host 0.0.0.0 --port 8000`)
- **Documentation:** Complete interactive docs at http://localhost:8000/docs
- **Usage:** Integrate XAI scoring into ML pipelines or external apps

Example: Scoring a new dataset via Python requests:

```
files = {'dataset_file': open('your_dataset.csv', 'rb')}
data = {
'domain': 'healthcare',
'fidelity_weight': 0.3,
'stability_weight': 0.3,
'user_rating_weight': 0.2,
'simplicity_weight': 0.2
}
r = requests.post('http://localhost:8000/score-dataset', files=files, data=data)
print(r.json())
```

Table 15: Description and examples of API endpoints.

Endpoint	Method	Description	Example cURL
/	GET	Show system/API info	`curl http://localhost:8000/`
/health	GET	Health check/status	`curl http://localhost:8000/health`
/load-data	POST	Upload benchmark/rating files (Excel, CSV)	`curl -F 'excel_file=@benchmark.xlsx' -F 'ratings_file=@xai_results.csv' http://localhost:8000/load-data`
/score-dataset	POST	Score a dataset (CSV)	`curl -F 'dataset_file=@your_dataset.csv' -F 'domain=finance' -F 'fidelity_weight=0.4' -F 'stability_weight=0.3' -F 'user_rating_weight=0.2' -F`

			'simplicity_weight=0.1' http://localhost:8000/score-dataset
/score-features	POST	Estimate XAI scores by provided dataset features	curl -H "Content-Type: application/json" -d '{"feature_characteristics": { ... }}' http://localhost:8000/score-features
/api/methods	GET	Get list of supported XAI methods	curl http://localhost:8000/api/methods
/api/domains	GET	List supported domains and bonuses	curl http://localhost:8000/api/domains

Example: Scoring a dataset

```
curl -F 'dataset_file=@your_dataset.csv' \
  -F 'domain=healthcare' \
  -F 'fidelity_weight=0.3' \
  -F 'stability_weight=0.3' \
  -F 'user_rating_weight=0.2' \
  -F 'simplicity_weight=0.2' \
  http://localhost:8000/score-dataset
```

3.4 StreamStory XAI Dashboard

3.4.1 Description

In earlier iterations of the StreamStory XAI Dashboard, our approach to understanding complex industrial processes relied on a screenshot-based method. Transition timelines from our Hierarchical Markov Chain model were captured as images and interpreted by LLMs. This, however, proved to have a significant flaw: language models often produced inaccurate or hallucinated interpretations when processing visual data without its underlying structural context. An analysis that is not reliable is not useful in an industrial setting where safety and efficiency are paramount. To solve this, we developed a new framework that eliminates this unreliable step entirely. Instead, it focuses on direct data extraction from the hierarchical Markov chain models, ensuring our analysis is grounded in the actual process data and providing a trustworthy foundation for generating insights.

3.4.2 Technical Specification

This new pipeline implements an advanced pattern mining and clustering framework that analyses these models to identify operational patterns across multiple time scales. It combines statistical pattern recognition, machine learning, and explainable AI techniques with language model integration to provide human-interpretable insights. The true usefulness of this approach lies in its ability to turn vast, complex sensor data into a clear, strategic asset. For a plant manager or engineer, it acts as an early warning system. Rather than reacting to failures, they can see subtle patterns that point to developing issues, allowing them to move from a reactive maintenance schedule to a truly predictive one. This direct application in predictive maintenance and process optimization, by identifying patterns that precede failures or inefficiency, is where the framework provides its most tangible value.

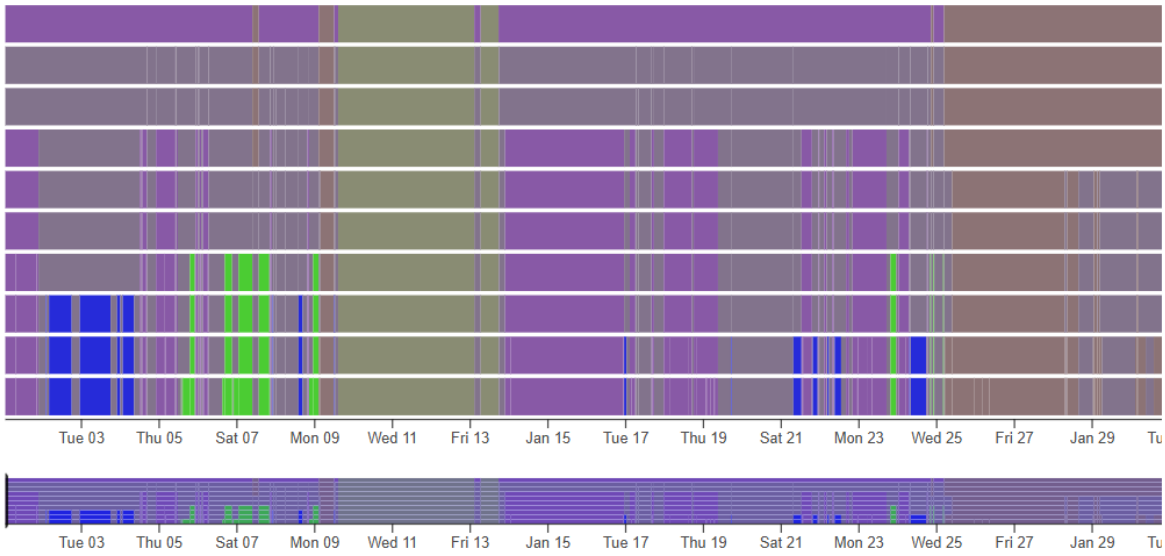


Figure 25: StreamStory state transitions history.

The framework operates in a sequence of phases. It begins by loading the StreamStory JSON model files, which contain the state history (Figure 26), transition probabilities, and hierarchical relationships between states. The pipeline converts this input into a standard format, applying Laplace smoothing to the transition matrices to ensure numerical stability, especially when dealing with rare events. From there, a feature engineering phase uses a sliding window to calculate log-likelihood scores from the Markov chain probabilities, creating behavioral signatures that capture the system's dynamics over time. To ensure these signatures are clear, median filtering is applied to reduce noise, and to preserve the integrity and temporal consistency of the analysis, any time periods with missing data are simply discarded rather than being estimated.

Each temporal scale then undergoes an independent clustering analysis using Agglomerative Hierarchical Clustering. This process automatically determines the optimal number of clusters based on the data's structure, grouping similar operational behaviours together. The quality of these clusters is validated through multiple mechanisms, including silhouette scores to measure cluster separation and cohesion, and reliability scoring to assign confidence levels to each scale's clustering results. The system also performs detailed statistical analysis of each identified cluster, calculating its prevalence to show the percentage of time spent in that state, its duration statistics to reveal its temporal characteristics, and its centroid to define its core behavioural signature. This multi-dimensional characterization provides a complete picture of each operational pattern. The independent analysis of each scale is a key innovation, allowing the system to identify both short-term events and long-term trends, and to see how they might relate to each other through their temporal alignment.

Once patterns are identified, the framework focuses on explaining them in a way that is useful for everyone, from the control room to the boardroom. The intuition here is to first find the "what," then the "why," and finally translate it into "what to do."

First, the PrefixSpan algorithm is used to extract the most frequent sequences of events within each cluster. This reveals the underlying logic of a specific operational state. For a plant operator, this is incredibly useful because it shows the typical "recipe" or path that leads to a certain condition, like an anomaly. By recognizing the start of a known problematic sequence, they can intervene before the issue becomes critical. For a manager, these frequent patterns highlight recurring operational procedures that may be inefficient or risky, providing data-backed evidence for process improvement initiatives.

Next, to understand not just what happened, but why it happened, the system incorporates SHAP. The intuition behind SHAP is to assign credit or blame to each factor that contributed to an outcome. It calculates feature importance scores that pinpoint which specific sensor readings or system states were the primary drivers for a particular pattern. This moves the system from a "black box" to a transparent partner. For an operator, this means no more guesswork; instead of seeing a generic "high temperature" alarm, they can see that a specific sensor is the main cause, allowing them to investigate the right component. For a manager, this helps in resource allocation. If SHAP consistently shows that one particular valve is linked to production slowdowns, it becomes a clear priority for maintenance or replacement.

The final step is an LLM integration that acts as a translator, turning all the technical findings into actionable business insights. The intuition is to bridge the gap between complex data and practical decision-making. The LLM generates human-readable descriptions, categorizes states, and provides context-aware explanations and recommendations by integrating sensor descriptions and domain knowledge. For an operator, this provides clear, unambiguous instructions in plain language. For a manager, it delivers concise summaries and recommendations that can be used directly in reports and strategic meetings, without them needing to be data science experts.

The results of this entire process can be exported as CSV files for traditional analysis or as JSON data to power the interactive dashboard. The deterministic nature of the clustering ensures these results are reproducible, which is essential for any industrial application where consistency is required. Future work will explore the integration of LLM agents to enhance this user-system interaction even further.

The framework's findings are presented in an intuitive and interactive web-based dashboard (Figure 27), designed to make complex data exploration simple for both technical and non-technical users. The interface supports a range of functionalities to facilitate a deep dive into the process data.

Figure 26: Clusters in XAI dashboard.

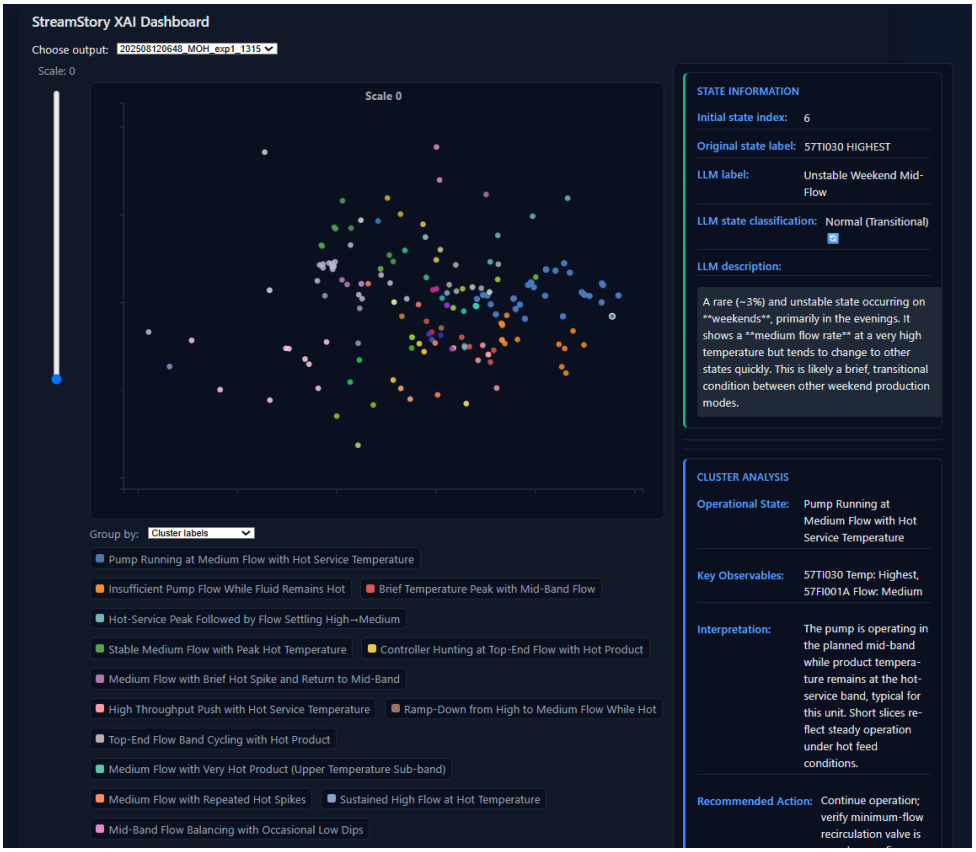


Figure 27: XAI Dashboard- Side menu for selected datapoint, including descriptions and LLM explanations related to current state and cluster.

Figure 28: XAI Dashboard- SHAP explanations and extracted patterns for each cluster in selected scale.

The main view is centred around a 2D scatter plot (Figure 27), which visualizes the high-dimensional clustering results. Each point on the plot represents a moment in time, color-coded by the operational pattern it belongs to. This provides an immediate visual summary of the system's dominant states.

A user can interact with the dashboard in several ways. A dropdown menu allows for the selection and loading of different output files. A vertical scale slider is a key feature, enabling the user to move between different time scales of the analysis. Changing the scale re-renders the entire view, showing patterns that occur over seconds, minutes, or hours, and revealing how short-term events might contribute to long-term trends.

The dashboard is designed for drill-down analysis. When a user clicks on any data point in the scatter plot, a sidebox (Figure 28) is instantly populated with detailed information. This includes the original state label, the more descriptive name assigned by the LLM, the LLM's classification of the state (e.g., Normal, Anomalous), and a detailed, plain-language description of what was happening in the process at that moment. The sidebox also displays the enhanced summary for the entire cluster, providing interpretation and recommended actions. A dynamic legend allows users to toggle the visibility of different clusters, helping to focus the analysis on specific patterns of interest. Furthermore, a "Group by" dropdown lets the user switch the plot's color-coding from the algorithm's cluster labels to the LLM's validation labels, making it easy to spot all anomalous states at a glance.

Below the main chart, a comprehensive Cluster Analysis Statistics section provides a deeper level of detail for each identified pattern. This area displays the prevalence of each cluster, its key statistical features, and visualizations of its feature distributions. Crucially, it also presents the results of the PrefixSpan and SHAP analyses (Figure 28), showing the most frequent sequential patterns that define

a cluster and a bar chart of the feature importances that explain why the system grouped certain behaviours together. This combination of interactive visualization and detailed statistical evidence makes the dashboard a powerful tool for exploring, understanding, and acting upon complex industrial data.

4 Pilots Applications

4.1 ML/AI Analytics

4.1.1 Pilot 1

4.1.1.1 ML Profiler (UC1)

The clustering analysis for Pilot 1 produced four well-differentiated customer segments based on their Recency, Frequency, and Monetary (RFM) scores. The spatial distribution of these clusters in the RFM feature space is shown in Figure 29, presenting a 3D scatter plot of the four clusters using the training dataset. This version was chosen instead of the testing dataset because the larger number of points in the training set produces a clearer and more continuous visualization of the cluster boundaries. The same distribution pattern was observed in the testing dataset, and its corresponding plot is provided in Annex A.

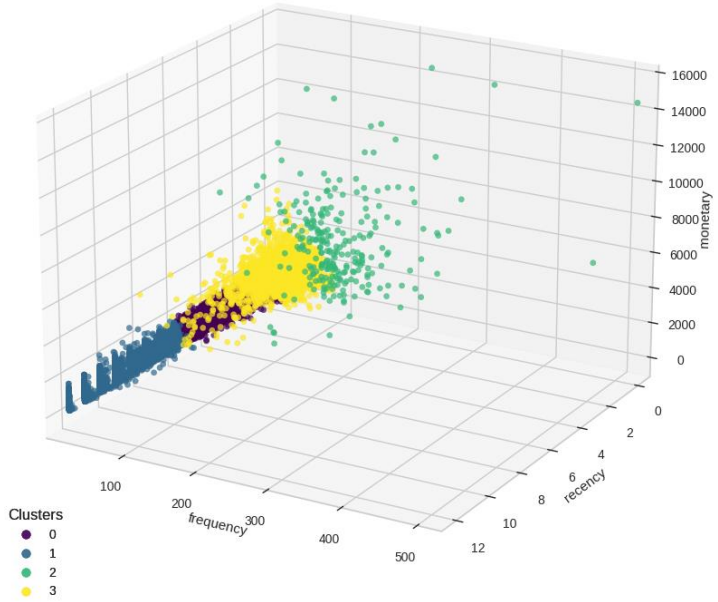


Figure 29: 3D scatter plot of the four clusters in the Pilot 1 UC1 RFM feature space using the training dataset.

To explore the differences between clusters in each individual metric, boxplots were generated for the RFM values (Figure 30) based on the training dataset, to visualize the differences following Figure 29. Analysis of each metric individually revealed distinct behavioural profiles:

- Cluster 0 consists of moderate users with room to increase their engagement and value.
- Cluster 1 exhibits very high recency values (indicating a long time since last activity) and low engagement, consistent with inactive customers.
- Cluster 2 contains high-frequency, high-monetary customers, representing the most valuable and engaged segment.
- Cluster 3 is a small group of low-activity customers with limited transactions and spending.

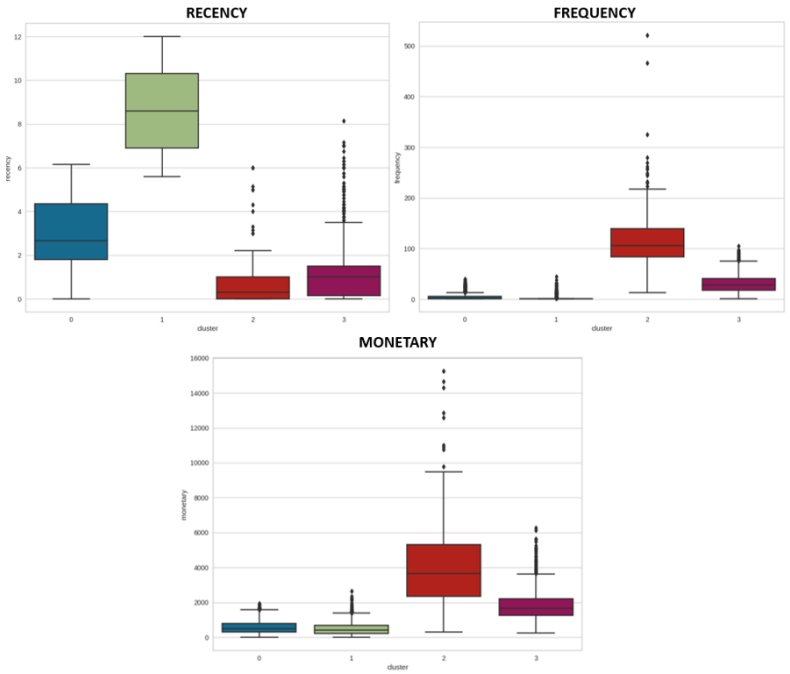


Figure 30: Boxplots illustrating the variation of Recency, Frequency, and Monetary values across the four identified clusters for Pilot 1 UC1 using the training dataset.

The four customer clusters formed the basis for targeted behavioural analysis and the design of a recommendation system. To better understand the characteristics and preferences of each cluster, the pilot team defined a set of business-driven questions, and the relevant variables were analysed using the full customer dataset. For each cluster, the proportion of customers exhibiting each possible value was calculated, providing probability estimates that offer actionable insights to support decision-making, personalise offers, and prioritise engagement strategies.

For example, Table 16 answers the question *"Which customers are more likely to subscribe to a light plan?"*. The analysis shows that customers in Cluster 3 have the highest adoption rate (13.68%), while Clusters 0 and 2 have notably lower rates (2.93% and 3.68% respectively).

Table 16: Percentage of customers subscribed to a light plan per cluster assignation.

light_plan	Cluster 2	Cluster 0	Cluster 1	Cluster 3
0	96.07%	97.07%	91.83%	86.32%
1	3.68%	2.93%	8.17%	13.68%

Similarly, Table 17 and Table 18 address the question *"Which customers are more likely to use the card online?"*. The results are presented separately for the last 12 months (Table 17) and the last 3 months (Table 18), enabling temporal comparison of online transaction behaviour.

Table 17: Percentage of customers using a card online per cluster assignation in the last 12 months.

online_value_l12m_binned	Cluster 2	Cluster 0	Cluster 1	Cluster 3
0-55	80.0%	89.0%	60.0%	34.0%
55-314	9.0%	6.0%	21.0%	45.0%
314-14295	10.0%	6.0%	20.0%	21.0%

Table 18: Percentage of customers using a card online per cluster assignation in the last 3 months.

online_value_l3m_binned	Cluster 2	Cluster 0	Cluster 1	Cluster 3
0-10	82.0%	86.0%	67.0%	42.0%
10-118	9.0%	6.0%	18.0%	25.0%
118-3966	9.0%	7.0%	15.0%	34.0%

These examples illustrate how the clustering model enables the extraction of customers' insights. By associating behavioural probabilities with each customer segment, the organization can prioritize marketing actions, optimize resource allocation, and design personalized product recommendations that address the specific needs and behaviours of each group.

4.1.1.2 ML Risk assessment (UC2)

The results presented in this section refer to the final TabNet model developed for Pilot 1 UC2, aiming to predict the variable *flag_atraso*. This model was optimized to maximize recall, aiming to identify as many at-risk customers as possible while managing the trade-off with precision.

Table 19 presents the classification metrics for the TabNet classifier evaluated on the test dataset using a threshold optimized for high recall. The following metrics are included:

- **Precision**: For class 1 (True), precision is 0.04, meaning that only 4% of customers predicted as at-risk were truly at-risk. This indicates that most customers flagged will require further verification before taking action, as many will be false positives. In contrast, class 0.0 (False) achieved a high precision of 0.98, meaning that almost all customers predicted as safe were indeed safe, providing a high level of confidence in negative predictions.
- **Recall**: For class 1, the model correctly identified 62% of all actual at-risk customers. This is important because it ensures that most high-risk cases are detected. For class 0, the model correctly identified 57 % of safe customers, with the remainder being incorrectly flagged as at-risk.
- **F1-score**: For class 1, the low F1-score (0.07) reflects the imbalance between high recall and very low precision; the model prioritises finding at-risk customers even if that means many false alarms. For class 0, the F1-score is 0.72, showing a better balance between high precision and moderate recall for safe customers.
- **Support**: The number of true samples for each class in the dataset, highlighting the strong class imbalance.
- **Balanced accuracy**: 0.57, the average recall across both classes, offering a fairer view of model performance in imbalanced scenarios.
- **Macro average**: The unweighted average across both classes, treating them equally regardless of number of class samples.
- **Weighted average**: Each metric averaged and weighted by class size, which is dominated by safe customers and therefore higher for precision and accuracy.

Table 19: Classification report of Pilot 1 UC2 TabNet model.

Metric	Precision	Recall	F1-score	Support
Class 0 (False)	0.98	0.57	0.72	1956
Class 1 (True)	0.04	0.62	0.07	53
Balanced accuracy			0.57	2009

Macro Avg	0.51	0.60	0.40	2009
Weighted Avg	0.96	0.57	0.71	2009

When applied to the testing dataset, the model correctly identified 33 high-risk customers out of 53 actual cases. The detailed distribution of predictions is shown in the confusion matrix in Table 20.

Table 20: Confusion matrix of Pilot 1 UC2 TabNet model.

	Predicted False	Predicted True
Actual False	1121	835
Actual True	20	33

This recall-oriented configuration was intentionally selected to minimize the risk of missing potential default cases, even at the cost of a higher false positive rate (835 cases). While precision for the positive class was low (0.04), the approach ensures that most at-risk customers are flagged for further review.

Figure 31 illustrates the prediction output for each customer in the dataset, with an additional column indicating the model's predicted risk flag. This view enables direct inspection of individual predictions and facilitates further analysis by the pilot.

Number of dependent	Age	Nationality	Profession Monthly Expenses	Requested Amount	Profession	Housing Type	Net Monthly Income	Fixed Monthly Charges	Postal Code	Number Financial Commitments	Universo Monthly Payment	UpdatedOn	Fraude
0.0	27.0	PRT	284.32	1750.0	EMPREG. ESCRITORIO/COMERCIO	LOCATARIO	1750	512,97	4580245.0	20	0	2024-04-12 16:00	FALSE
0.0	46.0	PRT		5500.0	EMPREG. ESCRITORIO/COMERCIO	ALOJADO PELA FAMILIA	1156,17	284,32	5000584.0	82,9435	0	2024-04-11 15:00	FALSE
2.0	40.0	PRT	547.38	100.0	EMPREG. ESCRITORIO/COMERCIO	PROPRIETARIO	2333,33	734,64	4400478.0	524,332	0	2024-03-14 15:30	FALSE
0.0	24.0	PRT	234.44	1500.0	OPERARIO/TECNICO	PROPRIETARIO	921,67	234,44	4630296.0	0	0	2024-04-12 13:00	FALSE
0.0	58.0	PRT		500.0		ALOJADO PELA FAMILIA	583,33	123,99	2530535.0	27,25	0	2024-03-15 12:13	FALSE
0.0	25.0	PRT		2000.0	EMPREG. ESCRITORIO/COMERCIO	ALOJADO PELA FAMILIA	1061,67	278,16	2565697.0	175,5535	0	2024-04-17 15:00	FALSE
1.0	31.0	PRT		729.0	MOTORISTA/SEGURANCA	ALOJADO PELA FAMILIA	1053,5	514,86	2530134.0	391,8	0	2024-03-15 16:43	TRUE
0.0	21.0	PRT	368.93	2000.0	OPERARIO/TECNICO	ALOJADO PELA FAMILIA	1108,33	368,93	2670488.0	40	0	2024-03-20 17:49	FALSE
0.0	53.0	PRT		1512.0	OPERARIO/TECNICO	ACESSO A PROPRIEDADE	2566,67	566,72	2560038.0	657,7	0	2024-03-16 10:44	FALSE
1.0	44.0	PRT		1041.0	OPERARIO/TECNICO	PROPRIETARIO	1166,67	665,59	4635751.0	140,67	0	2024-03-16 11:17	TRUE
1.0	41.0	PRT	664.11	1000.0	EMPREG. ESCRITORIO/COMERCIO	ACESSO A PROPRIEDADE	1120	890,36	2725567.0	745,9	0	2024-05-10 00:00	FALSE
0.0	26.0	PRT		1500.0	OPERARIO/TECNICO	ALOJADO PELA FAMILIA	1062,83	504,91	3700244.0	189,5	0	2024-04-14 12:00	FALSE
0.0	66.0	PRT	940.87	1495.0		LOCATARIO	1750	603,32	2560061.0	321,509	0	2024-03-17 12:38	FALSE

Figure 31: Fraud prediction results for each customer in the dataset, with the predicted risk flag appended as a new column.

Although the dataset presented severe class imbalance and weak correlation between features and the target, the final TabNet model demonstrated an ability to capture relevant patterns and identify high-risk customers with moderate recall. These results provide a viable baseline for a credit risk scoring system. Future improvements, such as integrating more predictive features, incorporating behavioural and transaction-level data, or increasing the representation of the minority class, are expected to enhance both recall and precision.

4.1.2 Pilot 2

The clustering results obtained in Pilot 2 formed the basis for analysing customer behaviour in relation to parking habits in Athens. Building the segmentation using the RFM metrics enabled the grouping of customers with similar activity levels, spending patterns, and engagement frequency into four distinct segments. Figure 32 presents the distribution of customers in the RFM feature space for the training dataset. This dataset was chosen as its larger sample size provides a clearer analysis of the cluster boundaries and density. The testing dataset exhibits a similar pattern, and its corresponding plot can be found in Annex B.

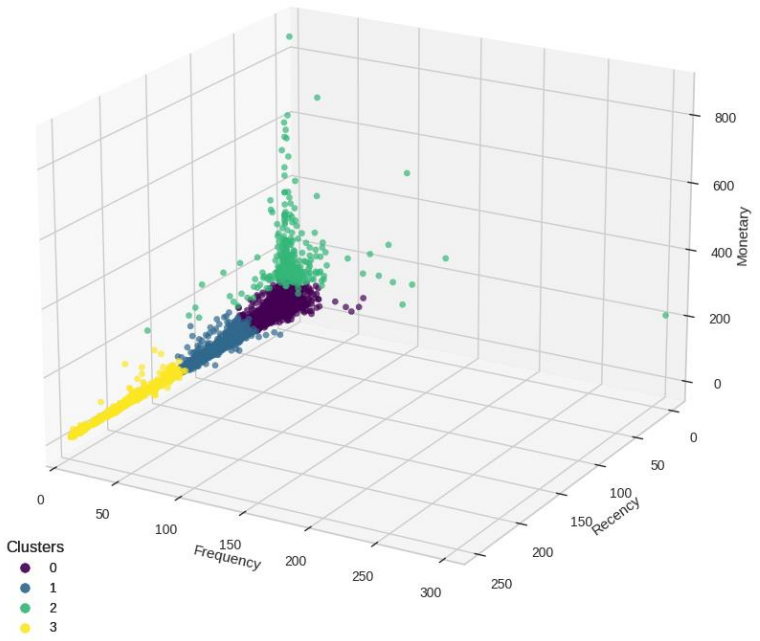


Figure 32: 3D scatter plot of the four clusters in the Pilot 2 RFM feature space using the training dataset.

To better understand how each metric contributes to the segmentation, boxplots of Recency, Frequency, and Monetary values were generated (Figure 33) using the training dataset. These visualizations allow for a better interpretation of the differences between clusters following the overall distribution shown in Figure 32. The four clusters identified exhibited the following profiles:

- Cluster 0 consists of low-usage customers with very limited spending, reflecting minimal interaction.
- Cluster 1 groups customers who have not engaged with the service for a long period (high recency) and contribute little revenue, indicating possible lost customers.
- Cluster 2 represents the most active and profitable users, characterized by frequent transactions and the highest overall spending.
- Cluster 3 contains occasional users who generate moderate spending and interact periodically, offering potential for targeted engagement.

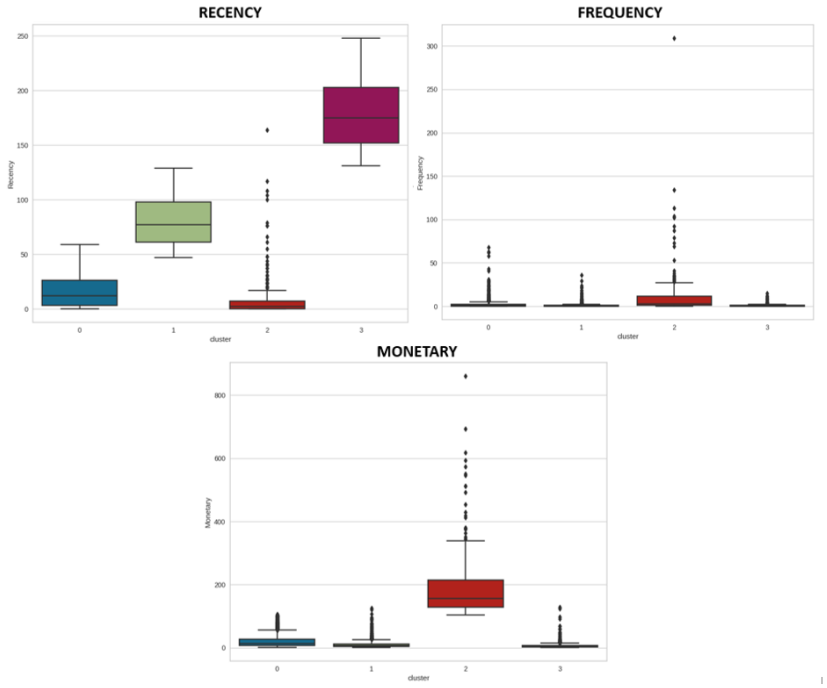


Figure 33: Boxplots illustrating the variation of Recency, Frequency, and Monetary values across the four identified clusters for Pilot 2 using the training dataset.

Having access to historical parking transactions for each customer, an analysis took place to determine the most frequently used parking zones for each cluster. This information provides valuable input for location-specific marketing actions, such as discounts offerings or underutilized areas engagement. The outcome of this analysis is presented in Figure 34, which illustrates the number of customers in each cluster using each parking zone in the testing dataset.

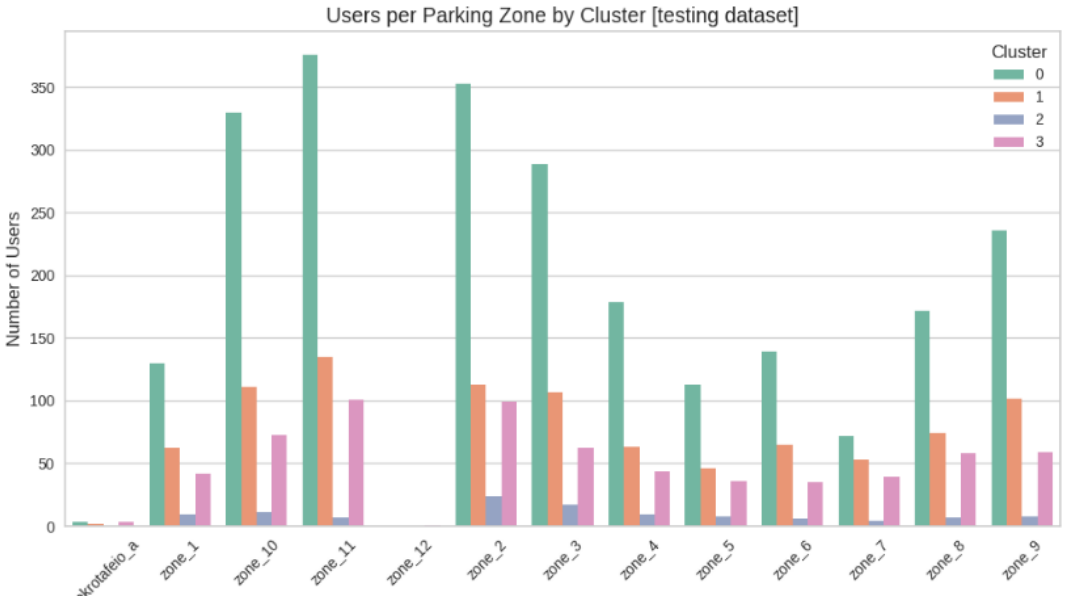


Figure 34: Most used parking zones in Athens per customers based on the cluster assignations.

The distribution reveals clear differences in location preferences between clusters, with certain zones attracting a high concentration of active and high-value users, while others are predominantly used by low-engagement customers.

4.1.3 Pilot 4

The models developed in Pilot 4 were designed to automatically extract specific numerical variables from CDTI PDF documents and return them following a specific Excel format. Each model was dedicated to a single variable, using a question-to-value approach.

Internal validation on the real pilot dataset demonstrated that four out of the five implemented extractors achieved perfect accuracy. As shown in Table 21, the "Minimum duration" model achieved 98.03% accuracy (149 correct predictions out of 152 test samples), while "Maximum duration", "Minimum disbursement", "Maximum advance without guarantee", and "Maximum advance with guarantee" all reached 100% accuracy. Some models benefiting from "soft" lemmatization and others performing best without any lemmatization, but all of them were truncated to a maximum of 1024 tokens.

Table 21: Accuracy of the trained information extraction models on the Pilot 4 test dataset, showing the best-performing configuration for each variable (with or without lemmatization).

Model	Lemmatized	Performance
Minimum duration	No	98,03%
Maximum duration	No	100%
Minimum disbursement	No	100%
Maximum advance without guarantee	Soft lemmatization	100%
Maximum advance with guarantee	Soft lemmatization	100%

The remaining variables were not implemented successfully for specific reasons.

- For "Non-repayable CDTI funds tranche" and "Non-repayable European funds tranche", the relevant sections were lengthy and heavily tabular, containing numerous conditional cases. Meeting the 1,024-token limit required truncation that removed essential information, while hard lemmatization altered key phrases and reduced the model's ability to interpret the content.
- For "Maximum advance" and "Maximum disbursement", no consistent extraction could be achieved despite multiple preprocessing approaches. The most plausible cause is that the synthetic dataset generation, while effective for simpler cases, did not fully capture the structural and semantic variability present in the PDFs for these variables. Additional real PDF samples and pre-labelled Excel outputs would likely be needed to build models robust enough for these fields.

Given these constraints, expanding the dataset with more annotated examples would be the most viable route to improve coverage. However, the effort required to source and annotate such material is non-trivial. Despite these limitations, the strong performance of the successfully implemented models demonstrates that the pipeline is highly effective for structured extraction and can be extended to additional variables once sufficient training data are available.

4.1.4 Pilot 7

4.1.4.1 ML Anomaly detector

The anomaly detection models developed in Pilot 7 are designed to automatically identify abnormal behaviour in industrial machinery based on sensor data. These models analyse temperature sensor readings over fixed 24-hour periods, detecting deviations from regular patterns that may indicate potential faults or maintenance needs.

To facilitate end-user interaction, the models were integrated into a web-based application implemented using Dash. It easily allows visualisation of detected anomalies without requiring programming knowledge or manual execution of scripts. The application connects to the deployed anomaly detection API and provides an intuitive interface for anomaly review.

The interface, shown in Figure 35, allows users to:

- Select the machine from the available list (left drop-down menu).
- Specify the date to evaluate.
- Choose the sensor to analyse from the corresponding machine (also a drop-down menu).

Once the parameters are selected, the system retrieves the anomaly detection results for the chosen machine, date, and sensor, displaying them on the right-hand panel. When anomalies are detected, they appear as spikes or deviations in the plotted signal, enabling operators to quickly identify the exact times at which abnormal behaviour occurred.

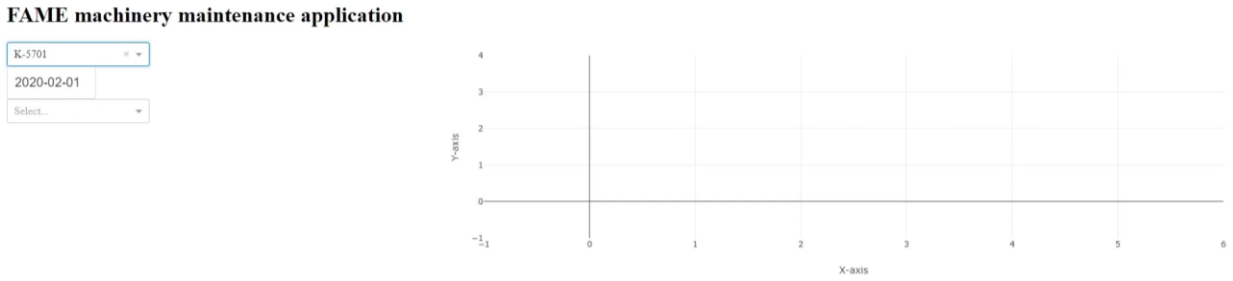


Figure 35: Web interface for anomaly detection, enabling selection of machine, date, and sensor (left panel) and displaying detected anomalies (right panel).

An example is presented in Figure 36, showing the results for sensor 57TI003 of the K-5701 machine, on 1 February 2020. The anomaly detection model identified four distinct anomalies during the day. These anomalies are clearly visible as sharp spikes in the sensor readings, indicating sudden deviations from the normal operating range. This visualisation allows maintenance teams to focus on specific time intervals for further investigation, reducing diagnostic effort and enabling proactive intervention.

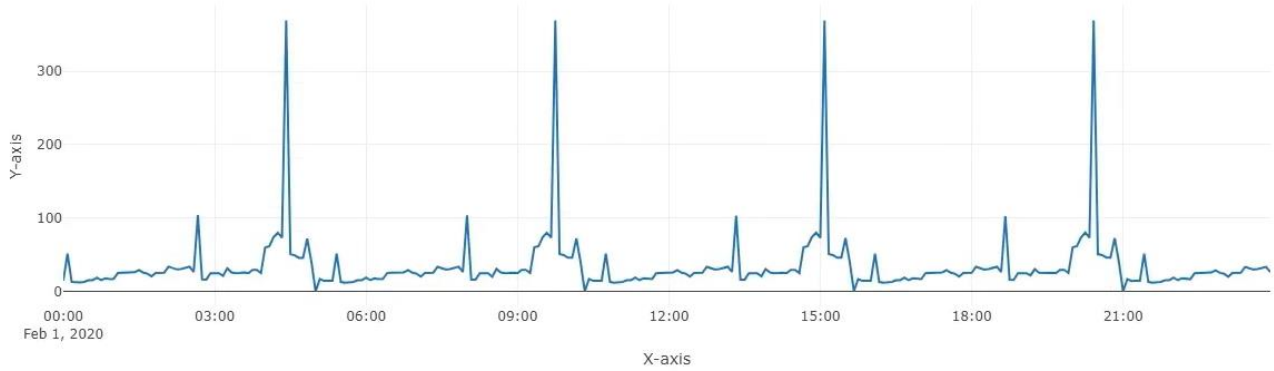


Figure 36: Example anomaly detection results for machine K-5701, sensor 57TI003, on 01/02/2020, with four detected anomalies.

4.1.4.2 ML Forecasting

The forecasting models in Pilot 7 are LSTM-based multivariate predictors designed to estimate the next week's temperature sensor values using the preceding two weeks of high-frequency historical data. Quantitative evaluation using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) confirmed that the implemented models effectively capture the temporal dynamics of the temperature sensors across all machines. Table 22 summarises the performance metrics for each sensor across the five models.

Table 22: Performance of the Pilot 7 forecasting models. Values show Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) for each sensor over the test period.

Machine	Sensor	MAE	RMSE
K-2201	22TI118	365.67	597.28
	22TI42	377.28	566.61
K-3201	32TI444	625.95	807.02
	32TI439	439.13	523.74
K-3301	33TI610	222.09	296.35
K-5701	57TI030	616.42	949.66
	57TI003	748.41	1244.10

Table 22 summarises the performance metrics for each sensor across the five models. The lowest errors were achieved for machine K-3301 (sensor 33TI610), the only model trained to forecast a single sensor. This narrower prediction scope likely improved performance by allowing the model to specialise in one signal's temporal patterns (Figure 53). In contrast, K-5701 displays the highest MAE and RMSE values. However, these must be interpreted in the context of its much higher operational range, as seen in Figure 37, with values fluctuating between roughly 2,000 and 14,000. For K-3201, the larger error for sensor 32TI444 aligns with its time series plot (Figure 52), where predictions diverge more from the ground truth during high-variability periods, resulting in a broader and more asymmetric error distribution. By comparison, K-2201 (Figure 51) achieves lower errors for both sensors, aided by more stable operational trends and fewer abrupt shifts.

Across all cases, the implemented forecasting models demonstrate strong generalisation capabilities across diverse machines and sensor behaviours, with consistent trend-following performance. The analysis also shows that the models are able to capture anomalous events when there is a progressive change before the event. For example, in sensor 57TI003 (Figure 37), the first anomaly defined as values dropping below the red threshold line, was correctly predicted, as the forecast also crossed the threshold. However, the second anomaly, caused by a sudden and sharp drop, was missed, reflecting the difficulty of predicting abrupt, non-repetitive events underrepresented in the training data.

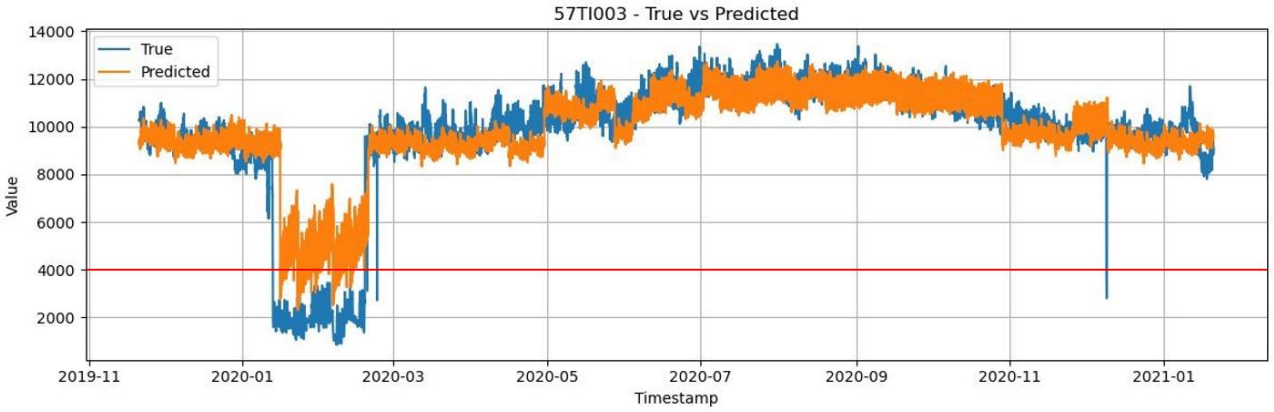


Figure 37: True versus predicted values for sensor 57TI003, illustrating anomaly prediction. The red horizontal line indicates the anomaly threshold.

While the models are not perfect, the results indicate that they robustly follow the trends of all monitored sensors, successfully reproducing the patterns of the machines. This performance makes the forecasting system a reliable tool for short-term planning and anomaly anticipation.

4.2 SAX Techniques

4.2.1 Pilots 1 and 2

The result decision tree models trained on the two ML k-means models developed for Pilots 1 and 2 are illustrated below.

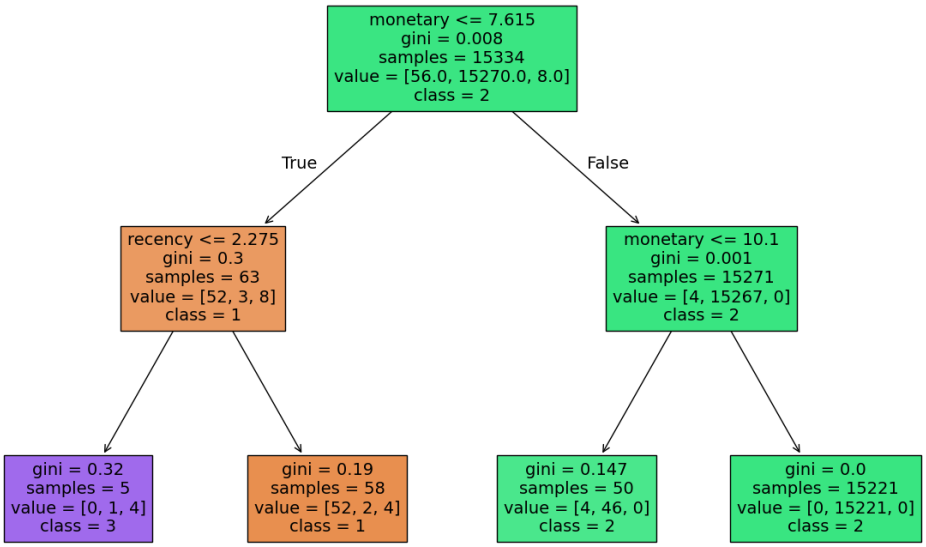


Figure 38: Decision tree explaining the classification model in Pilot 1.

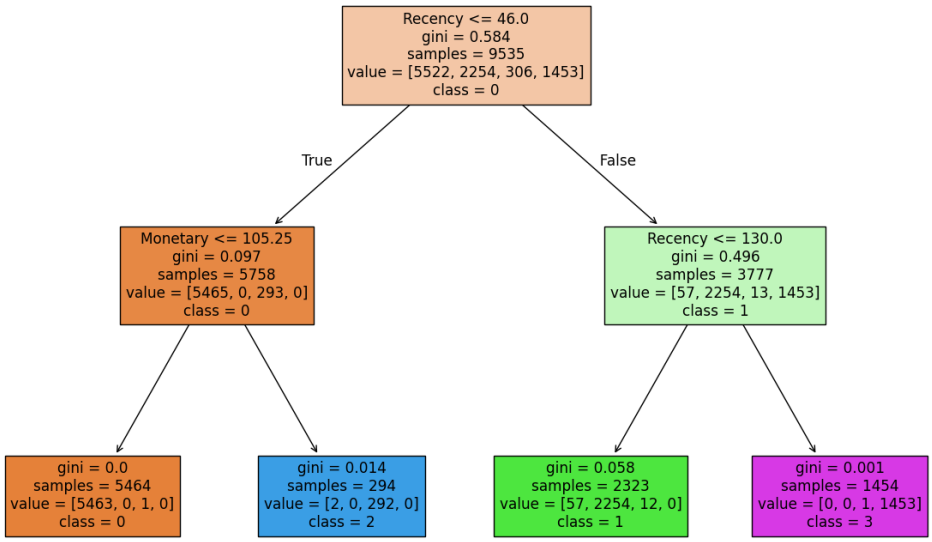


Figure 39: Decision tree explaining the classification model in Pilot 2.

4.2.2 Pilot 7

The methodology for generating transaction data suitable for causal discovery operates through a multi-stage pipeline that transforms raw multivariate time series into structured transaction events. This approach uses the StreamStory Markov chain framework to create a representation that captures temporal dependencies and state transitions in a format suitable for causal inference algorithms.

The process begins with data preprocessing and a hybrid feature selection approach, which is a necessary preliminary step to ensure the quality of the subsequent analysis. By combining multiple criteria, it identifies the most relevant variables for causal analysis. Variance thresholding removes features with minimal variation, as static or near-constant variables provide little information about temporal dynamics. To prevent multicollinearity issues that could obscure true causal relationships, a correlation-based selection step eliminates redundant features. Finally, an SVD-based selection uses singular value decomposition to identify features that capture the most significant linear variance, ensuring the selected feature set has sufficient informational content for the modeling stage.

Once the feature set is optimized, the methodology employs the StreamStory framework to construct Markov chain models that capture the temporal evolution of the system. These models operate at multiple scales, a feature that is particularly useful for causal discovery as it enables the identification of relationships that may manifest only at specific temporal resolutions, such as short-term correlations versus long-term trends. The number of initial states is a key parameter that controls the granularity of the state space, with more states providing finer resolution at the cost of potentially higher computational complexity.

The core of this methodology is the transformation of the Markov chain state sequences into transaction data. Instead of treating the time series as a continuous flow of observations, the system identifies specific states that serve as transaction boundaries. Each transaction represents a sequence of states that begins at a designated target state and continues until the next occurrence of that same state. This transformation is what converts the continuous temporal data into discrete, bounded sequences that can be analyzed effectively using causal discovery techniques.

The transaction generation process applies these cuts at specific states across multiple scales, creating a hierarchical representation where each scale can reveal different aspects of the system's dynamics. A filtering mechanism is implemented to ensure that each generated transaction set contains significantly more transactions than unique states. This criterion provides the necessary data density for reliable causal inference while maintaining the statistical validity of the analysis.

The resulting data structure for each transaction includes both entry and exit events, along with metadata such as scale indices, state identifiers, and timestamps. This dual-event representation enables the analysis of both the initiation and completion of causal sequences, providing richer information about the temporal characteristics of the relationships. The system also maintains mappings between the abstract state labels and more interpretable identifiers, which facilitates the final interpretation of any discovered causal patterns.

The final output consists of structured transaction files that can be used directly with various causal discovery algorithms. These files contain the temporal sequences, state transitions, and associated metadata in a standardized format, simplifying integration with other causal inference tools. The multi-scale nature of the transactions allows for a more comprehensive investigation of causal relationships at different temporal resolutions, from fine-grained, immediate effects to long-term causal patterns. This methodology provides a robust approach to transforming continuous temporal observations into discrete transaction events while preserving the essential temporal and causal structure of the underlying system. Such transaction files are formed to match the intended structure of an ‘event log’, serving as input for recently developed process and causal discovery techniques, as presented in Section 3.2.

4.3 XAI Scoring Framework

To assess the effectiveness of the proposed XAI Scoring Framework, we conducted a series of benchmarking experiments using a diverse set of tabular datasets and explainability techniques.

4.3.1 Datasets

We utilized a diverse set of tabular datasets from the UCI [9] repository, each representing distinct domains and varying levels of feature complexity. All datasets underwent standardized preprocessing procedures, including categorical encoding and normalization, to ensure consistency. Various ML models were trained on each dataset, after which the selected XAI techniques were applied to interpret the model predictions. This approach allowed us to evaluate the performance of the explainability methods in a domain-agnostic manner.

4.3.1.1 Dataset Distribution Across Domains

The utilized datasets span multiple domains, as illustrated in Figure 40. Notably, the health and medicine domain exhibits the largest number of datasets (over 50), followed by computer science, business, and physics and chemistry. This distribution highlights the prominence of medical and clinical use cases in current AI research, a trend corroborated by our survey-based data.

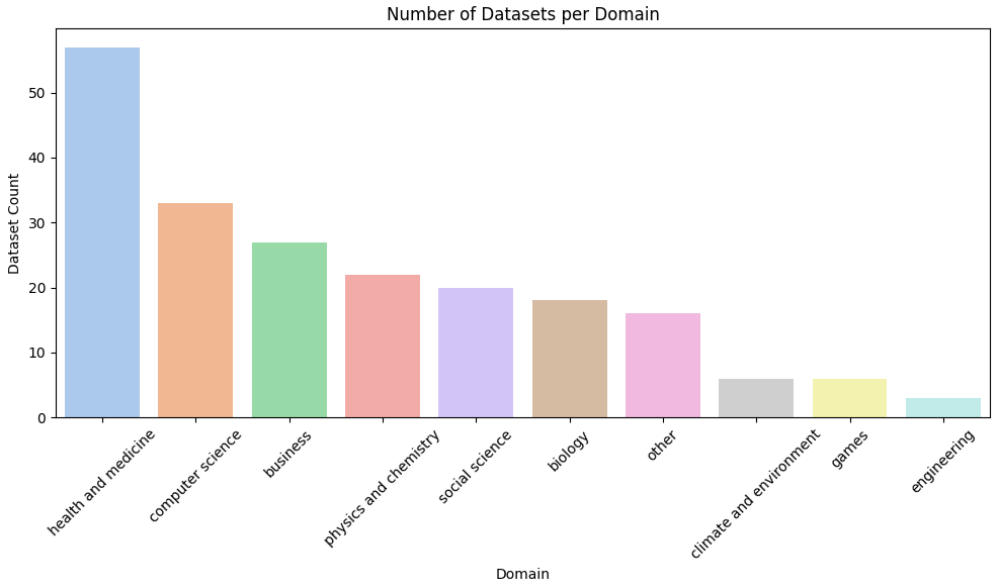


Figure 40: Number of datasets per domain. The health and medicine domain has the highest dataset count, reflecting a significant interest in clinical AI applications.

4.3.2 Quantitative Benchmarking of XAI Methods

Figure 41 presents the average fidelity of four XAI methods—SHAP, LIME, PFI, and PDP—across multiple domains. The results highlight a clear variation in method performance depending on the domain. In health and medicine, SHAP demonstrates consistently high fidelity, suggesting it aligns well with clinical data. Conversely, PDP exhibits notably higher fidelity in business applications, indicating it may be particularly effective for the kinds of features and relationships found in that domain. LIME and PFI maintain relatively moderate yet steady performance across most domains, with occasional spikes in areas such as biology or computer science. Overall, these findings underscore that no single XAI method dominates in every context, reinforcing the importance of domain-specific considerations when selecting an explainability technique.

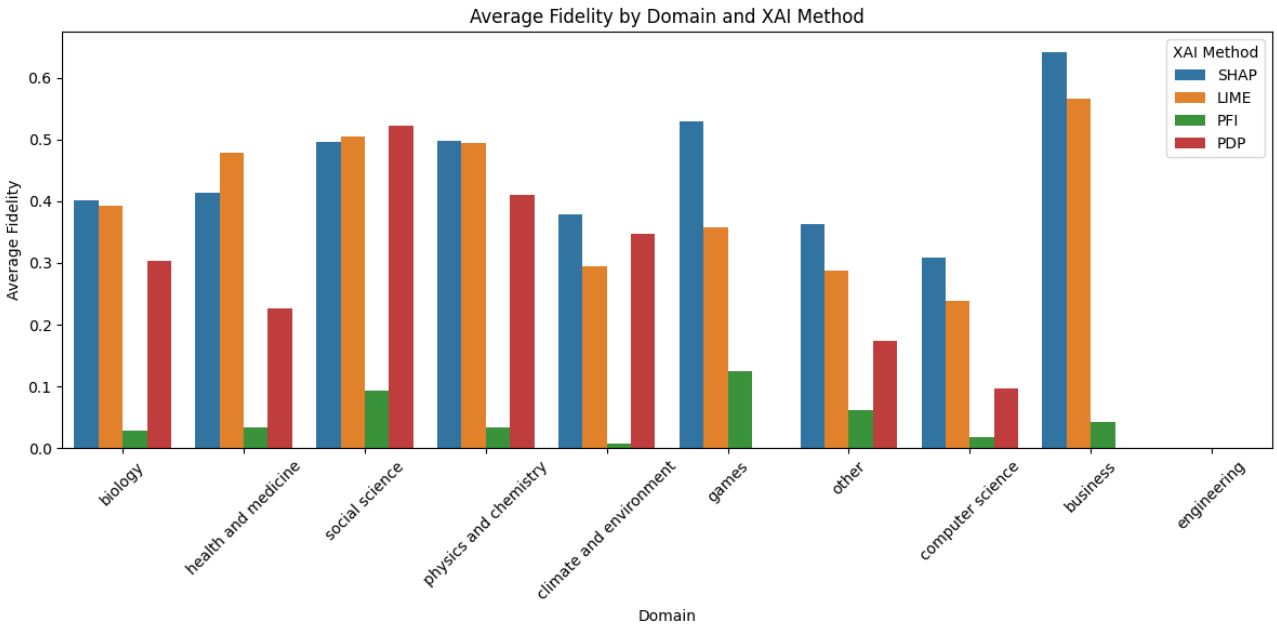


Figure 41: Domain-specific performance variations, with SHAP achieving 0.82 fidelity in healthcare datasets versus PDP's 0.71-0.74 in business domains.

Table 23 provides an additional breakdown for selected datasets. For example, on a Heart Disease dataset within the health and medicine domain, SHAP attains a fidelity of 0.82, while LIME exhibits a lower simplicity value (5.1), indicating more concise explanations. PFI excels in stability (0.93), underscoring its consistency under feature perturbations.

Table 23: Sample of quantitative explainability scores for SHAP, LIME, PFI, and PDP. Fidelity measures alignment with the model, Simplicity indicates fewer features (lower is better), and Stability measures consistency.

Dataset	Method	Fidelity	Simplicity	Stability
Heart Disease	SHAP	0.82	7.3	0.91
	LIME	0.79	5.1	0.88
	PFI	0.76	4.6	0.93
	PDP	0.74	6.0	0.87
Wine Quality	SHAP	0.85	6.8	0.89
	LIME	0.78	5.3	0.85
	PFI	0.74	4.1	0.92
	PDP	0.71	5.7	0.84

4.3.3 Qualitative User Ratings and Interpretability

We measure user-perceived interpretability by aggregating virtual personas ratings (interpretability, understanding, trust). Figure 42 displays these average interpretability scores by domain. PDP appears to be the highest-scoring method in nearly every domain, indicating that users find its partial dependence approach particularly intuitive or easy to understand. While LIME and SHAP also achieve relatively strong scores in several domains (e.g., business, health and medicine), their performance is slightly outpaced by PDP in most cases. PFI, meanwhile, maintains moderate interpretability levels overall.

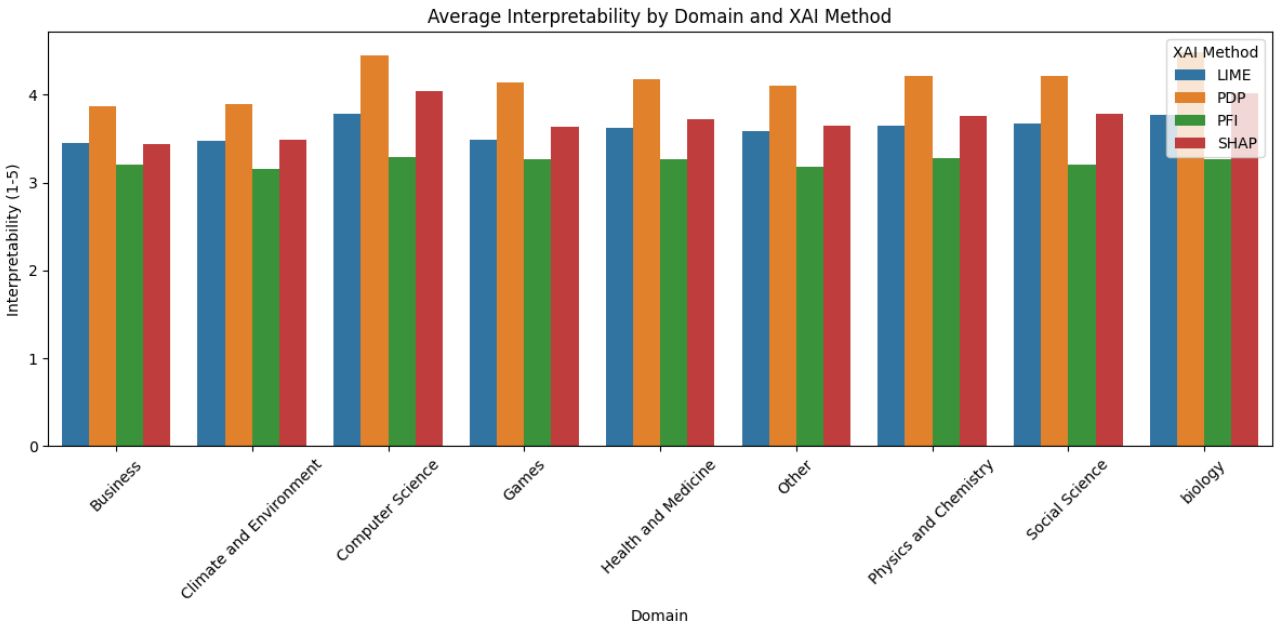


Figure 42: Average interpretability (1-5 scale) by domain and XAI method. Higher bars indicate that end-users (virtual personas) found the explanations more understandable and trustworthy.

All these data sources are integrated into a unified repository keyed by dataset_id.

4.4 StreamStory XAI Dashboard

4.4.1 Pilot 7

This analysis examines the operational behavior of the K-5701 pump at the MOH OIL refinery across two distinct time periods, captured in two models. The first model was built using data from January 2017 (Figure 43), and the second was built using data from March to the end of April 2017 (Figure 44). For both, the data was resampled to 30-minute intervals, and automatic feature selection was used to build the StreamStory models.

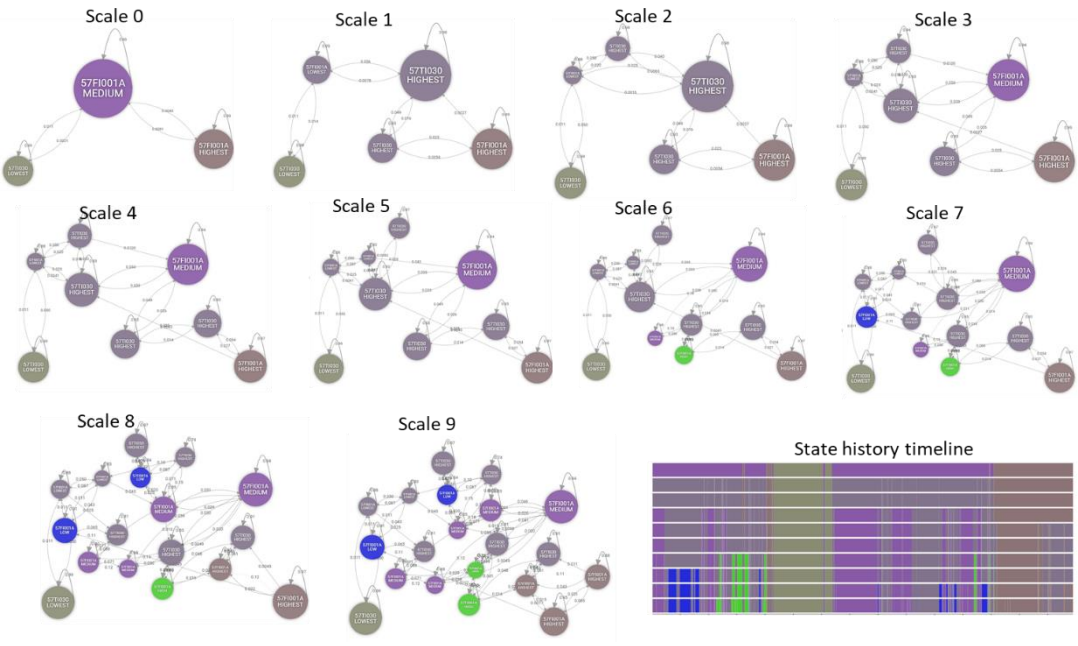


Figure 43: StreamStory model based on MOH data (January 2017).

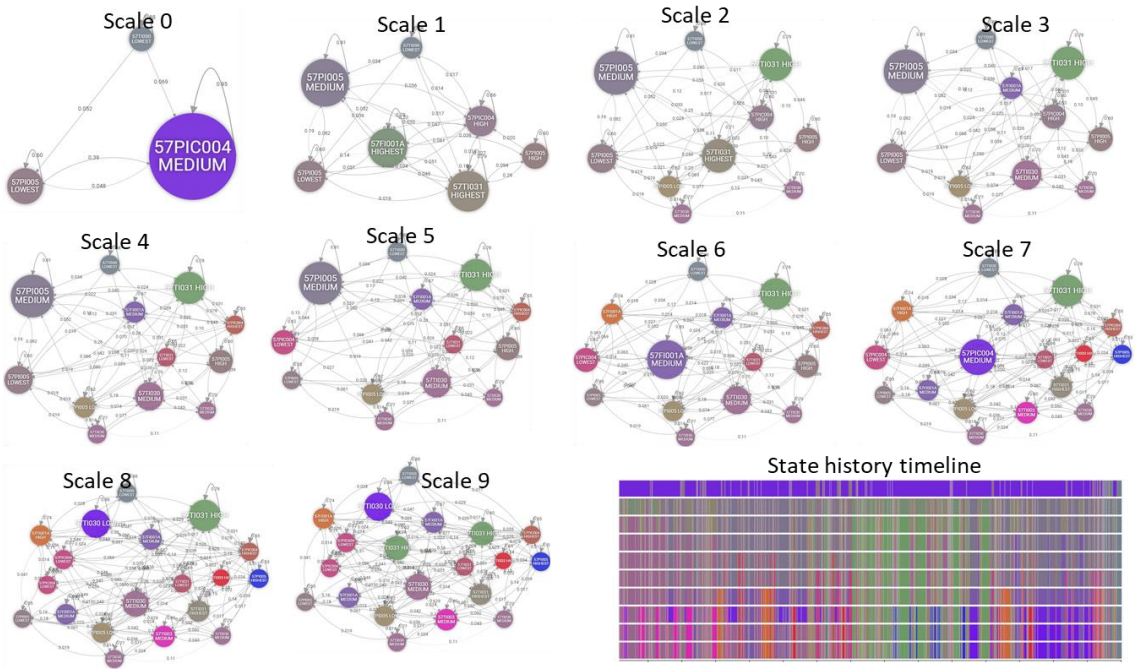


Figure 44: StreamStory model based on MOH data (March-April 2017).

It is important to note that this methodology is not designed to pinpoint a single root cause for any given issue. Instead, its strength lies in revealing the complex dynamics and evolving patterns within the system. By interpreting these patterns across various time scales, we can build a comprehensive understanding of the pump's performance, which can help guide more focused engineering investigations and strategic planning.

The data from the first model suggests a period where the primary operational challenge for the K-5701 pump may have been related to maintaining stable flow and pressure while handling hot product. At the most immediate time scales (Scales 0-2), the analysis reveals frequent, short-term fluctuations. We see a constant interplay between the pump's flow rate and the product temperature, with patterns like "Brief Temperature Peak with Mid-Band Flow" being common. This points to a highly dynamic environment where the control system is constantly making small adjustments. For a plant operator, this view offers a detailed picture of the pump's moment-to-moment efforts to maintain a steady state.

Moving to the mid-range scales (Scales 3-5), these rapid, individual events begin to form more defined operational behaviors. The short-term fluctuations are now seen as part of larger patterns, such as "Controller Hunting at Top-End Flow with Hot Product." What might appear as random noise at lower scales could be interpreted here as a systematic instability. This might suggest that the control loop is tuned in a way that causes the pump to overshoot and undershoot its setpoints continuously. This is useful information for an operator, as it suggests that an investigation into controller tuning could be a valuable next step.

At the highest scales (Scales 6-9), the analysis provides a long-term, strategic overview. The dominant pattern, covering almost 86% of the time at Scale 9, is "Pump K-5701 Holding Mid-Range Throughput." This indicates that, despite the instabilities seen at lower scales, the pump's primary function is being met most of the time. The cross-scale story, however, is the most revealing. The persistent, low-level flow and temperature cycling seen in the lower scales aggregate into the broader "unstable" operational clusters at the mid-scales. This could mean that the equipment is under a form of chronic stress, which might lead to inefficiency and increased wear over time. For planning purposes, the analysis suggests the pump is operating, but its operational mode could potentially impact its long-term health and lead to unplanned downtime in the future.

The second model suggests a shift in operational challenges, moving from flow control to thermal and pressure management under very high demand. The lowest scales (0-2) are dominated by events related to temperature and pressure extremes, with a high frequency of patterns like "Pump Discharge Temperature at Upper Range" and "Discharge Pressure at Maximum (Short Spikes)." This indicates that the pump is frequently operating at or near its thermal and pressure limits. Compared to the flow-centric issues in the first model, this points to a different kind of stress, one that may be driven more by the overall process conditions than the pump's own controls.

At the mid-range scales (3-5), these individual spikes are grouped into sustained periods of high-stress operation. Clusters like "High-Demand, Very Hot Operation" become prominent, showing that the brief events at lower scales are not isolated incidents but are part of longer-lasting operational states. The frequent appearance of "Suction Pressure Collapse with Flow Swing (Cavitation Risk)" at Scale 1 is particularly noteworthy. This provides a critical insight for operators, as suction pressure loss is a known precursor to cavitation, a condition that can cause severe damage to a pump's internals.

The highest scales (6-9) confirm this narrative. The dominant state at Scale 8 is "Pump K-5701 Discharge Pressure Held at Upper PIC Band." This long-term view shows that the system is consistently running against its high-pressure constraints. The cross-scale analysis reveals a clear story of how the constant high-temperature and high-pressure events at the lower scales are the building blocks of the sustained high-stress operation seen at the higher scales. This provides a useful perspective for strategic planning, suggesting that the operational envelope for this pump has shifted and that it is now consistently running in a higher-risk state, which could have implications for equipment health and process stability.

Comparing the two models reveals a significant evolution in the pump's operational challenges. The core function of pumping hot product remains the same in both periods. However, the nature of the instability and the potential sources of operational risk appear to have changed. The first model is characterized by what could be interpreted as internal control instability. The primary issue seems to be related to the pump's own control system and its efforts to maintain a stable flow rate, as evidenced by the "Controller Hunting" patterns. This might point toward factors localized to the pump and its immediate controls.

In contrast, the second model is defined by what appears to be external process stress. The pump seems to be reacting to more extreme process conditions, namely very high temperatures and pressures. The emergence of patterns related to suction pressure collapse could suggest that issues are originating upstream of the pump. This difference is a key insight, suggesting that the operational context for the K-5701 pump changed between the two time periods. The challenge may have shifted from a localized control issue to a broader, systemic issue of managing a high-energy process fluid.

For operators, this distinction is valuable. The patterns in the first model might prompt them to look at the pump's control valve and tuning parameters, whereas the patterns in the second model might encourage them to look upstream for the source of the pressure and temperature excursions. For strategic planning, this comparison provides a data-driven narrative of an evolving operational environment. It highlights that the nature of operational stress can change over time. This information could be useful in guiding where to focus resources for reliability improvements, whether on individual equipment or on the broader process interactions. The analysis provides specific evidence to move from a general sense that "the pump is running hot" to a more nuanced understanding of the underlying operational dynamics, which can lead to more informed and effective decision-making.

5 Conclusions

D5.3 is the second and final deliverable in the series on *Trusted and Explainable AI Techniques*. This document reports the advancements and developments achieved under Tasks 5.1 and 5.2 from M16 to M33 of the project. The performance metrics achieved exceeded expectations, and a large number of assets were indexed in the FAME platform for broad exploitation. The following subsections present the specific conclusions drawn for each functionality separately.

5.1 AI/ML Analytics

Task 5.1 delivered a diverse portfolio of Machine Learning assets across the FAME pilots, covering customer profiling, risk assessment, information extraction, anomaly detection and forecasting. The implemented solutions were tailored to the specific datasets, objectives, and operational contexts of each pilot.

For Pilots 1 and 2, customer segmentations were developed using Recency, Frequency and Monetary (RFM) analysis and KMeans clustering, providing insights into customers' behaviour. The clustering approach demonstrated strong reusability, as the same methodology was successfully applied in two different domains where customer segmentation was required, making it highly suitable for deployment through the marketplace as a generic asset. This demonstrates one of the key features of the marketplace, namely the ability to monetize the same data asset across multiple scenarios.

Additionally, for Pilot 1, a TabNet-based risk assessment model was deployed, focused on identifying high-risk fraudulent customers. For Pilot 4 a question-to-value information extraction pipeline was implemented, leveraging a fine-tuned FLAN-T5 model with LoRA adaptation, and achieving near-perfect accuracy for multiple numerical fields in CDTI funding documents.

Finally, Pilot 7 incorporated two systems for temperature sensor monitoring: an anomaly detector for abnormal pattern recognition, and an LSTM-CNN forecasting model predicting one-week sensor trends from two weeks of historical data. The forecasting solution outperformed the pilot's previously deployed approach, providing more consistent alignment with observed values and the ability to anticipate certain anomalies.

5.2 SAX Techniques

The foundational work on SAX techniques within the FAME platform has opened new frontiers in the research of causal business processes and the use of LLMs for process explainability. The SAX4BPM library is a first-of-its-kind library that offers services to discover causal relations among activities in business processes and to derive explanations from multiple perspectives by leveraging the power of LLMs. Complementing the core algorithms, further work was pursued to establish its pragmatic usefulness in developing the needed instrumentation to assess the quality of its output, either as perceived by its users or more objectively via a designated benchmark. Research on SAX techniques within the FAME platform has had a significant scientific impact, as demonstrated by 10 publications in leading conferences, workshops, and journals [2–7, 19–22].

In terms of developed assets, SAX techniques have produced four assets released as open source and indexed in the FAME platform for broad exploitation: the SAX4BPM library, the survey for explanation quality, the benchmark dataset for causal business process reasoning, and the explanation of clustered instances.

5.3 XAI Scoring Framework

The XAI Scoring Framework has progressed from an early script-based demonstrator to a mature micro-service that quantifies explainability in a way that is both rigorous and user-centred. By fusing quantitative metrics with qualitative signals collected from a group of virtual personas, the service now produces a multidimensional XAI score for a new dataset. The entire pipeline is wrapped in a container image exposing two interoperable entry points: an API layer for automated workflows and a browser-based dashboard for analysts. Four core explanation techniques (SHAP, LIME, Permutation Feature Importance and Partial Dependence Plots) are fully supported, and the plug-in design allows additional methods to be added, ensuring that KPI 5.1 is met and can grow beyond the project's lifetime. Looking ahead, the consortium will exploit this outcome by maintaining the framework as an open-source module that partners can embed in commercial analytics stacks or offer as a stand-alone Software-as-a-Service checker for regulatory audits. Its lightweight deployment model, clear API contract and provenance-aware scoring logic give a concrete way to demonstrate transparent AI.

5.4 StreamStory XAI Dashboard

Part of the work on SAX techniques for time-series data resulted in the development of an explainability suite centered around the StreamStory framework. The primary component, the StreamStory XAI Dashboard, was applied in Pilot 7 to translate complex industrial sensor data into actionable insights using pattern mining, clustering, and LLM-based interpretation. The suite also resulted in release of the StreamStoryPyClient to simplify model interaction and the StreamStory Causal pipeline, which transforms temporal data to enable formal cause-and-effect analysis.

References

1. D5.1 – Trusted and Explainable AI Techniques I, accessible here: https://zenodo.org/records/16620533
2. Fournier, F., Limonad, L., Skarbovsky, I., David, Y.: The WHY in Business Processes: Discovery of Causal Execution Dependencies. Künstliche Intelligenz (1 2025)
3. Dirk Fahland, Fabiana Fournier, Lior Limonad, Inna Skarbovsky, Ava J.E. Swevels*; How well can a large language model explain business processes as perceived by users?*, Data & Knowledge Engineering, Volume 157, 2025, 102416, ISSN 0169-023X
4. Fabiana Fournier, Lior Limonad, Inna Skarbovsky; *Towards a Benchmark for Causal Business Process Reasoning with LLMs*, NLP4BPM24 workshop at BPM2024.
5. Lior Limonad, Fabiana Fournier, Hadar Mulian, George Manias, Spiros Borotis and Danai Kyrkou, Selecting the Right LLM for eGov Explanations, ICEDEG 2025 - Eleventh International Conference on eDemocracy & eGovernment, Bern, Switzerland, 18 - 20 June 2025.
6. Lior Limonad, Fabiana Fournier, The WHY in Business Processes: A new paradigm, 24th National Conference on Industrial Engineering and Management, Open University, Ra'anana, Israel, April 28, 2025
7. Y. David, F. Fournier, L. Limonad, I. Skarbovsky, The WHY in Business Processes: Unification of Causal Process Models, in: BPM Forum in BPM Conference (to appear), 2025.
8. S. Moon, M. Abdulhai, M. Kang, J. Suh, W. Soedarmadji, E. K. Behar, and D. M. Chan, "Virtual personas for language models via an anthology of backstories," arXiv preprint arXiv:2407.06576, 2024.
9. A. Asuncion, D. Newman et al., "Uci machine learning repository," 2007.
10. Moshkovitz, M., Dasgupta, S., Rashtchian, C., & Frost, N. (2020, November). Explainable k-means and k-medians clustering. In International conference on machine learning (pp. 7055-7065). PMLR.
11. Jyoti Arora, Divya Varshney, Analysis of K-Means and K-Medoids Algorithm For Big Data, Procedia Computer Science, Volume 78, 2016, Pages 507-512.
12. Leo Breiman, Random Forests, Machine Learning, Vol. 45, No. 1, 2001, pp. 5-32, Springer.
13. Jerome H. Friedman, Greedy Function Approximation: A Gradient Boosting Machine, The Annals of Statistics, Vol. 29, No. 5, October 2001, pp. 1189-1232, DOI: 10.1214/aos/1013203451
14. David Opitz, Richard Maclin, Popular Ensemble Methods: An Empirical Study, Journal of Artificial Intelligence Research, Volume 11, 1999, Pages 169-198, DOI: 10.1613/jair.614
15. Sercan O. Arik, Tomas Pfister, TabNet: Attentive Interpretable Tabular Learning, Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 6679–6687, arXiv:1908.07442
16. Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, "LoRA: Low-Rank Adaptation of Large Language Models," arXiv preprint arXiv:2106.09685, 2021.
17. Gopikrishna Pavuluri, Gayathri Annem, A Deep Learning Approach to Video Anomaly Detection using Convolutional Autoencoders, arXiv preprint arXiv:2311.04351, 2023.
18. Elmaz, F., Eyckerman, R., Casteels, W., Latré, S., & Hellinckx, P. (2021). CNN-LSTM architecture for predictive indoor temperature modeling. Building and Environment, 204, 108327. https://doi.org/10.1016/j.buildenv.2021.108327

19. Lior Limonad, Fabiana Fournier, Juan Manuel Vera Díaz, Inna Skarbovsky, Shlomit Gur, and Raquel Lazcano*; Monetizing Currency Pair Sentiments through LLM Explainability*, AIFin@ECAI24.
20. Guy Amit , Shlomit Gur*; eXplainable Random Forest*, HAII5.0@ECAI24
21. Dirk Fahland, Fabiana Fournier, Lior Limonad, Inna Skarbovsky, and Ava J.E. Swevels; *Why are my Pizzas late?*, PMAI@IJCAI23
22. Fabiana Fournier, Lior Limonad, and Yuval David, Agentic AI Process Observability: Discovering Behavioral Variability, in PMAI25 workshop (to appear)

Annexes

A. Additional clustering validation figures for Pilot 1

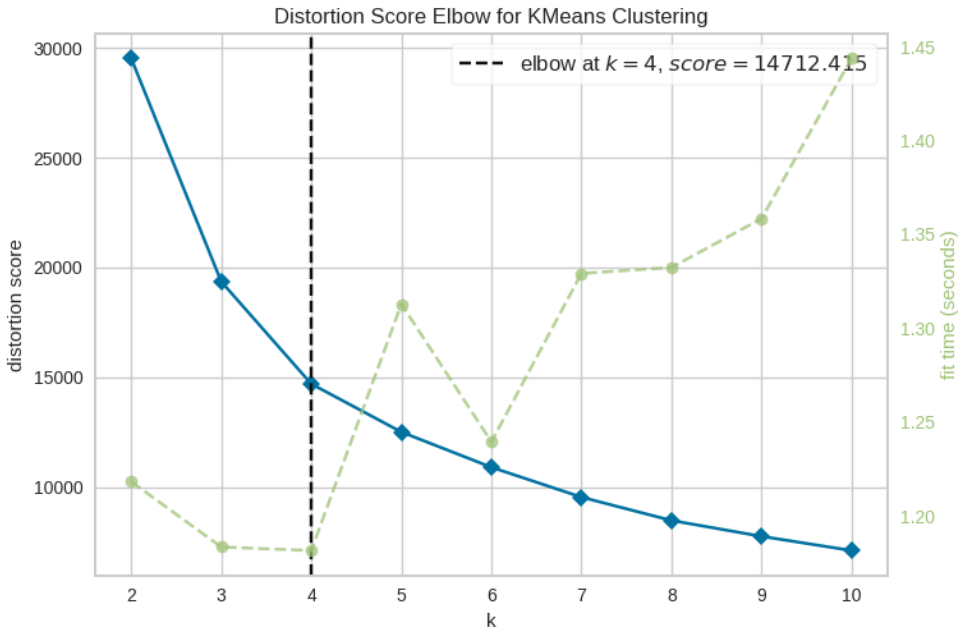


Figure 45: Results of the elbow method for KMeans clustering, indicating a change in slope at K = 4, supporting the cluster choice suggested by the silhouette method.

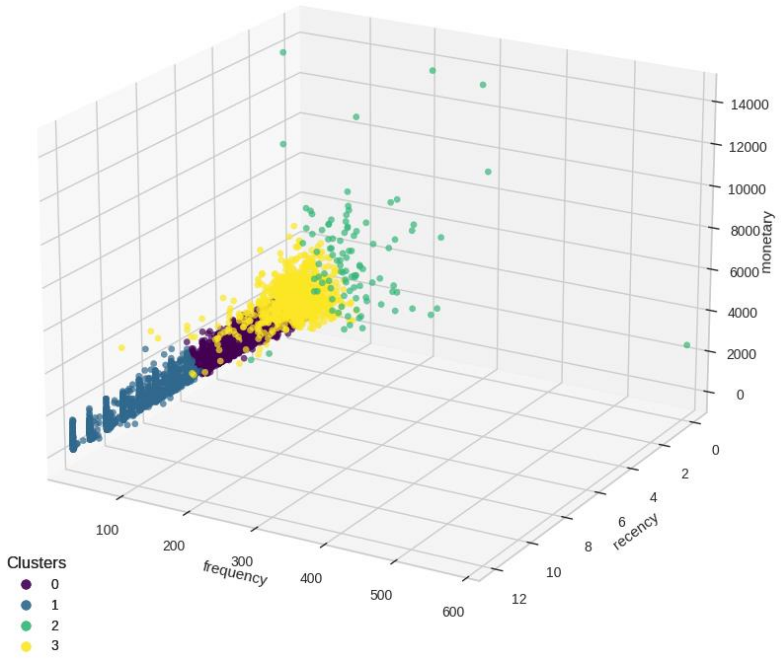


Figure 46: Visualization of the Pilot 1 four clusters in the RFM feature space for the testing dataset. The smaller sample size results in a less dense representation than in the training plot.

B. Additional clustering validation figures for Pilot 2

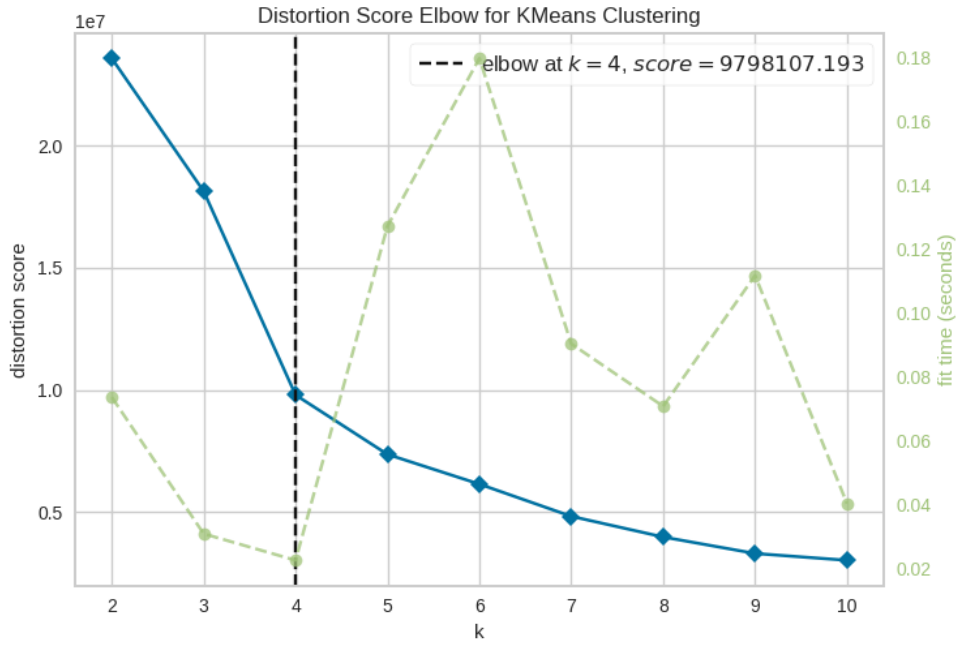


Figure 47: Results of the elbow method for KMeans clustering, indicating a change in slope at K = 4, supporting the cluster choice suggested by the silhouette method.

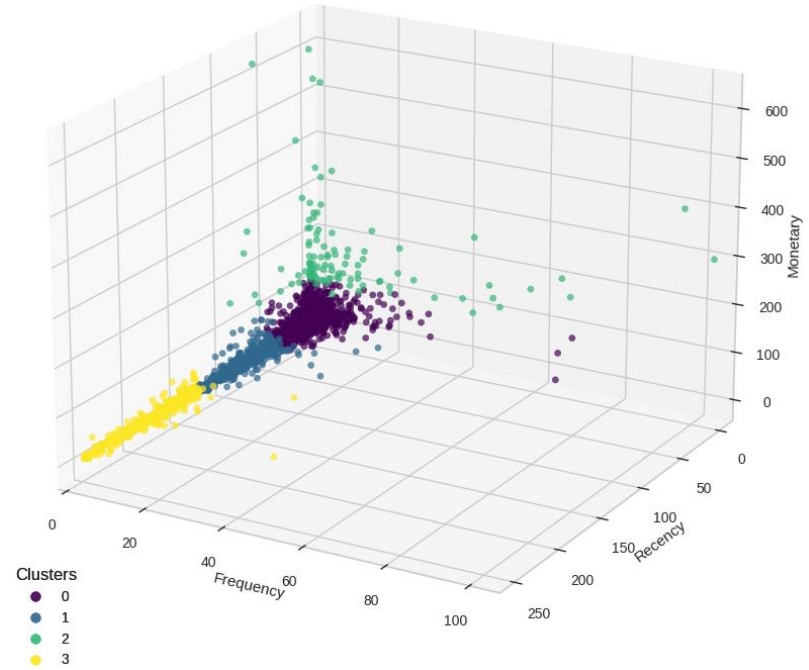


Figure 48: Visualization of the Pilot 2 four clusters in the RFM feature space for the testing dataset. The smaller sample size results in a less dense representation than in the training plot.

C. Risk assessment model evaluation summary for Pilot 1 UC2

Table 24 summarises the performance of all models tested during the development of the UC2 risk assessment model. For each model, standard evaluation metrics such as ROC-AUC and F1-score are provided, along with confusion matrix components (True Positives, False Positives, False Negatives, True Negatives). Derived precision and recall values are also included to facilitate comparison between models, particularly in the context of imbalanced classification.

Table 24: Detailed evaluation metrics for all models tested in UC2. Includes confusion matrix components and derived scores (ROC-AUC, F1-score, Precision, Recall) used to guide model selection and threshold optimization.

Model	ROC-AUC	F1-score	Precision	Recall	TP	FP	FN	TN
Random Forest	0.50	N/A	N/A	0.000	0	0	53	1956
Gradient Boosting	0.51	N/A	N/A	0.000	0	0	53	1956
Voting (Hard)	0.56	0.066	0.038	0.245	13	328	40	1628
Voting (Soft)	0.58	0.084	0.053	0.208	11	197	42	1759
Balanced RF	0.59	0.055	0.029	0.509	27	904	26	1052
TabNet (High Precission)	0.61	0.080	0.069	0.094	5	67	48	1889
TabNet (High Recall)	**0.62**	**0.072**	**0.038**	**0.623**	**33**	**835**	**20**	**1121**

D. Information extraction API response example for Pilot 4

This annex presents a sample JSON response returned by the *POST /extract_information* endpoint. The example contains two samples with the variables extracted from the input CDTI PDF. Each key corresponds to either a condition specification (e.g., *ID. TIPOLOGIA*, *COMUNIDAD*) or a target variable (e.g., *Duración_min (meses)*), with the extracted values assigned.


```
{
    "message": "File processed successfully",
    "document_type": "LICa",
    "sections": [
        "Duración de los proyectos",
        "Anticipos de la ayuda"
    ],
    "output": [
        {
            "ID. TIPOLOGIA": "LICa",
            "TIPOLOGIA": "LÍNEA DIRECTA DE EXPANSIÓN",
            "TIPO AYUDA": "Ayudas Parcialmente Reembolsables de Innovación",
            "PYME": "S",
            "COMUNIDAD": "ANDALUCIA",
            "FONDOS PROPIOS": "PRESUPUESTO-APORTACIÓN CDTI",
            "Duración_min (meses)": 9,
            "Duración_max (meses)": 24,
            "Desembolso_min (hito, mes)": 9,
            "Potencial anticipo max sin aval (%)": 0.5,
            "Potencial anticipo max con aval (%)": 0.75
        },
        {
            "ID. TIPOLOGIA": "LICa",
            "TIPOLOGIA": "LÍNEA DIRECTA DE EXPANSIÓN",
            "TIPO AYUDA": "Ayudas Parcialmente Reembolsables de Innovación",
            "PYME": "S",
            "COMUNIDAD": "CANARIAS",
            "FONDOS PROPIOS": "PRESUPUESTO-APORTACIÓN CDTI",
            "Duración_min (meses)": 9,
            "Duración_max (meses)": 24,
            "Desembolso_min (hito, mes)": 9,
            "Potencial anticipo max sin aval (%)": 0.5,
            "Potencial anticipo max con aval (%)": 0.75
        }
    ],
```


Figure 49: Example of JSON output format generated by the information extraction API for Pilot 4.

E. Forecasting API response example for Pilot 7

The example below illustrates a JSON response generated by the /predict endpoint when the response_format parameter is set to "json". Each record corresponds to one forecasted time step and contains the predicted date and time and the forecasted value of each sensor for that timestamp.

```
[
  {
    "Timestamp": "2021-01-25 15:05:00",
    "22TI118": 43.073341369628906,
    "22TI42": 61.650299072265625
  },
  {
    "Timestamp": "2021-01-25 15:10:00",
    "22TI118": 43.339942932128906,
    "22TI42": 62.62474822998047
  },
  {
    "Timestamp": "2021-01-25 15:15:00",
    "22TI118": 43.15877914428711,
    "22TI42": 61.63727951049805
  },
  {
    "Timestamp": "2021-01-25 15:20:00",
    "22TI118": 43.26405334472656,
    "22TI42": 62.42198181152344
  },
  {
    "Timestamp": "2021-01-25 15:25:00",
    "22TI118": 43.153438568115234,
    "22TI42": 62.610538482666016
  },
  {
    "Timestamp": "2021-01-25 15:30:00",
    "22TI118": 42.995182037353516,
    "22TI42": 62.11643600463867
  },
  {
    "Timestamp": "2021-01-25 15:35:00",
    "22TI118": 43.50305938720703,
    "22TI42": 61.57859420776367
  },
```


Figure 50: Example JSON output from the Forecasting API for machine K-2201, including predictions for sensors 22TI118 and 22TI42 over the first few forecasted time steps.

F. Forecasting models results for Pilot 7

This annex presents the complete evaluation results for the remaining forecasting models developed in Pilot 7. Each figure includes a comparison of true versus predicted values and the corresponding prediction error distribution for the associated sensors.

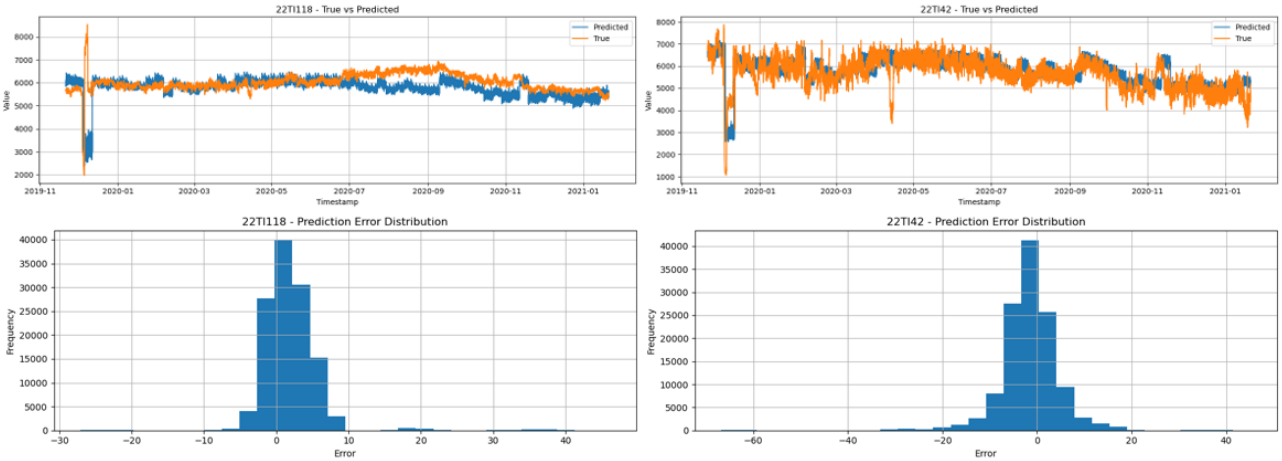


Figure 51: True versus predicted time series and prediction error distributions for the machine K-2201 and sensors 22TI118 (left) and 22TI42 (right).

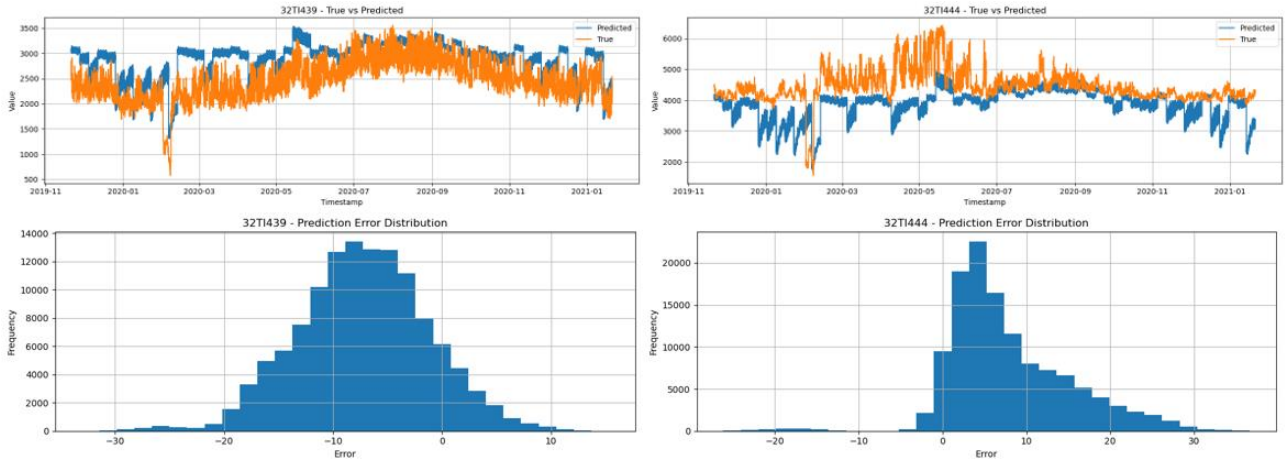


Figure 52: True versus predicted time series and prediction error distributions for the machine K-3201 and sensors 32TI439 (left) and 32TI444 (right).

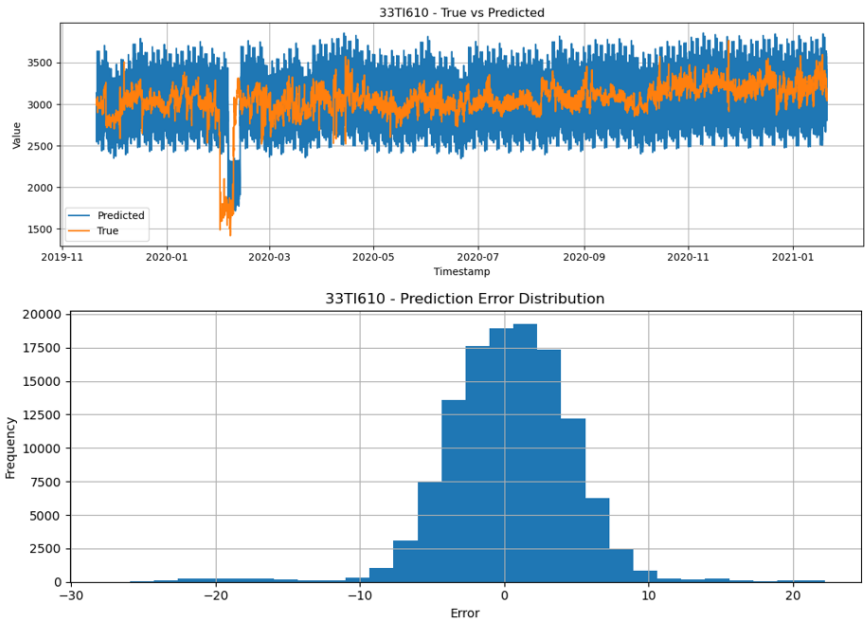


Figure 53: True versus predicted time series and prediction error distributions for the machine K-3301 and the sensor 33TI610.