Feature Engineering for Agents: An Adaptive Cognitive Architecture for Interpretable ML Monitoring (SUPPLEMENTARY FILE)

Gusseppe Bravo-Rocca[1][0000−0001−6824−1124], Peini Liu[1][0000−0003−0058−8732], Jordi Guitart[1,2][0000−0003−0751−3100], Rodrigo M Carrillo-Larco[4][0000−0002−2090−1856], Ajay Dholakia[3][0009−0007−8973−6063], and David Ellison[3][0000−0002−0752−5569]

[1] Barcelona Supercomputing Center, Barcelona, Spain
{gusseppe.bravo, peini.liu, jordi.guitart}@bsc.es
[2] Universitat Politcnica de Catalunya, Barcelona, Spain
[3] Lenovo Infrastructure Solutions Group, Morrisville, NC, USA
{adholakia, dellison}@lenovo.com
[4] Emory University, GA, USA
rmcarri@emory.edu

1 Dataset Generation

This section details our process for creating synthetic datasets that simulate realistic distribution drift scenarios. We employed a Large Language Model (LLM) to generate a Python script based on a carefully crafted system prompt. This approach allowed us to create datasets that are both realistic and controllable, enabling rigorous testing of our ML monitoring system.

1.1 System Prompt

We used the following prompt to instruct the LLM in generating our dataset creation script:

```
Write a Python script to create a synthetic dataset that simulates
distribution drift for a particular real-world scenario. Topics like
finance, healthcare, weather patterns, social media trends, or any
topic most likely to suffer distribution drift.
Use sklearn's make_classification to generate the initial data and
follow these specifications. The dataset should follow these
requirements:

It should have 5 features.
1 label.
6 columns in total.
Some features should contain drift, but not the label.
Make the dataset as realistic as possible.
```

```
Create detailed descriptions as shown in the Python script below.
Include functions to:
\begin{itemize}
\item \texttt{generate_data}: Create the classification dataset
with the specified features and label.
\item \texttt{scale_features}: Scale features to realistic ranges
based on domain-specific requirements.
\item \texttt{create_dataframe}: Convert the raw data into a
pandas DataFrame.
\item \texttt{introduce_drift}: Simulate drift in the dataset by
altering feature distributions over time.
\end{itemize}
Also, include unit tests to validate the dataset generation, feature
scaling, and introduced drift.
```

This prompt was designed to ensure the generation of a dataset that not only simulates distribution drift but also maintains realistic properties specific to the chosen domain.

1.2 Generated Python Script

Based on the system prompt, the LLM produced the following Python script:

```python
import numpy as np
import pandas as pd
from sklearn.datasets import make_classification
from typing import Tuple
import unittest
SEED = 42
NUM_SAMPLES = 1000
FEATURES = ['Age', 'Income', 'Education Level', 'Employment
    Status', 'Marital Status']
LABEL = 'ProgramEligibility'
def generate_data(seed: int, num_samples: int) -> Tuple[np.
    ndarray, np.ndarray]:
X, y = make_classification(n_samples=num_samples, n_features=
    len(FEATURES), n_classes=2, random_state=seed)
return X, y
def scale_features(X: np.ndarray) -> np.ndarray:
X[:, 0] = np.round(np.interp(X[:, 0], (X[:, 0].min(), X[:,
    0].max()), (18, 65)))  # Age
X[:, 1] = np.round(np.interp(X[:, 1], (X[:, 1].min(), X[:,
    1].max()), (20000, 70000)))  # Income
return X
def create_dataframe(X: np.ndarray, y: np.ndarray) -> pd.
    DataFrame:
df = pd.DataFrame(X, columns=FEATURES)
df[LABEL] = y
return df
```

```
def introduce_drift(df: pd.DataFrame) -> pd.DataFrame:
np.random.seed(SEED)
df['Income'] *= np.random.normal(loc=1.5, scale=0.5, size=len
    (df))
return df
class TestDataGeneration(unittest.TestCase):
def test_generate_data(self):
X, y = generate_data(SEED, NUM_SAMPLES)
assert X.shape == (NUM_SAMPLES, 5)
assert y.shape == (NUM_SAMPLES,)
Additional test methods would be included here
```

Code Listing 1.1: Python Code for Dataset Generation

This script implements the core functionalities requested in the prompt, including data generation, feature scaling, dataframe creation, and drift introduction. It also includes a basic unit test to validate the data generation process.

1.3 Dataset Characteristics

The generated dataset simulates a scenario related to program eligibility, with features including Age, Income, Education Level, Employment Status, and Marital Status. The script introduces drift specifically in the Income feature, simulating a realistic scenario where economic conditions might change over time. Figure 1 visualizes the distribution change in the 'Income' feature before and

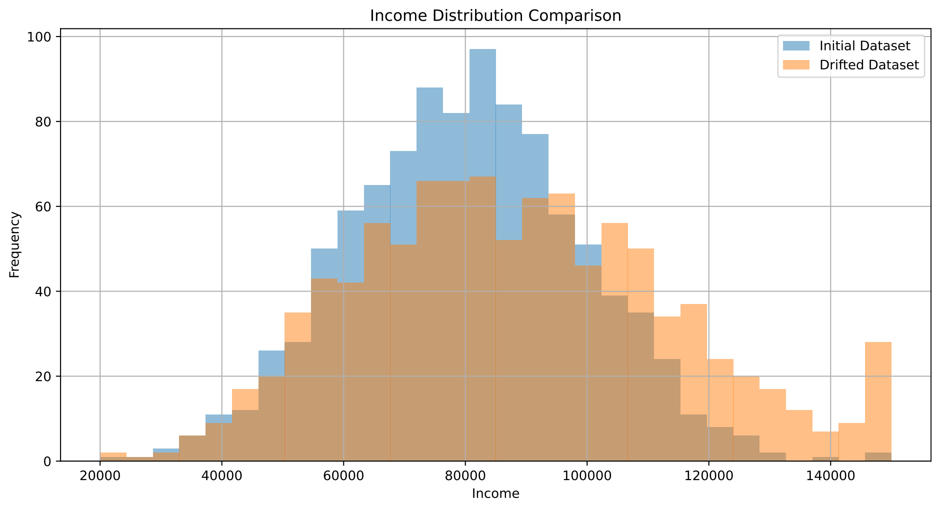


Fig. 1: Comparison of the 'Income' feature distribution between the initial and drifted datasets.

after introducing drift. This histogram clearly demonstrates the shift in income distribution, simulating a realistic scenario of economic change that our ML monitoring system needs to detect and adapt to.

1.4 Refinement Process

After generating the initial script, we conducted a thorough review and refinement process to ensure the dataset's realism and suitability for our experiments. This process included:

– Adjusting feature ranges to align more closely with real-world data.
– Enhancing the drift introduction mechanism to create more nuanced and realistic changes.
– Expanding the unit tests to cover all aspects of the data generation and drift introduction process.
– Verifying the statistical properties of the generated data to ensure it accurately represents the intended scenario.

1.5 JSON Structure for Financial Dataset

As part of our comprehensive approach to dataset generation, we created three distinct datasets: Financial (Loan Default Prediction), Eligibility, and Healthcare. The process described here for the Financial dataset was consistently applied across all three domains, leveraging our LLM to generate both the dataset creation scripts and the corresponding JSON metadata structures. The following JSON structure, generated by the LLM, represents the metadata for the synthetic Financial dataset used to predict loan defaults. It serves as an example of the detailed metadata created for each of our three datasets:

```
{
"NUM_SAMPLES": 1000,
"FEATURES": ["Age", "Income", "Credit Score", "Loan Amount",
"Loan Term", "Interest Rate", "Employment Length",
"Home Ownership", "Marital Status", "Dependents"],
"NUMERICAL_FEATURES": ["Age", "Income", "Credit Score",
"Loan Amount", "Loan Term",
"Interest Rate", "Employment Length"],
"CATEGORICAL_FEATURES": ["Home Ownership", "Marital Status",
    "Dependents"],
"LABEL": "Loan Default",
"COLUMN_TYPES": {
"Age": "int",
"Income": "float",
"Credit Score": "int",
"Loan Amount": "float",
"Loan Term": "int",
"Interest Rate": "float",
"Employment Length": "int",
"Home Ownership": "int",
"Marital Status": "int",
"Dependents": "int",
"Loan Default": "int"
},
"COLUMN_VALUES": {
"Age": "Ranging from 18 to 70 years.",
"Income": "Ranging from $20,000 to $150,000.",
"Credit Score": "Ranging from 300 to 850.",
"Loan Amount": "Ranging from $1,000 to $50,000.",
"Loan Term": "Ranging from 12 to 60 months.",
"Interest Rate": "Ranging from 3.5% to 25%.",
"Employment Length": "Ranging from 0 to 40 years.",
"Home Ownership": {"0": "Rent", "1": "Own", "2": "Mortgage"},
"Marital Status": {"0": "Single", "1": "Married", "2": "Divorced", "3": "Widowed"},
"Dependents": "Ranging from 0 to 5.",
"Loan Default": {"0": "No default", "1": "Default"}
},
"DATASET_TITLE": "Loan Default Prediction Data",
"DATASET_DESCRIPTION": "This dataset simulates the likelihood of borrowers defaulting on a loan based on attributes such as Age, Income, Credit Score, Loan Amount, Loan Term, Interest Rate, Employment Length, Home Ownership, Marital Status, and Dependents.",
"FEATURE_DESCRIPTIONS": {
"Age": "Age of the borrower in years, ranging from 18 to 70.",
"Income": "Annual income of the borrower in dollars, ranging from $20,000 to $150,000.",
"Credit Score": "Credit score of the borrower, ranging from 300 to 850.",
"Loan Amount": "Loan amount requested by the borrower in dollars, ranging from $1,000 to $50,000.",
"Loan Term": "Loan term in months, ranging from 12 to 60.",
"Interest Rate": "Interest rate of the loan in percentage, ranging from 3.5% to 25%.",
"Employment Length": "Number of years the borrower has been employed, ranging from 0 to 40.",
"Home Ownership": "Home ownership status, represented as 0 (Rent), 1 (Own), or 2 (Mortgage).",
"Marital Status": "Marital status, represented as 0 (Single), 1 (Married), 2 (Divorced), or 3 (Widowed).",
"Dependents": "Number of dependents, ranging from 0 to 5."
},
"LABEL_DESCRIPTION": "Indicates the likelihood of loan default, with 0 representing no default and 1 representing default."
}
```

Code Listing 1.2: JSON Metadata for Loan Default Prediction Dataset

This LLM-generated JSON structure, replicated for each of our three datasets with domain-specific adjustments, provides several key benefits for our dataset generation and subsequent analysis:

Comprehensive Feature Description: Each feature is thoroughly described, including its data type, range of values, and semantic meaning. This level of detail ensures clarity in interpreting the data and its implications.

Clear Distinction of Feature Types: The structure explicitly separates numerical and categorical features, facilitating appropriate preprocessing and analysis techniques for each type.

Detailed Value Ranges: For both numerical and categorical features, specific value ranges or categories are provided. This information is crucial for maintaining data integrity and realistic simulations.

Dataset Context: The inclusion of a dataset title and description provides immediate context for the data, making it easier for researchers and practitioners to understand the purpose and scope of the dataset.

Label Information: The structure clearly defines the target variable (Loan Default) and its interpretation, which is essential for supervised learning tasks.

1.6 Sample Dataset

To provide a concrete example of our generated data, we present a sample from the Loan Default Prediction dataset. This sample illustrates the structure and characteristics of the data used in our experiments, demonstrating how the LLM-generated metadata translates into actual data points. Table 1 shows a transposed view of five samples from the dataset:

Feature	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
Age	46	41	42	63	47
Income	76861.45	78687.41	75928.66	62082.39	51463.18
Credit Score	626	673	722	568	773
Loan Amount	31587.58	23096.78	17848.01	31876.20	23139.41
Loan Term	32	37	36	24	27
Interest Rate	8.94	16.42	17.48	10.66	8.88
Employment Length	17	16	14	22	27
Home Ownership	1	1	1	1	0
Marital Status	1	1	1	1	2
Dependents	2	3	2	2	2
Loan Default	1	0	0	0	1

Table 1: Transposed Sample of the Loan Default Prediction Dataset

This sample data demonstrates several key aspects of our synthetic dataset:

Range Adherence: The values for each feature adhere to the ranges specified in the JSON metadata. For instance, Age values fall within the 18-70 range, and Credit Scores are between 300 and 850.

Categorical Encoding: Categorical variables like Home Ownership and Marital Status are encoded as integers, as specified in the JSON structure. For example, Home Ownership values of 0 and 1 correspond to "Rent" and "Own" respectively.

Realistic Distributions: The data exhibits realistic distributions and relationships. For instance, there's a general trend of higher incomes corresponding to higher credit scores, although this relationship isn't perfect, mimicking real-world variability.

Label Distribution: The Loan Default label (the target variable) shows a mix of 0 (no default) and 1 (default), reflecting the binary classification nature of the problem.

Feature Interactions: The sample data suggests potential interactions between features. For example, Sample 4 has the highest age but a relatively low credit score, which might influence the loan default prediction.

2 Monitoring Logs for Drift and SHAP Values

These logs provide crucial insights into the changes in data distribution and feature importance over time.

2.1 Drift Detection logs

Table 2 summarizes the drift detection results using the Kullback-Leibler (KL) divergence:

Feature	**Drift Score**	**Threshold**	**Drift Detected**	**Statistical Test**
Age	0.0388	0.1	No	Kullback-Leibler
Credit Score	0.0778	0.1	No	Kullback-Leibler
Employment Length	0.1042	0.1	Yes	Kullback-Leibler
Income	0.1308	0.1	Yes	Kullback-Leibler
Interest Rate	0.1221	0.1	Yes	Kullback-Leibler
Loan Amount	0.0647	0.1	No	Kullback-Leibler
Loan Term	0.0699	0.1	No	Kullback-Leibler
Home Ownership	0.1856	0.1	Yes	Kullback-Leibler
Marital Status	5.6558	0.1	Yes	Kullback-Leibler
Dependents	0.1291	0.1	Yes	Kullback-Leibler

Table 2: Drift detection report based on the Kullback-Leibler divergence test. Drift is detected when the drift score exceeds the threshold.

2.2 SHAP Value Analysis

Table 3 presents the SHAP values for both the reference and current datasets, including their ranking positions:

Feature	Reference SHAP Value	Reference Position	Current SHAP Value	Current Position
Income	0.1398	1	0.1676	1
Loan Term	0.1079	2	0.0887	2
Age	0.0816	3	0.0535	5
Employment Length	0.0775	4	0.0772	3
Credit Score	0.0573	5	0.0526	6
Marital Status	0.0414	6	0.0735	4
Loan Amount	0.0309	7	0.0303	7
Interest Rate	0.0220	8	0.0180	9
Dependents	0.0210	9	0.0185	8
Home Ownership	0.0036	10	0.0033	10

Table 3: SHAP values for the reference and current datasets. The SHAP value represents the impact of each feature on the prediction.

2.3 Implications for Model Monitoring

The combination of drift detection and SHAP value analysis provides a comprehensive view of how the data distribution and feature importance evolve over time. This information is crucial for:

- Identifying features that require closer monitoring or potential retraining.
- Understanding how changes in feature distributions affect their predictive importance.
- Guiding feature engineering efforts to maintain or improve model performance.
- Informing decisions about model updates or retraining to adapt to significant data drift.

Our CAMA system leverages these insights to provide actionable recommendations for maintaining model performance and reliability in the face of evolving data distributions.

3 Ground Truth Report

This sections provides a comprehensive and detailed analysis of the actual dataset used in the study, that is, our ground truth report. It includes an overview of the dataset's features, label information, and key insights from the data. This report serves as the baseline for comparison with model-generated reports and helps to understand the data's structure, distribution, and significance for predictive modeling. It may include statistical analyses, drift detection, and feature importance metrics, giving a thorough understanding of the dataset before applying machine learning techniques. The following is an example:

```
## Comprehensive Report

### Executive Summary

This report analyzes a dataset aimed at predicting loan defaults based on
```

```
various borrower attributes. Key features include Age, Income, Credit Score,
Loan Amount, Loan Term, Interest Rate, Employment Length, Home Ownership,
Marital Status, and Dependents. The primary objective is to identify patterns
and insights that can help in understanding the likelihood of loan defaults.

### Dataset Synopsis

- **Title**: Loan Default Prediction Data
- **Features Analyzed**: Age, Income, Credit Score, Loan Amount, Loan Term,
  Interest Rate, Employment Length, Home Ownership, Marital Status, Dependents
- **Label Variable**: Loan Default

This dataset simulates the likelihood of borrowers defaulting on a loan based
on attributes such as Age, Income, Credit Score, Loan Amount, Loan Term,
Interest Rate, Employment Length, Home Ownership, Marital Status, and
Dependents. The 'Loan Default' variable serves as the label, indicating whether
a borrower is likely (1) or not likely (0) to default on the loan.

### Label Description

- **Loan Default**: Indicates the likelihood of loan default, with 0
  representing no default and 1 representing default.
(Here the LLM should provide an explanation if the label has issues or not.
Remember that the label was not used in the drift analysis and SHAP values.)

The label 'Loan Default' appears to be well-defined with clear binary outcomes
(0 or 1). However, without using it in drift analysis and SHAP values, it is
crucial to ensure the consistency and accuracy of this label during model
training and testing.

### Feature Analysis

#### Feature Name: Age
- **Description**: Age of the borrower in years, ranging from 18 to 70.
- **Type**: Numerical
- **Possible Values**: Ranging from 18 to 70 years.
- **Data Type**: int

**Distribution Drift Analysis**
- **Statistical Test**: Kullback-Leibler divergence
- **Drift Score**: 0.03883719590118
- **Detection**: No drift detected
- **Current vs. Reference Distribution**:
  - **Current**: {27.0, 31.0, 35.0, 39.0, 43.0, 47.0, 51.0, 55.0, 59.0,
    63.0, 67.0}
  - **Reference**: {18.0, 23.2, 28.4, 33.6, 38.8, 44.0, 49.2, 54.4,
    59.6, 64.8, 70.0}
- **Interpretation**: The current distribution shows higher frequencies in the
  middle age ranges (35-55 years) compared to the reference distribution, which
  indicates a broader spread across all ages.

**Feature Attribution Analysis**
- **Method**: Tree SHAP
- **Training Data**: 0.08155174483476563 (Rank 3)
- **Current Data**: 0.05350981388279517 (Rank 5)
- **Interpretation**: Age was a significant factor during training but has
  reduced in importance in the current data, potentially due to other
  influencing factors.

**Overall Interpretation**:
Age does not show significant distribution drift, indicating stability in the
age demographics of borrowers.

[MORE ANALYSIS GOES HERE]
```

4 Evaluation set

It provides a structured prompt designed to generate an assessment of the ground truth report. It tasks a system with creating a set of 39 multiple-choice questions and corresponding answers that evaluate the key insights and comparisons presented in the report. The questions cover various aspects of the dataset, including numerical comparisons between current and reference distributions, and the answers are formatted in JSON for further use. This section ensures a comprehensive evaluation of the report's content through a variety of detailed questions. It follows the MMLU benchmark as follows:

```
System: You are an expert in data science.
Human:
    Generate 39 detailed multiple-choice questions and concise answers
    based on the following comprehensive report.
    Each question should have five options (A, B, C, D, E) and should
    cover various aspects of the report. Ensure the correct answer
    is clearly indicated and is unique.
    Make sure to include numerical comparisons and insights drawn
    from the current and reference distributions
wherever applicable:

[GROUND TRUTH REPORT]
```

```
Your output should be in json format (‘‘‘json and ‘‘‘ tags) as follows:

[
  {’question’: ’question here’, ’options’: [’A) option1’, ’B) option2’,
  ’C) option3’, ’D) option4’, ’E)
option5’], ’answer’: ’A’},
  {’question’: ’question here’, ’options’: [’A) option1’, ’B) option2’,
  ’C) option3’, ’D) option4’, ’E)
option5’], ’answer’: ’C’},
  ...
  {’question’: ’question here’, ’options’: [’A) option1’, ’B) option2’,
  ’C) option3’, ’D) option4’, ’E)
option5’], ’answer’: ’E’},
]

Only output the json (using ‘‘‘json and ‘‘‘ tags), no explanations.
Think step by step to solve your task.
```

The execution of the previous prompt yields the following JSON file:

```
[
    {
        "question": "Which feature is used to predict loan default and represents the age of the borrower?",
        "options": [
            "A) Income",
            "B) Credit Score",
            "C) Loan Amount",
            "D) Age",
            "E) Employment Length"
        ],
        "answer": "D"
    },

    {
        "question": "Which feature has the highest SHAP value in the current data for predicting loan default?",
        "options": [
            "A) Loan Amount",
            "B) Loan Term",
            "C) Credit Score",
            "D) Age",
            "E) Income"
        ],
        "answer": "E"
    },
    {
        "question": "Which statistical test is used for
           distribution drift analysis in this report?",
        "options": [
            "A) Chi-Square Test",
            "B) T-Test",
            "C) ANOVA",
            "D) Kullback-Leibler Divergence",
            "E) Mann-Whitney U Test"
        ],
        "answer": "D"
    },


    {
        "question": "Which feature represents the interest
           rate of the loan in percentage?",
        "options": [
            "A) Loan Term",
            "B) Loan Amount",
            "C) Interest Rate",
            "D) Age",
            "E) Income"
        ],
        "answer": "C"
    },
    {
        "question": "What is the drift score for the Marital
           Status feature?",
        "options": [
            "A) 0.18557356469873026",
            "B) 5.655843738731566",
            "C) 0.1290888567959812",
            "D) 0.10422809774139326",
            "E) 0.12211093048448328"
        ],
        "answer": "B"
    },
    {
        "question": "Which feature's current distribution
           shows a higher concentration of borrowers with
           shorter employment durations?",
        "options": [
            "A) Age",
            "B) Income",
            "C) Loan Amount",
            "D) Employment Length",
            "E) Marital Status"
        ],
        "answer": "D"
    },
    {[MORE QUESTIONS AND ANSWERS GO HERE]}

]
```

5 Generated Report

This report was generated by our method **CAMA**, using the **Llama3-70b model** and the **financial dataset**.

5.1 Loan Default Prediction Data - Deep Insight Report

```
## Overview

## Summary
The Loan Default Prediction dataset consists of 10 features and a target variable,
Loan Default, which indicates the likelihood of a borrower defaulting on a loan.
The features include Age, Income, Credit Score, Loan Amount, Loan Term, Interest
Rate, Employment Length, Home Ownership, Marital Status, and Dependents. The
dataset has undergone significant changes between the reference and current
datasets, with drift detected in several features, including Income, Loan Term,
Interest Rate, Employment Length, Home Ownership, Marital Status, and Dependents.
The SHAP values indicate that Income, Credit Score, and Loan Amount are the most
important features in predicting loan default.

## Conclusion
The analysis of the Loan Default Prediction dataset reveals significant changes in
the distribution of several features between the reference and current datasets.
The drift detected in these features may impact the performance of the loan default
prediction model, and retraining the model with the updated dataset is recommended.
The SHAP values highlight the importance of Income, Credit Score, and Loan Amount
in predicting loan default, and these features should be closely monitored.
Furthermore, the changes in the distribution of features such as Employment Length,
Home Ownership, Marital Status, and Dependents may require adjustments to the model
to ensure accurate predictions. Overall, the analysis provides valuable insights
into the dataset and informs strategies for improving the loan default prediction
model.

## Details

### Label Insight

The label "Loan Default" in the "Loan Default Prediction Data" dataset is a
categorical variable that indicates the likelihood of a borrower defaulting
on a loan. It takes two possible values:
```

```
- 0: Represents "No default", meaning the borrower is not likely to default
  on the loan.
- 1: Represents "Default", meaning the borrower is likely to default
  on the loan.

In other words, a value of 0 indicates a low risk of loan default, while a
value of 1 indicates a high risk of loan default. The label is of type
integer (int), which is a common representation for categorical variables
in machine learning and data analysis.

Based on the available information, there are no apparent issues with the label.
The description, possible values, and data type are all clear and consistent.

### Feature Analysis: Marital Status

#### Feature Description

- Name: Marital Status
- Description: Marital status, represented as 0 (Single), 1 (Married),
  2 (Divorced), or 3 (Widowed)
- Type: Categorical
- Possible Values: {'0': 'Single', '1': 'Married', '2': 'Divorced',
  '3': 'Widowed'}
- Data Type: int

#### Tool Results

**get_drift_report**

The get_drift_report tool was used to analyze the Marital Status feature for
data drift between the reference and current datasets. The results are as
follows:

- Column Name: Marital Status
- Column Type: Categorical
- Statistical Test: Kullback-Leibler divergence
- Drift Score: 5.655843738731566
- Drift Detected: True
- Current Distribution: {'x': [0, 1, 2, 3], 'y': [2, 1, 0, 197]}
- Reference Distribution: {'x': [0, 1, 2, 3], 'y': [20, 538, 237, 5]}

**get_shap_values**

The get_shap_values tool was used to calculate the mean(|SHAP value|) for
```

```
the Marital Status feature to show its average impact on the model's predictions:

- Reference: {'value': 0.041422401537971096, 'position': 6}
- Current: {'value': 0.07354211915327408, 'position': 4}

The results indicate that the Marital Status feature's impact on the model's
predictions has increased in the current dataset.

#### Overall Feature Analysis

The Marital Status feature has drifted, and its increased importance in the
models predictions may require adjustments to improve performance.
```

6 Decision Procedure

6.1 Refactoring

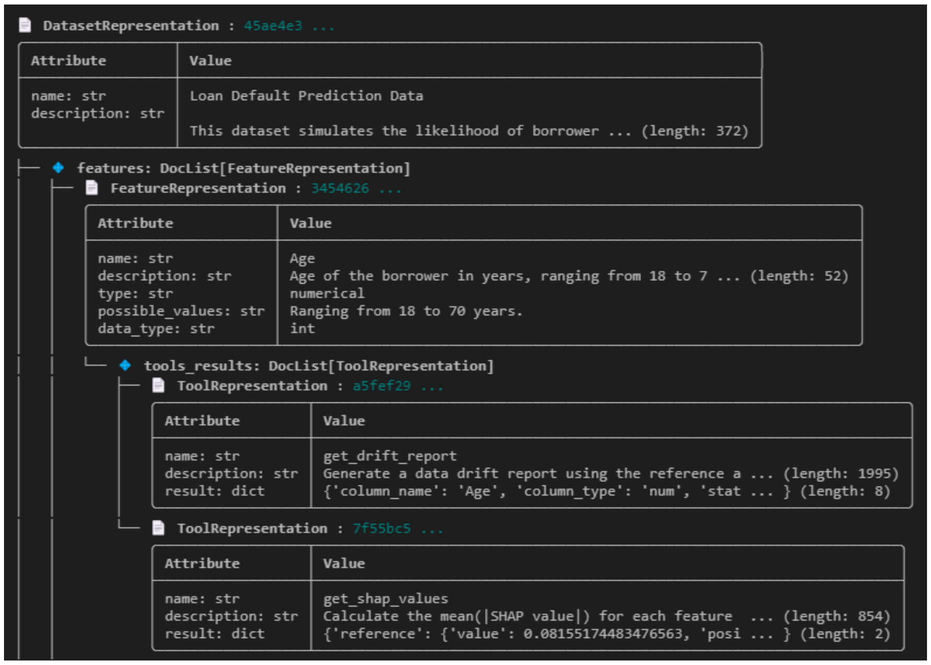
In the refactoring process, different Python classes are used to represent the dataset, features, and the tools applied to those features. See the results in Figure 2. The following describes the refactored structure:

– **DatasetRepresentation**: This class organizes the dataset metadata, including the dataset's name, description, and its features. Each feature is represented using the *FeatureRepresentation* class, which stores details like name, type (numerical or categorical), possible values, and the results from tools applied to the feature (e.g., drift detection, SHAP values).
– **FeatureRepresentation**: For each feature (such as "Age"), this class holds metadata such as a description, data type, and a list of tools results applied to that feature. It captures the analytical tools applied to each feature and their outputs.
– **ToolRepresentation**: Each tool result (like drift detection, SHAP value analysis) is stored as a *ToolRepresentation*, which contains the tool's name, a description of what the tool does, and the actual result of the tool applied to the feature.

The structure allows for a well-organized representation of both the dataset's features and the results from various analytical tools, enabling better insights and interpretability.

The figure illustrates the hierarchical structure after refactoring:

– At the **Dataset Level**, the dataset's name and description (e.g., "Loan Default Prediction Data") are shown, along with its purpose (predicting the likelihood of loan defaults based on borrower features).
– At the **Feature Level**, each feature is represented by attributes like description, type, and possible values (e.g., "Age" ranges from 18 to 70 years).
– Each feature is associated with **Tool Results**, such as ‘get_drift_report‘ and ‘get_shap_values‘, showing the results of analyzing the feature for drift and SHAP importance values.



```
📄 DatasetRepresentation : 45ae4e3 ...
╭──────────────────┬───────────────────────────────────────────────────────────────╮
│ Attribute        │ Value                                                         │
├──────────────────┼───────────────────────────────────────────────────────────────┤
│ name: str        │ Loan Default Prediction Data                                  │
│ description: str │                                                               │
│                  │ This dataset simulates the likelihood of borrower ... (length: 372) │
╰──────────────────┴───────────────────────────────────────────────────────────────╯
├── 🔷 features: DocList[FeatureRepresentation]
│   ├── 📄 FeatureRepresentation : 3454626 ...
│   │   ╭──────────────────────┬──────────────────────────────────────────────────────────╮
│   │   │ Attribute            │ Value                                                    │
│   │   ├──────────────────────┼──────────────────────────────────────────────────────────┤
│   │   │ name: str            │ Age                                                      │
│   │   │ description: str     │ Age of the borrower in years, ranging from 18 to 7 ... (length: 52) │
│   │   │ type: str            │ numerical                                                │
│   │   │ possible_values: str │ Ranging from 18 to 70 years.                             │
│   │   │ data_type: str       │ int                                                      │
│   │   ╰──────────────────────┴──────────────────────────────────────────────────────────╯
│   │   └── 🔷 tools_results: DocList[ToolRepresentation]
│   │       ├── 📄 ToolRepresentation : a5fef29 ...
│   │       │   ╭──────────────────┬──────────────────────────────────────────────────────────────╮
│   │       │   │ Attribute        │ Value                                                        │
│   │       │   ├──────────────────┼──────────────────────────────────────────────────────────────┤
│   │       │   │ name: str        │ get_drift_report                                             │
│   │       │   │ description: str │ Generate a data drift report using the reference a ... (length: 1995) │
│   │       │   │ result: dict     │ {'column_name': 'Age', 'column_type': 'num', 'stat ... } (length: 8) │
│   │       │   ╰──────────────────┴──────────────────────────────────────────────────────────────╯
│   │       └── 📄 ToolRepresentation : 7f55bc5 ...
│   │           ╭──────────────────┬──────────────────────────────────────────────────────────────╮
│   │           │ Attribute        │ Value                                                        │
│   │           ├──────────────────┼──────────────────────────────────────────────────────────────┤
│   │           │ name: str        │ get_shap_values                                              │
│   │           │ description: str │ Calculate the mean(|SHAP value|) for each feature  ... (length: 854) │
│   │           │ result: dict     │ {'reference': {'value': 0.08155174483476563, 'posi ... } (length: 2) │
│   │           ╰──────────────────┴──────────────────────────────────────────────────────────────╯
```

Fig. 2: Data representation after refactoring.

6.2 Implementation Details

The refactoring process is implemented using Python classes. Here are key excerpts from the implementation:

```python
class ColumnRepresentation(BaseDoc):
    name: str = None
    description: str = None
    type: str = None
    possible_values: Any = None
    data_type: str = None

    def as_dict(self):
        return {
            'name': self.name,
            'description': self.description,
            'type': self.type,
            'possible_values': self.possible_values,
            'data_type': self.data_type,
        }

class ToolRepresentation(BaseDoc):
    name: str = None
    description: str = None
    result: Any = None

    def as_dict(self):
        return {
            'name': self.name,
            'description': self.description,
            'result': self.result
        }

class FeatureRepresentation(ColumnRepresentation):
    tools_results: DocList[ToolRepresentation]

    def as_dict(self):
        column_metadata = super().as_dict()
        column_metadata['tools_results'] = [tool.as_dict()
            for tool in self.tools_results]
        return column_metadata

class DatasetRepresentation(BaseDoc):
    name: str = None
    description: str = None
    features: DocList[FeatureRepresentation] = None
    label: ColumnRepresentation = None

    def as_dict(self):
        return {
            'name': self.name,
            'description': self.description,
            'features': [feature.as_dict() for feature in
                self.features],
            'label': self.label.as_dict()
        }
```

Code Listing 1.3: Key classes for dataset representation

These classes form the backbone of our refactored data representation, allowing for a structured and easily interpretable format for dataset metadata and analysis results.

7 Break down

The Break Down step is designed to analyze individual features of the dataset in detail. It uses a step-by-step prompt to guide the analysis process. This step generates comprehensive reports for each feature, including:

– Feature description (name, type, possible values, data type)
– Detailed analysis of results from various tools (e.g., drift detection, SHAP values)
– Insights and interpretations based on the tool results

– Overall feature analysis summarizing key findings

The process uses a template that incorporates dataset information, feature details, and tool results to ensure consistent and thorough analysis across all features.

```
You are an expert data scientist tasked with generating a comprehensive
and detailed report for a dataset analysis. Use the following guidelines to
structure your response:
Context: You are analyzing a feature within a dataset using various tools.
The results from these tools need to be compiled into a cohesive report.
Objective: Generate a detailed report that describes the feature,
presents the results from various analysis tools, and provides
insights based on these results within the context of the overall dataset.
Style: Write in a professional, analytical style typical of data science
reports. Be clear, concise, and well-structured with proper headings and
subheadings. Include technical details and data-driven insights.
Tone: Maintain an objective, authoritative, and informative tone
throughout the report, reflecting your expertise in data science.
Audience: Your report is for data scientists, business analysts, and
stakeholders interested in the dataset analysis results.
Assume the audience has a good understanding of data science
concepts and terminology.
Response Format: Structure your report as follows:
1. Feature Description
   - Name
   - Description
   - Type
   - Possible Values
   - Data Type
2. Tool Results (for each tool)
   - Detailed analysis of results
   - Insights and interpretations
3. Overall Feature Analysis
   - Summary of key findings

Do not add more sections. Use the provided information to fill in
the details for each section. Ensure that your analysis is thorough and
provides valuable analysis for each tool's results.
""")

tool_results = "\n".join([
    f"- {tool['name']}:\n  Description: {tool['description']}\n  Result:
    {tool['result']}"
    for tool in feature_info['tools_results']
])
```

```
tools_names = ", ".join([tool['name'] for tool in feature_info['tools_results']])

human_template = textwrap.dedent("""
Generate a comprehensive data science report using
the following information:

Dataset Information:
Title: {dataset_title}
Description: {dataset_description}
Feature Information:
- Name: {name}
- Description: {description}
- Type: {type}
- Possible Values: {possible_values}
- Data Type: {data_type}
Tool Results:
{tool_results}
Tools used: {tools_names}
""")
```

7.1 Compile

The Compile step aggregates the detailed feature-level analyses into a concise, overarching summary of the dataset. This step focuses on:

– Providing a high-level dataset overview (title, description, number of features)
– Summarizing key findings from individual feature reports
– Presenting a concise label report

The compile step emphasizes brevity and clarity, distilling the wealth of information from the break down step into a digestible format suitable for quick understanding of the dataset's key characteristics and any significant findings from the analysis.

```
You are a data scientist tasked with generating a concise summary
based on the provided dataset information. Be concise, make it short
and to the point.

### Dataset Overview:
- **Title**: {dataset_title}
- **Description**: {dataset_description}
- **Number of Features**: {num_features}
- **Label**: {label_name} - {label_description}
```

```
### Feature Reports:
{escape_curly_braces(feature_reports_section)}

### Label Report:
{escape_curly_braces(label_report)}
```

8 Results

This section presents the performance comparison of various LLMs and methods applied to each of the three datasets: the Financial dataset (representing an easy difficulty level), the Eligibility dataset (representing a medium difficulty level), and the Healthcare dataset (representing a difficult level). The models were evaluated based on multiple metrics, including accuracy, the percentage of unknown predictions, the number of tokens used, and the time taken for each method. The tables below (Table 4, 5 and 6) provide detailed performance metrics for each model and method, offering insights into their effectiveness across different datasets.

The tables below summarize the performance across different methods, including CAMA, Standard, Chain of Thought (CoT), Reflection, React, Self-Discover (S.Discover), and Plan Execute (P.Execute), for each dataset. The results highlight the effectiveness of CAMA in achieving higher accuracy while maintaining a balance between token usage and processing time compared to the other methods.

Table 4: Performance comparison for Financial dataset
Note: The financial dataset represents an easy difficulty level.

Model	Metric	CAMA	Standard	CoT	Reflection	React	S.Discover	P.Execute
llama-3.2-1b	**Accuracy (↑)**	**71.8**	43.6	23.1	35.9	59.0	43.6	38.5
	Unknown (↓)	**25.6**	56.4	76.9	53.8	33.3	53.8	61.5
	Tokens (↓)	28.1	5.2	4.8	13.8	**4.3**	18.2	37.3
	Time (↓)	6.2	**1.2**	1.9	4.2	2.5	5.6	43.4
llama3-8b	**Accuracy (↑)**	**87.2**	30.8	38.5	48.7	7.7	46.2	69.2
	Unknown (↓)	**0.0**	69.2	61.5	48.7	92.3	51.3	23.1
	Tokens (↓)	24.2	4.8	4.8	13.8	**4.3**	24.6	37.3
	Time (↓)	5.1	**1.2**	**1.2**	4.2	2.5	5.7	15.2
llama3-70b	**Accuracy (↑)**	**94.9**	33.3	51.3	48.7	20.5	43.6	43.6
	Unknown (↓)	**0.0**	66.7	48.7	48.7	79.5	53.8	53.8
	Tokens (↓)	21.6	**4.8**	5.2	14.6	8.9	23.1	14.3
	Time (↓)	59.5	**2.1**	48.0	146.3	17.7	171.8	148.6
gpt-4o-mini	**Accuracy (↑)**	**84.6**	69.2	61.5	30.8	48.7	41.0	41.0
	Unknown (↓)	**0.0**	30.8	38.5	66.7	46.2	59.0	59.0
	Tokens (↓)	24.2	**5.7**	**5.7**	16.5	9.6	23.9	510.6
	Time (↓)	**19.7**	20.7	27.7	62.6	32.5	56.8	238.1

Table 5: Performance comparison for Eligibility dataset
Note: The eligibility dataset represents a medium difficulty level.

Model	Metric	Cama	Standard	CoT	Reflection	React	S.Discover	P.Execute
llama-3.2-1b	Accuracy (↑)	**46.2**	0.0	7.7	15.4	0.0	15.4	15.4
	Unknown (↓)	**38.5**	100.0	92.3	84.6	100.0	84.6	84.6
	Tokens (↓)	14.8	5.0	**2.2**	15.2	33.2	15.9	19.7
	Time (↓)	5.0	**0.6**	1.3	2.7	80.7	88.2	45.6
llama3-8b	Accuracy (↑)	**94.9**	15.4	0.0	7.7	5.1	2.6	51.3
	Unknown (↓)	**0.0**	84.6	100.0	92.3	94.9	97.4	41.0
	Tokens (↓)	11.6	5.3	5.2	**3.4**	4.5	13.9	27.7
	Time (↓)	4.8	0.9	**0.8**	3.6	2.3	18.9	55.6
llama3-70b	Accuracy (↑)	**92.3**	84.6	76.9	76.9	12.8	25.6	56.4
	Unknown (↓)	**0.0**	**0.0**	10.3	7.7	87.2	64.1	35.9
	Tokens (↓)	14.8	**2.2**	**2.2**	8.9	4.1	12.8	8.8
	Time (↓)	95.5	**16.9**	26.5	100.5	20.6	130.1	93.6
gpt-4o-mini	Accuracy (↑)	**94.9**	79.5	84.6	7.7	12.8	79.5	76.9
	Unknown (↓)	**0.0**	17.9	12.8	92.3	82.1	20.5	20.5
	Tokens (↓)	12.1	3.1	**2.2**	9.0	4.5	19.0	474.8
	Time (↓)	24.0	**12.2**	18.1	52.6	21.5	174.6	324.3

Table 6: Performance comparison for Healthcare dataset
Note: The healthcare dataset represents a difficult level.

Model	Metric	Cama	Standard	CoT	Reflection	React	S.Discover	P.Execute
llama-3.2-1b	Accuracy (↑)	**48.7**	25.6	10.3	23.1	23.1	5.1	5.1
	Unknown (↓)	**43.6**	74.4	87.2	71.8	74.4	94.9	94.9
	Tokens (↓)	31.9	**4.5**	**4.5**	13.5	33.2	17.9	14.1
	Time (↓)	5.7	**2.7**	**2.7**	52.1	22.3	89.2	50.7
llama3-8b	Accuracy (↑)	**89.7**	38.5	20.5	28.2	15.4	17.9	17.9
	Unknown (↓)	**2.6**	56.4	79.5	64.1	82.1	79.5	79.5
	Tokens (↓)	23.7	5.0	**4.5**	12.8	8.4	22.2	39.7
	Time (↓)	5.8	**1.2**	8.8	27.2	15.1	43.6	44.0
llama3-70b	Accuracy (↑)	**89.7**	30.8	48.7	51.3	5.1	15.4	35.9
	Unknown (↓)	**0.0**	61.5	48.7	41.0	89.7	82.1	61.5
	Tokens (↓)	21.0	**5.1**	5.4	13.5	8.2	22.6	88.7
	Time (↓)	61.5	**18.6**	31.4	119.8	25.4	69.0	504.9
gpt-4o-mini	Accuracy (↑)	**92.3**	56.4	76.9	10.3	2.6	7.7	10.3
	Unknown (↓)	**0.0**	38.5	7.7	89.7	97.4	89.7	74.4
	Tokens (↓)	24.2	**5.5**	**5.5**	15.2	9.3	24.8	502.2
	Time (↓)	38.0	24.5	25.8	55.9	**23.6**	72.0	458.5