Process Mining Project 2:
Extracting and Evaluating Recommendations from Decision Trees

Lorenzo Fasol lorenzo.fasol@studenti.unitn.it
Stefano Camposilvan stefano.camposilvan@studenti.unitn.it

University of Trento
Process Mining and Management 2025/2026

January 2026

Contents

1 Introduction

This report documents the implementation of **Project Typology #2: Extracting and Evaluating Recommendations** for the 2025/2026 Process Mining and Management course offered by the University of Trento. The project addresses two main objectives:

1. **Prediction:** Build a predictive model to classify ongoing process instances based on their expected cycle time (fast vs. slow).
2. **Recommendation:** For traces predicted as negative (slow), extract actionable recommendations from the decision tree structure to guide the process towards a positive outcome.

1.1 Dataset Description

The project uses the `Production` manufacturing log with the following characteristics:

Attribute	**Value**
Domain	Manufacturing process
Label criterion	Average cycle time
Positive class (`true`)	Fast execution (cycle time $<$ average)
Negative class (`false`)	Slow execution (cycle time $\geq$ average)
Training traces	177 (80%)
Testing traces	43 (20%)
Unique activities	26 (including "unknown")

Table 1: Dataset characteristics

2 System Architecture

The implementation follows a modular architecture with clear separation of concerns:

- `config.py`: Centralized module including configurable file paths and constants, such as prefix lengths $[5, 10]$ and the hyperparameter search grid.
- `utils.py`: Data pipeline utilities for XES import, boolean encoding, and CSV export.
- `functions.py`: Core logic for model training, testing, tree visualization, positive path extraction, recommendation generation, and evaluation.
- `main.py`: Pipeline orchestrator that executes the workflow for each prefix length.

2.1 Execution Pipeline

The system executes the following steps for each prefix length:

1. Load training and testing XES files
2. Apply boolean encoding to create prefix representations

3. Optimize Decision Tree hyperparameters via GridSearchCV (5-fold CV)
4. Train the model with optimal parameters
5. Evaluate model performance on test prefixes
6. Extract all positive paths from the trained tree
7. Generate recommendations for negatively-predicted traces
8. Evaluate recommendations against full-length traces

3 Methodology

3.1 Encoding mechanism

Boolean encoding is used as traces' encoding mechanism, to preprocess said trace prefixes and create training and test data. Using this technique, each prefix is encoded as a binary vector where:

- Each position corresponds to a unique activity in the log
- Value 1 indicates the activity was executed within the prefix
- Value 0 indicates the activity was not executed

In addition to actual unique activities in the log, an **unknown** activity was also added, so as to represent possible new activities in future test logs.
Boolean encoding allows for:

1. **Transparency:** Binary features directly map to decision tree splits ("Was Activity X executed?"), enabling straightforward interpretation.
2. **Recommendation Extraction:** Tree conditions naturally translate to actionable recommendations ("execute Activity X" or "do not execute Activity Y").
3. **Compatibility:** Works seamlessly with varying trace lengths (shorter traces simply have fewer activities marked as executed).
4. **Efficiency:** Simple binary representation enables fast training and inference.

Alternative encodings (frequency-based, index-based, or neural embeddings) would produce continuous or sequential representations that obscure which specific activities trigger classification decisions, complicating recommendation extraction.

3.2 Hyperparameter Optimization

The Decision Tree classifier is optimized using `GridSearchCV` with 5-fold cross-validation. The search space includes:

Parameter	Values	Purpose
`criterion`	gini, entropy	Impurity measure for splits
`max_depth`	3, 5, 10, 15	Tree complexity control
`min_samples_split`	2, 5, 10	Minimum samples to split a node
`min_samples_leaf`	1, 2, 4, 5	Minimum samples in leaf nodes

Table 2: Hyperparameter search space ($2 \times 4 \times 3 \times 4 = 96$ combinations)

The optimization metric is **F1-score (macro-averaged)** to balance precision and recall across both classes.

3.3 Recommendation Extraction Algorithm

The recommendation extraction process consists of four steps:

3.3.1 Step 1: Extract Positive Paths

The decision tree is traversed recursively to identify all paths leading to leaves that predict the positive class (`true`). For each path, we store:

- The sequence of conditions: (activity, required_presence) pairs
- The confidence score: proportion of positive samples at the leaf

$$\text{confidence}(path) = \frac{\#\text{positive samples at leaf}}{\#\text{total samples at leaf}} \tag{1}$$

3.3.2 Step 2: Filter Compatible Paths

For each negatively-predicted prefix, we then filter positive paths to retain only those **compatible** with the current execution state. A path is **incompatible** if it requires an activity to be absent, but that activity has already been executed in the prefix:

$$\text{Incompatible} \iff \exists (a, req) \in path : (\text{encoding}[a] = 1) \wedge (req = 0) \tag{2}$$

This situation represents an irreversible constraint, as an activity cannnot be "un-executed".

Encoding[a]	Requirement	Status	Interpretation
1	1	Compatible	Already executed as required
0	1	Compatible	Not yet executed, can be done
1	0	**Incompatible**	Already executed, path requires absence
0	0	Compatible	Not executed, not required

Table 3: Compatibility matrix for path filtering

3.3.3 Step 3: Select Best Path

Among compatible paths, we select the one with the **highest confidence**:

$$\text{best\_path} = \arg \max_{p \in \text{compatible\_paths}} \text{confidence}(p) \tag{3}$$

In case of ties in confidence, different tie-breakers were possible (e.g., shortest path, fewest required changes, or most actionable activities). In this project, random selection was used to avoid systematically favoring specific activities; however, deterministic tie-breakers could improve reproducibility and may increase practical usefulness depending on the operational setting.

3.3.4 Step 4: Generate Recommendations

From the selected path, we extract conditions not yet satisfied in the prefix:

- If $req = 1$ and encoding$[a] = 0$: recommend "Activity a **has to be executed**"
- If $req = 0$ and encoding$[a] = 0$: recommend "Activity a **does not have to be executed**"

Conditions already satisfied (encoding matches requirement) are omitted from recommendations.

3.4 Recommendation Evaluation

Recommendations are evaluated by comparing suggested actions against the **actual continuation** of each trace. The evaluation considers only traces that were predicted as negative (and thus received recommendations).

3.4.1 Classification Scheme

- **Recommendation Followed:** All recommended activities were executed (or avoided) as suggested in the full trace
- **Ground Truth Positive:** The full trace has label `true` (fast cycle time)

	Ground Truth = true	**Ground Truth = false**
Followed	TP	FP
Not Followed	FN	TN

Table 4: Recommendation evaluation confusion matrix

3.4.2 Metrics

$$\text{Accuracy}_{\text{recc}} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$\text{Precision}_{\text{recc}} = \frac{TP}{TP + FP} \tag{5}$$

$$\text{Recall}_{\text{recc}} = \frac{TP}{TP + FN} \tag{6}$$

$$\text{F-measure}_{\text{recc}} = \frac{2 \times \text{Precision}_{\text{recc}} \times \text{Recall}_{\text{recc}}}{\text{Precision}_{\text{recc}} + \text{Recall}_{\text{recc}}} \tag{7}$$

4 Tests and Results

To evaluate the proposed approach, prefix lengths of 5 and 10 were selected. This choice allows analyzing the trade-off between prediction reliability and recommendation feasibility. In fact, prefix length 5 represents an early-stage intervention point, where limited information is available but corrective actions are still feasible. Prefix length 10, on the other hand, represents a later-stage intervention point, where richer contextual information enables more accurate predictions, but fewer actions remain possible.

The following sections report and compare the obtained results for both prefix lengths in terms of prediction and recommendation performance.

4.1 Hyperparameters Optimization

The GridSearchCV optimization identified the best configurations reported in Table 5.

Parameter	**Prefix 5**	**Prefix 10**
criterion	entropy	entropy
max_depth	15	15
min_samples_leaf	1	1
min_samples_split	10	2

Table 5: Best hyperparameters from GridSearchCV

Both models selected entropy and a maximum depth of 15, indicating the need for expressive decision boundaries. The smaller `min_samples_split` value for prefix 10 reflects the higher confidence enabled by additional process context.

4.2 Model Prediction

Prefix Length	**Accuracy**	**Precision**	**Recall**	**F-measure**
5	0.6279	0.8333	0.6250	0.7143
10	0.8372	0.8788	0.9062	0.8923

Table 6: Model prediction performance on test prefixes

Key observations:

- Accuracy improves by more than 21% from prefix 5 to prefix 10.
- Recall increases significantly, showing that more fast cases are correctly identified.
- The F-measure confirms a much better balance between precision and recall at prefix 10.

These results confirm that prediction accuracy strongly benefits from additional contextual information. Early prefixes lack sufficient evidence to reliably distinguish between fast and slow traces, while later prefixes capture more discriminative activities and process patterns.

4.3 Positive Paths Extraction

Prefix Length	Positive Paths	Max Confidence
5	20	1.0000
10	26	1.0000

Table 7: Positive paths extracted from decision trees

Prefix 10 produces a higher number of positive paths, indicating a more expressive model capable of finer distinctions among fast executions. However, this also increases the strictness of constraints when generating compatible recommendations.

4.4 Recommendation Generation

Prefix Length	Negative Predictions	With Recommendations	No Compatible Path
5	19	16 (84.2%)	3 (15.8%)
10	10	4 (40.0%)	6 (60.0%)

Table 8: Recommendation generation coverage

Key observations:

- Prefix 5 generates more negative predictions and therefore more recommendation opportunities.
- Prefix 10 generates fewer negative predictions but suffers from much lower recommendation coverage due to incompatibility with executed activities.

4.5 Recommendation Evaluation

Prefix	TP	FP	TN	FN	Acc	Prec	Rec	F1
5	0	0	7	12	0.3684	0.0000	0.0000	0.0000
10	0	0	7	3	0.7000	0.0000	0.0000	0.0000

Table 9: Recommendation evaluation metrics

Although recommendation accuracy improves at prefix 10, no true positive recommendation was observed for either prefix length.

5 Analysis and Discussion

5.1 Prediction vs Recommendation Accuracy

A clear contrast emerges between prediction and recommendation performance:

- Prediction accuracy improves substantially with longer prefixes.
- Recommendation accuracy improves only because fewer cases are evaluated.
- Recommendation precision and recall remain zero due to the absence of true positives.

This contrast highlights an important conceptual distinction: predicting an outcome is fundamentally easier than prescribing effective corrective actions. The prediction task benefits directly from additional contextual information, while the recommendation task is constrained by feasibility, controllability, and causal validity.

The improvement in prediction accuracy for longer prefixes can be explained by the fact that more activities, quality checks, and rework indicators are observed. These activities implicitly encode delays, deviations, and inefficiencies that strongly correlate with cycle time. Early prefixes, instead, either contain mostly standard activities that are common to both fast and slow traces, or too few activities to actually learn to distinguish between cases in a reliable way.

In contrast, recommendation accuracy does not follow the same trend. Although the classifier becomes more accurate, the recommendation module must operate within the remaining degrees of freedom of the process. As the prefix length increases, many future choices are already fixed, and the opportunity to alter the outcome through alternative activity patterns becomes increasingly limited.

5.2 Qualitative Recommendation Inspection

Recommendations extracted from decision trees describe activity presence patterns associated with fast executions, but they do not guarantee that such activities are controllable or feasible. In practice, these recommendations represent *idealized optimal paths* derived from historical correlations.

5.2.1 Example: Case124 (Prefix 5)

- **Prediction:** slow
- **Ground truth:** fast
- **Recommendations:**
  - Turning Q.C. must be executed
  - Turning & Milling - Machine 10 must be executed
  - Grinding Rework - Machine 27 must not be executed

- **Actual continuation:** Turning & Milling Q.C., Laser Marking, Lapping, Turning & Milling - Machine 4, Packing, Final Inspection Q.C.

From a process perspective, the recommendations are intuitively reasonable: they encourage quality control execution and avoidance of rework, which are typically associated with better performance. However, they assume that machine assignment and quality control routing are freely controllable, which is rarely the case in real manufacturing environments.

The actual execution reflects operational constraints such as machine availability, queue lengths, production planning priorities, and resource allocation decisions that are not encoded in the processed training and testing data. This illustrates that recommendations are often optimal from a historical data perspective but not from an operational feasibility perspective.

5.3 Why True Positives Are Absent

The absence of true positives (TP = 0) is a direct consequence of both methodological and practical factors:

1. **Strict evaluation criterion:** A recommendation is considered followed only if all suggested conditions are satisfied. Partial compliance is penalized, which significantly reduces the chance of observing TP cases.

2. **Resource and time constraints:** Recommendations such as "use Machine 10" implicitly assume resource availability. In practice, machines may be busy, under maintenance, or allocated to higher-priority jobs.

3. **Hidden business rules:** Scheduling policies, customer priorities, and contractual obligations influence routing decisions but are not captured in the event log.

4. **Correlation vs causation:** Decision trees learn correlations. A recommended activity may be associated with fast cases without being the actual cause of fast completion.

Therefore, even when recommendations are theoretically optimal, they are rarely executable exactly as prescribed.

5.4 Interpretation of Recommendation Accuracy Results

The apparent improvement in recommendation accuracy for prefix 10 is mainly due to a reduction in FN cases rather than an increase in TP. This means that the system becomes better at identifying situations where recommendations are not followed and the outcome remains negative, but it does not become better at generating effective, outcome-changing advice.

This confirms that the recommendation module currently behaves more as a diagnostic tool than as a true prescriptive system.

5.5 Prefix Length Trade-offs

A general comparison of results for the two used prefix lengths are provided in the table below:

Aspect	**Prefix 5**	**Prefix 10**
Prediction Accuracy	Lower	Higher
Recommendation Coverage	Higher	Lower
Intervention Time	Larger	Smaller
Decision Flexibility	Higher	Lower
Process Rigidity	Lower	Higher

Table 10: Trade-offs between prefix lengths

It is observable that prefix 5 offers greater flexibility, as many decisions are still open, but prediction reliability is limited. Prefix 10, on the other hand, offers more reliable predictions but operates in a much more constrained decision space, where most corrective actions are no longer feasible. This highlights a classic problem of predictive process monitoring: early intervention is uncertain, while late intervention is accurate but ineffective.

6 Conclusions

The proposed project investigated a decision-tree-based predictive process monitoring approach with integrated recommendation generation. The experimental results highlight a clear distinction between the effectiveness of prediction and that of recommendation.

From the prediction perspective, the results confirm that model performance strongly benefits from longer prefixes. As more contextual information becomes available, the classifier is able to capture increasingly discriminative patterns related to cycle time, leading to a significant improvement in accuracy, recall, and F-measure. This confirms the suitability of decision trees as transparent and interpretable models for predictive monitoring tasks.

From the recommendation perspective, however, the results reveal important limitations. Although recommendations are logically consistent with historical fast executions, they rarely translate into actionable or effective interventions in practice. The absence of true positive recommendations demonstrates that correlation-based patterns are insufficient to guarantee operational feasibility or causal impact. Recommendations represent idealized optimal paths derived from historical data, but real process executions are constrained by resource availability, scheduling policies, non-value-added activities, and business rules that are not captured in the event log.

The comparison between prefix lengths further highlights a fundamental trade-off. Early prefixes offer greater flexibility and a wider intervention window, but suffer from limited predictive reliability. Later prefixes provide much more accurate predictions, yet operate in a rigid decision space where most corrective actions are no longer feasible. This confirms that early intervention is uncertain, while late intervention is accurate but often ineffective.

Overall, the obtained results suggest that decision-tree-based models are highly effective for transparent and interpretable prediction, but that recommendation generation re-

quires richer process representations, causal validation, and explicit feasibility constraints in order to become practically useful.

6.1 Future Work

Future research should focus on extending the proposed approach in several directions:

- Incorporating sequence-aware and temporal learning models to better capture ordering and duration effects.
- Introducing causal analysis techniques to distinguish actionable factors from mere correlations.
- Designing softer evaluation criteria that account for partial compliance with recommendations.
- Integrating resource, workload, and scheduling information to improve recommendation feasibility.
- Validating recommendations through expert feedback and human-in-the-loop experimentation.

These extensions would contribute to transforming predictive monitoring systems from descriptive and diagnostic tools into truly prescriptive decision-support systems.