Process Mining and Intelligence Project

Emotion Based Music Selection

Ettore Ricci Francesco Boldrini Paolo Palumbo Zahra Omrani

February 3, 2025

Contents

1 BPMN modeling

1.1 Process landscape

[*Ettore Ricci, Paolo Palumbo, Francesco Boldrini, Zahra Omrani*]

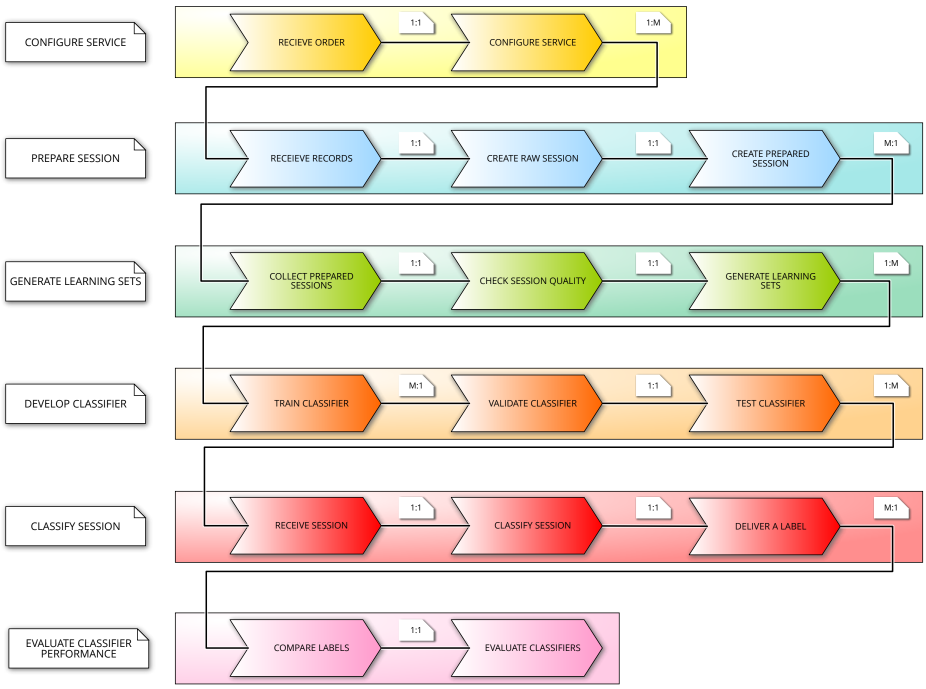


Figure 1: Process landscape

1.2 Process model

1.2.1 Prepare session

[*Ettore Ricci, Paolo Palumbo*]

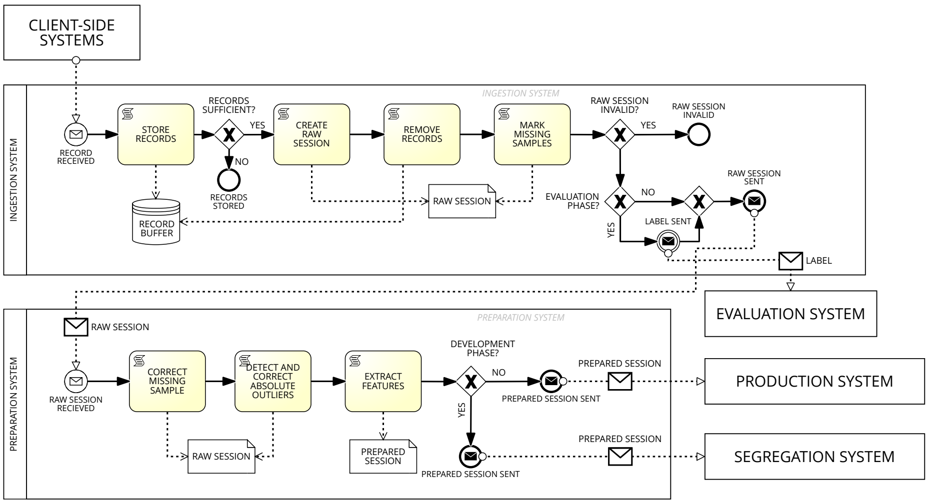


Figure 2: Business Diagram of the "Prepare session" process

1.2.2 Generate learning sets

[*Ettore Ricci, Paolo Palumbo*]

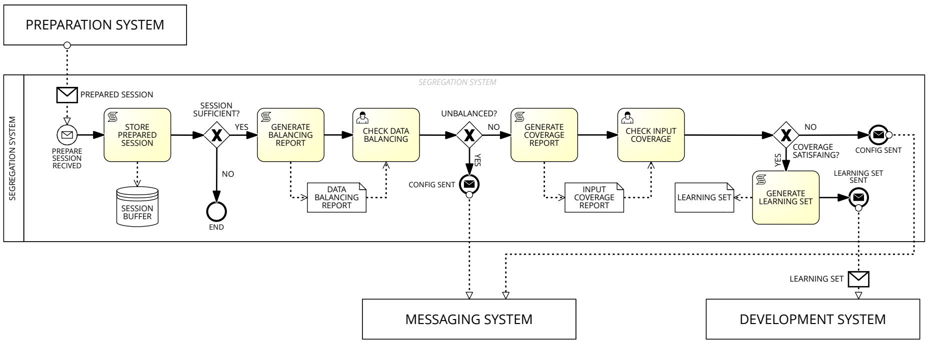


Figure 3: Business Diagram of the "Generate learning sets" process

1.2.3 Develop classifier

[*Ettore Ricci, Paolo Palumbo*]

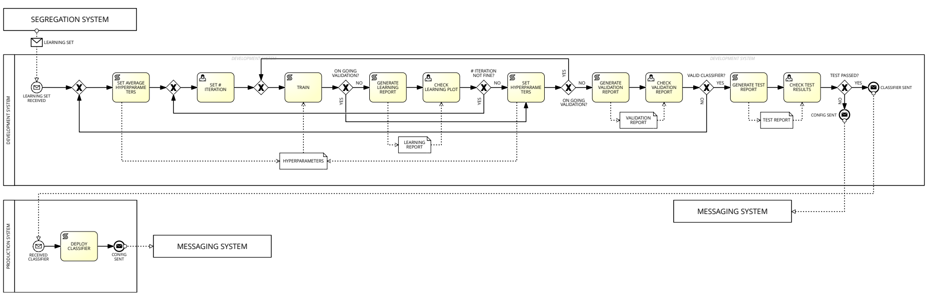


Figure 4: Business Diagram of the "Develop classifier" process

1.2.4 Classify session

[*Ettore Ricci, Paolo Palumbo*]

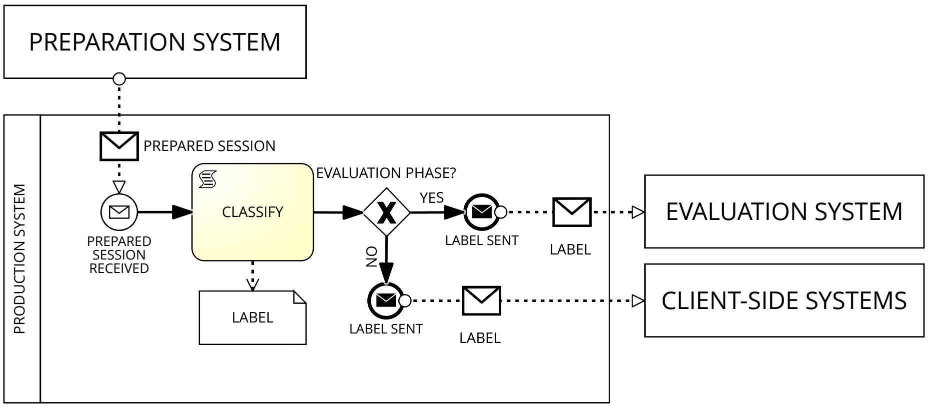


Figure 5: Business Diagram of the "Classify session" process

1.2.5 Evaluate classifier performance

[*Ettore Ricci, Paolo Palumbo*]

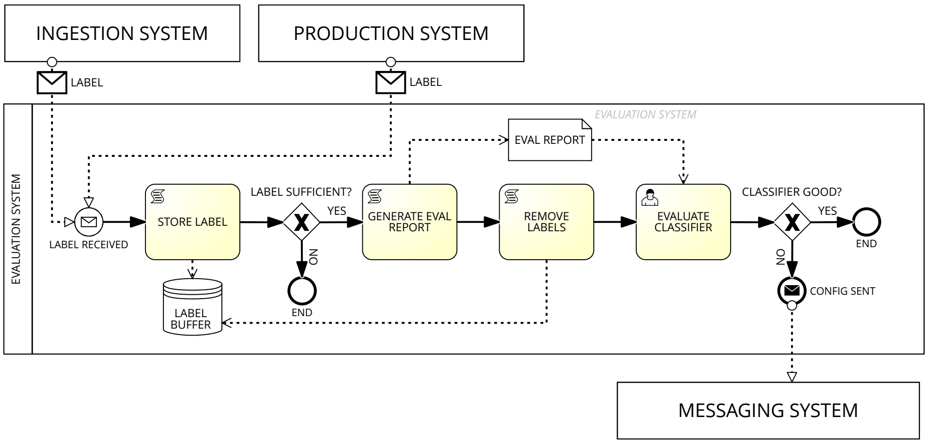


Figure 6: Business Diagram of the "Evaluate classifier performance" process

1.2.6 Configure systems

[*Ettore Ricci, Paolo Palumbo*]

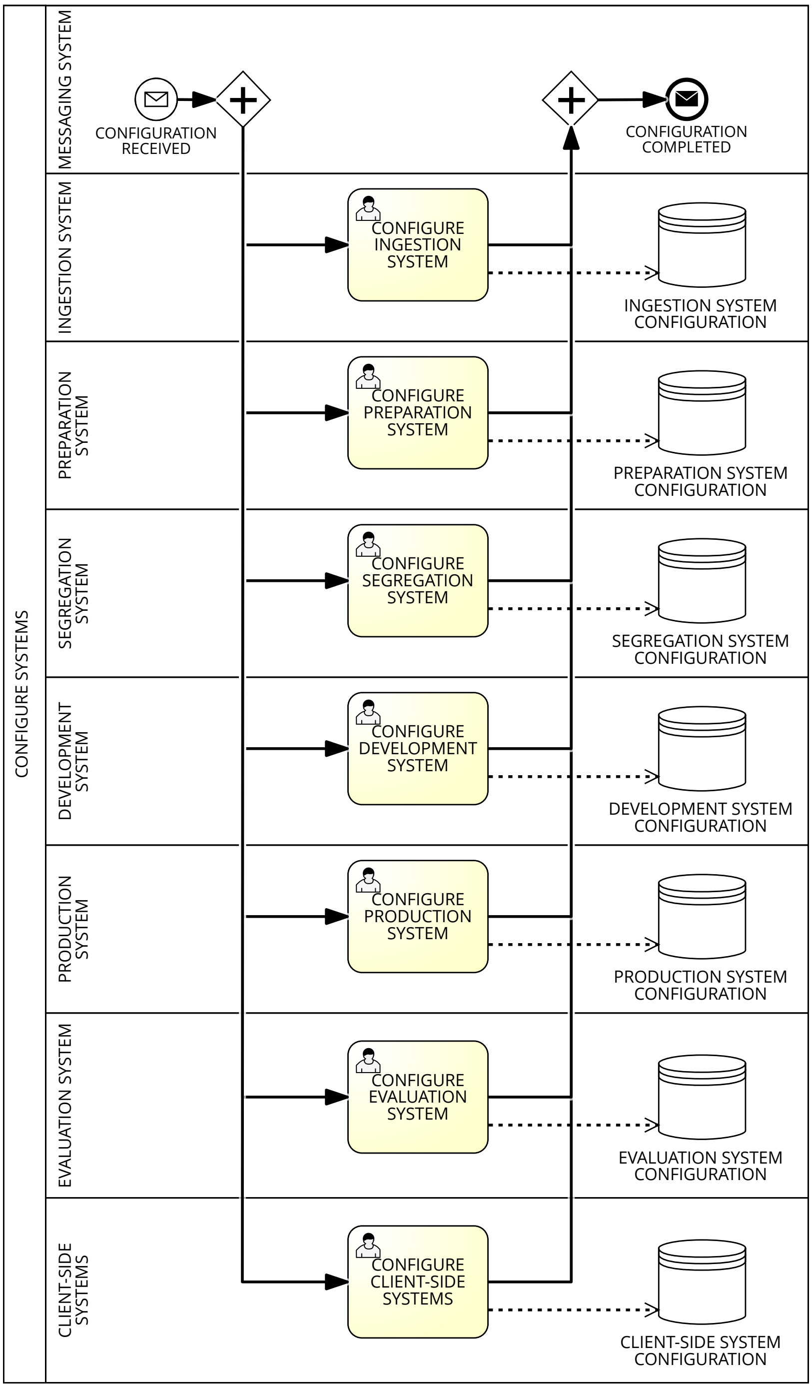


Figure 7: Business Diagram of the "Configure systems" process

2 Data modeling

2.1 Process model

2.1.1 Prepare session

[*Ettore Ricci*]

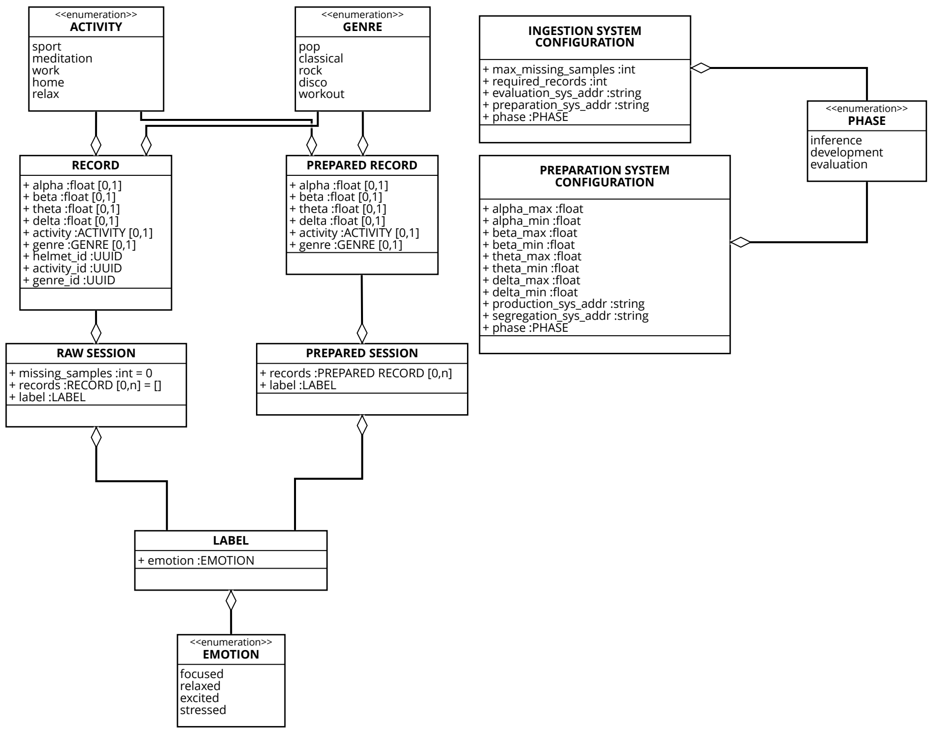


Figure 8: Data Model of the "Prepare session" process

2.1.2 Generate learning sets

[*Paolo Palumbo*]

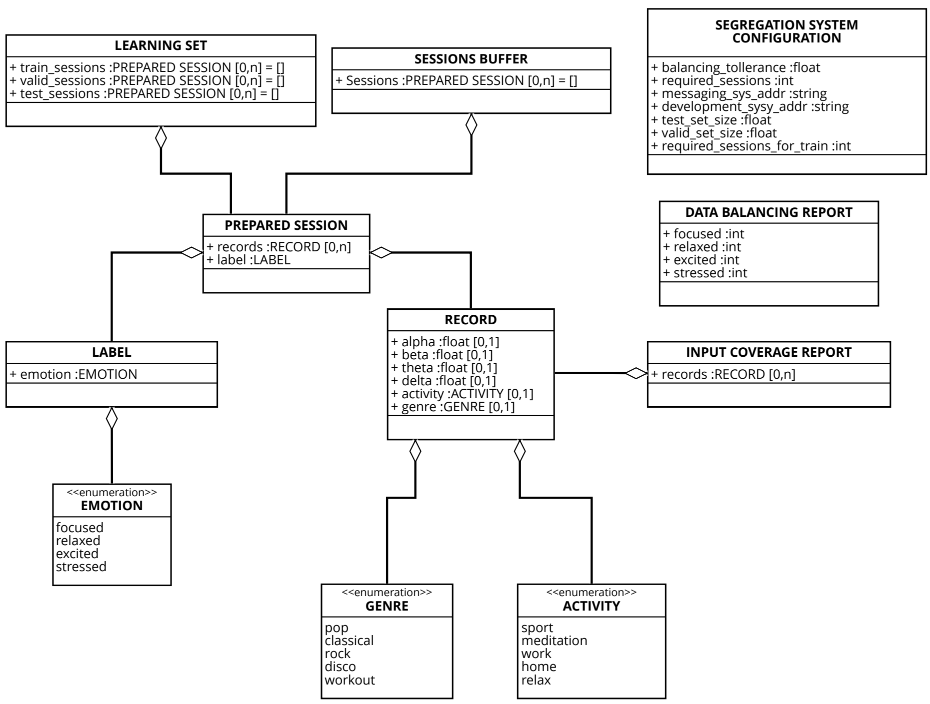


Figure 9: Data Model of the "Generate learning sets" process

2.1.3 Develop classifier

[*Paolo Palumbo*]

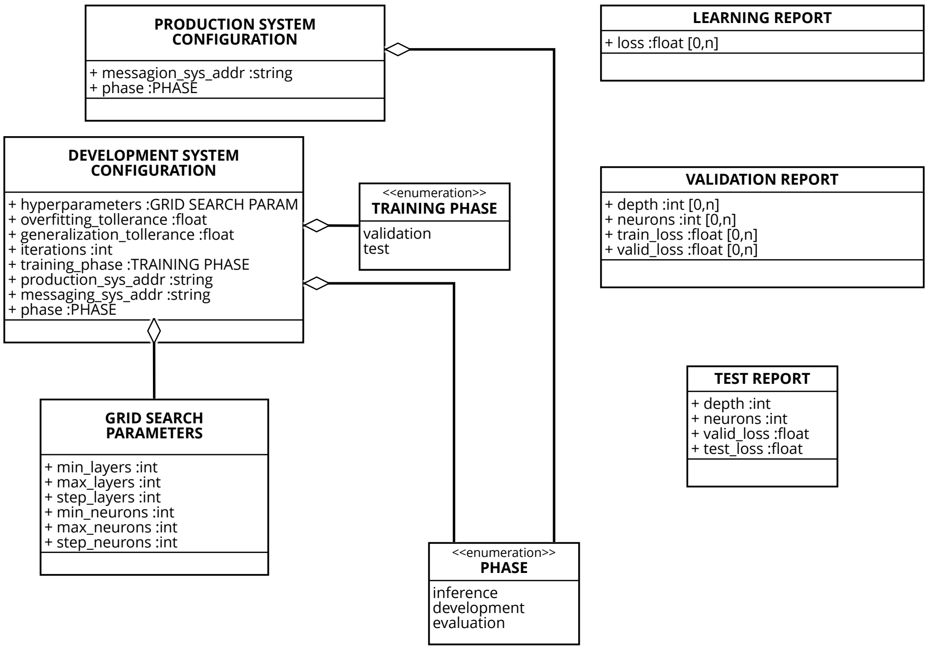


Figure 10: Data Model of the "Develop classifier" process

2.1.4 Classify session

[*Francesco Boldrini*]

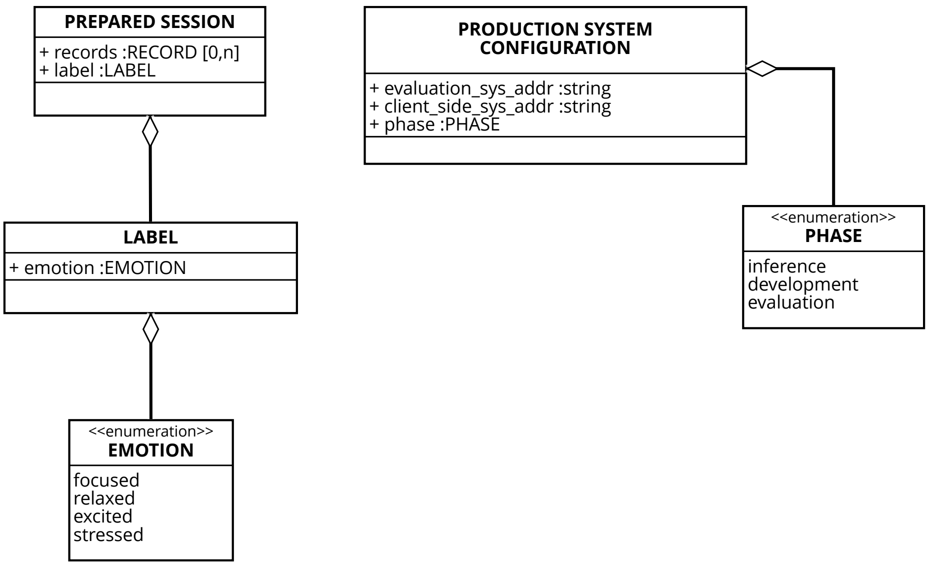


Figure 11: Data Model of the "Classify session" process

2.1.5 Evaluate classifier performance

[*Zahra Omrani*]

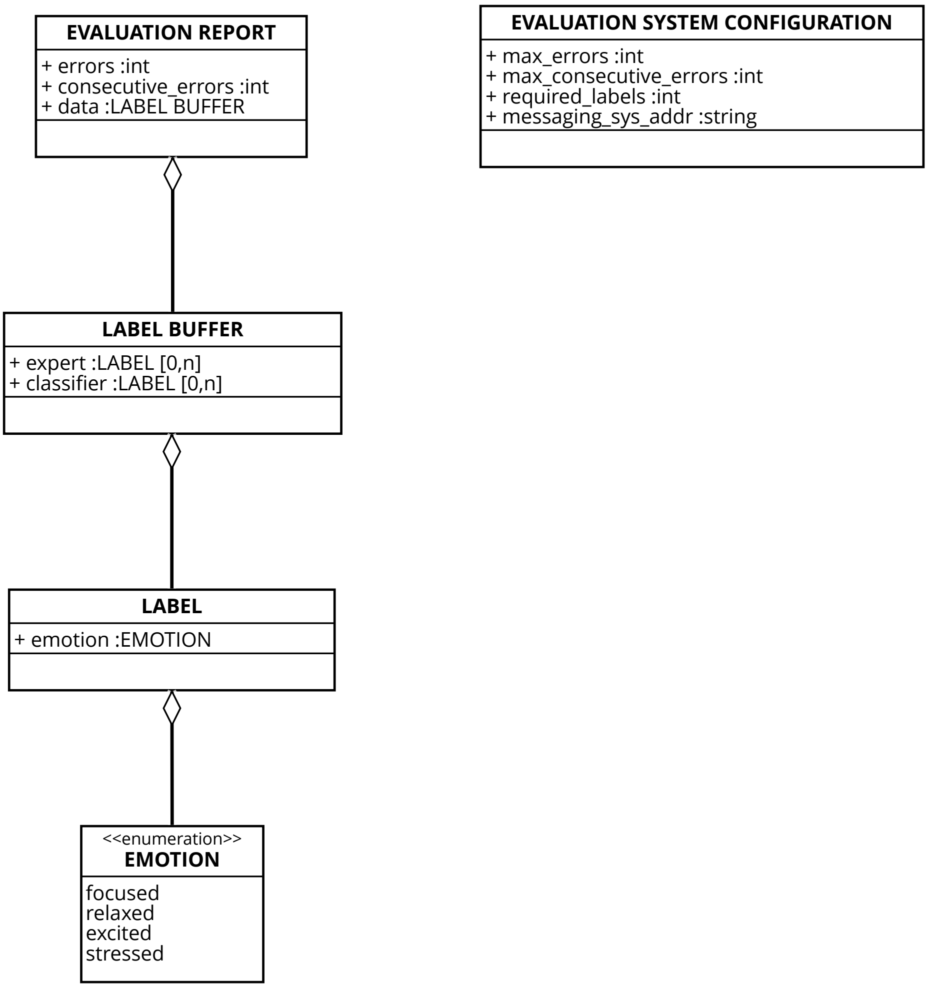


Figure 12: Data Model of the "Evaluate classifier performance" process

3 Task level modeling

3.1 Roles and salaries

[*Ettore Ricci, Paolo Palumbo*]

Position	**Description**	**Salary**	**Normalized Salary**
Clerk	Handles administrative tasks, organizes documentation, and assists with data entry and labeling. Ensures smooth operations by coordinating communication and managing resources.	$52,000.00	1.00
Data analyst	Prepares, analyzes, and visualizes data to extract insights. Collaborates on cleaning datasets, identifying trends, and supporting model validation.	$60,000.00	1.15
ML engineer	Builds, tests, and deploys machine learning models, optimizing performance and scalability. Integrates AI solutions into production systems with a focus on efficiency.	$130,000.00	2.50
Data scientist	Designs and experiments with AI models, applying advanced techniques to solve project challenges. Collaborates with experts to integrate domain knowledge and refine outputs.	$123,000.00	2.37
Domain expert (Neurologist)	Provides medical expertise to guide AI development and validate results. Ensures solutions align with clinical standards and address neurological challenges.	$267,000.00	5.13
Minimum		$52,000.00	1.00

Table 1: Salary and normalized salary for each position

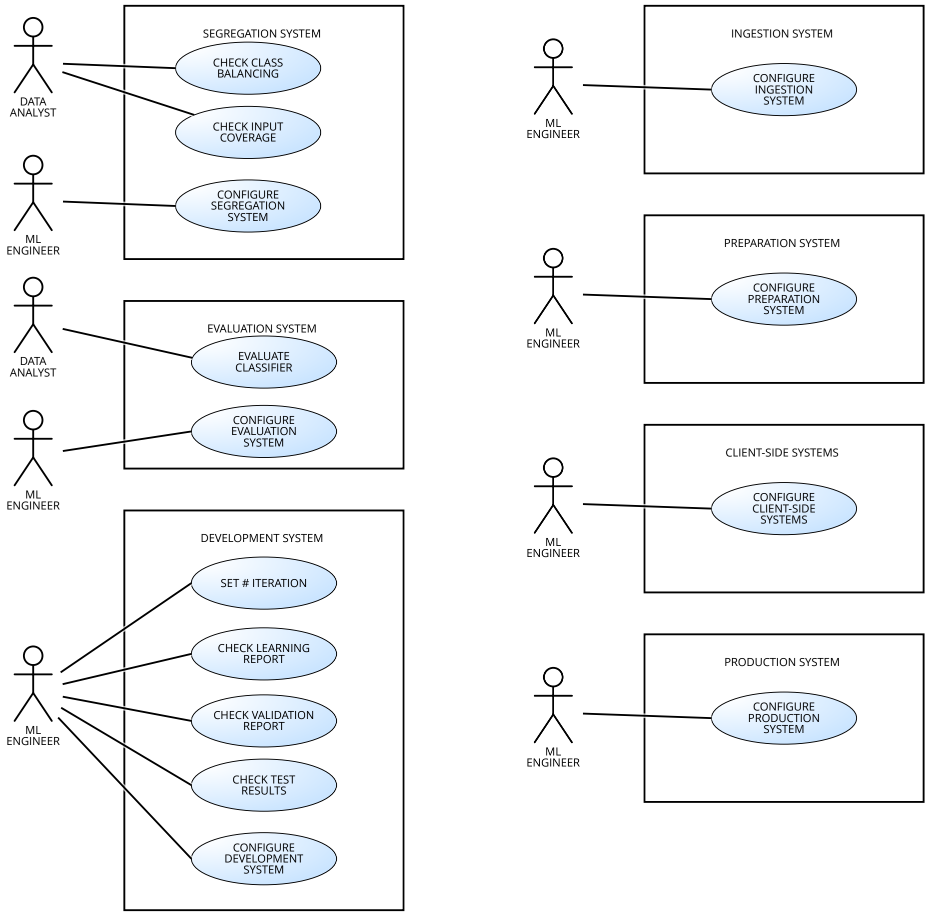


Figure 13: Use case diagram

3.2 Segregation system

3.2.1 Check data balancing

[*Ettore Ricci, Paolo Palumbo*]

The task is performed by a Data Analyst.

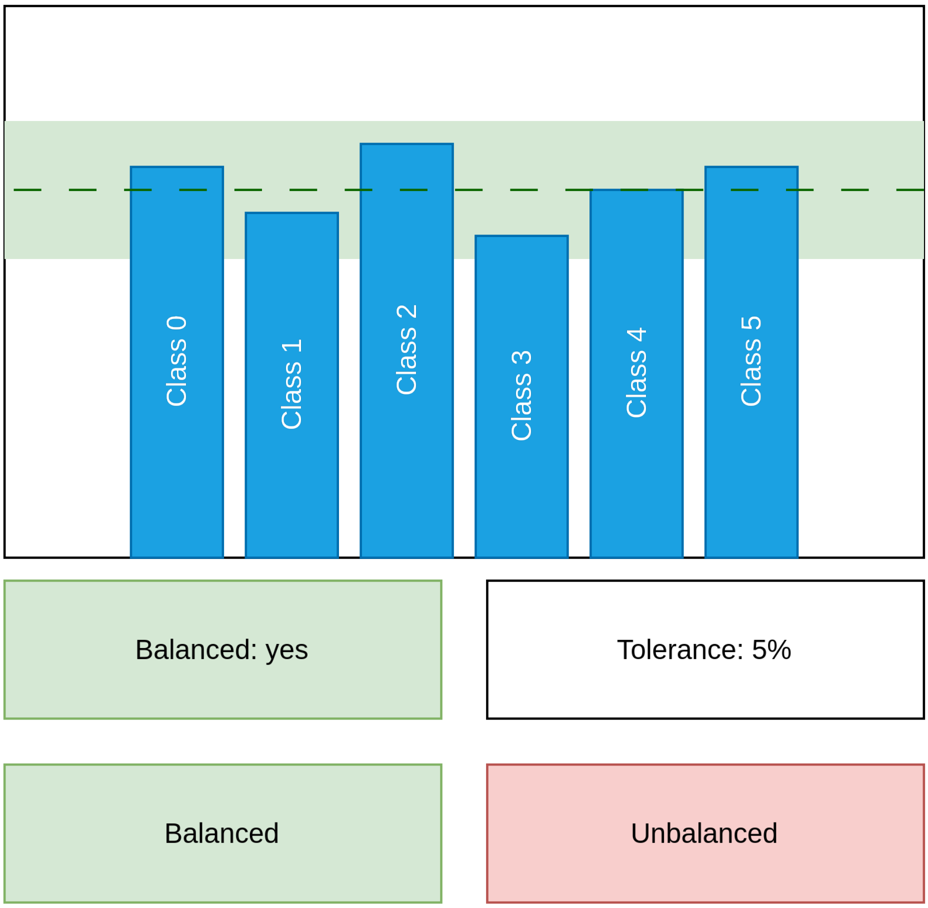


Figure 14: ”Check data balancing” mock-up form

Step	O	CL	S	SC
1 ACTOR opens ”Check data balancing” form.	1	1	1.15	1.15
2 SYSTEM shows the report.				
3 SYSTEM shows a hint whether the data is balanced or not.				
4 ACTOR checks the hint to see if the data is balanced or not.	1	2	1.15	2.30
5.1 IF the data is balanced.	0.2			
5.1.1 ACTOR clicks ”Balanced” button.	0.2	1	1.15	0.23
5.2 ELSE	0.8			
5.2.1 ACTOR clicks ”Unbalanced” button.	0.8	1	1.15	0.92
7 SYSTEM shows a confirmation dialog.				
8 ACTOR closes the form.	1	1	1.15	1.15
	Human task cost			5.74

Table 2: Detailed use case for ”Check data balancing” task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.2.2 Check input coverage

[*Ettore Ricci, Paolo Palumbo*]
The task is performed by a Data Analyst.

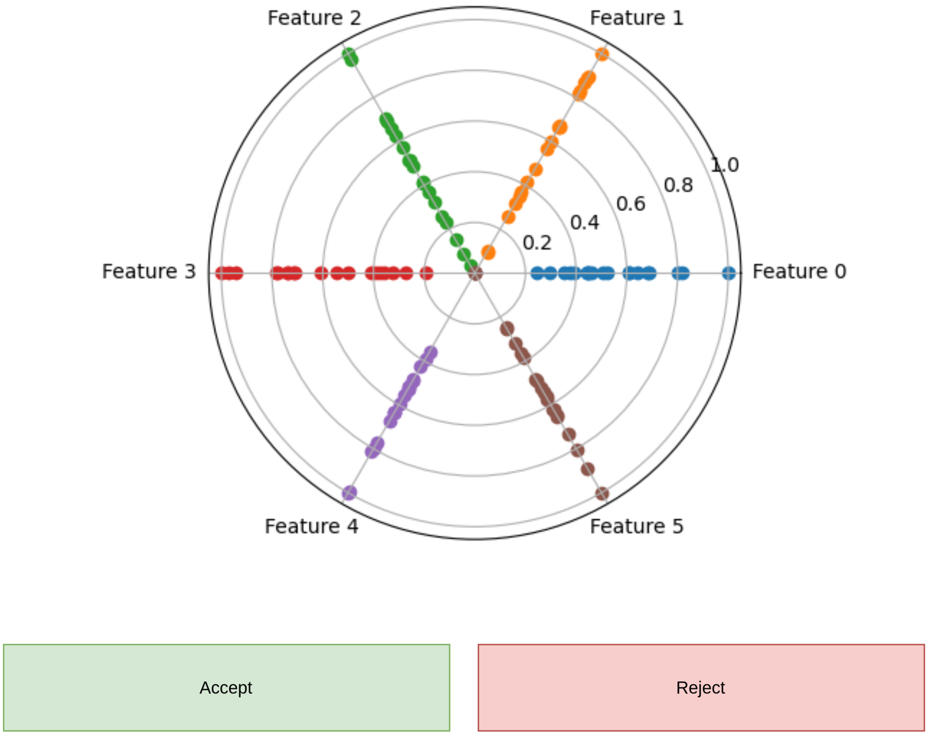


Figure 15: "Check input coverage" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens "Check input coverage" form.	1	1	1.15	1.15
2 SYSTEM shows a radar scatter plot of the input distribution.				
3 FOR EACH radius in the radar scatter plot:	6			
3.1 ACTOR checks if the distribution is uniform on the radius.	6	4	1.15	27.6
3.1.1 IF the distribution is not uniform as expected.	4			
3.1.1.1 THEN the input coverage is not satisfied.	4			
4.1 IF the input coverage is satisfied.	0.33			
4.1.1 ACTOR clicks "Accept" button.	0.33	1	1.15	0.38
4.2 ELSE	0.66			
4.2.1 ACTOR clicks "Reject" button.	0.66	1	1.15	0.76
5 SYSTEM shows a confirmation dialog.				
6 ACTOR closes the form.	1	1	1.15	1.15
			Human task cost	31.04

Table 3: Detailed use case for "Check input coverage" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.2.3 Configure Segregation System

[*Francesco Boldrini, Zahra Omrani*]

This task is performed by a ML Engineer.

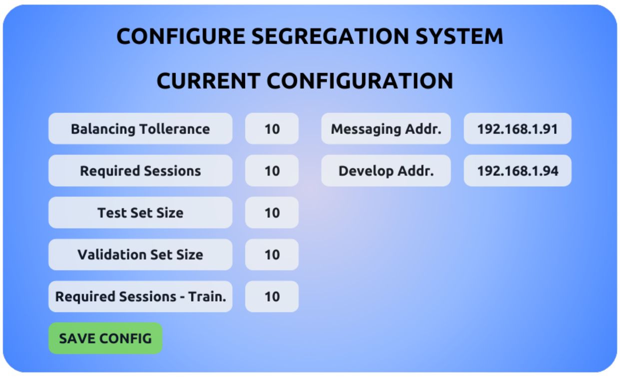


Figure 16: "Configure Segregation System" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens the "Configure Segregation System" form.	1	1	2.50	2.50
2 SYSTEM displays the current configuration.				
3 ACTOR sets the `balancing_tolerance`.	1	4	2.50	10
4 ACTOR sets the `required_sessions`.	1	4	2.50	10
5 ACTOR sets the `messaging_sys_addr`.	1	1	2.50	2.50
6 ACTOR sets the `development_sys_addr`.	1	1	2.50	2.50
7 ACTOR sets the `test_set_size`.	1	4	2.50	10
8 ACTOR sets the `valid_set_size`.	1	4	2.50	10
9 ACTOR sets the `required_sessions_for_train`.	1	4	2.50	10
10 SYSTEM validates the configuration.				
10.1 IF the configuration is correct and properly formatted:				
10.1.1 SYSTEM displays a confirmation message.				
10.2 ELSE (if the configuration is incorrect):				
10.2.1 SYSTEM displays an error message and aborts the process.				
11 ACTOR saves the form.	1	1	2.50	2.50
			Human task cost	60

Table 4: Detailed use case for "Configure Segregation System" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.3 Development system

3.3.1 Set iteration number

[*Zahra Omrani*]
The task is performed by a ML engineer.

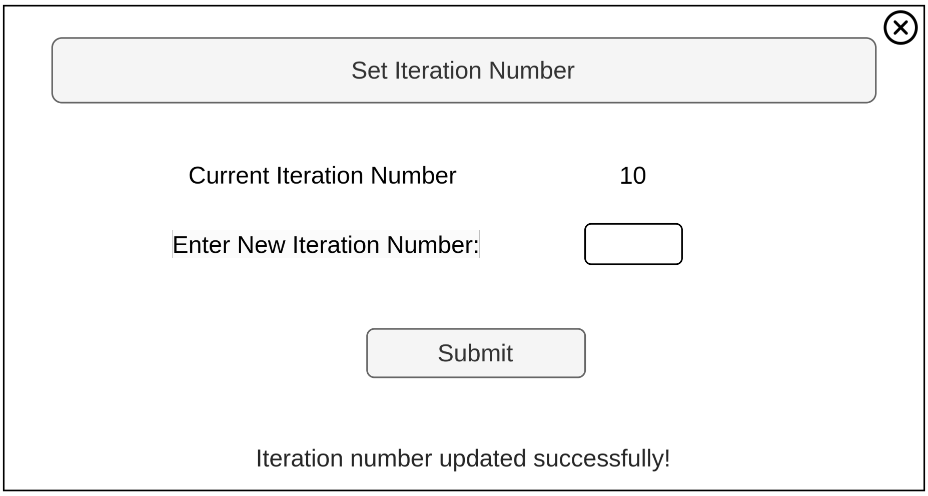


Figure 17: "Set iteration number" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens "Set Iteration Number" form.	1	1	2.5	2.5
2 SYSTEM displays the current iteration number.				
3.1 IF it's the first configuration:				
3.1.1 ACTOR inputs the desired number of iterations based on task complexity and previous experience.	0.002	3	2.5	0.015
3.2 ELSE (subsequent configurations):				
3.2.1 ACTOR inputs the number based on the established learning curve.	0.998	1	2.5	2.495
4 ACTOR clicks "Submit" button to confirm the iteration number.	1	1	2.5	2.5
5 SYSTEM shows a confirmation dialog.				
6 ACTOR closes the form.	1	1	2.5	2.5
			Human task cost	10.01

Table 5: Detailed use case for "Set iteration number" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.3.2 Check learning report

[*Paolo Palumbo*]

The task is performed by a ML engineer.

Figure 18: "Check learning report" mock-up form

Step	O	CL	S	SC
1 ACTOR opens "Check training report" form.	1	1	2.50	2.50
2 SYSTEM shows the training loss curve.				
3 ACTOR checks the learning curve.	1	3	2.50	7.50
3.1 IF the loss is flat for at least half of the iterations:	0.4			
3.1.1 THEN ACTOR clicks "Overfit" button.	0.4	1	2.50	1.00
3.2 IF the loss is not flat at the end of the iterations:	0.4			
3.2.1 THEN ACTOR clicks "Underfit" button.	0.4	1	2.50	1.00
3.3 ELSE	0.2			
3.3.1 ACTOR clicks "Approved" button.	0.2	1	2.50	0.50
4 SYSTEM shows a confirmation dialog.				
5 ACTOR closes the form.	1	1	2.50	2.50
		Human task cost		15

Table 6: Detailed use case for "Check training report" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.3.3 Check validation report

[*Ettore Ricci*]

This task is performed by a ML engineer.

	ID	Depth	Neurons	Train MSE	Valid MSE	Delta MSE
1	954	3	4000	0.13	0.14	0.01
2	321	4	3000	0.23	0.24	0.01
3	5	3	1000	0.35	0.35	0.00
4	764	2	2000	0.24	0.45	0.21
5	202	3	2500	0.20	0.47	0.27

Reject

Overfitting Tolerance:	0.10

Figure 19: "Check validation report" mock-up form

Step	O	CL	S	SC
1 ACTOR opens "Check validation report" form.	1	1	2.5	2.5
2 SYSTEM shows the best 5 models sorted by increasing Validation Loss.				
3 FOR EACH model in the list:	5			
3.1 ACTOR calculates model Validation Loss minus the Training Loss	1	3	2.5	7.5
3.2 IF 3.1 is less than the Overfitting Tolerance and the Best Model is not selected.	1	3	2.5	7.5
3.2.1 THEN ACTOR selects the model as the Best Model.	1	1	2.5	2.5
4 FOR EACH model in the list aside form the previous:	4			
4.1 ACTOR calculates model Validation Loss minus the Training Loss	1	3	2.5	7.5
4.2 IF 4.1 is less than the Overfitting Tolerance and the Second Best Model is not selected.	1	3	2.5	7.5
4.2.1 THEN select the model as the Second Best Model.	0.25	1	2.5	0.625
5 ACTOR calculates if the Validation Loss of the Second Best Model is one order of magnitude greater than the Validation Loss of the Best Model.	1	3	2.5	7.5
6.1 IF the Best Model is not selected.	0.05	1	2.5	0.125
6.1.1 ACTOR clicks "Reject" button.	0.05	1	2.5	0.125
6.2 ELSE IF the Second Best Model is not selected or **5** is true	0.3	3	2.5	2.25
6.2.1 ACTOR clicks on the Best Model.	0.3	1	2.5	0.75
6.3 ELSE	0.65	3	2.5	4.875
6.3.1 ACTOR clicks on the least complex model among the Best Model and the Second Best Model.	0.65	3	2.5	4.875
7 SYSTEM shows a confirmation dialog.				
8 ACTOR closes the form.	1	1	2.5	2.5
	Human task cost			175.5

Table 7: Detailed use case for "Check validation report" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.3.4 Check test results

[*Ettore Ricci*]
This task is performed by a ML engineer.

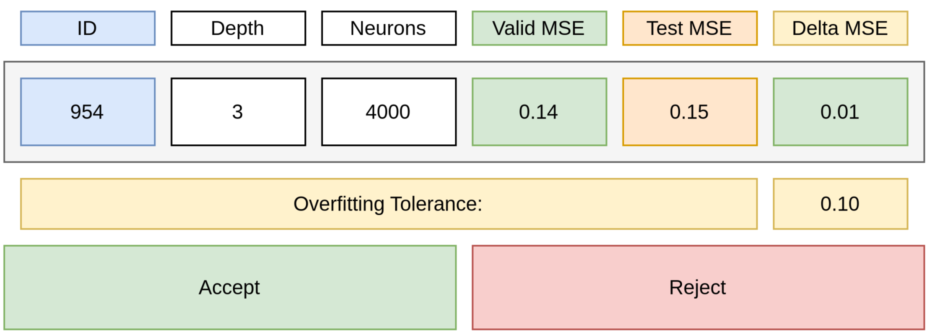


Figure 20: "Check test results" mock-up form

Step	O	CL	S	SC
1 ACTOR opens "Check test results" form.	1	1	2.5	2.5
2 SYSTEM shows the test results.				
3 ACTOR checks if the difference between the test results and the validation results is within overfitting tolerance.	1	2	2.5	5
4.1 IF the test results is not satisfactory.	0.01			
4.1.1 ACTOR clicks "Reject" button.	0.01	1	2.5	0.025
4.2 ELSE	0.99			
4.2.1 ACTOR clicks "Approve" button.	0.99	1	2.5	2.475
5 SYSTEM shows a confirmation dialog.				
6 ACTOR closes the form.	1	1	2.5	2.5
	Human task cost			12.5

Table 8: Detailed use case for "Check test results" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.3.5 Configure Development System

[*Francesco Boldrini, Zahra Omrani*]
This task is performed by a ML Engineer.

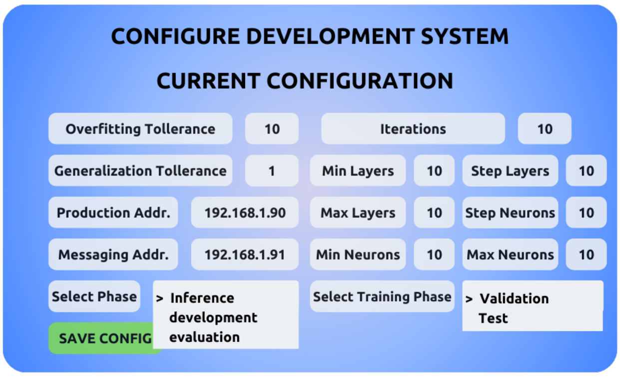


Figure 21: "Configure Development System" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens the ”Configure Development System” form.	1	1	2.50	2.50
2 SYSTEM displays current configuration.				
3 ACTOR sets the `min_layers`.	1	4	2.50	10
4 ACTOR sets the `max_layers`.	1	4	2.50	10
5 ACTOR sets the `min_neurons`.	1	4	2.50	10
6 ACTOR sets the `step_layers`.	1	4	2.50	10
7 ACTOR sets the `step_neurons`.	1	4	2.50	10
8 ACTOR sets the `max_neurons`.	1	4	2.50	10
9 ACTOR sets the `overfitting_tolerance` parameter.	1	4	2.50	10
10 ACTOR sets the `generalization_tolerance` parameter.	1	4	2.50	10
11 ACTOR sets the `iterations` parameter.	1	4	2.50	10
12 ACTOR choose the `training_phase` parameter from the drop down (`validation, test`).	1	1	2.50	2.50
13 ACTOR sets the `production_sys_addr` parameter.	1	1	2.50	2.50
14 ACTOR sets the `messaging_sys_addr` parameter.	1	1	2.50	2.50
15 ACTOR choose the `phase` parameter from the drop down (`inference, develop, evaluation`) .	1	1	2.50	2.50
16.1 SYSTEM IF config is correct and correctly formatted.				
16.1.1 SYSTEM shows a confirmation message.				
16.2 ELSE				
16.2.1 SYSTEM shows error message and aborts.				
17 ACTOR saves the form.	1	1	2.50	2.50
			Human task cost	105

Table 9: Detailed use case for ”Configure Development” task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.4 Evaluation system

3.4.1 Evaluate classifier performance

[*Zahra Omrani*]

This task is performed by a Data Analyst.

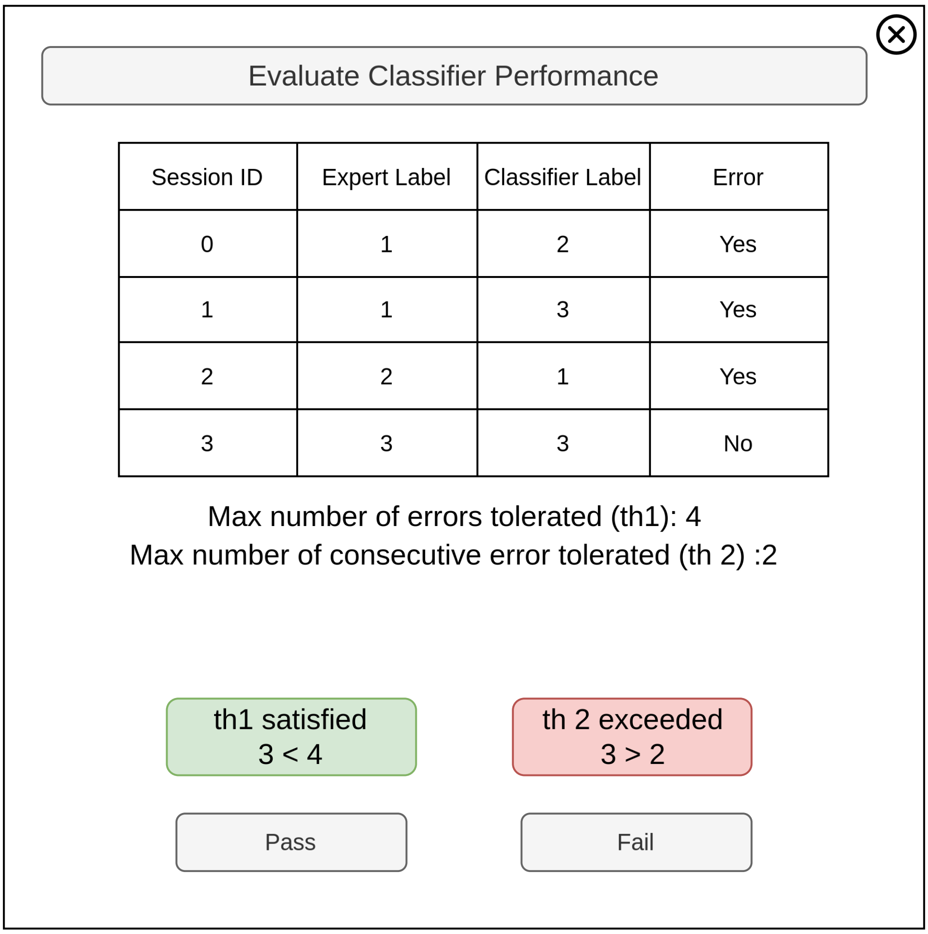


Figure 22: "Evaluate Classifier Performance" mock-up form

Step	O	CL	S	SC
1 ACTOR opens the "Evaluate Classifier Performance" form.	1	1	1.15	1.15
2 SYSTEM displays a table of sessions with Expert Label (ground truth) and Classifier Label (predicted label). The difference between the labels (if any) represents an error.				
3.1 ACTOR checks the total errors threshold color.	1	2	1.15	2.30
3.2 ACTOR checks the consecutive errors threshold color	1	2	1.15	2.30
3.3 IF at least one threshold is red				
3.3.1 ACTOR clicks the "Fail" button.	0.14	1	1.15	0.161
3.4 ELSE				
3.4.1 ACTOR clicks the "Pass" button.	0.86	1	1.15	0.989
4 SYSTEM shows a confirmation dialog.				
5 ACTOR closes the form.	1	1	1.15	1.15
		Human task cost		8.05

Table 10: Detailed use case for "Evaluate Classifier Performance" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.4.2 Configure Evaluation System

[*Francesco Boldrini, Zahra Omrani*]

This task is performed by a ML Engineer.

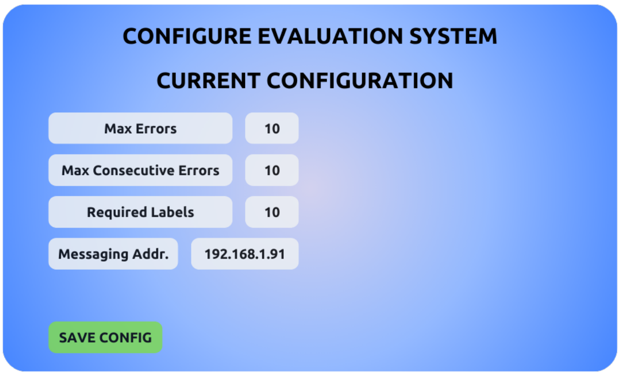


Figure 23: "Configure Evaluation System" mock-up form

Step	O	CL	S	SC
1 ACTOR opens the "Configure Evaluation System" form.	1	1	2.50	2.50
2 SYSTEM displays current configuration.				
3 ACTOR sets the `max_errors` parameter.	1	4	2.50	10
4 ACTOR sets the `max_consecutive_errors` parameter.	1	4	2.50	10
5 ACTOR sets the `required_labels` parameter.	1	4	2.50	10
6 ACTOR sets the `messaging_sys_addr` parameter.	1	1	2.50	2.50
7.1 SYSTEM IF config is correct and correctly formatted.				
7.1.1 SYSTEM shows a confirmation message.				
7.2 ELSE				
7.2.1 SYSTEM shows error message and aborts.				
9 ACTOR saves the form.	1	1	2.50	2.50
	Human task cost			37.50

Table 11: Detailed use case for "Configure Evaluation" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.5 Client-Side Systems

3.5.1 Configure Client-Side Systems

[*Francesco Boldrini, Zahra Omrani*]
This task is performed by a ML Engineer.

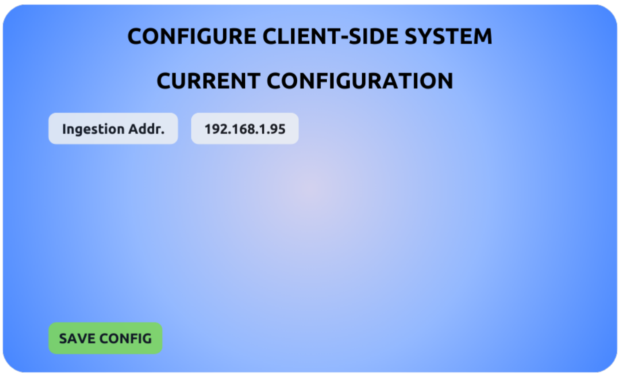


Figure 24: "Configure Client-Side Systems" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens the "Configure Client-Side System" form.	1	1	2.50	2.50
2 SYSTEM displays current configuration.				
3 ACTOR sets the `ingestion_sys_addr` parameter.	1	1	2.50	2.50
4.1 SYSTEM IF config is correct and correctly formatted.				
4.1.1 SYSTEM shows a confirmation message.				
4.2 ELSE				
4.2.1 SYSTEM shows error message and aborts.				
5 ACTOR saves the form.	1	1	2.50	2.50
	Human task cost			7.50

Table 12: Detailed use case for "Configure Client-Side Systems" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.6 Production System

3.6.1 Configure Production Systems

[*Francesco Boldrini, Zahra Omrani*]

This task is performed by a ML Engineer.

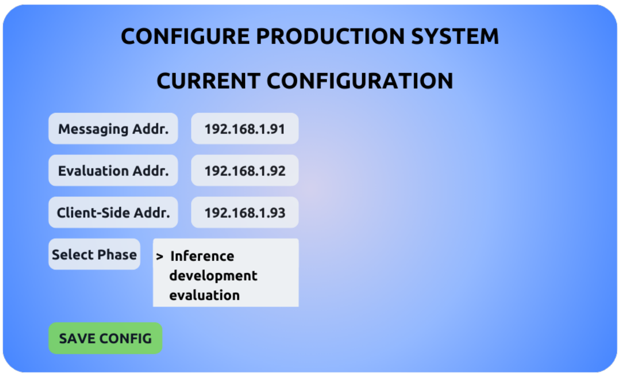


Figure 25: "Configure Production System" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens the "Configure Production System" form.	1	1	2.50	2.50
2 SYSTEM displays current configuration.				
3 ACTOR sets the `production_sys_addr` parameter.	1	1	2.50	2.50
4 ACTOR sets the `messaging_sys_addr` parameter.	1	1	2.50	2.50
5 ACTOR choose the `phase` parameter from the drop down (`inference, develop, evaluation`).	1	1	2.50	2.50
6.1 SYSTEM IF config is correct and correctly formatted.				
6.1.1 SYSTEM shows a confirmation message.				
6.2 ELSE				
6.2.1 SYSTEM shows error message and aborts.				
7 ACTOR saves the form.	1	1	2.50	2.50
	Human task cost			12.50

Table 13: Detailed use case for "Configure Production" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

3.7 Ingestion System

3.7.1 Configure Ingestion System

[*Francesco Boldrini, Zahra Omrani*]
This task is performed by a ML Engineer.

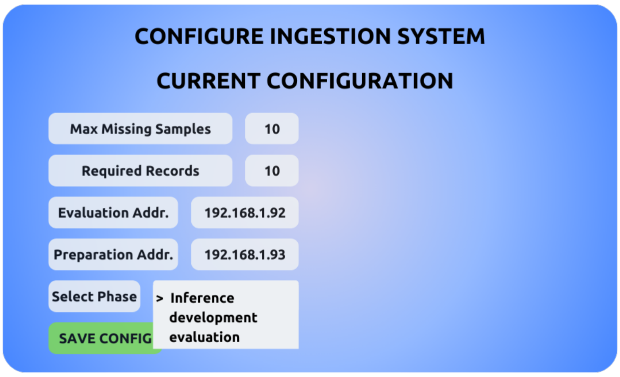


Figure 26: "Configure Ingestion System" mock-up form

Step	**O**	**CL**	**S**	**SC**
1 ACTOR opens the "Configure Ingestion System" form.	1	1	2.50	2.50
2 SYSTEM displays the current configuration.				
3 ACTOR sets the `max_missing_samples`.	1	4	2.50	10
4 ACTOR sets the `required_records`.	1	4	2.50	10
5 ACTOR sets the `evaluation_sys_addr`.	1	1	2.50	2.50
6 ACTOR sets the `preparation_sys_addr`.	1	1	2.50	2.50
7 ACTOR selects the `phase` from the dropdown (`inference`, `development`, `evaluation`).	1	1	2.50	2.50
8.1 SYSTEM IF the configurations are correct and properly formatted:				
8.1.1 SYSTEM displays a confirmation message.				
8.2 ELSE (if the configurations are incorrect):				
8.2.1 SYSTEM displays an error message and aborts the process.				
9 ACTOR saves the form.	1	1	2.50	2.50
Human task cost				32.5

Table 14: Detailed use case for "Configure Ingestion System" task

3.8 Preparation System

3.8.1 Configure Preparation System

[*Francesco Boldrini, Zahra Omrani*]

This task is performed by a ML Engineer.

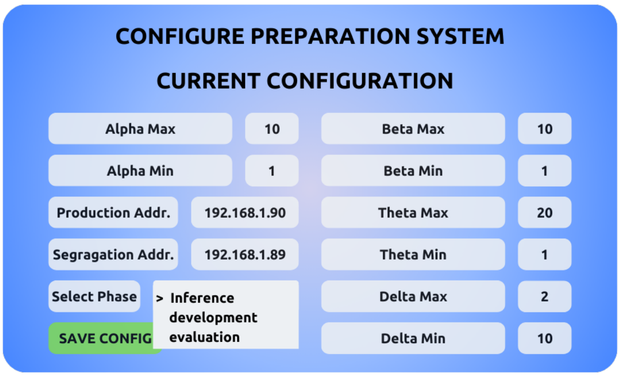


Figure 27: "Configure Preparation System" mock-up form

Step	O	CL	S	SC
1 ACTOR opens the "Configure Preparation System" form.	1	1	2.50	2.50
2 SYSTEM displays the current configuration.				
3 ACTOR sets the `alpha_max`.	1	4	2.50	10
4 ACTOR sets the `alpha_min`.	1	4	2.50	10
5 ACTOR sets the `beta_max`.	1	4	2.50	10
6 ACTOR sets the `beta_min`.	1	4	2.50	10
7 ACTOR sets the `theta_max`.	1	4	2.50	10
8 ACTOR sets the `theta_min`.	1	4	2.50	10
9 ACTOR sets the `delta_max`.	1	4	2.50	10
10 ACTOR sets the `delta_min`.	1	4	2.50	10
11 ACTOR sets the `production_sys_addr`.	1	1	2.50	2.50
12 ACTOR sets the `segregation_sys_addr`.	1	1	2.50	2.50
13 ACTOR selects the `phase` from the dropdown (`inference`, `develop`, `evaluation`).	1	1	2.50	2.50
14 SYSTEM IF the configuration is correct and properly formatted:				
14.1 SYSTEM displays a confirmation message.				
14.2 ELSE (if the configuration is incorrect):				
14.2.1 SYSTEM displays an error message and aborts the process.				
15 ACTOR saves the form.	1	1	2.50	2.50
Human task cost				92.5

Table 15: Detailed use case for "Configure Preparation System" task

4 Simulation

4.1 Collapsed workflow

[*Ettore Ricci, Paolo Palumbo, Francesco Boldrini*]

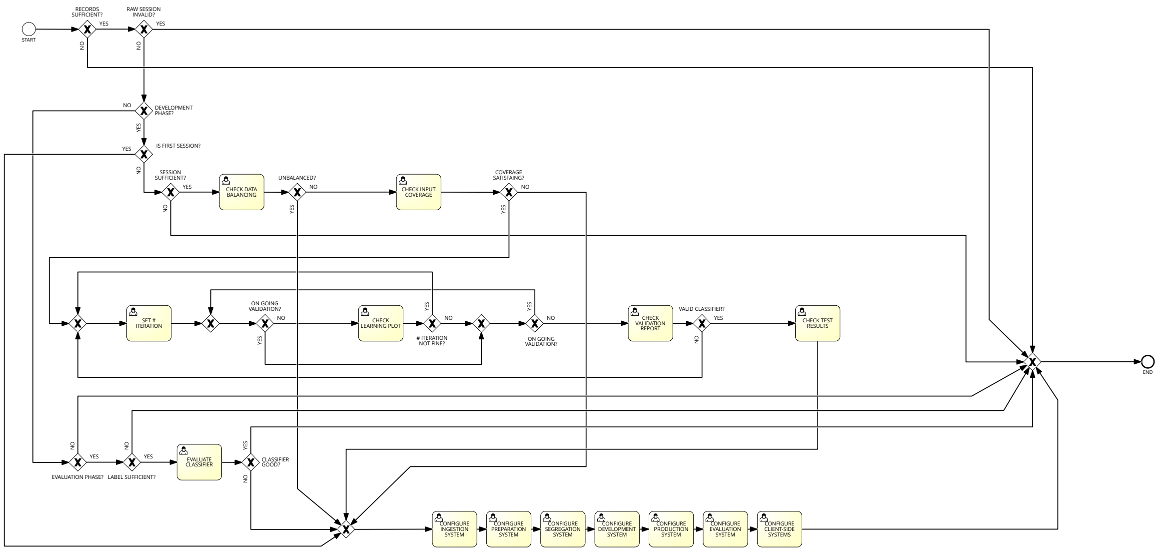


Figure 28: Collapsed workflow

4.2 AS-IS Simulation

[*Francesco Boldrini, Zahra Omrani*]

NB: we set the total number of initial process instances to 6852, as with our assumptions for the two initial gates, where we discard 10% of the sessions twice, we need 6852 sessions to start, to work with the documentation's assumptions of the 5550 good sessions for all phases altogether.

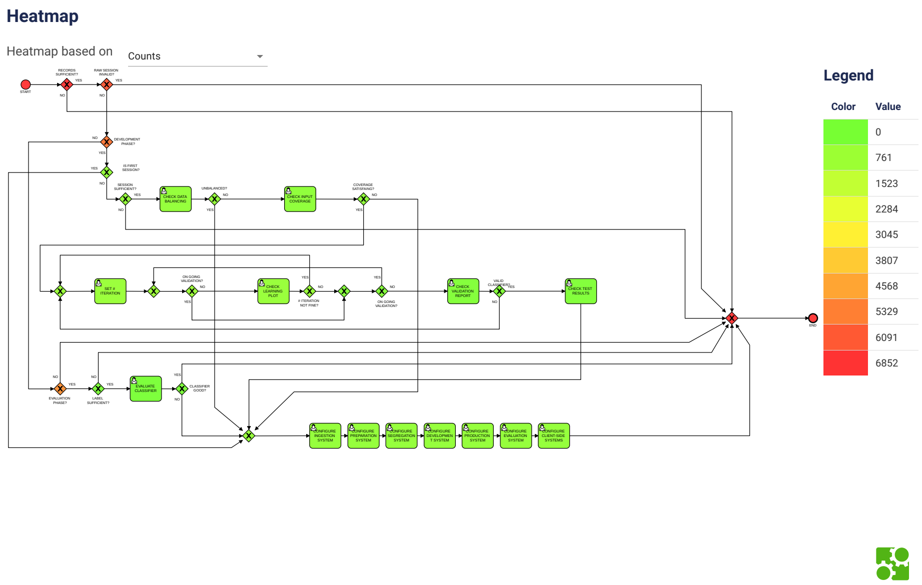


Figure 29: AS-IS Heatmap of the counts of the parameters

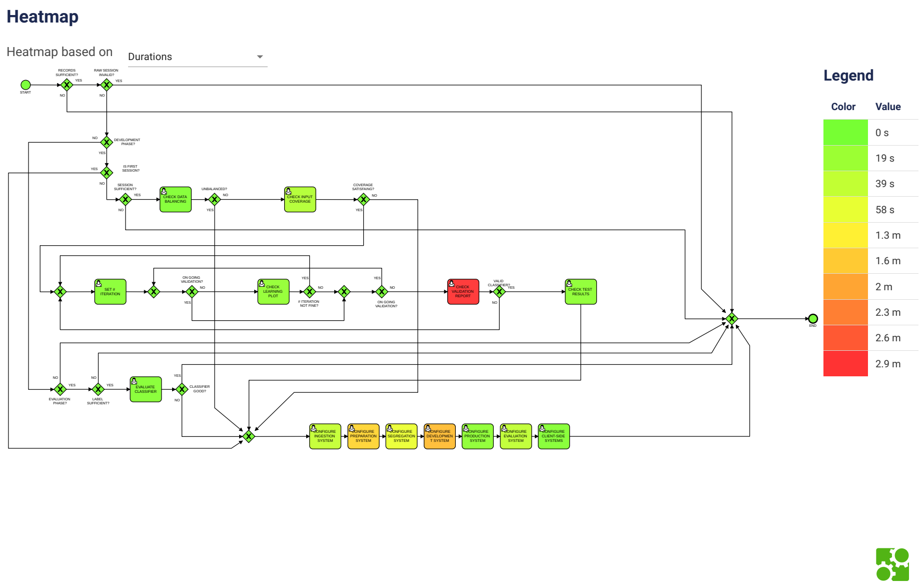


Figure 30: AS-IS Heatmap of the time spent in each passage [Durations]

Scenario Statistics

	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	0 seconds	20.3 minutes	36.9 seconds
Process instance cycle times excluding off-timetable hours	0 seconds	20.3 minutes	36.9 seconds
Process instance costs	0 EUR	0 EUR	0 EUR

Activity Durations, Costs, Waiting times, Deviations from Thresholds

Name		Waiting time			Duration			Duration over threshold			Cost			Cost over threshold		
	Count	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
CHECK DATA BALANCING	528	0 s	0 s	0 s	5.5 s	5.7 s	6 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK INPUT COVERAGE	434	0 s	0 s	0 s	29.5 s	31 s	32.6 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK LEARNING PLOT	604	0 s	0 s	0 s	14.3 s	15 s	15.7 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK TEST RESULTS	150	0 s	0 s	0 s	11.9 s	12.5 s	13.1 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK VALIDATION REPORT	159	0 s	0 s	0 s	2.8 m	2.9 m	3.1 m	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE CLIENT-SIDE SYSTEMS	537	0 s	0 s	0 s	7.1 s	7.5 s	7.9 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE DEVELOPMENT SYSTEM	537	0 s	0 s	0 s	1.7 m	1.7 m	1.8 m	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE EVALUATION SYSTEM	537	0 s	0 s	0 s	35.6 s	37.6 s	39.4 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE INGESTION SYSTEM	537	0 s	0 s	0 s	38 s	40 s	42 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE PREPARATION SYSTEM	537	0 s	0 s	0 s	1.5 m	1.5 m	1.6 m	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE PRODUCTION SYSTEM	537	0 s	0 s	0 s	11.9 s	12.5 s	13.1 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE SEGREGATION SYSTEM	537	0 s	0 s	0 s	57 s	59.9 s	1 m	0 s	0 s	0 s	0	0	0	0	0	0
EVALUATE CLASSIFIER	29	0 s	0 s	0 s	7.7 s	8.1 s	8.4 s	0 s	0 s	0 s	0	0	0	0	0	0
SET # ITERATION	658	0 s	0 s	0 s	9.5 s	10 s	10.5 s	0 s	0 s	0 s	0	0	0	0	0	0

Figure 31: AS-IS Simulation Results

Figure 32: AS-IS Scenario Statistics

Parameter	% of the Gate	Motivation
# Iteration not fine?	20%	According to the assumptions of the documentation, we set the % of iters that are not fine to 20%
Classifier Good?	86%	According to the assumptions of the docs, classifiers are good 86 of the time
Coverage satisfying?	33%	According to the assumptions of the docs, coverage is satisfying 33% of the time
Development Phase?	9%	In this gate, out of 5550, 500 are in the development phase
Is First Session?	1%	In this gate, out of 500, 1 is in the first session, but the gate won't load less than 1% on BIMP
Labels sufficient?	71%	Given 5550 good sessions, and assuming that those will yields 5 proper classifiers, given the probabilities at the preceding gates and the rounding because of necessity of the gate to have at least 1% as value, we need to have 71% of the sessions to have sufficient labels to respect the documentation's assumptions.
On going Validation?	90%	We set validation to 90% as most of the times this step involves the autonomous systems and not humans
Raw session Invalid?	10%	We assume that 90% of the times the raw session is valid.
Records Sufficient?	90%	We assume that 90% of the times the records are sufficient.
Session Sufficient?	99%	We set sessions sufficient to 99% as with the document's assumptions, we would need roughly 545 sessions to have 5 final good classifiers and here we already start with 500, which is already lower than what we would need.
Unbalanced?	20%	The documentation assumes that 20% of the classes are balanced
Valid Classifier?	95%	The documentation assumes that 95% of the classifiers are valid

4.3 Modeling the TO-BE Process

[*Francesco Boldrini, Zahra Omrani*]

In the context of our application, "Emotion Based Music Selection", we suppose that during the initial configuration phase, we acquire the data through the ECG sensors and the user's schedules, currently playing music and other relevant informations.

This data could be processed by a research center through data-mining approaches: this would mean simplifying the process and making it more efficient, as we could use existing similar classifiers to initialize ours, rather than starting from scratch.

In fact similar networks and classifiers may work well with similar parameters over similar tasks.

For each category it is possible to define some improvement(s):

1. Hand-Off level Improvement(s): Re-use sessions from similar networks in the same category, rather than collecting other sessions.

2. Service Level Improvement(s): Use hyperparameters from similar networks in the same category, rather than starting from scratch.

3. Task Level Improvement(s): Reduce the cognitive effort necessary for the configurations, by starting from default parameters obtained from other similar networks, rather than starting from scratch.

4.3.1 Hand-Off level Improvement(s)

[*Francesco Boldrini, Zahra Omrani*]

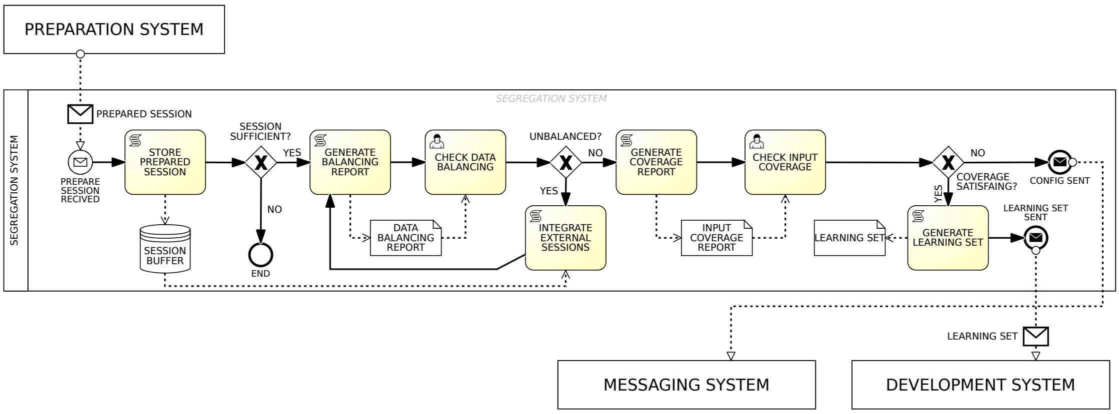


Figure 33: Change to the Generate Learning Sets

We modified the workflow in such a way that our saved previous sessions can be reused in the case of unbalanced data, rather than awaiting for the message system to respond to the issue arisen in the workflow.

This cuts on necessary times to respond to this erroneous situation, as the system can autonomously respond to the issue, rather than waiting for a human to intervene, thus improving the system's efficiency and re-use of data.

4.3.2 Service Level Improvement(s)

[*Francesco Boldrini, Zahra Omrani*]

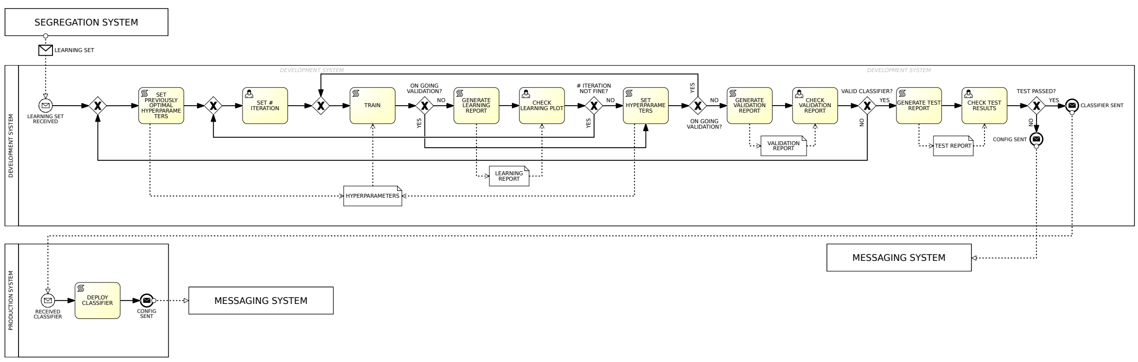


Figure 34: Change to the Develop Classifier

The possibility of using hyperparameters from similar trained networks in the same category, rather than starting from scratch, is a great improvement in the service level.

The search for optimized parameters in the network no longer involves brute-forcing the optimization through a grid-search approach but rather re-uses a functioning network's parameters, saving time and computational resources.

4.3.3 Task Level Improvement(s)

[*Francesco Boldrini, Zahra Omrani*]

The task level improvement(s) involve reducing the cognitive effort necessary for the configurations, by starting from default parameters obtained from other similar networks, rather than starting from scratch.

In particular, we removed the need for a grid search in the check validation report, as we no longer need to check amongst the 5 best networks, but simply have to verify that the network respects the overfitting tolerance, as we are using parameters from a similar network in the same category.

Step	O	CL	S	SC
1 ACTOR opens "Check validation report" form.	1	1	2.5	2.5
2 SYSTEM shows the model trained on optimal parameters				
3 ACTOR calculates model Validation Loss minus the Training Loss	1	3	2.5	7.5
4.1 IF 3 is less than the Overfitting Tolerance	0.95	3	2.5	7.125
3.1.1 THEN ACTOR Confirms the selected model.	0.95	3	2.5	7.125
3.2 ELSE ACTOR Rejects the selected model.	0.05	3	2.5	0.375
6 SYSTEM shows a confirmation dialog.				
7 ACTOR closes the form.	1	1	2.5	2.5
			Human task cost	$27.125 < 175.5$

Table 16: Detailed use case for "Check validation report" task
O - Occurrence, CL - Cognitive Level, S - Normalized Salary, SC - Step Cost

Furthermore, by using suggested parameters in the configuration phase, we can reduce the cognitive level necessary for the configurations to 2 from 4, by starting from default parameters obtained from other similar networks, rather than needing extensive evaluations (level 4 cognitive level) to find the optimal parameters.

Step	O	CL	S	SC
1 ACTOR opens the "Configure Preparation System" form.	1	1	2.50	2.50
2 SYSTEM displays the current configuration.				
3 ACTOR sets the `alpha_max`.	1	2	2.50	5
4 ACTOR sets the `alpha_min`.	1	2	2.50	5
5 ACTOR sets the `beta_max`.	1	2	2.50	5
6 ACTOR sets the `beta_min`.	1	2	2.50	5
7 ACTOR sets the `theta_max`.	1	2	2.50	5
8 ACTOR sets the `theta_min`.	1	2	2.50	5
9 ACTOR sets the `delta_max`.	1	2	2.50	5
10 ACTOR sets the `delta_min`.	1	2	2.50	5
11 ACTOR sets the `production_sys_addr`.	1	1	2.50	2.50
12 ACTOR sets the `segregation_sys_addr`.	1	1	2.50	2.50
13 ACTOR selects the `phase` from the dropdown (`inference`, `develop`, `evaluation`).	1	1	2.50	2.50
14 SYSTEM IF the configuration is correct and properly formatted:				
14.1 SYSTEM displays a confirmation message.				
14.2 ELSE (if the configuration is incorrect):				
14.2.1 SYSTEM displays an error message and aborts the process.				
15 ACTOR saves the form.	1	1	2.50	2.50
			Human task cost	$52.5 < 92.5$

Table 17: TO-BE detailed use case for "Configure Preparation System" task

4.4 TO-BE Simulation

[Francesco Boldrini, Zahra Omrani]

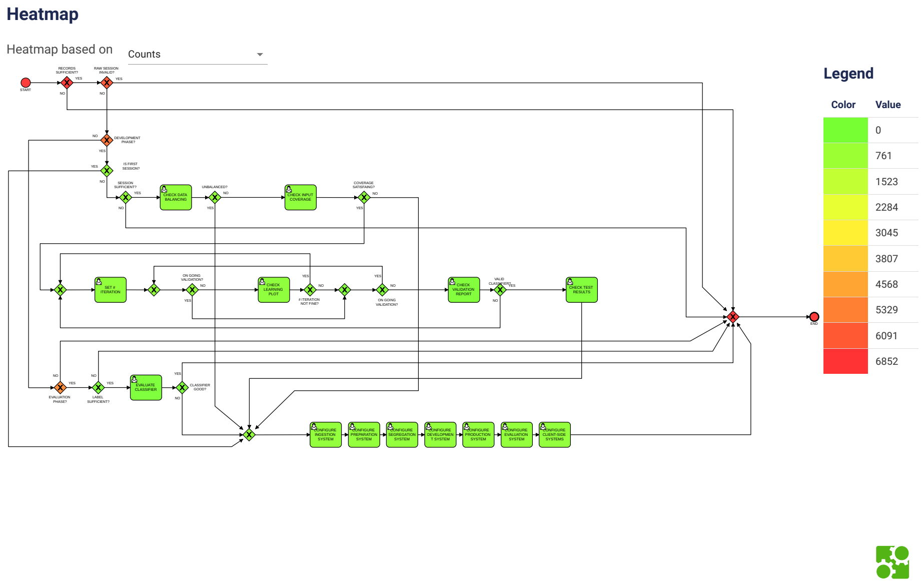


Figure 35: TO-BE Heatmap of the counts of the parameters

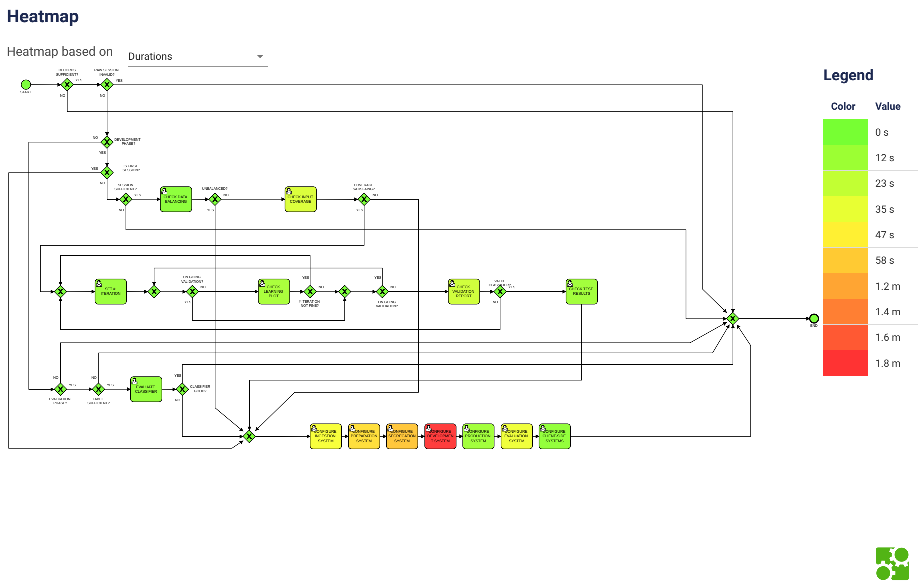
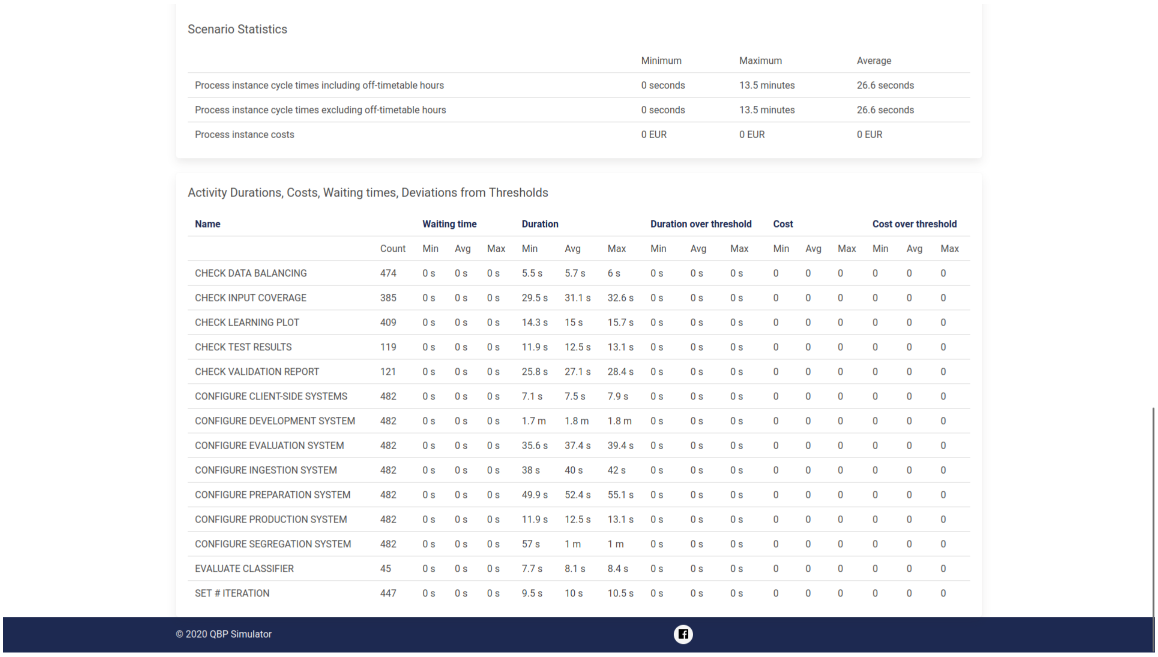


Figure 36: TO-BE Heatmap of the time spent in each passage [Durations]



Scenario Statistics

	Minimum	Maximum	Average
Process instance cycle times including off-timetable hours	0 seconds	13.5 minutes	26.6 seconds
Process instance cycle times excluding off-timetable hours	0 seconds	13.5 minutes	26.6 seconds
Process instance costs	0 EUR	0 EUR	0 EUR

Activity Durations, Costs, Waiting times, Deviations from Thresholds

Name		Waiting time			Duration			Duration over threshold			Cost			Cost over threshold		
	Count	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
CHECK DATA BALANCING	474	0 s	0 s	0 s	5.5 s	5.7 s	6 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK INPUT COVERAGE	385	0 s	0 s	0 s	29.5 s	31.1 s	32.6 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK LEARNING PLOT	409	0 s	0 s	0 s	14.3 s	15 s	15.7 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK TEST RESULTS	119	0 s	0 s	0 s	11.9 s	12.5 s	13.1 s	0 s	0 s	0 s	0	0	0	0	0	0
CHECK VALIDATION REPORT	121	0 s	0 s	0 s	25.8 s	27.1 s	28.4 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE CLIENT-SIDE SYSTEMS	482	0 s	0 s	0 s	7.1 s	7.5 s	7.9 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE DEVELOPMENT SYSTEM	482	0 s	0 s	0 s	1.7 m	1.8 m	1.8 m	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE EVALUATION SYSTEM	482	0 s	0 s	0 s	35.6 s	37.4 s	39.4 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE INGESTION SYSTEM	482	0 s	0 s	0 s	38 s	40 s	42 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE PREPARATION SYSTEM	482	0 s	0 s	0 s	49.9 s	52.4 s	55.1 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE PRODUCTION SYSTEM	482	0 s	0 s	0 s	11.9 s	12.5 s	13.1 s	0 s	0 s	0 s	0	0	0	0	0	0
CONFIGURE SEGREGATION SYSTEM	482	0 s	0 s	0 s	57 s	1 m	1 m	0 s	0 s	0 s	0	0	0	0	0	0
EVALUATE CLASSIFIER	45	0 s	0 s	0 s	7.7 s	8.1 s	8.4 s	0 s	0 s	0 s	0	0	0	0	0	0
SET # ITERATION	447	0 s	0 s	0 s	9.5 s	10 s	10.5 s	0 s	0 s	0 s	0	0	0	0	0	0

© 2020 QBP Simulator

Figure 37: TO-BE Simulation Results

Figure 38: TO-BE Scenario Statistics

5 Process mining

[*Ettore Ricci, Paolo Palumbo*]

We mined the logs generated by the simulation of the collapsed workflow.

We modified the simulation configuration to make the 100 tokens flow through every path of the workflow. The most important gateways that we changed are listed in the following table.

Gateway	Yes	No
RAW SESSION INVALID	5%	95%
RECORD SUFFICIENT	95%	5%
SESSION SUFFICIENT	95%	5%
IS FIRST SESSION	20%	80%
COVERAGE SATISFYING	70%	30%
DEVELOPMENT PHASE	70%	30%

Table 18: Gateways configuration

5.1 Transaction mining

[*Ettore Ricci, Paolo Palumbo*]

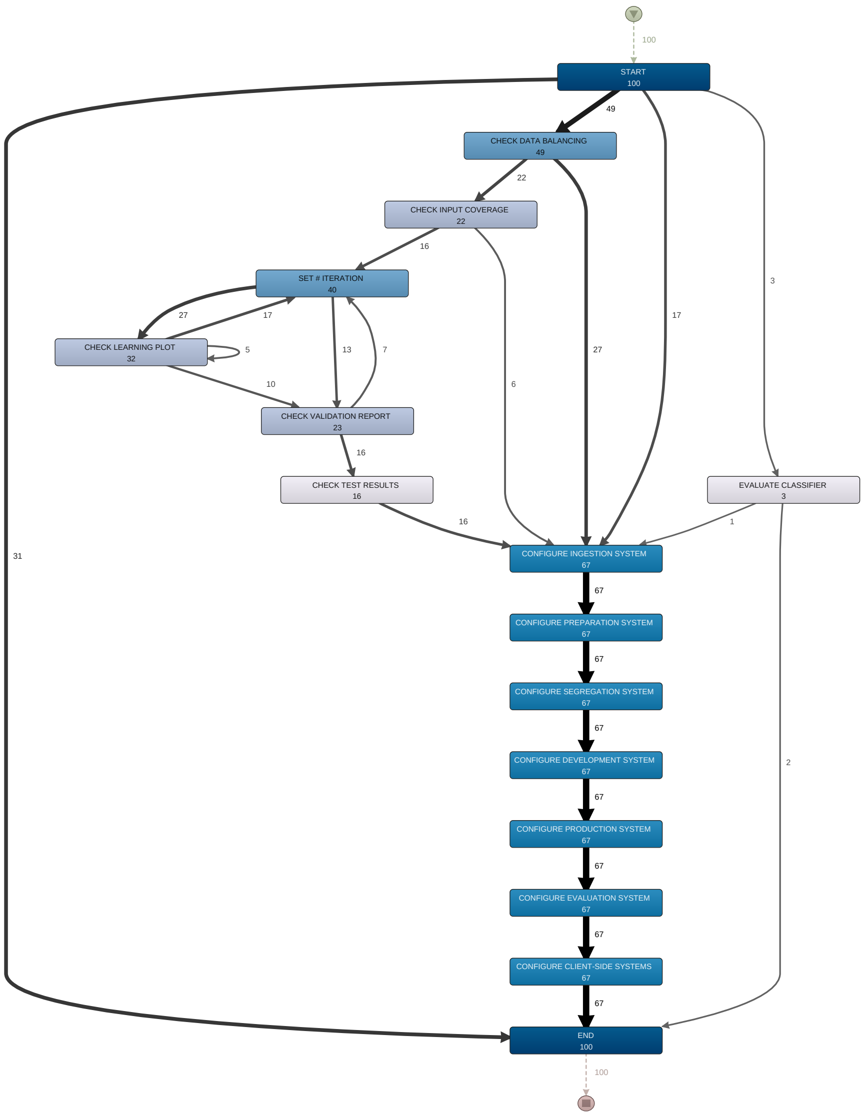


Figure 39: Disco analysis

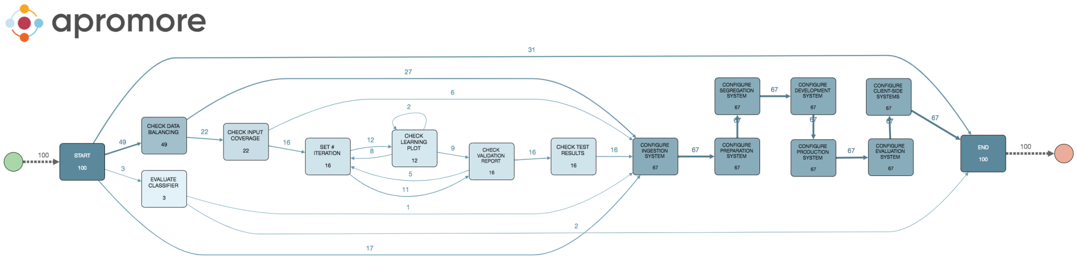


Figure 40: Apromore analysis

As we can see, the two transition maps mined from Disco and from Apromore are identical. The only difference stays in the frequencies because in Disco the frequencies are calculated as the total number of times a transition is executed, even on the same token; while in Apromore the frequencies are calculated as the number of individual tokens that execute a transition. This behavior can be changed with a setting in both tools.

5.2 BPMN mining

[*Ettore Ricci, Paolo Palumbo*]

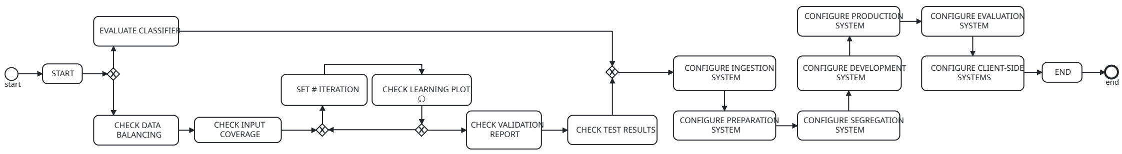


Figure 41: ProM mined BPMN model

We mined the logs using the "Heuristics Miner ProM6" mining algorithm.

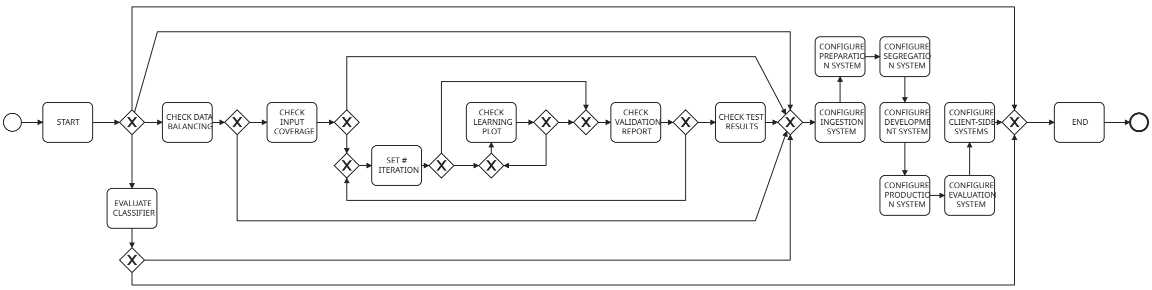


Figure 42: Apromore mined BPMN model

The BPMN model mined from Apromore is more detailed and covers more cases than the one mined from ProM. The key differences between the ProM model and the Apromore one are that the ProM model is missing the paths that skip the training and the configuration as well as one of the two paths that skip only the training. Furthermore, the training loop is much simpler in the ProM model, as it is missing every path that restarts the training after "CHECK VALIDATION REPORT".

5.3 Conformance checking

[*Ettore Ricci, Paolo Palumbo*]

Tool	Fitness	Generalization	Precision	Simplicity
Apromore	0.9928	0.9837	0.8199	62
ProM	0.7313	0.9902	0.8653	39

Table 19: Comparison of the process mining tools

5.4 Violations

[*Ettore Ricci, Paolo Palumbo*]

We modified the logs to introduce 3 violations in the workflow. The violations are the following:

1. Skipping the dataset creation (”CHECK DATA BALANCING” and ”CHECK INPUT COVERAGE”) using data from another user.
2. Skipping ”SET # ITERATIONS” and ”CHECK LEARNING PLOT” by using early stopping.
3. Skipping ”CHECK DATA BALANCING” by using a resampling technique.

Each violation is introduced 3 times in the logs.
These 3 violations can be beneficial in terms of time and resources:

- The first violation can make the costs of the training significantly lower for the client, because using an old dataset allows us to skip the labeling of the new data and it usually is very expensive. Also the manual check of the dataset is skipped saving additional time and resources. It must be noted that this violation can be a problem for the privacy of the clients and also result in worse models if the data of the new user has different characteristics from the old one.
- The second violation can make the training faster, because we do not need anymore to check the learning plot manually and we can train each model only once instead of trying multiple times with different number of iterations. Also, the method previously used to determine the number of iterations was based on an heuristic and it can be prone to errors.
- The third violation can reduce the time and costs of the dataset creation, also making the training possible with unbalanced datasets.

CaseID	Violation	Fitness ProM	Fitness Apromore
10	1	0.91	0.87
20	1	0.85	0.84
47	1	0.86	0.86
53	2	0.91	0.93
63	2	0.84	0.82
88	2	0.91	0.93
6	3	0.91	0.93
72	3	0.91	0.85
81	3	0.94	0.87

Table 20: Cases, violations and fitness on models generated by ProM and Apromore

10	START
10	CHECK DATA BALANCING
10	CHECK INPUT COVERAGE
10	SET # ITERATION
10	CHECK LEARNING PLOT
10	CHECK VALIDATION REPORT
10	CHECK TEST RESULTS
10	CONFIGURE INGESTION SYSTEM
10	CONFIGURE PREPARATION SYSTEM
10	CONFIGURE SEGREGATION SYSTEM
10	CONFIGURE DEVELOPMENT SYSTEM
10	CONFIGURE PRODUCTION SYSTEM
10	CONFIGURE EVALUATION SYSTEM
10	CONFIGURE CLIENT-SIDE SYSTEMS
10	END

20	START
20	CHECK DATA BALANCING
20	CHECK INPUT COVERAGE
20	SET # ITERATION
20	CHECK LEARNING PLOT
20	CHECK VALIDATION REPORT
20	SET # ITERATION
20	CHECK LEARNING PLOT
20	SET # ITERATION
20	CHECK VALIDATION REPORT
20	CHECK TEST RESULTS
20	CONFIGURE INGESTION SYSTEM
20	CONFIGURE PREPARATION SYSTEM
20	CONFIGURE SEGREGATION SYSTEM
20	CONFIGURE DEVELOPMENT SYSTEM
20	CONFIGURE PRODUCTION SYSTEM
20	CONFIGURE EVALUATION SYSTEM
20	CONFIGURE CLIENT-SIDE SYSTEMS
20	END

47	START
47	CHECK DATA BALANCING
47	CHECK INPUT COVERAGE
47	SET # ITERATION
47	CHECK VALIDATION REPORT
47	CHECK TEST RESULTS
47	CONFIGURE INGESTION SYSTEM
47	CONFIGURE PREPARATION SYSTEM
47	CONFIGURE SEGREGATION SYSTEM
47	CONFIGURE DEVELOPMENT SYSTEM
47	CONFIGURE PRODUCTION SYSTEM
47	CONFIGURE EVALUATION SYSTEM
47	CONFIGURE CLIENT-SIDE SYSTEMS
47	END

53	START
53	CHECK DATA BALANCING
53	CHECK INPUT COVERAGE
53	SET # ITERATION
53	CHECK LEARNING PLOT
53	SET # ITERATION
53	CHECK LEARNING PLOT
53	CHECK VALIDATION REPORT
53	CHECK TEST RESULTS
53	CONFIGURE INGESTION SYSTEM
53	CONFIGURE PREPARATION SYSTEM
53	CONFIGURE SEGREGATION SYSTEM
53	CONFIGURE DEVELOPMENT SYSTEM
53	CONFIGURE PRODUCTION SYSTEM
53	CONFIGURE EVALUATION SYSTEM
53	CONFIGURE CLIENT-SIDE SYSTEMS
53	END

63	START
63	CHECK DATA BALANCING
63	CHECK INPUT COVERAGE
63	SET # ITERATION
63	CHECK VALIDATION REPORT
63	SET # ITERATION
63	CHECK LEARNING PLOT
63	CHECK LEARNING PLOT
63	SET # ITERATION
63	CHECK LEARNING PLOT
63	CHECK VALIDATION REPORT
63	SET # ITERATION
63	CHECK LEARNING PLOT
63	SET # ITERATION
63	CHECK LEARNING PLOT
63	CHECK LEARNING PLOT
63	SET # ITERATION
63	CHECK LEARNING PLOT
63	SET # ITERATION
63	CHECK VALIDATION REPORT
63	CHECK TEST RESULTS
63	CONFIGURE INGESTION SYSTEM
63	CONFIGURE PREPARATION SYSTEM
63	CONFIGURE SEGREGATION SYSTEM
63	CONFIGURE DEVELOPMENT SYSTEM
63	CONFIGURE PRODUCTION SYSTEM
63	CONFIGURE EVALUATION SYSTEM
63	CONFIGURE CLIENT-SIDE SYSTEMS
63	END

88	START
88	CHECK DATA BALANCING
88	CHECK INPUT COVERAGE
88	SET # ITERATION
88	CHECK LEARNING PLOT
88	SET # ITERATION
88	CHECK LEARNING PLOT
88	SET # ITERATION
88	CHECK LEARNING PLOT
88	SET # ITERATION
88	CHECK VALIDATION REPORT
88	CHECK TEST RESULTS
88	CONFIGURE INGESTION SYSTEM
88	CONFIGURE PREPARATION SYSTEM
88	CONFIGURE SEGREGATION SYSTEM
88	CONFIGURE DEVELOPMENT SYSTEM
88	CONFIGURE PRODUCTION SYSTEM
88	CONFIGURE EVALUATION SYSTEM
88	CONFIGURE CLIENT-SIDE SYSTEMS
88	END

6	START
6	CHECK DATA BALANCING
6	CHECK INPUT COVERAGE
6	SET # ITERATION
6	CHECK VALIDATION REPORT
6	CHECK TEST RESULTS
6	CONFIGURE INGESTION SYSTEM
6	CONFIGURE PREPARATION SYSTEM
6	CONFIGURE SEGREGATION SYSTEM
6	CONFIGURE DEVELOPMENT SYSTEM
6	CONFIGURE PRODUCTION SYSTEM
6	CONFIGURE EVALUATION SYSTEM
6	CONFIGURE CLIENT-SIDE SYSTEMS
6	END

72	START
72	CHECK DATA BALANCING
72	CHECK INPUT COVERAGE
72	SET # ITERATION
72	CHECK LEARNING PLOT
72	CHECK LEARNING PLOT
72	CHECK LEARNING PLOT
72	SET # ITERATION
72	CHECK LEARNING PLOT
72	SET # ITERATION
72	CHECK LEARNING PLOT
72	CHECK VALIDATION REPORT
72	SET # ITERATION
72	CHECK LEARNING PLOT
72	CHECK LEARNING PLOT
72	SET # ITERATION
72	CHECK VALIDATION REPORT
72	CHECK TEST RESULTS
72	CONFIGURE INGESTION SYSTEM
72	CONFIGURE PREPARATION SYSTEM
72	CONFIGURE SEGREGATION SYSTEM
72	CONFIGURE DEVELOPMENT SYSTEM
72	CONFIGURE PRODUCTION SYSTEM
72	CONFIGURE EVALUATION SYSTEM
72	CONFIGURE CLIENT-SIDE SYSTEMS
72	END

81	START
81	CHECK DATA BALANCING
81	CHECK INPUT COVERAGE
81	SET # ITERATION
81	CHECK LEARNING PLOT
81	CHECK VALIDATION REPORT
81	SET # ITERATION
81	CHECK LEARNING PLOT
81	SET # ITERATION
81	CHECK LEARNING PLOT
81	SET # ITERATION
81	CHECK LEARNING PLOT
81	CHECK VALIDATION REPORT
81	CHECK TEST RESULTS
81	CONFIGURE INGESTION SYSTEM
81	CONFIGURE PREPARATION SYSTEM
81	CONFIGURE SEGREGATION SYSTEM
81	CONFIGURE DEVELOPMENT SYSTEM
81	CONFIGURE PRODUCTION SYSTEM
81	CONFIGURE EVALUATION SYSTEM
81	CONFIGURE CLIENT-SIDE SYSTEMS
81	END

Figure 43: Screenshots of the logs while introducing the violations. The highlighted activities are the ones removed from the logs.

Tool	Fitness
Apromore	0.9875
ProM	0.7256

Table 21: New fitness with violations included in the logs

Figure 44: Violations in the Apromore model visualized with ProM

Figure 45: Violations in the ProM model visualized with ProM

5.4.1 Transaction mining with violations

[*Ettore Ricci, Paolo Palumbo*]

Mining the logs with the violations included, we get these results:

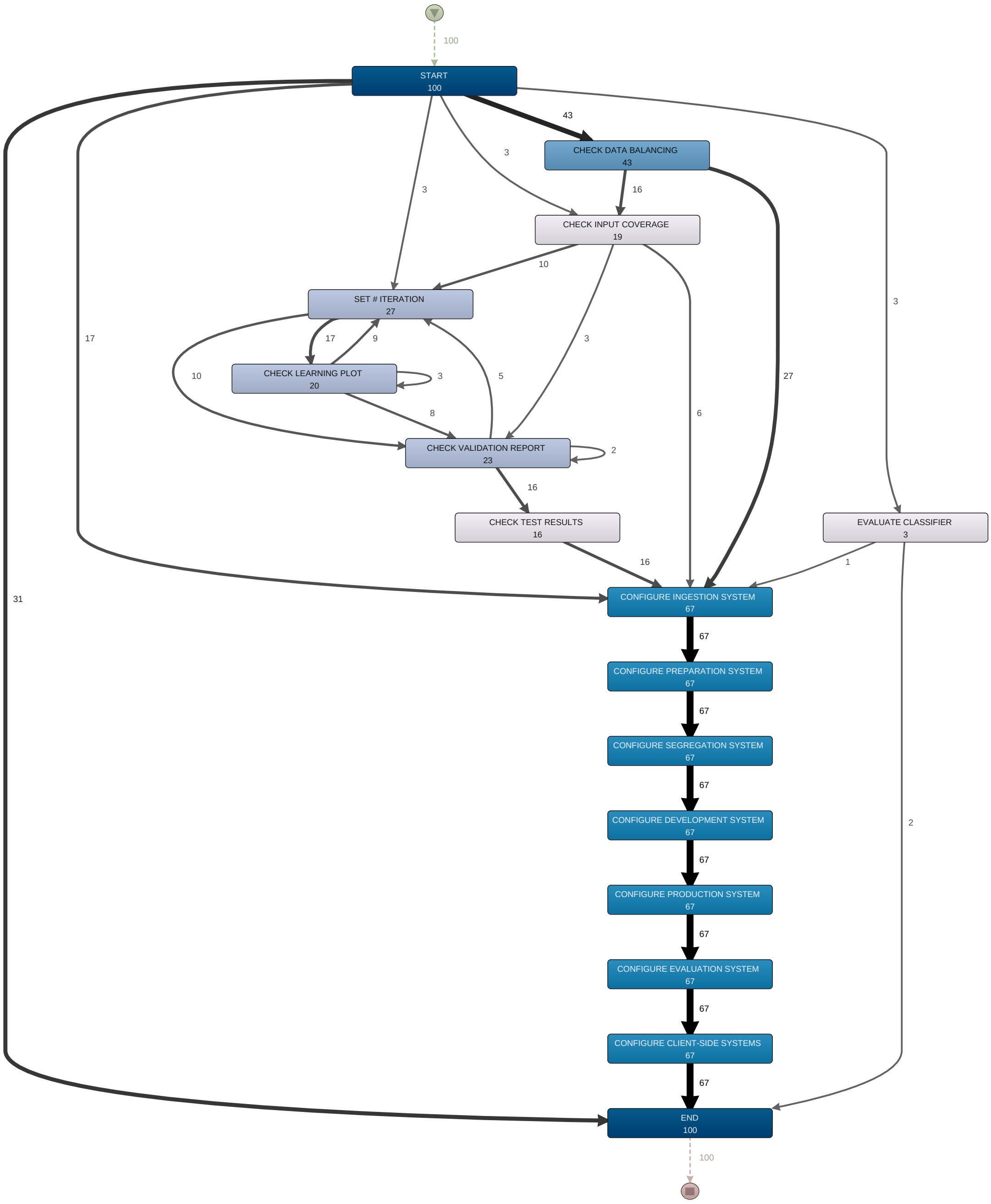


Figure 46: Disco transition map mined with violations

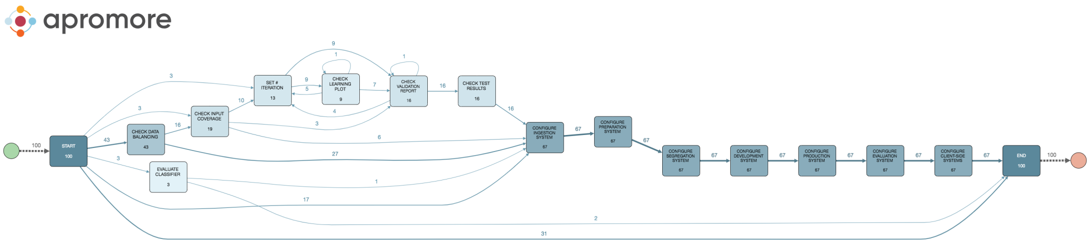


Figure 47: Apromore transition map mined with violations

5.4.2 BPMN mining with violations

[*Ettore Ricci, Paolo Palumbo*]

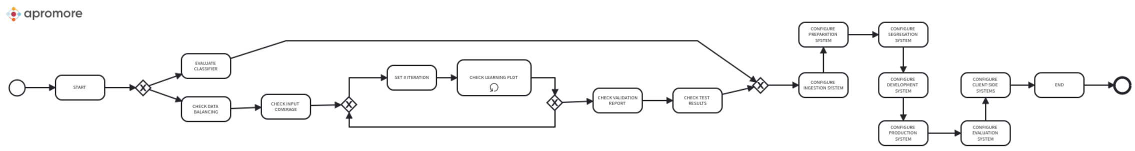


Figure 48: ProM mined BPMN model with violations

As we can see, the BPMN model mined from ProM with the violations included does not change at all from the one without the violations.

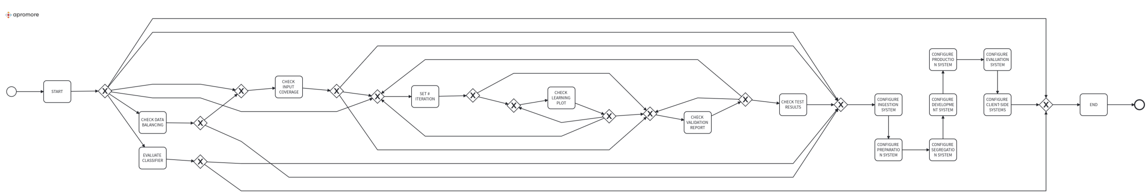


Figure 49: Apromore mined BPMN model with violations

On the other hand, the BPMN model mined from Apromore with the violations included changes according to the violations, ultimately having a higher fitness.

5.4.3 Conformance checking with violations

[*Ettore Ricci, Paolo Palumbo*]

Tool	Fitness	Generalization	Precision	Simplicity
Apromore	1	0.9780	0.6742	69
ProM	0.7256	0.9909	0.8941	39

Table 22: Comparison of the process mining tools with violations

As expected, the fitness of the ProM mined model is the same as the one calculated with the same log, on the old model. Because the ProM model is much simpler than the Apromore one, its Generalization and Precision are higher while the Simplicity is lower. The Apromore model got more complex because of the violations, making its Generalization and Precision lower than the old model, also the Simplicity is a bit higher. The Apromore model, however, has a perfect fitness, because it is able to capture all the possible paths of the workflow, even with the violations.

Because the ProM model did not change, we won't include its cases as they are the same as the ones from the old model.

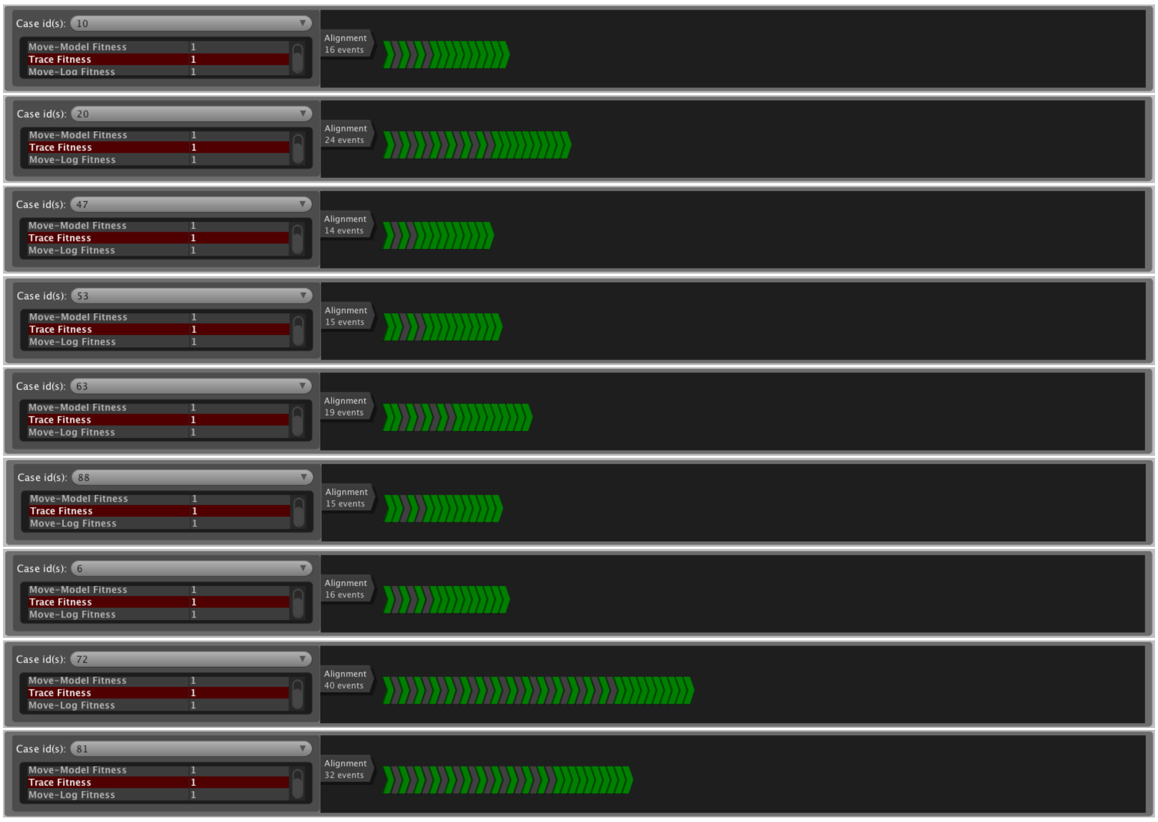


Figure 50: Violations in the new Apromore model visualized with ProM