Towards Explainable Test Case Prioritisation with Learning-to-Rank Models

Aurora Ramírez[1], Mario Berrios[1], José Raúl Romero[1], Robert Feldt[2]
[1] University of Córdoba, Spain
[2] Chalmers University of Technology, Sweden

3RD INT. WORKSHOP ON ARTIFICIAL INTELLIGENCE IN SOFTWARE TESTING – ICST

DUBLIN, 20/04/2023

Content

1. Introduction
2. Explainability needs in TCP
3. Prelimiary results
4. Discussion

Machine learning for TCP

- Historical analysis of test case properties, previous runs, SUT characteristics…
- Recent interest in ML for TCP under different learning paradigms: supervised learning, reinforcement learning and learning-to-rank
- These techniques are usually "black-box" (sacrifice for higher accuracy)

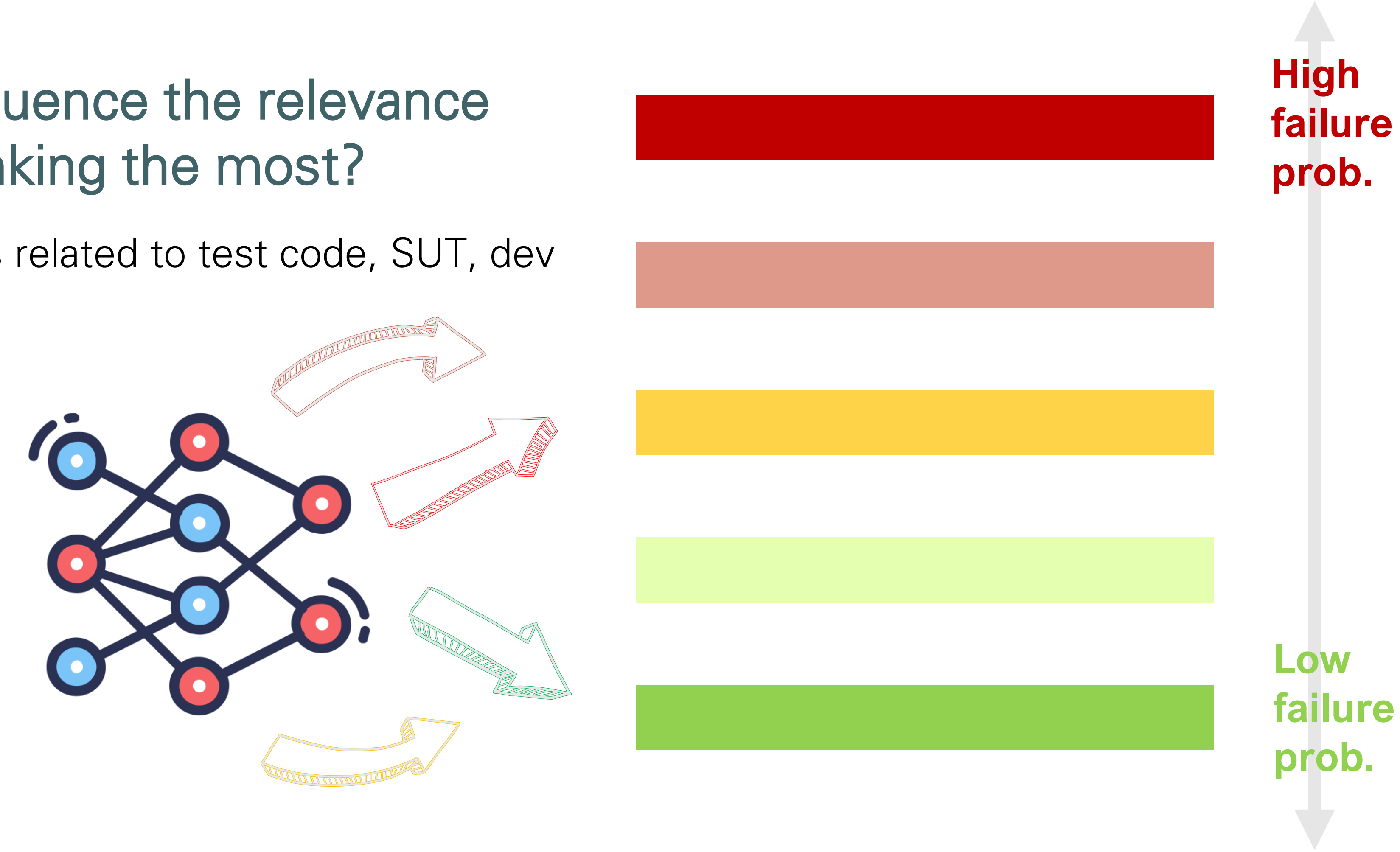
Explainable artificial intelligence (XAI)

- Understanding how ML models work → *global explanations*
- Understanding why particular predictions are returned → *local explanations*
- Understanding predictions with different outcomes → *contrastive explanations*
- Understanding how predictions could change → *counterfactual explanations (what-if)*

Explaining TCP for one build

1A. Which features influence the relevance prediction of a ranking the most?

Global explanation | > 100 features related to test code, SUT, dev



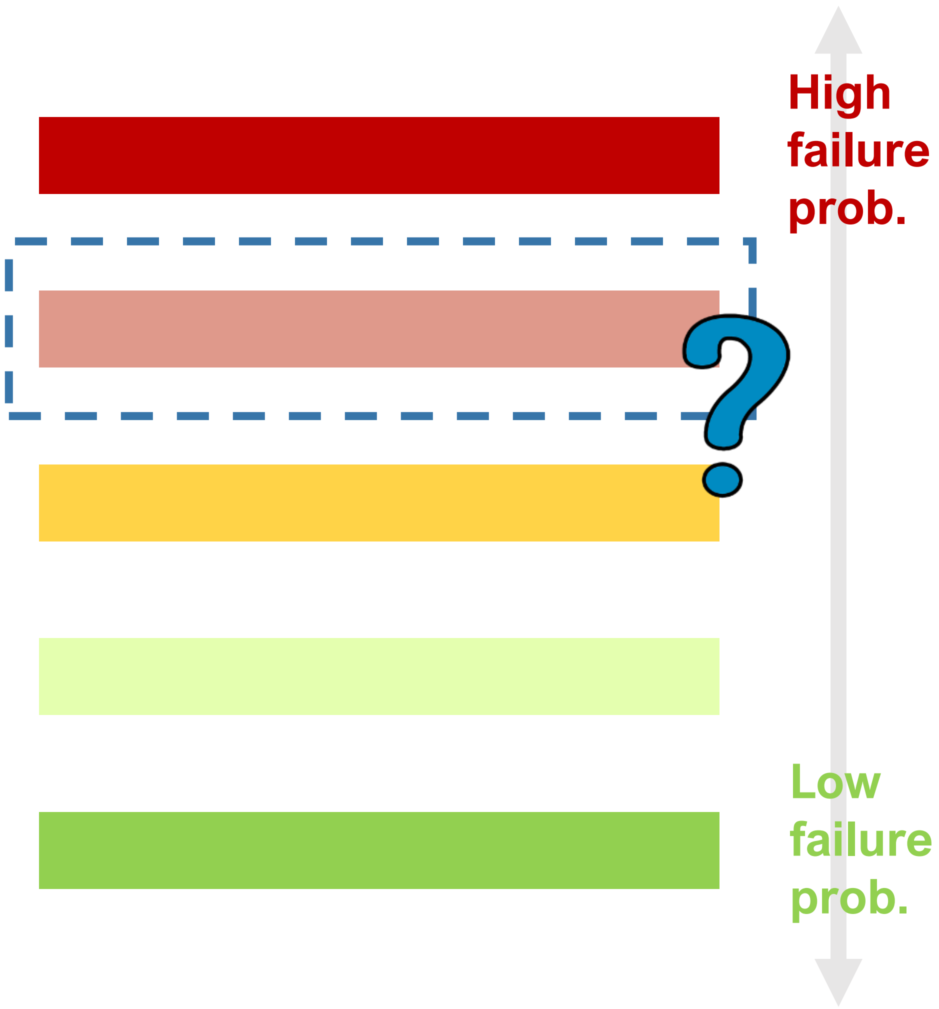
Explaining TCP for one build

1A. Which features influence the relevance prediction of a ranking the most?

Global explanation | > 100 features related to test code, SUT, dev

1B. Why is a test case placed in a certain position in the ranking?

Local explanations | Feature contribution of top-ranked test cases



Explaining TCP for one build

1A. Which features influence the relevance prediction of a ranking the most?

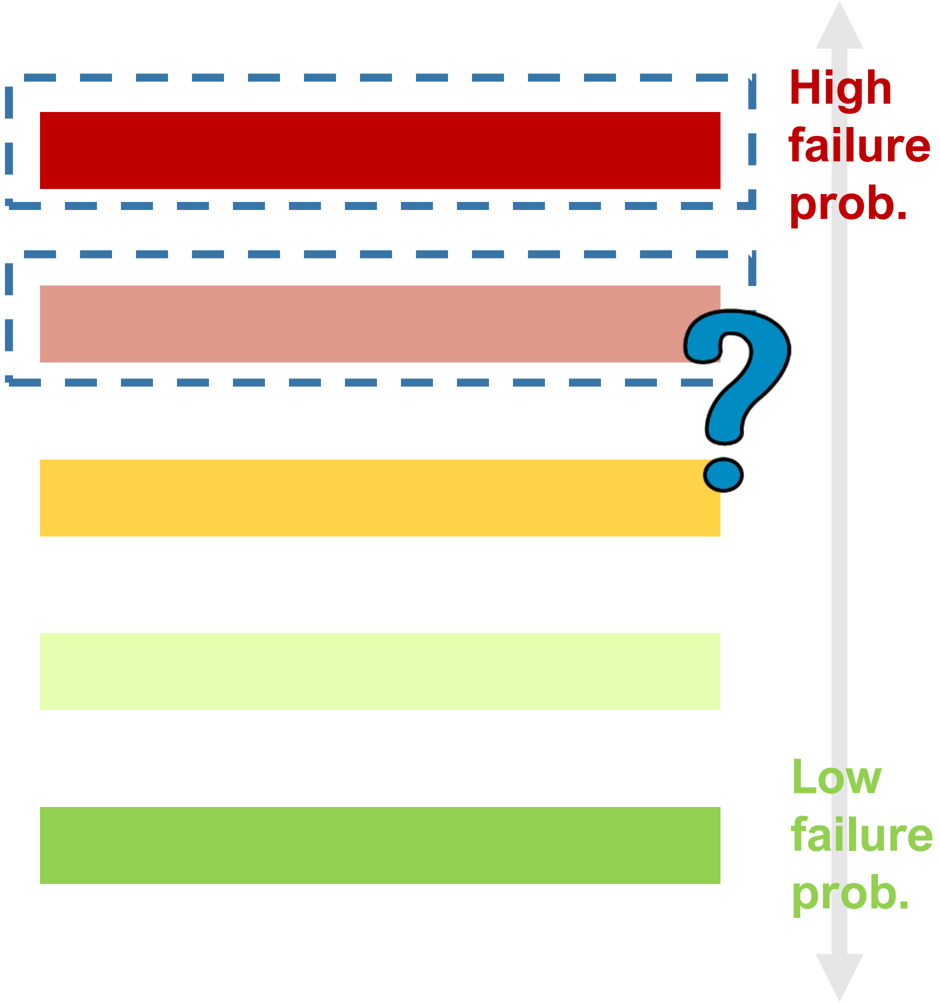
Global explanation | > 100 features related to test code, SUT, dev

1B. Why is a test case placed in a certain position in the ranking?

Local explanations | Feature contribution of top-ranked test cases

1C. Why is a test case ranked above another test case?

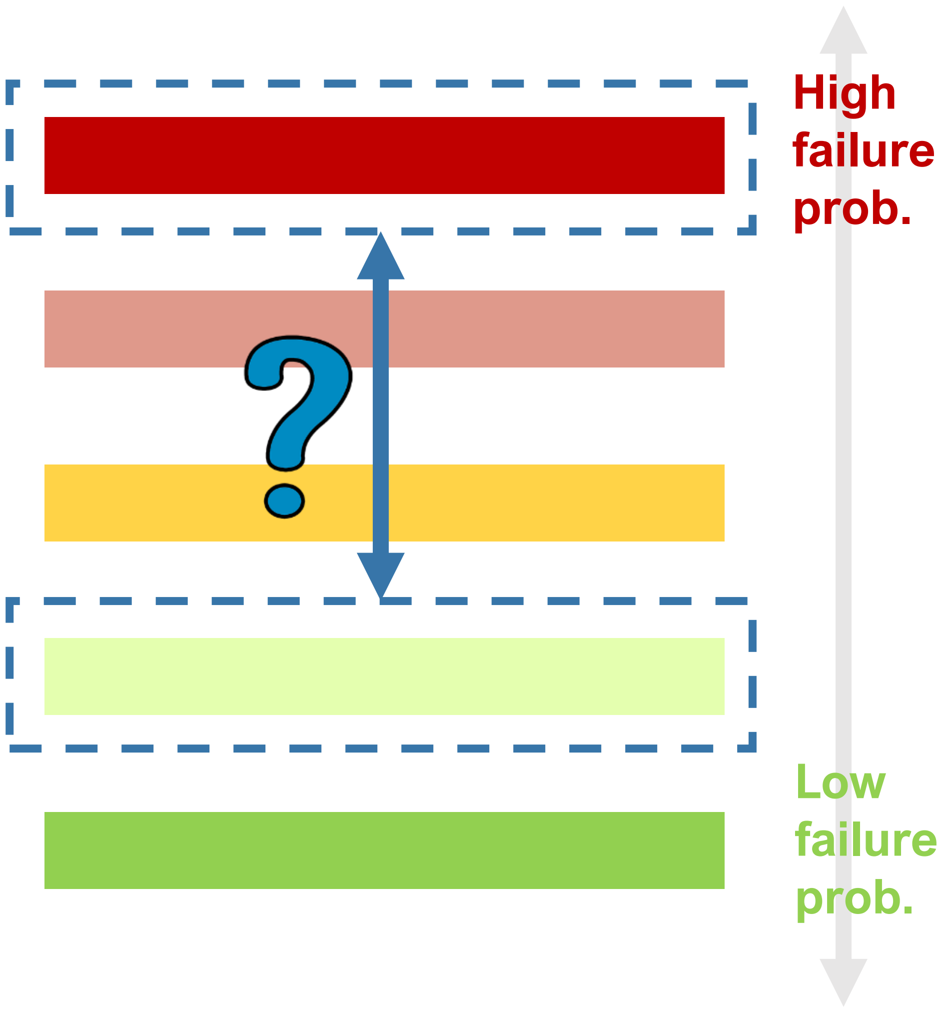
Comparison of local explanations | Are relative positions relevant?



Explaining TCP for one build

1D. Why is one test case likely to fail and another test case to pass?

Contrastive explanations | Consider positional distance between failing/pass test cases



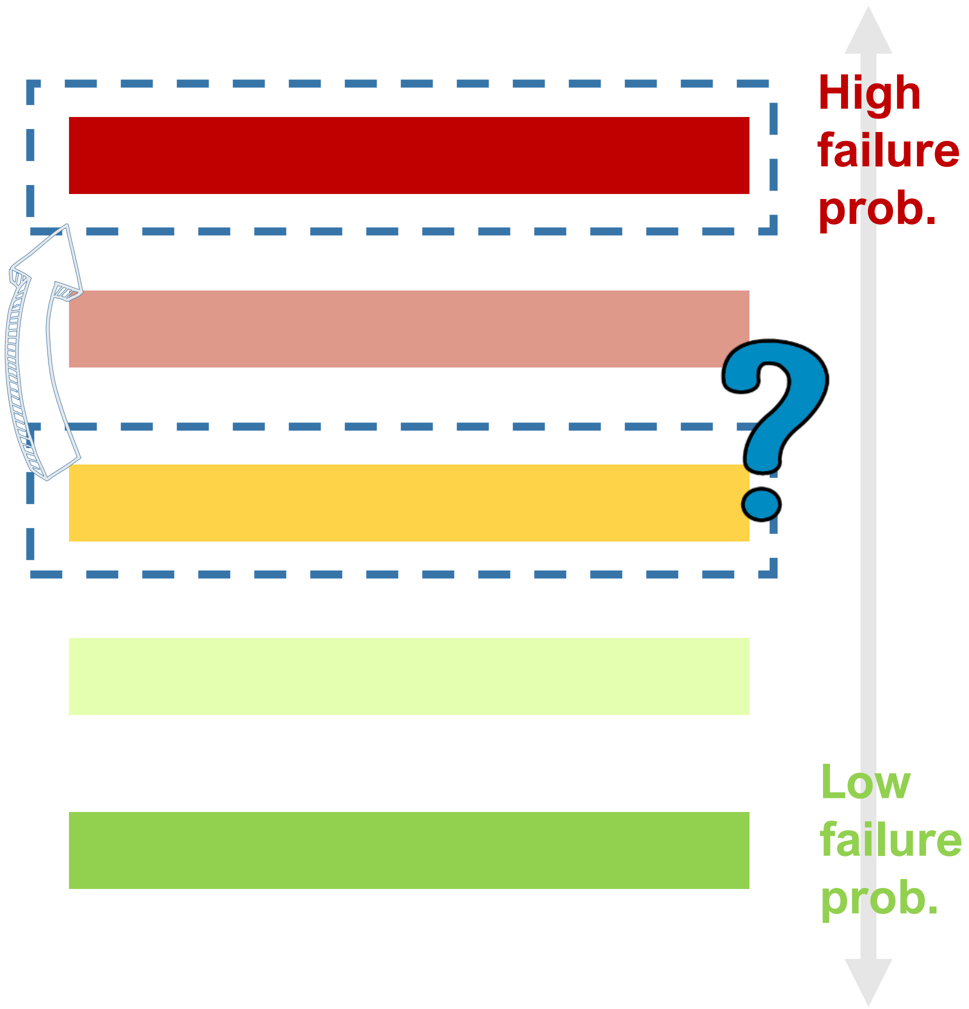
Explaining TCP for one build

1D. Why is one test case likely to fail and another test case to pass?

Contrastive explanations | Consider positional distance between failing/pass test cases

1E. What properties would need to be different to place a test case higher in the ranking?

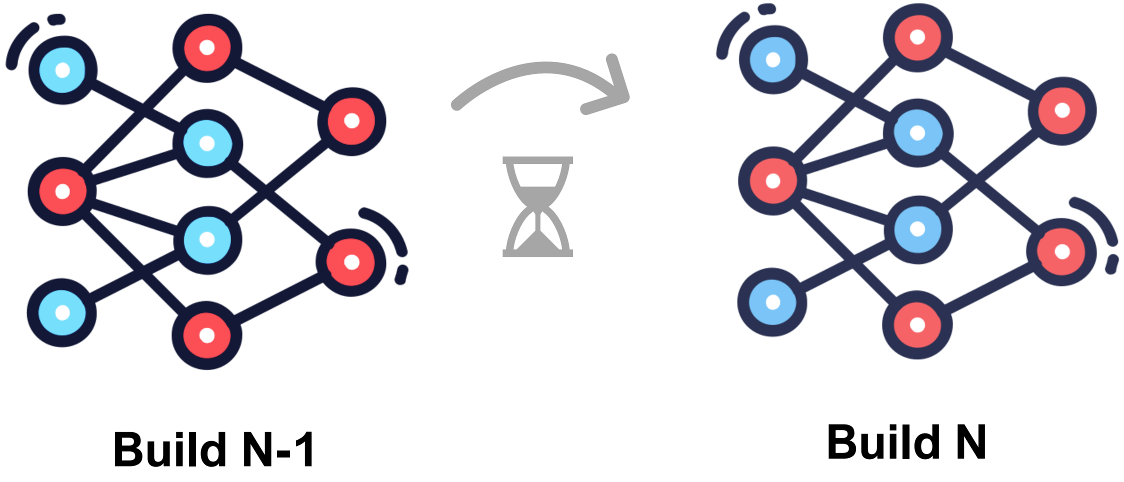
Counterfactual explanations | Features and values that increase test case relevance in the ranking



Explaining TCP across builds

2A. How does the contribution of features to LTR models vary over time?

Global explanations | Evolution of feature contributions as models are retrained



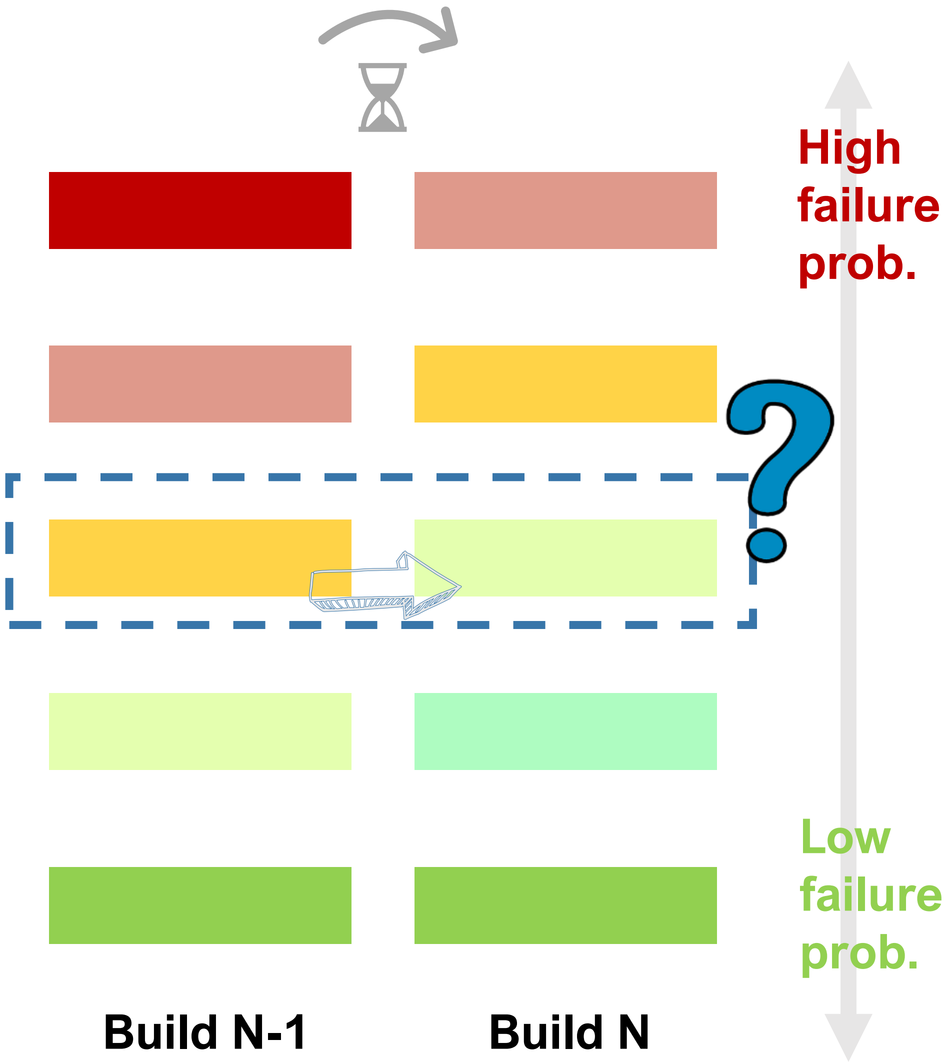
Explaining TCP across builds

2A. How does the contribution of features to LTR models vary over time?

Global explanations | Evolution of feature contributions as models are retrained

2B. Why did a test case fail on build T-1 and not in build T?

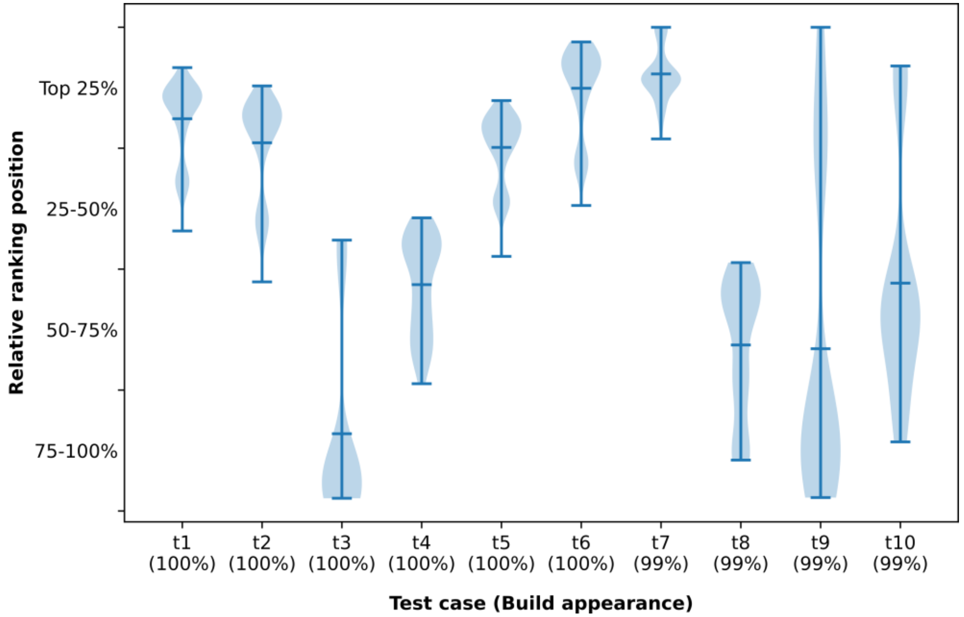
Local explanations | Variation of feature contributions for a particular test case in consecutive builds

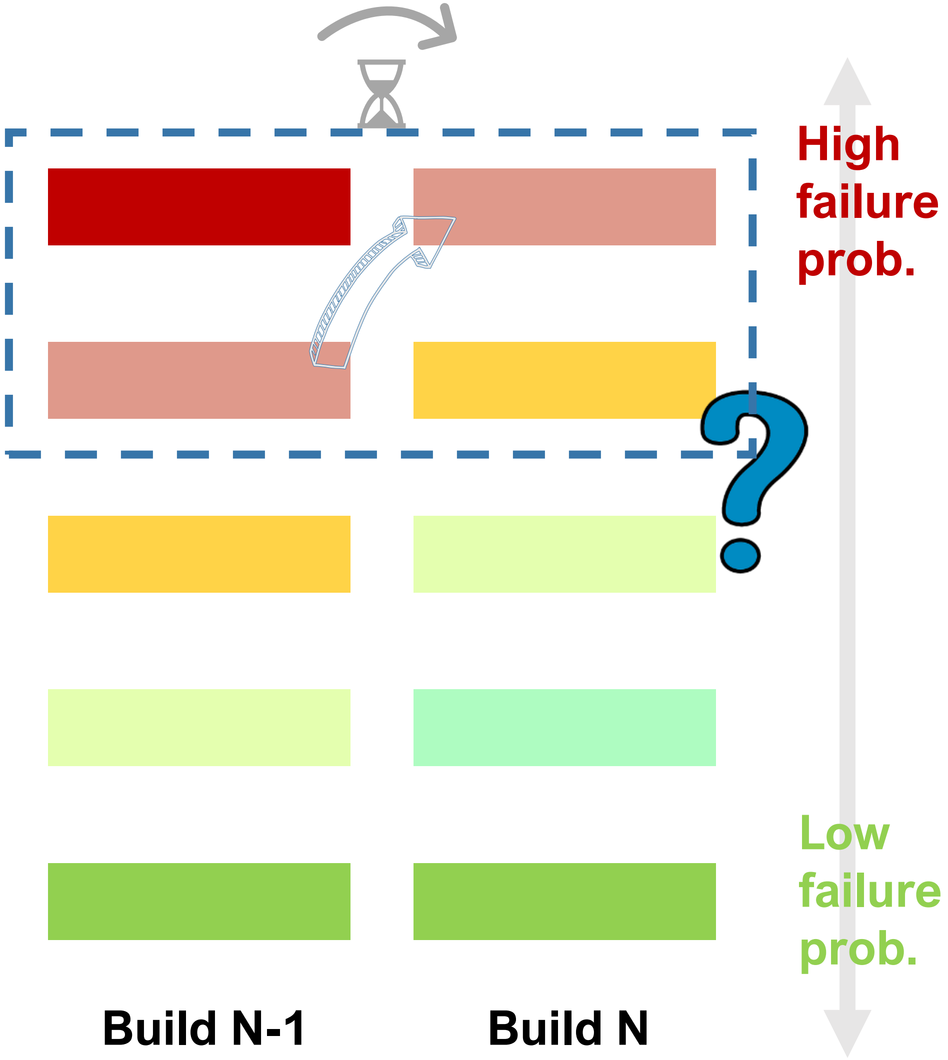


Explaining TCP across builds

2C. Why was a test case ranked in different positions in the builds T-1 and T?

Local explanations | Dependence to changes in the test suite





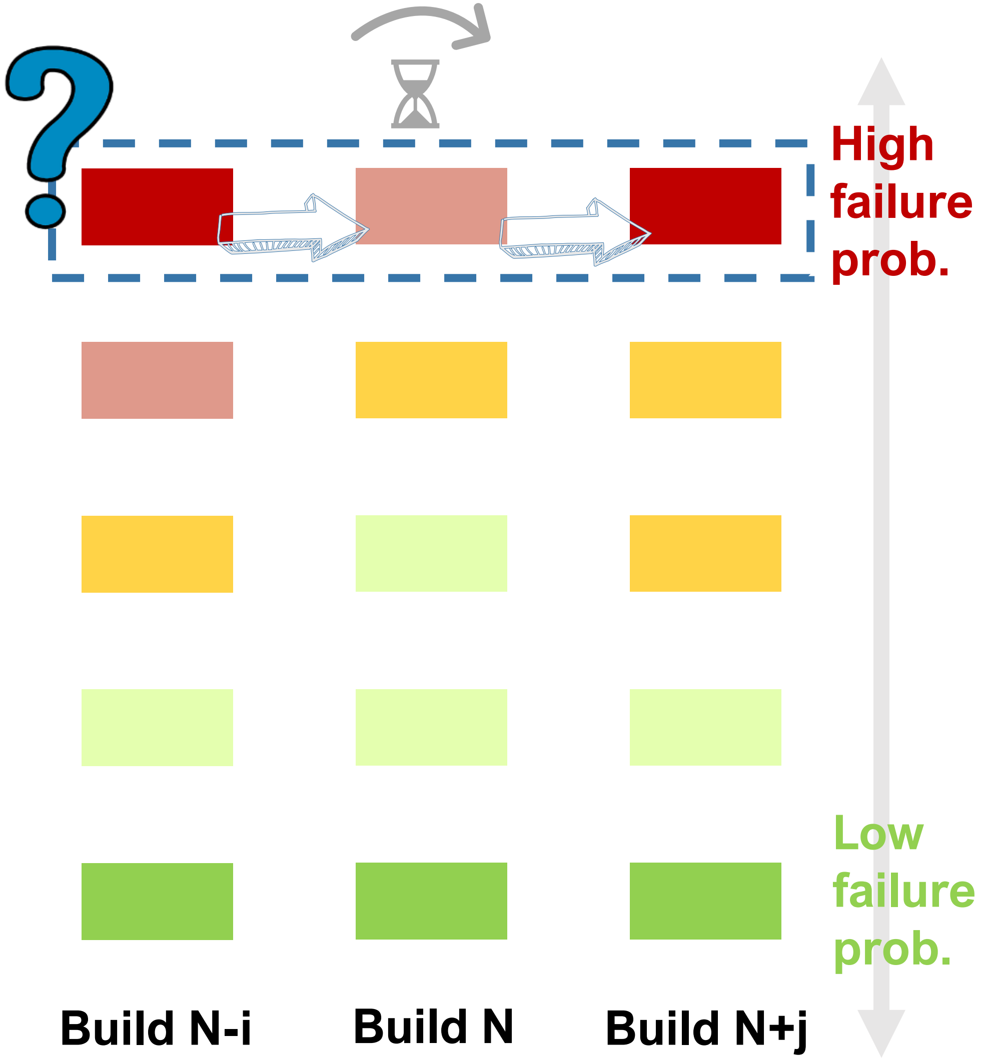
Explaining TCP across builds

2C. Why was a test case ranked in different positions in the builds T-1 and T?

Local explanations | Dependence to changes in the test suite

2D. How do feature contributions vary between different builds for a given test case?

Local explanations | Variation of feature contributions for a particular test case along its execution



1 Dataset　2 Machine learning　3 Explanation

System selection

→ From Yaraghi et al. TSE paper: 25 Java systems

→ *Angel* : 308 builds, highest failure rate (40%)

→ 150 features: test case history, source code metrics, coverage information

Data preparation

→ Test cases in training builds are sorted by verdict and execution time

→ Hold-out strategy: Test on build N, train with all previous builds

→ Preserve last 20% of the training builds for validation

A. S. Yaraghi, M. Bagherzadeh, N. Kahani, and L. Briand, "Scalable and Accurate Test Case Prioritization in Continuous Integration Contexts" IEEE Transactions on Software Engineering, pp. 1–24, 2022.

Learning to rank with LambdaMART

- → Combination of LambdaRank + Boosted trees
- → Implementation from **LightGBM** (default params)
- → Model performance: NDGC metric
- → Run on every failed build (124)
- → Choosing one build (35 tests) for inspection:
  - ✓ 5th Best performance (NDGC=0.9896)
  - ✓ First one to put all failed test cases on top
- → Scripts and full results:

https://github.com/tepia-taxonomy/aist23-workshop

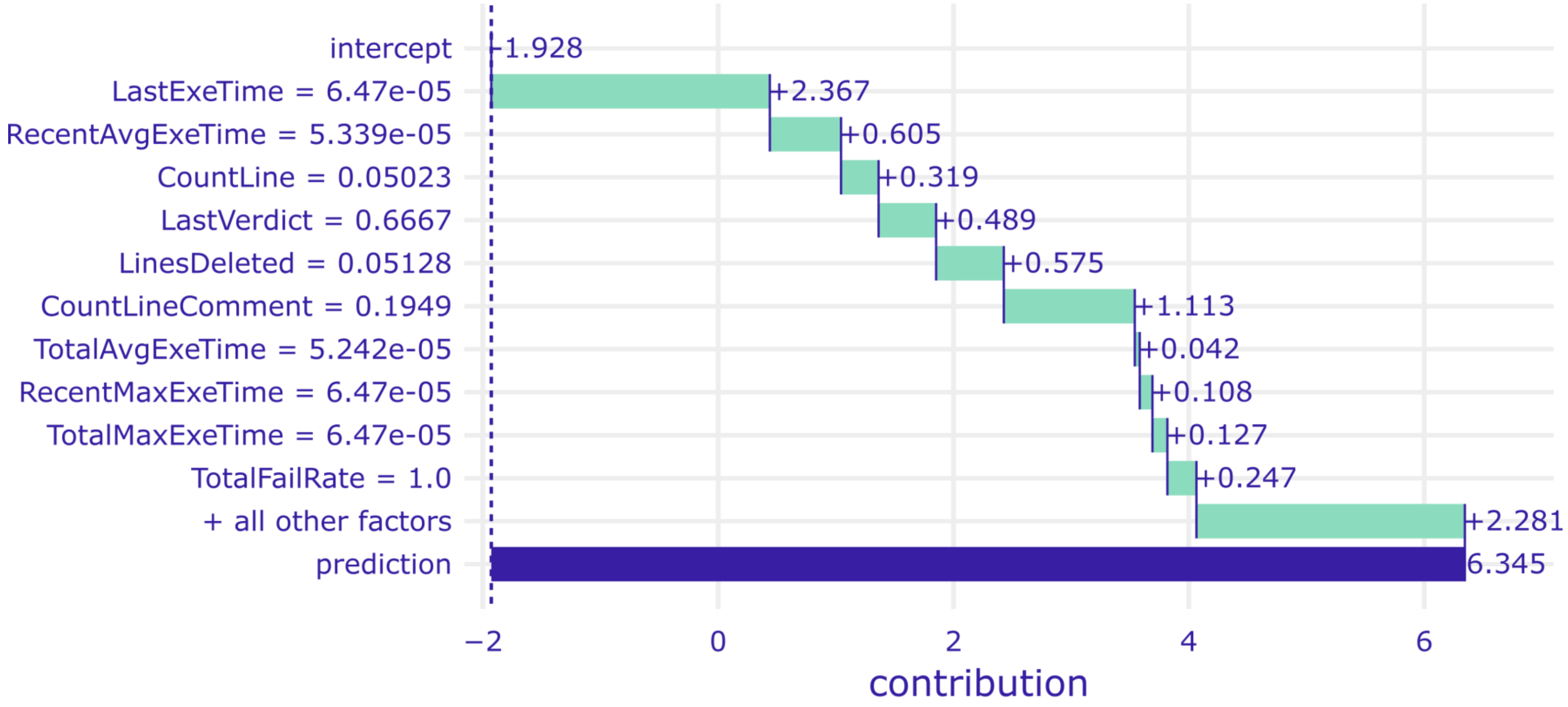
Test case	Relevance	Ex. Time	Position
5141	6.3452	28	1
2953	5.5933	70	2
5140	5.5489	60	3
2732	2.6534	6	4
2161	2.2648	24	5
…			
2963	-7.3633	70403	34
2955	-8.5646	102292	35

Preliminary results

1 Dataset 2 Machine learning 3 Explanation

Local explanations with Break Down (DALEX) – Scenario 1B

- Top features in terms of contribution appear in the global model explanation too
- Some important features refer to test code (instead of SUT code) and test case properties



Local explanations with Break Down (DALEX) – Scenario 1C

→ We compute the cosine similarity between the feature contributions returned by Break Down
→ Failed test cases (positions 1-3) have very similar explanations
→ Similarity is reduced as the test cases are more distant in the ranking (1-4 vs. 1-35)

Test case position	Test case position	Explanation similarity
TC 1	TC 2	0.9563
TC 1	TC 3	0.9948
TC 2	TC 3	0.9509
TC 1	TC 4	0.8330
TC 1	TC 35	-0.7225

Current challenges

- Current explainable methods are not "prepared" to inspect learning-to-rank model/outputs
- TCP is a time-dependent problem that makes comparisons across builds difficult
- Do we need black-box models always?

Future work

- Explore contrastive and counterfactual generation methods to study additional scenarios
- Include the temporal dimension in our analysis
- Generalise conclusions based on more systems and black-box models
- Develop specific explainable methods for TCP and evaluate them with testers

Towards Explainable Test Case Prioritisation with Learning-to-Rank Models

Questions?

AIST 2023

Thank you!

Aurora Ramírez

aramirez@uco.es

@aurora_rq

www.uco.es/users/aramirez/en

Proyecto PID2020-115832GB-I00 y Ayuda RED2018-102472-T financiados por MCIN/ AEI /10.13039/501100011033/ y por FEDER Una manera de hacer Europa