THOR

Predictive Test selection using AI

(Yogesh, Manish, Rahul, Shiv, Karan, Ankit)

AGENDA

Idea ?

To develop a better method, we consider a different question:

What is the likelihood that a given test finds a regression with a particular code change?

Why ?

- Latency in Code Validation and Delivery
- Resource Unavailability
- report of source code impacting.
- Resources

What?

- Developer Productivity
- Smart pre-commits Validation using CI

How ?

- We created a predictive model that estimates the probability of each test failing for a newly proposed code change.

- Instead of defining the model manually, we built it by using a data set containing results of tests on historical code changes and then applying standard machine learning techniques.

- During training, our system learns a model based on features derived from previous code changes and tests.

- When the system is analyzing new code changes, we apply the learned model to a feature-based abstraction of the code change.

- For any particular test, the model is then able to predict the likelihood of detecting a regression.

- To do this, the system uses decision tree machine learning algorithm.

How ?

- JIRA -> TC'name,
  -> Git ID -> Sources

- Source -> TC (using mapped DB)

- Finally get the List of Test cases to runs

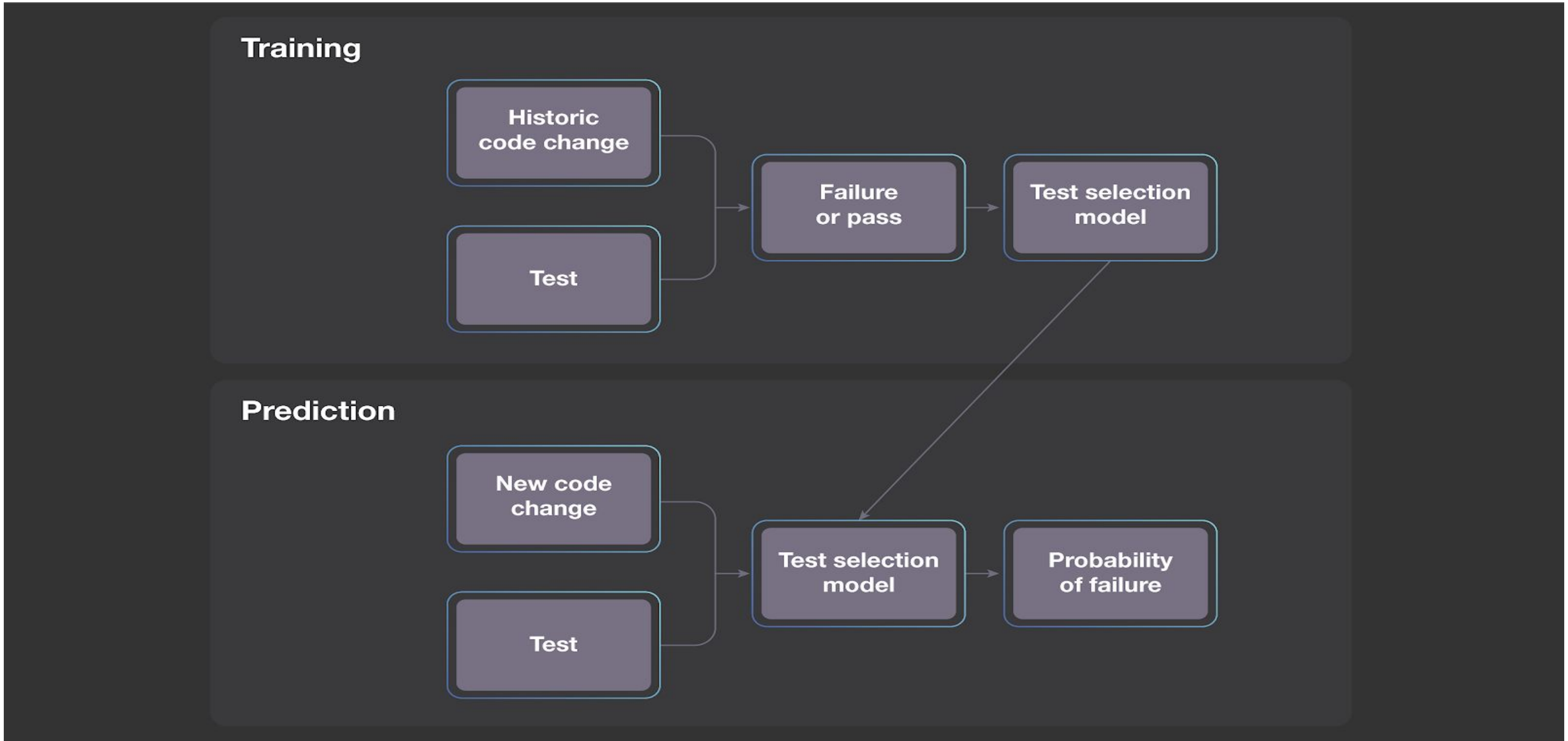
- JITA -> TC Run Time

- TC -> Update in CI Data sets -> Give the results

Workflow

- Developer raises Gerrit review request.
- Tool engine sends an API request to Gerrit and identifies code churn.
- Tests to run is decided based on modified files.
- Dataset is created for decision tree machine learning algorithm.
- Tests and Source files are passed as input data to machine learning core module.
- Tests to run are selected by machine learning core module.
- CI engine sends an API request to Jita to create testset dynamically using the tests selected.
- CI engine sends testset to Jita to run Nutest.
- CI engine publishes test run results in Gerrit request.

How ?



Training

Historic code change

Test

Failure or pass

Test selection model

Prediction

New code change

Test

Test selection model

Probability of failure

Benefits

- Enable engineers to create and deploy new features more quickly and effectively.

- Increase the efficiency of testing infrastructure.

-

Thank You