OCTOBER 9 – 11, 2023 | #PNCQC
PORTLAND, OR

IMPROVING ENTERPRISE SCALE TEST AUTOMATION WITH ML-BASED PREDICTIVE ANALYTICS

DMITRIY GUMENIUK
EPAM SYSTEMS

Dmitriy Gumeniuk

Head of Testing Products @ EPAM Systems

Based in New York Metro Area

Leads development of a portfolio of accelerators focused on the usage of Machine Learning and Neural Networks in test automation.

While having 16 years of experience in software development, has been contributing to DevTestOps communities by actively speaking at events and organizing the **DelEx** Conference,
aimed at inspiring DevTestOps practices.

delex

/ghdmitry

/ghdmitry

Dmitriy Gumeniuk

Phonetic Spelling: «DMEE-tree»

In the Team of:

Adam

Tariq

delex

/ghdmitry

/ghdmitry

ReportPortal.io – AI-powered Test Automation Dashboard

Drill4J – Test Gap and Impact analysis, regression minimization tool

Healenium – self-healing capabilities for Selenium-based test cases

TDSpora.ai – ML-based synthetic data generation based on production data

QASpace – JIRA plugin for neat Test Case Management

Challenges of Test automation at scale
and
practical AI/ML to overcome them

Challenges

1. **Complexity of Test Scenarios.** As apps complexity grow, crafting realistic automated tests becomes increasingly difficult.

2. **Management of Test Data.** Data is the backbone of any testing process. In an enterprise setting, handling vast amounts of test data across different databases, services, and environments becomes a laborious task.

3. **Integration with Existing Systems.** Automated tools must work with a mix of modern and legacy systems, requiring custom setups.

4. **Resource Constraints.** Budget, time, and skilled personnel limit the scope and maintenance of automated testing.

5. **Maintainability of Test Scripts.** Test scripts need frequent updates to stay relevant as software evolves.

6. **Noise and False Positives.** Automated tests can yield misleading results, requiring extra analysis to identify real issues.

7. **Understanding the ‘Why’ Behind Failures.** Traditional automated systems are excellent at pointing out 'what' has failed but are usually not designed to diagnose 'why' it has failed. This requires human intervention, further complicating the automation process.

1. **Complexity of Test Scenarios.** As apps complexity grow, crafting realistic automated tests becomes increasingly difficult.

2. **Management of Test Data.** Data is the backbone of any testing process. In an enterprise setting, handling vast amounts of test data across different databases, services, and environments becomes a laborious task.

3. **Integration with Existing Systems.** Automated tools must work with a mix of modern and legacy systems, requiring custom setups.

4. **Resource Constraints.** Budget, time, and skilled personnel limit the scope and maintenance of automated testing.

5. **Maintainability of Test Scripts.** Test scripts need frequent updates to stay relevant as software evolves.

6. **Noise and False Positives.** Automated tests can yield misleading results, requiring extra analysis to identify real issues.

7. **Understanding the 'Why' Behind Failures.** Traditional automated systems are excellent at pointing out 'what' has failed but are usually not designed to diagnose 'why' it has failed. This requires human intervention, further complicating the automation process.

Overcomes

- **Intelligent Test Data Management**
- **Self-Healing and Adaptive Test Scripts**
- **Enhanced Failure Diagnosis**
- **False Positive Identification**
- **Test Failure Categorization**

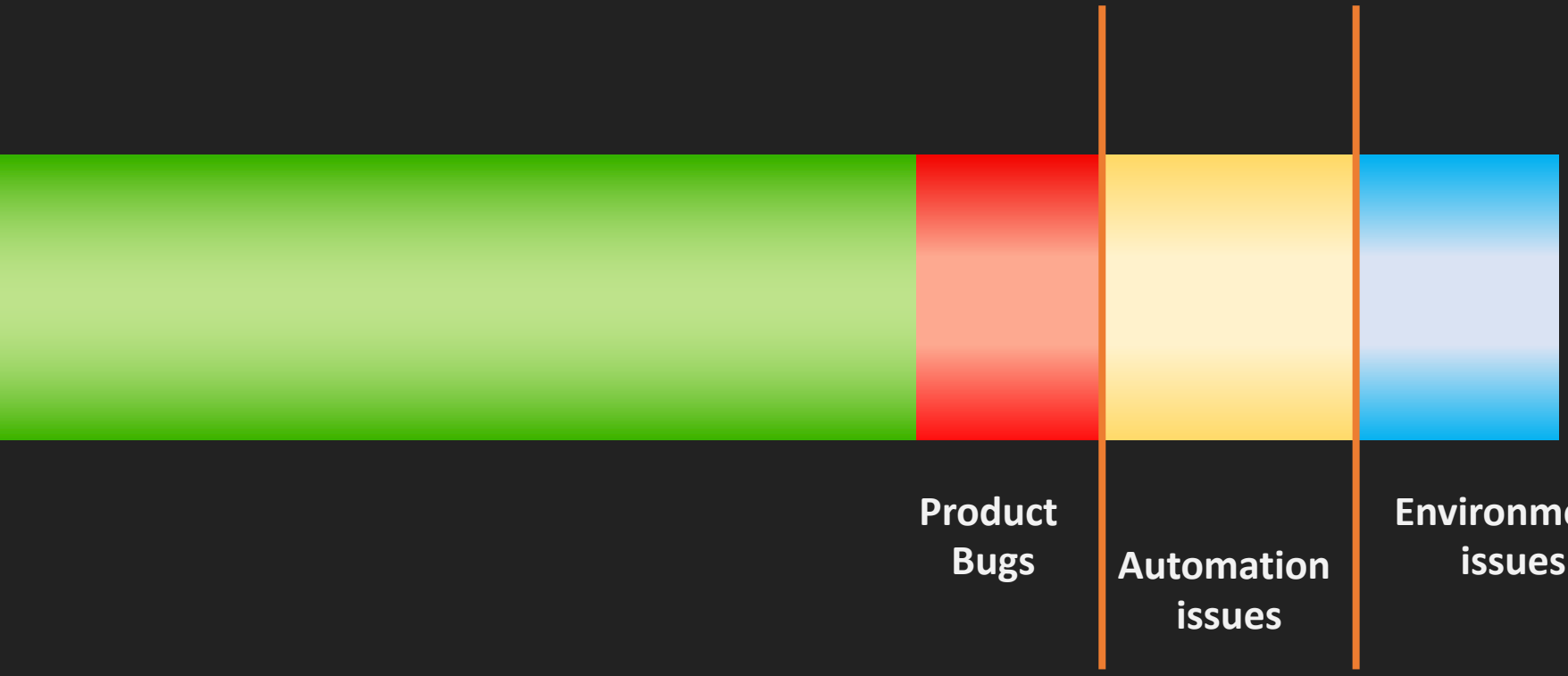
Understanding the

‘Why’

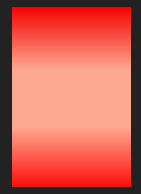
Behind Failures

Triage of failed results

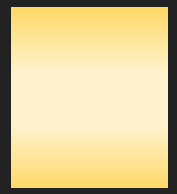
Triage of failed results



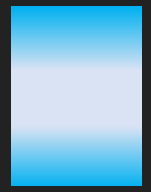
Triage of failed results



Product Bugs – **actual value** of test automation in finding product bugs



Automation issues – **technical dept** of QA automation team
(outdated tests, broken locator, incorrect test data, waiters, etc.)



Environment issues – **infrastructure dept** and inconsistency
(configuration issues, limited budget, unstable connection,
version mismatches, etc.)

How ?

How?

Machine Learning vs

```
java.lang.AssertionError: Invalid Upc Service Navigation link redirection. expected [true] but
found [false]
org.testng.Assert.fail(Assert.java:94)
org.testng.Assert.failNotEquals(Assert.java:513)
org.testng.Assert.assertTrue(Assert.java:42)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavigationServi
ceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpringC
ontextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
org.testng.internal.Invoker.invokeMethod(Invoker.java:642)
org.testng.internal.Invoker.invokeTestMethod(Invoker.java:811)
org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1137)
org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:129)
org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:112)
org.testng.TestRunner.privateRun(TestRunner.java:753)
org.testng.TestRunner.run(TestRunner.java:607)
org.testng.SuiteRunner.runTest(SuiteRunner.java:368)
org.testng.SuiteRunner.runSequentially(SuiteRunner.java:363)
org.testng.SuiteRunner.privateRun(SuiteRunner.java:321)
org.testng.SuiteRunner.run(SuiteRunner.java:270)
org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1284)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1280)
org.testng.TestNG.runSuitesLocally(TestNG.java:1209)
org.testng.TestNG.runSuites(TestNG.java:1124)
org.testng.TestNG.run(TestNG.java:1096)
my.project.web.ta.run.TestNgRunner.main(TestNgRunner.java:32)
```

```
java.lang.AssertionError: Invalid Upc Service Navigation link redirection. expected
[true] but found [false]
org.testng.Assert.fail(Assert.java:94)
org.testng.Assert.failNotEquals(Assert.java:513)
org.testng.Assert.assertTrue(Assert.java:42)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNa
vigationServiceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:1
00)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.jav
a:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(Abstract
TestNGSpringContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java
:201)
org.testng.internal.Invoker.invokeMethod(Invoker.java:642)
org.testng.internal.Invoker.invokeTestMethod(Invoker.java:811)
org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1137)
org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:129)
org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:112)
org.testng.TestRunner.privateRun(TestRunner.java:753)
org.testng.TestRunner.run(TestRunner.java:607)
org.testng.SuiteRunner.runTest(SuiteRunner.java:368)
org.testng.SuiteRunner.runSequentially(SuiteRunner.java:363)
org.testng.SuiteRunner.privateRun(SuiteRunner.java:321)
org.testng.SuiteRunner.run(SuiteRunner.java:270)
org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1284)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1280)
org.testng.TestNG.runSuitesLocally(TestNG.java:1209)
org.testng.TestNG.runSuites(TestNG.java:1124)
org.testng.TestNG.run(TestNG.java:1096)
my.project.web.ta.run.TestNgRunner.main(TestNgRunner.java:32)
```

Product Bug

```
NoElementFoundException: missing button on UI, unable to locate the Refresh button for
the table.
org.testng.Assert.fail(Assert.java:94)
org.testng.Assert.failNotEquals(Assert.java:513)
org.testng.Assert.assertTrue(Assert.java:42)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavi
gationServiceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100
)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:
189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTe
stNGSpringContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:2
01)
org.testng.internal.Invoker.invokeMethod(Invoker.java:642)
org.testng.internal.Invoker.invokeTestMethod(Invoker.java:811)
org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1137)
org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:129)
org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:112)
org.testng.TestRunner.privateRun(TestRunner.java:753)
org.testng.TestRunner.run(TestRunner.java:607)
org.testng.SuiteRunner.runTest(SuiteRunner.java:368)
org.testng.SuiteRunner.runSequentially(SuiteRunner.java:363)
org.testng.SuiteRunner.privateRun(SuiteRunner.java:321)
org.testng.SuiteRunner.run(SuiteRunner.java:270)
org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1284)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1280)
org.testng.TestNG.runSuitesLocally(TestNG.java:1209)
org.testng.TestNG.runSuites(TestNG.java:1124)
org.testng.TestNG.run(TestNG.java:1096)
my.project.web.ta.run.TestNgRunner.main(TestNgRunner.java:32)
```

Automation Issue

Case study

2.7Tb

Obfuscated production
test automation logs

Experimental Setup

20 production project at larger scale

6 combinations of programming language, test runners, tools and test types:

Java + testNG + Selenium – e2e UI testing
Java + JBehave + Selenium – e2e UI testing, in BDD
.Net + Specflow + Selenium – e2e UI testing in BDD
.Net + NUnit + Selenium – e2e UI testing in BDD
Python + Pytest + API calls – API testing
JavaScript + Mocha + WebDriverIO - UI testing

Primary Data types collected:

- **Automated Testing Results** including logs of execution, specifically all error logs and stack traces of the failed test cases, test case IDs, parent test suites, test parameters and attributes
- **Manual categorization** to serve as a benchmark, manual categorizations were also conducted for a subset of tests, focusing on the areas where machine learning-based recommendations were provided.

Primary Data types collected

- Manual categorization
  - Product
  - Automation
  - Environment

- Automated Testing Results

```
java.lang.AssertionError: Invalid Upc Service Navigation link redirection. expected [true] but
found [false]
org.testng.Assert.fail(Assert.java:94)
org.testng.Assert.failNotEquals(Assert.java:513)
org.testng.Assert.assertTrue(Assert.java:42)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavigationSe
rviceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpri
ngContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
org.testng.internal.Invoker.invokeMethod(Invoker.java:642)
org.testng.internal.Invoker.invokeTestMethod(Invoker.java:811)
org.testng.internal.Invoker.invokeTestMethods(Invoker.java:1137)
org.testng.internal.TestMethodWorker.invokeTestMethods(TestMethodWorker.java:129)
org.testng.internal.TestMethodWorker.run(TestMethodWorker.java:112)
org.testng.TestRunner.privateRun(TestRunner.java:753)
org.testng.TestRunner.run(TestRunner.java:607)
org.testng.SuiteRunner.runTest(SuiteRunner.java:368)
org.testng.SuiteRunner.runSequentially(SuiteRunner.java:363)
org.testng.SuiteRunner.privateRun(SuiteRunner.java:321)
org.testng.SuiteRunner.run(SuiteRunner.java:270)
org.testng.SuiteRunnerWorker.runSuite(SuiteRunnerWorker.java:52)
org.testng.SuiteRunnerWorker.run(SuiteRunnerWorker.java:86)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1284)
org.testng.TestNG.runSuitesSequentially(TestNG.java:1280)
org.testng.TestNG.runSuitesLocally(TestNG.java:1209)
org.testng.TestNG.runSuites(TestNG.java:1124)
org.testng.TestNG.run(TestNG.java:1096)
my.project.web.ta.run.TestNgRunner.main(TestNgRunner.java:32)
```

75 / 25

Training Set / **Validation Set**

Meta data:

- TestCaseID, parent names, attributes, parameters

Metrics for Evaluation

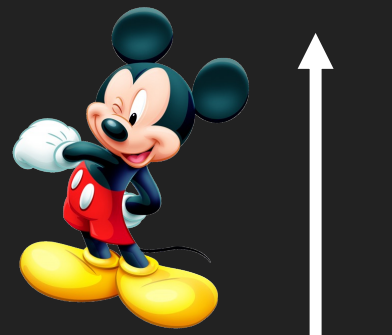
1. **Accuracy:** **t**he ratio of correctly classified defects to the total number of defects.
2. **Precision:** How many of the flagged issues were indeed defects.
3. **Recall:** **o**f all the defects, how many were correctly identified by the machine learning algorithms.
4. **F1-Score:** metric that combines both precision and recall to provide a single measure of a model's performance.
5. **Time-Saved:** The amount of manual effort saved by automating defect categorization and recommendation processes.

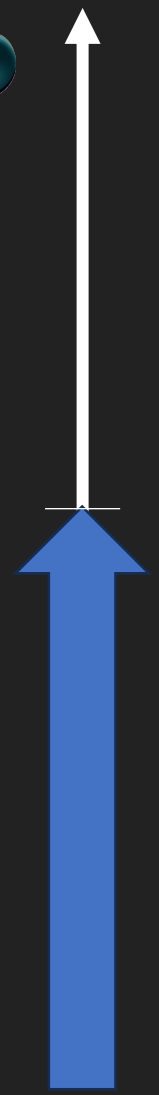
Logs vectorization

Mickey woke up in a magical world full of colorful gems. Intrigued, Mickey grabbed a map left by a mysterious sorcerer.
"To the gem palace!" Mickey exclaimed.
Along the way, he met a talking squirrel. "Follow me, Mickey, to avoid traps," the squirrel advised. Grateful, Mickey continued cautiously. Soon, they reached the palace. Guarded by enchanted statues, Mickey knew he had to be smart. Remembering the map's riddle, Mickey sang a tune and the statues danced away. Inside, Mickey found a gem that granted happiness and shared it with everyone.

TF-IDF = 0.06 * 4 = 0.24

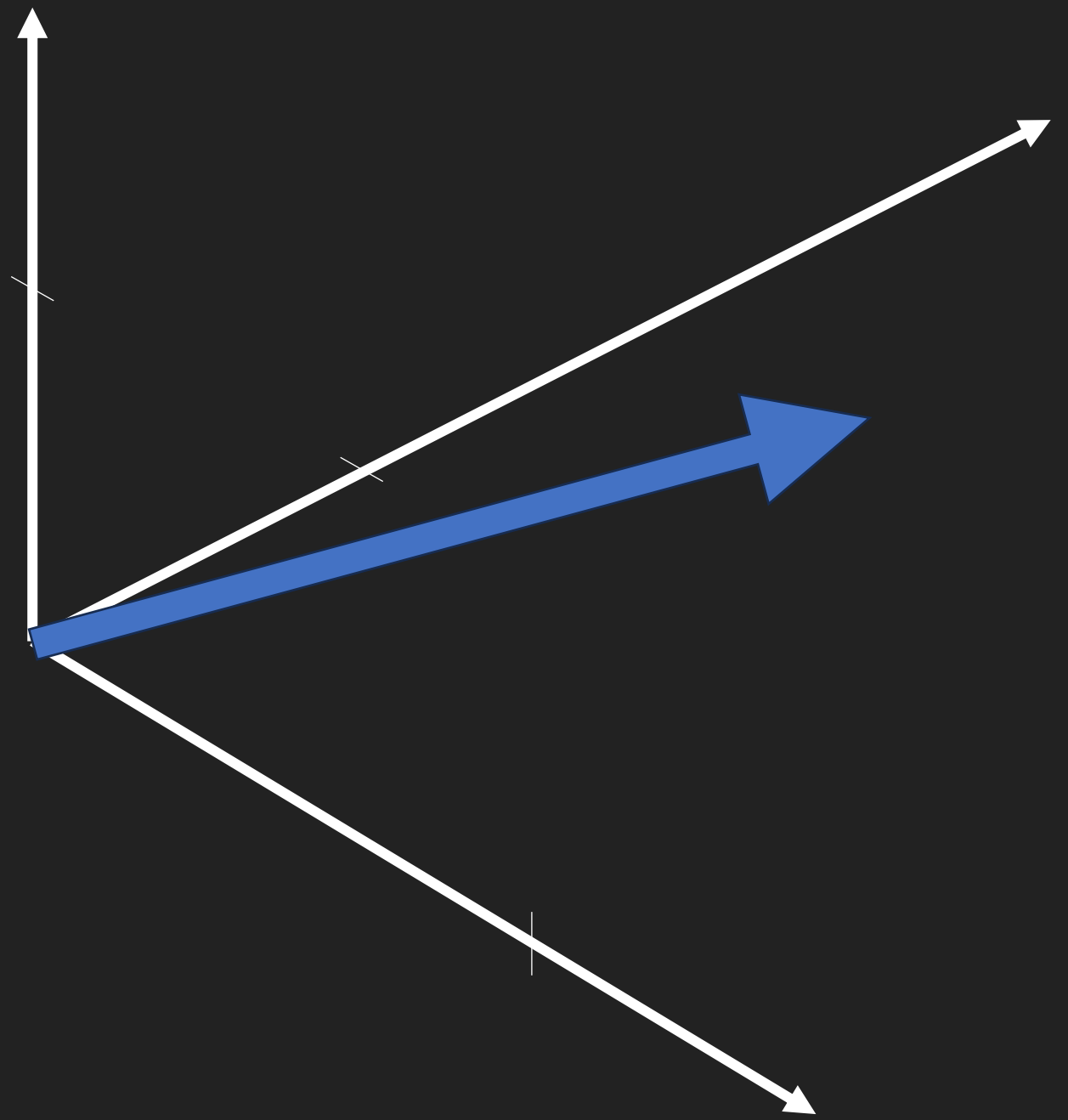
TF = 6 / 100 = 0.06
IDF = Log (10 000 000/ 1000) = 4

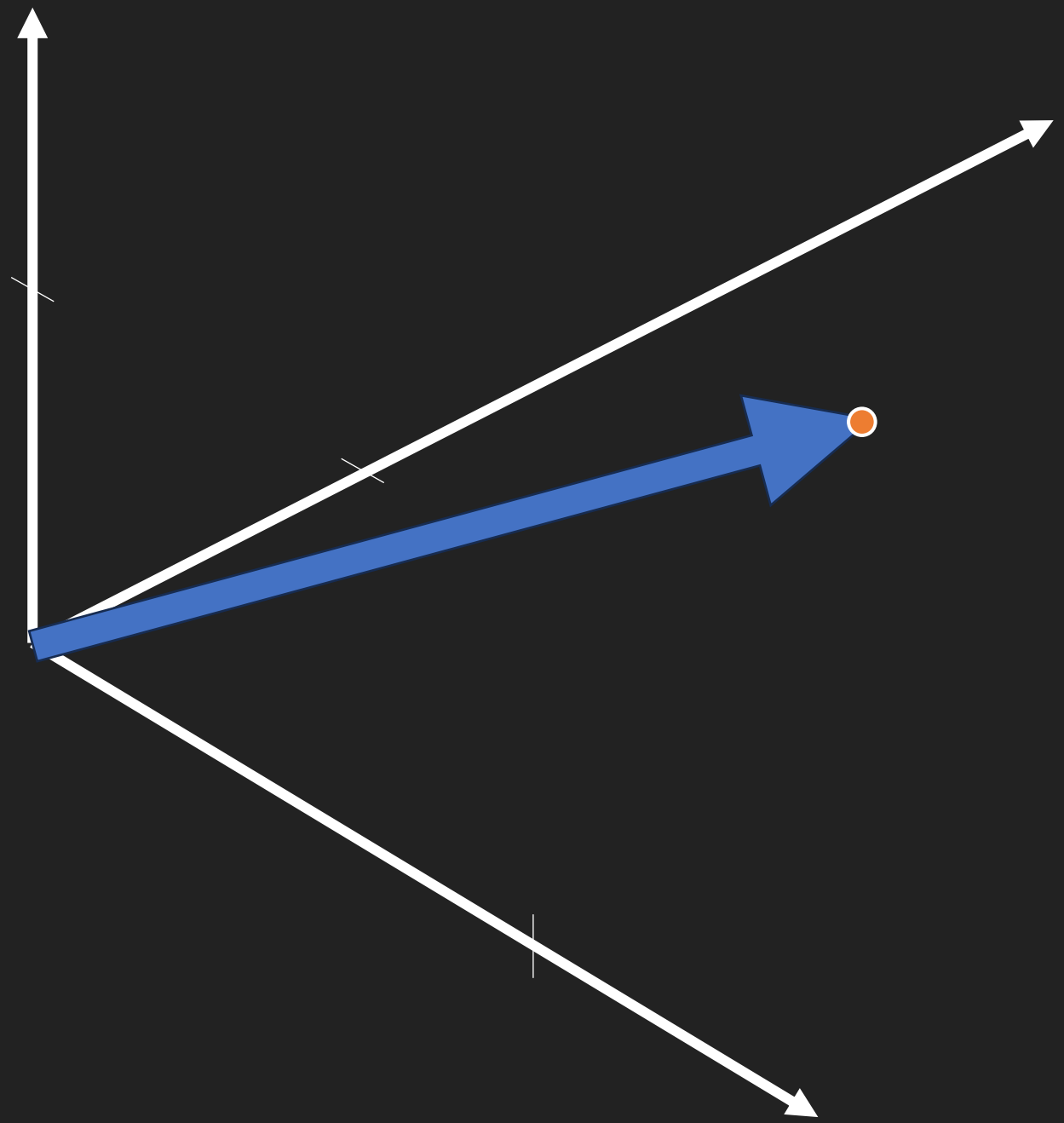


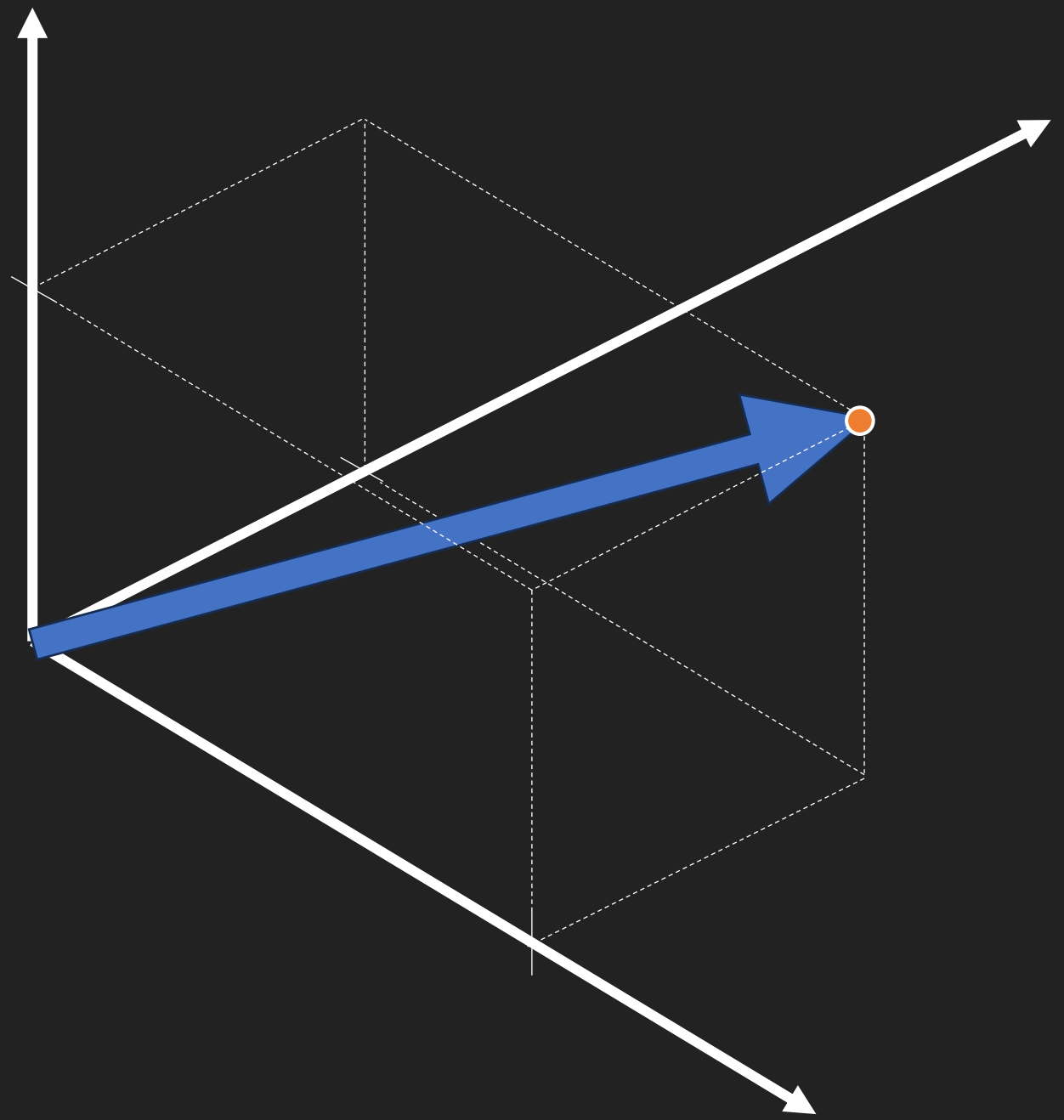


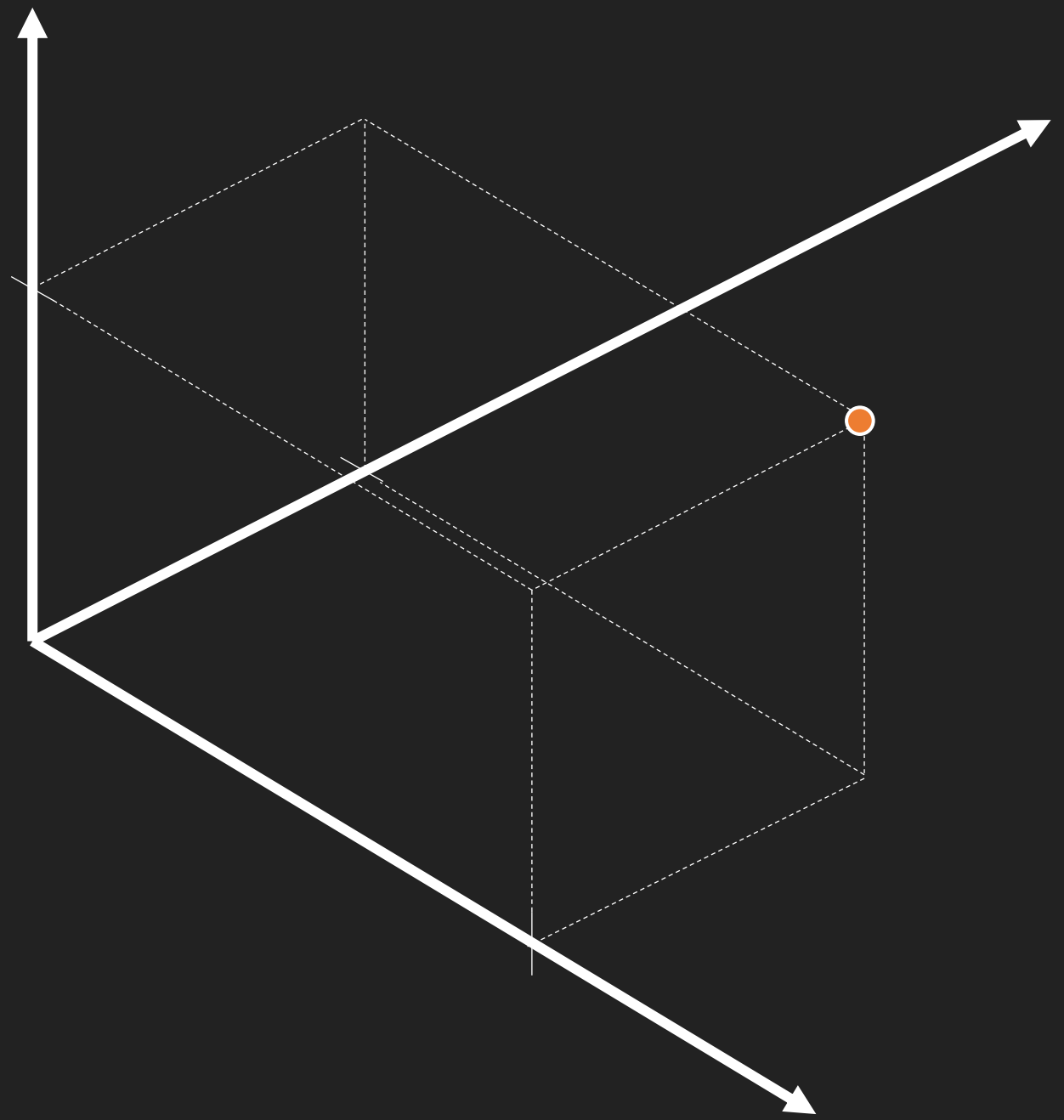
TF-IDF = 0.06 * 4 = 0.24

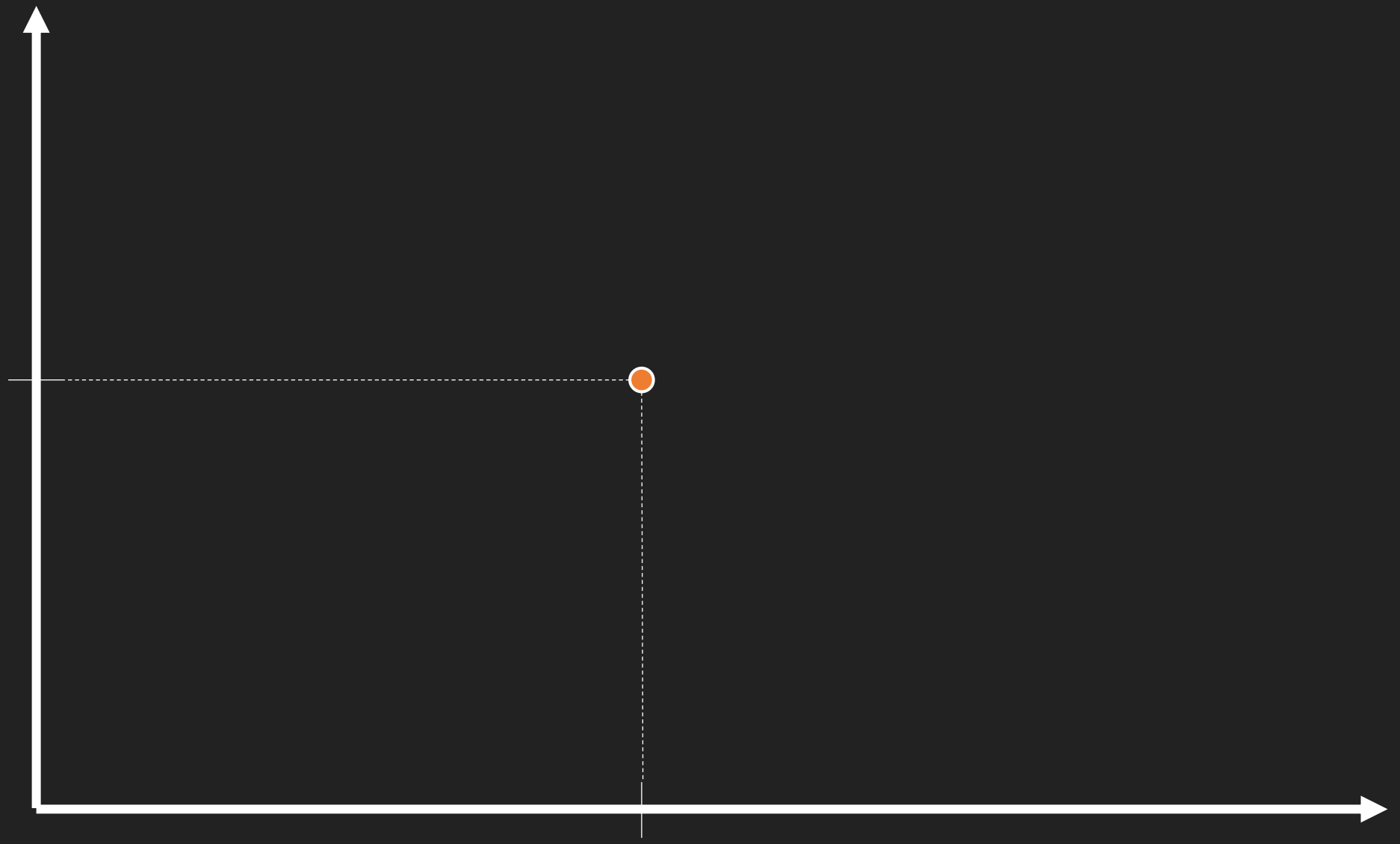
gems

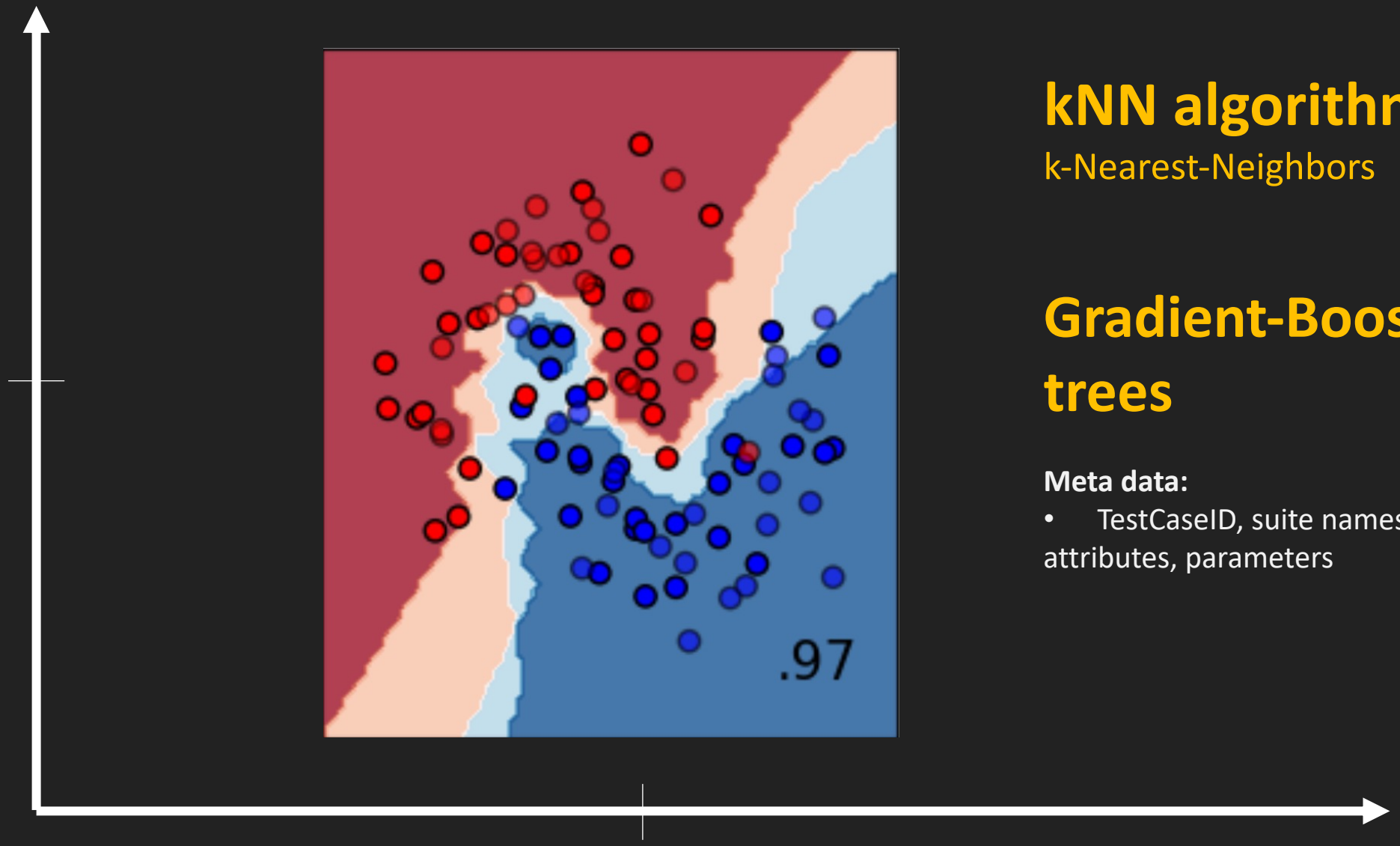












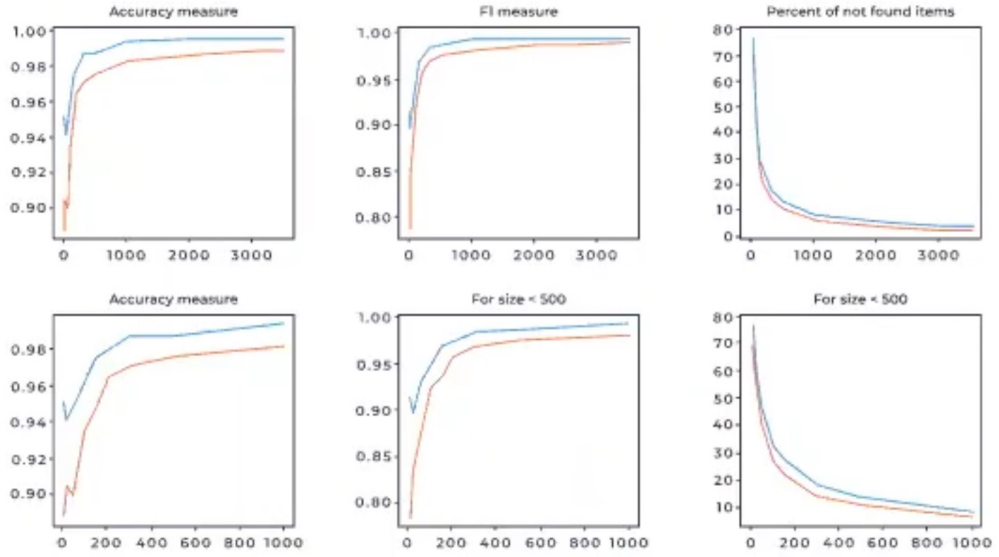
kNN algorithm

k-Nearest-Neighbors

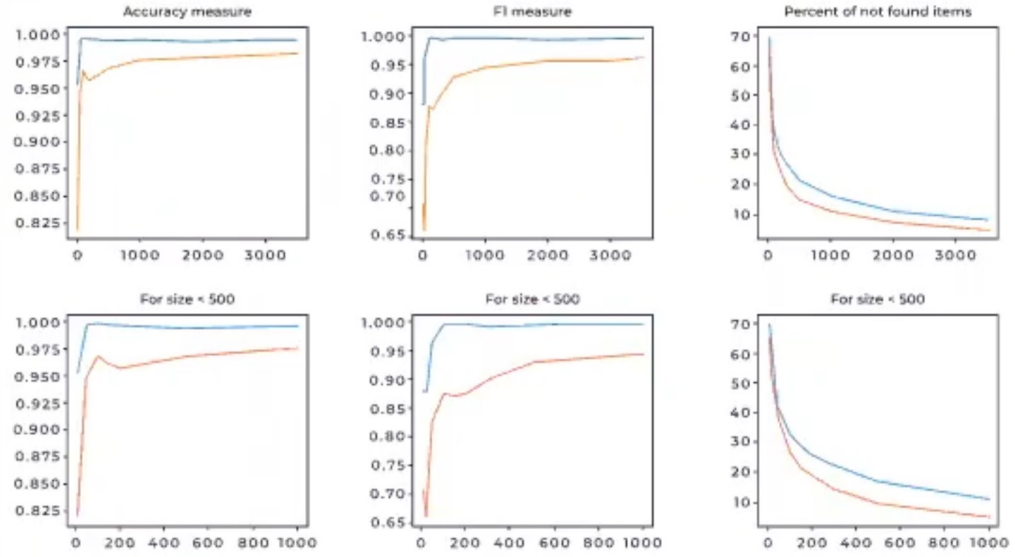
Gradient-Boosted trees

Meta data:

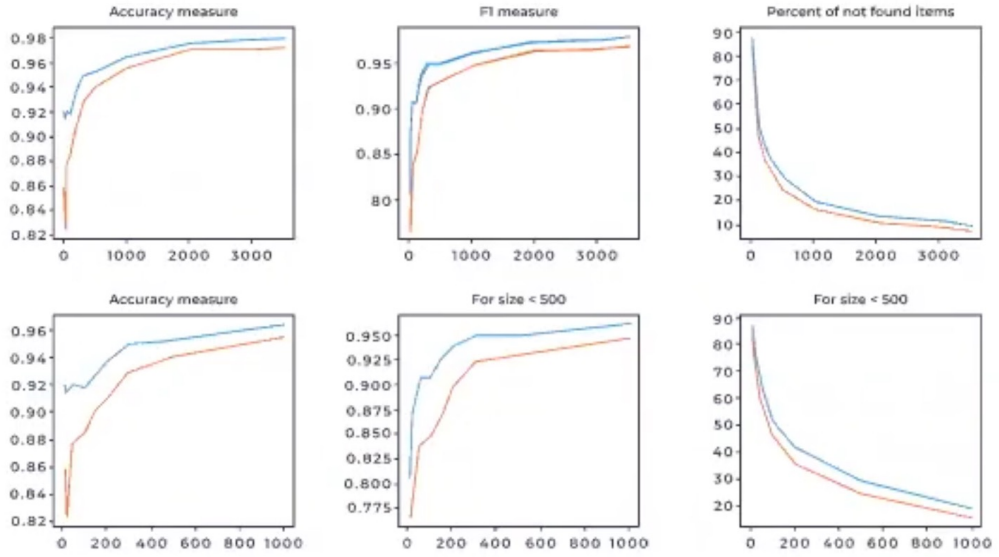
- TestCaseID, suite names, attributes, parameters



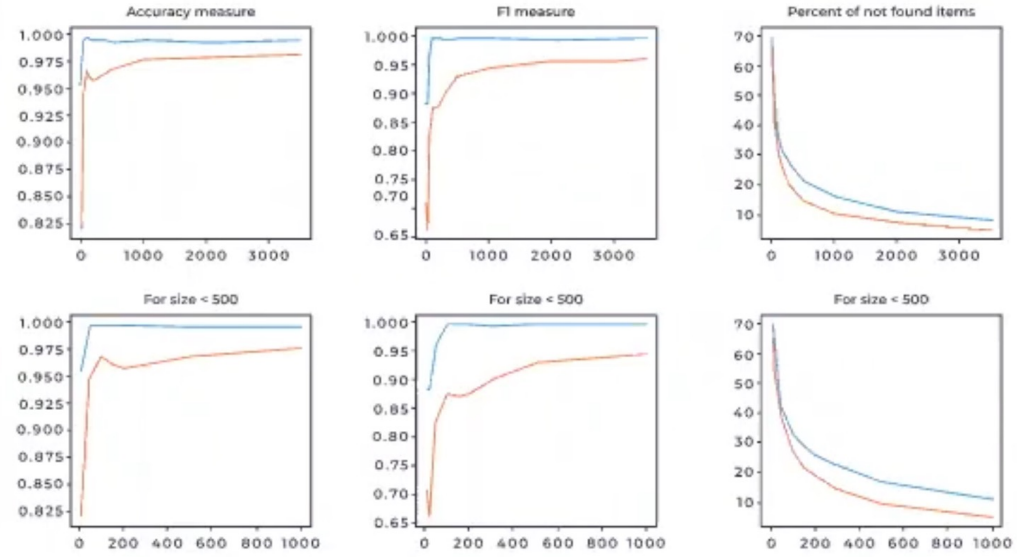
java+testng



net+selenium



.net + specflow



java + jbehave

Automation issue

```
org.openqa.selenium.NoSuchElementException: Unable to locate element:
{"method":"xpath","selector":"//input[@id='username']"}
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavigationSe
rviceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpri
ngContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
```

Product Bug

```
java.lang.AssertionError: Invalid Upc Service Navigation link redirection. expected [true] but found [false]
org.testng.Assert.fail(Assert.java:94)
org.testng.Assert.failNotEquals(Assert.java:513)
org.testng.Assert.assertTrue(Assert.java:42)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavigationServiceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpringContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
```

Environment

```
java.sql.SQLException: Connection timed out
at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
At
com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:
122)
at com.example.tests.UserDataTest.testUserDataRetrieval(UserDataTest.java:56)
framessun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpri
ngContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
```

Product Bug

```
Failed. expected [true] but was [false]

org.testng.Assert.fail(Assert.java:94)
my.project.web.tests.navigation.checkLinksFromServiceNavigationBarAreClickable(MainNavigationSe
rviceNavigationTest.java:61)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpri
ngContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
```


```
Failed.
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.testng.internal.MethodInvocationHelper.invokeMethod(MethodInvocationHelper.java:100)
org.testng.internal.MethodInvocationHelper$1.runTestMethod(MethodInvocationHelper.java:189)
org.springframework.test.context.testng.AbstractTestNGSpringContextTests.run(AbstractTestNGSpri
ngContextTests.java:175)
org.testng.internal.MethodInvocationHelper.invokeHookable(MethodInvocationHelper.java:201)
```

```
Failed. expected [true] but was [false]
```

1. Garbage in – Garbage out
2. Make your logs and errors meaningful

Results

Up to 90%

Reduction of test analysis efforts

Up to x2

Speed up of regression cycle

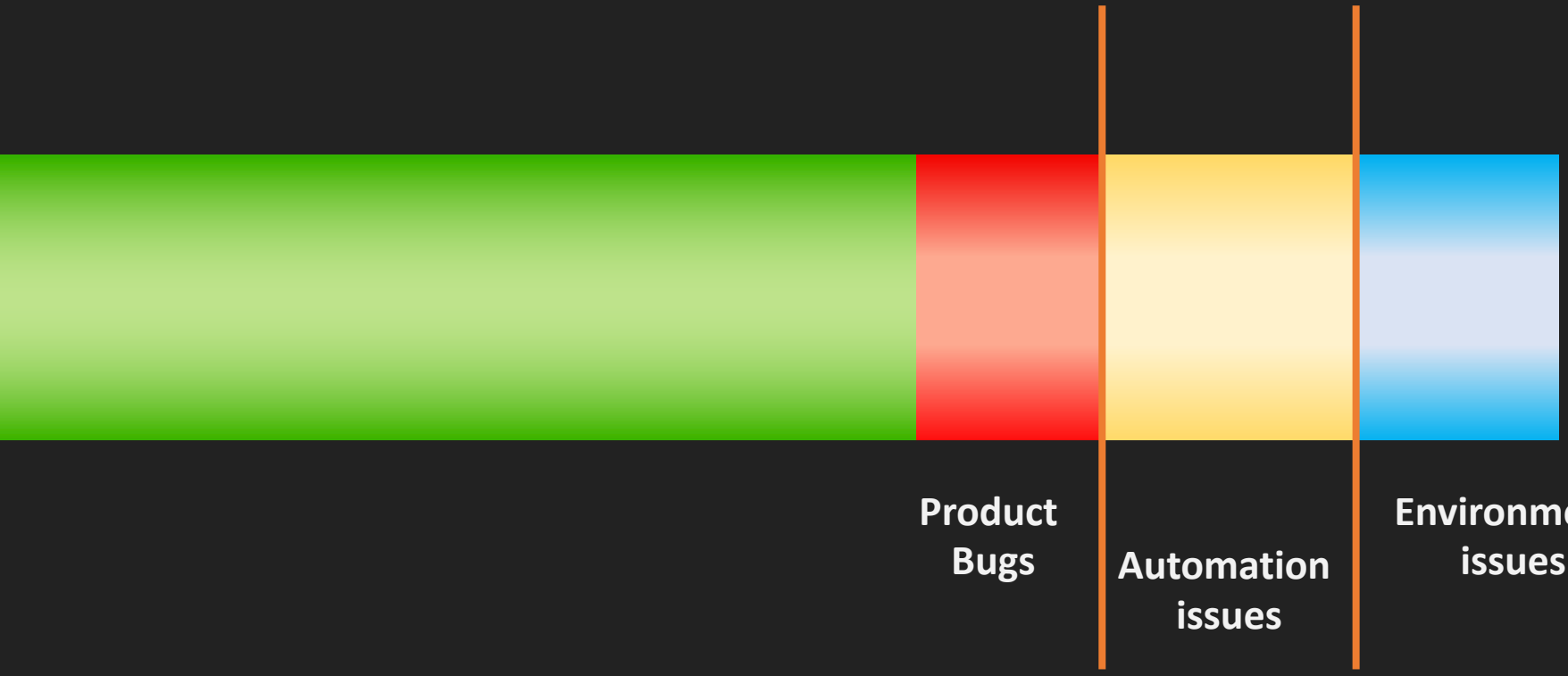
1. **Log Quality**
   - Generic error logs hinder algorithms from deducing meaningful insights.
   - A mere enrichment of error descriptions in assertions or exceptions managed by try-catch led to an 80% surge in categorization accuracy.
   - The principle is clear: If a QA engineer unfamiliar with the failing test can't discern the issue based on the log description, machine learning will likely struggle as well.

1. **The Human Factor:** Human uncertainty in categorizing test failures often leads to the selection of inappropriate categories. These categories often undergo transformation after 2-3 test runs. The "Friday Bug" anomaly was identified, where engineers, in a bid to wrap up their week, hastily marked a segment of failed tests as defect-free. This behavior significantly blurred clustering and reduced precision.

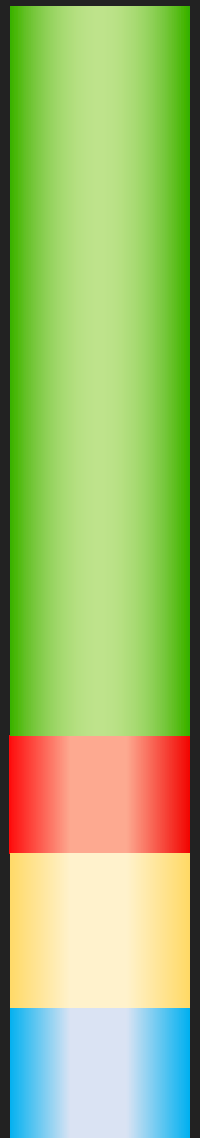
2. **Incremental Learning:** is essential due to changes in the project landscape, such as the introduction of new tests, application evolution, and enhanced coverage.

3. **Decisions Based on Recency.** For the ML model it does make sense to prioritize latest categorization decision over the old ones via boosting the scoring. Decisions related to tests were heavily influenced by the recency of the decision timestamp..

Triage of failed results



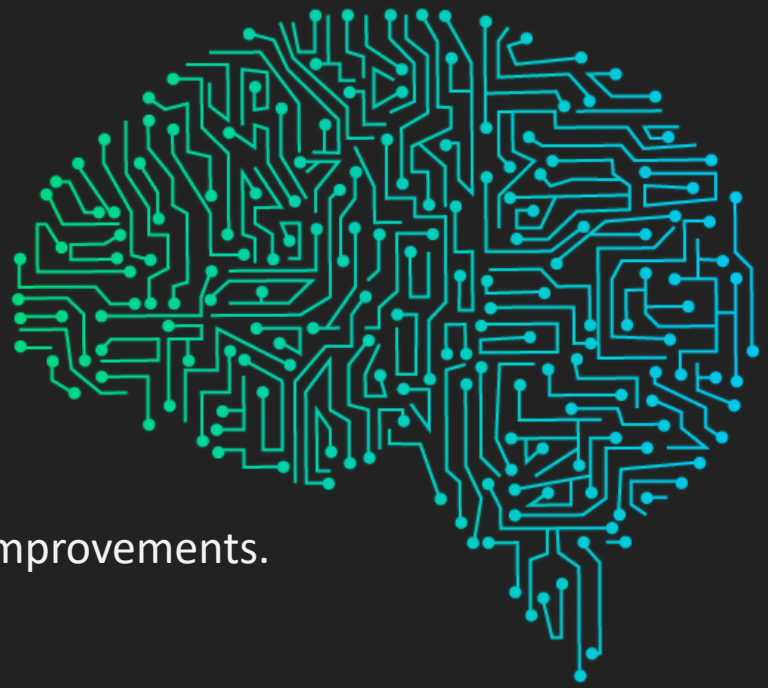
Triage of failed results



Goal:
- categorize each and every failed test case

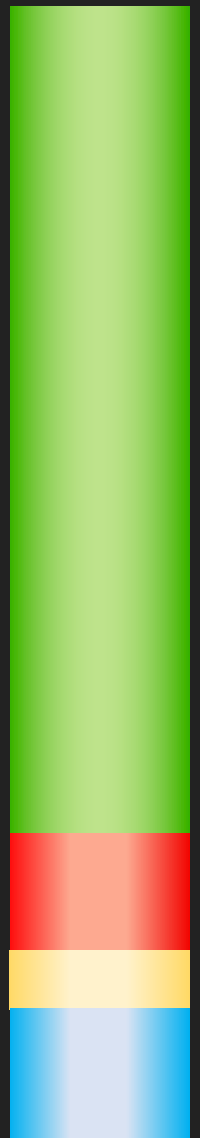
Why?
- Known issues
- Multiply failures because of the one specific issue
- Just boring

Solution
- **ML algorithms** to triage known issues



Result of ML categorization:
- Known issues categorized automatically
- Unknown issues are grouped for bulk investigation
- Team focuses on new failures
- Released team effort is focused on stabilization and improvements.

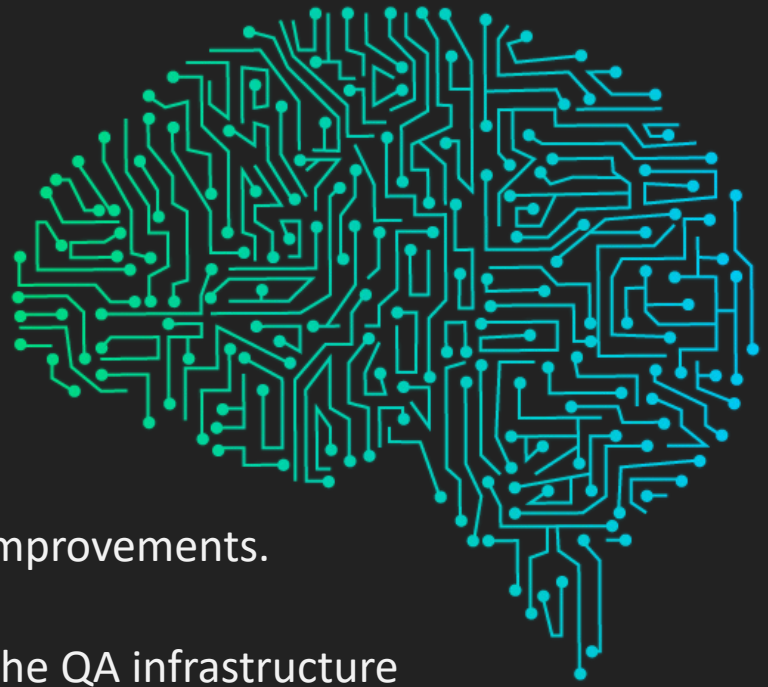
Triage of failed results



Goal:
- categorize each and every failed test case

Why?
- Known issues
- Multiply failures because of the one specific issue
- Just boring

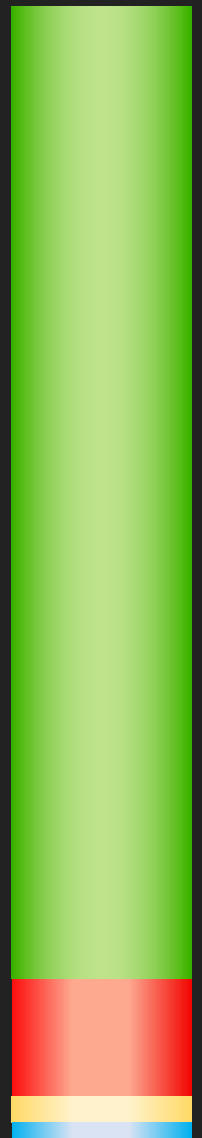
Solution
- **ML algorithms** to triage known issues



Result of ML categorization:
- Known issues categorized automatically
- Unknown issues are grouped for bulk investigation
- Team focuses on new failures
- Released team effort is focused on stabilization and improvements.

Improved visibility returns as investments into the QA infrastructure

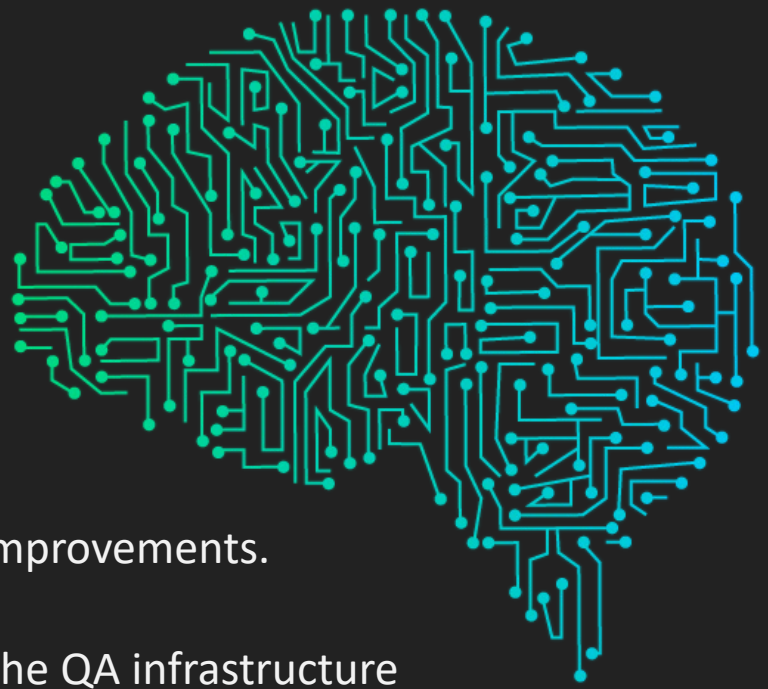
Triage of failed results



Goal:
- categorize each and every failed test case

Why?
- Known issues
- Multiply failures because of the one specific issue
- Just boring

Solution
- **ML algorithms** to triage known issues



Result of ML categorization:
- Known issues categorized automatically
- Unknown issues are grouped for bulk investigation
- Team focuses on new failures
- Released team effort is focused on stabilization and improvements.

Improved visibility returns as investments into the QA infrastructure

Future Directions

1. **Advanced Machine Learning Models**:
   - **Deep Learning Integration:** While the current focus has been on kNN and Boosting Trees algorithms, there's potential to integrate deep learning models.
   - **Natural Language Processing**: GPT-like, and NLP techniques can be employed to categorize error logs, especially those written in natural language

2. **Application Logs as a Rich Data Source** can serve as a treasure trove of information. These logs can provide deeper insights into system behavior, user interactions, and potential error patterns.

3. **Visual AI for Enhanced Analysis**, as an Image recognition technique can be employed to detect error patterns or pop-ups in application screenshots. This "Visual AI" can be pivotal in identifying visual discrepancies that might be missed in traditional testing.

4. **Automated Test Generation with Generative Large Language Models**, aimed at a language generation, not just analyze but also generate test cases based on software requirements, user behavior, and historical defect data.

5. **Self-Optimizing Test Suites beyond self-healing test scripts**, envision test suites that can self-optimize, reordering tests based on dependencies, historical fail rates, or execution time, ensuring the most efficient test run possible.

6. **Enhanced Integration with Development and Operations**, allowing for real-time feedback loops. This can lead to immediate defect resolution during the development phase itself, further streamlining the DevOps pipeline.

http://ReportPortal.io

reportportal

http://ReportPortal.io

Dmitriy Gumeniuk

/ghdmitry

/ghdmitry