Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering
InIT Institut für angewandte
Informationstechnologie

Fehlervorhersage in Java-Code mittels Machine Learning

Tobias Meier (meierto3), Yacine Mekesser (mekesyac)

4. Juli 2016

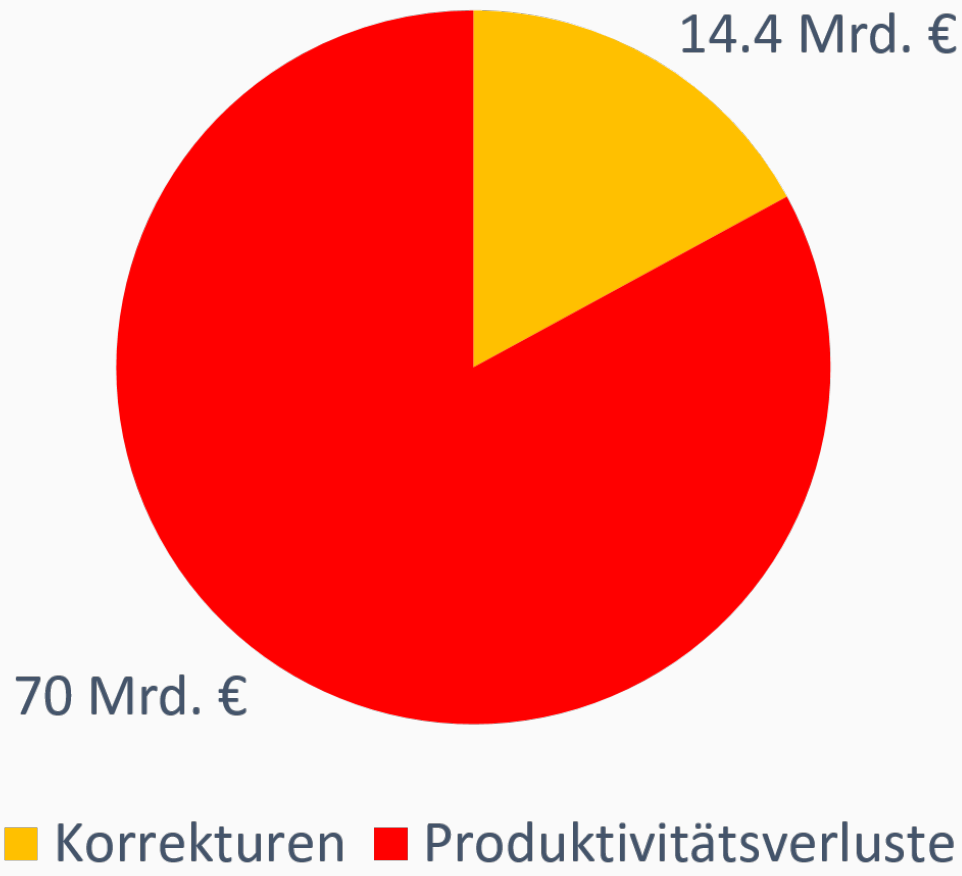
ZHAW

Inhaltsverzeichnis

Einleitung

Softwarefehler I

- 84.4 Mrd. € jährliche Verluste in DE



Softwarefehler II

Softwarefehler nehmen zu:

- Komplexität
- Wachsende Codebase
- Agile Softwareentwicklung

Idee

Frühwarnsystem für Softwarefehler

- Qualität
- Effizienz
- Günstiger

Beobachtungen

- Textverständnis mit Machine Learning
- Open-Source-Projekte
  - Öffentliche Code Repositories
  - Issue-Tracking-Systeme

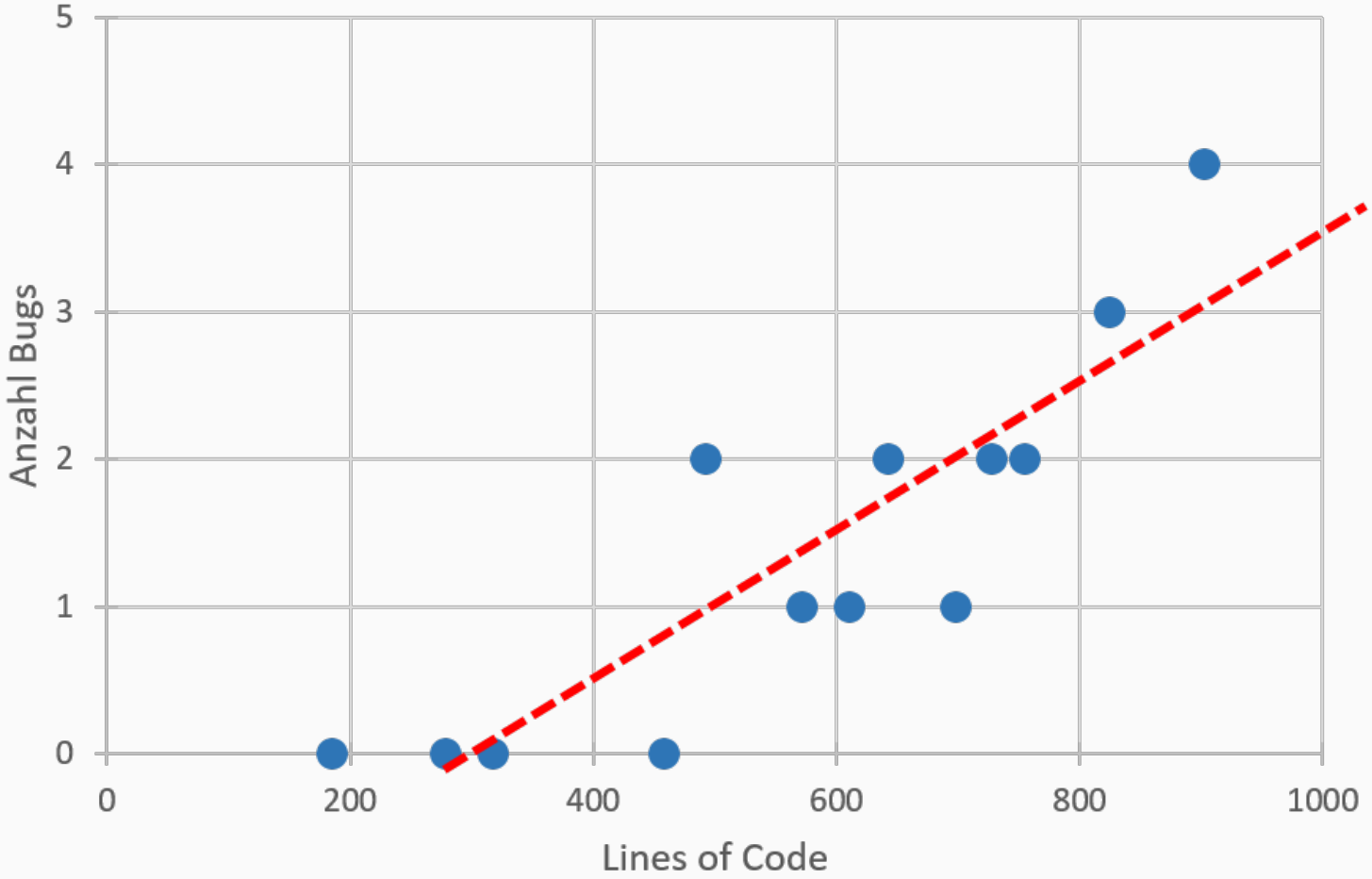
Ansatz

- Vorhersage von Bugs
- Codeanalyse mit ML
- Beschränkung auf Java-Projekte

Grundlagen

Machine Learning

Ermöglicht Computern aus Beispielen zu lernen und dann zu verallgemeinern.



Vorgehen

Stand der Technik

Statischen Codeanalyse

checkstyle

CODE CLIMATE

Machine Learning

- Ideen und Potential vorhanden
- Keine schlüsselfertige Produkte

Ziele

- Grundstein für weitere Arbeiten legen
- Toolset für projektbezogenes ML bereitstellen
- Einbeziehung von Textanalyse-Features

Herausforderungen

- ML Datensatz
  - Code und Issue-Tracking
- Features
- ML-Pipeline

Architektur

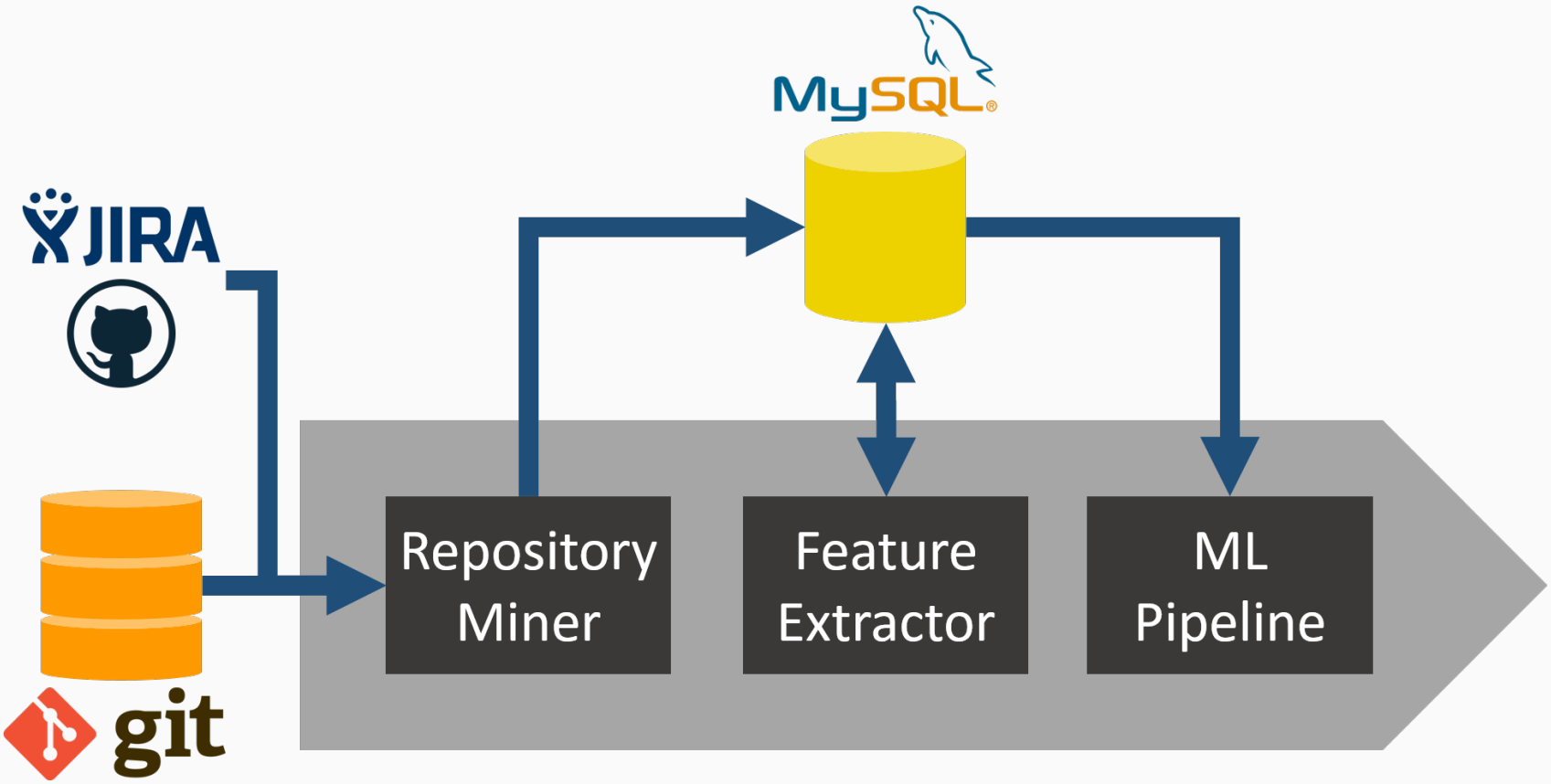
Lösungsansatz

Drei separate Tools:

- Repository Miner
- Feature Extractor
- ML-Pipeline

Ermöglicht Modularität , Flexibilität & Effizienz

Systemarchitektur

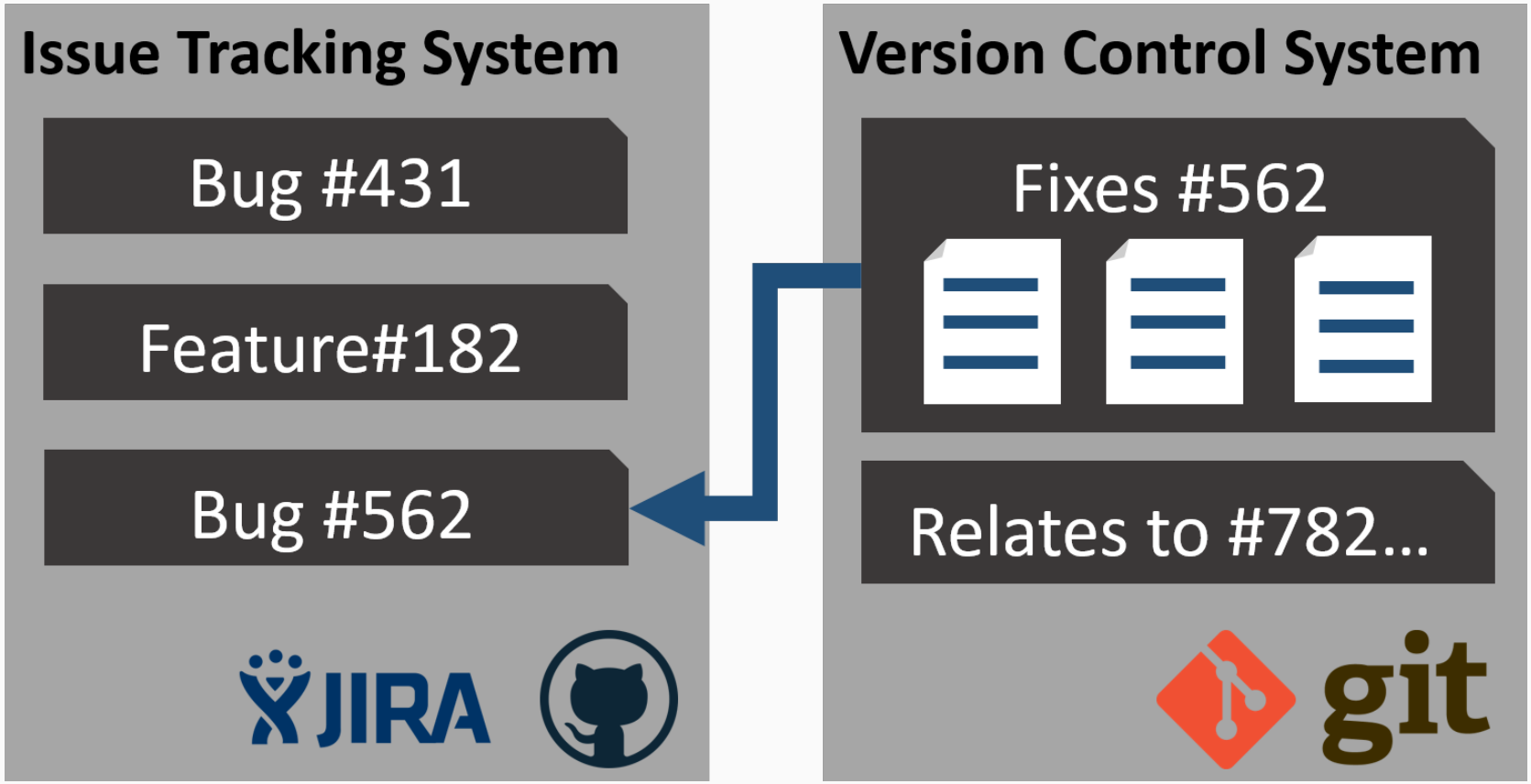


Repository Miner

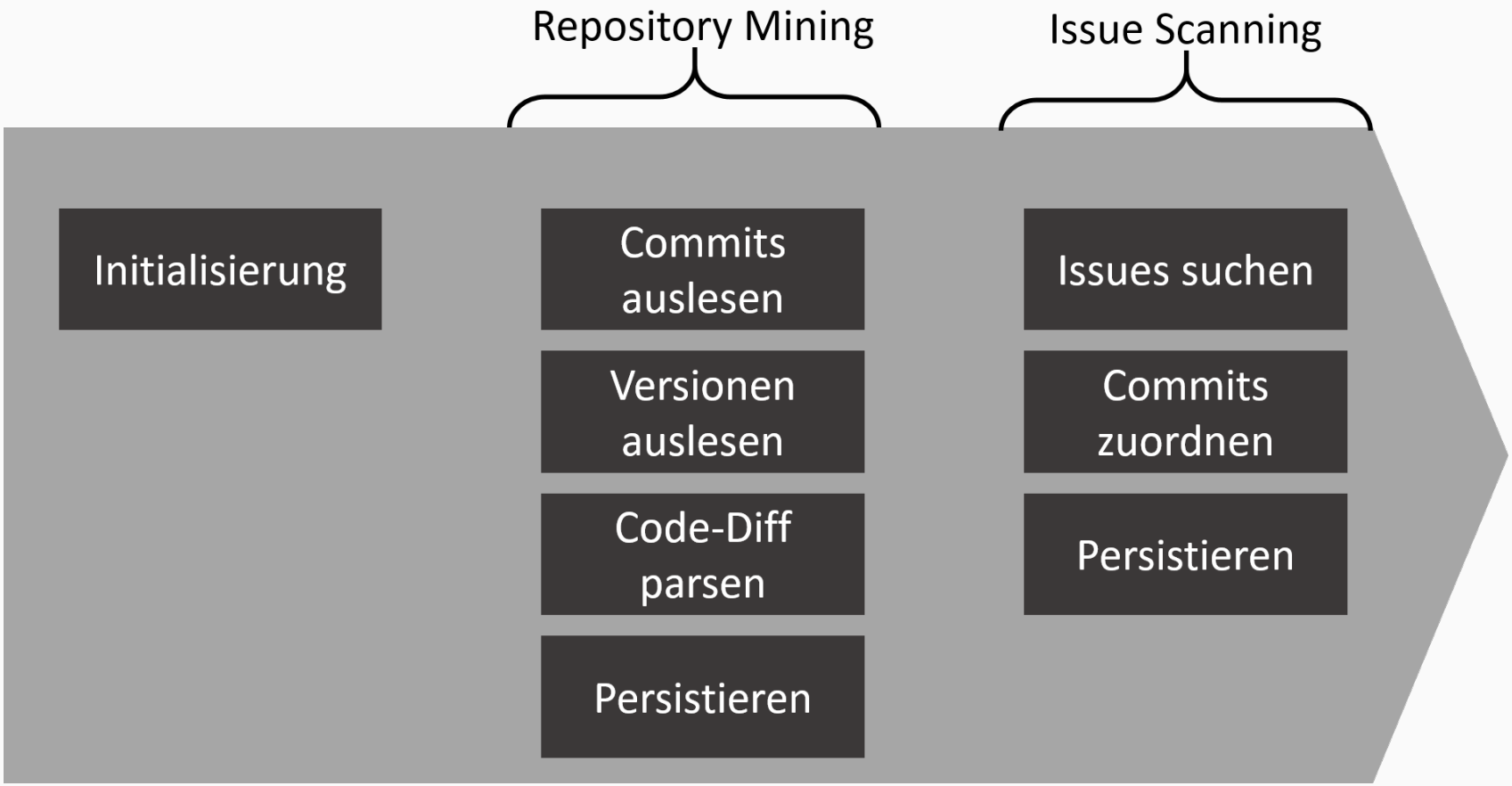
Repository-Miner

- Anbindung an Git, GitHub & Jira
- Schnellen Zugriff bieten
- Daten strukturiert und normalisiert speichern

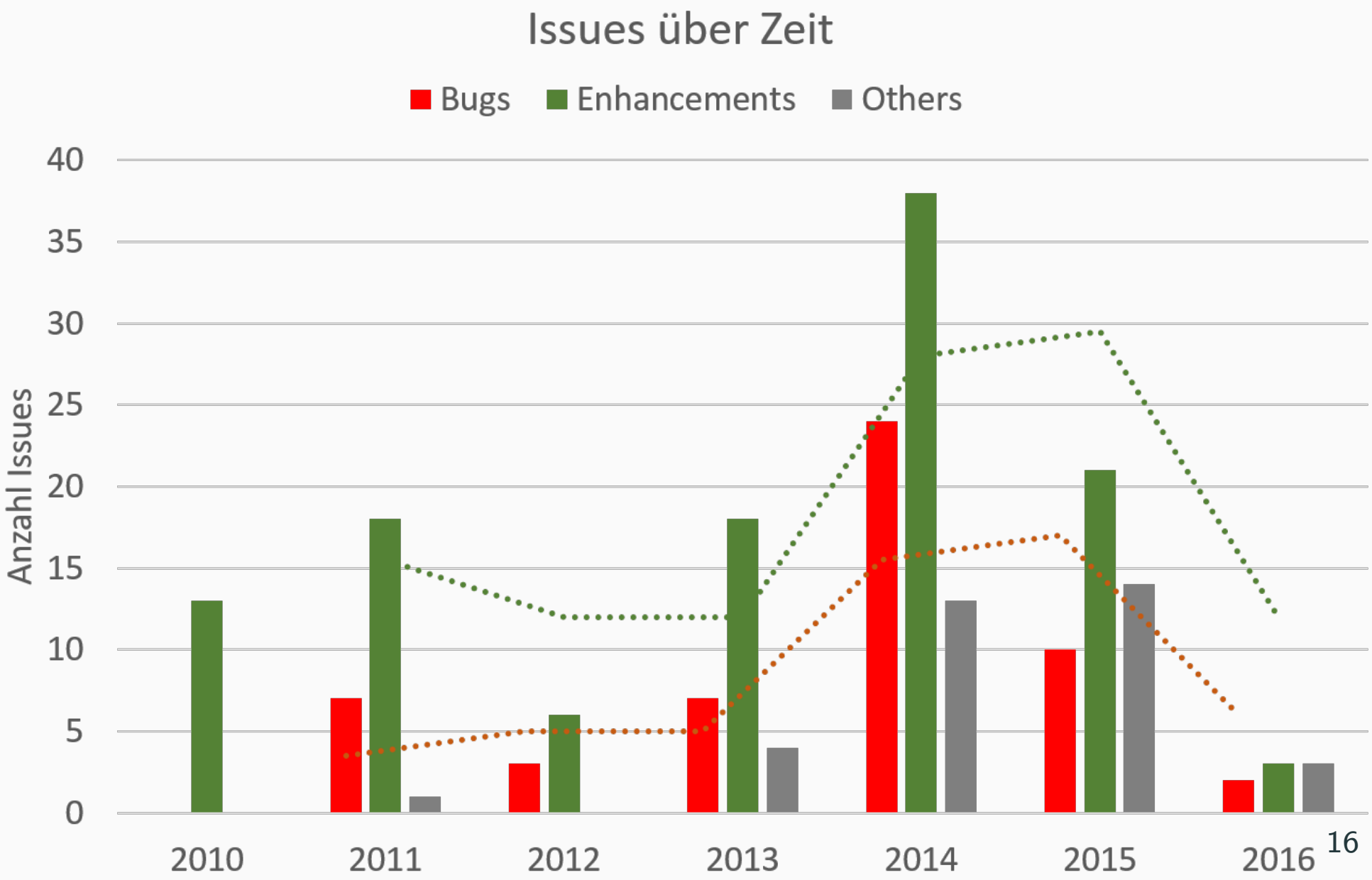
Herausforderungen



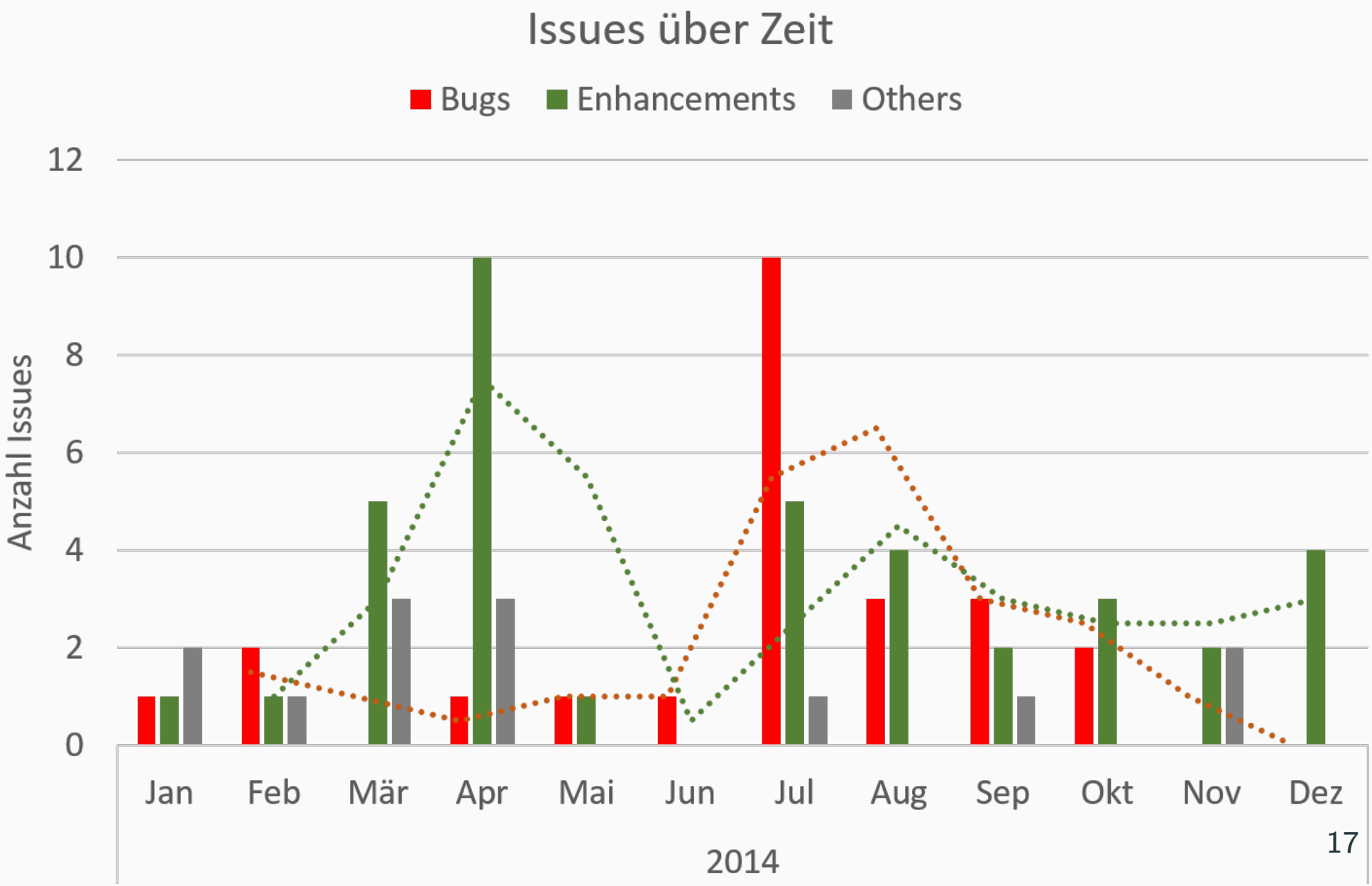
Workflow


Repository Mining
Issue Scanning
Initialisierung
Commits auslesen
Versionen auslesen
Code-Diff parsen
Persistieren
Issues suchen
Commits zuordnen
Persistieren

Erkenntnisse



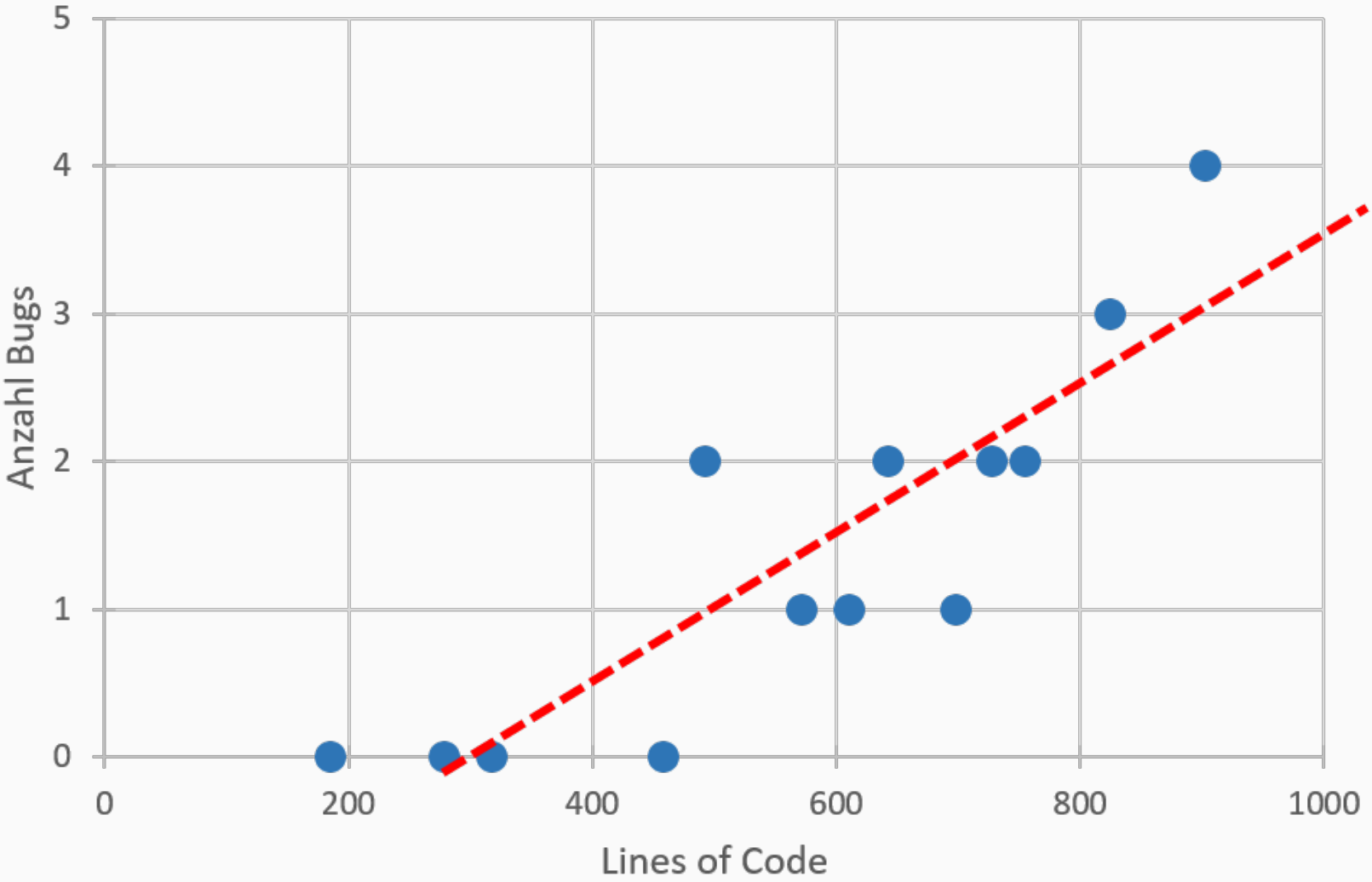
Erkenntnisse



Features

Features

- Messbare Eigenschaften einer Beobachtung



Lines-of-Code-Features

- Anzahl Codezeilen
- Grosse Java Files - viele Fehler?
- Einfache Implementation

Objektorientierte Features

- Kopplung zwischen Objekten
- Mangel an Kohäsion in Klasse
- Anzahl Methoden pro Klasse

Code-Complexity-Features

- Halstead
  - Gegenüberstellung von Operatoren & Operanden
- McCabe
  - Anzahl Kontrollfluss-Statements

Anzahl-und-Typen-Features

- Anzahl Imports, Interfaces, Methoden, Klassenvariablen
- Komplexität von Klasse

Temporale Features

- Anzahl Bugs im letzen Monat
- Anzahl Tage seit letztem Commit
- Datenbasis: Historie der Versionsverwaltung

Textanalyse-Features

- Länge von Namen (Methoden, Variablen)
- Text- und Sprachanalyse

N-Grams

- Bug-Mustererkennung
- Verschiedene Abstraktionslevels

```
Code (Java): public class MyClass..

Code (AST):
(55, TypeDeclaration)
  (83, Modifier)
    (43, SimpleType)
      (42, SimpleName)

N-Gram: 55_83_43_42
```

Feature Extractor

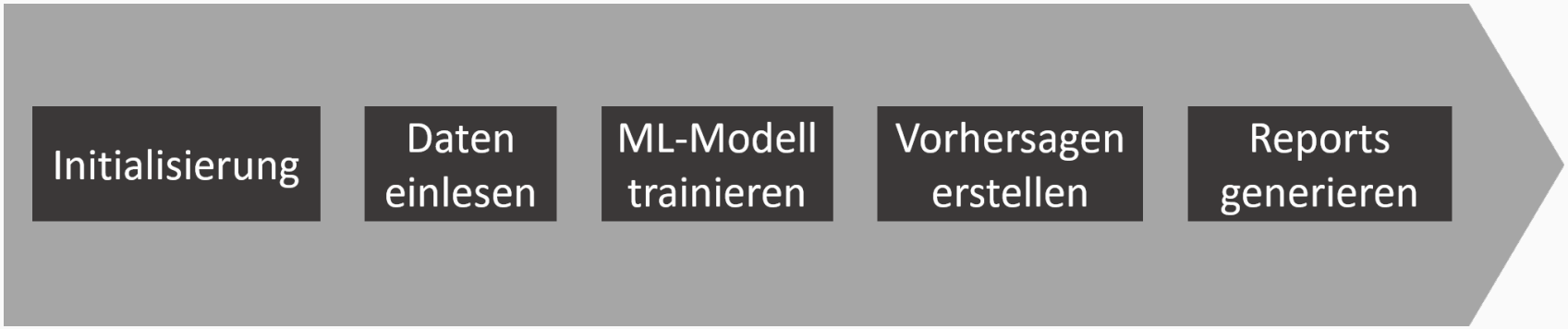
- Berechnet Features aus der Datenbasis
- AST-Parsing
- Modularer Aufbau

ML-Pipeline

Herausforderungen

- Umgang mit grossen Datensets
  - Datenset-Caching
  - Sparse-Matrizen
- Einfache Konfiguration verschiedener ML-Algorithmen
  - Zentrales Config-File
- Resultate müssen bewertet und analysiert werden
  - Verschiedene Charts & Reports

Workflow

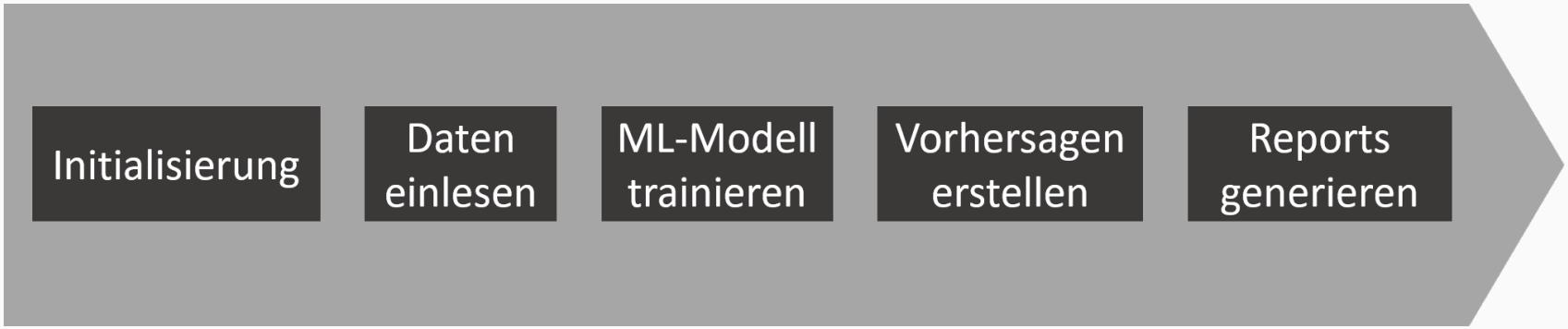

Initialisierung
Daten einlesen
ML-Modell trainieren
Vorhersagen erstellen
Reports generieren

Demo

Testdaten

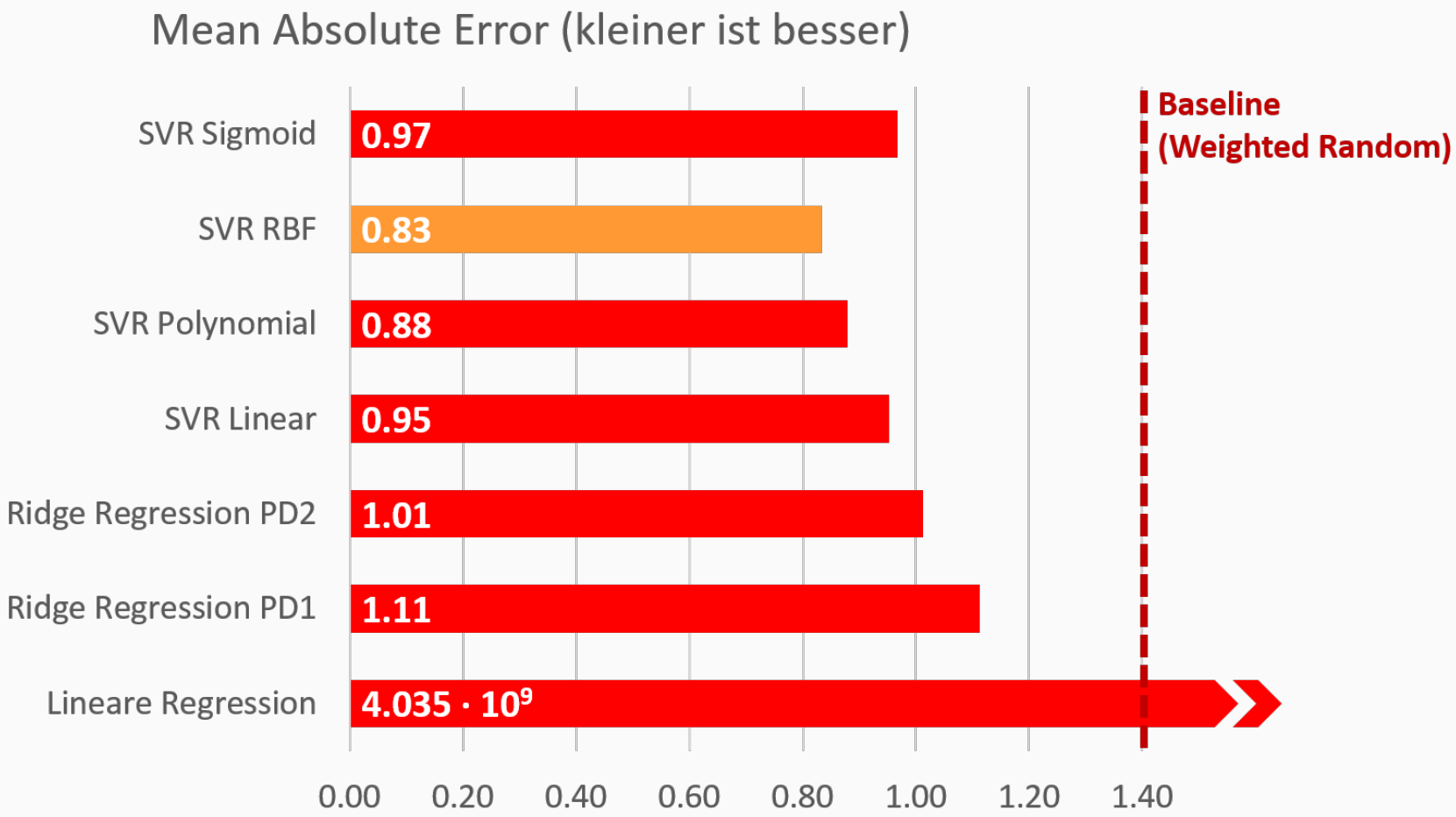
- Projekt: Elasticsearch
  - $>$ 6 Jahre
  - $\sim$ 22 000 Commits
  - $\sim$ 10 000 Issues (GitHub)
- Trainingsset: 01.10.2014 - 31.12.2014
  - 620 Commits
  - 3 414 Dateiversionen
- Testset: 01.01.2015 - 31.01.2015
  - 165 Commits
  - 950 Dateiversionen

Workflow der ML-Pipeline

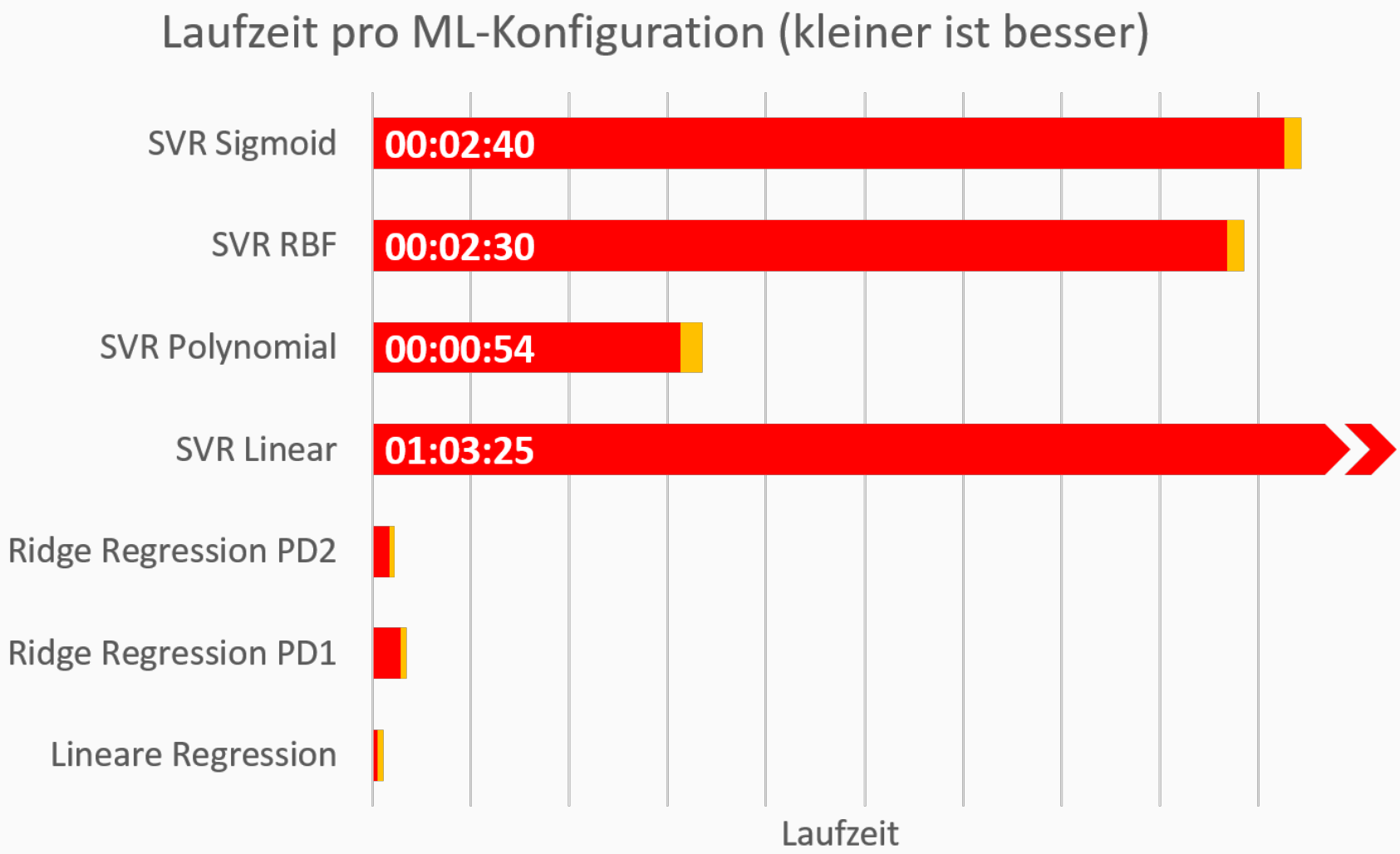


Resultate

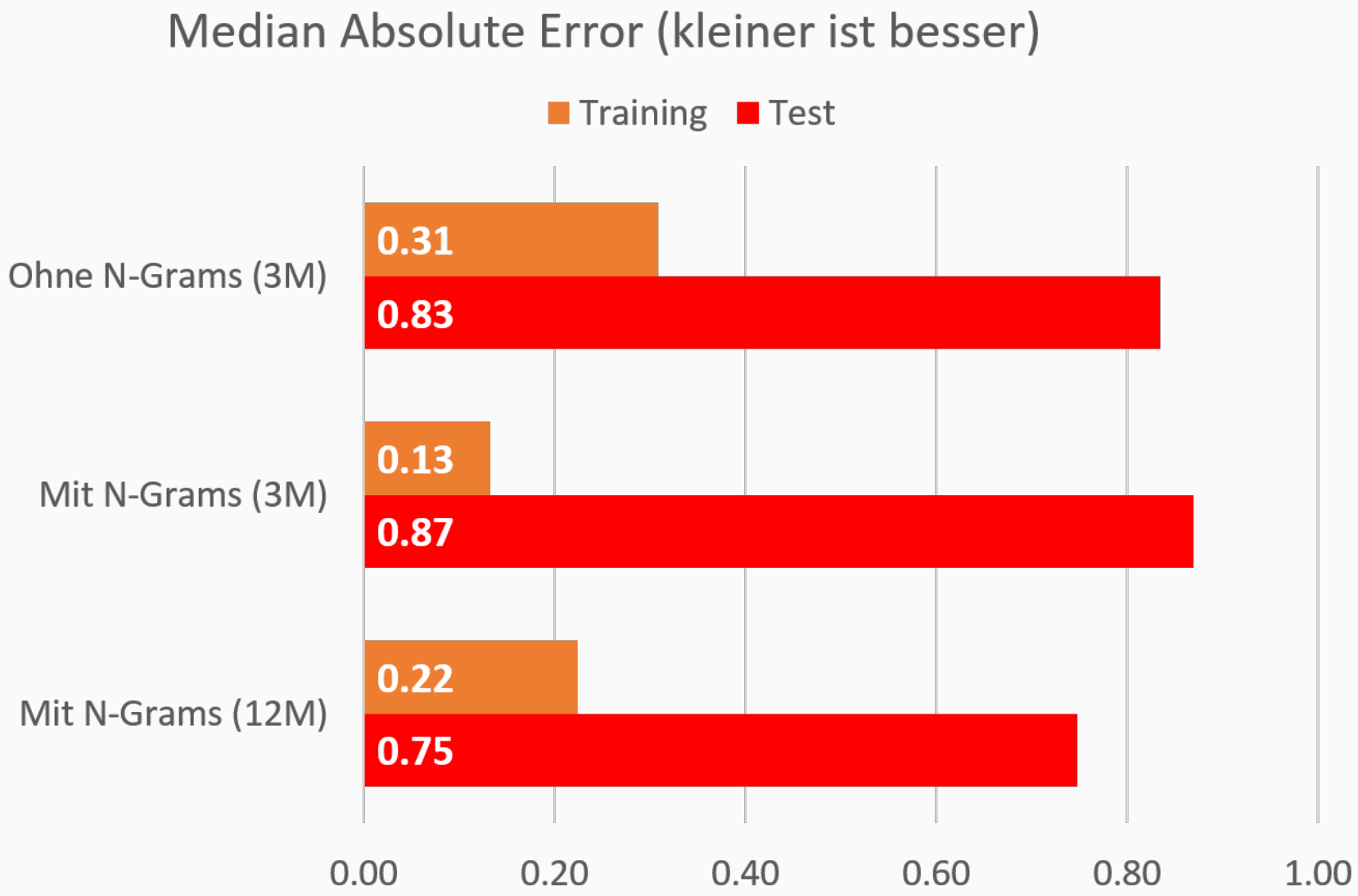
Vergleich von ML-Algorithmen



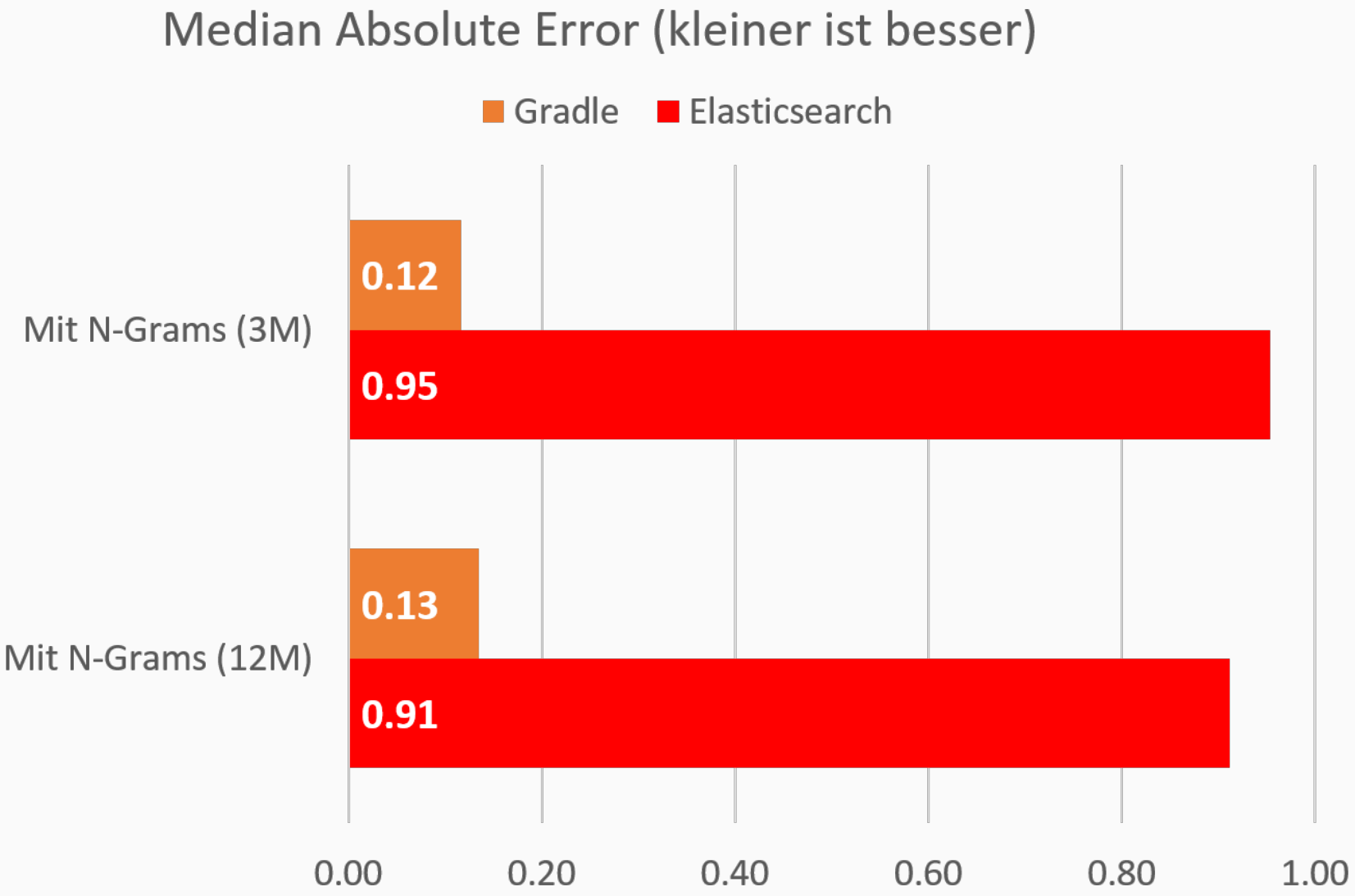
Laufzeit von ML-Algorithmen



Performance von N-Grams



Vergleich von Projekten



Zusammenfassung

Diskussion

- Tool-Set für projektbezogenes Lernen
- Grundlage für weitere Arbeiten
- Textanalyse-Features
- einige Experimente

Ausblick

- Heatmap
- Fehlerlokalisierung
- Andere Programmiersprachen
- Cloud-Anwendung

Fragen und Diskussion