Technical Debt at Scale

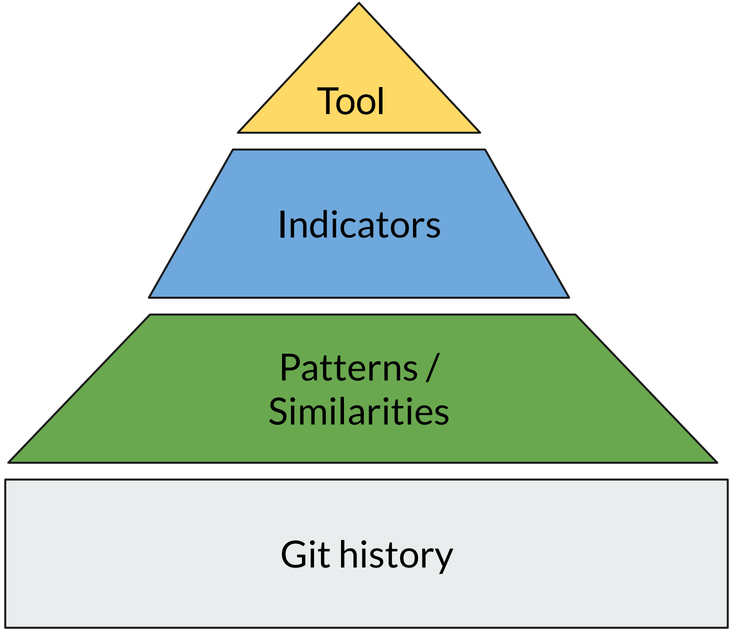
Jonas Grunert - Code Repository Mining - 20.7.2020 - SS 2020

Agenda

1. Recap
2. Data Mining Setup
3. Crawled Data
4. Cluster Analysis
5. Prediction
6. Outlook
7. Future Work

Recap

Building a tool, that warns of tech debt, based upon indicators, which can be found in similarities and patterns in the Git history



Data Mining Setup

Collect Metainformation

- per Commit
- Using PyDriller

- More meta information may be helpful

Calculate TD-Index

- per Commit
- Using Sonarqube

- Long running analysis in comparison to PyDriller

SQALE Calculation

- Built upon different indices
- Calculated by Sonarqube
- Quality Index and Business Index exist
- Quality index used by Sonarqube

Repository Selection

Calculate TD-Index

- Long running repos
- JavaScript Staple Repositories
- At least 2000 commits
- Older than 3 years

Repositories selected

- lodash
- svelte
- rollup
- axios
- parcel

Crawled Data

Commit data

Project, commit hash,
commit message

Author, committer,
dates

Code lines changed
Added lines
Removed lines

Hunks Count
Files changed

Git over time data

Code lines changed
Over 3 commits
Over 5 commits

Added lines
Over 3 commits
Over 5 commits

Removed lines
Over 3 Commits
Over 5 Commits

Contributors
Over 3 commits
Over 5 commits

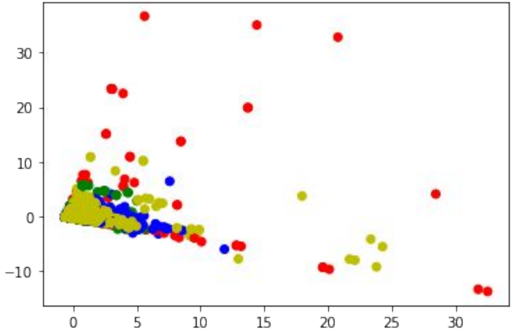
Tech Debt data

Sonarqube data
(SQALE)

Delta Maintainability
Index

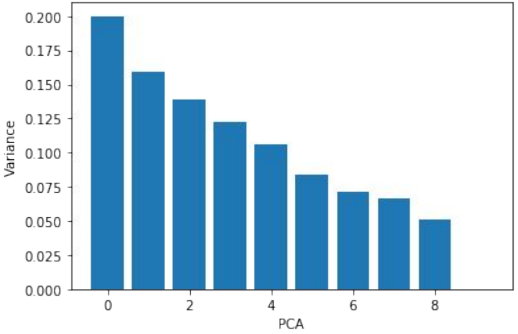
Cluster Analysis

Normalized PCA



Color respond to SQALE rating
No clusters visible

Variance



Low variance on PC1
Low decline in variance

- Unlikely to find a correlation / make a prediction
- More change centric metrics probably needed

Prediction

Predicting SQALE Number or Complexity Number
20% Testdata

Decision tree

- ~60% accuracy for SQALE
- Way lower for complexity (~40%)
- Bump of 20% accuracy with “% comments” and “% duplicated lines”
- Accuracy plateaus with a max depth of 20 nodes

Random forest

- ~60% accuracy for SQALE
- Way lower for complexity (~40%)
- Grid search did enable ~5% accuracy gain

Linear Regression

- R^2: ~0.57
- Errors mostly to the correct trend
- May be better at predicting an increase or decrease

Outlook

Crawler

- More diff based metrics e.g. word count
- Easier to obtain tech debt metric
- More contributor based metrics

Data preperation

- In/decrease of tech debt
- Histogram visualization
- Smaller repositories

Data analysis

- PC1 Variance to about 60%
- Additional visualization
- Prediction of increase or decrease of tech debt

Summary

Already done

- Data gathering
- Complex data analysis
- Simple prediction models

Future work

- Analyze different Tech Debt metrics
- Fasten data gathering
- Analyze prediction
- Develop into a git hook