UNIVERSITÄT KOBLENZ · LANDAU

CHALMERS UNIVERSITY OF TECHNOLOGY

Feature-based defect prediction

using Machine Learning methods

Introductory talk

Winter semester 2019 / 2020

11th December 2019

Stefan Strüder

1 Introduction & Motivation

Introduction & Motivation

CHALMERS
UNIVERSITY OF TECHNOLOGY

- software defects trigger financial loss and reputation damage

Amazon 1p glitch: Software error sees hundreds of items sold for fractions of their value (independent.co.uk, [1])

Retailers say they could be bankrupted by the fault in software that claims to 'auto-optimise' listings

Software

How one bad algorithm cost traders $440m (theregister.co.uk, [2])

A look at the worst software testing day ever

Having 'Null' as a license plate is about as much of a nightmare as you'd expect

The license plate was 'null,' but the tickets were anything but (theverge.com, [3])

Introduction & Motivation

CHALMERS
UNIVERSITY OF TECHNOLOGY

- great interest in tools for detecting faulty code
- development of techniques for default detection and prediction
  - mostly based on methods of machine learning
  - creation of a data set for the training of classifiers
    - data basis: defect-free and faulty historical data
- wide range of learning methods available (Son et al., 2019; Challagulla, Bastani, Yen, & Paul, 2008)
  - Decision Tree based
  - Bayesian methods
  - Regression, k-Nearest-Neighbor, Artificial Neural Networks

Introduction & Motivation

- aim of the thesis
  - prediction technique for software defects
  - in consideration of software features
  - based on methods of machine learning
- promising approach
  - defect prediction based on "past" of the software
  - properties of defect-prone features
  - defect susceptibility of features

2 Background

Machine Learning classification

Defect prediction using Machine Learning

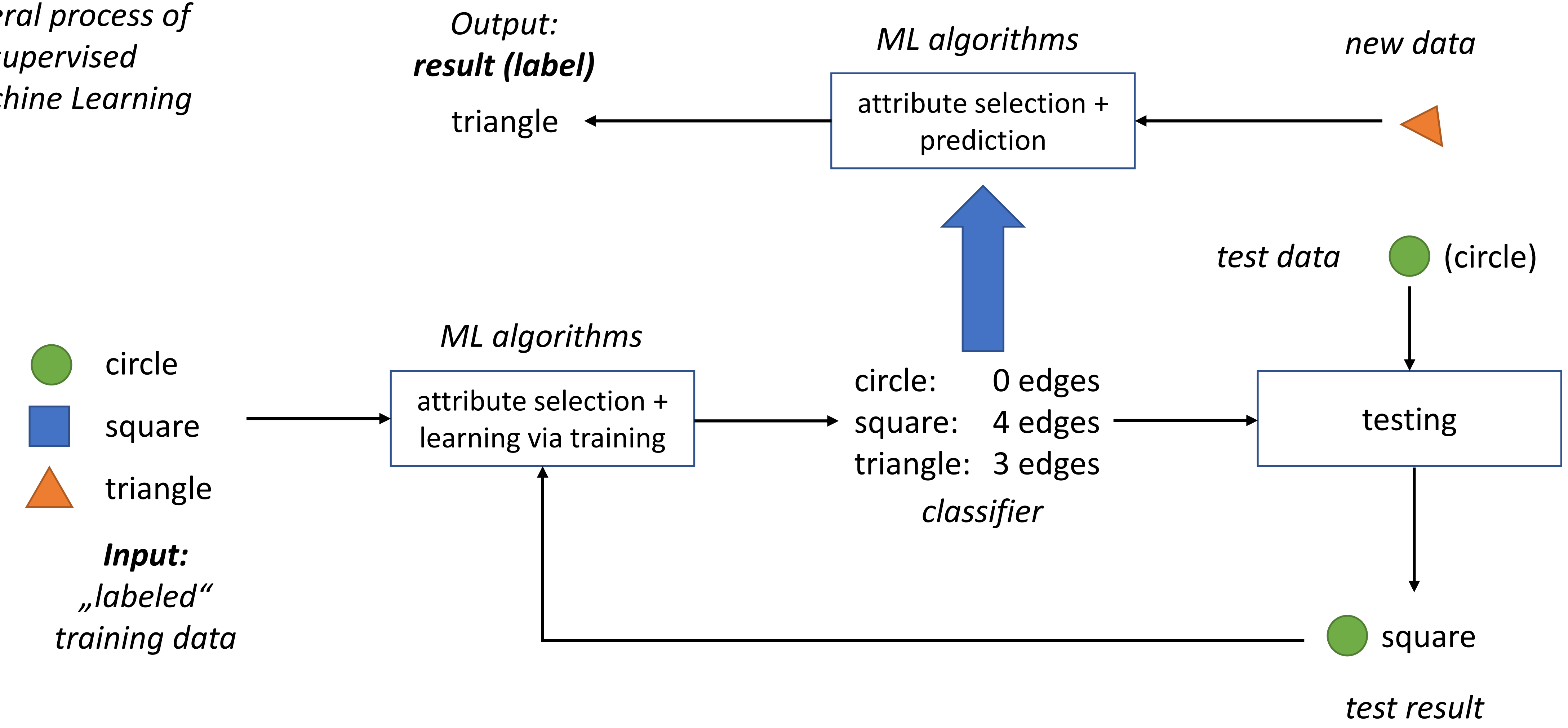
Background | Machine Learning classification

- supervised Machine Learning (Hammouri, Hammad, Alnabhan, & Alsarayrah, 2018)
  - development of a derivation function
    - conclusions from in- and output within a training data set
    - prediction for new input data
  - common algorithms
    - Naïve Bayes
    - Decision Trees
    - Artificial Neural Networks

Background | Machine Learning classification

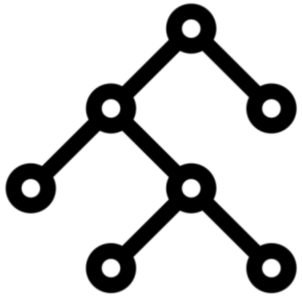
CHALMERS
UNIVERSITY OF TECHNOLOGY

general process of supervised Machine Learning



- Naïve Bayes (Hammouri, Hammad, Alnabhan, & Alsarayrah, 2018)
  - probabilistic classifier
  - based on Bayes theorem →
  - independence of attributes
- decision trees (Hammouri, Hammad, Alnabhan, & Alsarayrah, 2018)
  - hierarchical and predicative
  - attributes of the data as branches
  - decisions as leaf nodes

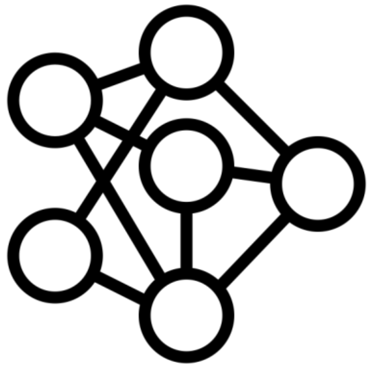
$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$



(source of the icon see bibliography)

Background | Machine Learning classification

- Artificial Neural Networks (Hammouri, Hammad, Alnabhan, & Alsarayrah, 2018; Jukes, 2017)
  - inspired by biological neural networks
  - non-linear classifier
  - consisting of quantity of processing units (neurons)
    - parallel execution for the development of expenditures
    - signal transmission through connections
    - calculations based on the sum of the inputs of all neurons



(source of the icon see bibliography)

Background | ML defect prediction

CHALMERS
UNIVERSITY OF TECHNOLOGY

- defect prediction using Machine Learning
  - Challagulla, V. U. B., Bastani, F. B., Yen, I. L., & Paul, R. A. (2008). **Empirical assessment of machine learning based software defect prediction techniques.** International Journal on Artificial Intelligence Tools, 17(2), 389–400. https://doi.org/10.1142/S0218213008003947
  - Son, L. H., Pritam, N., Khari, M., Kumar, R., Phuong, P. T. M., & Thong, P. H. (2019). **Empirical study of software defect prediction: A systematic mapping.** Symmetry, 11(2). https://doi.org/10.3390/sym11020212
- feature-based defect prediction using Machine Learning
  - Queiroz, R., Berger, T., & Czarnecki, K. (2016). **Towards predicting feature defects in software product lines.** FOSD 2016 - Proceedings of the 7th International Workshop on Feature-Oriented Software Development, Co-Located with SPLASH 2016, 58–62. https://doi.org/10.1145/3001867.3001874

Background | ML defect prediction

feature-based process of supervised machine learning according to Queiroz et. al.

„labeled“ data set based on releases of the tool Busybox			
feature	**release**	**metrics**	**label**
SIGHUP	2.1	12,4,8,9,2	defective
SIGHUP	2.2	14,6,10,6,7	faultless
SIGINT	2.1	5,2,4,6,4	faultless
SIGINT	2.2	6,3,5,10,6	defective
…	…	…	…
SIGQUIT	2.1	8,2,4,3,5	defective
SIGQUIT	2.2	15,5,8,4,8	faultless
…	…	…	…

training data

test data

attribute selection + learning via training

attributes: process metrics

- COMM
- ADEV
- DDEV
- EXP
- OXP

ML algorithms

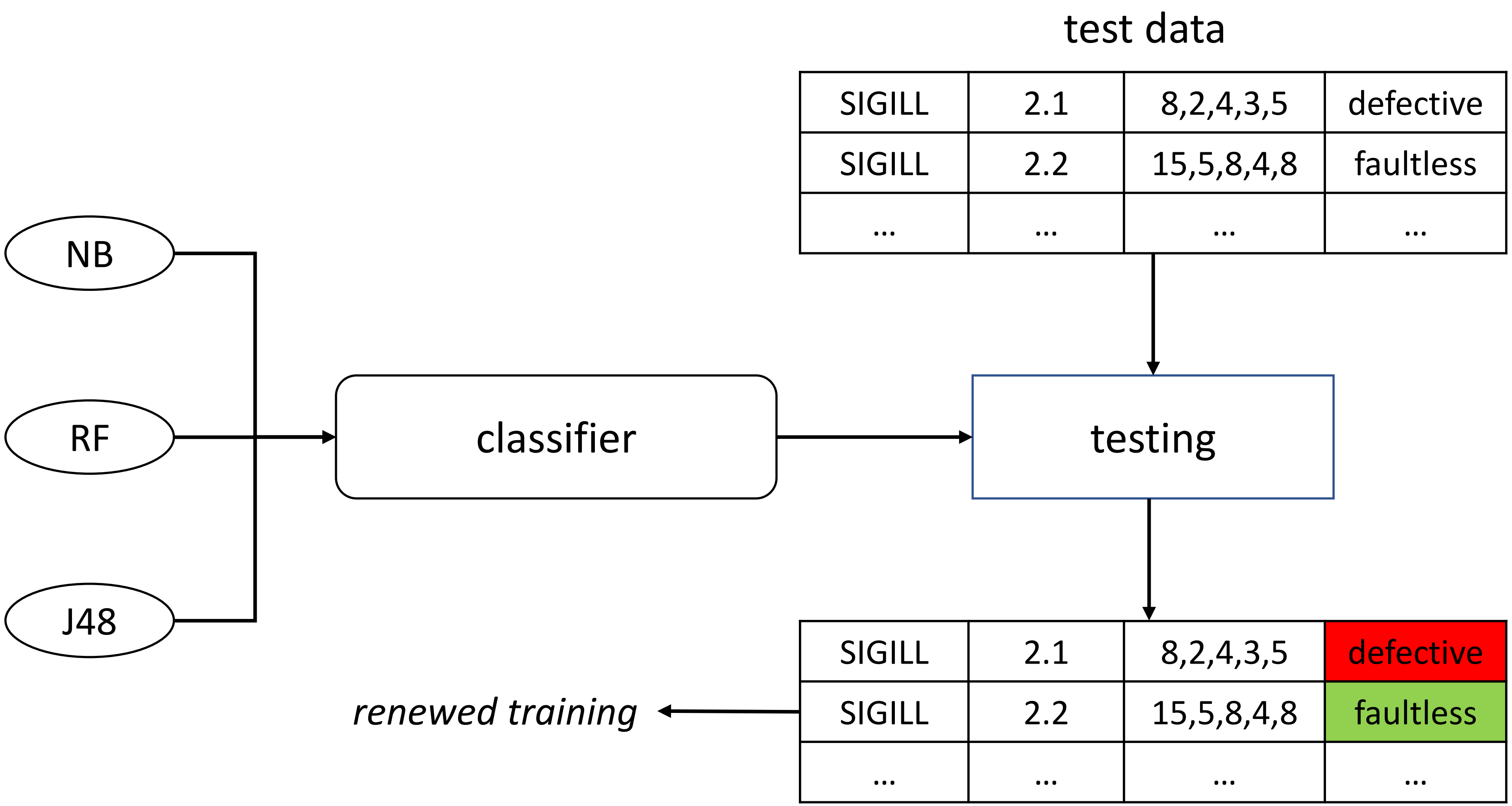
NB
Naïve Bayes

RF
Random Forest

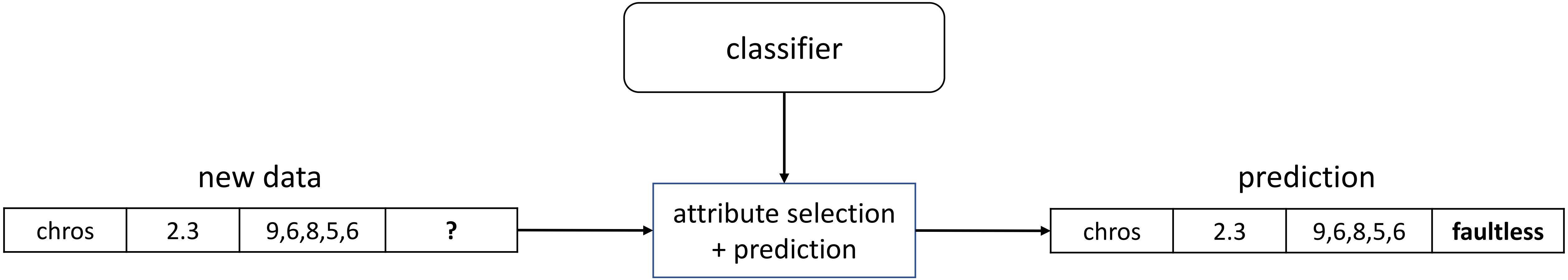
J48
Decision Tree

Background | ML defect prediction

feature-based process of supervised machine learning according to Queiroz et. al.



feature-based process of supervised machine learning according to Queiroz et. al.



attributes: process metrics

- COMM
- ADEV
- DDEV
- EXP
- OXP

3 Goal and approach

Goal

Methodology

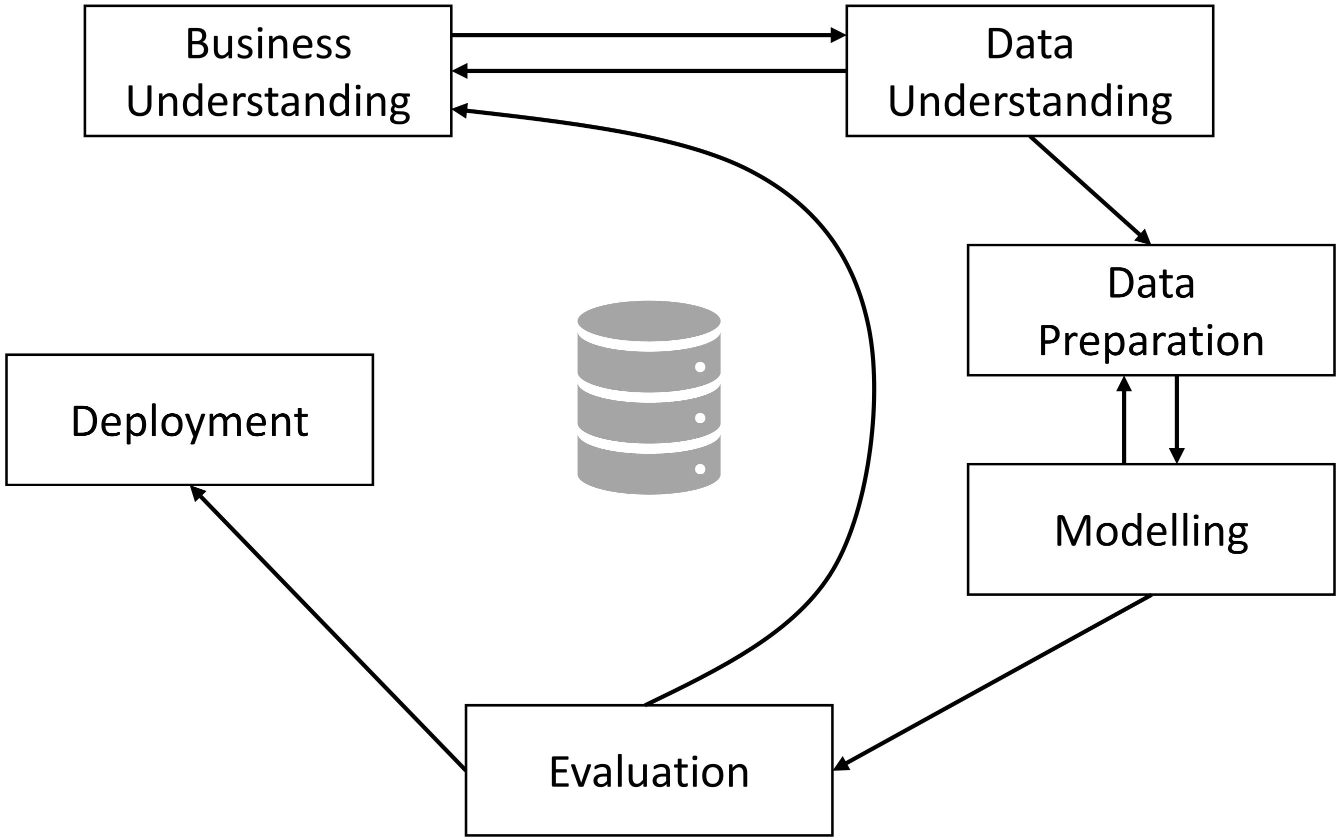
Creation of data set

Goal and approach | Goal

- overarching goal
  - development of a prediction technique for defects in feature-based software
  - using Machine Learning methods
- data basis: commits of versioning systems (Git)
  - commit: provision of an updated version of a software product
  - faulty and defect-free commits for learning classifiers
- three research objectives
  - creation of data set, training of classifiers, evaluation of classifiers
  - + preparation and follow-up

Goal and approach | Methodology

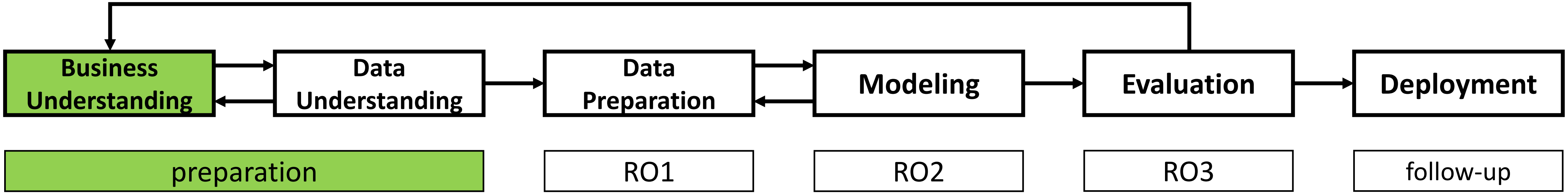
Cross-Industry Standard Process for Data Mining (CRISP-DM) process modell



(Chapman et al., 2000)

Goal and approach | Methodology

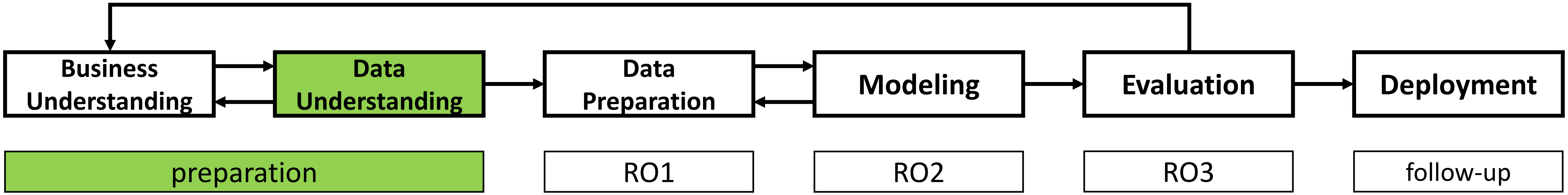
CHALMERS
UNIVERSITY OF TECHNOLOGY



- Business Understanding (preparation)
  - general familiarization with the topic
  - formulation of research objectives

(source of the icon see bibliography)

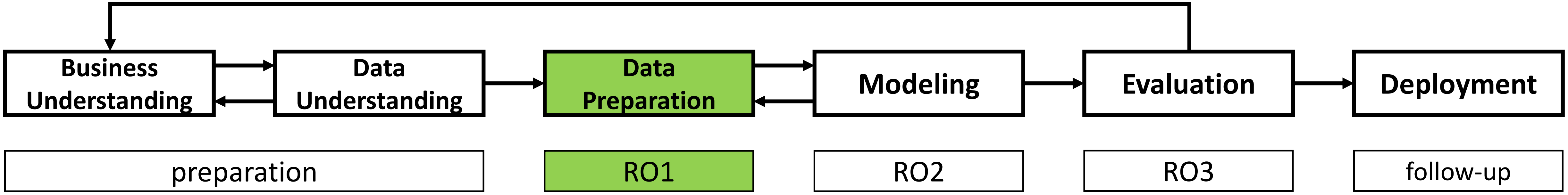
Goal and approach | Methodology



- Data Understanding (preparation)
  - search + review of relevant data and ready-made data sets
  - search focus: Git repositories

(source of the icon see bibliography)

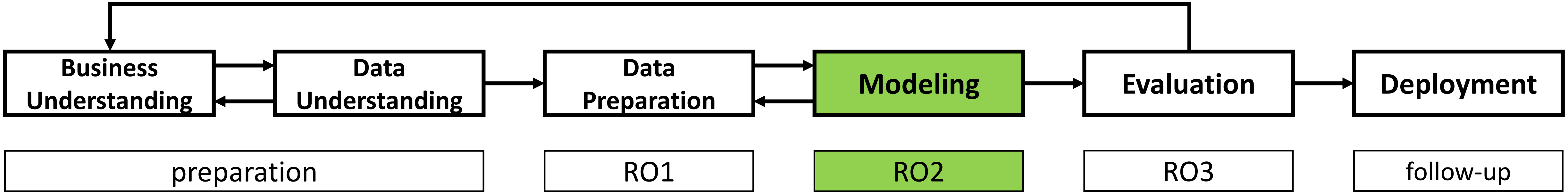
Goal and approach | Methodology



- Data Preparation (research objective 1)
  - processes for optimizing the data set
  - creation of the final data set

(source of the icon see bibliography)

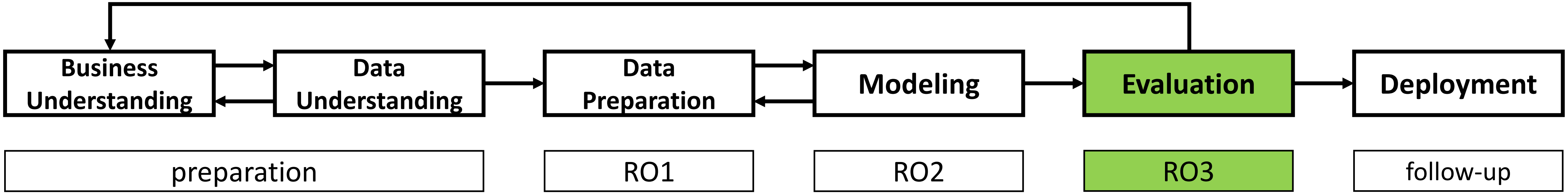
Goal and approach | Methodology



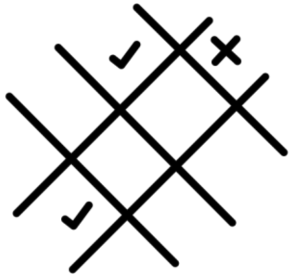
- Modeling (research objective 2)
  - application of the created data set
  - training of Machine Learning algorithms

(source of the icon see bibliography)

Goal and approach | Methodology

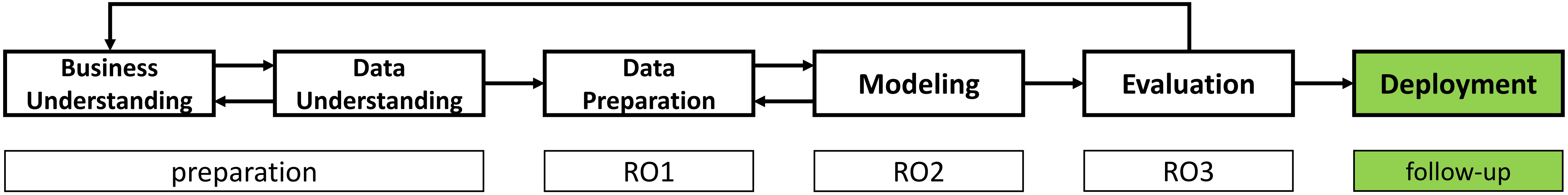


- Evaluation (research objective 3)
  - evaluation of Machine Learning Algorithms
  - comparison of algorithms



(source of the icon see bibliography)

Goal and approach | Methodology

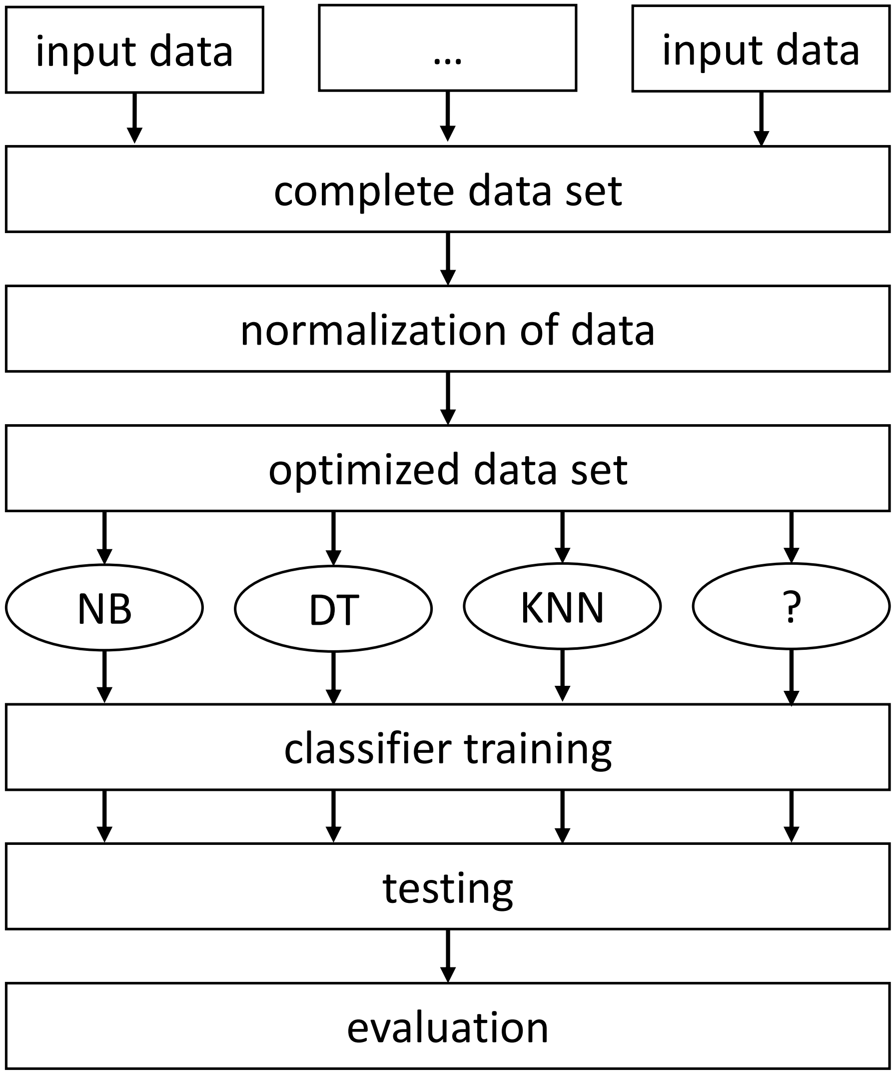


- Deployment (follow-up)
  - preparation of the written elaboration and final presentation
  - holding the colloquium

(source of the icon see bibliography)

Goal and approach | Methodology

applied machine-learning method



commits of feature-based software

consisting of training + test data

Preprocessing, calculation of metrics

metrics as attributes

training of classifiers

application of test data

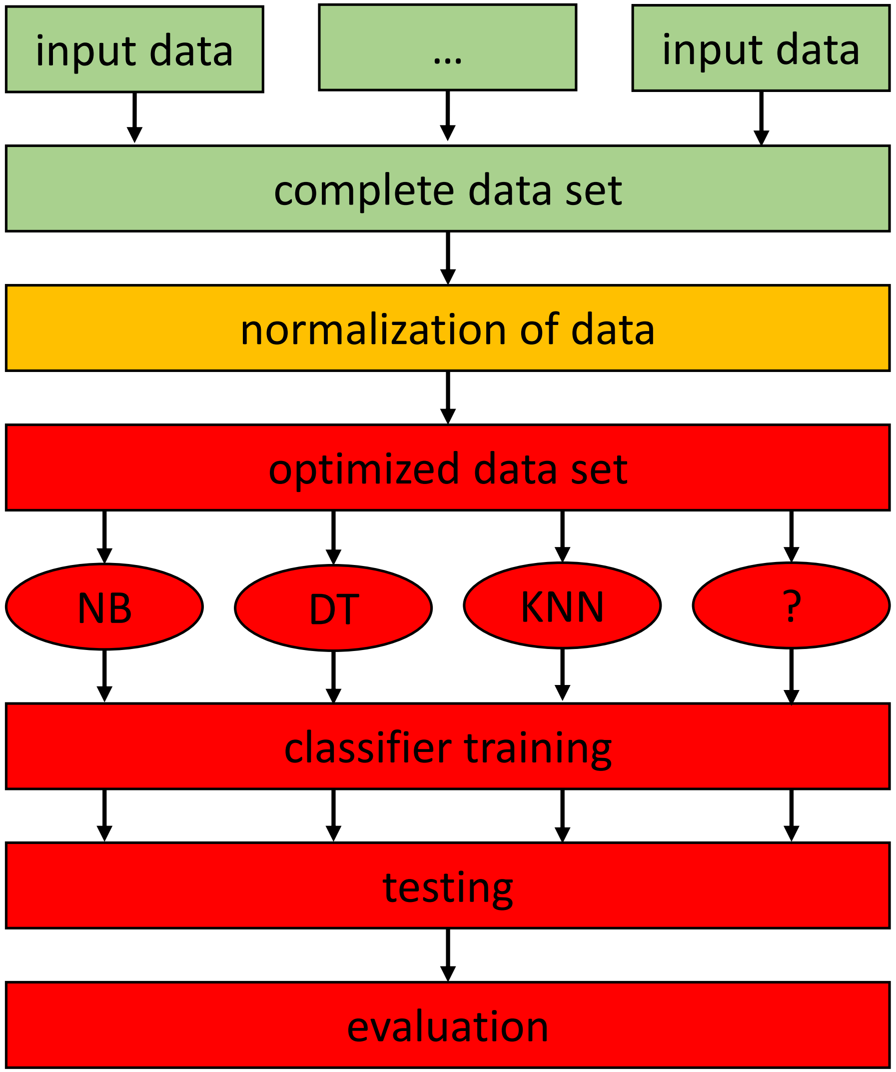
Selection of the most performant classifier

(Ceylan, Kutlubay, & Bener, 2006)

Goal and approach | Methodology

Current state

preparation ✓



commits of feature-based software

consisting of training + test data

Preprocessing, calculation of metrics

metrics as attributes

training of classifiers

application of test data

Selection of the most performant classifier

(Ceylan, Kutlubay, & Bener, 2006)

Goal and approach | Creation of data set

- usage of PyDriller for repository mining
  - Python framework by (Spadini, Aniche, & Bacchelli, 2018)
  - https://github.com/ishepard/pydriller
  - easy extraction of commits, developers, diffs and source code (and more)
  - well documented (https://pydriller.readthedocs.io)

```python
for commit in RepositoryMining("link_to_repo").traverse_commits():
        for m in commit.modifications:
                print(
                     "Author {}".format(commit.author.name),
                     " modified {}".format(m.filename),
                     " with a change type of {}".format(m.change_type.name),
                     " and the complexity is {}".format(m.complexity)
                )
```

Goal and approach | Creation of data set

CHALMERS
UNIVERSITY OF TECHNOLOGY

- historical data of 12 software projects (preprocessor conditionals)
  - feature-based software
    - based on preprocessor conditionals
  - selection criterion: previous use in literature (Hunsen et al., 2016; Liebig et al., 2010; Queiroz et al., 2017)
  - extracted from Git, GitHub, GitLab and Sourceforge repositories
  - divided into commits per release
    - based on tag structure of git repositories

Blender	Busybox	Emacs	Gimp	Gnumeric	Gnuplot
Irssi	Libxml2	Lighttpd	Mpsolve	Parrot	Xfig

Goal and approach | Creation of data set

	Purpose	Data source	#Releases	#Commits	#Corrective
Blender	3D modelling tool	GitHub mirror	11	19119	3697
Busybox	UNIX tool package	Git repository	13	4593	1447
Emacs	Text editor	GitHub mirror	7	11344	3638
Gimp	Photo editor	GitLab repository	14	7295	1226
Gnumeric	Spreadsheet	GitLab repository	8	6025	1211
Gnuplot	Plotter	GitHub mirror	4	4922	476
Irssi	IRC client	GitHub repository	7	367	72
Libxml2	XML parser	GitLab repository	10	732	359
Lighttpd	Webserver	Git repository	5	2285	1080
Mpsolve	Polynomial solver	GitHub repository	8	668	101
Parrot	VM	GitHub repository	4	2765	735
Xfig	Graphics editor	Sourceforge	7	18	0

Goal and approach | Creation of data set

CHALMERS
UNIVERSITY OF TECHNOLOGY

- data was stored in MySQL database
  - one table for each software project

Column	Description	Column	Description
change_type	Type of change (added, deleted, modified, renamed)	lines_added	number of lines added to file
commit_author	responsible developer	lines_reomved	number of lines removed from file
commit_hash	unique identifier of commit	name	software name
commit_msg	commit message	nloc	lines of code of file
cycomplexity	cyclomatic complexity of changed file	release_number	associated release number based on tags
diff	diff of changed file	status	normal (false) or corrective (true) commit (commit_msg containing "bug", "fix", "error" or "fail")
filename	name of changed file		

Goal and approach | Creation of data set

- next up
  - extract modified features of commits (text mining -> identify `#IFDEF`)
  - choose relevant metrics
    - `COMM ADEV DDEV EXP OXP` (from ML process according to Queiroz et. al.)
    - What other metrics that can represent features could be considered?
    - How to calculate metrics based on available data?
    - Which Machine Learning algorithms should be considered?

Thank you for your attention.

Time for questions.

Bibliography

CHALMERS
UNIVERSITY OF TECHNOLOGY

Literature

- Ceylan, E., Kutlubay, F. O., & Bener, A. B. (2006). Software defect identification using machine learning techniques. Proceedings - 32nd Euromicro Conference on Software Engineering and Advanced Applications, SEAA, 240–246. https://doi.org/10.1109/EUROMICRO.2006.56
- Challagulla, V. U. B., Bastani, F. B., Yen, I. L., & Paul, R. A. (2008). Empirical assessment of machine learning based software defect prediction techniques. International Journal on Artificial Intelligence Tools, 17(2), 389–400. https://doi.org/10.1142/S0218213008003947
- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). CRISP-DM 1.0. CRISP-DM Consortium, 76. https://doi.org/10.1109/ICETET.2008.239

Bibliography

CHALMERS
UNIVERSITY OF TECHNOLOGY

Literature

- Hammouri, A., Hammad, M., Alnabhan, M., & Alsarayrah, F. (2018). Software Bug Prediction using machine learning approach. International Journal of Advanced Computer Science and Applications, 9(2), 78–83. https://doi.org/10.14569/IJACSA.2018.090212
- Hunsen, C., Zhang, B., Siegmund, J., Kästner, C., Leßenich, O., Becker, M., & Apel, S. (2016). Preprocessor-based variability in open-source and industrial software systems: An empirical study. Empirical Software Engineering (Vol. 21). https://doi.org/10.1007/s10664-015-9360-1
- Liebig, J., Apel, S., Lengauer, C., Kästner, C., & Schulze, M. (2010). An analysis of the variability in forty preprocessor-based software product lines. Proceedings - International Conference on Software Engineering, 1, 105–114. https://doi.org/10.1145/1806799.1806819
- Queiroz, R., Passos, L., Valente, M. T., Hunsen, C., Apel, S., & Czarnecki, K. (2017). The shape of feature code: an analysis of twenty C-preprocessor-based systems. Software and Systems Modeling, 16(1), 77–96. https://doi.org/10.1007/s10270-015-0483-z

Bibliography

CHALMERS
UNIVERSITY OF TECHNOLOGY

Literature

- Queiroz, R., Berger, T., & Czarnecki, K. (2016). Towards predicting feature defects in software product lines. FOSD 2016 - Proceedings of the 7th International Workshop on Feature-Oriented Software Development, Co-Located with SPLASH 2016, 58–62. https://doi.org/10.1145/3001867.3001874
- Son, L. H., Pritam, N., Khari, M., Kumar, R., Phuong, P. T. M., & Thong, P. H. (2019). Empirical study of software defect prediction: A systematic mapping. Symmetry, 11(2). https://doi.org/10.3390/sym11020212
- Spadini, D., Aniche, M., & Bacchelli, A. (2018). PyDriller: Python framework for mining software repositories. In Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2018 (pp. 908–911). New York, New York, USA: ACM Press. https://doi.org/10.1145/3236024.3264598

Bibliography

CHALMERS
UNIVERSITY OF TECHNOLOGY

Online sources

[1] https://www.independent.co.uk/news/uk/home-news/amazon-1p-glitch-software-error-sees-hundreds-of-items-sold-for-fractions-of-their-value-9923730.html (independent.co.uk, 14.12.2014)

[2] https://www.theregister.co.uk/2012/08/03/bad_algorithm_lost_440_million_dollars/ (theregister.co.uk, 03.08.2012)

[3] https://www.theverge.com/tldr/2019/8/14/20805543/null-license-plate-california-parking-tickets-violations-void-programming-bug (theverge.com, 14.08.2019)

Bibliography

CHALMERS
UNIVERSITY OF TECHNOLOGY

Icon sources

- Reading by Arafat Uddin from the Noun Project
- Data Analysis by Brennan Novak from the Noun Project
- Data by fizae from the Noun Project
- Machine Learning by Juicy Fish from the Noun Project
- evaluation by Michael Rojas from the Noun Project
- Writing by Kmg Design from the Noun Project