Towards Base Rates in Software Analytics

Early results and challenges from studying Ohloh

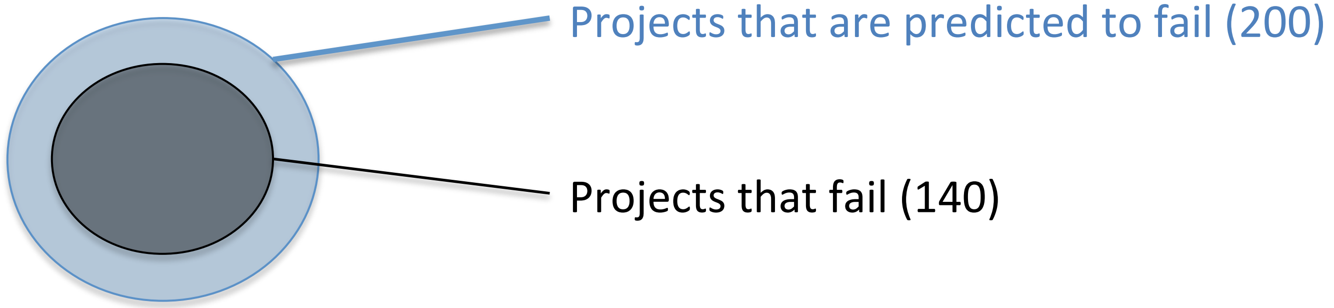
Magiel Bruntink (Uni. van Amsterdam)

Benevol 2013

I agree with Harald and Gregorio, so:
Find my data, code and replication details (Wiki) on
github.com/OhlohAnalytics

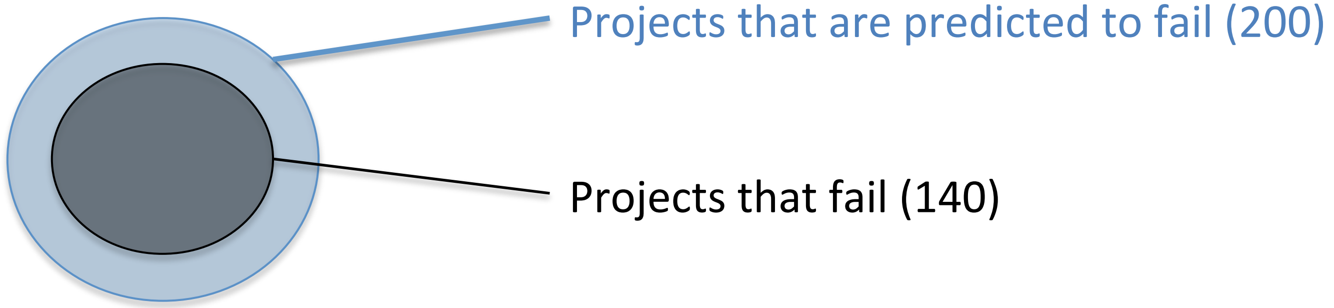
Why should we care about base rates in software analytics?

"Our research on 300 projects shows that our method predicts project failure with 70% precision."



Why should we care about base rates in software analytics?

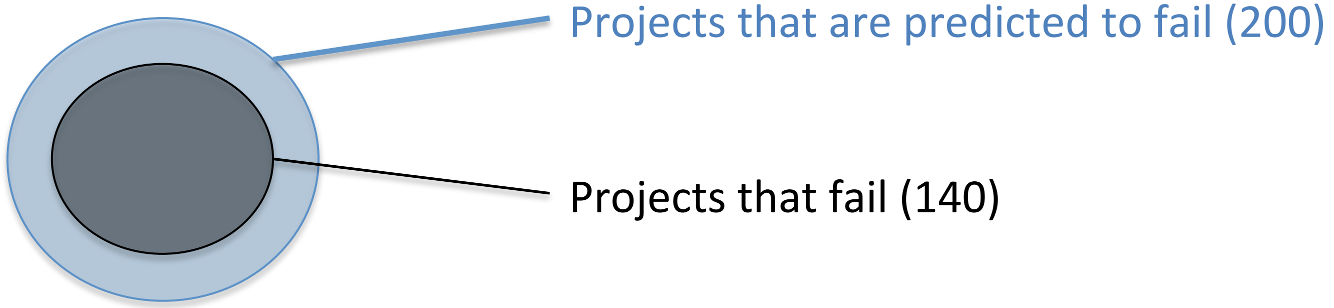
Let's say the method says "your project will fail."
What is the chance of failure?



Why should we care about base rates in software analytics?

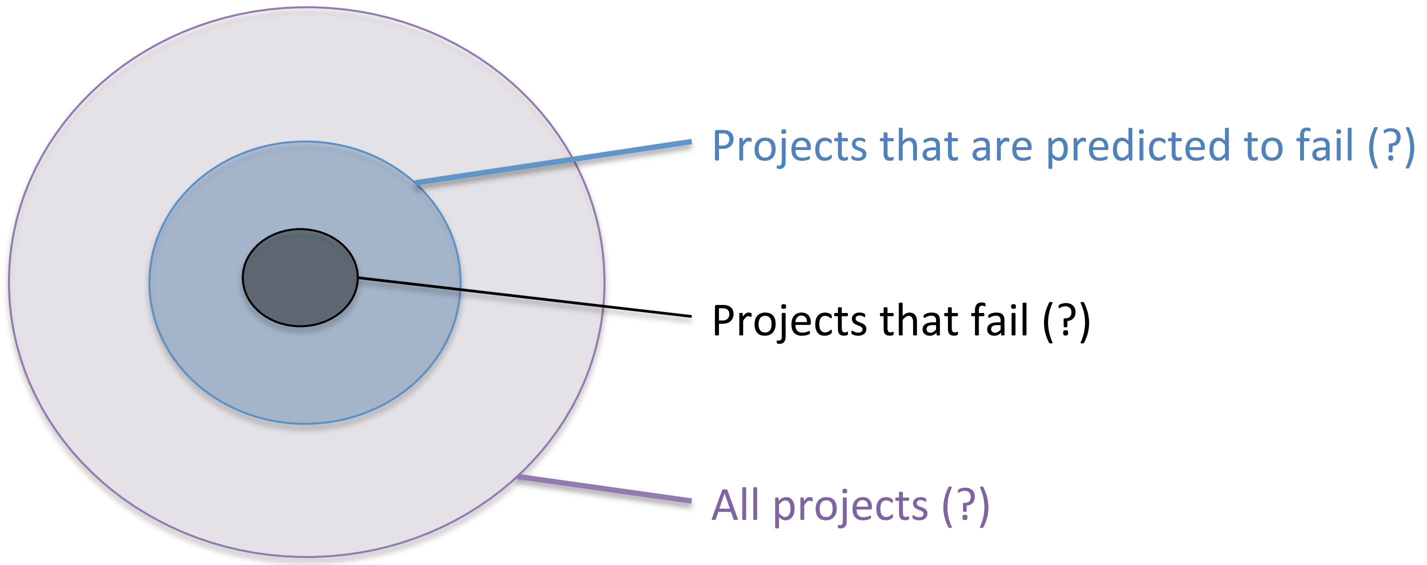
70% is not the right answer!

This is the *base rate fallacy.*



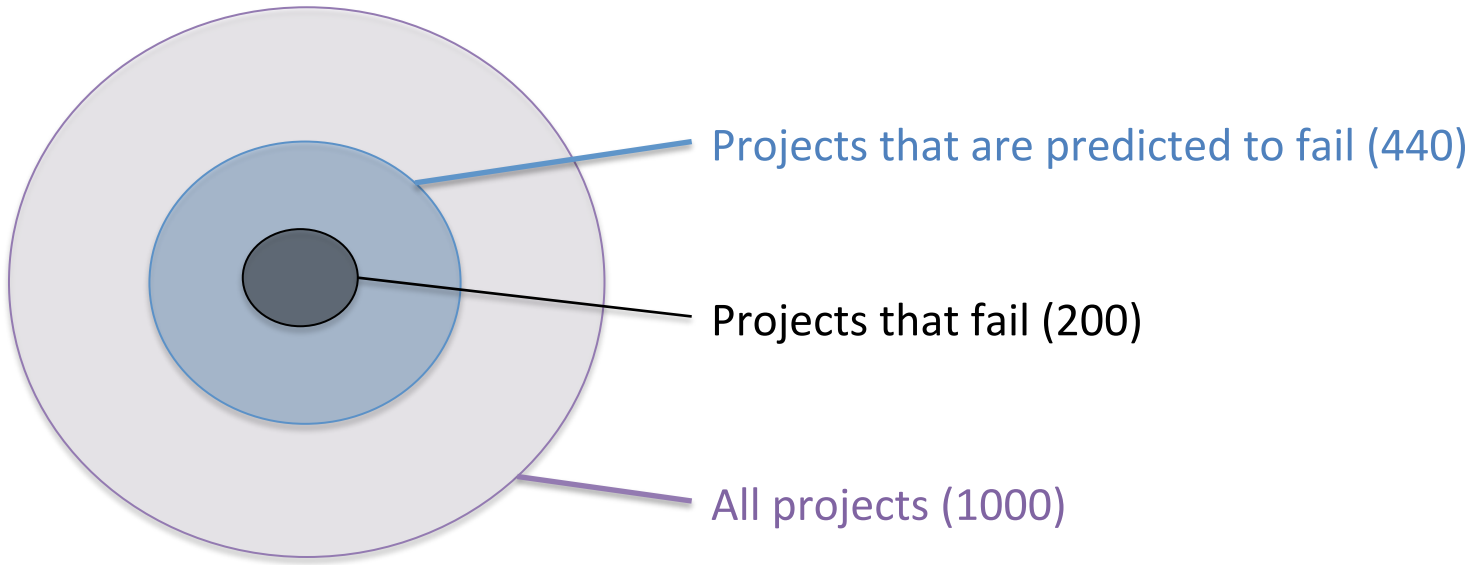
Why should we care about base rates in software analytics?

You have to take into account the *base rate*:
"How many projects fail in the first place?"



Why should we care about base rates in software analytics?

Projects that fail out of 1000:	200 (example base rate)
Projects that do not fail but test positive:	(1000-200) * (100%-70%) = 240
Project failure chance given positive test:	200 / (200+240) = **45% or less**



What is/are software analytics?

Software Analytics is a research focussing on:

- Data on software artefacts (code, documents) and projects (people, activities)
- Appropriate statistics and data-driven methods
- Actionable insight to users, developers, decision makers, etc.

Example of software analytics in practice:

- Software Improvement Group

Don’t we know these things?

It turns out, there is work in this area:

- “Survival analysis on the duration of open source projects”, Samoladas et. al., 2010
- “Reclassifying Success and Tragedy in FLOSS Projects”, Wiggins and Crowston, 2010
- “A statistical examination of the properties and evolution of libre software”, Herraiz Tabernero, 2008
- “Software Assessments, Benchmarks and Best Practices”, Capers Jones, 2007
- …

But it can, and should, be extended further:

- Data sets are small, or focussed on 1-10 languages (most research)
- Or focussed on only open source (most research)
- Or data sets are not available at all (Capers Jones)

How to start obtaining base rates?

Through large-scale data collection and careful analysis

Leveraging the ocean of data now available on open source software (600 K projects on Ohloh)

Researching the application of analyses from other fields such as medicine or economics

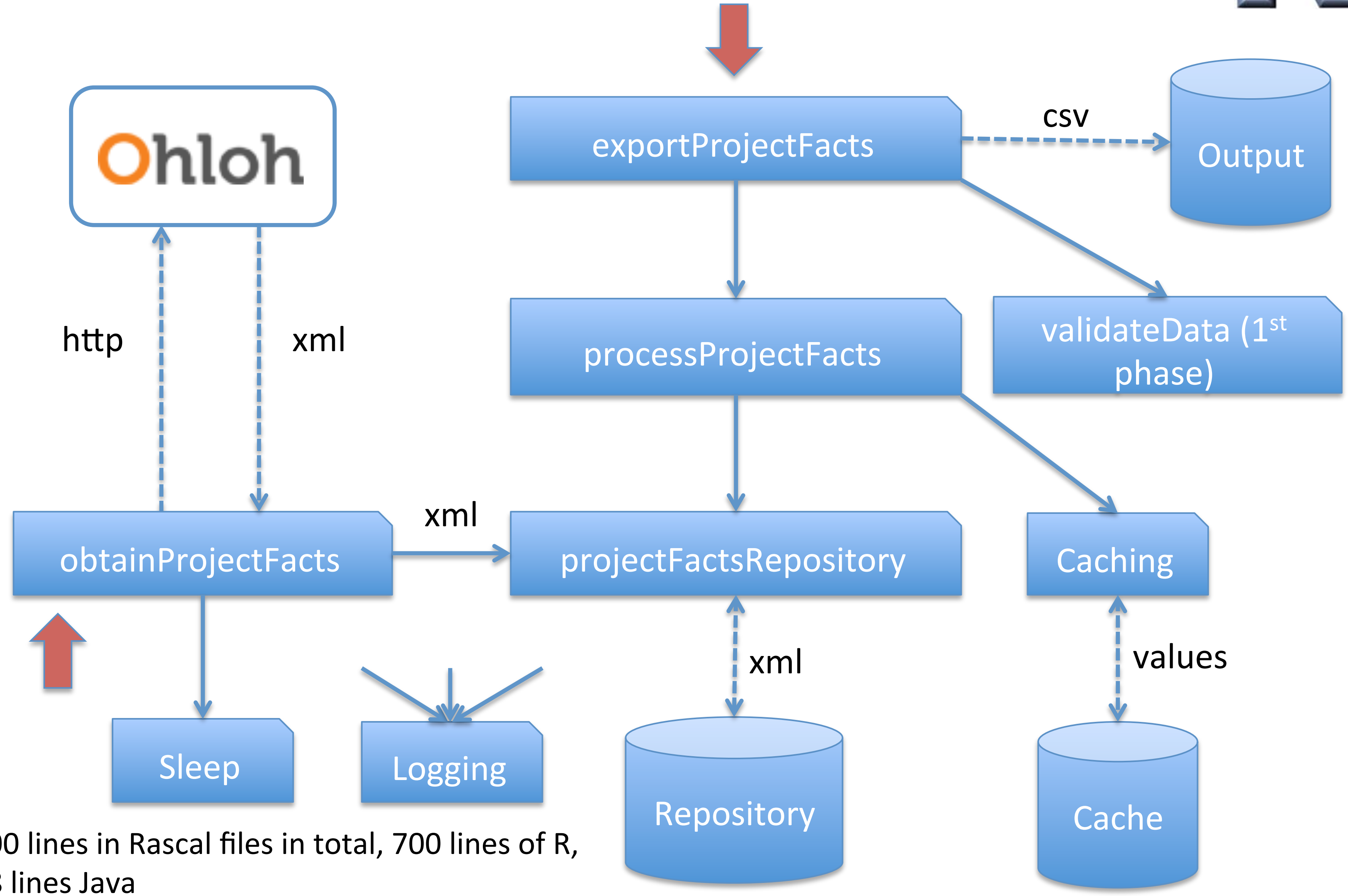
Gathering some data

Work in progress: collecting and analysing data from Ohloh http://www.ohloh.net/

Data set: monthly record of history for 12,360 open source projects

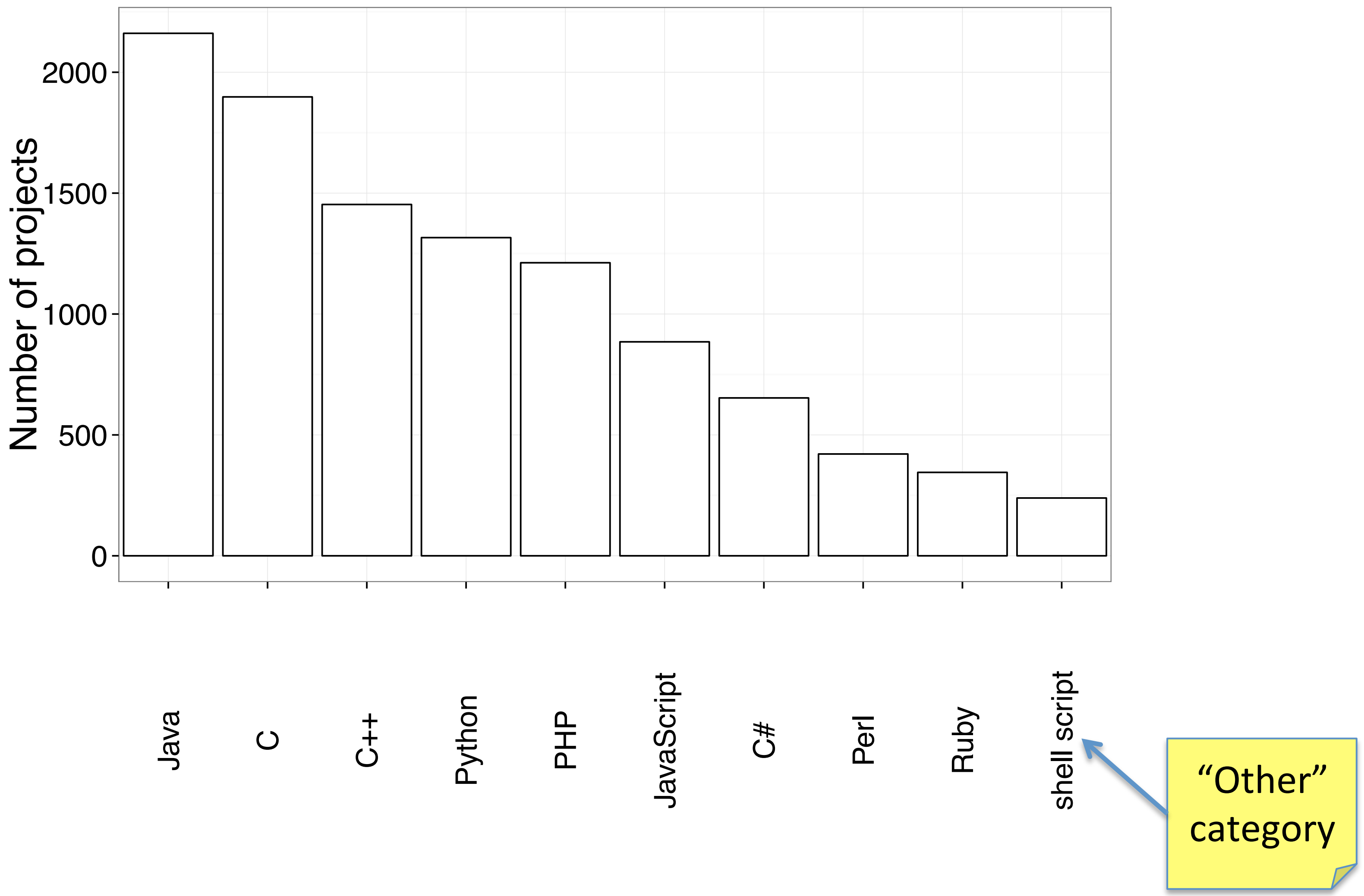
- Code/comments/blanks added, deleted, total
- Number of contributors
- Number of commits
- Main programming language
- Other meta data

Architecture



700 lines in Rascal files in total, 700 lines of R, 18 lines Java

Data set by 10 most popular main programming languages on Ohloh



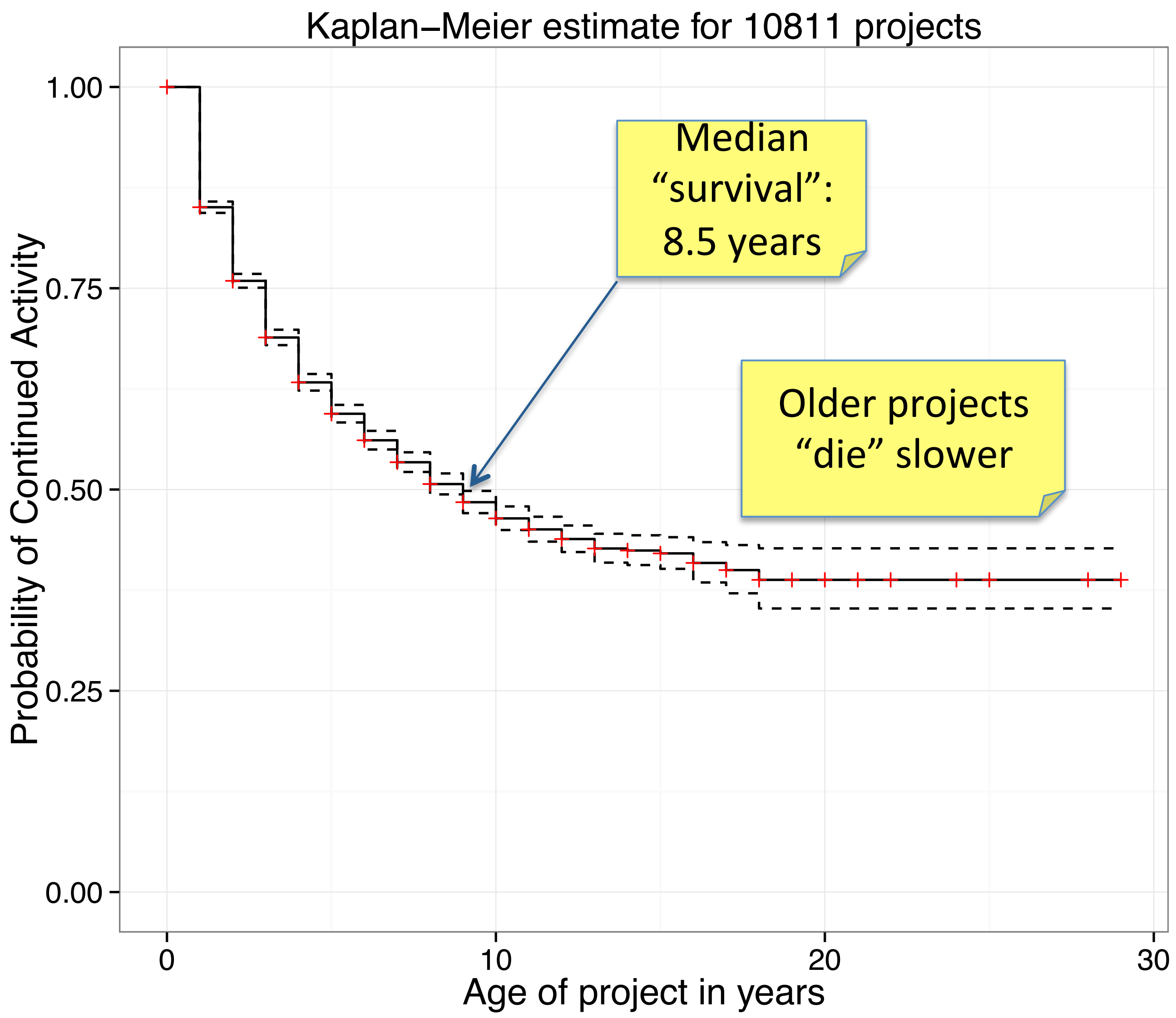
Early results: *Project inactivity*

Question: *What is the rate of OSS projects becoming inactive?*

Metric: *Probability of Continued Activity*

– Measured by a Kaplan-Meier estimate based on (right-censored) inactivity events. A project is considered to suffer from inactivity if it has 0 commits in a year (of age).

Early results: *Project inactivity*



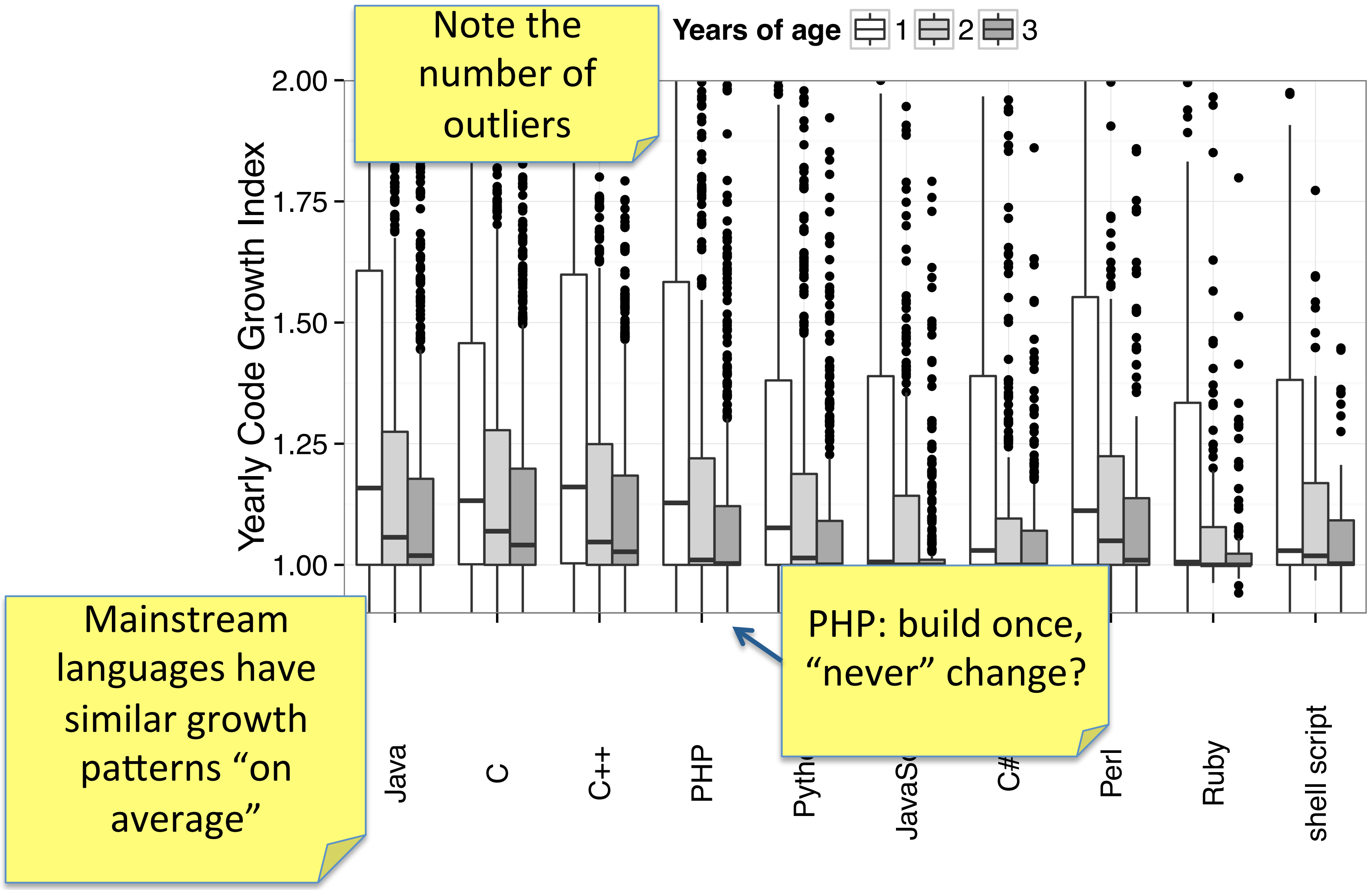
Early results: *Code growth*

Question: *What is the yearly code growth rate of OSS projects?*

Metric: *Indexed Code Growth*

- An index of the code size at the end of a year compared to the beginning of the year. For example, a value of 1.05 represents 5% code growth since the beginning of the year.

Early results: *Code growth*



The 10 most used main programming languages in the data set

Challenge: *data quality*

Initial investigation into the Ohloh data reveals that at least 15% of the cases are suspect or wrong

- Inconsistencies (LOC do not always add up?)
- Implausibilities (LOC negative?)
- Source repositories badly configured (SVN)
- Missing data
- Events where code is moved or imported

How to deal with this problem for big software datasets?

- Looking into machine learning tools (e.g., Gaussian processes) to automate detection of issues
- Build up a manually-verified sample and compare
- Cross-validate with other datasets

Challenge: *beyond open source*

Base rates should not be limited to only open source software

How to obtain sufficient industrial software data? *This is a call to industry to share data.*

Question: Are industrial software and open source software (projects) really different?

Conclusion

Base rates are needed to avoid fallacies

Open source data enables doing this research

Challenges ahead, and help of industry is needed!

Thanks! Questions?

Find the data, code and replication details on

github.com/OhlohAnalytics

Magiel Bruntink

University of Amsterdam

M.Bruntink@uva.nl

020 525 8201