LTAT.05.008:
Software Analytics

Ezequiel Scott
ezequiel.scott@ut.ee
Motivation
Motivation

How to take advantage of the data?

How to support decisions?
Data analytics
Some definitions...

**Data Science** is a field that comprises of everything that related to data cleansing, preparation, and analysis.

**Big Data** is something that can be used to analyze insights which can lead to better decision and strategic business moves.

**Data Analytics** involves automating insights into a certain dataset as well as supposes the usage of queries and data aggregation procedures.
Analytic Maturity Continuum
*analytics

• Web analytics
• Game analytics
• Business analytics
• Learning analytics
• Text analytics
• Analytics

... 

• Software analytics – why not?
Developers tools...
What are the new trends?

Developers
Solo coders → Social coders

Software development
Code centric → Data centric

Visualization
Standalone → Ubiquitous
The (r)evolution of social media in software engineering.

Defining Software Analytics (SA)

- **SA** refers to analytics specific to the domain of **software systems** taking into account **source code**, **static** and **dynamic characteristics** as well as related **processes** of their development and evolution. [...] Similar to other domain-specific analytics fields (e.g., business analytics), it is based on analyzing big data commonly managed, for example, by IDEs and software repositories.” – Wikipedia

- **SA** aims to obtain insightful and actionable information from **software artifacts** that help practitioners accomplish tasks related to **software development**, systems, and users.” – D. Zhang et al. "Software Analytics in Practice". IEEE Software

- **SA** is analytics on **software data** for managers and software engineers with the aim of empowering software development individuals and teams to gain and share insight form their data to make better decisions.” – R. Buse, T. Zimmermann. "Information Needs for Software Development Analytics". ICSE, 2012
Applications of Software Analytics

- combining software product information with apps store data\(^1\);\(^2\);
- using process data to predict overall project effort\(^3\);
- using software process models to learn effective project changes\(^4\);
- using operating system logs that predict software power consumption\(^5\);
- exploring product line models to configure new applications\(^6\);
- mining natural language requirements to find links between components\(^7\);
- mining performance data\(^8\);\(^9\);
- using XML descriptions of design patterns to recommend particular designs\(^10\);
- using email lists to understand the human networks inside software teams\(^11\);
- linking emails to source code artifacts and classifying their content\(^12\);
- using execution traces to learn normal interface usage patterns\(^13\);
- using bug databases to learn defect predictors that guide inspection teams to where the code is most likely to fail\(^14\)–\(^16\) and to classify changes as clean or buggy\(^17\);
- using security data to identify indicators for software vulnerabilities\(^18\);
- using visualization to support program comprehension\(^19\);
- using software ontologies to enable natural language queries\(^20\); and
- mining code clones to assess the implications of cloning and copy/paste in software.\(^21\);\(^22\)

How to support software analytics?

We rely on the process of knowledge discovery from data (KDD)

How to support software analytics?

We rely on the process of knowledge discovery from data (KDD)

KD1: data selection and data pre-processing

• Data selection
  where data relevant to the analysis task are retrieved

• Data integration*
  where multiple data sources may be combined

• Data cleaning*
  to remove noise and inconsistent data

(*) part of the pre-processing
Data selection – What sources are relevant to Software Analytics?
Development environment – Example #1
Development environment – Example #2

1. check in
2. indicate change
3. check out
4. report results
5. set status
6. notify

version control system (e.g. Subversion)

developers

build servers

feedback

CI server (Hudson)

developers

team leads, testers, clients, management, etc.

LTAT.05.008 / Software Analytics / © Ezequiel Scott 2018
Development environment – Example #3
## Data sources

### Repositories of software engineering data.

<table>
<thead>
<tr>
<th>Repository</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug Prediction Dataset</td>
<td><a href="http://bug.inf.usi.ch">http://bug.inf.usi.ch</a></td>
</tr>
<tr>
<td>Eclipse Bug Data</td>
<td><a href="http://www.st.cs.uni-saarland.de/softevo/bug-data/eclipse">www.st.cs.uni-saarland.de/softevo/bug-data/eclipse</a></td>
</tr>
<tr>
<td>FLOSSMetrics</td>
<td><a href="http://flossmetrics.org">http://flossmetrics.org</a></td>
</tr>
<tr>
<td>FLOSSMole</td>
<td><a href="http://flossmole.org">http://flossmole.org</a></td>
</tr>
<tr>
<td>International Software Benchmarking Standards Group (IBSBG)</td>
<td><a href="http://www.isbsg.org">www.isbsg.org</a></td>
</tr>
<tr>
<td>ohloh</td>
<td><a href="http://www.ohloh.net">www.ohloh.net</a></td>
</tr>
<tr>
<td>PROMISE</td>
<td><a href="http://promisedata.googlecode.com">http://promisedata.googlecode.com</a></td>
</tr>
<tr>
<td>Qualitas Corpus</td>
<td><a href="http://qualitascorpus.com">http://qualitascorpus.com</a></td>
</tr>
<tr>
<td>Software Artifact Repository</td>
<td><a href="http://sir.unl.edu">http://sir.unl.edu</a></td>
</tr>
<tr>
<td>SourceForge Research Data</td>
<td><a href="http://zerlot.cse.nd.edu">http://zerlot.cse.nd.edu</a></td>
</tr>
<tr>
<td>Sourcerer Project</td>
<td><a href="http://sourcerer.ics.uci.edu">http://sourcerer.ics.uci.edu</a></td>
</tr>
<tr>
<td>Tukutuku</td>
<td><a href="http://www.metriq.biz/tukutuku">www.metriq.biz/tukutuku</a></td>
</tr>
<tr>
<td>Ultimate Debian Database</td>
<td><a href="http://udd.debian.org">http://udd.debian.org</a></td>
</tr>
</tbody>
</table>

GitHub – What data can be extracted?

- Commits
- Pull requests
- Issues...
GitHub – How to extract data?

• Web scraping
• Using GitHub official API
  • REST
  • JAVA
  • Python
• GHTorrent
• Google Big Data Platform
• Microsoft Power BI
GitHub – Web scraping

• Web scraping a web page involves **fetching** it and **extracting** from it.

• Typically it refers to automated processes implemented using a **bot** or **web crawler**.

↓ Difficult to analyse

↓ Time consuming

↓ Legal issues

↓ Methods to prevent scraping
GitHub – Using official API

A **RESTful API** is a web service implemented using HTTP protocol and the principles of REST. It is a collection of resources that employ HTTP methods *(GET, PUT, POST, DELETE)*
GitHub – GitHub API for JAVA

We can use the GitHubDataExtractor project to retrieve data from Github repositories.

• The project relies on the Github API for Java
• You can download the GitHubDataExtractor from here
• Import the project into your favorite Java IDE (e.g. Eclipse) and then add the required libraries to the build path
GitHub – GitHub API for JAVA

• There are two important classes:
  • **RRCalc** – just the main class
  • **CommitDataCollection** – the class in charge of collecting the commit data, it does the hard job

• In **RRCalc**, we set important data up such as the username, repository, the credentials, dates, etc.

• In **CommitDataCollection**, we use the Github API to connect with the Github services and obtain all the data from the repository
GitHub API for JAVA – HOW?

First, we have to create an object for the repository and set the credentials up:

```java
RepositoryService repservice = new RepositoryService();
repservice.getClient()
    .setCredentials(GitCredits[0], GitCredits[1]);

RepositoryId repo = new RepositoryId(repoOwner, repoName);
```
GitHub API for JAVA – HOW? (cont.)

Then, we can use different services for retrieving the data from the repository.

There are three services available: **Commit**, **Issue**, and **Pull**. All of them require credentials.

```java
// For downloading commits
CommitService commitservice = new CommitService();
commitservice.getClient().setCredentials(GitCredits[0], GitCredits[1]);

// For downloading pulls
PullRequestService pullservice = new PullRequestService();
pullservice.getClient()
    .setCredentials(GitCredits[0], GitCredits[1]);
```
Finally, we can retrieve all the data from each service and store it in List objects. It makes finding elements easier to do.

```java
// For downloading commits
List<RepositoryCommit> commitList = commitservice.getCommits(repo);

// For downloading issues
List<RepositoryIssue> issueList = issueservice.getIssues();

// For downloading pulls
List<PullRequest> pullList = pullservice.getPullRequests(repo, "closed");
```
GitHub API for JAVA – HOW? (cont.)

Once we have obtained the lists with the data, we can retrieve all the info from the commit/issue/pull objects.

```java
// Getting the SHA key from the i-commit
String sha = commitList.get(i).getSha();

// Getting the author from the i-commit
commitList.get(i).getCommit().getAuthor().getName();

// Getting the message from the i-commit
String message = commitList.get(i).getCommit().getMessage();

...
GitHub API for JAVA – HOW? (cont.)

Once we have obtained the lists with the data, we can retrieve all the info from the commit/issue/pull objects.

```java
// Getting the SHA key from the i-commit
String sha = commitList.get(i).getSha();

// Getting the author from the i-commit
commitList.get(i).getCommit().getAuthor().getName();

// Getting the message from the i-commit
String message = commitList.get(i).getCommit().getMessage();
```

...
GitHub API for Python

PyGitHub is a Python library to access the GitHub API v3

$ pip install pygithub

from github import Github

# First create a Github instance:
g = Github("user", "password")

# or using an access token
g = Github("access_token")

# Then play with your Github objects:
for repo in g.get_user().get_repos():
    print(repo.name)
GHTorrent

• Torrential downpour from GitHub

• GitHub Torrent Data (ghtorrent.org)

• Collected by Georgios Gousios from TU Delft

• Available in
  • MySQL dumps (5 Billions Records)
  • MongoDB (10TB of entity data)
  • Azure Data Lake Store
    https://github.com/Microsoft/ghinsights
  • Google Big Query http://ghtorrent.org/gcloud.html
JIRA – What data can be extracted?
JIRA – How to extract data?

- Web scraping
- Using JIRA official API
  - REST
  - Java
  - Python
JIRA Java API

- JIRA has its own Java API
  https://developer.atlassian.com/server/jira/platform/java-apis/
- It is typically used when building JIRA Server add-ons
- Use the Maven artefact

```xml
<dependency>
  <groupId>com.atlassian.jira</groupId>
  <artifactId>jira-api</artifactId>
  <version>${atlassian.product.version}</version>
  <scope>provided</scope>
</dependency>
```
Python Jira Library is a python library to access the JIRA API

```
$ pip install jira

from jira import JIRA

jira = JIRA('https://jira.atlassian.com')

issue = jira.issue('JRA-9')
print issue.fields.project.key
print issue.fields.issuetype.name
print issue.fields.reporter.displayName
```
# Rate Limits – Github and JIRA

<table>
<thead>
<tr>
<th></th>
<th>GitHub*</th>
<th>JIRA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal API</td>
<td>Search API</td>
</tr>
<tr>
<td>Unauthenticated</td>
<td>60 request per hour</td>
<td>10 per minute</td>
</tr>
<tr>
<td>requests</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basic Authentication or OAuth</td>
<td>5000 request per hour</td>
<td>30 per minute</td>
</tr>
</tbody>
</table>

(*) https://developer.github.com/v3/#rate-limiting
(**) https://jira.atlassian.com/browse/JRASERVER-41876
## Extraction Methods – GitHub and JIRA

<table>
<thead>
<tr>
<th>Method</th>
<th>GitHub</th>
<th>JIRA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web scrapping</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>REST API</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>JAVA</td>
<td>Github API for Java</td>
<td>Jira API</td>
</tr>
<tr>
<td>Python</td>
<td>PyGitHub</td>
<td>Python Jira Library</td>
</tr>
<tr>
<td>Massive dump</td>
<td>GHTorrent</td>
<td>❌?</td>
</tr>
</tbody>
</table>
LTAT.05.008: Software Analytics Practice Session

Ezequiel Scott
ezequiel.scott@ut.ee
Practice Session

The KDD process
Context

Data Mining + Software Engineering

The Scrum Framework:

1. **INPUTS FROM CUSTOMERS, TEAM, MANAGERS & EXECVS.**
   - PRODUCT OWNER
   - THE TEAM

2. **PRODUCT BACKLOG**
   - Prioritized list of what is required: features, bugs...

3. **SPRINT PLANNING MEETING**
   - Team selects starting at top as much as it can commit to deliver by end of Sprint

4. **TASK BREAKOUT**

5. **SPRINT BACKLOG**

1-4 week SPRINT

**SPRINT MASTER**

- DAILY STAND UP MEETING
- SPRINT REVIEW
- FINISHED WORK
- SPRINT RETROSPECTIVE

Sprint end date and team deliverable do not change

JIRA

GitHub
## JIRA Dataset

<table>
<thead>
<tr>
<th>Team</th>
<th>User Name</th>
<th>User Story</th>
<th>Story Points</th>
<th>Status</th>
<th>Created</th>
<th>Resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>SpringXD</td>
<td>User1</td>
<td>As a user, I want...</td>
<td>5</td>
<td>TO DO</td>
<td>2016-03-31T22:35:55.000+0000</td>
<td>2016-03-31T22:35:55.000+0000</td>
</tr>
<tr>
<td>SpringXD</td>
<td>User2</td>
<td>As a manager, I want...</td>
<td>3</td>
<td>DOING</td>
<td>2016-03-14T18:09:51.000+0000</td>
<td>2016-03-14T18:09:51.000+0000</td>
</tr>
<tr>
<td>SpringXD</td>
<td>User1</td>
<td>As an admin, I want...</td>
<td>2</td>
<td>DONE</td>
<td>2016-03-03T13:22:14.000+0000</td>
<td>2016-03-03T18:41:19.000+0000</td>
</tr>
<tr>
<td>SpringXD</td>
<td>User3</td>
<td>As a user, I want...</td>
<td>1</td>
<td>DONE</td>
<td>2016-02-29T10:00:18.000+0000</td>
<td>2016-03-13T10:24:15.000+0000</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Interesting Problems

I. Estimating story points from User Stories

II. Recommending task to developers

III. Exploring the GitHub social network

IV. Sentiment analysis from commits
Problem (I)

Estimating **Story Points** from User Stories

\(<\text{User}, \text{Description}, \text{Type}, \ldots, \text{Estimates}\>\)

Given some user story → Estimate (Story Points)

“Recommend an estimate to junior (?) developers”

Ezequiel Scott - ezequiel.scott@ut.ee
Problem (II)

Recommend **task** to developers

<table>
<thead>
<tr>
<th>Dev 1</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tasks' features</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>Task 1</td>
</tr>
<tr>
<td>Task 2</td>
</tr>
<tr>
<td>Task 3</td>
</tr>
<tr>
<td>Task 4</td>
</tr>
<tr>
<td>Task 5</td>
</tr>
</tbody>
</table>

=

<table>
<thead>
<tr>
<th>Dev 1</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tasks' features</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>Dev 1</td>
</tr>
<tr>
<td>Dev 2</td>
</tr>
<tr>
<td>Dev 3</td>
</tr>
</tbody>
</table>
Problem (III)

Exploring the GitHub social network
Problem (IV)

Sentiment analysis from commits

Discovering people's opinions, emotions, and feelings about a product or service

Sentiment Analysis of Commit Comments in GitHub: An Empirical Study

Emitza Guzman, David Azócar, Yang Li
Technische Universität München
Faculty of Informatics
Garching, Germany
emitza.guzman@mytum.de, dazocar@gmail.com, liya@in.tum.de