Software Development Effort and Quality Prediction Using Bayesian Nets and Small Local Qualitative Data

Łukasz Radliński
University of Szczecin, Poland
lukrad@uoo.univ.szczecin.pl
Agenda

• Background
• Bayesian Nets
• Dataset and method
• Results
  – accuracy
  – most important predictors
• Conclusions
Background

• To predict software development effort & quality
  – large and homogeneous datasets
    • difficult in practice - $$$
  – assumptions for statistical methods
    • are software engineers aware of constraints?

• A need for:
  – modeling method – easy to use
  – building a model – flexible (expert/data)
  – clear presentation for model and results

• Solution: Bayesian nets
Aim

Can Bayesian nets be effectively used to predict software development effort and software quality?

Two assumptions:

1. Available dataset is \textbf{small} but contains \textbf{local} data about past projects from a single company
2. A model is \textbf{automatically} generated from available data without expert input
Bayesian nets

Two perspectives:

1. **graphical**: a directed acyclic graph consisting of a set of nodes (variables) and directed links between pairs of nodes

2. **numeric**: each node is defined in terms of conditional probability distributions given its parents
Bayesian net topologies

Naive Bayesian Classifier – NBC

Converging star – CS

Causal BN – CBN

Dynamic BN – DBN

Łukasz Radliński: Software Development Effort and Quality Prediction Using Bayesian Nets and Small Local Qualitative Data
## Recent Bayesian nets for software engineering

<table>
<thead>
<tr>
<th>Author</th>
<th>Main problem analyzed</th>
<th>BN topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radliński, 2009</td>
<td>types of defects</td>
<td>NBC</td>
</tr>
<tr>
<td>Stewart, 2002</td>
<td>effort, productivity</td>
<td>NBC</td>
</tr>
<tr>
<td>Pai &amp; Dugan, 2007</td>
<td>fault content, fault proneness</td>
<td>CS</td>
</tr>
<tr>
<td>Zhou et al., 2008</td>
<td>change coupling</td>
<td>CS</td>
</tr>
<tr>
<td>Cockram, 2001</td>
<td>effectiveness of inspections</td>
<td>CBN</td>
</tr>
<tr>
<td>Dabney et al., 2006</td>
<td>defect rate</td>
<td>CBN</td>
</tr>
<tr>
<td>del Salgado Martinez &amp; del Aguina Cano, 2008</td>
<td>need for requirements review</td>
<td>CBN</td>
</tr>
<tr>
<td>Fenton et al., 2004</td>
<td>trade-off between: size, effort, quality</td>
<td>CBN</td>
</tr>
<tr>
<td>Fenton et al., 2007, 2008b</td>
<td>defects, partly: effort</td>
<td>CBN/DBN</td>
</tr>
<tr>
<td>Mendes &amp; Mosley, 2008; Mendes, 2008</td>
<td>web development effort</td>
<td>CBN</td>
</tr>
<tr>
<td>Pai et al., 2005</td>
<td>maturity of requirements</td>
<td>CBN</td>
</tr>
<tr>
<td>Radliński, 2007</td>
<td>trade-off between: size, effort, quality</td>
<td>CBN</td>
</tr>
<tr>
<td>Wagner, 2009</td>
<td>various aspects of software quality</td>
<td>CBN</td>
</tr>
<tr>
<td>Woof et al., 2002</td>
<td>testing process</td>
<td>CBN</td>
</tr>
<tr>
<td>Bai et al., 2002</td>
<td>failures</td>
<td>DBN</td>
</tr>
<tr>
<td>Bibi &amp; Stamelos, 2004</td>
<td>effort</td>
<td>DBN</td>
</tr>
<tr>
<td>Fenton et al., 2008a; Radliński et al., 2008</td>
<td>defects</td>
<td>DBN</td>
</tr>
<tr>
<td>Hearty et al. 2009</td>
<td>project velocity (functionality)</td>
<td>DBN</td>
</tr>
</tbody>
</table>
Dataset

- PROMISE Repository of Empirical Software Engineering Data:
  - [http://promisedata.org/?p=44](http://promisedata.org/?p=44) (extended)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

29 cases

4 dependent variables

29 predictors

- effort
- productivity rate (effort / project size)
- number of defects
- defect rate (number of defects / project size)

- project size
- project type

27 factors - process and people quality
Research procedure

1. Prepare data
2. Split dataset
3. Generate a BN from learning subset
4. Test model using testing subset
5. Calculate and analyze evaluation measures
6. Identify most important predictors

- Continue with another data split? (Yes/No)

Background | Bayesian nets | Data & method (current) | Results | Conclusions
Accuracy – individual samples

Mean magnitude of relative error

Median magnitude of relative error
Accuracy – individual samples

![Graph showing the relationship between set number and pred(25)](image)

- Pred(25) vs. Set number
- Comparing effort, productivity, defects, and defect rate

Legend:
- Effort
- Productivity
- Defects
- Defect rate
Accuracy – aggregated

<table>
<thead>
<tr>
<th></th>
<th>MMRE</th>
<th>MdMRE</th>
<th>Pred(25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>effort</td>
<td>0.97</td>
<td>0.60</td>
<td>0.17</td>
</tr>
<tr>
<td>productivity</td>
<td>0.77</td>
<td>0.40</td>
<td>0.27</td>
</tr>
<tr>
<td>defects</td>
<td>2.14</td>
<td>0.79</td>
<td>0.14</td>
</tr>
<tr>
<td>defect rate</td>
<td>0.83</td>
<td>0.53</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Accuracy – comparison

**Background**

**Bayesian nets**

**Data & method**

**Results**

**Conclusions**

Łukasz Radliński: Software Development Effort and Quality Prediction Using Bayesian Nets and Small Local Qualitative Data

13
Most important predictors: best performing Bayesian nets

**The best**
- Development staff training
- Project planning
- Configuration management
- Productivity rate
- Defect rate

**2nd best**
- Testing process well defined
- Process maturity
- Project planning
- Programmer capability
- Productivity rate
- Defect rate

**Results**
- Effort
- Defects
- Requirements stability

**Background**
- Bayesian nets
- Data & method
- Conclusions
## Most frequent factors in Markov blankets

<table>
<thead>
<tr>
<th>Factor</th>
<th>Dependent variable</th>
<th>effort</th>
<th>productivity rate</th>
<th>defects</th>
<th>defect rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>effort</td>
<td></td>
<td>–</td>
<td>0</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>productivity rate</td>
<td></td>
<td>0</td>
<td>–</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>defects</td>
<td></td>
<td>1</td>
<td>3</td>
<td>–</td>
<td>0</td>
</tr>
<tr>
<td>defect rate</td>
<td></td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>–</td>
</tr>
<tr>
<td>dev. staff motivation</td>
<td></td>
<td>7</td>
<td>1</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>project size</td>
<td></td>
<td>5</td>
<td>8</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>requirements stability</td>
<td></td>
<td>5</td>
<td>0</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>complexity of new functionality</td>
<td></td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>project planning</td>
<td></td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>dev. staff experience</td>
<td></td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>vendor management</td>
<td></td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
Conclusions

• Automatic generation of BNs from empirical data
  – easy to use
  – low volume of data does not allow to exploit these BNs in other environments
  – but relationships identified can still be a useful base

• To improve accuracy – stronger input from expert
  – BN topology by expert, parameters from data
  – BN structure and parameters from data, then an expert may adjust parameters
Thank you

questions?