Specialization and Extrapolation of Software Cost Models

Tim Menzies, Portland State, USA  
Zhihao Chen, USC, USA  
Dan Port, U.Hawaii, USA  
Jairus Hihn, JPL, USA

Motivation

"Software costing is a quality issue.”  
- Get it wrong and everything suffers; e.g. no $$ for QA

"Most software is costed like the weather." [Boehm, 2000]  
- Tomorrow will be like today, times some “deltas”.  
- Is it safe to cost new projects via extrapolation of old ones?

"Stop model conflation.”  
- As time goes by, most cost models get more elaborate;  
- Experience should tell us when to add or PRUNE variables.

Example

Is software complexity a useful cost driver?  
- In NASA data sets, CPLX=high (usually);  
- No information in this variable;  
- Prune it?

Prune rows? (only use data from related projects?)

Specialization

The "wrapper": [Kohavi & John, 1997]  
1) Pick a learner;  
   • here: LSR (not M5') on log(NUMS)  
2) Include some more attributes;  
3) Try learning with just those attributes;  
4) If better then { Stale = 0 }  
   else if ( ++Stale > 5 )  
   then {Stale=0; forget last 5 includes}  
5) Goto 2)

Why specialize?

Fewer variables  
= smaller theories  
• more explainable, easier processing.

Many variables are noisy, redundant, under-sampled (e.g. cplx)  
• they confuse, rather than clarify, the generalization process.

Lower variance  
[Miller 2002]

Better estimates

- Especially for small data sets.  
- And small data sets are the industrial norm.

Less variation

- In the learned models.  
- 30 * 90% sub-samples of the data  
- Leam "effort = β₀ + β₁x₁ + β₂x₂ +.. “  
- Plot the β variance:

Conclusion

- For generalization via LSR:
  - no extrapolation without prior specialization.