Software Quality Engineering:

Testing, Quality Assurance, and Quantifiable Improvement

Jeff Tian, tian@engr.smu.edu
www.engr.smu.edu/~tian/SQEbook

Chapter 21. Risk Identification for Quantifiable Quality Improvement

- Basic Ideas and Concepts
- Traditional Statistical Techniques
- Newer/More Effective Techniques
- Tree-Based Analysis of ODC Data

Risk Identification: Why?

- Observations and empirical evidences:
  - ▷ 80:20 rule: non-uniform distribution:
    - – 20% of the modules/parts/etc. contribute to
    - – 80% of the defects/effort/etc.
  - ▷ implication: non-uniform attention
    - – risk identification
    - – risk management/resolution

- Risk Identification in SQE:
  - ▷ 80:20 rule as implicit hypothesis
  - ▷ focus: techniques and applications

Risk Identification: How?

- Qualitative and subjective techniques:
  - ▷ Causal analysis
  - ▷ Delphi and other subjective methods
- Traditional statistical techniques:
  - ▷ Correlation analysis
  - ▷ Regression models:
    - – linear, non-linear, logistic, etc.
- Newer (more effective) techniques:
  - ▷ Statistical: PCA, DA, TBM
  - ▷ AI-based: NN, OSR
  - ▷ Focus of our Chapter.

Risk Identification: Where?

- 80% or target:
  - ▷ Mostly quality or defect (most of our examples also)
  - ▷ Effort and other external metrics
  - ▷ Typically directly related to goal
  - ▷ Resultant improvement
- 20% or contributor:
  - ▷ 20%: risk identification!
  - ▷ Understand the link
  - ▷ Control the contributor:
    - corrections/defect removal/etc.
    - future planning/improvement
    - remedial vs preventive actions

Statistics Basics

- Random variables
  - ▷ used in statistical analysis/modeling and in risk identification in particular
  - ▷ used as either i.v./p.v. or d.v./r.v.
  - ▷ i.v. or p.v.: independent variable – also called predictor (variable)
  - ▷ d.v. or r.v.: dependent variable – also called response (variable)

- Statistical distributions:
  - ▷ 1d: normal, exponential, binomial, etc.
  - ▷ 2d: independent vs. correlated
  - ▷ covariance, correlation (coefficient)

Traditional Technique: Correlation

- Correlation coefficient (c.c.):
  - ranges between -1 and 1
  - positive: move in same direction
  - negative: move in opposite direction
  - 0: not correlated (independent)
- Correlation analysis:
  - use correlation coefficient
  - linear (Pearson) correlation vs. non-parametric (Spearman) correlation
  - based on measurement type/distribution:
    - non-normal distribution
    - ordinal measurement etc.

Traditional Technique: Correlation

- Correlation analysis: applications
  - ▷ understand general relationship
    – e.g., complexity-defect correlation
  - ▷ risk identification also
  - ▷ cross validation (metrics etc.)

- Correlation analysis: assessment
  - ▷ only partially successful
  - ▷ low correlation, then what?
  - ▷ data skew: 0-defect example
  - ▷ uniform treatment of data

$\Rightarrow$ Other risk identification techniques needed.

Traditional Technique: Regression

- Regression models:
  - ▷ as generalized correlation analysis
  - ▷ n i.v. combined to predict 1 d.v.
  - ▷ forms of prediction formula
    $\Rightarrow$ diff. types of regression models

- Types of regression models:
  - ▷ linear: linear function

$$y = \alpha_0 + \alpha_1 x_1 + ... + \alpha_n x_n + \epsilon$$

  - ▷ log-linear: linear after log-transformation
  - ▷ non-linear: non-linear function
  - ▷ logistic: represent presence/absence of categorical variables

Traditional Technique: Regression

- Regression analysis: applications
  - ▷ similar to correlation analysis
  - ▷ multiple attribute data

- Regression analysis: assessment
  - ▷ only partially successful
  - ▷ similar to correlation analysis
  - ▷ often marginally better (R-sqr vs c.c.)
  - ▷ same kind of problems
  - ▷ data transformation problem
  - ▷ synthesized metrics $\sim$ regression model?

$\Rightarrow$ Other risk identification techniques needed.

New Techniques

- New statistical techniques:
  - ▷ PCA: principal component analysis
  - ▷ DA: discriminant analysis
  - ▷ TBM: tree-based modeling
- AI-based new techniques:
  - ▷ NN: artificial neural networks.
  - ▷ OSR: optimal set reduction.
  - ▷ Abductive-reasoning, etc.
- Focus of our Chapter.

New Techniques: PCA & DA

- Not really new techniques,
  but rather new applications in SE.

- PCA: principal component analysis
  - ▷ Idea of linear transformation.
  - ▷ PCA to reduce dimensionality.
  - ▷ Effectively combined with DA and other techniques (NN later).

- DA: discriminant analysis
  - ▷ Discriminant function
  - ▷ Risk id as a classification problem
  - ▷ Combine with other techniques

New Techniques: PCA & DA

- PCA: why?
  - ▷ Correlated i.v.'s ⇒ unstable models
  - ▷ Extreme case: linearly dependent ⇒ singularity
  - ▷ linear transformation (PCA) ⇒ uncorrelated PCs (or domain metrics)

- PCA: how?
  - ▷ Covariance matrix: Σ
  - ▷ Solve $|\Sigma - \Lambda| = 0$ to obtain eigenvalues λ_j along the diagonal for the diagonal matrix Λ
  - ▷ λ_j's in decreasing value
  - ▷ Decomposition: $\Sigma = C^T \Lambda C$
  - ▷ C: matrix of eigenvectors (transformation used)

New Techniques: PCA & DA

- Transformation to PCs/data matrix D:
  - ▷ $D = ZT$, where
  - ▷ Z is the original data matrix
  - ▷ T is the transformation matrix
- Obtaining PCA results: Λ, C, T can be calculated by statistical packages/tools.
- PCA result interpretation/usage:
  - ▷ Eigenvalues $\approx$ explained variance.
  - ▷ Uncorrelated PCs
    $\Rightarrow$ good/stable (linear/other) models

New Techniques: PCA & DA

	pc1	pc2	pc3	pc4
eigenvalue λ_i	2.352	1.296	1.042	0.711
% of variance	55.3%	16.8%	10.8%	5.1%
cumulative % of variance	55.3%	72.1%	82.9%	88.0%

- PCA example: Table 21.1 (p.357)
  - ▷ Eigenvalues sorted in decreasing order.
  - ▷ First few (3-5) principal components (PCs) explain most of the variance.
  - ▷ Dimension reduction
    $\Rightarrow$ simplified models with a few PCs.

New Techniques: PCA & DA

- DA: general ideas
  - ▷ Define discriminant function.
  - ▷ Classify into G_1 and G_2
    - G_1: not fault-prune
    - G_2: fault-prune

- Sample discriminant function:
  - ▷ Assign d_i to G_1, if $\frac{f_1(d_i)}{f_2(d_i)} > \frac{\pi_2}{\pi_1}$,
  - ▷ Otherwise, assign d_i to G_2.
  - ▷ d_i: i-th module's principal-component values (i-th row of the D matrix above).
  - ▷ π_k: prior probability(membership in G_k).
  - ▷ $f_k(d_i)$: probability($d_i \in G_k$).

New Techniques: PCA & DA

- DA: how?
  - ▷ Discriminant function example above.
  - ▷ Other/similar definitions possible.
  - ▷ Minimize misclassification rate in model fitting and in prediction.
  - ▷ Good results (Khoshgoftaar et al., 1996).

- PCA&DA: Summary and Observations:
  - ▷ Positive/encouraging results, but,
  - ▷ Much processing/transformation needed.
  - ▷ Much statistics knowledge.
  - ▷ Difficulty in data/result interpretation.

New Technique: NN

- NN or ANN: artificial neural networks
  - ▷ Inspired by biological computation
  - ▷ Neuron: basic computational unit – different functions
  - ▷ Connection: neural network
  - ▷ Input/output/hidden layers

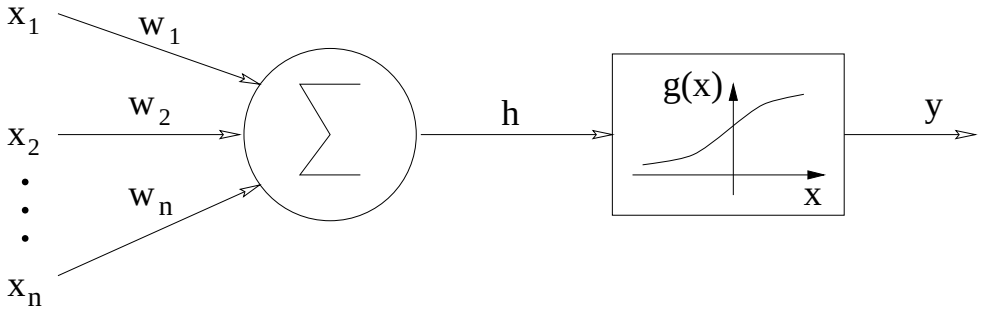
- NN applications:
  - ▷ AI and AI problem solving
  - ▷ In SQE: defect/risk identification

New Technique: NN

- Computation at a neuron:
  - ▷ Weighted sum of input: $h = \sum_{1}^{n} x_i$ (may include constant)
  - ▷ Then activation function $y = g(h)$
    - – threshold, piecewise-linear,
    - – Gaussian, sigmoid (below), etc.

$$y = \frac{1}{1 + e^{-\beta x}}$$

- Illustration: Fig 21.1 (p.358)



New Technique: NN

0. Initialize the weights to small random values.
1. Repeat steps 2 $\sim$ 6 until the error in the output layer is below a pre-specified threshold or a maximum number of iterations is reached.
2. Randomly choose an input.
3. Propagate the signal forward through the network.
4. Compute the errors in the output layer.
5. Compute the deltas for the preceding layers by propagating the errors backward.
6. Update the weights based on these deltas.

- Overall NN computation: Fig 21.2 (p.359)
  - ▷ input layer: raw data feed
  - ▷ other layers: computation at n neurons
  - ▷ minimize prediction error at the output layer via backward propagation

New Technique: NN

- NN study (Khoshgoftaar and Szabo, 1996):
  - ▷ train NN with Kernel.1 data
    – using both raw data and PCA data
  - ▷ apply NN to Kernel.2 and Kernel.3

- Result comparison: Table 21.2 (p.359)
  - ▷ NN superior to linear regression.
  - ▷ NN+PCA superior to NN on raw data.

System	Model data	Output error			
		mean	std.dev	min.	max.
Kernel.2	raw	11.4	6.6	0.19	32.8
	PCA	7.1	5.6	0.05	42.8
Kernel.3	raw	11.0	6.3	0.12	31.6
	PCA	4.7	4.1	0.02	26.2

New Technique: TBM

- TBM: tree-based modeling
  - ▷ Similar to decision trees
  - ▷ But data-based (derived from data)
  - ▷ Preserves tree advantages:
    - – easy to understand/interpret
    - – both numerical and categorical data
    - – partition ⇒ non-uniform treatment

- TBM applications:
  - ▷ Main: defect analysis
    TBDMs (tree-based defect models)
  - ▷ Past: psychology, SE-Amadeus, etc.
  - ▷ Reliability: TBRMs (Ch.22)

- Risk identification **and** characterization.

New Technique: TBM

- TBM for risk identification:
  - ▷ Assumption in traditional techniques:
    - linear relation
    - uniformly valid result
  - ▷ Reality of defect distribution:
    - isolated pocket
    - different types of metrics
    - correlation/dependency in metrics
    - qualitative differences
  - ▷ Need new risk id. techniques.

- TBM for risk characterization:
  - ▷ Identified, then what?
  - ▷ Result interpretation.
  - ▷ Remedial/corrective actions.
  - ▷ Extrapolation to new product/release.
  - ▷ TBDMs appropriate.

New Technique: TBM

- TBDMs: tree-based defect models using tree-based modeling (TBM) technique

- Decision trees:
  - ▷ multiple/multi-stage decisions
  - ▷ may be context-sensitive
  - ▷ natural to the decision process
  - ▷ applications in many problems
    - – decision making & problem solving
    - – decision analysis/optimization

- Tree-based models:
  - ▷ reverse process of decision trees
  - ▷ data $\Rightarrow$ tree
  - ▷ idea of decision extraction
  - ▷ generalization of "decision"

New Technique: TBM

- Technique: tree-based modeling
  - ▷ Tree: nodes=data-set, edges=decision.
  - ▷ Data attributes:
    – 1 response & n predictor variables.
  - ▷ Construction: recursive partitioning
    – with tree growing and pruning.

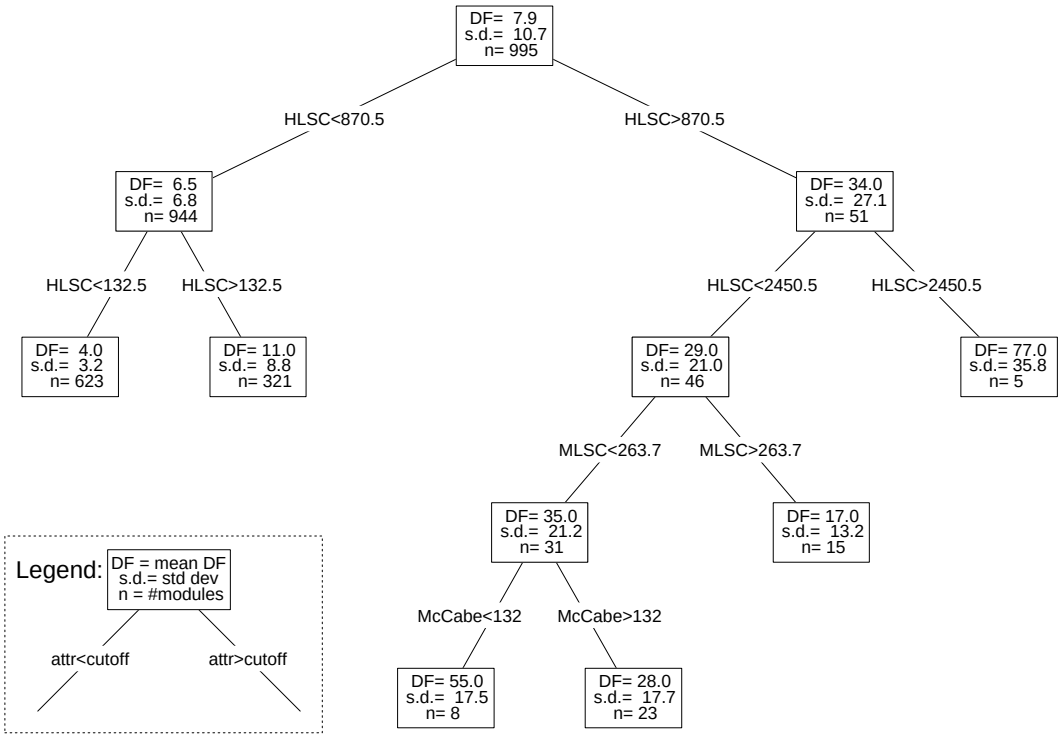
- TBM usage: relating r.v. to p.v.:
  - ▷ $Y = Tree(X_1, \ldots, X_n)$
  - ▷ understanding vs. predicting
  - ▷ identification and characterization

- Types of data $\Rightarrow$ tree type
  - ▷ numerical r.v.: regression tree
  - ▷ categorical r.v.: classification tree

TBM Algorithm: Fig 21.3 (p.360)

0. *Initialization.* Set the list, `Slist`, to contain only the complete data set as its singleton element. Select the size and homogeneity thresholds T_s and T_h for the algorithm.
1. *Overall control.* Repeatedly remove a data set from `Slist` and execute step 2 until `Slist` becomes empty.
2. *Size test.* If $|S| < T_s$, stop; otherwise, execute steps 3 through 6. $|S|$ is the number of data points in set S.
3. *Defining binary partitions.* A binary partition divides S into two subsets using a *split condition* defined on a specific predictor p. For numerical p, it can be defined with a cutoff value c: Data points with $p < c$ form one subset (S_1) and those with $p \geq c$ form another subset (S_2). If p is a categorical variable, a binary partition is a unique grouping of all its category values into two mutually exclusive subsets S_1 and S_2.
4. *Computing predicted responses and prediction deviances.* The predicted response value $v(S)$ for a set S is the average over the set; that is, $v(S) = \frac{1}{|S|}\sum_{i \in S}(v_i)$; and the prediction deviance is $D(S) = \sum_{i \in S}(v_i - v(S))^2$, where v_i is the response value for data point i.
5. *Selecting the optimal partition.* Among all the possible partitions (all predictors with all associated cutoffs or binary groupings), the one that minimizes the deviance of the partitioned subsets is selected; that is, the partition with minimized $D(S_1) + D(S_2)$ is selected.
6. *Homogeneity test:* Stop if $\left(1 - \frac{D(S_1)+D(S_2)}{D(S)}\right) \leq T_h$ (that is, stop if there is no substantial gain in prediction accuracy in further partitioning); otherwise, append S_1 and S_2 to $Slist$.

TBM Example



- TBDM example: Fig 21.4 (p.361)
  - ▷ defect prediction for IBM-NS
  - ▷ 11 design/size/complexity metrics.

TBM Example

Node:	Split Conditions or Subset Characteristics	# of modules	Predicted DF
rlll:	870.5 $<$ HLSC $<$ 2450.5 $\wedge$ MLSC $<$ 263.7, $\wedge$ McCabe $<$ 132	8	55.0
rr:	$HLSC >$ 2450.5	5	77.0

- Table 21.3 (p.361): high-risk subsets.
- Design and control complexity as main predictors to characterize high-risk.
- Key "selling" points of TBM:
  - ▷ intuitiveness and interpretation – compare to PCA, NN
  - ▷ quantitative & qualitative info.
  - ▷ hierarchy/importance/organization

New Technique: OSR

- OSR: optimal set reduction
  - ▷ pattern matching idea
  - ▷ clusters and cluster analysis
  - ▷ similar to TBM but different in:
    - – pattern extraction vs. partition

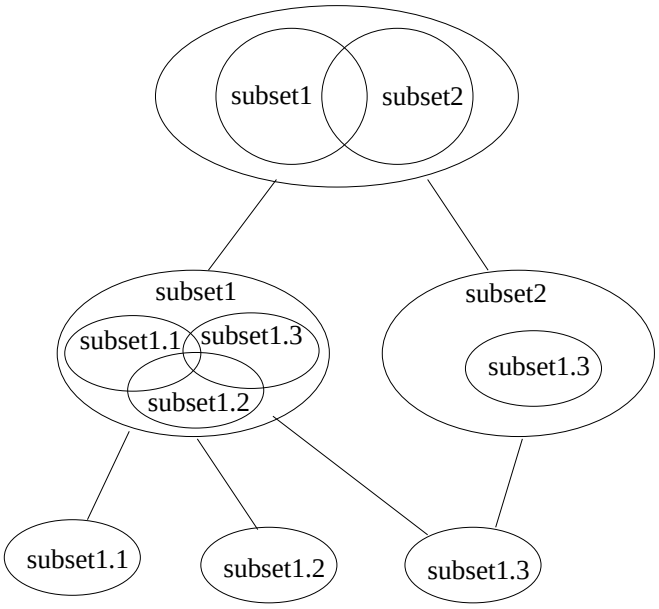
- algorithm sketch: Fig 21.5 (p.362)

Step 1. Both the dependent (response) variable and the explanatory (predictor or independent) variables are discretized by using cluster analysis or some other method if they are continuous.

Step 2. Select all statistically significant subsets defined by a pattern whose entropy (or uniformity) is within a threshold of the minimal entropy.

Step 3. Step 2 is repeated until no significant gain can be made in entropy reduction.

New Technique: OSR



- OSR illustration: Fig 21.6 (p.363)

- Organization/modeling results:
  - ▷ no longer a tree, see example above
  - ▷ general subsets, may overlap
  - ▷ details and some positive results: see Briand et al. (1992)

Risk Identification: Comparison

- Comparison: cost-benefit analysis
  $\approx$ comparing QA alternatives (Ch.17).

- Comparison area: benefit-related
  - ▷ accuracy
  - ▷ early availability and stability
  - ▷ constructive information and guidance for (quality) improvement

- Comparison area: cost-related
  - ▷ simplicity
  - ▷ ease of result interpretation
  - ▷ availability of tool support

Comparison: Accuracy

- Accuracy in assessment:
  - ▷ model fits data well
    - – use various goodness-of-fit measures
  - ▷ avoid over-fitting
  - ▷ cross validation by review etc.

- Accuracy in prediction:
  - ▷ over-fitting $\Rightarrow$ bad predictions
  - ▷ prediction: training and testing sets
    - – within project: jackknife
    - – across projects: extrapolate
  - ▷ minimize prediction errors

Comparison: Usefulness

- Early availability and stability
  - ▷ to be useful must be available early
  - ▷ focus on control/improvement
  - ▷ apply remedial/preventive actions early
  - ▷ track progress: stability

- constructive information and guidance
  - ▷ what: assessment/prediction
  - ▷ how to improve?
    - – constructive information
    - – guidance on what to do
  - ▷ example of TBRMs

Comparison: Usability

- Can't explain in a few words
  $\Rightarrow$ difficulties with reception/deployment

- Simplicity & result interpretation?
  - ▷ technique easy to use/understand
  - ▷ what does it (the result) mean?
  - ▷ training effort involved
  - ▷ causal and other connections

- Tool and other support:
  - ▷ availability of easy-to-use tools
  - ▷ other support: process/personnel/etc.
  - ▷ direct impact on deployment

Comparison Summary

(Table 21.4)

Technique	Benefit/Performance		
	accuracy	stability	guidance
correlation	poor	fair	fair
regression	poor	poor	poor
PCA/DA	good	excellent	fair
NN	good	fair	poor
TBM	good	good	excellent
OSR	good	fair	excellent

(Table 21.4 continued)

Technique	Cost/Usability		
	simplicity	interp.	tool sup.
correlation	simplest	easiest	wide
regression	simple	moderate	wide
PCA/DA	moderate	moderate	moderate
NN	complex	hard	moderate
TBM	moderate	easy	moderate
OSR	complex	moderate	limited

- Summary: Table 21.4 (p.364)

Recommendations and Integration

- General recommendations:
  - ▷ TBM good balance ⇒ prime candidate.
  - ▷ Other techniques might be useful too.
  - ▷ A suite of techniques might be suitable.

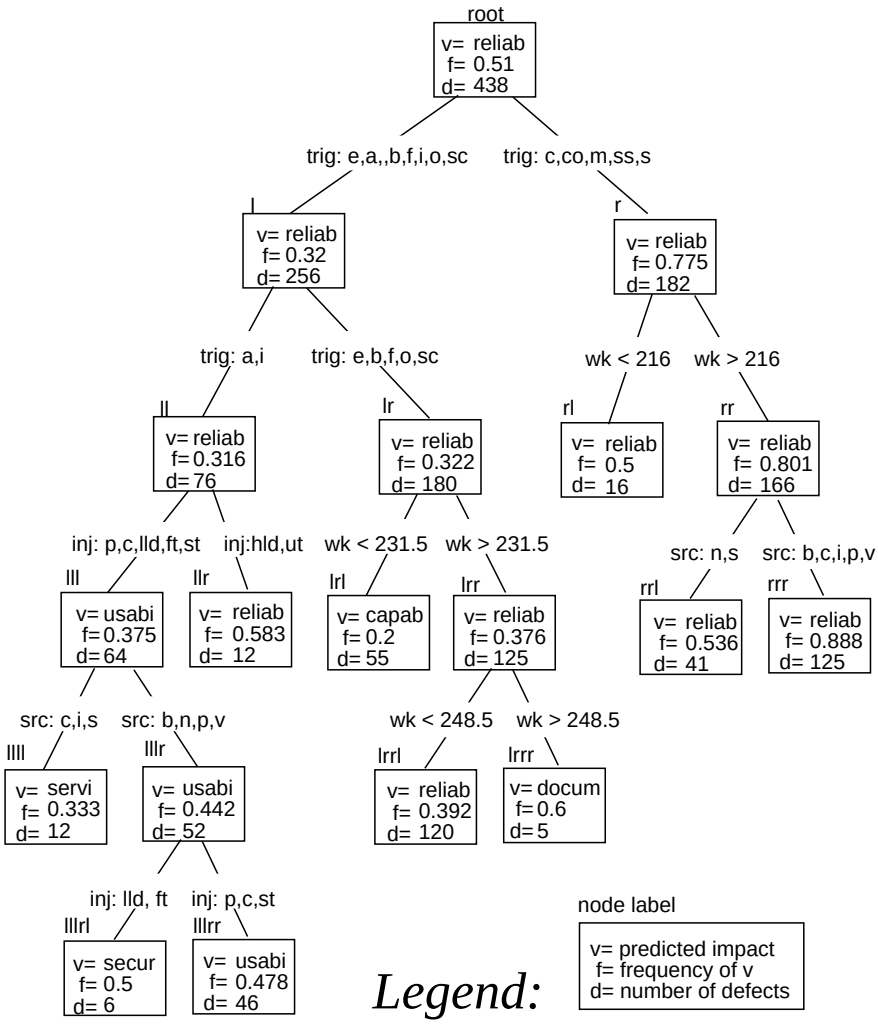
- Lifecycle integration:
  - ▷ Process and data availability ⇒ inspection/testing/other QA data.
  - ▷ Experience/infrastructure/tools/etc. for implementation/technology transfer.
  - ▷ Similar techniques for other problems – e.g., identifying effort, schedule risks.
  - ▷ Tailoring to individual process/product

Tree-Based ODC Data Analysis

- Continuation of ODC analysis:
  - ▷ IBM Toronto data from ODC (Ch.20)
  - ▷ 1-way → 2-way → n-way analyses
    - – combinatorial explosion
  - ▷ Better focus on n-1 linkage:
    - – 1 response variable: impact
    - – n (=6 here) predictor variables
  - ▷ ODC attributes in Table 20.6 (p.347)
    - – all except "severity" used
    - – impact-severity analysis already done: see Table 20.7 (p.351)

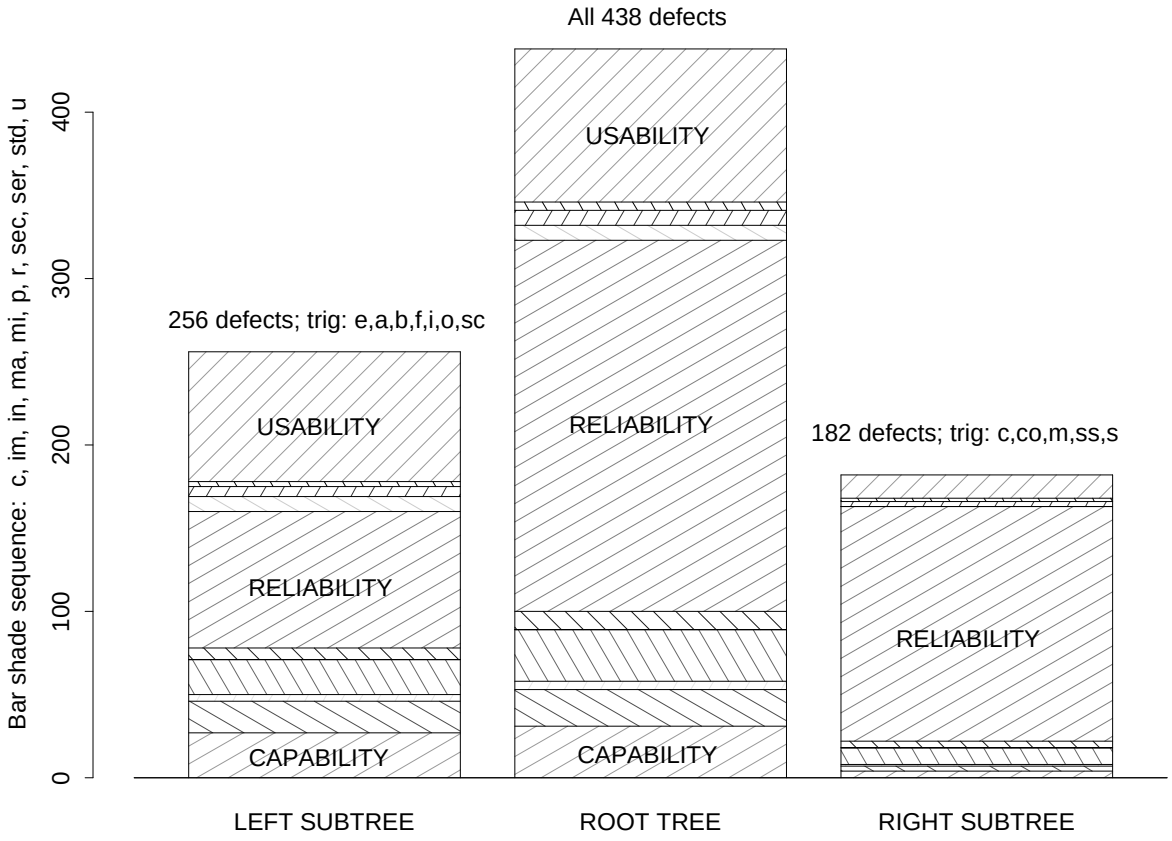
- Tree-based ODC modeling
  - ▷ Classification trees (instead of regression trees)
  - ▷ Change in distribution

Tree-Based ODC Data Analysis



- Overall result: Fig 21.7 (p.366)
  - dominant impact shown in tree nodes.

Tree-Based ODC Data Analysis



- Fig 21.8 (p.367): impact distribution
  - ▷ Primary partition: defect trigger
  - ▷ High homogeneity of right subtree
  - ▷ Problem identification: left subtree

Tree-Based ODC Data Analysis

- Result interpretation:
  - ▷ Overall result: Fig 21.7 (p.366)
  - ▷ Dominant impact: tree nodes.
  - ▷ Impact distribution: bars.
  - ▷ Confidence: frequency and cardinality.
- Usage of modeling results:
  - ▷ Passive tracking and correction
  - ▷ Active problem identification and quality control
- Other ODC trees possible (for different r.v.)