Lecture 2: Metrics and Measurement

17-313: Foundations of Software Engineering
Fall 2023

Administrivia

- Slack
  - Add profile picture (if you want).
  - Ask questions in #general or #project-one. Use threads.
- Homework 1 is released. It is due Wed Aug 30, 11:59 pm (one week!)
  - This is an individual assignment
  - Get started early, ask for help, and check the #project-one channel; chances are decent your questions have been asked by others! Office hours are running
- Complete the team formation survey today!
- Reading for next Sunday will be posted on the website
  - Gradescope quiz due before class.

Learning Goals

Use measurements as a decision tool to reduce uncertainty

Understand difficulty of measurement; discuss validity of measurements

Provide examples of metrics for software qualities and process

Understand limitations and dangers of decisions and incentives based on measurements

Software Engineering: Principles, practices (technical and non-technical) for **confidently** building **high-quality** software.

What does this mean?
How do we know?
→ *Measurement* and metrics are **key** concerns**.**

Outline

- Case Study: AV Vehicles
- Definitions: Measurements and Metrics
  - Examples: Code Complexity
  - Measurement scales
- Why measure?
- Risks and challenges
- Measures and incentives

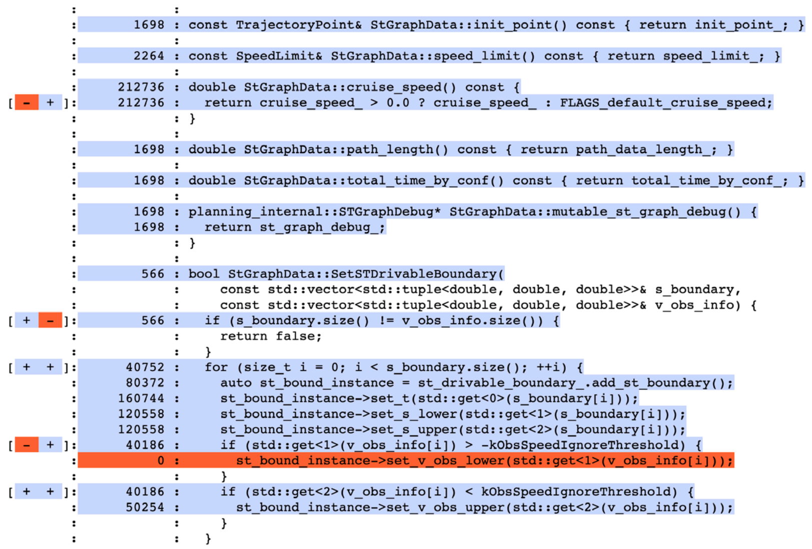
Case study: Autonomous Vehicles

AV Software is ______________________

How can we judge AV software quality (e.g. safety)?

Test coverage

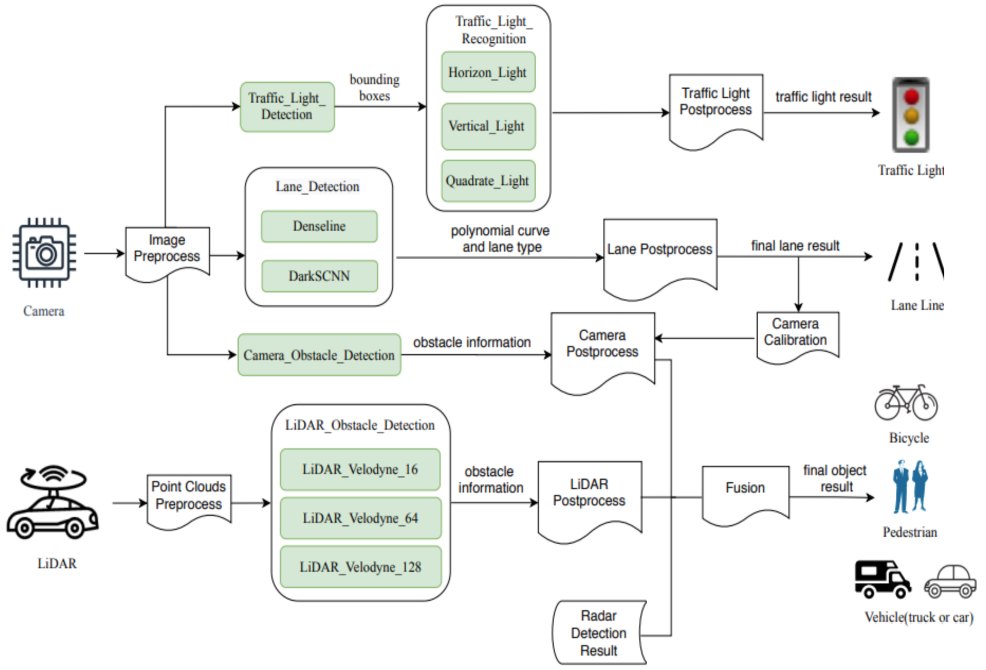
- Amount of code executed during testing.
- Statement coverage, line coverage, branch coverage, etc.
- E.g., 75% branch coverage → 3/4 if-else outcomes have been executed



```
         :          :
         :     1698 : const TrajectoryPoint& StGraphData::init_point() const { return init_point_; }
         :          :
         :     2264 : const SpeedLimit& StGraphData::speed_limit() const { return speed_limit_; }
         :          :
         :   212736 : double StGraphData::cruise_speed() const {
[ -  + ]:   212736 :   return cruise_speed_ > 0.0 ? cruise_speed_ : FLAGS_default_cruise_speed;
         :          : }
         :          :
         :     1698 : double StGraphData::path_length() const { return path_data_length_; }
         :          :
         :     1698 : double StGraphData::total_time_by_conf() const { return total_time_by_conf_; }
         :          :
         :     1698 : planning_internal::STGraphDebug* StGraphData::mutable_st_graph_debug() {
         :     1698 :   return st_graph_debug_;
         :          : }
         :          :
         :      566 : bool StGraphData::SetSTDrivableBoundary(
         :          :     const std::vector<std::tuple<double, double, double>>& s_boundary,
         :          :     const std::vector<std::tuple<double, double, double>>& v_obs_info) {
[ +  - ]:      566 :   if (s_boundary.size() != v_obs_info.size()) {
         :          :     return false;
         :          :   }
[ +  + ]:    40752 :   for (size_t i = 0; i < s_boundary.size(); ++i) {
         :    80372 :     auto st_bound_instance = st_drivable_boundary_.add_st_boundary();
         :   160744 :     st_bound_instance->set_t(std::get<0>(s_boundary[i]));
         :   120558 :     st_bound_instance->set_s_lower(std::get<1>(s_boundary[i]));
         :   120558 :     st_bound_instance->set_s_upper(std::get<2>(s_boundary[i]));
[ -  + ]:    40186 :     if (std::get<1>(v_obs_info[i]) > -kObsSpeedIgnoreThreshold) {
         :        0 :       st_bound_instance->set_v_obs_lower(std::get<1>(v_obs_info[i]));
         :          :     }
[ +  + ]:    40186 :     if (std::get<2>(v_obs_info[i]) < kObsSpeedIgnoreThreshold) {
         :    50254 :       st_bound_instance->set_v_obs_upper(std::get<2>(v_obs_info[i]));
         :          :     }
         :          :   }
```

Model Accuracy

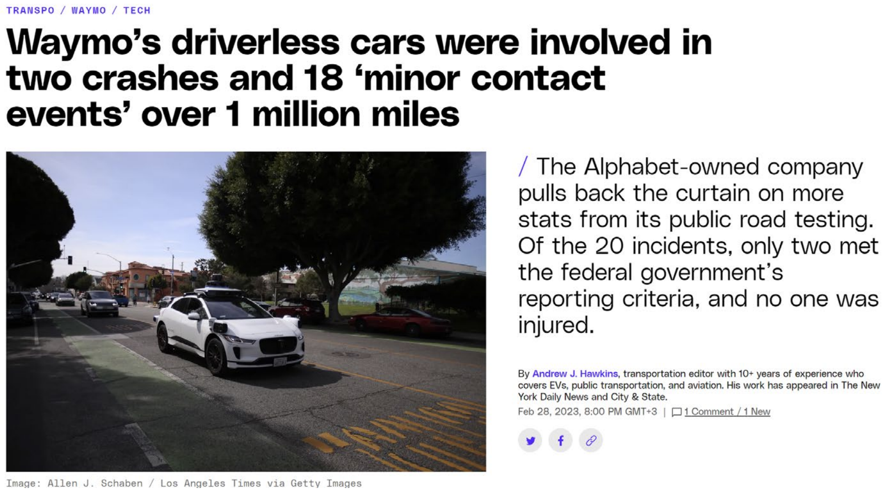
- Train machine-learning models on labelled data (sensor data + ground truth).
- Compute accuracy on a separate labelled test set.
- E.g., 90% accuracy implies that object recognition is right for 90% of the test inputs.



Source: Peng et al. ESEC/FSE'20

Failure Rate

- Frequency of crashes/fatalities
- Per 1000 rides, per million miles, per month (in the news)



TRANSPO / WAYMO / TECH

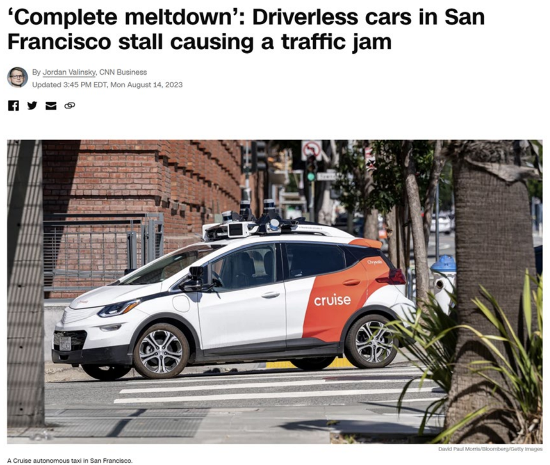
Waymo's driverless cars were involved in two crashes and 18 'minor contact events' over 1 million miles

/ The Alphabet-owned company pulls back the curtain on more stats from its public road testing. Of the 20 incidents, only two met the federal government's reporting criteria, and no one was injured.

By Andrew J. Hawkins, transportation editor with 10+ years of experience who covers EVs, public transportation, and aviation. His work has appeared in The New York Daily News and City & State.

Feb 28, 2023, 8:00 PM GMT+3 | 1 Comment / 1 New

Image: Allen J. Schaben / Los Angeles Times via Getty Images



'Complete meltdown': Driverless cars in San Francisco stall causing a traffic jam

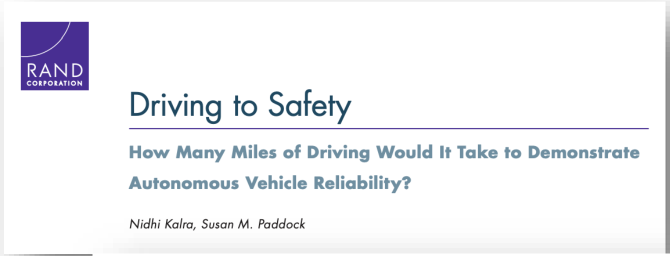
By Jordan Valinsky, CNN Business

Updated 3:45 PM EDT, Mon August 14, 2023

David Paul Morris/Bloomberg/Getty Images

A Cruise autonomous taxi in San Francisco.

Mileage



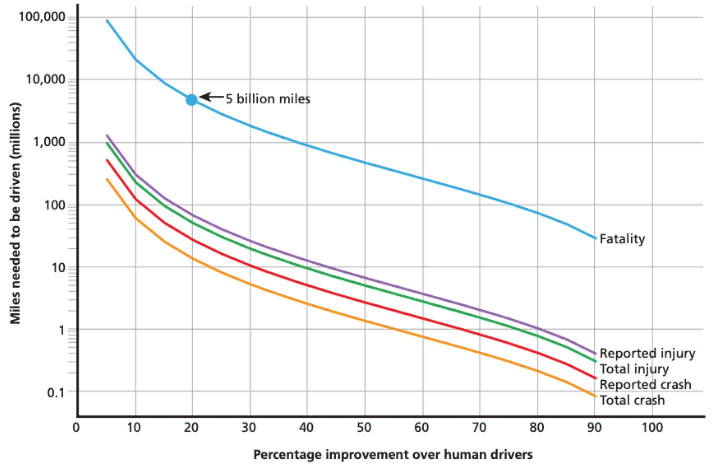
RAND CORPORATION

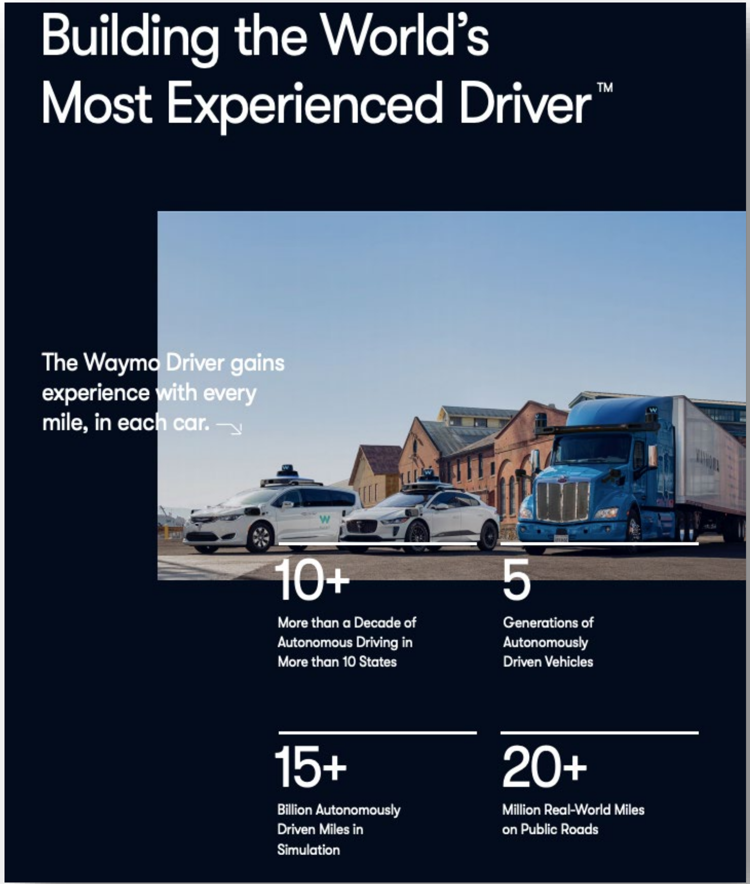
Driving to Safety

How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?

Nidhi Kalra, Susan M. Paddock

Figure 3. Miles Needed to Demonstrate with 95% Confidence that the Autonomous Vehicle Failure Rate Is Lower than the Human Driver Failure Rate





Building the World's Most Experienced Driver™

The Waymo Driver gains experience with every mile, in each car.

10+ More than a Decade of Autonomous Driving in More than 10 States

5 Generations of Autonomously Driven Vehicles

15+ Billion Autonomously Driven Miles in Simulation

20+ Million Real-World Miles on Public Roads

Source: waymo.com/safety (September 2021)

Activity

Think of “pros” and “cons” for using various metrics to judge AV software.

Test coverage	Age of codebase
Model accuracy	Time of most recent change
Failure rate	Frequency of code releases
Size of codebase	Amount of code documentation
Mileage	Number of contributors

- Group discussion, 5 minutes
- Write down two pros and two cons for three metrics on a piece of paper
- Write down your Andrew IDs.
- Keep the paper. You must hand in the paper at the end of the lecture.

Outline

- Case Study: AV Vehicles
- ➢ **Definitions: Measurements and Metrics**
    - Examples: Code Complexity
    - Measurement scales
- Why measure?
- Risks and challenges
- Measures and incentives

MEASUREMENT AND METRICS

institute for SOFTWARE RESEARCH

Carnegie Mellon University School of Computer Science

What is Measurement?

- **Measurement is the empirical, objective assignment of numbers, according to a rule derived from a model or theory, to attributes of objects or events with the intent of describing them.** – Craner, Bond, “Software Engineering Metrics: What Do They Measure and How Do We Know?”
- **A quantitatively expressed reduction of uncertainty based on one or more observations.** – Hubbard, “How to Measure Anything …”

Software Quality Metrics

- IEEE 1061 definition: **“A software quality metric is a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given attribute that affects its quality**.”
- Metrics have been proposed for many quality attributes; may define own metrics

What software qualities do we care about? (examples)

- Scalability
- Security
- Extensibility
- Documentation
- Performance
- Consistency
- Portability
- Installability
- Maintainability
- Functionality (e.g., data integrity)
- Availability
- Ease of use

What process qualities do we care about? (examples)

- On-time release
- Development speed
- Meeting efficiency
- Conformance to processes
- Time spent on rework
- Reliability of predictions
- Fairness in decision making

- Measure time, costs, actions, resources, and quality of work packages; compare with predictions
- Use information from issue trackers, communication networks, team structures, etc...

Everything is measurable

- If X is something we care about, then X, by definition, must be detectable.
    - How could we care about things like “quality,” “risk,” “security,” or “public image” if these things were totally undetectable, directly or indirectly?
    - If we have reason to care about some unknown quantity, it is because we think it corresponds to desirable or undesirable results in some way.
- If X is detectable, then it must be detectable in some amount.
    - If you can observe a thing at all, you can observe more of it or less of it
- If we can observe it in some amount, then it must be measurable.

Douglas Hubbard, How to Measure Anything, 2010

EXAMPLES:
CODE COMPLEXITY

Lines of Code

- Easy to measure

```
> wc –l file1 file2…
```

LOC	projects
450	Expression Evaluator
2,000	Sudoku
100,000	Apache Maven
500,000	Git
3,000,000	MySQL
15,000,000	gcc
50,000.000	Windows 10
2,000,000,000	Google (MonoRepo)

Normalizing Lines of Code

- Ignore comments and empty lines
- Ignore lines < 2 characters
- Pretty print source code first
- Count statements (logical lines of code)
- See also: cloc

```
for (i = 0; i < 100; i += 1) printf("hello"); /* How many lines of code is this? */
```

```
/* How many lines of code is this? */

for (
                i = 0;
                i < 100;
                i += 1
    ) {
                printf("hello");
}
```

Normalization per Language

Language	Statement factor (productivity)	Line factor
C	1	1
C++	2.5	1
Fortran	2	0.8
Java	2.5	1.5
Perl	6	6
Smalltalk	6	6.25
Python	6	6.5

Source: “Code Complete: A Practical Handbook of Software Construction“, S. McConnell, Microsoft Press (2004) and http://www.codinghorror.com/blog/2005/08/are-all-programming-languages-the-same.html u.a.

Halstead Volume

- Introduced by Maurice Howard Halstead in 1977
- Halstead Volume =
  number of operators/operands *
  log2(number of distinct operators/operands)
- Approximates size of elements and vocabulary

Halstead Volume - Example

-
  ```
  main() {
          int a, b, c, avg;
          scanf("%d %d %d", &a, &b, &c);
          avg = (a + b + c) / 3;
          printf("avg = %d", avg);
  }
  ```

Operators/Operands: main, (), {}, int, a, b, c, avg, scanf, (), "…", &, a, &, b, &, c, avg, =, a, +, b, +, c, (), /, 3, printf, (), "…", avg

Cyclomatic Complexity

- Proposed by McCabe 1976
- Based on control flow graph, measures linearly independent paths through a program
    - ~= number of decisions
    - Number of test cases needed to achieve branch coverage

```
if (c1) {
                f1();
 } else {
                f2();
 }
 if (c2) {
                f3();
 } else {
                f4();
 }
```

M = edges of CFG – nodes of CFG + 2*connected components

"For each module, either limit cyclomatic complexity to [X] or provide a written explanation of why the limit was exceeded."
– NIST Structured Testing methodology

Object-Oriented Metrics

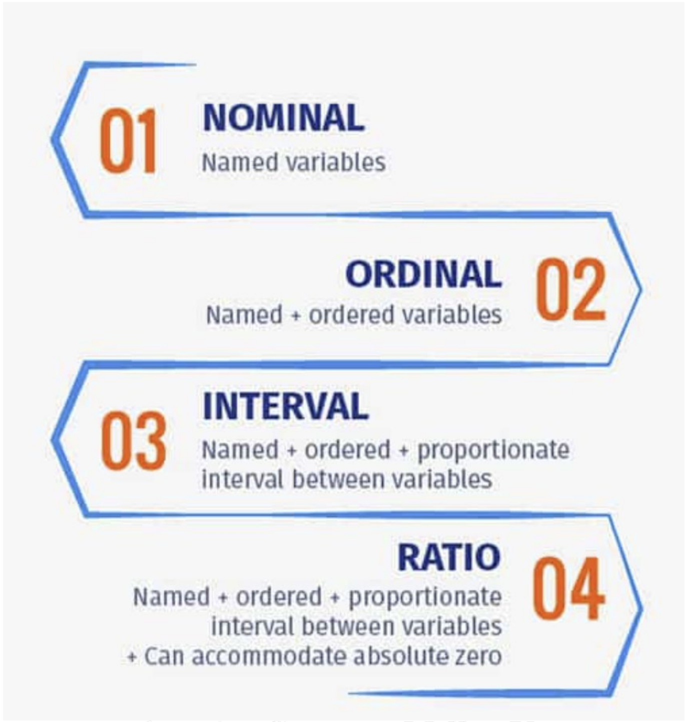
- Number of Methods per Class
- Depth of Inheritance Tree
- Number of Child Classes
- Coupling between Object Classes
- Calls to Methods in Unrelated Classes
- …

Measurement scales

Measurement scales: why they are important

- Scale: the type of data being measured.
- **The scale dictates what sorts of analysis/arithmetic is legitimate or meaningful.**
- Your options are:
  - Nominal: categories
  - Ordinal: order, but no magnitude.
  - Interval: order, magnitude, but no zero.
  - Ratio: Order, magnitude, and zero.

Measurement scales: what you can do


01
NOMINAL
Named variables
ORDINAL
Named + ordered variables
02
03
INTERVAL
Named + ordered + proportionate interval between variables
RATIO
Named + ordered + proportionate interval between variables + Can accommodate absolute zero
04

Provides:	Nominal	Ordinal	Interval	Ratio
The "order" of values is known		✔	✔	✔
"Counts," aka "Frequency of Distribution"	✔	✔	✔	✔
Mode	✔	✔	✔	✔
Median		✔	✔	✔
Mean			✔	✔
Can quantify the difference between each value			✔	✔
Can add or subtract values			✔	✔
Can multiple and divide values				✔
Has "true zero"				✔

Outline

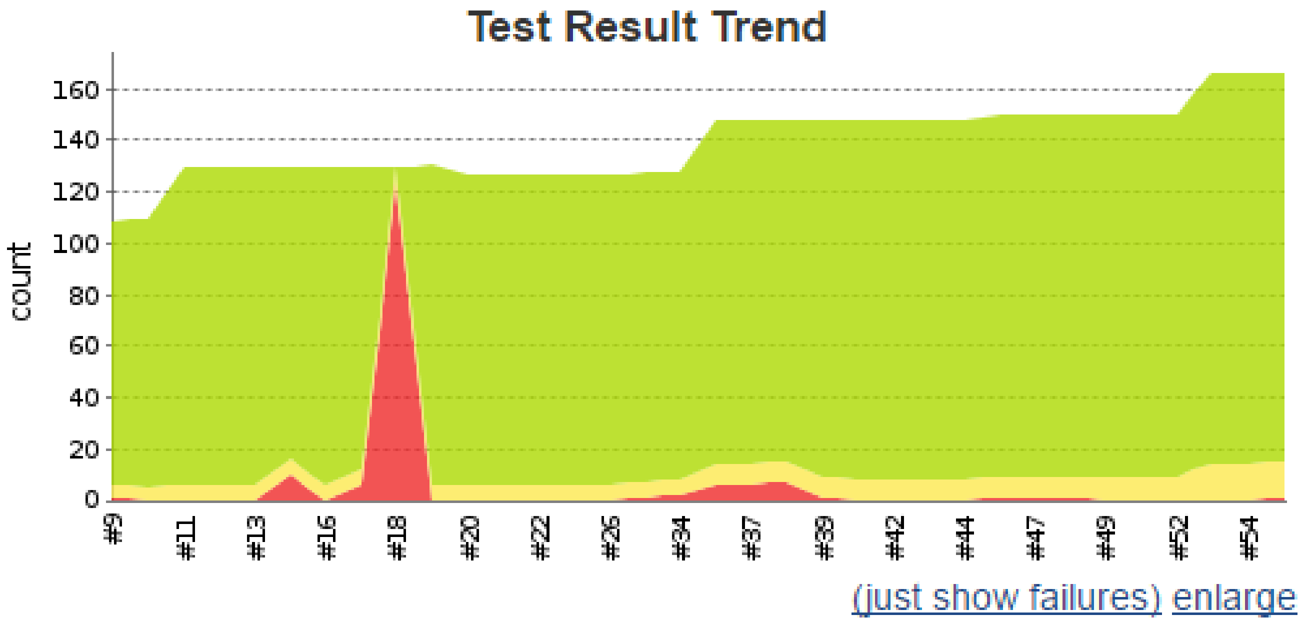
- Case Study: AV Vehicles
- Definitions: Measurements and Metrics
  - Examples: Code Complexity
  - Measurement scales
- ➢ **Why measure?**
- Risks and challenges
- Measures and incentives

WHY MEASURE?

Measurement for Decision Making

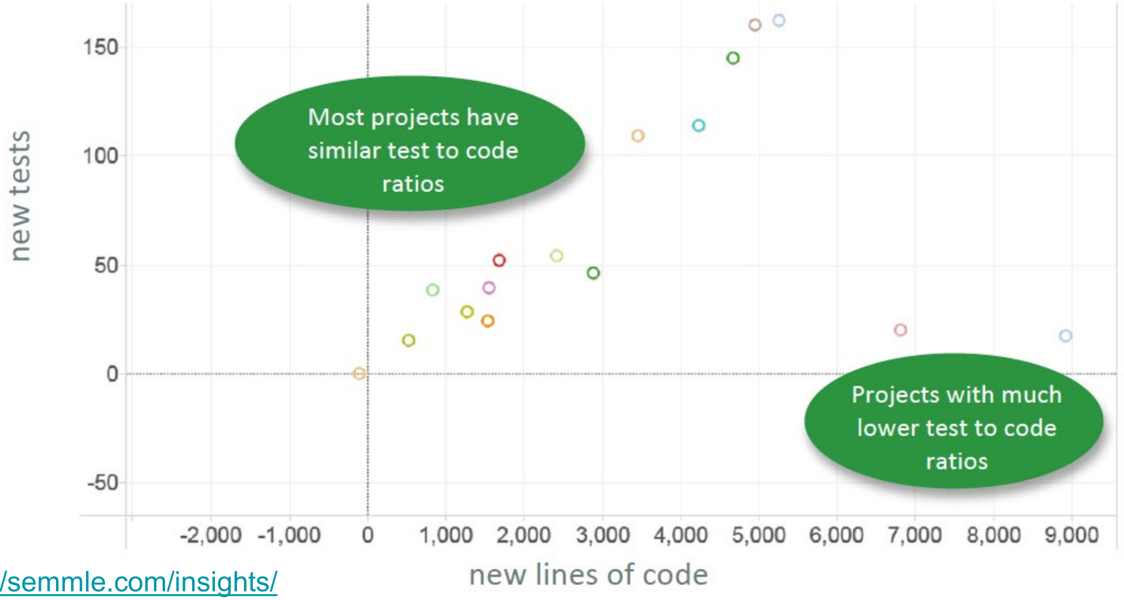
- Fund project?
- More testing?
- Fast enough? Secure enough?
- Code quality sufficient?
- Which feature to focus on?
- Developer bonus?
- Time and cost estimation? Predictions reliable?

Trend analyses



Benchmarking against standards

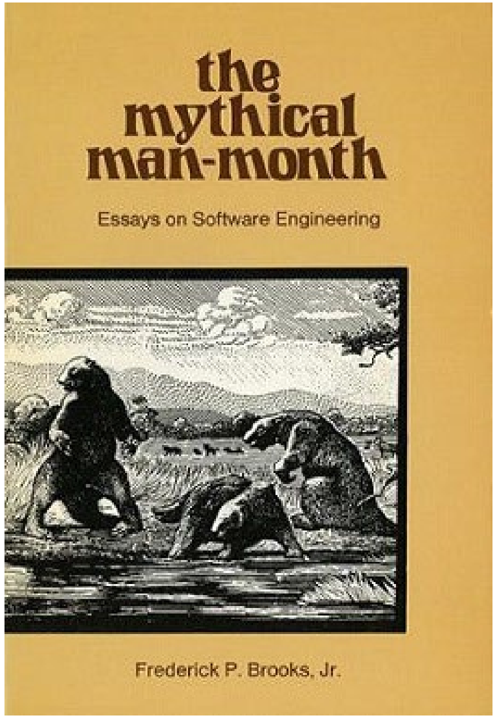
- Monitor many projects or many modules, get typical values for metrics
- Report deviations



https://semmle.com/insights/

Antipatterns in effort estimation

- IBM in the 60’s: Would account in “person-months”
  e.g. Team of 2 working 3 months = 6 person-months
- LoC ~ Person-months ~ $$$
- Brooks: “*Adding manpower to a late software project makes it later*.”



Outline

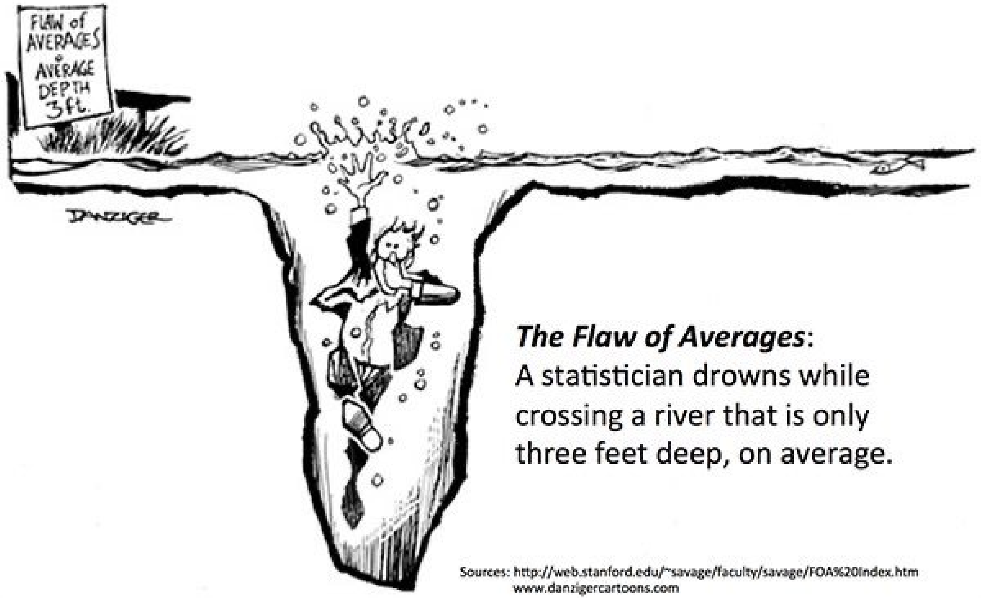
- ✓ Case Study: AV Vehicles
- ✓ Definitions: Measurements and Metrics
  - ✓ Examples: Code Complexity
  - ✓ Measurement scales
- ✓ Why measure?
- ➢ **Risks and challenges**
- ● Measures and incentives

The streetlight effect

- A known observational bias.
- People tend to look for something only where it's easiest to do so.
  - If you drop your keys at night, you'll tend to look for it under streetlights.

Bad metrics: What could possibly go wrong?

- Bad statistics: A basic misunderstanding of measurement theory and what is being measured.
- Bad decisions: The incorrect use of measurement data, leading to unintended side effects.
- Bad incentives: Disregard for the human factors, or how the cultural change of taking measurements will affect people.

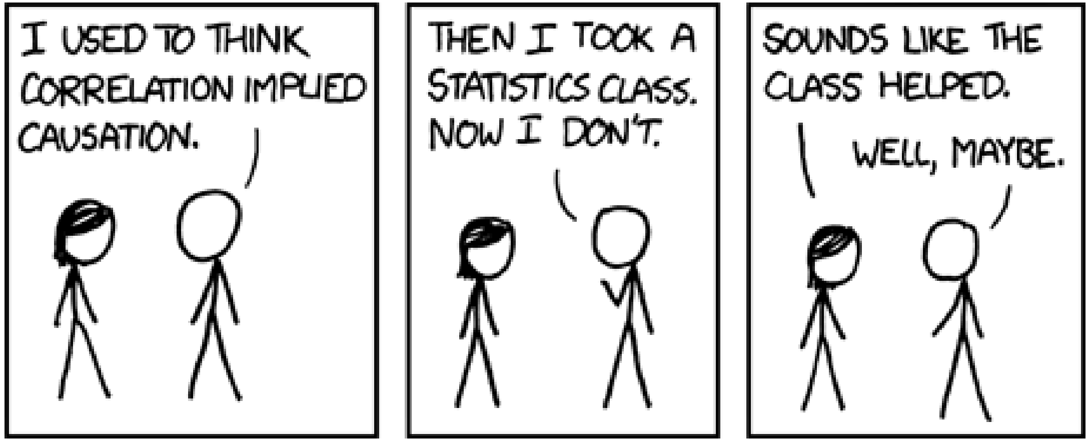


The Flaw of Averages: A statistician drowns while crossing a river that is only three feet deep, on average.

Sources: http://web.stanford.edu/~savage/faculty/savage/FOA%20Index.htm
www.danzigercartoons.com

Making inferences

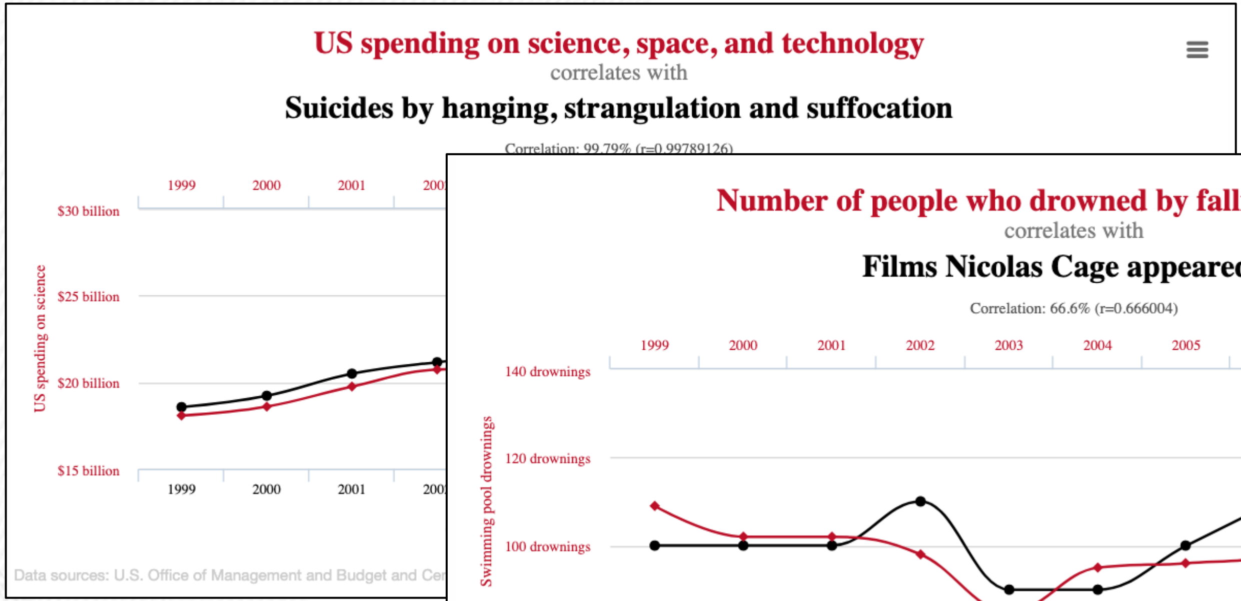
http://xkcd.com/552/

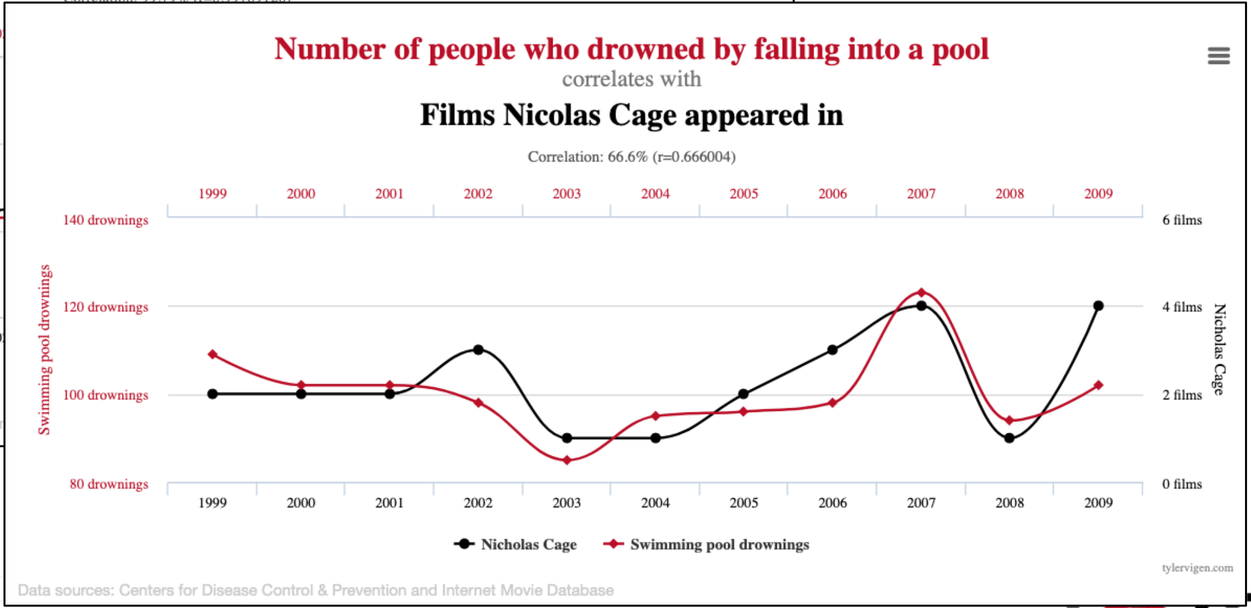

I USED TO THINK CORRELATION IMPLIED CAUSATION.
THEN I TOOK A STATISTICS CLASS. NOW I DON'T.
SOUNDS LIKE THE CLASS HELPED.
WELL, MAYBE.

To infer causation:

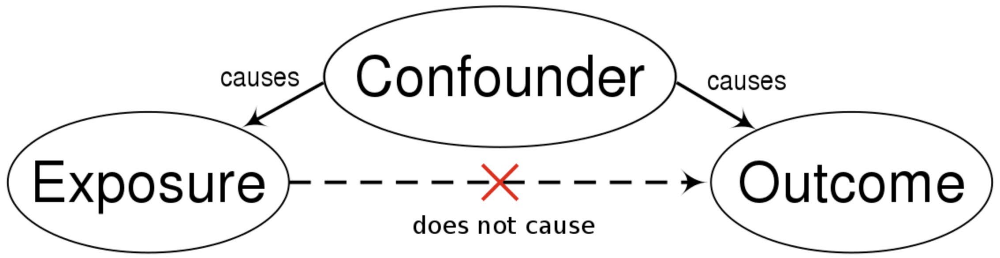
- Provide a theory (from domain knowledge, independent of data)
- Show correlation
- Demonstrate ability to predict new cases (replicate/validate)

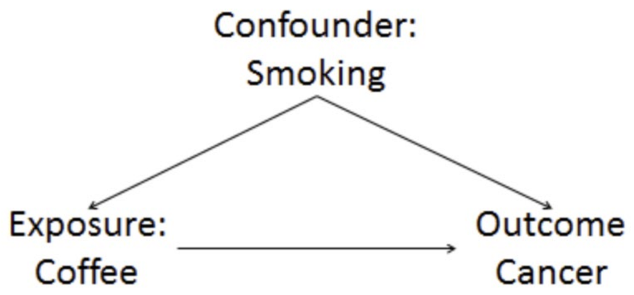
Spurious Correlations





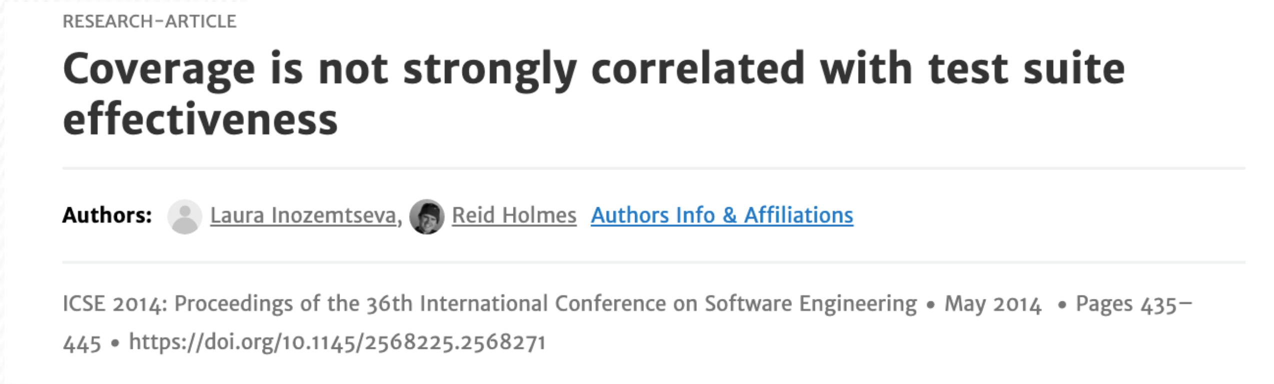
Confounding variables





- If you look only at the coffee consumption → cancer relationship, you can get very misleading results
- Smoking is a confounder

“We found that there is a low to moderate correlation between coverage and effectiveness when the number of test cases in the suite is controlled for.”


RESEARCH-ARTICLE

Coverage is not strongly correlated with test suite effectiveness

Authors: Laura Inozemtseva, Reid Holmes Authors Info & Affiliations

ICSE 2014: Proceedings of the 36th International Conference on Software Engineering • May 2014 • Pages 435–445 • https://doi.org/10.1145/2568225.2568271

Most studies did not account for the confounding influence of test suite size

Measurements validity

- *Construct validity* – Are we measuring what we intended to measure?
- *Internal validity* – The extent to which the measurement can be used to explain some other characteristic of the entity being measured
- *External validity* – Concerns the generalization of the findings to contexts and environments, other than the one studied

Measurements reliability

- Extent to which a measurement yields similar results when applied multiple times
- Goal is to reduce uncertainty, increase consistency
- Example: Performance
  - Time, memory usage
  - Cache misses, I/O operations, instruction execution count, etc.
- Law of large numbers
  - Taking multiple measurements to reduce error
  - Trade-off with cost

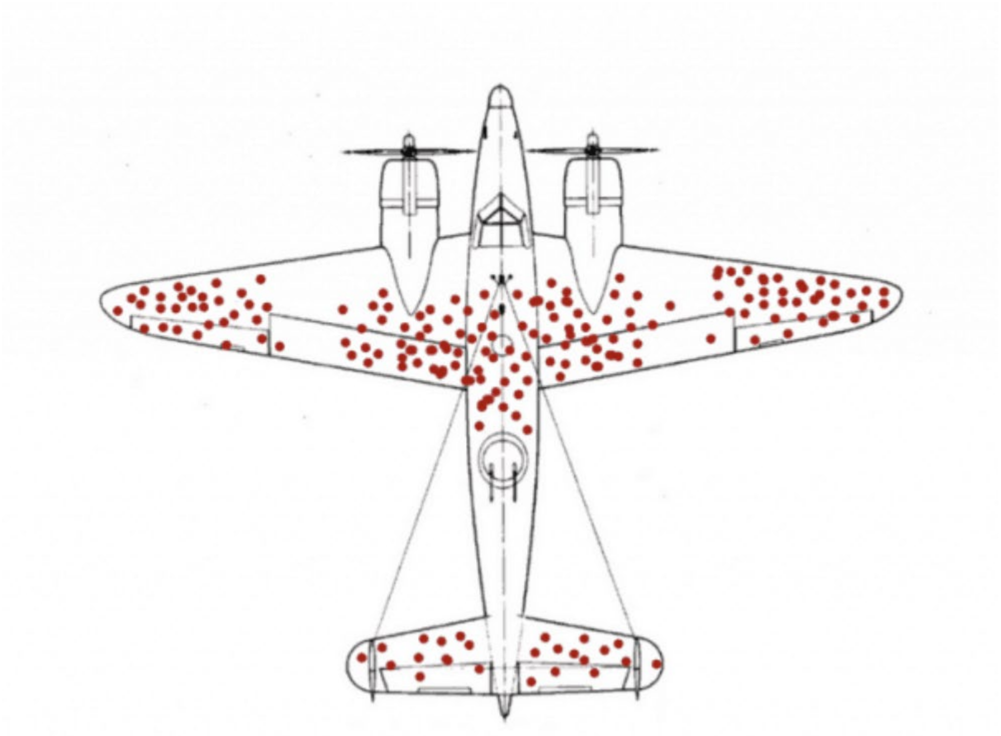
From The Director of "The Thin Blue Line" and "Gates Of Heaven"
"IF THERE IS ONE MOVIE THAT OUGHT TO BE STUDIED BY MILITARY AND CIVILIAN LEADERS AROUND THE WORLD AT THIS TREACHEROUS HISTORICAL MOMENT, IT IS 'THE FOG OF WAR', ERROL MORRIS'S PORTRAIT OF FORMER UNITED STATES DEFENSE SECRETARY, ROBERT S. MCNAMARA."
"SMART AND COMPLETELY FASCINATING!"
"STUNNING! SUPERBLY MADE!"
"SPLENDID! A MASTERPIECE!"
THE FOG OF WAR
BEST DOCUMENTARY FEATURE
Directed by Errol Morris
BEST ORIGINAL SCORE
Philip Glass
BEST EDITING
Karen Schmeer
Doug Abel
Chyld King
2003 Official Selection AFI Film Festival - Fri., Nov. 14, 2003 7:30pm

McNamara fallacy

- Measure whatever can be easily measured.
- Disregard that which cannot be measured easily.
- Presume that which cannot be measured easily is not important.
- Presume that which cannot be measured easily does not exist.

https://chronotopeblog.com/2015/04/04/the-mcnamara-fallacy-and-the-problem-with-numbers-in-education/

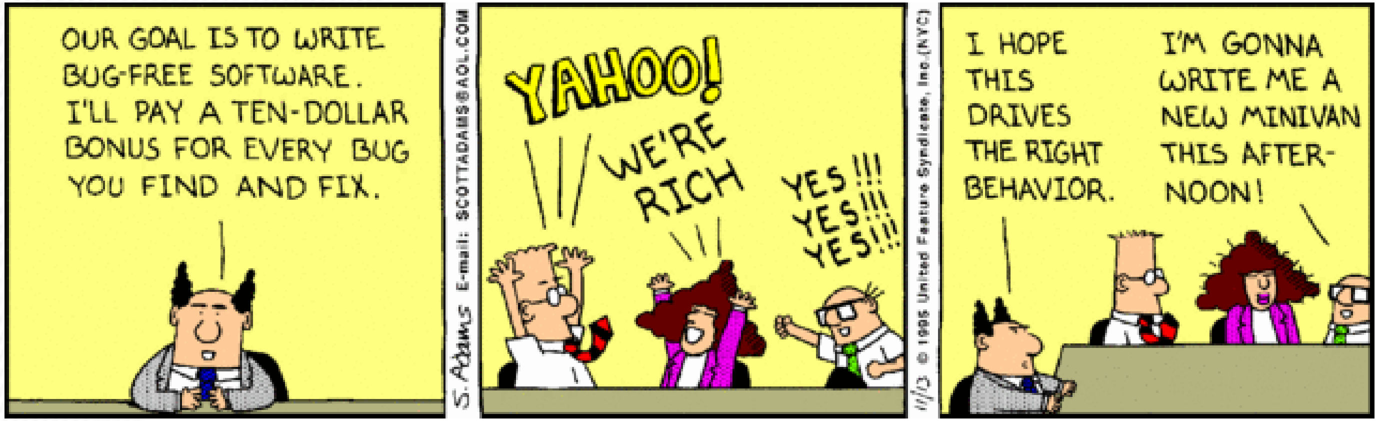
Survivorship bias



Outline

- ✓ Case Study: AV Vehicles
- ✓ Definitions: Measurements and Metrics
  - ✓ Examples: Code Complexity
  - ✓ Measurement scales
- ✓ Why measure?
- ✓ Risks and challenges
- ➢ **Measures and incentives**

Goodhart's law: "When a measure becomes a target, it ceases to be a good measure."


OUR GOAL IS TO WRITE BUG-FREE SOFTWARE. I'LL PAY A TEN-DOLLAR BONUS FOR EVERY BUG YOU FIND AND FIX.
YAHOO!
WE'RE RICH
YES!!! YES!!! YES!!!
I HOPE THIS DRIVES THE RIGHT BEHAVIOR.
I'M GONNA WRITE ME A NEW MINIVAN THIS AFTERNOON!
S. Adams E-mail: SCOTTADAMS@AOL.COM
11/13 © 1995 United Feature Syndicate, Inc.(NYC)

http://dilbert.com/strips/comic/1995-11-13/

The New York Times

The Price of Wells Fargo's Fake Account Scandal Grows by $3 Billion

The bank reached a settlement with federal prosecutors and the Securities and Exchange Commission after abusing customers.

Share full article 199

Wells Fargo used fraud to open up fake accounts and force customers into services

Productivity Metrics

- Lines of code per day?
  - Industry average 10-50 lines/day
  - Debugging + rework ca. 50% of time
- Function/object/application points per month
- Bugs fixed?
- Milestones reached?

Incentivizing Productivity

- What happens when developer bonuses are based on
  - Lines of code per day?
  - Amount of documentation written?
  - Low number of reported bugs in their code?
  - Low number of open bugs in their code?
  - High number of fixed bugs?
  - Accuracy of time estimates?

Summary

- Measurement is difficult but important for decision making
- Software metrics are easy to measure but hard to interpret, validity often not established
- Many metrics exist, often composed; pick or design suitable metrics if needed
- Careful in use: monitoring vs incentives
- Strategies beyond metrics

What you need to know

Use measurements as a decision tool to reduce uncertainty

Understand difficulty of measurement; discuss validity of measurements

Provide examples of metrics for software qualities and process

Understand limitations and dangers of decisions and incentives based on measurements

Questions to consider (HW1)

- What properties do we care about, and how do we measure it?
- What is being measured? Does it (to what degree) capture the thing you care about? What are its limitations?
- How should it be incorporated into process?
- What are potentially negative side effects or incentives?