SAND2023-05296C

Sandia National Laboratories

Exceptional service in the national interest

Modeling Scientific Software Architecture from Regression Tests using Data Mining Techniques

Is feature "X" ready for my intended use?

Matthew Mosby

With Jake Healy, Tony Nguyen and Chris Siefert

ASC Sustainable Scientific Software Conference

NLIT Summit, Milwaukee, WI June 27-30, 2023

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

IS FEATURE "X" READY FOR MY INTENDED USE?

Feature (n): user input to a scientific software program that activates a specific capability or behavior

Examples:

- Material model formulation
- Active physics (e.g., contact)
- Solver selection
- Time integration scheme
- Discretization

As a ***user*** of SciSoft, how can I be confident that a given ***feature*** is ready for use?

What evidence is available to me for deciding between two similar features?

Typical Evidence:

- Overall software test coverage
- Identify that the feature is tested
- SME assertion that the feature is ready

Is this evidence sufficient?

A (VERY) SIMPLE MOTIVATING EXAMPLE

Feature: Elastic/Linearly plastic material

Credibility Evidence:

- ✓ The overall code coverage is 90%
- ✓ The model is used in several tests
- ○ The code SME isn't available

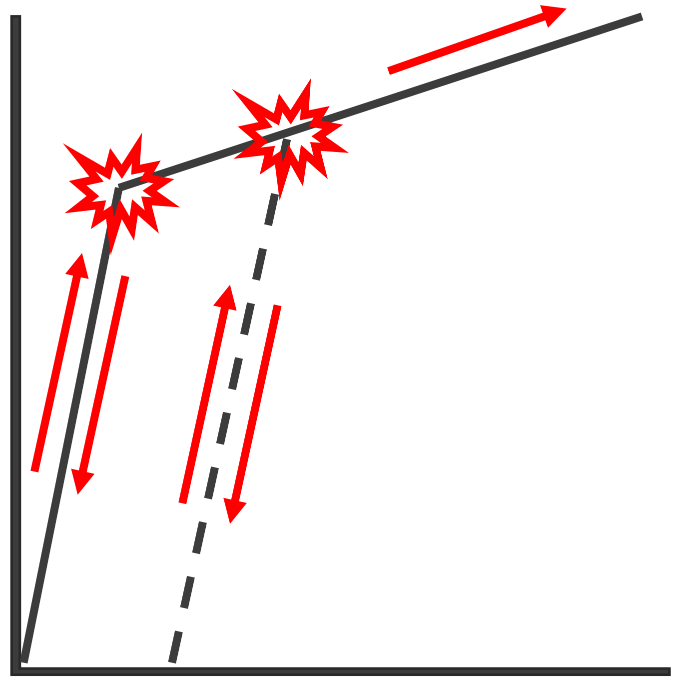
Quiz time:

How many conditions are in this model? 5

Could those branches be in the 10% missing code coverage? Absolutely

What can a user assert about the quality of tests this model was used in?

Nothing, this is why the SME is involved!



What would change if the user were presented with the following?

Estimated coverage of <feature>: 30%

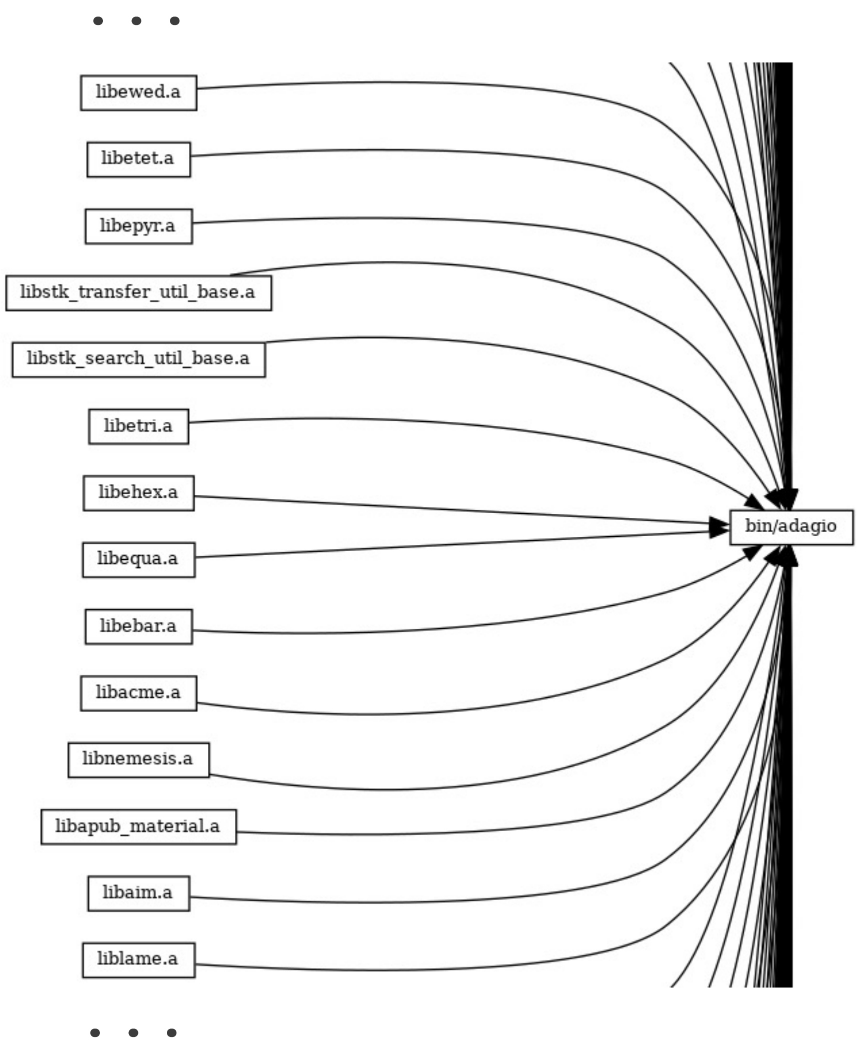
Goal: Enable such feedback

WHAT DOES ANY OF THAT HAVE TO DO WITH SOFTWARE ARCHITECTURE?

- SciSoft is complex, long-lived & changing
- SciSoft is often written by scientists and not *computer scientists*
- ***Features*** are often difficult to test in isolation

Architecture (n): the relationship between a user-facing ***feature*** and its software ***implementation***

Understanding the ***architecture*** is a prerequisite to gathering ***feature***-level readiness evidence



Library-level dependency graph of the SIERRA/SM application

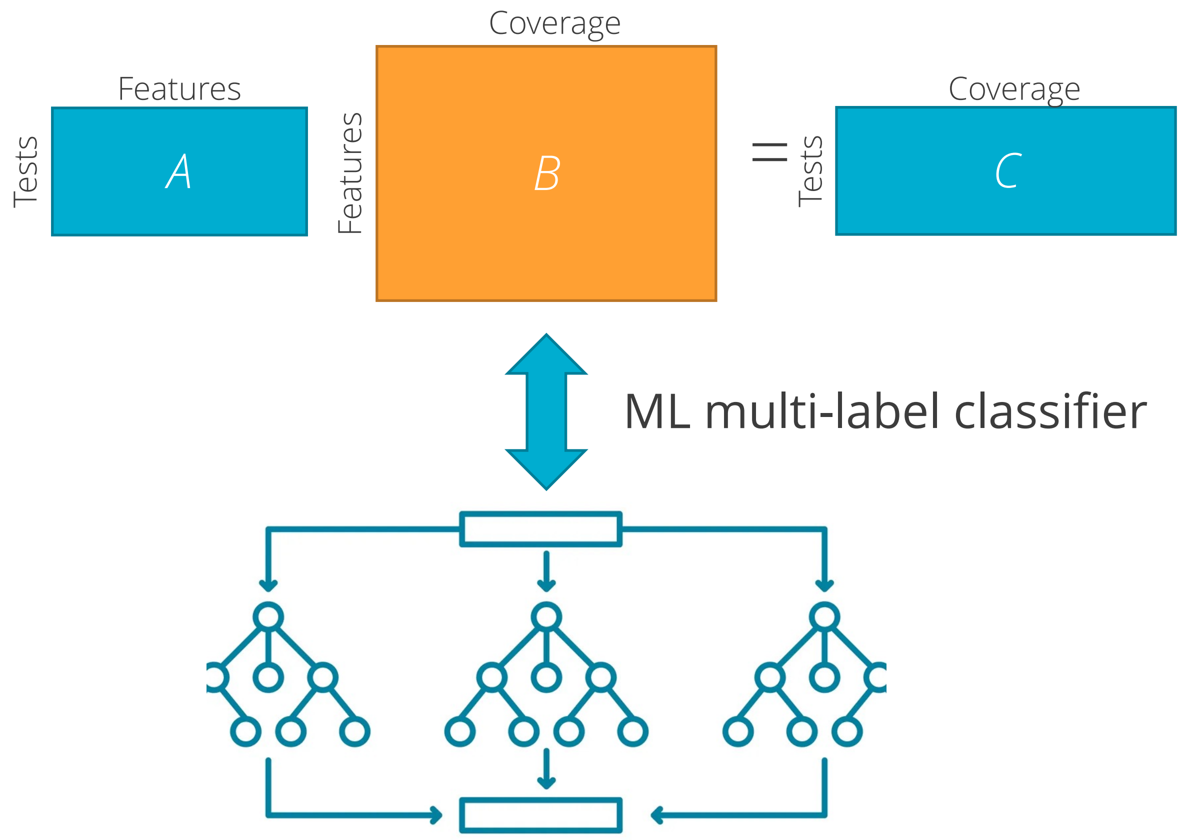
THE GENERAL APPROACH – MINE THE REGRESSION TESTS

- SciSoft typically have test suites
- Instrument the code/tests to provide:
  - Features used by an execution
  - Code coverage from an execution
- Run the instrumented test suite
- Per-test records form training data
- Apply ML algorithms to construct a model of the ***architecture***

ML multi-label classifier → "Given a feature (set), estimate the coverage set"

General form: $\min_B \|g(A,B) - C\|$

Conceptual linear form:



WHAT CONSTITUTES A FEATURE? HOW TO IDENTIFY THEM?

What

- Up to interpretation by user and/or SME
  - General, e.g., model formulation
  - Specific, e.g., sub-option/setting

How

- Feature annotation by SME or automatic
- Automatic annotation *strongly* preferred
  - Always up to date
  - Supports *user*-annotation of input
  - Extension to library-level SciSoft APIs

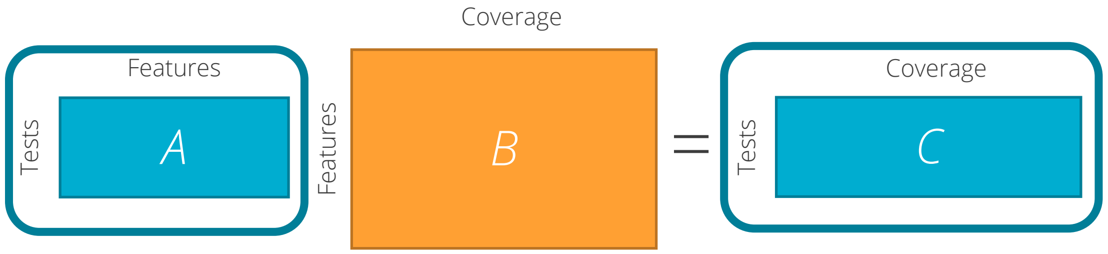
Feature identification requirements

- Feature keys generated by unique context
- Keys *don't* encode parameter/option *values*
- Keys can be mapped back to input command

Input	Key (e.g.)
`begin material steel`	0d4f3dd
`    density          = 0.000756`	b16b186
`    begin parameters for model ml_ep_fail`	698c7e7
`      youngs modulus      = {youngs}`	3a1dac5
`      poissons ratio      = {poissons}`	<...>
`      yield stress        = {yield}`	<...>
`      ...`	<...>
`    end`	<none>
`end`	<none>

FORMING THE TRAINING DATA (A, C)

Conceptual linear form:



Constructing A

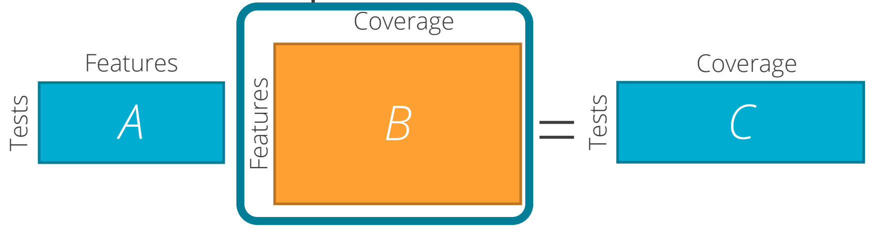
- Run tests logging what features are used
- Label columns of *A* with feature keys
- Each test results in a (sparse) row of *A*

Constructing C

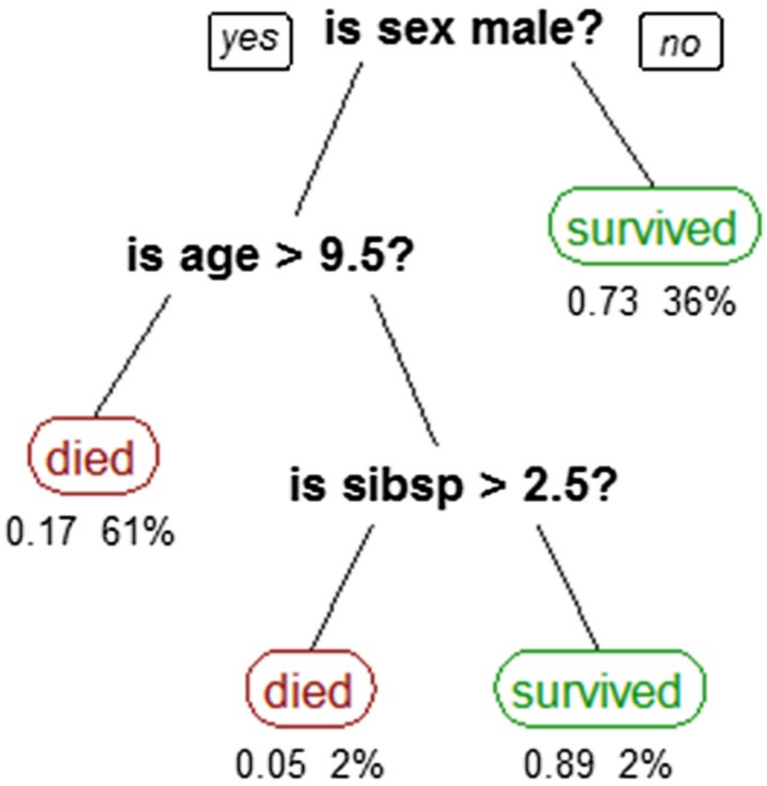
- Optional level of detail:
  - File, Function, Edge, Line
  - Greater detail increases dimension
- Run tests with coverage instrumentation
- Label columns of *C* with coverage keys
- Each test provides a (sparse) row of *C*

MODELING THE ARCHITECTURE (B)

Conceptual linear form:



- Architecture modeling can be framed as a ML *multi-label classification* problem
- We are *using* available classifiers
- Decision Tree classifiers are good for our type of data
  - Use series of splits based on parameter influence
  - Suffer from variance and bias – can be reduced by ensembles

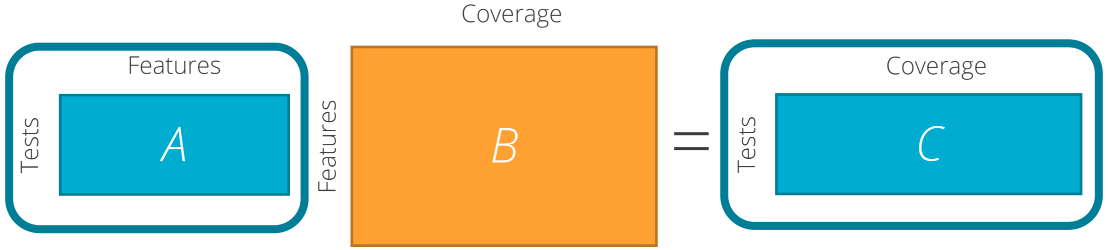


Titanic passenger survival model as a decision tree classifier [1]

[1] Lovinger, J., Valova, I. Infinite Lattice Learner: an ensemble for incremental learning. *Soft Comput* 24, 6957–6974 (2020).

ASIDE: SOFTWARE SUSTAINABILITY & CREDIBILITY BENEFITS FROM *ABILITY* TO AUTOMATICALLY GATHER THE TRAINING DATA

Conceptual linear form:



Benefits from *A*

- Feature coverage database
- Statistics on how apps used “in the wild”
- Identification of weak/untested features

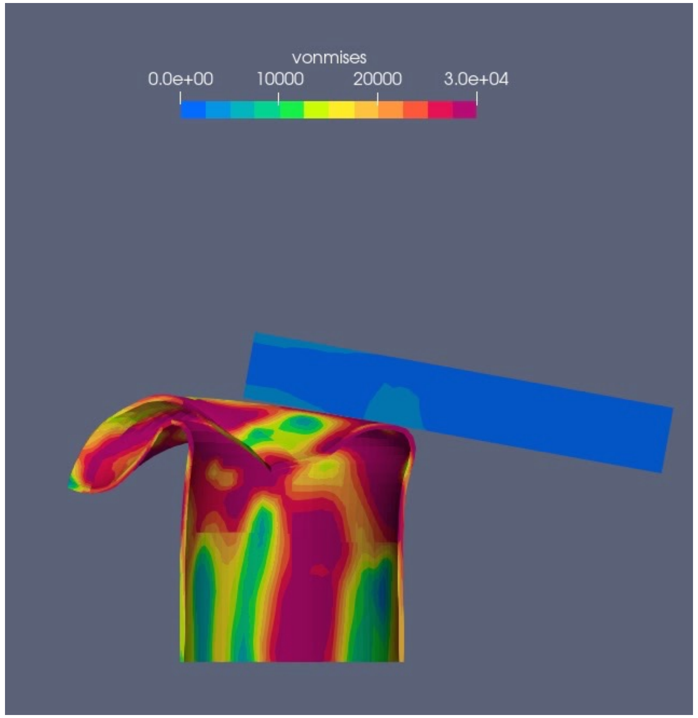
Benefits from *C*

- Supports optimal test-suite construction
  - Faster CI for large projects
  - Targeted change-based testing

Automatically gathering this data is foundational to a variety of potential user- & developer-facing credibility and productivity tools

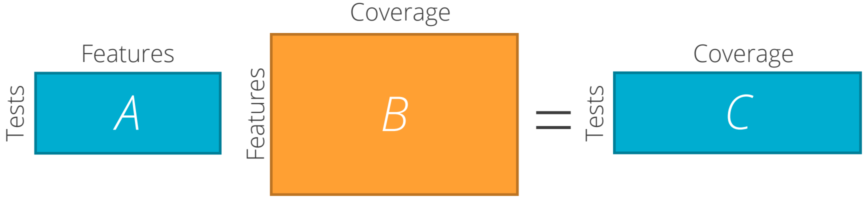
THE DATASET: SIERRA/SM (SOLID MECHANICS) APP

SIERRA Engineering Mechanics Code Suite	
Source Lines of Code (C/C++)	~2M
# Regression and unit tests	~20k
% Source line coverage	~75%



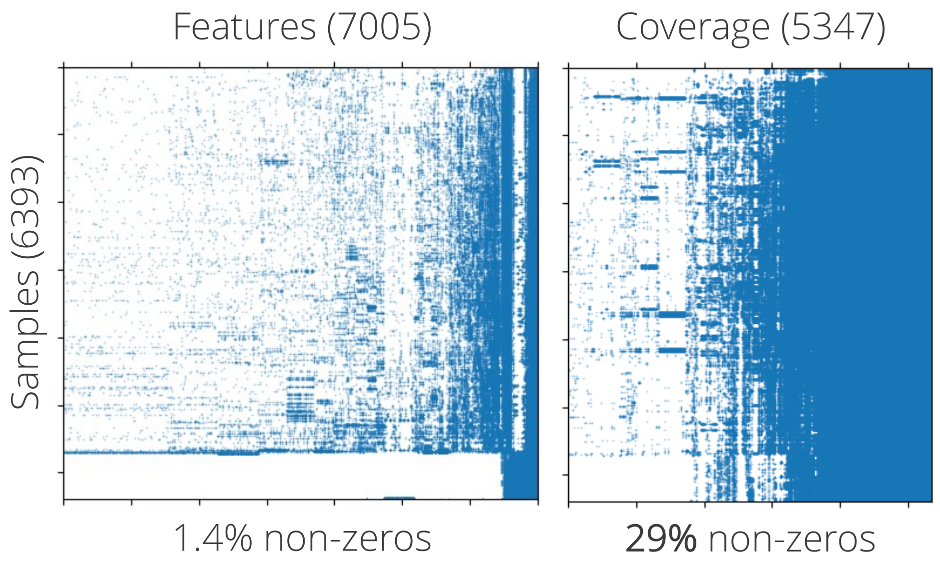
- Focus on Solid Mechanics app
- Focusing on the “small” tests in the suite
- Use Feature Coverage Tool for building *A*
- Use `LLVM CoverageSanitizer` for *B*
    - Custom callback for **file**-level coverage

Conceptual linear form:

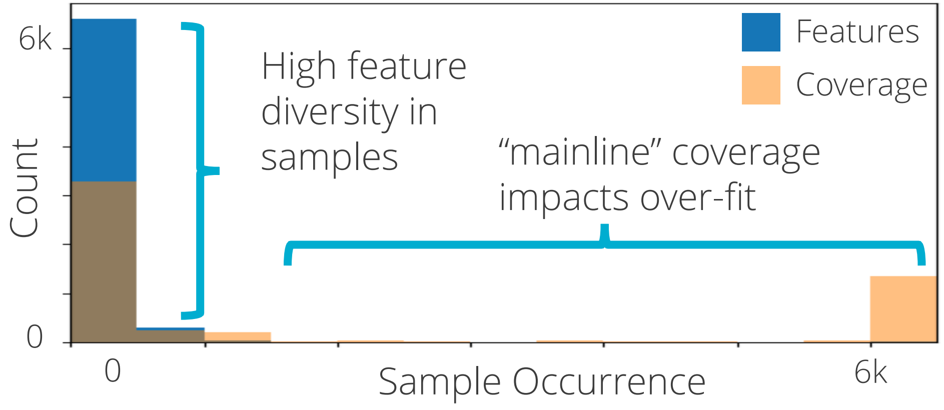


SIERRA/SM Dataset details	
# Tests (samples)	6393
# Features	7005
# Covered Files (labels)	5347

INITIAL RESULTS



Sparsity pattern of training data.
Note high coverage density.



- Examine performance of two classifiers
- Split test with all data to train/test and 30% reserved for training
- Use native "score" – fraction of 100% correct sample predictions

Classifier	Test %	Train Score	Test Score
`ExtraTrees`[1]	0%	0.64	-
	30%	0.68	0.24
`RandomForest`[2]	0%	0.64	-
	30%	0.68	0.24

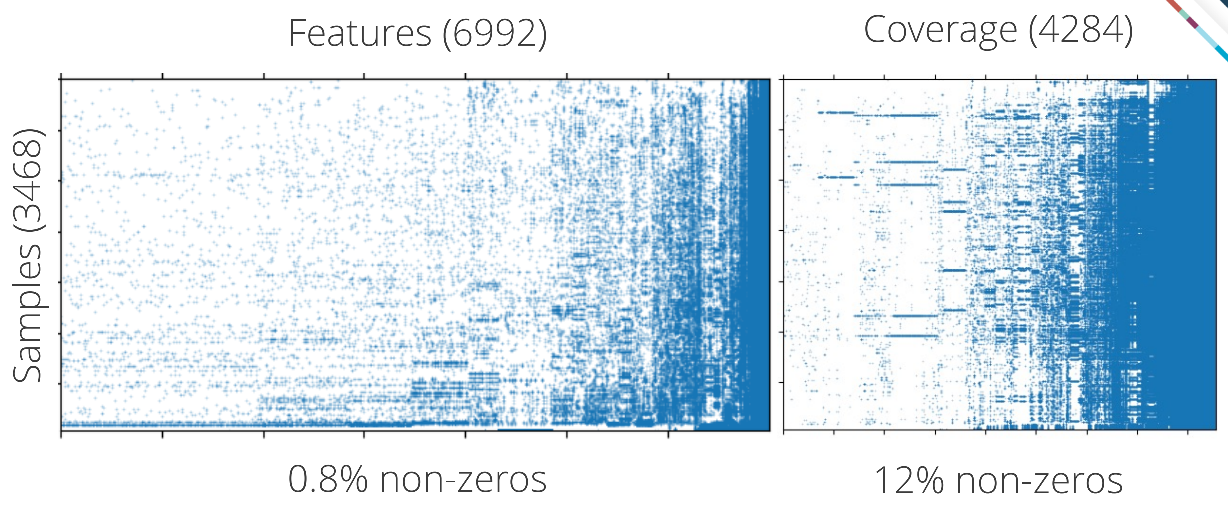
Can filtering the training data improve fit/predictions?

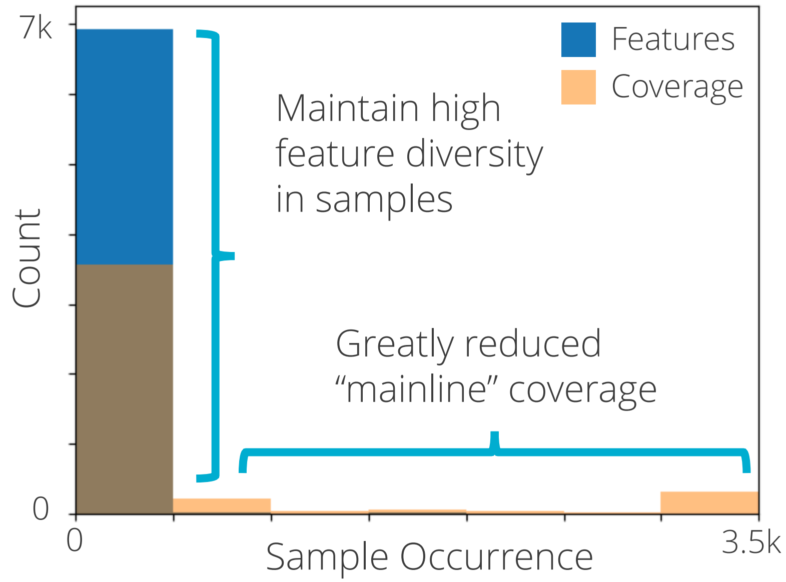
[1] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

FILTERING THE TRAINING DATASET

- Desire automatic approaches
- Identify duplicate samples
- Reduce duplicate samples with union of coverage data
  - 1-core vs. N-core cases of the same test
- Remove "mainline" features and coverage
  - Filter out columns with >= 99% fill
  - Removed features: boilerplate, e.g., 'begin sierra'
  - Removed coverage: libraries (e.g., ioss, stk), parsing

Sparsity pattern of reduced training data.





FILTERING THE TRAINING DATASET

SIERRA/SM Reduced Dataset Details	
# Tests (samples)	3468
# Features	6992
# Covered Files (labels)	4284

Classifier performance on reduced dataset

Classifier	Test %	Train Score	Test Score	Full Sample Score
ExtraTrees[1]	0%	1.0	-	0.622
	30%	1.0	0.183	-
RandomForest[2]	0%	0.999	-	0.621
	30%	0.998	0.169	-

Reduced training dataset provides better fit and maintains accuracy for the full sample set

[1] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

EXPERIMENT: IDENTIFICATION OF FEATURE SUPPORT

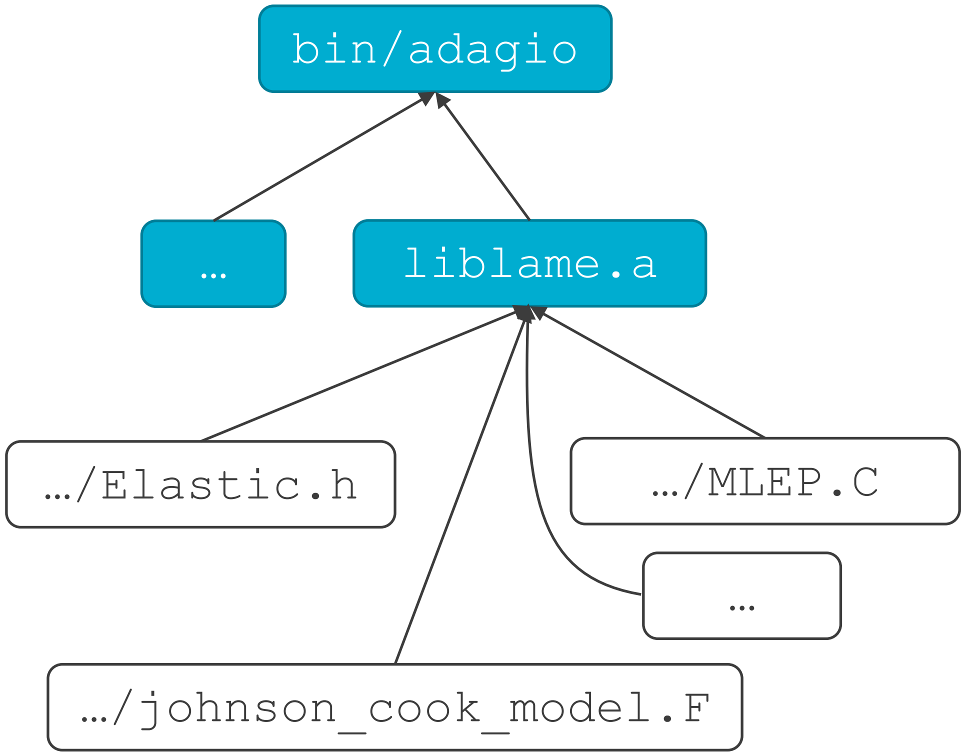
- SIERRA/SM has typical structure of a 30 year old SciSoft package
- Materials ***do*** have a well-defined interface
  - All models implemented in Lamé library
  - Source file names relate to model name
  - Able to construct/verify known label set

Given a feature key for a material model,

If the model is accurate,

Then the model will predict Lamé library files that support that model

Bonus: If the model is *precise*, the prediction doesn't contain *other* sources



RESULTS: IDENTIFICATION OF FEATURE SUPPORT

Material: `elastic`	
# Samples	**2305**
# Correct Lamé labels	7 of 7
# Wrong Lamé labels	0
# Non-Lamé labels	259

Material: `johnson_cook`	
# Samples	**61**
# Correct Lamé labels	5 of 9
# Wrong Lamé labels	2
# Non-Lamé labels	291

Material: `mlep`	
# Samples	**68**
# Correct Lamé labels	5 of 10
# Wrong Lamé labels	2
# Non-Lamé labels	245

Material: `dsa`	
# Samples	**28**
# Correct Lamé labels	5 of 7
# Wrong Lamé labels	2
# Non-Lamé labels	245

"# Samples" is the number of times the specified material appears in the training data

Too much noise/bias in the model → Predicts everything is elastic

USE SME GUIDANCE TO FOCUS TRAINING DATASET FOR MATERIALS

- Know that all material models are implemented in ‘`lame/`’ directory
- Know feature correlation, i.e., model options are associated with the specific model
- Reduce feature set to possible materials
- Reduce coverage set to files in ‘`lame/`’
- Train sub-model with reduced dataset

How might we automatically detect these reduced spaces to improve accuracy?

SIERRA/SM Material Only Dataset	
# Tests (samples)	3468
# Features (materials)	135
# Covered Files (labels)	681

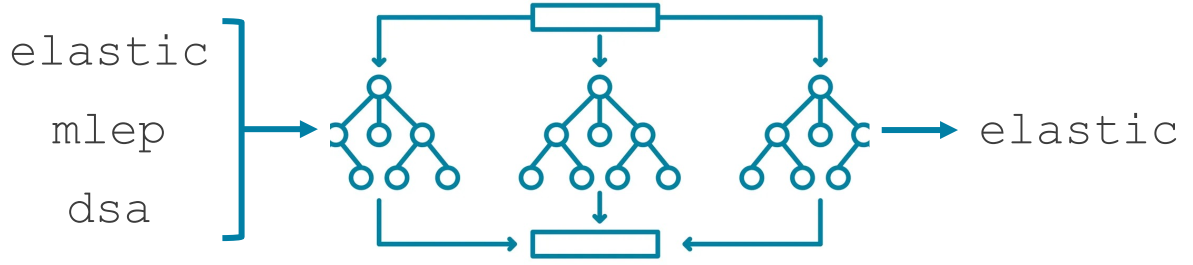
Model	# Correct
`elastic`	7 of 7
`mlep`	10 of 10
`*johnson_cook`	9 of 9
`*dsa`	7 of 7
`jc + mlep`	8 of 14

* Model never used alone in sample set

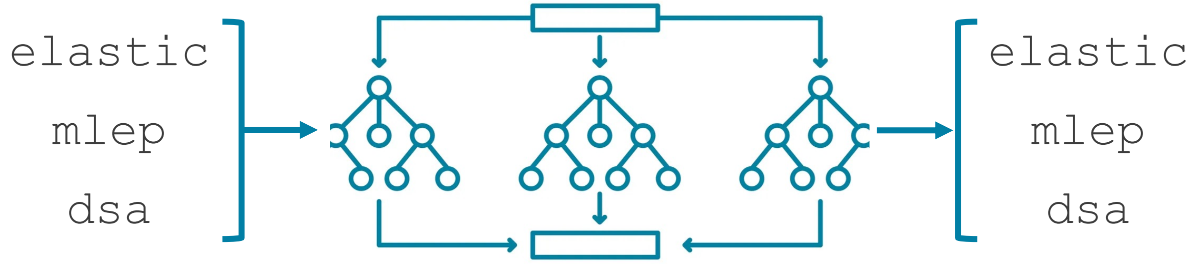
SUMMARY

- Can predict which source files cover a given input deck with ~60% accuracy
- Current model is noisy
  - Identifies a lot of library-type files
  - Overpredicts coverage for specific features
  - Bias from sample feature distribution
- Can improve new developer productivity
  - Provide pointers to where a feature is implemented, even if not super specific
- Poor predictor of specific features without sub-modeling
- Segmented models can improve accuracy

All Features Model



Materials Only Model



FUTURE WORK

Open Questions

- Can we using unsupervised learning to automatically discover correlated features and construct piecewise models spanning the feature space?
    - How could we sustainably incorporated SME knowledge?
- Is overprediction of file coverage acceptable? To what extent? Can we train to this metric?

Next Steps to Provide User Feedback

- Query full coverage data given files supporting a feature
- Develop coverage metric meaningful to an end user
- Integrate information into other user-facing credibility tools

ACKNOWLEDGEMENTS

Jake Healy

Tony Nguyen

Chris Siefert

LABORATORY DIRECTED RESEARCH & DEVELOPMENT

WHERE INNOVATION BEGINS

CIS Project 229302

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC (NTESS), a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration (DOE/NNSA) under contract DE-NA0003525. This written work is authored by an employee of NTESS. The employee, not NTESS, owns the right, title and interest in and to the written work and is responsible for its contents. Any subjective views or opinions that might be expressed in the written work do not necessarily represent the views of the U.S. Government. The publisher acknowledges that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this written work or allow others to do so, for U.S. Government purposes. The DOE will provide public access to results of federally sponsored research in accordance with the DOE Public Access Plan.

THANKS!