Global Sensitivity Analysis of xLPR using Metamodeling

Presented To:

CNSC-IAEA 2nd International Seminar on Probabilistic Methodologies for Nuclear Applications
Ottawa, Ontario, Canada

Presented By:

Christopher Casarez
Dominion Engineering, Inc.

October 26, 2017

Additional Authors:

Thomas Ligon, Markus Burkardt, Glenn White
Dominion Engineering, Inc.

Craig Harrington
Electric Power Research Institute

Dominion Engineering, Inc.
12100 Sunrise Valley Dr. #220
Reston, VA 20191
703.657.7300
www.domeng.com

Background

Motivation

- xLPR (eXtremely Low Probability of Rupture) is a complex probabilistic model for evaluating leak-before-break (LBB) in large dissimilar metal welds with active degradation mechanisms in US nuclear power plants
- The xLPR team is currently working on applying the code to the LBB problem in the US
- As part of applying this model to production analyses and to further validate the model, sensitivity analyses are being conducted

Background

Sensitivity Analysis

- Reasons to perform a sensitivity analysis:
  - Identify inputs that warrant greatest level of scrutiny, validation, and further sensitivity analysis
  - Identify inputs that are not key to the results
  - Model validation
  - Improve understanding of model behavior
  - Reduction of model complexity (e.g., set “unimportant” inputs to constant values)
  - Inform advanced Monte Carlo sampling strategies (e.g., importance sampling)
- Available techniques:
  - One-at-a-time
  - Local partial derivatives (e.g., Adjoint Modeling)
  - Variance-based (e.g., Sobol method)
  - Linear regression
  - Metamodels

Background

Sensitivity Analysis using Metamodels

- Why machine learning metamodeling?
  - Can handle correlated inputs
  - Accurately reflects non-monotonicity, non-linearity, and interactions
  - Importance measures reflect the whole input space
  - Several machine learning models automatically generate sensitivity metrics and down-select input variables based on information gained as part of the model fitting process
  - Fitted model can be used in place of the original model to compute quantitative sensitivity measures at lower computational cost
- Focus of this presentation: using built-in sensitivity metrics generated during fitting

Background

Analysis Activities

- Run the probabilistic code and collect results
- Implement metamodeling code
  - Import results from probabilistic code runs
  - Transform results to prepare for input to metamodel fitting (e.g., accounting for spatially sampled variables)
  - Fit the metamodel, including parameter optimization using cross-validation
  - Extract and report input importance metrics
- Evaluate
  - Examine goodness of fit metrics
  - Compare importance ranking results from alternate metamodels
  - Compare importance ranking results across different outputs of interest
- Iterate
  - Collect more inputs
  - Analyze different outputs
  - Run different discrete configurations of the probabilistic code
  - Use different metamodels / different metamodel parameters

Machine Learning Models

Selection and Implementation

- Python 3.6 using Scikit Learn Package*
- Machine learning models implemented:
  - Gradient Boosting Decision Trees
  - Random Forest Decision Trees
  - Linear Support Vector Machines
- All models used are classifiers (as opposed to regressors)
- All models include metrics for feature selection / feature importance

*Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

Metamodeling xLPR

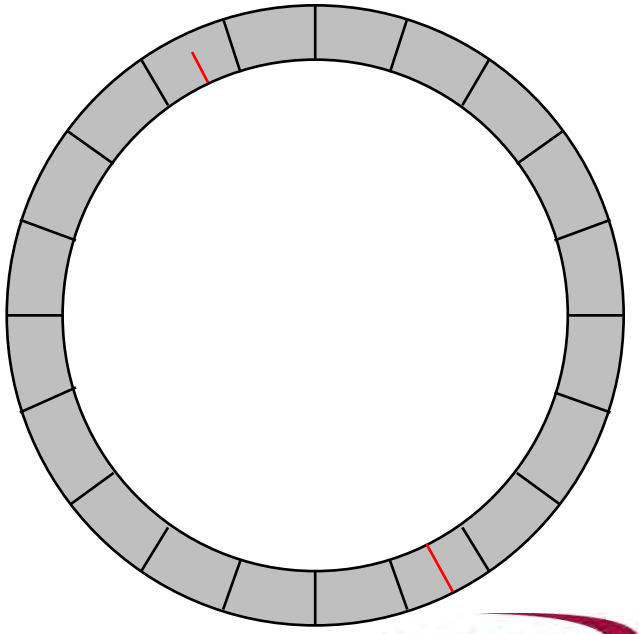
Data Analyzed

- Initial work focused on subset of 60 inputs:
  - Inputs that are expected to have high importance
  - Distributed inputs
  - Constant inputs uniformly distributed from 0.8 to 1.2 times constant value
- Outputs analyzed:
  - Occurrence leak
  - Occurrence rupture (with and without ISI)

Metamodeling xLPR

Spatially Distributed Inputs / Outputs

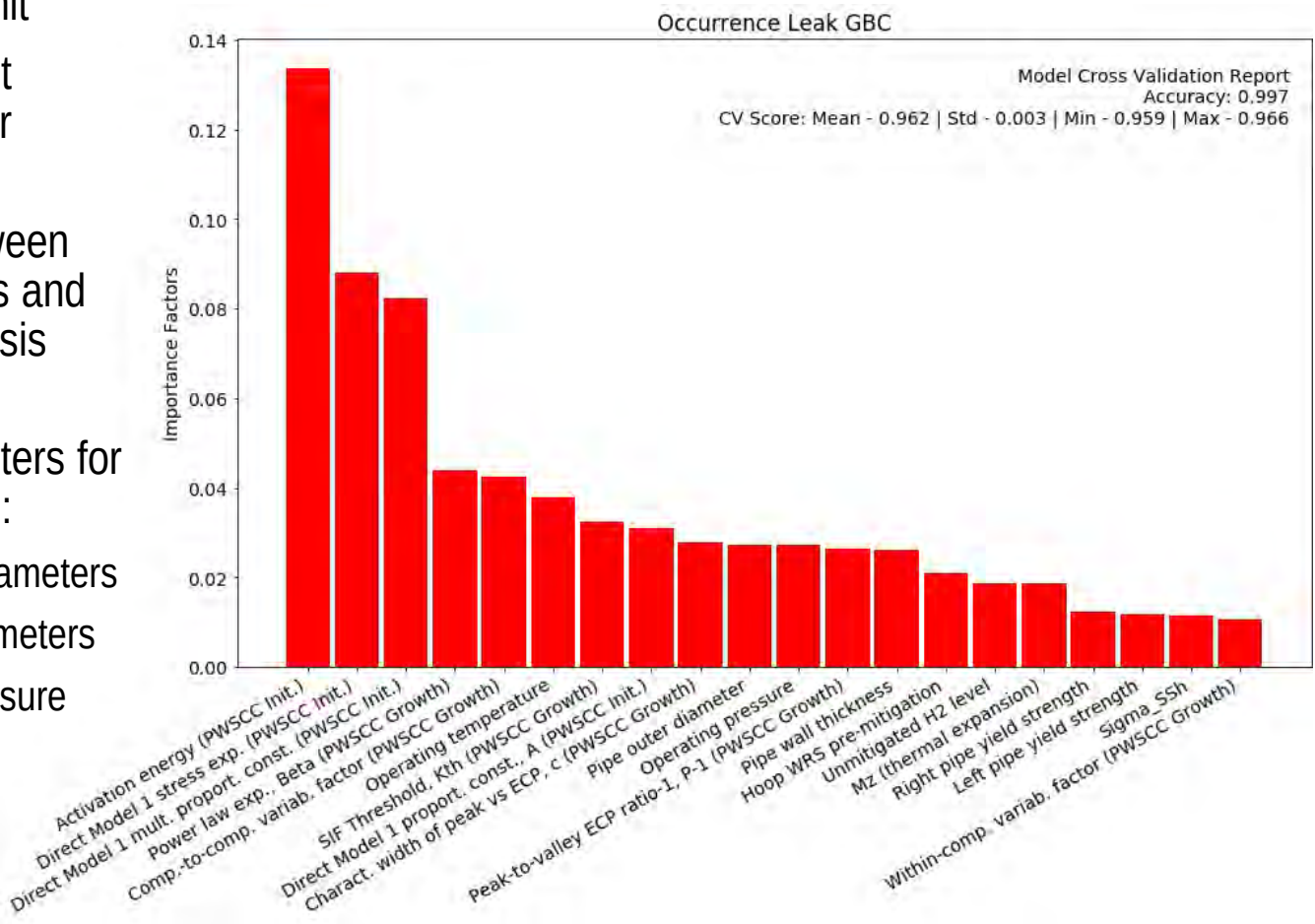
- Pipe section split into 19 subunits that can potentially crack
- Some inputs sampled on a subunit basis
- Some outputs also available on a subunit basis
- Aggregation methodology for subunit inputs / outputs
  - Pipe subunit inputs and outputs: Analyze each pipe subunit and crack direction separately and average feature importance metrics
  - Pipe subunit inputs and global outputs: Average input across all pipe subunits (and crack types) and perform single analysis to determine feature importance



Metamodeling xLPR

Leak Output

- Output: Leak (through wall crack) in any pipe subunit
- Analyzed using Gradient Boosted Trees Classifier (GBC)
- Allows comparison between averaging subunit inputs and averaging subunit analysis outputs
- Top importance parameters for averaged subunit inputs:
  - PWSCC initiation parameters
  - PWSCC growth parameters
  - Operating Temp/Pressure
  - Pipe OD / Thickness
  - WRS (Hoop)
  - Pipe yield strength

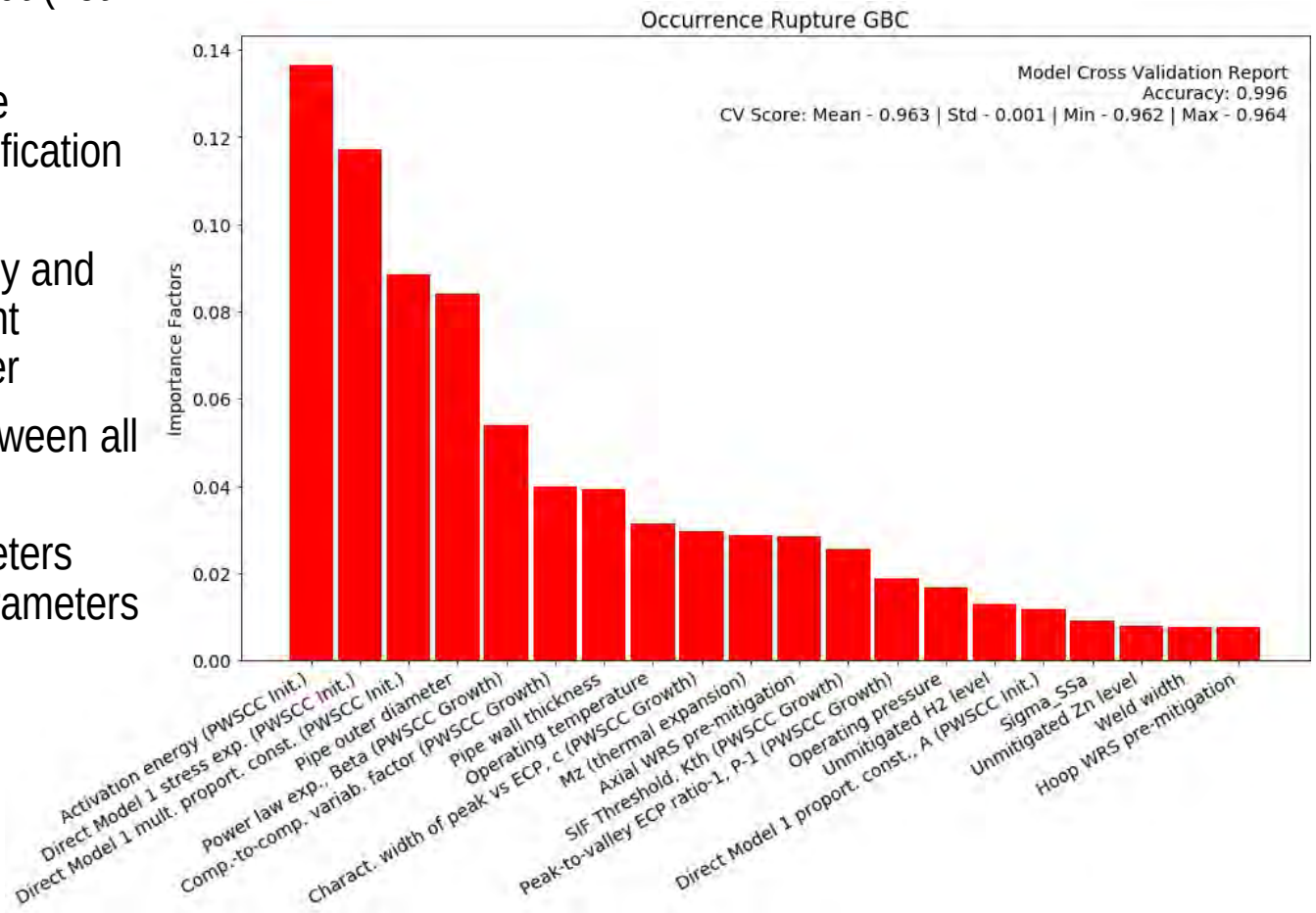


Dominion Engineering, Inc.

Metamodeling xLPR

Rupture Output

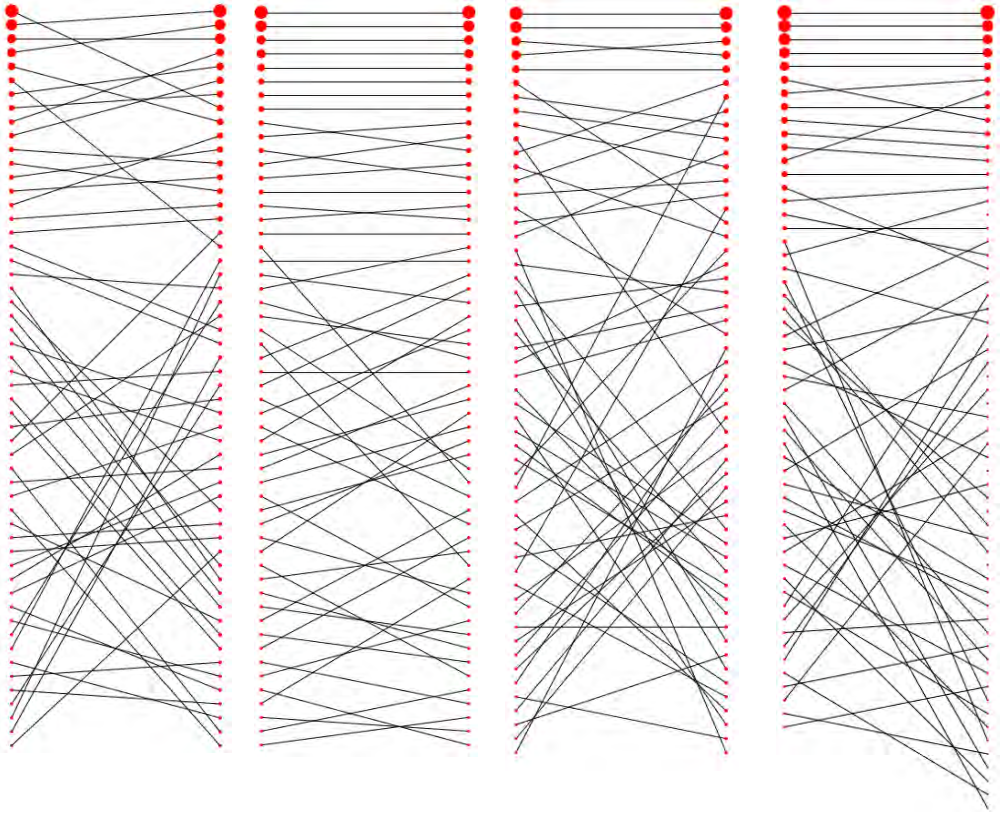
- Rupture full model output (not subunit basis)
- Analyzed using all three machine learning classification algorithms
- Best prediction accuracy and CV score using Gradient Boosted Trees Classifier
- General agreement between all three fitted models
- Top importance parameters consistent with leak parameters
  - PWSCC initiation
  - Axial WRS ranked above Hoop (opposite of leak)



Metamodeling xLPR

Results Visualization

- Importance factor results compared between two analyses to show changes in the relative ordering of inputs
- Useful for:
  - Comparison between alternate metamodeling approaches
  - Determining differences in sensitivity between different outputs of interest
  - Comparing runs with different model settings (e.g., different ISI intervals)

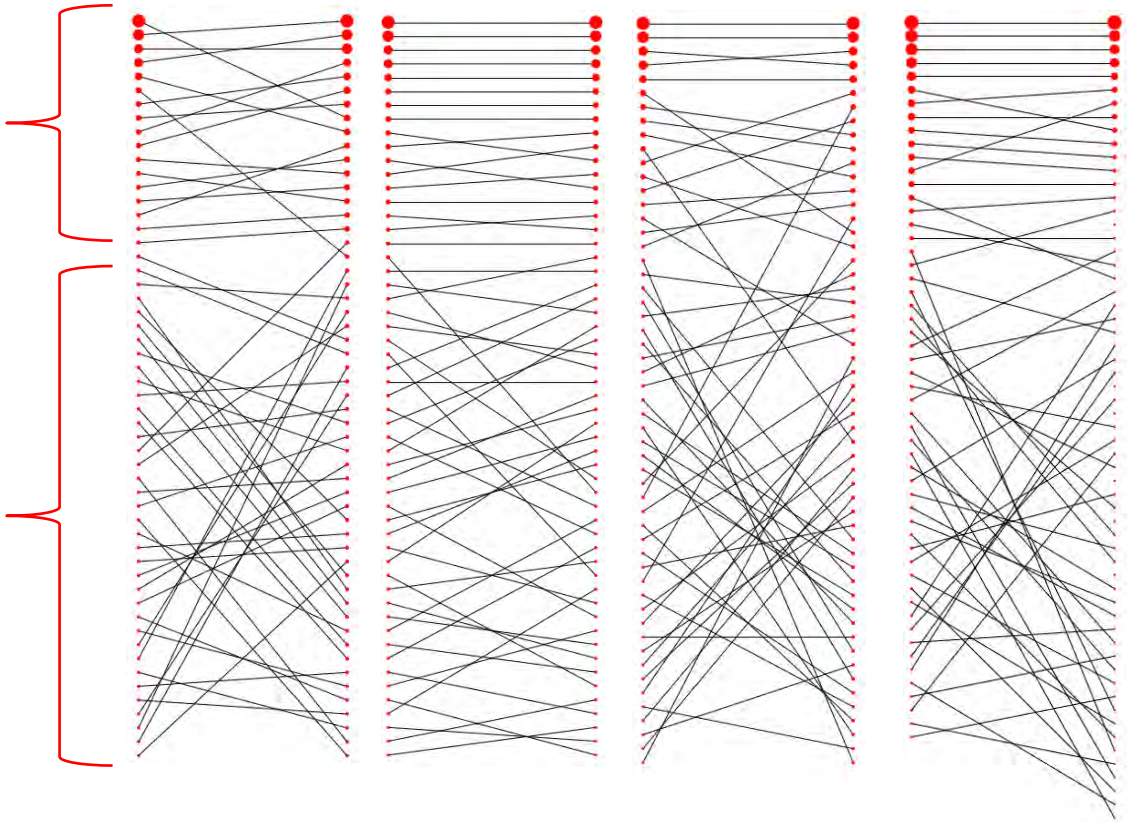


Metamodeling xLPR

Results Visualization

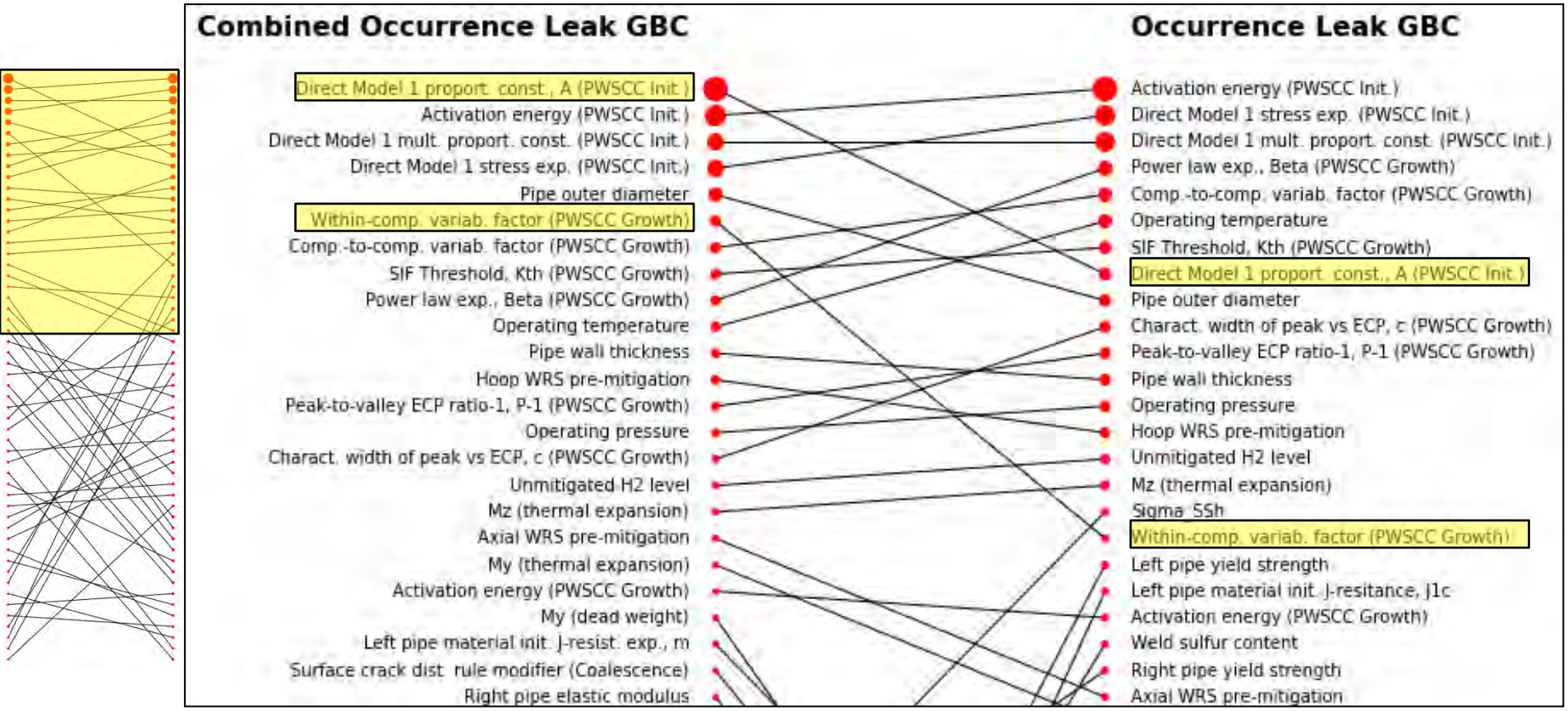
Most important inputs consistently drive result

Scatter indicates low confidence in relative ranking (“in the noise”)



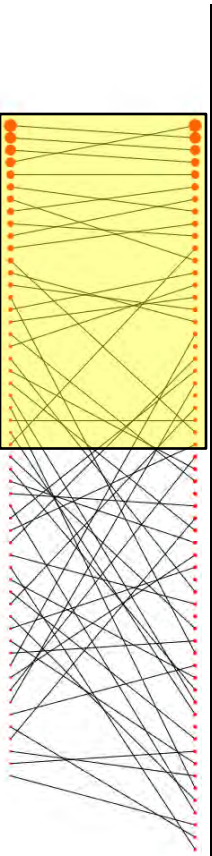
Metamodeling xLPR

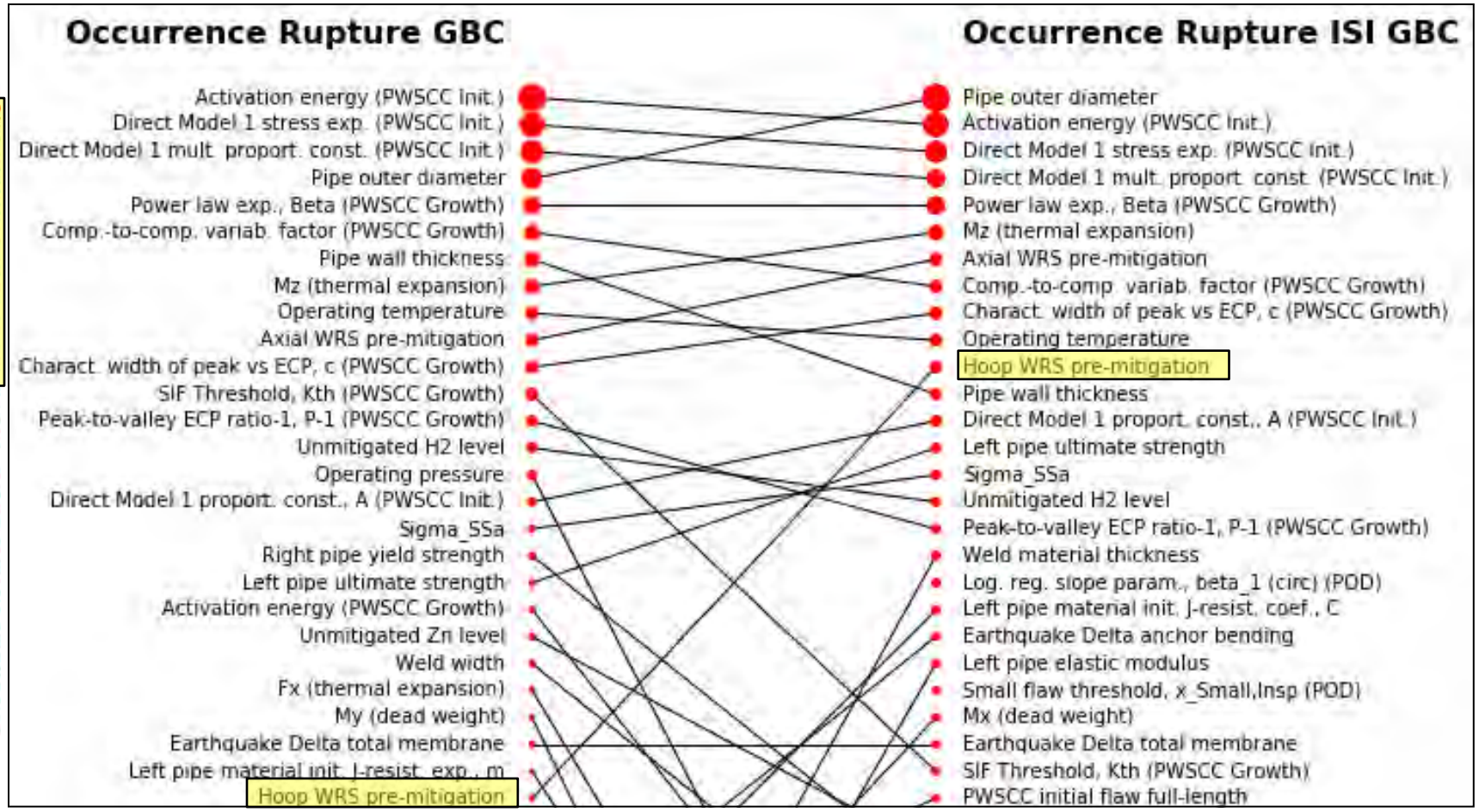
Leak Comparison – Combined Subunit Results vs Input Averaging



Metamodeling xLPR

Rupture Comparison – No ISI vs. ISI





Metamodeling xLPR

Conclusions

- Key findings
  - Relative comparisons (e.g. Axial vs. Circ, Rupture with/without ISI) are very useful for sanity checking the model
  - Relatively high confidence in the identification of highest-impact inputs but low confidence in ordering of low-impact inputs
- General challenges
  - Input distributions need to be selected carefully to get informative results
    - A default real-world analysis input set is probably not sufficient
  - Special consideration needed for inputs that are not continuous variables (e.g., settings flags)
- xLPR-specific challenges
  - Prediction of simulation-wide outcomes using subunit-level sampled values
  - Consideration of all inputs would be time-intensive (labor to extract sampled values and simulation time to adequately cover full input space)

Metamodeling xLPR

Potential Future Improvements

- Include more inputs in the machine learning model
- Examine other outputs of interest (e.g., leak rate jump indicator)
- Examine alternate configurations that can't be covered automatically using input distributions
- Use more advanced methods to improve on the relative rank importance metric (e.g., variance decomposition)

Questions?

Credit: XKCD, https://xkcd.com/1838/

Backup Slides

Machine Learning Models

Optimizing Model Fitting

- Machine learning algorithms include parameters to control how models fit to data
- Cross validation used to optimize model parameters to achieve good prediction while minimizing overfitting
  - Splits the input data (xLPR realizations) into *N* random equal folds (sets)
  - Machine learning model fit to *N*-1 folds
  - Model used to predict outcomes for data in the unfitted fold and scored based on prediction accuracy
  - Process is repeated for the N fold combinations to determine an aggregate score
  - Low likelihood of overfitting if high prediction accuracy of unfitted data

Machine Learning Models

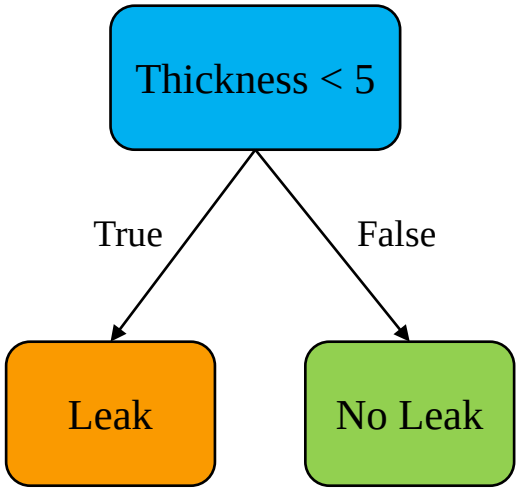
Feature Selection

- Feature selection is used to reduce number of inputs used to fit a model to a set of data
  - The feature selection methods highlight the inputs for which the metamodel prediction is more sensitive
- Methods include:
  - Feature importance: subset of machine learning algorithms directly provide metrics for relative importance of (input) features on model prediction
  - Recursive feature elimination (RFE): series of regression fits using a machine learning model where least important input features for model prediction are incrementally removed from sequential regressed fits
  - Principal component analysis (PCA): statistical procedure that transforms the input matrix (that possibly contains correlated variables) into a set of linearly uncorrelated “principal components”

Machine Learning Models

Decision Tree based Models

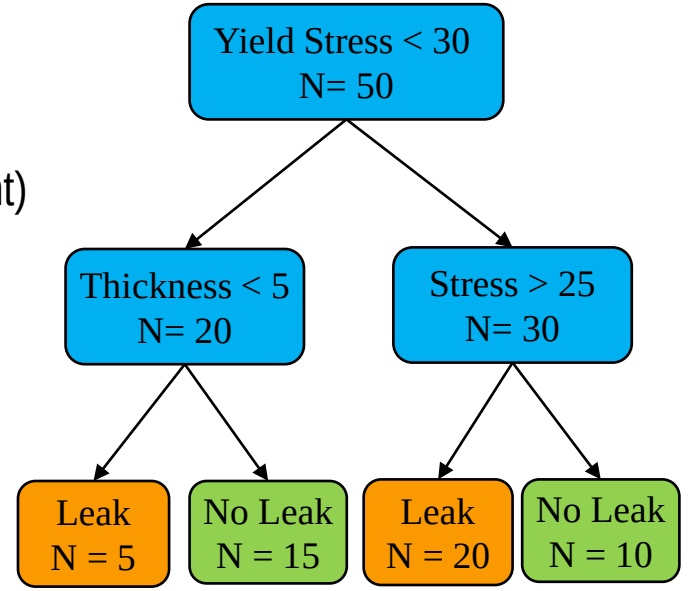
- Decision Tree based machine learning algorithms perform predictions using an *ensemble* of decision trees:
  - Each decision tree is a *weak learner* that does not accurately classify the entire sample population
  - The combined contribution from an ensemble of many weak learners can result in a more accurate prediction
  - Susceptible to overfitting if too many or large decision trees included in ensemble
  - Algorithm parameters control how trees are trained
- Examples:
  - Gradient Boosting Decision Trees
  - Random Forest Decision Trees
  - Adaboost Decision Trees



Machine Learning Models

Gradient Boosting Details

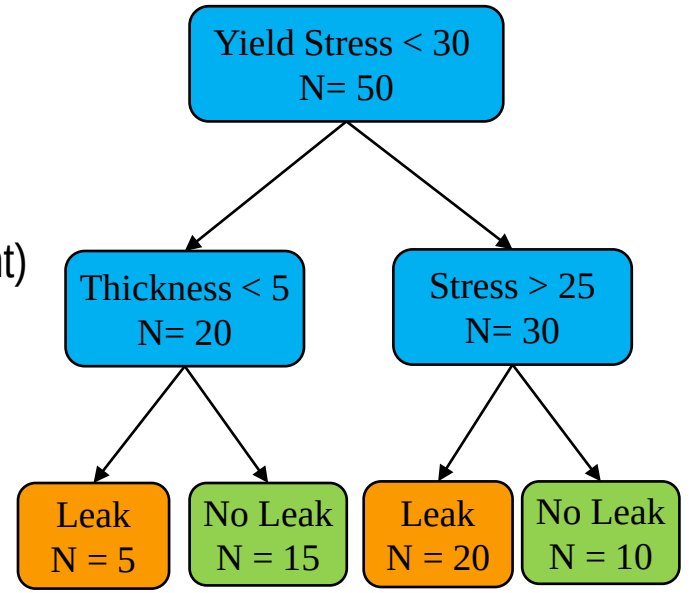
- Trains ensemble of sequentially added decision trees by minimizing a loss function using steepest descent
- Each additional tree intended to reduce error in previous trees
- Number of parameters control how many / how the trees are constructed during the training:
  - Tree specific parameters:
    - Tree depth (number of decision points in tree)
    - Minimum number of samples to split (decision point)
    - Minimum number of leaf samples (tree end point)
    - Max features to consider for a decision point
  - Boosting parameters:
    - Number of trees
    - Learning rate (relative weight of each tree)



Machine Learning Models

Random Forest Details

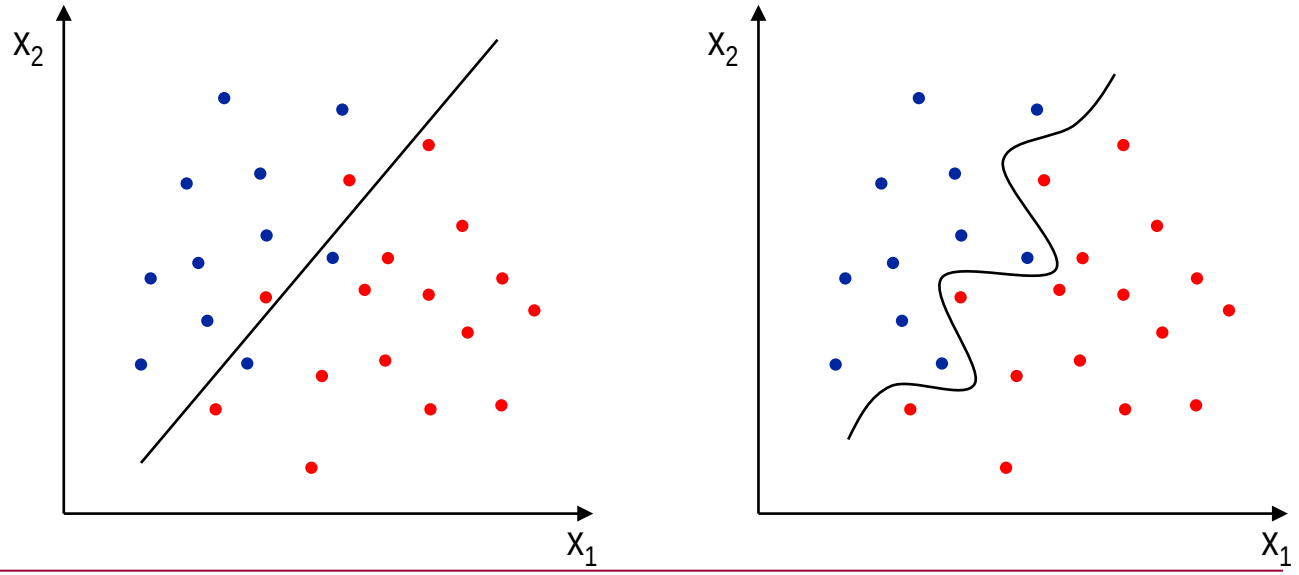
- Trains ensemble of decision trees using bagging (each tree is trained to subsamples of the input dataset with replacement) and each tree only considers a random subset of the input features
- Prediction is based on average or mode of the tree results
- Number of parameters control how many / how the trees are constructed during the training:
  - Tree specific parameters:
    - Tree depth (number of decision points in tree)
    - Minimum number of samples to split (decision point)
    - Minimum number of leaf samples (tree end point)
    - Max features to consider for a decision point
  - Ensemble parameters:
    - Number of trees



Machine Learning Models

Support Vector Machines

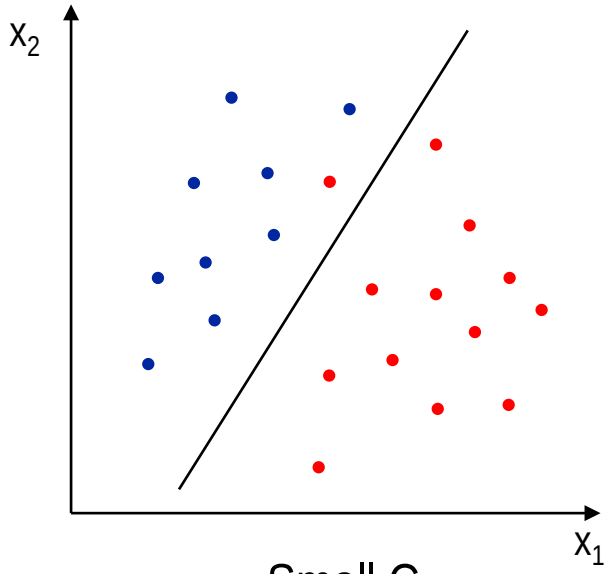
- Support vector machines develop hyperplanes in multi-dimensional space to differentiate training data for classification or regression
  - Hyperplanes can be linear or non-linear (e.g., polynomial)
  - Maximizes the margin (distance / loss function) between the hyperplane and the target classes



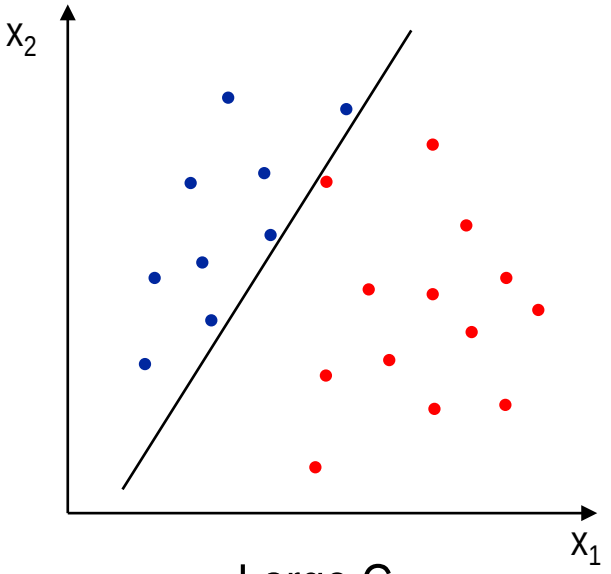
Machine Learning Models

Linear Support Vector Machines

- Linear SVM inputs are normalized to range from 0 - 1
- Linear SVM includes single controlling parameter C
  - Small values maximize margin separating the hyperplane from data
  - Large values minimize misclassification and allow smaller margins



Small C



Large C