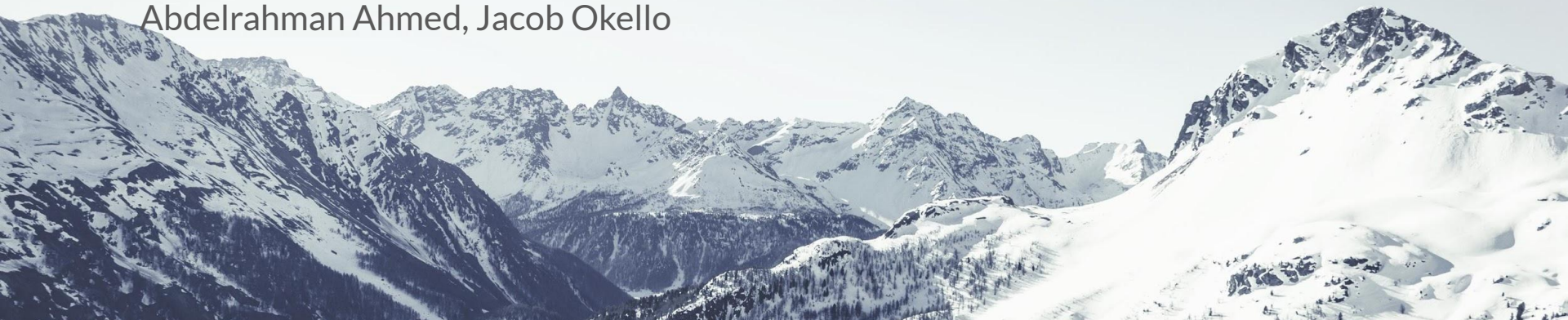




Interpretability Methods to Create White-Box Models.

Abdelrahman Ahmed, Jacob Okello



Outlines



- Interpretability
- White vs Black Box
- Interpretability Methods
- Generalized Linear Model
- GAM
- Implementation of GAM - Example.

Means Interpretability



Is extent to which a human can understand the cause of a decision made by a model.

Which is **Importance** :

- Debugging and improving models
- Trust and transparency
- Regular compliance
- Ethical AI deployment

Black vs White Box



-> Black-box:

- Complex architecture , High accuracy but low understandable.
- Difficult to diagnose errors if happen, Hard to ensure fairness.
- **Black box** is only look at the description of the software which is cannot go inside the main software and maintain it.

-> White-box:

- Understandable by Human which is kind of testing that look inside the software and how code **written & uses Info perform testing.**
- it can't reveal errors due to missing paths. Where missing paths are a part of a software specification that are not implemented.
 - **“Because it is focused on the code and not on the specification.”**



GLM

Generalized Linear Models (GLM)



- Type of mathematical model used for **Regression (predicting continuous outcomes)** and **probabilistic classification (assigning categories with associated probabilities)**.
- **GLM** based on rule-based feature model OR other mean Decision rules which is way for representing data patterns. “**Make model easy to understand**”.
- Rules make model easy to interpret.
- GOOD at build model more accurate and simple enough to understand it.

Introduction



Decision rules are crucial for interpretable models in supervised learning.

- Decision rules have been widely used as building blocks for supervised learning models due to their interpretability and ability to capture nonlinear dependences and interactions.
- we consider models that are linear combinations of decision rules, also referred to as rule ensembles, within the framework of generalized linear models (GLM).

Rule ensembles combine decision rules within the framework of generalized linear models (GLM).

- Rule ensembles integrate multiple decision rules into a cohesive model, leveraging the structure of GLMs to maintain interpretability while improving predictive performance.

This approach trades off prediction accuracy and rule set complexity.

- The goal is to find a balance between making accurate predictions and keeping the model simple enough to be easily interpretable.

PROBLEM



Our problem is formulation trades off predictive accuracy against the complexity of the rule ensemble, measured in terms of the number of rules as well as their lengths.

The main challenge in learning rule-based models is due to the exponential size of the space of rules, corresponding to all possible conjunctions of the input features.

To find best set of rules for model.



- To make balancing accuracy and complexity:
- So we will use the technique of column generation (CG):

is used to intelligently search the space of rules and produce useful ones as needed, as opposed to using a large. trying out every possible combination of rules (which would take forever).

- **CG** is used to avoid the need to generate a huge number of potential rules or trying out rules one by one.
- Solve problem of find the best set of rules using **integer programming (a type of optimization problem)** or a **heuristic (a rule of thumb or shortcut that usually gives a good answer)**.

Methodology



Column Generation (CG):

- CG is a technique for iteratively adding new rules to the model. It helps in efficiently searching a large rule space and selecting the most relevant rules on-the-fly.

Used to search the rule space and generate useful rules on-demand.

- Instead of starting with a predefined set of rules, CG finds and adds rules as needed, improving the model's adaptability.

GLM is re-fit as rules are generated, allowing existing rules to be reweighted or discarded.

- The model is continuously updated with new rules, ensuring that it remains optimal and discards any redundant or less useful rules.

Integer Programming & Heuristics:

- Solving the CG subproblem can be complex, so we use advanced techniques like integer programming or heuristics to find solutions efficiently.

CG subproblem solved using integer programming or heuristic optimizing the same objective.

- These methods help in finding the best possible rules quickly, even within a vast search space.

Results



- They tested their methods using logistic and linear regression (**types of models commonly used in statistics and machine learning**).
- Their methods performed better in terms of balancing accuracy and complexity compared to other rule-based or existing rule ensemble algorithms.
- **In simpler terms, GLMs** found a clever way to build models that are both accurate and easy to understand by using rules, and they tested these methods to show that they work better than other methods out there.

Generalized Linear Rule Models



Problem Context:

- **Objective:** We consider the standard supervised learning problem of predicting a target variable $Y \in \mathcal{Y}$ using input features $X = (X_1, \dots, X_d)$.
- **Data:** training dataset of samples (x_i, y_i) , $x_i = (x_{i1}, \dots, x_{id})$, $i = 1, \dots, n$.
 - The output space \mathcal{Y} may be discrete or continuous.
 - We assume that all features X_j have been binarized to take values in $\{0, 1\}$.

The goal is to predict a target variable Y using input features $X=(X_1,\dots,X_d)$. The relationship between Y and X is modeled using Generalized Linear Models (GLMs) which follow an exponential family distribution.

Exponential Family Distribution



For a **GLM**, the target variable Y conditioned on X follows an exponential family distribution: **Mathematical Formulation:**

Exponential Family: $p_{Y|X}(y|x) = h(y) \exp(\eta y - \Phi(\eta))$,

Variables: Denote by A_k the variable corresponds to conjunction $k \in K$. -

Here, Y conditioned on X follows an exponential family distribution, where η is the **canonical parameter** and $\Phi(\eta)$ is the **log-partition function**.

$h(y)$: The base measure, which is a function of the data y alone.

η : The canonical parameter, which is a function of the data x and model parameters β .

$\Phi(\eta)$: The log-partition function, also known as the normalizer.

Canonical Parameter: $\eta = \sum_{k \in K} \beta_k A_k$ The canonical parameter η is a linear combination of the conjunctions A_k of X ,

with coefficients β_k . Where β_k are the coefficients to be determined, and A_k are conjunctions of the binary features X_j .

log-partition function: The primary role is a function that normalizes the probability distribution, ensuring that the total probability sums to one.

Generalized Linear Rule Models



Model Prediction:

- **Prediction Function:** $y^{\wedge}(X) = E[Y | X] = \Phi'(\eta)$

The prediction function $\mathbf{y}^{\wedge}(\mathbf{X})$ is the conditional mean of \mathbf{Y} , which is the derivative of the log-partition function $\Phi(\eta)$.

Important Goal



We aim to find the best coefficients $\beta=(\beta_1,\beta_2,...,\beta_k)$ to predict Y .

Objective Function



Coefficient Estimation:

- **Objective:** Minimize the negative log-likelihood on training data.

To find the coefficients β , we minimize the negative log-likelihood of the observed data. The log-likelihood for the data can be written as:

$$\ell(\beta) = \sum_{i=1}^n (y_i \eta_i - \Phi(\eta_i))$$

The negative log-likelihood is:

$$-\ell(\beta) = - \sum_{i=1}^n (y_i \eta_i - \Phi(\eta_i))$$

- **Regularization:** Use ℓ_1 regularization to ensure a sparse solution.

Combined Optimization Problem



Combining the negative log-likelihood with the ℓ_1 regularization term, the optimization problem becomes:

where:
$$\min_{\beta} \left[\frac{1}{n} \sum_{i=1}^n \left(\Phi \left(\sum_{k \in K} \beta_k a_{ik} \right) - y_i \sum_{k \in K} \beta_k a_{ik} \right) + \sum_{k \in K} \lambda_k |\beta_k| \right]$$

- n is the number of training samples.
- a_{ik} is the value taken by the conjunction A_k for the i -th instance.
- **First Term:** Measures the fit of the model to the data.
- **Second Term:** Applies the regularization to promote sparsity.

Simplified Interpretation of β



β_k : Coefficients for the rules A_k .

Goal: Find β_k values that balance:

- Accurate predictions (good fit to the data).
- Simplicity (fewer non-zero β_k coefficients).

Sparse Solution: The regularization encourages many β_k to be zero, leading to a model with fewer rules.

Efficient Computation: Techniques like column generation can help manage large sets of possible rules efficiently.

Model Instantiations



- **Generalized Additive Model Using First-Degree Rules.**
- **General Rule Ensemble Using Column Generation.**
- **No restrictions on conjunctions.**
- **CG avoids complete enumeration by intelligent search.**

Generalized Additive Model Using First-Degree Rules



Generalized Additive Model Using First-Degree Rules:

- This version of our model only uses simple, single-condition rules, making it easy to interpret and understand.

Conjunctions correspond to binarized features.

- Each rule is a combination of binarized features, representing specific conditions in the data.

Simplicity and Sparsity:

- Removing linearly dependent rules contributes to sparser and simpler rule ensembles.

Including Non-Binarized Numerical Features:

- This variant is evaluated in the results section,

Free of interaction terms.

- This model does not consider interactions between features, focusing solely on their individual contributions.

General Rule Ensemble Using Column Generation



The use of column generation (CG) to create rule ensembles without restrictions on the conjunctions of features.

General Rule Ensemble Using Column Generation:

- A more complex model that allows for any combination of conditions, generated dynamically through CG.

No restrictions on conjunctions.

- This model can include rules with multiple conditions, capturing more complex patterns in the data.

CG avoids complete enumeration by intelligent search.

- Instead of considering all possible rules, CG intelligently searches and selects the most relevant ones, making the process efficient.

General Rule Ensemble Using Column Generation



Objective:

- Obtain unrestricted rule ensembles by considering all possible conjunctions K of features X .

Challenge:

- K is exponentially large, making it impractical to enumerate variables when the feature dimension d is not small.

Solution: Column Generation (CG):

- **Concept:**
 - a. Developed for large linear programs (LPs), CG leverages the sparsity of optimal LP solutions.
 - b. Originally designed to handle large linear programs (LPs) by managing many columns.
 - c. CG capitalizes on the sparsity of optimal LP solutions.

Procedure:

- **Initial Step:** Solve a restricted problem with a limited set of candidate columns.
- **Dual Solution:** Use the optimal dual solution to identify missing columns that could improve the solution.
- **Marginal Benefit:** Compute the partial derivative (marginal benefit) of adding a missing column.
- **Termination:** If the partial derivative for the best missing column is non-negative, the process stops.

General Rule Ensemble Using Column Generation



Formulating the Column Generation Problem:

- The key is to design a subproblem that searches through missing columns without fully enumerating them.

Optimization Problem Transformation:

- Transform problem by using B^+ and B^- , solve it for a S subset K , and check for optimality.

Optimality Conditions:

- Need to make sure that optimality conditions are met in the restricted problem.

Column Generation Algorithm



Alternates between solving the restricted problem and searching for new columns.

- The algorithm works by repeatedly refining the model and adding new rules (columns) that improve its performance.

Solves the CG subproblem using integer programming or heuristic.

- Each iteration involves solving a subproblem to find the best new rule, using sophisticated optimization techniques.

Terminates with a certificate that the problem is solved to optimality.

- The process continues until it reaches a point where no further improvements can be made, ensuring the model is optimal.

Column Generation Approaches



Objective:

- Solve the column generation subproblem using either integer programming (IP) or a heuristic algorithm.

Implemented in Java using LibLinear and Cplex.

- Our implementation leverages powerful software libraries for linear programming and optimization, ensuring robustness and efficiency.

Implementation:

- **Software Tools:**
 - **LibLinear:** Used for solving regularized logistic regression problems.
 - **Cplex:** Used for solving the integer programming subproblem.

Heuristic algorithm performs limited search of the rule space in order of increasing conjunction degree.

- The heuristic approach starts with simpler rules and gradually considers more complex ones, making the search process manageable and effective.

Classification Experiments



Evaluated performance-complexity trade-offs.

- We analyzed how our models balance prediction accuracy with the complexity of the rule set.

Reported accuracy and Brier score.

- Key performance metrics include accuracy (correct predictions) and the Brier score (measuring probabilistic predictions' accuracy).

Measured rule ensemble complexity in terms of the number and length of rules.

- We assessed the complexity by counting the number of rules and their lengths, aiming for simplicity without sacrificing performance.



Results Summary

Proposed methods achieve better accuracy-complexity trade-offs.

- Our models provide a good balance, maintaining high accuracy while keeping the rule set manageable.

Competitive with less interpretable benchmark models.

- Despite being more interpretable, our models perform competitively with less interpretable but traditionally more powerful models.

Simpler models often available at a relatively small performance cost.

- We found that simpler, more interpretable models can be almost as effective as more complex ones, making them practical for real-world applications.



Relationships Between GLMs, Decision Trees, and Linear Regression

Generalized Linear Models (GLMs)

- **Definition:** GLMs are a broad class of models that generalize **linear regression** to allow for **response variables** that have **error distribution models other than a normal distribution**. They consist of three components:
 1. **Random Component:** Specifies the distribution of the response variable (e.g., normal, binomial, Poisson).
 2. **Systematic Component:** A linear combination of the predictor variables (similar to linear regression).
 3. **Link Function:** A function that relates the mean of the response variable to the linear predictors.



Examples of GLMs:

- **Linear Regression:** A GLM with a normal distribution and identity link function.
- **Logistic Regression:** A GLM with a binomial distribution and logit link function.
- **Poisson Regression:** A GLM with a Poisson distribution and log link function.



Linear Regression

- **Relationship to GLMs:**

- **Special Case of GLMs:** Linear regression is a specific type of GLM where the response variable is **normally distributed**, and the **link function** is the identity function. This means the predicted values are directly equal to the linear predictors.

- **Formula:** $y = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n + \epsilon$

$$= w_0 + \sum_{i=1}^n w_i x_i$$

-
- where ϵ is the normally distributed error term and n the number of features.



Linear Regression

Objective of Linear Regression Learning Algorithm

- **Goal:** Determine weights to accurately predict target variable for all patients in the training set.

Techniques:

1. Gradient Descent
2. Closed-form Solution (e.g., Newton equation)

Gradient Descent:

- **Advantages:** Scales well with a large number of features and training examples.
- **Process:** Updates weights to minimize squared error between predicted and actual target variables.

- **Outcome:** Guarantees finding optimal weights by minimizing squared error (least squares method).

Implementation:

- **Tool:** Scikit-Learn in Python
- **Dataset:** Standardized diabetes dataset (zero mean and unit variance)
- **Note:** Feature standardization is crucial for many machine learning models, including linear and logistic regression, and neural networks.



Decision Trees

Definition: Decision trees are a non-parametric supervised learning method used for classification and regression. They recursively split the data into subsets based on the value of input features, forming a tree-like structure.

Relationship to GLMs:

- **Different Approach:** Decision trees do not assume a specific distribution for the response variable or a specific form for the relationship between predictors and the response variable. Instead, they use a hierarchical, rule-based approach to model the data.
- **Model Complexity:** Unlike GLMs, decision trees can capture non-linear relationships and interactions between variables without requiring a pre-specified form.



Generalized Additive Models (GAMs)

Even though the models covered above perform reasonably, there not that good. By interpreting the models, we have also uncovered some shortcomings.

The linear regression model does not seem to handle features that are highly correlated with each other. The decision tree model performs worse than linear regression, and it seems to have overfit on the training data.



Generalized Additive Models (GAMs)

● **Definition:** GAMs extend GLMs by allowing non-linear functions of the predictor variables while maintaining additivity. This means each predictor can be modeled with a non-linear function, but the overall model remains additive.

Formula: where $f(i)$ are smooth functions of the predictors.

$$y = w_0 + f_1(x_1) + f_2(x_2) + \dots + f_n(x_n)$$



INTELLIGIBLE MODELS

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_1^N$ denote a training dataset of size N , where $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$ is a feature vector with p features and y_i is the target (response). We use x_j to denote the j th variable in the feature space.

Generalized additive models (GAMs) are the gold standard for intelligibility when low-dimensional terms are considered [4, 5, 6]. Standard GAMs have the form

$$g(E[y]) = \beta_0 + \sum f_j(x_j), \quad (1)$$



INTELLIGIBLE MODELS- GAMS

where g is the link function and for each term f_j , $E[f_j] = 0$. Generalized linear models (GLMs), such as logistic regression, are a special form of GAMS where each f_j is restricted to be linear. Since the contribution of a single feature to the final prediction can be easily understood by examining f_j , such models are considered intelligible.

To improve accuracy, pairwise interactions can be added to standard GAMS, leading to a model called GA²Ms [6]:

$$g(E[y]) = \beta_0 + \sum_i f_j(x_j) + \sum_{i \neq j} f_{ij}(x_i, x_j). \quad (2)$$



Comparing GLMs, Linear Regression, and Decision Trees

GLMs vs. Linear Regression:

- Linear regression is a type of GLM.
- Both use a linear combination of predictors, but GLMs can handle different types of response variable distributions and link functions.

GLMs vs. Decision Trees:

- **Assumptions:** GLMs rely on specific assumptions about the distribution of the response variable and the form of the relationship between predictors and response. Decision trees do not.
- **Flexibility:** Decision trees can model complex, non-linear relationships without pre-specifying the form of the relationship, unlike GLMs.
- **Interpretability:** GLMs provide a clear interpretation of coefficients, whereas decision trees provide interpretability through decision rules.



Summary

Linear Regression: A special case of GLMs where the response is normally distributed, and the link function is identity.

Decision Trees: A non-parametric method different from GLMs that models data using a hierarchical, rule-based approach, capturing non-linear relationships and interactions.

GLMs and GAMs: GLMs can be extended to GAMs to include non-linear functions of predictors while maintaining additivity.



Intelligible Models for Healthcare:

Predicting Pneumonia Risk and Hospital 30-day Readmission

Key Points:

- Tradeoff between accuracy and intelligibility in machine learning models.
- High-performance generalized additive models with pairwise interactions (GA2Ms) used for intelligibility and state-of-the-art accuracy.
- Two case studies: Pneumonia risk prediction and 30-day hospital readmission prediction.



Motivation for Intelligible Models in Healthcare

Key Points:

- Historical challenge: High accuracy models (e.g., neural nets) vs. intelligible models (e.g., logistic regression).
- Example: Neural nets outperformed traditional methods but were too risky for clinical use due to lack of intelligibility.
- Importance of understanding, validating, and trusting models in mission-critical healthcare applications.



Case Study 1: Pneumonia Risk Prediction

Title: Case Study: Predicting Pneumonia Risk

Key Points:

- Dataset: 14,199 pneumonia patients, 46 features.
- Goal: Predict probability of death (POD) - to inform hospitalization decisions.
- GA²M models achieved highest (Area under the curve -is the [definite integral](#) of the [concentration](#) of a [drug](#) in [blood plasma](#) as a function of time) AUC (0.857) while remaining intelligible.
- Discovered and corrected misleading patterns (e.g., asthma indicating lower risk due to ICU admissions).



GA2M Model Insights for Pneumonia

Title: Insights from GA2M Model on Pneumonia

Key Points:

- Important features: Age, asthma, BUN levels, cancer, chronic lung disease, heart rate, respiration rate.
- Asthma pattern corrected due to intelligibility.
- Discovered new issues (e.g., chronic lung disease, history of chest pain).
- Model is modular and editable by experts.



Case Study 2: 30-Day Hospital Readmission

Title: Case Study: Predicting 30-Day Hospital Readmission

Key Points:

- Dataset: 195,901 patients in training, 100,823 in testing, 3,956 features.
- Goal: Predict which patients are likely to be readmitted within 30 days.
- GA2M models used to provide intelligible predictions for individual patients.



GA2M Model Insights for Readmission

Title: Insights from GA2M Model on Readmission

Key Points:

- Examined predictions for three patients with varying risks.
- Top terms contributing to risk: Number of hospital visits, specific medications, underlying conditions.
- Intelligible predictions allow understanding of individual patient risk factors.



Discussion - How To Interpret Risk Scores

How To Interpret Risk Scores

- Each term in the model returns a risk score (log odds) added to the patient's total predicted risk.
- Risk scores above zero increase risk; below zero decrease risk.
- Terms added to a baseline risk, converted to probability.
- Baseline rates for pneumonia and 30-day readmission near 0.1 (TotalRiskScore = -2.197).
- TotalRiskScore > -2.2: Higher than average risk.
- TotalRiskScore < -2.2: Lower than average risk.
- Example: TotalRiskScore = 0, probability = 0.5.



Discussion - How To Interpret Risk Scores

A patient with TotalRiskScore = 0 (including the |

offset) has quite high risk:


$$p = 1/(1+\exp(-1*\text{TotalRiskScore}))$$

$$= 1/(1 + \exp(0))$$

$$= 0.5.$$

RiskScore	Probability	RiskScore	Probability
-5.0	0.0067	+5.0	0.9933
-4.0	0.0180	+4.0	0.9820
-3.0	0.0474	+3.0	0.9526
-2.0	0.1192	+2.0	0.8808
-1.0	0.2689	+1.0	0.7311
0.0	0.5000		

Table 3: Risk scores (log odds) and the corresponding probabilities.



Discussion - Modularity

Modularity

- Average risk score for each feature/pair of features set to zero by subtracting mean score.
- A bias term added so average predicted probability matches observed baseline rate.
- Models are identifiable and modular.
- Each term can be removed without biasing predictions.
- Only remaining term if all are removed: Bias term predicting the observed baseline rate.
- Adding terms increases model discriminativeness without altering the prior.
- Important for interpretability: Negative scores decrease risk, positive scores increase risk compared to baseline.



Discussion - Sorting Terms by Importance

Sorting Terms by Importance

- For fewer than 50 terms, show in familiar logical groupings.
- For many terms, sort by importance (drop in AUC, term skill, maximum contribution).
- Sorting helps identify key patient characteristics quickly.
- Example: 30-day readmission model with over 4000 terms, relevant terms for each patient are often less than 100.
- Importance for single patient: Sort terms by impact on risk, aiding expert understanding.



Discussion - Feature Shaping vs. Expert Discretization

Feature Shaping vs. Expert Discretization

- Cost-Effective HealthCare - CEHC pneumonia study used logistic regression with expert-defined boolean variables.
- Example intervals for age: 18-39, 40-54, etc.
- GA2M model with continuous features had higher accuracy (AUC drop of 0.01 with discretized features).
- GA2M's accuracy due to better shaping of continuous features.

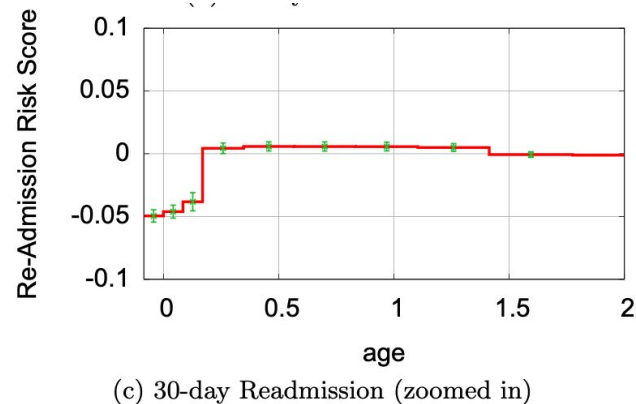
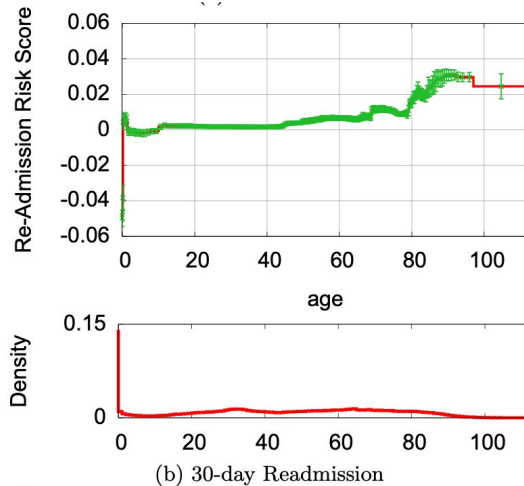
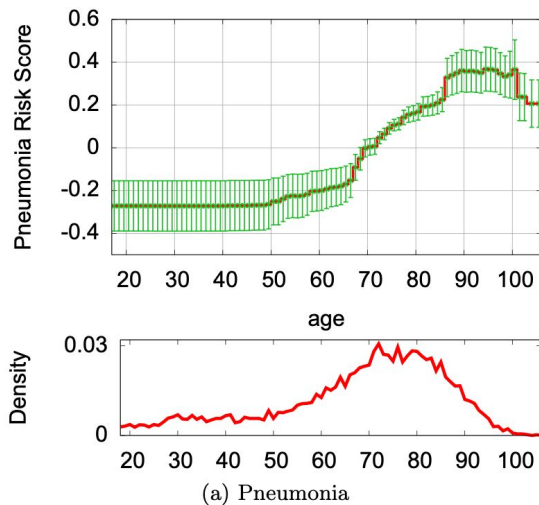


Discussion - Deep Dive: Risk as a Function of Age

Risk as a Function of Age

- Age critical in pneumonia models, less so in 30-day readmission.
- Pneumonia model: Risk low and constant from 18-50, rises from 50-66, then quickly from 66-90, levels off above 90.
- Detailed risk profiles, jumps at specific ages (e.g., 67, 86).
- Readmission model: Little effect on risk between 2-50, rises slowly from 50-80, slight increase above

Figure 3: Risk as a function of Age for the Pneumonia and 30-day Readmission problems.





Discussion - Deep Dive: Risk as a Function of Age (continued)

Detailed Analysis of Age in Models

- **Pneumonia Model (Figure 3(a)):**
 - Risk profile shows:
 - Constant low risk from 18-50.
 - Slow rise from 50-66.
 - Rapid increase from 66-90.
 - Leveling off at very high risk above 90.
 - Notable jumps in risk at ages 67 and 86.
 - Possible reasons for jumps discussed: retirement age effects, healthcare access, or model artifact.



30-Day Readmission Model (Figure 3(b)):

- Age less influential compared to pneumonia.
- Little impact on readmission risk between 2-50.
- Slow rise from 50-80, slight increase above 80.
- Larger positive impact on risk seen at age 90 and above.
- Infants age 0-2 months show decreased risk of readmission.



Discussion - Shaping with Splines

Shaping with Splines

- **Generalized Additive Models (GAMs)** often use splines.
- Splines allow control over regularization, error bars, and capture basic trends.
- Comparison with GA2M models:
 - GA2M captures more detail and structure.
 - Higher accuracy in risk prediction compared to spline models.
 - Example terms: Age, pH, temperature (Figure 4).



Discussion - Shaping with Splines

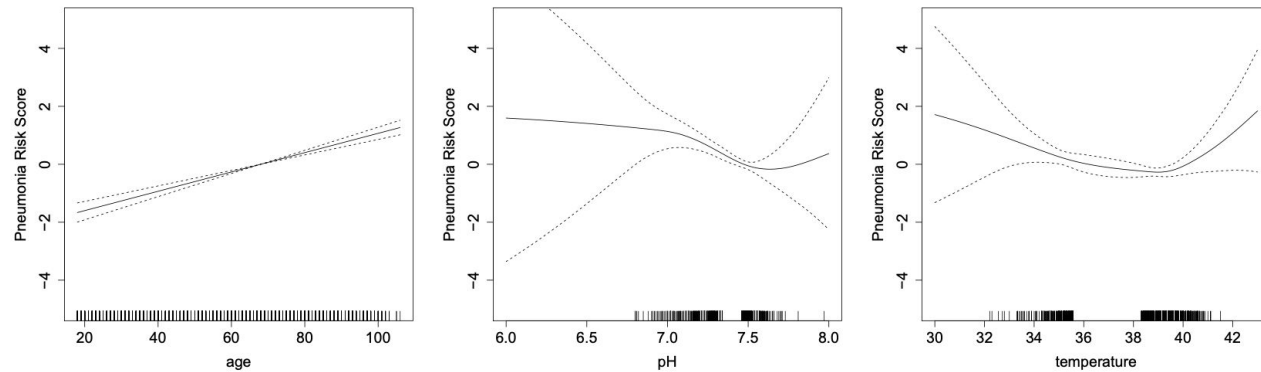


Figure 4: Selected splines in pneumonia dataset.



Discussion - Shaping with Splines

Comparison of Spline GAM and GA²M on Pneumonia Data

- **Spline GAM Model:**
- **Terms:** Age, pH, Temperature
- **Trends Captured:**
- Age: Risk increases with age.
- pH: Risk is lowest around 7.6.
- Temperature: Fever risk rises above 40°C.
- **Limitations:**
- Misses nuanced details.
- May not properly model temperature in the normal range (36°C-38°C).
- Accuracy closer to logistic regression.
- **GA²M Model:**
- **Benefits:** More nuanced model for age.
- Captures additional detail.
- Increased accuracy reflects genuine structure in data.
- **Conclusion:** GA²M captures more detailed relationships, resulting in higher accuracy compared to spline GAM



Discussion - Correlation Does Not Imply Causation

Correlation Does Not Imply Causation

- Models are based on correlation, not causation.
- Changes in features can alter model terms.
- Interpretation challenges:
 - Overfitting.
 - Correlation with other variables.
 - Interactions with unmeasured variables.



Discussion - Correlation Does Not Imply Causation

- Need for cautious interpretation:
 - Detailed shape plots raise questions.
 - Clear distinction between correlation and causation needed.
 - Future directions: automate analysis, add causal inference.



Limitations of Generalized Additive Models (GAMs)

Overfitting: GAMs can overfit, but this can be managed with regularization.

- **Sensitivity to Feature Range:**
 - GAMs lose predictive power with feature values outside the training set range.
 - Outlier values particularly affect performance.
- **Predictive Power:**
 - For mission-critical tasks, GAMs might not be powerful enough.
 - More robust black-box models may be necessary for such tasks.



Conclusions

GA2Ms in Healthcare:

- Achieve state-of-the-art accuracy.
- Maintain intelligibility crucial for deployment.

Pneumonia Case Study:

- Reveals patterns missed by other models.
- Editable for reduced deployment risk.



Conclusions

30-Day Readmission Task:

- Manageable model despite complexity (4000+ terms).
- Focus on key terms for patient-specific predictions.

Future Directions:

- Automate analysis to validate model insights.
- Integrate causal analysis for deeper understanding.



Conclusions

Acknowledgments:

- Contributors from University of Pittsburgh and Columbia University Medical Center.



Questions and Discussion

- Open floor for questions.
- Discuss implications for healthcare applications.
- Explore challenges and opportunities in model interpretability and deployment.

These slides summarize the discussion on interpreting risk scores, modularity in model construction, sorting terms by importance, the impact of age on pneumonia and readmission risks, modeling with splines, and the distinction between correlation and causation. They conclude with insights into GA2Ms' potential in healthcare and future research directions



Thank you.

