Toward Interactive Visualizations for Explaining Machine Learning Models

Ashley Ramsey
University of Nebraska at Omaha
Computer Science

Akshay Kale
University of Nebraska at Omaha
Computer Science

Yonas Kassa
University of Nebraska at Omaha
Computer Science

Robin Gandhi
University of Nebraska at Omaha
Cybersecurity

Brian Ricks
University of Nebraska at Omaha
Computer Science
bricks@unomaha.edu

ABSTRACT

Researchers and end users generally demand more trust and transparency from Machine learning (ML) models due to the complexity of their learned rule spaces. The field of eXplainable Artificial Intelligence (XAI) seeks to rectify this problem by developing methods of explaining ML models and the attributes used in making inferences. In the area of structural health monitoring of bridges, machine learning can offer insight into the relation between a bridge's conditions and its environment over time. In this paper, we describe three visualization techniques that explain decision tree (DT) ML models that identify which features of a bridge make it more likely to receive repairs. Each of these visualizations enable interpretation, exploration, and clarification of complex DT models. We outline the development of these visualizations, along with their validity by experts in AI and in bridge design and engineering. This work has inherent benefits in the field of XAI as a direction for future research and as a tool for interactive visual explanation of ML models.

Keywords

Explainable AI, Data Visualization, Bridge Health, Decision Trees

INTRODUCTION

Visualizing the rule space learned by training a complex machine learning model in an understandable and comprehensive way is a major challenge in the broader field of AI. This area of research is part of eXplainable Artificial Intelligence (XAI) and it is critical to building trust in AI systems and the rules they use to generate inferences when subject to new situations. Explainability is particularly critical where Machine learning (ML) is used in emergency and crisis management scenarios where quick and dependable decisions are of utmost importance (Linardos et al. 2022). One reason that XAI is important is that people are generally unwilling to place their trust in AI systems that have uninterpretable models (Arrieta et al. 2020). It is easy to see why mistrust is the baseline; when an AI is a completely black-box model, the end user and even the AI specialists who made the AI may be unable to tell exactly how or why the AI produced its classifications or conclusions. For example, the European Union has passed a law that gives their citizens a right to receive an explanation of an algorithm's decision if it significantly affects them (Goodman and Flaxman 2016). This is especially important in that it is global in nature; any company processing EU citizens' data is expected to comply (Goodman and Flaxman 2016).

However, XAI should not be viewed as a barrier to the implementation and advancement of AI. Rather, it is a practice to ensure that the model learned matches with previously established evidence-based practices. For

example, Murdoch et al. describe a real study in which a ML model was developed to determine mortality risk for hospital patients with pneumonia (Murdoch et al. 2019). Counter intuitively, the ML model learned that patients with asthma have a lower mortality risk relative to patients without asthma; what the model didn't learn was that this reduced mortality risk is due to the intensive care that patients with asthma and pneumonia receive. Without XAI practices, this AI would have suggested these patients receive less intensive care, thus putting them at higher risk. XAI and the understanding it creates can be used to directly improve the quality of ML models and ensure it is functioning as intended.

Broadly speaking, there are two categories of methods for providing explanations of an AI: transparent models and post-hoc explanations (Arrieta et al. 2020). According to Barredo, a model is transparent when the programmer makes changes or creates the AI to be self explanatory (Arrieta et al. 2020). An example of a transparent model could be a simple decision tree, in which a human could look at the rules in a decision tree in their entirety and accurately trace the model's decisions, with a clear understanding of the attributes the model emphasized in making a classification or prediction (Arrieta et al. 2020). Post-hoc analysis, on the other hand, is necessary when a model is too complex to understand, as is often the case with neural networks. A post-hoc explanation entails generating a textual, visual, or other type of explanation to simplify or describe a complex ML model. In the context of this paper, we outline the development of visual explanations with focus on model transparency and post-hoc explanations of a machine learning model for infrastructure health monitoring.

RELATED WORK

The body of literature surrounding XAI and interpretable machine learning is expansive. The following research efforts are relevant to the research of this project.

Explainable AI

Arrieta et al. have surveyed the XAI field (Arrieta et al. 2020). It includes an overview of popular machine learning methods and how XAI can be used with them. The authors say explaining an AI to an audience is not as productive as explaining the AI's decisions using strategies suited for a end users, who often lacks a prior understanding of AI. The authors also emphasize that XAI as a practice is only a part of creating responsible AI, which needs to uphold principles of fairness, privacy, accountability, ethics, transparency, and safety. In discussing developments and applications of ML in disaster management, Arrieta et al. (Linardos et al. 2022) emphasized that developed models should be explainable and understandable by domain experts and decision makers as such operations affect human life.

Lipton similarly discusses the importance of model interpretability by describing the most crucial outcomes of interpretability as trust, causality, transferability, informativeness, and fair and ethical decision making (Lipton 2018). These qualities are presented as goals that drive the need for interpretability in artificial intelligence. Lipton then establishes that interpretability is achieved by developing a transparent model, meaning understandable algorithmically or as a whole, or applying post-hoc methods of interpreting data. Given the complexity of some machine learning models, decomposability is put forth as a less strict alternative way of bringing transparency to a model. Decomposability describes a quality of a model in which it can be broken down or simplified into parts that are transparent or intuitive in their own capacity. In the realm of post-hoc interpretability, Lipton identifies four primary types: text explanations, visualizations, local explanations, and explanations by example. Visualizations, which are the primary concern of this paper, can vary greatly depending on the model and its outputs. Local explanations, such as the visualization (detailed later) attempt to explain smaller sections of a model in order to capture the behavior of the full model.

Hohman et al. introduce the idea that the focus in XAI research settings is usually on developing a machine learning model and interpreting the output (Hohman et al. 2020). The authors emphasize that the dataset is also very important to producing a quality and useful model. In some situations the dataset will not change. In these cases, the authors suggest data iteration, in which the dataset or the way the data is manipulated is adjusted alongside the model. This article is relevant to the project as something to keep in mind during the research. The visualizations of this project have informed changes to the ML models, and vice versa.

Explainability in the Social Sciences

A significant portion of the literature surrounding XAI in the social sciences calls for the involvement of the social sciences in development of new XAI techniques. In this field, Miller outlines four major findings (Miller 2019). First, explanations are contrastive. When asking questions, people will usually ask why something happened instead

of something else. In practice, this could mean an AI could provide explanations justifying why it makes one choice over possibly the second most likely choice. Second, explanations are selected with bias. Meaning, humans don't often expect a complete profile of how a decision came to be. Rather, people choose at most a few causes as the explanation for a decision. Third, using probability or statistical relationships to form the basis of an explanation is less useful than referring to the potential causes. Lastly, explanations are social. This implies an interaction that takes into account the background of the explainer during the explanation.

Wang et al. build upon these core ideas of social explanation in AI (Wang et al. 2019). They propose a framework for creating XAI, utilizing the knowledge in psychology and philosophy as a basis. The authors describe two types of human reasoning - System 1 and System 2 reasoning. System 1 reasoning is quick and based in intuition. It is defined by cognitive shortcuts and biases. System 2 reasoning is much slower and more rational, which necessitates more effort and thought. Using the proposed framework, the researchers developed an XAI visualization for medical diagnosis. The study found, in accordance with Miller's findings, that users often sought contrastive explanations. They also found that, to support System 2 reasoning, it is better to show the reasoning behind a prediction before the prediction itself. The authors find that it is easier for users to build the appropriate amount of trust in AI if they have support for manual decision making alongside the AI's decisions. This allows them to come to their own conclusions and compare with the AI. Lastly, the authors recommend integrating multiple kinds of explanations into XAI methods if possible.

In the recent work of Liao et al., the authors apply explanatory knowledge from the social sciences in an attempt to find gaps in XAI methods that are currently available (Liao et al. 2020). In this study, user needs for explainability are represented as common user questions and used as a study probe. The researchers interviewed 20 UX and design practitioners regarding users' need to understand AI and the explanatory techniques in use currently. They found that a special emphasis was placed on the need for XAI to be socially interactive. Another finding is that, contrary to the common focus of XAI development, users tended to focus more on which features were used and how as opposed to the technical details of the model. The authors suggest that, moving forward, XAI should address user needs with a question based framework. This would allow users to engage with the AI in a socially interactive manner.

Mittelstadt et al. contribute to this discussion, noting that recent work in XAI has focused on model simplification (Mittelstadt et al. 2019). However, they argue that XAI research would be better focused on interactive methods of interpretability that present the ability for informed dialogue between relevant parties. They claim that offering a simplified or local model as an approximation of a given machine learning model can be misleading. As an alternative, the authors suggest contrastive explanations, emphasizing that these explanations are responsive to human desire for contrastive, selective, and social explanations.

APPROACH

In the area of XAI, this paper focuses on the explanation of a collection of decision trees models (Kale 2022). These machine learning models process data from the National Bridge Inventory (NBI) of the United States. NBI contains a yearly government survey of bridge health, containing just under a hundred features of a bridge. Examples of features include scour critical rating and year built (Bridges and Structures 2021). NBI also includes evaluation for condition ratings of bridges as a way to assess the state of the bridge's deck, superstructure, and substructure over time. By tracking instances of improvements in the three listed condition ratings, we can determine when a repair was made. This is critically important information for the decision trees of this project, as it provides the decision trees with labeled examples of bridges that will and will not receive repairs.

Therefore, the complex decision trees of this project are the machine learning models with the purpose of predicting whether a bridge is likely to need repairs in the immediate future. After processing the NBI data and constructing a set of rules that determine the health of each bridge, the decision tree for a given state (out of Illinois, Indiana, Kansas, Minnesota, Missouri, Nebraska, Ohio, and Wisconsin), a given repair type (out of Substructure, Deck, and Superstructure), and a given classification criterion (out of entropy and gini) is constructed with a unique structure and evaluation of which features are more important for a classification. The result is a collection of 48 different decision trees, averaging 442 rules per tree, with an upper bound of 1,171 rules, that our work seeks to explain through visualization (Table 1).

Providing an interactive visual display of this complex machine learning model is crucial for human understanding of the data it produces and the AI itself. A tool of this kind should do more than present data; it should allow bridge engineers to interact with the data in a deliberate, meaningful way. Presenting the data in a visually concise format further can allow users to make discoveries about the dataset that would be otherwise difficult or impossible to find. Because of the scale of the data handled in this project (decision trees can be as large as a thousand rules in size),

State	Average Number of Rules
IL	227.3
IN	621.5
KS	679.5
MN	92.8
MO	264.2
NE	467.7
OH	1037.5
WI	145.0
All	441.9

Table 1. This table shows the average number of rules for a given state across all trees using that state. For example, Illinois shows the average number of rules for all six Illinois trees.

Figure 1. This flowchart shows the expected flow of user interaction with these visualizations. In each visualization, the user is shown a smaller window of data, such that an observation from the heatmap can be followed up with more specific information from the tree visualization and a visual showing the interaction between two features.

specialized visualizations needed to be created to reduce complexity while retaining all of the information present in the decision trees. This research proposes a three visualization suite for understanding and interacting with the bridge health decision trees. A brief summary of each is included in this section, and each is described in greater detail in Work Conducted section.

The first visualization is a post-hoc visualization called the Bridge Repair Features Heatmap, which can display information from any number of trees at once (See Figure 1, left). This visualization aims to sort different features by importance in the machine learning models' decision making processes. Bridge engineers can use this visualization to compare, across states and repairs, what features are most relevant.

A second visualization is a model-based visualization that takes a given decision tree and allows bridge engineers to explore it in its entirety (See Figure 1, center). Since these decision trees are too large to visualize in their entirety, several navigation features are included to truncate the decision tree, reduce the amount of information on screen, or clarify what the user is seeing. The primary purpose of this visualization is to give bridge engineers a highly customizable overview of the model, from which they can generate more insights or questions that make use of the next visualization.

The third visualization, called the Two Feature Matrix, seeks to give the user the ability to visualize the interaction between two different features for a given decision tree (see Figure 1, right). In a decision tree, every feature is dependent on the rules of the features that come before it. The Two Feature Matrix simplifies this by traversing the tree using every possible value pair for two given features, assuming that other rules regarding other features are both true and false, and collecting every leaf node (or classification) of the tree. In this fashion, we can find the number of positive and negative classifications for the value pair for the chosen features, and plot these in a heatmap graph.

WORK CONDUCTED

The three visualizations described in the previous section, as well as their supporting modules, are described in detail in this section. Before continuing on to the visualizations, we will briefly discuss the libraries used to construct these visuals.

The primary library used in this work is d3, originally described in "D3 Data Driven Documents" by Michael Bostok, Vadim Ogievetsky, and Jeffery Heer (Bostock et al. 2011). D3 is a visualization library that offers programmers fine control over their visualizations. In the context of this project, D3 facilitates the construction of the visualizations by simplifying the process of interacting with the Scalable Vector Graphic (SVG) tags that display the visualizations.

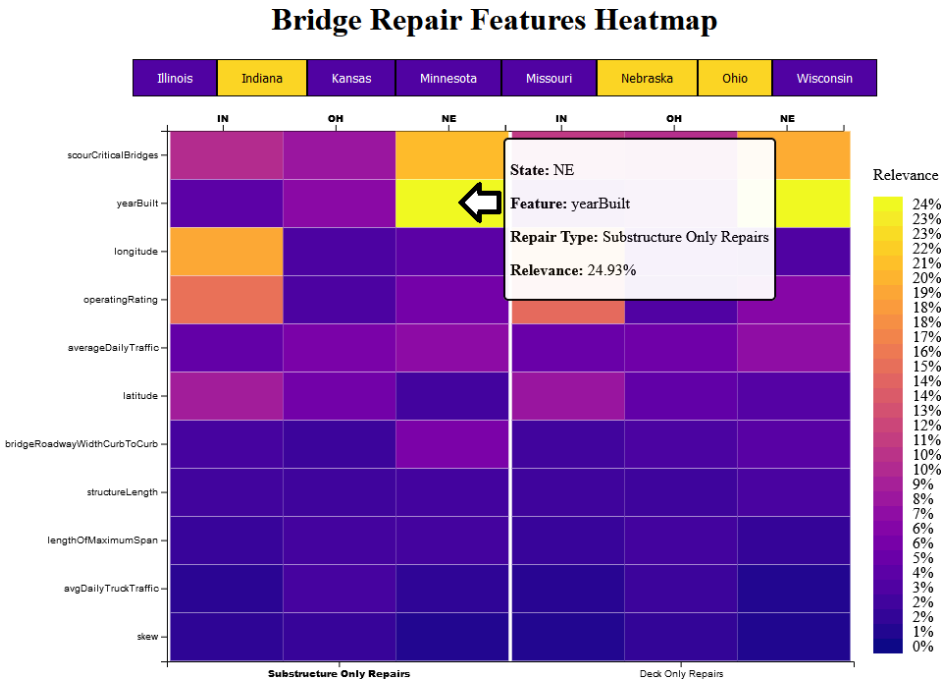

Figure 2. This image shows the feature heatmap with Indiana, Ohio, and Nebraska selected, all repair types shown, hiding superstructure repairs, and sorting by substructure repairs. Additionally, a tooltip is displayed. If the user clicks on a data point, they will be taken to the corresponding decision tree, as in Figure 7.

Feature Heatmap Visualization

The first visualization of this work is a heatmap visualization. Generally, heatmaps have two axes. In this two axis grid system, a data point is plotted as a colored rectangle. The color of the rectangle is determined by each axis value at the intersection, which in turn can be determined by a data set or a function. The first visualization of this work extends this definition by adding a third dimension. Our feature heatmap, as it is called, makes use of three axes: an axis for states, an axis for repair types, and an axis for bridge features, such as in Figure 2.

The feature heatmap is a powerful, post-hoc visualization tool for exploring the decision trees of this work. The goal of this visualization is to give bridge engineers an overview of the different decision trees and allow for easy comparison of the decision trees for two or more state-repair-classification triads. Additionally, an AI specialist can make use of a tool like this and adapt the ideas and code of this work to their own machine learning models to aid in generating user friendly explanations.

An example of the feature heatmap can be seen in Figure 2. This visualization takes an input from the decision trees in the form of two csv files, one corresponding to decision trees using entropy as a classification criterion and the other to trees using gini index as a classification criterion. Entropy and gini based classification critera are mathematical equations that are used in the construction of a decision tree to find a good rule to use to split the population into two subgroups that are easier to classify with subsequent rules. The gini index accomplishes this by minimizing impurities in divided samples each time a new rule is needed to obtain accurately classified samples (Suryakanthi 2020). Entropy, on the other hand, is a measure of the disorder within a set of samples, which is also minimized in the creation of the decision tree (Suryakanthi 2020). Each tree in this work has a version that was constructed using entropy and another constructed using the gini index.

Relevance is a value for the features of a decision tree, determined by the "Mean Decrease Impurity importance" described by Louppe et al. (Louppe et al. 2013). This is a measure of the average effectiveness in splitting a population into two more easily classified groups, for the rules involving a specific feature. For example, if yearBuilt is often successful in splitting a group of bridges into one category of bridges that is more likely to need repairs and another category of bridges that is less likely to need repairs, then yearBuilt will have a high relevance value for this specific decision tree. This process is handled by the scikit-learn library in the construction of the decision trees. Since not every decision tree has every feature, in cases where the tree lacks a feature, that feature's relevance value is set to zero as a pre-processing step. These values are then normalized for each tree, such that they add up to one.

Having read in the data, the outer portion of the heatmap (such as axes and labels) is created. Since the visualization is produced very quickly, changes in options such as states simply rebuilds the visualization. When the sorting

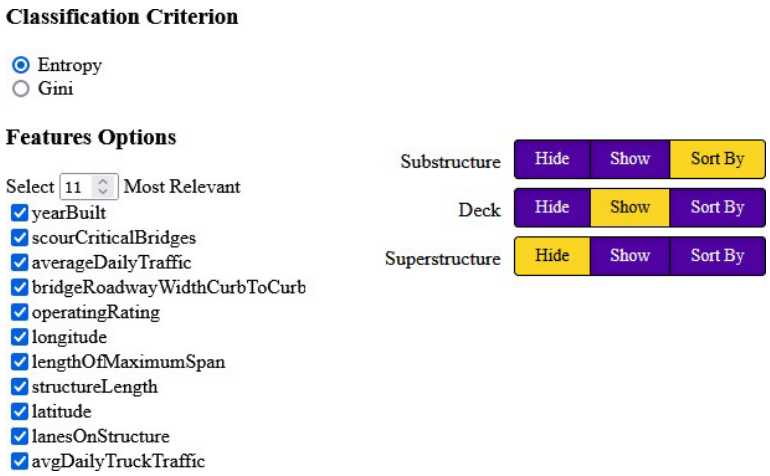

Figure 3. These options, which exist left of and underneath the feature heatmap respectively, offer the user control over classification criterion, number of features, specific features, and repair types they would like for the visualization.

repair type is changed using the three way radio in Figure 3, the features are sorted by average relevance value for a repair type across selected states, as seen in Figure 2.

With the skeleton of the feature heatmap created, the inner data points are filled in. Each data point has a tooltip (see Figure 2), and the user can click a data point to see the associated decision tree visualization for that data column, filtered to only show rules that use the selected feature. For example, if the user clicked the NE-yearBuilt-Substructure data point while the feature heatmap was set to the Entropy classification criterion, as Figure 2 shows, they would be taken to the decision tree in Figure 7.

The first options panel, in Figure 3, shows the options for choosing classification criterion as well as features. The classification criterion is a radio button, because the relevance values for trees of entropy vs gini are kept separate to avoid visualizing a fourth factor dividing the data. Also in this options panel are feature options. These are presented so that the user can select or deselect specific features, or adjust the tree such that it presents more or less relevant features. This allows the user multiple avenues of visualizing only the features that they want to see at a given time.

The next options panel, found in Figure 3, has the options for repair types. This is a table component containing three sets of radio buttons. These radio buttons are linked such that only one repair type can be set as the sorting repair type. When the user selects a different repair type for sorting, the previously selected repair type changes to "Show" to allow for the user to sort by their desired repair type. When only a single state is selected, sorting by a repair type simply sorts features of all columns according to the relevance of that feature in the selected "Sort By" column. In cases where multiple states are selected, as in Figure 4, the average feature relevance across states for the sorting repair type decides the order of the features.

In the final options panel, above the heatmap in Figure 2, the user can toggle on or off any number of states. At least one state must be selected, and if the user tries to deselect all, the last remaining state will not be deselected. This options panel could be extended to include a larger number of states, or similar categories of data, although at a certain number of categories, the developer risks both an overwhelming amount of options and a possibly unreadable heatmap.

Tree Visualization

The second visualization in our XAI suite is the decision tree visualization. This visualization takes the form of an interactive binary tree visualization. In general, a binary tree is a type of graph made up of nodes and edges. Each node represents a data point and each edge connects two data points. In a binary tree, each node has at most three connections - one to a "parent" node in the previous layer (except for the root, which has no parent) and up to two "child" nodes in the following layer. In the context of this work, we are visualizing decision trees, where nodes represent conditions (called rules), such as "year built >1993" and each edge represents the previous condition being true or false. By combining several consecutive rules, a decision tree can classify samples of a population into groups.

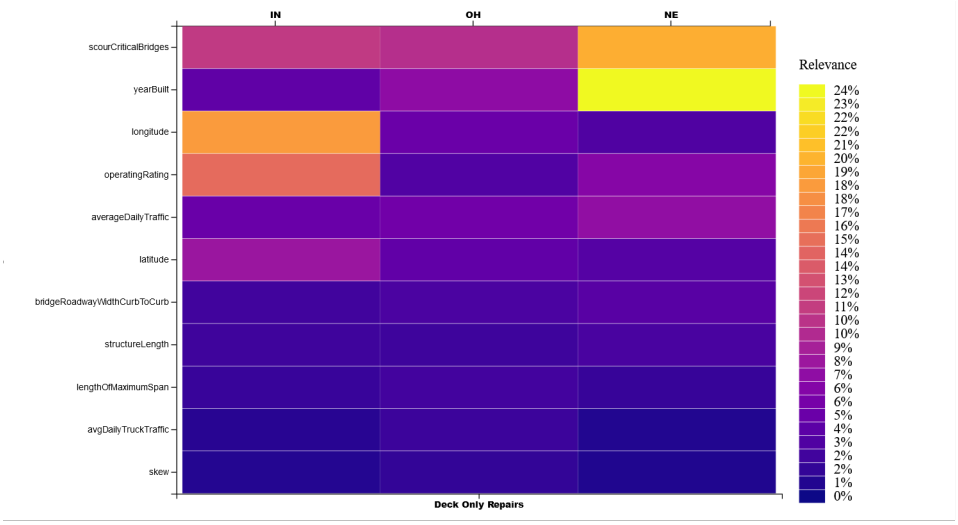

Figure 4. This figure shows the feature heatmap's ability to adjust to user input, as the user has selected to sort by Deck Only Repairs while hiding the other two repair types. This allows them to focus on data specific to a given repair or two while possibly bringing in data from multiple states.

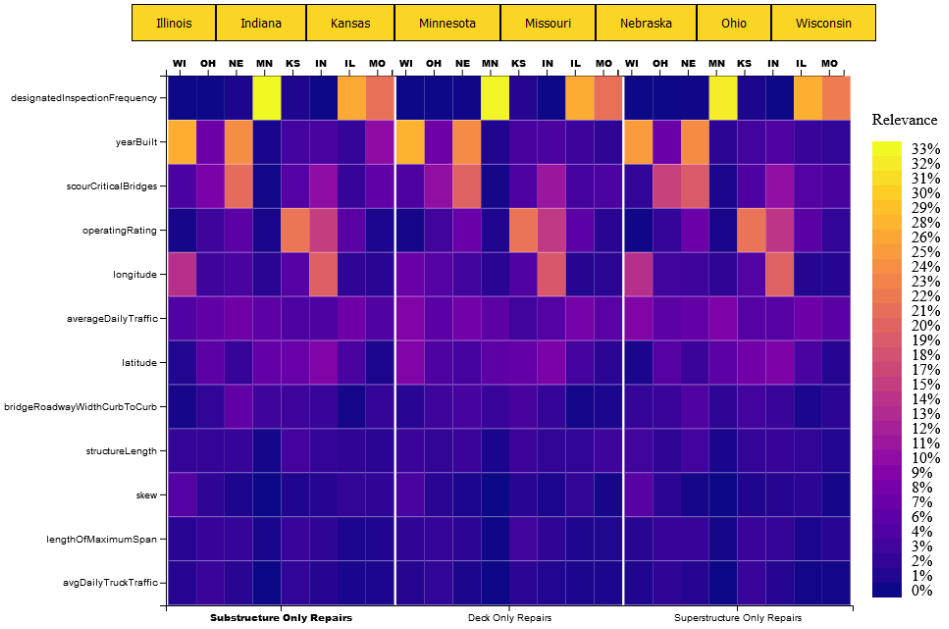

Figure 5. In this figure, every state is selected to demonstrate the flexibility of this visualization.

The decision tree visualization is a model-based visualization with the purpose of bringing clarity and ease of exploration to large decision trees for understanding bridge repair trends. This allows bridge engineers to establish a clear understanding of what exactly a decision tree is and how it operates. This is accomplished primarily by providing the user several options for managing complexity. We will go over these options in greater detail in this section.

When initializing the visualization, if the user arrived at the visualization from the feature heatmap, the identifying information about the tree is taken from the URL. Additionally, the settings from the feature heatmap are stored in the window's local storage for retrieval, should the user use the back button in the options panel. The data is then retrieved from the corresponding tree (or if no tree is provided, the default NE-Substructure-Entropy tree) and the features in the options panel are initialized. The feature checkboxes are colored according to the feature relevance for the corresponding column in the feature heatmap to maintain continuity.

From here, the tree data is used to create the tree visualization. This is a process that can be broken into creating the tree and updating the tree. The basis for the tree is created by appending an SVG with the appropriate dimensions to the body of the page and preparing the tree as a d3 hierarchy. When updating the tree, two containers are created as children of the svg. In the first container, every connection between nodes is created as two connected "paths" (in the svg parlance, not the graph theoretic meaning), shown in Figure 10. The next container is assigned a child node for every rule in the tree, shown in Figure 11.

To give a sense of scale, the leftmost tree in Figure 6 shows an example decision tree of about average size for this work. This image may look daunting at first, but the user has many tools at their disposal to reduce the decision tree down to manageable and comprehensible sizes with only relevant information. For instance, the sizes of the nodes and the edges are scaled according to the number of classifications, or leaf nodes, that exist among the descendants of the given node. For edges, the size is tied to the node that is further from the root of the tree. This feature allows users to see at a glance which portions of the tree have the most classifications.

The majority of the tools we provide the user focus on reducing the size of the decision tree. For example, Figure 6 shows a decision tree where the minimum classifications value in the options panel is set to 11. This means that, at every node in the decision tree, if that node has fewer than 11 descendant leaf nodes, all descendent nodes of the given node are hidden (see Figure 9 for an example of node hiding). This allows for a much simpler view of the tree, wherein the user can distinctly see which rules are splitting the bridge population into two balanced or unbalanced subgroups. Figure 6 also shows another feature for managing the decision tree's size, where the user is given control over the maximum depth of the tree. This option hides all subtrees that exceed a certain depth, which reduces the number of nodes on screen, leaving the user with a much simpler tree to learn from.

Another option available to the user is the ability to select which features they want to see represented in the tree. This option can be seen at work in Figures 7, 10, and 14. In each of these examples, one or more features have been filtered from the decision tree visualization in order to create a simpler model for understanding decision making process.

In addition to path tooltip, previously described and found in Figure 10, this visualization includes a node tooltip, found in Figure 11. The node tooltip is split into two key parts. The first is a table of every rule that has subdivided the population up to this node. This table is simplified, such that a collection of rules involving the same feature only takes up one line of the table. This gives the user a quick summary of the kinds of bridges that get classified in this subtree. Additionally, the tooltip includes two numbers corresponding to the number of descendant leaf nodes classifying a bridge as needing a repair or not needing a repair. This is also reflected in the coloration of the node in question. In the example image, since two thirds of descendant leaf nodes recommend no repairs, two thirds of the node in the tooltip is colored blue, and the other third is colored red.

Two Feature Matrix

The third and final visualization is called the "two feature matrix." This model-based visualization is a heatmap that plots the interaction between two different features in a range of values for a given decision tree as a colored block. Since this is a more experimental visualization, we will describe the goals, methods, and use cases of collecting the necessary data in greater detail.

Even given the many tools available to the user for reducing the complexity of the decision tree visualization, the visualization itself can still be daunting to try to understand. The two feature matrix is an attempt to further simplify the complexity of the tree visualization and to extract more specific details about the interaction between two features. Consider a decision tree, where only the rules that involve Feature A and Feature B are visible to the user. This decision tree might look something like the decision tree in Figure 14 or in Figure 12.

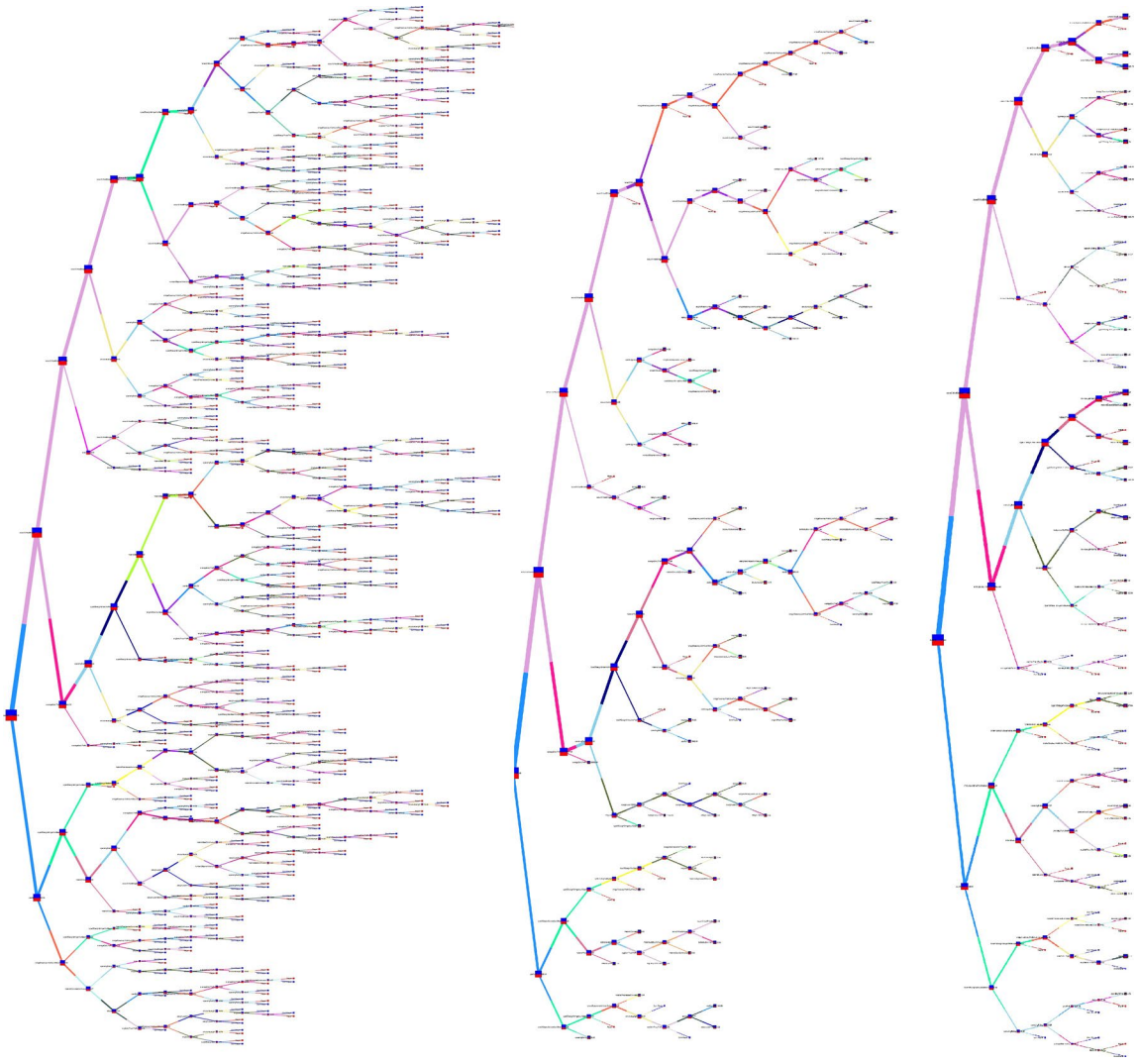

Figure 6. This figure shows the complete decision tree for Nebraska, Substructure repairs, and Entropy classification criterion (left), the tree reduced using the minimum classifications option (middle), and the tree reduced using the maximum depth option (right).

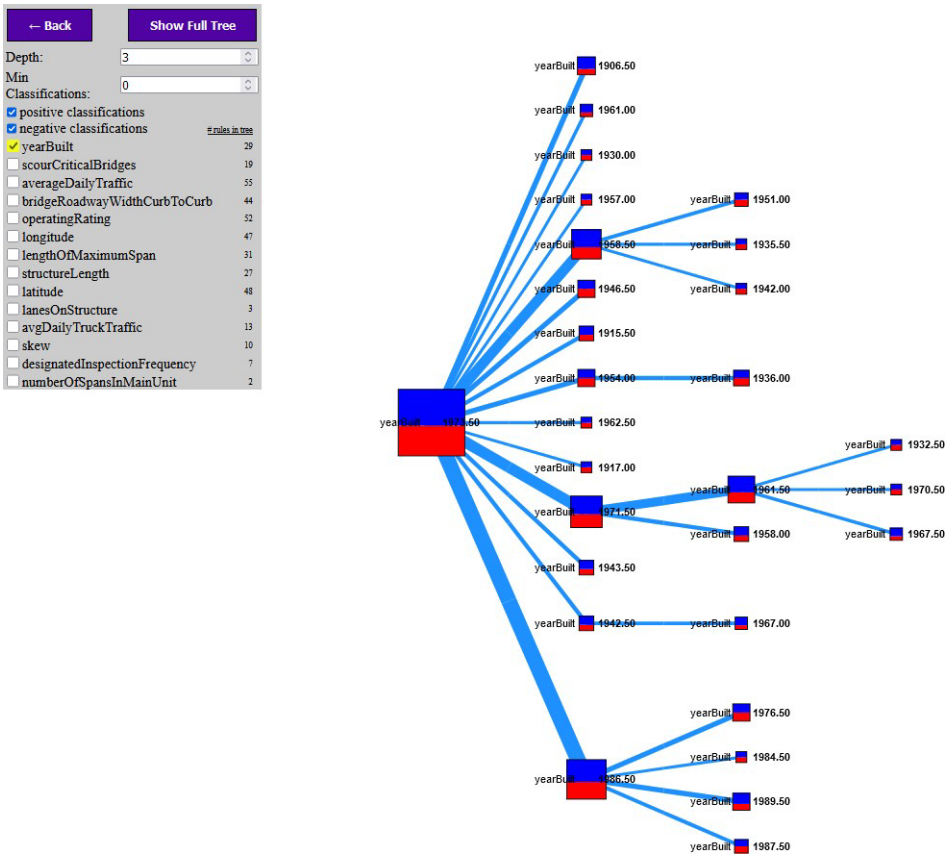

Figure 7. This is the decision tree visualization for NE-Substructure-Entropy with only "yearBuilt" selected. This is what the user would see if they clicked on the datapoint for NE-Substructure-Entropy and yearBuilt in the feature heatmap, also shown in Figure 2.

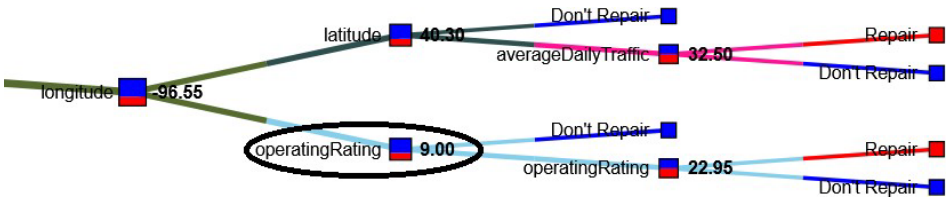

Figure 8. This image shows a subtree of the larger decision tree visualization. When the user clicks the circled node, this subtree looks like the subtree in Figure 9.

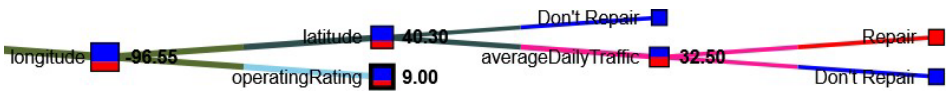

Figure 9. This image shows a subtree of the larger decision tree visualization with a collapsed "operatingRating" node. This is denoted by a thick, black border around the node. The user may click the collapsed node to return this subtree to the subtree in Figure 8.

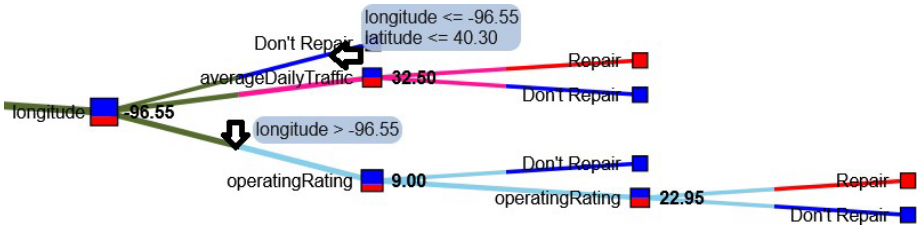

Figure 10. This image shows two possible scenarios for edge tooltips in the visualization. The lower tooltip shows the decision made in the parent node to lead to the child node. The upper tooltip shows the same, as well as any decisions that were removed from the tree by toggling features off.

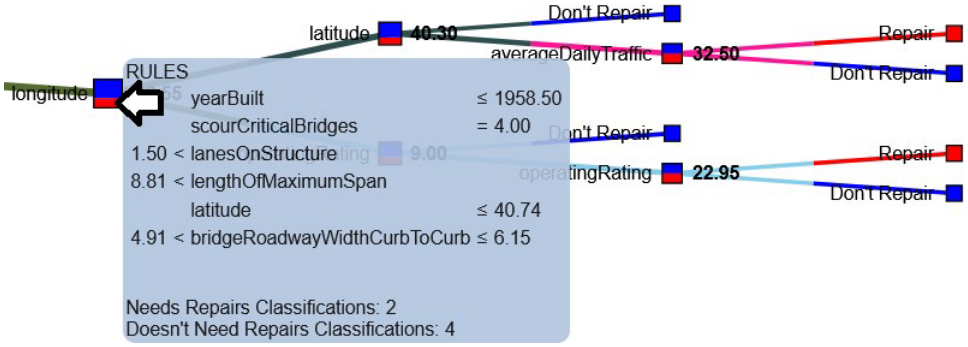

Figure 11. This is the tooltip that appears when hovering over a node. Note how rules has been simplified.

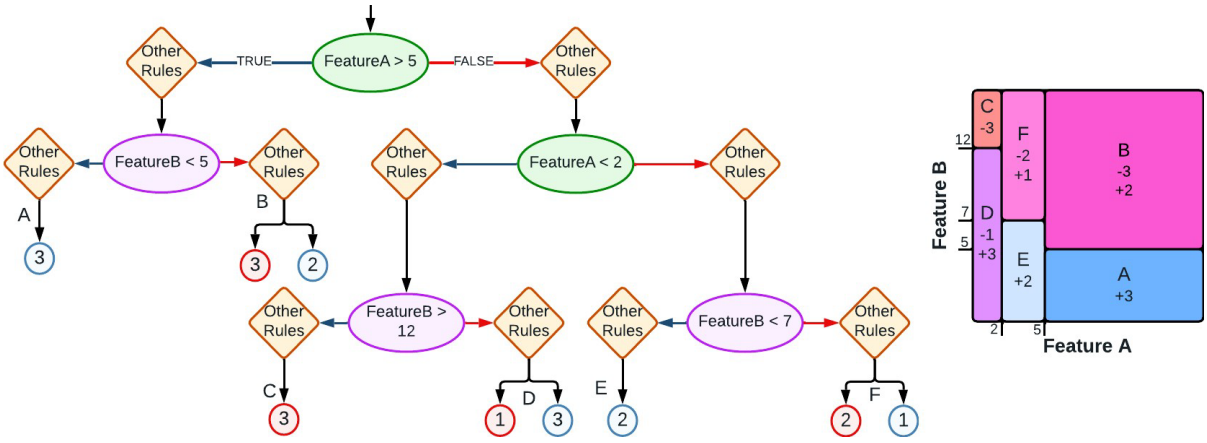

Figure 12. The data gathering technique for the two feature matrix finds every classification node matching a combination of values for Feature A and Feature B, and then plots these values for every possible combination. The areas are sectioned off from one another based on the specific classification nodes between two areas.

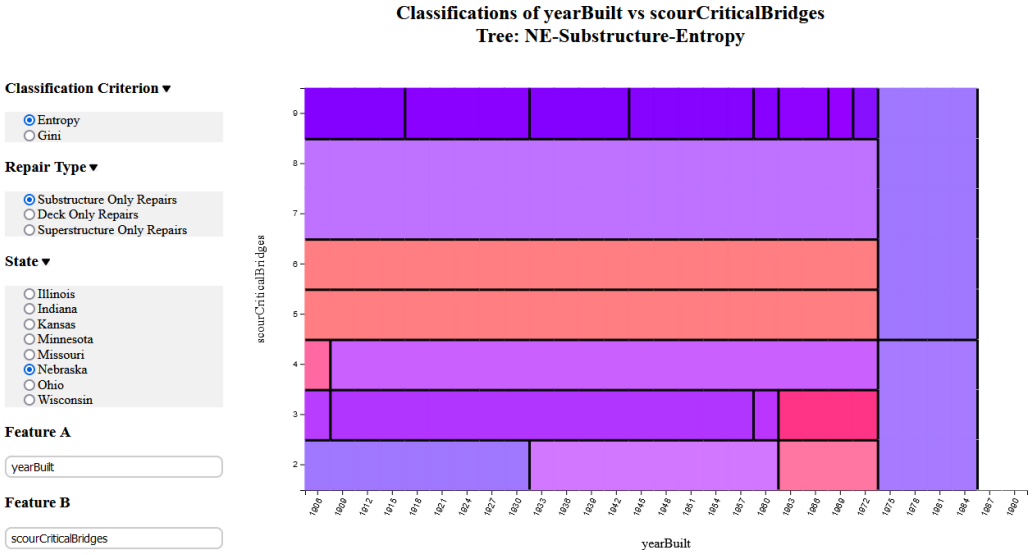

Figure 13. This is the two feature matrix for the NE-Substructure-Entropy decision tree with yearBuilt and scourCriticalBridges selected. The user can select any tree and any two features for the visualization.

In Figure 12, what the two feature matrix shows is, for every possible value pair of yearBuilt and scourCriticalBridges, of the deepest nodes corresponding to one of these features, which are accessible by this value pair. This process assumes that every rule that does not use one of these two features will be both true and false, meaning both paths are taken. For example, in the left image of Figure 12, if we test every value from 1-16 for each feature (i.e. 256 tests of the tree, since we are using every combination of 16 values for A and 16 values for B), the resulting visualization in the two feature matrix would look like the right image of Figure 12.

As a more concrete, smaller example using the same two images, consider the value pair FeatureA = 3, FeatureB = 1. Following the decision tree in Figure 12, the first condition would be false, leading us to the right subtree. When we encounter other rules, we take both the true and false path, which could lead to us exploring multiple subtrees. In this case, we only have one subtree. At the second rule, we take the right path since our condition is false. And finally, at the third rule, we take the left path since the condition is true. This results in us arriving at the "E" group of classification nodes. This corresponds to the E section of the two feature matrix in Figure 12. We will discuss the coloring of these regions in greater detail later in this section.

In more complex decision trees, we often encounter situations where taking both paths at a rule that doesn't match our features will lead us to multiple groups of classification nodes. In this case, we simply sum the matching classifications. Figure 14 shows a real decision tree, where we see that this situation occurs quite often. In this case, there are 29 nodes at the deepest visible level. The corresponding two feature matrix, in Figure 13, shows 23 unique regions (including the white region where no classification nodes were reachable). These numbers don't match up, because each region corresponds to some combination of node groups out of the 29 groups in the decision tree. Within a bounding region, each data point shares the exact same classification nodes as others in the region.

To create the visualization, we first fetch the required data for the selected decision tree and features. This includes the data ranges (i.e. 0-100, counting by 2) and a list of all the classifications in the chosen tree. With this data, we can find all classification nodes that match data values for each value pair in the data range for each feature, as described in the previous example. Then, two arrays are created: one that includes data points that have a neighbor above that doesn't match their classification nodes and another for data points with non-matching neighbors to the right. Using the data generated, the visualization is constructed, placing boundaries between data points specified by the two arrays. The finished visualization can be seen in Figures 15 and 16.

The coloring of the visualization is done using Hue Saturation Value (HSV) color channels. Using this method of coloring, we can provide a scale of color from blue to red, corresponding to no repair needed and repair needed, and we can provide a scale from saturated to unsaturated, corresponding to having many classification nodes and having few classification nodes. Using this coloring method, we can see for which values of the two selected features are resulting in strong or weak recommendations for or against repair.

An example of this color scheme at work is apparent in Figure 16. In this example, for the NE-Deck-Entropy tree, we can see a clear demarcation in hue where at a longitude of -95 and before 1967, the decision tree includes more

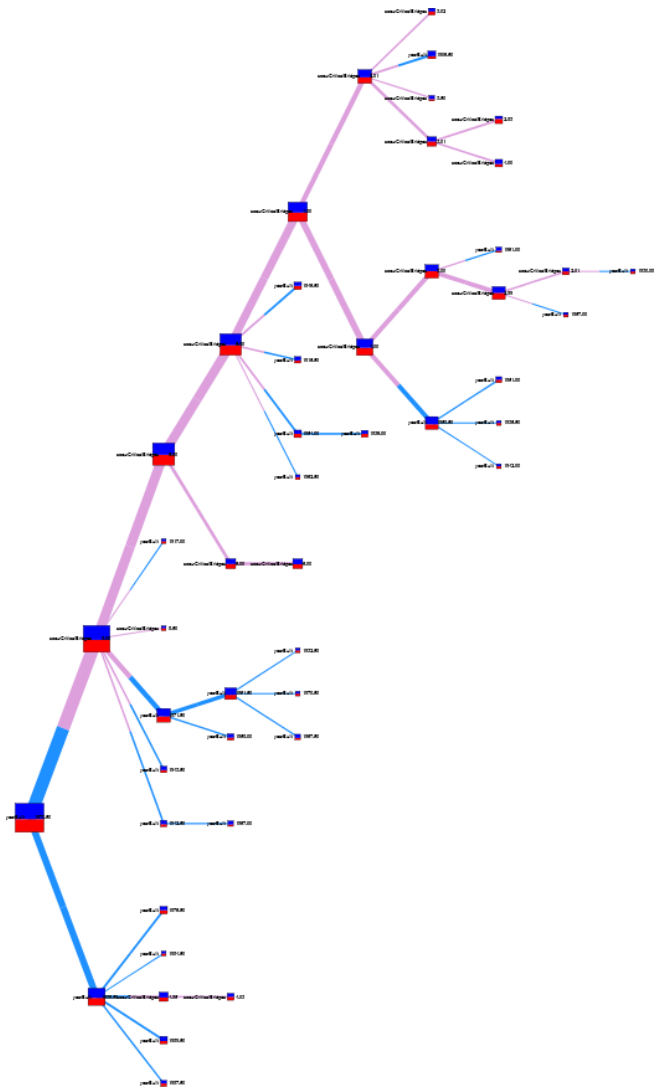

Figure 14. This is an example of a decision tree, filtered to only show yearBuilt features (in blue) and scourCriticalBridges (in pink).

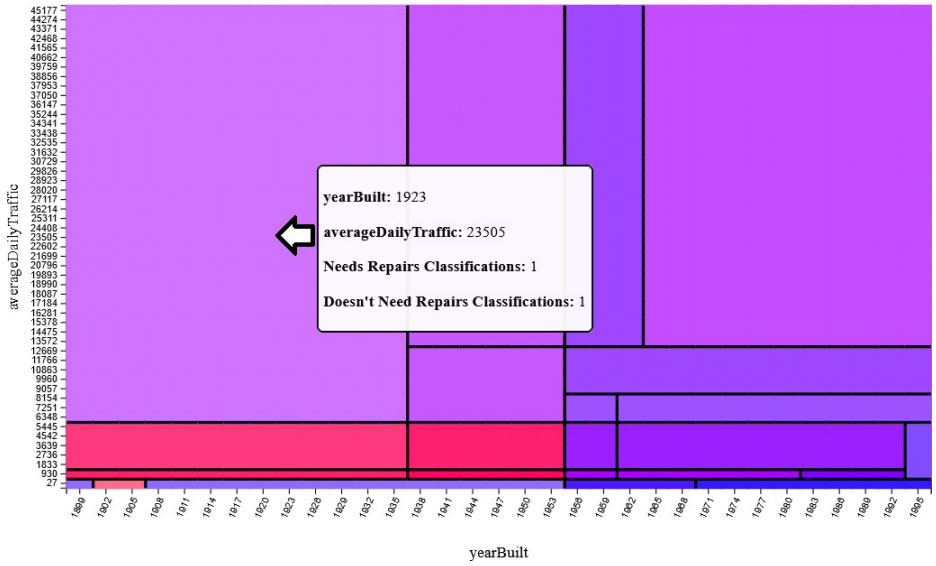

Figure 15. This image shows an example of the information provided in a tooltip. The tooltip is specific to a given data point, as opposed to the entire area, so the values of the features may differ within an area, though the classification numbers will stay the same.

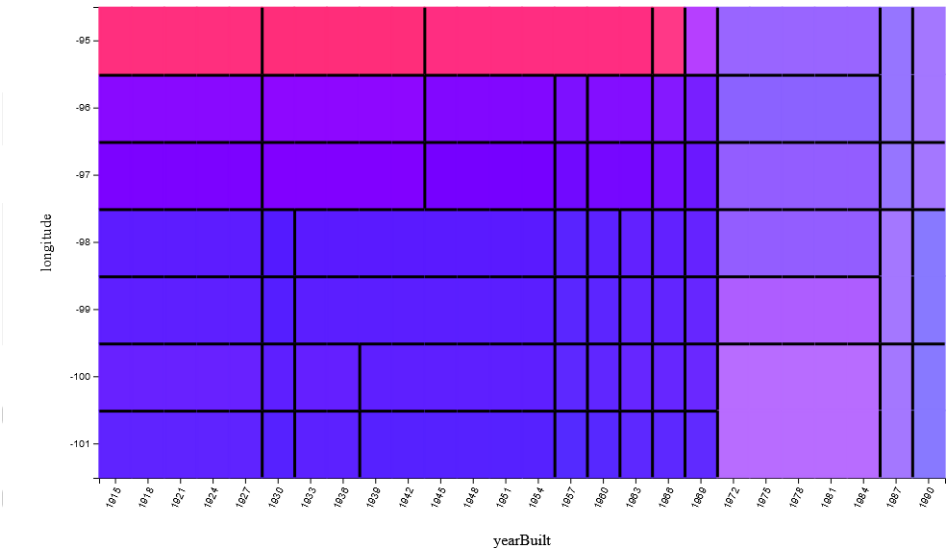

Figure 16. This is another example of the two feature matrix, with clear boundaries in both number of classifications (based on saturation) and type of classification (based on hue).

classifying nodes suggesting repairs for bridges. Additionally, it is clear to see from the saturation of the colors that the decision tree generally includes more classification nodes for bridges built before 1970 suggesting that for these older bridges more rules involving more features may be necessary to accurately assess whether repairs are needed.

EVALUATING RESULTS WITH EXPERT FEEDBACK

Results in this paper have been presented at a variety of conferences and meetings and received a great deal of feedback from bridge engineers. In this section, we will discuss some of these in greater detail.

This research includes the input of a bridge engineer at a state department of transportation. This engineer has been involved in multiple meetings for this project, where we present the visualizations and receive their feedback, perspectives, and interpretation of the data. They have been invaluable in providing direction in this project. For instance, in the early development of the feature heatmap visualization, they brought insightful information to the table, such as questions about the meaning of feature relevance, which was at the time a value that was not normalized, not displayed in the tooltip, and not tracked in the legend. This inquiry prompted the development of those useful features. Additionally, the bridge engineer's exploration of the decision tree visualization helped us recognize the level of complexity present in decision trees, prompting development of many of the size-reducing methods. This also informed the creation of the two feature matrix visualization, as a different method of managing the level of information presented to the user.

Though this work has seen multiple presentations, there are two that we believe provide face validity for this work. The first was at the Midwest Bridge Preservation Partnership (MWBPP). It consists of a group of representatives from highway agencies, transport agencies, relevant industries, suppliers, consultants, and academia. During this meeting, we presented the feature heatmap and the decision tree visualization. The attendees expressed deep interest in this project and posed thought provoking questions and directions for future research. We gave another presentation to a large group of industry experts, many of whom had years of experience in the area of bridge maintenance. A common sentiment among these industry experts is the benefit of extending these visualizations into similar fields. One such area is dam health, which has similarly detailed records and could benefit from the predictive machine learning models and model explanations of this work. We acknowledge that more structured user studies are required to further improve the visualizations. The detailed descriptions along with the code base made available through this work will enable such research.

CONCLUSION AND FUTURE WORK

Interpretability in artificial intelligence is very important since it is understood to be a prerequisite for users' trust in machine learning models. In explainable AI, there are two primary types of explanation - model-based explanation and post-hoc explanation. This paper provides three interactive visualization techniques for decision tree explanations using both methods of explanation. Additionally, these visualizations are based in the psychology behind explanations so that they provide the tools for the user to generate their own effective explanations.

For researchers looking to continue work in a similar vein, we would recommend developing different visual tools for explaining machine learning outputs. As outlined in the related works, the field of XAI is moving toward explanations that are rooted in the psychology of explanation.

Within this project, we have two recommendations for future work. The first is generalization of the visualizations in this paper. Generalizing methods in this paper for use in other domains, such as dam health, has been suggested on multiple occasions. The work needed to complete this would vary based on the visualization, but this would typically require adapting hard-coded feature and state data to accept a more general form of input data. The second recommendation is to adjust the size of the tree visualization's nodes and paths and color of the two feature matrix visualization. These aspects of the visualizations are currently tied to the number of classifying nodes in the tree rather than the actual bridges classified. By basing tree node and path size as well as two feature matrix coloring on data from bridges classified by the models, this would ground the visualizations in more tangible data.

In this paper, we demonstrated the potential of interactive visualizations as important explanation techniques for constructing practical explainable AI (XAI) systems. Our results have been demonstrated to expert stakeholders and the feedback suggests a promising potential for our approach. We hope that the findings presented in this paper can provide insights and motivations for future research on interactive tools for explaining various machine learning models.

ACKNOWLEDGEMENT

This work is partially supported by contracts W912HZ21C0060 and W912HZ23C0005, US Army Engineering Research and Development Center (ERDC), and Award Number 1762034 from the National Science Foundation.

REFERENCES

Arrieta, A. B. et al. (2020). "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Information fusion* 58, pp. 82–115.

Bostock, M., Ogievetsky, V., and Heer, J. (2011). "D³ Data-Driven Documents". In: *IEEE Transactions on Visualization and Computer Graphics* 17.12, pp. 2301–2309.

Bridges, O. of and Structures (2021). *National Bridge Inventory*. url: https://www.fhwa.dot.gov/bridge/nbi.cfm (visited on 05/10/2021).

Goodman, B. and Flaxman, S. (2016). "EU regulations on algorithmic decision-making and a "right to explanation"(2016)". In: *arXiv preprint arXiv:1606.08813*.

Hohman, F., Wongsuphasawat, K., Kery, M. B., and Patel, K. (2020). "Understanding and visualizing data iteration in machine learning". In: *Proceedings of the 2020 CHI conference on human factors in computing systems*, pp. 1–13.

Kale, A. (Mar. 2022). "Building Interpretable Methods For Identifying Bridge Maintenance Patterns". In: *UNO Student Research and Creative Activity Fair*.

Liao, Q. V., Gruen, D., and Miller, S. (2020). "Questioning the AI: informing design practices for explainable AI user experiences". In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–15.

Linardos, V., Drakaki, M., Tzionas, P., and Karnavas, Y. L. (2022). "Machine Learning in disaster management: Recent developments in methods and applications". In: *Machine Learning and Knowledge Extraction* 4.2, pp. 446–473.

Lipton, Z. C. (2018). "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery." In: *Queue* 16.3, pp. 31–57.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). "Understanding variable importances in forests of randomized trees". In: *Advances in neural information processing systems* 26.

Miller, T. (2019). "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial intelligence* 267, pp. 1–38.

Mittelstadt, B., Russell, C., and Wachter, S. (2019). "Explaining explanations in AI". In: *Proceedings of the conference on fairness, accountability, and transparency*, pp. 279–288.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). "Definitions, methods, and applications in interpretable machine learning". In: *Proceedings of the National Academy of Sciences* 116.44, pp. 22071–22080.

Suryakanthi, T. (Jan. 2020). "Evaluating the Impact of GINI Index and Information Gain on Classification using Decision Tree Classifier Algorithm*". In: *International Journal of Advanced Computer Science and Applications* 11.

Wang, D., Yang, Q., Abdul, A., and Lim, B. Y. (2019). "Designing theory-driven user-centric explainable AI". In: *Proceedings of the 2019 CHI conference on human factors in computing systems*, pp. 1–15.