Interactive Visualization and Validation Techniques for Algorithmic Decision Making: Outcome-Explorer and SliceTeller Approaches

Rishabh Tiwari

Abstract—The use of algorithmic decision-making systems has become widespread, which creates a need for explanations that can help users comprehend the reasoning behind these complex models. Unfortunately, most of the current explanations are challenging for laypeople without specific expertise in machine learning to understand, which could lead to incorrect interpretations. In response to this issue, a new predictive and interactive model based on causality, which is intrinsically interpretable and doesn't require auxiliary models, has been presented. Outcome Explorer [1], a causality-guided interactive interface, was developed to address this problem. It was tested through think-aloud sessions with three expert users and a user study with 18 non-expert users. The tool effectively supported the explanation needs of expert users and enabled non-expert users to easily understand the inner workings of a model. To meet the critical product requirements for release, ensure fairness for different groups, and achieve consistent performance in various scenarios, thoroughly evaluating machine learning models is crucial. To address this, a new tool called SliceTeller [10] has been introduced. It allows users to debug, compare, and improve machine learning models using critical data slices. SliceTeller automatically identifies problematic slices in the data, helps users understand why models fail, and presents a new algorithm, SliceBoosting, to estimate trade-offs when prioritizing optimization over certain slices. The system empowers model developers to compare and analyze different model versions during iterations, allowing them to choose the best model version for their applications. The power of SliceTeller in debugging and improving product-quality ML models has been demonstrated through three use cases, including two real-world product development scenarios.

1 Introduction

Designing an explainable AI (XAI) [6], [2] platform to support non-expert users is challenging because they have different goals, reasons, and skill sets for interpreting a machine learning model than expert users. For instance, a machine learning practitioner with significant data science expertise will want to interpret a model to ensure its accuracy and fairness. On the other hand, a non-expert user will want to understand the service the model provides and gain trust in it. Since most of these users lack background knowledge in data science and machine learning, they need an easy-to-understand visual representation of the model. Machine learning is widely used in critical applications, such as autonomous driving, medical imaging, industrial fire detection, and credit scoring. Therefore, it is essential to evaluate these models thoroughly to assess their capabilities and limitations before deployment. Model mistakes can cause serious consequences in the real world, such as safety issues in driver assistance and industrial systems, misdiagnoses in medical analysis, and biases against minorities.

The goal of Outcome Explorer is to provide an interactive and predictive model that can explain the reasoning behind complex algorithmic decision-making systems in a way that is easy to understand for both expert and non-expert users. The aim is to ensure accurate interpretations of the models and promote transparency and trust. And, the goal of SliceTeller is to provide a tool that enables users to debug, compare, and improve machine learning models using critical data slices. It helps identify problematic slices in the data, understands why models fail, and presents an efficient algorithm, SliceBoosting, to estimate trade-offs when prioritizing the optimization over certain slices. The goal is to empower model developers to compare and analyze different model versions during iterations, allowing them to choose the best model version for their applications.

2 Motivation

Machine Learning models can be complex and difficult to understand and interpret for a non-expert user as well as for people who have expertise until it is clear and understandable or the output is trustworthy. Transparent decision-making is critical, such as in fields like medicine, finance and law. Outcome Explorer solves the problem of interpretability and transparency in decision-making algorithms. Real world Machine Learning applications need to be thoroughly evaluated for model release in order to ensure fairness and to achieve a consistent performance in various scenarios. Discovering where a model fails, understanding why they fail and mitigating these problems, SliceTeller is a tool, that allows users to debug, compare and improve ML models driven by critical data slices.

3 Related work

Explainable AI (XAI) tools and methods are divided into two categories: post-hoc explanation and explanation via interpretability. SHAP [3] and LIME [5] are examples of post-hoc explanation methods that are model-agnostic and can explain the predictions of any machine learning model. However, recent research has argued that these methods can be misleading and make model understanding even more complicated. On the other hand, several visual analytics approaches [8] have shown that they make causal analysis more accessible and intuitive for non-expert users, providing them with visual representations of the data and relationships, improving their ability to make data-driven decisions. Nevertheless, these visual analytics approaches neglect temporal variables, and users still want interactive causal graph customization.

Several visual analytics (VA) techniques have been developed to support the creation and analysis of data slices, including Manifold [9] and Errudite [7]. Manifold allows data subgroup analysis based on human interaction but do not provide automated methods for data slicing. Errudite focuses on providing a domain-specific language for data grouping with unstructured text data analysis, while SliceTeller mainly focuses on structured data. However, most of these works focus on data exploration rather than providing model or data optimization recommendations.

4 Background

In this section we will be discussing about the background of both the papers including the formative studies done with non-expert users in Outcome explorer. We will also be putting an emphasis on the system workflow and the data and domain requirements for the SliceTeller tool.

4.1 Outcome Explorer

Outcome Explorer uses Pearl's Structural Causal Model (SCM) [4] to define causal relationships between variables. In SCM, causal relationships are represented by a directed acyclic graph (DAG), where

Seminar Visual Computing, March 1–3, 2023.
Visual Computing Group, Heidelberg University, Germany.

variables are categorized as exogenous (U) or endogenous (V). Exogenous variables have no parents in the DAG and are independent and unexplained by the model, while endogenous variables are fully explained by the model and represent the causal effects of exogenous variables. The authors describe how the causal structure between variables can be obtained through three different approaches: using domain expertise or prior knowledge, using automated algorithms that utilize conditional independence tests to find causal structure, or a combination of both. They also explain how to use SEM to parameterize the model once the causal structure is learned, and how to use the do operator to simulate a causal intervention on the model.

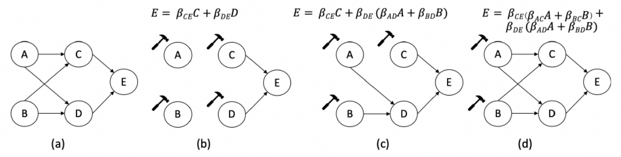

$E = \beta_{CE}C + \beta_{DE}D$
$E = \beta_{CE}C + \beta_{DE}(\beta_{AD}A + \beta_{BD}B)$
$E = \beta_{CE}(\beta_{AC}A + \beta_{BC}B) + \beta_{DE}(\beta_{AD}A + \beta_{BD}B)$
A
B
C
D
E
(a)
(b)
(c)
(d)

Fig. 1. Prediction in a causal model. The hammer icon represents intervention. (a) True causal model. (b) Interventions on all feature variables. The causal links leading to nodes C and D are removed since the values of C and D are set externally. (c) Interventions on nodes A, B, and C. (d) Interventions on nodes A and B. In the path equations above models (b)-(d), the β are standardized regression coefficients estimated from the data [1].

4.1.1 Formative study with Non-Expert users

During the research, interviews were conducted with non-expert users to understand their expectations of decision-making interfaces. It was concluded that users generally prefer automated systems over human assistance because they are quick and efficient. However, they still require human assistance when the automated system is unable to provide them with the necessary information. Users also expressed the need for transparency in decision-making, as they feared that automated systems might not provide them with optimal results. Moreover, users often received different decisions from their friends, which could be due to the different capabilities of each user.

4.2 SliceTeller

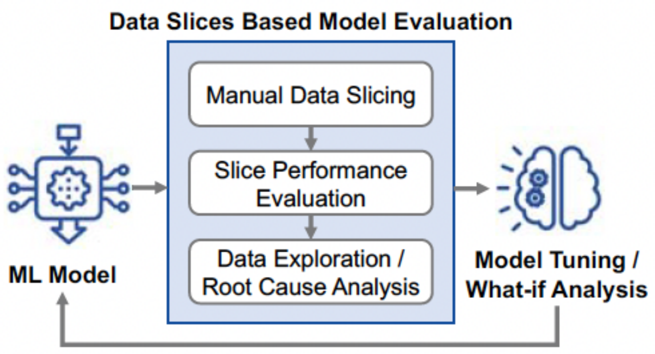

Data Slices Based Model Evaluation
Manual Data Slicing
Slice Performance Evaluation
Data Exploration / Root Cause Analysis
ML Model
Model Tuning / What-if Analysis

Fig. 2. Product MLOps engineer's workflow for model validation and iteration over critical data slices. Experts sliced their data based on product and domain requirements, computed model performances per slice, and explored the data to identify the root causes for potential model mistakes. Based on these observations, they would iterate over the model, by retraining while re-prioritizing certain data slices over others. [10].

We now describe the development of a data slice-driven approach for machine learning model validation, called SliceTeller. This approach was developed through collaboration with three MLOps engineers(Fig 2 shows the MLOps Engineer's workflow) working on critical applications such as autonomous driving and fire detection. The engineers faced challenges in thoroughly evaluating their models, understanding failure cases, and optimizing their models for critical use cases. Based on these observations, the authors identified four key desiderata for a data slice-driven model validation system: (1) automatic identification of data slices, (2) exploration and valuation of data slices, (3) slice-based model optimization, and (4) slice-based model comparison. SliceTeller was developed to meet these requirements, providing a user-friendly interface for exploring and optimizing machine learning models in critical applications.

4.2.1 System Workflow

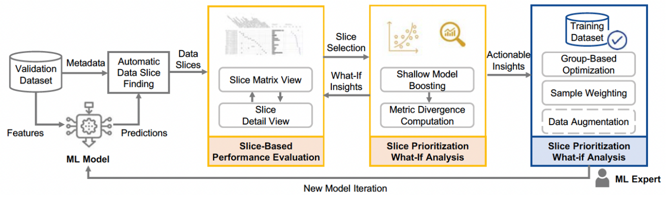


Fig. 3. Workflow of the model analysis and improvement using SliceTeller. The validation data, together with the model predictions, are used for the automatic slice identification. The produced data slices can be explored using our VA solution (Slice Matrix and Slice Detail View). Users can prioritize groups of data slices and quickly evaluate the effect of this action on the rest of the model slices. Finally, experts can use the insights gained from the system to fine tune the model and continue the analysis with SliceTeller. [10].

SliceTeller is a tool that presents the evolution of machine learning models by analyzing and comparing data slices. Figure 3 represents the workflow of the model showing the input which consists of a validation dataset with validation data, metadata, and ground truth labels. The system uses an automatic slice finding algorithm to identify data slices where performance measures are different from the overall model performance. Then, a visual analysis system allows users to visualize and summarize the data slices using the Slice Matrix View and Slice Distribution View. The user can test mitigating measures on critical slices using the SliceBoosting algorithm. Finally, the user can export the selected slices back to their programming environment to make changes to data, hyperparameters, or the model and compare models using SliceTeller.

5 Module Design

This sections deals with the module design of Outcome Explorer tool and SliceTeller. Both the tool designs will be discussed below.

5.1 Outcome Explorer

In the Fig. 4 and 5, Outcome-Explorer has formulated five design guidelines to support both expert and non-expert users in creating and interpreting predictive causal models. The guidelines include creating a two-module design, allowing expert users to interactively create and evaluate the model, designing a visualization and interaction for the causal DAG to interpret the model accurately, supporting explanation queries from non-expert users, and allowing non-expert users to freely change input features while evaluating the realism of their chosen configuration.

5.2 SliceTeller

SliceTeller is a visualization tool that helps users explore and analyze problematic data slices in machine learning models. In the Fig 6, the main components of SliceTeller are the Slice Matrix and the Slice Detail View. The Slice Matrix provides a summary of all the data slices with a performance metric that diverges from the overall model. It uses an adaptation of the UpSet matrix encoding, where sets are represented as rows, and set members, as columns. The user can drill down on the slices in order to explore one or more data slices simultaneously

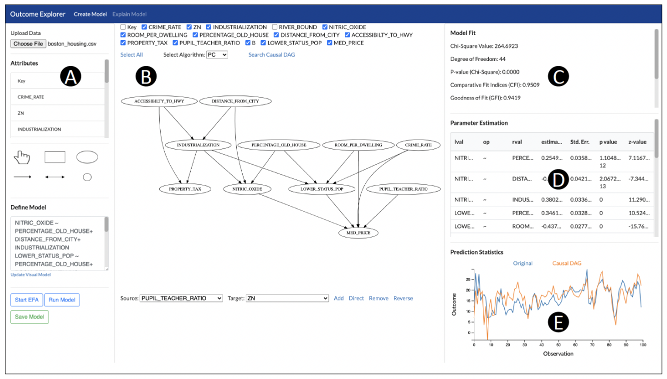


Fig. 4. Model Creation Module of Outcome-Explorer. A) Control Panel. B) Causal structure obtained from the search algorithms. Users can interactively add, remove, and direct edges in the causal structure. C) Model fit measures obtained from Structural Equation Modelling (SEM). D) Parameter estimation for each relations (beta coefficients). E) A line chart showing the prediction accuracy of the Causal Model on the test set. [1].

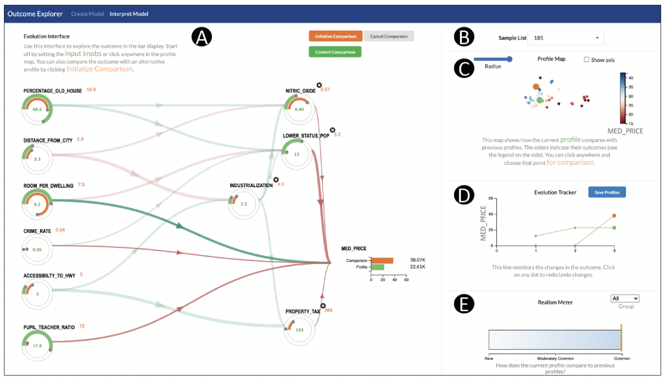


Fig. 5. Model Interpretation Module of Outcome-Explorer. A) Interactive causal DAG showing causal relations between variables. Each node includes two circular knobs (green and orange) to facilitate profile comparisons. The edge thickness and color depict the effect size and type of each edge. B) Sample selection panel. C) A biplot showing the position of green and orange profiles compared to nearest neighbors. D) A line chart to track the model outcome and to go back and forth between feature configuration. E) Realism meter allowing users to determine how common a profile is compared to other samples in the dataset. [1].

using the Slice Detail View. The System Menu allows users to switch between data slices from multiple ML models, summarize model slices, and perform What-if analyses to estimate the effect of optimizing the model for a particular data slice. SliceTeller identifies data slices using DivExplorer and enables users to interactively analyze and diagnose model performance issues.

6 Methodology

6.1 Outcome Explorer

The profile map in Figure 5(C) is a biplot that displays the nearest neighbors of a profile (DG4). A PCA is run on the selected points to compute the biplot. Users can adjust the radius of the neighborhood through the ”Radius” slider, and neighbors are colored based on outcome value. Users can hover over any point to compare it with the green profile (DG4) and click on any point to set that point as a comparison case for a more detailed analysis. The green and orange disks move around the map as the user changes profiles in the causal model.

The Evolution Tracker (Figure 5(D)) is a line chart with two lines for two profiles in the system. Users can save a particular state in the tracker by clicking on the ”Save Profiles” button and can click on any point to move back and forth between different states.

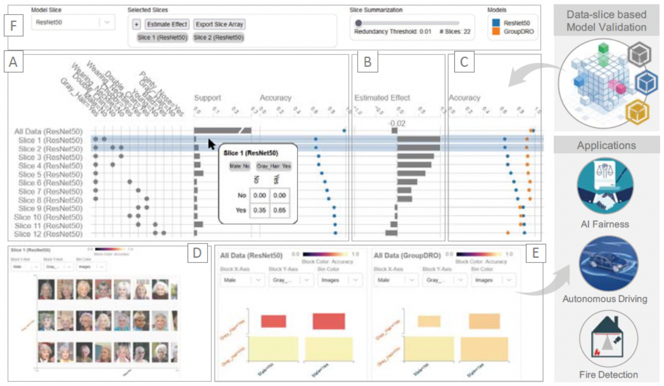


Fig. 6. SliceTeller applied to the comparison of two machine learning models (ResNet50 and GroupDRO) for hair color classification (gray hair, not gray hair), trained on the CelebFaces Attributes Dataset (CelebA). (A) Slice Matrix: The data slices (represented as rows), slice descriptions (encoded as columns), and slice metrics (Support and Accuracy). Slices are sorted by model accuracy. (A - Tooltip) Confusion matrix for Slice 1. (B) Estimated effects of optimizing the model for two data slices (Slices 1 and 2, highlighted in blue). (C) Accuracy comparison between the two models, ResNet50 and GroupDRO. (D) Slice Detail View containing image samples from a data slice. (E) Slice Detail View containing the comparison of two data slices using the MatrixScape visualization. (F) System menu, containing options for model selection, effect estimation of focusing on a slice during model training, and data slice summarization. [10].

The Similarity (or Realism) Meter is used to determine how common a user’s profile is compared to the existing population captured by the dataset (DG5). It is a safety check to prevent unreasonable expectations generated by the interactive interpretation module. A Gaussian Mixture Model is fitted on the existing data points in multivariate space, and the probability of a datapoint belonging to a component is calculated using the Expectation-Maximization algorithm. P(Cijx) is interpreted as a scale of how ”real” a datapoint is to the other members of a component and is translated to a human understandable meter with ”Rare,” ”Moderately Common,” and ”Common” as the interpretations.

6.1.1 Exploring Outcomes and What-if Analysis

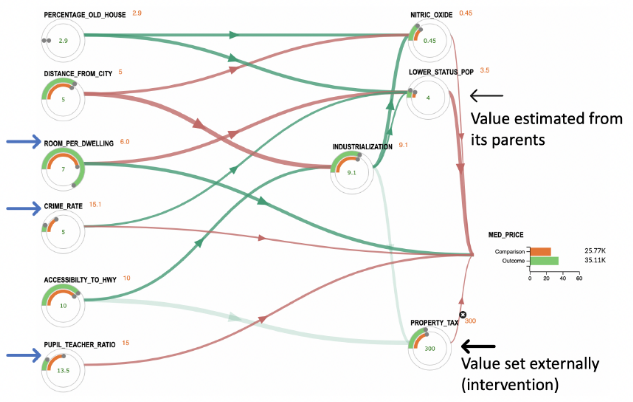


Fig. 7. Asking what-if questions in Outcome-Explorer. A user can keep one profile (green) fixed, and change the other profile (orange) to ask what-if questions. The blue arrows indicate the changes in the orange profile. Note that property tax is set to 300 by the user. As a result, changing its parents will not affect property tax. The other endogenous variables are estimated from their parents. [1]

A case study has been used in the Outcome Explorer to conduct a what-if analysis. We start by understanding the causal relationships between variables and then began adjusting the values of various knobs in the interface. We observe that changing the value of an internal node caused the edges leading to that node to become blurred. After finding an ideal neighborhood with a median price of \$35,000, we wanted to adjust the variables to bring the median price down to the budget of \$25,000. We used the comparison mode to create an orange profile identical to the green profile and iteratively changed the variables to achieve the desired median price. We then used the tracker and realism meter to monitor the changes she was making.

Outcome Explorer allows for what-if analysis by enabling users to keep one profile fixed (the green one) while changing the other (the orange one) to ask hypothetical questions. The blue arrows in the interface indicate the changes in the orange profile. The user can set the values externally to conduct the analysis, and changing the parents will not affect the external value. Endogenous variables are estimated from their parents. Overall, Outcome Explorer provides a powerful tool for analyzing hypothetical scenarios and understanding how changes to one variable can affect the outcome of a system.

6.2 SliceTeller

6.2.1 SliceBoosting Algorithm

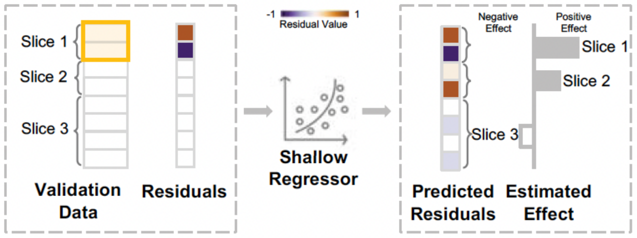


Fig. 8. Illustration of SliceBoosting algorithm to estimate model optimization effect. Given the selected slice 1, we train a shallow regressor to estimate that, under the ideal scenario where the optimization model correctly fits to slice 1, how will the performance on slice 2 and 3 be affected. We design the prediction target of the regressor as the residuals of the original model predictions to the ground truth validation labels. To focus on the effect estimation of slice 1, we set the residual values only for slice 1, while keeping the residuals of all other slices as 0. The predicted residuals from the regression are in the range of $[-1,1]$. We then aggregate the sample-level residual predictions to obtain slice-level estimation results: how will the accuracy on these slices increase or decrease? [10]

SliceBoosting is a machine learning algorithm(refer to Figure 8) that enables fast experimentation with different data slices to evaluate trade-offs between them, without the need to train multiple models from scratch. The algorithm achieves this by training a shallow model to approximate the residuals (errors) of the full model on selected validation slices, in a similar approach to boosting. The shallow model is trained to fit the residuals of the original model on the selected slices, which are calculated as the difference between the ground truth validation labels and the predicted labels. To focus on the effect estimation of slice 1, the residual values are set only for slice 1, while keeping the residuals of all other slices as 0. The predicted residuals from the regression are in the range of $[-1,1]$. By focusing on the selected slices, the algorithm can estimate the effect of subgroup optimization and allow for quick experimentation. The algorithm is based on two assumptions: 1) the validation set has a similar distribution to the training data and 2) the optimization approach is powerful enough to steer the model to make correct predictions on the selected validation slices.

6.2.2 Model Optimisation

We discuss about a tool called SliceTeller for optimizing machine learning models by improving performance on selected "slices" of data while minimizing the trade-off for the averaged model performance on the entire dataset. Two optimization methods are introduced: importance weighting and group DRO in the Figure 9.

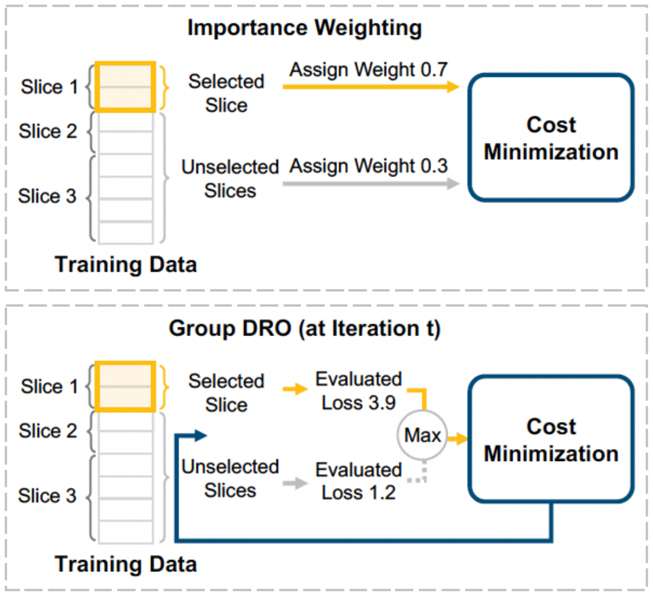


Fig. 9. Illustration of the model optimization methods considered in our work. During re-training, they prioritize slices in the training data according to user's decision. [10].

Importance weighting changes the loss function by assigning heavier weights to the training samples in the worst-performing slices. The weight for a slice is calculated as:

$$w_S = \frac{M \times n_{train}}{N_{train}}$$

where N_{train} is the number of samples in the training set, M is the total number of slices, and n_{train} is the number of samples in slice S. The modified expected loss is then defined as:

$$E(x_{train}, y_{train}, S) \sim P[w_S l(\theta; (x_{train}, y_{train}))]$$

where P is the distribution of training data X_{train} and l is the loss.

Group DRO optimizes for the worst-case loss over the groups in the training data, rather than optimizing for the averaged loss over entire training data. The expected loss is defined as:

$$\max_S E(x_{train}, y_{train}) \sim P_S[l(\theta; (x_{train}, y_{train}))]$$

where P_S is the distribution of the worst-performing slices. During training, the optimization can be conducted by either recording the historical losses of all groups or utilizing gradient ascent.

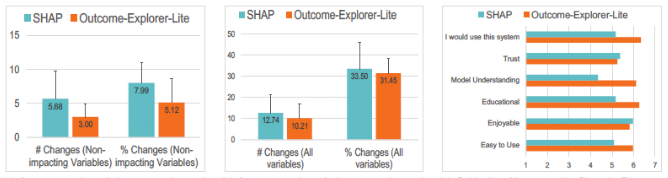
7 Evaluation and Results

Many different results for expert and non-expert users were presented in the two studies. These included testing several test cases with expert evaluation and the input from the non-expert users who have no prior knowledge of operating a Machine learning model.

7.1 Outcome Explorer

7.1.1 Study with Non-Expert Users

A user study was conducted to validate three hypotheses about the Outcome-Explorer tool. The Figure 10 shows that the tool was compared to the SHAP explanation technique in a within-subjects experiment with two conditions: one using SHAP and another using Outcome-Explorer-Lite. The study included 18 participants with no machine



(a) Average number of changes and magnitude of changes on non-impacting variables (b) Average number of changes and magnitude of changes on all variables (c) Subjective Measures (1: Strongly Disagree, 7:Strongly Agree).

Fig. 10. Study Results. The average number of changes and the average magnitude of changes % made to (a) non-impacting variables, and (b) all variables to reach the target outcomes. (c) Average self-reported subjective measures. Error bars show +1 SD. [1]

learning expertise and evaluated the tool's impact on actual decision-making tasks related to the Boston housing and PIMA Diabetes datasets. The study also collected subjective measures such as model understanding, trust, and usability.

The first hypothesis (H1) was that Outcome-Explorer would improve a user's understanding of the embedded predictive causal model compared to SHAP. The study found that Outcome-Explorer visualized the interplay of variables better than SHAP, which only estimates feature contributions, and does not explain why some variables do not affect the outcome. The study measured the number and magnitude of changes made to non-impacting variables, and the results showed that Outcome-Explorer users made fewer changes and smaller magnitude changes, indicating better understanding.

The second hypothesis (H2) was that Outcome-Explorer would increase a user's efficiency in reaching a desired outcome compared to SHAP. The study did not find a significant difference in efficiency between the two tools.

The third hypothesis (H3) was that Outcome-Explorer would be easy to use. The study found that participants rated Outcome-Explorer as more user-friendly and easier to use than SHAP.

Overall, the study found that Outcome-Explorer had advantages over SHAP in terms of model understanding and ease of use but did not show significant differences in efficiency for actual decision-making tasks. The study provides evidence that Outcome-Explorer could benefit non-expert users in understanding predictive causal models.

7.1.2 Study with Expert Users

Three ML practitioners, with post-graduate degrees and research experience in XAI, Fairness, and Data Ethics, were invited to examine Outcome-Explorer. The participants were familiar with statistical causal analysis, and sessions were conducted via Skype with the tool deployed on a web server. The participants chose one dataset and performed tasks in the Creation and Interpretation Modules while providing feedback. The participants found the tool to be comprehensive, engaging, and fun, with potential use for expert users in the ML pipeline. Participants also emphasized the need for appropriate disclaimers to communicate the implications of causal relations to end-users.

7.2 SliceTeller

This section showcases the effectiveness of SliceTeller in analyzing, validating, and enhancing ML models through two use cases. The first use case demonstrates how a hypothetical user can utilize SliceTeller to verify the existence of data biases in image classification problems. The next case study, derived from interviews with the MLOps engineers working on real industry products, illustrate how SliceTeller can be used to improve the accuracy of models on crucial data slices.

7.2.1 Bias Detection for AI Fairness in Image Classification Models

This use case describes how a PhD student interested in computer vision, used SliceTeller to identify a gender-related bias in ResNet50's performance on the CelebFaces Attributes Dataset (CelebA). CelebA contains over 202,000 face images of 10,177 celebrities, along with 40

Slice	Description	ResNet-50	Group DRO
0	All Data	**0.98 / 0.98**	0.95 / 0.95
1	Gray_Hair=Yes, Male=No	0.65 / 0.5	**0.91 / 0.81**
2	Gray_Hair=Yes, Double_Chin=No, Wearing_Necktie=No	0.65 / 0.69	**0.90 / 0.93**

Fig. 11. Table 1. Accuracy of Image Classification Models(Val/Test) [10]

binary attribute annotations, including gender, skin color, and smiling, for each image. ResNet50 model was trained to classify gray hair and achieved an overall validation accuracy of 0.98. However, SliceTeller revealed that the model performed poorly on Slice 1, which represents females with gray hair. It was found that the low accuracy was due to the highly skewed data distribution in that slice. Using SliceTeller's effect estimation functionality, Slices 1 and 2 were optimized together, which improved the performance of the worst eight slices significantly. A GroupDRO model was retrained that dynamically optimized the two slices and achieved higher accuracy on the worst slices. The performance of both models were compared using SliceTeller and it was confirmed in the Fig 11 that the optimized model improved the classification of gray-haired females while having a small trade-off on overall model performance.

7.2.2 Case 2: Ultrasonic Object Height Classification for Autonomous Driving

Slice	Description	Model_1	Model_2	Model_3
0	All Data	0.88 / 0.74	0.92 / **0.84**	**0.95** / 0.84
1	Clutter=High, Weather=Sunny, Temp=(20.6, 29.0]	0.53 / 0.60	0.82 / **0.87**	**0.96** / 0.85
2	Clutter=High, Speed=(1.5, 3.4]	0.56 / 0.67	0.82 / **0.86**	**0.89** / 0.84
7	Object=Charging Curbstone	0.67 / 0.36	0.94 / 0.85	**0.91 / 0.87**
10	Object=Containerside	0.70 / 0.66	0.97 / 0.99	**1.00 / 1.00**
139	Weather=Rain	**0.98** / 0.83	0.88 / 0.88	0.95 / **0.90**
142	Object=Wall	**0.99 / 0.88**	0.81 / 0.80	0.93 / 0.83

Fig. 12. Table 2. Accuracy of Image Classification Models(Val/Test) [10]

In this scenario, the goal is to predict object height as a binary label for autonomous driving using ultrasonic sensors. The expert used XGBoost model due to the limited compute power of the car's onboard processor. The data was split into three sets: training (60%), validation (30%), and testing (10%). The expert used SliceTeller to analyze the dataset, which identified 145 data slices. The expert used Slice Summarization to reduce the number of data subsets for analysis and obtained 11 distinct data slices. The expert noticed that some data slices performed significantly worse than the overall model. They then used the SliceBoosting algorithm to estimate the effect of training the model with higher weights on these slices. A new XGBoost model was trained using the importance weighting method with higher sample weights to the samples belonging to the slices that performed worse. The expert then trained another model, weighting the samples by the previously optimized slices, as well as critical slices for autonomous driving. They visualized the worst data slices from all three models and found that Model 3 performs better than the other models on the worst data slices. Table 2 shows the performance of the three models on the data slices of interest for validation and test data.

8 Limitation

Outcome Explorer, a visual interface that leverages causal modeling for interpretable decision-making, has a few limitations that need to be addressed. Firstly, it is computationally intensive, requiring large sample sizes to generate reliable results. Secondly, the tool makes assumptions about causality that may not hold true in all situations, leading to potentially misleading or inaccurate outcomes. Finally, the interface may be difficult for non-technical users to interpret, limiting its practical utility in certain settings. These limitations need to be addressed to enhance the effectiveness and usability of Outcome Explorer.

SliceTeller has some limitations that need to be addressed, including limited generalization, lack of representation, lack of data, and potential overfitting. Since SliceTeller validates a model on a specific subset of data, the model's performance may not generalize well to new and unseen data. This is due to the possibility of the model overfitting to the specific characteristics of the data slice it was trained on. Moreover, validation on a small data slice may not provide a good representation of the entire dataset, which could lead to the model underperforming or overperforming on unseen data. Another limitation is that slicing the data may result in a dataset that is too small to work with, which can lead to poor model performance and lack of confidence in the results.

9 Comparison of two methods

Outcome Explorer and Slice Teller are two methods that focus on explaining the behavior of predictive models. However, they differ in their target users, design goals, and features.

Outcome Explorer targets both expert and non-expert users and provides a two-module design that supports the creation and interpretation of a causal model. Its design goals include creating a transparent and interactive interface that allows users to understand the causal relations in the model, make decisions, and evaluate the realism of input feature configurations. It also provides several features that allow users to compare their results with other results in the same field, and to explore the causal DAG through visualization and interaction. For instance, a doctor can compare a patient's disease with the side effects of other patients to predict the outcome.

On the other hand, Slice Teller focuses only on expert users and provides a data slicing technique to handle large datasets. Its design goals include providing detailed information of the data slices, helping users to identify and correct errors, and allowing experts to conclude at a faster time. It is particularly useful in image-based fire detection applications. For example, SliceTeller can aid in image-based fire detection, identifying potential fires at an early stage.

Overall, while Outcome Explorer targets a wider range of users and provides more comprehensive features to support model creation and interpretation, Slice Teller is more focused on data handling and analysis, particularly for experts dealing with large datasets. Both methods have their strengths and limitations, and the choice between them would depend on the specific needs and expertise of the users.

10 Conclusion

The paper describes two different systems, Outcome-Explorer and SliceTeller. Outcome-Explorer is an interactive visual interface that makes use of causal models to provide explanations for decisions made by machine learning models. The authors suggest that the system could be used by bank advisors or insurance agents to discuss options with clients, or made available on a bank or insurance website. They plan to investigate how complex a model can get while still being understandable by non-expert users. On the other hand, SliceTeller is a visual analytics tool designed for data slice-driven validation of ML models. It allows users to identify problematic data slices, investigate failure cases, understand optimization trade-offs, and iterate on new model solutions. The authors worked closely with industry ML Ops engineers and domain experts to develop and improve SliceTeller, and plan to incorporate it into ML development workflows to facilitate fast product iteration and model release.

11 Future Work

Future work for Outcome Explorer, some potential improvements include expanding the system's functionality to handle more complex models and datasets, incorporating more advanced statistical analysis techniques, and incorporating real-time data streams for dynamic model evaluation. Additionally, future research could explore ways to enhance the user interface to make it more intuitive and user-friendly.

As for SliceTeller, some potential improvements include incorporating additional machine learning algorithms and techniques, expanding the system's capabilities to handle more diverse types of data, and integrating more advanced optimization techniques. Additionally, future research could explore ways to enhance the system's scalability and performance, as well as ways to improve its visualization and reporting capabilities. Finally, it may be useful to investigate how SliceTeller can be integrated into existing ML development workflows and automated deployment pipelines.

12 My Opinion

In my opinion, Outcome-explorer is an effective tool for both experts and non-experts to understand algorithm functioning and trust the outcomes. However, there is room for improvement to make the model more transparent and user-friendly for non-expert users who may struggle with understanding the variables and DAG usage. Nonetheless, the tool provides helpful comparison and what-if analysis features for all users to trust and reuse it.

Regarding SliceTeller, my opinion is that the model still requires fine-tuning to achieve zero-errors and ensure customer-defined quality over handcrafted slices. I believe that the tool is still complex for expert users as well, and there is a need for improvement in the slice-boosting algorithm to enhance accuracy and efficiency. Overall, it is a commendable data implementation tool that helps identify flaws and improve outcomes.

References

[1] M. N. Hoque and K. Mueller. Outcome-explorer: A causality guided interactive visual interface for interpretable algorithmic decision making. *IEEE Transactions on Visualization and Computer Graphics*, 28(12):4728–4740, 2021.

[2] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, and S. Friedler. Problems with shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pp. 5491–5500. PMLR, 2020.

[3] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[4] J. Pearl. Causality: models, reasoning, and inference, 1980.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

[6] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[7] T. Wu, M. T. Ribeiro, J. Heer, and D. S. Weld. Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 747–763, 2019.

[8] X. Xie, F. Du, and Y. Wu. A visual analytics approach for exploratory causal analysis: Exploration, validation, and applications. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1448–1458, 2020.

[9] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert. Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE transactions on visualization and computer graphics*, 25(1):364–373, 2018.

[10] X. Zhang, J. P. Ono, H. Song, L. Gou, K.-L. Ma, and L. Ren. Sliceteller: A data slice-driven approach for machine learning model validation. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):842–852, 2022.