# InterpretStack: Interpretable Exploration and Interactive Visualization Construction of Stacking Algorithm

Yu Wang Fudan University Shanghai, China 21210980075@m.fudan.edu.cn

Jing Lu Fudan University Shanghai, China jinglu20@fudan.edu.cn

Le Liu Northwestern Polytechnical University Xian, Shaanxi, China lel@nwpu.edu.cn

Junping Zhang Fudan University Shanghai, China jpzhang@fudan.edu.cn

# ABSTRACT

Stacking is a common ensemble method in ensemble learning that improves the predictive performance of the model by combining the outputs of multiple base models through a meta model. However, selecting appropriate base models and designing an effective Stacking structure remains a major challenge. Additionally, the complex non-linear structure of Stacking makes it difficult to interpret its decision-making process. To address these issues, we propose InterpretStack, an interactive visual analysis system that enhances the interpretability of Stacking methods while improving model performance InterpretStack integrates classic and effective feature importance assessment methods and hyperparametric space exploration methods. It introduces human decision-making into the design of Stacking structures and the selection of features and models through interactive exploration and human-computer interaction during Stacking generation. From the perspective of feature importance, model importance, and model correlation, the system enhances the interpretability of the Stacking decision-making process by scoring feature importance, model correlation coefficients, and SHAP values. We validate the effectiveness of InterpretStack through a comprehensive case study and user study, demonstrating significant improvements in model interpretability and performance.

#### VLDB Workshop Reference Format:

Yu Wang, Jing Lu, Le Liu, Junping Zhang, and Siming Chen. InterpretStack: Interpretable Exploration and Interactive Visualization Construction of Stacking Algorithm. VLDB 2024 Workshop: Relational Models.

#### VLDB Workshop Artifact Availability:

The source code, data, and/or other artifacts have been made available at [URL\\_TO\\_YOUR\\_ARTIFACTS.](URL_TO_YOUR_ARTIFACTS)

Proceedings of the VLDB Endowment. ISSN 2150-8097.

Siming Chen Fudan University Shanghai, China simingchen@fudan.edu.cn

# 1 INTRODUCTION

Ensemble learning, exemplified by methods like Stacking [\[7\]](#page-8-0) and its variants, has become pivotal in machine learning. It leverages the collective wisdom of multiple models to enhance predictive accuracy while mitigating bias and variance. Stacking, in particular, aggregates predictions from diverse base models [\[26\]](#page-8-1), which are subsequently used as features in a meta model to generate final predictions. These base models can vary widely—from support vector machines [\[19\]](#page-8-2) to logistic regression [\[14\]](#page-8-3) and neural networks [\[9\]](#page-8-4)—offering flexibility in model composition.

While Stacking traditionally involves a single layer of base models feeding into a meta model, advanced configurations employ multi-layer stacking to accommodate complex datasets, thereby enhancing model fitting capabilities. However, these sophisticated designs necessitate careful consideration to prevent overfitting, especially with limited or noisy data [\[7\]](#page-8-0).

Other ensemble methods like Bagging [\[4\]](#page-8-5) and Boosting [\[8\]](#page-8-6) also enhance model performance through sampling and iterative learning, respectively. Unlike Stacking, Bagging reduces variance by aggregating predictions from multiple samples of the same type of model, while Boosting iteratively corrects model biases with sequentially trained models of the same type.

Despite its strengths, Stacking's opacity presents challenges in interpretability, crucial in domains like healthcare. While efforts such as Chatzimparmpas et al.'s StackGenVis[\[5\]](#page-8-7) have introduced tools to visualize and refine stacking models, these tools often fall short in providing comprehensive explanations for decision-making processes, limiting their utility in critical applications. Recent works, such as MetaStackVis[\[22\]](#page-8-8), have focused on visually-assisted performance evaluation of metamodels, providing valuable insights into model selection and evaluation. However, these approaches often lack interactive exploration and comprehensive decision-making support.

In contrast, InterpretStack integrates feature importance assessment, model correlation analysis, and SHAP values into an interactive visual analysis system, enabling users to make informed decisions throughout the stacking model design and evaluation process. This holistic approach addresses the limitations of existing tools by offering a more detailed and interactive exploration of stacking ensemble methods.

This work is licensed under the Creative Commons BY-NC-ND 4.0 International License. Visit<https://creativecommons.org/licenses/by-nc-nd/4.0/> to view a copy of this license. For any use beyond those covered by this license, obtain permission by emailing [info@vldb.org.](mailto:info@vldb.org) Copyright is held by the owner/author(s). Publication rights licensed to the VLDB Endowment.

<span id="page-1-0"></span>

Figure 1: Interactive visual analytics system InterpretStack: (a) the DataSet and Performance metric manager view displays the data set selection and performance evaluation metric information, (b) the Features view displays the feature importance and provides feature selection, (c) the Base Model view displays the Pearson correlation coefficient between the base models and provides base model selection, (d) the meta model view provides meta model selection, (e) the Stacking Components view displays and provides the stacking structure design including feature and model selection, model update, layer selection, etc, (f) The Evaluation view displays the performance evaluation results of the base model training set and validation set, (g) the History Stacking view displays the performance evaluation result ratio of each stacking combination validation set.

To summarize, The contributions of this paper include:

- Providing insights into Stacking's decision-making process through feature and model importance metrics.
- Streamlining Stacking model design with interactive visualizations and automated hyperparameter tuning.
- Developing an interactive visual analytics system for iterative model refinement and performance evaluation.

Chapter 2 reviews ensemble learning, Stacking methodologies, and related interactive AI applications. Chapter 3 outlines system requirements and design criteria, while Chapter 4 details the technical methods employed, including Bayesian optimization for hyperparameter tuning. Chapter 5 validates the system's efficacy through performance benchmarks, and Chapter 6 presents findings from a user study. Finally, Chapter 7 concludes with a summary and future directions for enhancing Stacking's utility and interpretability.

# 2 RELATED WORK

This section reviews related work on Ensemble Learning and Stacking and LLM-empowered vInteractive Artificial Intelligence.

#### 2.1 Ensemble Learning and Stacking

Ensemble learning, encompassing methods like Bagging and Boosting, has significantly enhanced model accuracy and robustness by aggregating predictions from multiple models. Stacking, a key ensemble technique, combines base model predictions using a metamodel to improve overall performance [\[26\]](#page-8-1). Since its inception, Stacking has been extensively applied and rapidly evolved with various enhancements, such as multi-level Stacking and integration with neural networks [\[10\]](#page-8-9). Researchers have continuously proposed improvements, including methods that combine ensemble learning and Stacking [\[30\]](#page-8-10).

Wolpert [\[26\]](#page-8-1) first introduced Stacking in 1992, presenting a twostage model combination process. Breiman [\[4\]](#page-8-5) further advanced ensemble learning, introducing Bagging, which aggregates predictions from models trained on different data samples. While Bagging and Stacking differ in approach, both aim to enhance predictive performance by combining multiple models.

Stacking's application spans various fields. For instance, Zhao et al. [\[29\]](#page-8-11) used Stacking in speech recognition, combining deep learning with ensemble methods. Alaoui et al. [\[1\]](#page-8-12) applied Stacking for network attack detection, using CNNs[\[15\]](#page-8-13) and LSTMs[\[11\]](#page-8-14) to enhance prediction accuracy. In financial risk assessment, Zhang et al. [\[28\]](#page-8-15) improved credit risk control by integrating multiple learners, resulting in better accuracy and generalization.

# 2.2 Interactive Artificial Intelligence

Interactive AI systems enable real-time, bidirectional communication between humans and machines, enhancing user experience and model performance understanding. For example, Li et al. [\[16\]](#page-8-16) proposed interactive visualizations to study neural network loss functions, aiding in network optimization. Kahng et al. [\[13\]](#page-8-17) developed a tool to explore GAN models, while Park et al. [\[20\]](#page-8-18) created HyperTendril to link model performance with hyperparameters. Amershi et al. [\[2\]](#page-8-19) introduced Modeltracker, an interactive tool for analyzing machine learning models.

As machine learning models grow more complex, ensuring their interpretability has become crucial. Researchers have developed tools and methods to visualize and explain model decisions. Hohman et al. [\[12\]](#page-8-20) reviewed visualization techniques for deep learning, introducing tools like TensorBoard and Netron. Zeiler et al. [\[27\]](#page-8-21) proposed visual methods to understand CNNs, and Rauber et al. [\[24\]](#page-8-22) developed Netdissect for visualizing neural network activities. Molnar et al. [\[18\]](#page-8-23) detailed explanatory approaches in "Interpretable Machine Learning."

Despite extensive work on model interpretability, few studies focus on explaining ensemble learning decisions, particularly for Stacking. Addressing this gap, we designed an interactive visual analysis system to aid in Stacking model design, understand its decision-making process, and enhance performance.

# 3 TASK ANALYSIS AND DESIGN REQUIREMENTS

In order to realize a visual analysis system that can be interactively explored and can take into account the characteristics, models, parameter selection and structural design in the process of Stacking generation, we summarize the following tasks and requirements that the visual analysis system needs to meet.

# 3.1 Task Analysis

To effectively utilize the Stacking method, several key tasks need to be addressed:

T1: Find the appropriate solution for the Stacking method. Identify suitable Stacking solutions for different scenarios, including dataset division, feature engineering, base model selection, structural design, and ensemble strategy.

T2: Reduce computing resources and workload. Optimize resource usage and reduce workload, crucial given the complexity of models used in Stacking.

T3: Provide explanation for the decision-making process of the Stacking method. Ensure interpretability, especially in critical fields like medicine.

T4: Manage Stacking Evaluation Metrics. Provide various evaluation metrics to comprehensively assess model performance across different tasks.

T5: Base model backtracking. Track and manage different base model combinations and their performance, allowing users to revisit and refine previous models.

T6: Base model and Stacking performance comparison Compare performance across base models and Stacking methods to guide feature and model selection, avoiding redundant features and ensuring proper model fitting.

# 3.2 Design Requirements

Based on the tasks identified, we propose the following design requirements for the visual analysis system:

R1: Improve users' understanding and insights, and provide more perspectives and dimensions. Provide users with multi-dimensional feedback and data stories to facilitate new exploration and a deeper understanding of data.

R2: Efficient and intuitive operator interface. Design an interface that is easy to use and understand, reducing the learning curve and encouraging deeper exploration and decision-making.

R3: Provide a variety of visual coding methods to help quickly obtain information. Use various visual coding techniques to enhance data readability and help users quickly capture and comprehend information.

R4: Support user interaction and exploration to help users design models based on information. Enable users to interact with and explore the system, allowing for feature, model selection, and structural design based on evaluation results.

R5: Consider the background of the target audience of the system to improve user experience. Design the system with the needs of users familiar with the Stacking method in mind, emphasizing professionalism and customization to improve user experience.

# 4 INTERPRETSTACK:OVERVIEW AND APPLICATION

<span id="page-2-0"></span>

Figure 2: Interactive visualization system frame. It mainly includes the calculation and transmission process of internal data in the process of user interaction and the construction of the system, which consists of two parts, the front-end is mainly user interaction and visual display, and the back-end is data processing and model training.

Based on the task analysis and system design requirements outlined in Chapter 3, we present an interactive visual analysis process tailored to regression tasks in supervised learning. This process generates an effective stacking method to address specific problems, implementing the visual analysis system through interactive user engagement.

<span id="page-3-0"></span>

Figure 3: Interactive visual analysis system workflow, flow for Stacking method consists of a complete generation process of the system, including the selection of datasets, the selection of Stacking method performance evaluation metrics, the selection of the number of Stacking layers, as well as the input characteristics and the selection of specific base and meta models in the case of the selected number of layers.

# 4.1 Overview And System Frame

As highlighted in the task analysis, the system aims to find appropriate Stacking solutions, provide explanations for its decision-making process, and help users reduce computing resources and workload. To achieve this, we propose a framework that integrates front-end and back-end technologies, resulting in the interactive visual analysis system, InterpretStack.

Firstly, to address the need for different datasets to use various evaluation metrics, we allow users to select evaluation metrics and measure the performance of the final Stacking.

Secondly, to enhance interpretability and performance, we incorporate Catboost feature importance score[\[23\]](#page-8-24) to help users understand the dataset initially. To ensure the diversity of base models and reduce redundant features in meta-model inputs, we use the Pearson correlation coefficient[\[6\]](#page-8-25) to measure base model correlations. Additionally, we include SHAP[\[17\]](#page-8-26) values to better explain the Stacking decision-making process, providing insights into feature contributions for both base and meta models.

Furthermore, to improve Stacking performance, users often choose complex base models or Stacking structures, increasing resource consumption and workload. To mitigate this, we introduce Bayesian optimization to optimize the posterior distribution of the Gaussian process, reducing the number of hyperparameter explorations and user workload while ensuring model performance.

Finally, in the base model selection, we provide Lasso, Ridge, Random Forest, SVM, and XGBoost, where Lasso1 and Lasso2 refer to the top models in hyperparameter exploration. For the meta model, we offer Average, Linear, Lasso, and Ridge.

Combined with the above methods, we design the system framework shown in Fig. [Figure 2.](#page-2-0) The front-end handles the visual

display of data and provides interactive user operations, while the back-end manages data processing and model training. Data transmission between the front-end and back-end is implemented through Flask and Axios.

# 4.2 Workflow

InpretStack (Fig. [Figure 1\)](#page-1-0) contains seven sub-views: Dataset and Performance Metrics Manager (T4), Features(T1 and T3), Base Model (T1 and T3), Meta Model(T1), Stacking Components (T1, T2 and T3), Evaluation(T5), History Stacking(T6). The workflow of InterpretStack can be seen in Fig. [Figure 3.](#page-3-0) The workflow consists of two parts, namely preliminary Stacking design and iterative update of Stacking.

In the subsequent subsection, to help users understand, we introduce the system InterpretStack with the Urban Traffic[\[25\]](#page-8-27) data set, which comes from the UCI Machine Learning Repository[\[3\]](#page-8-28), tasked with predicting Slowness in traffic, contains 18 variables and 135 instances.

# 4.3 DataSet And Performance Metrics Manager

In order to solve the design of stacking methods for regression task in supervised learning in different domains, we design a DataSet And Performance Metrics Manager view, as seen in Fig. [Figure 1](#page-1-0) a, which can provide data set selection. Users can select the data set that has been designed by the system or upload the data set, but when uploading the data set, ensure that the original data has been completely processed and can be directly input into the model. The view also provides users with a choice of evaluation metrics, users can customize the performance evaluation metrics of the Stacking method according to their needs, for each evaluation

metric there is a slider bar through which the user can control the level of participation of this indicator in the final evaluation scheme. Among the evaluation indicators are: MSE, RMSE, MAE,  $R^2$ , MAPE, MSPE.

# 4.4 Features

The Features view (Fig. [Figure 1-](#page-1-0)b) presents feature importance scores using a bar chart generated by the CatBoost method. Users can select features by clicking small circles under each column chart, with a gray ring indicating selection. The 'SelectAll' button enables quick selection of all features, and clicking 'Commit' submits the chosen features. This facilitates informed feature selection for model training in subsequent layers, aiding in finding suitable solutions for the Stacking method.

# 4.5 Base Model

The Base Model view (Fig. [Figure 1-](#page-1-0)c) displays a correlation matrix of base model predictions using Pearson correlation coefficients. Users can filter base models by clicking on feature names and select all features with the 'SelectAll' button. This view aids in understanding base model dependencies and selection, crucial for optimizing the stacking method.

# 4.6 Meta Model

In different scenarios and stacking designs, diverse ensemble methods or meta models are necessary to enhance stacking performance. For instance, complex stacking structures may benefit from simpler meta models to boost performance. Determining the appropriate meta model requires extensive experimentation.

The Meta Model view (Fig[.Figure 1-](#page-1-0)e) displays the selected meta model for that layer. The backend initiates training for each selected model as a meta model for stacking, calculating performance on the validation set, presented in Fig. [Figure 1-](#page-1-0)g. This view streamlines integration strategy or meta model selection, addressing Task Requirements.

# 4.7 Stacking Components

The Stacking Components view (Fig[.Figure 1-](#page-1-0)e) is pivotal in the system, encompassing two key processes: preliminary stacking structure design and training, and subsequent stacking structure update and retraining. Users start by selecting the number of layers using the Layer Selector. They then choose features for the input layer in the Features view and submit them for base model training. Once base models are selected, they proceed to the Meta Model view to select the desired meta model. The output layer of the Stacking Components view displays the selected meta model, base models, and input features for the first selected meta model.Users can further explore and update the Stacking design by clicking on each base model in the Stacking Components view to observe feature contributions. They can adjust the base models, features, and meta models accordingly. Updates are reflected in the Evaluation View, where users can compare the performance of the updated models with previous results.

This iterative process empowers users to experiment with Stacking designs, improving its performance while providing interpretability and interactivity in the decision-making process.



Figure 4: Stacking Components view, which contains the models and features selected for each layer of Stacking, and the feature contributions of each model.

# 4.8 Evaluation View



#### Figure 5: Stacking

The Evaluation View (Fig. [Figure 1-](#page-1-0)f) facilitates base model backtracking and performance evaluation comparison. It displays the performance evaluation results of all base model training sets and validation sets from the previous layer of the output layer in the Stacking Components view. Users can visualize these results in histograms to identify overfitting or underfitting issues and make necessary adjustments to improve model performance. After updating the base model, users can track its performance through the Evaluation View, aiding in refining the model to better fit the dataset.

# 4.9 History Stacking

The History Stacking view (Fig. [Figure 1-](#page-1-0)g) shows the record of the model performance evaluation results of each designed Stacking method.

We use a line chart to compare the model performance evaluation results of stacking for each design. The abscissa represents the stacking of the last update, and the ordinate represents the model performance evaluation results of the stacking on the validation set. In this view, there are two types of polylines compared: one represents the Stacking composed of different meta models, used to compare their roles, and the other represents the performance of the best-performing base model in each stacking, evaluating whether the meta model or integration strategy has improved the overall predictive ability.

# 5 CASE STUDY

This section will verify the system, while verifying the explorability, interactivity and effectiveness of the system, while explaining and verifying the function and effect of the system from the perspective of the entire generation process of Stacking and the final Stacking performance at the end of exploration, as well as the interpretability of the stacking decision-making process.

<span id="page-5-0"></span>

Name	Type	Description
X	numeric	X-axis spatial coordinates of for-
		est areas (on hectares)
$\overline{Y}$	numeric	Y-axis spatial coordinates of for-
		est areas (on hectares)
month	discrete	Record the month of the fire
		("jan" to "dec")
day	discrete	Record the day of the week
		("mon" to "sun")
<b>FFMC</b>	numeric	Combustion hazard rating, ex-
		pressed as FM10
<b>DMC</b>	numeric	Radiator dryness, expressed in
		10 months
DC	numeric	Dryness, expressed in 10
		months
<b>ISI</b>	numeric	Initial expansion index
temp	numeric	Temperature, in degrees Celsius
RH	numeric	Relative humidity, expressed as
		a percentage
wind	numeric	Wind speed, kilometers per sec-
		ond
rain	numeric	Rainfall, measured in millime-
		ters per square meter
area	numeric	The area of a forest fire, mea-
		sured in hectares

Table 1: Forest Fires data set variable description

# 5.1 Introduction to Data Set

The dataset used in this study originates from the UCI Machine Learning Repository [\[3\]](#page-8-28), specifically the Forest Fires dataset [\[21\]](#page-8-29). It documents forest fire occurrences in residential areas of northern Portugal between 1998 and 2017. The dataset comprises 517 samples and includes 13 feature variables, detailed in Table [Table 1.](#page-5-0) The predictor variable for this regression task is the area affected by the forest fires. During data preprocessing, ordinal encoding was applied to the categorical variables "month" and "day".

# 5.2 Preliminary Stacking construction

In the DataSet And Performance Metrics Manager (Fig[.Figure 6](#page-6-0) a), we selected MAE as the primary performance metric for the Forest Fires dataset and submitted it for Catboost model importance scoring. Features were comprehensively assessed using Shap values and Catboost scores to refine subsequent model updates. Initial base model correlations are illustrated in Fig[.Figure 6-](#page-6-0)b. The SVM model was excluded initially due to its inefficiency with large datasets, necessitating substantial time and computing resources for hyperparameter exploration. Post-submission, Shap values were reviewed to finalize Stacking design across individual base model training phases (Fig. [Figure 6-](#page-6-0)e).

From Fig. [Figure 6-](#page-6-0)b, under the Forest Fires dataset, temperature (temp) emerged as the most significant feature based on Catboost model importance scores, contrasting with rain, which scored lower (Fig. [Figure 6-](#page-6-0)c). Notably, base models within the same category exhibited high similarity, whereas differences were evident between models like LASSO and Ridge, reflecting varied dataset insights (Fig. [Figure 6-](#page-6-0)f). Among different base model types, LASSO and Ridge demonstrated superior fit to the training set, with Random Forest (rf) and XGBoost (xgb) models following closely in prediction performance (Fig. [Figure 6-](#page-6-0)g).

# 5.3 Stacking further explores and updates

To enhance the Stacking method's performance, we conducted additional model adjustments. Initially, from Fig[.Figure 6-](#page-6-0)e, it was evident that the input features of the linear model were contributing minimally, suggesting overfitting. Consequently, all input features were removed and the model was retrained, resulting in improved Stacking performance, surpassing even the best performing base model (Fig[.Figure 6-](#page-6-0)b).

Further, we selected base model features based on Shap values to eliminate those with minimal contribution, as depicted in Fig. [Fig](#page-6-1)[ure 7-](#page-6-1)c. This optimization significantly enhanced each base model's performance on the validation set while maintaining consistency with the training set, indicative of improved predictive capability.

Simultaneously, integrating these refined base model predictions as features into the linear meta model showed no significant change in Stacking method performance (Fig[.Figure 7-](#page-6-1)c), despite enhancing the best performing base model's performance (baseBase). This indicated redundancy in the meta model's input information. Analysis from Stacking Components (Fig[.Figure 7-](#page-6-1)c) revealed high Shap contributions from lasso1 and lasso2 models, contrasting with minimal contributions from ridge1 and ridge2 models. Given the similarity between lasso and ridge models, and the distinctiveness introduced by Random Forest and XGBoost algorithms, it was inferred that redundant information from lasso and ridge predictions was affecting meta model performance. Therefore, lasso2, ridge1, and ridge2 models were removed, and the linear model was retrained (Fig. [Fig](#page-6-1)[ure 7-](#page-6-1)d). This adjustment yielded a more balanced contribution from each base model, resulting in significantly improved Stacking performance surpassing that of the best performing individual base model.

<span id="page-6-0"></span>

<span id="page-6-2"></span>Figure 6: Forest Fires dataset preliminary stacking design results.

<span id="page-6-1"></span>

Figure 7: Stacking further explores and updates for Forest Fires dataset. Including the updated Stacking structure design and evaluation results of the base model features and meta model features.

# 5.4 Decision understanding of this stacking design

From the Fig. [Figure 7-](#page-6-1)d, it can be found that the final Stacking middle layer consists of five base models, namely lasso1, rf1, rf2, xgb1 and xgb2, and the meta model is a linear regression model , in which the feature selection of each base model is shown in Fig. [Figure 8.](#page-6-2) Among them, the feature selection and the feature contribution of rf1, rf2, xgb1 and xgb2 models is similar, and the lasso1 model is



slightly different, according to the base model feature contribution and the contribution of the base model prediction results to the meta model, it can be concluded that for the task of predicting forest fire area, in this Stacking method, temperature (temp), radiator dryness (DMC), combustion hazard level (FFMC), wind speed (wind), dryness (DC) , relative humidity (RH), X-axis spatial coordinates (X) of forest area, preliminary deployment index (ISI) and month of fire occurrence all contributed to the final model effect, among which temperature (temp) and radiator dryness (DC) contributed the most to the model, while preliminary deployment index (ISI)

contributed relatively high to the Stacking prediction performance despite the low importance score of Catboost characteristics.

# 6 USER STUDY

To verify the interactivity, fluency, and effectiveness of the system, and its ability to solve T1, T2, and T3, we conducted a user study of InterpretStack.

# 6.1 Study Overview

We recruited 16 participants for the study, which consisted of four key phases: an initial questionnaire, system introduction, independent system usage, and a follow-up questionnaire.

Initial Questionnaire: Participants were asked about their familiarity with ensemble learning and stacking:

- Q1. Awareness of the Stacking method (Yes/No).
- Q2. Previous use of Stacking to enhance model performance (Yes/No).
- Q3. Difficulty in optimizing Stacking for better performance (Rated from 1 to 10, higher scores indicating greater difficulty).
- Q4. Understanding of Stacking's decision-making process (Rated from 1 to 10, higher scores indicating better comprehension).

System Introduction: Participants received a detailed online introduction to InterpretStack, covering its functionalities and usage process.

Independent System Usage: Participants explored the system to optimize Stacking performance and understand its decisionmaking process.

Follow-up Questionnaire: Participants assessed various aspects of InterpretStack:

- Q1. System interactivity.
- Q2. System fluency.
- Q3. System effectiveness.
- Q4. Level of confusion during system exploration (1-10 scale).
- Q5. Ability to enhance Stacking performance.
- Q6. Efficacy in facilitating understanding of Stacking.
- Q7. Capacity to reduce computing resources and workload.
- Q8. Suggestions for improvement.

### 6.2 Results and Suggestions

From the initial questionnaire (Q1 and Q2), 15 of 16 participants were familiar with stacking, and 7 had used it to enhance model performance. This suggests participants have prior experience with stacking and seek to leverage it for improved performance. As indicated in Fig. [Figure 9,](#page-7-0) participants found the system's workflow straightforward.

Users generally found the system interactive and fluent but suggested it could be more effective. They recommended actionable insights beyond Shap values to optimize stacking without requiring extensive knowledge of datasets and methodologies.

Initially, many participants found stacking challenging to understand and optimize. However, after using InterpretStack, they

<span id="page-7-0"></span>

Figure 9: The results of the user study. The left side shows the questionnaire results before using the system, and the right side shows the questionnaire results after using the system.

appreciated its ability to reduce workload, enhance stacking performance, and clarify stacking decision-making processes.

Despite overall approval, some participants suggested adding more tooltips for guidance and exploring ways to expedite hyperparameter searches.

# 7 DISCUSSION

Interactive machine learning enhances algorithm accuracy and robustness by merging human intuition with machine learning. InterpretStack exemplifies this by aiding users in iteratively refining Stacking methods and understanding the process. However, it has some limitations worth exploring.

Firstly, while InterpretStack helps with feature and model selection for better performance and interpretability, future improvements could include sample-level information, such as scatter plots of model predictions and Shap value contributions for individual samples. Additionally, expanding to accommodate classification tasks could increase its utility.

Secondly, the system's design could be improved with more flexible interaction options, like drag-and-drop for Stacking structure design.

Lastly, large datasets challenge system efficiency due to timeintensive hyperparameter tuning. Pre-training models or incorporating parallel computing could enhance feedback speed and user experience.

# 8 CONCLUSION

We introduced InterpretStack, an interactive visual analysis system for optimizing stacking methods in machine learning. It helps users reduce workload and computing resources while enhancing model performance and understanding. Users can choose evaluation metrics, select input features, and base models using Catboost feature importance scores and Pearson correlation coefficients. They can design stacking layers and compare the performance of base and meta models. Iteration for improved performance is guided by Shap values and other metrics. Additionally, the stacking method's decision-making process is clarified through model inputs and Shap values. Finally, we verify the system's interactivity, fluency, and effectiveness through case studies and user evaluations.

## ACKNOWLEDGMENTS

This work is supported by Natural Science Foundation of China (NSFC No.62202105) and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0103).

#### **REFERENCES**

- <span id="page-8-12"></span>[1] Rokia Lamrani Alaoui and El Habib Nfaoui. 2022. Web attacks detection using stacked generalization ensemble for LSTMs and word embedding. Procedia Computer Science 215 (2022), 687–696.<https://doi.org/10.1016/j.procs.2022.12.070> 4th International Conference on Innovative Data Communication Technology and Application.
- <span id="page-8-19"></span>[2] Saleema Amershi, Max Chickering, Steven M Drucker, Bongshin Lee, Patrice Simard, and Jina Suh. 2015. Modeltracker: Redesigning performance analysis tools for machine learning. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. 337–346.
- <span id="page-8-28"></span>[3] Arthur Asuncion and David Newman. 2007. UCI machine learning repository.
- <span id="page-8-7"></span><span id="page-8-5"></span>[4] L Breiman. 1996. Bagging Predictors. Machine Learning 24, 2 (1996), 123–140. [5] Angelos Chatzimparmpas, Rafael M Martins, Kostiantyn Kucher, and Andreas Kerren. 2020. Stackgenvis: Alignment of data, algorithms, and models for stacking
- ensemble learning using performance metrics. IEEE Transactions on Visualization and Computer Graphics 27, 2 (2020), 1547–1557.
- <span id="page-8-25"></span>[6] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. Noise reduction in speech processing (2009), 1–4.
- <span id="page-8-0"></span>[7] Thomas G Dietterich et al. 2002. Ensemble learning. The handbook of brain theory and neural networks 2, 1 (2002), 110–125.
- <span id="page-8-6"></span>[8] Yoav Freund, Robert Schapire, and Naoki Abe. 1999. A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 14, 771-780 (1999), 1612.
- <span id="page-8-4"></span>[9] Kevin Gurney. 2018. An introduction to neural networks. CRC press.
- <span id="page-8-9"></span>[10] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2011. Transforming Auto-Encoders. In Artificial Neural Networks and Machine Learning – ICANN 2011, Timo Honkela, Włodzisław Duch, Mark Girolami, and Samuel Kaski (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 44–51.
- <span id="page-8-14"></span>[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural computation 9, 8 (1997), 1735–1780.
- <span id="page-8-20"></span>[12] Fred Hohman, Minsuk Kahng, Robert Pienta, and Duen Horng Chau. 2018. Visual analytics in deep learning: An interrogative survey for the next frontiers. IEEE transactions on visualization and computer graphics 25, 8 (2018), 2674–2693.
- <span id="page-8-17"></span>[13] Minsuk Kahng, Nikhil Thorat, Duen Horng Chau, Fernanda B Viégas, and Martin Wattenberg. 2018. Gan lab: Understanding complex deep generative models<br>using interactive visual experimentation. IEEE transactions on visualization and computer graphics 25, 1 (2018), 310–320.
- <span id="page-8-3"></span>[14] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. 2002. Logistic regression. Springer.
- <span id="page-8-13"></span>[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradientbased learning applied to document recognition. Proc. IEEE 86, 11 (1998), 2278– 2324.
- <span id="page-8-16"></span>[16] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the loss landscape of neural nets. Advances in neural information processing systems 31 (2018).
- <span id="page-8-26"></span>[17] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. Advances in neural information processing systems 30 (2017).
- <span id="page-8-23"></span>[18] Christoph Molnar. 2020. Interpretable machine learning. Lulu. com.
- <span id="page-8-2"></span>[19] William S Noble. 2006. What is a support vector machine? Nature biotechnology 24, 12 (2006), 1565–1567.
- <span id="page-8-18"></span>[20] Heungseok Park, Yoonsoo Nam, Ji-Hoon Kim, and Jaegul Choo. 2020. Hyper-Tendril: Visual analytics for user-driven hyperparameter optimization of deep neural networks. IEEE Transactions on Visualization and Computer Graphics 27, 2 (2020), 1407–1416.
- <span id="page-8-29"></span>[21] Anbal Morais Paulo Cortez. 2008. Forest Fires. UCI Machine Learning Repository. <https://doi.org/10.24432/C5D88D>
- <span id="page-8-8"></span>[22] Ilya Ploshchik, Angelos Chatzimparmpas, and Andreas Kerren. 2023. MetaStack-Vis: Visually-Assisted Performance Evaluation of Metamodels. In 2023 IEEE 16th Pacific Visualization Symposium (PacificVis). 207–211. [https://doi.org/10.1109/](https://doi.org/10.1109/PacificVis56936.2023.00030) [PacificVis56936.2023.00030](https://doi.org/10.1109/PacificVis56936.2023.00030)
- <span id="page-8-24"></span>[23] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. Advances in neural information processing systems 31 (2018).
- <span id="page-8-22"></span>[24] Paulo E Rauber, Samuel G Fadel, Alexandre X Falcao, and Alexandru C Telea. 2016. Visualizing the hidden activity of artificial neural networks. IEEE transactions on visualization and computer graphics 23, 1 (2016), 101–110.
- <span id="page-8-27"></span>[25] Renato Sassi Ricardo Ferreira, Andrea Martiniano. 2018. Behavior of the urban traffic of the city of Sao Paulo in Brazil. UCI Machine Learning Repository. <https://doi.org/10.24432/C5902F>
- <span id="page-8-1"></span>[26] David H. Wolpert. 1992. Stacked generalization. Neural Networks 5, 2 (1992), 241–259. [https://doi.org/10.1016/S0893-6080\(05\)80023-1](https://doi.org/10.1016/S0893-6080(05)80023-1)
- <span id="page-8-21"></span>[27] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13. Springer, 818– 833.
- <span id="page-8-15"></span>[28] Tao Zhang and Jiaheng Li. 2021. Credit Risk Control Algorithm Based on Stacking Ensemble Learning. In 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA). 668–670. [https://doi.org/10.1109/ICPECA51329.](https://doi.org/10.1109/ICPECA51329.2021.9362514) [2021.9362514](https://doi.org/10.1109/ICPECA51329.2021.9362514)
- <span id="page-8-11"></span>[29] Jun Zhao, Qingguo Yan, Qinwei Dong, Xianguang Zha, Jun Wu, Zejia He, Xindong Zhao, and Xiaowen Zhang. 2022. Speech Instruction Recognition Method based on Stacking Ensemble Learning. In 2022 9th International Conference on Dependable Systems and Their Applications (DSA). 546–553. [https:](https://doi.org/10.1109/DSA56465.2022.00079) [//doi.org/10.1109/DSA56465.2022.00079](https://doi.org/10.1109/DSA56465.2022.00079)
- <span id="page-8-10"></span>[30] Zhi-Hua Zhou. 2012. Ensemble Methods: Foundations and Algorithms. CRC Press.