




Wenwen Gao · Shangsong Liu · Yi Zhou · Fengjie Wang · Feng Zhou · Min Zhu 

GBDT4CTRVis: visual analytics of gradient boosting decision tree for advertisement click-through rate prediction

Received: 30 November 2023 / Accepted: 2 March 2024
© The Visualization Society of Japan 2024

Abstract Gradient boosting decision tree (GBDT) is a mainstream model for advertisement click-through rate (CTR) prediction. Since the complex working mechanism of GBDT, advertising analysts often fail to analyze the decision-making and the iterative evolution process of a large number of decision trees, as well as to understand the impact of different features on the prediction results, which makes the model tuning quite challenging. To address these challenges, we propose a visual analytics system, GBDT4CTRVis, which helps advertising analysts understand the working mechanism of GBDT and facilitate model tuning through intuitive and interactive views. Specifically, we propose instance-level views to hierarchically explore the prediction results of advertising data, feature-level views to analyze the importance of features and their correlations from various perspectives, and model-level views to investigate the structure of representative decision trees and the temporal evolution of information gain during model prediction. We also provide multi-view interactions and panel control for flexible exploration. Finally, we evaluate GBDT4CTRVis through three case studies and expert evaluations. Feedback from experts indicated the usefulness and effectiveness of GBDT4CTRVis in helping to understand the model mechanism and tune the model.

Keywords Click-through rate prediction · Gradient boosting decision tree · Model tuning · Visual analytics

1 Introduction

Online advertising has been the major source of revenue for Internet businesses. In order to reduce advertisement (ad) costs and increase ad revenue, advertising analysts usually build click-through rate

W. Gao · S. Liu · Y. Zhou · F. Wang · F. Zhou · M. Zhu (✉)
College of Computer Science, Sichuan University, Yihuan Road, Chengdu 610065, Sichuan, China
E-mail: zhumin@scu.edu.cn

W. Gao
E-mail: gaowenwen@stu.scu.edu.cn

S. Liu
E-mail: piny@stu.scu.edu.cn

Y. Zhou
E-mail: zhouyi2@stu.scu.edu.cn

F. Wang
E-mail: wangfengjie@stu.scu.edu.cn

F. Zhou
E-mail: zhoulfeng@stu.scu.edu.cn

Published online: 29 March 2024

(CTR) prediction models to predict the probability of ads being clicked by users. The ads with higher probabilities are presented to users first, thus making the advertising more accurately exposed to target users.

Gradient boosting decision tree (GBDT) (He et al. 2014; Ling et al. 2017; Qiu et al. 2018; Wang et al. 2019) is a widely used CTR prediction model. While tuning the GBDT-based CTR prediction model, advertising analysts need to comprehensively analyze the model's prediction results, the features (i.e., the data attributes in the advertising CTR prediction dataset) involved in training, and the structural information of the model to understand the decision mechanism and further determine suitable hyperparameters. They often use several data analysis tools such as jupyter notebook (Rule et al. 2018; Wang et al. 2023) to complete the above tasks through extensive coding and trial-and-error experiments, which are time-consuming and cumbersome. In addition, due to the complex working mechanism of GBDT, advertising analysts always have difficulty analyzing the decision-making and the iterative evolution process of a large number of decision trees, and understanding the impact of different features on the model's prediction results.

Visual analytics have been demonstrated to be helpful in understanding the model mechanism and facilitating the process of model tuning (Yuan et al. 2021, 2023; Yang et al. 2022; Li et al. 2022), which generally involves the analysis of input data features and model structure. In terms of features, the analysis is mainly about the importance. For example, INFUSE (Krause et al. 2014) designs a glyph to display the contribution of features to the prediction model. RFSeer (Zhang 2021) helps users leverage features to understand the prediction results of random forest. However, these studies have not analyzed the correlations among features, and they fail to cover the need to distinguish the three main types of features in the field of online advertising: ad, medium, and user. Concerning the model structure, visualization studies of tree models related to our work, such as iForest (Zhao et al. 2019) and BOOSTVis (Liu et al. 2018), can reveal the static structure of decision trees. But it is difficult for analysts to explore the dynamic evolution process of tree ensemble models, such as changes in information gain.

To address these issues, we propose GBDT4CTRVis (Fig. 1), a visual analytics system that can help advertising analysts understand the working mechanism of GBDT and facilitate the process of model tuning. First, we construct a GBDT-based CTR prediction model and collect the log data generated during the training process. Second, we map the prediction results of data instances (i.e., a piece of advertising data) to a data overview view, and combine a Data Statistics View and a data list view to explore the advertising data instances hierarchically. Then, we use a list combined with the dual-axis plot to display feature importance, feature distribution, and the impact of feature values on prediction results. We use a node-link chart to show the correlations among ad, medium, and user features. Next, we cluster decision trees and use icicle charts to display the structure of representative decision trees and use a streamgraph to express the temporal evolution of information gain during the model prediction process. We also provide multi-view interactions and panel control for flexible exploration. With this system, advertising analysts can quickly obtain insights into model tuning and perform validation. Finally, we verify the effectiveness and usefulness of GBDT4CTRVis through three case studies and expert evaluations.

In summary, our contributions include:

- We summarize the analytical requirements of advertising analysts in tuning the GBDT-based CTR prediction model, and propose a multilevel visual analytics pipeline to provide a visual analytics solution for GBDT model tuning.
- We propose a set of novel visualization methods that combine the three main participants (ad, medium and user) in online advertising campaigns to reveal the working mechanism of the GBDT-based CTR prediction model at three levels: instance, feature, and model.
- We implement GBDT4CTRVis, an interactive visualization system integrating the above methods, to help advertising analysts understand the working mechanism of the GBDT-based CTR prediction model and streamline the tuning process. We demonstrate the usefulness and effectiveness of GBDT4CTRVis through three case studies and expert evaluations.

2 Related work

In this section, we survey the most relevant papers to our work: model interpretation visualization. We organize the relevant literature into two parts based on different dimensions of interpretation, namely feature



Fig. 1 The user interface of GBDT4CTRVis. GBDT4CTRVis assists advertising analysts in understanding the working mechanism of the GBDT-based CTR prediction model on multiple levels and facilitating the model tuning process. (1) Instance level: explore the prediction results of advertising data instances hierarchically through data overview (B), data statistics (C), and data details (D). (2) Feature level: analyze the importance of features (E) and the correlations (F). (3) Model level: display the most representative K decision trees (G), observe the evolution of the tree size (H), and present the evolution of the information gain during the model prediction (I). Analysts can also adjust hyperparameters and evaluate prediction performance through the control panel (A)

and model structure. These parts are feature-based model interpretation visualization and tree-structured model interpretation visualization.

2.1 Feature-based model interpretation visualization

Feature visualization aims to assist analysts in understanding the impact of different features. It can help analysts understand the importance of features and how models make decisions.

Regarding feature importance, INFUSE (Krause et al. 2014) performs feature selection by comparing different classification methods and designs a glyph to display the contribution of features to the model. Krause et al. (2016) and Hohman et al. (2019) investigate the effect of each feature on prediction results by changing the value of the feature. Additionally, there are methods such as LIME (Ribeiro et al. 2016) and SHAP (Lundberg and Lee 2017) that calculate feature importance from a local perspective.

In terms of understanding the model, Zhang (2021) helps users understand the prediction results of random forest from the perspective of features by analyzing feature importance, distribution of feature segmentation points, and feature statistics. Rauber et al. (2017) use t-SNE dimensionality reduction to downscale the high-dimensional feature maps of different layers of the network to a two-dimensional space for observation, which is used to understand and interpret multilayer perceptron and convolutional neural networks (CNNs). Liu et al. (2017) introduce a visual analytics system, CNNVis, which enables analysts to observe the most salient features learned by each type of neuron from images for the purpose of understanding CNN. Similar researches include RNNVis (Ming et al. 2017), DeepNLPVis (Li et al. 2022), etc.

From the above research, existing studies can analyze feature importance and help understand the working mechanism of the model through features. However, they have not explained the relationship between features and their impact on prediction results. The calculation of feature importance depends on the analyzed model and cannot be directly transferred to GBDT. Moreover, the online advertising domain has to distinguish between the features of three subjects: ad, medium, and user during feature analysis, yet existing studies have difficulty distinguishing the role of different categories of features in the model. Advertising analysts still need to use matplotlib in data analysis tools to customize the content they want, which may lead to issues such as data redundancy, visual clutter, and a lack of interactive features in the visualizations.

2.2 Tree-structured model interpretation visualization

Tree-based machine learning models include decision tree models and tree ensemble models. Decision tree models use a single tree to make decisions, while tree ensemble models are composed of multiple decision trees. Visualizing the structure of decision trees and tree ensemble models can help understand how the model makes decisions based on features and identify potential problems in the model.

The most common method for visualizing a single decision tree is intuitive node-link charts (Liu et al. 2018; Ware et al. 2001; Zhang et al. 2009; Wang et al. 2022). In addition, PaintingClass (Teoh and Ma 2003) uses a variant of the icicle chart to reveal local tree structures centered on nodes of interest. BaobabView (Elzen and Wijk 2011) uses a node-link tree with confusion matrices to support interactive construction and analysis of decision trees. Mühlbacher et al. (2018) use flow charts to represent the flow of samples on decision trees and embed pixel-level design in tree charts to help users evaluate the complexity and accuracy of decision trees. Jia et al. (2020) design a foldable tree visualization to display data flow through a proxy decision tree. Wang et al. (2022) develop an interactive tool based on the sunburst and node-link charts to display large decision trees.

For visualizing tree ensemble models, EnsembleMatrix (Talbot et al. 2009) provides interactive visualization of the confusion matrix that enabled users to compare different classifiers at a class level. Höferlin et al. (2012) visualize the class distribution of data at each stage of the Adaboosting model using a cascading scatter plot. Lee et al. (2016) propose an interactive visualization framework that visualizes the model structure and prediction statistics at each step of the gradient-boosting tree learning process for users. BOOSTVis (Liu et al. 2018) clusters multiple decision trees and supports the interactive exploration of representative decision trees with different structures. iForest (Zhao et al. 2019) provides information on all decision trees and reveals multiple decision paths to understand the generation process of final prediction results in random forests. RFSeer (Zhang 2021) uses sunburst charts as the main view to provide a multidimensional interpretation of the model structure of random forests.

The existing methods for visualizing decision trees and tree ensemble models are able to display the static structure and information of decision trees. However, they are difficult to explore the training progress of tree ensemble models and the evolution of prediction results. Therefore, they cannot meet the needs of advertising analysts to analyze the iterative evolution process of GBDT.

3 System design

In this section, we describe the design requirements, system overview, data preprocessing, model construction, and analysis and computation.

3.1 Design requirements

GBDT4CTRVIS is designed for advertising analysts. To summarize the key pain points of advertising analysts in understanding the mechanism of GBDT and model tuning, we have thorough discussions with advertising analysts (E1–E3) and their close collaborator (E4) from a large internet company.

E1–E3 work in the computational advertising industry, their main responsibility is to combine machine learning technology with the advertising business to continuously optimize the effectiveness of the models for more accurate and efficient ad delivery. Specifically, they aim to gain insights into the working mechanism of GBDT to facilitate model tuning or understand the driving factors behind advertising CTR. E4 is an advertising product manager, who works closely with E1–E3. He needs to understand the results of the model constructed by the advertising analysts, devise a deployment strategy, and ensure that the advertising campaign achieves the expected performance (i.e., business metrics) at a lower cost.

Here, we summarize three analysis requirements as follows to guide the design of our system.

R1: Explore the model's prediction results at the instance level. Advertising analysts need to evaluate the model's performance by combining the original data instances and the model's prediction results to discover whether there are biases or errors in the model's predictions. They need to observe the model's applicability range from the data instances. They also need to identify the differences and similarities between instances to provide a basis for further customization of advertising delivery.

R2: Analyze model decision-making basis at the feature level. Advertising analysts need to identify the importance of features and discover correlations among them, which can help avoid inputting redundant

features into the model and optimize the feature engineering. They also need to distinguish the features of the three categories in advertising campaigns: ad, medium, and user. Understanding the role of different types of features in the model helps to comprehend the decision-making basis, and internal operational mechanism of the model and determine appropriate model hyperparameters.

R3: Understand the model's decision-making mechanism at the model level. GBDT is an ensemble learning model that iteratively trains multiple decision trees and combines them for prediction. In each iteration, GBDT trains a new decision tree based on the residuals of the previous round's model to approach the true label value continuously. Understanding the decision-making mechanism of GBDT in each iteration, such as how to select features for node splitting, the evolution of information gain, and so on, can optimize and improve the model in a targeted manner.

3.2 System overview

According to the design requirements in Sect. 3.1, we develop an interactive visual analytics system called GBDT4CTRVis to assist advertising analysts in understanding how the GBDT-based CTR prediction model works and simplifying the process of model tuning. The system pipeline is shown in Fig. 2 and consists of four main modules: (1) data preprocessing, (2) model construction, (3) analysis and computation, and (4) visualization and interaction.

The data preprocessing module performs data cleaning and down sampling on the advertising click-through rate prediction dataset, and then the processed data are used for training the GBDT model in the model construction module. The trained model is utilized in the analysis and computation module for further analysis, as well as in generating views for Data Statistics View, Data List View, and Feature Importance View. The data generated from model training in the model construction module is used by the analysis and computation module for decision tree clustering, feature importance and correlations calculations, dimensionality reduction. The results of decision tree clustering and the decision tree data obtained in model construction (*tree_info*) are jointly used in drawing three views for model-level views. The results of feature importance and correlations calculations are used to draw Feature Importance View and Feature Correlation View. The dimensionality reduction coordinates of the samples obtained are used in the Data Overview View. In the visualization and interaction module, users can adjust the model's hyperparameters through a control panel to obtain the model's re-training results and update the data and views. They can explore the model's predictive performance from the perspectives of instance, feature, and model, obtain insights for improving the model, and validate it.

3.3 Data preprocessing

In this paper, we use the advertising CTR prediction dataset publicly available from the Huawei 2020 DIGIX algorithm competition,¹ recommended by the domain experts we collaborate with. The dataset contains 7 days of advertising delivery and clicks data, with approximately 41.91 million records, and each record has 36 fields.

Among the 36 fields, one is the label for advertising click behavior (with values of 1 or 0, representing whether the user clicked on the corresponding ad), and the remaining 35 fields can be divided into three categories of features: ad, medium, and user. These features include basic information about the displayed ad (such as the advertiser, industry, ad material, etc.), user information (such as demographic attributes, device name, etc.), and media information (such as application platform, advertising slot, etc.). In addition, the dataset has undergone discretization and desensitization processing through label encoding, where all feature values are numeric labels. To make better use of the original data in subsequent analysis, we divide the features into unordered discrete categorical data or ordered discrete numerical data based on their specific meanings.

Before constructing the GBDT model, we first perform the following data processing steps: (1) Data cleaning: removing duplicate data from the original dataset, filling in missing values and invalid data; (2) Data sampling: the original dataset contains a total of 41,907,133 samples, of which only 1,445,488 are positive samples (*label* = 1) for clicking on ads, while the number of negative samples is about 28 times that of the positive samples. Imbalanced sample categories can easily lead to model bias (Japkowicz and Stephen 2002), i.e., bias toward the category with more samples, and difficult to identify categories with fewer

Par100 <https://www.kaggle.com/datasets/louischen7/2020-digix-advertisement-ctr-prediction>.

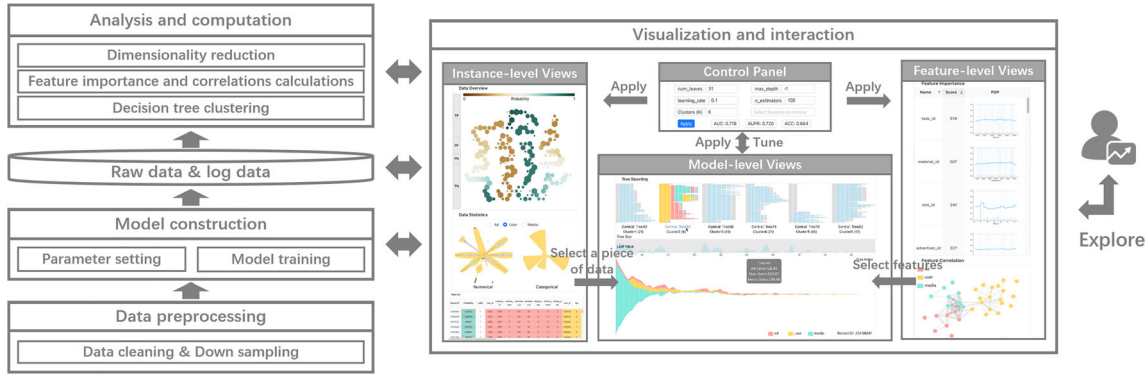


Fig. 2 Pipeline of GBDT4CTRVis. There are four modules included in the pipeline: data preprocessing, model construction, analysis and computation, and visualization and interaction. Our system supports the understanding of the working mechanism of the GBDT-based CTR prediction model from three levels: instance, feature, and model. Rich interactions are also provided to facilitate model tuning

samples. To balance the positive and negative samples, we adopt a downsampling strategy (Japkowicz and Stephen 2002), randomly selecting an equal number of negative samples (a total of 2,890,976 samples) to participate in subsequent data partitioning; (3) Dataset partitioning: all positive samples and an equal number of sampled negative samples are randomly divided into 10 parts, with 7 parts used as the training set, 1 part as the validation set, and 2 parts as the test set. The preprocessed and split dataset will be used for training the GBDT model, conducting further analysis in the analysis and computation module, and generating visualizations in the data statistics view, data list view, and feature importance view.

3.4 Model construction

The core idea of the GBDT is to combine multiple decision trees in a sequential manner, so that each new tree can correct the errors of the previous trees, thereby improving the overall predictive performance. The implementation process of the GBDT is as follows:

Inputs: (a) Training dataset: $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$, where N represents the number of samples. For a sample (x_i, y_i) , $x_i \in \mathbb{R}^d$ represents the set of features sized d and y_i represents the corresponding label, $i = 1, 2, \dots, N$. (b) Loss function: $L(y_i, f(x_i)) = \frac{1}{2}(y_i - f(x_i))^2$. (c) Number of decision trees: M . (d) Limits on the depth and number of leaf nodes for each tree. (e) Learning rate: $\mu \in [0, 1]$.

Output: The ensemble of decision trees $f_M(x)$, where the m -th tree is denoted as $T_m(x)$ and the ensemble model at m -th iteration is denoted as $f_m(x)$, $m = 1, 2, \dots, M$.

Steps:

S1: Initialize the model. As formulated in $f_0(x) = \arg \min_c \sum_{i=1}^N L(y_i, c)$, a constant value c is estimated by minimizing the loss function to initialize a tree with only the root node.

Subsequently, by repeating the following Steps 2–4, new decision trees are continuously constructed, and the ensemble model is updated. Each iteration attempts to reduce the residuals that were not fitted by the model in previous iterations.

S2: Compute residuals. For each sample, calculate the residuals $r_{mi} = -\frac{\partial L(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)} = (f(x_i) - y_i)$, $i = 1, 2, \dots, N$. The residuals measure the difference between the current model's predictions and the labels.

S3: Iteratively build decision trees. Taking r_{mi} as the new labels and replacing each training sample by (x_i, r_{mi}) , a new decision tree $T_m(x)$ can be constructed. The construction process includes the following substeps: (a) *Feature selection.* At each node, the best splitting feature is selected by calculating the information gain of each feature, which is used to measure the importance of the features. (b) *Splitting nodes.* Based on the selected best splitting features, the samples at the nodes are assigned to the left and right child nodes. (c) *Termination conditions.* According to the limiting conditions such as the maximum depth of decision trees, the maximum number of leaf nodes, determining whether to continue splitting the child nodes or stop splitting to form leaf nodes.

S4: Update the model. The newly constructed decision tree is combined with the current model using weighted aggregation, resulting in a more powerful model $f_m(x) = f_{m-1}(x) + \mu T_m(x)$, where μ is the learning rate to dampen the influence of the new tree. This allows the updated model to better fit the residuals that were not captured by the previous one.

S5: Obtain the final model. Repeat the iterations until reaching predetermined stopping conditions, such as reaching the maximum number of iterations or achieving convergence of residuals. At that point, the final model $f_M(x)$ is obtained.

We implement the GBDT model by LightGBM (Ke et al. 2017), which is an efficient implementation version of GBDT. Meanwhile, we enable users to adjust four hyperparameters of the model: the maximum number of leaf nodes of each decision tree (*num_leaves*, set to 31), the maximum depth of each decision tree (*max_depth*, set to -1, indicating no limit), the number of decision tree M (*n_estimators*, set to 100), and the learning rate μ (*learning_rate*, set to 0.1). The values of these hyperparameters are the built-in defaults in LightGBM.

During the training process of GBDT, a large number of decision trees are generated, which contain a wealth of information about the decision-making process. For subsequent visual analysis, GBDT4CTRVis utilizes built-in methods, i.e., *lightgbm.Booster.dump_model*, to export the logged data of the trained model, including feature importance (*feature_importances*) and decision tree data (*tree_info*). The importance score of each feature is calculated as the number of times that feature is selected as the splitting feature in all decision trees. The more frequently a feature is chosen for splitting, the greater its involvement in the decision-making process, and thus, the more important. The decision tree data includes the node connection structure of each tree, the predicted values of the leaf nodes, as well as the splitting feature, information gain, and sample count for each node. These data and the model's prediction results will be further processed in the analysis and computation module.

3.5 Analysis and computation

In this module, we use the raw data and log data generated from data preprocessing and model construction to perform decision tree clustering, feature importance and correlations calculations, and dimensionality reduction.

3.5.1 Decision tree clustering

Zhang–Shasha algorithm (Zhang and Shasha 1989) is used to measure the similarity of tree structures based on tree edit distance in this paper, which is one of the commonly used methods for measuring tree structural similarity. It is simpler and more practical compared to the Tree Kernel method, which requires designing a good kernel function. Specifically, based on the node connection structure of each decision tree collected in the previous section, the tree edit distance (the minimum number of editing operations required to transform one tree into another) is calculated using the dynamic programming approach (Bille 2005) to quantify the distance between decision trees, resulting in a tree distance matrix. Subsequently, the more flexible and adaptable K-Medoids algorithm (Park and Jun 2009) is applied to this matrix to cluster the trees, allowing users to customize the number of cluster centroids K based on the granularity of observations (default is 6 for preliminary display). The cluster centroids represent the representative decision trees. The results of the clustering, along with the decision tree data obtained in the model construction (*tree_info*), are jointly used for drawing three views in the model-level views.

3.5.2 Feature importance and correlations calculations

The importance score of each feature can be obtained from the training log data of GBDT, which represents the number of times the feature is selected as the splitting feature in all decision trees. The importance scores of features are used to create the Feature Importance View. In addition, we use Spearman's correlation coefficient (Spearman 1961) to measure the correlation between the features. Because Spearman's correlation coefficient is non-parametric, it does not require the assumption that the data is normally distributed, does not consider specific numerical values, and is not sensitive to outliers. It can measure the correlation between variables that do not have a linear relationship. Compared to other correlation measures such as Pearson, the Spearman correlation coefficient is more suitable for the characteristics of the advertising CTR prediction dataset. The computation of the Spearman coefficient takes the training data samples as input and outputs a correlation coefficient matrix, which describes the degree of correlation between different features in the advertising CTR prediction dataset and is used to create the Feature Correlation View.

3.5.3 Dimensionality reduction

To facilitate the exploration of data instances by users, we employ a flexible nonlinear dimension reduction algorithm called Uniform Manifold Approximation and Projection (UMAP) (McInnes et al. 2018) to reduce the dimension of the data samples. We choose UMAP as the dimension reduction method because the advertising CTR prediction dataset typically contains numerous nonlinear relationships and complex interactions. Linear methods, such as PCA, assume linear relationships between data points and fail to capture the nonlinear structures in the dataset effectively. Additionally, compared to the nonlinear t-SNE (Maaten and Hinton 2008) dimension reduction method, UMAP can handle large-scale datasets more efficiently while maintaining the quality of dimension reduction. UMAP has two main steps: learning the manifold structure of the high-dimensional space and finding the corresponding low-dimensional representation. When learning the manifold structure of the high-dimensional space, the number of neighboring points can be specified by adjusting the UMAP hyperparameter $n_neighbors$. When finding the corresponding low-dimensional representation, UMAP provides another hyperparameter min_dist to specify the minimum distance between sample points in the low-dimensional space. Experimental results show that the default values for both parameters yield good results. Therefore, we use the default values: $n_neighbors = 15$ and $min_dist = 0.1$. The data samples are reduced to a two-dimensional plane space by UMAP, and the resulting coordinate data is used in the subsequent visualization method of the hexagonal binning chart.

4 Visualization

We design a novel interactive visual analytics system, GBDT4CTRVIS, to implement the above analysis requirements. GBDT4CTRVIS can help advertising analysts understand the working mechanism of the GBDT-based CTR prediction model through the instance, feature, and model level and facilitate the tuning process. As shown in Fig. 1, GBDT4CTRVIS contains three levels of views: (1) Instance-level views, which use a hexagonal binning chart, two rose charts, and a table-based chart to explore advertising data instances from the perspectives of data overview, data statistics, and data details respectively (R1); (2) Feature-level views, which use a list combined with dual-axis plot to display feature importance and use a node-link chart to show intra-class and inter-class correlations among three types of features: ad, medium, and user (R2); (3) Model-level views, which use icicle charts to display the structure of representative decision trees and use streamgraph to express the temporal evolution of information gain in the model prediction process (R3). GBDT4CTRVIS also provides a control panel that allows users to adjust hyperparameters and validate their insights.

4.1 Instance-level views

Instance-level views provide an overview of the prediction results and the instance clusters. By analyzing the distribution of feature values and detailed information of the instance cluster, instance-level views help to analyze the prediction results of the model, construct user profiles for different clusters, and further provide insights for model tuning at the data level (R1). Instance-level views include data overview view (Fig. 1B), data statistics view (Fig. 1C), and data list view (Fig. 1D).

Data overview view (Fig. 1B) integrates the design of the confusion matrix and hexagonal binning chart, providing users with an overview of the prediction results of data instances. It consists of a vertical confusion matrix, a color legend, and a hexagonal binning chart. As shown in Fig. 3a, *the confusion matrix* consists of four rectangles arranged vertically from top to bottom, representing four classification results: true positive (TP), false positive (FP), false negative (FN), and true negative (TN). The height of each rectangle encodes the number of samples included in each type of classification result, with a longer rectangle indicating more samples. *The color legend* specifies the color encoding for the hexagonal binning chart, using two colors and their gradients to display the prediction scores of samples, where the prediction score of 0 is represented by dark brown, 1 is represented by dark green, and scores between 0 and 1 are linearly mapped using a gradient color between these two colors. *The hexagonal binning chart* uses hexagons to aggregate similar data samples that fall within its boundaries after dimensionality reduction by UMAP. Hexagons located closely together representing data samples have similar features and prediction scores. The area of the hexagon encodes the number of samples, with a larger area indicating more samples.

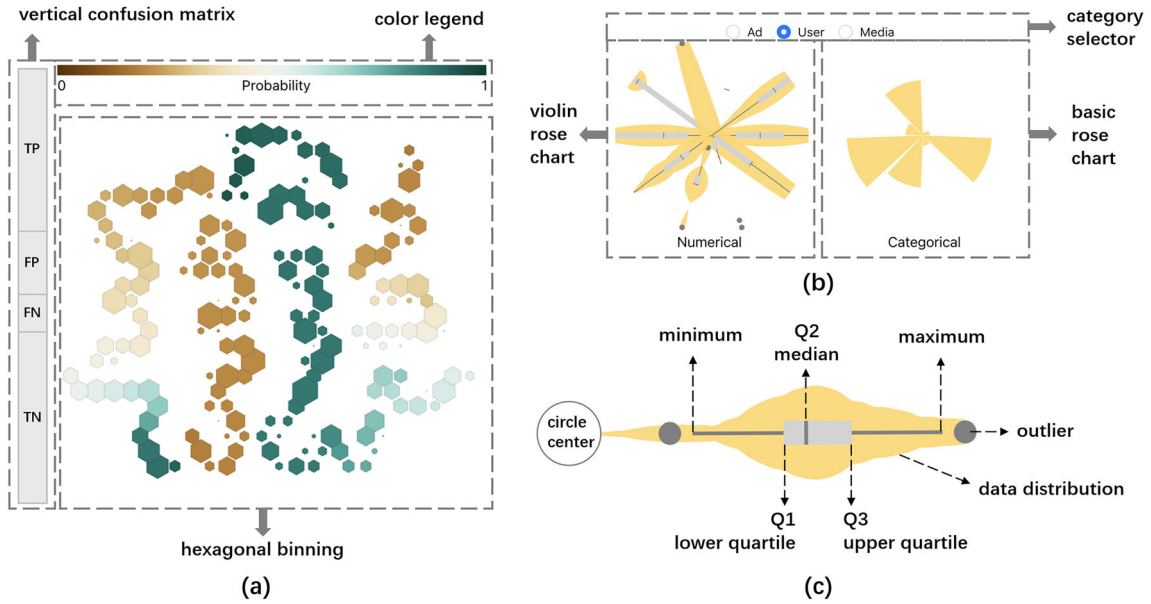


Fig. 3 Design details of data overview view and data statistics view

Detailed information about the data within each hexagon can be viewed through the Data List View (Fig. 1D) for further analysis of the dimension reduction results.

Design alternatives. Two alternative designs are considered for displaying the dimensionality reduction results of the advertisement data. The first design directly uses scatter plot, but due to the large amount of data, many points overlap and cannot be displayed, losing density information, and putting pressure on the browser’s rendering. Experts expressed their desire to obtain more information about the occluded data. The second design uses triangular or square binning to aggregate the data, aggregating the scatter plot into a coarser graphical representation to encode more visual information and alleviate browser pressure. However, research (Carr et al. 1986) has shown that hexagonal binning is more effective in aggregating data. Sensitivity issues introduced by the presentation of aggregated data (such as the division of points on hexagon boundaries) can be addressed by examining detailed information in the Data List View. In addition, domain experts have suggested that the sensitivity issues introduced by data aggregation are negligible in the task of overviewing the prediction results of clusters of instances compared to the occluded data in the scatter plot. Therefore, we visualize the dimensionality reduction results of the data based on hexagonal binning.

Data statistics view (Figs. 1C and 3b) uses two rose charts to display statistical information on numerical and categorical features. It consists of a category selector, a violin rose chart, and a basic rose chart. *The category selector* is used to switch the types of features to be presented in the rose charts, including three categories of features: ad, medium, and user, corresponding to the background colors of pink, yellow, and cyan in the rose charts. *The violin rose chart* is used to display statistical information on numerical features, consisting of several violin plots in a radial layout. As shown in Fig. 3c, each violin plot integrates the design of the box plot and density plot, creating a symmetrical graphical representation. In the box plot, the side of the gray rectangle near the circle center represents the lower quartile $Q1$, the side far from the circle center represents the upper quartile $Q3$, and the line segment in the middle of the rectangle represents the median $Q2$. The line segments on the symmetrical axis represent the minimum and maximum values within the normal data range, with the closest end to the center indicating the minimum value and the furthest end representing the maximum value. We use the interquartile range (IQR), which is the range between the first quartile ($Q1$) and the third quartile ($Q3$), to define the normal range of the data. Specifically, the normal range is defined as $[Q1 - 1.5 * IQR, Q3 + 1.5 * IQR]$. Data points outside the normal data range are considered outliers and represented by gray dots. In the density plot, the curvature encodes the shape of the data distribution, with higher curvature indicating denser data. *The basic rose chart* is used to show statistical information on categorical features, where each petal represents a categorical feature, and the sector’s radius encodes the frequency of the most frequently occurring value in that categorical feature.

Design alternatives. In the design iteration process, we first use box plots to present the distribution of data and information about outliers. However, our experts point out that box plots display statistical values such as the median, quartiles, etc., but they cannot discern the specific shape of the data distribution (such as symmetry, kurtosis, etc.). In addition, for categorical data, they are more interested in the values that occur most frequently for each feature, as these values can help them quickly match large number of users. Therefore, we have designed different visualization schemes for numerical and categorical features. For numerical features, we combine the design of box plots and density plots; for categorical data, we use a basic rose chart to present the most frequent values for each feature.

Data list view (Fig. 1D) presents detailed information on raw data in a table format. As shown in Fig. 1D, each row in the table represents one sample in the dataset, with the first three columns representing its ID, prediction score, and actual label, respectively, and the following columns representing features. The background color of the second column represents the prediction score of the sample, using the same color mapping as in Data Overview View. The background color of the feature columns represents the category of the feature, using the same color mapping as in Data Statistics View with the rose diagrams.

Interaction: Analysts can click on the hexagons or confusion matrix of interest in (Fig. 1B) to update the data source for the Data Statistics View (Fig. 1C) and Data List View (Fig. 1D). Then, analysts can analyze the user profile of the cluster, identify outliers in the cluster data through Fig. 1C, and view detailed information through Fig. 1D. When a data record in Fig. 1D is clicked, GBDT4CTRVis updates the streamgraph described in Sect. 4.3 for further analysis. We also reduce the visual burden and improve space utilization by employing techniques such as highlighting, tooltip, zooming, dragging (panning), filtering, etc.

4.2 Feature-level views

Feature-level views are designed to help users analyze the importance ranking of all features, the impact of different values of a single feature on prediction results, as well as the intra-class and inter-class correlations between features, thereby providing insights for model tuning at the feature level (R2). Feature-level views include Feature Importance View (Fig. 1E) and Feature Correlation View (Fig. 1F).

Feature importance view (Fig. 1E) presents the importance ranking of features and the impact of individual features on prediction results, based on a list combined with dual-axis plot. As shown in Fig. 1E, this view consists of a scrollable list, with each column representing the feature name, its importance score, and a partial dependence plot (PDP) in a dual-axis plot. Each row represents a feature, sorted by default from highest to lowest feature importance. In the PDP, the lines represent the impact of the feature value on the model prediction results. The dark blue line represents the overall impact of the feature on the prediction results (across all samples), while a light blue line represents the prediction of a single sample on that feature. The gray rectangle represents the distribution of the feature, with categorical features represented by discrete bar charts and numerical features represented by continuous histograms. The “dual-axis plot” refers to the combination of the line chart with the bar chart or histogram.

Design alternatives. For the design iterations, we first use the traditional PDP to analyze the overall impact of different feature values on model predictions, represented by the dark blue lines in the graph. However, domain experts have reported that they also want to observe the prediction of single samples on each feature and the distribution of feature values, to avoid misinterpretation caused by overemphasizing regions with very little data in the PDP. To address this, we overlay light blue lines on the PDP to represent the prediction of single samples on each feature value. Additionally, we incorporate gray histograms and bar charts respectively for numerical and categorical features to meet the experts’ requirements.

Feature correlation view (Fig. 1F) uses a node-link chart to explore the intra-class and inter-class correlations between features, consisting of a legend in the upper left corner and a node-link network on the right side. As shown in Fig. 1F, in the node-link network, each circle represents a feature, and the line between the two circles represents the correlation between the corresponding features. The width of the line encodes the strength of the correlation. And the larger the correlation, the thicker the line. When hovering over the corresponding element, GBDT4CTRVis displays the corresponding tooltip. The rectangle in the upper left corner indicates the color legend. The circles and rectangles corresponding to three types of features: ad, medium, and user, are also respectively colored pink, yellow, and cyan.

Design alternatives. We have considered optimizing the traditional feature correlation matrix heatmap to analyze feature correlations, but this method produces redundant data (upper and lower symmetric triangle matrices) due to the large number of features, which not only wastes space but also increases visual burden.

Although only displaying one of the symmetrical triangles can solve this problem, it still cannot intuitively compare the within-class and between-class feature correlations. After discussing with experts, we use node-link chart to address these issues.

Interaction: Feature Importance View displays all features by default, sorted by feature importance from highest to lowest score. Analysts can filter the displayed features by clicking the button to the right of “Name” and change the sorting rules by clicking the button to the right of “Score”. When clicking a feature represented by a row in the Feature Importance View, GBDT4CTRVis highlights the corresponding rectangle of the node split by that feature in the Icicle View described in Sect. 4.3. Meanwhile, GBDT4CTRVis magnifies the corresponding feature nodes in the Feature Correlation View, and pops up a tooltip showing the feature name for user analysis.

4.3 Model-level views

Model-level views assist users in analyzing the structure of a single representative decision tree, observing the overall structural evolution of the decision trees, and exploring the evolution of the information gain of an instance in the model prediction process, to better understand the model’s decision process, and further provide insights for model tuning at the model level (R3). Model-level views include Icicle View (Fig. 1G), Area View (Fig. 1H), and Streamgraph View (Fig. 1I).

Icicle view (Fig. 1G) displays the most representative K decision trees obtained using the method described in Sect. 3.5.1. As shown in Fig. 1G, K horizontally arranged icicle charts are displayed from left to right to represent the structures of the K decision trees. The two lines of text below each icicle chart indicate the index number of the cluster center and the number of trees contained in that cluster, respectively. In the icicle chart, each rectangle represents a node in the tree structure, with the leftmost rectangle representing the root node and the rightmost rectangles representing leaf nodes. The area of a rectangle encodes the number of samples passing through that node during model training, with a larger area indicating a greater number of samples. Icicle View has both normal mode and highlight mode. In normal mode, the rectangles representing the root and leaf nodes are coded in gray, while those representing the internal nodes are coded in blue. In highlight mode, the color of each rectangle corresponding to a non-leaf node is determined by the splitting feature of that node, with ad, medium, and user features being colored pink, yellow, and cyan, respectively.

Design alternatives. Two alternative designs are considered to visualize the structure of the decision tree. The first alternative design is the node-link chart, but it suffers from layout difficulties and poor readability when dealing with large decision trees (e.g., deep and wide trees). The second alternative design is the rectangular tree map, but it exhibits a low space utilization rate due to the presence of significant blank areas when visualizing unbalanced decision trees. Additionally, in a smaller display space, the layout lines between complex rectangular tree map nodes may intersect and overlap, making the chart difficult to comprehend. Therefore, we opted for the concise, user-friendly, clearly hierarchical, and space-efficient icicle chart to visualize the structure of the decision tree.

Area view (Fig. 1H) is located below the Icicle View and is used to observe the evolution of tree size. The horizontal axis represents the sequence number of all decision trees, and the vertical axis represents the size of each tree, i.e., the number of all nodes, with the area colored gray by default. When a user selects a certain instance, GBDT4CTRVis superimposes a layer of blue area chart on the Area View. The horizontal axis likewise represents the sequence number of all decision trees, while the vertical axis represents the values of the leaf nodes reached by the sample through each decision tree, with positive values located above the horizontal axis and negative values located below.

Streamgraph view (Fig. 1I) is used to present the evolution of the information gain during the model prediction process. As shown in Fig. 1I, from left to right, the white dotted lines represent the 0-th to $(n - 1)$ -th decision trees generated in sequence during the training process of the prediction model, among which the dotted lines representing cluster centers are thicker and more opaque. When a user moves the mouse over a certain dotted line, GBDT4CTRVis highlights the line in black and pops up a tooltip with the sum of the information gain from different classes splitting features when the data sample passes through that decision tree. At the corresponding position of each tree, the overall width of the river band indicates the sum of the information gain of all splitting nodes (non-leaf nodes) on the decision path as the data sample passes through that tree, and the different categories of splitting features are colored with the corresponding colors as described previously.

Design alternatives. A streamgraph is a type of stacked area graph. During the design iteration process, we consider directly using a stacked area graph or ThemeRivers, which are similar to streamgraphs, to present the evolution of the information gain. However, when it comes to the task of comparing the evolution of information gain for different feature categories, streamgraphs are more readable than stacked area graphs or ThemeRivers (Thudt et al. 2016). Therefore, we ultimately use a streamgraph for visualization.

Interaction: GBDT4CTRVis provides highlight and tooltip interactions for each rectangle in the Icicle View, the text below the icicle charts, and each dotted line in the Streamgraph View. When a user moves the mouse over the first line of text label below, GBDT4CTRVis highlights the corresponding dotted line in the Streamgraph View at the appropriate position and a tooltip will be displayed. When a dotted line is clicked, GBDT4CTRVis highlights the decision path of the data sample in the Icicle View corresponding to that decision tree. When the mouse is moved over the second line of text label below, Streamgraph View marks all the positions of the decision trees in that cluster with black dotted lines.

4.4 Control panel

We provide a control panel to assist analysts in model tuning and evaluating model performance. As shown in Fig. 1A, analysts can adjust the maximum number of leaf nodes of the decision tree (*num_leaves*), maximum depth of the decision tree (*max_depth*), number of decision trees (*n_estimators*), learning rate (*learning_rate*), number of representative decision trees (*clusters_k*), and features participating in model training. After clicking the button “Apply”, analysts can check the adjusted model performance and the updated views. The control panel provides three commonly used evaluation metrics for CTR prediction models: AUC (area under the ROC curve), AUPR (area under the precision–recall curve), and ACC (accuracy) for analysts to refer to.

5 Evaluation

To investigate the usefulness and effectiveness of GBDT4CTRVis, we invited four experts who worked with us during the design process (E1–E4) mentioned in Sect. 3.1, two machine learning model researchers (E5 familiar with GBDT, E6 not familiar with GBDT), and four visual analytics researchers (E7–E10, among them, two individuals have experience in model visual analysis, while the other two have experience in the visual analysis of advertising data) to participate in the evaluation experiment. There are 5 males and 5 females, with an average age of 26.3, with an average domain experience of more than 3 years. They include industry experts, researchers, and graduate students. The entire evaluation process took an average of approximately 70 min.

The procedure for the study consisted of 4 steps. First, we spent 5 min briefly introducing the background and research content of our work to the experts. Second, we spent 15 min demonstrating the visual encoding and interaction of GBDT4CTRVis through an example. Next, experts spent 40 min freely exploring the data described in Sect. 3.3. We encouraged the experts to think aloud and say what they thought and did during their exploration. Finally, we invited the experts to complete a questionnaire to quantitatively evaluate the visual encoding, interaction design, and system functionality, and collect their feedback and suggestions.

We summarize three representative case studies found by our experts in Sect. 5.1 to fully illustrate how experts use GBDT4CTRVis to explore the GBDT-based CTR prediction model and tune the model. Then, we report the results of the questionnaires in Sect. 5.2.

5.1 Case study

5.1.1 Exploring prediction results of data instances

Exploring prediction results of data instances is a starting point for analyzing model performance and data characteristics. In this case, we describe how E1 uses GBDT4CTRVis combined with ad data instances to analyze the prediction results of the GBDT-based CTR prediction model and perform tuning (R1).

E1 started by analyzing the prediction results of the model with default hyperparameters (Fig. 1). He selected a hexagon with deeper color and larger area (Fig. 4a) in the Data Overview View (Fig. 1B) and

analyzed the feature statistics information of this data cluster in the Data Statistics View (Fig. 4b). From the basic rose chart on the right side of the user type, E1 found that all samples in this data cluster have a career value of 9 and a gender value of 3. In addition, these samples have a *net_type* and *purchase_tag* value of 2, and most samples have a city value of 287 (these values are coded for anonymization). E1 further checked and validated the specific feature values of each instance by dragging the horizontal slider to the feature of user type in Data List View (Fig. 4c), and found that the values of these features were indeed largely consistent. E1 explained that the model performs well in predicting data with these data features. He also indicated that the same ads can be pushed to other similar users based on the features of the ad type data corresponding to these users.

Furthermore, E1 clicked on the first instance (record “4227959”) to analyze its prediction result. As shown in Fig. 5a, E1 found that this sample could generate more information gain in the first 18 decision trees, and the subsequent decision trees corrected the deviation between the predicted value and the true value. However, this sample produced negative prediction scores between the 80-th and 90-th decision trees, but the evolution of information gain indicated that the deviation had not been fully corrected. E1 speculated that it might be due to insufficient basic learners. If there were more basic learners, perhaps more detailed information could be captured. E1 adjusted the number of basic learners (*n_estimators*) from the default 100–150 in the control panel (Fig. 5b). After checking the predicted values of the leaf nodes in the updated blue area chart, E1 found that the new decision trees did learn more detailed information. And the model’s evaluation metrics AUC increased from the original 0.718–0.722, AUPR increased from 0.720 to 0.724, and ACC increased from 0.664 to 0.666. In addition, E1 found that the fitting speed of predicted values to residuals was slow and suggested increasing the learning rate of the model. Then E1 verified his speculation through similar adjustment operations.

Overall, E1 was able to successfully combine ad data instances to analyze the prediction results of the GBDT-based CTR prediction model. He could understand the mechanism of the model well from the perspective of data, propose hypotheses for model tuning, and verify them through the control panel.

5.1.2 Analyzing and selecting features

Analyzing the importance of features in model prediction and the correlation between features can help analysts identify key factors that affect model performance, and select suitable features for model training. In this case, we describe how E2 used our system to analyze the role of ad data features in model prediction and select features for training (R2).

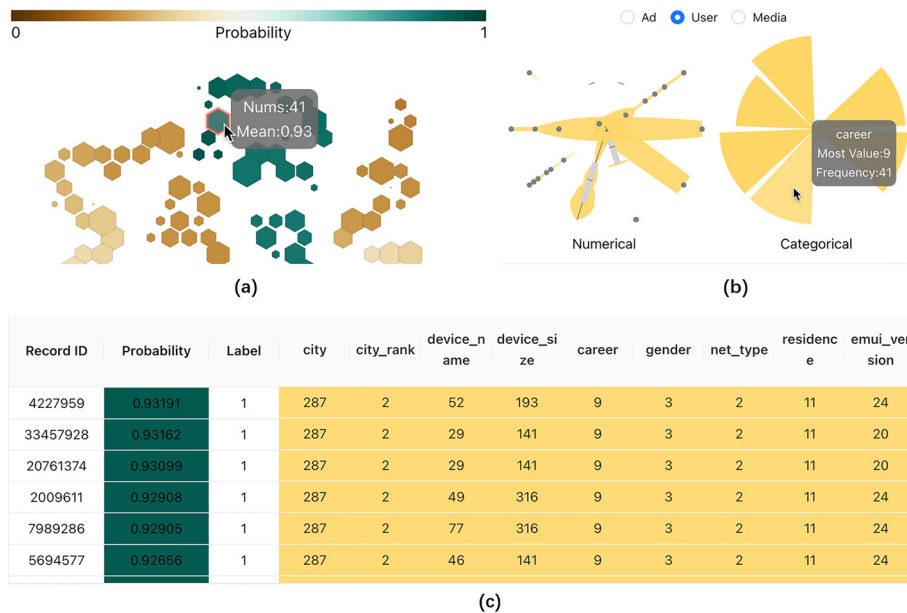


Fig. 4 Explore the prediction results of ad data instances hierarchically

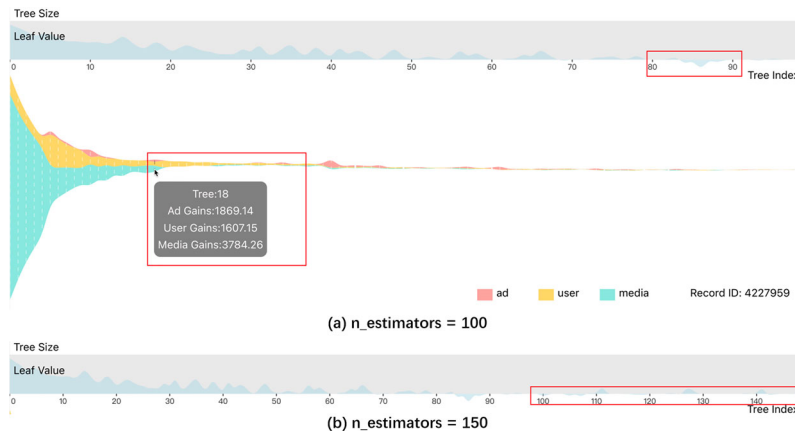


Fig. 5 Increase base learners to tune the model. **a** Analysis of leaf value and information gain evolution of an instance in prediction. **b** Area view after adjusting the hyperparameter $n_estimators$

E2 found from the descending ranking (Fig. 6) that $task_id$, $material_id$, and $slot_id$ had a significant impact on the model's prediction results. He clicked on these three features and found from the Icicle View (Fig. 7, where the rectangles corresponding to the features were highlighted) that these features were indeed used multiple times as the basis for decision tree node splitting. E2 also found from the Feature Correlation View (where the corresponding feature nodes were enlarged) that these features were distributed around the periphery of the view and had no obvious correlation with other features, indicating that these features were more distinctive.

E2 analyzed the densely distributed nodes in the center of the Feature Correlation View and found that the app_score feature of the media type was strongly correlated with multiple features (Fig. 8). He clicked on one of the links and found that the correlation coefficient between app_score and app_first_class was 1, indicating that they had a very strong correlation. E2 also noticed from the corresponding PDP that they had the same data distributions and impacts on the model's prediction results, indicating that they were redundant features. E2 speculated that eliminating one of the features may be able to improve the model performance.

To verify his hypothesis, E2 removed the feature app_score , which was strongly correlated with other features and had no contribution to the model's prediction results, from the control panel while keeping the other default model hyperparameters unchanged. He found that the model's performance improved slightly, with AUC increasing from 0.718 to 0.719, AUPR increasing from 0.720 to 0.721, and ACC remaining at 0.664.

In conclusion, E2 believed that GBDT4CTRVis provided an intuitive and effective view that could help him analyze the importance and correlation of features to some extent, thus gaining insights into feature selection.

5.1.3 Analyzing and tuning model structures

Analyzing the model structure can help to tune the model in a targeted manner. This case aims to describe how E3 used GBDT4CTRVis to diagnose and tune the model structure of the GBDT-based CTR prediction model (R3).

From the Icicle View (Fig. 1G) under default hyperparameters, E3 found that the overall model structure was relatively simple, and there was a large difference in the number of samples contained in the left and right nodes of most representative decision tree root nodes. E3 said this may cause the model to fail to fully learn the differences between samples, and he felt that the maximum number of leaves could be adjusted to increase the complexity and fitting ability of the model. To verify his hypothesis, E3 adjusted the maximum number of leaves (num_leaves) from the default 31–100 in the control panel and found that the evaluation indicators AUC, AUPR, and ACC of the model were all improved: AUC increased from the original 0.718–0.726, AUPR increased from 0.720 to 0.730, and ACC increased from 0.664 to 0.670. Then, E3 observed the structure of the model again and found a decision tree with a depth of 31 (Fig. 9). He said that too deep trees might cause overfitting of the model, reduce the speed of model operation, and lower the generalization ability of the model. E3 then set max_depth to 12 in the control panel and approximately waited for a few

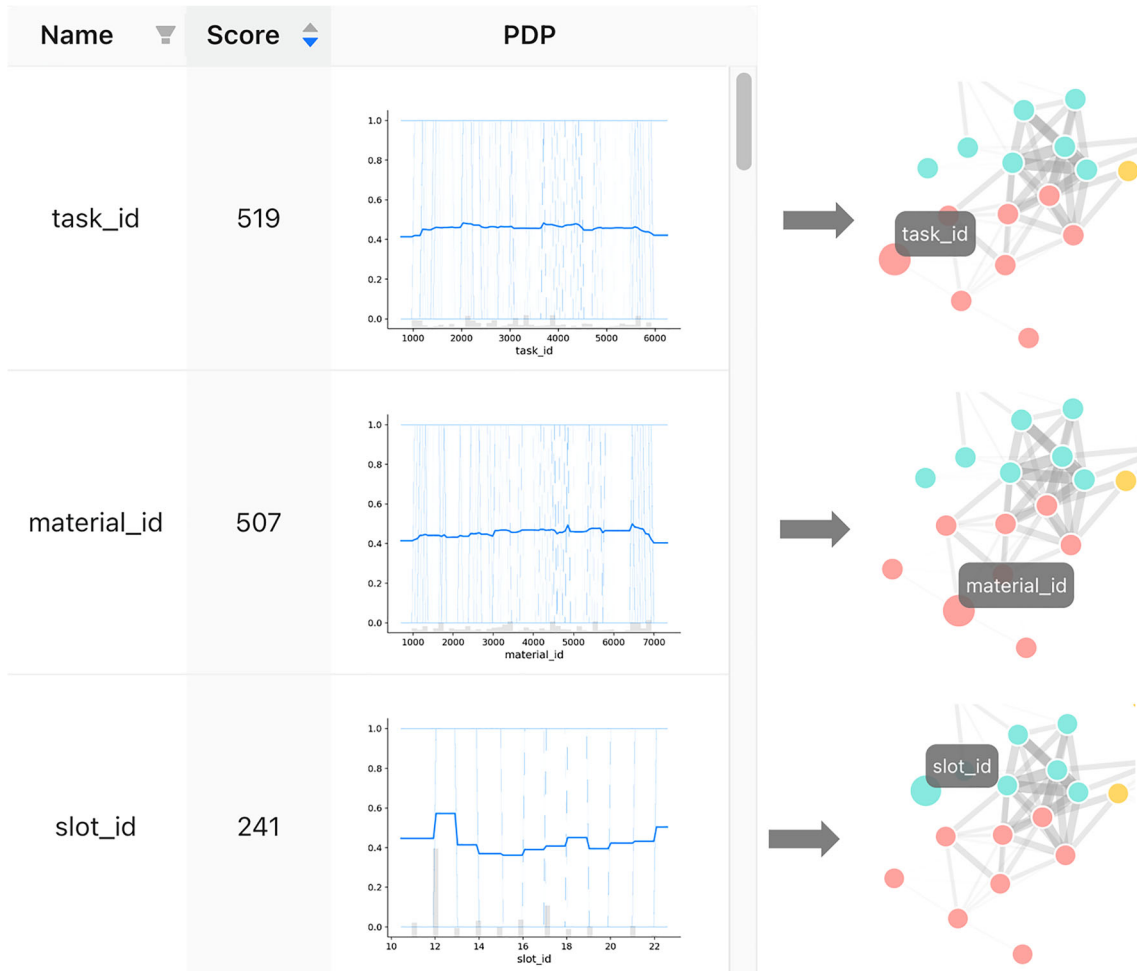


Fig. 6 Importance and correlation analysis of features



Fig. 7 Split node distribution of important features

tens of seconds (mainly waiting for the result of Icicle View). Then, he found that the model performance was consistent with before, but the depth of the representative decision tree did not exceed 12, and the distribution of nodes was more reasonable. E3 indicated that this hyperparameter combination simplified the structure of the base learner and improved computational efficiency while ensuring that the model evaluation indicators did not decrease.

After learning about the analysis content of E1 and E2, E3 set *num_leaves* to 100, *max_depth* to 12, *n_estimators* to 150, and *learning_rate* to 0.25 on the control panel, and removed redundant features. He found that the performance of the model was further improved, and the three evaluation indicators on the test set were higher than when the above hyperparameters worked alone. Specifically, AUC increased from 0.718 to 0.729, AUPR increased from 0.720 to 0.733, and ACC increased from 0.664 to 0.672. E3 said that in the field of ad CTR prediction with imbalanced positive and negative samples, a 0.1% improvement in

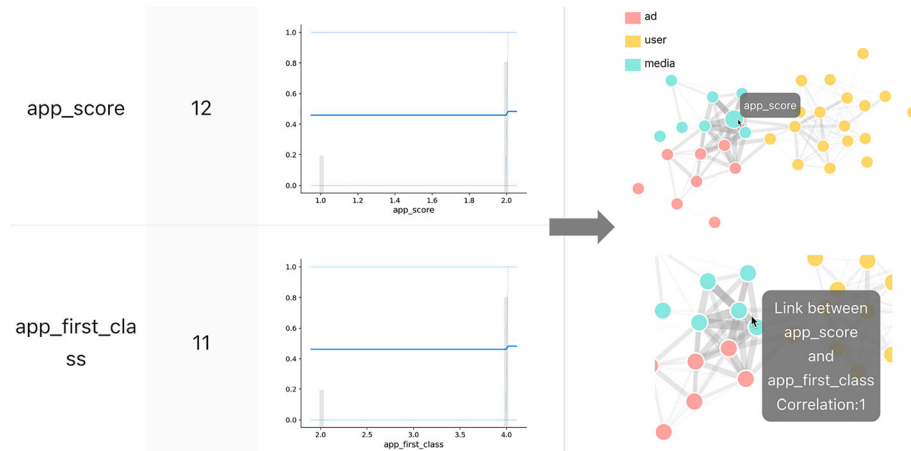


Fig. 8 Analysis of features with strong correlations

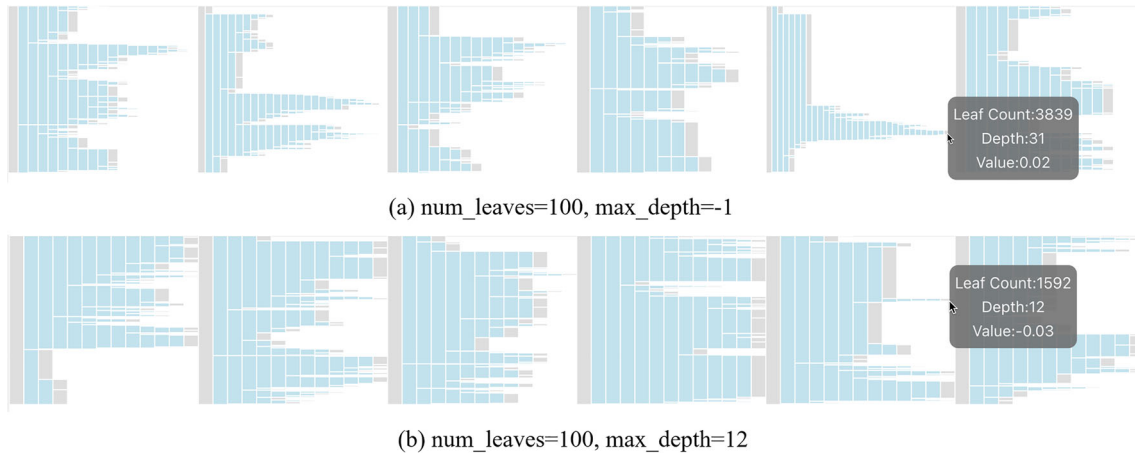


Fig. 9 Icicle view after adjusting the *num_leaves* and *max_depth*

these indicators could already bring enormous benefits. Existing research (Ling et al. 2017) has also demonstrated this point.

Therefore, E3 recognized the effectiveness of GBDT4CTRVIS in analyzing and tuning the GBDT-based CTR prediction model structure. He also appreciated our system design and analysis pipeline. He believed that GBDT4CTRVIS could indeed help advertising analysts understand the basic working mechanism of the GBDT-based CTR prediction model from the three levels of instance, feature, and model, and simplify the process of model tuning.

5.2 Expert evaluation

We referred to related work in the field of visual analytics (Jin et al. 2023; Zhang et al. 2023; Li et al. 2021; Zhao et al. 2022) and designed 9 questions based on the system’s usefulness and effectiveness, as shown in Table 1. Experts need to rate each question based on the 5-point Likert scale (1–5 represents “strongly disagree” to “strongly agree”).

The results of the questionnaire are shown in Fig. 10. Overall, GBDT4CTRVIS is perceived as being easy to learn (Q1), easy to understand (Q2), and easy to interact with (Q3), with average scores of 4.4, 4.1, and 4.2, respectively, and variances of 0.24, 0.49, and 0.36. In terms of system responsiveness (Q4), two participants gave a score of 2 because they felt that the response time of around a few tens of seconds exceeded their waiting threshold, while the average score for this question is 3.6, with a variance of 1.04. Participants found that GBDT4CTRVIS allows easy exploration of model predictions at the instance level (Q5) and model level (Q7), with average scores of 4.2 and 4.1, and variances of 0.36 and 0.49, respectively.

Table 1 Questionnaire of the expert evaluation

No.	Question
Q1	I think it's easy to learn the system
Q2	I think it's easy to understand the visual design of the system
Q3	I think it's easy to interact with the system
Q4	I am satisfied with the responsiveness of the system
Q5	I think it's easy to explore the model's prediction results at the instance level
Q6	I think it's easy to analyze the model decision-making basis at the feature level
Q7	I think it's easy to understand the model's decision-making mechanism at the model level
Q8	I think it's easy to perform model tuning through the system
Q9	I think this system is very helpful in improving the effectiveness of ad CTR prediction models

However, the average score for exploring model predictions at the feature level (Q6) is relatively lower, which is 3.9, with a variance of 0.49, as participants felt that the exploration at the feature level should provide more views that are closely related to the raw data, enabling them to gain more information. Additional, participants appreciated the system's role in model tuning (Q8) and improving ad CTR prediction model performance (Q9), with average scores of 4.4 and 4.1, and variances of 0.24 and 0.49, respectively.

We report the results of the questionnaire survey and expert evaluation below from the perspectives of usefulness and effectiveness.

Usefulness. Almost all experts expressed that they can easily learn and use this system ($Q1_mean = 4.4$, $Q3_mean = 4.2$), and the design of the views is intuitive and easy to understand ($Q2_mean = 4.1$). “The design of GBDT4CTRVis is very user-friendly, and I can quickly grasp the use of the system.” -E8. “The visualization is divided from three perspectives: the instance, model, and feature are easy to understand and remember, and the color matching of the system is also quite identifiable.” -E4. In terms of system response speed ($Q4_mean = 3.6$), two machine learning experts, E5 and E6, expressed that further optimization is needed because they frequently modify model hyperparameters through the control panel to observe the effects, but the response time for the system to return calculation results exceeds their waiting threshold. Mainly waiting for the results of Icicle View, it usually takes about a few tens of seconds. They said it was tolerable and noted that it could be improved with higher performance computing resources. On this issue, experts in the visualization field gave high scores because they adjust model hyperparameters less frequently and attach more importance to the interaction of the system. The statistical results of Q1–Q4 indicate that the vast majority of experts gave high evaluations of the system's usefulness.

Effectiveness. The vast majority of experts believed that they could effectively explore the model's prediction results at the instance level through the system ($Q5_mean = 4.2$). “GBDT4CTRVis met the need for analyzing the prediction results of data instances. I could quickly adjust model hyperparameters by analyzing these results,” said E1. However, expert E9 suggested that the visualization design of the rose chart could be further optimized, such as using an adaptive layout according to the number of features. Most experts believed that the system could analyze the model's decision basis at the feature level, including the importance, correlation, distribution of features, and the impact of feature values on prediction results ($Q6_mean = 3.9$). “I really liked the feature-level design because GBDT4CTRVis visualized all aspects of the features I cared about,” said E2. However, E7 mentioned that the effect of some partial dependence plots was not significant enough, and complementary experiments could be conducted on datasets with real

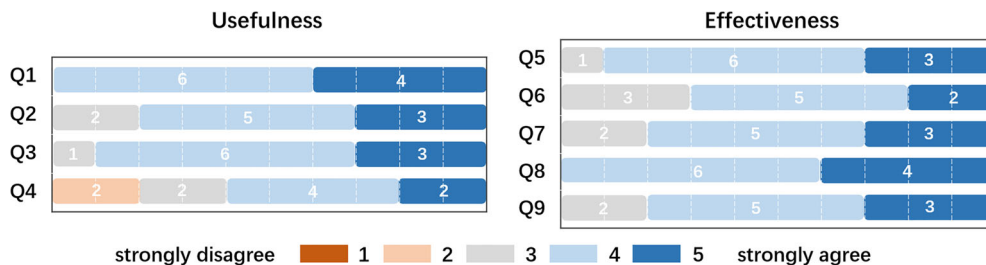


Fig. 10 Statistics of the questionnaire results, 1–5 represents “strongly disagree” to “strongly agree”. The white numbers represent the number of individuals who gave that score

continuous features. In terms of understanding the model’s decision mechanism at the model level, most experts believed that they could diagnose and tune the model through the system ($Q7_mean = 4.1$). “I found multiple ways to improve the model while exploring the system, such as tuning the tree structure based on the insights from the Icicle View,” said E3. “If there were other visualization views of the decision tree structure and node split, perhaps more effective insights could be obtained,” E2 suggested. E2 also suggested supplementing the prompt box in the river chart to specify which features produced information gain. All experts acknowledged the effectiveness of this system in model tuning ($Q8_mean = 4.4$). E1 and E3 both stated that GBDT4CTRVis allowed them to intuitively understand the model’s working mechanism, discover directions for model tuning, and verify their findings through the control panel. Expert E5 mentioned that existing hyperparameter optimization solutions such as grid search could be integrated into the control panel to provide more references. Experts believed that the system was helpful in improving the accuracy of ad CTR prediction ($Q9_mean = 4.1$), and E2 also hoped that the system could support the analysis of custom datasets.

In addition, we discussed the advantages and disadvantages of GBDT4CTRVis compared to the existing tool Jupyter Notebook with experts. They unanimously agreed that GBDT4CTRVis integrated a complete analysis pipeline, allowing them to focus on thinking and analysis. The visual design of GBDT4CTRVis also met their analytical needs well, but the analysis provided by GBDT4CTRVis was limited. On the other hand, Jupyter Notebook required repetitive code modifications to obtain the desired data, and the existing visualization tools used in Jupyter Notebook, such as Matplotlib, suffered from issues such as data redundancy, visual clutter, and lack of interactivity in generating views, which affected work efficiency. However, it allowed for wider range of data exploration through coding.

In general, experts recognized the usefulness and effectiveness of the prototype system, believing that the system was easy to operate and could effectively meet the requirements described in Sect. 3.1. Experts also made some constructive comments to guide our future work.

6 Discussion

In this section, we first discuss the lessons learned from our design research and then discuss the generalizability and limitation of GBDT4CTRVis.

6.1 Lessons learned

Working closely with domain experts throughout the entire process provides us with valuable insights in developing GBDT4CTRVis. Firstly, in the requirement analysis phase, it is crucial to communicate and collaborate iteratively with domain experts multiple times to summarize and optimize requirements, as they may not be able to describe their needs clearly at the beginning. Furthermore, in this process, visual analytics specialists need to proactively acquire necessary background knowledge of domain business or models. Also, involving intermediaries such as product managers who bridge the gap between technology and business can help eliminate the gap between visual analytics specialists and business requirements, facilitating communication between visual analytics specialists and target users. Secondly, as for visual design, considering domain experts’ cognitive preferences and usage habits is essential to improve user experience and efficiency. For example, in the Data Statistics View, we combine domain experts’ different analysis preferences for numerical and categorical data and design violin rose charts and basic rose charts, respectively, to visualize the corresponding data. Additionally, associating model interpretability research with domain data analysis can promote the practical application of model interpretability in the field. It enables domain experts to gain more contextual information in model tuning or other downstream tasks, such as assisted decision-making.

6.2 Generalizability

Although GBDT4CTRVis is used to understand the working mechanism of GBDT in the problem of advertisement CTR prediction and simplify the model tuning process, it can also be applied to other scenarios. First, our system can be used to analyze other GBDT-based binary classification tasks. For example, stock price trend prediction (predicting whether the stock price will rise or fall, to help investors formulate investment strategies), loan default prediction (predicting whether the borrower will default,

helping banks assess risk and formulate loan strategies), disease diagnosis (predicting whether a patient has a certain disease based on the patient's clinical indicators and signs), and attack detection (predicting whether network traffic contains malicious attack behavior to strengthen network security protection), and so on. The main work required is to process the original data into the format required by the system using the method described in Sect. 3. Secondly, we summarize a multi-level visual analytics pipeline(i.e., the three levels of instance, feature, and model), which can be transferred to other visual analytics studies for model tuning of machine learning models.

6.3 Limitation

Case studies and expert evaluations confirm the usefulness and effectiveness of GBDT4CTRVis, but there are still some limitations. (1) System response time: The time complexity of tree edit distance is high, which affects the response time of the Icicle View after system parameter adjustment. It is possible to consider using GPU to improve computational efficiency and optimize the tree edit distance algorithm to enhance user experience. (2) Model tuning efficiency: The current model tuning solutions are relatively simple. We can further improve the efficiency of model tuning by combining common hyperparameter optimization solutions such as grid search and adding other model hyperparameters such as splitting criterion. (3) Visual and interactive design: We can optimize the layout design of the rose chart, increase the significance of some partial dependence plots, provide a magnifying glass function for the river chart, provide an overview of the distribution of feature importance, and provide a view interface for analyzing each decision tree or a custom dataset analysis interface for users to enrich the functionality and user experience of the system. (4) Tree structure similarity measure: The tree edit distance only considers the labels and structures of trees when measuring the similarity between tree structures, ignoring the semantic information of the trees, which can more accurately measure the similarity between decision trees. It is possible to consider using Tree Kernel methods or other methods that can capture the semantic information of trees for optimization.

7 Conclusion and future work

We propose a visual analytics system, GBDT4CTRVis, which helps advertising analysts understand the working mechanism of the GBDT-based CTR prediction model from three levels: instance, feature, and model, thereby facilitating the process of model tuning. Based on the background of online advertising, this system combines the three main participants (ad, medium and user) in online advertising campaigns with the construction and tuning process of the CTR prediction model, allowing analysts to intuitively explore the ad data instances and their corresponding prediction results. Users can analyze the role of data features in model prediction from four aspects: the importance of features, the impact of feature values on prediction results, the distribution of feature values, and intra-class and inter-class correlations between features. Users can also understand the model's decision mechanism at the model level, analyze the model evolution, gain insight into model tuning, and validate it by controlling the panel. Our evaluation results show that GBDT4CTRVis can effectively help advertising analysts understand the working mechanism of the GBDT-based CTR prediction model and simplify the process of model tuning.

In future work, we plan to improve the response speed of the system by optimizing the tree edit distance algorithm and using GPUs for acceleration. We will also improve the system by providing hyperparameter optimization guidance and enriching the existing views to provide more functionality and more user-friendly interactions. Additionally, we will incorporate the semantic information of trees into the tree structure similarity measurement methods to more accurately measure the similarity between decision trees.

Acknowledgements We thank all the reviewers and our participants for their time and valuable input and appreciate Huawei's advertisement click-through rate prediction dataset, which is publicly available at the 2020 DIGIX algorithm competition.

References

- Bille P (2005) A survey on tree edit distance and related problems. *Theor Comput Sci* 337(1):217–239. <https://doi.org/10.1016/j.tcs.2004.12.030>
- Carr DB, Littlefield RJ, Nicholson WL (1986) Scatterplot matrix techniques for large n. In: Proceedings of the seventeenth symposium on the interface of computer sciences and statistics on computer science and statistics. Elsevier North-Holland, Inc., pp 297–306. <https://doi.org/10.5555/26036.26072>

- Elzen S, Wijk JJ (2011) Baobabview: interactive construction and analysis of decision trees. In: 2011 IEEE conference on visual analytics science and technology (VAST), pp 151–160. <https://doi.org/10.1109/VAST.2011.6102453>
- He X, Pan J, Jin O, Xu T, Liu B, Xu T, Shi Y, Atallah A, Herbrich R, Bowers S, Candela JQ (2014) Practical lessons from predicting clicks on ads at Facebook. In: Proceedings of the eighth international workshop on data mining for online advertising. ADKDD'14. Association for Computing Machinery, New York, pp 1–9. <https://doi.org/10.1145/2648584.2648589>
- Höferlin B, Netzel R, Höferlin M, Weiskopf D, Heidemann G (2012) Inter-active learning of ad-hoc classifiers for video visual analytics. In: 2012 IEEE conference on visual analytics science and technology (VAST), pp 23–32. <https://doi.org/10.1109/VAST.2012.6400492>
- Hohman F, Head A, Caruana R, DeLine R, Drucker SM (2019) Gamut: a design probe to understand how data scientists understand machine learning models. In: Proceedings of the 2019 CHI conference on human factors in computing systems. CHI '19. Association for Computing Machinery, New York, pp 1–13. <https://doi.org/10.1145/3290605.3300809>
- Japkowicz N, Stephen S (2002) The class imbalance problem: a systematic study. *Intell Data Anal* 6(5):429–449. <https://doi.org/10.5555/1293951.1293954>
- Jia S, Lin P, Li Z, Zhang J, Liu S (2020) Visualizing surrogate decision trees of convolutional neural networks. *J Vis* 23(1):141–156. <https://doi.org/10.1007/s12650-019-00607-z>
- Jin Z, Wang Y, Wang Q, Ming Y, Ma T, Qu H (2023) Gnnlens: a visual analytics approach for prediction error diagnosis of graph neural networks. *IEEE Trans Vis Comput Graph* 29(6):3024–3038. <https://doi.org/10.1109/TVCG.2022.3148107>
- Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y (2017) Lightgbm: a highly efficient gradient boosting decision tree. In: Proceedings of the 31st international conference on neural information processing systems. NIPS'17. Curran Associates Inc., Red Hook, pp 3149–3157. <https://doi.org/10.5555/3294996.3295074>
- Krause J, Perer A, Bertini E (2014) Infuse: interactive feature selection for predictive modeling of high dimensional data. *IEEE Trans Vis Comput Graph* 20(12):1614–1623. <https://doi.org/10.1109/TVCG.2014.2346482>
- Krause J, Perer A, Ng K (2016) Interacting with predictions: visual inspection of black-box machine learning models. In: Proceedings of the 2016 CHI conference on human factors in computing systems. CHI '16. Association for Computing Machinery, New York, pp 5686–5697. <https://doi.org/10.1145/2858036.2858529>
- Lee T, Johnson J, Cheng S (2016) An interactive machine learning framework. <https://doi.org/10.48550/arXiv.1610.05463>
- Li H, Xu M, Wang Y, Wei H, Qu H (2021) A visual analytics approach to facilitate the proctoring of online exams. In: Proceedings of the 2021 CHI conference on human factors in computing systems. CHI '21. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3411764.3445294>
- Li Z, Wang X, Yang W, Wu J, Zhang Z, Liu Z, Sun M, Zhang H, Liu S (2022) A unified understanding of deep NLP models for text classification. *IEEE Trans Vis Comput Graph* 28(12):4980–4994. <https://doi.org/10.1109/TVCG.2022.3184186>
- Ling X, Deng W, Gu C, Zhou H, Li C, Sun F (2017) Model ensemble for click prediction in bing search ads. In: Proceedings of the 26th international conference on world wide web companion. WWW '17 companion. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, pp 689–698. <https://doi.org/10.1145/3041021.3054192>
- Liu M, Shi J, Li Z, Li C, Zhu J, Liu S (2017) Towards better analysis of deep convolutional neural networks. *IEEE Trans Vis Comput Graph* 23(1):91–100. <https://doi.org/10.1109/TVCG.2016.2598831>
- Liu S, Xiao J, Liu J, Wang X, Wu J, Zhu J (2018) Visual diagnosis of tree boosting methods. *IEEE Trans Vis Comput Graph* 24(1):163–173. <https://doi.org/10.1109/TVCG.2017.2744378>
- Lundberg SM, Lee S-I (2017) A unified approach to interpreting model predictions. In: Proceedings of the 31st international conference on neural information processing systems. NIPS'17. Curran Associates Inc., Red Hook, pp 4768–4777. <https://doi.org/10.5555/3295222.3295230>
- Maaten L, Hinton G (2008) Visualizing data using t-SNE. *J Mach Learn Res* 9(86):2579–2605
- McInnes L, Healy J, Saul N, Großberger L (2018) Umap: uniform manifold approximation and projection. *J Open Source Softw* 3(29):861. <https://doi.org/10.21105/joss.00861>
- Ming Y, Cao S, Zhang R, Li Z, Chen Y, Song Y, Qu H (2017) Understanding hidden memories of recurrent neural networks. In: 2017 IEEE conference on visual analytics science and technology (VAST), pp 13–24. <https://doi.org/10.1109/VAST.2017.8585721>
- Mühlbacher T, Linhardt L, Möller T, Piringer H (2018) Treepod: sensitivity-aware selection of pareto-optimal decision trees. *IEEE Trans Vis Comput Graph* 24(1):174–183. <https://doi.org/10.1109/TVCG.2017.2745158>
- Park H-S, Jun C-H (2009) A simple and fast algorithm for k-medoids clustering. *Expert Syst Appl* 36(2, Part 2):3336–3341. <https://doi.org/10.1016/j.eswa.2008.01.039>
- Qiu X, Zuo Y, Liu G (2018) Etcf: an ensemble model for CTR prediction. In: 2018 15th International conference on service systems and service management (ICSSSM), pp 1–5. <https://doi.org/10.1109/ICSSSM.2018.8465044>
- Rauber PE, Fadel SG, Falcão AX, Telea AC (2017) Visualizing the hidden activity of artificial neural networks. *IEEE Trans Vis Comput Graph* 23(1):101–110. <https://doi.org/10.1109/TVCG.2016.2598838>
- Ribeiro MT, Singh S, Guestrin C (2016) “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. KDD '16. Association for Computing Machinery, New York, pp 1135–1144. <https://doi.org/10.1145/2939672.2939778>
- Rule A, Tabard A, Hollan JD (2018) Exploration and explanation in computational notebooks. In: Proceedings of the 2018 CHI conference on human factors in computing systems. CHI '18. Association for Computing Machinery, New York, pp 1–12. <https://doi.org/10.1145/3173574.3173606>
- Spearman C (1961) The proof and measurement of association between two things. In: Jenkins JJ, Paterson DG (eds) *Studies in individual differences: the search for intelligence*. Appleton-Century-Crofts, New York, pp 45–58. <https://doi.org/10.1037/11491-005>
- Talbot J, Lee B, Kapoor A, Tan DS (2009) Ensemblematrix: interactive visualization to support machine learning with multiple classifiers. In: Proceedings of the SIGCHI conference on human factors in computing systems. CHI '09. Association for Computing Machinery, New York, pp 1283–1292. <https://doi.org/10.1145/1518701.1518895>

- Teoh ST, Ma K-L (2003) Paintingclass: interactive construction, visualization and exploration of decision trees. In: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining. KDD '03. Association for Computing Machinery, New York, pp 667–672. <https://doi.org/10.1145/956750.956837>
- Thudt A, Walny J, Perin C, Rajabiyazdi F, MacDonald L, Vardeleon R, Greenberg S, Carpendale S (2016) Assessing the readability of stacked graphs. In: Proceedings of the 42nd graphics interface conference. GI '16. Canadian Human-Computer Communications Society, Waterloo, CAN, pp 167–174. <https://doi.org/10.5555/3076132.3076164>
- Wang X, Hu G, Lin H, Sun J (2019) A novel ensemble approach for click-through rate prediction based on factorization machines and gradient boosting decision trees. In: Shao J, Yiu ML, Toyoda M, Zhang D, Wang W, Cui B (eds) Web and big data. Springer, Cham, pp 152–162. https://doi.org/10.1007/978-3-030-26075-0_12
- Wang ZJ, Zhong C, Xin R, Takagi T, Chen Z, Chau DH, Rudin C, Seltzer M (2022) Timbertrek: exploring and curating sparse decision trees with interactive visualization. In: 2022 IEEE visualization and visual analytics (VIS), pp 60–64. <https://doi.org/10.1109/VIS54862.2022.00021>
- Wang F, Liu X, Liu O, Neshati A, Ma T, Zhu M, Zhao J (2023) Slide4n: creating presentation slides from computational notebooks with human-ai collaboration. In: Proceedings of the 2023 CHI conference on human factors in computing systems. CHI '23. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3544548.3580753>
- Ware M, Frank E, Holmes G, Hall M, Witten IH (2001) Interactive machine learning: letting users build classifiers. *Int J Hum Comput Stud* 55(3):281–292. <https://doi.org/10.1006/ijhc.2001.0499>
- Yang W, Ye X, Zhang X, Xiao L, Xia J, Wang Z, Zhu J, Pfister H, Liu S (2022) Diagnosing ensemble few-shot classifiers. *IEEE Trans Vis Comput Graph* 28(9):3292–3306. <https://doi.org/10.1109/TVCG.2022.3182488>
- Yuan J, Chen C, Yang W, Liu M, Xia J, Liu S (2021) A survey of visual analytics techniques for machine learning. *Comput Vis Med* 7:3–36. <https://doi.org/10.1007/s41095-020-0191-7>
- Yuan J, Liu M, Tian F, Liu S (2023) Visual analysis of neural architecture spaces for summarizing design principles. *IEEE Trans Vis Comput Graph* 29(1):288–298. <https://doi.org/10.1109/TVCG.2022.3209404>
- Zhang T (2021) Visual interpretation and analysis of random forest. Master's thesis, University of Electronic Science and Technology of China
- Zhang K, Shasha D (1989) Simple fast algorithms for the editing distance between trees and related problems. *SIAM J Comput* 18(6):1245–1262. <https://doi.org/10.1137/0218082>
- Zhang J, Gruenwald L, Gertz M (2009) VDM-RS: a visual data mining system for exploring and classifying remotely sensed images. *Comput Geosci* 35(9):1827–1836. <https://doi.org/10.1016/j.cageo.2009.02.006>
- Zhang C, Wang X, Zhao C, Ren Y, Zhang T, Peng Z, Fan X, Ma X, Li Q (2023) Promotionlens: inspecting promotion strategies of online e-commerce via visual analytics. *IEEE Trans Vis Comput Graph* 29(1):767–777. <https://doi.org/10.1109/TVCG.2022.3209440>
- Zhao X, Wu Y, Lee DL, Cui W (2019) iforest: interpreting random forests via visual analytics. *IEEE Trans Vis Comput Graph* 25(1):407–416. <https://doi.org/10.1109/TVCG.2018.2864475>
- Zhao J, Fan M, Feng M (2022) Chartseer: interactive steering exploratory visual analysis with machine intelligence. *IEEE Trans Vis Comput Graph* 28(3):1500–1513. <https://doi.org/10.1109/TVCG.2020.3018724>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.