GBMVis: Visual Analytics for Interpreting Gradient Boosting Machine

Yulu Xia[1], Kehan Cheng[1], Zhuoyue Cheng[1],

Yunbo Rao[2], and Jiansu Pu[1]

[1] VisBig Lab, Department of Computer Sci&amp; Tech University of Eletronic Science and Tech of China

[2] School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China

Abstract. The gradient boosting machine (GBM) composed of multiple weak learners is an efficient and widely used machine learning method. As a key factor in the prediction process of the gradient boosting machine, feature affects the performance of the gradient boosting machine when splitting nodes. The idea of GBM gives it a natural advantage to discover a variety of distinguishing features and feature combinations. Once the gradient boosting machine has the correct features, other factors play a relatively weak role. However, the GBM is a complicated and tedious process with diverse structure and attributes of decision tree, leading the model to be less interpretable, especially for high risk areas such as medical diagnosis and financial analytics that require transparent prediction. To tackle this issue, we have proposed an interactive visual analytic system, GBMVis, to help experts quickly analyze and of the gradient boosting machine. In addition to providing information about the features, we have also provided a visualization of the structure of boosting trees, which aims to display the major data flow in the gradient boosting machine. We have demonstrated the effectiveness of our system in a real dataset.

Keywords: Gradient Boosting Machine • Boosting Tree • Feature • Prediction

1 Introduction

Gradient boosting machine (GBM), an ensemble machine learning model that consists of many independent weak learners, has been proved to be widely utilized in different areas, such as regression methods and classification [22]. GBM is one of the best traditional machine learning algorithms for fitting real distributions. It is an algorithm that classifies or regresses data by using an additive model (i.e., a linear combination of basic functions) and continuously reducing the error generated by the training process [9]. Due to its effectiveness, GBM is applied to commercial tasks. Before deep learning was a big deal, GBM was a big hit in various competitions because of its stable performance, diverse applications and the function of filtering features. For example, GBM appears in the solution of the champion team at KDD Cup in

2016[18]. Besides, gradient boosting machine has been adopted by 9 of the first 14 teams in Kaggle competition since 2016[23].

Despite the wide use and high efficiency, the GBM still meets the function flaw and the performance limitation in the practical application. The first comes from the complex structure of the GBM model and its algorithm. Although GBM has the advantage of stable performance, it also has some limitations. For a large number of identity features, GBM cannot store them effectively due to the tree depth and tree limitation (to prevent overfitting). The second challenge is the output of features engineering. Compared to deep learning models, GBM lacks a certain encoder capability, which is a different feature point organization method that gives flexibility in constructing the network, and corresponding network is usually constructed to better encode the features according to the specific problem [17]. Therefore, to generate useful features is a difficult work with manually selecting process. The third issue stems from the need to interpret the GBM model completely and intuitively. One of the drawbacks of GBM algorithm is that it requires careful tuning of parameters, and the training time may be longer owing to confusing information, which affects performance analysis.

To tackle challenges mentioned above, we develop GBMVis, an interactive visualization system, to help users and machine learning experts interpret gradient boost machine from different perspectives. Specifically, our contributions in this paper are summarized as follows:

- An interactive visualization system, GBMVis, that assists users in interpreting gradient boost decision tree models and predictions through three levels: overview level, feature level and prediction level.
- A combination of diverse feature bar chart design that explain the relationship between feature and prediction.
- A usage scenarios and qualitative user research, which proves the effectiveness and usefulness of GBMVis on incomplete data sets.

2 Background

In this paper, the prediction model is from the GBM plus logistic regression (LR) model adopted by He et. al. in 2014[9]. The model structure is shown below as Fig. 1. It is a hybrid model with input features transformed by boosted decision tree method and output of each tree treated as categorical input to a linear classifier, respectively. We choose boosted decision tree approach due to its high effectiveness in feature transform. The algorithm we use in this article is Lightgbm[1], which is a specific implementation of the GBM model

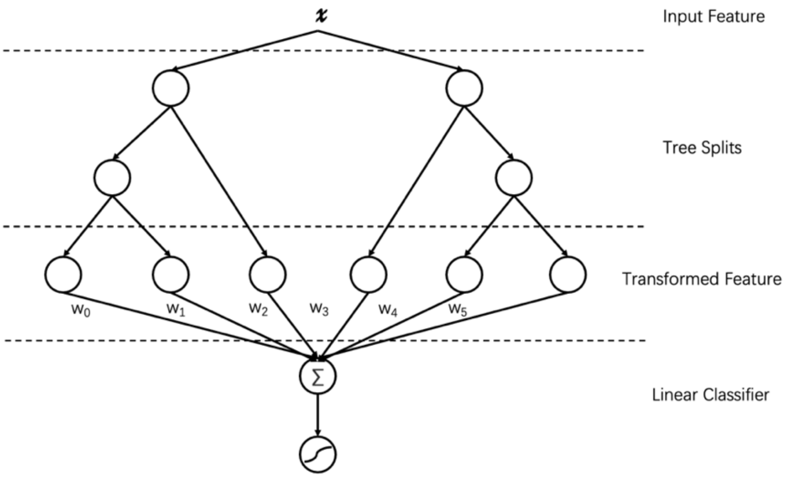


Fig. 1. Prediction model. There are two trees in the figure, and x is an input sample. After traversing the two trees, the x samples fall on the leaf nodes of the two trees, and each leaf node corresponds to LR one-dimensional features, then by traversing the trees, all the LR features corresponding to that sample are obtained. These features are used as input into the linear classifier for classification.

The fundamental algorithm in GBM is two-class logistic regression and classification [6]. The loss function is

$$L(y,F) = \log(1 + \exp(-2yF)), \qquad y \in \{-1,1\} \tag{1}$$

where

$$F(x) = \frac{1}{2}\log\left[\frac{\Pr(y=1|x)}{\Pr(y=-1|x)}\right] \tag{2}$$

The pseudo-response is

$$\tilde{y}_i = -\left[\frac{\partial L\big(y_i,\ F(x_i)\big)}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} = \frac{2y_i}{1+\exp\big(2y_i F_{m-1}(x_i)\big)} \tag{3}$$

line search is

$$\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} \log\Big(1+\exp\Big(-2y_i\big(F_{m-1}(x_i) + \rho h(x_i;a_m)\big)\Big)\Big) \tag{4}$$

Then use separate updates in each terminal node and treat regression trees as base learners:

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} \log\big(1+\exp\big(-2y_i(F_{m-1}(x_i) + \gamma)\big)\big) \tag{5}$$

(5) has no close-form solution. Then approximate the loss function by a single Newton-Raphson step:

$$\gamma_{jm} = \sum_{x_i \in R_{jm}} \tilde{y}_i \Big/ \sum_{x_i \in R_{jm}} |\tilde{y}_i|(2 - |\tilde{y}_i|) \tag{6}$$

$\tilde{y}_i$ is from (3). the whole algorithm is below.

ALGORITHM 1 (L2-TreeBoost)

```
$F_0(x) = \frac{1}{2}\log\frac{1+\bar{y}}{1-\bar{y}}$
For m =1 to M do:
$\tilde{y}_i = \frac{2y_i}{1+\exp(2y_i F_{m-1}(x_i))}, i = 1, N$
$\{R_{jm}\}_1^J$ = J-terminal node $tree(\{\tilde{y}_i, x_i\}_1^N)$
$\gamma_{jm} = \sum_{x_i \in R_{jm}} \tilde{y}_i \Big/ \sum_{x_i \in R_{jm}} |\tilde{y}_i|(2 - |\tilde{y}_i|), j = 1, J$
$F_m(x) = F_{m-1}(x) + \sum_{j=1}^{J} \gamma_{jm} 1(x \in R_{jm})$
end For
end Algorithm
```

FM(x) has relationship to the odds of log in (2), which can be transformed to yield probability estimates

$$\rho_+(x) = \widehat{Pr}(y = 1\,|x) = \frac{1}{1 + e^{-2F_M(x)}} \tag{7}$$

$$\rho_-(x) = \widehat{Pr}(y = -1\,|x) = \frac{1}{1 + e^{2F_M(x)}} \tag{8}$$

Then it can be classified:

$$\hat{y}(x) = 2 \cdot 1[c(-1,1)\rho_+(x) > c(1,-1)\rho_-(x)] - 1 \tag{9}$$

c($\hat{y}$,y) is the cost related to $\hat{y}$ when the truth is y.

3 Related Work

In this section, we review existing work on tree-based model and model prediction visualizations.

3.1 Visualization of tree-based model

Feature Interpretation. Feature interpretation methods are mainly divided into 2 categories: feature importance and partial dependence plots.

Feature importance calculates a score for each feature to reflect its impact on model predictions [2, 3, 8]. Common criteria for evaluating the feature importance are Mean Decrease Accuracy (MDA) [2] and Mean Decrease Impurity (MDI) [3]. MDA is a model-independent method, which randomly arranges the values of features and measures the degree of reduction in prediction accuracy. MDI [16] calculates the average decreased impurity (e.g., Gini impurity [8]) to determine the feature used to divide each node in the decision tree [16] and its split value. In this paper, the result of feature importance calculation contains numbers of times the feature is used in a model. We rank the features according to their importance in predictions to find appropriate split values for each node, so that effectively boosting trees.

Another approach to reveal the relationship between features and predictions is the partial dependence plots (PDPs) [6, 7], which reflects how features affect predictions. PDP is generally visualized with a line chart, where the x-axis represents the value range of the feature, and the y-axis represents the predicted probability [13]. Prospector adds two reference lines to the original data to represent a standard deviation of the mean in two directions. The red horizontal reference line represents the average predicted risk of the model, and the black vertical line represents the average observed value of the input data. Due to the intuitiveness of the line chart, we also use it to describe the partial dependence information.

Data Flow Interpretation. Zhao et al. [22] proposes a visual analytic system, iForest, to interpret random forest models and predictions. iForest designs a pixel-based bar chart, which summarizes the decision paths to reveal the underlying working mechanism of random forest. Unlike boosting trees, decision trees used in random forest are classification trees, while boosting trees are regression trees. In this paper, we focus on the decision path of the regression tree, including feature discretization.

BaobabView[20], which visualizes a single decision tree, develops a flexible layout method for trees that use color-banded edges with variable width to visualize the flow of the data through the tree, and designs visual annotations to the confusion matrix. In order to display the tree structure more intuitively, BOOSTVis[15] uses the node link tree visualization, and also considers the visualization of the tree as a whole. In this paper, we pay more attention to the interpretability of the boosting tree and reveal its influence on the prediction results through visualization techniques, rather than the structural details of the tree.

3.2 Visualization for model prediction

INFUSE [12] proposes a visual design, which assist experts in understanding how predictive features are ranked across feature selection algorithms, cross-validation folds, and classifiers. Each visual object represents a feature, and the information obtained from the algorithm can be reflected in its design and layout. Prospector [13] proposes novel visual representations to interactively diagnose partial dependence and compare multiple models. It also supports localized inspection to understand the rules for predicting specific data points. To help data scientists to understand their models, EXPLAINEXPLORE [5] supports a variety of different data sets and machine learning models. Experts could use explanations to diagnose the model and find problems.

In this paper, diagnosing and improving the model is not our main purpose. We propose an interactive visual analytic system, which can reflect the model prediction results, and analyze the features used in model prediction through visualization. We also show the structure of trees in GBM.

4 Design Goals

After a comprehensive summary of papers collected from machine learning, visualization and human-computer interaction fields, we propose the following design goals.

G1: Uncover the relationship between features and model prediction. In order to better understand GBM, users should first understand the general knowledge of the model and be able to evaluate the predictions of the model [14]. During the training process, GBM learns the mapping between features and predictions, which reflects the model behaviors. Displaying input features and their relationships with model predictions enables users to understand and measure the impact of features on prediction results. For example, users may want to know which features are important in the prediction, or which minor changes in feature values will not significantly affect the prediction results, so that users can modify the model or delete unnecessary features. Therefore, uncovering the relationship between features and predictions is helpful to explain GBM.

G2: Reveal the internal mechanism of the model. Making GBM transparent not only needs to reveal the relationships between features and predictions, but also reveals the internal mechanism of the model [14]. Experts should be able to review the model prediction to ensure the correctness of the decision process. For example, Data scientists working in banks may need to understand the model predictions of loan applications to assess whether the loan applicants are in good standing, so that they can decide whether to apply for loans for them. The working principle of GBM can be described by the structure and decision path of the boosting tree. For example, the information gain of each branch node of the tree describes its splitting performance. Analyzing the decision path from root to leaf can help explain the rules of certain features and the prediction process of GBM. Therefore, displaying boosting trees can help experts understand the underlying working mechanism of GBM.

G3: Provides case-based analysis. Case-based analysis is the most effective part of the decision-making program [10]. Its idea is to extend the solution of similar problems to new problems [11]. When interpreting GBM, users can compare new cases with similar cases in the training data to evaluate predictions [4]. However, there are multiple measures for calculating the similarity of GBM. For example, the similarity can be measured by calculating how many common leaf nodes reach [19]. Therefore, providing different similar cases is helpful for users to evaluate predictions from various angles.

5 Analytical Tasks

T1. Encode feature importance. Feature importance helps users build an understanding of the model(G1). Users may not be interested in the internal mechanism and structure of the model, but are more familiar with the features. Feature importance reflects the impact of a feature on the prediction result, which is consistent with the user's subjective perception and can increase the user trust in the model.

T2. Encode partial dependence information. Feature importance refers to the influence of a certain feature on model prediction, which is a numerical value, and partial dependence information can reflect how this feature affects prediction(G1). Partial dependence information can be used to answer similar questions as follows: If all other features remain unchanged, what effect does latitude and longitude have on housing prices? In the two different groups of people, is the difference in health predicted by the model caused by their debt level, or is there another reason? What partial dependence information reflects is the change of the mean value. If the features of some training data increase the prediction probability, and some decrease, they may cancel each other out. Therefore, in practical applications, users need to be patient to observe and interpret.

T3. Encode split point distribution. The split points distribution is another key clue to reveal the relationship between features and predictions (G1). In this paper, we use histogram algorithm to split features. The basic idea of the histogram algorithm is to discretize the continuous feature values into k integers, and then construct a histogram with a width of k. When traversing the data, the discretized value will be used as an index to accumulate statistics in the histogram. After traversing the data once, the histogram accumulates the required statistics. Then the algorithm traverses to find the optimal split point according to the discrete value of the histogram.

T4: Review training data value distribution. Reviewing this distribution is useful for both feature(G1) and case-based analysis (G3). The training data value distribution can also be used as evidence when partial dependence information and the split point distribution cannot correctly reflect the relationship between features and predictions. For example, if there is interaction between variables, it is possible that the partial dependence plot is completely flat, but the feature importance is very large. Furthermore, examining the distribution of similar training data is also helpful to check the model predictions.

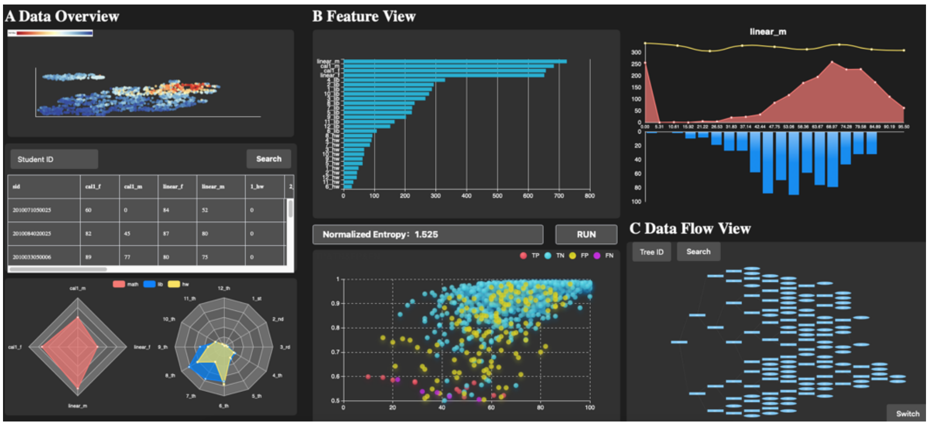
T5: Analyze prediction errors. During the inspection process, the user may need to check whether the test data is correctly predicted. If a set of data is incorrectly predicted, they can check the feature values and distribution of the test data to determine which features caused the error (G1). Understanding the prediction error of the model allows users to improve model inputs and parameters.

T6: Encode structures of tree. Each tree's structure is unique, including the depth of the tree, the number of branches of the tree, the features that appear on the path from the root to the leaves, and the segmentation threshold of the features in each node. These structures can give users a deeper insight into the underlying working mechanism(G2) of GBM.

T7: Provide interactive inspection of the model. GBMVis provides interactive inspections of the three design goals listed above. Users may want to know the impact of different input features on the prediction results(G1). They may also be interested in a single tree in GBM (G2), or interactively analyze similar cases (G3). Therefore, interactive inspection can help users better understand the model.

6 Visual Design

As shown in Fig. 2., GBMVis contains three main views: 1) Data Overview, which shows the training data after dimensionality reduction. We provide a search function that allows users to understand a single piece of data and view its features; 2) Feature View, which describes the relationship between features and prediction results from multiple perspectives; 3) Data Flow Path, which aims to enable users to observe the flow of data during the prediction process by showing the structure of the boosting trees, and understand GBM more intuitively. In addition, users can explore the working mechanism through multi-graph interactions.



sid	call_f	call_m	linear_f	linear_m	1_hw
2010071050025	60	0	84	52	0
2010084020025	82	45	87	80	0
2010033050006	89	77	80	75	0

Fig. 2. Using GBMVis to interpret GBM with student card records dataset. (A) a Data Overview's visual encoding for a) general data information viewing b) feature values of a single data. By entering a single data in the search box, the upper and lower parts of the data respond simultaneously. (B) a Feature View's visual encoding for a) feature importance, b) partial dependence information, c) training data value distribution, d) split point distribution, e) prediction errors. (C) a Data Flow View's visual encoding for structure of boosting trees.

6.1 Data Overview

Data overview summarizes data information in order to give users a comprehensive grasp of all data. The data we use is multidimensional data. If we visualize it directly, it not only reduces the response efficiency of the system, but also makes it difficult to display all the data on the visual charts. Therefore, according to the amount and type of data we have, we use the t-SNE dimensionality reduction technique for multidimensional data [21]. t-SNE uses Principal Component Analysis (PCA) to reduce multidimensional data to two dimensions and then maps it to the visualization chart. The t-SNE enables the user to visualize the distribution of the data. The objective function of t-SNE used to can be expressed as：

$$C = \sum_{i} KL(P_i \parallel Q_i) = \sum_{i} \sum_{j} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \tag{10}$$

As in Fig. 2A., we show a data table that allows users to easily browse information about the data by searching. The features of a single piece of data are displayed using a radar chart. The radar chart shows a particular data more clearly than the traditional display of values. The user can modify the data displayed in the radar chart by searching.

6.2 Feature View

In Feature View, users can see the values of individual features to analyze the relationship between features and predicted results. This helps users to understand the model. For example, the user may want to know which features are most important for model learning (T1) and how the features affect the prediction (T2). In addition, split point distribution for each feature (T3) and training data value distribution (T4) reflect the relationship between features and predictions. Therefore, we designed Feature View to interpret such information to help users understand the model.

In Fig. 2B., we visualize the three plots showing the feature information as one graph. From top to bottom, they are the line plot, area plot, and histogram, which represent the partial dependence information, training data value distribution, and split point distribution, respectively. The line graph represents the partial dependence information. For a feature $f^m \in F = \{f^1, f^2, \dots, f^M\}$, let $C = F - \{f\}$ to be the complement set of f， and the partial dependence can be calculated as:

$$PDP_{f^m}(\alpha) = \frac{1}{N} \sum_{i=1}^{n} GBM(x_i^C, x_i^m = \alpha) \tag{11}$$

where N denotes the size of the training data and the function $GBM(x)$ denotes the GBM models, which takes the training data as input and outputs the probabilities. Partial dependence calculates the average value of the prediction f, x_i^C is fixed and the value of feature m for each data is set to α. In this figure, the y-axis indicates the prediction of the model. In Fig. 2B., we use area plots to show the distribution of the training data values. The x-axis represents the feature values within a certain range represented by each bin, and the y-axis represents the number of data items within a certain range of feature values. We use histogram algorithm to count the number of split points of feature.

6.3 Data Flow View

In the Data Flow View, we present the structure of the boosting trees to allow users to gain a clearer understanding of the data flow in the GBM model during construction and prediction. The structure of the boosting tree is designed to reveal the structure and properties of the decision paths, which allows users to examine the order in which features appear in different decision paths. This is crucial for measuring the importance of features. As shown in Fig. 2C., the depth of the tree increases from left to right.

7 Case Study

In this section, we describe a usage scenario and a qualitative case study that demonstrate the effectiveness of GBMVis. The data set we use comes from the behavior and score of students at the University of Electronic Science and Technology of China.

The dataset is the four-year student card records of the students who enrolled in the University of Electronic Science and Technology of China in 2009 and 2010 (9,457 in total), and the scores of all the students' courses. There are about 3 million records, including hot water in the teaching building and access to the library. The number of times a student goes to the library and the teaching building can reflect his learning status. We think that the more times a student goes to the library or teaching building, the more time he spends on studying. The score data includes mid-term and final scores. We first process the training data and simplify the model by combining similar or redundant features. Finally, we generated the following features: the mid-term and final scores of calculus and linear algebra in the first semester, and the number of access to the library and hot water in the teaching building in the first twelve months. The reason why we only choose the first year's math courses scores and behaviors as input features is that the first year is a critical period, which largely determines the student's final GPA. Math is also a key course, which widens the gap between students' scores. However, too many features can overfit the model. The label to be predicted is whether the student's grade point average (GPA) reached 60 points at graduation. This is a binary variable, where 60 points or less are marked as positive.

First, we find that the mid-term score of linear algebra in the first semester(linear_m) ranked top by observing the Feature View, so linear_m is the most important feature related to prediction(T1). Then we check the split point distribution(T3) and training data value distribution(T4). We find most data are distributed over 50, of which 60 have the highest statistic. This is consistent with the fact that students with good math scores have a higher GPA when they graduate. But in partial dependence plot(T2), the change of linear_m did not have much impact on the prediction result. Since the PDP curve has been fluctuating in a relatively stable interval, we need to examine the reason in conjunction with the analysis of prediction errors(T5).

Through the scatter chart, we observe that the linear_m of the students in the TN group (GPA higher than 60) are all concentrated above 60, while the students in the TP group (GPA lower than 60) are almost below 60. However, the FN group (which is incorrectly predicted to have GPAs higher than 60) has a similar score distribution to the TP group, but the prediction results are different. In order to check the model further, we click on a certain point (T7) in the FN group to view other feature values of the student. We first check the feature values of the student whose student ID is 292311017, and find that the student's math score is above 60 three times, but linear_m is less than 60. It can be seen that although linear_m has the highest importance, other math scores still affect the prediction. Then, we choose the student whose student ID is 2901304009. His math scores are all below 60. By observing his behavior data, we find that he went to the library 6 times in the fourth month(4_lib). The fourth month is the last month of the first semester, and the final exam will be held that month. This shows that the model mistakenly believes that the student spends more time studying, so his GPA is high. Understanding the prediction errors of the model can help users improve the model. If users, especially machine learning beginners, are interested in the working mechanism of the model, they can view the structure of the boosting tree(T6). But if users do not understand machine learning, they can also understand the prediction through the previous steps.

8 Conclusion

In this paper, we have presented an interactive visualization tool, GBMVis, that helps users and machine learning experts interpret gradient boosted machine model. The tool first assists analyze the input feature automatically, hence boosting the feature engineering process. In addition, the main decision path is visualized to display the data flow in gradient boosting machine, which clearly show the basic mechanism of GBM procedure. The result of the visualization indicates that our tool can effectively explain the relationship between the input feature and the prediction result and demonstrate all key feature combination and decision path intuitively.

For future work of GBMVis system, there are three aspects to be improved. Firstly, we are going to expand the dataset and use another set of data for case study to further validate the effectiveness of our tool. Secondly, according to the experimental results of He et al. [9], in addition to using GBM, we also use logistic regression for feature classification, and this part should be visualized and added to GBMVis to completely explain the feature processing process. Thirdly, the visualization view of the GBMVis needs to be added more interactivity, such as adding motion pictures to help experts adjust parameters and change settings more easily and quickly.

References

1. Microsoft LightGBM. https://github.com/Microsoft/ LightGBM. Accessed: 31-Mar-2017.
2. Breiman, Leo. "Random forests." *Machine learning* 45.1 (2001): 5-32.
3. Breiman, Leo. "Manual on setting up, using, and understanding random forests v3. 1." *Statistics Department University of California Berkeley, CA, USA* 1 (2002): 58.

4. Caruana, Rich, et al. "Case-based explanation of non-case-based learning methods." *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 1999.
5. Collaris, Dennis, and Jarke J. van Wijk. "ExplainExplore: visual exploration of machine learning explanations." *2020 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, 2020.
6. Friedman, Jerome H. "Greedy function approximation: a gradient boosting machine." *Annals of statistics* (2001): 1189-1232.
7. Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. No. 10. New York: Springer series in statistics, 2001.
8. Genuer, Robin, Jean-Michel Poggi, and Christine Tuleau-Malot. "Variable selection using random forests." *Pattern recognition letters* 31.14 (2010): 2225-2236.
9. He, Xinran, et al. "Practical lessons from predicting clicks on ads at facebook." *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. 2014.
10. Kim, Been, Cynthia Rudin, and Julie Shah. "The bayesian case model: A generative approach for case-based reasoning and prototype classification." *arXiv preprint arXiv:1503.01161* (2015).
11. Kolodner, Janet. *Case-based reasoning*. Morgan Kaufmann, 2014.
12. Krause, Josua, Adam Perer, and Enrico Bertini. "INFUSE: interactive feature selection for predictive modeling of high dimensional data." *IEEE transactions on visualization and computer graphics* 20.12 (2014): 1614-1623.
13. Krause, Josua, Adam Perer, and Kenney Ng. "Interacting with predictions: Visual inspection of black-box machine learning models." *Proceedings of the 2016 CHI conference on human factors in computing systems*. 2016.
14. Lipton, Zachary C. "The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery." *Queue* 16.3 (2018): 31-57.
15. Liu, Shixia, et al. "Visual diagnosis of tree boosting methods." *IEEE transactions on visualization and computer graphics* 24.1 (2017): 163-173.
16. Louppe, Gilles, et al. "Understanding variable importances in forests of randomized trees." *Advances in neural information processing systems 26* (2013).
17. Ng, Andrew. "Machine Learning and AI via Brain simulations." *Accessed: May* 3 (2013): 2018.
18. Sandulescu, Vlad, and Mihai Chiru. "Predicting the future relevance of research institutions-The winning solution of the KDD Cup 2016." *arXiv preprint arXiv:1609.02728* (2016).
19. Tan, Sarah, et al. "Tree space prototypes: Another look at making tree ensembles interpretable." *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*. 2020.
20. Van Den Elzen, Stef, and Jarke J. Van Wijk. "Baobabview: Interactive construction and analysis of decision trees." *2011 IEEE conference on visual analytics science and technology (VAST)*. IEEE, 2011.
21. Van der Maaten, Laurens, and Geoffrey Hinton. "Visualizing data using t-SNE." *Journal of machine learning research* 9.11 (2008).
22. Zhao, Xun, et al. "iForest: Interpreting random forests via visual analytics." *IEEE transactions on visualization and computer graphics* 25.1 (2018): 407-416.
23. No free hunch–the official blog of kaggle.com. http://blog. kaggle.com/category/winners-interviews/. Accessed: 31-Mar-2017.