Tree Ensemble Explainability

Alexander Moore [1] Yaxiong Cai [1] Kristine Jones [1] Vanessa Murdock [1]

Abstract

Complex machine learned models play an increasingly important role in modern technologies, consuming large amounts of data to provide a plethora of useful services. While these systems are highly effective, many of them are black boxes and give no insight into how they make the choices they make. Moreover, those that do often do so at the model-level rather than the instance-level. In this work, we present a method for deriving instance-level explanations for tree ensemble models and examine its applications. Tree ensemble models such as Random Forests and Boosted Trees are used across industry with great success; adding a level of insight simultaneously boosts model effectiveness and consumer trust.

1. Model Explainability

As machine learned models drive more and more of our everyday experiences, they have the potential to greatly enhance our lives. These models depend on vast amounts of data about who we are, where we go, what we like, and how we interact with technology. The technologies for targeting, personalization and other types of inference have become so useful that we are increasingly comfortable providing our data to support them.

While the technology is at times astoundingly sophisticated, the aggregation and inference over our personal data carries with it an inherent risk of violating our privacy and trust. As we push the boundaries of what machine learning can be used for, we reveal the limitations and biases inherent in the system. For this reason, it becomes increasingly important to provide insight into how the system made its decisions. Understanding the factors that led to a machine-learned decision allows us to correct biases, or remove our personal data. Revealing the reasons behind model decisions has the potential to increase user trust and engagement with the system. This type of transparency is healthy, and of benefit both to the models and to their consumers.

Much previous work in presenting model explanations centers around rule-based systems, which are used in domains where the model is an aid to human decision-making, such as for medical diagnoses. In such systems, the model is not the domain expert - rather, the human is the expert - so the focus is on making the models as understandable as possible at the expense of accuracy (Caruana et al., 2015).

Many modern technolgies, such as search engines, and recommender and forecasting systems are built on highly complex, highly accurate tree ensembles. These models do not lend themselves to human-understandable explanations at the instance level. Rather than present the exact features from the model itself, such a system may reveal heuristics derived from a single feature, or use the results of a second more interpretable model whose decisions are not related to the original model.

Ideally we would like to have our cake, and eat it too. We would like a model that is both a highly accurate predictor, and yields reasonably accurate explanations. One approach is to start with a model that lends itself to interpretation, and drive its accuracy to be on par with the state-of-the art. This is the approach of Lou et al. (Lou et al., 2012), building off of Generalized Additive Models (GAMs). In GAMs, the distributions of values for a given feature function can be plotted in two dimensions, which is a visualization of the model a domain expert can easily interpret. However, not all systems lend themselves to this type of visualization. In particular, if the consumer of the model is not a domain expert, the visualizations will not be meaningful. Further, as the number of features increases, the number of 2D plots produced also increases. There is a limit to how many can be viewed before the information becomes overwhelming.

In this paper, we define model explainability as two distinct processes:

1. Determining which features contributed to a model's decision at the instance level.

[1]Microsoft, Redmond, Washington, USA. Correspondence to: Alexander Moore <almoore@microsoft.com>, Yaxiong Cai <yaca@microsoft.com>, Kristine Jones <krjones@microsoft.com>, Vanessa Murdock <vanmur@microsoft.com>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

2. Producing a human-readable representation of the salient features.

We focus on the first process and ask the question: *What were the features that contributed to the model's decision for a given instance?*

We start with tree ensembles, which are known to be highly accurate for many classes of problems, and are in wide use in industry. We use the structure of the model to derive instance-level explanations, examining changes in expected output at each node along the path through the tree. Changes in expectation are added to the influence of the splitting feature, and these changes are aggregated across all nodes in all trees. Additionally, we maintain feature ranges within which feature contributions would remain unchanged to provide additional insights. In this work we focus on Random Forests, but the same methodology applies to other tree ensemble models. Additionally, while our case study is classification, the same methods apply to other tasks.

2. Related Work

In a trio of papers exploring Generalized Additive Models (GAMs), Lou and Caruana et al. (Lou et al., 2012; 2013; Caruana et al., 2015) propose that GAMs are especially suited for model explanations because they lend themselves to visualizing the distributions of the values of the feature functions. The feature distributions can be plotted in 2D, making these visualizations very easy for a person looking at them to interpret. GAMs are not as accurate as full-complexity models, such as Random Forests and Boosted Trees, but they are far more accurate than rule-based systems, which are currently used for systems that must be human-interpretable, such as medical diagnoses. In (Lou et al., 2012) the authors show that the accuracy of GAMs can be improved with complex shape functions, with no harm to the interpretability of the models.

Since GAMs lend themselves to visualization, the contribution of their work is to push their accuracy to be on par with full-complexity models, without sacrificing efficiency, so that the models can scale and provide accurate results, while still being human interpretable. To this end, the authors propose GA^2Ms, which allow the inclusion of pairs of features, where the features interact (Lou et al., 2013). This yields more accurate models, while still being able to be visualized, because pairwise feature interactions can be shown on a heatmap. The authors further investigate efficient methods for discovering the most informative pairwise feature interactions.

In the third paper (Caruana et al., 2015) the authors present applications of GA^2Ms for classifying patients as likely to respond to treatment for pneumonia, and for classifying patients as likely to be readmitted to the hospital. In this work, the authors focus on a case study in the medical domain, which is one of the primary applications of the research into intelligible models. Because the model is used as an aid to human reasoning, it is not useful if it does not provide insight into the decision, and the model's predictions can be less accurate since the care provider will use the model input as one factor in deciding care for a patient.

For many systems, there will be no human intervention, and the system's utility is first and foremost dependent on its accuracy. There are many instances where a system would benefit from providing the user insight into the model decision, that are not at liberty to change the modeling architecture. Tree Ensembles are widely used across industry, for many search, personalization and recommendation systems. In contrast to the work of Caruana et al., we focus on providing instance-level explanations for decisions made by Random Forests and Boosted Trees. Other approaches to this problem exist: Ribeiro et al. (Ribeiro et al., 2016) proposed a method for instance-level explanations of black-box classification models, and work by Hara et al. (Hara and Hayashi, 2016) has focused on providing similar insights by building a simpler model for interpretation. Our approach differs from the black-box approach by taking into account the structure of tree ensemble models, allowing for other insights besides a raw feature ranking. It differs from both approaches by directly interpreting the original model rather than building an additional interpretable model.

It should be noted that work on feature selection (for an overview see (Forman, 2003)) is not sufficient here, because feature selection provides weighting of features at the model level, but does not give insight into the decision for a given instance.

3. Decision Tree Models

A predictive decision tree is a directed acyclic graph satisfying the following conditions:

- There is a single node, called the root, which has no incoming edges.
- Every other node in the graph has exactly one incoming edge.
- There is only one path from the root to any given node.
- The nodes at the bottom of the tree, called *leaves*, have no outgoing edges. Output predictions are stored in these nodes.

Each non-leaf node splits on one or more features, but in this work we focus on univariate splits. A split on feature

x_{ij} is described by $x_{ij} \leq t$, where t is some threshold value. Instances satisfying the inequality progress to the left subtree, while instances which do not satisfy the inequality progress to the right. Instances continue through the tree until they reach a leaf node, at which point they are assigned a prediction $F(\mathbf{x_i})$ according to the leaf node's output.

Decision tree learning involves deciding which feature to split on at each node n. This is done by selecting the feature j and threshold t that minimize a function $S(\mathbf{x}, \mathbf{y}, j, t)$ across all incoming instances $(\mathbf{x}, \mathbf{y})$. The choice of split function does not affect our algorithm. Our experiments use:

$$S_j = \sum_{i \in L}(y_i - \mu_L)^2 + \sum_{i \in R}(y_i - \mu_R)^2 \tag{1}$$

where μ_L is the mean y_i of instances with $x_{ij} \leq t$, and μ_R is the mean y_i of instances with $x_{ij} > t$.

3.1. Tree Ensembles

A tree ensemble is a collection of K decision trees whose output $F(\mathbf{x})$ for an instance $\mathbf{x}$ takes the form:

$$F(\mathbf{x}) = f(\{F_k(\mathbf{x}), k = 1..K\}) \tag{2}$$

where $F_k(\mathbf{x})$ is the prediction of the k^{th} tree for instance $\mathbf{x}$. In this work, we consider tree ensembles of the form:

$$F(\mathbf{x}) = f(\sum_{k=1}^{K} F_k(\mathbf{x})) \tag{3}$$

where f is a monotonically increasing function.

3.2. Random Forests

Random forests (Breiman, 2001) are tree ensemble models for which each tree is trained independently. Trees are each trained on a bootstrap sample of the data using a random subset of the features. In this work, outputs were written as indicated above, with:

$$f(v) = \frac{1}{1 + e^{-\alpha v + \beta}} \tag{4}$$

where v is the summed output and $\alpha, \beta \in R$.

4. The Explainability Model

The model we present aims to describe, for a given instance $\mathbf{x_i}$, the influence that each feature j's particular value x_{ij} had on the final output $F(\mathbf{x_i})$. We assume a tree ensemble model has already been trained, and that its structure cannot be modified. The tree ensemble output takes the form

$$F(\mathbf{x_i}) = f(\sum_{k=1}^{T} F_k(\mathbf{x_i})) \tag{5}$$

where T is the number of trees in the ensemble, f is a monotonically increasing function and F_k is the output of the k^{th} tree. Our model works as follows:

1. Assign output $O_k(n)$ to each node n in trees $k = 1..K$ prior to prediction time.
2. At prediction time, consider each tree one at a time, tracing the path of a given instance down the tree, and monitoring the change in output at each node. The output change for each node is added to the influence of that feature. The sum of all changes for a given feature is that feature's contribution.

Just as there are multiple ways to assign outputs to leaf nodes, there are multiple ways to assign outputs to internal nodes. In this work we use expected value.

More formally, assume there is some prior distribution $\mathcal{P}_k$ across all leaf nodes in the k^{th} tree. Let $P_k(l)$ be the prior probability of arriving in leaf node l in the k^{th} tree, and $\sum_l P_k(l) = 1$. Similarly, let $O_k(l)$ be the output at leaf node l. Consider a node n in tree k for which both children are leaf nodes. The expected output at node n is defined as the expectation across the leaves n_l and n_r, namely:

$$E_k[n] = \frac{P_k(n_l)O_k(n_l) + P_k(n_r)O_k(n_r)}{P_k(n_l) + P_k(n_r)} \tag{6}$$

More generally, the expected output at any node n can be written:

$$E_k[n] = \frac{\sum_{l \in L(n)} P_k(l)O_k(l)}{\sum_{l \in L(n)} P_k(l)} \tag{7}$$

where $L(n)$ is the set of all leaf nodes reachable from n. Let $P_k(n)$ be defined as $\sum_{l \in L(n)} P_k(l)$ for every node n. If n_l and n_r are the children of node n, the expectation can be rewritten as:

$$E_k[n] = P_k(n_l)E[n_l] + P_k(n_r)E[n_r] \tag{8}$$

This form yields the same result as Equation 7 but allows for more efficient computation. If traversing the tree in post-order, the expectations can be computed in time complexity $O(|N_k|)$ where N_k is the set of all nodes in the tree.

Given this definition of $E_k[n]$, the influence of a node n splitting on feature j for a given instance $\mathbf{x}$ is computed as the change in expected output from that node to the next node along $\mathbf{x}$'s path. More concretely, consider a tree k and an instance $\mathbf{x}$, and let $n_q(\mathbf{x})$ be the q^{th} node along the path $\mathbf{x}$ takes through tree k from the root node to the leaf node. The influence $\text{Infl}_k(n_q)$ of node n_q's split on feature j in tree k is:

$$\text{Infl}_k(n_q) = \Delta E_k[n_q \rightarrow n_{q+1}] = E_k[n_{q+1}] - E_k[n_q] \quad (9)$$

The overall influence of a feature j in tree k is the sum of the influence of all nodes along $\mathbf{x}$'s path that split on feature j, namely:

$$\text{FeatInfl}_k(j) = \sum_{n \in Path_k(\mathbf{x})} \text{Infl}_k(n) I_k(j,n) \quad (10)$$

where $Path_k(\mathbf{x})$ is the set of all nodes along instance $\mathbf{x}$'s path through tree k, and $I_k(j,n)$ is an indicator function which is 1 if node n splits on feature j and 0 otherwise. The feature influence across all trees is simply:

$$\text{FeatInfl}(j) = \sum_{k=1..K} \text{FeatInfl}_k(j) \quad (11)$$

4.1. Feature Influence Ranges

An extension of this method allows us to gain additional insight by generating feature influence ranges. These are ranges $v_{i,j,min} \leq x_{ij} \leq v_{i,j,max}$ such that the path direction after each node split on feature j for instance $\mathbf{x_i}$ would remain unchanged. In other words, as long as $v_{i,j,min} \leq x_{ij} \leq v_{i,j,max}$, all feature influence contributions by nodes $n \in Path(\mathbf{x_i})$ splitting on feature j would remain unchanged. These ranges are computed for tree k as follows:

$$\begin{aligned} v_{k,i,j,min} &= max(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : I_k(n,j) = 1, L_k(n) = 0) \quad (12) \\ v_{k,i,j,max} &= min(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : I_k(n,j) = 1, L_k(n) = 1) \quad (13) \end{aligned}$$

where t_n is the split threshold for node n and $L_k(n)$ indicates a split where $\mathbf{x_i}$ went left. Aggregating across all trees to find the narrowest range, we have:

$$\begin{aligned} v_{i,j,min} &= max(v_{k,i,j,min}, k = 1..K) \quad (14) \\ v_{i,j,max} &= min(v_{k,i,j,max}, k = 1..K) \quad (15) \end{aligned}$$

We can gain more insight by generating two ranges for each feature j in tree k: One for cases where $\text{Infl}_k(n) > 0$ and one for cases where $\text{Infl}_k(n) < 0$. Let these be called $v^+_{k,min}$, $v^+_{k,max}$, $v^-_{k,min}$ and $v^-_{k,max}$ with implicit dependencies on i, j and implicit $I_k(n,j) = 1$. We then have:

$$\begin{aligned} v^+_{k,min} &= max(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : \text{Infl}_k(n) > 0, L_k(n) = 0) \quad (16) \\ v^+_{k,max} &= min(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : \text{Infl}_k(n) > 0, L_k(n) = 1) \quad (17) \\ v^-_{k,min} &= max(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : \text{Infl}_k(n) < 0, L_k(n) = 0) \quad (18) \\ v^-_{k,max} &= min(t_n \forall n \in Path_k(\mathbf{x_i}) \\ &\qquad : \text{Infl}_k(n) < 0, L_k(n) = 1) \quad (19) \end{aligned}$$

Aggregation is once again:

$$\begin{aligned} v^+_{min} &= max(v^+_{k,min}, k = 1..K) \quad (20) \\ v^+_{max} &= min(v^+_{k,,max}, k = 1..K) \quad (21) \\ v^-_{min} &= max(v^-_{k,,min}, k = 1..K) \quad (22) \\ v^-_{max} &= min(v^-_{k,,max}, k = 1..K) \quad (23) \end{aligned}$$

which gives us the range values for each feature, partitioned by contribution direction. The following section demonstrates the above methods on a public data set.

5. Results

To test this model, we used the publicly available UCI 1995 Adult Census dataset (Lichman, 2013), wherein the classification task is to determine whether an individual is making over \$50K/year or not. The training set contained 32,561 rows, and the test set contained 16,281 rows with positive class ratios of 24% and 23.6% respectively and labels $y_i \in \{-1, 1\}$. Categorical variables were expanded into 0/1 indicators. The set of features used was as follows:

Numerical Features	Categorical Features
work class	capital gain
education	capital loss
native country	hours-per-week
marital status (mar)	education num
occupation (job)	age
relationship (rel)	
sex	
race	

Note that for each of the categorical features, an individual will have all values. For example, the same individual will have the features "marital status (Married) = 1" and "marital status (Never married) = 0".

We trained a Random Forest model of 100 trees with at most 20 leaves per tree, at least 10 instances per leaf, a bootstrap sample size of 70% and a feature sample size of 70%, achieving 90% AUC, and an Accuracy of 85.8%. The prior distributions $\mathcal{P}_k$ was computed using the training data. Examining the top 3 positive and negative outputs of the model on instances in the test set, we have:

Person A (95.1%) (Makes >$50K)

Rank	Feature	Influence	Min	Max
1	capital gain	+74.9	7073.5	∞
2	education num	+22.0	12.5	∞
3	mar (Married)	+19.3	0.5	∞
1	rel (Husband)	-12.27	$-\infty$	0.5
2	job (Prof special)	-0.73	$-\infty$	0.5
3	capital loss	-0.60	$-\infty$	1740.5

Person B (50.3%) (Makes ≤$50K)

Rank	Feature	Influence	Min	Max
1	education num	+35.19	12.5	∞
2	rel (Husband)	+19.35	0.5	∞
3	mar (Married)	+19.33	0.5	∞
1	hours-per-week	-38.42	$-\infty$	30.5
2	capital gain	-5.39	$-\infty$	5036.5
3	workclass (Self-emp)	-3.19	0.5	∞

Person C (9.15%) (Makes ≤$50K)

Rank	Feature	Influence	Min	Max
1	mar (Married)	+22.39	0.5	∞
2	age	+5.97	36.5	59.5
3	mar (Never married)	+1.68	$-\infty$	0.5
1	rel (Husband)	-12.97	$-\infty$	0.5
2	education num	-10.73	$-\infty$	9.5
3	job (Other service)	-4.32	0.5	∞

6. Discussion

The results of our method are often intuitive and can provide insights into the model. For Person A, the method indicates high capital gains were a strong positive influence. This makes intuitive sense– an individual with capital gains has owned and sold capital assets, implying affluence. The second influencer, years of education, also makes sense. In the dataset, college graduates had 13 years of education, so having over 12.5 years of education implies Person A graduated from college. Additionally, Person A is married, suggesting either a higher household income or a more stable life.

On the other hand, the strongest negative influencer was the individual not being a husband. This illustrates an unfortunate bias in the model. "Husband" frequently comes up as an influence, but "Wife" does not. In the training data, 40% of individuals were husbands, while only 5% were wives. A similar proportion of husbands and wives made over $50K, and among those making over $50K 75% were husbands while 10% were wives. Since husbands are 8 times as prevalent in the data as wives, one might expect the model to focus on their attributes. The reason for husbands being more prevalent could link to societal biases. The other negative influences, not being in a specialty and having low capital losses, are also intuitive but have relatively low influence.

Person B was more difficult for the model to discern. On the one hand, they are a college graduate, a husband and are married, all of which are taken as positive influencers. On the other hand, they work 30 or fewer hours per week (as opposed to the usual 40+), have less than 5036.5 in capital gains, and are self-employed. Once again, the influencers make intuitive sense– someone who works fewer hours, has fewer capital assets and is self-employed may not make $50K/year.

Person C is another obvious case for the model. Even though they are married (a big positive signal), are between the ages of 37 and 59 (late in career but before retirement) and are not currently *not* married, there are numerous negative signals. They are not a husband, have a high school education and work in the service industry.

While a person thinking of the characteristics of an example might choose a different ranking of influencers, the feature ranking gives insight into the model's process. This can be useful to model developers looking to improve performance or to identify bias in the data, or to end-users wondering what influenced a prediction. In this case, marriage and relationships seem to be important indicators, along with education and capital asset activity.

7. Conclusion

In this work, we defined the problem of explainability for tree ensembles, and proposed a method for per-instance explainability. We demonstrated the results in a brief case study, as a proof of concept. We leave the full evaluation of the explanations for future work, along with producing human-readable summaries of the model explanations.

References

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., and Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, pages 1721–1730, Sydney, Australia.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of machine learning research*, 3(Mar):1289–1305.

Hara, S. and Hayashi, K. (2016). Making tree ensembles interpretable. *WHI 2016. arXiv preprint arXiv:1606.05390*.

Lichman, M. (2013). UCI machine learning repository.

Lou, Y., Caruana, R., and Gehrke, J. (2012). Intelligible models for classification and regression. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, Beijing, China.

Lou, Y., Caruana, R., Gehrke, J., and Hooker, G. (2013). Accurate intelligible models with pairwise interactions. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, Chicago, Illinois.

Ribeiro, M. T., Singh, S., and 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*, pages 1135–1144. ACM.