Orthogonal Decision Trees

Hillol Kargupta*and Haimonti Dutta
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County, 1000 Hilltop Circle Baltimore, MD 21250
{hillol, hdutta1}@cs.umbc.edu

Abstract

This paper introduces orthogonal decision trees that offer an effective way to construct a redundancy-free, accurate, and meaningful representation of large decision-tree-ensembles often created by popular techniques such as Bagging, Boosting, Random Forests and many distributed and data stream mining algorithms. Orthogonal decision trees are functionally orthogonal to each other and they correspond to the principal components of the underlying function space. This paper offers a technique to construct such trees based on eigen-analysis of the ensemble and offers experimental results to document the performance of orthogonal trees on grounds of accuracy and model complexity.

1. Introduction

Decision tree [8] ensembles are frequently used in data mining and machine learning applications. Boosting [4, 3], Bagging[1], Stacking [10], and random forests [2] are some of the well-known ensemble-learning techniques. Many of these techniques often produce large ensembles that combine the outputs of a large number of trees for producing the overall output. Large ensembles pose several problems to a data miner. They are difficult to understand and the overall functional structure of the ensemble is not very "actionable" since it is difficult to manually combine the physical meaning of different trees in order to produce a simplified set of rules that can be used in practice. Moreover, in many time-critical applications such as monitoring data streams [9], particularly for resource constrained environments [5], maintaining a large ensemble and using it for continuous monitoring are computationally challenging. So it will be useful if we can develop a technique to construct a redundancy free meaningful compact representation of large ensembles. This paper offers a technique to do that.

This paper presents a technique to construct redundancy-free decision trees-ensembles by constructing orthogonal decision trees. The technique first constructs an algebraic representation of trees using multi-variate discrete Fourier bases. The new representation is then used for eigen-analysis of the covariance matrix generated by the decision trees in Fourier representation. The proposed approach converts the corresponding principal components to decision trees using a technique reported elsewhere [5]. These trees are functionally orthogonal to each other and they span the underlying function space. These orthogonal trees are in turn used for accurate (in many cases with improved accuracy) and redundancy-free (in the sense of orthogonal basis set) compact representation of large ensembles.

Section 2 presents a brief overview of the Fourier spectrum of decision trees. Section 3 describes the construction of orthogonal decision trees. Section 4 presents experimental results. Finally, Section 5 concludes this paper.

2. Fourier Transform of Decision Trees

This section briefly discusses the background material [5] necessary for the development of the proposed technique to construct orthogonal decision trees. The proposed approach makes use of linear algebraic representation of the trees. In order to do that that we first need to convert the trees into a numeric tree just in case the attributes are symbolic. This can be done by simply using a codebook that replaces the symbols with numeric values in a consistent manner. Since the proposed approach of constructing orthogonal trees uses this representation as an intermediate stage and eventually the physical tree is converted back, the exact scheme for replacing the symbols (if any) does not matter as long as it is consistent.

Once the tree is converted to a discrete numeric function, we can also apply any appropriate analytical transformation if necessary. Fourier transformation is one such interesting possibility. Fourier bases are orthogonal functions that can be used to represent any discrete function. Consider the set of all ℓ-dimensional feature vectors where the i-th feature can take λ_i different categorical values. The Fourier basis set that spans this space is comprised of $\Pi_{i=0}^{\ell}\lambda_i$ basis functions. Each Fourier basis function is defined as,

$$\psi_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x}) = \frac{1}{\sqrt{\Pi_{i=1}^{l}\lambda_i}}\Pi_{m=1}^{l}\exp^{\frac{2\pi i}{\lambda_m}x_m j_m}$$

where $\mathbf{j}$ and $\mathbf{x}$ are strings of length ℓ; x_m and j_m are m-th attribute-value in $\mathbf{x}$ and $\mathbf{j}$, respectively; $x_m, j_m \in \{0, 1, \cdots \lambda_i\}$ and $\overline{\lambda}$ represents the feature-cardinality vector, $\lambda_0, \cdots \lambda_\ell$; $\psi_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x})$ is called the $\mathbf{j}$-th basis function. The

*Also affiliated with AGNIK, LLC, USA.

vector $\mathbf{j}$ is called a *partition*, and the *order* of a partition $\mathbf{j}$ is the number of non-zero feature values it contains. A Fourier basis function depends on some x_i only when the corresponding $j_i \neq 0$. If a partition $\mathbf{j}$ has exactly α number of non-zeros values, then we say the partition is of order α since the corresponding Fourier basis function depends only on those α number of variables that take non-zero values in the partition $\mathbf{j}$.

A function $f : \mathbf{X}^\ell \rightarrow \Re$, that maps an ℓ -dimensional discrete domain to a real-valued range, can be represented using the Fourier basis functions: $f(\mathbf{x}) = \sum_{\mathbf{j}} w_{\mathbf{j}} \overline{\psi}^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x})$. where $w_{\mathbf{j}}$ is the Fourier Coefficient (FC) corresponding to the partition $\mathbf{j}$ and $\overline{\psi}^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x})$ is the complex conjugate of $\psi^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x})$; $w_{\mathbf{j}} = \sum_{\mathbf{x}} \psi^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x}) f(\mathbf{x})$. The *order* of a Fourier coefficient is nothing but the order of the corresponding partition. We shall often use terms like *high order* or *low order* coefficients to refer to a set of Fourier coefficients whose orders are relatively large or small respectively. Energy of a spectrum is defined by the summation $\sum_{\mathbf{j}} w^2_{\mathbf{j}}$. Let us also define the inner product between two spectra $\mathbf{w}_{(1)}$ and $\mathbf{w}_{(2)}$ where $\mathbf{w}_{(i)} = [w_{(i),1} w_{(i),2}, \cdots w_{(i),|J|}]^T$ is the column matrix of all Fourier coefficients in an arbitrary but fixed order. Superscript T denotes the transpose operation and $|J|$ denotes the total number of coefficients in the spectrum. The inner product, $< \mathbf{w}_{(1)}, \mathbf{w}_{(2)} >= \sum_{\mathbf{j}} w_{(1),\mathbf{j}} w_{(2),\mathbf{j}}$. We will also use the definition of the inner product between a pair of real-valued functions defined over some domain Ω. This is defined as $< f_1(\mathbf{x}), f_2(\mathbf{x}) >= \sum_{\mathbf{x} \in \Omega} f_1(\mathbf{x}) f_2(\mathbf{x})$.

Fourier transformations of bounded-depth decision trees have several properties that makes it an efficient one. More details can be found elsewhere [6, 7].

Let us also note that,

1. the Fourier spectrum of a decision tree can be efficiently computed [5] and
2. the Fourier spectrum can be directly used for constructing the tree [7].

In other words, we can go back and forth between the tree and its spectrum. This is philosophically similar to the switching between the time and frequency domains in the traditional application of Fourier analysis for signal processing.

Fourier transformation of decision trees also preserves inner product. The functional behavior of a decision tree is defined by the class labels it assigns. Therefore, if $\{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_{|\Omega|}\}$ are the members of the domain Ω then the functional behavior of a decision tree $f(\mathbf{x})$ can be captured by the vector $[f]_{x \in \Omega} = [f(\mathbf{x}_1) f(\mathbf{x}_2) \cdots f(\mathbf{x}_{|\Omega|})]^T$, where the superscript T denotes the transpose operation. The following section describes a Fourier analysis-based technique for constructing redundancy-free orthogonal representation of ensembles.

3 Removing Redundancies from Ensembles

Existing ensemble-learning techniques work by combining (usually a linear combination) the output of the base classifiers. They do not structurally combine the classifiers themselves. As a result they often share a lot of redundancies. The Fourier representation offers a unique way to fundamentally aggregate the trees and perform further analysis to construct an efficient redundancy-free representation.

Let $f_e(\mathbf{x})$ be the underlying function representing the ensemble of m different decision trees where the output is a weighted linear combination of the outputs of the base classifiers. Then we can write,

$$\begin{aligned} f_e(\mathbf{x}) &= \alpha_1 \tau_{(1)}(\mathbf{x}) + \alpha_2 \tau_{(2)}(\mathbf{x}) + \cdots + \alpha_m \tau_{(m)}(\mathbf{x}) \\ &= \alpha_1 \sum_{j \in \mathcal{J}_1} w_{(1),\mathbf{j}} \overline{\psi}^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x}) \cdots + \alpha_m \sum_{\mathbf{j} \in \mathcal{J}_m} w_{(m),\mathbf{j}} \overline{\psi}^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x}). \end{aligned}$$

Where α_i is the weight of the i^{th} decision tree and Z_i is the set of all partitions with non-zero Fourier coefficients in its spectrum. Therefore, $f_e(\mathbf{x}) = \sum_{j \in \mathcal{J}} w_{(e),\mathbf{j}} \overline{\psi}^{\overline{\lambda}}_{\mathbf{j}}(\mathbf{x})$, where $w_{(e),\mathbf{j}} = \sum_{i=1}^{m} \alpha_i w_{(i),\mathbf{j}}$ and $\mathcal{J} = \cup_{i=1}^{m} \mathcal{J}_i$. Therefore, the Fourier spectrum of $f_e(\mathbf{x})$ (a linear ensemble classifier) is simply the weighted sum of the spectra of the member trees.

Consider the matrix D where $D_{i,j} = \tau_{(j)}(\mathbf{x_i})$, where $\tau_{(j)}(\mathbf{x_i})$ is the output of the tree $\tau_{(j)}$ for input $\mathbf{x_i} \in \Omega$. D is an $|\Omega| \times m$ matrix where $|\Omega|$ is the size of the input domain and m is the total number of trees in the ensemble.

An ensemble classifier that combines the outputs of the base classifiers can be viewed as a function defined over the set of all rows in D. If $D_{*,j}$ denotes the j-th column matrix of D then the ensemble classifier can be viewed as a function of $D_{*,1}, D_{*,2}, \cdots D_{*,m}$. When the ensemble classifier is a linear combination of the outputs of the base classifiers we have $F = \alpha_1 D_{*,1} + \alpha_2 D_{*,2} + \cdots \alpha_m D_{*,m}$, where F is the column matrix of the overall ensemble-output. Since the base classifiers may have redundancy, we would like to construct a compact low-dimensional representation of the matrix D . However, explicit construction and manipulation of the matrix D is difficult, since most practical applications deal with a very large domain.

In the following we demonstrate a novel way to perform a PCA of the matrix D, defined over the entire domain. The approach uses the Fourier spectra of the trees and works without explicitly generating the matrix D.

The following analysis will assume that the columns of the matrix D are mean-zero. This restriction can be easily removed with a simple extension of the analysis. Note that the covariance of the matrix D is $D^T D$. Let us denote this covariance matrix by C. The (i, j)-th entry of the matrix,

$$C_{i,j} = < D(*,i), D(*,j) > = < \tau_{(i)}(\mathbf{x}), \tau_{(j)}(\mathbf{x}) > = \sum_{\mathbf{p}} w_{(i),\mathbf{p}} w_{(j),\mathbf{p}} = < \mathbf{w}_{(i)}, \mathbf{w}_{(j)} > \quad (1)$$

Now let us the consider the matrix W where $W_{i,j} = w_{(j),(i)}$, i.e. the coefficient corresponding to the i-th member of the partition set $\mathcal{J}$ from the spectrum of the tree $\tau_{(j)}$. Equation 1 implies that the covariance matrices of D and W are identical. Note that W is an $|\mathcal{J}| \times m$ dimensional matrix. For most practical applications $|\mathcal{J}| << |\Omega|$. Therefore analyzing W using techniques like PCA is significantly easier. The following discourse outlines a PCA-based approach.

PCA of the matrix W produces a set of eigenvectors which in turn defines a set of Principal Components, $V_1, V_2, \cdots V_k$. Let $\gamma_{(j),q}$ be the j-th component of the q-th eigenvector of the matrix $W^T W$.

$$V_q = \sum_{j=1}^{n} \gamma_{(j),q} D(*,j) = \left[\sum_{\mathbf{i}} a_{\mathbf{i},q} \overline{\psi}^{\lambda}_{\mathbf{i}}(\mathbf{x}) \right]_{\mathbf{x} \in \Omega}.$$

Where $a_{\mathbf{i},q} = \sum_{j=1}^{n} \gamma_{(j),q} w_{(j),\mathbf{i}}$. The eigenvalue decomposition constructs a new representation of the underlying domain where the feature corresponding to column vector V_q is $v_q = \sum_{\mathbf{i}} a_{\mathbf{i},q} \overline{\psi}^{\lambda}_{\mathbf{j}}(\mathbf{x})$ i.e., $V_q = [v_q]_{\mathbf{x} \in \Omega}$. Note that v_q is a linear combination of a set of Fourier spectra and therefore it is also a Fourier spectrum. Also note that V_q-s are orthogonal which is proved in the following.

The inner product between V_q and V_r for $q \neq r$ is, $< V_q, V_r > = < [v_q]_{\mathbf{x}}, [v_r]_{\mathbf{x}} > = 0$. Therefore, we conclude that the spectra corresponding to the orthonormal basis vectors V_q and V_r are themselves orthonormal. Let f_q and f_r be the functions corresponding to the spectra $\mathbf{a}_q$ and $\mathbf{a}_r$. In other words, $f_q(\mathbf{x}) = \sum_{\mathbf{i}} a_{\mathbf{i},q} \psi_{\mathbf{i}}(\mathbf{x})$ and $f_r(\mathbf{x}) = \sum_{\mathbf{i}} a_{\mathbf{i},r} \psi_{\mathbf{i}}(\mathbf{x})$. Therefore, we can also conclude that, $< V_q, V_r > = < \mathbf{a}_q, \mathbf{a}_r > = < f_q(\mathbf{x}), f_r(\mathbf{x}) >$. This implies that the inner product between the output vectors of the corresponding functions are also orthonormal to each other.

The principal components $V_1, V_2, \cdots V_k$ computed using the eigenvectors of the covariance matrix C are orthogonal to each other themselves. Since each of these principal components is a Fourier spectrum in itself we can always construct a decision tree from this spectrum using technique noted in Section 2 and detailed elsewhere [5]. Although the tree looks physically different from the Fourier spectrum, they are functionally identical. Therefore, the trees constructed from the principal components $V_1, V_2, \cdots V_k$ also maintain the orthogonality condition. These orthogonal trees now can be used to represent the entire ensemble in a very compact and efficient manner. The following section reports some experimental results.

Method of classification	Error Percentage
C4.5	24.5989 (%)
Bagging (40 trees)	20.85 (%)
Aggregated Fourier Trees (40 trees)	19.78(%)
Orthogonal Decision Trees	8.02(%)

Table 1. Classification error for SPECT data.

4. Experimental Results

This section reports the experimental performance of orthogonal decision trees on the Single Proton Emission Computed Tomography (SPECT) data set. [1] The following four different experiments were performed to test classification accuracies: (1) **C4.5 classifier**, (2) **Bagging**, (3) **Aggregated Fourier Tree:** The training set was uniformly sampled, with replacement and C4.5 decision trees were built on each sample. A Fourier representation of each tree was obtained(preserving approximately 99(%) of the total energy), and these were aggregated with uniform weighting, to obtain a Fourier tree. The classification accuracy of this aggregated Fourier tree was reported. (4) **Orthogonal Decision Tree:** The matrix containing the Fourier coefficients of the decision trees (obtained from step 3 above) was subjected to principle component analysis. Orthogonal trees were built, corresponding to the significant components and they were combined using an uniform aggregation scheme. The accuracy of the orthogonal trees was reported.

We report classification accuracies using 10-fold cross-validation and tree complexity, in terms of the number of nodes in the tree. In case of the orthogonal trees, tree complexity refers to the average number of nodes in each tree projected along significant components.

The dataset of 267 SPECT image sets (corresponding to different patients) was processed to extract 22 binary feature patterns that summarize the original SPECT images. The training data set consisted of 80 instances while the test data consists of 187 instances. The class label is binary.

The Figure 1 illustrates four decision trees built on the uniformly sampled training data set(each of size 20). The first decision tree, has a complexity 7 and considers attribute 7, 15 and 10 as ideal for splits. Before pruning, only one instance is mis-classified giving an error of 5(%). After pruning, there is no change in structure of the tree. The estimated error percentage is 28.5(%). The second, third and fourth decision trees have complexities 7, 5, and 3 respectively. An illustration of an orthogonal decision tree obtained from the first principle component, is shown in Figure 2.

Table 1 illustrates the error percentage obtained in each of the four different classification schemes. For bagging,

[1] Available from the University of California Irvine, Machine Learning Repository.

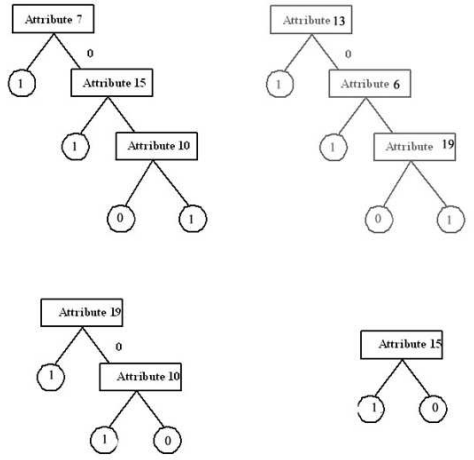

Figure 1. Decision trees built from the four different samples of the SPECT data set.

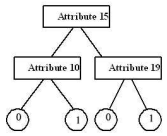

Figure 2. An orthogonal Decision Tree.

only 40 trees are used in the ensemble, since this gives the best classification accuracy for the data set.

For orthogonal trees, the coefficient matrix was projected onto the first five most significant principal components. The equivalent eigenvectors captured, 99.4416(%), 0.3748(%), 0.0925(%), 0.0240(%), 0.0156(%) of the variance respectively.

Table 2 illustrates the tree complexity for this data set. The aggregated Fourier tree and the orthogonal trees were found to be smaller in complexity, thus reducing the complexity of the ensemble.

5 Conclusions

This paper introduced the notion of orthogonal decision trees and offered a methodology to construct them. Orthogonal decision trees are functionally orthogonal to each other and they provide an efficient redundancy-free representation of large ensembles that are frequently produced by techniques like Boosting [4, 3], Bagging[1], Stacking [10], and random forests [2].

Method of classification	Tree Complexity
C4.5	13
Bagging (average of 40 trees)	5.06
Aggregated Fourier Trees (40 trees)	3
Orthogonal Decision Tree	3

Table 2. Tree complexity for SPECT data.

The proposed approach exploits the earlier work done by the first author and his colleagues [5] which showed that the Fourier transform of decision trees can be efficiently computed and we can also compute the tree back from its Fourier spectrum [7]. Although, the paper considers the Fourier representation, this is clearly not the only available linear representation around. However, our prior work shows that it is particularly suitable for representing decision trees. This work also opens up several new possibilities. Linear systems theory offers many tools for analyzing properties like stability and convergence.

Acknowledgments

The authors acknowledge supports from NSF CAREER award IIS-0093353, NSF grant IIS-0203958, and NASA grant NAS2-37143.

References

[1] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

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

[3] H. Drucker and C. Cortes. Boosting decision trees. *Advances in Neural Information Processing Systems*, 8:479–485, 1996.

[4] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[5] H. Kargupta and B. Park. A fourier spectrum-based approach to represent decision trees for mining data streams in mobile environments. *IEEE Transactions on Knowledge and Data Engineering*, 16(2):216–229, 2002.

[6] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM*, 40:607–620, 1993.

[7] B. H. Park and H. Kargupta. Constructing simpler decision trees from ensemble models using fourier analysis. In *Proceedings of the 7th Workshop on Research Issues in Data Mining and Knowledge Discovery, ACM SIGMOD*, pages 18–23, 2002.

[8] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[9] W. N. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classificaiton. In *Seventh ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, 2001.

[10] D. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.