Fourier Representation for Meta-Level Analysis of Data Mining Models

Hillol Kargupta
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
1000 Hilltop Circle Baltimore, MD 21250, USA
Phone: (410) 455-3972, Fax:(410) 455-3969
E-mail: hillol@cs.umbc.edu

Abstract

This paper offers a linear representation-based approach for advanced meta-analysis of data mining models. The contribution is particularly relevant to data mining applications that produce ensemble of dynamically changing classifiers, clusters, and other models. Mining data streams, distributed data, and privacy-sensitive data are some examples where we often face this scenario. These ensembles are usually difficult to interpret and translate into useful knowledge. Even if the mining algorithm produces a single model, dynamically changing environments like data streams often produce a temporally varying model that needs to analyzed and understood in the context of past models. This may help in detecting changes in the underlying data distribution, checking the stability of the monitoring/mining system, identifying operating regimes, and triggering different monitoring models.

The paper offers a linear representation-based approach for representing and performing meta-analysis of discrete structure-based data mining models. It particularly considers the Fourier representation of discrete structures like decision trees as an example. It points out that Fourier spectrum of decision trees can be used to efficiently perform many useful operations like principle component analysis-based visualization, aggregation, construction of efficient redundancy-free representation of an ensemble of decision trees and detecting unstable behavior of the models, that are currently very difficult to perform, if not impossible.

1 Introduction

Many data mining applications that deal with environments like data streams, distributed data, and privacy-sensitive data, produce either an ensemble of models or a temporally varying model. The difference and similarities among the models may help in identifying changes in different properties of the data such as underlying distribution, regime, outliers, and many other aspects. Therefore, understanding the dynamics of the models is important in many data mining applications. There are also many techniques like Boosting [6, 4], Bagging[2], Stacking [25], and random forests [3] that produce an ensemble of classifiers and they are regularly applied to static data sets for increasing classifier accuracy.

This paper offers a methodology to perform advanced meta-analysis of discrete model structures generated by data mining techniques. It explores construction of an embedding of such structures in linear representation and provides a mechanism to perform a rich variety of algebraic operations on a collection of these data mining models. Today, the state-of-the-art in dealing with collection of models is primarily restricted to combining only the outputs (e.g. class label assigned by the classifier, cluster membership in case of a clustering problem) using various methods like the Bagging [2], Stacking [25], order statistics-based techniques [24], voting [21], mutual information-maximization [23].

This paper claims that we need to go beyond treating the model-ensembles as a blackbox. We need to analyze and understand the characteristics of the models in a common framework. This philosophical motivation behind this work is also very important since most large-scale practical data mining applications do tend to deploy various techniques that eventually produces a collection of heterogeneous models.

This paper considers a small but important component of the problem. It considers ensemble of decision trees [15]. Decision trees are very popular and are widely used in many data mining applications for classification. The paper develops a Fourier representation-based approach that is suitable for analyzing the dynamically changing single or ensemble of decision trees. However, the paper discusses only the later case explicitly since an ensemble of multiple time varying models can be viewed as a generalization of a single time varying model. This paper offers a methodology that allows us to perform many useful operations on a collection of decision trees for aggregating, understanding, and manipulating the ensemble. The specific contributions of this particular paper include the following:

1. a similarity-preserving transformation of an ensemble of decision trees,
2. principle component analysis-based visualization of an ensemble,
3. aggregation, and construction of efficient redundancy-free representation of an ensemble, and
4. detecting unstable behavior of the models.

Section 2 offers a framework for aggregating and manipulating ensemble of decision trees using Fourier basis. Section 3 describes a novel scheme for PCA-based visualization of an ensemble of decision trees. Section 4 presents an efficient eigenanalysis-based scheme for removing redundancy in a decision tree-ensemble. Section 5 discusses a technique to monitor the stability of the functional behavior of an ensemble using matrix perturbation theory. Section 6 presents some experimental results. Finally, Section 7 concludes the paper.

2 Linear Representations for Aggregating, and Understanding Ensemble of Models

An ensemble-classifier is a collection of classifiers that work together in order to perform the classification tasks. The member classifiers of an ensemble are called base classifiers. This section considers ensemble of classifiers represented using discrete structures and proposes a framework to aggregate, understand, and manipulate them using formal algebraic operations. It particularly considers linear representations of decision trees (e.g., CART[1], ID3[16], and C4.5 [17]) for demonstrating the possibility of going beyond the traditional ensembles that just combine the outputs of the models. This section considers decision trees since it is a popular technique to learn classifiers from data and it is represented by a discrete structure. Learning decision trees from distributed and stream data often produces large ensembles [5, 13, 18, 22]. The rest of this paper considers the Fourier representation of decision trees which allows efficient representation, aggregation, and manipulation of tree-ensembles.

2.1 Decision Trees as Numeric Functions

A decision tree defined over a domain of categorical attributes can be treated as a numeric function. First note that a decision tree is a function that maps its domain members to a range of class labels. Sometimes, it is a symbolic function where features take symbolic (non-numeric) values. However, a symbolic function can be easily converted to a numeric function by simply replacing the symbols with numeric values in a consistent manner.

Once the tree is converted to a discrete numeric function, we can also apply any appropriate analytical transformation as necessary. Fourier transformation is one such interesting possibility. Fourier representation of a function is a linear combination of the Fourier basis functions. The weights, called Fourier coefficients, completely define the representation. Each coefficient is associated with a Fourier basis function that depends on a certain subset of features defining the domain. This section reviews the Fourier representation of decision tree ensembles, introduced elsewhere [8, 10].

2.2 A Brief Review of the Fourier Basis

Fourier bases are orthogonal functions that can be used to represent any discrete function. In other words, it is a functionally complete representation. 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 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:

$$f(\mathbf{x}) = \sum_{\mathbf{j}} w_{\mathbf{j}} \overline{\psi}_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x})$$

where $w_{\mathbf{j}}$ is the Fourier Coefficient (FC) corresponding to the partition $\mathbf{j}$ and $\overline{\psi}_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x})$ is the complex conjugate of $\psi_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x})$;

$$w_{\mathbf{j}} = \sum_{\mathbf{x}} \psi_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x}) f(\mathbf{x}).$$

The Fourier coefficient $w_{\mathbf{j}}$ can be viewed as the relative contribution of the partition $\mathbf{j}$ to the function value of $f(\mathbf{x})$. Therefore, the absolute value of $w_{\mathbf{j}}$ can be used as the "significance" of the corresponding partition $\mathbf{j}$. If the magnitude of some $w_{\mathbf{j}}$ is very small compared to other coefficients, we may consider the $\mathbf{j}$-th partition to be insignificant and neglect its contribution.

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 $\sum_{\mathbf{j}} w_{\mathbf{j}}^2$. 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

$$< f_1(\mathbf{x}), f_2(\mathbf{x}) >= \sum_{\mathbf{x} \in \Omega} f_1(\mathbf{x}) f_2(\mathbf{x}).$$

The following section considers the Fourier spectrum of decision trees and discusses some of its useful properties.

2.3 Properties of Decision Trees in the Fourier Domain

This section considers the Fourier spectrum of decision trees with finite depths, bounded by some constant. The underlying functions in such decision trees can be represented by a constant depth Boolean AND and OR circuit (or equivalently AC^0 circuit). Linial et al. [11] noted that the Fourier spectrum of AC^0 circuit has very interesting properties and proved the following lemma.

Lemma 1 ***(Linial, 1993)*** *Let M and d be the size and depth of an AC^0 circuit. Then*

$$\sum_{\{\mathbf{j} \mid o(\mathbf{j}) > t\}} w_{\mathbf{j}}^2 \leq 2M 2^{-t^{1/d}/20}$$

where $o(\mathbf{j})$ denotes the order of the partition $\mathbf{j}$ and t is a non-negative integer. The term on the left hand side of the inequality represents the energy of the spectrum captured by the coefficients with order greater than a given constant t.

Lemma 1 essentially states the following property about decision trees: The energy captured by all high order Fourier coefficients is small. This is because the energy of the Fourier coefficients of higher order decays exponentially. This observation suggests that the spectrum of a Boolean decision tree (or equivalently bounded depth function) can be approximated by computing only a small number of low order Fourier coefficients. So Fourier basis offers an efficient numeric representation of a decision tree in the form of a linear function that can be easily stored and manipulated. The exponential decay property of Fourier spectrum also holds for non-Boolean decision trees. The complete proof is available elsewhere [14].

There are two additional important characteristics of the Fourier spectrum of a decision tree:

1. The Fourier spectrum of a decision tree can be efficiently computed [10].

2. The Fourier spectrum can be directly used for constructing the tree [14].

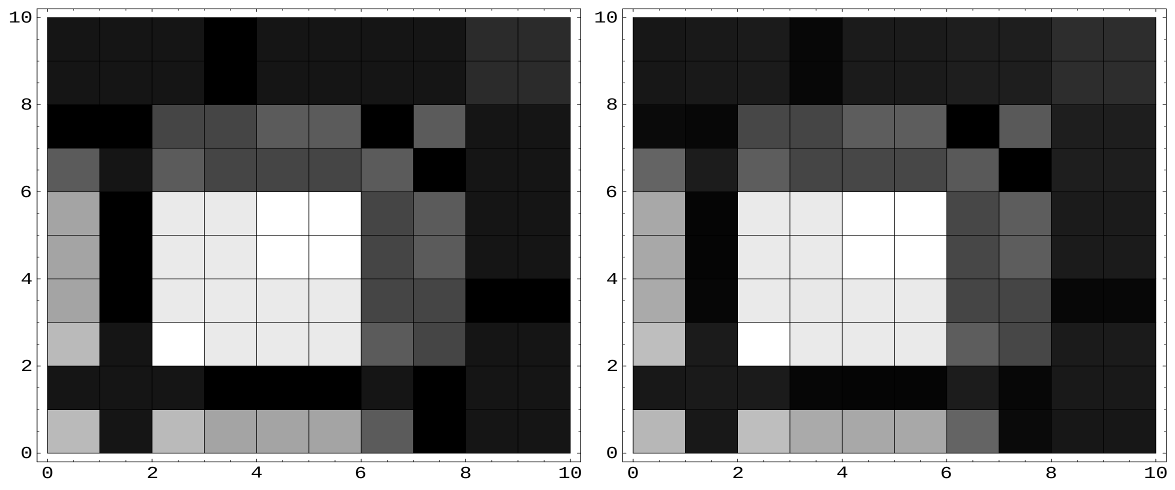

Figure 1: Inner product matrices: Computed using the (Left) output of the decision trees and (Right) the Fourier spectra of the respective trees.

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 lemma proves that the inner product between two such vectors is identical to the same in between their respective Fourier spectra.

Lemma 2 *Given two functions* $f_1(\mathbf{x}) = \sum_{\mathbf{j}} w_{(1),\mathbf{j}} \overline{\psi_{\mathbf{j}}}^{\lambda}(\mathbf{x})$ *and* $f_2(\mathbf{x}) = \sum_{\mathbf{j}} w_{(2),\mathbf{j}} \overline{\psi_{\mathbf{j}}}^{\lambda}(\mathbf{x})$ *in Fourier representation. Then* $< f_1(\mathbf{x}), f_2(\mathbf{x}) > = < \mathbf{w}_{(1)}, \mathbf{w}_{(2)} >$.

Proof:

$$\begin{aligned}
< f_1(\mathbf{x}), f_2(\mathbf{x}) > &= \sum_{\mathbf{x}\in\Omega} f_1(\mathbf{x}) f_2(\mathbf{x}) \\
&= \sum_{\mathbf{x}\in\Omega} \sum_{\mathbf{j},\mathbf{i}} w_{(1),\mathbf{j}} \overline{\psi_{\mathbf{j}}}^{\lambda}(\mathbf{x}) w_{(2),\mathbf{i}} \overline{\psi_{\mathbf{i}}}^{\lambda}(\mathbf{x}) \\
&= \sum_{\mathbf{j},\mathbf{i}} w_{(1),\mathbf{j}} w_{(2),\mathbf{i}} \sum_{\mathbf{x}\in\Omega} \overline{\psi_{\mathbf{j}}}^{\lambda}(\mathbf{x}) \overline{\psi_{\mathbf{i}}}^{\lambda}(\mathbf{x}) \\
&= \sum_{\mathbf{j}} w_{(1),\mathbf{j}} w_{(2),\mathbf{j}} \\
&= < \mathbf{w}_{(1)}, \mathbf{w}_{(2)} > .
\end{aligned}$$

■

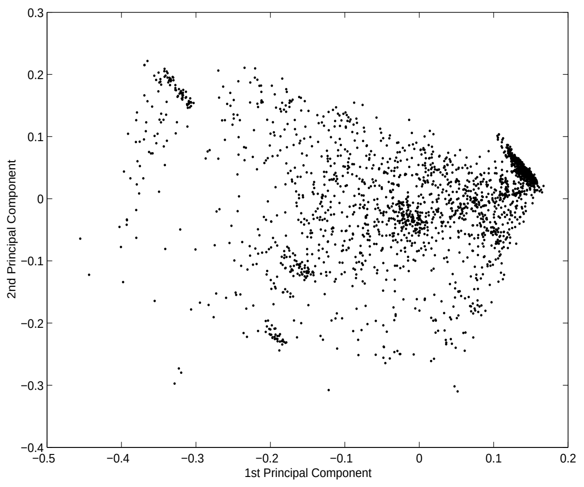

Figure 2: Visualization of an ensemble of decision trees using PCA. Each point represents a single decision tree.

The fourth step is true since Fourier basis functions are orthonormal. The Fourier spectrum of a decision tree offers a real valued representation that allows a wide range of different data analysis techniques for analyzing and understanding the ensemble of decision trees. The rest of this paper considers several such possibilities.

3 Visualizing Ensemble of Decision Trees

Visual inspection of ensemble models for identifying their relative similarities and dissimilarities is one of the basic needs for better understanding of an ensemble. This section offers a technique for visualizing decision tree-ensembles using the Principal Component Analysis (PCA) [7].

Consider a set of decision trees $\tau_{(1)}, \tau_{(2)}, \cdots, \tau_{(m)}$; let $\mathbf{w}_{(1)}, \mathbf{w}_{(2)}, \cdots \mathbf{w}_{(m)}$ be their respective spectra. In order to visualize the functional behavior of any given tree $\tau_{(i)}$ we need to somehow represent the vector $[\tau_{(i)}]$. The inner product between $[\tau_{(1)}]$ and $[\tau_{(2)}]$ provides a measure of their similarities. Therefore, the inner product matrix can be something useful for studying the ensemble. Unfortunately, for most real-life applications explicit operations using $[\tau_{(i)}]$ are not practical since the domain of $\tau_{(i)}$ is usually very large. However, Lemma 2 offers a practical way to solve this problem. Since Fourier spectrum preserves inner product we can operate in the Fourier domain efficiently without explicitly dealing with the $[\tau_{(i)}]$-s.

The inner product matrix is useful for measuring pair-wise similarity between trees. However sometimes we may need to represent the trees in a new embedding. PCA is a popular technique to construct a smaller dimensional representation of high dimensional data. Although PCA may not be directly applicable to the discrete structures—trees, the technique works fine with the representation of decision trees in the Fourier domain.

Let $\mathcal{J}$ be the union of all partitions with non-zero coefficients from all the m spectra under consideration and $|\mathcal{J}|$ is the cardinality of the set $\mathcal{J}$. Consider an arbitrary but fixed ordering of all the members of $\mathcal{J}$. Let us now define the matrix A such that $A_{i,j} = \mathbf{w}_{(i),(j)}$, where $\mathbf{w}_{(i),(j)}$ denotes the Fourier coefficient corresponding to the j-th partition in the ordering from the spectrum of the tree τ_i. After translating the column-means of the matrix A to zero, we get the new matrix $\mathbf{A}$. Therefore, $\mathbf{A}_{i,j} = A_{i,j} - \mu_j$, where μ_j is the mean of the j-th column of matrix A. This is a real valued $m \times |\mathcal{J}|$ matrix.

The covariance matrix of $\mathbf{A}$ is therefore $\mathbf{A}^T\mathbf{A}$. This is a symmetric matrix with an eigenvalue decomposition. A straight forward application of PCA on $\mathbf{A}$ and its subsequent projection along the dominant eigenvectors can be used to create a compact, smaller dimensional representation of the trees. Figure 2 shows a two-dimensional representation of an ensemble of trees using the two most dominant eigenvectors.

Visualization of trees is not the only thing we can do using the Fourier representation of trees. It also offers us a way to create linear combinations the trees and develop the notion of redundancy-free orthogonal trees. The following section outlines these possibilities.

4 Removing Redundancies from Ensembles

Combining the output of base models is the central issue in ensemble learning. There exist several well known techniques [2, 19, 25, 24] to do that. All of these techniques combine the output of the base classifiers in different ways. They do not structurally combine the classifiers themselves. The Fourier representation offers a unique way to do that.

The Fourier spectrum of a linear combination of decision tree classifiers can be computed by first computing the Fourier spectrum of every tree and then aggregating them using the chosen scheme for constructing the ensemble. 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,

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

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,

$$f_e(\mathbf{x}) = \sum_{j\in\mathcal{J}} w_{(e),\mathbf{j}}\overline{\psi}_{\mathbf{j}}^{\overline{\lambda}}(\mathbf{x}),$$

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 $f_e(\mathbf{x})$ (an linear ensemble classifier) is simply the weighted sum of the spectra of the member trees.

The base models of an ensemble often share redundancy resulting from similar observations noted at different sites. Continuous data stream environments may also introduce redundancy in the generated models because of the underlying periodicity in the data. Therefore, removing redundancy from the base models may be useful for creating the ensemble. The following part of this section explores a Fourier spectrum-based approach to do that.

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. We can try to construct an approximation of D using only the available training data. One such approximation of D and its Principal Component Analysis-based projection is reported elsewhere [12]. Their technique performs PCA of the matrix D, projects the data in the representation defined by the eigenvectors of the covariance matrix of D, and then performs linear regression for computing the coefficients $\alpha_1, \alpha_2, \cdots$, and α_m.

While the approach is interesting, it has a serious problem. First of all, the construction of an approximation of D even for the training data is computationally prohibiting for most large scale data mining applications. Moreover, this is an approximation since the matrix is computed only over the observed data set of the entire 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, Lemma 2, 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,

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

The third step is true by Lemma 2. 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

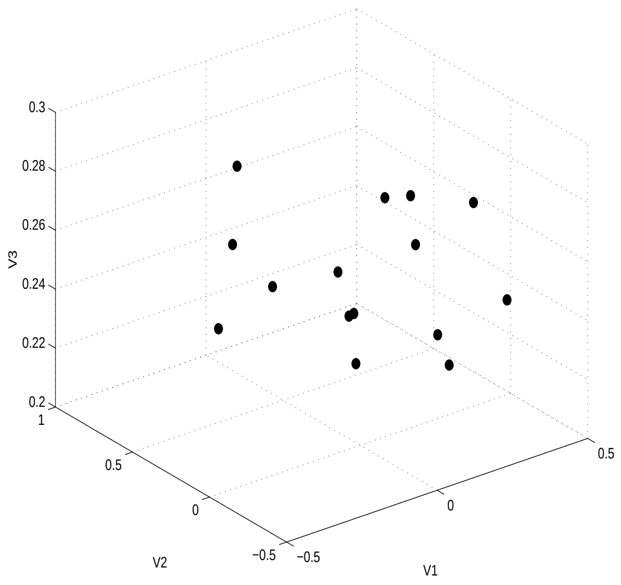

Figure 3: A collection of fifteen decision trees represented as a linear combination of the three orthogonal functions (V_q-s).

matrix W^TW.

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

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 $v_q = \sum_{\mathbf{i}} a_{\mathbf{i},q} \overline{\psi}_{\mathbf{j}}^{\lambda}(\mathbf{x})$ 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,

$$\begin{aligned} < V_q, V_r > &= < [v_q]_{\mathbf{x}}, [v_r]_{\mathbf{x}} > \\ &= \sum_{\mathbf{i},\mathbf{j}} a_{\mathbf{i},q} a_{\mathbf{j},r} \sum_{\mathbf{x}} \psi_{\mathbf{i}}(\mathbf{x}) \psi_{\mathbf{j}}(\mathbf{x}) \\ &= \sum_{\mathbf{i}} a_{\mathbf{i},q} a_{\mathbf{i},r} \\ &= < \mathbf{a}_q, \mathbf{a}_r > \\ &= 0. \end{aligned}$$

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 $f_r(\mathbf{x}) = \sum_{\mathbf{i}} a_{\mathbf{i},r}\psi_{\mathbf{i}}(\mathbf{x})$. Then by Lemma 2 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 above analysis offers a way to construct the Fourier spectra of a set of functions that are orthogonal to each other and therefore redundancy-free. These functions also define a basis and can be used to represent any given decision tree in the ensemble in the form of a linear combination.

Figure 3 reports an experiment with an ensemble of fifteen different trees. Some of these trees were generated in such a way that they share some functional similarities. In other words, some of the trees come up with the same class labels for the same domain member. This is very natural for most ensembles. We constructed the W matrix and performed the analysis presented in this section. The most dominant eigenvector captures 85% of the variance. The second and third most dominant eigenvectors capture 2.6% and 2.7% respectively of the variance. Each of the fifteen base models of an ensemble is represented in the form of a linear combination of those top three V_q-s and shown in a three dimensional graph (Figure 3).

Also note that most ensemble-based classifier learning techniques create a linear combination of the trees. Since the V_q-s define a linear basis, any such ensemble classifier can be constructed using a linear combination of V_q-s. We can also compute the coefficients corresponding to each V_q by simply performing linear regression. Visualization and construction of regression-based aggregation are not the only things that we can do using this approach. It can be used for studying many interesting properties of the classifier, particularly in the data stream and distributed data mining scenario. The following section considers yet another application of the framework and explores the problem of monitoring the functional stability of an ensemble classifier.

5 Stability Analysis of Ensembles

The stability issue of an ensemble presented in this section considers the changes in the functional behavior of an ensemble-classifier introduced by changing its base classifiers. The discussion is particularly relevant to temporally varying applications like mining data streams.

Consider a scenario where decision tree learning techniques are applied on a continuous stream of data resulting in the construction of a stream of different trees. At time t we have an ensemble of decision trees $\tau_{(1)}, \tau_{(2)}, \cdots, \tau_{(m)}$. At time $t+1$ we get another tree $\tau_{(m+1)}$. We would like to consider including it in the ensemble. We have several possibilities. We can either simply grow the size of the ensemble by including it in the pool. On the other hand, we can replace an existing base-classifier ($\tau_{(o)}$) using $\tau_{(m+1)}$. There are different schemes to select the appropriate $\tau_{(o)}$. For example the SEA technique reported elsewhere [22] replaces the worst performing base-classifier by the new one. On the other hand Fan [5] uses a moving window-based approach. For the sake of this discussion, let us assume that we replace some model in ensemble $\tau_{(o)}$ by $\tau_{(m+1)}$ since the model of infinitely growing ensemble does not work well in practice.

Now we wonder what kind of changes in the functional behavior of the system we should expect as a result of this change. In some applications, we may want to have a reasonably stable, robust classifier once we have observed the stream for a reasonable period of time and the underlying distribution is invariant. On the other hand, if the underlying distribution has changed then we sure want to detect that quickly and modify the ensemble content as necessary.

Since the overall ensemble classifier is a linear combination of the V_q-s we can write the classifier as follows:

$$f_e(\mathbf{x}) \quad = \quad [\psi_{\mathbf{j}}(\mathbf{x})]^T_{\forall \mathbf{j}} VB.$$

V is an orthogonal matrix where columns correspond to k different significant V_q-s and B is the corresponding $k \times 1$ coefficient matrix that defines the linear combination of V_q-s. The i-th entry of matrix B is nothing but the weight associated with V_i which defines the overall ensemble-classifier.

The matrix V directly depends on the eigenvectors of the matrix W^TW. On the other hand, the coefficient matrix B depends on the aggregation scheme. The matrix $[\psi_{\mathbf{j}}(\mathbf{x})]_{\forall \mathbf{j}}$ is simply the column matrix of Fourier basis function values. Therefore, for a given specific scheme to combine the base classifiers, the matrix V controls the functional behavior of the ensemble-classifier. If the eigenvectors of W^TW change dramatically then the behavior may change accordingly. Therefore, monitoring the changes in the dominant eigenvectors is very important for detecting major changes in the ensemble behavior. The following discussion presents a way to do that in a computationally efficient manner using matrix perturbation theory.

Let us note that W^TW is a symmetric matrix. Let γ_i and λ_i be the i-th ranking eigenvector and the corresponding eigenvalue such that $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_k$. Therefore γ_1 represents the most dominant eigenvector, γ_2 represents the second-most dominant eigenvector, and so on. Let W_n be the spectra-matrix produced by replacing the spectrum of $\tau_{(o)}$ by that of $\tau_{(m+1)}$ in the matrix W. Let us define $E = W_n^T W_n - W^TW$. Note that E is symmetric. In other words, the change in the corresponding covariance matrix W^TW is symmetric. Let $\lambda_1^{'} \geq \lambda_2^{'} \geq \cdots \lambda_k^{'}$ be the dominant eigenvalues of the perturbed matrix W_n; let $\gamma_1^{'} \geq \gamma_2^{'} \geq \cdots \gamma_k^{'}$ be the corresponding eigenvectors. Using matrix perturbation theory [20] we can write,

$$||\gamma_1 - \gamma_1^{'}|| \quad \leq \quad \frac{4||E||_F}{\delta - \sqrt{2}||E||_F} \tag{2}$$

$$|\lambda_1 - \lambda_1^{'}| \quad \leq \quad \sqrt{2}||E||_F \tag{3}$$

Where δ is the difference between the λ_1 and λ_2 of the matrix W^TW, sometimes called its eigengap. The Frobenius norm of a matrix E is defined as $||E||_F = (\sum_i \sum_j E_{ij}^2)^{1/2}$. Equation 2 assumes that $\delta > \sqrt{2}||\Delta^T\Delta||_F$.

Now note that,

$$
\begin{aligned}
||E||_F &= (2\sum_{i\neq o}(<\mathbf{w}_i, (\mathbf{w}_o - \mathbf{w}_{m+1})>)^2 + \\
&\quad (<\mathbf{w}_o, \mathbf{w}_o> - <\mathbf{w}_{m+1}, \mathbf{w}_{m+1}>)^2)^{1/2}
\end{aligned}
\tag{4}
$$

We can extend the above analysis for second-most dominant eigenvector using a similar approach by considering the complementary eigenspace comprised of all the eigenvectors except the most dominant one and then applying Equation 2. The approach can be further extended to eigenvectors with subsequent rankings.

Therefore in order to monitor the changes in the underlying basis that spans the entire classifier space, we simply need to do the following things:

1. Maintain the eigengap (δ) information.

2. Every time we decide to replace tree $\tau_{(o)}$ by another tree $\tau_{(m+1)}$ we simply compute $||E||_F$ using Equation 4 and then check the bounds given by Equations 2 and 3.

This gives us an upper bound of the change in the eigenvectors. If the bound is high enough then we may want to trigger necessary actions needed to deal with a change in the behavioral regime. Note that the test does not require frequent expensive computation of eigenvectors unless the tests predict a major change in the underlying basis. If the upper bound on the change in the dominant eigenvectors is large only then we may need to recompute the new basis and the eigengap information. This property is highly desirable for dealing with data stream scenarios where response-time is particularly critical.

6 Experimental Results

This section presents results from experiments performed using NASDAQ 100 stock quote data. The data set contains 20 features where each feature corresponds to a unique stock in NASDAQ. The original real-valued data is pre-processed and transformed to discrete Boolean data by encoding percentages of changes in stock quotes. We used the discretized value of Yahoo stock as the class label. A test data set of 112,000 instances is used to simulate the stream. A new decision tree is constructed from every 800 instances. The new tree replaces the oldest tree in the ensemble. Our objective is to monitor the changes in the dominant eigenvector of the ensemble. We can do that by continuously performing eigen analysis for every new ensemble. However, as we noted earlier the theoretical bounds offered by Equations 2 and 3 can be used instead of computing the new eigenstates every time the ensemble is changed. In this section we report the behavior of the bounds and also a relatively simple estimator for the changes that seems to work accurately for the data set considered here.

Figures 4 and 5 were generated by an experiment starting with an ensemble of 55 decision trees. Another 85 trees were used to replace one tree at a time in the ensemble for creating the temporally varying ensemble generated from the stream. In the experiment, we first computed the eigengap (δ). Each time the spectrum of a new tree replaces an existing one in the ensemble, we compute $||E||_F$ according to Equation 4, and then get the upper bounds and the real values of the perturbation in both dominant eigenvectors and eigenvalues using Equations 2 and 3. At the end of the experiment, the two figures were produced to illustrate those bounds and

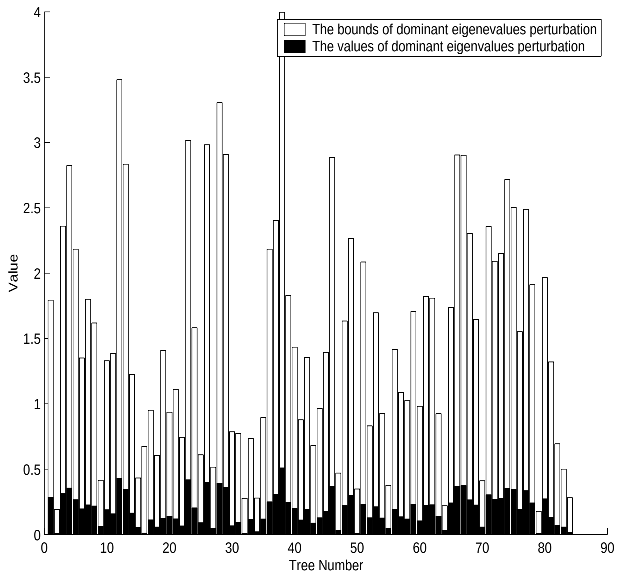

Figure 4: The actual changes and the theoretical upper bounds of the changes in the dominant eigenvalues corresponding to the eigenstates representing the ensemble at different times (tree number). Note that the content of the ensemble is changing over time.

real values. As the figures show, the bounds as given by Equations 2 and 3 are correct but very conservative. In real practice if we use these bounds then we will certainly be able to detect large changes in the eigenstates; however, since the bounds are conservative we are likely to trigger the eigenstate-recomputation process often even when the actual change is not that large.

Figures 6 and 7 illustrate the results of an experiment for mining over a stock data-stream where we estimate the changes in the eigenvectors and eigenvalues using a linear function of the upper bounds provided by Equations 2 and 3. The experimental set up was identical to the one used for the previous set of experiments. The only thing that is different in this case is the introduction of the scaling factors α_1 and α_2. These two factors are multiplied with the upper bounds provided by Equations 6 and 7 respectively.

In all the experiments reported here, we obtained these coefficients by simply dividing the the actual change in the eigenvectors (or eigenvalues) by the corresponding theoretical bound and averaging this over a few observations. In our experiments we always used the first 10 observations to compute these scaling factors. This is a relatively simple technique to estimate the scaling factors; but it seem to work very well for the data set we considered. The resulting scaling factors for the eigenvector and eigenvalue upper-bounds are 0.1490 and 0.1290 respectively. Figure 6 and 7 show the estimated bounds of the perturbation in dominant eigenvectors, and their actual perturbation values. As seen from the two graphs, our estimated bounds quite precisely estimated the actual changes in the eigenstates. The following section concludes this paper.

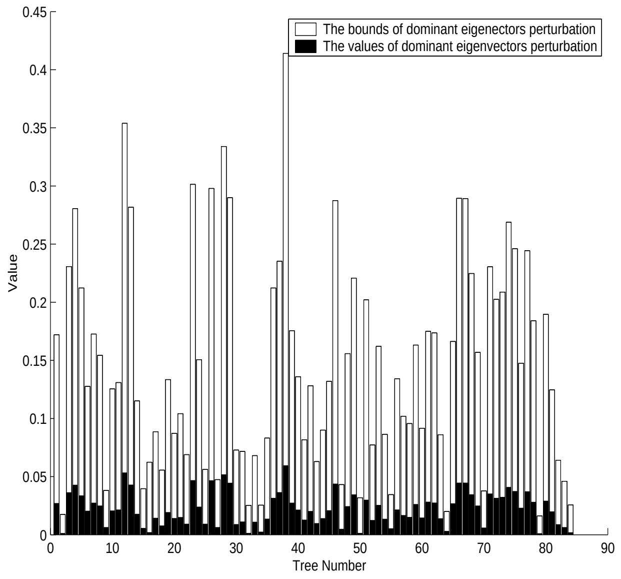

Figure 5: The actual changes and the theoretical upper bounds of the changes in the dominant eigenvectors corresponding to the eigenstates representing the ensemble at different times (tree number). Note that the content of the ensemble is changing over time.

7 Conclusions

This paper explores a relatively new area—meta-level analysis of data mining models for applications that generate a collection of such models. Data stream mining, distributed data mining, privacy-sensitive data mining are some of the common applications that usually face this situation. Most of the available techniques to deal with ensemble of data mining models treat the ensembles as a "blackbox" and do not go beyond providing a way to combine the output behavior of the base models. However, this alone does not serve the purpose at least for a data mining practitioner. Most real-life data mining applications require understanding the real "meaning" of the models in the context of the application domain. Moreover, in applications where the ensemble is changing over time, it is imperative to be able to efficiently represent, access, and monitor the models.

This paper pointed out that the Fourier basis is a very appropriate representation for decision trees that allows many interesting and useful operations on ensembles in a computationally efficient manner. Earlier work of the author [9, 8] showed that the Fourier representation of decision trees can be effectively used for physically aggregating multiple tree-structures into a single spectrum using a non-heuristic-based approach. Our earlier work also showed that the Fourier spectrum of a decision tree can be computed efficiently and it is also possible to compute the tree back from its Fourier spectrum [14].

This paper further extends our previous work and shows many different applications of the framework. It first showed that the Fourier transformation preserves the inner product between a pair of decision trees. It also demonstrated that a PCA-based dimension reduction of the Fourier representation of the ensemble may be suitable for low dimensional visualization of an ensemble. Next it identified a way to remove redundancy from an ensemble for constructing a basis set of orthogonal functions that spans the entire ensemble. This allows us to construct

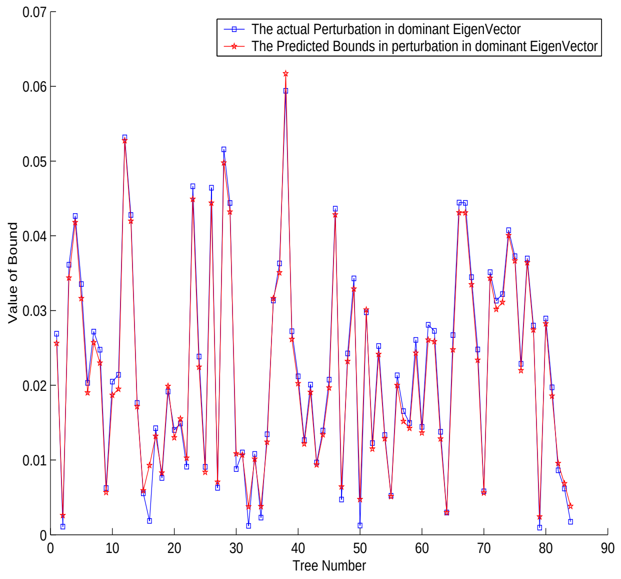

Figure 6: The changes in the most dominant eigenvector and its estimated upper bounds for a time varying decision tree ensemble. The estimated Bound is computed by multiplying the bound with an estimated scaling factor $\alpha_1 = 0.1490$.

the overall ensemble classifier in much more efficient manner with little redundancy in the representation. This offers an efficient representation of the ensemble, often needed for fast response in many real-time data mining applications. This also allows a meaningful way to visualize the trees in a low dimensional space. Finally, the paper considers the problem of detecting significant changes in the performance of an ensemble due to changes in its content. This is particularly important for the time varying applications like data stream mining. It offers a computationally efficient way to detect significant changes in the underlying basis without necessarily performing eigen-analysis every time.

Although, the paper considers the Fourier representation, this is clearly not the only available linear representation around. Data mining applications deal with many other discrete structures (e.g. graphs) that can also benefit from appropriate linear decompositions. Eigenvectors and Wavelets are other interesting choices for representing ensembles that need further investigations.

Acknowledgments

The authors acknowledge supports from the United States National Science Foundation CAREER award IIS-0093353, NASA (NRA) NAS2-37143, TRW Research Foundation, and TEDCO, Maryland Technology Development Center. The author would also like to thank Jessica Ryan for producing Figure 3, Ligong Wang for generating Figures 4 & 5, and Suraj Amonkar for producing Figures 6 & 7.

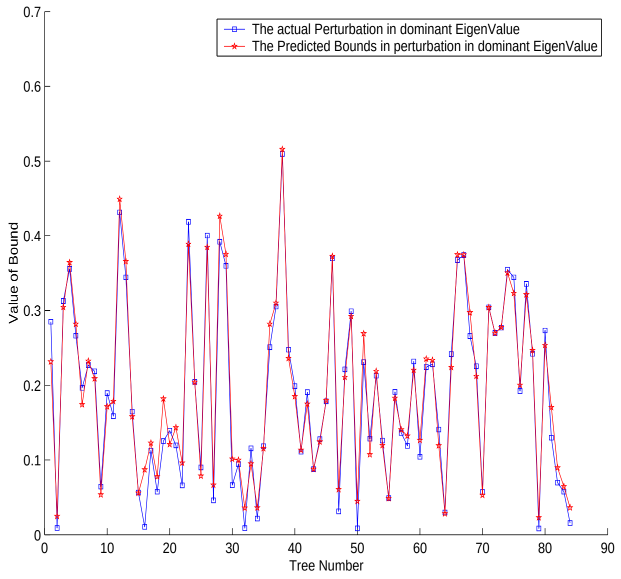

Figure 7: The changes in the most dominant eigenvalue and its estimated upper bounds for a time varying decision tree ensemble. The estimated Bound is computed by multiplying the bound with an estimated scaling factor $\alpha_2 = 0.1290$.

References

[1] L. Breiman, J. H. Freidman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

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

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

[4] Harris Drucker and Corrina Cortes. Boosting decision trees. *Advances in Neural Information Processing Systems*, 8:479–485, 1996.

[5] Wei Fan, Sal Stolfo, and Junxin Zhang. The application of adaboost for distributed, scalable and on-line learning. In *Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, California, 1999.

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

[7] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24, 1933.

[8] H. Kargupta and B. Park. Mining time-critical data stream using the Fourier spectrum of decision trees. In *Proceedings of the IEEE International Conference on Data Mining*, pages 281–288. IEEE Press, 2001.

[9] H. Kargupta, B. Park, D. Hershberger, and E. Johnson. Collective data mining: A new perspective towards distributed data mining. In *Advances in Distributed and Parallel Knowledge Discovery, Eds: Kargupta, Hillol and Chan, Philip*. AAAI/MIT Press, 2000.

[10] H. Kargupta, B. H. Park, S. Pittie, L. Liu, D. Kushraj, and K. Sarkar. Mobimine: Monitoring the stock market from a PDA. *ACM SIGKDD Explorations*, 3(2):37–46, January 2002.

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

[12] Christoper J. Merz and Michael J. Pazzani. A principal components approach to combining regression estimates. *Machine Learning*, 36(1–2):9–32, 1999.

[13] B. Park, Ayyagari R., and H. Kargupta. A Fourier analysis-based approach to learn classifier from distributed heterogeneous data. In *Proceedings of the First SIAM Internation Conference on Data Mining*, Chicago, US, 2001.

[14] B. H. Park and H. Kargupta. Constructing simpler decision trees from the Fourier spectrum of ensemble models: Theoretical issues and application in mining data streams. In communication (Shorter version published in SIGMOD DMKD'02 Workshop, 2002.

[15] J. Ross Quinlan. Learning efficient classification procedures. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach.* Tioga Press, Palo Alto, CA, 1983.

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

[17] J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kauffman, 1993.

[18] J. Ross Quinlan. Bagging, boosting and C4.5. In *Proceedings of AAAI'96 National Conference on Artificial Intelligence*, pages 725–730, 1996.

[19] Padhraic Smyth and David Wolpert. Linearly combining density estimators via stacking. *Machine Learning*, 36(1–2):59–83, 1999.

[20] W. Stewart and J. Sun. *Matrix Perturbation Theory.* Academic Press, 1990.

[21] S. Stolfo et al. Jam: Java agents for meta-learning over distributed databases. In *Proceedings Third International Conference on Knowledge Discovery and Data Mining*, pages 74–81, Menlo Park, CA, 1997. AAAI Press.

[22] W. Nick Street and YongSeog 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.

[23] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining partitionings. In *Proceedings the 18th National Conference on Artificial Intelligence (AAAI)*, July, Edmonton, Canada, 2002. AAAI.

[24] K. Tumer and J. Ghosh. Robust order statistics based ensemble for distributed data mining. In *Advances in Distributed and Parallel Knowledge Discovery, Eds: Kargupta, Hillol and Chan, Philip.* MIT, 2000.

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