Density Estimation Trees

Parikshit Ram
Georgia Institute of Technology
Atlanta GA, 30332
p.ram@gatech.edu

Alexander G. Gray
Georgia Institute of Technology
Atlanta GA, 30332
agray@cc.gatech.edu

ABSTRACT

In this paper we develop *density estimation trees* (DETs), the natural analog of classification trees and regression trees, for the task of density estimation. We consider the estimation of a joint probability density function of a d-dimensional random vector X and define a piecewise constant estimator structured as a decision tree. The integrated squared error is minimized to learn the tree. We show that the method is nonparametric: under standard conditions of nonparametric density estimation, DETs are shown to be asymptotically consistent. In addition, being decision trees, DETs perform automatic feature selection. They empirically exhibit the interpretability, adaptability and feature selection properties of supervised decision trees while incurring slight loss in accuracy over other nonparametric density estimators. Hence they might be able to avoid the curse of dimensionality if the true density is sparse in dimensions. We believe that density estimation trees provide a new tool for exploratory data analysis with unique capabilities.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Models; G.3 [**Probability and Statistics**]: Nonparametric statistics

General Terms

Algorithms, Experimentation

Keywords

Decision trees, density estimation, data analysis

1. INTRODUCTION

Three most fundamental tasks of machine learning are classification, regression, and density estimation. Classification and regression are instances of supervised data analysis where a training set of examples is provided. The learning task is to estimate a function using the training set which also performs well on a test set. The third task, density estimation, is an instance of unsupervised learning. This is generally harder because one does not have any instance of the ground truth regarding the quantity being estimated. Decision trees [1] have been widely used in the supervised setting for classification and regression. *In this paper we introduce, derive, and explore the natural analog of classification trees and regression trees for the unsupervised task of density estimation.* To our knowledge this analogy has never been explored rigorously, though several other ideas for density estimation involving hierarchical schemes have been proposed. Our hope is that the unique advantages of decision trees in the supervised setting will transfer to the unsupervised setting, to give rise to a new nonparametric density estimation method with interesting capabilities.

Density estimation. The problem of density estimation can be defined as estimating an unknown distribution f on $\mathcal{X}$[1] with a close approximation $\hat{f}\colon \mathcal{X} \to \mathbb{R}_+$ given a set of N IID (independent and identically distributed) observations $\{X_1, X_2, \ldots, X_N\} \subset \mathcal{X}$ drawn from f. Estimating the probability density of the given data is a fundamental task in multivariate statistics. It often appears as a subroutine in other inference – for example, in classification, one is required to find $\hat{P}(C|X) = \frac{\hat{p}(X|C)P(C)}{\hat{p}(X)}$, where C is one of the possible K classes and $\hat{p}(X|C)$ is the class-conditional density of the data X [2, 3, 4]. It is also widely used for conducting exploratory analyses such as outlier or cluster detection.

Decision trees. Decision trees [1] were developed for classification [5] and regression [6]. They are primarily used for the purpose of supervised learning and have been widely and successfully used in nonparametric classification and regression. The piecewise constant estimators and the simplistic model of univariate binary splits of the data in decision trees lead to relatively *less accurate* estimators with unknown convergence rates [7]. Techniques such as bagging are used to boost this accuracy. More sophisticated methods with better accuracies and clearer asymptotic properties such as local linear regression [8] and support vector machines [9] exist for regression and classification respectively.

Nonetheless, anecdotally, decision trees remain one of the more widely used methods in practice [7], possibly even the most widely used. This is due to the inherent *intuitiveness*, *adaptability* and *interpretability* of the models learned. Moreover, they are capable of dealing with *mixed* categorical, discrete and continuous variables within one unified framework. They also perform *feature selection* and are easy to implement. These properties come at the cost of accuracy in prediction but still make decision trees very desirable and

[1]If all the attributes of the data are continuous then $\mathcal{X} \subset \mathbb{R}^d$

KDD'11, August 21–24, 2011, San Diego, California, USA.

Table 1: Characteristics of Methods for Density Estimation

Methods	np	Accuracy	Interpretability			Adaptability		Speed	
			COD	VI	Rules	ABD	AWD	Training	Query
MoG	×	low	✓	×	×	×	×	fast EM algorithm	very fast $O(d^2)$
Histogram	✓	low	×	×	×	×	×	fast $O(MdN)$	very fast $O(d)$
KDE (FBW)	✓	medium	×	×	×	×	×	slow $O(HdN^2)$	slow $O(dN)$
KDE* (FBW)	✓	medium	×	×	×	×	×	fast $O(HdN)$	medium $O(d \log N)$
KDE (VBW)	✓	highest	×	×	×	✓	✓	very slow $O(H^d dN)/q$	slow $O(dN)$
KDE* (VBW)	✓	high	×	×	×	✓	✓	slow $O(H^d d \log N)/q$	medium $O(d \log N)$
local r-KDE	✓	very high	×	×	×	✓	✓	fast CV step/q	slow $O(dN)$
global r-KDE	✓	medium/high	×	×	×	✓	×	t fast CV steps	slow $O(dN)$
DET	✓	medium	✓	✓	✓	✓	✓	slow LOO-CV	fast $O(D_T)$

practical. Another advantage of decision trees is the *efficient test querying* once the model (in this case a tree) has been trained.

Decision trees trade some accuracy for a simple, interpretable model. Considering that density estimation is widely used in exploratory analysis, giving up accuracy (which is not a useful concept in the absence of the ground truth) for understanding is acceptable.

1.1 Nonparametric Density Estimators

Histograms and kernel density estimators (KDEs) [10] are simpler nonparametric (np) techniques, whereas variants of KDEs such as rodeo-KDEs [11] and wavelet based methods [8] are more complex nonparametric estimators. A number of nonparametric Bayesian methods have also been developed for the task of density estimation [12]. Mixtures of Gaussians (MoGs) are widely used parametric density estimators. Even though the nonparametric methods have been shown to be significantly more accurate than the parametric methods and require fewer assumptions on the data than parametric methods, MoGs are widely used because of their interpretability as clusters (as well as their simple implementation).

Adaptability and Speed. Adaptability has two implications in density estimation: (1) *adaptable between dimensions (ABD)* – the estimation process should treat dimensions differently depending on their influence on the density (for example, addition of unimportant dimensions of uniform noise should not affect the estimator) (2) *adaptable within a dimension (AWD)* – determining regions of interest for a given dimension and adjusting the estimator to the local behavior of the density in that dimension (for example, regions with fast changing density vs. regions with flat densities).

MoGs are not known to adapt between or within dimensions. Fixed-bandwidth KDEs (FBW) are accurate nonparametric estimators but are not adaptable between or within dimensions because of the restriction of using a multivariate kernel with the same bandwidth in every dimension (spherical kernel) for every point. The bandwidth is chosen through leave-one-out cross-validation (LOO-CV) by picking the one with the best CV loss among the H different values tried. A significantly high value of H is required to obtain accurate results. LOO-CV for the naïve implementation of KDE takes $O(HdN^2)$. Spatial partitioning tree data structures have been used for fast-approximate computation of KDEs (KDE*s) [13, 14]. For cover-trees [15], this process requires $O(HdN)$ time [16].

Various KDEs with adaptive bandwidths have been developed but are complicated in practice. The nearest-neighbor KDEs [10] are locally adaptive by choosing bandwidths based on their k^{th}-nearest-neighbor distance. However, they are not adaptive over dimensions since they still use a spherical multivariate kernel. Moreover, the estimate does not represent a probability density function (pdf), and is discontinuous and heavy tailed. A truly adaptive (ABD) KDE would require an optimal bandwidth for each dimension (VBW). This means the kernel used would be elliptical, adapting to the density of the data. This makes the VBW-KDEs adaptable between dimensions. If H bandwidths are tried in *each* dimension, the training time required for the naïve implementation of KDEs would be $O(H^d dN)$. To make this locally adaptive for each dimension(AWD), the bandwidth estimation would be required for each query q. Even the faster methods (KDE*) would require $O(H^d d \log N)$ training time for each query. This makes the computational cost of CV for VBW-KDEs intractable even in medium dimensions.

A recent algorithm [11] uses *rodeo* [17] to greedily select adaptive bandwidths for a KDE in each dimension. Under certain sparsity conditions on the unknown density function, the resulting estimator is shown to empirically perform well and obtain near optimal convergence rate for KDEs. The local version (local r-KDE) of this estimator computes the optimal bandwidth in every dimension for every query point. The paper demonstrates rodeo's adaptability – implicitly identifying the dimensions of interest during the process of bandwidth selection as well as identifying regions of interest within a single dimension, selecting smaller bandwidths where the density varies more rapidly. However, this technique is expensive, requiring a rodeo step for every single query. The rodeo step is iterative and we are not aware of its runtime bound. However, it is empirically faster than LOO-CV for VBW-KDEs. The global-rodeo [11] (global r-KDE) improves efficiency by using a fixed bandwidth within each dimension, and estimating it by averaging the estimated bandwidths for t training points in every dimension. Hence, the resulting estimator loses local adaptability within a dimension (property 2 of adaptability). The training time now involves t rodeo steps instead of a rodeo step for each query. The accuracy of the estimate obviously depends on the t (larger number training queries imply more accurate estimates and slower training).

KDEs can be made adaptable, however, at the cost of computational complexity. Moreover, given the estimated bandwidths, the time taken for a single query using the naïve implementation of a KDE is $O(dN)$. The cover-tree data structure provides an approximate estimate in $O(d \log N)$ query time. For $O(N)$ queries, the query cost can be amor-

tized over the queries using a tree-data-structure over the queries, requiring a total time of $O(dN)$ [16][2].

Decision trees are known for their adaptability over the data (over features by implicit feature selection and within dimensions by choosing different leaf size in different regions). In this paper, we will demonstrate their adaptability for the task of density estimation. However, training decision trees using LOO-CV is an expensive step involving the growing of the tree, and the subsequent cost-complexity pruning [1, 7]. However this cost is amortized over the multiple empirically efficient queries. The query time for a decision tree estimator is $O(D_T)$ where D_T is the depth of a decision tree T. The worst case upper bound for D_T is $O(N)$, but is empirically seen to be much tighter in practice (much more closer to $O(\log N)$). Hence decision trees also bring efficiency to nonparametric density estimation at the cost of some accuracy. Histograms are also fast nonparametric estimators during training as well as query time. They require $O(MdN)$ training time for trying M different bin-widths and choosing the one with the best CV error. The query time is a blazing $O(d)$. However, they lack the adaptability of decision trees, and become prohibitively inaccurate as the number of dimensions increases.

Interpretability. We also propose the use of decision trees for density estimation to introduce interpretability in the nonparametric setting. The decision trees provide an interesting overlap between the accuracy of nonparametric methods and the simplicity and interpretability of parametric methods. Interpretability in density estimation can be useful in the providing the following information about the density of the data: (1) *detecting clusters and outliers (COD)* (2) *providing relative variable/dimension importance (VI)* – identifying dimensions that significantly affect the density (3) *providing simple univariate rules* – these simple rules can be used, for example, for directly specifying some chunk of a huge data set which sits on a DBMS database. In the case of a DET, this chunk might represent a cluster.

MoGs are known to detect clusters but are incapable of imparting any interpretability regarding the dimensions. The aforementioned method using rodeo [11] implicitly does identify dimensions of interest. However, it is hard to obtain relative importance of dimensions by just looking at the values of the estimated bandwidths. Moreover, cluster and outlier detection is complicated with KDEs – an exhaustive scan of the whole space provides regions of high (clusters) and low density (outliers). So KDEs are not interpretable density estimators. When accuracy is required with no concern over computation time, adaptive-KDEs are the method of choice. However, they might be too costly for exploratory analysis.

We attempt to achieve the properties of interpretability, while retaining adaptability, by using the CART-style univariate splits which gives rise to the uniquely readable rule structure of decision trees. This key property is not shared by the several other methods for density estimation involving hierarchical schemes which have been proposed over the years. Determining dimensions of interest is a direct by-product of the decision tree framework via the measure of *relative importance* of predictor variables [1][3]. While linear models such as linear regression and linear SVMs yield such a capability via examination of the linear coefficients, this capability is harder to find in nonparametric methods. A sorted list of the leaves of a DET can easily identify the clusters and the outliers in the data.

[2]It is important to note that the spatial partitioning tree building has a one-time cost of $O(dN \log N)$.

[3]We will define this measure in the technical section of the paper.

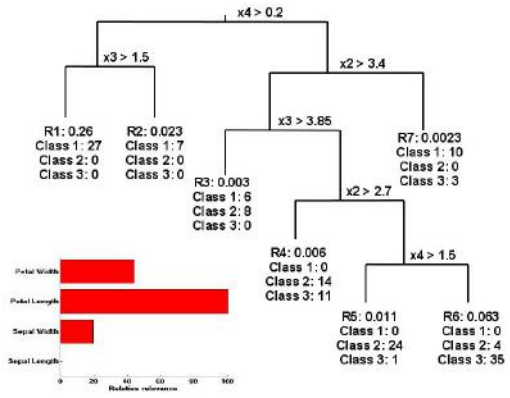


Figure 1: Density estimate in form of a decision tree and the relative variable importance.

All the aforementioned properties of these estimators are summarized in Table 1. We further motivate the desirability of interpretability of a decision tree in the following example:

Motivating example. Consider the density estimates for the **Iris** data set (150×4)[4] as a set of rules depicted in Figure 1. Each leaf node contains the piecewise constant density estimate. More explicitly, $\hat{f}(x) = 0.026\ \forall x \in R_1$ (say). Here the leaf $R_1 = \{x\colon x_4 \leq 0.2\ \&\ x_3 \leq 1.5\}$. We have also listed the class memberships of the points in the leaf nodes[5]. This figure also presents the relative importance of the attributes.

The tree representation lets us easily find regions of relatively high (R_1) and low densities (R_3). The subtree corresponding to the regions R_1 and R_2 contains purely class 1, representing the well-known linearly separable class in the Iris data. The chart containing the relative importance of the predictor variables implies that the *petal length* has the highest importance with regards to the density, whereas, the *sepal length* has no influence on the density (example of feature deletion).

Thus we see that density estimation in the decision tree framework provides much more information about the data than just the density estimates. There are several other cases where it would be useful to get this kind of information about the data: (1)*Astronomy* – performing density estimation on quasar data is a common task in astronomy [18]. Common features in the data are colors. The feature-importance analysis corresponding to the density will indicate the unimportant color features. This information can be useful in deciding which filters are useful in future analyses, thereby, conserving resources. (2)*Bioinformatics* – consider a metabolomics data set containing pa-

[4]We will continue to represent the size of data sets with N point with d attributes/dimensions in the form $(N \times d)$ throughout this paper.

[5]We intend to perform density estimation only on the points in the unsupervised setting. However, we provide class memberships of the points in the node to motivate a well known property of the Iris data set.

tient information with over 20000 features, where the task is to differentiate cancer patients from healthy patients [19]. Density estimation performed over all patients' profiles can reveal which features are responsible for the main variation within, say, cancer patients, or within healthy patients. This is different from feature selection for classification which would select features which most differentiate the patients with cancer from the rest. In addition, cluster and outlier detection performed on this data set will produce interesting and possibly crucial information.

Moreover, a representative partition of the state space by itself sheds light on the underlying structure of the distribution. Such information is particularly valuable in high dimensional problems where direct visualization of the data is difficult.

Remark. DETs are similar to variable bin-width histograms, but are restricted only to a hierarchical partitioning of the data (hence possibly having lower accuracy than the best possible variable bin-width histogram, although our experiments suggest otherwise).

1.2 Overview

The focus of this paper is to develop a rigorous decision-tree-structured density estimator and demonstrate the usefulness and interpretability of the same.

In Section 2, we discuss existing connections between density estimation and decision trees. In the following section, we define an estimator based on the decision tree and apply the decision-tree framework for density estimation using this estimator. We also provide certain asymptotic properties of the decision-tree-structured estimator. Section 4 contains experimental results displaying the performance of DETs with some comparisons with histograms and KDEs. The adaptability of DETs is demonstrated by using some synthetic data sets. High dimensional image data sets are used to demonstrate the DETs' interpretability and their application to classification. Training and querying time of DETs are compared with the fastest training and querying method for KDEs (KDE*s). In the final section, conclusions are discussed along with certain open questions.

2. FURTHER RELATED WORK

Decision trees have been used alongside density estimation in earlier works. For example, Kohavi, 1996 [20] and Smyth et al., 1995 [21] use decision trees for the supervised task of classification, and density estimation (with naïve Bayes classifiers and KDEs respectively) is used solely for the purpose of obtaining a smoother and more accurate classification and class-conditional probabilities in contrast to the piecewise constant non-smooth estimates of the standard classification trees. Decision trees been used for the task of diagnosing extrapolation [22] by building a classification tree differentiating the given data set from a random sample from a uniform distribution in the support of the data set. This tree provides a way to compute a measure for extrapolation. This measure can be indirectly used to compute the density, but the tree still performs classification. Decision trees have also been used in the supervised setting for estimating the joint probability (of the data and the label of the training examples) [23] for censored data by replacing the standard loss function (for example 0-1 loss or Gini-index) with the negative loglikelihood of the joint density.

The idea of having a nested hierarchy of partitions of the support of the data has been used for dicretization of univariate data [24]. The set of observations are partitioned on the basis of the density using the loglikelihood loss function. But the focus is solely on univariate data. Decision trees have also been used in an unsupervised setting to perform hierarchical clustering [25] but do not trivially translate to density estimation. Siedl et al., 2009 [26] propose a novel method for indexing the density model in the form of a tree (called Bayes tree) for fast access with desired level of accuracy. This tree is grown bottom-up from the estimate of a KDE for the whole data set. The Bayes tree uses MoGs to index the density at the intermediate levels (increasing the number of Gaussians with the depth). This tree successfully locates clusters in the data. However, being a MoG, it fails to determine relative relevance of the dimensions. Pólya trees [27] are hierarchical spatial partitioning trees but used as a prior for probability measures over the data space. Density estimation is done either by computing the posterior mean density given this prior or "learning" a fixed tree topology and computing the piecewise constant estimate conditional on this topology.

DETs are related in spirit, however, they will be learned by directly minimizing the density estimation loss and the estimates are finally obtained directly from the tree. Moreover, the aforementioned methods, though hierarchical in nature, do not share the interpretability and feature selection properties of decision trees, which are based on univariate splits.

3. DENSITY ESTIMATION TREE

This section provides the road map to perform density estimation using decision trees. We define an estimator and the corresponding loss function to be minimized during the training of the decision tree for continuous and mixed data. Following that, we explain the process of learning the optimal decision tree over the given sample and provide an asymptotic property of the DET.

3.1 Continuous Features

Definition 1. *The piecewise constant density estimate of the decision tree T built on a set $\mathcal{S}$ of N observations in $\mathbb{R}^d$ is defined as*

$$\hat{f}_N(x) = \sum_{l \in \tilde{T}} \frac{|l\,|}{N V_l} \mathbb{I}(x \in l\,) \tag{1}$$

where $\tilde{T}$ is the set of leaves of the decision tree T representing the partitions of the data space, $|l\,|$ is the number of observations of $\mathcal{S}$ in the leaf l, V_l is the volume of the leaf l within the d dimensional bounding box of $\mathcal{S}$ and $\mathbb{I}(\cdot)$ is the indicator function.

A decision tree T requires a notion of a loss function $R(T)$ which is minimized using a greedy algorithm to construct a tree on the set of observations. For the unsupervised task of density estimation, we consider the Integrated Squared Error (ISE) loss function [10]. The ISE gives a notion of overall distance between the estimated and the true density and is a favored choice in nonparametric density estimation for its inherent robustness in comparison to maximum-likelihood-based loss functions [8]. However, other distance functions such as the KL-divergence can be used as the loss function. We will explore this in the longer version of the paper.

The task of learning a DET would involve solving the following optimization problem:

$$\min_{\hat{f}_N \in \mathcal{F}_N} \int_X \left(\hat{f}_N(x) - f(x)\right)^2 dx \tag{2}$$

where $\mathcal{F}_N$ is the class of estimators of form in Definition 1 that can learned with any set of N observations. After expanding the square and the following Monte-Carlo substitution $\int_X \hat{f}_N(x) f(x) dx \approx \frac{1}{N}\sum_{i=1}^N \hat{f}_N(X_i)$ (where $\{X_i\}_{i=1}^N$ is the training set), the objective function in Eq. 2 is replaced by the following consistent plug-in estimator of the ISE [10][6]:

$$\min_{\hat{f}_N \in \mathcal{F}_N} \left\{ \int_X \left(\hat{f}_N(x)\right)^2 dx - \frac{2}{N}\sum_{i=1}^N \hat{f}_N(X_i) \right\} \tag{3}$$

Using the piecewise constant estimator from Definition 1 (which is constant within each leaf l), the objective function in Eq. 3 is replaced with the following:

$$\sum_{l \in \tilde{T}} \left\{ \int_l \left(\frac{|l|}{NV_l}\right)^2 dx - \frac{2}{N}\frac{|l|}{NV_l} \cdot |l| \right\} \tag{4}$$

by substituting

$$\hat{f}_N^2(x) = \sum_{l \in \tilde{T}} \left(\frac{|l|}{NV_l}\right)^2 \mathbb{I}(x \in l)$$

(since the cross terms in the expansion of $\hat{f}_N^2(x)$ vanish because of the indicator function) and simple calculation shows that

$$\sum_{i=1}^N \hat{f}_N(X_i) = \sum_{i=1}^N \sum_{l \in \tilde{T}} \frac{|l|}{NV_l} \cdot \mathbb{I}(X_i \in l) = \sum_{l \in \tilde{T}} \frac{|l|}{NV_l} \cdot |l|$$

By making the following substitution

$$\int_l \left(\frac{|l|}{NV_l}\right)^2 dx = \left(\frac{|l|}{NV_l}\right)^2 \int_l dx = \left(\frac{|l|}{NV_l}\right)^2 \cdot V_l,$$

the estimator of the ISE for DET has the following form:

$$\sum_{l \in \tilde{T}} \left\{ -\frac{|l|^2}{N^2 V_l} \right\} \tag{5}$$

Defining Eq. 5 as the error $R(T)$ of the tree, the greedy surrogate of the error for any node t (internal or otherwise) can be defined as

$$R(t) = -\frac{|t|^2}{N^2 V_t}. \tag{6}$$

The tree is grown in a top down manner by maximizing the reduction in this greedy surrogate of the error over the given observations.

3.2 Mixed Features

For density estimation over data with mixed features, we define a novel density estimator and a loss function involving ordinal and categorical data along with continuous data that can be used to learn DETs with mixed data:

[6]The term $\int_X (f(x))^2 dx$ is removed from the objective function since it is independent of the estimate and hence doesn't affect the optimum.

DEFINITION 2. *Let $\mathcal{S} \subset \mathbb{R}^d \times \mathbb{Z}^{d'} \times \mathbb{C}^{d''}$ with d real features, d' ordinal features and d'' categorical features. The piecewise constant density estimator of the decision tree T built on $\mathcal{S}$ is defined as*

$$\hat{f}_N(x) = \sum_{l \in \tilde{T}} \frac{|l| \cdot \mathbb{I}(x \in l)}{N \cdot V_{l_d} \cdot \prod_{j=1}^{d'_l} R_{l_j} \cdot \prod_{i=1}^{d''_l} M_{l_i}} \tag{7}$$

where V_{l_d} is the volume of the leaf l within the d dimensional bounding box of the real part of $\mathcal{S}$, R_{l_j} is the range of the ordinal values in the j^{th} of the d'_l ordinal dimensions present in l and M_{l_i} is the number of categories present in l for the i^{th} of the d''_l categorical dimensions present in l.

The error at a node t corresponding to the ISE is then obtained as

$$R(t) = -\frac{|t|^2}{N^2 \cdot V_{t_d} \cdot \prod_{j=1}^{d'_t} R_{t_j} \cdot \prod_{i=1}^{d''_t} M_{t_i}} \tag{8}$$

where V_{t_d} is the volume of the node t in the d real dimensions, R_{t_j} is the range of the ordinal values in the j^{th} of the d'_t ordinal dimensions present in t and M_{t_i} is the number of categories present in t for the i^{th} of the d''_t categorical dimensions present in t.

3.3 Tree Construction

We use the tree learning algorithm presented in Breiman, et al.,1984 [1]. The splitting, pruning and the cross-validation procedures are modified to work with this new loss function and for the unsupervised task of density estimation.

Splitting. For growing the tree, each node is split into two children. Let S be the set of all univariate splits.

DEFINITION 3. *[1] The best split s^* of a node t is the split in the set of splits S which maximally reduces $R(T)$.*

This is done by greedily reducing $R(t)$ for all the terminal nodes t of the current tree. Hence, for any currently terminal node, we need to find a split s^* for a node t into t_L and t_R such that

$$s^* = \arg\max_{s \in S} \{R(t) - R(t_L) - R(t_R)\} \tag{9}$$

where $|t| = |t_L| + |t_R|$. For continuous and ordinal dimensions, this optimization is performed by trying every $|t| - 1$ possible splits of the data in each of the dimensions. For categorical dimensions, the splits are performed in the manner similar to the CART model [1]. The splitting is stopped when the node size $|t|$ goes below a certain threshold (say $N_{\min}$).

Pruning. To avoid overfitting, we use the minimal cost-complexity pruning [1]. The regularized error of a subtree rooted at a node t is defined as

$$R_\alpha(t) = R(\tilde{t}) + \alpha \cdot |\tilde{t}| \tag{10}$$

where α is a regularization parameter (to be estimated through cross-validation) and $\tilde{t}$ is the set of leaves in the subtree rooted at t. The value of α is gradually increased and a subtree rooted at t is pruned for the value of α where $R_\alpha(t) = R_\alpha(\{t\})$, the regularized error of the pruned subtree. Since the size of the initial tree constructed by the splitting algorithm described previously is finite, the number of possible values of α at which a prune occurs is finite

and can be calculated efficiently. Hence we only need to select the optimal α from a finite set of values (See Section 3.3 in Breiman, et al., 1984 [1] for complete details).

Cross-validation. The leave-one-out cross-validation (LOO-CV) estimator of the density estimation loss function in Eq. 3 is given by

$$\hat{J}(\alpha) = \int_X \left(\hat{f}_N^{\alpha}(x)\right)^2 dx - \frac{2}{N}\sum_{i=1}^{N} \hat{f}_{(-i)}^{\alpha}(X_i) \qquad (11)$$

where $\hat{f}_N^{\alpha}$ is the estimator with the decision tree T pruned with a regularization parameter α, and $\hat{f}_{(-i)}^{\alpha}$ is the estimator with the decision tree $T_{(-i)}$ built without the training example X_i pruned with the regularization parameter α. This LOO-CV estimator is obtained from Silverman, 1986 [10] by switching the regularization parameter (replacing the bandwidth with α). The best sized tree is the tree T pruned with the parameter α^* such that:

$$\alpha^* = \arg\min_{\alpha} \hat{J}(\alpha) \qquad (12)$$

3.4 Asymptotic Properties

We show that the method is *nonparametric* – it is consistent under mild assumptions on the model class of the input distribution f. Consistency is typically shown [28] for a nonparametric estimator $\hat{f}_N$ obtained from a set of N observations by showing that

$$\Pr\left(\lim_{N\to\infty} \int_X \left(\hat{f}_N(x) - f(x)\right)^2 dx = 0\right) = 1. \qquad (13)$$

We prove the consistency of DETs on data with continuous features. The proof of consistency of the density estimator proposed in Definition 1 follows arguments similar to those used to show the consistency of regression trees [1] .

THEOREM 1. *The estimator $\hat{f}_N$ defined in Definition 1 satisfies Eq. 13.*

PROOF. Given a fixed positive integer d_1, let $\mathcal{B}$ denote the collection of all sets $t \subset X$ that can be described as the solution set to a system of d_1 inequalities, each of the form $b^T x \leq c$ for $b \in \mathbb{R}^d$ and $c \in \mathbb{R}$. Now in a decision tree T, every leaf $l \in \tilde{T}$ is the solution set of a system of d_1 inequalities of the form $b^T x \leq c$ with $b \in \mathbb{R}^d$ with just one entry equal to 1 and the rest of the entries equal to 0. Therefore, $\tilde{T} \subset \mathcal{B}$.

Let X_n, $n \geq 1$, be a random sample from a density function f on X. For $N \geq 1$, let $\hat{F}_N$ denote the empirical distribution of $X_n, 1 \leq n \leq N$, defined on a set $t \subset X$ by

$$\hat{F}_N(t) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}(X_n \in t) = \frac{|t|}{N} = \int_t \hat{f}_N(x)dx \qquad (14)$$

where $|t|$ is the number of random samples in the set $t \cap \{X_n\}_{n=1}^N$ and $\hat{f}_N(x)$ is the estimator given in Definition 1.

According to a general version of the Glivenko-Cantelli theorem [29],

$$\Pr\left(\lim_{N\to\infty} \sup_{t\in\mathcal{B}} |\hat{F}_N(t) - \int_t f(x)dx| = 0\right) = 1. \qquad (15)$$

By Eq.14 and 15, we get

$$\Pr\left(\lim_{N\to\infty} \sup_{t\in\mathcal{B}} |\int_t \hat{f}_N(x)dx - \int_t f(x)dx| = 0\right) = 1$$

$$\Rightarrow \Pr\left(\lim_{N\to\infty} \sup_{t\in\mathcal{B}} \int_t |\hat{f}_N(x) - f(x)|dx \geq 0\right) = 1.$$

Assuming that $\lim_{N\to\infty} \Pr(\text{diameter}(t) \geq \epsilon) = 0$, hence $\Pr\left(\lim_{N\to\infty} \int_t dx = 0\right) = 1$, we get the following with probability 1

$$\lim_{N\to\infty} \sup_{t\in\mathcal{B}} \int_t |\hat{f}_N(x) - f(x)|dx \quad \leq$$

$$\lim_{N\to\infty} |\hat{f}_N(x') - f(x')| \cdot \int_t dx \text{ for some } x' \in t \quad = \quad 0$$

This assumption is commonly used for the consistency of data-partitioning estimators [30] and is justified since as $N \to \infty$, the diameter of any leaf node would become smaller and smaller since the leaf node can only have a bounded number of points. Hence

$$\Pr\left(\lim_{N\to\infty} \sup_{t\in\mathcal{B}} \int_t |\hat{f}_N(x) - f(x)|dx = 0\right) = 1$$

$$\Rightarrow \Pr\left(\lim_{N\to\infty} \int_X \left(\hat{f}_N(x) - f(x)\right)^2 dx = 0\right) = 1.$$

Hence $\hat{f}_N$ satisfies Eq. 13. □

4. EXPERIMENTS

In this section, we demonstrate the performance of DETs under different conditions using synthetic and real data sets. Estimation accuracies of DETs are compared with existing nonparametric estimators. We exhibit the interpretability of DETs with two real data sets. Furthermore, density estimates of DETs are applied to classification and subsequent labelling accuracies are presented. Finally, the speed of training and querying DETs are presented on several real data sets and compared with an existing method. We only consider continuous and ordinal data in this paper for the lack of space. Experiments with categorical and mixed data will be presented in the longer version of the paper.

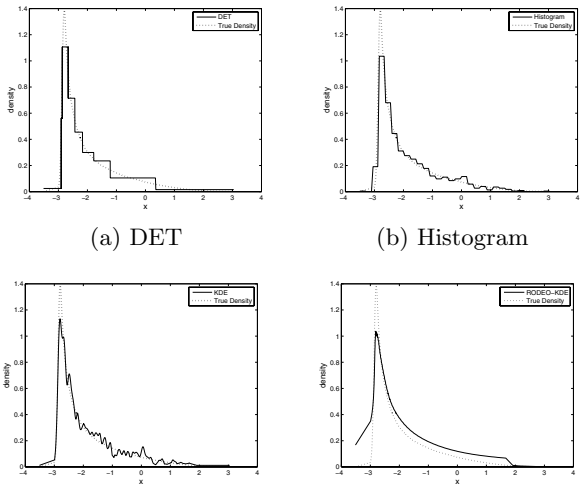
4.1 Estimation Accuracy: Synthetic Examples

For the unsupervised task of density estimation, estimation accuracy can only be computed on synthetic data sets. We choose the best-sized tree through LOO-CV. To compute estimation error on synthetic data for density queries, we use 2 measures: *Root Mean Squared Error* (RMSE) and *Hellinger Distance* (HD). We compare DETs only with other nonparametric density estimators since parametric estimators would involve choosing the model class (like choosing the number of Gaussians in MoGs). We report the comparison between DETs, histograms (bin-width selected by LOO-CV) and KDEs (using the Gaussian kernel) with the bandwidth selected by unbiased LOO-CV using the ISE criterion (FBW) and by local rodeo (local r-KDE) (VBW) on synthetic data sets.

Example 1. (Strongly skewed distribution in one dimension) This distribution is defined as

$$X \sim \sum_{i=0}^{7} \frac{1}{8}\mathcal{N}\left(3\left(\left(\frac{2}{3}\right)^i - 1\right), \left(\frac{2}{3}\right)^{2i}\right).$$

Figure 2 shows the estimated density functions by a DET, a histogram and two KDEs for a sample size $N = 1000$. Because of the high skewness of the density function, the KDE (FBW) and the histogram fail to fit the very smooth



(a) DET (b) Histogram

(c) KDE (FBW) (d) local r-KDE

Figure 2: The density estimates obtained with 1000 points for Example 1 using DET, Histogram and KDEs compared to the true density.

Table 2: Estimation errors for Histograms, KDEs and DETs with increasing number of observations

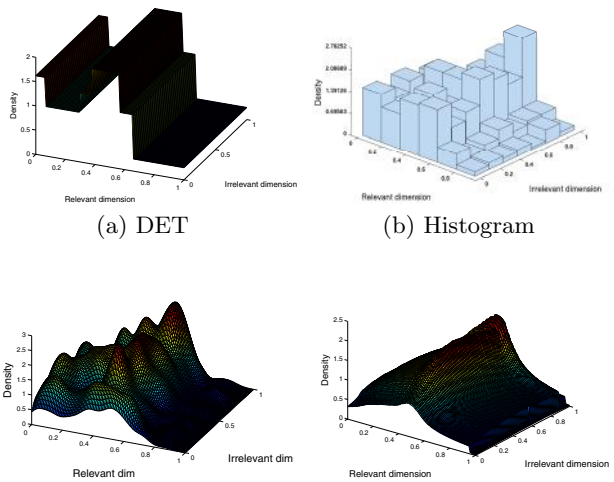
Type	N	Hist	KDE(FBW)	local r-KDE	DET
RMSE	10^2	0.2213	0.1748	0.1318	0.2548
	10^3	0.1158	0.0949	0.1339	0.1090
	10^4	0.0565	0.0235	0.0596	0.0527
HD	10^2	0.1292	0.2365	0.0967	0.1187
	10^3	0.0832	0.0684	0.2343	0.0278
	10^4	0.01	0.0130	0.1634	0.0072

tail. The fixed bandwidth/binwidth results in a highly wiggly estimate of the very smooth tail. The DET provides a piecewise constant estimate, but it adjusts to the different parts of the density function by varying the leaf size – the tree has small leaves closer to the spike where the density function changes rapidly, having larger leaves where the density function does not change rapidly. This demonstrates the adaptability of DETs within the same dimension. The local r-KDE exhibits the same adaptability by obtaining different bandwidths at different regions, hence capturing the smooth tail of the true density.

Table 2 provides the estimation errors with increasing sample sizes for the different methods. The RMSE values are fairly close and the HD values for the DETs are in fact better than that of KDE using unbiased LOO-CV. This can be attributed to the adaptive nature of DET with respect to the leaf sizes. The error values of local r-KDE are better than DET when the data set size is small (in case of larger data sets, the local method might be overfitting over this instance of the data set). The accuracies of the histogram estimators and DETs are comparable in this univariate example. But we will demonstrate in the next example that the performance of histograms decline with increasing dimensions.

Example 2. (Two dimensional data – Mixture of Beta distributions in one dimension, uniform in the other dimension) We create a (600×2) data set by sampling one dimension as a mixture of Beta distributions and the other dimension as a uniform distribution

$$X_1 \sim \frac{2}{3}\mathcal{B}(1,2) + \frac{1}{3}\mathcal{B}(10,10); X_2 \sim \mathcal{U}(0,1).$$



(a) DET (b) Histogram

(c) KDE (FBW) (d) local r-KDE

Figure 3: Perspective plots of the density estimates for a DET, a Histogram and two KDEs on the data set in Example 2.

Figure 3 shows the 2-dimensional density estimates of a DET, a histogram and two KDEs. The DET fits the irrelevant uniform dimension perfectly, while closely approximating the mixture of Beta distributions in the relevant dimension. The KDE with fixed bandwidth and the histogram estimator completely fail to fit the irrelevant dimension. The local r-KDE does a better job of fitting the irrelevant dimension compared to the KDE (FBW), but still does not entirely capture the uniform density.

4.2 Interpretability: Variable Importance

The decision-tree framework defines a *measure of relevance* for each of the predictor variables [1] as following:

DEFINITION 4. *The measure of relevance $\mathcal{I}_d(T)$ for each attribute X_d (the d^{th} attribute) in a decision tree T is defined as*

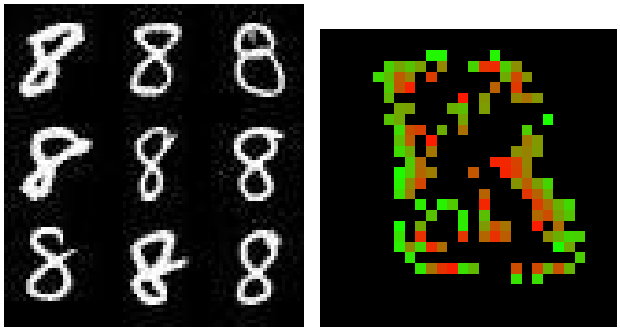
$$\mathcal{I}_d(T) = \sum_{t=1}^{|\tilde{T}|-1} \hat{\imath}_t^2 \mathbb{I}(d(t) = d) \tag{16}$$

where the summation is over the $|\tilde{T}| - 1$ internal nodes of the tree T, and the attribute $d(t)$ used to partition node t obtains $\hat{\imath}_t^2$ improvement in squared error over the constant fit in node t.

We use this measure to present the interpretability of the DETs on two real data sets.

Example 3. (Iris data set) We perform density estimation on this 4 dimensional data set and compute variable importance for each of the dimensions. Figure 1 displays the tree and the relative variable importance. The interpretation has been explained in Section 1.1.

Example 4. (MNIST digits – image data) Each image has 28-by-28 pixels providing a 784-dimensional data set. We perform density estimation on the 5851 images of the digit 8 (left panel of Figure 4). This is also an instance of performing density estimation with *ordinal data* since the pixel values are discrete. We compute the variable importance of each of the dimensions (pixels in this case). The right panel of Figure 4 displays the results obtained. The black pixels indicate pixels with zero variable importance in density



(a) Originals (b) Variable Importance

Figure 4: Relative importance of the predictor variables (pixels in this case) in density estimation for images of digit 8.

Table 3: Classification Accuracy: r-KDE vs. DET

Case	# Train	# Test	r-KDE	DET
1v7	240	121	0.93	0.91
2v7	237	119	0.97	0.87
3v8	238	119	0.81	0.81
5v8	237	119	0.86	0.72
8v9	236	118	0.75	0.77
All	1347	450	0.70	0.73

estimation. For the pixels with non-zero variable importance, the color transitions for green (indicating low relative variable importance) to red (indicating high relative variable importance). The marginal densities of many pixels are close to point masses. Hence the density does not vary at all in those dimensions, and the relative variable importance depicted in Figure 4 indicates that the DET is capable of capturing this property of the data.

4.3 Classification

As mentioned earlier, density estimation is a common subroutine in classification where the class-conditional density $\hat{p}(x|C)$ is to be estimated for the query point x using the training set X. The estimated class $\hat{C}(x) = \arg\max_C P(x|C) \cdot P(C)$. We compare the accuracy of classification between the density estimates computed by DETs and local r-KDEs.

Example 5. (Opt-digits – image data (1797×64) [31]) We conduct binary as well as multi-class classification. For binary classification, we consider the following cases: 1 vs. 7, 2 vs. 7, 3 vs. 8, 5 vs. 8, 8 vs. 9. We perform multi-class classification using all the classes (0-9). Table 3 lists the classification accuracies for the different tasks. In most cases, the accuracies are close for the two methods of density estimation, with the r-KDEs being more accurate than the DETs. But in the last two cases, the DETs outperform the r-KDEs. Moreover, the whole experiment (training and testing) with the DETs is much faster than the one using the local r-KDEs. For example, in the last experiment performing multi-class classification required ~ 8 seconds using DETs and ~ 500 seconds using r-KDEs[7].

4.4 Speed

We compare DETs to KDE*s(FBW) using the fast approximate dual-tree algorithm [32] with *kd*-trees. We use LOO-CV to estimate the optimal bandwidth for the KDE*s

[7]The DET is implemented in C++ and the rodeo-KDE is implemented using MATLAB on the same machine.

Table 4: Timings (in seconds) for training(T) and all queries(Q) with KDE*s and DETs

Data set	T(KDE*)	T(DET)	Q(KDE*)	Q(DET)
sj2	93.77	485.77	11.79	0.0026
Colors	101.68	427.02	12.11	0.0022
Bio	421.22	1213.84	5.7	0.0069
Pall7	1240.75	1719.26	10.67	0.0046
psf	18039.27	32456.43	77.48	0.975

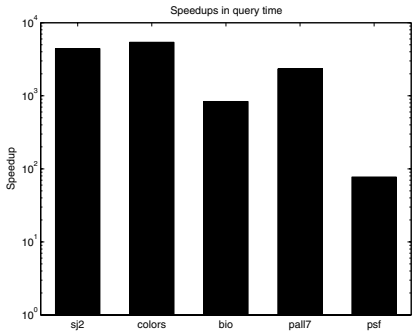


Figure 5: Query Time speedup for the DTE over the KDE. Note that the KDE is faster than the DTE in case of training.

(FBW). Real world data sets drawn from the SDSS repository [33] (SJ2 (50000×2), PSF (3000000×2)) and the UCI Machine Learning Repository [31] (Colors (50000×2), dimension reduced version of the Bio dataset (100000×5), Pall7 (100000×7)) are used. The query set is same as the training set in all the experiments and the KDE*s provide the leave-one-out estimates.

For the KDE*, we use around $H \approx 10$ bandwidths and chose the one with the best CV error. Usually, a much larger number of bandwidths are tried and the training times scale linearly with H. For the DETs, we use 10-fold cross-validation since we were not able to find an efficient way to conduct LOO-CV with decision trees and it would require prohibitively long time for these large data sets. This is one major limitation of this method. Howeover, for sufficiently large data sets, our experiments indicate that DETs trained with LOO-CV are only slightly more accurate than those with 10-fold CV.

The absolute timing values (in seconds) for training(T) and querying(Q) are given in Table 4. The speedups of the DETs over the KDE*s in test/query time are shown in Figure 5. As can be seen from the results, the training time for the decision tree algorithm is significantly larger than the LOO-CV for the KDE*s, but the DETs provide up to 3.5 orders of magnitude speedup in query time. The KDE*s are fast to train, but their work comes at test time. This experiment demonstrates the efficient querying of decision trees while also indicating that the task of training decision trees using LOO-CV is quite challenging.

5. DISCUSSION AND CONCLUSIONS

This framework of decision-tree-structured density estimation provides a new way to estimate the density of a given set of observations in the form of a simple set of rules which are inherently intuitive and interpretable. This framework has the capability of easily dealing with categorical, discrete as well as continuous variables and performs automatic fea-

ture selection. On having these rules, the density of a new test point can be computed cheaply. All these features stem from the fact that the density estimation is performed in the form of a decision tree. Along with that, the DETs are easily implementable like supervised decision trees. Although these advantages come with the loss of accuracy, the DETs are shown to be consistent, and hence are quite reliable in the presence of enough data.

Future directions of improvement include the reduction of the discontinuities in the density estimate because of the piecewise constant estimator and the boundary bias since the DETs put no mass outside the span of the data. Boundary bias is a common problem for almost all density estimators. One possible remedy is to use a normalized KDE at each of the leaves instead of using the piecewise constant estimator (similar to the approach in Smyth et.al. [21].

An analysis of the convergence rate of the DETs would quantify the amount of loss of accuracy to account for the simplicity of the estimator. Being effectively a variable bin-width histogram, we conjecture that the convergence rate for the DETs would be $o(N^{-2/3})$ for univariate data and better than histograms in higher dimensions since we demonstrate that the DETs can effectively model uninformative dimensions easily without requiring that extra number of points as imposed by the curse of dimensionality. Moreover, we are actively working on obtaining density dependent bounds on the depth of the DETs to quantify the runtimes for training and test query.

Overall, this method for density estimation has immediate application to various fields of data analysis (for example, outlier and anomaly detection)[8] and machine learning due to its simple and interpretable solution to the fundamental task of density estimation.

6. REFERENCES

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

[2] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[3] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[4] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley New York, 1973.

[5] J. H. Friedman. A Recursive Partitioning Decision Rule for Nonparametric Classification. *Transactions on Computers*, 1977.

[6] J. H. Friedman. A tree-structured approach to nonparametric multiple regression. *Smoothing Techniques for Curve Estimation*, 1979.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[8] L. Wasserman. *All of Nonparametric Statistics*. Physica-Verlag, 2006.

[9] C. Cortes and V. N. Vapnik. Support vector networks. *Machine Learning*, 1995.

[10] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986.

[11] H. Liu, J. Lafferty, and L. Wasserman. Sparse Nonparametric Density Estimation in High Dimensions Using the Rodeo. In *AISTATS*, 2007.

[12] P. Müller and F.A. Quintana. Nonparametric Bayesian data analysis. *Statistical Science*, 2004.

[13] A. G. Gray and A. W. Moore. Nonparametric density estimation: Toward computational tractability. In *SIAM Data Mining*, 2003.

[14] D. Lee and A. G. Gray. Fast high-dimensional kernel summations using the monte carlo multipole method. In *Advances in NIPS*, 2008.

[15] A. Beygelzimer, S. Kakade, and J.C. Langford. Cover Trees for Nearest Neighbor. *ICML*, 2006.

[16] P. Ram, D. Lee, W. March, and A. Gray. Linear-time algorithms for pairwise statistical problems. In *Advances in NIPS*. 2009.

[17] J. Lafferty and L. Wasserman. Rodeo: Sparse Nonparametric Regression in High Dimensions. *Arxiv preprint math.ST/0506342*, 2005.

[18] R. Riegel, A. G. Gray, and G. Richards. Massive-Scale Kernel Discriminant Analysis: Mining for Quasars. In *SIAM Data Mining (SDM)*, 2008.

[19] W. Guan, M. Zhou, C. Y. Hampton, B. B. Benigno, L. D. Walker, A. G. Gray, J. F. McDonald, and F. M. Fernandez. Ovarian Cancer Detection from Metabolomic Liquid Chromatography/Mass Spectrometry Data by Support Vector Machines. *BMC Bioinformatics Journal*, 2009.

[20] R. Kohavi. Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid. In *KDD*, 1996.

[21] P. Smyth, A. G. Gray, and U. M. Fayyad. Retrofitting decision tree classifiers using kernel density estimation. In *ICML*, 1995.

[22] G. Hooker. Diagnosing extrapolation: Tree-based density estimation. In *SIGKDD*, 2004.

[23] A.M. Molinaro, S. Dudoit, and M.J. Van der Laan. Tree-based multivariate regression and density estimation with right-censored data. *Journal of Multivariate Analysis*, 2004.

[24] G. Schmidberger and E. Frank. Unsupervised discretization using tree-based density estimation. *Lecture Notes in Computer Science*, 2005.

[25] J. Basak and R. Krishnapuram. Interpretable hierarchical clustering by constructing an unsupervised decision tree. *IEEE transactions on knowledge and data engineering*, 2005.

[26] T. Seidl, I. Assent, P. Kranen, R. Krieger, and J. Herrmann. Indexing density models for incremental learning and anytime classification on data streams. In *ICEDT: Advances in Database Technology*, 2009.

[27] W.H. Wong and L. Ma. Optional Pólya tree and Bayesian inference. *The Annals of Statistics*, 2010.

[28] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, pages 1065–1076, 1962.

[29] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 1971.

[30] L. Devroye, L. Györfi, and G. Lugosi. *A probabilistic theory of pattern recognition*. Springer Verlag, 1996.

[31] C.L. Blake and C.J. Merz. UCI repository of machine learning databases. 1998.

[32] A. G. Gray and A. W. Moore. '*n*-body' problems in statistical learning. In *Advances in NIPS*, 2000.

[33] R. Lupton, J.E. Gunn, Z. Ivezic, G.R. Knapp, S. Kent, and N. Yasuda. The SDSS Imaging Pipelines. *Arxiv preprint astro-ph/0101420*, 2001.

[8]We will demonstrate these applications in the longer version of the paper.