Doctoral Thesis

Submitted to the faculty at Virginia Tech

Lucas Roberts

Department of Statistics

Virginia Tech

rlucas7@vt.edu

September 3, 2014

Abstract

In this thesis we provide a historical survey of the statistics and computer science research in decision trees. We review both the greedy and Bayesian approaches to decision tree induction and propose a novel modification to previous Bayesian decision tree methods. Our approach facilitates covariate selection explicitly in the model, something not present in most previous research. We define a transformation that allows us to use priors from linear models tp facilitate covariate selection in decision trees. Using this transform, we modify many common approaches to variable selection in the linear model and bring these methods to bear on the problem of explicit covariate selection in decision tree models. We also provide theoretical guidelines, including a theorem, which gives necessary and sufficient conditions for consistency of decision trees in infinite dimensional spaces. Our examples and case studies use both simulated and real data cases with moderate to large numbers of covariates. The examples support the claim that the approach presented here is to be preferred in large dimensional datasets. Moreover, the approach approach shown here has, as a special case, the most commonly used previous approach.

Contents

List of Figures vi

List of Tables ix

List of Symbols x

List of Abbreviations xii

1 Introduction 1

1.1 Related Work 2

1.1.1 An Historical Path Through the Literature 3

1.1.2 A Brief Overview of Variable Selection Methods 8

2 Preliminaries 15

2.1 Greedy Induction 15

2.1.1 Impurity Functions 16

2.1.2 Induction 18

2.1.3 A Simple Example 19

2.2 Bayesian Approaches 21

2.2.1 The CGM approach 21

2.2.2 Integrated Likelihood 23

2.2.3 The Process Prior 24

2.2.4 Node Likelihoods and Priors 25

2.2.5 A Bayesian Zero Inflated Poisson Model 27

2.3 Previous Variable Selection 29

2.4 Derivations 30

2.4.1 ZIP Derivations 30

3 Bayesian Decision Tree Models 33

3.1 The Likelihood . . . 33
3.2 The Additive Logistic Transform . . . 34
3.3 The Tree Prior . . . 35

4 Dirichlet Variable Selection For Decision Trees: The DiVaS method 41
4.1 Related Work . . . 43
4.2 Model Details . . . 45
4.3 Ensuring Consistent Classifiers . . . 49
4.4 A Simulated Example . . . 54
4.4.1 Choosing Covariates . . . 58

5 A Case Study of the DiVaS Model 60
5.1 Discussion of Results . . . 62

6 Additive Logistic Variable Selection: The ALoVaS method 63
6.1 Normal Distributions Transformed to the Unit Simplex: ALT and ALN Dynamics. 66
6.2 A Simple Sampler Approach . . . 69
6.2.1 The General Strategy . . . 69
6.3 Slice Sampling Gibbs Updates . . . 72
6.4 A Stochastic Search Variable Selection Approach . . . 75
6.5 Parameter Expansions For the Lasso and Horseshoe Priors . . . 76
6.6 Regularization Posteriors . . . 77
6.7 The Model . . . 80
6.8 ALoVaS Algorithms Pseudocode . . . 82
6.8.1 The Lasso Prior . . . 83
6.8.2 The Horseshoe Prior . . . 83
6.8.3 The Stochastic Search Prior . . . 85

7 ALoVaS Examples and a Case Study 86
7.1 Simulated Examples . . . 86

7.1.1 Multimodal Simulation Study 86
7.1.2 Class Unbalancedness Simulation Study 89
7.2 Case Study 93
7.3 Conclusions 97

8 Synthesis: Comparing the DiVaS and ALoVaS Methods **99**
8.1 Theoretical differences and a simulation study 99
8.2 Practical differences and a simulation study 101
8.3 Recommendations 103

9 Discussion **105**

List of Figures

1 A plot of the three impurity functions: entropy (solid line), Gini (thatched line), and misclass probability (dots). 17

2 A simple decision tree 20

3 Illustrating the induction process 23

4 ALN plots with various multivariate normal parameters. Subfigure (a) contains multivariate standard normal draws. Subfigure (b) plots multivariate normal draws with zero mean and large independent variances. In subfigure (c), contains unit independent variances with mean vector $\mu = (-2, 2, 0)$. Finally, in subfigure (d) we add correlations while keeping the mean vector as a zero vector. 36

5 The binary heap node numbering system used in our simulations. 37

6 Plots of the log of the number of possible trees (solid line) less than or equal to a given depth n, compared to exponential (45 degree line), quadratic (dash-dotted line) and linear functions (even-dashed line). Note the vertical axis is on a log scale. 39

7 A shatter coefficient plot 51

8 Maximum depth of samplers trees with maximum depth set at 2. 52

9 Maximum depth of samplers trees with maximum depth set at 3. 52

10 Maximum depth of samplers trees with maximum depth set at 4. 52

11 Maximum depth of samplers trees with maximum depth set at 5. 52

12 Maximum depth of samplers trees with maximum depth set at 6. 52

13 Maximum depth of samplers trees with maximum depth set at 7. 52

14 Maximum depth of sampled trees with maximum depth set at 10. 53

15 A plot of the true DGP 55

16 The tree found by a greedy optimization. 55

17 The best tree using the weighted method. 56

18 The best tree found by the CGM method. 56

19 Covariate inclusion probabilities for the 100 covariate example. 57
20 Covariate inclusion probabilities for the 400 covariate example. 57
21 The greedy algorithm tree for the internet ads dataset 60
22 The CGM algorithm tree for the internet ads dataset 60
23 The weighted method tree for the internet ads dataset 60
24 Estimated covariate weights for the internet ads dataset 60
25 In (a) we plot the points in the Euclidean space and the primed points represent the points mapped to the probability simplex, represented in the plot by the surface of the triangular plane given by the equation $x + y + z = 1, x, y, z > 0$. The primes indicate points on the probability simplex after applying the additive logistic transformation to the split counts. In (b) we plot an example decision tree splitting on two of three possible covariates. 65
26 ALN plot with a zero mean vector 68
27 ALN plot with a negative one mean vector 68
28 ALN plot with a mean vector $(2, 0)^T$ 69
29 ALN plot with mean vector $(2, 0)^T$ 69
30 ALN plot with a mean vector of $(-2, 2)^T$ 70
31 ALN plot $\Sigma = \text{Diag}(0.01,100)$ 70
32 ALN plot Σ numerically singular 71
33 Similar to the case in Figure 31 but with the variances reversed 71
34 ALN plot with a zero vector mean and $\Sigma = \text{Diag}(100, 100)$ 72
35 ALN plot approximating the CGM model 72
36 Results for the zero mean prior 74
37 Results for the informative prior 74
38 The two optimal trees for the multimodal simulation study. 89
39 The points on the three dimensional simplex space we use as unbalanced classes in the simulation study. 90

40 Box plots of the errors in prediction with the hold out data from the sampled trees after burn-in in the unbalanced simulation study. The three box plots are the prediction errors for the sampled trees for SSVS, lasso, and horseshoe within each set of class probabilities. The fractions displayed with each set of four box plots label the correspondence with the proportion used to simulate each of the three response classes in the simulated data. Also these proportions are the points on the simplex displayed in Figure 39. Each letter in the figure, corresponding to the different box plots is a different sparsity value, corresponding to different sparsity levels in the number of covariates. Figure (a) corresponds to 50% sparsity, Figure (b) 60%, Figure (c) 70%, and Figure (d) 80%. 91

41 Box plots of the errors in prediction with the hold out data from the sampled trees after burn-in in the unbalanced simulation study. This is for the case of 90% (a) sparsity, 95% (b) sparsity, and (c) 99% sparsity. 92

42 The decision trees fitted to the internet ads dataset. The CGM (a) is much deeper than the remaining three trees fit using the ALoVaS method. The SSVS (b), Lasso (c) and horseshoe (d) all select the 1245th variable along with a second variable. The second variable differs in each of the three ALoVaS variants we use but the fact that the 1245th variable is common to all three gives further credence to the claim that a choice between the SSVS, lasso, or horseshoe types of ALoVaS is largely a choice of personal preference. 95

43 A plot of the Correlation space where non-zero prior probability is placed in the DiVaS and ALoVaS models. 99

44 write the caption here. 101

45 write the caption here. 101

List of Tables

1 A simple decision tree example data 19

2 Misclassification probabilities for $d = 100$. 56

3 Misclassification Probabilities of the three tree fitting methods for $d = 400$ 58

4 Empirical covariate selection with 100 simulations 59

5 Misclassification probabilities for internet ads dataset 61

6 The set J of covariates using Equation 57 61

7 Multimodal simulation study results, each entry represents the proportion of time the Markov chain stays in the two optimal trees given in Figure 38. The * indicates that 3 cannot be divided by a number to get 60% sparsity exactly. 88

8 Class unbalancedness simulation study results, each entry represents the MSE for that scenario and model. 94

List of Symbols

$\|\underline{x}\|_\nu = \left(\sum_{i=1}^d \|x_i\|^\nu\right)^{1/\nu}$	The ν norm of the vector $\underline{x}$. 10
$\underline{a}$	A column vector of entries a_i for $i = 1, \ldots, d$. 9
$\mathbb{1}[A]$	Indicator function for the set A. 10
$N_{[a,b]}(A\|B, C)$	Normal density on A, with mean B, and variance C, truncated to $[a, b]$. 13
$\sim$	Distributed as, or, Distributed according to the density. 13
$\propto$	Proportional to 13
$R(\mathcal{T}_i)$	The risk of the ith tree. 18
Inv–Gamma$(A\|B, C)$	Inverse gamma density on A, with shape B, and scale C. 26
Multinomial$(A\|B, C)$	Multinomial mass function on A, with count B and probabilities C. 27
Dirichlet$(A\|B)$	Dirichlet density on A with concentration parameters B. 27
$s(\mathcal{A}, n)$	The shatter coefficient for sets $\mathcal{A}$ on sample size n. 50
$V_\mathcal{A}$	The VC dimension for functions in the class $\mathcal{A}$. 50
$v_n = \frac{\sum_{i=1}^n \mathbb{1}(y_i = \hat{\mathcal{T}}(X_i))}{n}$	The empirical error of the classifier. 50
$v = \Pr(\mathcal{T}(X) = y)$	The theoretical (Bayes) error of the classifier. 50
$\mathcal{H}(x)$	The entropy function. 50
$J(\underline{y}\|\underline{x})$	The Jacobian of the transformation from $\underline{y}$ to $\underline{x}$. 66
$\mathbb{R}$	The real number system. 66
$\mathbb{S}^d$	The d-dimensional simplex. 66
$Diag(a_i)$	A square, diagonal matrix with diagonal entries a_i. 68
$\mathcal{O}(A)$	Big "O" of A. 68
$\odot$	Hadamard product of two vectors (element-wise product). 69
$MVN(\underline{a}\|\underline{b}, C)$	Multivariate normal on $\underline{a}$, with mean $\underline{b}$, and covariance C. 70
$S\beta(c\|\alpha, \beta, a, b)$	Scaled beta distribution on c with parameters α, β, scaled to the support $[a, b]$. 71
Φ, (Φ^{-1})	Cumulative density (quantile) function for the standard normal. . 72
$Unif(A, B)$	Continuous uniform density on the region $[A, B]$. 73

Exponential(λ) The exponential density with rate λ. 76
Laplace(μ, σ) Laplace density with location μ and scale σ. 76

List of Abbreviations

CS	Computer science	1
GLM	Generalized linear model	1
GLMM	Generalized linear mixed model	1
UCI	University of California Irvine	2
ALN	Additive logistic regression	2
CART	Classification and Regression Tree	3
CGM	Chipman, George and McCulloch	5
DMS	Denison, Mallick and Smith	5
MH	Metropolis-Hastings	5
MCMC	Markov chain Monte Carlo	5
FS	Forward selection	9
SSVS	Stochastic search variable selection	10
RJ-MCMC	Reversible jump Markov chain Monte Carlo	12
LAR	Least angle regression	12
LASSO	Least absolute shrinkage and selection operator	12
UUP	Uniform uncertainty principle	13
MSE	Mean squared error	20
VIMP	Variable importance	19
ZIP	Zero-inflated Poisson	28
GP	Gaussian process	42
VC	Vapnik-Chervonenkis	50
DGP	Data generating process	54
MVN	Multivariate normal	75
ZINB	Zero-inflated negative binomial	28
DiVaS	Dirichlet variable selection	2
ALoVaS	Additive logistic variable selection	2
ALT	Additive logistic transformation	64

GL	Gramacy and Lee	43
DGP	Data generating process	54
MTD	Markov transition density	82
CHAID	Chi-squared automatic interaction detection	100
GIG	Generalized inverse Gaussian	83

1 Introduction

This thesis outlines variable selection as a genre of research in the intersecting domains of statistics, computer science (CS) and other data sciences, including several related subfields of CS and statistics. Our goal is to develop methods to perform *explicit* variable selection within a decision tree model. We emphasize "explicit" because there are several *ad hoc* methods that are currently common in applied practice. These *ad hoc* methods perform variable selection using decision trees, usually with little theoretical justification, and no notion of measure on the individual variables.

Towards our goal, we give an historical overview of decision trees, surveying literature from both CS and statistics, spanning seven decades. While an exhaustive literature review would be a Herculean task, we seek to examine a representative sample of literature to give the reader sufficient knowledge of the choices and strategies applied by researchers. What will emerge is a confluence of several fields including mathematics, statistics, CS and related subdomains applying their own methods of research into this diverse subdomain of study.

We are fundamentally looking toward variable selection as the goal, but a reasonable question to ask is "who cares?"; why should we think that variable selection is worth studying? Moreover, why should we study variable selection in this very specific subdomain of decision trees? The simplest answer is that datasets are getting bigger. Cheap computing power and the march towards an interconnected web of business, socialization and life has nearly automated many data collection processes. This automation has created a 21st century gold rush into any academic pursuit that trains an individual to work with data. Thus, big datasets are here to stay.

Big data can mean a large number of observations, or a large number of variables. In this thesis we are mainly concerned with a large number of variables. Specifically, we study methods to choose subsets of variables from a decision tree model in reasonable ways.

There are many well known statistical models, perhaps the most common is the linear model. A straight forward extension to the linear model is the transformed linear model, known as the generalized linear model (hereafter abbreviated GLM). Further extensions allow for random effects, known as GLMMs (the extra M stands for 'mixed'). The earliest developed variable selection

techniques are usually considered to be the forward, backward, and stepwise techniques. These three techniques were originally studied for linear regression models in the 1960s. Examining historically, as we do in the next subsection, we will see that decision trees were also one of the first forms of variable selection when the landmark paper by Morgan and Sonquist [**?**] was published. This groundbreaking work would not be fully appreciated in the research community for nearly two decades. Finally in the 1990s and the first decade of this century, the methods employed in this original paper would be in widespread use, in both academia and industry.

This introductory chapter only outlines the material to be discussed in the subsequent thesis. We give a literature review of decision trees in the proceeding two subsections of this chapter. The final subsection of this chapter provides a brief literature review of variable selection methods. Chapter 2 gives an overview of the various decision tree models discussed in this thesis and gives some technical details about the models. Chapter 3 presents variable selection methods for decision trees using a Dirichlet type prior approach for covariate selection. In Chapter 3 we discuss a class of methods for variable selection proposed by the author of this thesis. Chapter 4 presents case studies of decision tree variable selection methods using simulated data and using data taken from the UCI machine learning repository [**?**]. Chapter 5 presents an alternative approach using normal distributions as a variable selection distribution and transforming to a probability scale using a transform we call the ALN transform. We call this method the additive logistic variable selection (ALoVaS) method. This chapter shows how the class of methods from Chapter 3 can be used to solve the decision tree variable selection problem using non-Dirichlet priors for the probability of selecting a covariate. Chapter 6 presents a case study of the ALoVaS method applied to the internet ads dataset. Chapter 7 Compares and contrasts the DiVaS and ALoVaS methods. Finally, Chapter 8 discussed and synthesizes the results and conclusions of the thesis and points to future work.

1.1 Related Work

This subsection details two important aspects of this thesis: decision trees and variable selection. Little work has focused on variable selection in decision trees and, for this reason, we separate the literature review into two components. In subsequent chapters, we provide more details on the

few implementations of variable selection in decision tree problems. We begin with an historical review of the decision tree literature. Then, in a subsequent subsection, we survey the literature on the variable selection problem.

1.1.1 An Historical Path Through the Literature

In this section we review decision tree literature from an historical perspective. We begin with the earliest decision tree paper in the statistics literature, the paper by Morgan and Sonquist [**?**]. Their proposal suggested that the linear model is often an inadequate method to handle complex survey data analysis. The authors outline several complications with survey data, including high correlations among the covariates, known to cause instability in linear models, and complex interactions, which make the linear model difficult to interpret and complicated to calculate. Furthermore, their paper cites only one previous author who had a similar but not the same approach. They mention an applied statistics paper from 1959 by Belson [**?**], also an economist. Belson's work certainly predates Morgan and Sonquist's, but the aim of both papers appears to be to partition recursively complex survey data, applying an empirical and simple approach instead of a theoretical approach to analyzing large complicated data.

After the pioneering work of Morgan and Sonquist and that of Belson, the next researcher to begin looking at decision tree methods of data analysis was Kass [**?**]. Although Kass' paper does not present a graphic of a decision tree, the method he proposes is that of recursive partitioning of the data. He gives suggestions and recommendations for how to carry out this procedure and notes the need for further research in this area. Kass follows this paper up with another paper in 1980 [**?**]. Kass [**?**] studies recursive partitioning again but this time specifically on categorical data. Conclusions and results are similar to those in Kass [**?**]. It is worth noting that although Breiman, Friedman, Olshen and Stone's 1984 CART book [**?**] is often considered the work that introduced decision trees into the statistics literature, here we have noted at least four papers prior to the 1984 publication of this book and we by no means claim to be exhaustive in our review. Moreover, it is equally surprising that Breiman noted that they published the CART book as a book because the authors thought that no statistics journal would publish the work [**?**].

The work of Breiman et al. [**?**] was pioneering in several aspects. The last chapter contains a theoretical proof of the consistency of decision trees as the number of observations, typically denoted n, grows arbitrarily large. Breiman et al. provided a new framework, which involved growing a full tree and then pruning back to optimality. This is the first instance of a consistent stopping rule in the decision tree literature. In addition, the book presents specifics of algorithmic implementation. Also many practical issues such as stopping criteria are thoroughly discussed, indicating why the full growth and then pruning approach is preferred over previously proposed simpler stopping rules. The book is an excellent reading for theoretical statisticians and applied data analysts. Moreover, the low cost of the book and the practical interpretability of decision trees made the method popular among researchers in several fields.

Other work around the same time included the pioneering work of Quinlan [**?**] and his textbook [**?**]. Both Quinlan [**?**] and Quinlan [**?**] discuss induction learning for decision trees. Quinlan's research differs from the statistical approaches in two aspects. The first fundamental differences is that Quinlan uses multiway splits compared to only binary decision tree splits in the previous statistics literature. Second, Quinlan uses an information theoretic approach to justify decision trees whereas the statistics literature takes a nonparametric approach.

Attempting to improve the prediction errors of CART trees, Loh and Vanichsetakul proposed using linear combinations of covariates as splitting rules instead of the simple axis orthogonal splits of the CART methodology. The fundamental differences between the CART method and the work of Loh and Vanichsetakul [**?**] are:

- Multiway, or more than two way splits are possible at each node.
- A direct stopping rule is used.
- Loh and Vanichsetakul's method estimates missing values.
- The tree splits are linear combinations and may contain both categorical and continuous covariates.
- Loh and Vanichsetakul's method has no invariance to monotonic transformations;

- Computationally faster than Breiman et al.'s method.

Furthermore, Loh and Vanichsetakul used statistical inference approaches such as F ratios to choose splits and to stop the tree from growing. This work was criticized by Breiman and Friedman [**?**] on several aspects, the most important of which are the lack of robustness, no proof of a consistent stopping rule, and lack of invariance to monotonic transformations.

In the subsequent decade much research was done, both empirically and theoretically regarding the efficacy of decision tree methods. The next major extension was done in the year 1998. Two groundbreaking papers were published, one by Chipman, George and McCulloch [**?**], and another by Denison, Mallick and Smith [**?**], hereafter referred to as CGM and DMS , respectively. Both articles brought Bayesian computational techniques to bear on the problem of decision tree induction. Much Bayesian computational work was accomplished following the groundbreaking Gibbs sampler first published in 1990 by Gelfand and Smith [**?**], such as Metropolis-Hastings (MH) samplers [**?**, **?**] and reversible jump methods [**?**]. The papers by CGM and DMS brought the 8+ years of research in Markov chain Monte Carlo (MCMC) methods into decision tree search methods. CGM proposed a novel process prior and gave a set of proposal functions that exhibit useful cancellations in the MH ratio. In a similar vein, DMS proposed a reversible jump algorithm with a similar proposal function that appears to mix more efficiently while searching the space of trees. These two papers are the genesis of our developments in the current thesis.

Around the same time (the 1990's), the machine learning community was experiencing rapid growth. During this time of rapid growth, Leo Breiman helped to bridge the gap between the statistics community and the machine learning community, publishing fundamental work proposing the bagging method [**?**]. During the same year, 1998, Ho proposed a similar generalization known as the random subspace method [**?**]. Both algorithms use resampling methods similar to the bootstrap [**?**, **?**], but build a decision tree on each subsampled dataset and then aggregate the predictions from the resulting trees to improve prediction and stability of the estimator. Although more refined and developed, a similar approach is the random forest model [**?**]. The random forest model is the subject of current research into the theoretical properties of the resulting collection of trees and their predictions [**?**, **?**]. Interestingly, this research has led to the conclusion that the pruning rule

makes a consistent decision tree but that an analyst may substitute averaging, instead of pruning by using many fully grown trees and averaging the predictions, leading to a consistent model. This model is very similar to bagging and differs primarily in implementation details. Also, the practical performance of these random forest variations has empirically been shown to outperform the bagging method [**?**].

The authors Chipman, George, and McCulloch (CGM) [**?**] did further further fundamental research in developing Bayesian decision trees. In 2000 CGM formulated another modification of their previous model, this time proposing another clever prior that encouraged shrinkage and sharing of information by nodes close together in the tree [**?**]. Then in 2002, these authors' proposed another modification to their previous work suggesting that perhaps constant models in each terminal node were not effective or general enough to effectively model observed data. Instead, they suggested to allow GLM's in each terminal node and called the resulting models *treed* models [**?**]. Treed models generalize classic decision tree models to include linear or generalized linear models within each terminal node. It is worth noting that the previous models proposed using constant functions are in fact special cases of treed models. They are linear regression models with only an intercept term in each terminal node. Furthermore, the amount of available data decreases the further into the tree you traverse, so either tree regularization or node model regularization is necessary to combat the "small n, big p" problem that occurs in terminal nodes of large trees.

Several further refinements to the Bayesian approach were proposed during the subsequent decade. Wu, Tjelmland and West [**?**] offered an improved proposal function that contains a radical restart move that grows a new tree from scratch when the current chain is stuck in a local maximum. This is accomplished by a "pinball prior", which is one of the unique and practically useful aspects of the paper. As a further improvement, to aid the mixing of the Markov chain Monte Carlo and to aid the chain in traversing from one local maximum to another, Leman, Chen and Levine proposed a new MCMC algorithm called the multiset sampler [**?**]. The multiset sampler is able to achieve a move from one local max to another by allowing the chain to be in two states of the Markov chain at a single instant. The multiset sampler was originally developed for evolutionary inference in the reconstruction of phylogenetic trees, however this specific application was referred to as

the evolutionary forest algorithm. The extension from phylogenetic trees to CART trees is fairly straightforward. Moreover, Gramacy and Lee [**?**] extended the treed model to include Gaussian processes and gave an application to simulation of rocket boosters. Gramacy and Taddy [**?**] provide an R package called 'tgp' that implements the ideas in Gramacy and Lee [**?**] and provides several extensions and special cases.

Several other important works deserve to be mentioned during this timeframe (the 2000's). In the computer science domain, Dobra advanced a scalable algorithm to do decision tree induction called SECRET [**?**]. In addition to providing scalable algorithms for decision tree induction, Dobra offered a novel modification to decision trees: a probabilistic split at each node. This creates fuzzy classifiers and fuzzy regions in the covariate space around which splits are made. The probabilistic splits create regions of splits in the covariate space. A similar phenomenon is observed in bagged trees [**?**], although Dobra's approach is simpler and preserves the interpretability of the single tree. Along different lines, Gray and Fan proposed genetic algorithms to build decision trees in their TARGET algorithm [**?**]. Genetic algorithms are roughly similar to the population approaches [**?**] except that they rely on an analogy with evolutionary processes to guide their development.

During the same decade (the 2000's), the stigmergic approach to decision tree induction was investigated empirically. The stigmergic approach generally works by agents, or a population approach, whereby each agent is able to communicate with other agents through the environment in which the agent acts. In the case of the ant colony optimization algorithm [**?**], stigmergy is achieved by ants leaving pheromone trails that influence the behavior of subsequent ant agents. While ant colony optimization approaches do not generally yield the best performance in the training data, they are generally competitive with other algorithms in test data prediction and provide differing and often insightful trees [**?**]. Several authors have modified algorithms to build decision trees using the antminer system [**?**] [**?**]. The antminer system is publicly available in Matlab code at the link http://www.antminerplus.com/ .

Two additional statistical approaches have been advocated recently: the EARTH algorithm and the GUIDE algorithm [**?**] [**?**]. EARTH is an algorithm that nonparametrically selects covariates to include in the model. Doksum, Tang, and Tsui [**?**] provide a theorem that shows the covariate se-

lection consistency of the EARTH algorithm as the sample size, $n \to \infty$, and as the dimensionality of the data, $d \to \infty$.

The GUIDE algorithm [**?**], proposed by Loh, is similar to the original work of Loh and Vanichsetakul [**?**]. The GUIDE algorithm is a general unbiased interaction detector that claims to have superior performance at detecting interactions and incorporating those into the model by splits that are linear combinations of covariates. Unfortunately, there is no guarantee of consistency of the GUIDE algorithm.

Decision trees have long been applied to survival data. The explosion of cheap genome sequencing and generally cheap data collection and storage has made variable selection methods increasingly useful in the context of survival trees. The work of Ishwaran, Kogalur, Gorodeski, Minn, and Lauer defines a new quantity called a maximal subtree and use the inherent topology of the tree and the maximal subtree to measure variable importance [**?**]. Ishwaran et al. advocate a bootstrapping approach to variable selection in the context of random survival forests. The authors noted good empirical performance and provide probability approximations to calculations necessary in their simulations.

Finally, Taddy, Gramacy, and Polson have extended the decision tree literature into the time series domain [**?**]. The authors propose to embed decision trees into a dynamic stochastic process. The authors suggest that the underlying tree of the model is updated by alternating grow, split, and do nothing moves. Taddy et al. also provide an illustrative example using car crash data. Time series applications of decision trees appears to be a fruitful area of research.

1.1.2 A Brief Overview of Variable Selection Methods

In order to understand the methods we will employ in further chapters, we will give a brief overview of variable selection methods for linear models. We focus on linear models because the majority of the research into variable selection has been conducted on these models. Extensions have been done for certain methods on GLMs, however this material is usually more complicated and specialized. Also these extensions have not been applied to all the methods we discuss. In subsequent chapters, we will see how the material we introduce here can be applied to decision

trees using an appropriately defined transform.

Perhaps the earliest variable selection method, besides the modest proposal of Morgan and Sonquist [**?**], is the forward selection method (FS). The forward selection method and many variations appear in the early 1960's from several references making it very difficult to identify the person who originally proposed this method. References in other languages are not included in this review, further obfuscating the designation of first proposal. The FS method is well know to run into difficulties when several covariates are highly correlated with each other [**?**]. There are several nice benefits to the FS method, such as computational feasibility and readily available, high quality computer code that implements this technique. Nonetheless, several researchers have pointed out the sometimes dubious nature of the resulting output [**?**].

A related method to forward search is backwards search, which operates analogously to the forward search, except the model starts with all covariates in the model. At each iteration the variable with the lowest correlation with the response is removed from the model. Also, the stepwise method devised by Efroymson [**?**] represents a middle ground between forward and backward search by sequentially adding and deleting variables. The stepwise method tries at each step to include or exclude a variable, based upon an F statistic value. The backward and stepwise searches are also known to encounter similar difficulties as FS [**?**] does. Highly correlated covariates may produce dubious results. A nice overview of all three methods, along with several other subset selection approaches can be found in Miller [**?**]. The second edition of Miller's book contains many updates, including chapters on Bayesian and regularization methods.

Ridge regression is another popular approach used in subset selection problems [**?**]. The ridge regression estimator is defined as

$$\underline{\hat{\beta}} = \underset{\underline{\beta}}{argmin}\ (\underline{y} - X\underline{\beta})^T\Sigma^{-1}(\underline{y} - X^T\underline{\beta}) + \lambda\underline{\beta}^T\Gamma\underline{\beta}, \tag{1}$$

for $\lambda \geq 0$ some scalar (constant), with $\underline{y}$ an $n \times 1$ vector, $\underline{\beta}$ a $d \times 1$ vector and X an $n \times d$ matrix. The objective function (1) has the closed form solution $\underline{\hat{\beta}} = (X^TX + \lambda\Gamma)^{-1}X^T\underline{y}$ and Γ is a matrix chosen to be conformable for addition with X^TX. Ridge regression combats the effect of high correlation among the columns of X, allowing the matrix (X^TX) to be inverted. This method is often most useful when $d < n$ and regression coefficients are desired. Note that estimated $\underline{\hat{\beta}}$

vectors with zero entries are unattainable in this penalized regression method when Γ is a full rank matrix.

Frank and Friedman [**?**] proposed another penalized regression approach to ridge regression called bridge regression. Frank and Friedman also gave an optimization algorithm that solves the objective function. The objective function to solve was defined as

$$\underline{\hat{\beta}} = \underset{\underline{\beta}}{argmin} \ (\underline{y} - X\underline{\beta})^T\Sigma^{-1}(\underline{y} - X^T\underline{\beta}) + \lambda|\underline{\beta}|_\nu^\nu, \tag{2}$$

where the second term denotes a ν norm and $\nu \geq 0$ is a specified constant. This technique later became known as "bridge" regression, because the objective function bridges between several well known estimators by choosing various values of ν. For example $\nu = 0, 1, 2$ correspond to subset selection, the lasso, and ridge regression, respectively. We note that the $\nu = 0$ case is interpreted as $\lim_{\nu\to 0} |\underline{\beta}|_\nu^\nu = |\underline{\beta}|_0$, the limiting case of the ν-norm where the norm counts the number of non-zero elements of the vector.

During the 1990's Bayesian approaches became practical because of advances in computational statistics, especially developments in Gibbs sampling and MH algorithms. These advances lead to several researchers proposing Bayesian variable selection techniques. The first of these advances in the variable selection literature was the stochastic search variable selection (SSVS) approach of George and McCulloch [**?**]. The SSVS approach relies on

$$\delta(x_0) = \lim_{\sigma\to 0} \phi(x_0; \sigma) = d\mathbb{1}[x \geq x_0], \tag{3}$$

where $\phi(a; b)$ denotes a Gaussian density evaluated at the point a, with standard deviation b and mean zero [**?**, **?**]. The notation $\delta(x_0)$ will be used to denote the Dirac delta functional. Essentially we are trying to determine if $\beta_j = 0$, or if $\beta_j \neq 0$ and we might wish to assign a point mass probability to $\beta_j = 0$. Thus, a reasonable approximation is to use a two component mixture of normal distributions, with one normal having a large variance compared to the other. Using a latent variable representation, George and McCulloch gave a Gibbs sampling algorithm that samples subsets of predictors and thereby provides variable selections. The main drawback is that George and McCulloch only offered SSVS for the Gaussian linear model. Extensions in the literature indicate the method can be applied to GLMs and to problems where the number of covariates is

larger than the number of observations, also called the '$p > n$ problem', such as gene selection [**?**, **?**].

Historically, the 1990s were a fruitful period of research and methods that were proposed earlier but were not yet computationally feasible were rediscovered. Breiman advocated better subset selection using the nonnegative garrote [**?**, **?**]. As indicated by the title, Breiman's procedure [**?**] selected better subsets compared to backward search and subset selection. The subsets selected are the non zero values of the estimated coefficients, conventionally denoted as $\underline{\hat{\beta}}$. In the non-negative garrote problem these estimates of $\underline{\beta}$ are obtained by solving the objective function

$$\underset{\forall j: c_j \geq 0}{argmin} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{d} c_j \hat{\beta}_j x_{ij})^2 + \lambda \sum_{j=1}^{d} c_j, \tag{4}$$

where λ is a specified constant, or estimated by some other means. The estimate $\hat{\beta}_j$ denotes the jth least squares estimate. Alternately we can use an estimate of β_j obtained by means other than least squares, for example using a ridge estimator. Once $\hat{c}_j$ is estimated, the non-negative garrote estimated is $\hat{\beta}_j^{\text{NNG}} = \hat{c}_j \hat{\beta}_j^{\text{LS}}$. Under an orthogonal covariate assumption ($X^T X = I$), there are closed form solutions to the objective function (4) that have nice interpretations as hard thresholding rules. The threshold is determined as a function of the Lagrange multiplier λ. These threshold formulae are derived in an the Appendix. Breiman compared the non-negative garrote method against subset selection and backward search procedures, indicating positive results for the non-negative garrote. Yuan and Lin [**?**] and Xiong [**?**] give further results for the non-negative garrote. Yuan and Lin used the theory of duality to give oracle type properties of the nonnegative garrote estimator. Briefly, the oracle property states that $\Pr(\hat{\beta} = \hat{\beta}_{\text{global}}) \to 1$ as $\lambda_n \to \lambda$. In other words, this means the local estimator becomes the global estimator for a suitably constructed sequence of regularization parameters. Xiong studied iterating the non-negative garrote procedure and also how the degrees of freedom influence the prediction risk of estimates.

We now move to discuss Bayesian approaches to variable selection. In the Bayesian approach, the constraint in the optimization problem is motivated as the negative logarithm of a prior measure placed on the parameter(s) of interest. We start first in the context of sampling methods. Besides the method proposed by George and McCulloch [**?**], there is another popular method applied to

variable selection problems in a Bayesian context. This alternate method is known as reversible jump Markov chain Monte Carlo (RJ-MCMC). This method was first suggested by Green [**?**]. Green showed that mixtures of normal distributions can be modeled using the RJ-MCMC sampler. Specifically, the RJ-MCMC algorithm eliminates the need to specify the number of mixtures in the normal distribution, effectively eliminating tuning parameters from normal mixture distribution problems.

One of the most fruitful areas of research in the last 15 years has been the lasso, which stands for *least absolute selection and shrinkage operator*. Motivated by the non-negative garrotte, the lasso was proposed by Tibshirani in 1996 [**?**]. Many subsequent papers give properties of the lasso, or proposed alternate methods to solve the objective function. The objective function is defined as

$$\underset{\underline{\beta}}{argmin} \sum_{i=1}^{n}(y_i - \sum_{j=1}^{d} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{d} |\beta_j|_1, \tag{5}$$

where the $\lambda \geq 0$, is a constant or tuning parameter. Tibshirani solved the optimization by doing a grid search across several values of λ. Unhappy with the computational complexity of the linearly constrained quadratic programming optimization approach suggested by Tibshirani [**?**], Efron et al. [**?**] proposed the least angle regression algorithm, hereafter referred to as LAR, to solve the lasso optimization problem. The LAR algorithm uses a homotopy method to solve the objective function, Equation 86. Along similar lines, both Osborne, Presnell and Turlach, and Zhao and Yu used duality theory to prove properties of the lasso estimators [**?**, **?**]. Bunea, Tsybakov, and Wegkamp [**?**] and Candes and Plan [**?**] have also derived oracle and optimality properties of the lasso problem estimators. These optimality results state that lasso solutions are within a constant factor of the true values, with the constant usually being a number of the form $1+\epsilon$ and $1 \geq \epsilon \geq 0$. Tibshirani [**?**] originally noted a Bayesian approach to interpreting Equation 86. Park and Casella [**?**] gave further Bayesian lasso results including a marginal maximum likelihood (empirical Bayes) method for estimating the tuning parameter λ. An empirical Bayes argument is used to justify this method of estimating λ. Finally, Zhou and Hastie [**?**] combined the penalties of the lasso and of ridge regression and called this objective function the elastic net. The elastic net can be interpreted as a convex combination of a lasso ($|\underline{\beta}|_1$) penalty and a ridge regression ($|\underline{\beta}|_2^2$) penalty. Zhou and

Hastie [**?**] provided a transformation to convert the elastic net problem into a lasso problem so that the LAR algorithm can be used to solve the elastic net objective function efficiently.

The flurry of regularization papers on the lasso and the non-negative garrote methods inspired Candes and Tao [**?**]. Candes and Tao [**?**] advocated an alternate method of estimating regressors called the Dantzig selector. The Dantzig selector estimates a sparse vector of coefficients, denoted $\hat{\underline{\beta}}^{\mathrm{D}}$, by solving the objective function

$$\underset{\underline{\beta}}{argmin}\ |\underline{\beta}|_1 + \lambda|X^T(\underline{y} - X\underline{\beta}) - k_p\sigma|_\infty. \tag{6}$$

Candes and Tao [**?**] suggested that this objective function be reformulated as a linear program. Linear programs have several highly reliable software applications to estimate the optimum. Candes and Tao [**?**] also gave a primal-dual interior point algorithm to solve the objective function with publicly available Matlab code. The authors emphasized a uniform uncertainty principle (UUP) and derived oracle and optimality results based on the UUP. Bickel, Ritov, and Tsybakov [**?**] and Koltchinksii [**?**] provided further theoretical analysis of the Dantzig selector, including optimality results and oracle inequalities under different conditions than Candes and Tao [**?**].

The last estimator we discuss is the horseshoe estimator, arising from the horseshoe prior. This prior was proposed by Gelman as a way to combat a numerical difficulty in MCMC sampling [**?**]. Carvalho, Polson, and Scott [**?**, **?**] describe the general setup. The horseshoe estimator arises via the hierarchical probability representation described in Equations 101-104 beneath

$$\underline{y}|\underline{\beta} \sim N(X\underline{\beta}, \sigma^2 I), \tag{7}$$

$$\beta_j|\lambda_j, \tau \sim N(0, \lambda_j^2\tau^2), \tag{8}$$

$$\pi(\lambda_j) \propto \frac{1}{1+\lambda_j^2}\mathbb{1}[\lambda_j \geq 0], \text{ and} \tag{9}$$

$$\pi(\tau) \propto \frac{1}{1+\tau^2}\mathbb{1}[\tau \geq 0]. \tag{10}$$

Here it should be made clear that the horseshoe prior is a half-Cauchy distribution. It is worth noting that the local scale priors λ_j for $j = 1, \ldots, d$ are on the standard deviation scale and this prior is one of a general class of half-t densities [**?**, **?**]. Carvalho et al. [**?**] showed that the horseshoe

prior has several appealing properties, including sparsity properties shared by the lasso estimator, while also having other desirable properties *not* shared by the lasso. Carvalho et al. [**?**, **?**] gave examples demonstrating the utility of this method for variable selection and indicated superior performance compared to the lasso in the datasets examined. Moreover, Polson [**?**] argued that the half-t prior, or the half Cauchy prior as a special case, should become the reference (default) prior for variable selection problems. Polson justified this argument based on the many desirable properties of the horseshoe, some of which are not shared with other estimators such as the lasso.

The attentive reader may notice that the majority of material in this subsection pertains to linear regression models. Decision tree models are inherently nonlinear in nature so it is reasonable to wonder: Will we use any of the material discussed in this section? The short answer is "yes," but not directly. Our approach will be in terms of linking these sparse linear models with decision trees and variable selections. The ALoVaS method provides exactly this link and we will be elaborate in subsequent chapters. We explain our decision tree model in the next chapter and show how we exploit sparsity in said model by using modifications to distributions to encourage sparsity.

We conclude this section by noting some review papers on the variable selection problem within the literature. O'Hara and Silanpää [**?**] gave a recent notable Bayesian survey. This paper covered many of the methods discussed above in some detail and compared them all from a Bayesian perspective. Miller [**?**] gave a book length treatment on variable selection methods, with the second edition of the book including some Bayesian methods and the lasso, but not all methods discussed in this section. In particular, Miller [**?**] described the forward, backward and stepwise searches in detail and gave several useful references to the literature. From the CS literature, the review by Dash and Liu [**?**] provided a useful reference into the CS field developments in variable selection. Dash and Liu [**?**] also provided a useful framework to compare subset/variable selection approaches. George [**?**] provided a good overview and major references in the variable or subset selection field current to the year 2000.

2 Preliminaries

This chapter provides an overview of the models used for decision tree induction and variable selection. We begin by discussing the earliest methods of induction, greedy algorithms. With the advance of time (about a decade), greedy induction approaches became fully implemented through high quality Fortran code. Bayesian approaches to building decision trees were not possible until advances in computing power and the rediscovery of sampling based approaches to Bayesian statistics became popular again. While the work of Breiman et al. [**?**] always contained a Bayesian flavor, no probability measure over trees was ever proposed. That is until the groundbreaking works of Denison, Mallick and Smith [**?**] and Chipman, George, and Mcculloch [**?**].

During the same time (the 1990s and the 2000s), researchers in Machine Learning and Statistics were developing theory beyond that published by Breiman et al. [**?**], that ensured decision trees' consistency. In this case consistency means that the error rate of the tree converges to the Bayes error. The initial impetus for this work stemmed from the landmark paper of Vapnik and Chervonenkis [**?**], and subsequent work by Vapnik [**?**], though it would take over a decade for the full power of their combinatorial approach to be widely used in both Machine Learning and Statistics.

This chapter presents necessary preliminaries for the reader to understand the developments in subsequent chapters. Therefore the reader may omit this chapter on a first reading and refer back to the necessary subsections when subsequent chapters recall the preliminaries. Section 2.1 and Section 2.2 review the necessary material on greedy and Bayesian approaches to decision tree induction. The reader who is not familiar with these two approaches to decision trees should read these two subsections before moving on to the rest of the thesis.

2.1 Greedy Induction

In this section we answer the question: how does greedy induction work? Once the reader has completed this section they should, with sufficient time, be able to reproduce computer code that implements decision trees using greedy strategies. We begin with impurity functions.

2.1.1 Impurity Functions

An impurity function corresponds to a likelihood function in statistics or to an objective function in machine learning. This function measures, in some sense, how many good observations lie in a node of the tree and how many bad observations lie in a node of the tree. Formally, an impurity function, denoted ϕ, must satisfy three axioms:

1. $\phi(1/2, 1/2) \geq \phi(p, 1-p)$ for any $p \in [0, 1]$.

2. $\phi(0, 1) = \phi(1, 0) = 0$.

3. $\phi(p, 1-p)$ non-decreasing for $p \in [0, 1/2]$ and non-increasing for $p \in [1/2, 1]$.

Figure 1displays the impurity functions given in Breiman et al. [**?**] are:

1. Entropy: $\phi(p, 1-p) = -plog(p) - (1-p)log(1-p)$.

2. Gini: $\phi(p, 1-p) = 2p(1-p)$.

3. Misclass probability: $\phi(p, 1-p) = min(p, 1-p)$.

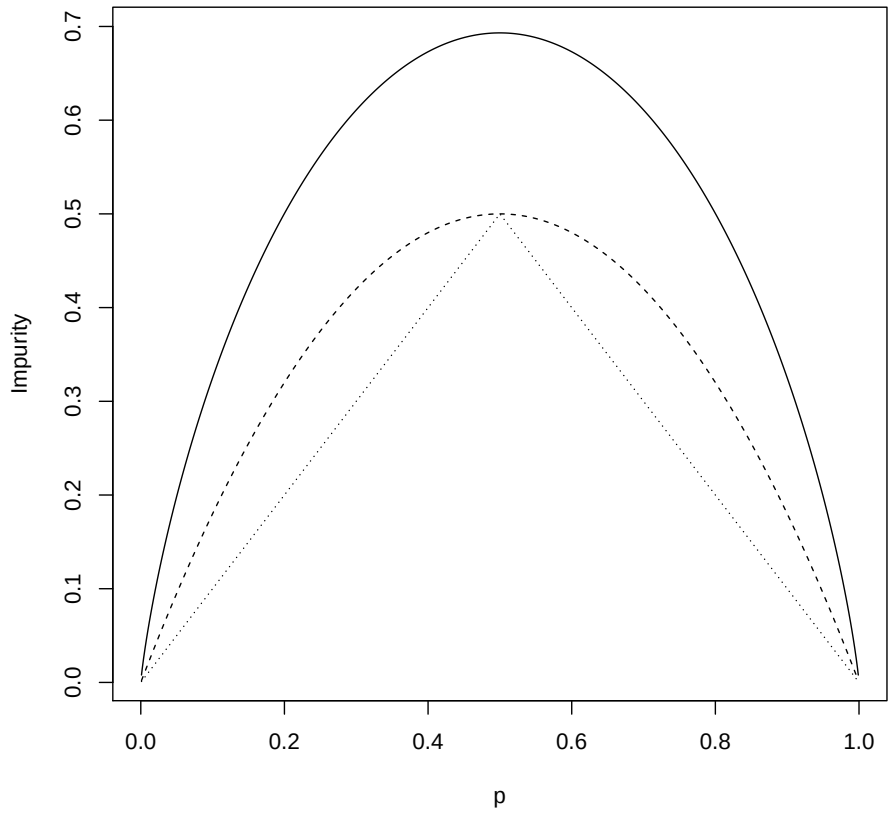


Figure 1: A plot of the three impurity functions: entropy (solid line), Gini (thatched line), and misclass probability (dots).

To build a regression tree, in place of an impurity function, we use the mean squared error criteria denoted

$$\text{MSE} : \phi(y_i, \bar{y}) = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}.$$

The reader should note that the greedy approach to decision tree induction always tries to optimize impurity functions. The optimization problems are known to be NP-Hard (for an introduction to complexity theory confer Garey and Johnson [**?**]). Briefly, NP-Hard optimization problems are considered some of the hardest optimization problems to solve. These optimization problems increase exponentially with the number of observations in the problem. Therefore, exact methods

will nearly always take too long to compute and thus greedy strategies are used to approximate the global optimum, assuming one exists, with a local optimum.

2.1.2 Induction

The induction of decision trees proceeds by solving the objective function

$$\underset{t,s}{argmax}\Delta\phi = \phi - \pi_L\phi_L - \pi_R\phi_R. \tag{11}$$

Here the variables t and s scroll over all covariates in the data and all midpoints between successive observations (or at observed points) of the tth covariate respectively. The proportions π_L and π_R represent the number of data points going into the node to the left (L) and right (R) of the current node if the chosen split is on covariate t and observation s. The impurity functions are similarly subscripted. Thus, if there are 100 observations in the current node and as a result of the split on covariate t, at observation s, 70 observations go into the left child node and 30 observations go into the right child node, then the two proportions are $(\pi_L, \pi_R) = (0.7, 0.3)$.

The tree induction process continues until no more data points are incorrectly classified, or a predetermined stopping rule is met. A common stopping rule is to stop when there are less than 5 observations in a terminal node. Once the full tree has been built, the second stage of the process now starts. This is known as the pruning stage. Now that the full tree is grown, we progressively prune back terminal nodes of the tree until the root node occurs. Several related approaches have been proposed in the literature to select the optimal tree via pruning. The most common is to select the tree using the regularized risk estimate given in Equation 12

$$R(\mathcal{T}_i, \alpha) = R(\mathcal{T}_i) + \alpha|\mathcal{T}_i|. \tag{12}$$

Here the α parameter is a regularization parameter with larger values of α given greater penalty to the number of terminal nodes in the tree, here denoted $|\mathcal{T}_i|$. The notation $R(\mathcal{T}_i)$ denotes the risk of the tree, which is usually calculated as the sum of squared errors across all terminal nodes in a regression setting or the sum of the impurity function values in each terminal of the tree in the classification case. We choose the value of α leading to the smallest regularized risk ($R(\mathcal{T}_i, \alpha)$).

The value of α is chosen over a grid of positive values by minimizing Equation 12 on holdout data, using a cross validation approach,

Discussion of consistency of this pruning rule can be found in Devroye et al. [**?**], Breiman et al. [**?**], Gey [**?**], and Suavé and Tuleau-Malot [**?**]. All the theoretical results require controlling the complexity of the decision tree, $|\mathcal{T}|$, and allowing the number of data points $n \to \infty$. However, the results in Devroye et al. [**?**] also give explicit bounds for the error of decision tree classifiers for finite values of n.

The pruning rule discussed here, and the induction process overall, is an implicit form of model selection. This is implicit because the selected variables are the variables left in the tree after pruning. Those variables considered important are the variables that remain and those variables not selected are considered not important. Breiman et al. [**?**] define no measure of importance on each variable, so it is difficult to rank variables based on importance. Breiman [**?**] proposed such a measure, called variable importance (abbreviated VIMP) but in the context of random forests and not for a single decision tree. We propose different methods to perform explicit variable selection for Bayesian decision trees in later chapters.

2.1.3 A Simple Example

In this subsection we work through a simple example to give the reader a flavor of the calculations necessary to induct a decision tree.

Consider the following data

i	y_i	x_1	x_2
1	1	2	3
2	2	5	6
3	5	8	9

Table 1: A simple decision tree example data.

We have three observations and two covariates within each observation. The response is a

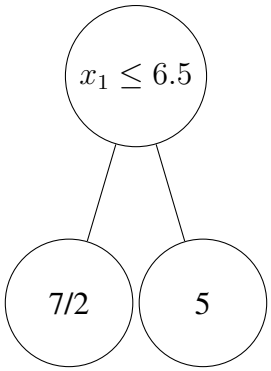


Figure 2: The decision tree after the first split.

continuous random variable, so this will be a regression tree approach. We will examine potential split points by looking at the midpoints between two observed values of the covariates. We begin by sorting the data in increasing order for both covariates. Fortunately, in this case, the data is already in order for both covariates, so no sorting is necessary. We now examine the possibility of a split point on x_1. Using the MSE impurity we calculate ϕ, ϕ_L, and ϕ_R. A sum of squared error calculation shows $\phi = 26/3$, which is a constant value for all calculations we perform. Now for a split between observation 1 and 2,

$$\Delta\phi = 26/3 - (1/3)(1-1)^2 - (2/3)((2-7/2)^2 + (5-7/2)^2) = 26/3 - 9/3 = 17/3.$$

For a split between observation 2 and 3,

$$\Delta\phi = 26/3 - (2/3)((1-3/2)^2 + (2-3/2)^2) - (5-5)^2(1/3) = 26/3 - 1/3 = 25/3$$

Now, because the data is sorted, the same $\Delta\phi$ values will result for potential splits on x_2. Therefore, we, somewhat arbitrarily, choose to split on the covariate with the smaller index, x_1. Thus, our first split is on the value $\{x_1 : x_1 \leq 6.5\}$ and the tree at this point looks like that in Figure 2. Note that Figure 2 displays the mean of the values in each terminal node. The mean value is the predicted value for observations falling into the given terminal node.

If we had more than 3 observations, we would then continue calculating $\Delta\phi$s for the data that falls into each of the two resultant terminal nodes, continuing until there is only one observation in each terminal node, or until there is some specified number of observations in each terminal node.

Although any number is possible, the specified minimum number of observations in each terminal node is usually 5. Once this process is completed, the induction step is finished, and the pruning process begins.

2.2 Bayesian Approaches

This section describes a Bayesian approach to decision trees. In the previous chapter we presented a greedy algorithm to fit decision trees. Besides the observed error in a greedy decision tree, there is nothing to describe the fit of the model, or to provide a measure over the decision tree. This section provides both of these quantities. We begin by defining the CGM model and calculating necessary quantities for the algorithm. Furthermore, there is no explicit model selection, which will be the main contribution of this thesis.

2.2.1 The CGM approach

We begin by defining notation and measures on each quantity of the tree. We assume that the tree topology and split rules are conditionally independent. Based on fundamentals of probability we have the following relationships

$$\begin{aligned} \Pr(\mathcal{T}_i|\underline{y}, X) &\propto \Pr(\mathcal{T}_i)\Pr(\underline{y}|\mathcal{T}_i, X) \\ &\propto \Pr(\mathcal{T}_i)\int_\Theta \Pr(\underline{y}|\mathcal{T}_i, X, \theta)\pi(\theta)d\theta, \end{aligned} \tag{13}$$

where $\Pr(\mathcal{T}_i)$ denotes the prior measure on trees and $\Pr(\underline{y}|\mathcal{T}_i, X)$ denotes the integrated likelihood of the tree. Finally, $\Pr(\underline{y}|\mathcal{T}_i, X, \theta)$ and $\pi(\theta)$ denote the tree likelihood and the prior measure on node parameters, respectively. CGM [**?**] defined this conditional decomposition and showed how to use this to construct an algorithm to sample Bayesian decision trees. We now define the aspects of the model described in the CGM paper [**?**].

The decision tree model has two main components, the tree $\mathcal{T}$ with b terminal nodes, and the parameters in each terminal node, $(\theta_1, \ldots, \theta_b)$. The two main likelihoods in each terminal node are the normal and the multinomial, for continuous and categorical responses, respectively.

We denote the responses in each terminal node as the vector of vectors $Y \equiv (Y_1, \ldots, Y_b)$. Then $Y_i = (y_{i1}, \ldots y_{in_i})$ and the main relation is the independence breakdown

$$f(Y|\mathcal{T}, X, \theta) = \prod_{i=1}^{b} f(Y_i|\mathcal{T}, X, \theta_i) = \prod_{i=1}^{b} \prod_{j=1}^{n_i} f(y_{ij}|\mathcal{T}, X, \theta_i). \tag{14}$$

The two likelihoods are given by

$$f(y_{ij}|\mathcal{T}, X, \theta_i) = N(\mu_i, \sigma_i), \tag{15}$$

and the multinomial likelihood is

$$f(y_{i1}, \ldots, y_{in_i}|\mathcal{T}, X, \theta_i) = \prod_{j=1}^{n_i} \prod_{k=1}^{K} p_{ik}^{\mathbb{1}(y_{ij}=k)}. \tag{16}$$

In Equation 16, p_{ik} denotes the probability y_{ij} of being in category k in terminal node i and $\mathbb{1}(A)$ denotes the indicator function for the set A.

We now proceed to define the tree prior. We start with a tree consisting of a single node, the root node. We then imagine the tree growing by randomly choosing terminal nodes to split on. To grow a tree we must specify two functions, the growing function and the splitting function. The splitting function is denoted $p_{\text{split}}(\eta, \mathcal{T})$ and the rule function is denoted $p_{\text{rule}}(\rho|\eta, \mathcal{T})$. The rule function provides a criteria to determine which of the two child nodes the observed data go into. If the observed covariate value is less than the rule value, then the observation go into the left child node. Similarly, if the observed covariate value is greater than the rule value, then that observation goes into the right child node. Growing a tree (called induction) consists of iterating these steps. creating two new children from a terminal node and assigning a rule to the terminal node (now a parent of two terminal nodes). Figure 3 illustrates one iteration of the induction process graphically.

The probability measure on the potential splits of the tree is

$$p_{\text{split}}(\eta, \mathcal{T}) = \alpha(1 + d_\eta)^{-\beta}, \;\; \alpha > 0, \beta \geq 0, \tag{17}$$

where d_η denotes the depth of the node η and α, and β are scalars. The probability of the specific rule, denoted ρ is

$$p_{\text{rule}}(\rho|\eta, \mathcal{T}) \propto \underbrace{\Pr(\text{split on covariate})}_{=p_{\text{split}}} \underbrace{\Pr(\text{split on a value given a covariate})}_{=p_{\text{rule}}}. \tag{18}$$

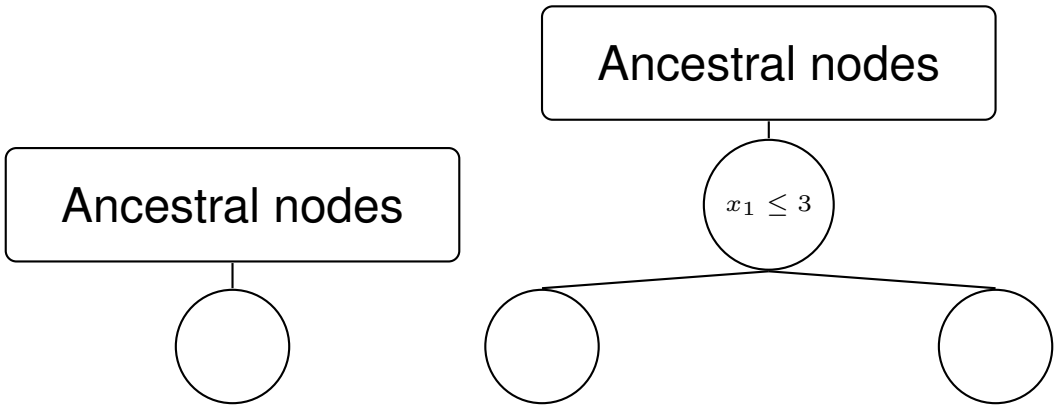


Figure 3: A terminal node in the decision tree before (left) and after (right) a split on a terminal node.

Here CGM recommends using a discrete uniform prior on p_{split} and splitting uniformly amongst the splitting values (in p_{rule}) that do not result in an empty terminal node. While we choose the same proposal mechanism for the p_{rule} quantity, the main point of this thesis is to examine and propose alternate specifications for p_{split}. The data sets modeled in CGM [**?**] and DMS [**?**], and other modifications in the literature, deal with data with a small number of predictors. In this thesis we are concerned with a large number of predictors, so we will focus on variable selection, which will ultimately explain how we specify quantity p_{split} in Equation 18.

2.2.2 Integrated Likelihood

We will now focus on the integrated likelihood, which is the quantity

$$\Pr(Y_i|\mathcal{T}, X) = \int_{\Theta} \Pr(Y_i|\mathcal{T}_i, X, \theta)\pi(\theta)d\theta. \tag{19}$$

To evaluate the integral in Equation 19 we must first define a prior, denoted $\pi(\theta)$, for the parameters in each terminal node. There are two possible priors for the case of the normal likelihood that will result in a conjugate prior/posterior. These are normal and normals-gamma distributions, or equivalently, normal-inverse gamma distributions depending upon the given parametrization.

2.2.3 The Process Prior

Assuming we have a closed form solution for the integral in Equation 19, we can use Bayes' rule to determine

$$\Pr(\mathcal{T}|Y,X) \propto \Pr(Y|X,\mathcal{T})\Pr(\mathcal{T}). \tag{20}$$

We now have an effective means of searching the posterior space over trees to determine the high posterior trees. We can do so by using the Metropolis-Hastings rule

$$\mathcal{T}^{i+1} = \begin{cases} \mathcal{T}^*, & \text{with probability } \alpha(\mathcal{T}^*,\mathcal{T}^i) = \min\left(\frac{q(\mathcal{T}^*,\mathcal{T}^i)}{q(\mathcal{T}^i,\mathcal{T}^*)}\frac{\Pr(Y|X,\mathcal{T}^*)}{\Pr(Y|X,\mathcal{T}^i)}\frac{\Pr(\mathcal{T}^*)}{\Pr(\mathcal{T}^i)}, 1\right) \\ \mathcal{T}^i, & \text{with probability } 1-\alpha(\mathcal{T}^*,\mathcal{T}^i). \end{cases} \tag{21}$$

To evaluate the normalization constant would require summing Equation 20 across all possible trees. This sum includes $\mathcal{O}(nd\frac{4^h}{h^{3/2}})$ terms, with h denoting the maximum height of the trees, n denoting the number of observations, and d denoting the number of covariates. This is an infeasible sum for most data sets, and for all data sets examined in this thesis. For the function $q(-|-)$, which is called the proposal function, we use q to propose a new tree $\mathcal{T}^*$. In Equation 51, $q(\mathcal{T}^*|\mathcal{T})$ denotes proposing a new tree $\mathcal{T}^*$, starting from the current tree $\mathcal{T}$. Our proposal mechanism is as follows:

- The grow step chooses at random one of the terminal nodes and proposes to append two new child nodes with a certain probability that could depend on the tree depth, splitting on a chosen covariate.

- The prune step works in reverse of the grow. A terminal node is selected at random and that node and the node's sibling are pruned to the immediate parent of the two child nodes.

- The change step randomly picks an internal node and attempts to change the split rule at the node with that of another observation, possibly on a different covariate.

- The swap step randomly selects an internal node that is not the root node and proposes to swap the split rules of the parent-child pair. If both child nodes split on the same covariate, then both the children and the parent node's rules are swapped.

- The rotate step randomly chooses a left or right rotation move. Then this step randomly chooses an admissible internal node and rotates.

The rotate operation for binary trees was first introduced in Sleater and Tarjan [**?**] and was introduced into Bayesian decision trees in GL [**?**]. A good introduction and several practical uses of the rotate move can be found in Cormen, Lieserson, Rivest and Stein [**?**]. The proposal of Gramacy and Lee [**?**] only allows a rotate move for the specific case when a swap move is proposed and the parent child pair both split on the same covariate. We modify this rule and allow rotate to be a separate operation of the transition kernel and not a special swap move case. The proposal mechanism of CGM uses the grow, prune, change and swap moves only. We also allow swap moves in our proposal. In addition, neither of CGM nor Gramacy and Lee [**?**] included weights on each covariate in their examples or model specifications. CGM sampled each covariate and split value uniformly, at random.

The probability measure on the tree is

$$\Pr(\mathcal{T}) = \prod_{\eta \in \mathcal{N}} p_{\text{rule}}(\rho|\eta, \mathcal{T}) p_{\text{split}}(\eta, \mathcal{T}), \tag{22}$$

where $\mathcal{N}$ denotes the set of nodes in tree $\mathcal{T}$. The probability measure on each split, here denoted $p_{\text{split}}(\eta, \mathcal{T})$, uses Equation 45. Similarly, the measure on each rule, here denoted $p_{\text{rule}}(\rho|\eta, \mathcal{T})$, uses Equation 18. All that is left to specify is the likelihood in each node and the prior structure for the parameters in each node, both of which are done in the next subsection.

2.2.4 Node Likelihoods and Priors

CGM discuss three models. Two of the models use Gaussian priors and Gaussian likelihoods and one of the models uses a Dirichlet prior and a multinomial likelihood. The two Gaussian models differ in that one has a single variance and the other has a different variance for each node. As noted by Lee [**?**], in a greedy optimization context, sometimes the data suggest a different model than either a Gaussian or a multinomial-Dirichlet. If the experiment suggests analyzing data using an alternate model, the Bayesian context easily handles these alterations, once the corresponding

likelihood and prior are specified. Lee [**?**], proposed a zero inflated poisson (ZIP) model to analyze the solder data. Note our Bayesian model can easily handle extensions such as ZIP response data and also permits covariate selection, provided the integrated likelihood is available in closed form.

We begin with the Gaussian likelihood and Gaussian prior model. We define the likelihood as

$$N[y_{ij}|\mu_i, \sigma^2]. \tag{23}$$

Also, we define the prior for μ_i as

$$N[\mu_i|\bar{\mu}, \sigma^2]. \tag{24}$$

Furthermore, we define the prior for σ^2 as

$$\text{Inv-Gamma}(\sigma^2|\alpha, \beta). \tag{25}$$

All that remains is to evaluate the integral

$$\prod_{i=1}^{b} \int_0^{\infty} \int_{-\infty}^{\infty} \prod_{j=1}^{n_i} N[y_{ij}|\mu_i, \sigma^2] N[\mu_i|\bar{\mu}, \sigma^2] \text{Inv-Gamma}(\sigma^2|\nu/2, \nu\lambda/2) d\mu_i d\sigma^2. \tag{26}$$

For the Gaussian prior and likelihood we can explicitly calculate the marginal likelihood. Being able to marginalize the node parameters explicitly allows us to implement a Metropolis-Hastings algorithm without resorting to complicated, specialized algorithms, or numerical integrations. Straightforward analytic manipulations yield the solution to Equation 26 written in Equation 27

$$\frac{ca^{b/2}}{\prod_{i=1}^{b} \sqrt{n_i + a}} \times \left(\sum_{i=1}^{b} \left(\sum_{j=1}^{n_i} (y_{ij} - \bar{y}_i)^2 \right) + \frac{(\bar{y}_i - \bar{\mu})^2 (n_i a)}{n_i + a} \right)^{-(\nu+n)/2}. \tag{27}$$

If we assume instead that the variances might change from node to node, then the stated model is misspecified. Let us denote the variance in each node as σ_i^2 and keep all other notations from the stated model specification. Then the model is specified using

$$N[y_{ij}|\mu_i, \sigma_i^2]. \tag{28}$$

Also, we define the prior for μ_i as

$$N[\mu_i|\bar{\mu}, \sigma_i^2]. \tag{29}$$

Furthermore, we define the prior for the σ_i^2s as

$$\text{Inv-Gamma}(\sigma_i^2|\nu/2, \nu\lambda/2) \tag{30}$$

and now we evaluate the integral equation

$$\prod_{i=1}^{b} \int_0^\infty \int_{-\infty}^\infty \prod_{j=1}^{n_i} N[y_{ij}|\mu_i, \sigma_i^2] N[\mu_i|\bar{\mu}, \sigma_i^2] \text{Inv-Gamma}(\sigma_i^2|\nu/2, \nu\lambda/2) d\mu_i d\sigma_i^2. \tag{31}$$

The result of computing the integrals in Equation 31 is

$$\prod_{i=1}^{b} \pi^{n_i/2} (\lambda\nu)^{\nu/2} \sqrt{\frac{a}{n_i + a}} \frac{\Gamma((n_i + \nu)/2)}{\Gamma(\nu/2)} \times \left(\sum_{j=1}^{n_i} (y_{ij} - \bar{y}_i)^2 + \frac{(\bar{y}_i - \bar{\mu})^2 (n_i a)}{n_i + a} + \nu\lambda \right)^{(n_i+\nu)/2}. \tag{32}$$

These are the two "regression" models for the Bayesian decision trees given in CGM [**?**].

The classification model discussed in CGM [**?**] defines the likelihood, prior, and integrated likelihood as

$$y_{i1}, \ldots, y_{in_i}|\mathcal{T} \sim \text{Multinomial}(Y_i|\underline{n}, \underline{p}), \tag{33}$$

$$\underline{p}|\mathcal{T} \sim \text{Dirichlet}(\underline{p}|\underline{\alpha}), \tag{34}$$

and

$$\Pr(Y|\mathcal{T}, X) = \left(\frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\prod_{k=1}^K \Gamma(\alpha_k)} \right)^b \prod_{i=1}^{b} \left(\frac{\prod_{k=1}^K \Gamma(n_{ik} + \alpha_k)}{\Gamma(n_i + \sum_{k=1}^K \alpha_k)} \right), \tag{35}$$

respectively.

If we wanted to model the data using a different data generating process, for example a zero-inflated Poisson, we could do so by specifying a different likelihood, prior, and by computing the integrated likelihood. For a zero-inflated Poisson model defining a likelihood, prior, and calculating the integrated likelihood is possible using gamma priors for the rate (λ) and beta priors for zero inflation components (ϕ).

2.2.5 A Bayesian Zero Inflated Poisson Model

Lee and Jin [**?**] reconsidered impurity functions in light of the connection to likelihood functions. Lee and Jin [**?**] proposed to use likelihood functions instead of impurity functions that model

the data generating process. Towards this end they considered the soldering data from Chambers and Hastie [**?**]. The response of interest in this case is a collection of counts on manufactured circuit boards. This response has many zero values and Lee and Jin [**?**] proposed using a zero inflated (ZIP) Poisson likelihood to model the measured counts. Lee and Jin [**?**] optimized using a greedy algorithm and they found the fit and holdout prediction to be better using the ZIP model in each terminal node. If we are to use a Bayesian approach to this problem, we need to define the likelihood, the prior, and the integrated likelihood. We now define these three quantities.

The likelihood for a single observation is

$$f(y|\lambda,\phi) \propto \mathbb{1}(y=0)\left(\phi + (1-\phi)\exp\left(-\lambda\right)\right) + \mathbb{1}(y>0)\left(\exp\left(-\lambda\right)\frac{\lambda^y}{y!}\right). \tag{36}$$

The priors for λ and ϕ are

$$\pi(\phi,\lambda) \propto \underbrace{\frac{\phi^{\alpha-1}(1-\phi)^{\beta-1}\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}}_{=\text{A beta prior}} \times \underbrace{\frac{\lambda^{\alpha_\lambda-1}\exp\left(-\lambda/\beta_\lambda\right)}{\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}}}_{=\text{A gamma prior}}. \tag{37}$$

Now we need to calculate the integrated likelihood, which means that we must evaluate

$$\int_0^1\int_0^\infty \left(\mathbb{1}(y=0)\left(\phi + (1-\phi)\exp\left(-\lambda\right)\right) + \mathbb{1}(y>0)\left(\exp\left(-\lambda\right)\frac{\lambda^y}{y!}\right)\right) \tag{38}$$
$$\frac{\phi^{\alpha-1}(1-\phi)^{\beta-1}\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \times \frac{\lambda^{\alpha_\lambda-1}\exp\left(-\lambda/\beta_\lambda\right)}{\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}} d\lambda d\phi.$$

Let j index the observed zero counts. Furthermore, let $\bar{y}_+$ denote the average of the non-zero counts and n_0 and n_+ denote the number of zeros and non-zeros in the data respectively. Now we assume that the observations are $i.i.d.$ and simple calculations lead to the conclusion that

$$\begin{aligned}\Pr(Y|X,\mathcal{T}) = &\left[\sum_{j=0}^{n_0}\binom{n_0}{j}\frac{\Gamma(\alpha+\beta)\Gamma(\alpha+j)\Gamma(n_0+\beta-j)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha+\beta+n_0)}\left(\frac{n_0-j+\beta_\lambda^{-1}}{\beta_\lambda}\right)^{\alpha_\lambda}\right] \\ &+ \frac{\Gamma(\alpha_\lambda+n_+\bar{y}_+)}{\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}}\left(n_+ + 1/\beta_\lambda\right)^{\alpha_\lambda+n_+\bar{y}_+}.\end{aligned} \tag{39}$$

A similar calculation may be performed for a response variable that is distributed as a zero-inflated negative binomial random variable (ZINB).

2.3 Previous Variable Selection

Previous approaches to variable selection have focused primarily on the linear model or GLM or GLMM models, all of which we briefly reviewed in Chapter 1. Hereafter we will refer to all three types of models as linear. The correspondence between variable selection in linear models and in Bayesian decision trees is a simple correspondence between zeros in the linear model, or zero means of the normal and transformed values on the unit simplex. This correspondence will be detailed in the next subsection of this chapter.

Ishwaran et al. [**?**] propose a modification to Variable Importance (VIMP) criteria that allows some very basic theory to describe the VIMP's analytical properties. VIMP was first proposed by Breiman [**?**] as a method to assess which covariates are important in a dataset. The difficulty with both Ishwaran et al's method and Breiman's method is that they are both very much black box techniques. VIMP basically takes a split in a decision tree on a specific covariate and randomly decides if the observation should go to the left or the right child node. This is done for a collection of observations and the random predictions are averaged against the actual predictions. The resulting numeric estimates for each covariate are the VIMP estimates. A ranking of these values provides a ranking of the covariates.

A similar method of ranking covariates was proposed by Taddy, Gramacy, and Polson [**?**]. Taddy, Gramacy and Polson proposed a Bayesian approach and used the Bayesian equivalent to $\Delta\phi$ from this chapter, written here for clarity,

$$\Delta\phi = \int \phi ds - \pi_L \int \phi ds - \pi_R \int \phi_R ds. \tag{40}$$

Taddy, Gramacy and Polson proposed using samples and estimating the integrals with Monte Carlo approximations. In this case the approach is using a greedy measure on a Bayesian problem, something we find confusing. In the Bayesian approach the quantity $\Delta\phi$ has little meaning, because we are using an MCMC approach to search across trees. The resulting estimates of the covariates' importance will be the same as the greedy approach, which we find undesirable for many reasons. The method proposed by Taddy, Gramacy, and Polson is worth comparing against our approach because it will have similar difficulties to CGM in high-dimensional data. We will do compare

their approach to a time series version of our model in future work.

2.4 Derivations

This subsection contains the mathematical details for calculating the integrals to get the closed form solutions for the integrated likelihood equations of the ZIP model described in this chapter.

2.4.1 ZIP Derivations

In this section we provide the derivation for the integrated likelihood for the Bayes ZIP (zero inflated Poisson) tree model. Now let us define our notation, j will index all observations or only the observed zero observations with the upper limit being n or n_0 if the index is all observations or observed zero counts only, respectively. Also j' will exclusively index the non-zero observations. The total number of non-zero observations is denoted n_+, so that $n_+ + n_0 = n$, the total number of observations. Finally, let $\bar{y}_{i+}$ denote the sample mean of the non-zero count observations in terminal node i.

The integrated likelihood is:

$$\begin{aligned}
\Pr(Y|X,\mathcal{T}) &= \prod_{i=1}^{b}\int_0^1\int_0^\infty \prod_{j=1}^{n_i}\left[\mathbb{1}[y_{ij}=0](\phi+(1-\phi)\exp(-\lambda))+\mathbb{1}[y_{ij}>0]\frac{\exp(-\lambda)\lambda^{y_{ij}}}{y_{ij}!}\right]\pi(\phi_i,\lambda_i)d\lambda_i d\phi_i \\
&= \prod_{i=1}^{b}\int_0^1\int_0^\infty \underbrace{\prod_{j=1}^{n_0}(\phi+(1-\phi)\exp(-\lambda))\pi(\phi_i,\lambda_i)d\lambda_i d\phi_i}_{=(1)} \\
&\quad + \prod_{i=1}^{b}\int_0^1\int_0^\infty \underbrace{\prod_{j'=1}^{n_+}\frac{\exp(-\lambda)\lambda^{y_{ij'}}}{y_{ij'}!}\pi(\phi_i,\lambda_i)d\lambda_i d\phi_i}_{=(2)}\,.
\end{aligned}$$

We will first tackle (1), then (2).

$$
\begin{aligned}
(1) &= \int_0^1 \int_0^\infty \prod_{j=1}^{n_0} (\phi + (1-\phi)\exp(-\lambda))\pi(\phi_i, \lambda_i) d\lambda_i d\phi_i \\
&= \int_0^1 \int_0^\infty (\phi + (1-\phi)\exp(-\lambda))^{n_0} \pi(\phi_i, \lambda_i) d\lambda_i d\phi_i \\
&= \int_0^1 \int_0^\infty \sum_{j=1}^{n_0} \binom{n_0}{j} \phi^j (1-\phi)^{n_0-j} \exp(-(n_0-j)\lambda))\pi(\phi_i)\pi(\lambda_i) d\lambda_i d\phi_i.
\end{aligned}
$$

Now, we take $\pi(\phi_i)$ to be a beta(α, β) prior and $\pi(\lambda_i)$ to be a gamma$(\alpha_\lambda, \beta_\lambda)$ prior. This simplifies matters greatly. The last equation above with a double integral becomes:

$$
\begin{aligned}
&\int_0^1 \int_0^\infty \sum_{j=1}^{n_0} \binom{n_0}{j} \phi^j (1-\phi)^{n_0-j} \exp(-(n_0-j)\lambda)) \frac{\Gamma(\alpha+\beta)\phi^{\alpha-1}(1-\phi)^{\beta-1}}{\Gamma(\alpha)\Gamma(\beta)} \frac{\lambda^{\alpha_\lambda - 1}\exp(-\lambda/\beta_\lambda)}{\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}} d\lambda_i d\phi_i \\
&= \sum_{j=1}^{n_0} \binom{n_0}{j} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}} \underbrace{\int_0^1 \phi^{j+\alpha-1}(1-\phi)^{\beta+n_0-j-1} d\phi_i}_{\text{a beta kernel}} \underbrace{\int_0^\infty \lambda^{\alpha_\lambda - 1}\exp(-(n_0 - j + \beta_\lambda^{-1})\lambda) d\lambda_i}_{\text{a gamma kernel}} \\
&= \underbrace{\sum_{j=1}^{n_0} \binom{n_0}{j} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}} \frac{\Gamma(\alpha+j)\Gamma(\beta+n_0-j)\Gamma(\alpha_\lambda)}{\Gamma(\alpha+\beta+n_0)} (n_0 - j + \beta_\lambda^{-1})^{\alpha_\lambda}}_{=(1)}.
\end{aligned}
$$

Now with the first piece simplified, we move on to piece (2).

$$
\begin{aligned}
(2) &= \int_0^1 \int_0^\infty \prod_{j'=1}^{n_+} \frac{\exp(-\lambda)\lambda^{y_{ij'}}}{y_{ij'}!}\pi(\phi_i,\lambda_i)d\lambda_i d\phi_i \\
&= \int_0^\infty \prod_{j'=1}^{n_+} \frac{\exp(-\lambda)\lambda^{y_{ij'}}}{y_{ij'}!}\pi(\lambda_i)d\lambda_i \\
&= \int_0^\infty \frac{\exp(-n_+\lambda)\lambda^{n_+\bar{y}_{i+}}}{\prod_{j'=1}^{n_+} y_{ij'}!}\pi(\lambda_i)d\lambda_i \\
&= \int_0^\infty \frac{\exp(-n_+\lambda)\lambda^{n_+\bar{y}_{i+}}}{\prod_{j'=1}^{n_+} y_{ij'}!}\frac{\lambda^{\alpha_\lambda-1}\exp(-\lambda/\beta_\lambda)}{\Gamma(\alpha_\lambda)}d\lambda_i \\
&= \frac{\int_0^\infty \exp(-(n_+ + \beta_\lambda^{-1})\lambda)\lambda^{n_+\bar{y}_{i+}+\alpha_\lambda-1}d\lambda_i}{\Gamma(\alpha_\lambda)\prod_{j'=1}^{n_+} y_{ij'}!} \\
&= \underbrace{\frac{\Gamma(n_+\bar{y}_{i+}+\alpha_\lambda)(n_+ + \beta_\lambda^{-1})^{n_+\bar{y}_{i+}+\alpha_\lambda}}{\Gamma(\alpha_\lambda)\prod_{j'=1}^{n_+} y_{ij'}!}}_{=(2)}.
\end{aligned}
$$

This is all the result we need putting the two parts together now gives the closed form for the integrated likelihood. We rewrite Equation 39 here for completeness,

$$
\begin{aligned}
\Pr(Y|X,\mathcal{T}) = &\left[\sum_{j=0}^{n_0}\binom{n_0}{j}\frac{\Gamma(\alpha+\beta)\Gamma(\alpha+j)\Gamma(n_0+\beta-j)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha+\beta+n_0)}\left(\frac{n_0-j+\beta_\lambda^{-1}}{\beta_\lambda}\right)^{\alpha_\lambda}\right] \\
&+ \frac{\Gamma(\alpha_\lambda+n_+\bar{y}_+)}{\Gamma(\alpha_\lambda)\beta_\lambda^{\alpha_\lambda}}(n_+ + 1/\beta_\lambda)^{\alpha_\lambda+n_+\bar{y}_+}.
\end{aligned}
$$

3 Bayesian Decision Tree Models

In Sections 3.1 and 3.2 we describe how we use the ALT to link sparse linear models with covariate selection probabilities facilitating sparse Bayesian decision trees. In Section 3.3 we describe the tree prior and the local updates we use to explore the graph of permissible trees. Finally, in Section 6.6 we give several examples of regularization priors we will study in our simulated examples.

3.1 The Likelihood

We begin by reviewing the model proposed in Chipman et al. [**?**], and show how our approaches generalize upon their model.

The response data is usually assumed to be observed from a normal or multinomial likelihood function. Nonetheless, other forms of response data such as zero inflated models are possible [**?**]. The likelihood is denoted

$$\mathcal{L}(\mathcal{T}, \underline{q}, \underline{\theta} | \underline{y} X) = \Pr(\underline{y} | X, \mathcal{T}, \underline{q}, \underline{\theta}). \tag{41}$$

Here the $\underline{y}$ denotes the vector of responses, X denotes the matrix of covariates, and $\mathcal{T}$ denotes the tree. The vector $\underline{q}$ is the vector of covariate selection probabilities and $\underline{\theta}$ denotes the vector of parameters in the terminal nodes of the tree. The parameters that we are interested in are: $\mathcal{T}$, $\underline{\theta}$, and $\underline{q}$. We factorize the prior in a similar fashion to Chipman et al. using the breakdown

$$\Pr(\mathcal{T}, \underline{q}, \underline{\theta}) \propto \Pr(\underline{\theta} | \mathcal{T}) \Pr(\underline{q} | \mathcal{T}) \Pr(\mathcal{T}). \tag{42}$$

The framework in Chipman et al. [**?**] proposed assumed $\underline{q} \propto 1/p$. Although we find this prior to be a strong *a priori* assumption to take with a large dimensional covariate space, this prior will be seen as a special case of the SSVS prior presented in Section 6.7. Practical difficulties with the uniform prior were already noted in the discussion of Chipman et al. [**?**], although, little work has been done in the interim to remedy this drawback. We modify the framework of Chipman et al. to allow the $\underline{q}$ to vary, depending on their relevance. We specify a prior on the μ_j, a linear space, and use the ALT to determine (indirectly) the prior on the probability space of the $\underline{q}$. This

prior specification on the μ_j allows one to use the methods of the sparse linear model literature but instead apply these methods to decision trees.

The prior density for $\underline{q}$, the covariate selection probabilities, is defined in Subsection 3.2. Similar to Chipman et al., we place the uniform prior, denoted $\Pr(\mathcal{T})$, on the decision tree space. We sample the decision trees using the stochastic process defined in Section 3.3. To monitor the convergence of the Markov chain used to sample the space of decision trees, it is necessary to be able to calculate the marginal density for $\mathcal{T}$ given the covariate selection probabilities $\underline{q}$. This requires choosing conjugate priors for the vector $\underline{\theta}$ so that a closed form marginal density, $\Pr(\underline{y}|X, \underline{q}, \mathcal{T})$ is available. Thus, if the response data are continuous, we choose a Gaussian density for the prior on $\underline{\theta}$. Similarly, if the response data are categorical, we choose a Dirichlet density as the prior density for $\underline{\theta}$.

3.2 The Additive Logistic Transform

The ALT, defined in Equation 92, is a mapping from a Euclidean space to a probability space. The ALT was proposed previously as a fundamental transform for use in compositional data analysis [**?**].

The additive logistic normal (ALN) density is obtained as a result of transforming a multivariate normal density using the ALT. The ALN density is given in Equation 43, however we will rarely find use for this equation. Rather, the density on $\underline{\mu}$ is a multivariate normal density with a linear scale. When a probability scale is needed, we apply the ALT to the normal random variates, $\underline{\mu}$, to obtain probabilities.

$$f(\underline{q}|\mu, \Sigma) = \left[(2\pi)^p |\Sigma| \left(\prod_{j=1}^{p} q_j \right)^2 \right]^{-1/2} \exp\left[-\frac{1}{2} (log(\underline{q}_{(p)}/q_p) - \underline{\mu})^T \Sigma^{-1} (log(\underline{q}_{(p)}/q_p) - \underline{\mu}) \right]. \tag{43}$$

In Equation 43 the notation $\underline{q}_{(p)}$ indicates a vector with the pth entry removed. This density is the result of applying the transformation $q_j = \frac{e^{\mu_j}}{1+\sum_{i=1}^{p-1} e^{\mu_i}}$, for $i = 1, \ldots, p-1$, and the p-dimensional vector, $\underline{\mu}$, has a multivariate normal density with mean vector $\underline{\mu}'$ and covariance matrix Σ. The ALN distribution is defined on the $p-1$ dimensional simplex so that $\sum_{j=1}^{p} q_j = 1$. Various

settings of the parameters $\underline{\mu}'$ and Σ correspond to probabilities concentrated on different regions of the $p-1$ simplex. In practice we do not make use of Equation 43, instead we sample from multivariate normals and transform our random variables using the ALT. Thus the transformed sampled variates, now representing covariate selection probabilities, have a density defined by Equation 43. To help the reader gain intuition into this distribution, we generate plots for the cases where $\underline{q}$ is a 3-dimensional vector. The simplex plots illustrate the effect of changes in the multivariate normal parameters on the density in the simplex space.

In Figure 4, the four sub-figures represent sampled observations from a normal distribution after applying the ALT. The extreme points in the simplex correspond to sparse covariate selection probabilities and therefore result in sparse decision trees. In (c), we see data concentrated mostly around the point of the simplex $(1/3, 1/3, 1/3)$, corresponding to independent standard normals. In sub-figure (d), we see the effect of increasing the variance of the normal densities while maintaining zero means. The larger variances make the draws more likely to come from regions near the edge of the simplex. In sub-figure (e), we see the effect of adding correlation to the multivariate normal density while keeping the mean vector equal to zero. In this parametrization the density lies close to a plane and after transformation to the simplex space the density is concentrated around a line. In sub-figure (f), we see the result of shifting the mean from the zero to the point $(-2, 2, 0)$. In our simulation studies the parameter changes will be less pronounced and less isolated. Often we will have the four types of changes presented in these images occurring simultaneously.

3.3 The Tree Prior

This section reviews the tree prior first described in Chipman et al. [**?**], for those readers that are not familiar with the model. Those readers familiar with the paper may safely skip this subsection.

The tree prior is defined as

$$\Pr(\mathcal{T}|\underline{q}) = \prod_{\eta \in N} \Pr{}_{\text{split}}(\eta) \Pr{}_{\text{rule}}(\eta, j, r_j|\underline{q}), \tag{44}$$

where η ranges over all nodes in the tree, denoted by the index set N, and the two probabilities in

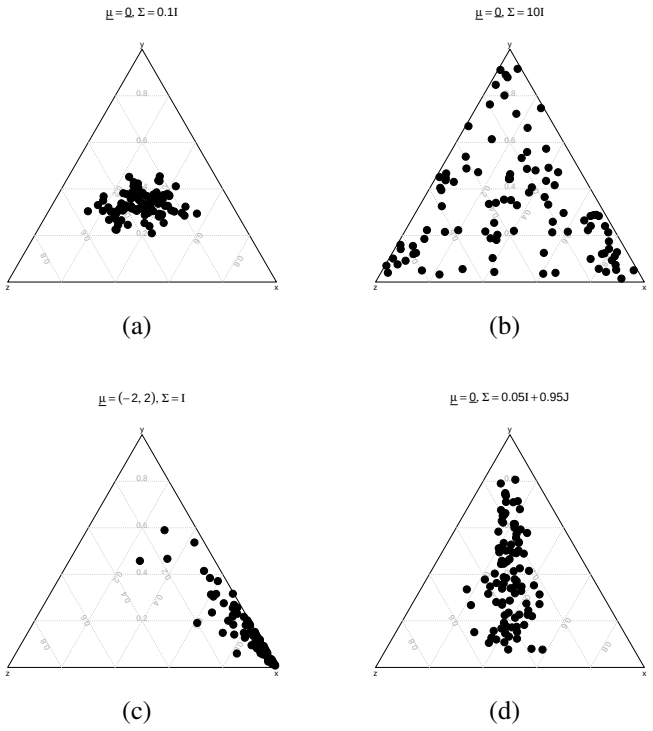


Figure 4: ALN plots with various multivariate normal parameters. Subfigure (a) contains multivariate standard normal draws. Subfigure (b) plots multivariate normal draws with zero mean and large independent variances. In subfigure (c), contains unit independent variances with mean vector $\mu = (-2, 2, 0)$. Finally, in subfigure (d) we add correlations while keeping the mean vector as a zero vector.

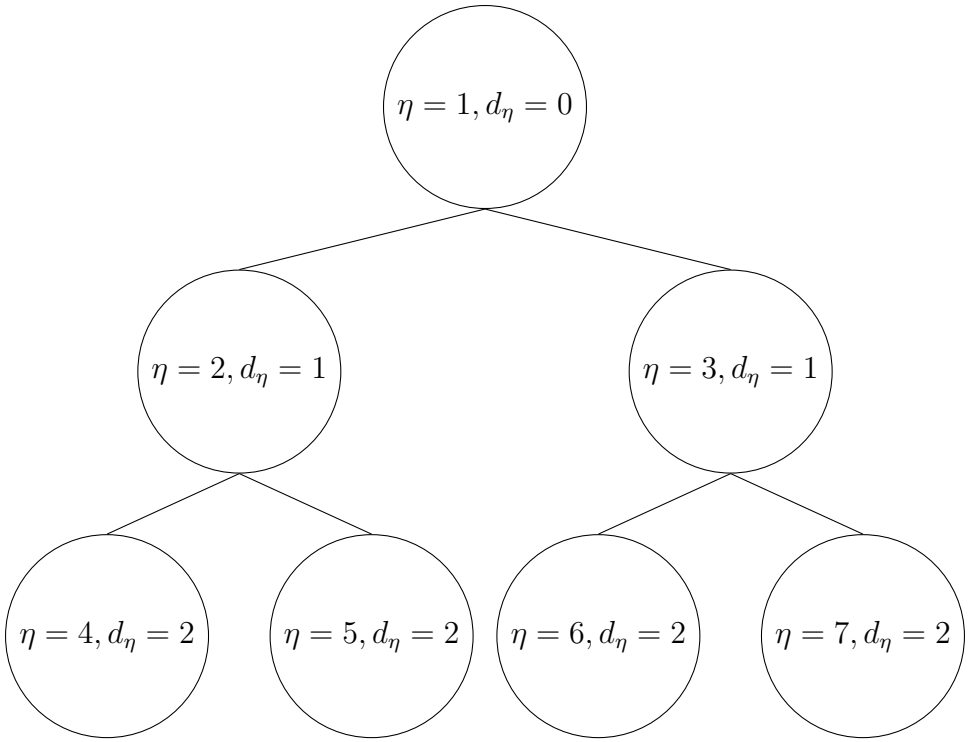


Figure 5: The binary heap node numbering system used in our simulations.

Equation 44 are the probabilities of a split at the node η and the probability of selecting a specific split rule at node η respectively. Also, r_j denotes the splitting value of the split rule at node η on the jth covariate. The probability of a split in a tree at node η is defined by

$$\Pr{}_{\text{split}}(\eta) = \alpha(1 + d_\eta)^{-\beta}, \tag{45}$$

where the quantities $\alpha > 0$ and $\beta \geq 0$ are fixed values specified *a priori* and $d_\eta = \lfloor lg(\eta) \rfloor$ is the depth of the node η ($\lfloor - \rfloor$ denotes the floor function). In this paper lg denotes base 2 logarithms and log denotes base e logarithms. We number the nodes in the tree according to the binary heap node numbering scheme used in many binary tree applications. For a good review of the binary heap node numbering system see Cormen, Lieserson, Rivest, and Stein [**?**]. We define the root node, labeled node number one, to have depth zero. Furthermore, nodes two and three are the left and right child nodes of node one respectively. Both nodes two and three are defined to have depth one. The depths of other nodes in the tree follow similarly.

In addition to the process prior defined on the tree, we define a process prior on the selection of covariates for split rules within a tree. The function $\Pr_{\text{rule}}(\eta, j, r_j|\underline{q})$ is decomposed into two components, one selecting the jth covariate which we use to propose a split at the node, and a second component selecting a specific split value (r_j). This corresponds to the conditional probability decomposition

$$\Pr_{\text{rule}}(\eta, j, r_j|\underline{q}) = \underbrace{\Pr_{\text{cov}}(j, \eta|\underline{q})}_{\text{ALN density}} \underbrace{\Pr_{\text{split value}}(r_j|j, \eta)}_{\text{Uniform}}. \tag{46}$$

The tree prior, denoted $\Pr(\mathcal{T})$, is defined uniformly over the space of admissible trees. Exact enumeration of all admissible trees is only possible for very small values of n, the number of observations. The number of admissible trees is a cumulative sum of Catalan numbers less than or equal to a given depth. This sum grows at the rate $\mathcal{O}(4^n/n^{3/2})$ [**?**], the same rate as the Catalan numbers themselves, displayed in Figure 6, and becomes intractable quickly. Fortunately, Markov chain Monte Carlo sampling allows us to sample from the space of trees using a Metropolis-Hastings rule with local proposals on the space of trees. We now detail the structure of the proposal functions.

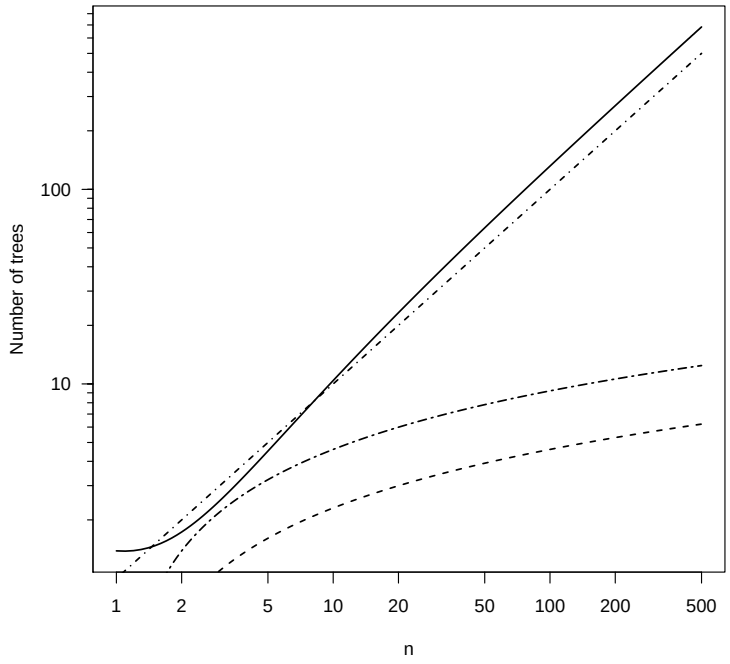


Figure 6: Plots of the log of the number of possible trees (solid line) less than or equal to a given depth n, compared to exponential (45 degree line), quadratic (dash-dotted line) and linear functions (even-dashed line). Note the vertical axis is on a log scale.

Given a tree $\mathcal{T}$, we sample a new tree adjacent to $\mathcal{T}$ by proposing local updates. The local updates we use are the following:

- The Grow step: choose at random one of the terminal nodes and propose to append two new child nodes with probabilities given by Equation 45.
- The Prune step: the reverse of the grow step, a terminal node is selected at random and the selected node and that selected node's sibling are both pruned to the immediate parent of the two child nodes. The parent node becomes a terminal node.
- The Change step: randomly pick an internal node and propose to change the split rule at the selected node with that of another observation, possibly on a different covariate.

- The Swap step: randomly select an internal node that is not the immediate parent of any terminal node and proposes to swap the split rules of the parent-child pair. Otherwise, when both child nodes split on the same covariate, both child node's rules and the parent node's rules are swapped.

- The Rotate step: randomly choose a left or right rotation move, each with probability $1/2$. Then a rotatable node is chosen and a rotate move is performed.

The rotate operation for binary trees was first introduced in Sleator and Tarjan [**?**] and was implemented with Bayesian decision trees by Gramacy and Lee [**?**]. The rotate move was originally suggested by Knight, Kustra, and Tibshirani [**?**] as another possible move that might improve mixing. A good introduction and several practical uses of the rotate move can be found in Cormen, Lieserson, Rivest, and Stein [**?**]. The proposal functions in Gramacy and Lee [**?**] perform a rotate move only when a swap move is proposed and the parent-child pair both split on the same covariate. We modify this and allow rotate moves to be a separate proposal function and not a special swap move case. The proposal mechanism of Chipman et al. uses the grow, prune, change and swap moves only. In addition, neither Chipman et al. [**?**], Denison et al. [**?**], nor Gramacy and Lee [**?**] included weights on each covariate in their examples or model specifications. They sampled each covariate and each split value uniformly across all covariates.

4 Dirichlet Variable Selection For Decision Trees: The DiVaS method

Many data analysis procedures, originally designed for inference in low dimensional spaces, create difficulties when dealing with high dimensional data. Often data analysts use low dimensional intuition and apply similar logic to high dimensional data, which can often lead to erroneous conclusions. The approach outlined in this thesis not only moves around the high dimensional space efficiently, but also indicates with a high degree of probability which covariates are useful in determining the response at the tree nodes. In addition, we retrieve summary data indicating which dimensions are useful assisting the data analyst in the interpretation of results. Our primary focus in this thesis is on inferring a structural relationship between dimensions and the response of interest, with a preference towards interpretability.

There are perhaps two predominant approaches to dealing with high dimensional data. The first attempts to only include dimensions that are important based on some metric of choice, usually with a filter run over the data to screen out some dimensions. Examples of this are principal components analysis [**?**] and various other approaches surveyed by Dash and Liu [**?**]. The second approach attempts to fit a model to the best dimensions sequentially. Examples of this approach include forward, backward and stagewise searches for regression models [**?**] [**?**] [**?**]. The lasso [**?**] is a problem which can be considered in either category, depending upon the approach taken. If we use the LAR algorithm to fit the model for all values of λ then the method can be considered iterative. If instead, we fit a model for the lasso objective function choosing a single value of λ, for example by cross validation, then the method can be thought of as a filtering approach. Our approach is similar to both approaches and operates by choosing a middle ground between these two extremes. As a byproduct of choosing this middle ground we are able to search the space of models more effectively and give intuitive summary results indicating which covariates are useful, while returning models to the analyst for use in prediction and inference.

In this chapter we follow a Bayesian framework. We define a new prior measure over trees, which facilitates comparison across trees based on a global search of the tree space, and automat-

ically induces coherent covariate selection. We assume a working familiarity with Markov chain Monte Carlo (MCMC) methods. For a good review of MCMC approaches to machine learning problems, we recommend the paper by Andrieu, De Freitas, Doucet and Jordan [**?**]. We show through an example that greedily grown decision trees can fail to recover good trees in high dimensional data. Moreover, we find that previously implemented Bayesian approaches to decision tree learning have been applied only to low dimensional spaces [**?**] [**?**] [**?**]. Applying the same approaches to high dimensional data is inappropriate and we demonstrate this through examples. These methods fail to move around the large dimensional space efficiently causing poor mixing and only finding local optima.

Nearly all the previous approaches to decision tree induction have not explicitly incorporated model selection into the tree framework [**?**] [**?**] [**?**] [**?**] [**?**]. A notable exception is the recent paper by Gramacy and Taddy [**?**] where binary indicators are used to perform model selection within a terminal node. Specifically, Gramacy and Taddy define a mixture of a Gaussian process (GP) and the GP's limiting linear model, and use a point mass mixture to choose between the GP and the limiting linear model within each terminal node. This is in contrast with our approach, which aims to perform variable selection at the level of the decision tree itself and not for each model in each terminal node. Another notable exception is the work of Ishwaran, Kogalur, Gorodeski, Minn, and Lauer, who define a new quantity called a maximal subtree, and use the inherent topology of the tree and the maximal subtree to measure variable importance [**?**]. Ishwaran et al. work in the context of random survival forests, built using bootstrapped data. Most researchers have used implicit selection, for which predictors are chosen at the end of the greedy induction strategy as the predictors which are useful. Those not selected are considered not useful. Currently, most tree algorithms rely on a pruning rule for determining which covariates are useful. This implicit approach has the drawback of only providing "yes/no" answers to inclusion, and does not allow one to order the dimensions in some meaningful fashion. In our work we explicitly model these probabilities of inclusion and exclusion in the decision tree. This not only allows for useful predictor selection when choosing models, but also improves the efficiency of searching in large dimensions. We note that we include a prune function in our proposal which affects the selection of useful covariates.

However, in large dimensions, a better way to improve the efficiency of the searching algorithm is to treat covariates differently based on their utility in the model.

This chapter follows from the original Bayesian decision tree model of Chipman, George and Mcculloch [**?**] (referred to hereafter as CGM), and Gramacy and Lee [**?**](hereafter referred to as GL). Specifically, we propose a generalization of the models used by CGM and of GL. We allow rotate moves at all permissible nodes of the tree and varying selection probabilities on each covariate. In this manuscript we confine ourselves to categorical response data. All results can easily be extended and applied similarly to continuous data. In addition, we study the approach of CGM to large dimensional datasets. To our knowledge, the largest dimensions of data that the aforementioned approaches have been studied on are at most 15 dimensions. We find our method to be more effective than the CGM approach for large dimensional data. We also study the efficacy of a general rotate move in the transition kernel of the Markov chain.

The remainder of this paper unfolds as follows. Section 4.1 reviews previous greedy and Bayesian approaches to decision tree learning. Section 4.2 shows our model and describes how this is a modification of the algorithm proposed by CGM [**?**]. In this and the next chapter we place emphasis on the algorithmic details of our procedure and only briefly describe some of the mathematical properties. Section 4.3 states necessary conditions for consistency of the decision trees we propose. Section 4.4 shows how high dimensional data can create problems for both greedy and Bayesian approaches that have been previously proposed, and gives an example. Section 7.2 applies our approach to a dataset taken from the UCI machine learning repository. In Section 5.1 we state conclusions and point towards future work.

4.1 Related Work

Some of the most well known approaches to decision tree learning are two approaches invented in the 1980's; these are Breiman, Friedman, Olshen, and Stone's cart method [**?**] and Quinlan's C4.5 algorithms and variants [**?**]. The work of Breiman et al. [**?**] showed the practical and theoretical approaches to decision tree induction. In Breiman et al. [**?**], the theoretical framework for discussing decision trees is given in terms of general impurity functions, allowing results to apply to

specific impurities suggested by Quinlan [**?**]. For a theoretical treatment of consistency, the work of Devroye, Lugosi, and Györfi [**?**] provided the foundation and application of the theory to certain tree methods.

The consistency of tree classifiers was also discussed by Breiman et al. [**?**]. They showed that, under fairly general conditions, tree classifiers are asymptotically consistent, meaning trees can recapture the correct classifier if they are given enough data. The practical problem with this is that we usually only have a finite amount of data. Although we do not overcome this practical limitation, the results in Section 4.3 improve the rate of convergence by exploiting sparsity. Quinlan's approach modified the impurity functions proposed by Breiman [**?**], and also allowed for multiway splits at nodes in the tree [**?**]. Empirically, we have found that the entropy impurity function approach of Quinlan [**?**] performs better if there is a difference, although there is usually little or no difference in terms of the final tree.

In our probabilistic approach, we quantify *a priori* uncertainty in the classifier with a prior probability measure over trees. By defining a prior over trees, we induce a posterior distribution over trees via Bayes' rule shown in Equation 47, where θ denotes parameters in the model and D denotes observed data.

$$\Pr(\theta|D) = \frac{\Pr(D|\theta)\Pr(\theta)}{\Pr(D)} \tag{47}$$

The posterior allows us to *search* the space of trees, which is distinctly different from the greedy induction approach. In the Bayesian approach to decision tree induction several researchers have proposed various methods for searching over the tree space. The space of trees is a large discrete space, so MCMC algorithms are usually applied to search this space. CGM [**?**] proposed a clever process prior on the tree space, denoted $\Pr(\mathcal{T})$, to induce a posterior which moves between a varying number of dimensions. Let $\mathcal{T}$ denote the tree and θ the collection of parameters in each terminal node (mean or class probabilities) and D denotes data. Applying Bayes' rule, we have the result

$$\Pr(\mathcal{T}|D) = \frac{\int \Pr(D|\theta,\mathcal{T})\Pr(\theta|\mathcal{T})\Pr(\mathcal{T})d\theta}{\Pr(D)}. \tag{48}$$

Note that the integral in the numerator of Equation 48 treats the tree prior as a constant with respect

to θ. The integrated product of the other two quantities in the numerator is called the integrated likelihood and will be referred to in Section 4.2. In the same year, Denison, Mallick, and Smith [**?**] proposed a reversible jump algorithm that has also shown success in searching tree space. As modifications to the work of CGM [**?**], GL [**?**] proposed a Gaussian process model in each terminal node, along with modifying the transition kernel to include rotations in one special case of swap moves.

Bagging and boosting are two other common methods that might be used for performing model selection in the context of decision tree induction [**?**] [**?**]. Often analysts will use these algorithms as black boxes and use the frequency of selected covariates in the resulting trees as measures of importance of decision trees. In the bagging context and the related random subspace method [**?**], one randomly subsamples either observations or dimensions and builds trees on this subsampled data. One of the difficulties with data containing many predictors is that, when we randomly sample with replacement collections of predictors to build trees, we select approximately 63% of the dimensions [**?**]. This is usually still a large feature space inheriting the same problems we initially faced. Bagging and boosting methods are primarily used in the context of prediction. As initially stated, our primary motivation is on inferring a mechanistic relationship between dimensions and the response of interest with a preference towards simple and easily understood models. Hence, inference is the primary goal, with prediction being secondary.

4.2 Model Details

In this section, after we define our approach, we detail the transition kernels proposed by CGM [**?**] and by GL [**?**]. We highlight the differences between their approaches and our approach.

Let us define the prior measure over a decision tree as

$$\Pr(\mathcal{T}) = \prod_{\eta \in N} \Pr{}_{\text{split}}(\eta) \Pr{}_{\text{rule}}(\eta, s_k),$$

where η ranges over all nodes in the tree, and the two probabilities are probabilities of a split at the node η, and the specific rule at node η, respectively. Also, s_k denotes the randomly selected

covariate. The probability of a split in a tree is

$$\Pr{}_{\text{split}}(\eta) = \alpha(1 + d_\eta)^{-\beta},$$

where the quantities $\alpha > 0$ and $\beta \geq 0$ are parameters, and $d_\eta = \lfloor lg(\eta) \rfloor$ is the depth of the node η under the usual binary heap node numbering (cf. Cormen, Lieserson, Rivest and Stein [**?**]). In this document lg denotes a base 2 logarithm, and log denotes a base e logarithm.

For our model we rely on the multinomial Dirichlet conjugate pair

$$\Pr(\underline{p}) \Pr(c_i|\underline{p}) = \underbrace{K \prod_{i=1}^{d} p_i^{\alpha_i - 1}}_{\text{Dirichlet}} \times \underbrace{\binom{n}{c_1, \ldots, c_d} \prod_{i=1}^{d} p_i^{c_i}}_{\text{Multinomial}},$$

where the c_i's are counts and K is a normalizing constant.

The integrated likelihood quantity from the numerator of Equation 48 is a multinomial Dirichlet conjugate pair. Therefore, the integral is available in closed form. We take the Dirichlet parameters (α_is) to be a vector of 1s although other alternatives are possible. The $\Pr_{\text{rule}}(\eta, s_k)$ is decomposed into two components, one selecting the jth covariate to use to propose a split at the node, and a second component selecting a specific split value. This corresponds to the conditional probability decomposition

$$\Pr{}_{\text{rule}}(\eta, s_k) = \underbrace{\Pr{}_{\text{cov}}(s_k)}_{\text{multinomial}} \Pr{}_{\text{splitvalue}}(\eta|s_k). \tag{49}$$

The multinomial distribution in Equation 49 to be a multinomial and we define a prior of choosing covariates to split on as a Dirichlet distribution. These two probability densities are conjugate pairs. This is brought about by assuming the structure $\Pr{}_{\text{cov}}(s_k) \equiv \Pr{}_{\text{cov}}(s_k|s_{k-1}, s_{k-2}, \ldots, s_1)$. To our knowledge, this refinement of the CGM model has not been employed in the literature. The work of Ishwaran et al. used the exact same breakdown as Equation 49 but proposed a different probability density on the covariates. Also Ishwaran et al.'s method was used in the context of random survival trees, decision trees made using bootstrap sampled survival data [**?**].

When we sample a Markov chain we collect trees sampled under the dynamics of Equation 51, a Metropolis-Hastings rule [**?**]. We often find that the observed counts of splits on each covariate

drown out the information from the prior. This is an undesirable effect of the proposed model, because the observed split counts are not observed *a priori*, but during the course of the MCMC algorithm. To combat this, we find it necessary to define the parameter $\widetilde{\alpha}$. This parameter is defined by setting the prior expectation of the probability of splitting proportional to the expectation of the unobserved split probabilities on a covariate written here in Equation 50

$$\widetilde{\alpha} = \frac{C \sum_{j=1}^{d} \alpha_j}{\sum_{i,j} s_{ij}}. \tag{50}$$

The α_j are the current concentration parameters of the Dirichlet distribution and s_{ij} denotes the observed frequency of splits on dimension j at iteration number i. This implies that the posterior for the covariate weights follows a modified Dirichlet distribution,

$$\Pr(\underline{p}|\alpha, \underline{s}_i) = K(\widetilde{\alpha}) \binom{n}{s_{i1}, \ldots, s_{id}} \prod_{j=1}^{d} p_j^{\widetilde{\alpha} s_{ij}}.$$

Here $K(\widetilde{\alpha})$ is a normalizing constant. The extent to which the parameter $\widetilde{\alpha}$ (Equation 50) will impact the chain will depend on the length of each Markov chain. The parameter $\widetilde{\alpha}$ has a greater impact on chains that are run for a greater number of iterations.

The constant C in Equation 50 must be greater than zero and is a specified constant governing the ratio of exploration and exploitation conducted on the covariates of the Markov chain. All MCMC algorithms have the property of exploring the space until a region of high posterior probability is found. Once found, the algorithm exploits the local concavity of this probability region. The extent to which an MCMC algorithm exploits and explores will determine how well the posterior distribution is recovered. The higher the value of C, the more exploration the Markov chain will conduct, and the smaller the value, the more exploitation. In addition, for data sets of differing dimensions, we may wish to use the dimensionality to guide the choice of the exploration and exploitation. Larger dimensional data sets will require a larger value of C to encourage exploration. If the analyst has good prior information in choosing α_j's, i.e. which covariates are useful, then the chain should move more towards exploitation, encouraging the Markov chain to find just the right splits and not focus too much on selecting the right covariates.

The approaches of CGM and GL used discrete uniform probability masses on each covariate

and on each available split value in the current node. We modify this by allowing the selection of covariates to vary from covariate to covariate, while keeping the discrete uniform prior on split values. We take the prior measure on the covariates to be a Dirichlet distribution and the observed counts for splits in the tree from the Markov chain as the pseudo counts of a multinomial likelihood. Briefly, the benefits are:

- Improved searching of the Markov chain.
- Better inferential and predictive trees.
- Easier interpretation of classifiers.

We discuss the model and specification of parameters further in Section 4.4 and in Chapter 7.2.

The transition kernel describes the dynamics governing the state transitions of a Markov chain. To understand this let us introduce some notation. Let X' denote the current state of the Markov chain, and X some new future state. Then $k(X'|X)$ denotes the probability of moving from the current state X' into the new state X, which may be viewed as a conditional density or mass function. In this paper, the states of the Markov chain are trees, denoted as$\mathcal{T}'$ (current) and $\mathcal{T}$ (new). The transition kernel $k(\mathcal{T}'|\mathcal{T})$ is defined by the Metropolis-Hastings rule

$$\alpha(\mathcal{T}'|\mathcal{T}) = min\left(1, \frac{\Pr(\mathcal{T}|D)q(\mathcal{T}'|\mathcal{T})}{\Pr(\mathcal{T}'|D)q(\mathcal{T}|\mathcal{T}')}\right). \tag{51}$$

In Equation 51, $q(\mathcal{T}'|\mathcal{T})$ denotes a proposal function proposing a new tree $\mathcal{T}'$, starting from the current tree $\mathcal{T}$. Our proposal mechanism is as follows:

- The grow step chooses at random one of the terminal nodes and proposes to append two new child nodes with a certain probability that could depend on the tree depth, splitting on a chosen covariate.
- The prune step works in reverse of the grow, a terminal node is selected at random and that node and the node's sibling are pruned to the immediate parent of the two child nodes.
- The change step randomly picks an internal node and attempts to change the split rule at the node with that of another observation, possibly on a different covariate.

- The swap step randomly selects an internal node that is not the root node and proposes to swap the split rules of the parent-child pair. If both child nodes split on the same covariate, then both children and the parent node's rules are swapped.

- The rotate step randomly chooses a left or right rotation move. Then it randomly chooses an admissible internal node and rotates.

Sleater and Tarjan [**?**] introduced the rotate operation for binary trees in computer science. GL [**?**] first introduced the rotate operation into the Bayesian decision tree literature. A good introduction and several practical uses of the rotate move can be found in Cormen, Lieserson, Rivest and Stein [**?**]. The proposal of Gramacy and Lee [**?**] only allows a rotate move for the specific case when a swap move is proposed and the parent child pair both split on the same covariate. We modify this and allow rotate to be a separate operation of the transition kernel and not a special swap move case. The proposal mechanism of CGM uses the grow, prune, change and swap moves only. We also allow swap moves in our proposal. In addition, neither of these papers included weights on each covariate in their examples or model specifications. They sampled each covariate and split value uniformly at random.

4.3 Ensuring Consistent Classifiers

In this section we aim to answer two questions. The first question is what is the behavior of our method as the number of dimensions $d \to \infty$. We generally tend to think of data as having a fixed dimensionality, but as data storage costs decrease, the number of variables tends to increase. We assume that the data has a large number of dimensions and use the theoretical quantity $d \to \infty$ as a guidepost. We use a working example throughout this section, a stump tree, which is a tree containing only one split rule. The second working example is a decision tree with an arbitrary number of splits. Through these two examples we see when the theory discussed in this section works and when the theory fails.

We begin with a few definitions originally provided by Vapnik and Chervonekis [**?**], and subsequently studied by many others [**?**] [**?**]. Let us define the shatter coefficient as the maximum

number of sets obtained by intersecting a finite collection of points with functions from a specified class of functions and denote this as $s(\mathcal{A}, n)$. For our work it will suffice to look at half spaces on $\mathbb{R}^d$, which have shatter coefficient 2^n for $n < d$, but once $n > d$, the shatter coefficient no longer grows exponentially, instead growing polynomially in n. We say a class of functions $\mathcal{A}$ has Vapnik-Chervonekis (VC) dimension $V_{\mathcal{A}}$, where $V_{\mathcal{A}}$ denotes the largest value in the *exponent* of the shatter coefficient, such that the exponent equals the sample size. Formally, let us define this value as $V_{\mathcal{A}} \stackrel{def}{=} max\{n : s(\mathcal{A}, n) = 2^n\}$. Intuitively, the VC dimension denotes a phase change in the shattering dynamics as a function of the sample size, as seen, for example, in Figure 7. This phase change occurs when the growth in the shatter coefficient as a function of n, the sample size, changes from exponential to polynomial. For half spaces on $\mathbb{R}^d$, $V_{\mathcal{A}} = d$ [**?**]. We will use the shatter coefficient and VC dimension in error bounds stated in this section.

Devroye, Györfi, and Lugosi [**?**] provided the following result:

Theorem 1

For $0 < k^2/(k-1) < n\varepsilon^2$,

$$\Pr\left(\sup_{A\in\mathcal{A}}|v_n - v| > \varepsilon\right) \leq 4ks(\mathcal{A}, n)\exp\left[\frac{-n\varepsilon^2}{2k^2}\right]. \tag{52}$$

The notation $v_n \stackrel{def}{=} (1/n)\sum_{i=1}^{n} \mathbb{1}(y_i = \hat{\mathcal{T}}(X_i))$ denotes the empirical error of the estimated classifier and the notation $v \stackrel{def}{=} \Pr(\mathcal{T}(X) = y)$ denotes the Bayes error of the data. This result tells us that if the family of classifiers has a finite VC dimension, or equivalently a finite shatter coefficient, then the classifier is consistent. Here k and ε are specified constants, usually we take $k = 2$ and $\varepsilon = 0.01$. Now we know that a halfspace split in $\mathbb{R}^d$ has $V_{\mathcal{A}} = d$. Define the entropy function as,$\mathcal{H}(x) = -xlog(x) - (1-x)log(1-x))$ for $0 \leq x \leq 1$ with $\mathcal{H}(1) = \mathcal{H}(0) = 0$. A useful inequality is

$$s(\mathcal{A}, n) \leq \exp[n\mathcal{H}(V_{\mathcal{A}}/n)], \tag{53}$$

for $V_{\mathcal{A}} < 2n$. We see that, for $d < 2n$, this inequality holds and, thus, the right hand side of inequality 52 will go to 0, as $n \to \infty$. For our stump, the tree will capture the Bayes error as $n \to \infty$, if the model is correct. We also see that, for $n < d$, the shatter coefficient for half spaces

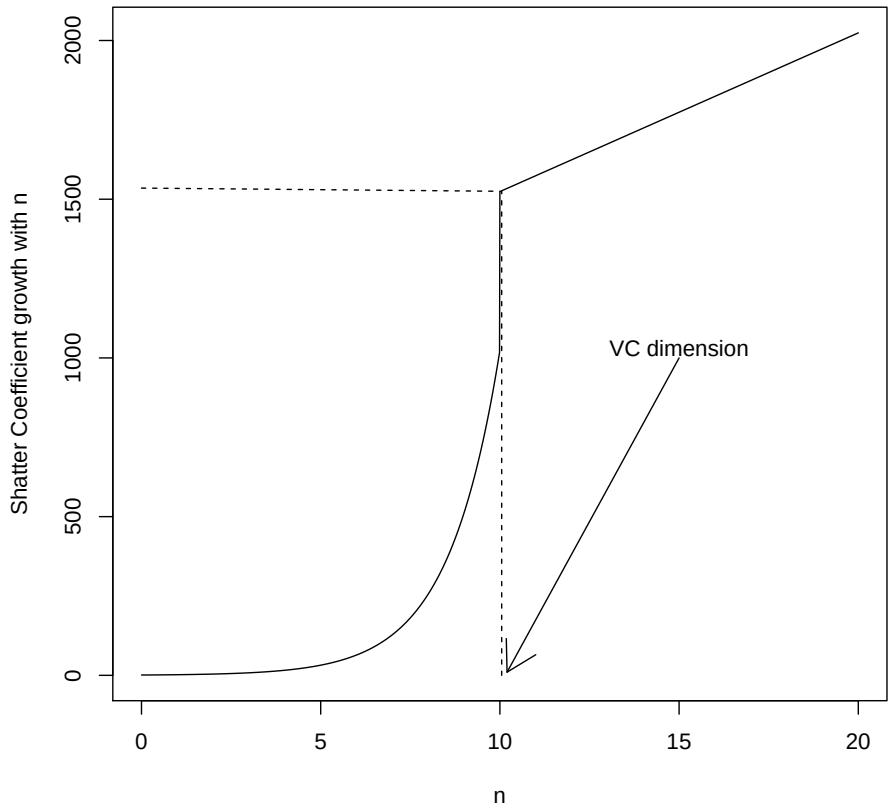


Figure 7: The shatter coefficient function for half spaces in $\mathbb{R}^d$, as a function of n. In this plot $d = 10$.

is 2^n. If $d \to \infty$ and $n \to \infty$, but $n < d$, we have no consistency guarantee, according to this bound. In the second case, let us consider a decision tree of arbitrary depth. Theorem 13.5 of Devroye, Györfi, and Lugosi tells us that for $\mathcal{A} = \{\mathcal{A}_1 \cap \mathcal{A}_2\}$,

$$s(\mathcal{A}, n) \leq s(\mathcal{A}_1, n)s(\mathcal{A}_2, n). \tag{54}$$

For decision trees with two splits on halfspaces, the shatter coefficient is bounded above by $2^{2n} = 4^n$, for sample sizes less than d. For sample sizes larger than d we will have consistency as $n \to \infty$, provided that the decision tree does not grow arbitrarily large. As a simple requirement we ensure that the trees do not grow arbitrarily large, by ensuring that trees are shallower than K,

for some constant K. Note that this constraint was also required by Denison, Mallick and Smith's reversible jump algorithm [**?**]. Our simulation studies indicate that there is little to no sensitivity in the specification of K. Figures 8 -14 show the results of simulating from the posterior distribution over trees and varying the maximum depth, K. We evaluate for values of $K = 2, 3, 4, 5, 6, 7, 10$. For small values of K such as 2 or 3, the setting of maximum depth appears to limit the sampler from exploring possible trees. However, once K is set larger than 6, there appears to be little impact restricting the sampler from exploring certain trees. Of course, for any given data set the value of K that is acceptable will likely change, so our conclusion is that K should be large enough. Our assumption is $K = 20$ or $K = 30$ should be enough for most problems. Simply stated, K should be large. Too small values of K can cause problems. A value of $K = 10$ appears sufficient for our purposes and causes no problems.

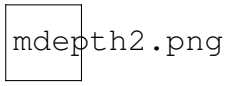


Figure 8: Maximum depth of samplers trees with maximum depth set at 2.

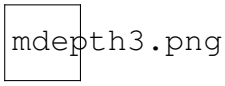


Figure 9: Maximum depth of samplers trees with maximum depth set at 3.

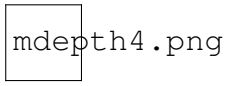


Figure 10: Maximum depth of samplers trees with maximum depth set at 4.

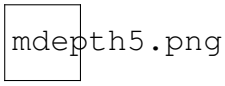


Figure 11: Maximum depth of samplers trees with maximum depth set at 5.

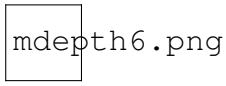


Figure 12: Maximum depth of samplers trees with maximum depth set at 6.

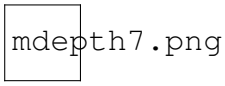


Figure 13: Maximum depth of samplers trees with maximum depth set at 7.

Figure 14: Maximum depth of sampled trees with maximum depth set at 10.

Up to this point, the material in this section is nothing new, but contained fully in the work of Devroye, Györfi, and Lugosi [**?**]. Now let us define the sparse shatter coefficient as the shatter coefficient using only dimensions where $S_j = 1$, denoted $s(\mathcal{A}, n|S_{j=1}^d))$ and $S_j \in \{0, 1\}$. We now use this sparse definition to carry over results into cases where sparsity holds in classification problems.

Theorem 2

For $0 < k^2/(k-1) < n\varepsilon^2$ and $\mathbb{E}(S_j) = p_j$, the bound

$$\Pr\left(\underset{A\in\mathcal{A}}{sup}|v_n - v| > \varepsilon\right) \leq 4ks(\mathcal{A}, n|S_{j=1}^d)) \exp\left[\frac{-n\varepsilon^2}{2k^2}\right] \tag{55}$$

holds, if and only if $\sum_{j=1}^d p_j < \infty$, where d can be finite or infinite. In the case $d < \infty$ everything looks nearly the same and the proof follows verbatim from the derivation in Devroye, Györfi, and Lugosi. The case $d = \infty$ is delicate and requires Kolmogorov's three series theorem [**?**].

The second error bound (Equation 55) differs from the first (Equation 52) in that now we are only using some subset of the covariates to classify the observations. Explicitly, the number of effective dimensions, denoted as d^*, can be bounded almost surely, but the total number of dimensions $d \to \infty$. Each $S_j \in \{0, 1\}$, so if $\Pr(S_j = 1) = 1$, then we are back into the case of the first theorem, which is non sparse shattering. In our case, we relax the assumption that all $\Pr(S_j = 1) = 1$ and instead allow $\Pr(S_j = 1) = p_j$, for $0 \leq p_j \leq 1$. This can be thought of as a convex relaxation of a non-convex, computationally difficult optimization problem. The sum $\sum_j^d S_j$ is now a random sum, since each S_j is a random variable. Also, this random sum is the sparse VC dimension for splits of the form $(\infty, a_i]$, the splits we use to construct our classifiers. Passing to the limit as $d \to \infty$, we want the random infinite series to converge, otherwise we will not have a consistent classifier. The Kolmogorov three series theorem tells us that convergence almost surely of the random series $\sum_{j=1}^\infty S_j$ occurs if and only if the series $\sum_{j=1}^\infty p_j$ converges,

thus necessary and sufficient conditions are $\sum_{j=1}^{\infty} p_j < \infty$.

We have necessary and sufficient conditions for convergence and therefore for consistency of our sparse classifier in infinite dimensional data. We need only ensure that $\sum_{j=1}^{\infty} p_j < \infty$. Provided $\sum_{j=1}^{\infty} p_j < \infty$ the stump tree will be consistent with infinite dimensional data as $n \to \infty$ Furthermore, the tree with arbitrary but finite depth will also be consistent, provided that the tree does not grow too deep, less than 2^K for some large but finite K. Intuitively this result should make sense. For infinite dimensional data, we must have most dimensions be irrelevant for the classification task, in order for the resulting classifiers to be consistent. In other words, most $p_j \to 0$ and some $p_{j'} \to c > 0$ $(0 < c < 1)$ as $d \to \infty$. Our approach, without the $\widetilde{\alpha}$, ensures precisely this condition. Furthermore, this is not a weakness of our approach, because this theoretical result holds for $d \to \infty$. In our real data cases d is finite and usually fixed. The $\widetilde{\alpha}$ is a practical aspect of our approach and is used to improve exploration and exploitation in fixed, finite dimensional data.

4.4 A Simulated Example

This section details a simulated example where we know which covariates are important and which are not important. Of course, this will never be the case in practice, yet, for verification, we find this setting useful.

For our example, we note that a greedily built decision tree with a linear classifier can universally fail, if we are willing to make d large enough. To accomplish this for each covariate after d^*, we will simulate from a mixture of two normals, where each mixing probability is $1/2$, and the means of the two normals are taken from a continuous uniform distribution on a large enough region (l, u). If we can generate data where each covariate has observations in two groups that are separated enough and some of the responses are also separated, then we will incorrectly select a split rule based on this covariate, when in reality this covariate is independent of the response by construction. This will occur more frequently for larger d. Fortunately, our method will still select the correct covariates and as a result, select the correct tree.

Let us define the data generating process (DGP) as that given in Figure 15.

The probability of being the majority class in any of the four regions of the Figure 15 is 90%.

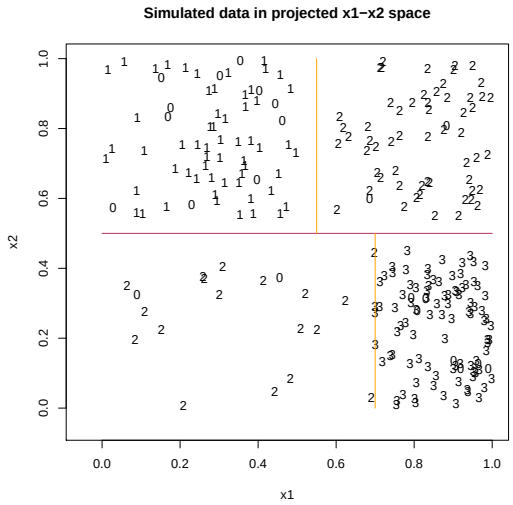


Figure 15: A plot of the true DGP

Therefore no perfect classifier exists, yet very good classifiers are possible. We next generate additional covariates independently according to the mixture strategy described in the previous paragraph, with the continuous uniform supported on the interval $(0, 20)$. We look at the cases of $d = 100$ and 400. The trees generated by the three tree methods are shown in Figures 16-18. The best tree found by our method is the generative tree of the data, and therefore this classifier achieves the Bayes error rate for this example. We simulate several Markov chains are simulated to guard against trapping in local optima. This is not a new strategy for MCMC with many local optima and was proposed by CGM [**?**] for decision tree MCMC samplers to aid exploration of the decision tree space and to prevent getting stuck in one of the many local optima. We tried a few chain lengths (500, 1000, and 2000) of each chain and fitted 10 chains on each data set. The results were the same, regardless of the chain length.

threetree1.pdf

Figure 16: The tree found by a greedy optimization.

To compare the three methods, we calculate misclassification probabilities for each of the three methods. We took a random sample of 250 observations as hold out, or test data, to calculate

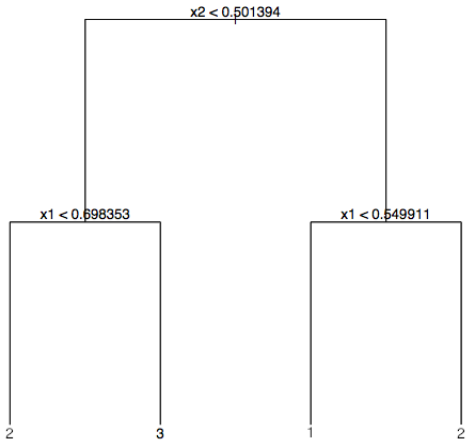


Figure 17: The best tree using the weighted method.

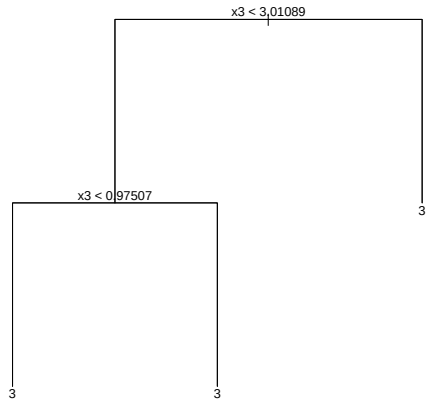


Figure 18: The best tree found by the CGM method.

Method	Misclass Prob.
Greedy	11.6%
CGM	62.4%
Weighted	10.8%

Table 2: Misclassification probabilities of the three tree fitting methods for $d = 100$.

misclassification probabilities. The misclassification probabilities are calculated by dropping all data points from the test data set through the resulting trees. We fit the trees using only the 250 observations from the training data. The tables of misclassification probabilities are displayed in Tables 2 and 3.

We see that the pruning rule fails to collect the correct covariates as important in the greedily grown tree. The problem is that the deterministic procedure is overwhelmed by candidate splits and thus overfits. In fact, the misclassification probability *within* the training data is 4.8%, better than the Bayes rate! However, this method performed poorly on hold out data compared to the weighted method, as shown in Tables 2 and 3. The greedy method eventually finds some suboptimal splits.

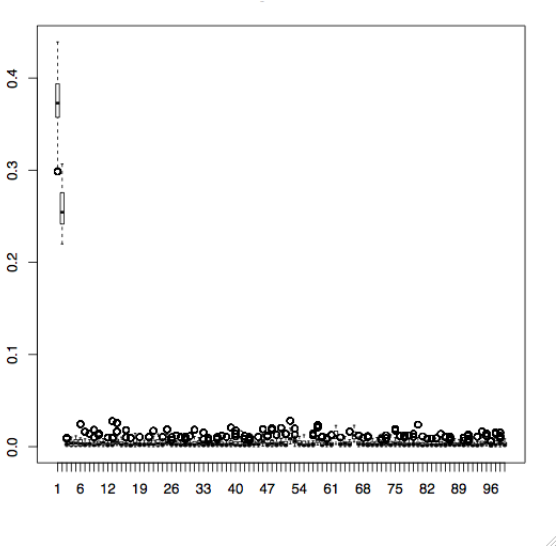


Figure 19: Covariate weights on the 100 dimensional example. Note the first two covariates are selected with high probability and the rest with miniscule probability.

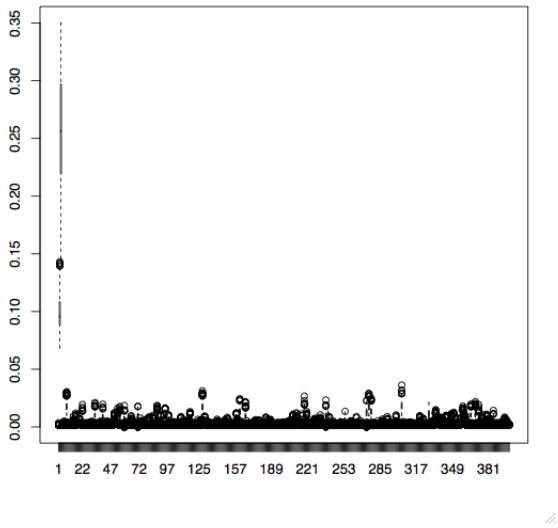


Figure 20: Covariate weights on the 400 dimension simulated example. Note the first two covariates are selected with high probability and the rest with negligible probability.

Of course, in this example we know the truth, and these extra splits are far from the truth. In addition, we see that the CGM approach fails on these trees. Much like the greedy method, the CGM approach is also overwhelmed with possible splits. Each new covariate adds exponentially more possible trees. This plethora of trees often allows one to find many locally optimal trees, but at the cost of exponentially increased search time. A benefit of our method is the improved search by effectively ignoring certain covariates and preventing some trapping of the chain in local optima. By allowing non-uniform probabilities of selecting covariates, the algorithm spends more time searching through covariates that have good splits, while not spending time in other covariates that have mostly useless splits. This behavior leads to a more efficient search of the tree space and, consequently, better predictive trees than those found by the CGM search approach. The trees built using our weighted approach are better or competitive in prediction but simpler for inference compared with those found by a greedy search as indicated by the results in Table 3.

Method	Misclass Prob.
Greedy	24.8%
CGM	57.2%
Weighted	10.0%

Table 3: Misclassification Probabilities of the three tree fitting methods for $d = 400$.

4.4.1 Choosing Covariates

A natural question to ask is: How should one choose covariates? This question is important because there are often covariates which have estimated probabilities that might be considered large enough to be important but also could be small enough to be considered not important. We approach this question from the context of the learning achieved during the sampling. We call the prior moment estimator defined here by Equation 56 as

$$\Pr(\text{covariate } j \text{ included}) = \frac{\alpha_j}{\sum_{j=1}^{d} \alpha_j}. \tag{56}$$

Nonetheless, the prior moment estimator does not take into account the learning performed by the MCMC algorithm. We now discuss a method that accounts for the learning occurring as the MCMC algorithm progresses.

Note that Equation 56 is simply the maximum likelihood estimator of the *a priori* probabilities of each covariate being selected to split on in the decision tree. To choose which covariates to use (in some subsequent model building), one might select those covariates where the *a posteriori* probabilities are larger than the *a priori* probabilities. This then indicates that information on the covariates was learned during the course of the MCMC algorithm. Formally, select the set of covariates for inclusion according to the rule

$$J = \left\{ \text{all } j : \frac{\alpha'_j}{\sum_{j=1}^{d} \alpha'_j} > \frac{\alpha_j}{\sum_{j=1}^{d} \alpha_j} \right\}, \tag{57}$$

where the $\alpha'_j = \alpha_j + \widetilde{\alpha} s_j$ denote posterior concentration parameters for each covariate.

Other methods for covariate selection are possible but we do not discuss them here. We present a small simulation study to determine the efficacy of the proposed method. We simulate data as

described in the earlier portion of this section, where we know the specified covariates that should be selected, and those that should not. We ran the simulation 100 times and present the average percentage correctly selected by using the rule in Equation 57 for our weighted method. We also compared our weighted method results to simple frequency estimates using bootstrap samples and using the CGM method for fitting decision trees. The results are presented in Table 4. We fit a collection of trees using a bootstrap resample of 100 and using 1000 trees and calculate the number of times the correct covariates are selected. These entries are the subscripted '100' and '1000' entries in Table 4.

	Weights	Boot_{100}	Boot_{1000}	Greedy	CGM
$\Pr$(covariate 1 selected)	0.32140	0.12405	0.12664	0.12178	0.13333
$\Pr$(covariate 2 selected)	0.33806	0.10795	0.10835	0.12376	0.06667
$\Pr$(covariates 1 and 2 selected)	0.15429	0.08712	0.08365	0.08614	0.06667
$\Pr$(any other covariates selected)	0.34054	0.76799	0.76502	0.75446	0.80000

Table 4: The empirical performance of the tree fitting methods on the variable selection problem using 100 simulations.

From Table 4 it is clear that the weighted method outperforms the four other methods in correctly selecting the important covariates. Also, our weighted method selects the correct covariates jointly better than the other methods. Moreover, our weighted method selects incorrect covariates less often than the other method. It is important to note that theoretically we do not want the entries in the last line to approach zero. If we did this then the MCMC would get immediately stuck in a local maximum of the likelihood space and would not find many good local maxima. Our weighted method is able to more evenly balance the search for new local maxima with the search for the correct splits and tree topology within the basin of attraction of the current local maximum. Weighting the covariates leads to improved searching of the tree space and leads to better trees in terms of inference, which are competitive in terms of prediction.

5 A Case Study of the DiVaS Model

In this section we compare our method against the greedy approach and the CGM approach using the publicly available internet ads dataset from the UCI data repository [**?**]. The internet ads data set contains 1558 covariates, some numeric and some categorical. The data set was first used in a paper by Kushmerick [**?**] and has since been used in several statistical classification studies. The UCI machine learning repository contains a more complete listing of papers using this dataset.

As noted by Kushmerick [**?**], there are several reasons for wanting to remove advertisements from webpages. Some reasons are: Images tend to dominate a pages total download time, users dislike paying for services indirectly through advertisers, preferring direct payment for services rendered, and malicious software can be unintentionally placed on a user's machine through images masked as advertisements.

We used a random subset of internet advertisements data to fit a greedy tree, a CGM tree, and a tree fit using our weighted method. We first removed all observations that contain missing values. We then took a 50% random sample of training and test data. Figures 21-23 plot the fitted trees. The misclassification probabilities are calculated by dropping all data points from the test data set through the trees built using the training data. The resulting trees are fit using only the training data.

Figure 21: The greedy tree.

Figure 22: The tree found by the CGM algorithm on the internet ads dataset.

Table 5 shows that the best tree both in terms of simplicity, and in terms of misclassification

Figure 23: Best tree using the weighted method on the internet ads dataset.

Figure 24: Estimated covariate weights using the internet ads dataset.

Method	Misclass Prob.
Greedy	5.578%
CGM	7.2%
Weighted	5.598%

Table 5: Misclassification probabilities of the three tree fitting methods on the internet ads dataset.

Cov Num	2*	1400*	1261	247	1432	1394*	121	434	476	430	864	240
Rank	607.78*	563.94*	111.91	42.00	26.78	21.99*	14.66	12.97	12.12	11.84	11.28	10.15
Cov Num	1162*	964	787	1060	69	1201	45	1015	1314	139	410	1028
Rank	8.74*	7.33	7.05	6.77	6.48	6.48	5.92	5.92	5.92	5.64	5.64	5.64
Cov Num	55	1460	1504	1214	543	123	1362	967	368	949	253	813
Rank	4.79	4.51	4.51	3.66	3.38	3.10	3.10	2.54	2.26	2.26	1.41	1.13

Table 6: The set J of possibly included covariates from Equation 57. Covariates are listed in decreasing order of the value used to include them in J. Stars indicate covariates used in Figure 23. The rank is a numeric value that follows from Equation 57.

probabilities, is the tree fir using our weighted method. The CGM approach is not as good at predicting as the trees found by either the greedy or our weighted approach. Amongst the weighted and greedy approaches, the tree found by the greedy algorithm is far more complex than the tree found by our weighted method. The weighted method is essentially as good in terms of prediction error as the greedy method and is simpler to understand and interpret. Moreover, the greedy method and the CGM method do not provide output explicitly indicating which covariates are useful in the model.

Figure 24 presents the estimated probability of selection from our model. In this case there are three covariates that are clearly important in the model. Table 6 ranks the covariates selected according to the rule from Equation 57 in decreasing order. The covariates used in the tree in Figure 23 correspond to a subset of these covariates and are starred in the table. The starred covariates are: the width of the image, the url tag "tkaine+bars", the url tag "www.irish-times.com", and the url tag "click". The tag "click" is not particularly surprising because many ads are in the form of images

that pop up a secondary webpage if you click on the image. The "Irish Times" is a newspaper and a perusal of their current webpage indicates that they have many images that accompany news articles and we can reasonable assume the website was similar when the dataset was created. The width field is also not surprising, wide windows facilitate better views of images. It is not very clear what the field "tkaine+bars" means. Perhaps some of the images come from a collection of webpages during Tim Kaine's race for mayor of Richmond, Virginia, in 1998. Admittedly, more information on this field would be desirable.

5.1 Discussion of Results

In this chapter we studied the application of MCMC data partitioning models to large dimensional data. We showed that, in large dimensional data sets, the methods used on low dimensional examples will not work effectively. We provided a simple modification that greatly improves the accuracy and utility of the partitioning scheme in large dimensional datasets.

When running MCMC algorithms on large dimensional data, we spend more time searching the covariates that are important (high probability) and ignoring the covariates that are not useful (the low probability ones). Moreover, the values of the probabilities for each covariate are on a probability scale providing intuitive interpretation of the values sampled from the posterior distribution of weights. Sparsifying our tree search procedure provides us with several benefits. Firstly, we are able to separate the covariates into groups of high and low probability. Secondly, this gives us simplified interpretation of the tree models by searching for simple trees and eases inference for the analyst. Thirdly, we explore the state space more efficiently than the nonsparse greedy approach or the CGM search approach.

Finally, we note that our approach to sparsity is different from the approach in Chipman, George and McCulloch [**?**]. In their work the goal was to model using short trees, so sparsity was achieved as a byproduct of constraining the complexity of the decision tree. Furthermore, a simple tree was their primary goal and no measure of usefulness on each covariate was desired. The inferential difficulties of the pruning rule noted in the introduction still apply to the Chipman et al. [**?**] method, whereas our dimension weighting does not have this difficulty.

6 Additive Logistic Variable Selection: The ALoVaS method

In this chapter we describe a decision tree method to explicitly handle variable selection in high-dimensional data, called the Additive Logistic Variable Selection (ALoVaS) method. This method allows us to reuse regularization approaches, prominent in the linear model literature, and apply them to decision trees. Although Bayesian methods have been available for use with decision trees since the seminal papers of Chipman, George, and McCulloch [**?**] and Denison, Mallick, and Smith [**?**], there has been little work to alleviate the practical difficulties of applying Bayesian decision trees for high-dimensional data. A notable exception is the approach of Roberts and Leman [**?**] who used a Dirichlet prior on the covariate selection probabilities. Also, the non-Bayesian approaches described by Ishwaran and coauthers [**?**, **?**, **?**] deserve mentioning these authors use an initial bootstrap sample to determine the covariate selection probabilities and then use these estimated covariate selection probabilities in a subsequent random forest algorithm. Moreover, in the discussion of Chipman et al., it is noted by Knight, Kustra, and Tibshirani [**?**] that the Bayesian approach to decision trees breaks down as the number of covariates increases.

In contrast to the literature on decision tree variable selection, the literature on variable selection for linear models is voluminous. Textbooks have been written on variable selection for linear models [**?**]. Each year several papers are published in major statistical journals developing properties of various variable selection methods and proposing modifications to improve known deficiencies.

Let us start with the linear model

$$\underline{y} = X\underline{\mu} + \underline{\epsilon}, \tag{58}$$

where $\underline{\epsilon}$ is an $n \times 1$ vector of random errors, typically assumed to follow a multivariate normal distribution. Also, X is an $n \times p$ matrix, also called the design matrix, which determines the mean value associated with each observed response, the associated element of the $n \times 1$ vector $\underline{y}$. It should be noted that in this section and throughout this paper we use the cell means model to describe the ANOVA linear model. Thus here X denotes an incidence matrix of full rank. It is well known that if n, the number of observations, is less than p, the number of covariates, then the

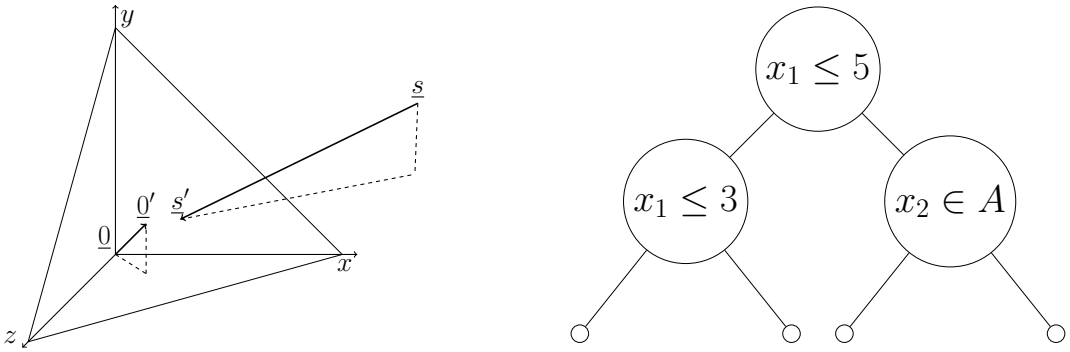
estimated values of the μ_j, for $j = 1, \ldots, p$, denoted $\hat{\mu}_j$, are not estimable. However, regularizing, or equivalently, putting a prior on the μ_j, results in estimable values for the $\hat{\mu}_j$. Moreover, a nice property of the linear model is that covariates deemed not useful can have their coefficients coded as zero, whereas useful covariates can be encoded with non-zero coefficients. Thus, a sparse linear model indicates knowledge that certain covariates are not useful. Sparse linear models are typically necessary when dealing with high-dimensional data, especially in the $p > n$ case.

The Additive Logistic Transform (ALT) , is used as the canonical inverse link function [**?**] with multi-category response data in a multi class logistic regression. The ALT is referred to as the softmax function in the machine learning literature [**?**]. Additionally, the ALT is fundamental to the analysis of compositional data [**?**]. The ALT is defined by the vector valued function given in Equation 92,

$$\underline{q} = [q_1, q_2, \ldots, q_{p-1}, q_p] = \underline{g}^{-1}(\underline{\mu}) = \left[\frac{e^{\mu_1}}{1 + S_{p-1}}, \frac{e^{\mu_2}}{1 + S_{p-1}}, \ldots, \frac{e^{\mu_{p-1}}}{1 + S_{p-1}}, \frac{1}{1 + S_{p-1}} \right], \quad (59)$$

where $S_{p-1} = \sum_{j=1}^{p-1} e^{\mu_j}$. Moreover, the ALT is an invertible mapping from the p dimensional real number system to a probability simplex. This allows us to calculate the distribution of the random variables obtained by the ALT via a change of variables. In this paper we focus on the Additive Logistic Normal (ALN) density, obtained by applying the ALT to a random vector with a multivariate normal density. The ALN density has proved successful in the correlated topic model [**?**, **?**]. Additionally, for our applications the ALN density allows us to model covariate selection probabilities that have both positive and negative correlations, whereas distributions like the Dirichlet density only permit negative correlations.

By focusing on the ALT, we are able to transform the decision tree variable selection problem into a problem of variable selection in a linear model. The upshot of this is that we may use variable selection methods, prevalent in the linear model literature, with decision trees. Thus, we can sample from a posterior distribution of the covariate selection probabilities and determine the utility of covariates, simultaneously performing variable selection and decision tree fitting.



(a) (b)

Figure 25: In (a) we plot the points in the Euclidean space and the primed points represent the points mapped to the probability simplex, represented in the plot by the surface of the triangular plane given by the equation $x + y + z = 1, x, y, z > 0$. The primes indicate points on the probability simplex after applying the additive logistic transformation to the split counts. In (b) we plot an example decision tree splitting on two of three possible covariates.

Figure 25 shows a plot of a decision tree and a plot of the probabilities associated with the tree on a two-dimensional simplex and the three-dimensional Euclidean space. If the observed data has only three covariates and we only observe the first two covariates, x_1 and x_2 in the estimated tree, then the covariate selection probability vector is approximately $\underline{p} = [0.50, 0.31, 0.19]$. The covariate that has no observed splits receives a lower assigned probability for the subsequent iteration, whereas the two covariates with larger numbers of splits receive proportionally higher covariate selection probabilities for the following iteration of the algorithm. Note the point $\underline{0}'$ in Figure 25 is the point $(1/3, 1/3, 1/3)$ which functions as the origin on the simplex in 3 dimensions.

This chapter proceeds as follows: subsection **??** describes and motivates the use of the ALT to model covariate selection probabilities. Subsection 6.2 gives the simple sampler with no regularization on the covariate selection means μ_j. Subsection **??** contains a description of the models used in this chapter. The reader may find it helpful to reread the material of Chapter 3,particularly the tree likelihood function and review the Chipman et al. Bayesian CART [**?**] model for clarity. In Subsection 6.7, we present the details of the ALoVaS model, we show the relation with the CGM model, and detail the simulation algorithms used. Then in Subsection 7, we present simulated

examples where we study the efficacy of our proposed method using the regularization priors we describe in Subsection **??**. Concluding Subsection 7, we give an example classifying webpages according to whether those webpages contain ads or not. Finally, in SubSection 7.3 we summarize our findings and discuss the limitations, extensions, and possible variations of the ALoVaS model.

6.1 Normal Distributions Transformed to the Unit Simplex: ALT and ALN Dynamics.

This chapter outlines the additive logistic transformation, which transforms a d dimensional multivariate normal distribution onto the unit simplex in $d+1$ dimensions. Note that the unit simplex in $d+1$ dimensions actually lies in a subspace of d dimensions because of the constraint that the sum of the probabilities equals one.

The goal of this chapter is to find a transform that moves the space $\mathbb{R}^d$ to the simplex $\mathbb{S}^d$. The simplex $\mathbb{S}^d$ is a space defined by the constraints $\{x_i : 0 < x_i < 1, \sum_{j=1}^{d} x_j < 1\}$, and the extra term $x_{d+1} = 1 - \sum_{j=1}^{d} x_j$ ensures the total sums to 1.

Define the notation $\underline{y}$, for the normal random variables that reside in the $\mathbb{R}^d$ dimensional space. Define the notation $\underline{x}$ for the vector that resides on $\mathbb{S}^d$, the simplex in d dimensions. We use an underline to indicate that the stated quantity is a column vector and capital greek letters (and I) will denote matrices (the identity matrix)

$$x_i = \frac{e^{y_i}}{1 + \sum_{j=1}^{d} e^{y_j}}. \tag{60}$$

Define the Jacobian of the transformation as

$$J(\underline{y}|\underline{x}) = \left(\prod_{j=1}^{d+1} x_j \right)^{-1}. \tag{61}$$

It is important to note that $\underline{y} \in \mathbb{R}^d$, whereas $\underline{x} \in \mathbb{S}^d$. The d dimensional normal has the usual parameters and density

$$f_{\underline{y}}(\underline{y}|\Sigma, \mu) = (2\pi)^{-d/2} |\Sigma|^{-1/2} \exp\left(-1/2(\underline{y} - \underline{\mu})^T \Sigma^{-1} (\underline{y} - \underline{\mu})\right). \tag{62}$$

Upon applying the transformation defined by Equation 60, we arrive at the additive logistic normal (ALN) distribution, with density

$$f_{\underline{x}}(\underline{x}|\Sigma,\mu) = \frac{(2\pi)^{-d/2}}{\sqrt{|\Sigma|}\prod_{j=1}^{d+1} x_j} \exp\left(-1/2(\log(\underline{x}_{(d+1)}/x_{d+1}) - \underline{\mu})^T\Sigma^{-1}(\log(\underline{x}_{(d+1)}/x_{d+1}) - \underline{\mu})\right). \tag{63}$$

The vector notation $\underline{x}_{(d+1)}$ denotes the vector in d dimensions that has the $d+1$ entry removed from the vector $\underline{x}$. It is important to note that this density function is defined on the space $\mathbb{S}^d$ and *not* on the space $\mathbb{R}^d$.

A useful property of this transform is that we can handle probabilities defined on the d dimensional simplex while working with a normal distribution. This is a common and comfortable probability distribution for most statisticians and applied scientists. We now wish to understand how the specification of the mean vector $\underline{\mu}$ and the covariance matrix Σ impact the structure of the ALN density. From simulation we can formulate the following conclusions:

- With $\Sigma = I$, increasing the mean vector in the positive direction in any one of the d components individually corresponds to shifting density towards the corner of the simplex associated with that covariate.

- With $\Sigma = I$, increasing the mean vector in the negative in *all* d components corresponds to shifting density towards the $d+1$ corner of the simplex.

- Keeping $\underline{\mu} = \underline{0}$, adjusting any of the variances corresponds to shifting towards a projected space of $\mathbb{S}^d$.

- With $\underline{\mu} = \underline{0}$, making one variance small corresponds to the shifting density towards the median of the simplex associated with the remaining d dimensions.

- Making the Σ matrix approximately singular and moving $\underline{\mu}$ in the negative direction for all components places most of the probability density along the median of simplex associated with first d dimensions.

- If $\Sigma = \text{Diag}(\sigma_j^2)$, as the σ_j^2 entries become smaller, the probabilities approach the CGM specification.

The ALN density can be simulated using the transformation defined in Equation 60 and the fact that it is easy to simulate from the vector normal distribution.

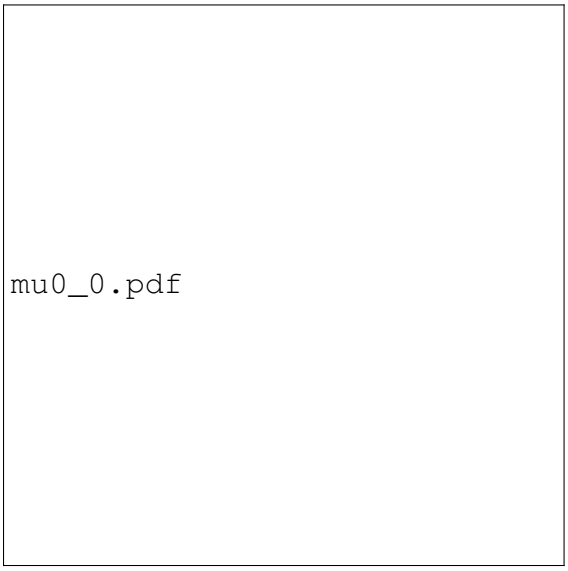


Figure 26: In this figure, $\underline{\mu}$ has all zero entries, with $\Sigma = I$, corresponding to the equiprobable case. Each probability is approximately $1/(d+1)$.

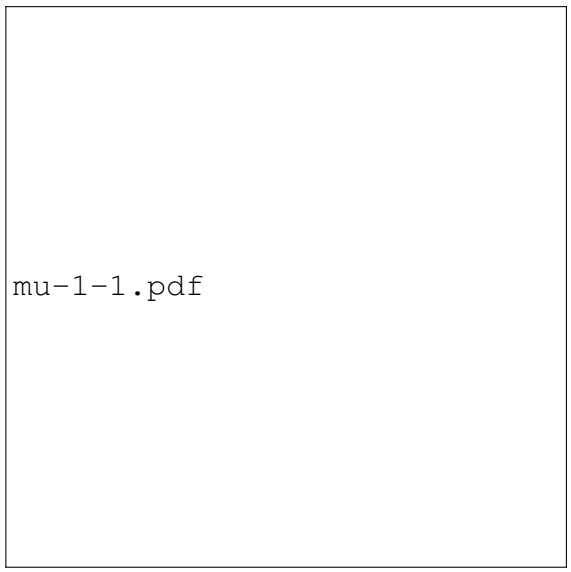


Figure 27: In this figure, $\underline{\mu} = (-1,-1)^T$, with $\Sigma = I$, corresponds to moving towards a sparser set of covariates.

The simulations indicate that we can scale everything at approximately an $\mathcal{O}(d)$ rate because of the diagonality of the variance covariance matrix, and no matrix inversions are required. The step that remains is to link this portion of the model with the (currently) observed data in the tree, so that better trees are favored over trees that are worse, when each is observed in the Markov chain. We will first work in the space $\mathbb{R}^d$ and then translate to probabilities by using Equation 60.

If we focus on the means and a collection of variances in a multivariate normal i.i.d. model, then we must fully specify the likelihood and the prior to form our posterior.

We define the likelihood of the tree by counting the tree's selected covariates and summing

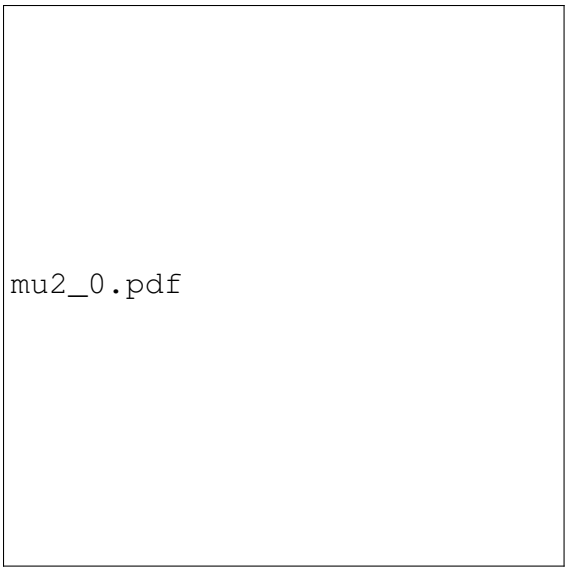


Figure 28: $\underline{\mu} = (2, 0)^T$, with $\Sigma = I$, moves the density towards one corner of the simplex.

Figure 29: $\underline{\mu} = (0, 2)^T$, with $\Sigma = I$, moves the density towards the other corner of the simplex.

across all observed splits. The counting leads to the likelihood taking on discrete values for the observed data. Let us define a multiplier that can take on an arbitrary positive or negative value in a compact region. We then multiply the counts of splits on each covariate by this quantity, effectively creating a mean which can take arbitrary values in $\mathbb{R}$.

6.2 A Simple Sampler Approach

This subsection derives the full conditional densities for each necessary update in the Gibbs loop to sample posterior weights on each dimension in the CGM decision tree sampler. Throughout this subsection we will use the notation $\odot$ to denote a Hadamard product of two matrices or vectors.

6.2.1 The General Strategy

There are many problems with using a Dirichlet prior and a Multinomial conjugate likelihood. The two most glaring problems are the implicit prior assumption of same scales on each covariate

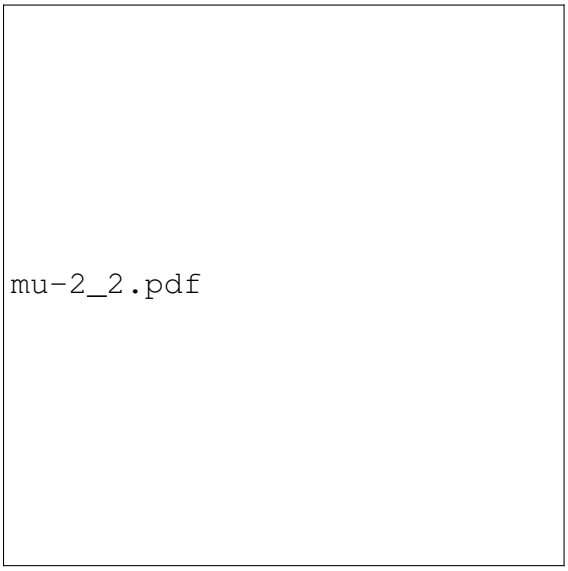


Figure 30: $\underline{\mu} = (-2, 2)^T$, with $\Sigma = I$, corresponds to most probability mass along a corner of the simplex and is a sparse representation.

Figure 31: $\underline{\mu} = \underline{0}$, with $\Sigma = \text{diag}(0.01, 100)$, corresponds to most density lying on a one dimensional subspace (the second covariate in the $\mathbb{R}$ space).

and the fact that all covariances or, equivalently, correlations must be negative. If the generative model of the data is a linear model with an interaction term and a decision tree model is fit to the data, then several splits will occur on the two interacting covariates. These splits will occur alternately until the curvature is sufficiently approximated [**?**]. This situation indicates a positive correlation between the two covariates. Higher order interactions will result in similar positive correlations between collections of covariates. Therefore we conclude the Dirichlet density as a posterior for the probability of selecting a covariate is an inferior model. Moreover, the initial motivation for modeling data with a decision tree was to handle survey data that contained many complex interactions that would be too computationally expensive to evaluate using linear model methods [**?**].

The likelihood will be denoted by

$$MVN(\underline{c} \odot \underline{s} | \underline{\mu}, \Sigma = \text{Diag}(\sigma_j^2)). \tag{64}$$

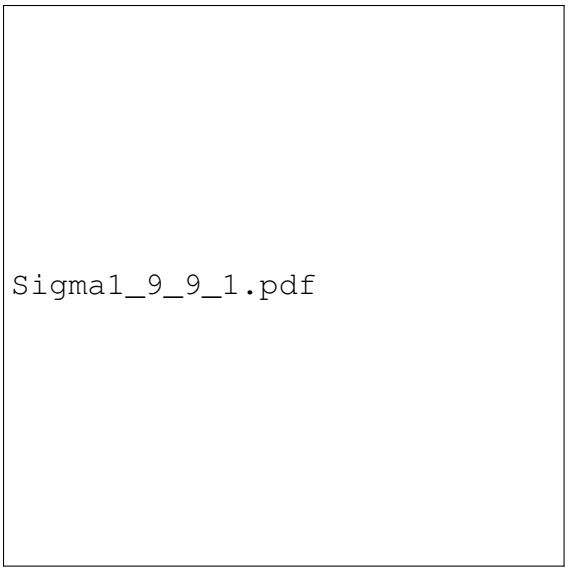


Figure 32: Here Σ is approximately singular and most of the probability mass in concentrated along the $d+1$th dimension in the $\mathbb{R}^{d+1}$ space.

Figure 33: Similar to the case in Figure 31 but with the variances reversed.

The prior is also a normal

$$\pi(\underline{\mu}|\underline{\mu}_p, \Sigma) = MVN(\underline{\mu}|\underline{\mu}_p, \Sigma). \tag{65}$$

The priors on the variances are $i.i.d.\sim$ Inv-Gamma$(\sigma_j^2|a_j, b_j)$. Finally the priors on the c_js are $i.i.d.$ scaled beta's with densities

$$S\beta(c_j|\alpha_j, \beta_j, -a, a) \equiv \pi(c_j|\alpha_j, \beta_j, -a, a) = \frac{(c_j+a)^{\alpha_j-1}(a-c_j)^{\beta_j-1}\Gamma(\alpha_j+\beta_j)}{(2a)^{\alpha_j+\beta_j-1}\Gamma(\alpha_j)\Gamma(\beta_j)}. \tag{66}$$

A special case of these scaled beta densities is when $\alpha_j = \beta_j = 1$, which yields uniform r.v.'s on the region $[-a, a]$. For each $j = 1, \ldots, d$, we first calculate $s_j = \epsilon + \sum_{\forall \eta \in \mathcal{T}} \mathbb{1}[\text{split on covariate j at node } \eta]$, for some fixed $\epsilon > 0$. Through basic Bayesian calculations we find that the full conditional distributionss are

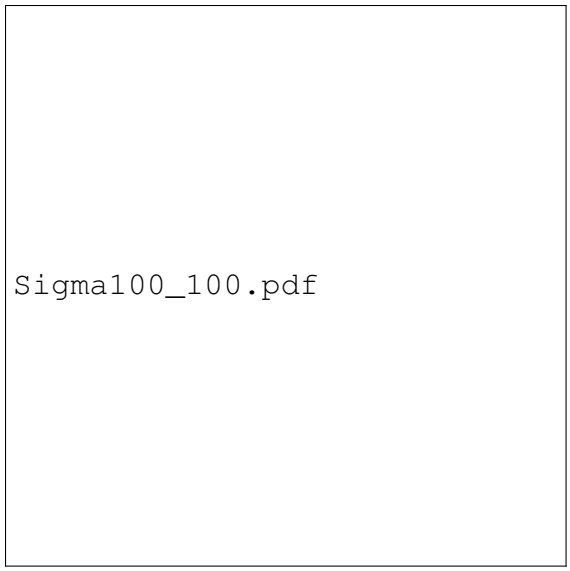


Figure 34: $\underline{\mu} = \underline{0}$, with $\Sigma = \text{diag}(100, 100)$, corresponds to encouraging sparse representations *a priori*.

Figure 35: Here $\Sigma = 0.1I$, and $\underline{\mu} = \underline{0}$, corresponds to roughly the CGM specification.

$$\pi(c_j|\mu_j, \sigma_j^2, a) \sim N_{[-a,a]}(c_j s_j|\mu_j, \sigma_j^2)(\text{a normal truncated to the region } (-a, a)), \tag{67}$$

$$\pi(\sigma_j^2|\mu_j, c_j, a) \sim \text{Inv-Gamma}(\sigma_j^2|2a_j + 2, \frac{(c_j s_j - \mu_j)^2 + (\mu_j - \mu_j^p)^2 + b_j}{2}), \text{ and} \tag{68}$$

$$\pi(\mu_j|\mu_j^p, c_j, \sigma_j^2) \sim N[c_j s_j + \mu_j^p, \sigma_j^2]. \tag{69}$$

6.3 Slice Sampling Gibbs Updates

Ideally we would like everything to be a Gibbs step. This can be accomplished numerically using numerical approximations to the cumulative density of the standard normal distribution, here denoted Φ, and Φ^{-1}, the quantile function of the standard normal cumulative density. However, we can accomplish this directly using the technique of parameter expansion set forth in Damien and Walker [**?**], which we review here for completeness.

We begin with the joint density of two random variables

$$f(x, y) \propto \mathbb{1}[0, \exp(-x^2/2)](y). \tag{70}$$

The marginal of x is a standard normal density. Through elementary probability calculations we find

$$f(y|X = x) \propto Unif(0, \exp(-x^2/2)), \tag{71}$$

and

$$f(x|Y = y) \propto Unif(-\sqrt{-2log(y)}, \sqrt{-2log(y)}). \tag{72}$$

The region upon which X is defined arises from the solution of the inequalities $\{0 \leq y \leq \exp(-x^2/2)\}$ for X, which is a quadratic equation in X. Similarly, if we have a truncated distribution truncated to the region $[a, b]$, we write the joint density

$$f(x, y) = Unif((0 \leq y \leq \exp(-x^2/2)) \times (a, b)). \tag{73}$$

Solving the resulting system of inequalities leads to the set

$$\{-\sqrt{-2log(y)}, \sqrt{-2log(y)}\} \cap \{a, b\}, \tag{74}$$

which results in the region for x

$$\{\max(a, -\sqrt{-2log(y)}), \min(b, \sqrt{-2log(y)})\}. \tag{75}$$

Simulating from a truncated normal is equivalent to simulating from two uniforms on the appropriate regions and evaluating a natural logarithm and a square root function at each iteration.

A special case of these scaled beta densities is when $\alpha_j = \beta_j = 1$, which yields uniform random variables on the region $[-a, a]$. For each $j = 1, \ldots, d$, we first calculate $s_j = \epsilon + \sum_{\forall \eta \in \mathcal{T}} \mathbb{1}[\text{split on covariate j at node } \eta]$, for some fixed $\epsilon > 0$. Through basic Bayesian calculations we find the full conditionals in closed form

$$\pi(u|c_j s_j, \mu_j, \sigma_j^2) = \text{Unif}(0, \exp\,(\frac{-(c_j s_j - \mu_j)^2}{2\sigma_j^2})), \tag{76}$$

$$\pi(c_j|u) = \text{Unif}(\max(-a, -\sqrt{-2log(u)}), \min(a, \sqrt{-2log(u)})), \tag{77}$$

$$\pi(\sigma_j^2|\mu_j, c_j, a) \sim \text{Inv-Gamma}(\sigma_j^2|2a_j + 2, \frac{(c_j s_j - \mu_j)^2 + (\mu_j - \mu_j^p)^2 + b_j}{2}), \text{ and} \tag{78}$$

$$\pi(\mu_j|\mu_j^p, c_j, \sigma_j^2) \sim N[c_j s_j + \mu_j^p, \sigma_j^2]. \tag{79}$$

The Gibbs sampling step, nested within the MH sampler, proceeds by sampling from Equations 76-79. At each iteration, once samples are drawn in sequence from the distributions given in Equations 76- 79, we take the posterior samples of μ_j and transform these onto the $[0, 1]$ scale, by using the ALN transform given in Equation 60.

We show results from a preliminary coding of the stated algorithm. We use two specifications for the prior means $\underline{\mu}^p$. One specification uses $\underline{\mu}^p = \underline{0}$ (Figure 36) and another uses $\underline{\mu}^p = (-2, -2, 2, \ldots, 2)$ (Figure 37). The difference in the results of sampled weights indicates that, if we can move from a negative to a positive value for the prior mean, we can greatly influence the selection of covariates. The graphic of posterior weights shown in Figure 37 is the correct set of weights for the data.

Figure 36: A zero mean prior. Note that the two covariates that should have large probabilities are covariates 1 and 2.

Figure 37: An informative prior. The first two prior means are 2 and the remaining prior means are set at -2.

6.4 A Stochastic Search Variable Selection Approach

Using the results of George and McCulloch we derive Gibbs updates for each of the parameters and have as a special case the approach of CGM if all the dimension indicators are selected to be the point masses at zero. In this case the ALN transform puts probability $1/(d+1)$ on each dimension.

We investigate three priors for use in variable selection with decision trees. They are:

- A stochastic search approach using method of George and McCulloch [**?**, **?**].
- A lasso prior on the means of the MVN, using parameter expansion to use Gibbs updates [**?**].
- A multivariate half-Cauchy prior using parameter expansion (Huang and Wand method) [**?**, **?**, **?**, **?**].

We will discuss the preliminary details of each method in sequence in this chapter.

Variable selection with SSVS facilitates a Bayesian approach to variable selection in decision trees. Prior methods have examined bootstrapping approaches and used some complicated math to allow the statistician to peer inside the black box method known as randomForest [**?**, **?**], often with little insight or understanding. The SSVS prior allows us to "test" whether the constant prior probability of selecting a covariate for all dimensions is appropriate for the given dataset. This means we can test, for any dataset, whether the CGM model of covariate selection is appropriate.

$$\pi(\mu_i|\tau_i, p_i, c_i) \propto p_i N(\mu_i, \tau_i^2) + (1-p_i)N(\mu_i, \psi_i^2 + \tau_i^2). \tag{80}$$

Here the $\psi_i > 0$ and the $\tau_i > 0$. Usually $p_i \equiv 1/2$ but it is also possible to put a prior on these hyper-parameters. The advantage of a mixture prior in this form is that Gibbs samples are readily available for each of the desired quantities facilitating fast sampling.

6.5 Parameter Expansions For the Lasso and Horseshoe Priors

In this section we propose to use parameter expansion to facilitate Gibbs sampling within the sampling framework for the posterior means (μ_j) for each covariate in the normal space. We use the ALN transform to convert the μ_j to probabilities using the ALT transform. We use the scale mixture of normals representation to write the Laplace prior is parameter expanded form. This representation is stated here for reference.

If $V \sim$ Exponential(1) and $Z \sim N(0,1)$ independent of V, then $X = \mu + b\sqrt{2V}Z \sim \text{Laplace}(\mu, b)$.

Equivalently we can write this as the hierarchy

$$X|V \sim N[\mu, \sigma^2 2V], \tag{81}$$

$$V \sim \text{Exponential}(1), \text{ and} \tag{82}$$

$$X \sim \text{Laplace}(\mu, \sigma). \tag{83}$$

What we want to examine here is whether shrinkage along an L_1 penalty will achieve similar results to the SSVS approach and give us meaningful results. The belief here is the same as in the SSVS approach, shrinkage will shrink means toward zero in the multivariate normal. These sampled values from the multivariate normal will then be transformed into something on the $[0,1]$ (probability) scale using the ALN transform. Values of zero, or near zero of the μ_j, transform from the Euclidean space into values of approximately $1/(d+1)$ in the simplex space. The probabilities with values around $1/(d+1)$ correspond to covariates that we are indifferent about. Recall that all probabilities must sum to 1, adding some difficulty to interpreting the estimated probabilities. Therefore if some values are excessively small this may coerce all the other probabilities to be larger. In this case it is then difficult to determine which covariates are non-informative. Further simulations regarding selecting covariates using the rule of Equation 57 will be conducted using the ALN transform and will be done on several simulated and real data examples.

The following result will be useful for parameter expansion. If

$X|a \sim$ Inv–Gamma$(\nu/2, \nu/a)$ and $a \sim$ Inv–Gamma$(1/2, A^2)$, then $\sqrt{X} \sim$ Half–Cauchy. This is expressed symbolically in the integral equation,

$$
\begin{aligned}
f(x) &= \int_0^\infty \underbrace{\frac{\nu^{\nu/2}}{a^{\nu/2}\Gamma(\nu/2)x^{(\nu/2)+1}} \exp\left(-\frac{\nu}{ax}\right)}_{=f(x|a)} \underbrace{\frac{\exp\left(-\frac{1}{aA^2}\right)}{A\sqrt{\pi}a^{3/2}}}_{=f(a)} da \\
&= \frac{\nu^{\nu/2}}{A\sqrt{\pi}\Gamma(\nu/2)x^{(\nu+2)/2}} \underbrace{\int_0^\infty a^{-\nu/2-1/2-1} \exp\left(-(1/a)(\nu/x + 1/A^2)\right) da}_{=\text{Inv-Gamma kernel}} \\
&= \frac{\nu^{\nu/2}\Gamma((\nu+1)/2)}{A\sqrt{\pi}\Gamma(\nu/2)x^{(\nu+2)/2}(\nu/x + 1/A^2)^{(\nu+1)/2}} \\
&\propto \frac{1}{\sqrt{x}x^{(\nu+1)/2}(\nu/x + 1/A^2)^{(\nu+1)/2}} \\
&= \frac{1}{\sqrt{x}(\nu + x/A^2)^{(\nu+1)/2}} \\
&= \frac{1}{\sqrt{x}(\nu + x/A^2)^{(\nu+1)/2}}.
\end{aligned}
$$

Making the change of variable $x = y^2$, which implies $dx/(2\sqrt{x}) = dy$, and substituting y and dy shows us that

$$
f(y) \propto (1 + (y/A)^2/\nu)^{-(\nu+1)/2}, \tag{84}
$$

for $y > 0$ which is the definition of a half-Cauchy density.

Huang and Wand [**?**] indicate that using an inverse-Wishart prior on the variances, with each standard deviation having a half-Cauchy prior, gives a scaled beta marginal for each correlation. At this point it is worth recalling that the t and half-t distributions have as special cases, the Cauchy and half-Cauchy distributions when the degrees of freedom parameter (ν) in the t and half-t is set to $\nu = 1$.

6.6 Regularization Posteriors

While this paper is focused on regularized trees, we will show the connection to various regularization frameworks used for regression models. In essence, the ALoVaS method can incorporate

any model-based regularization method. We briefly describe a few popular approaches for the sake of completeness.

Regularization, originally defined as a constrained optimization problem, has been a popular topic of research in the sparse linear model literature. The lasso model [**?**], a popular regularization technique, seeks to optimize an L_1 constrained least squares problem. The negative of the log posterior density of the lasso is given by Equation 86

$$-log(\pi(\underline{\mu}|\underline{y}, X, \lambda)) \propto \sum_{i=1}^{n} \left(y_i - \underline{x}_i^T \underline{\mu}\right)^2 + \underbrace{\lambda \sum_{j=1}^{p} |\mu_j|}_{-log \text{ Laplace prior}} \tag{85}$$

Interpreted in a Bayesian setting, the L_1 constraint is viewed as a double exponential or Laplace prior on the coefficients, μ_j. A nice property of the lasso model is that the *maximum a posteriori* estimated coefficients form a sparse vector, with λ controlling the degree of sparsity. Thus, one of the models that we will use in our simulations uses a lasso prior to constrain the covariate selection probabilities.

Since the publication of the lasso model, several other regularization techniques have been published. One that predates the lasso by several years is the Stochastic Search Variable Selection method (SSVS) [**?**], given in Equation 88 beneath

$$-log(\pi(\underline{\mu}|\underline{y}, X, \pi, \sigma, \psi)) \propto \sum_{i=1}^{n} \left(y_i - \underline{x}_i^T \underline{\mu}\right)^2 + \sum_{j=1}^{p} log \left\{ \underbrace{\pi_j N(0, \sigma^2) + (1 - \pi_j) N(0, \sigma^2 + \psi^2)}_{\text{Spike \& slab prior}} \right\}, \tag{86}$$

with $\sigma < \psi$,s and $N(A, B)$ denotes the likelihood of a normal random variate with mean A and variance B. This Bayesian method posits a prior distribution that is a point mass mixture prior. The point mass is at zero and the density is a normal distribution, with a Bernoulli random variable mixing the two. We present simulation studies that use the SSVS model. In addition to these two methods, SSVS and lasso, we also study another prior known as the horseshoe prior, originally proposed by Carvalho, Polson, and Scott [**?**], and further studied in Polson and Scott [**?**]. This prior may also be viewed as a constraint on the unknown coefficients, although the objective function is usually viewed in a Bayesian context as a posterior distribution over the coefficients. The horseshoe

model has the following negative log posterior,

$$-log(\pi(\underline{\mu}|\underline{y}, X, \sigma, \lambda)) \propto \sum_{i=1}^{n} \left(y_i - \underline{x}_i^T \underline{\mu}\right)^2 + \underbrace{\sum_{j=1}^{p} \left(\frac{\mu_j}{\lambda_j \sigma}\right)^2 + \frac{p+1}{2} log\left(1 + \frac{\lambda_j^2}{p}\right)}_{-log \text{ Horseshoe prior}}. \tag{87}$$

Although Carvalho et al. [**?**] state that the horseshoe prior is the case when $\sigma^2 = 1$, in their paper they refer to the general case of σ^2 as the horseshoe prior. We also call the general case the horseshoe prior. Additionally, λ^2 has a central half-t density with p degrees of freedom. For completeness, we write out the priors, including the hierarchical form of the horseshoe prior, which is the commonly used way to represent the horseshoe prior:

$$\pi(\mu_j) \propto N(0, \sigma_j^2)p_j + (1 - p_j)N(0, \sigma_j^2 + \psi), \tag{88}$$

$$\pi(\mu_j) \propto \exp(-\lambda|\mu_j|), \tag{89}$$

$$\begin{aligned} \pi(\mu_j|\lambda_j) &\propto N(0, \lambda^2\sigma^2), \\ \pi(\lambda_j|a) &\propto \text{Inv-}\Gamma(a, b), \\ \pi(a) &\propto \text{Inv-}\Gamma(c, d). \end{aligned} \tag{90}$$

The reader should note that the Laplace prior in the model indicates that this is a lasso type model and the lasso prior is in fact a double exponential, or Laplace prior. Other priors on the μ_j are possible and we consider the SSVS, or spike and slab prior given in Equation 88, and the horseshoe prior given in Equation 90, in addition to the lasso prior of Equation 89.

Recall, if X is distributed according to a t-distribution with p degrees of freedom, then $Y = |X|$ is distributed according to a half-t distribution with p degrees of freedom [**?**]. For our simulations, we use the mixture representation of the half-t distribution given in Huang and Wand [**?**] and given in Subsection 6.5. Interestingly, the three models can be interpreted as scale mixtures of normal priors. The Lasso is equivalent to a exponential scale mixture of normals, the SSVS is a point mass scale mixture of normals, and the horseshoe is a half-t scale mixture of normals. In all three cases we have a global regularization parameter, λ, ψ, and σ represent the regularization parameters for the lasso, SSVS and horseshoe priors respectively.

6.7 The Model

In this section we detail the ALoVaS model in the most general form. Then we detail the model for an arbitrary choice of priors for the means and variances. This allows the reader to see concrete examples of the ALoVaS method for clarity, while preserving the general applicability of the method.

We begin the model description by stating our posterior,

$$\pi(\mathcal{T}, \theta, \underline{q}|Y, X) \propto \mathcal{L}(Y|X, \mathcal{T}, \theta, \underline{q})\pi(\theta|\mathcal{T})\pi(\underline{q}|\mathcal{T})\pi(\mathcal{T}), \tag{91}$$

where Y denotes the response vector, X is the design matrix or the collection of covariates, $\mathcal{T}$ denotes the decision tree, and $\underline{q}$ denotes the vector of covariate selection probabilities.

Instead of defining $\pi(\underline{q}|\mathcal{T})$ in terms of $\underline{q}$, we define it on a linear scale, and once we have the linear scale parameters, we convert to a probability using the ALT

$$\underline{q} = [q_1, q_2, \ldots, q_{p-1}, q_p] = \underline{g}^{-1}(\underline{\mu}) = \left[\frac{e^{\mu_1}}{1+S_{p-1}}, \frac{e^{\mu_2}}{1+S_{p-1}}, \ldots, \frac{e^{\mu_{p-1}}}{1+S_{p-1}}, \frac{1}{1+S_{p-1}}\right], \tag{92}$$

where the notation is the same as in Section 6. The parameters on the linear scale are denoted μ_j and can be interpreted as log-odds of selecting covariate j. The number of splits on each covariate at each iteration are denoted as s_{jt}. We assume these counts, s_{jt} to have an approximately normal distribution once multiplied by a parameter c_j. The parameter c_j is distributed on the support $[-1, 1]$ and has a scaled-beta prior scaled and shifted to cover the support of c_j. This framework is written succinctly in Equations 93 and 94. The equations for s_{jt}, c_j, and μ_j are written out here for completeness

$$c_j s_{jt}|c_j \sim N\left(\mu_j = log(q_j/q_p), \sigma_j^2\right), \tag{93}$$

$$c_j \sim \text{ ScaledBeta}(-1, 1, \alpha, \beta), \text{ and} \tag{94}$$

$$\mu_j \sim g(\mu_j). \tag{95}$$

Here the density function g can be specified by the analyst. We detail the specification of g as a Laplace (Equation 89) prior, SSVS (Equation 88) prior, and horseshoe (Equation 90) prior. The form of g should now be recognized as a regularization prior. In previous Subsections 6.4-6.5, we included details for each of the three stated priors. Namely, the pseudocode for the lasso, SSVS, and horseshoe priors may be found in Sections 6.8.1, 6.8.3, and 6.8.2, respectively. Finally, we assume the usual Jeffreys' prior for the variances,

$$\pi(\sigma_j^2) \propto \quad \sigma_j^{-2}. \tag{96}$$

In practice inference should not be conducted on all the parameters in the model. There are many additional parameters in this model compared to the CGM version. For us, the main parameters of interest are $\mathcal{T}_i$, the ith decision tree, and $\underline{q}$, the covariate selection probabilities. The probabilities are a function of the parameters $\underline{\mu}, \underline{c}, \underline{\sigma}$ and any other hyper-parameters estimated or sampled over. For full implementation details of the ALoVaS methods we include pseudocode in Subsection 6.8 for the SSVS, lasso, and horseshoe methods. However, as noted in this section, different choices of $g(\mu_j)$ priors that regularize will result in different types of ALoVaS methods so the careful reader should be able to tailor the existing methods to whatever desired regularization prior.

The proposed trees are accepted or rejected based on the usual Metropolis-Hastings rule, which is written here for completeness

$$\alpha(\mathcal{T}|\mathcal{T}') = \min\left(\frac{\Pr(\mathcal{T}'|X, \underline{q}, \underline{y})\Pr(\mathcal{T}|\mathcal{T}')}{\Pr(\mathcal{T}|X, \underline{q}, \underline{y})\Pr(\mathcal{T}'|\mathcal{T})}, 1\right) \tag{97}$$

Detailed balance is a formal property imposed on Markov Chains after convergence. Intuitively, the detailed balance equation indicates that, after convergence, the process has the same distribution when run forwards in time as when run backwards in time. Formally, this is represented in Equation 98,

$$\alpha(\mathcal{T}^{(t)}|\mathcal{T}^{(t-1)}, \underline{q}^{(t-1)})\pi(\mathcal{T}^{(t)}|\underline{q}^{(t-1)})\pi(\underline{q}^{(t-1)}) = \pi(\underline{q}^{(t)}|\mathcal{T}^{(t-1)})\pi(\mathcal{T}^{(t-1)})\alpha(\mathcal{T}^{(t-1)}|\mathcal{T}^{(t-1)}, \underline{q}^{(t)}), \tag{98}$$

where the superscripts denote the (discrete time) nature of the Markov Chain. We are holding the $\underline{q}$ constant while transitioning from $\mathcal{T}^{(t)}$ to $\mathcal{T}^{(t-1)}$. Once the transition on $\mathcal{T}$ has been made, we update $\underline{q}$ using a Gibbs step. Thus, our algorithm is a Metropolis within Gibbs algorithm. The Metropolis step updates $\mathcal{T}$, the Gibbs step updates $\underline{q}$, and we iterate until convergence. It can be shown that any algorithm satisfying the detailed balance criteria is guaranteed to converge to the stationary distribution [**?**].

The Markov Transition Density (MTD) for q_j is given by Equation 99

$$\Pr(q_j^{(t)}|q_j^{(t-1)}) = \sum_{\mathcal{T}} \Pr(q_j^{(t)}|\mathcal{T}^{(t-1)}) \Pr(\mathcal{T}^{(t-1)}|q_j^{(t-1)}). \tag{99}$$

A similar MTD can be formulated for $\mathcal{T}$. When closed forms exist for these MTDs, and minorization and drift conditions are also satisfied, geometric convergence may be established. For examples and more details on this line of study, see the very readable introduction by Jones and Hobert [**?**]. For our example, the MTD for $\underline{q}$ involves evaluating an NP-Hard sum, thus exact convergence rates seem unlikely to be found.

In this section we write out the ALoVaS model for a specific regularization prior. In general, the prior on the set $\{\mu_j, \sigma_j^2\}_{j=1}^p$ will change, giving us different flavors of the ALoVaS method. In this paper we include a study of the lasso, horseshoe, and SSVS priors, but other variations are certainly possible. Moreover, we do not claim optimality of our results. Indeed, many of these priors are optimal only in specific circumstances. If a model is not optimal in a linear model, this will carry over to our ALoVaS model. Therefore, the ALoVaS method will be sub-optimal as well. This is not a drawback of our proposed ALoVaS method but is a drawback of the choice of the regularization prior in the given situation.

6.8 ALoVaS Algorithms Pseudocode

This subsection contains some technical details of regularization priors described in Section 3. Also in this appendix we explain the hierarchical expansions that we employ to facilitate Gibbs sampling.

6.8.1 The Lasso Prior

The lasso prior, also referred to as L_1 regularization, was described in [**?**] and has the posterior density

$$-log(\pi(\underline{\mu}|-)) \propto \sum_{j=1}^{p}(c_j s_{jt} - \mu_j)^2 + \lambda \sum_{j=1}^{p} |\mu_j|. \tag{100}$$

Sampling from this density directly is difficult, instead we work in a larger parameter space and recover this density marginally. The scale mixture of normal representation of the Laplace distribution was used in [**?**] and is a convenient way to represent the L_1 prior in an expanded form. We expand the L_1 prior to be a scale mixture of normals where the scale density is an exponential density. We then have the posterior density proportional to

$$-log(\pi(\underline{\mu}, \tau^2|-)) \propto \sum_{j=1}^{p}(c_j s_{jt} - \mu_j)^2 + \lambda^2 \sum_{j=1}^{p} \left(\frac{\mu_j}{\tau}\right)^2 + \frac{\lambda^2 \tau^2}{2}.$$

To sample from this joint posterior density we need to sample an additional parameter τ which has a generalized inverse Gaussian (GIG) density [**?**, **?**]. Note that integrating the prior over τ^2 allows us to recover the marginal L_1 prior on the μ_j. A GIG density can be easily and efficiently sampled using an accept-reject algorithm given by Devroye [**?**]. Sampling from μ_j is now accomplished by Gibbs sampling conditional on the previously sampled value of τ.

Algorithm 1: Laplace Sampler

Data: $\mathcal{T}_1, \underline{c}_1, \underline{\sigma}_1^2, \underline{\tau}_1^2, \underline{\mu}_1, \underline{q}_1$

for $i = 2, \ldots, \#$*samples* **do**

$\mathcal{T}_i \sim$ Tree Sampler$(\mathcal{T}_{i-1}, \underline{q}_{i-1})$;

$\underline{c}_i \sim N_{[-a,a]}(m, V)$;

$\underline{\sigma}^2 \sim$ Inv-Gamma$\left(\frac{2a+6}{2}, \frac{(c_{ij}s_{ij}-\mu_j)^2}{2} + \lambda^2 \frac{(\mu_j - \mu_j^p)^2}{\tau_j^2} + \theta\right)$;

$\underline{\tau}^2 \sim$ GIG$(\chi = \lambda^2 \mu_j, \psi = \lambda^2, \lambda' = 0)$;

$\underline{\mu} \sim$ Normal $\left(\frac{c_{ij}s_{ij}}{\sigma_i^2 \tau_{ij}^2}, \frac{\sigma_i^2 \tau_{ij}^2}{\lambda^2 + \tau_{ij}^2}\right)$;

$\underline{q}_i = \left[\frac{e^{\mu_1}}{1+\sum_{j=1}^{p-1} e^{\mu_j}}, \ldots, 1 - \sum_{j=1}^{p-1} q_j\right]$;

6.8.2 The Horseshoe Prior

The hierarchical representation of the horseshoe prior [**?**] is given by the following equations

$$y_j|\mu_j, \sigma^2 \sim N(\mu_j, \sigma^2), \tag{101}$$

$$\mu_j|\lambda_j, \tau \sim N(0, \lambda_j^2\tau^2), \tag{102}$$

$$\pi(\lambda_j) \propto \frac{1}{1+\lambda_j^2}\mathbb{1}[\lambda_j \geq 0], \tag{103}$$

$$\pi(\tau) \propto \frac{1}{1+\tau^2}\mathbb{1}[\tau \geq 0]. \tag{104}$$

This is a global-local scale mixture of normals [**?**]. Additionally, we may describe the Inv-Γ scale mixture representation of half-t densities [**?**]

$$f(\sigma^2|\nu, a) = \frac{(\nu/a)^{\nu/2}}{\Gamma(\nu/2)}(\sigma^2)^{-(\nu/2+1)}\exp\left[-\frac{\nu}{a\sigma^2}\right] \tag{105}$$

$$f(a|1/2, c) = \frac{c}{\Gamma(1/2)}a^{-3/2}\exp\left[-1/(ac^2)\right] \tag{106}$$

$$\text{then,} \tag{107}$$

$$f(\sigma|\nu, c) = \text{Half-t}(\nu, c). \tag{108}$$

It is worth noting that this scale mixture result given above in Equation 105 has the measure specified on σ^2, but the half-Cauchy is defined on σ.

Algorithm 2: Horseshoe prior Sampler

Data: $\mathcal{T}_1, \underline{c}_1, \underline{\sigma}_1^2, \underline{\tau}_1^2, \underline{\mu}_1, \underline{q}_1, \underline{\lambda}^2, a_\tau, c_\tau, a$

for $i = 2, \ldots, \#\textit{samples}$ **do**

$\mathcal{T}_i \sim$ Tree Sampler$(\mathcal{T}_{i-1}, \underline{q}_{i-1})$;

$\underline{c}_i \sim N_{[-a,a]}(\mu_j/s_{ij}, \sigma_j^2/s_{ij})$;

$\lambda_j^2 \sim$ Inv-$\Gamma(\nu_j + 1/2, \mu_j^2/2\tau^2 + 1/a_j)$;

$a_j \sim$ Inv-$\Gamma(1/2, 1/\lambda_j^2 + 1/c_j)$;

$\tau^2 \sim$ Inv-$\Gamma(\nu_\tau + p/2, \sum_{j=1}^p \left(\frac{\mu_j}{\lambda_j}\right)^2 + 1/a_\tau)$;

$a_\tau \sim$ Inv-$\Gamma(1/2, 1/\tau^2 + 1/c_\tau)$;

$\mu_j \sim$ Normal $\left(\frac{c_{ij}s_{ij}\mu_j V}{\sigma_i^2}, V = \frac{\sigma_{ij}^2\tau_i^2\lambda_j^2}{\lambda^2\tau_{ij}^2+\sigma_j^2}\right)$;

$\underline{q}_i = \left[\frac{e^{\mu_1}}{1+\sum_{j=1}^{p-1}e^{\mu_j}}, \ldots, 1 - \sum_{j=1}^{p-1} q_j\right]$;

6.8.3 The Stochastic Search Prior

The point mass mixture prior is defined by Equation 88 [**?**],

$$\pi(\mu_j) = \pi_0 N(0, \sigma_0^2) + (1 - \pi_0) N(0, \sigma^2). \tag{109}$$

Note the normal density limit $\lim_{\sigma_0 \to 0} N(0, \sigma_0^2) \xrightarrow{d} \delta(\beta_j)$ where $\delta(f)$ is the Dirac delta functional with point masses at the roots of the function f.

Algorithm 3: SSVS sampler.

Data: $\mathcal{T}_1, \underline{c}_1, \underline{\sigma}_1^2, \underline{\tau}_1^2, \underline{\mu}_1, \underline{q}_1$

for $i = 2, \ldots, \#$*samples* **do**

$\mathcal{T}_i \sim$ Tree Sampler$(\mathcal{T}_{i-1}, \underline{q}_{i-1})$;

$\sigma_j^2 \sim$ Inv-Γ $(\alpha_\sigma + 1/2, (c_j s_{ij} - \mu_j)^2 + 1/\beta_\sigma)$;

$y_j \sim$ Bernoulli $\left(\frac{1}{1 + \frac{N(\mu_j; 0, \sigma_{\neq 0}^2)(1 - \pi_{j0})}{N(\mu_j; 0, \sigma_0^2)\pi_{j0}}} \right)$;

if $y_j == 1$ **then**

$\mu_j \sim N(\frac{c_j s_{ij} \sigma_{\neq 0}^2}{\sigma_{\neq 0}^2 + \sigma_j^2}, \frac{\sigma_{\neq 0}^2 \sigma_j^2}{\sigma_{\neq 0}^2 + \sigma_j^2})$;

else $y_j == 0$ case

$\mu_j \sim N(\frac{c_j s_{ij} \sigma_0^2}{\sigma_0^2 + \sigma_j^2}, \frac{\sigma_0^2 \sigma_j^2}{\sigma_0^2 + \sigma_j^2})$;

$\pi_{j0} \sim$ Beta$(y_j + \alpha_\pi, 1 - y_j + \beta_\pi)$;

$\underline{q}_i = \left[\frac{e^{\mu_1}}{1 + \sum_{j=1}^{p-1} e^{\mu_j}}, \ldots, 1 - \sum_{j=1}^{p-1} q_j \right]$;

7 ALoVaS Examples and a Case Study

In this section we present simulation studies to illustrate properties of our methods. The two illustrated properties are handling multi-modality and handling highly non-balanced categorical response variable class probabilities. Following these simulation studies we present a case study using the internet ads dataset from the UC-Irvine machine learning database [**?**].

7.1 Simulated Examples

In this section we present two collections of simulations. The first collection aims to study the effects of having multiple best trees in the generative model. The second collection aims to study the effects of class imbalance when dealing with a categorical response.

7.1.1 Multimodal Simulation Study

In this collection of simulations we study is a class of simulations where more than one model explains the data equally well. We simulate data according to the model described in Wu, Tjelmland, and West [**?**], which exhibits multi-modality, also called the Rashomon effect by Breiman [**?**]. To simulate effectively from the data generating process, we need to be able to sample the two models in the same chain.

For this simulated example we take $x_{i1} \sim$ Unif$(.1, .4)$, for $i = 1, \ldots, 200$, and $x_{i1} \sim$ Unif$(.6, .9)$, for $i = 201, \ldots, 300$. Similarly, $x_{i2} \sim$ Unif$(.1, .4)$, for $i = 1, \ldots, 100$, $x_{i2} \sim$ Unif$(.6, .9)$, for $i = 101, \ldots, 200$, and $x_{i2} \sim$ Unif$(.1, .9)$, for $i = 201, \ldots, 300$. Finally, $x_{i3} \sim$ Unif$(.6, .9)$ for $i = 1, \ldots 200$, and $x_{i3} \sim$ Unif$(.1, .4)$, for $i = 201, \ldots, 300$. Then, given these values,

$$y_i = \begin{cases} 1 + N(0, 0.25) & \text{if } x_{1i} \leq 0.5 \text{ and } x_{2i} \leq 0.5, \\ 3 + N(0, 0.25) & \text{if } x_{1i} \leq 0.5 \text{ and } x_{2i} > 0.5, \\ 5 + N(0, 0.25) & \text{if } x_{1i} > 0.5. \end{cases}$$

We run our algorithm using the regularized methods on the number of splits and then transform all the sampled μ_j using the ALT. In this simulation, varying the sparsity is not something easily

parametrized. Whereas in the next simulation we can vary the number of active covariates, in this simulation we must fix the number of active covariates to 3 and vary the total number of covariates to accomplish a varying of the percentage of sparsity. Similarly to the next simulation, we compare the linear model methods: lasso, horseshoe, and SSVS. These various methods correspond to the rows within a sparsity level in Table 7.

		MSE
sparsity 50%	CGM	34.74%
	SSVS	12.43 %
	LASSO	5.89%
	HORSESHOE	7.35 %
sparsity 60%*	CGM	34.44 %
	SSVS	9.64%
	LASSO	6.19%
	HORSESHOE	9.22%
sparsity70%	CGM	27.13 %
	SSVS	8.41%
	LASSO	6.64%
	HORSESHOE	16.05%
sparsity 80%	CGM	27.64 %
	SSVS	8.69 %
	LASSO	6.12 %
	HORSESHOE	7.52 %
sparsity 90%	CGM	68.06 %
	SSVS	11.94 %
	LASSO	8.41%
	HORSESHOE	9.63%
sparsity 95%	CGM	47.37 %
	SSVS	7.08%
	LASSO	11.15%
	HORSESHOE	8.98%
sparsity 99%	CGM	47.51 %
	SSVS	6.82%
	LASSO	7.33%
	HORSESHOE	5.73%

Table 7: Multimodal simulation study results, each entry represents the proportion of time the

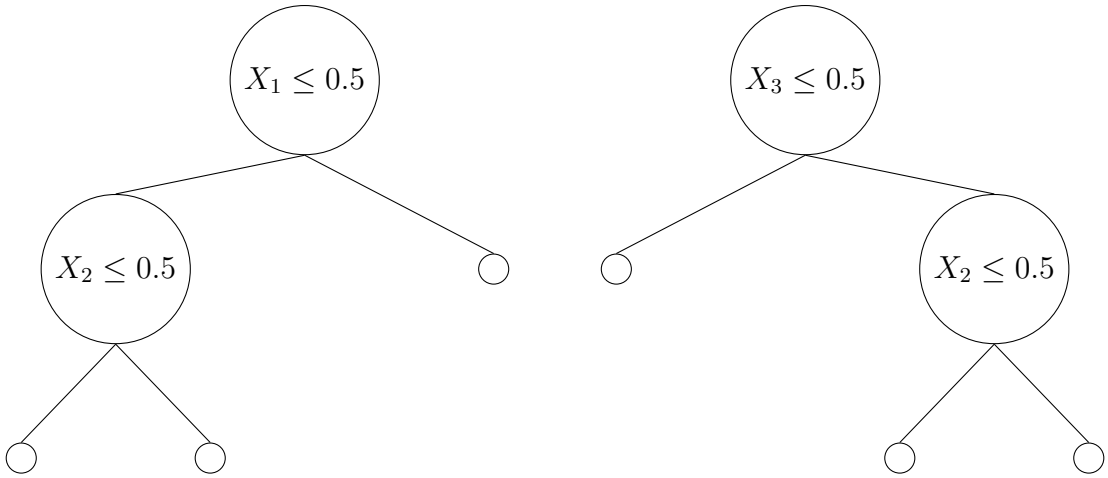


Figure 38: The two optimal trees for the multimodal simulation study.

For the results presented in Table 7 our burn-in is $1,000$. Additionally, we calculate MSEs using the 10,000 observations after burn in. Convergence diagnostics indicated a minimum sample size of 3746, so 10,000 samples is beyond adequate.

Moreover, in Table 7 a sentiment appears. There is no clear winner across sparsity levels. While the lasso and the horseshoe appear to be competitive at higher sparsity levels, the SSVS method might also be competitive at lower and higher sparsity levels but there is some degree of variability across the sparsity levels. Additionally, while all three models are able to attain one of the two modes, all three models have difficulty moving from one of the two modes in the underlying model to the other mode. This is exactly a result of the sparsity imposed on the fitted model. The two posterior trees in the two modes have different sets of covariates and thus the sparsity patterns will have difficulty switching from one to the other. A similar pattern can be found in sparse linear models. Typically, correlation constraints are imposed on the $(X^T X)$ matrix to ensure consistency of the selected sparsity pattern.

7.1.2 Class Unbalancedness Simulation Study

We next perform a simulation on unbalanced response class probabilities. We examine various states of unbalancedness of the categorical response variable. The intent of this simulation study is

to determine the effect of response class unbalancedness, a phenomenon widely seen in statistical practice. We simulate data according to the 3 category logistic regression model with a varying number of covariates. In this simulation we fix the number of active covariates to 2 so that varying the sparsity requires the number of covariates to range from 4 total (sparsity =50%) to 200 total (99% sparsity). For convenience we fix the coefficient vector at $\underline{\beta}^T = [1, 1, 0, \ldots, 0]$, varying the dimensionality of the vector as necessary for the desired sparsity. We include Figure 39 to give the reader a visual idea of the points on the three dimensional simplex where we study the class unbalanced problem. Note that by the self-similarity of the simplex space we don't need to study the areas of the simplex that are empty. Switching the labeling of the axes accomplishes this for us. In this case the settings where certain classes are extremely rare require a sample of $n = 10000$ per simulation. This setting is used for the highly unbalanced cases, as well as all the other cases in Figure 40.

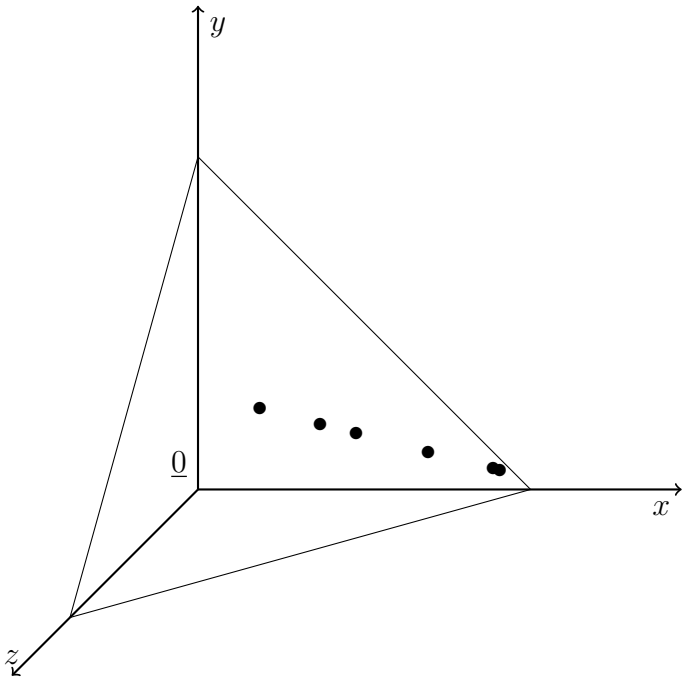


Figure 39: The points on the three dimensional simplex space we use as unbalanced classes in the simulation study.

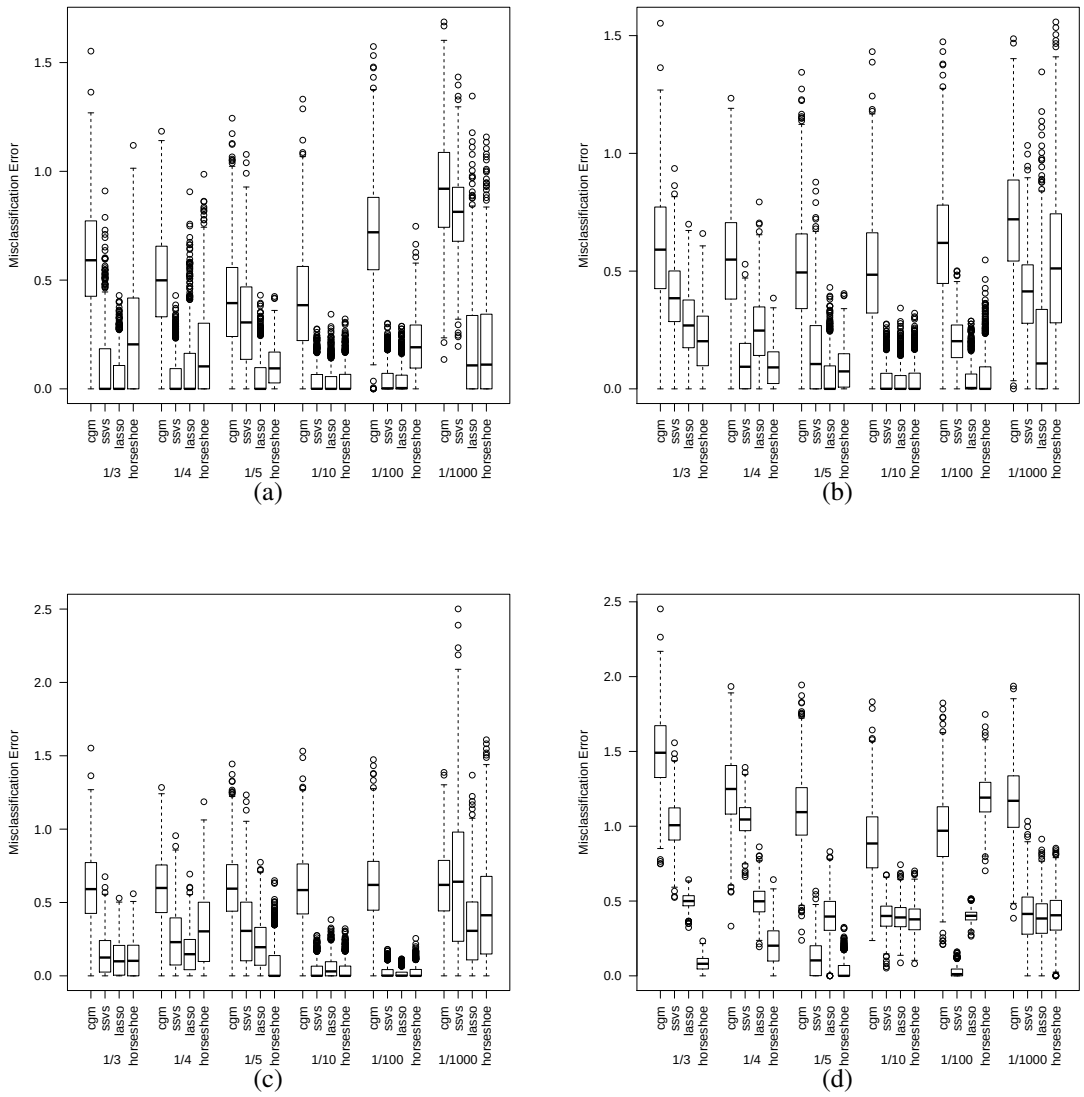
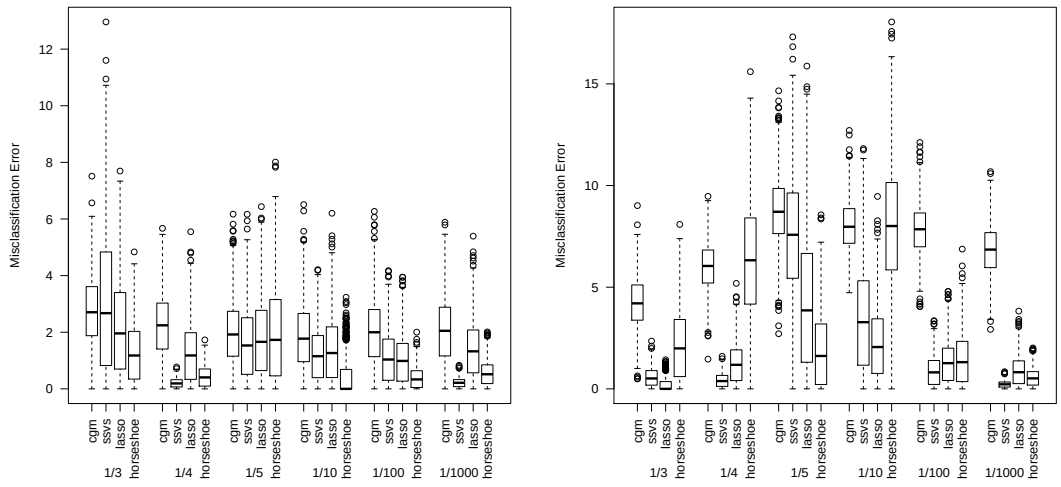
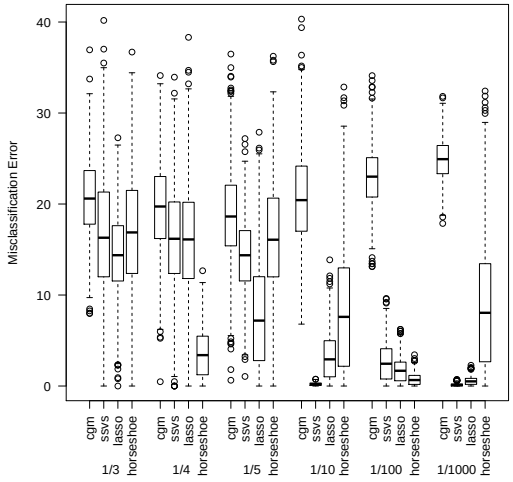


Figure 40: Box plots of the errors in prediction with the hold out data from the sampled trees after burn-in in the unbalanced simulation study. The three box plots are the prediction errors for the sampled trees for SSVS, lasso, and horseshoe within each set of class probabilities. The fractions displayed with each set of four box plots label the correspondence with the proportion used to simulate each of the three response classes in the simulated data. Also these proportions are the points on the simplex displayed in Figure 39. Each letter in the figure, corresponding to the different box plots is a different sparsity value, corresponding to different sparsity levels in the number of covariates. Figure (a) corresponds to 50% sparsity, Figure (b) 60%, Figure (c) 70%, and Figure (d) 80%.



(a) (b)



(c)

Figure 41: Box plots of the errors in prediction with the hold out data from the sampled trees after burn-in in the unbalanced simulation study. This is for the case of 90% (a) sparsity, 95% (b) sparsity, and (c) 99% sparsity.

Intuitively, the results of Figures 40 and 41 make sense. The less sparse the true model, the easier the model can classify the categories. Sparse models are not terribly difficult to classify. However, cases of extreme sparsity, such as those where 99+% sparsity exists, are likely to have a

far greater error rate. Interestingly, the degree of imbalance in the true, underlying model appear to affect the error as well. The explanation here is that, a decision tree can more easily partition a subset of the covariate space to classify responses that are rare as compared to more prevalent responses.

The results here are somewhat inconclusive on which specific model, ssvs, lasso, or horseshoe is best in a given scenario. It appears from the unbalanced simulation studies that, aside from Monte Carlo variability, there is no clear winner amongst the three ALoVaS models. It does appear that given a certain sparsity and perhaps other characteristics of the underlying model, such as correlation or class unbalanceness, it may be better to choose one type of ALoVaS model over the others. However, this information seems unlikely to be known *a priori* except in special circumstances. Moreover, any of the ALoVaS methods is likely to prove useful and provide improvements over the traditional Bayesian decision tree algorithm proposed by CGM.

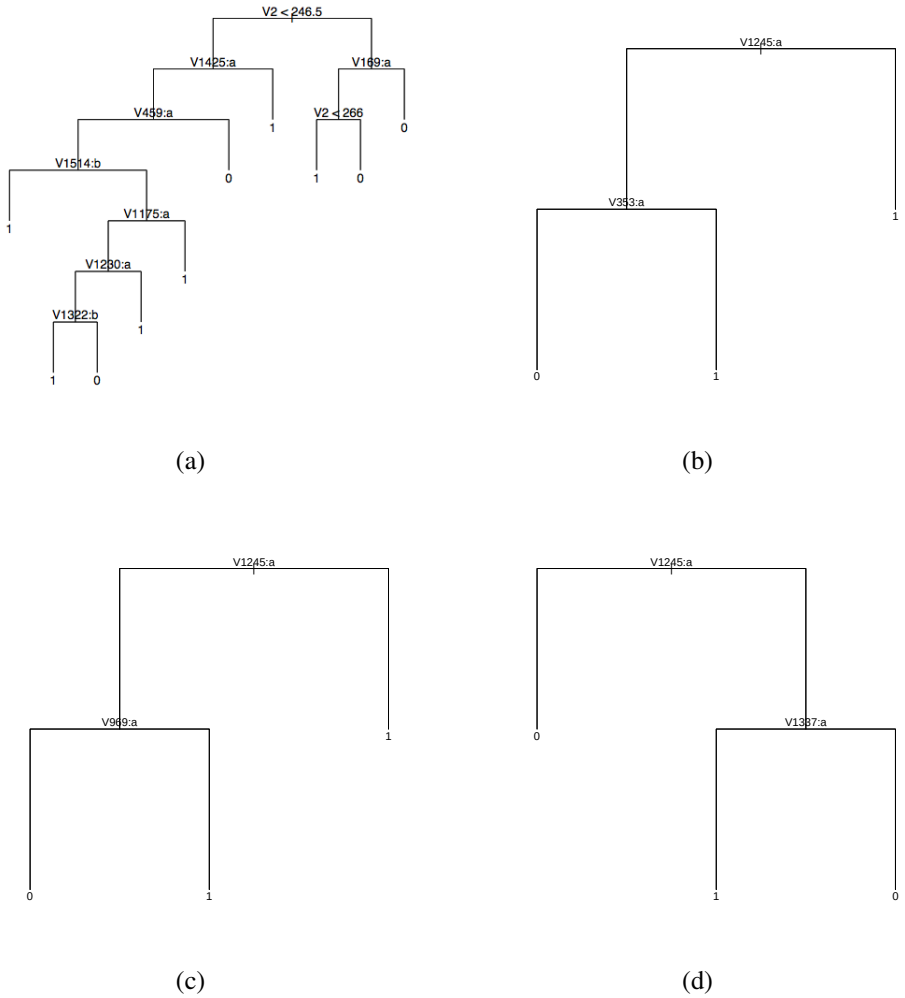
7.2 Case Study

In this section we compare the ALoVaS method against the CGM approach, a variable importance (VIMP) based frequentist method [**?**], and a non-decision tree based approach, the logistic-lasso. We compare these methods using the publicly available internet ads dataset from the UCI data repository [**?**]. The internet ads data set contains 1558 covariates, some numeric and some categorical. The data set was first used in a paper by Kushmerick [**?**] and has since been used in several statistical classification studies. A more complete description of the data and references to other papers using this dataset can be found at the UCI machine learning repository website, at the following link: `https://archive.ics.uci.edu/ml/datasets/Internet+Advertisements`.

As noted by Kushmerick [**?**], there are several reasons for wanting to remove advertisements from webpages. Some reasons are: images tend to dominate a pages total download time, users dislike paying for services indirectly through advertisers preferring direct payment for services rendered, and malicious software can be unintentionally placed on a user's machine through images masked as advertisements.

Model	Error
ALoVaS-SSVS	7.48%
ALoVaS-Lasso	5.12%
ALoVaS-Horseshoe	5.40%
VIMP	15.30%
CGM	7.20%
Logistic-lasso	11.20%

Table 8: Class unbalancedness simulation study results, each entry represents the MSE for that scenario and model.



(a) (b)

(c) (d)

Figure 42: The decision trees fitted to the internet ads dataset. The CGM (a) is much deeper than the remaining three trees fit using the ALoVaS method. The SSVS (b), Lasso (c) and horseshoe (d) all select the 1245th variable along with a second variable. The second variable differs in each of the three ALoVaS variants we use but the fact that the 1245th variable is common to all three gives further credence to the claim that a choice between the SSVS, lasso, or horseshoe types of ALoVaS is largely a choice of personal preference.

We use a random subset of the internet ads data to fit a CGM tree and an ALoVaS tree. We first remove all observations that contain missing values. We then take a 50% random sample of training and test data. The MSE is calculated as $\sum_i(\hat{y}_i - y_i)^2$/10,000, where i runs from draw $1,000$ to $11,000$. Thus our burn-in is $1,000$. Figure 42 shows the estimated trees. The misclassification

probabilities are calculated by dropping all data points from the test data set through the trees built using the training data. The resulting trees are fit using only the training data. In this example we do not know the ground truth, so comparison of misclassification rates against a ground truth is not possible. Nevertheless, we compare misclassification rates amongst the models shown in Table 8.

From the misclassification errors shown in Table 8, it is clear that some of the trees compete with each other in terms of prediction. However, once we examine the images of the fitted decision trees in Figure 42 ,we see that there is a dramatic difference in terms of the complexity of the tree for a given prediction error. The ALoVaS models are able to select trees with maximum depth of 2 and achieve comparable error rates to the CGM tree which has more splits on more variables.

Comparing the different decision trees selected by the three types of ALoVaS methods indicates that there is little practical difference between the SSVS, lasso, and horseshoe ALoVaS methods. Additionally, the variables selected by the three ALoVaS methods are very similar. Hence, the ALoVaS method is relatively insensitive to the regularization method used. Saying that, it is clear that there is a substantive performance increase over non-sparse competitors such as the CGM method and the logistic lasso. CGM is a nonlinear dense method, whereas logistic lasso is a linear, sparse method. In a nutshell, ALoVaS merges the benefits of both CGM and the lasso. The variable 1245 indicates that the webpage includes in the url the term 'psu.edu', showing that at the time the internet ads dataset was collected, Penn state's website was hosting advertisements. The other variable that is commonly occurring is variable 353, which indicates that the url term contains the text string 'images+ads', which seems to be an obvious choice of predictor. Further the variable 1337 indicates that the url string contained the set of characters 'sj', which is a difficult set of characters to interpret and is likely ancillary information.

The take home point of this section is that, not only is the ALoVaS method a useful conceptual method for performing variable selection when fitting Bayesian decision trees, but it is also practically useful. The ALoVaS method is capable of handling data with a large number of covariates and of performing covariate filtering by choosing various covariates. As a byproduct of the fitting method we also generate a sampled set of covariate selection probabilities which can be used if the method is to be compared with other methods that perform variable selection and use a similar

approach.

7.3 Conclusions

A lot of press is given to variable selection methods, both in terms of theoretical properties and in terms of practical results. However, these applications nearly always implement linear or generalized linear models. Non-linear models, such as decisions trees, are mathematically less tractable and thus theoretical results are more difficult to derive. Our ALoVaS method facilitates variable selection methods familiar to the researcher and allows their use with decision tree models. In non-sparse settings our method experiences similar difficulties as those experienced by methods such as the lasso and horseshoe. This is not a limitation of our method *per se* but rather an effect of model misspecification.

One difficulty with the ALoVaS method proposed in this article is that exact zeros and ones are impossible unless $\Pr(\mu_j \in \{-\infty, \infty\}) \neq 0$. This is a difficulty of all models using the ALN density and was noted by Atchison and Shen [**?**], as a point in need of remedy. Although exact zero covariate selection probabilities are not theoretically possible with our model, we are able to create sparse, simple decision trees via the ALN density by estimating zero probabilities as numerically negligible.

Of course, there are other distributions defined on the simplex that one might find useful to use for variable selection with decision trees. Perhaps using one of the distributions on the simplex from Jørgensen's [**?**] dispersion model apparatus would be a fruitful line of research. However, analytical intractability would result from using this type of distribution. Additionally, one would need to generalize the definition of this density to a simplex on more than 2 dimensions. Perhaps, if other densities on the simplex are deemed applicable, the application of those densities to decision tree variable selection could follow a line of reasoning similar to that described here.

There remain several avenues of future work based on the findings documented in this manuscript. We list three:

- Explore other regularization methods that have non-zero probability of setting $\mu_j = 0$.

- Study the effect of imposing normality on the split counts, s_{ij} instead of a Poisson or Negative Binomial model.

- Study the effects of multiplicity adjustments, false positives, and false negatives in the models detailed.

Regarding the first comment, the model proposed by Bhattacharya, Pati, Pillai, and Dunson [**?**] might be a useful first step. For the second item listed, the Gibbs sampling method proposed by Polson, Scott, and Windle [**?**] seems to be the way forward. Finally, regarding the last item, the arguments in Scott and Berger [**?**], as well as Efron's many articles and book length treatise [**?**, **?**, **?**, **?**], are logical starting points.

In this paper we proposed a novel framework for linking variable selection or regularization methods from linear models with variable selection in decision tree models. Moreover, we compared simulated and real data using our proposed ALoVaS method against common competitors, such as the logistic lasso and the CGM decision tree approach. Our ALoVas method showed improved prediction errors over the logistic lasso and the CGM methods. Moreover, our method provided sparser, less deep decision trees compared to other decision tree methods. This allows easier interpretation of the model both for trained analysts and for non-technical stakeholders. While it is inevitable that limitations exist with all methods, it appears that for sparse data generating processes where a simple structural model is desired, our ALoVaS method is to be preferred.

8 Synthesis: Comparing the DiVaS and ALoVaS Methods

In this section we compare the DiVaS and ALoVaS models. While both methods seem relatively similar, both methods performing variable selection for Bayesian decision trees, there are fundamental practical and theoretical differences between the two methods.

8.1 Theoretical differences and a simulation study

One of the most apparent theoretical differences between the DiVaS and ALoVaS models is the way the two models handle the correlation between covariate selection probabilities. One of the properties of the Dirichlet distribution is that $Corr(p_i, p_j) < 0$ for $i \neq j$. Thus, if the prior puts zero probability on non-negative covariate selection probability correlations, then the posterior will also place zero probability on these non-negative correlations. Figure 43 plots the relationship of the DiVaS and ALoVaS methods in the correlation space of two arbitrary covariate selection probabilities (p_i, p_j). This section elucidates the differences between the two methods and compares the similarities.

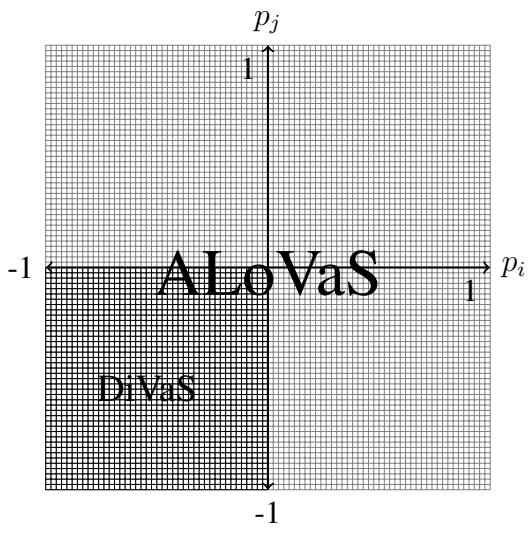


Figure 43: A plot of the Correlation space where non-zero prior probability is placed in the DiVaS and ALoVaS models.

One might be tempted to conclude that this is a element of minutæ that has no practical con-

sequence but consider a linear model data generating process that contains an interaction between two variables, x_1 and x_2

$$y_i = \beta x_1 x_2 + \epsilon_i, \tag{110}$$

where $\epsilon_i \sim N(0, \sigma^2)$. In this case the coefficient β controls the degree of curvature in the (x_1, x_2) space. The positive covariate selection probability results from using a constant function in each terminal node to approximate a curved two dimensional surface. The constant function requires alternating splits between x_1 and x_2 to approximate the curving of the (x_1, x_2) space. Thus, we see that there are really two important pieces to the model specification when fitting decision trees: specifying a tree model and specifying a terminal node model. Misspecification of one can lead to errors in the other, indicating that the two are also not independent.

A nice property of the ALoVaS method is that if p_i, p_j are the two covariate selection probabilities, and if these p_i, p_j have a logistic normal density, then the covariance between the two densities is

$$Cov(log(p_j/p_k), log(p_l/p_m)) = \sigma_{jl} + \sigma_{km} - \sigma_{jm} - \sigma_{kl}, \tag{111}$$

where σ_{ij} is the covariance parameter for the normal random variable before the application of the ALT. Clearly the four $\sigma_{ij} \geq 0$ in Equation 111, so that the covariance could be positive or negative.

At this point it is worth noting that the idea of an automatic interaction detector was first proposed in the Chi-squared Automatic Interaction Detection (CHAID) method [**?**]. The CHAID method claims to automatically detect interactions, yet popular documents [**?**] claim that CHAID and other decision tree methods cannot actually perform automatic interaction detection. The validity of this claim is not the point of this paragraph but it should be clear to the reader how one could visually inspect a fitted tree and determine if interaction is plausible. Moreover, if the ALoVaS method automatic interaction detection is plausible, by comparing the sampled values of $Cov(log(p_j/p_k), log(p_l/p_m))$, or equivalently $Corr(p_i, p_j)$, with the point zero, one has an indication of the degree of interaction. Additionally, we cannot determine if the interaction is positive or negative, only that an interaction is present. If there was a desire to determine the sign of the

Figure 44: write the caption here.

Figure 45: write the caption here.

interaction, then the analyst would need to provide estimates of the parameters in Equation 111. Alternately, we might estimate directly $Corr(p_i, p_j)$ with the Bayesian approach we take in the ALoVaS method. Either approach is feasible.

We now present a small simulation study. We sampled data according to the model describing in Equation 110. We simulated $n = 500$ observations from this model for a training dataset and another for a test, or hold-out dataset. We simulated data under the model with $\beta \in \{5, -5\}$. Figure 44 plots the two trees fitted under a greedy optimization for the two interaction coefficients.

We ran the ALoVaS and DiVaS algorithms for $11,000$ samples discarding the first 1000 as a burn-in sample. Convergence statistics were calculated using the coda package in R with convergence statistics indicating a minimal sample size of 3764 and 3271 for the ALoVaS and DiVaS chains respectively. Figure 45 presents the trees with the largest integrated likelihood for the two chains.

8.2 Practical differences and a simulation study

While the theoretical differences are important to academics and applied researchers who want to understand the limitations of a specific method, the questions most applied researchers tend to have are more practical in nature. We address the practical questions in this section.

Practically speaking, the ALoVaS and the DiVaS methods are different in how you think about selecting a variable to split on, or to remove, from the current decision tree. While the DiVaS method allows us to *quantify* these different scenarios, it does not allow us to think about these values in a manner we are familiar with. On the other hand, the ALoVaS method allows us a nice, linear framework to understand these differences. The primary difficulty lies in no standard, intu-

itive, basis to describe the simplex, the space of values in d dimensions such that $\sum_j q_j = 1$ and $q_j \geq 0$. Aitchison [**?**] discussed several bases, none of which are intuitive, whereas Euclidean geometric space is intuitive, ubiquitous, and statisticians usually have experience with the Euclidean space.

There are other more practical differences between the DiVaS and ALoVaS methods. For example, the ALoVaS method takes much longer to simulate. This increased simulation time should be no surprise given the additional number of parameters to simulate in the ALoVaS model relative to the DiVaS model. In the ALoVaS model we typically have means, variances, and covariances to simulate for each covariate and global parameters to simulate as well. This difference is one reason why the sampling takes longer. A second reason comes from the methods used to simulate the parameters from the DiVaS and ALoVaS methods. While the Dirichlet density is complicated because of the relationship with the gamma density there are highly optimized sampling codes for the Dirichlet density which have been used in production since at least the early 1990s and can be considered stable and highly optimized. In contrast to this, the lasso ALoVaS model requires sampling from a generalized inverse Gaussian density which is complicated to sample from. While several algorithms exist it is unclear what is the optimal algorithm for a given scenario. Nevertheless, there are well known accept-reject methods to sample the generalized inverse Gaussian density but they require more work to draw a sample (1.58 samples on average) than more straightforward sampling methods like the polar method most commonly used to sample the Dirichlet density. In addition the sampling for the generalized inverse Gaussian density typically requires evaluating special functions like the Bessel function adding to the computational time at each iteration.

A further practical difficulty that distinguishes the two methods is the correlations between the covariate selection probabilities. Recall the DiVaS method requires that the correlations between covariate selection probabilities are negative *a priori*. Whereas the ALoVaS method allows the correlation between any two covariate selection probabilities to be any value in the range $[-1, 1]$, the typical area of support for a correlation. The practical difficulty then with the DiVaS method is that it is impossible to know whether the covariate selection probabilities are negative, positive, or zero. Thus, using the DiVaS method seems to be asking for model misspecification errors. More-

over, if the model is linear with significant interactions, as shown earlier in this chapter and we estimate the response with a decision tree, then we will have positive covariate selection probabilities. In fact any model having a curvature in two or more of the covariates implies that when we build a decision tree we will have a positive covariate selection probability regardless of whether the generative model is linear or not, curvature of the model indicates positive covariate selection probabilities.

A reasonable question the reader may ask now is: if positive correlations of the covariate selection probabilities are a result of curvature when do negative correlation occur? In this case we would need covariates to split in the decision tree but we would want to select either x_1 or x_2 but not both. As one simple example this might occur if the generative model was

$$y_i = \beta_1 x_1 + \beta_2 x_2 + \epsilon_i, \tag{112}$$

for $i = 1, \ldots, n$. The two covariates x_1 and x_2 were such that $Corr(x_1, x_2) \neq 0$. In this case we see that knowledge of one covariate indicates, at least approximately,knowledge of the other covariate as well. We would then need the generative model to be linear because the correlation is a bilinear operator. We could choose to split on $x_1 < c$ for some constant c or we could choose to split on $x_2 < c'$ for some value c' because the two covariates are correlated. To reiterate, knowledge of one covariate implies approximate knowledge of the other covariate. Therefore, we would only need to split on one and not the other covariate. Thus, if we split on covariate x_1 for example, the prevalence of a split on covariate x_2 would not only be unlikely, but it would also be unnecessary, provided $|Corr(x_1, x_2)|$ was large enough.

8.3 Recommendations

The similarities and the differences between the ALoVaS and DiVaS methods having been pointed out, we now turn to the conclusions and recommendations we have for the use of the two methods.

While the DiVaS method is a theoretically sound method, several theoretical and practical difficulties arise when trying to implement the method. In contrast, the ALoVaS method has few theoretical difficulties and the practical difficulties are primarily in terms of simulation time which

will typically not be an issue for scientific problems and most problems with no dynamic time varying component. Therefore, problems involving decision trees where the data occur on a timescale of less than one day and especially less than one hour will likely not find use of the DiVaS and ALoVaS methods. Those needing these short timeframe models are likely to find some use of the time varying decision trees proposed by Gramacy, Taddy, and Polson [**?**]. Moreover, the DiVaS approach is more of an ad-hoc method lacking a unifying idea, whereas the ALoVaS method has one underlying concept, the ALT to transform from a linear scale to a probability scale.

The conclusion then seems straightforward. If one is to choose between the DiVaS and the ALoVaS methods, one would choose the ALoVaS method because of the theoretical soundness, the conceptual link with linear methods, and the ease of implementation. The key drawbacks are the run-time of the algorithm and the need to choose one of the regularization techniques discussed in this text or one of the plethora of other regularization techniques available in the statistics literature. However, as indicated in Section 7 the choice of which ALoVaS method to use is dependent largely on the underlying DGP, the knowledge of which would preclude the model fitting exercise. Also, our simulation studies presented in Section 7 largely indicate that, within a given sparsity level, the choice of which type of ALoVaS method to choose is largely irrelevant. To reiterate, any ALoVaS method, is to be preferred to the DiVaS method.

9 Discussion

In this thesis we demonstrated two methods of variable selection for Bayesian decision trees and the necessity of these methods in comparison to currently used methods such as the approaches of Chipman et al. [**?**] and Denison et al. [**?**]. Moreover, we showed the drawbacks of simplistic methods of variable selection like the pruning rule and bootstrapping. While we focused on the variable selection aspects of the study of decision trees, there are several avenues forward from here and this chapter discusses those paths.

The ability to provide good fitting decision trees with accurate predictions is of vital interest to practitioners in many fields. The use of simplistic models in the terminal nodes such as a constant mean model or a highest class probability model will inevitably lead to model misspecification in some datasets, such ass data with a large occurrence of zeros. The solution to this type of model misspecification problem is to use a zero-inflated model in each of the terminal nodes. The CGM model requires that the terminal node parameters be integrated over as shown in Equation 48 to facilitate Metropolis-Hastings samples. For the zero-inflated model this can be accomplished at least to the point of a finite series which can be calculated exactly if the number of zero observations are not too large. If the number of zero observations is too large, a numerical approximation would suffice. In future work, we intend to apply this model to several simulated and real datasets. These will include the solder data analyzed by [**?**], and the Nematode and vine root data collected by Giese et al. [**?**, **?**]. Additionally, some of these datasets could find use in applying the ALoVaS method to maintain a shallow tree and to select variables, while using an appropriate terminal node model, such as a zero-inflated model.

A second area of further research is studying the equivalence of the SSVS, lasso, horseshoe, and perhaps even the generalized double Pareto model proposed by Bhattacharya et al. [**?**]. By writing the negative log posteriors in proportional form, we find that the horseshoe prior and the lasso prior have similar forms when expanding the final term involving the logarithm. The appropriate Taylor expansion here is

$$log\left(1+(\lambda/\sigma)^2/p\right) = (\lambda/\sigma)^2/p + (\lambda/\sigma)^4/2p^2 + \cdots + (\lambda/\sigma)^{2k}/kp + \ldots. \tag{113}$$

By expanding the negative log posterior in this way, we see the asymptotic equivalence of the parameter expanded lasso and the horseshoe posteriors. It is reasonable to assume that a similar approximation will hold for the generalized double Pareto prior with density is given in Bhattacharya et al. [**?**]. While Carvalho et al. [**?**] emphasize the importance of both the singularity, or at least positive point mass probability at zero, as well as the heavy tail behavior, it is clear that the difference in the two posteriors will be governed by a power exponential density where the variance is proportional to $1/p$. Thus, in high-dimensional models, the difference will be negligible and the sparsity patterns will be similar. This seems somewhat at odds with the claims of Carvalho et al. and the claims of Bhattacharya et al. and this is an area in need of further clarification. Moreover, it remains to be seen if there is a similar asymptotic relationship with the stochastic search method. We posit the relationship exists but the difficulty lies in specifying the appropriate limiting form of the prior to give us the SSVS model while simultaneously encapsulating the lasso and horseshoe priors.

Appendix A: Algorithms Pseudo Code

Define $\mathcal{T}^0$ as the initialized tree, n as the number of iterations of each MCMC chain, $\underline{p}^0$ as the initialized probability weights and $\underline{\alpha}'$ as the initialized pseudo-counts for the splits in each dimension, s are the observed split counts from each sampled tree. As defaults we take $\underline{p}^0 \propto \underline{1}$, and $\underline{\alpha}' \propto \underline{1}$. Also N is the tree likelihood of the new or proposed tree and O is the old tree likelihood, both are on the log scale. Finally, b denotes the number of terminal nodes in the current ($\mathcal{T}^i$) decision tree.

Algorithm 4: DiVaS sampler.

Data: $n, \underline{p}^0, \mathcal{T}^0, \alpha^0, C$

for $i = 2, \ldots, \#samples$ **do**

$X \sim Discrete\_Uniform\,(1,5)$;

case (X=1) $\mathcal{T}' \leftarrow Grow(\mathcal{T}^i)$;

case (X=2) $\mathcal{T}' \leftarrow Prune(\mathcal{T}^i)$;

case (X=3) $\mathcal{T}' \leftarrow Change(\mathcal{T}^i)$;

case (X=4) $\mathcal{T}' \leftarrow Swap(\mathcal{T}^i)$;

case (X=5) $\mathcal{T}' \leftarrow Rotate(\mathcal{T}^i)$;

$N \leftarrow log(\Pr(D|\mathcal{T}'))$;

$O \leftarrow log(\Pr(D|\mathcal{T}^{i-1}))$;

case (X=1) $log(R) \leftarrow N - O + log(b)$;

case (X=2) $log(R) \leftarrow N - O + log(b+1)$;

case (X=3) $log(R) \leftarrow N - O + log(p_j) \; - \; log(p_{j'})$;

case (X=4 **or** X=5) $log(R) \leftarrow N \; - \; O$;

$U \leftarrow Continuous\_Uniform(0,1)$;

if $\{log(U) < log(R)\}$ $\mathcal{T}^i \leftarrow \mathcal{T}'$;

else $\mathcal{T}^i \leftarrow \mathcal{T}^{i-1}$;

$\underline{\alpha}^i \leftarrow \underline{\alpha}^{i-1} + \widetilde{\alpha}\underline{s}$;

$\underline{p}^i \leftarrow Dirichlet(\underline{\alpha}^i)$;

$\widetilde{\alpha} \leftarrow C \sum_{j=1}^{d} \alpha_j / \sum_{i,j} s_{ij}$;

The second pseudo-code listing contains the simple sampler approach from Chapter 6.1, the

notation is similar to the first pseudocode.

Algorithm 5: Simple sampler.

Data: $n, \underline{p}^0, \mathcal{T}^0, \alpha^0$

for $i = 2, \ldots, \#$*samples* **do**

$X \leftarrow Discrete\_Uniform\,(1, 5)\,;$

case (X=1) T′ $\leftarrow Grow(\mathcal{T}^i)$;

case (X=2) T′ $\leftarrow Prune(\mathcal{T}^i)$;

case (X=3) T′ $\leftarrow Change(\mathcal{T}^i)$;

case (X=4) T′ $\leftarrow Swap(\mathcal{T}^i)$;

case (X=5) T′ $\leftarrow Rotate(\mathcal{T}^i)$;

N $\leftarrow log(\Pr(D|\mathcal{T}'))$;

O $\leftarrow log(\Pr(D|\mathcal{T}^{i-1}))$;

case (X=1) log(R) $\leftarrow N - O + log(b)$;

case (X=2) log(R) $\leftarrow N - O + log(b+1)$;

case (X=3) log(R) $\leftarrow N - O + log(p_j) \; - \; log(p_{j'})$;

case (X=4 **or** X=5) log(R) $\leftarrow N \; - \; O$;

U$\leftarrow Continuous_Uniform(0, 1)$;

if {log(U) ¡ log(R)} T$^i \leftarrow \mathcal{T}'$;

else T$^i \leftarrow \mathcal{T}^{i-1}$;

for j $\leftarrow 1$ **to** d

u$_j \leftarrow Unif(0, \exp\,(-c_j s_j - \mu_j)^2/2\sigma_j^2)$

c$_j \leftarrow Unif(max(-a, -\sqrt{-2log(u_j)}), min(a, \sqrt{-2log(u_j)}))$

$\sigma_j^2 \leftarrow Inv - Gamma(2a_j + 2, ((c_j s_j - \mu_j)^2 + (\mu_j - \mu_j^p)^2)/2 + b_j)$

$\mu_j \leftarrow N[c_j s_j + \mu_j^p, \sigma_j^2]$

p$_j \leftarrow \exp(\mu_j)/(1 + \sum_{k=1}^{d} \exp(\mu_k))$

EndFor

p$_{d+1} \leftarrow 1 - \sum_{k=1}^{d} p_k$

–

Algorithm 9.1: SIMPLE SAMPLER(

for $i \leftarrow 1$ **to** n

$X \leftarrow Discrete\_Uniform\,(1,5)\,;$

case $(X = 1)$ $\mathcal{T}' \leftarrow Grow(\mathcal{T}^i)$;

case $(X = 2)$ $\mathcal{T}' \leftarrow Prune(\mathcal{T}^i)$;

case $(X = 3)$ $\mathcal{T}' \leftarrow Change(\mathcal{T}^i)$;

case $(X = 4)$ $\mathcal{T}' \leftarrow Swap(\mathcal{T}^i)$;

case $(X = 5)$ $\mathcal{T}' \leftarrow Rotate(\mathcal{T}^i)$;

$N \leftarrow log(\Pr(D|\mathcal{T}'))$;

$O \leftarrow log(\Pr(D|\mathcal{T}^{i-1}))$;

case $(X = 1)$ $log(R) \leftarrow N - O + log(b)$;

case $(X = 2)$ $log(R) \leftarrow N - O + log(b+1)$;

case $(X = 3)$ $log(R) \leftarrow N - O + log(p_j) \; - \; log(p_{j'})$;

) **case** $(X = 4$ **or** $X = 5)$ $log(R) \leftarrow N \; - \; O$;

$U \leftarrow Continuous\_Uniform(0,1)$;

if $\{log(U) < log(R)\}$ $\mathcal{T}^i \leftarrow \mathcal{T}'$;

else $\mathcal{T}^i \leftarrow \mathcal{T}^{i-1}$;

for $j \leftarrow 1$ **to** d

$u_j \leftarrow Unif(0, \exp\,(-c_j s_j - \mu_j)^2/2\sigma_j^2)$

$c_j \leftarrow Unif(max(-a, -\sqrt{-2log(u_j)}), min(a, \sqrt{-2log(u_j)}))$

$\sigma_j^2 \leftarrow Inv - Gamma(2a_j + 2, ((c_j s_j - \mu_j)^2 + (\mu_j - \mu_j^p)^2)/2 + b_j)$

$\mu_j \leftarrow N[c_j s_j + \mu_j^p, \sigma_j^2]$

$p_j \leftarrow \exp(\mu_j)/(1 + \sum_{k=1}^{d} \exp(\mu_k))$

EndFor

$p_{d+1} \leftarrow 1 - \sum_{k=1}^{d} p_k$

Appendix B: Non-negative Garrote Solutions when $X^TX = I$.

In this section we derive the non-negative garrote estimators under orthogonal designs. Recall a design matrix X is called orthogonal, or more properly, orthonormal, if $X^TX = I$. This implies that $\sum_i x_{ij}^2 = 1$ and $\sum_i x_{ij}x_{ik} = 0$ for $j \neq k$.

Recall the non-negative garrote objective function is

$$\underset{\forall j: c_j \geq 0}{argmin} \ \sum_{i=1}^{n}(y_i - \sum_{j=1}^{d} c_j \hat{\beta}_j x_{ij})^2 + \lambda \sum_{j=1}^{d} c_j, \tag{114}$$

Now for simplicity of exposition we will take $d = 2$ and also assume the y_i have had their mean subtracted from each observation, removing the intercept term from the model. This gives us the objective

$$\underset{\forall j: c_j \geq 0}{argmin} \ \underbrace{\frac{1}{2}\sum_{i=1}^{n}(y_i - c_1\hat{\beta}_1 x_{i1} - c_2\hat{\beta}_2 x_{i2})^2 + \lambda(c_1 + c_2)}_{=f(c_1,c_2)}, \tag{115}$$

Now we optimize over c_j by taking derivatives. This results in a system of 2 linear equations, the details follow:

$$\frac{\partial f}{\partial c_1} = \sum_i (y_i - c_1\hat{\beta}_1 x_{i1} - c_2\hat{\beta}_2 x_{i2})(-\hat{\beta}_1 x_{i1}) + \lambda \overset{\text{set}}{=} 0 \tag{116}$$

$$\frac{\partial f}{\partial c_2} = \sum_i (y_i - c_1\hat{\beta}_1 x_{i1} - c_2\hat{\beta}_2 x_{i2})(-\hat{\beta}_2 x_{i2}) + \lambda \overset{\text{set}}{=} 0 \tag{117}$$

Multiplying the extra terms through gives us the equations

$$\frac{\partial f}{\partial c_1} = \sum_i (-y_i\hat{\beta}_1 x_{i1} + c_1\hat{\beta}_1 x_{i1}\hat{\beta}_1 x_{i1} + c_2\hat{\beta}_2 x_{i2}\hat{\beta}_1 x_{i1}) + \lambda \overset{\text{set}}{=} 0 \tag{118}$$

$$\frac{\partial f}{\partial c_2} = \sum_i (-y_i\hat{\beta}_2 x_{i2} + c_1\hat{\beta}_1 x_{i1}\hat{\beta}_2 x_{i2} + c_2\hat{\beta}_2\hat{\beta}_2 x_{i2}x_{i2}) + \lambda \overset{\text{set}}{=} 0 \tag{119}$$

Now after some algebra we get the analogs to the "normal" equations in least squares

$$\hat{\beta}_1 \sum_i y_i x_{i1} = \sum_i (c_1 \hat{\beta}_1 x_{i1}^2 + c_2 \hat{\beta}_2 x_{i1} x_{i2}) + \lambda \tag{120}$$

$$\hat{\beta}_2 \sum_i y_i x_{i2} = \sum_i (c_1 \hat{\beta}_1 x_{i1} x_{i2} + c_2 \hat{\beta}_2 x_{i2}^2) + \lambda \tag{121}$$

Now applying the sum through to the RHS of both equations and applying the orthonormal conditions we get

$$\hat{\beta}_1 \sum_i y_i x_{i1} = c_1 \hat{\beta}_1^2 + \lambda \tag{122}$$

$$\hat{\beta}_2 \sum_i y_i x_{i2} = c_2 \hat{\beta}_2^2 + \lambda \tag{123}$$

The orthonormal conditions imply $\sum_i x_{ij}^2 = 1$, so we divide the $\sum_i y_i x_{ij}$ terms by this "1" term. Noting that if the x_{ij}'s are centered about their means we have

$$\hat{\beta}_j = \frac{\sum_i x_{ij} y_i}{\sum_i x_{ij}^2} \tag{124}$$

Solving for c_1 and c_2 gives the equations

$$1 - \frac{\lambda}{\hat{\beta}_1^2} = c_1 \tag{125}$$

$$1 - \frac{\lambda}{\hat{\beta}_2^2} = c_2 \tag{126}$$

and noting that $c_j \geq 0$ implies we take the positive part. These are the closed form solutions given in [**?**]. The reader can now easily generalize to the case with $d > 2$ covariates. □

The interested reader may work out the analogous results for the constraint $\sum_{j=1}^{d} c_j^2 \leq s$ for some constant s. The closed form solution in the orthonormal X case is also given in [**?**].

Appendix C

Using the result from Villa and Escobar [**?**], which states.

Theorem

Suppose $M_{x|y}(t) = C_1(t)\exp[C_2(t)Y]$ and there exists a $\delta > 0$ such that for $t \in (-\delta, \delta)$, $|C_i(t)| < \infty$ and $|M_y(C_2(t))| < \infty$ assuming $M_y(t)$ exists, then $M_x(t) = C_1(t)M_y(C_2(t))$. In more common MGF notation we have $M_x(t) = \mathbb{E}_y(M_{x|y}(t))$. See Villa and Escobar for the proof.

We an use this result to understand the mixture of normals. To do this we recall the forms of the MGFs for three common distributions:

$$M(t) = \exp\left(\mu t + \frac{\sigma^2 t^2}{2}\right) \tag{127}$$

$$M(t) = \frac{1}{1-\lambda t} \tag{128}$$

$$M(t) = \frac{\exp(\mu t)}{1 - b^2 t^2} \tag{129}$$

In MGF's 127-129, the means of the densities are μ, λ, and μ respectively. Also, the variances of the densities are σ^2, λ^2, and b^2.

Now applying the Theorem of Villa and Escobar we have

$$\begin{aligned}
\mathbb{E}\left(M_{x|v}(t)\right) &= \mathbb{E}\left(\exp\left(\mu t + 2v\sigma^2 t^2\right)\right) \\
&= \int_0^\infty \exp\left(-v + \mu t + 2v\sigma^2 t^2\right) dv \\
&= \exp(\mu t)\int_0^\infty \exp\left(-v + 2v\sigma^2 t^2\right) dv \\
&= \exp(\mu t)\int_0^\infty \exp\left(-v(1 - 2\sigma^2 t^2)\right) dv \\
&= \frac{\exp(\mu t)}{1-2\sigma^2 t^2}\underbrace{\int_0^\infty (1-2\sigma^2 t^2)\exp\left(-v(1-2\sigma^2 t^2)\right) dv}_{=1, \text{ because it is an exponential pdf}} \\
&= \frac{\exp(\mu t)}{1-2\sigma^2 t^2}
\end{aligned}$$

The result is shown.