

Supervised Learning via Bayesian Computation

Hugh Chipman, Acadia University

joint work with

Ed George (Wharton) & Rob McCulloch (Chicago)

*Coast to Coast Mathematics Seminar
November 2007*



Mathematics and Statistics at Acadia

FACULTY OF PURE AND APPLIED SCIENCES



Supervised Learning: “A sexy name for regression”

Using observed $(x_i, y_i)_{i=1}^n$ pairs, build a model for distribution of response variable Y given predictor variable(s) x .

Cases of special interest:

- Continuous response (a.k.a. regression):

$$Y = f(x) + \varepsilon$$

$$\varepsilon \sim N(0, \sigma^2) \quad (\text{independent Gaussian errors})$$

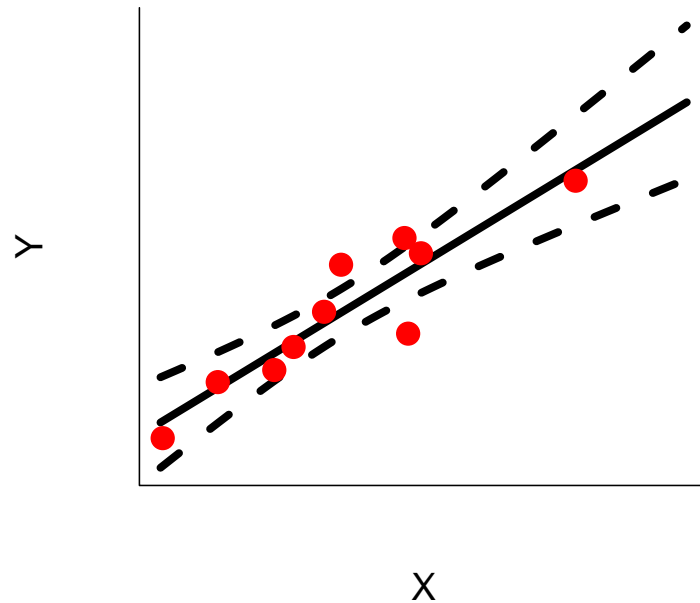
- Categorical response (a.k.a. classification)

Y takes categories labeled $1, 2, \dots, K$:

$$P(Y = k|x) = f(x), \quad (Y\text{'s are independent})$$

Just like regression:

- Basically we seek to solve the same problem as [Multiple Linear Regression](#): predict Y given a vector X of predictors.
- **Stat 101**: As with the linear model, we want two things:
 1. Good predictive accuracy
 2. Statistical inference
 - Intervals that reflect learning uncertainty: (predicted response) \pm (error bounds)
 - This is more than just an overall error - bounds can vary as a function of x .



Just like linear regression:

Let's review what we do in linear regression to build and use a model:

1. Specify a **functional form** for the model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon,$$

where ϵ is a Normally distributed “error” with mean 0 and variance σ^2 (*functional form includes probability model*)

2. Use data (i.e. observations of (Y, X_1, X_2) values for each individual) to **estimate all parameters** $(\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \hat{\sigma})$.
3. Use the data to make **uncertainty statements** about parameters or functions of them.
 - e.g., confidence interval for β_1
 - e.g., interval for prediction $\beta_0 + \beta_1 X_1 + \beta_2 X_2$
4. Report **overall performance measures** (e.g. mean squared error $\sum (y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \hat{\beta}_2 x_{2i}))^2$)

Just like linear regression:

We will follow the same strategy here:

1. **Functional form of model** - “tree model”, more complicated form than linear regression
2. **Estimate parameters** - complex model requires complex algorithms to estimate the model.
3. **Uncertainty statements** - via Bayesian statistical methods
4. **Overall performance measures** - similar to regression

In Machine Learning, “Estimation” is known as “Learning”.

- Statisticians put more emphasis on uncertainty statements, machine learners on estimation and overall performance.
- This talk attempts to bridge these approaches, combining uncertainty statements with **accurate** and **flexible** models.

Outline of the rest of the talk:

1. Tree models:
 - (a) Functional form
 - (b) Estimation (or learning) algorithms
 - (c) Uncertainty statements
2. Ensembles of trees:
 - (a) Functional form
 - (b) Brief description of estimation and uncertainty
 - (c) Overall performance
3. Active learning - an application of uncertainty.

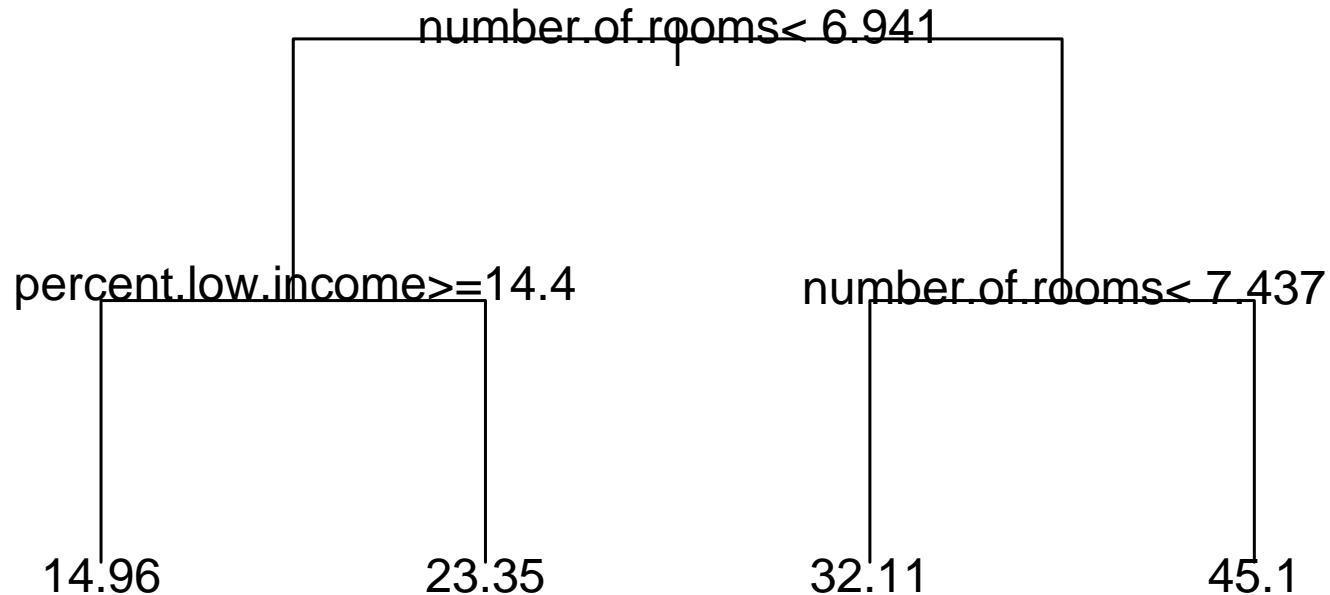
Part I: Tree models

Motivating Example: Boston Housing Prices

- Goal: Predict neighbourhood house price using demographic variables.
- Data:
 - y =median house price in the region (the response)
 - X is 13 predictors, measuring pollution, crime, house sizes, commute distance, racial diversity, tax rates, etc.
- Common “benchmark” problem.

Next slide: tree model for this data

Example: Boston Housing Prices:

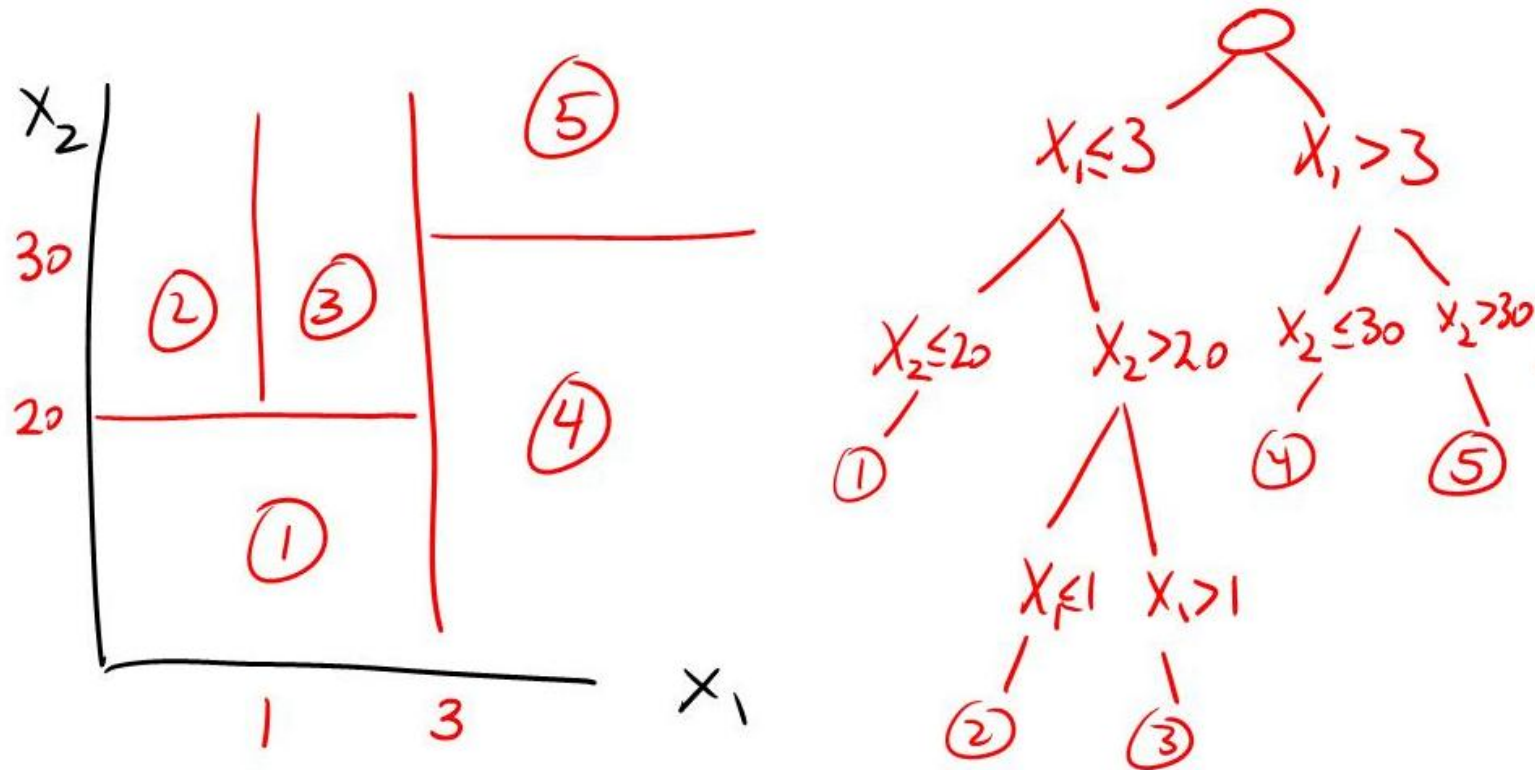


(“Real” tree models are typically larger)

How to generate predictions from a tree:

- Start at the top
- Answer each question, branching left if true, right if false.
- When you reach a terminal node, predict the response.
- Example: 5 rooms, 20% low income
⇒ predict $Y = 14.96$ (thousands of dollars)

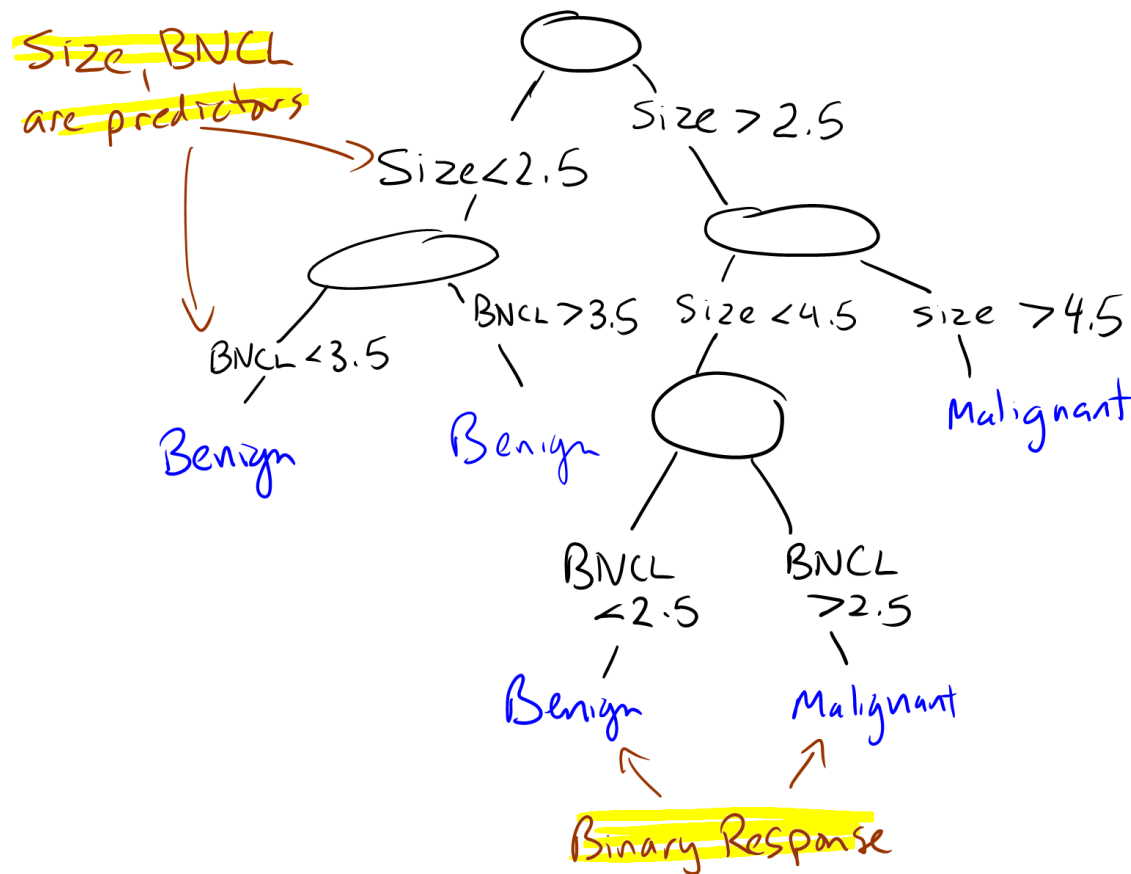
Another view of a tree: A partition of the predictor space



For continuous predictors X_1, X_2 , a tree divides the space into rectangular regions, making a constant prediction in each region.

Tree models:

Example for predicting tumors (benign/malignant) using 10 cellular characteristics



Trees are interpretable, adaptive, good with interactions and variable selection.

We must estimate (“learn”) the tree from data.

Aside: “Learning from data” was a fundamental shift from AI → ML.

Functional form of the model, continued:

The fine print: For a statistical model, we must specify

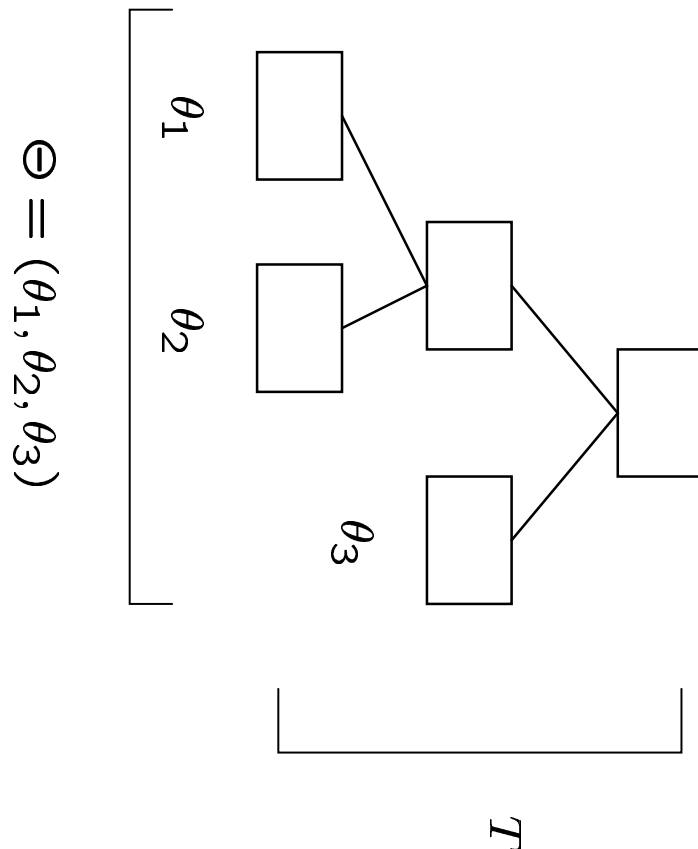
- Functional form (already done)
- Distribution of response Y given inputs X
- Unknown parameters to be estimated

Everything's a

Parameter



Functional form - parameterization:



T = Tree topology

$\Theta = (\theta_1, \theta_2, \theta_3)$

= terminal node params

Suppose we have an x that leads to node 1. Then

$$Y|x, \theta_1, \theta_2, \theta_3, T, \sigma \sim N(\theta_1, \sigma^2)$$

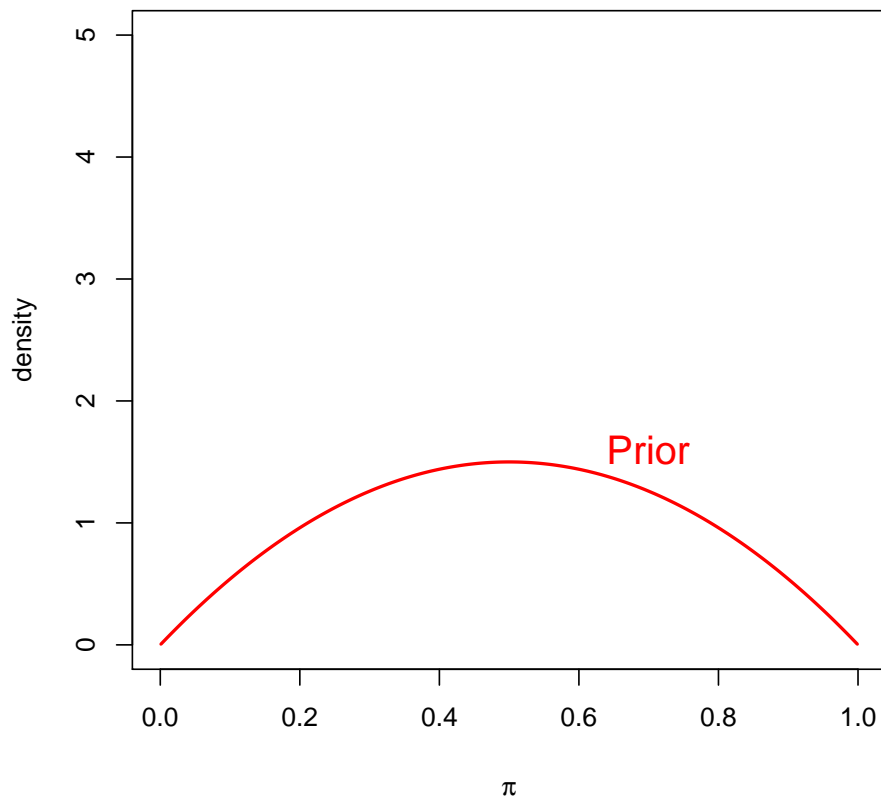
(distribution of Y given X .)

- Our goal will be to use data to estimate all parameters, and make inference about them.
- This is not trivial - it's a complex model so inference is tricky.

- We will use Bayesian methods - simple intro on next page.

Estimation and Uncertainty via Bayesian Statistics:

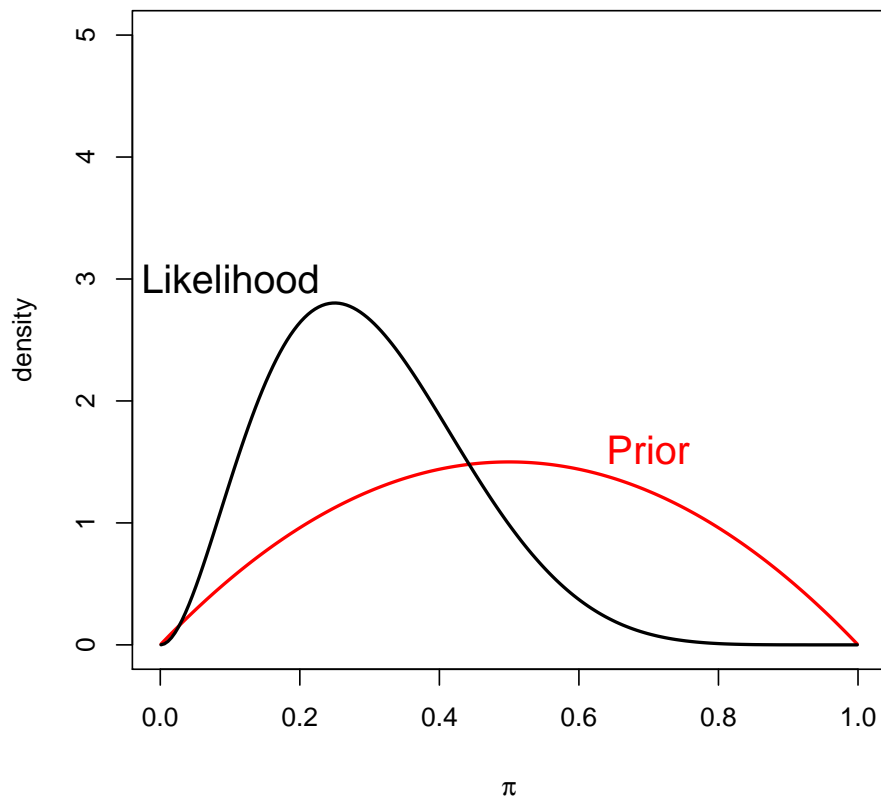
One parameter example: Estimate an unknown population proportion π . (e.g. what proportion of Canadians own a pet?)



- Prior distribution captures knowledge about π before observing any data.
- Here, I think π is more likely to be close to half than zero or 1.

Estimation and Uncertainty via Bayesian Statistics:

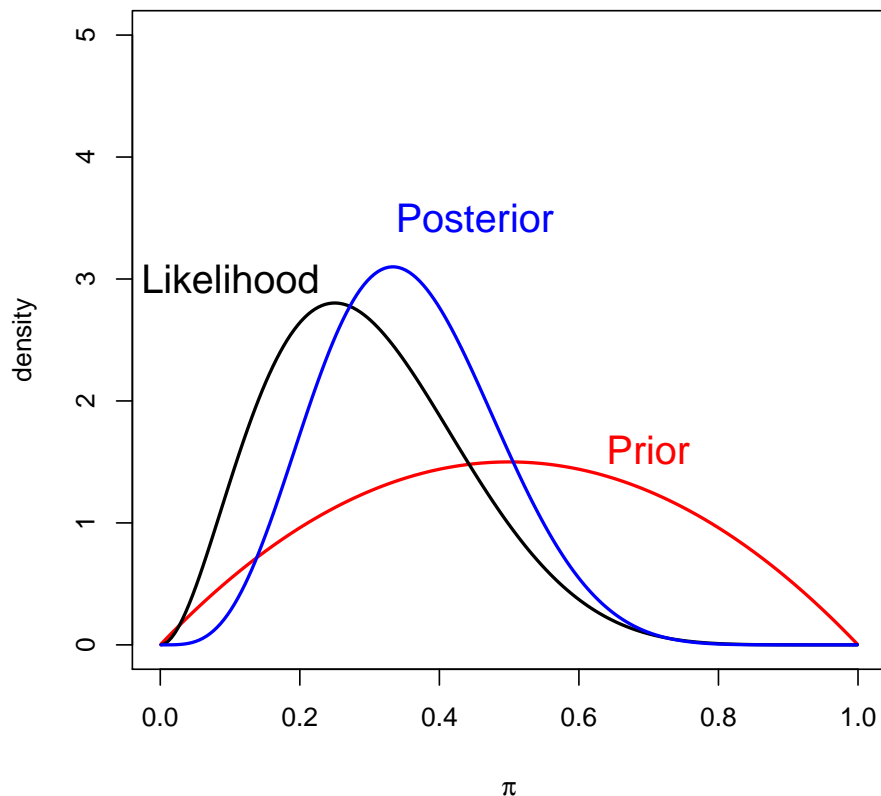
One parameter example: Estimate an unknown population proportion π . (e.g. what proportion of Canadians own a pet?)



- Suppose we observe 3 out of 10 “positive” responses. (3 out of 10 people own pets)
- Likelihood of the parameter π captures information from the data.
- $l(\pi) = P(\text{data}|\pi)$, viewed as a function of π .

Estimation and Uncertainty via Bayesian Statistics:

One parameter example: Estimate an unknown population proportion π . (e.g. what proportion of Canadians own a pet?)



- Posterior distribution combines information from the prior and likelihood, using Bayes' Theorem:

- $P(\pi|\text{data}) =$

$$\frac{P(\text{data}|\pi)P(\pi)}{P(\text{data})}$$

- Posterior =

$$\frac{\text{likelihood} \times \text{prior}}{\text{constant}}$$

For trees we will also:

1. Specify a prior distribution for (T, Θ) .
2. Compute a posterior distribution.

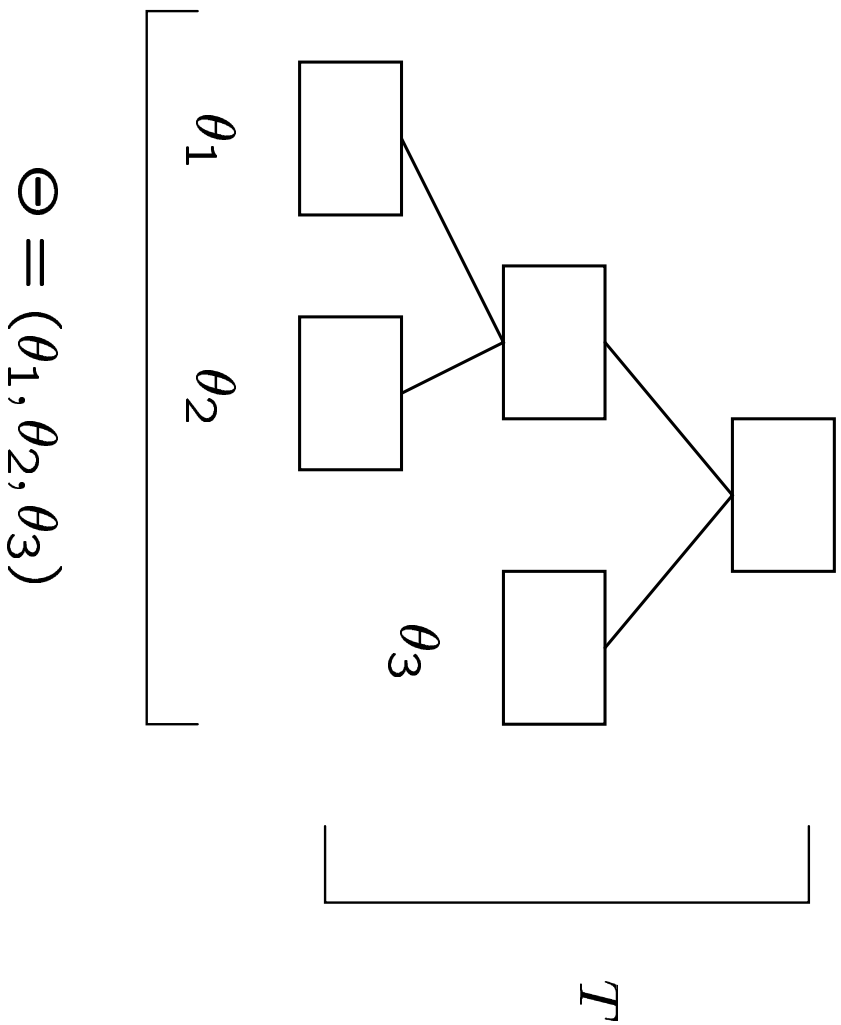
However...

- We have many more parameters (T, Θ , both of which are multivariate).
- Since trees are discrete objects, we must search for high probability trees.
- Contrast this with the “population proportion” example, which had a closed-form solution for a univariate parameter.

Prior specification:

We factor the joint prior on T, Θ as

$$P(T, \Theta) = P(T)P(\Theta|T)$$



Now we must specify a prior on

- the tree $P(T)$ and
- the terminal node parameters $P(\Theta|T)$.

Specifying the prior:

- Choosing prior parameters:
 - $P(T)$: How large a tree? What topology and rules?
 - $P(\Theta|T)$: What predictions will be made by terminal nodes?
 - $P(\Theta|T)$: How much noise (continuous responses)?
- All these prior distributions will be quite “uninformative” - that is, a wide range of values are possible.
- In practice, we develop automatic choices of prior parameters that seem to work well in a wide variety of applications (“Empirical Bayes”)

Next: Estimation and uncertainty, via computational methods

Computations for estimation and uncertainty

Markov chain Monte Carlo (MCMC):

- We want
 - $P(T, \Theta|Y)$ (the posterior probability of parameters given data Y .)
 - $P(T|Y)$ (“which trees are most probable?”)
 - $P(Y_{\text{new}}|Y)$, the predictive distribution of some future value of Y corresponding to a specific input value X_{new} .
- Not all these are available in closed form or easily manipulated expressions.
- Solution: Use an algorithm to sample values from the posterior distribution $P(T, \Theta|Y)$.

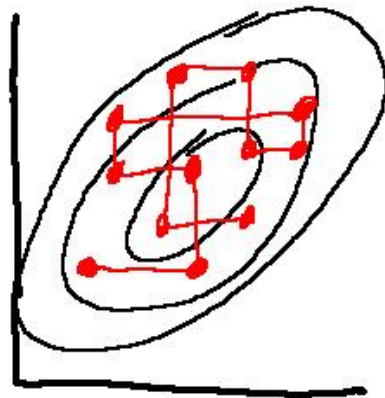
Computations for estimation and uncertainty

- MCMC: Construct a Markov chain

$$T^{(1)}, \Theta^{(1)} \rightarrow T^{(2)}, \Theta^{(2)} \rightarrow \dots$$

with stationary distribution $P(T, \Theta|Y)$.

- Gibbs sampling and Metropolis-Hastings popular MCMC approaches - see cartoons below.



Gibbs



Metropolis-Hastings

Computations for estimation and uncertainty

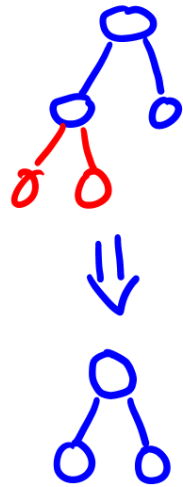
- We use the Metropolis-Hastings algorithm to move around the space of trees.
- This is a stochastic search, with the nice property that its sampling frequency corresponds to the posterior.
- Picture of move types on next page

Computations for estimation and uncertainty

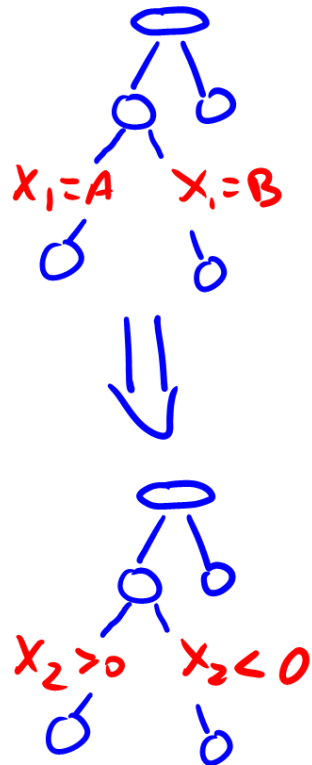
GROW



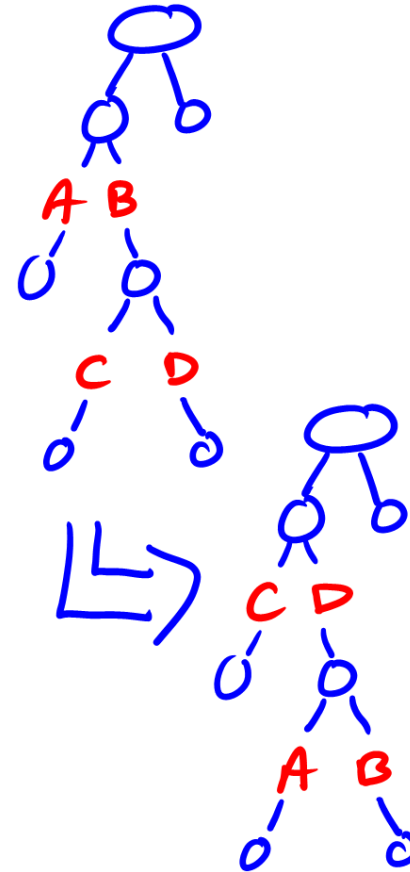
PRUNE



CHANGE



SWAP

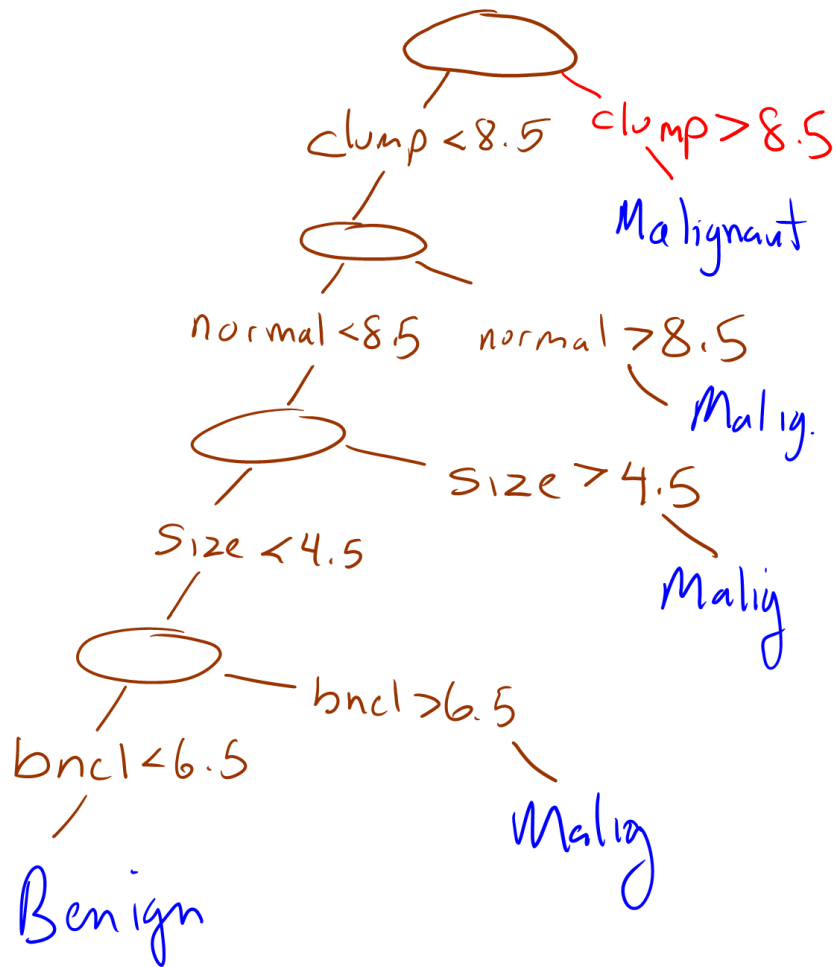


Computations for estimation and uncertainty

Compare this algorithm with conventional tree construction algorithms:

- Typically some sort of greedy, top-down search optimizes the top split, then recursively chooses daughter splits conditional on all those above.
- Grow a big tree, then prune it back to an appropriate size
- “Appropriate size” may seem to be an inference question.
 - A common technique: use part of the data to build the tree, and “hold out” a test set to choose an appropriate tree size.
 - This is often implemented as **“cross-validation”**.

Results:



- Cancer tumor example
- 19/683 observations misclassified
- Compare with 30/683 for same-sized tree constructed with greedy algorithm.
- We also get a posterior on trees
- Can “model average” across this posterior to get good predictions.

What have we learned so far?

- Bayesian learning presents a coherent framework for
 - Estimation of all model parameters
 - Stochastic search for high-probability parameter values
 - A mechanism to carry out statistical inference for complex models
- The story is the same for other flexible models: generalized additive models (Hastie and Tibshirani, 2000), neural networks (Neal, 1996), Gaussian processes (Gramacy 2006), KNN (Holmes and Adams, 2002)
- A good general reference is **Bayesian methods for non-linear classification and regression** (2002) by Denison, Holmes, Mallick, and Smith.

Part II: An extension: forests and ensemble models

“All models are wrong, but some are useful”

Tree models aren't always useful.

- Example: if the “true” model is

$$y = f(x) = x_1 + x_2 + x_3 + x_4 + x_5 + \varepsilon,$$

we need at least a 32-node tree (depth 5) to capture this structure (first split on X1, then X2, etc).

- A tree's strength is in interactions, but weakness is additivity.
- ... So we will consider a generalization...

Extension: “Forests” (a.k.a. Ensembles of trees)

- **Ensemble models** assume that $f(x)$ is actually a sum of m (often many) functions:

$$y = f(x) + \varepsilon$$

$$f(x) = g_1(x) + g_2(x) + \dots + g_m(x)$$

- Original ensemble motivation: we get a better prediction by averaging a “committee” of individual models (g_i 's).
- In practice, the g 's are often **Trees**.
- Machine Learners have come up with many clever ways to fit such models: Boosting, Bagging, Random Forests.
- ... but they aren't so much interested in uncertainty statements.

Ensembles as a statistical model: Bayesian Additive Regression Trees (BART)



Our data model is

$$Y = f(x) + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$

or more specifically,

$$Y = g(x; T_1, \Theta_1) + g(x; T_2, \Theta_2) + \dots + g(x; T_m, \Theta_m) + \epsilon.$$

with the errors being iid $\text{Normal}(0, \sigma^2)$

The parameters we have to estimate are:

- m Trees T_1, T_2, \dots, T_m .
- m sets of terminal node parameters $\Theta_1, \dots, \Theta_m$ (with b_j nodes in tree j).
- A single scale parameter σ for the residual variance.

OK. So what's different from the “one tree” model?

Model/Parameterization: It's a hugely overparameterized and unidentifiable model....

Priors:

- ... Thus priors will now also “regularize”
- For example, if we have a sum of $m = 200$ trees, we expect each tree to contribute only a small amount to the overall prediction.

MCMC:



- Adding components to a model is reasonably straightforward
- ... Just add an additional step where you draw the new parameter(s) conditional on the others.

Does it work?: A large empirical study

(conducted for *Neural Information Processing Systems 2006*)

Experimental comparison: 6 models × 42 datasets

- **The 6 models (“learners”):**
 - BART-default - (Bayesian Additive Regression Trees)
 - BART-cv (BART, but treat prior parameters like tuning parameters via cross-validation)
 - Random Forests
 - Boosting (Friedman’s gradient boosting machine)
 - Linear regression with lasso
 - Neural networks (single hidden layer)
- **The 42 datasets:**
 - From Kim, Loh, Shih and Chaudhuri (2006)
 - Up to 65 predictors and 6806 observations

Experimental comparison:

- **Guess what? BART does quite well:**
 - It's essentially a tie between everyone except the linear model, which did worse.
 - **Why we're excited:** one of the two BART versions didn't use cross-validation. We just used the default prior choices.
 - Also, we get inference, which we wouldn't have if we used cross-validation to choose our parameters.
- By the way, software is available (R library BayesTree)

So does the uncertainty buy you anything?

So does the uncertainty buy you anything?

Additional Goodies: The Boston Housing Example

- Goal: Predict neighbourhood house price using demographic variables.
- Data:
 - $y = \log$ median house price in the region (the response)
 - X is 13 predictors, measuring pollution, crime, house sizes, commute distance, racial diversity, tax rates, etc.
- Common “benchmark” problem.

Additional Goodies: The Boston Housing Example

Posterior distribution on the number of terminal nodes of the 200 trees (actually a draw from the posterior).

Number nodes	1	2	3	4	5	6
Relative freq (%)	3.5	54.5	32.5	8.5	0.5	0.5

This can be interesting because

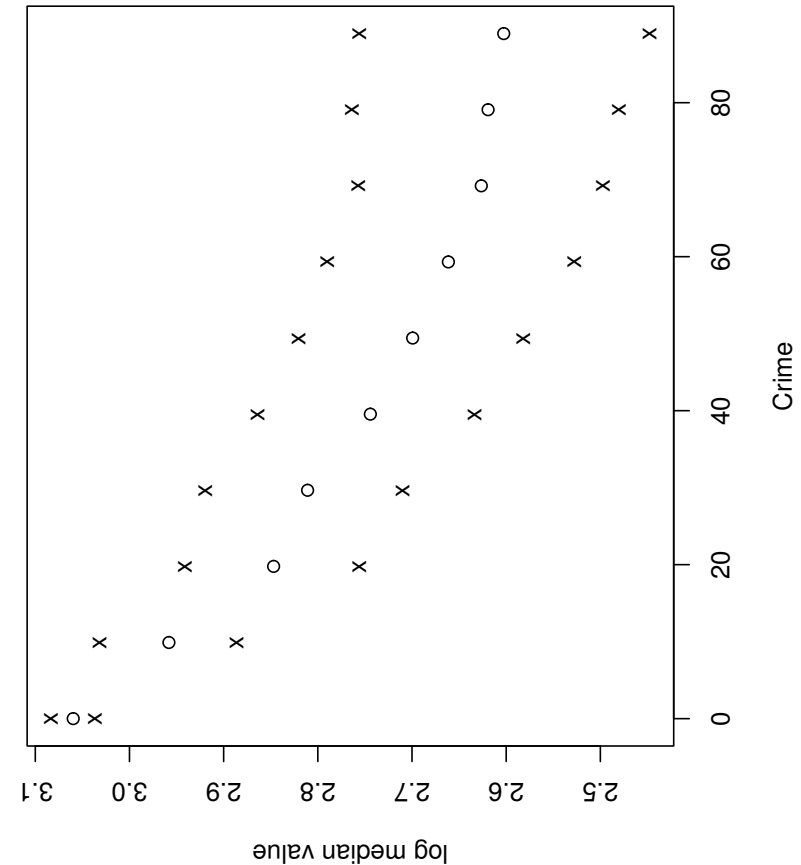
- a 1-node tree doesn't contribute to the model
- a 2 node tree is a main effect for one variable
- a 3 node tree is a two-way interaction
- ... etc.

In this case, there seems to be mostly main effects and some two-way interactions.

Additional Goodies: The Boston Housing Example

Partial Dependence Plots

- Want to measure the effect of one or two x 's on $f(x)$.
- Basically we margin over the other variables (Friedman 2001).
- Full posterior inference for such a plot is straightforward.
- Example: crime rate



- Almost all crime rates are in the 0-5 range.
- Bounds widen as we have less data (high crime rate).
- Interval width can also be a useful model diagnostic.

And Now for Something Completely Different....



Part III: Active Learning

Active Learning

The game we play:

- Same “regression” scenario as before: predict Y using X .
- The difference is that **we can sequentially choose the x 's at which we measure Y .**
- By “actively learning” (ie sequentially gathering data) we hope to build a better model with less data.
- Main idea: re-build the model as data is collected, and use it to decide where to sample more data.

Active Learning

Using the model to decide what data to collect:

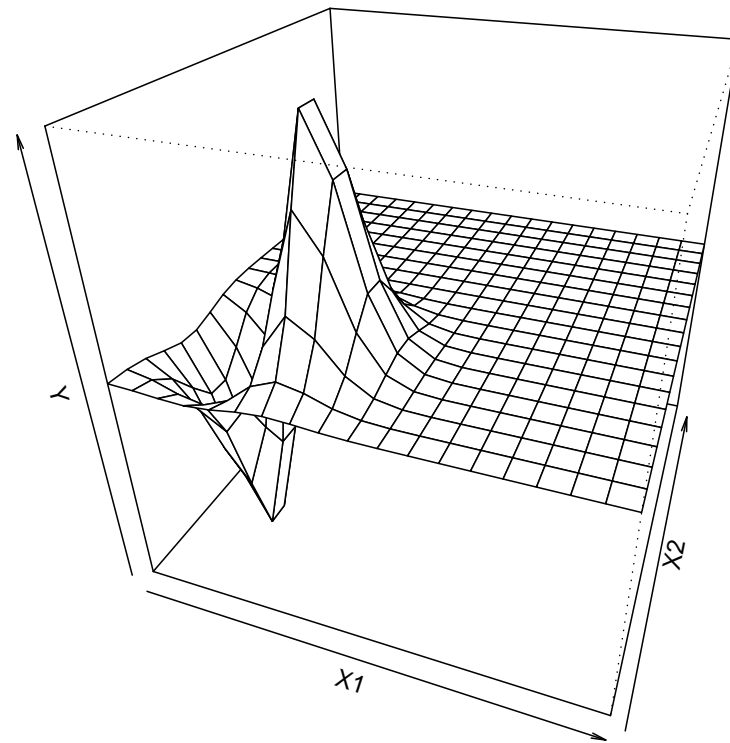
Two possible criteria:

1. Maximize variance of predicted response (“ALM” - MacKay (1992))
(*where do I know the least about Y ?*).
2. Maximize expected reduction in variance of predicted response, averaged over a candidate set C (“ALC” - Cohn (1996))
(*what data point will improve my model's predictions most?*)

We'll use # 1 here

2-D example (Gramacy and Lee 2006)

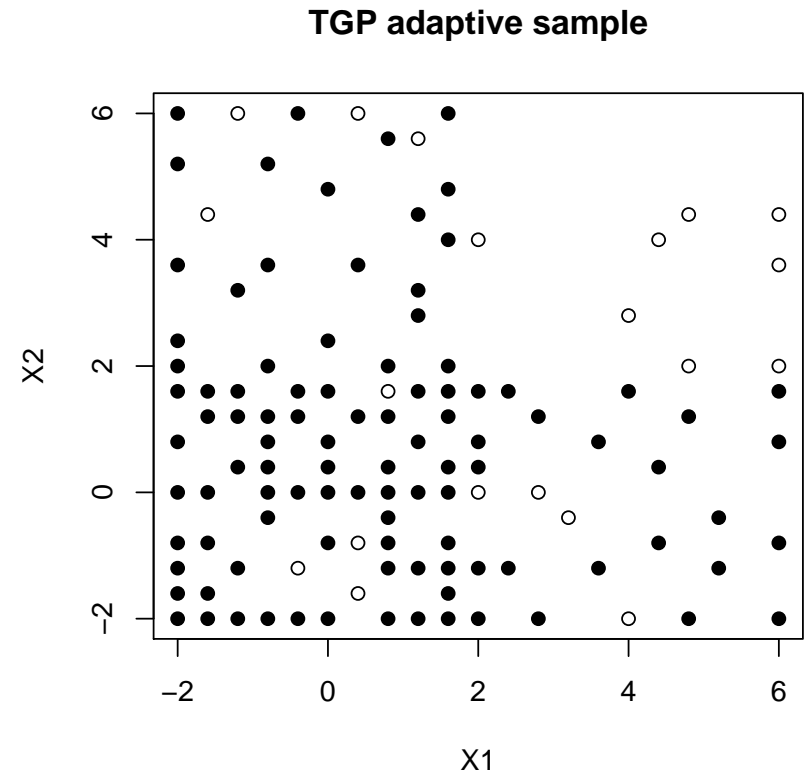
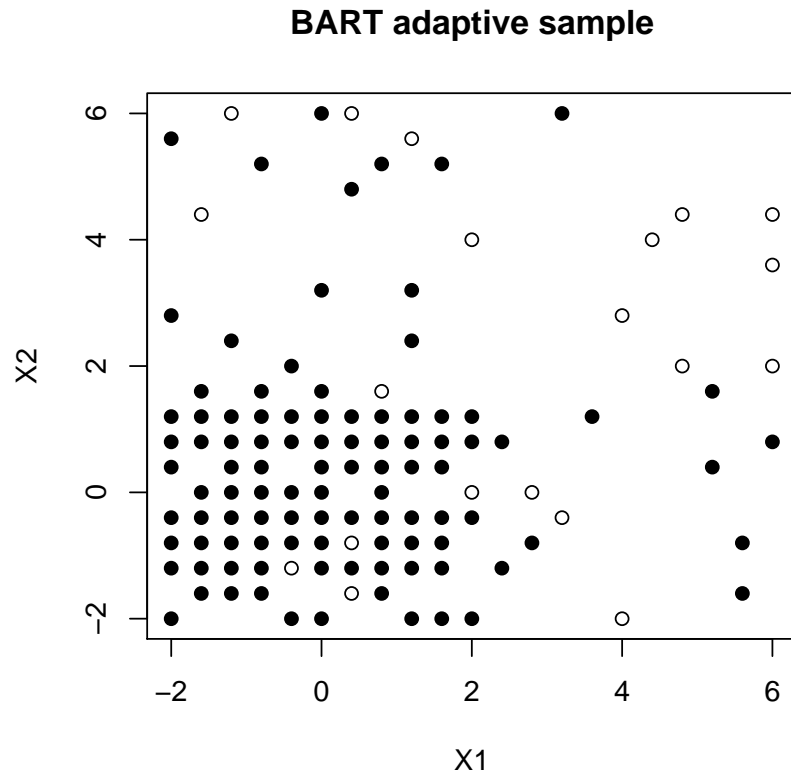
- Two predictors (X_1, X_2)
- Function (right) nearly constant in 75% of input space
- Initial random sample of 20 observations, followed by adaptive sampling of 100 observations.



We'll make comparisons with Gramacy and Lee's "Treed Gaussian Processes" (TGP)

2-D example (Gramacy and Lee 2006)

Points sampled by BART and TGP:



Test-set MSE's (right) indicate

- Both select good samples
- TGP fits better (smooth)

MSE	Model	
	TGP	BART
SRS sample	3.66	15.96
BART sample	0.35	2.84
TGP sample	0.40	2.93

The future

Extensions with BART:



Probit regression for binary response



nonparametric error distribution

More generally...

Computation:

- Fully general-purpose (ie automatic) MCMC algorithms don't exist
- Parallel computing

General directions:

- Funky new data structures (A few I've personally experienced below)
 - Functional data
 - (Social) network data

- Statistical and Machine Learners have each had a big impact on the other. Let's keep the tradition going!
- Statistics has a lot to offer:
 - Design of experiments
 - Mixed models
 - Time series
 - Diagnostics
 - Hierarchical modelling.
 - Incorporating models inside larger, more complex models (eg. BART inside a spatial model, Gaussian process models in trees)

Special thanks: University of Waterloo,
who provided computational resources to me during the strike
at Acadia