CSE/ISyE 6740
Computational Data Analysis

Feature Selection and Decision Tree

09/10/2025

Kai Wang, Assistant Professor in Computational Science and Engineering
kwang692@gatech.edu

Georgia Tech

Outline

- **Supervised Learning**
  - **Feature selection**
    - Information gain
    - Feature selection algorithm
  - **Decision tree**
    - How decision tree works
    - Threshold splits
    - Decision tree algorithm

Supervised Learning

Classification Problem

- Given a dataset $D = \{(x^1, y^1), (x^2, y^2), \dots, (x^n, y^n)\}, x \in \mathbb{R}^d, y \in \{1,2, \dots, K\}$
- Given a new feature x, can we infer the label y?
  - A function mapping $f: X \rightarrow Y$

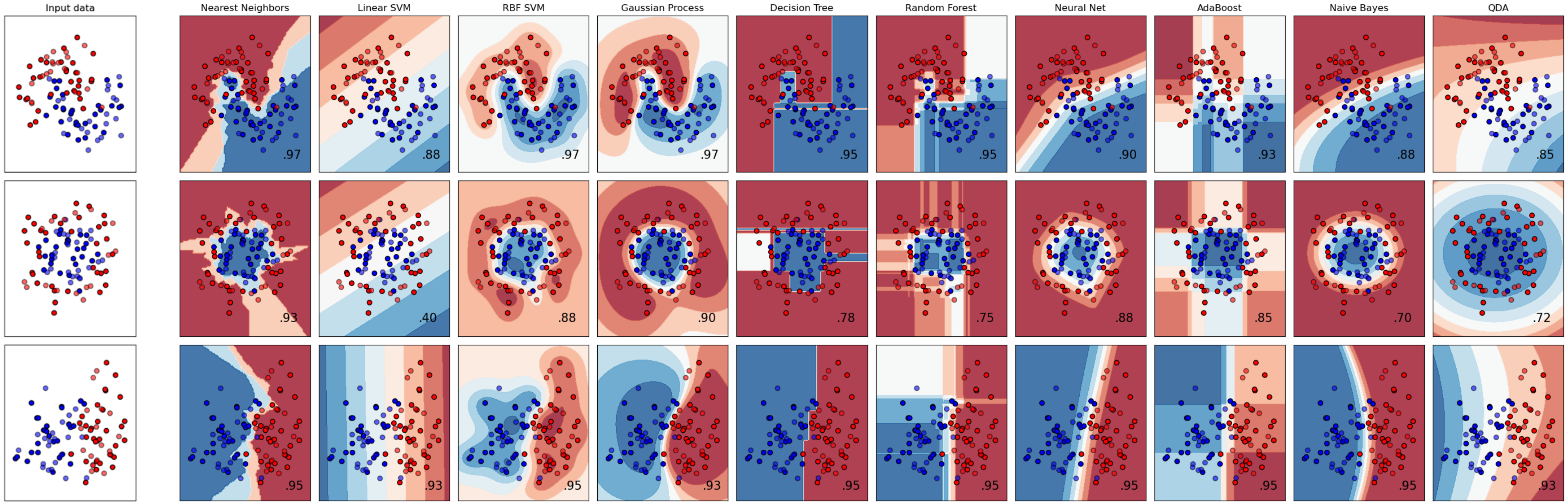


Image from scikit-learn

Feature Selection and Information Gain

Georgia Tech

Feature Selection

Task: classify emails as spam, work, …
Data: presence/absence of words

X

UNIGRAM:Viagra	0
UNIGRAM:the	1
BIGRAM:the presence	0
BIGRAM:hello Alex	1
UNIGRAM:Alex	1
UNIGRAM:of	1
BIGRAM:absence of	0
BIGRAM:classify email	0
BIGRAM:free Viagra	0
BIGRAM:predict the	1
…	
BIGRAM:emails as	1

Reduced X

UNIGRAM:Viagra	0
BIGRAM:hello Alex	1
BIGRAM:free Viagra	0

Task: predict chances of lung disease
Data: medical history survey

X

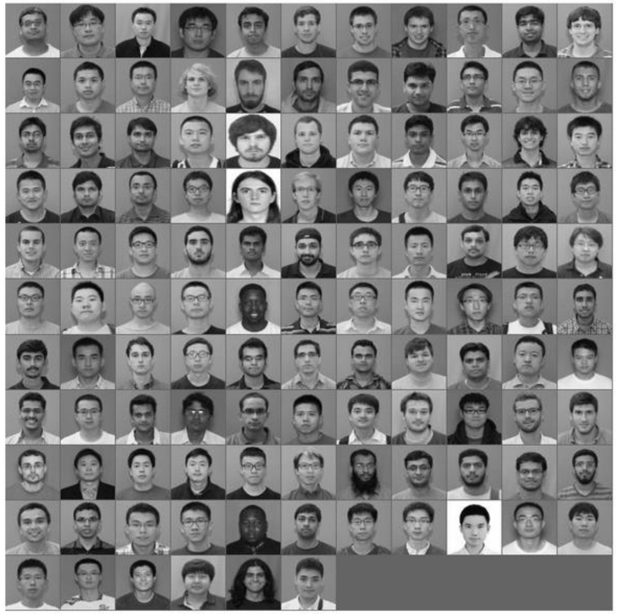
Vegetarian	No
Plays video games	Yes
Family history	No
Athletic	No
Smoker	Yes
Gender	Male
Lung capacity	5.8L
Hair color	Red
Car	Audi
…	
Weight	185 lbs

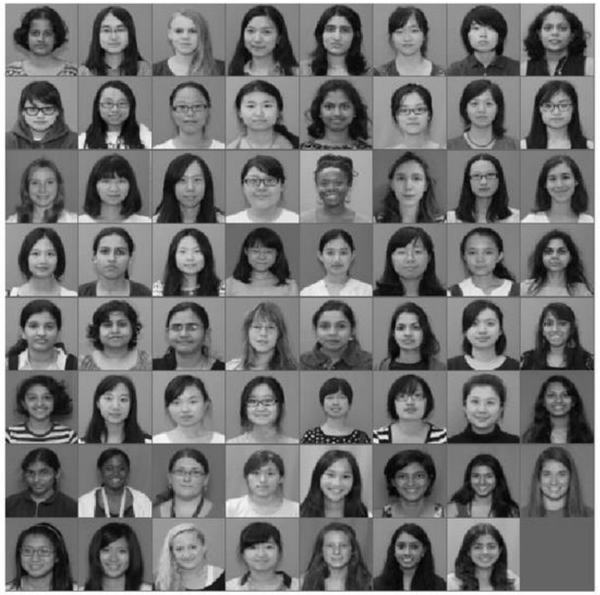
Reduced X

Family history	No
Smoker	Yes

Why Feature Selection?

- We are interested in features – we want to know which are relevant. If we fit a model, it should be interpretable.





Informativeness of a Feature

- We are uncertain about the label Y before seeing any input
  - Suppose we quantify using $H(Y)$
- Given a particular feature X_i, the uncertainty of Y changes
  - Suppose we quantify using $H(Y|X_i)$
- The reduction in uncertainty is the informativeness of feature X_i
  - $I(X_i, Y) = H(Y) - H(Y|X_i)$
- How to quantify uncertainty?

Entropy: Quantify Uncertainty

- **Entropy** $H(Y)$ of a random variable Y

$$H(Y) = -\sum_{k=1}^{K} P(y=k)\log_2 P(y=k) \qquad \textbf{(Shannon entropy)}$$

$$H(Y) = \mathbb{E}_y[-\log_2 P(y)] = -\int_{y\in D} P(y)\log_2 P(y)\,dy \qquad \text{(continuous case: \textbf{differential entropy})}$$

- $H(Y)$ is the expected number of bits needed to encode a randomly drawn value of Y (under most efficient code)

- **Information theory**
  - Most efficient code assigns $-\log_2 P(Y=k)$ bits to encode the message $Y=k$. So, expected number of bits to code one random Y is

$$-\sum_{k=1}^{K} P(y=k)\log_2 P(y=k)$$

Entropy: Quantify Uncertainty

- **“High entropy”**
  - Y is from a uniform like distribution
  - Flat histogram
  - Values sampled from it are less predictable

- **”Low entropy”**
  - Y is from a varied (peaks and valleys) distribution
  - Histogram has many lows and highs
  - Values sampled from it are more predictable

Examples for Computing Entropy

$$H(S) \coloneqq -p_+ \log_2 p_+ - p_- \log_2 p\_$$

Head	0
Tail	6

$$P(head) = \frac{0}{6} = 0 \qquad P(tail) = \frac{6}{6} = 1$$

$$\text{Entropy} = -0 \log 0 - 1 \log 1 = -0 - 0 = 0$$

Head	1
Tail	5

$$P(head) = \frac{1}{6} \qquad P(tail) = \frac{5}{6}$$

$$\text{Entropy} = -\frac{1}{6} \log \frac{1}{6} - \frac{5}{6} \log \frac{5}{6} = 0.65$$

Head	2
Tail	4

$$P(head) = \frac{2}{6} \qquad P(tail) = \frac{4}{6}$$

$$\text{Entropy} = -\frac{2}{6} \log \frac{2}{6} - \frac{4}{6} \log \frac{4}{6} = 0.92$$

Conditional Entropy

- **Conditional entropy** $H(Y|X)$ of a random variable Y given X

$$H(Y|X) = -\int_{x\in D_X}\left(\sum_{k=1}^{K} P(y=k|x)\log_2 P(y=k|x)\right)p(x)dx$$

$$H(Y|X) = -\int_{x\in D_X}\left(\int_{y\in D_Y} P(y|x)\log_2 P(y|x)\,dy\right)p(x)dx \qquad \text{(continuous case)}$$

- Quantify the uncertainty in Y after seeing feature X_i

- $H(Y|X_i)$ is the expected number of bits needed to encode a randomly drawn value of Y
  - Given X_i, and
  - Average over the likelihood of seeing particular value of X_i

Conditional Entropy

- Example

X\Y	0	1
0	$\frac{1}{4}$	$\frac{1}{4}$
1	$\frac{1}{2}$	0

- The **joint entropy** of (jointly distributed) random variables X and Y with $(X, Y) \sim p$

$$H(X, Y) = -\sum_{x,y} p(x, y) \log p(x, y)$$

- This is simply the entropy of the random variable $Z = (X, Y)$

$$H(X, Y) = \frac{1}{2}\log\frac{1}{1/2} + \frac{1}{4}\log\frac{1}{1/4} + \frac{1}{4}\log\frac{1}{1/4}$$

$$= \frac{1}{2} * 1 + \frac{1}{2} * 2 = 1.5$$

Conditional Entropy

- Example

X\Y	0	1
0	$\frac{1}{4}$	$\frac{1}{4}$
1	$\frac{1}{2}$	0

- What is $H(Y|X)$ and $H(X|Y)$?

$$H(Y|X) = \mathbb{E}_x[H(Y|X=x)]$$
$$= \frac{1}{2}H(Y|X=0) + \frac{1}{2}H(Y|X=1)$$
$$= \frac{1}{2}H\left(\frac{1}{2},\frac{1}{2}\right) + \frac{1}{2}H(1,0) = \frac{1}{2}$$

$$\neq$$

$$H(X|Y) = \mathbb{E}_y[H(X|Y=x)]$$
$$= \frac{3}{4}H(X|Y=0) + \frac{1}{4}H(X|Y=1)$$
$$= \frac{3}{4}H\left(\frac{2}{3},\frac{1}{3}\right) + \frac{1}{4}H(1,0) = 0.6887$$

Mutual Information: Reduction in Uncertainty

- **Mutual information**: the reduction in uncertainty of Y after seeing feature X_i

$$I(X_i, Y) = H(Y) - H(Y|X_i)$$

- The more the reduction in entropy, the more information a feature contains

- Is mutual information symmetric?
  - $I(X, Y) = I(Y, X)$?

A Feature Selection Algorithm

- Given a dataset $D = \{(x^1, y^1), (x^2, y^2), \dots, (x^n, y^n)\}, x \in \mathbb{R}^d, y \in \{1,2,\dots,K\}$

- For reach value of the label $y = k$
  - Estimate density $p(y = k)$

- For the i-th feature entry x_i (e.g., attendance) where $i \in \{1,2,\dots,d\}$
  - Estimate its density $p(x_i)$ using density estimation
  - For each value of the label $y = k$
    - Estimate the density $p(x_i|y = k)$
  - Score feature x_i using:

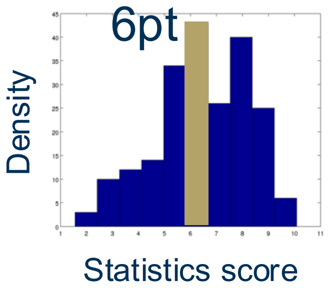
$$I_i = I(Y; X_i) = \int \sum_{k=1}^{K} p(x_i|y = k)p(y = k) \log_2 \frac{p(x_i|y = k)}{p(x_i)} \, dx_i$$

- Choose those feature x_i with high score I_i

A Feature Selection Algorithm

Student ID	Age	Gender	Study Hours/Week	Attendance (%)	Statistics Score
S001	19	Female	8	92	7.5
S002	21	Male	5	80	5.2

- Given a dataset $D = \{(x^1, y^1), (x^2, y^2), \dots, (x^n, y^n)\}, x \in \mathbb{R}^d, y \in \{1,2,\dots,K\}$

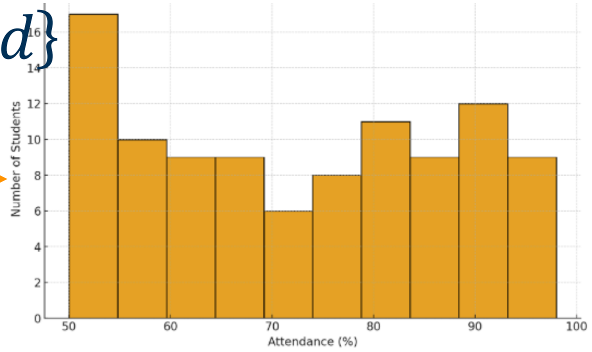


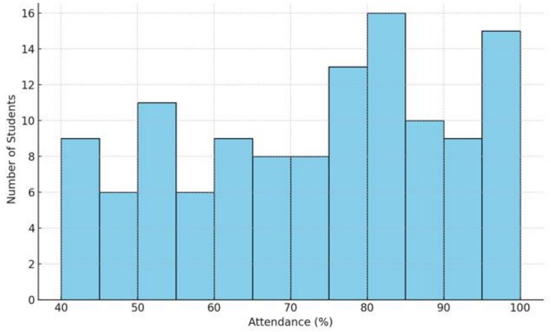
Among the students who get 6pt, what is the distribution of their attendance?

- For reach value of the label $y = k$
  - Estimate density $p(y = k)$
- For the i-th feature entry x_i (e.g., attendance) where $i \in \{1,2,\dots,d\}$
  - Estimate its density $p(x_i)$ using density estimation
  - For each value of the label $y = k$
    - Estimate the density $p(x_i|y = k)$
  - Score feature x_i using:

$$I_i = I(Y; X_i) = \int \sum_{k=1}^{K} p(x_i|y = k)p(y = k) \log_2 \frac{p(x_i|y = k)}{p(x_i)} dx_i$$

Why? Please verify it yourself!





- Choose the feature x_i with high score I_i

Classification and Decision Tree

Classification Problem

- Given a dataset $D = \{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}, x \in \mathbb{R}^d, y \in \{1,2,\ldots,K\}$
- Given a new feature x, can we infer the label y?
  - A function mapping $f: X \to Y$

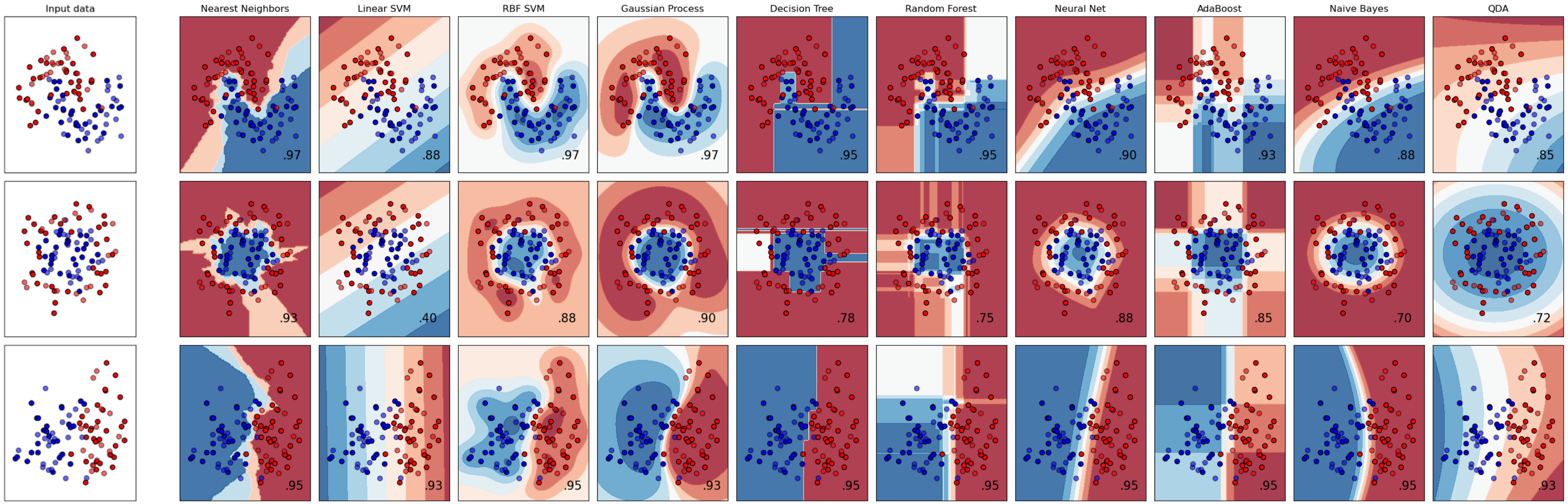
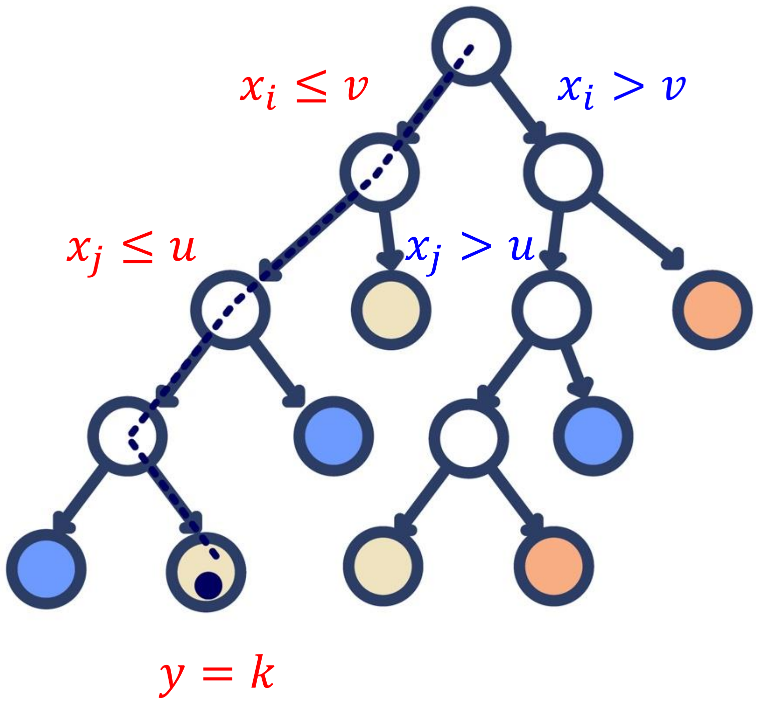


Image from scikit-learn

Hypotheses: Decision Trees!

- Each internal node tests an attribute/feature x_i
- One branch for each possible attribute value
  - $x_i \leq v$ or $x_i > v$
- Each leaf assigns a class y
- To classify input x:
  - Traverse the tree from root to leaf, output the labeled y



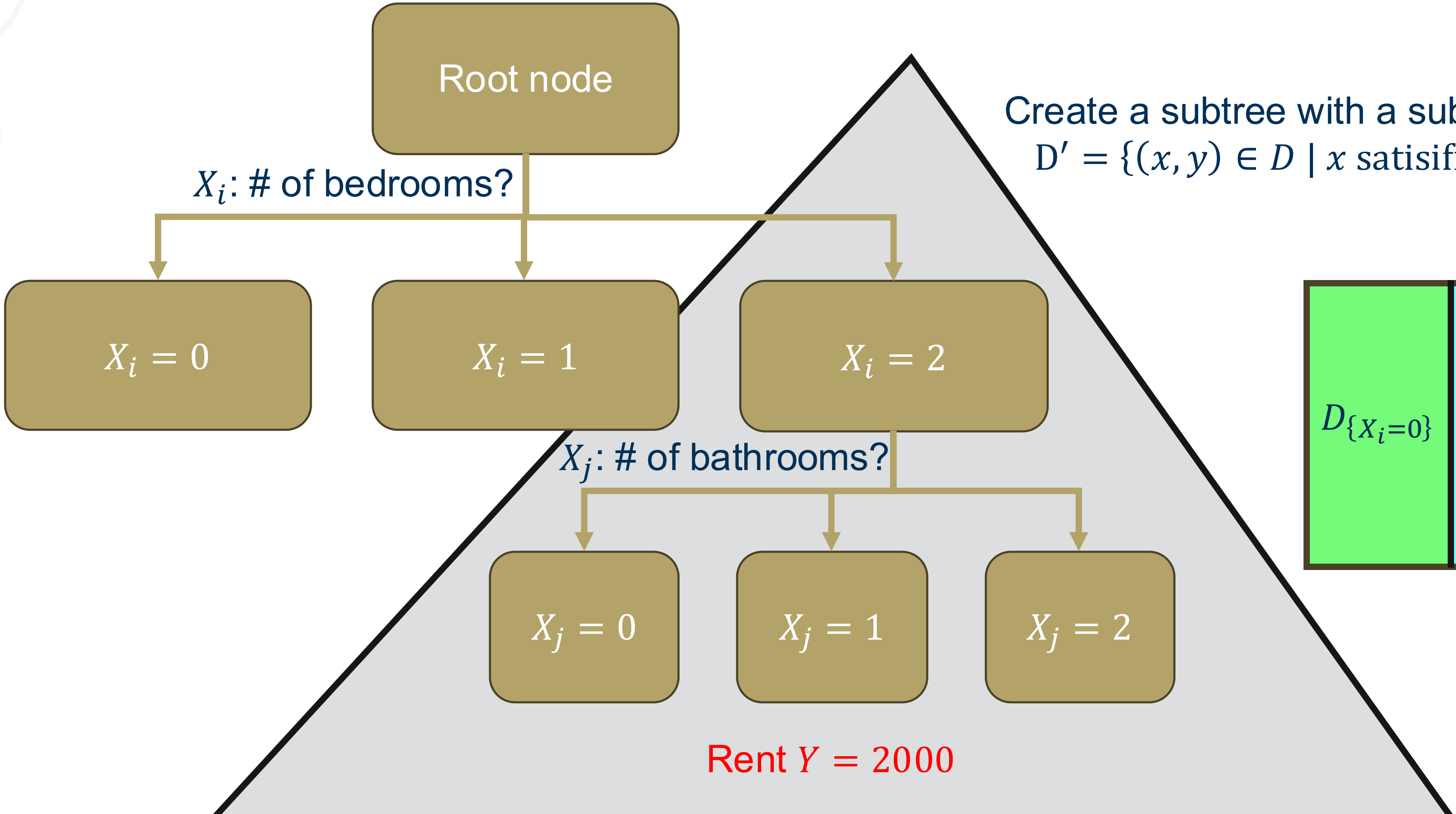
Learning Simplest Decision Tree: NP-hard

- Learning the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest ‘76]

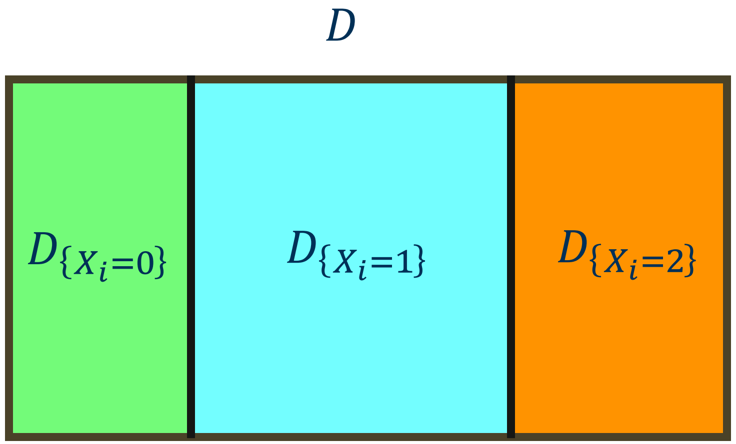
- A greedy heuristic:
  - Start from empty decision tree
  - Split on **next best attribute (feature)**
  - Recurse

Key Idea: Greedily Learn Trees Using Recursion

- Given a dataset $D = \{(x^1, y^1), \ldots, (x^n, y^n)\}$, predict Atlanta apartment rental price.



Create a subtree with a sub-dataset
$D' = \{(x, y) \in D \mid x \text{ satisifies conditions}\}$



Learning Decision Trees

- **Initialization**: start from empty decision tree with dataset D

- Based on D, compute information gain of each attribute (feature) X_i

$$I(X_i, Y) = H(Y) - H(Y|X_i)$$

- Pick **the best attribute (feature)**

$$i \coloneqq \arg\max_i I(X_i, Y) = \arg\max_i H(Y) - H(Y|X_i)$$

- Split the tree and the dataset at node X_i

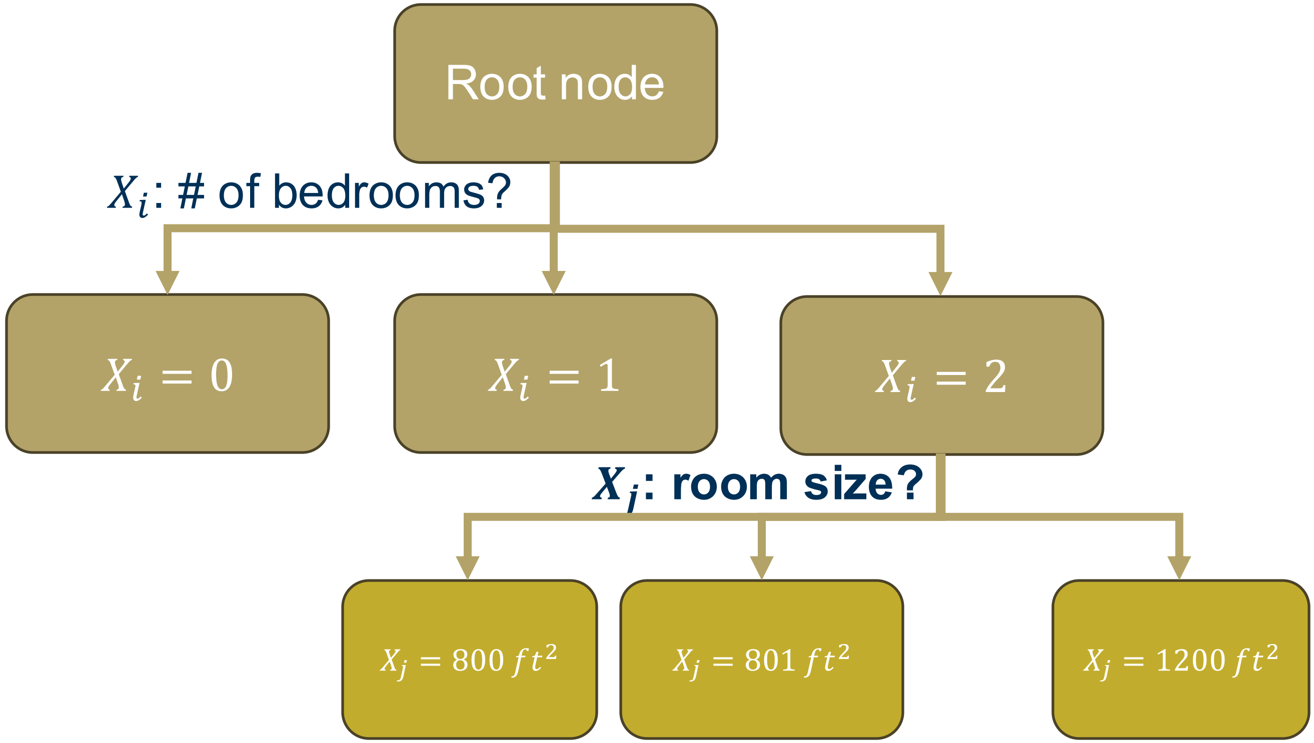
- Create subtrees and recurse through all subtrees

Real-Valued Features?

- What should we do if some of the features are real-valued?
  - E.g., room size? Distance to school?

- One branch for each numeric value?
  - Infinite number of possible split values!!!

  - Hopeless: hypothesis with such a high branching factor will easily overfit any dataset

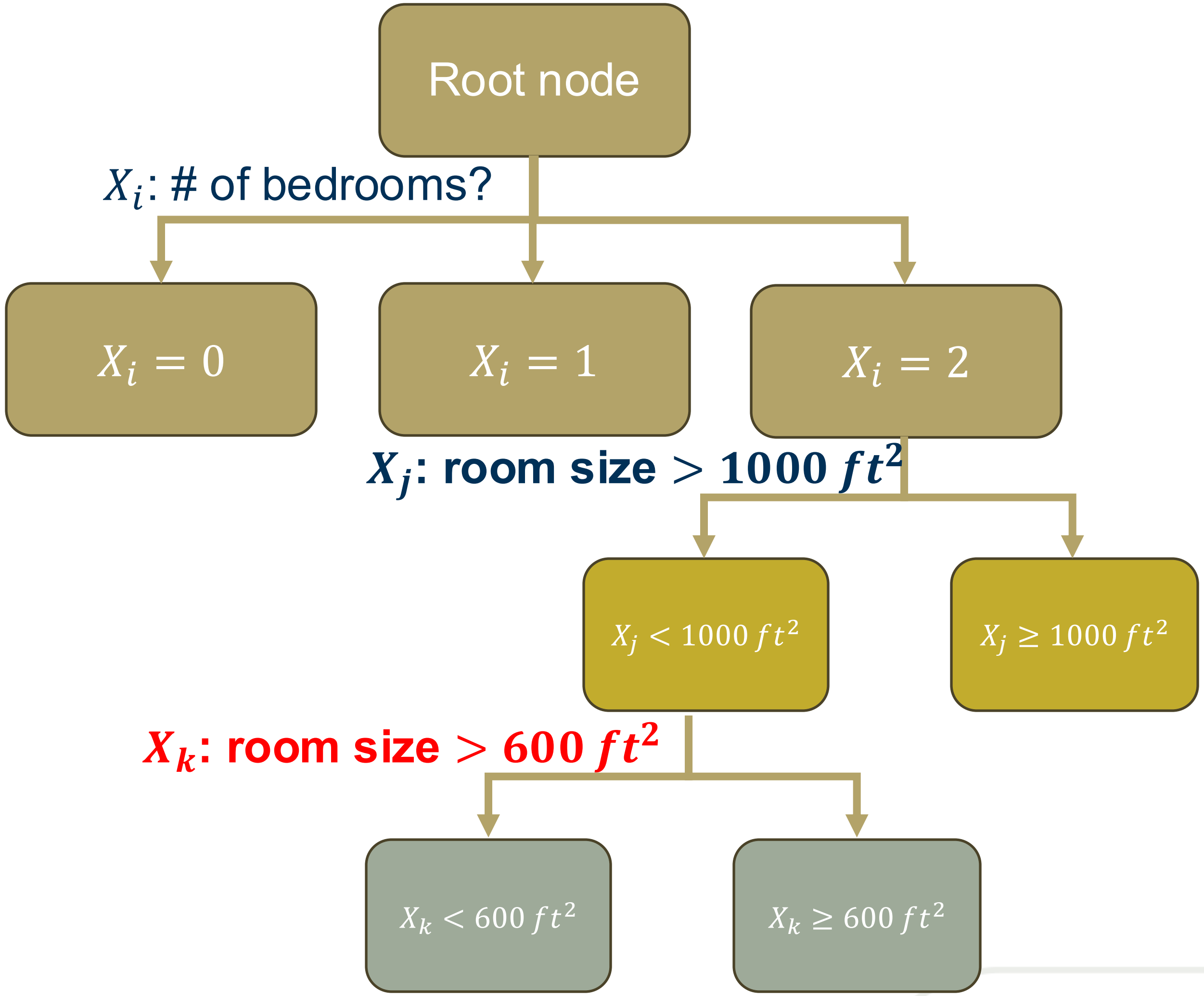


Infinite number of possible values!

Threshold Splits

- **Binary tree**: split on attribute X_i at value t
  - One branch $X_i < t$
  - Other branch $X_i \geq t$

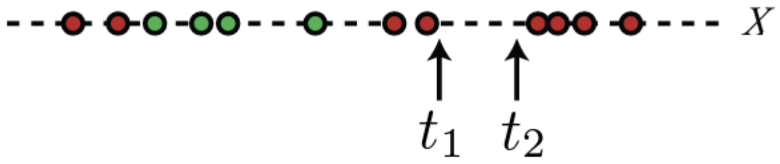
- Allow repeated splits on **same variable** along a path



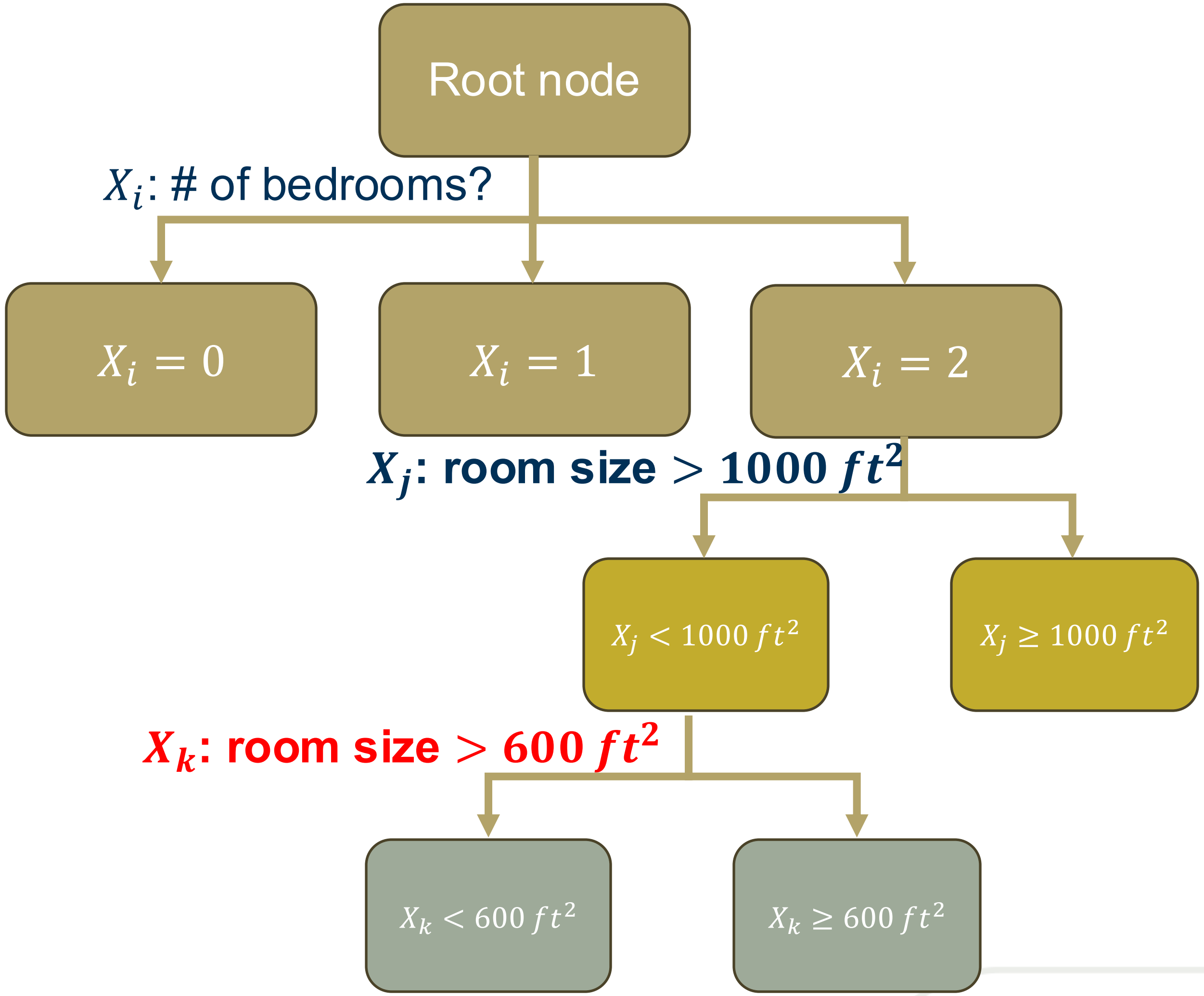
The Set of All Possible Thresholds

- **Binary tree**: split on attribute X_i at value t
  - One branch $X_i < t$
  - Other branch $X_i \geq t$

- But how do we pick the threshold t?
  - Sounds difficult?
  - But only a finite number of t's are important



  - Sort X_i in the dataset and consider splits between any two possible values



(Figures from Stuart Russell)

Learning Decision Trees

- **Initialization**: start from empty decision tree with dataset D

- Based on D, compute information gain of each attribute (feature) X_i and split t

$$I(X_i > t, Y) = H(Y) - H(Y|X_i > t)$$

- Pick **the best attribute (feature) and split t**

$$i, t \coloneqq \arg\max_{i,t} I(X_i > t, Y) = \arg\max_{i,t} H(Y) - H(Y|X_i > t)$$

- Split the tree and the dataset at node $X_i > t$

- Create subtrees and recurse through all subtrees

When to Stop?

- **Base Case One**:
  - If all data within the subtree have the same outputs, then **don't recurse**

- **Base Case Two**:
  - If all data within the subtree have exactly the same input features, then **don't recurse**

- **Base Case Three**: (**bad idea!!**)
  - If all attributes have small information gain, then **don't recurse**
  - Example: $Y = X_1\ XOR\ X_2$
  - Check:
    - Each of the splits X_1 and X_2 has 0 information gain
    - But splitting both can perfectly learn Y

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Georgia Tech

Decision Tree Summary

- Decision trees are one of the most popular ML tools
  - Easy to understand, implement, and use
  - Computationally cheap (to solve heuristically)

- Information gain to select attributes

- Can be used for regression and density estimation too

- Decision trees will overfit!!!
  - Must use tricks to find "**simple trees**", e.g.,
    - Fixed depth/Early stopping
    - Pruning
  - Use ensembles of different trees (random forests)