Decision Trees

Noa Lubin & Lior Sidi

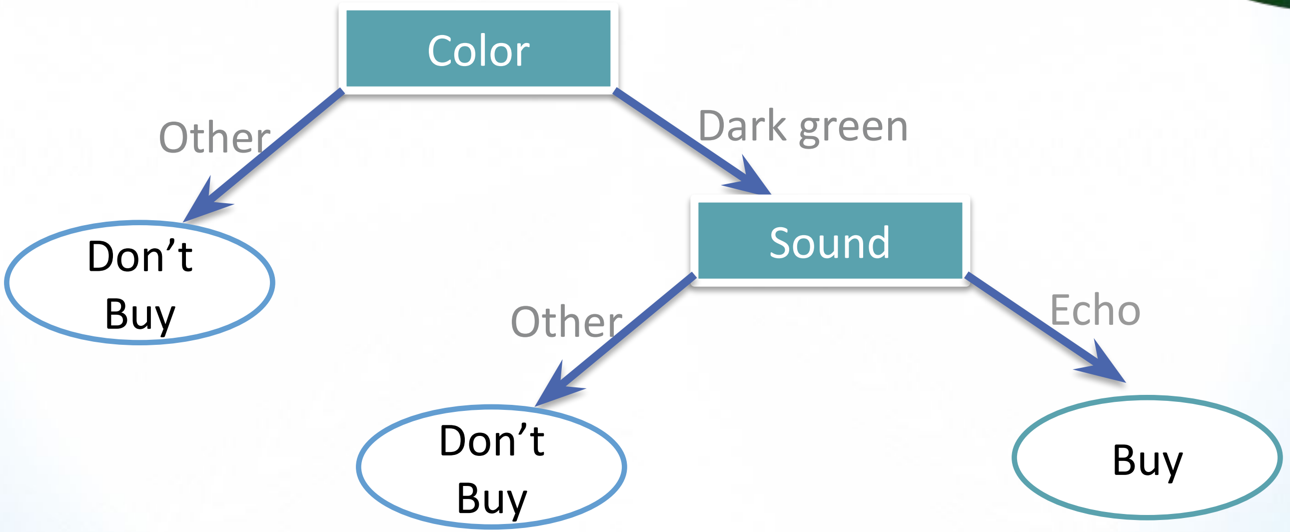
Agenda

- Motivation
- Information Theory
- Decision Trees
- Code
- Summary

Motivation

Decision Tree - What, Why, How

- Watermelon, to buy or not to buy?



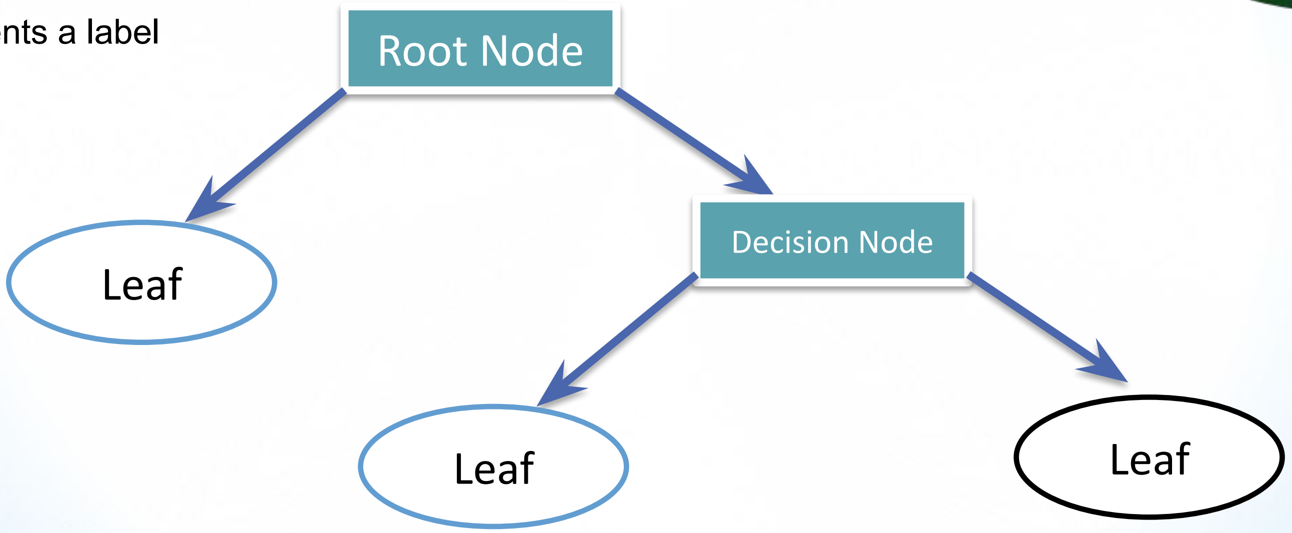
Y-DATA
SCHOOL OF DATA SCIENCE

Decision Trees Classification/regression

- Output can be both categorical or continuous
  - Classification Trees is when the predicted outcome is the class
  - Regression Trees is when the predicted outcome is considered a real number
- Model applied by traveling from a root node to a leaf
- Intuitive to understand

Terminology

- interior node corresponds to a feature
- leaf represents a label



Y-DATA
SCHOOL OF DATA SCIENCE

Decision Trees Features Can Be Categorical/Continuous

- Handles both categorical and continuous input
  - Decision split between category values
  - Decisions split based on comparison to some threshold

Y-DATA
SCHOOL OF DATA SCIENCE

Algorithm Outline

- Split the observations using the best feature
- Repeat for each child
- Stop when:
  - All the observations have the same target features value
  - There are no more features
  - There are no more observations

What is the Gain function?

- To decide how to split
- Decision tree objective is to select the split which results in **most homogeneous (lowest entropy) sub-nodes**
- in other words, the purity of the node increases with respect to the target variable

Information Theory

Information Theory Definition of Entropy

- Entropy the amount of information (bits) needed to hold the variable
- H(x) - the expectancy of the number of bits to hold the variable
- Ex:
  - Random variable with 4 values p(x) $<\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}>$

    Code $<00,01,10,11>$

    $H(x) = 2$
  - Random variable with 4 values p(x) $<\frac{1}{2},\frac{1}{4},\frac{1}{8},\frac{1}{8}>$

    Code $<0, 10, 110, 111>$

    $H(x) = \frac{1}{2}*1 + \frac{1}{4}*2 + \frac{1}{8}*3 + \frac{1}{8}*3 = 7/4$

SCHOOL OF DATA SCIENCE

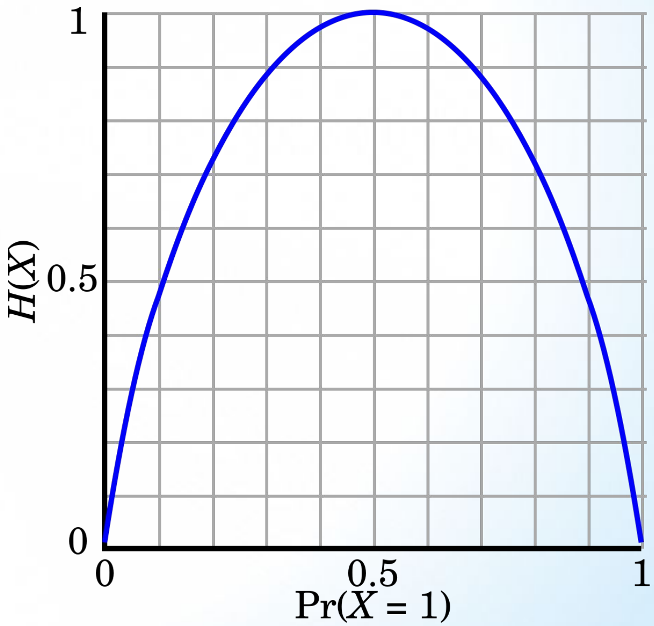
Entropy Mathematical Formulation

- Entropy is a measurement of uncertainty (chaos)
- Entropy is $H(x)=-\sum p(x)\log(p(x))$
- Example X is binary
  - with probabilities 0.5, 0.5

    $H(X)=-(0.5*\log 0.5 + 0.5*\log 0.5)=1$
  - with probabilities 0.2, 0.8

    $H(X)=-(0.2*\log 0.2 + 0.8*\log 0.8)=0.7$

What happens at the maximal point?



Entropy - Special Case

- What happens when the random variable is a binary bernoulli variable? ~ p
  Reminder: $H(x)=-\sum p(x)\log(p(x))$

- $H(x) = - p \log(p) - (1-p)\log(1-p)$ → looks familiar?

Information Gain

- **The Difference in Entropy**
- **information gain = reduction in entropy**

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in D_A} \frac{|S_v|}{|S|} Entropy(S_v)$$

Info-gain also works on categorical target variable, choosing the split with **maximal weighted gain in entropy reduction**, calculated as parent entropy minus sub-node entropy (given the splitting feature category), or simply **minimal weighted entropy**, where the Entropy is defined by

Y-DATA
SCHOOL OF DATA SCIENCE

Decision Tree

ID3 Classification Decision Tree Algorithm

- Iterative Dichotomiser 3 (ID3)
- Top-bottom algorithm
- Greedy algorithm
- Calculate the Information Gain for each feature
  - Gain(S, feature)= H(S)-H(S|f) , where f is a feature
- Choose feature with max Gain as the node and Split the set S into subsets based on the chosen feature values
- Recourse on subsets using remaining features

Y-DATA
SCHOOL OF DATA SCIENCE

Decision Tree Example

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Y-DATA
SCHOOL OF DATA SCIENCE

What will be our Root?

The information gain values for the 4 attributes are:
Gain(S,Outlook) =
Gain(S,Humidity) =
Gain(S,Wind) =
Gain(S,Temperature) =

where S denotes the collection of training examples

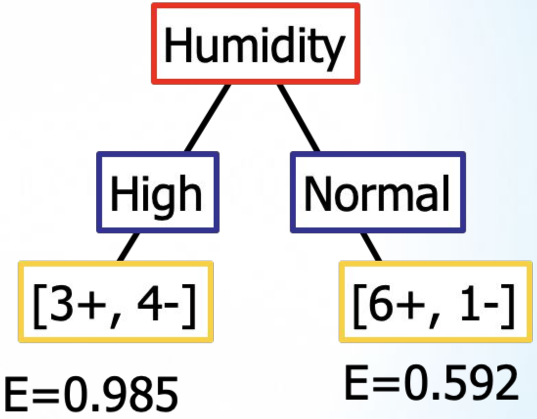
Play Tennis
No
No
Yes
Yes
Yes
No
Yes
No
Yes
Yes
Yes
Yes
Yes
No

Training Examples Entropy
S=[9+,5-]
H(S) = E
= -(5/14*log(5/14) + 9/14*log(9/14))
=0.940

Humidity

Humidity	Play Tennis
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

S=[9+,5-]
E=0.940

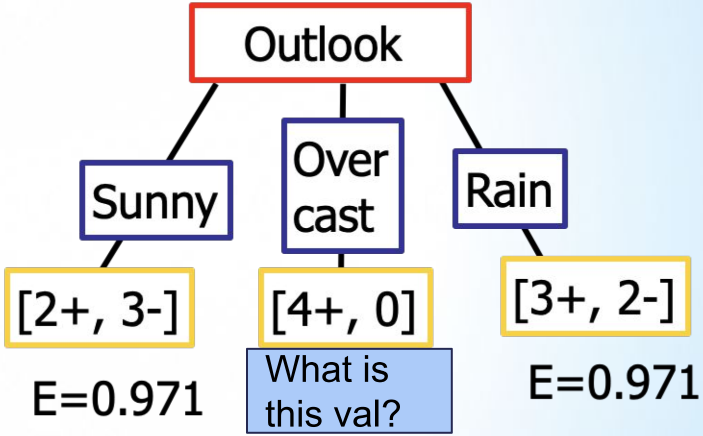


Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

Outlook

Day	Outlook	Play Tennis
D1	Sunny	No
D2	Sunny	No
D3	Overcast	Yes
D4	Rain	Yes
D5	Rain	Yes
D6	Rain	No
D7	Overcast	Yes
D8	Sunny	No
D9	Sunny	Yes
D10	Rain	Yes
D11	Sunny	Yes
D12	Overcast	Yes
D13	Overcast	Yes
D14	Rain	No

S=[9+,5-]
E=0.940

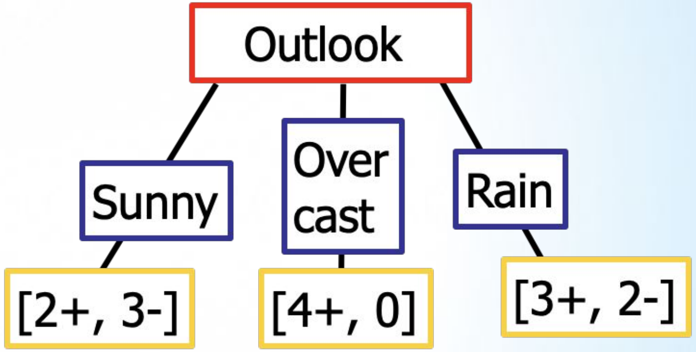


Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Outlook

Day	Outlook	Play Tennis
D1	Sunny	No
D2	Sunny	No
D3	Overcast	Yes
D4	Rain	Yes
D5	Rain	Yes
D6	Rain	No
D7	Overcast	Yes
D8	Sunny	No
D9	Sunny	Yes
D10	Rain	Yes
D11	Sunny	Yes
D12	Overcast	Yes
D13	Overcast	Yes
D14	Rain	No

S=[9+,5-]
E=0.940



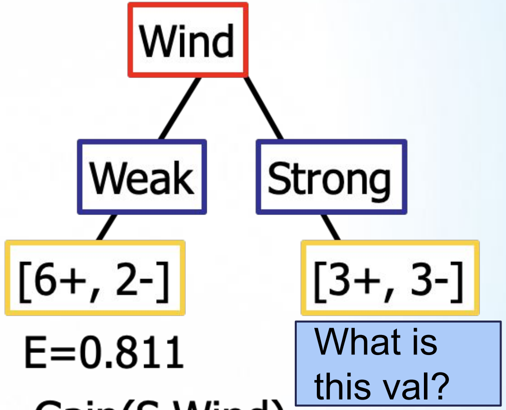
E=0.971 E=0.0 E=0.971

Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Last One We’ll Do Together: Wind

Wind	Play Tennis
Weak	No
Strong	No
Weak	Yes
Weak	Yes
Weak	Yes
Strong	No
Weak	Yes
Weak	No
Weak	Yes
Strong	Yes
Strong	Yes
Strong	Yes
Weak	Yes
Strong	No

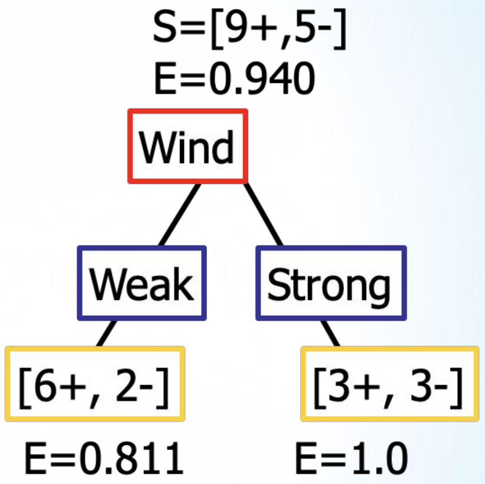
S=[9+,5-]
E=0.940



Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

Last One We'll Do Together: Wind

Wind	Play Tennis
Weak	No
Strong	No
Weak	Yes
Weak	Yes
Weak	Yes
Strong	No
Weak	Yes
Weak	No
Weak	Yes
Strong	Yes
Strong	Yes
Strong	Yes
Weak	Yes
Strong	No



Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

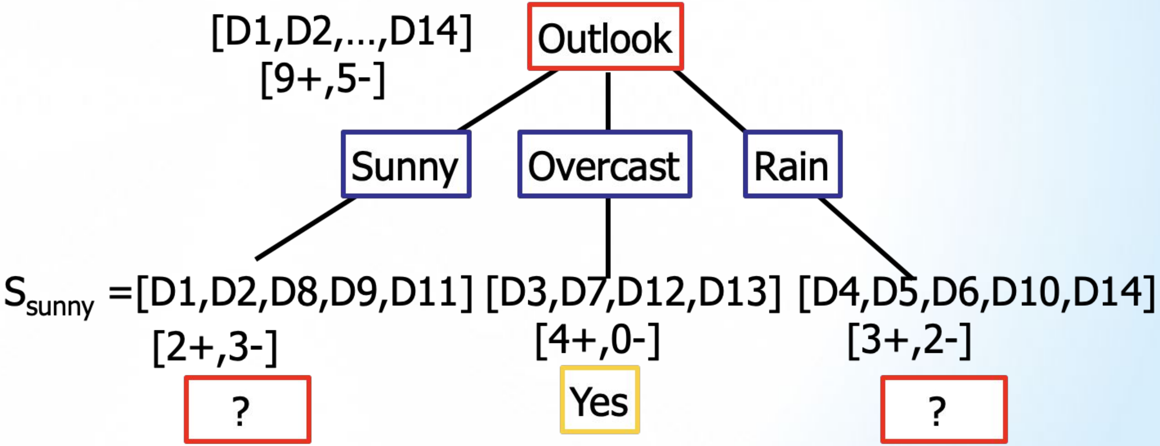
The Chosen Feature: Outlook

The information gain values for the 4 attributes are:

<u>Gain(S,Outlook) =0.247</u>
Gain(S,Humidity) =0.151
Gain(S,Wind) =0.048
Gain(S,Temperature) =0.029

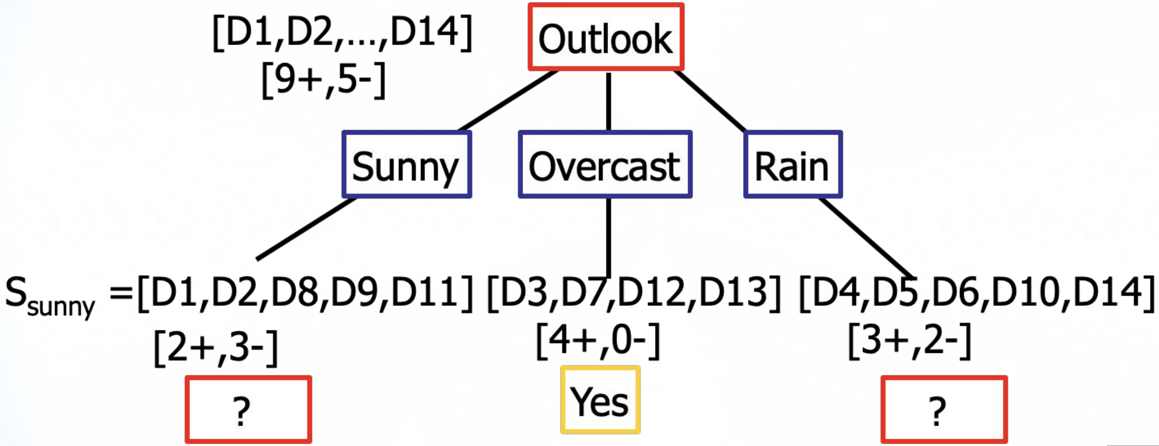
where S denotes the collection of training examples

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Y-DATA
SCHOOL OF DATA SCIENCE

Let's Continue to the Leftmost Node: Sunny

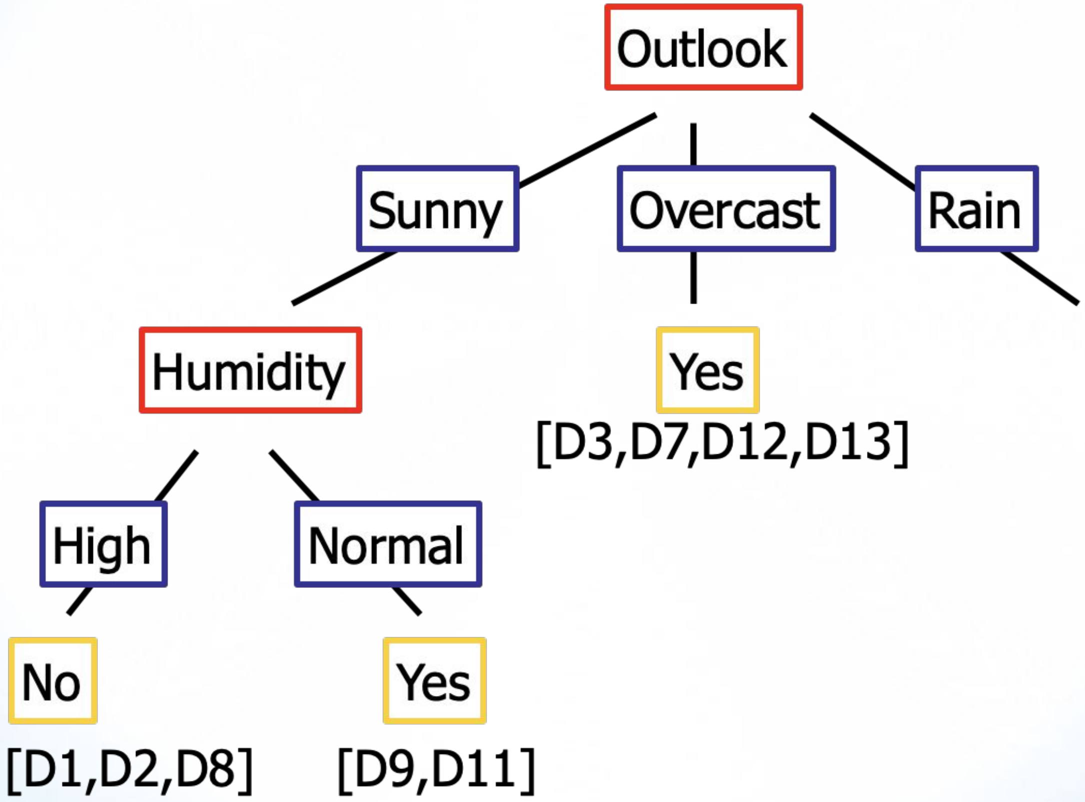


Gain(Sunny, Humidity)=0.970-(3/5)0.0 – 2/5(0.0) = 0.970

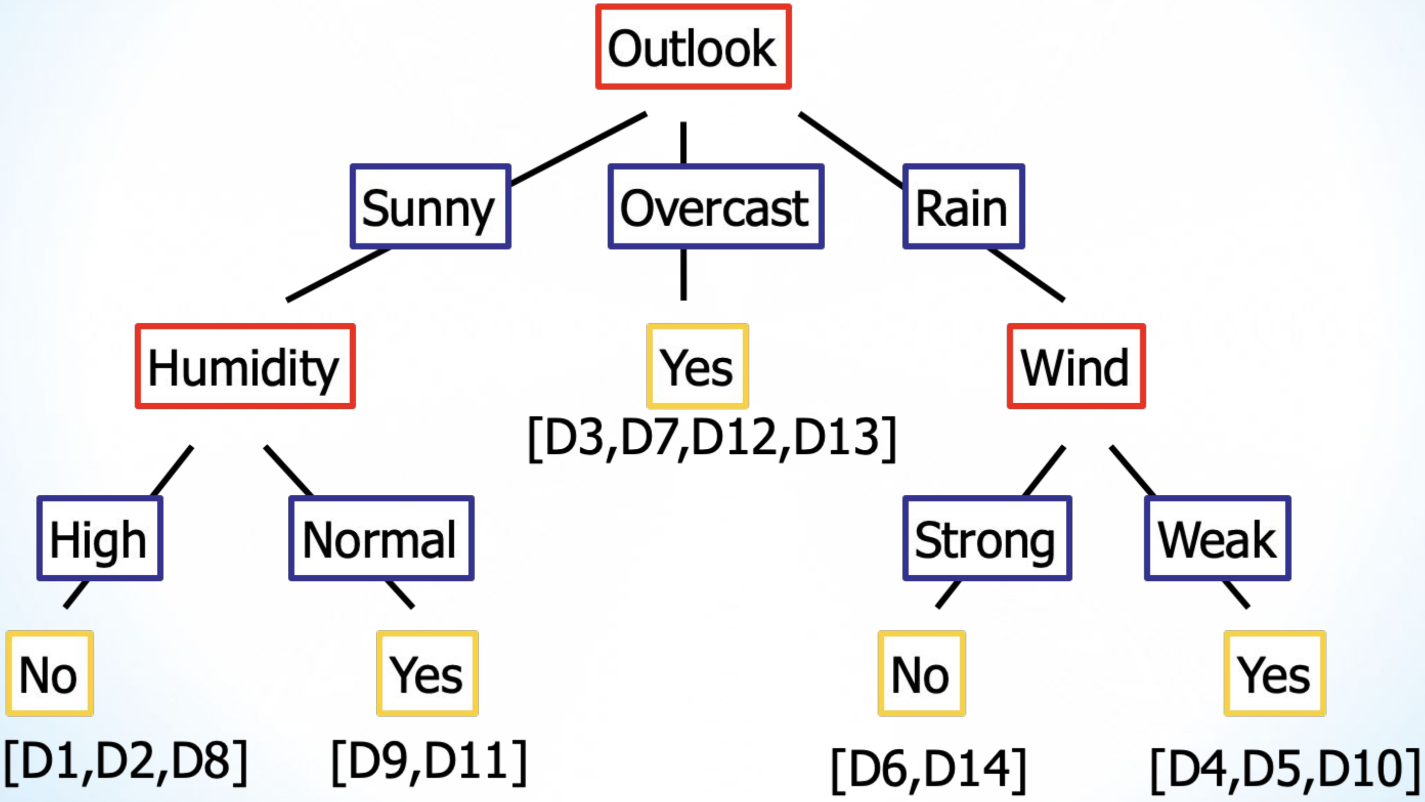
Gain(Sunny, Temp.)=0.970-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570

Gain(Sunny, Wind)=0.970= -(2/5)1.0 – 3/5(0.918) = 0.019

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No


Outlook
Sunny
Overcast
Rain
Humidity
Yes
[D3,D7,D12,D13]
High
Normal
No
Yes
[D1,D2,D8]
[D9,D11]

Y-DATA
SCHOOL OF DATA SCIENCE



Y-DATA
SCHOOL OF DATA SCIENCE

Decision Tree Algorithm

function DTL(*examples*, *attributes*, *default*) **returns** a decision tree

  if *examples* is empty **then return** *default*
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** MODE(*examples*)
  else
    best ← CHOOSE-ATTRIBUTE(*attributes*, *examples*)
    tree ← a new decision tree with root test *best*
    for each value v_i of *best* **do**
      $examples_i$ ← {elements of *examples* with $best = v_i$}
      subtree ← DTL($examples_i$, *attributes* − *best*, MODE(*examples*))
      add a branch to *tree* with label v_i and subtree *subtree*
    return *tree*

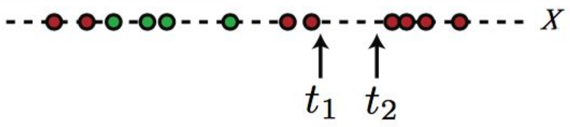
Handling split on Continuous Input

- Binary tree, split on attribute X
  - One branch: $X < t$
  - Other branch: $X \geq t$

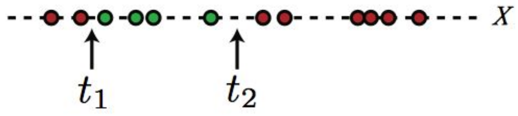
- Search through possible values of t… hard

Handling Splits on Continuous Input

- But only a finite number of t's are important



- Moreover, only splits between examples of different classes matter!
    - Sort data according to X into $\{x_1, \ldots, x_m\}$
    - Consider split points of the form $\frac{1}{2}(x_i + x_{i+1})$, only if $y_i \neq y_{i+1}$



(Figures from Stuart Russell)

Other Split Metrics

Classification only:

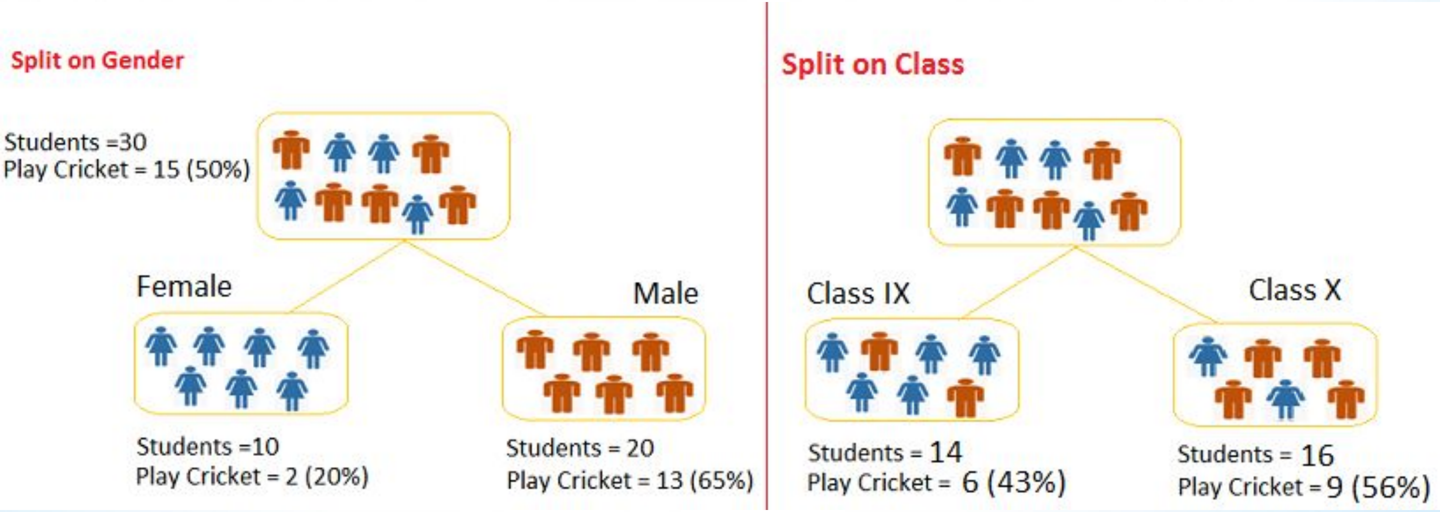
- **Information gain**
- **Gini index**

Regression and classification:

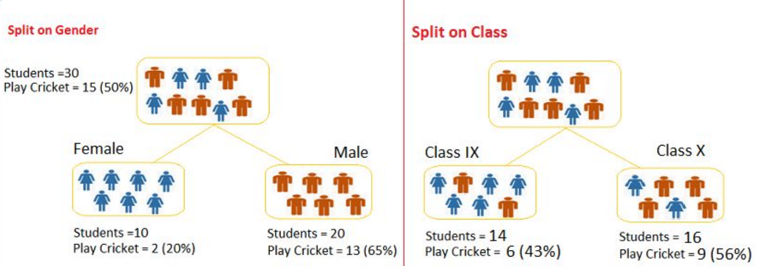
- **Reduction in variance**
- **Train error minimization** (mean squared or absolute error)

Example : Computing 4 different Gain functions

- Goal is to build a DT that predicts which student plays cricket
- Input with 2 attributes: gender, and class



Information Gain



- **Entropy for parent node** = -(15/30) log(15/30) – (15/30) log(15/30) = **1**

- Entropy for Female node = -(2/10) log(2/10) – (8/10) log(8/10) = 0.72
  Entropy for Male node = -(13/20) log(13/20) – (7/20) log(7/20) = 0.93
  Entropy for split Gender = Weighted entropy of sub-nodes
  = (10/30)*0.72 + (20/30)*0.93 = **0.86**

- Entropy for Class IX node = -(6/14) log(6/14) – (8/14) log(8/14) = 0.99
  Entropy for Class X node = -(9/16) log (9/16) – (7/16) log(7/16) = 0.99
  Entropy for split Class = (14/30)*0.99 + (16/30)*0.99 = **0.99**

$$H(X_m) = -\sum_k p_{mk} \log(p_{mk})$$

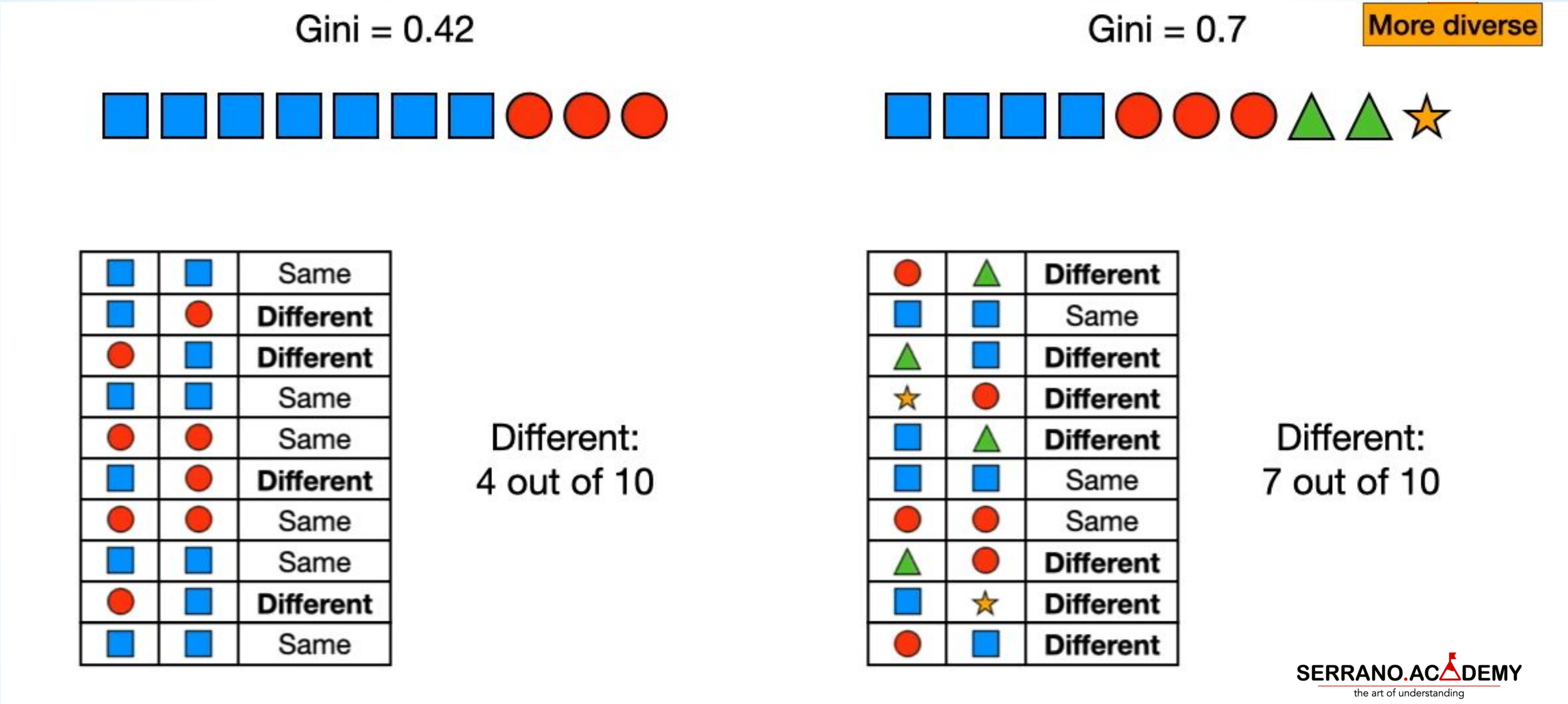
Gini Impurity

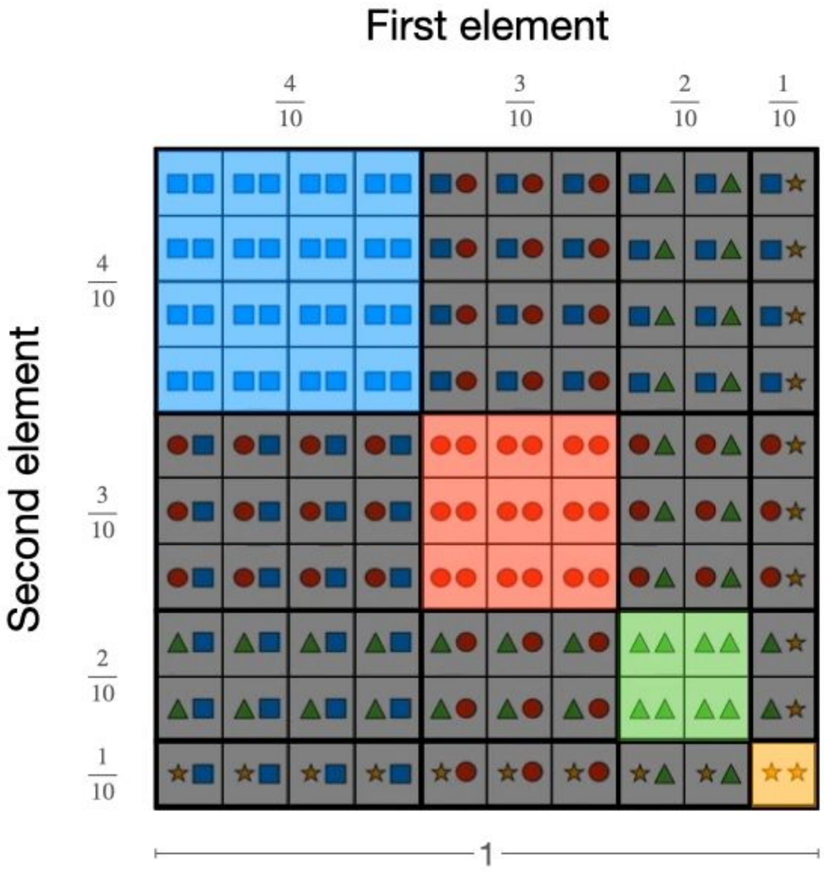
- Measures probability of picking two distinct elements

$$G = \sum_{i=1}^{C} p(i) * (1 - p(i)) = 1 - \sum_{i=1}^{C} (p_i)^2$$

We want to **Minimize Gini Impurity**

Intuition Behind the Math



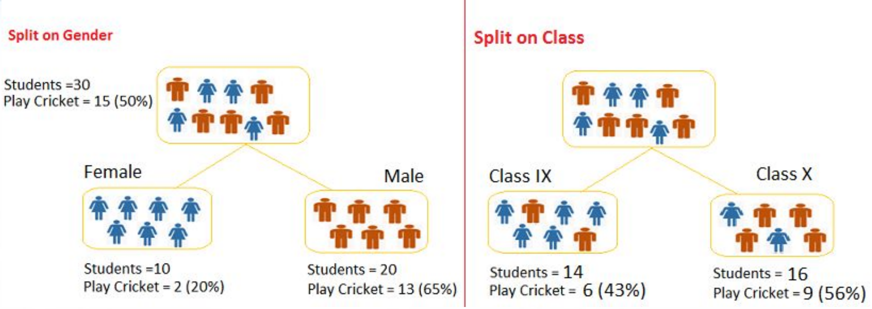


P(Both different) = P(Anything) - P(Both equal)

= 1 - P(Both blue) - P(Both red) - P(Both green) - P(Both yellow)

Y-DATA
SCHOOL OF DATA SCIENCE

Gini



Split on Gender

Calculate, Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68

Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55

Calculate weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 = **0.59**

Split on Class:

Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51

Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51

Calculate weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = **0.51**

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk}) = 1 - \sum_k p_{mk}^2$$

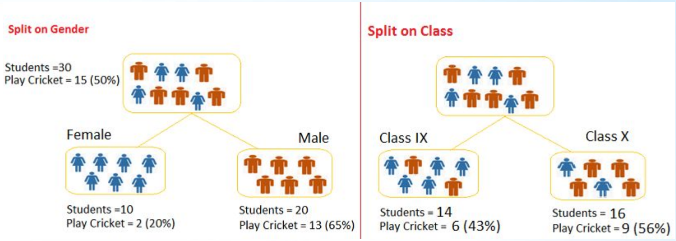
Reduction in variance

- Used for continuous target variables (regression problems) and classification
- Objective: Minimize variance of the target variable *x* due to the split at this node
- Uses usual variance formula: We want to **minimize the variance**

$$\text{Variance} = \frac{\Sigma(X - \overline{X})^2}{n}$$

- Use the mean target value as the predicted label

Reduction in Variance



Root node mean = (15*1 + 15*0)/30 = 0.5, variance = (15*(1-0.5)^2+15*(0-0.5)^2)/30 = **0.25**

Female node mean = (2*1+8*0)/10=0.2, variance = (2*(1-0.2)^2+8*(0-0.2)^2)/10 = 0.16

Male node mean = (13*1+7*0)/20=0.65, variance = (13*(1-0.65)^2+7*(0-0.65)^2)/20 = 0.23

Variance for split Gender = Weighted Variance of Sub-nodes

= (10/30)*0.16 + (20/30) *0.23 = **0.21**

Class IX node mean = (6*1+8*0)/14=0.43, variance = (6*(1-0.43)^2+8*(0-0.43)^2)/14= 0.24

Class X node mean = (9*1+7*0)/16=0.56, variance = (9*(1-0.56)^2+7*(0-0.56)^2)/16 = 0.25

Variance for split Class = (14/30)*0.24 + (16/30) *0.25 = **0.25**

$$\text{Variance} = \frac{\Sigma(X-\overline{X})^2}{n}$$

Train Error Minimization

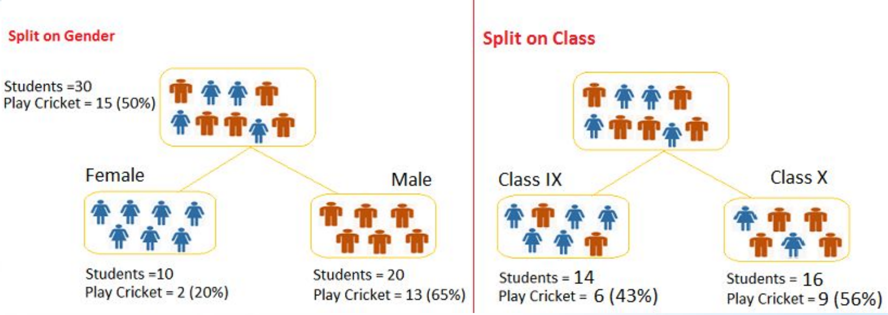
- Choose the split which reduces the train error assuming the nodes after the split are leafs
- For classification we can look at the misclassification rate
- For regression we can look at mean squared (MSE) or absolute error (MAE)

.

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - \bar{y}_m)^2$$

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} |y_i - \bar{y}_m|$$

Train Error Minimization



Root node error = 15/30 = **0.5**

Female node error = 2/10 = 0.2

Male node error = 7/20 = 0.35

Error for split Gender = Weighted Error of Sub-nodes

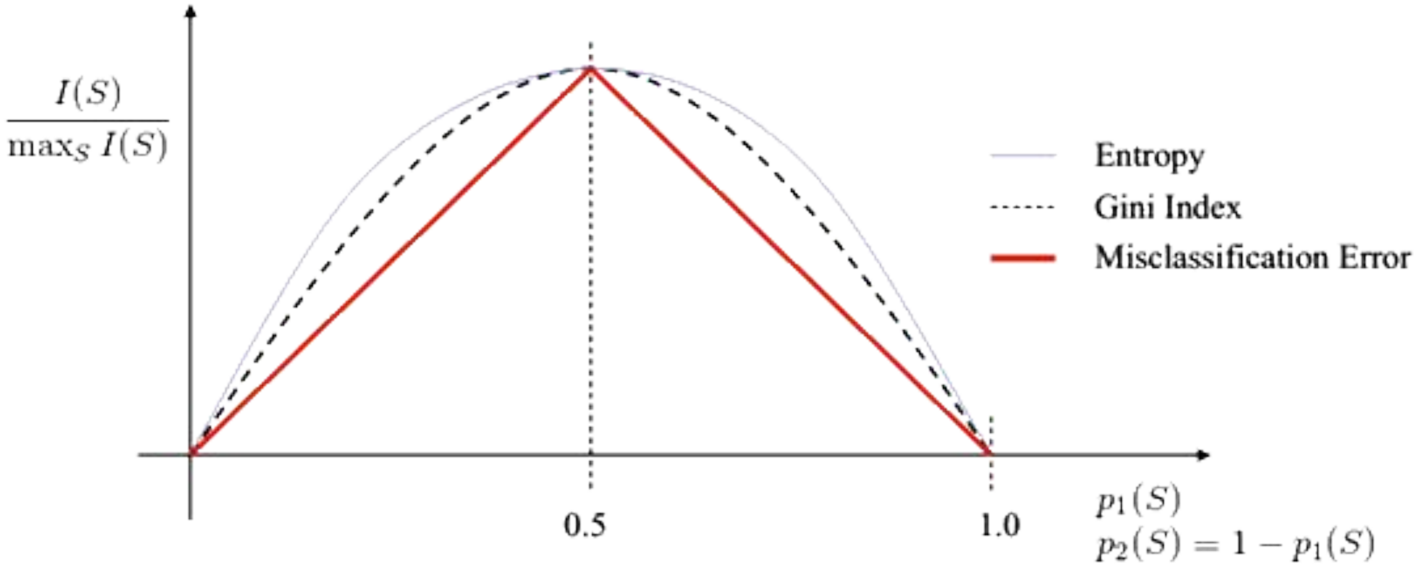
$$\boxed{= (10/30)*0.2 + (20/30)*0.35 = \mathbf{0.3}}$$

Class IX node error = 6/14 = 0.43

Class X node error = 7/16 = 0.44

Error for split Class = (14/30)*0.43 + (16/30) *0.44 = **0.435**

Comparison of Gain functions



Handling Missing Data

- In training
  - Ignore samples with missing values when computing the gain (but use them for other features)
  - Replace missing values with “?”
- In prediction
  - Get a final prediction probability for each class in each possible path, and create a final probability estimate for each class using a weighted sum of the different predictions

Rectilinear Decision Boundaries

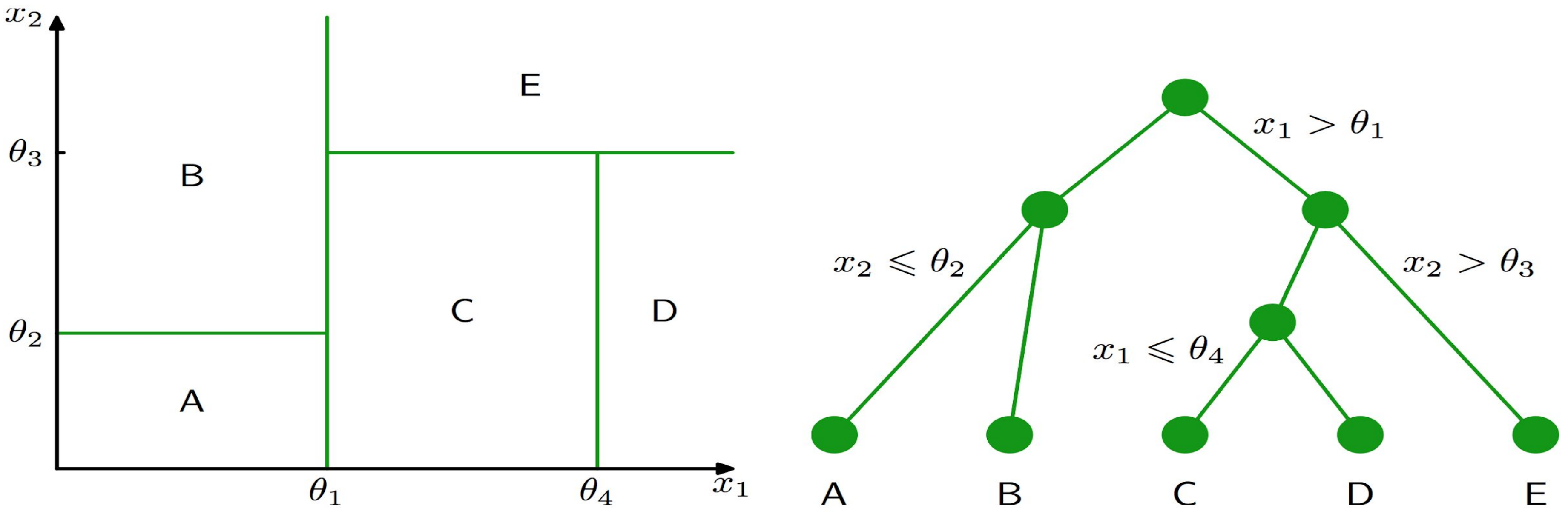


Figure credit: Chris Bishop, PRML

Which Size of Tree Would You Choose?

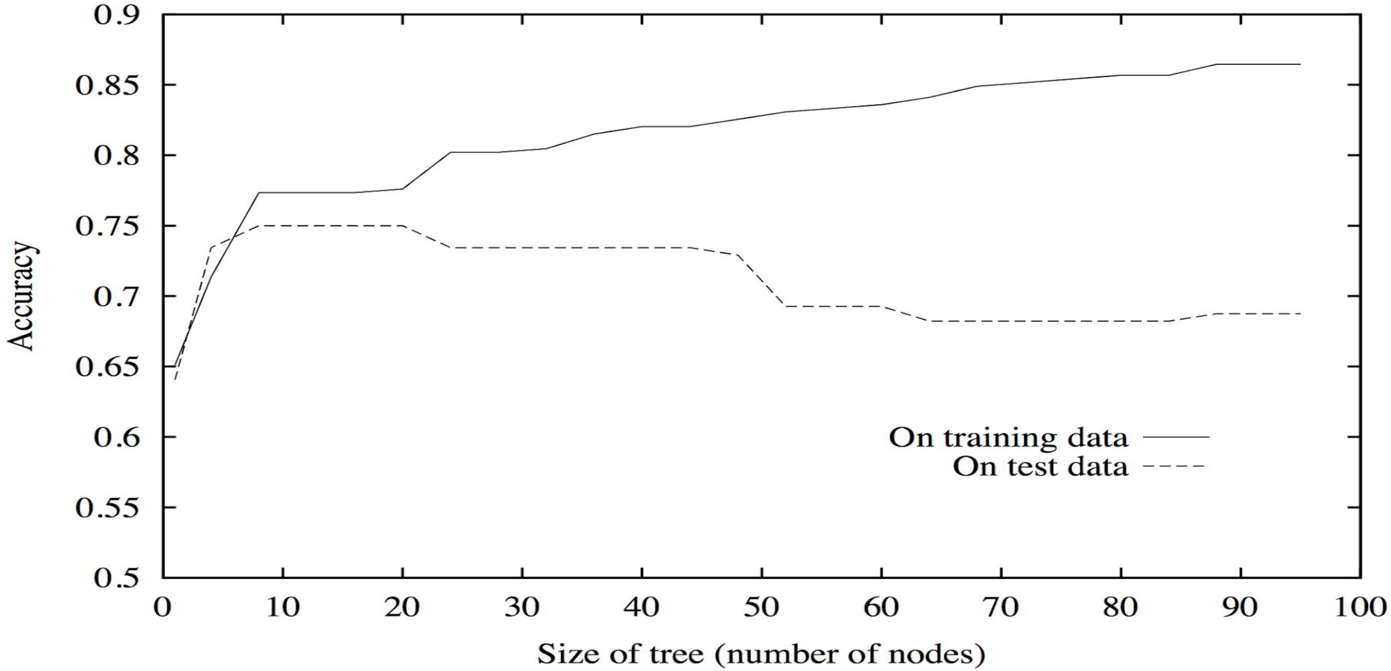


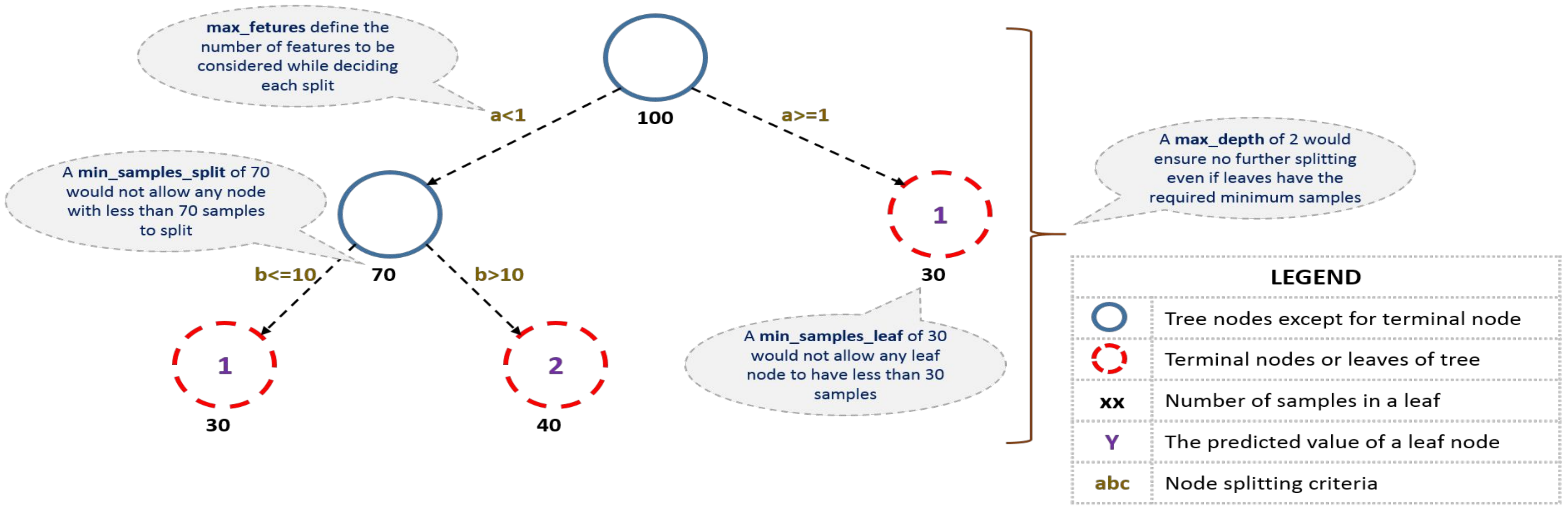
Figure credit: Tom Mitchell, 1997

Trees Can Fit Anything! Handling Overfitting

- Overfitting is key challenge
- Trees can reach 100% accuracy on training by making 1 leaf for each observation.
- 2 common ways to prevent overfitting in decision trees
  - Pre-pruning : Setting constraints on tree size / Stop when the information gain is small
  - Post-pruning : Tree pruning (applied post-building)

Y-DATA
SCHOOL OF DATA SCIENCE

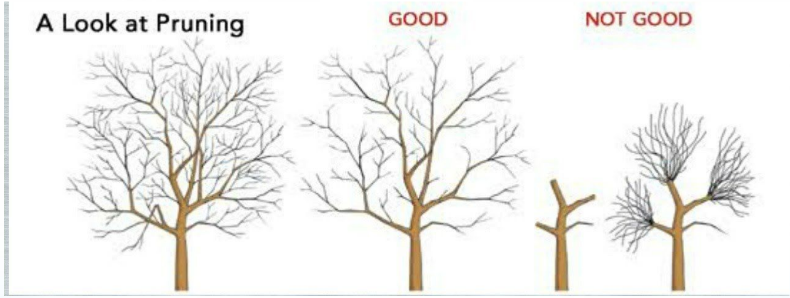
Pre-pruning: Constraints on Tree Size



Y-DATA
SCHOOL OF DATA SCIENCE

Post-Pruning

- Pruning is reducing the tree size after it has been built
- Pruning should reduce tree size without reducing predictive accuracy as measured by a cross-validation set



Y-DATA
SCHOOL OF DATA SCIENCE

Pruning Approaches

Cost-Complexity Pruning

Try to reduce the complexity of the tree (number of leaves) while not harming the accuracy too much (using some regularization coefficient)

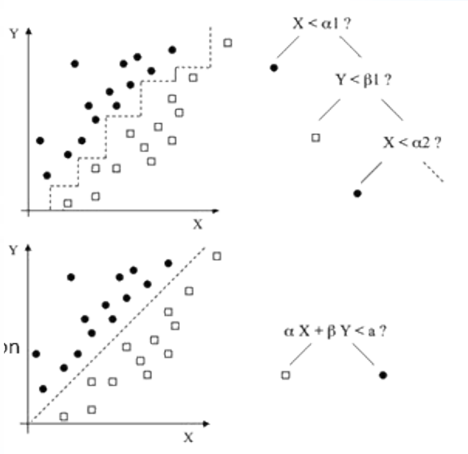
- $R_\alpha(T) = R(T) + \alpha \cdot |f(T)|$ where
  - $R(T)$ is the training/learning error
  - $f(T)$ a function that returns the set of leaves of tree T

Reduced Error Pruning

1. First make a decision tree to a large depth
2. Starting from the leaves, remove subtrees and replace their root node with its most popular class - keep the change if the prediction accuracy <u>on a validation set</u> is not harmed (or not harmed too much)
3. Continue until all sub-tree removals are harmful

Rectilinear Decision Boundaries

- May not be good model for some problems
- In such cases, it may be better to use other models such as Logistic Regression or Neural Networks

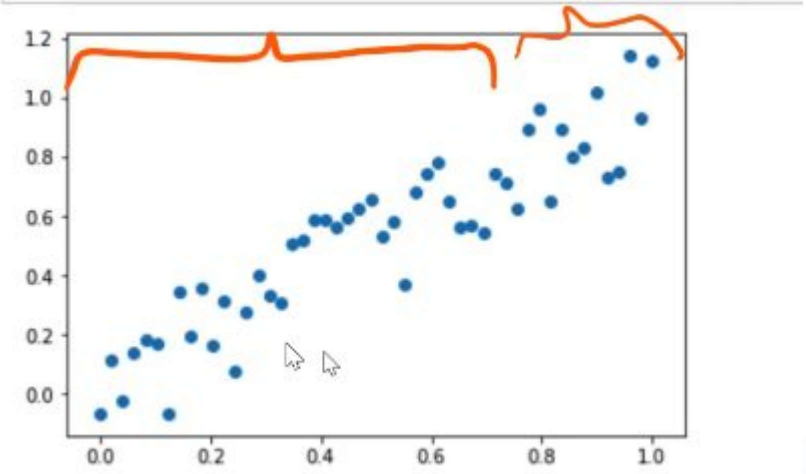


Extrapolation is a problem

What would you expect to see in the following case?

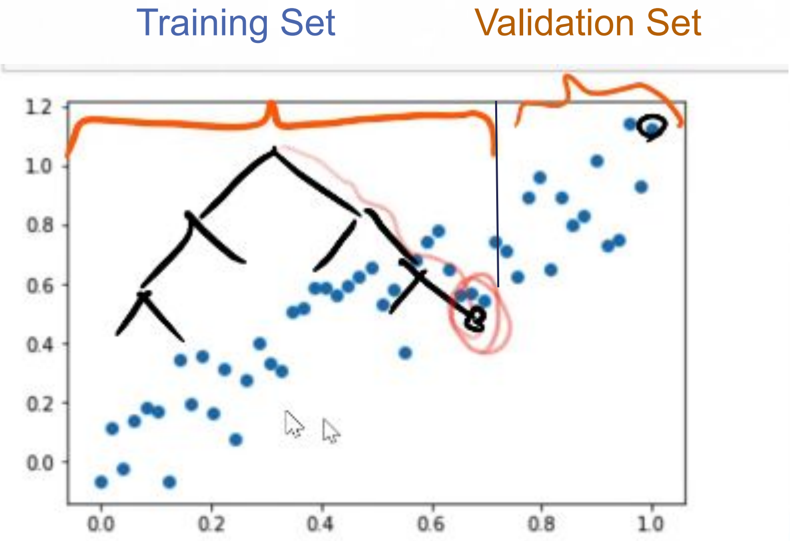
Training Set

Validation Set



Extrapolation is a problem

What would you expect to see in the following case?



Code

Let’s Think About This Together

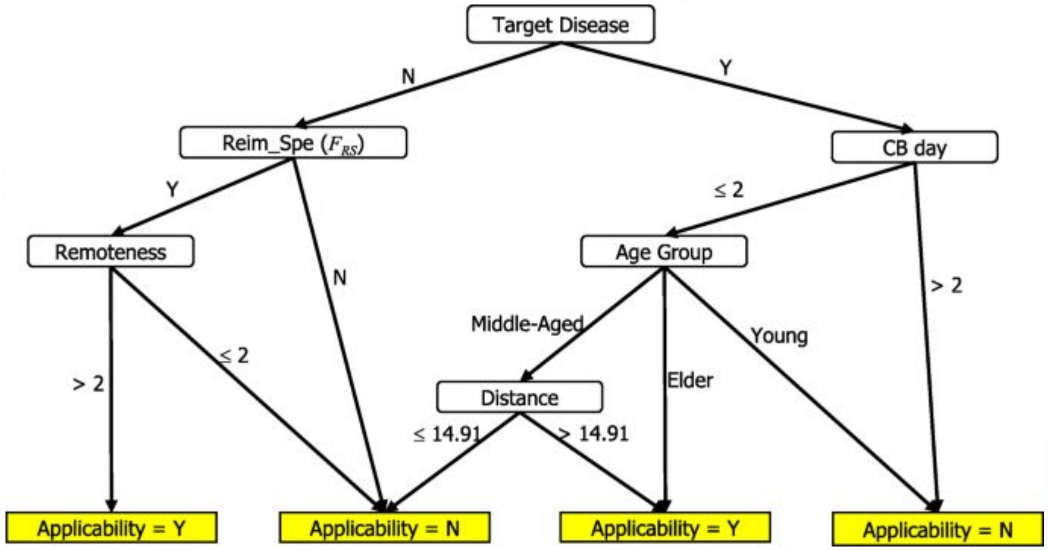
1. What are the main hyper-parameters?
2. Can it work for Multi-class data (relevant only for logistic)?
3. How does it handle categorical data?
4. How does it handle missing data?
5. Is it sensitive to outliers?
6. What if some features are correlated?
7. Is it prone to overfitting?
8. Is it interpretable?
9. Can it be parallelized?
10. Speed of training
11. Speed of prediction

Let’s Think About This Together

1. What are the main hyper-parameters? Split measure, depth, max features, min split, min leaf
2. Can it work for Multi-class data? Yes
3. How does it handle categorical data? Continuous we find best split.
4. How does it handle missing data? In train ignores, in test averages branches
5. Is it sensitive to outliers? No
6. What if some features are correlated? Handles well, will not pick feature for next split
7. Is it prone to overfitting? Yes
8. Is it interpretable? Yes
9. Can it be parallelized? No
10. Speed of training - Average
11. Speed of prediction - Fast

Example Information Theory: Wordle

Example: Patient Telehealth



Y-DATA
SCHOOL OF DATA SCIENCE

Summary

Summary

- Is an interpretable model - use it for EDA
- Can be used for classification and regression
- Is very prone to overfitting… you'll discuss the concept of random forest in the "ensemble" lecture which uses the advantage of trees without having a strong overfit

Decision Tree Pros & Cons

Pros	Cons
1. Simple 2. Explainable 3. Handles Categorical Data 4. Fast prediction 5. We can create rules based on trees	1. Tend to overfit the training data, which is solved by setting constraints on the model and pruning 2. Decision boundaries are rectilinear 3. Loses information when categorizing continuous input variables into different categories 4. In regression, it cannot predict beyond the range in the training data (bad extrapolation)