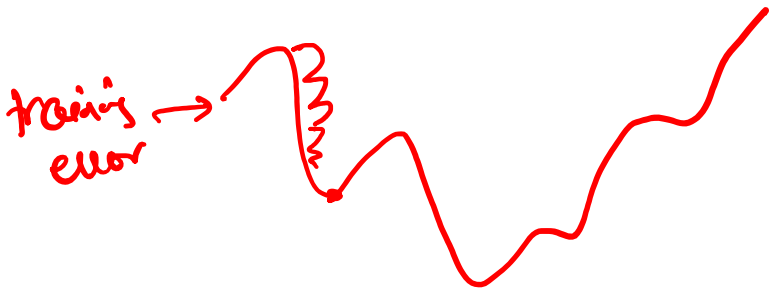
Tips and Tricks for training deep NNs

- First hypothesis (underfitting): better optimize
  - ➢ Increase the capacity of the neural network
  - ➢ Check initialization
  - ➢ Check gradients (saturating units and vanishing gradients)
  - ➢ Tune learning rate

- Second hypothesis (overfitting): use better regularization
  - ➢ Dropout
  - ➢ Data augmentation
  - ➢ Early stopping
  - ➢ Architecture search



- For many large-scale practical problems, you will need to use both: better optimization and better regularization!

Other optimization tips and tricks

- **Momentum:** use exponentially weighted sum of previous gradients

$$\overline{\nabla}_{\boldsymbol{\theta}}^{(t)} = \nabla_{\boldsymbol{\theta}} l(\mathbf{f}(\mathbf{x}^{(t)}), y^{(t)}) + \beta \overline{\nabla}_{\boldsymbol{\theta}}^{(t-1)}$$

can get pass plateaus more quickly, by "gaining momentum"

- **Initialization:** cannot initialize to same value, all units in a hidden layer will behave same; randomly initialize unif[-b,b]

- **Adaptive learning rates:** one learning rate per parameter

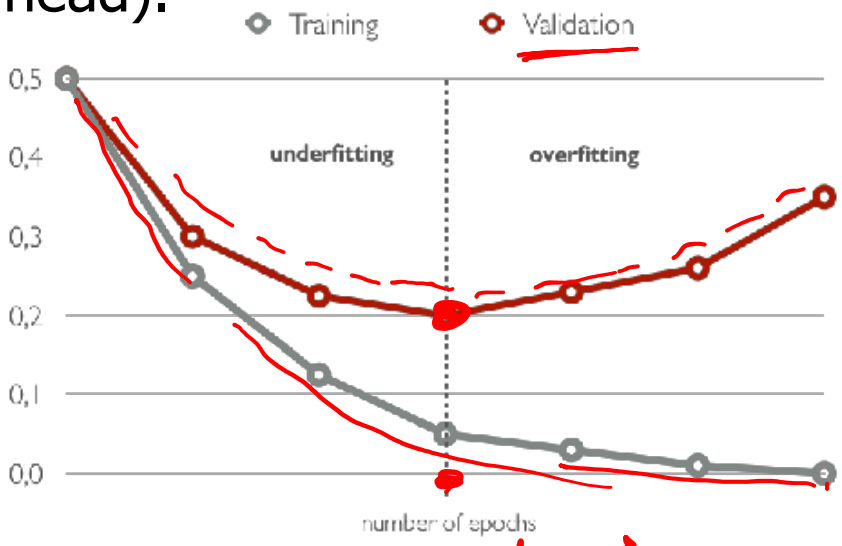
e.g. RMSProp uses exponentially weighted average of squared gradients

$$\gamma^{(t)} = \beta \gamma^{(t-1)} + (1-\beta) \left(\nabla_{\theta} l(\mathbf{f}(\mathbf{x}^{(t)}), y^{(t)}) \right)^2 \qquad \overline{\nabla}_{\theta}^{(t)} = \frac{\nabla_{\theta} l(\mathbf{f}(\mathbf{x}^{(t)}), y^{(t)})}{\sqrt{\gamma^{(t)} + \epsilon}}$$

Adam combines RMSProp with momentum

Tips and tricks for preventing overfitting

- **Dropout**
- **Data augmentation**
- **Early stopping:** stop training when validation set error increases (with some look ahead).



- **Neural Architecture search:** tune number of layers and neurons per layer using grid search or clever optimization

Classification:

p(Y|X)

X → Y

→ p(X)

→ p(X|Y)

D/ Logistic regression

G/ Bayes optimal classifier

G/ Gaussian Bayes classifier ← MLE, MAP

D/ Neural Nets, CNN,

G/ Naive Bayes

Decision Trees

Aarti Singh

Machine Learning 10-315

Oct 6 , 2021

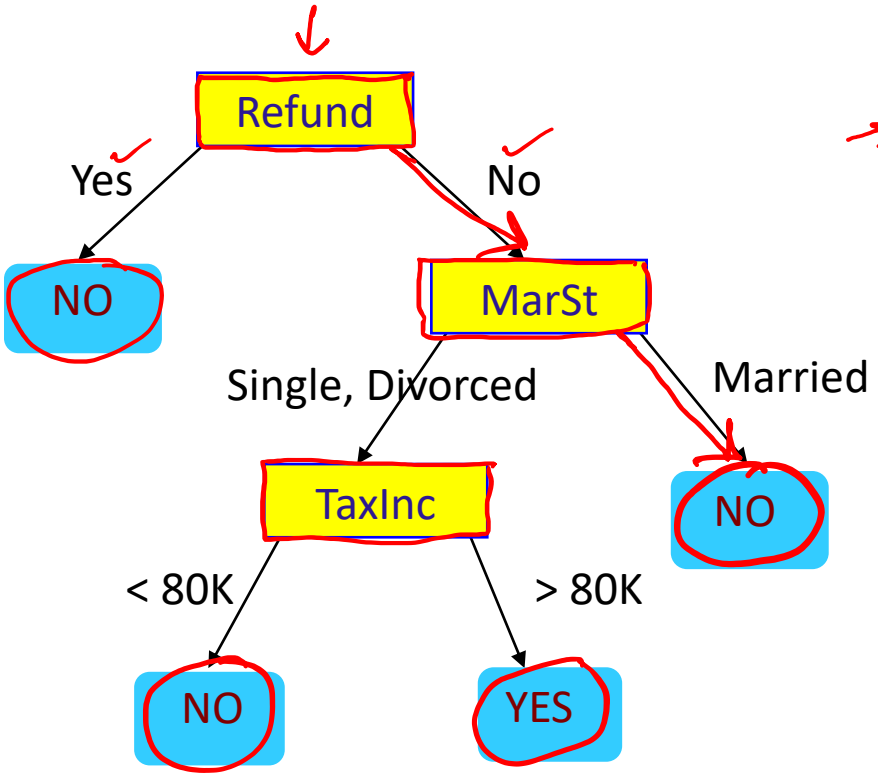
Decision Trees

- Start with discrete features, then discuss continuous

Representation

- What does a decision tree represent

Decision Tree for Tax Fraud Detection



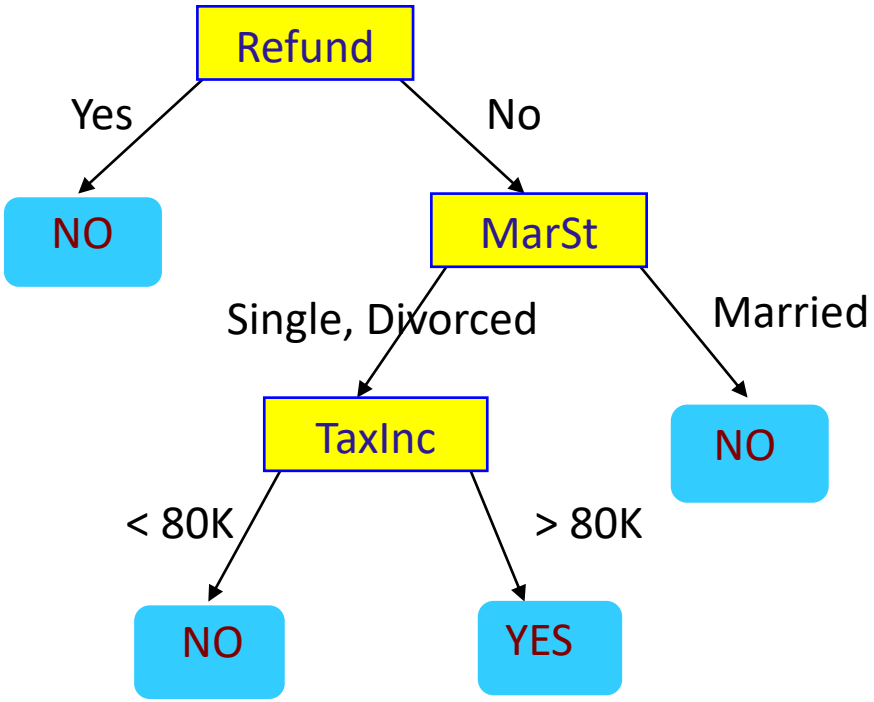
X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Mar	<80K	No

- Each internal node: test one feature X_i
- Each branch from a node: selects some value for X_i
- Each leaf node: prediction for Y

Prediction

- Given a decision tree, how do we assign label to a test point

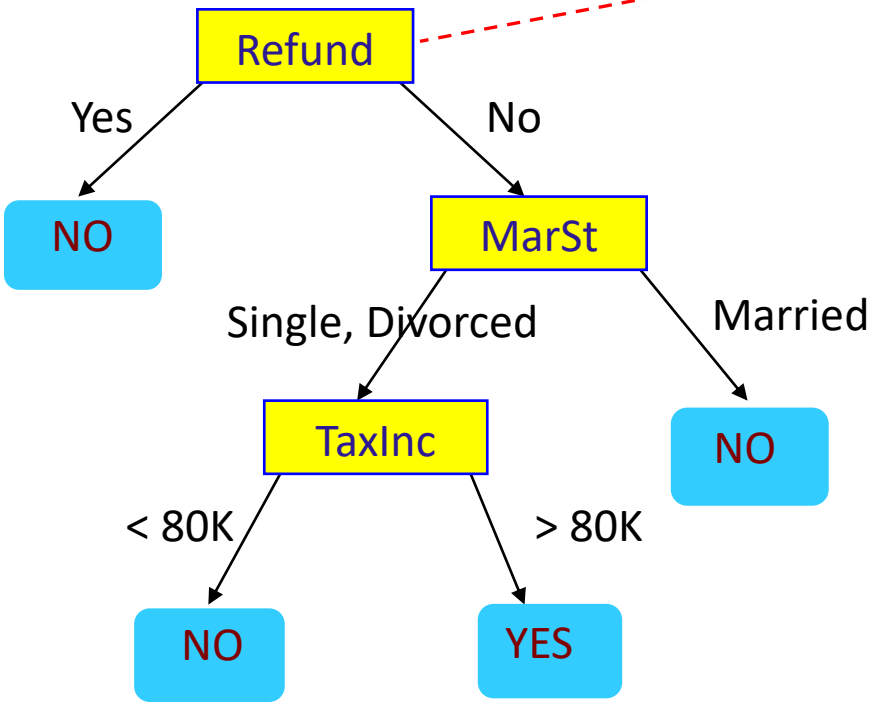
Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

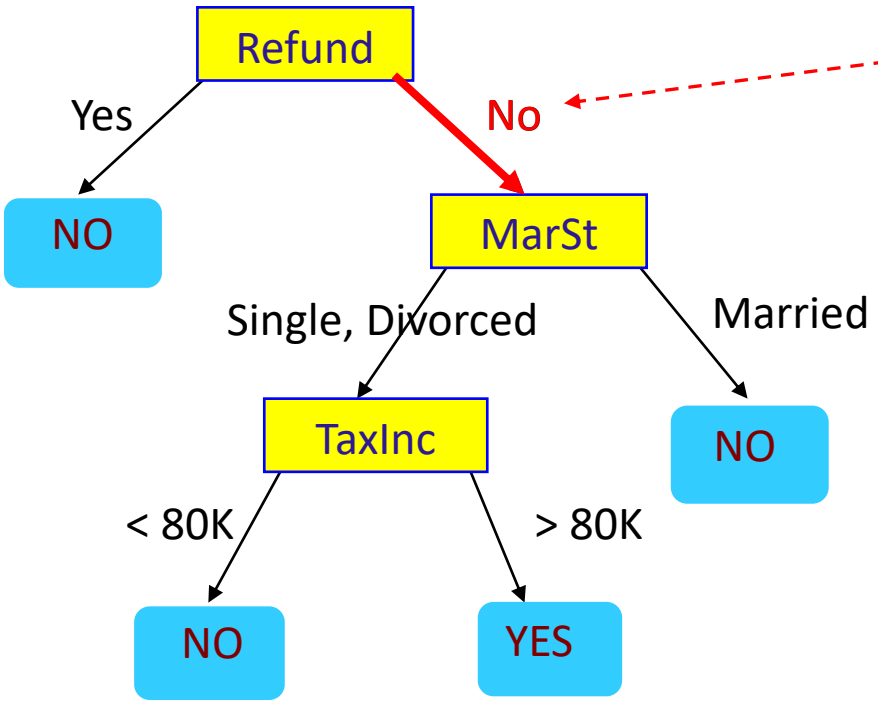
Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

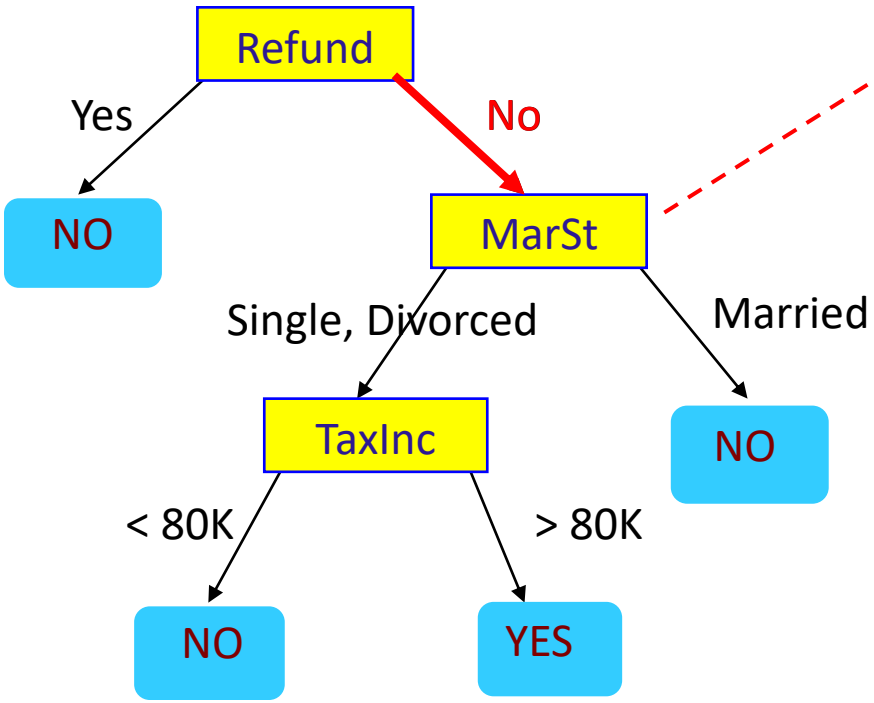
Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

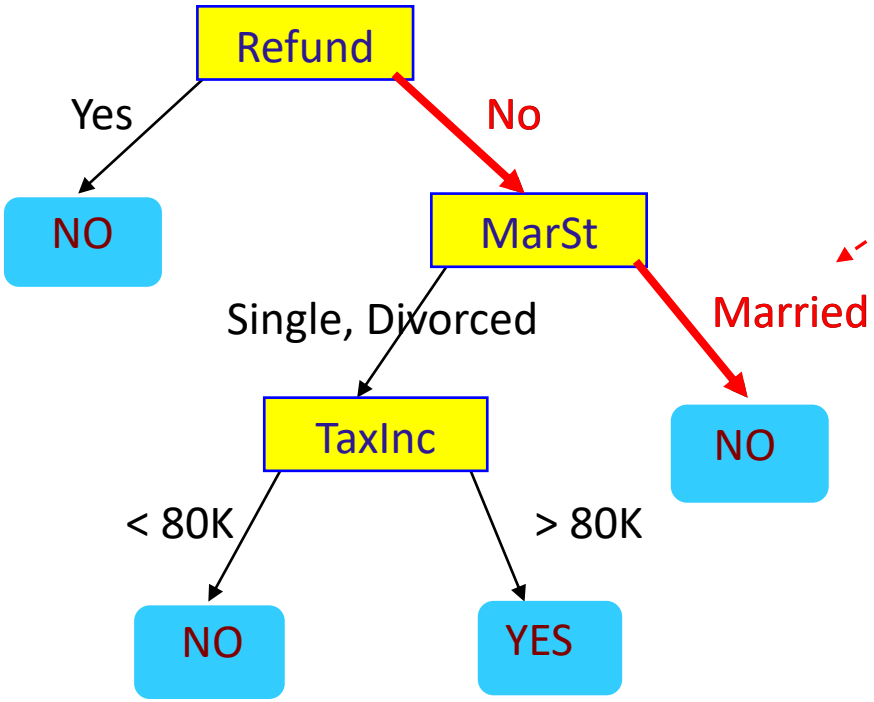
Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

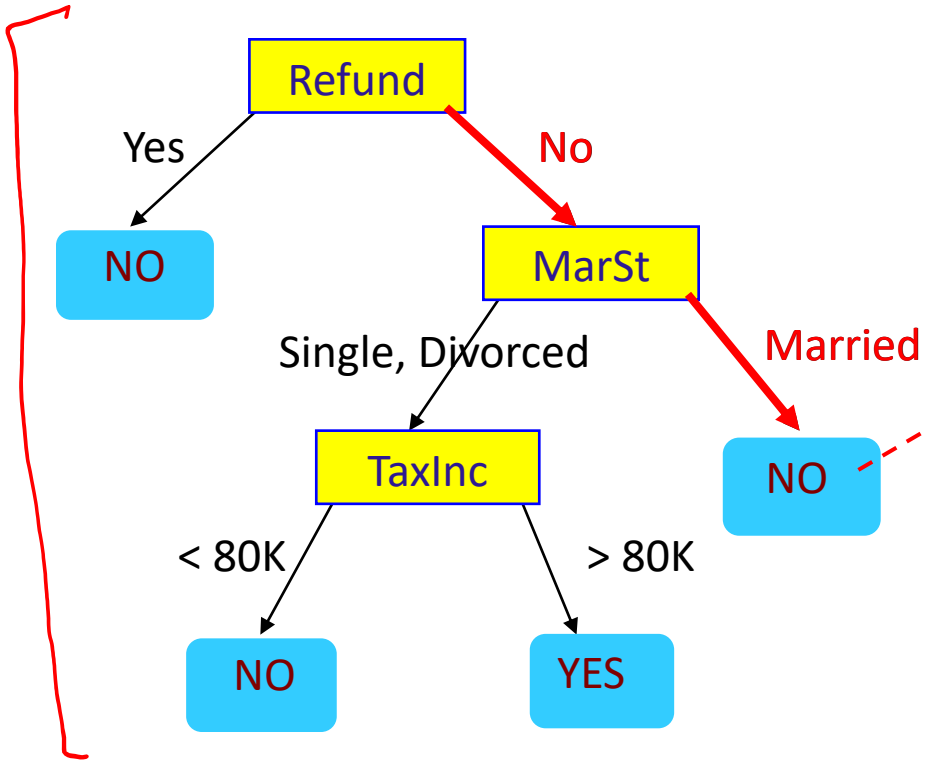
Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

Decision Tree for Tax Fraud Detection



Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

Assign Cheat to “No”

So far…

- What does a decision tree represent
- Given a decision tree, how do we assign label to a test point

Discriminative or Generative?

Now …

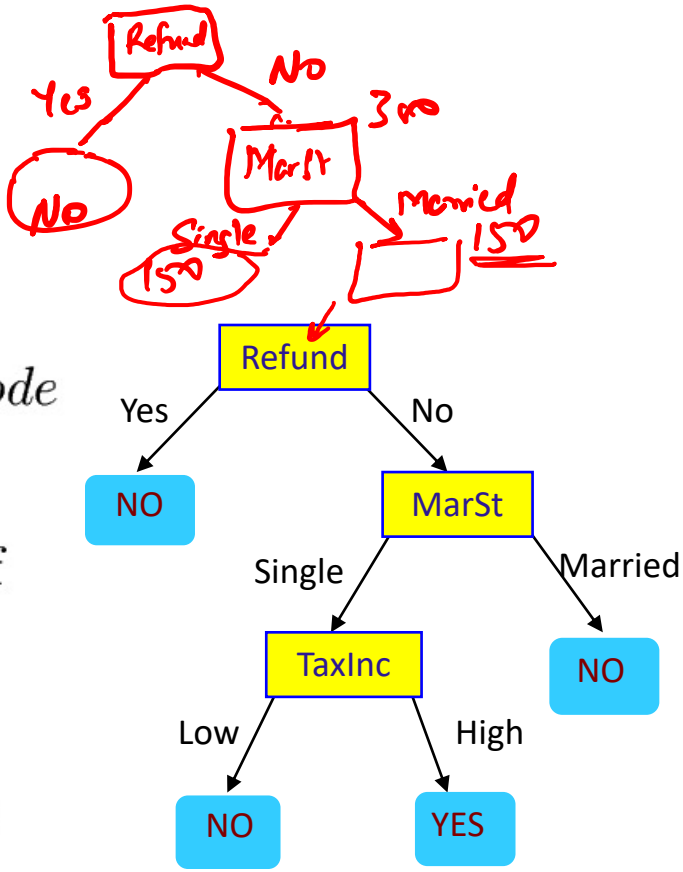
- How do we learn a decision tree from training data

How to learn a decision tree

- Top-down induction [ID3]

Main loop:

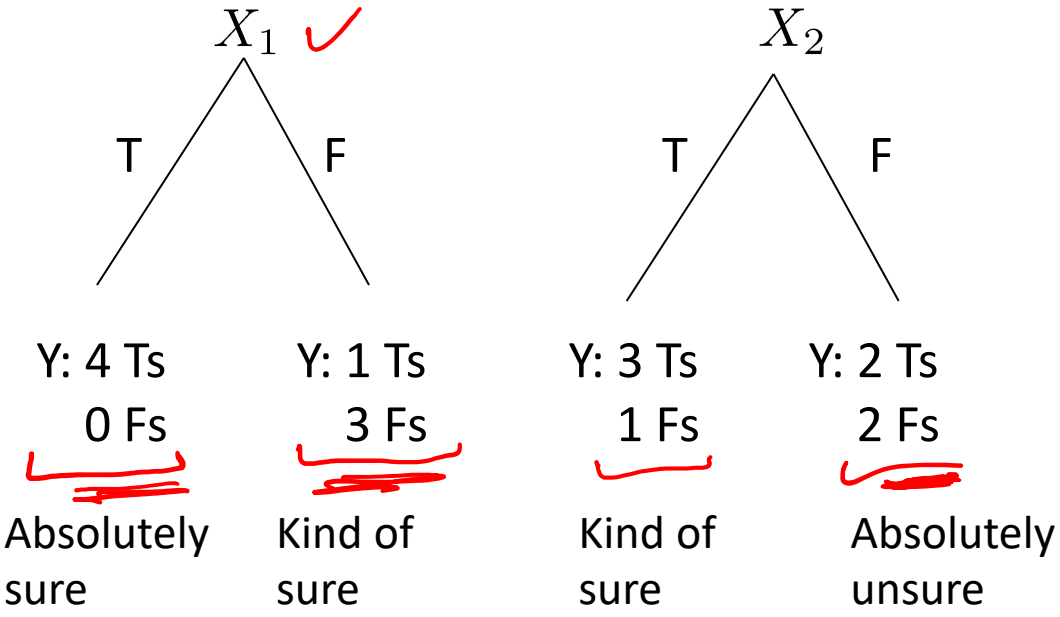
1. $X \leftarrow$ the "best" decision feature for next *node*
2. Assign X as decision feature for *node*
3. For each value of X, create new descendant of *node* (Discrete features)
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes
   (steps 1-5) after removing current feature



6. When all features exhausted, assign majority label to the leaf node

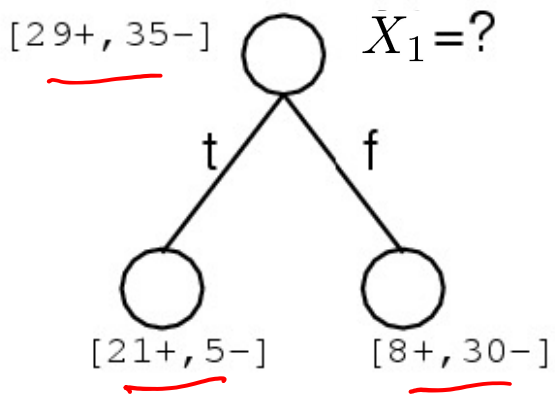
Which feature is best?

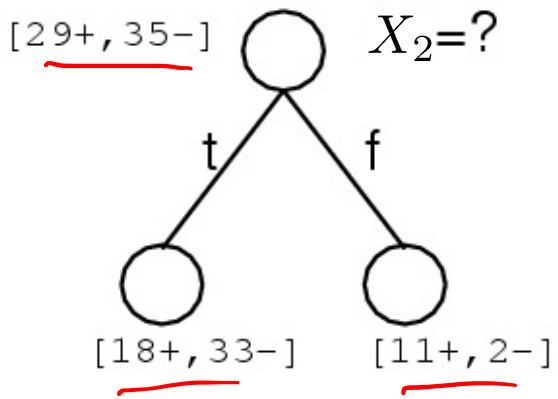
X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



Good split if we are more certain about classification after split –
Uniform distribution of labels is bad

Which feature is best?





Pick the attribute/feature which yields maximum information gain:

$$\arg\max_i I(Y, X_i) = \arg\max_i [H(Y) - H(Y|X_i)]$$

H(Y) – entropy of Y H(Y|X_i) – conditional entropy of Y

Andrew Moore's Entropy in a Nutshell

Low Entropy

..the values (locations of soup) sampled entirely from within the soup bowl

High Entropy

..the values (locations of soup) unpredictable... almost uniformly sampled throughout our dining room

Entropy

- Entropy of a random variable Y

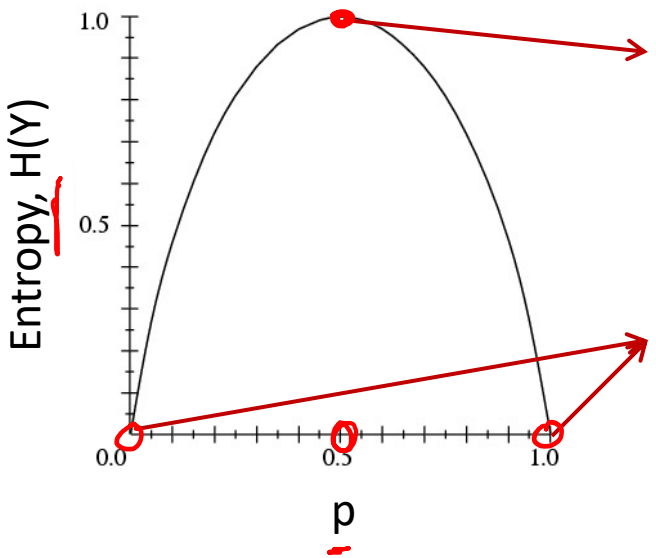
$$H(Y) = -\sum_{y} P(Y = y) \log_2 P(Y = y)$$

More uncertainty, more entropy!

Y ~ Bernoulli(p)

→ P(Y=1) = p

→ P(Y=0) = 1-p



Uniform

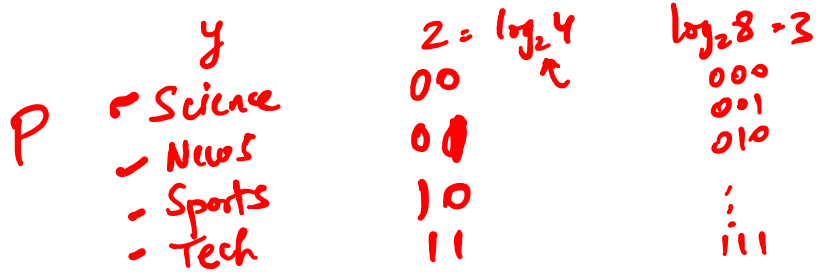
Max entropy

Deterministic

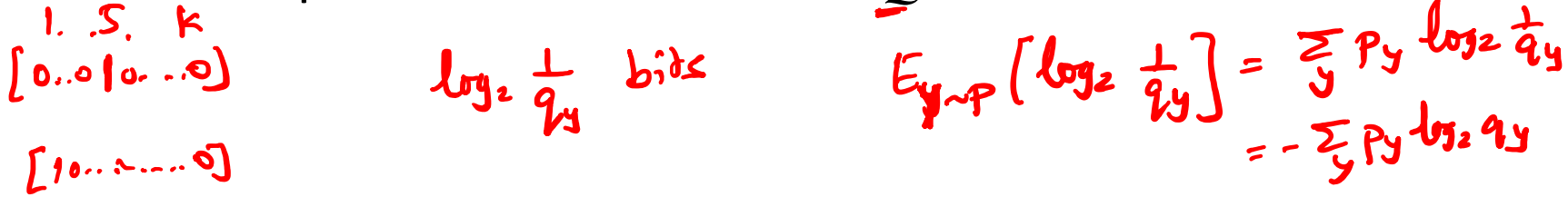
Zero entropy

Information Theory interpretation

Entropy: $H(Y) = H(P)$ is the expected number of bits needed to encode a randomly drawn value of $Y \sim P$ under most efficient code optimized for distribution P



Cross-Entropy: $H(P,Q)$ is the expected number of bits needed to encode a randomly drawn value of $Y \sim P$ under most efficient code optimized for distribution Q



Information Gain

- Advantage of attribute = decrease in uncertainty
  - Entropy of Y before split

$$H(Y) = -\sum_y P(Y = y) \log_2 P(Y = y)$$

  - Entropy of Y after splitting based on X_i
    - Weight by probability of following each branch

$$\begin{aligned} H(Y \mid X_i) &= \sum_x P(X_i = x) H(Y \mid X_i = x) \\ &= -\sum_x P(X_i = x) \sum_y P(Y = y \mid X_i = x) \log_2 P(Y = y \mid X_i = x) \end{aligned}$$

- Information gain is difference

$$I(Y, X_i) = H(Y) - H(Y \mid X_i)$$

Max Information gain = min conditional entropy

Which feature is best to split?

Pick the attribute/feature which yields maximum information gain:

$$\arg\max_i I(Y, X_i) = \arg\max_i [H(Y) - H(Y|X_i)]$$

$$= \arg\min_i H(Y|X_i)$$

Entropy of Y $\quad H(Y) = -\sum_y P(Y=y)\log_2 P(Y=y)$

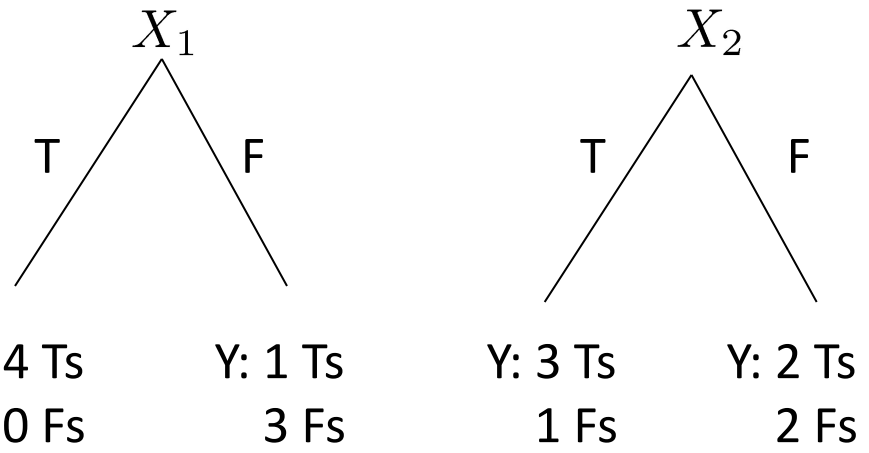
Conditional entropy of Y $\quad H(Y \mid X_i) = \sum_x P(X_i = x) H(Y \mid X_i = x)$

Feature which yields maximum reduction in entropy (uncertainty) provides maximum information about Y

Information Gain

$$H(Y \mid X_i) = -\sum_x P(X_i = x) \sum_y P(Y = y \mid X_i = x) \log_2 P(Y = y \mid X_i = x)$$

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



$$\widehat{H}(Y|X_1) = -\frac{1}{2}[1\log_2 1 + 0\log_2 0] - \frac{1}{2}[\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}]$$

$$\widehat{H}(Y|X_2) = -\frac{1}{2}[\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}] - \frac{1}{2}[\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}]$$

$$\widehat{H}(Y|X_1) < \widehat{H}(Y|X_2)$$