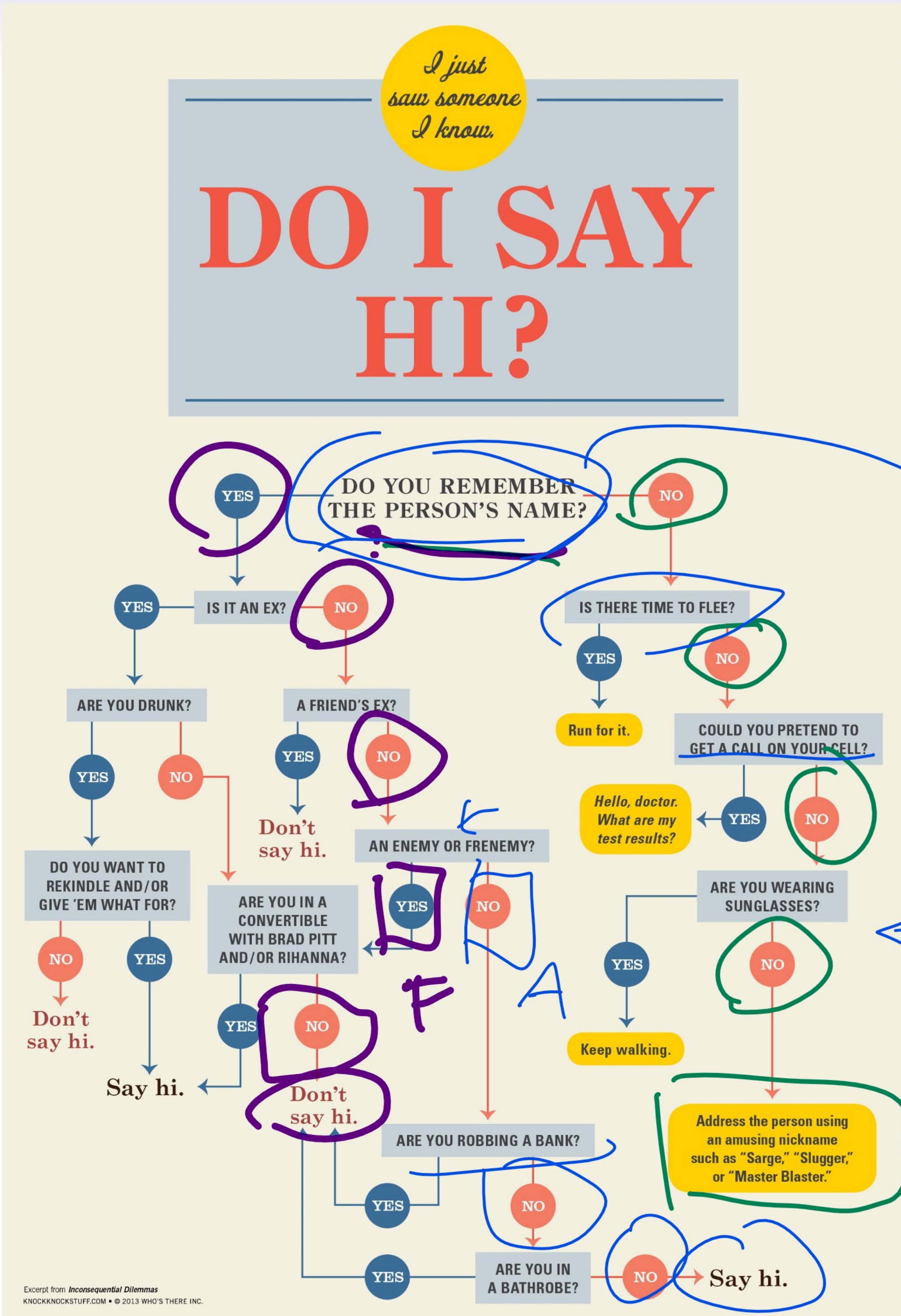
# CS540 Introduction to Artificial Intelligence
## Lecture 6

Young Wu
Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

May 26, 2020

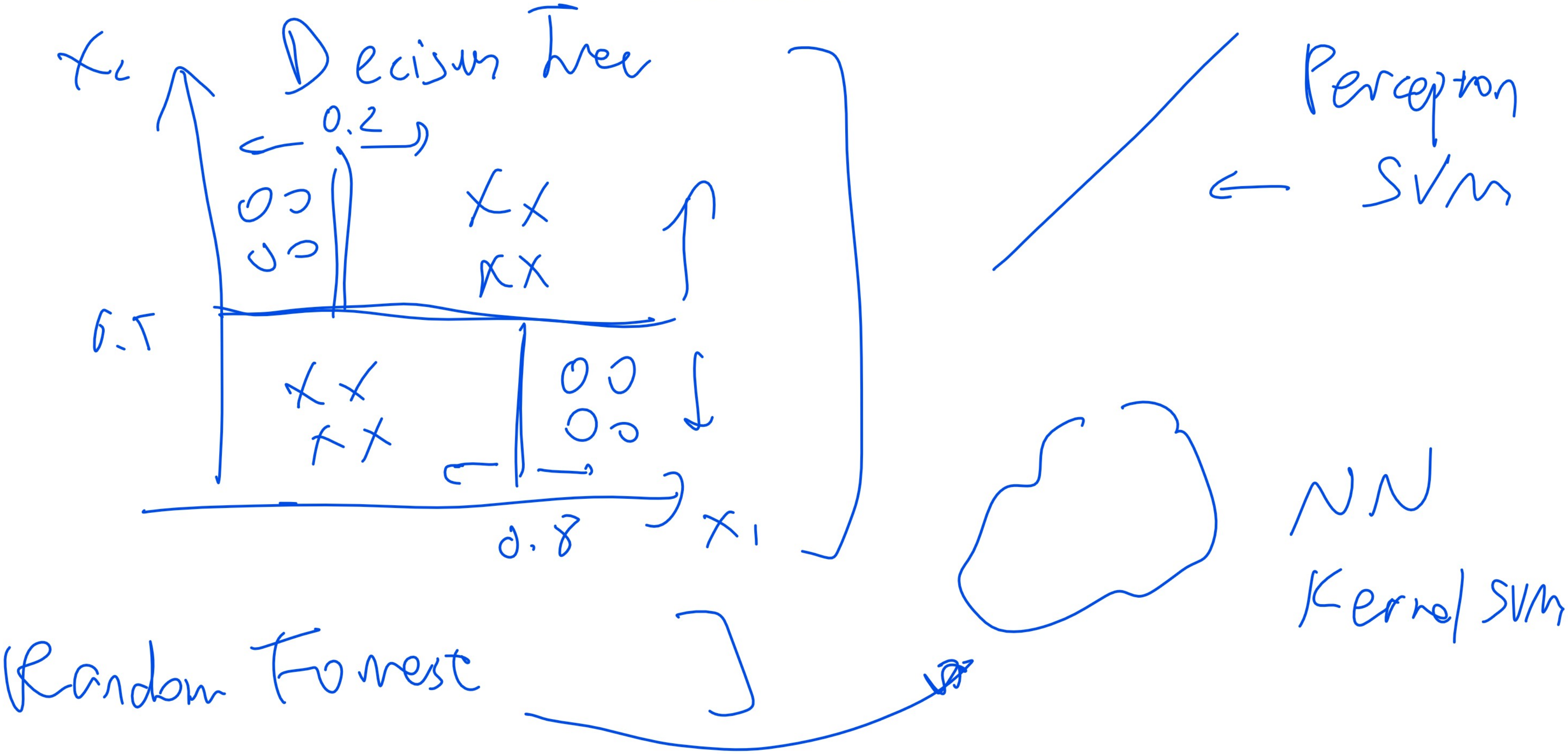
## ...on Tree

...tion


I just saw someone I know.
DO I SAY HI?
DO YOU REMEMBER THE PERSON'S NAME?
YES
NO
IS IT AN EX?
YES
NO
IS THERE TIME TO FLEE?
YES
NO
Run for it.
ARE YOU DRUNK?
YES
NO
A FRIEND'S EX?
YES
NO
Don't say hi.
COULD YOU PRETEND TO GET A CALL ON YOUR CELL?
Hello, doctor. What are my test results?
YES
NO
AN ENEMY OR FRENEMY?
YES
NO
DO YOU WANT TO REKINDLE AND/OR GIVE 'EM WHAT FOR?
NO
YES
Don't say hi.
Say hi.
ARE YOU IN A CONVERTIBLE WITH BRAD PITT AND/OR RIHANNA?
YES
NO
Don't say hi.
ARE YOU WEARING SUNGLASSES?
YES
NO
Keep walking.
Address the person using an amusing nickname such as "Sarge," "Slugger," or "Master Blaster."
ARE YOU ROBBING A BANK?
YES
NO
ARE YOU IN A BATHROBE?
YES
NO
Say hi.
Excerpt from Inconsequential Dilemmas
KNOCKKNOCKSTUFF.COM • © 2013 WHO'S THERE INC.

# Axes Aligned Decision Boundary

## Motivation



# Decision Tree

## Description

- Find the feature that is the most informative.
- Split the training set into subsets according to this feature.
- Repeat on the subsets until all the labels in the subset are the same.

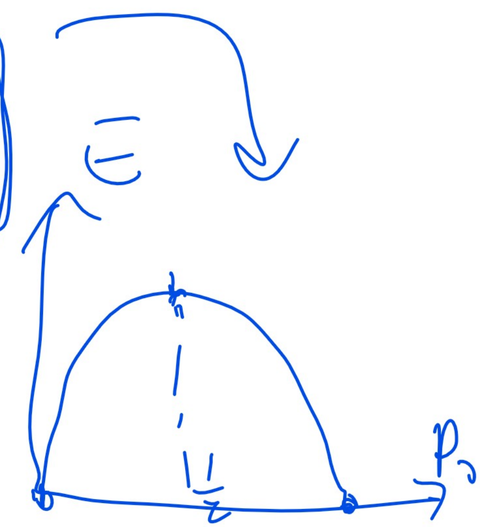
# Binary Entropy

## Definition

- Entropy is the measure of uncertainty.
- The value of something uncertain is more informative than the value of something certain.
- For binary labels, $y_i \in \{0, 1\}$, suppose $p_0$ fraction of labels are 0 and $1 - p_0 = p_1$ fraction of the training set labels are 1, the entropy is:

$$H(Y) = p_0 \log_2\left(\frac{1}{p_0}\right) + p_1 \log_2\left(\frac{1}{p_1}\right)$$
$$= -p_0 \log_2(p_0) - p_1 \log_2(p_1)$$

| | | $p_0$ | $p_1$ | E |
|---|---|---|---|---|
| certain | C | 1 | 0 | 0 |
| | | 0 | 1 | 0 |
| most uncertain | C | 1/2 | 1/2 | 1 |



# Measure of Uncertainty

## Definition

- If $p_0 = 0$ and $p_1 = 1$, the entropy is 0, the outcome is certain, so there is no uncertainty.
- If $p_0 = 1$ and $p_1 = 0$, the entropy is 0, the outcome is also certain, so there is no uncertainty.
- If $p_0 = \frac{1}{2}$ and $p_1 = \frac{1}{2}$, the entropy is the maximum 1, the outcome is the most uncertain.

# Entropy

## Definition

- If there are $K$ classes and $p_y$ fraction of the training set labels are in class $y$, with $y \in \{1, 2, ..., K\}$, the entropy is:

$$H(Y) = \sum_{y=1}^{K} p_y \log_2 \left( \frac{1}{p_y} \right)$$

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

# Conditional Entropy

## Definition

- Conditional entropy is the entropy of the conditional distribution. Let $K_X$ be the possible values of a feature $X$ and $K_Y$ be the possible labels $Y$. Define $p_x$ as the fraction of the instances that is $x$, and $p_{y|x}$ as the fraction of the labels that are $y$ among the ones with instance $x$.

$$H(Y|X = x) = -\sum_{y=1}^{K_Y} p_{y|x} \log_2 \left(p_{y|x}\right)$$

$$H(Y|X) = \sum_{x=1}^{K_X} p_x H(Y|X = x)$$

# Aside: Cross Entropy

## Definition

- Cross entropy measures the difference between two distributions.

$$H(Y,X) = -\sum_{z=1}^{K} p_{Y=z} \log_2 \left(p_{X=z}\right)$$

- It is used in logistic regression to measure the difference between actual label $Y_i$ and the predicted label $A_i$ for instance $i$, and at the same time, to make the cost convex.

$$H(Y_i, A_i) = -y_i \log(a_i) - (1 - y_i)\log(1 - a_i)$$

# Information Gain

## Definition

- The information gain is defined as the difference between the entropy and the conditional entropy.

$$I(Y|X) = H(Y) - H(Y|X).$$

- The larger than information gain, the larger the reduction in uncertainty, and the better predictor the feature is.

# Splitting Discrete Features

## Definition

- The most informative feature is the one with the largest information gain.

$$\arg\max_j I\left(Y|X_j\right)$$

- Splitting means dividing the training set into $K_{X_j}$ subsets.

$$\left\{(x_i, y_i) : x_{ij} = 1\right\}, \left\{(x_i, y_i) : x_{ij} = 2\right\}, ..., \left\{(x_i, y_i) : x_{ij} = K_{X_j}\right\}$$

# Splitting Continuous Features

## Definition

- Continuous features can be (arbitrarily) uniformly split into $K_X$ categories.
- To construct binary splits, all possible splits of the continuous feature can be constructed, and the one that yields the highest information gain is used.

$$\mathbb{1}_{\{X_j \leqslant x_{1j}\}}, \mathbb{1}_{\{X_j \leqslant x_{2j}\}}, \ldots, \mathbb{1}_{\{X_j \leqslant x_{nj}\}}$$

- One of the above binary features is used in place of the original continuous feature $X_j$.

# Choice of Thresholds

## Definition

- In practice, the efficient way to create the binary splits uses the midpoint between instances of different classes.
- The instances in the training set are sorted by $X_j$, say $x_{(1)j}, x_{(2)j}, \ldots, x_{(n)j}$, and suppose $x_{(i)j}$ and $x_{(i+1)j}$ have different labels, then $\frac{1}{2}\left(x_{(i)j} + x_{(i+1)j}\right)$ is considered as a possible binary split.

$$\mathbb{1}_{\left\{X_j \leqslant \frac{1}{2}\left(x_{(i)j}+x_{(i+1)j}\right)\right\}}$$

# Splitting Continuous Variables Diagram

## Definition

# ID3 Algorithm (Iterative Dichotomiser 3), Part I

## Algorithm

- Input: instances: $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$, feature $j$ is split into $K_j$ categories and $y$ has $K$ categories
- Output: a decision tree
- Start with the complete set of instances $\{x_i\}_{i=1}^n$.
- Suppose the current subset of instances is $\{x_i\}_{i \in S}$, find the information gain from each feature.

$$I(Y|X_j) = H(Y) - H(Y|X_j)$$

# ID3 Algorithm (Iterative Dichotomiser 3), Part II

## Algorithm

$$H(Y) = -\sum_{y=1}^{K} \frac{\#(Y=y)}{\#(Y)} \log\left(\frac{\#(Y=y)}{\#(Y)}\right)$$

$$H(Y|X_j) = -\sum_{x=1}^{K_j} \sum_{y=1}^{K} \frac{\#(Y=y, X_j=x)}{\#(Y)} \log\left(\frac{\#(Y=y, X_j=x)}{\#(X_j=x)}\right)$$

- Find the more informative feature $j^\star$.

$$j^\star = \arg\max_j I(Y|X_j)$$

# ID3 Algorithm (Iterative Dichotomiser 3), Part III

## Algorithm

- Split the subset $S$ into $K_{j^\star}$ subsets.

$$S_1 = \{(x_i, y_i) \in S : x_{ij^\star} = 1\}$$
$$S_2 = \{(x_i, y_i) \in S : x_{ij^\star} = 2\}$$
$$\dots$$
$$S_{K_{X_{j^\star}}} = \left\{(x_i, y_i) \in S : x_{ij^\star} = K_{X_{j^\star}}\right\}$$

- Recurse over the subsets until $p_y = 1$ for some $y$ on the subset.

# Pruning Diagram

## Disucssion

# Pruning

## Discussion

- Use the validation set to prune subtrees by making them a leaf. The leaf created by pruning a subtree has label equal to the majority of the training examples reaching this subtree.
- If making a subtree a leaf does not descrease the accuracy on the validation set, then the subtree is pruned.
- This is one of the simplest ways to prune a decision tree, called Reduced Error Pruning.

# Bagging

## Discussion

- Create many smaller training sets by sampling with replacement from the complete training set.
- Train different decision trees using the smaller training sets.
- Predict the label of new instances by majority vote from the decision trees.
- This is called bootstrap aggregating (bagging).

# Random Forrest

## Discussion

- When training the decision trees on the smaller training sets, only a random subset of the features are used. The decision trees are created without pruning.
- This algorithm is called random forests.

# Boosting

## Discussion

- The idea of boosting is to combine many weak decision trees, for example, decision stumps, into a strong one.
- Decision trees are trained sequentially. The instances that are classified incorrectly by previous trees are made more important for the next tree.

# Adaptive Boosting, Part I

## Discussion

- The weights $w$ for the instances are initialized uniformly.

$$w = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)$$

- In each iteration, a decision tree $f_k$ is trained on the training instances weighted by $w$.

$$f_k = \arg\min_f \sum_{i=1}^{n} w_i \mathbb{1}_{\{f(x_i) \neq y_i\}}$$

$$\varepsilon_k = \min_f \sum_{i=1}^{n} w_i \mathbb{1}_{\{f_k(x_i) \neq y_i\}}$$

# Adaptive Boosting, Part II

## Discussion

- The weights for the tree $f_k$ is computed.

$$\alpha_k = \log\left(\frac{1-\varepsilon_k}{\varepsilon_k}\right)$$

- The weights are updated according to the error $\varepsilon$ made by $f_k$, and normalized so that the sum is 1.

$$w_i = w_i e^{-\alpha_k\left(2\cdot\mathbb{1}_{\{f_k(x_i)=y_i\}}-1\right)}$$

# Adaptive Boosting, Part II

## Discussion

- The label of a new test instance $x_i$ is the $\alpha$ weighted majority of the labels produced by all $K$ trees: $f_1(x_i), f_2(x_i), ..., f_K(x_i)$.
- For example, if there are only two classes $\{0, 1\}$, and $\alpha$ is normalized so that the sum is 1, then the prediction is the following.

$$\hat{y}_i = \mathbb{1}_{\left\{\sum_{k=1}^{K} \alpha_k f_k(x_i) \geqslant 0.5\right\}}$$

# K Nearest Neighbor

## Description

- Given a new instance, find the $K$ instances in the training set that are the closest.
- Predict the label of the new instance by the majority of the labels of the $K$ instances.

# Distance Function

## Definition

- Many distance functions can be used in place of the Euclidean distance.

$$\rho\left(x, x'\right) = \left\|x - x'\right\|_2 = \sqrt{\sum_{j=1}^{m}\left(x_j - x_j'\right)^2}$$

- An example is Manhattan distance.

$$\rho\left(x, x'\right) = \sum_{j=1}^{m}\left|x_j - x_j'\right|$$

# Manhattan Distance Diagram

## Definition

# P Norms

## Definition

- Another group of examples is the $p$ norms.

$$\rho\left(x, x'\right) = \left(\sum_{j=1}^{m} \left|x_j - x_j'\right|^p\right)^{\frac{1}{p}}$$

- $p = 1$ is the Manhattan distance.
- $p = 2$ is the Euclidean distance.
- $p = \infty$ is the sup distance, $\rho\left(x, x'\right) = \max_{i=1,2,\ldots,m} \left\{\left|x_j - x_j'\right|\right\}$.
- $p$ cannot be less than 1.

# K Nearest Neighbor

## Algorithm

- Input: instances: $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$, and a new instance $\hat{x}$.
- Output: new label $\hat{y}$.
- Order the training instances according to the distance to $\hat{x}$.

$$\rho\left(\hat{x}, x_{(i)}\right) \leqslant \rho\left(\hat{x}, x_{(i+1)}\right), i = 1, 2, ..., n-1$$

- Assign the majority label of the closest $k$ instances.

$$\hat{y} = \text{ mode } \left\{y_{(1)}, y_{(2)}, ..., y_{(k)}\right\}$$