10-301/601: Introduction to Machine Learning Lecture 2 – Decision Trees: Model Definition

Henry Chai

5/12/25

Our first Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset

features (Family History, Resting Blood Pressure, Cholesterol); labels (Heart Disease?); data points (rows)

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- This classifier completely ignores the features…

Our first Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Majority vote classifier: always predict the most common label in the **training** dataset

features (Family History, Resting Blood Pressure, Cholesterol) — labels (Heart Disease?) — data points (rows)

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
Yes	Low	Normal	No	Yes
No	Medium	Normal	No	Yes
No	Low	Abnormal	Yes	Yes
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes

- The training error rate is $2/5$

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
Yes	Low	Normal	No	No
No	Medium	Normal	No	No
No	Low	Abnormal	Yes	Yes
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes

- The training error rate is 0!

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

Family History	Resting Blood Pressure	Cholesterol	Heart Disease?	Predictions
Yes	Low	Normal	No	No
No	Medium	Normal	No	No
No	Low	Abnormal	Yes	Yes
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes

- The training error rate is 0…

Our second Machine Learning Classifier

- A **classifier** is a function that takes feature values as input and outputs a label
- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote
- The memorizer (typically) does not **generalize** well, i.e., it does not perform well on unseen data points
- In some sense, good generalization, i.e., the ability to make accurate predictions given a small training dataset, is the whole point of machine learning!

Notation

- Feature space, $\mathcal{X}$
- Label space, $\mathcal{Y}$
- (Unknown) Target function, $c^*: \mathcal{X} \rightarrow \mathcal{Y}$
- Training dataset:

$$\mathcal{D} = \{(\boldsymbol{x}^{(1)}, c^*(\boldsymbol{x}^{(1)}) = y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}) \ldots, (\boldsymbol{x}^{(N)}, y^{(N)})\}$$

- Data point:

$$(\boldsymbol{x}^{(n)}, y^{(n)}) = \left(x_1^{(n)}, x_2^{(n)}, \ldots, x_D^{(n)}, y^{(n)}\right)$$

- Classifier, $h : \mathcal{X} \rightarrow \mathcal{Y}$
- Goal: find a classifier, h, that "best approximates" c^*

Evaluation

- Loss function, $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$
  - Defines how "bad" predictions, $\hat{y} = h(\boldsymbol{x})$, are compared to the true labels, $y = c^*(\boldsymbol{x})$
  - Common choices:

1. Squared loss (for regression): $\ell(y, \hat{y}) = (y - \hat{y})^2$
2. Binary or 0-1 loss (for classification):

$$\ell(y, \hat{y}) = \begin{cases} 1 & \text{if } y \neq \hat{y} \\ 0 & \text{otherwise} \end{cases}$$

Evaluation

- Loss function, $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
  - Defines how "bad" predictions, $\hat{y} = h(\boldsymbol{x})$, are compared to the true labels, $y = c^*(\boldsymbol{x})$
  - Common choices
  1. Squared loss (for regression): $\ell(y, \hat{y}) = (y - \hat{y})^2$
  2. Binary or 0-1 loss (for classification):

$$\ell(y, \hat{y}) = \mathbb{1}(y \neq \hat{y})$$

- Error rate:

$$err(h, \mathcal{D}) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\left(y^{(n)} \neq \hat{y}^{(n)}\right)$$

Notation: Example

- Memorizer: if a set of features exists in the **training** dataset, predict its corresponding label; otherwise, predict the majority vote

	x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ Predictions
	Yes	Low	Normal	No	No
$\boldsymbol{x}^{(2)}$	No	Medium	Normal	No	No
	No	Low	Abnormal	Yes	Yes
	Yes	Medium	Normal	Yes	Yes
	Yes	High	Abnormal	Yes	Yes

- $N = 5$ and $D = 3$
- $\boldsymbol{x}^{(2)} = \left(x_1^{(2)} = \text{"No"}, x_2^{(2)} = \text{"Medium"}, x_3^{(2)} = \text{"Normal"}\right)$

Our second Machine Learning Classifier: Pseudocode

- Memorizer:

def train($\mathcal{D}$):

  store $\mathcal{D}$

def majority_vote($\mathcal{D}$):

  return mode($y^{(1)}, y^{(2)}, \dots, y^{(N)}$)

def predict($\boldsymbol{x}'$):

  if $\exists\, \boldsymbol{x}^{(n)} \in \mathcal{D}$ s.t. $\boldsymbol{x}' = \boldsymbol{x}^{(n)}$:

    return $y^{(n)}$

  else

    return majority_vote($\mathcal{D}$)

Our third Machine Learning Classifier

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump: based on a single feature, x_d, predict the most common label in the training dataset among all data points that have the same value for x_d

Our third Machine Learning Classifier: Example

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x'_1, \dots, x'_D) = \begin{cases} ??? & \text{if } x'_1 = \text{“Yes”} \\ ??? & \text{otherwise} \end{cases}$$

Our third Machine Learning Classifier: Example

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x'_1, \dots, x'_D) = \begin{cases} \text{“Yes” if } x'_1 = \text{“Yes”} \\ ??? \ \text{otherwise} \end{cases}$$

Our third Machine Learning Classifier: Example

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes

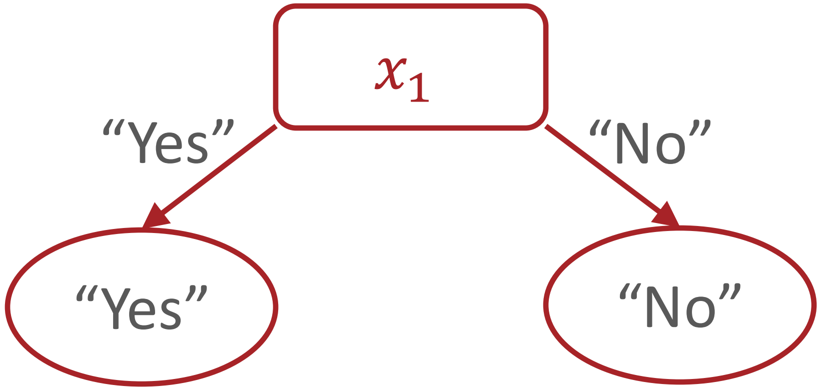
- Decision stump on x_1:

$$h(\boldsymbol{x}') = h(x_1', \ldots, x_D') = \begin{cases} \text{"Yes" if } x_1' = \text{"Yes"} \\ \text{"No" otherwise} \end{cases}$$

Our third Machine Learning Classifier: Example

- Alright, let's actually (try to) extract a pattern from the data

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?	$\hat{y}$ Predictions
Yes	Low	Normal	No	Yes
No	Medium	Normal	No	No
No	Low	Abnormal	Yes	No
Yes	Medium	Normal	Yes	Yes
Yes	High	Abnormal	Yes	Yes



Decision Stumps: Pseudocode

```
def train(𝒟):
    1. pick a feature, x_d
    2. split 𝒟 according to x_d

       for v in V(x_d), all possible values of x_d:
            𝒟_v = {(x^(i), y^(i)) ∈ 𝒟 | x_d^(i) = v}
    3. Compute the majority vote for each split

       for v in V(x_d), all possible values of x_d:
            ŷ_v = majority_vote(𝒟_v)
def predict(x′):

       for v in V(x_d), all possible values of x_d:
            if x′ = v: return ŷ_v
```

Where: $\mathcal{D}_v = \left\{\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D} \mid x_d^{(i)} = v\right\}$, $\hat{y}_v = \text{majority\_vote}(\mathcal{D}_v)$, predict($\boldsymbol{x}'$): if $\boldsymbol{x}' = v$: return $\hat{y}_v$

Decision Stumps: Questions

1. How can we pick which feature to split on?

2. Why stop at just one feature?

Decision Stumps: Questions

1. How can we pick which feature to split on?

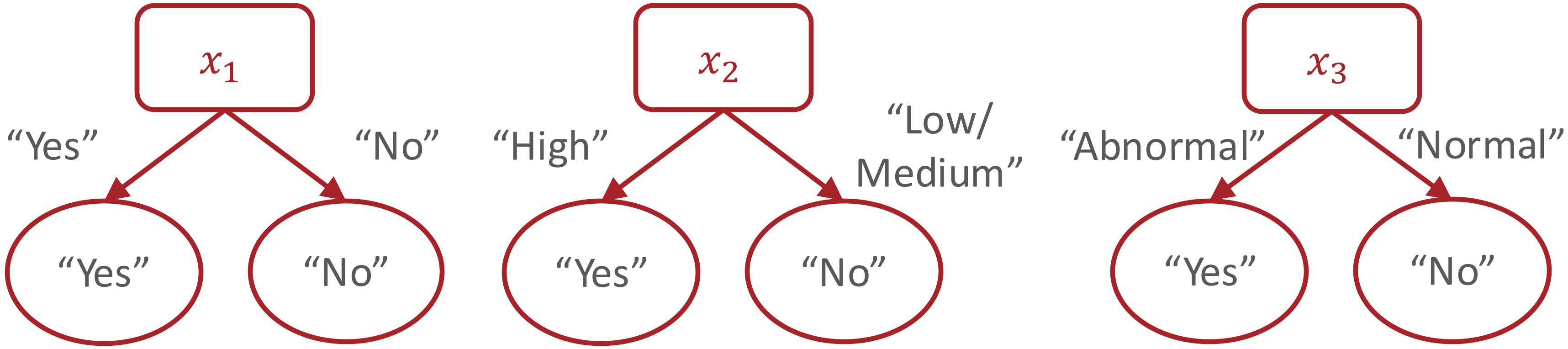
2. Why stop at just one feature?

Splitting Criterion

- A **splitting criterion** is a function that measures how good or useful splitting on a particular feature is *for a specified dataset*
- Insight: use the feature that optimizes the splitting criterion for our decision stump.

Training error rate as a Splitting Criterion

x_1 Family History	x_2 Resting Blood Pressure	x_3 Cholesterol	y Heart Disease?
Yes	Low	Normal	No
No	Medium	Normal	No
No	Low	Abnormal	Yes
Yes	Medium	Normal	Yes
Yes	High	Abnormal	Yes



Training error rate: 2/5

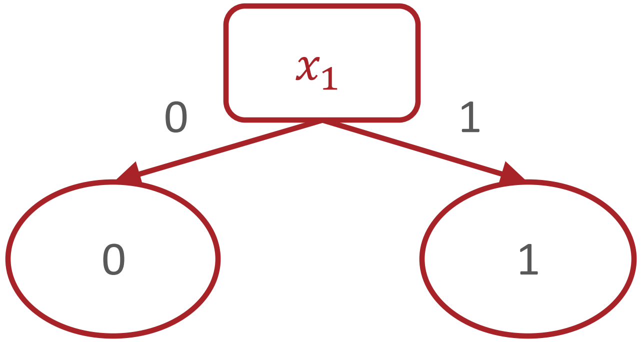
Training error rate: 2/5

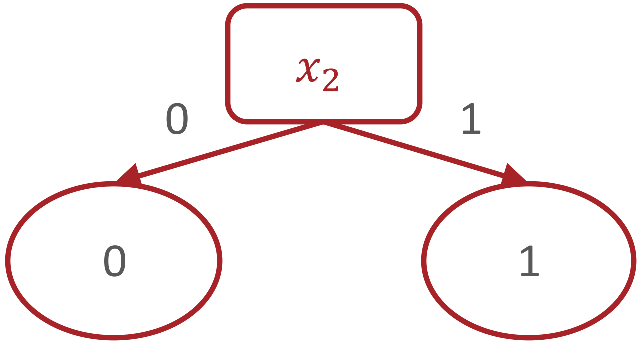
Training error rate: 1/5

Training error rate as a Splitting Criterion?

x_1	x_2	y
1	0	0
1	0	0
1	0	1
1	0	1
1	1	1
1	1	1
1	1	1
1	1	1

- Which feature would you split on using training error rate as the splitting criterion?





Training error rate: 2/8

Splitting Criterion

- A **splitting criterion** is a function that measures how good or useful splitting on a particular feature is *for a specified dataset*
- Insight: use the feature that optimizes the splitting criterion for our decision stump.
- Potential splitting criteria:
  - Training error rate (minimize)
  - Gini impurity (minimize) → CART algorithm
  - Mutual information (maximize) → ID3 algorithm

Splitting Criterion

- A **splitting criterion** is a function that measures how good or useful splitting on a particular feature is *for a specified dataset*
- Insight: use the feature that optimizes the splitting criterion for our decision stump.
- Potential splitting criteria:
    - Training error rate (minimize)
    - Gini impurity (minimize) → CART algorithm
    - Mutual information (maximize) → ID3 algorithm

Entropy

- The **entropy** of a *random variable* describes the uncertainty of its outcome: the higher the entropy, the less certain we are about what the outcome will be.

$$H(X) = -\sum_{v \in V(X)} P(X = v) \log_2\big(P(X = v)\big)$$

where X is a (discrete) random variable

$V(X)$ is the set of possible values X can take on

Entropy

- The **entropy** of a *set* describes how uniform or pure it is: the higher the entropy, the more impure or "mixed-up" the set is

$$H(S) = -\sum_{v \in V(S)} \frac{|S_v|}{|S|} \log_2\left(\frac{|S_v|}{|S|}\right)$$

where S is a collection of values,

$V(S)$ is the set of unique values in S

S_v is the collection of elements in S with value v

- If all the elements in S are the same, then

$$H(S) = -1 \log_2(1) = 0$$

Entropy

- The **entropy** of a *set* describes how uniform or pure it is: the higher the entropy, the more impure or "mixed-up" the set is

$$H(S) = -\sum_{v \in V(S)} \frac{|S_v|}{|S|} \log_2\left(\frac{|S_v|}{|S|}\right)$$

where S is a collection of values,

$V(S)$ is the set of unique values in S

S_v is the collection of elements in S with value v

- If S is split fifty-fifty between two values, then

$$H(S) = -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = -\log_2\left(\frac{1}{2}\right) = 1$$

Mutual Information

- The **mutual information** between *two random variables* describes how much clarity knowing the value of one random variables provides about the other

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

$$= H(Y) - \sum_{v \in V(X)} P(X = v) H(Y|X = v)$$

where X and Y are (discrete) random variables

$V(X)$ is the set of possible values X can take on

$H(Y|X = v)$ is the conditional entropy of Y given $X = v$

Mutual Information

- The **mutual information** between *a feature and the label* describes how much clarity knowing the feature provides about the label

$$I(y; x_d) = H(y) - H(y|x_d)$$

$$= H(y) - \sum_{v \in V(x_d)} f_v * H(Y_{x_d=v})$$

where x_d is a feature and y is the set of all labels

$V(x_d)$ is the set of possible values x_d can take on

f_v is the fraction of data points where $x_d = v$

$Y_{x_d=v}$ is the set of all labels where $x_d = v$

Mutual Information: Example

x_d	y
1	1
1	1
0	0
0	0

$$I(x_d, Y) = H(Y) - \sum_{v \in V(x_d)} f_v * H\left(Y_{x_d=v}\right)$$

$$= 1 - \frac{1}{2} H\left(Y_{x_d=0}\right) - \frac{1}{2} H\left(Y_{x_d=1}\right)$$

$$= 1 - \frac{1}{2}(0) - \frac{1}{2}(0) = 1$$

Mutual Information: Example

x_d	y
1	1
0	1
1	0
0	0

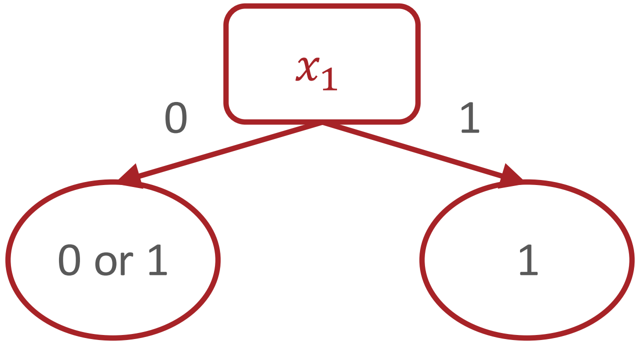
$$I(x_d, Y) = H(Y) - \sum_{v \in V(x_d)} f_v * H\left(Y_{x_d=v}\right)$$

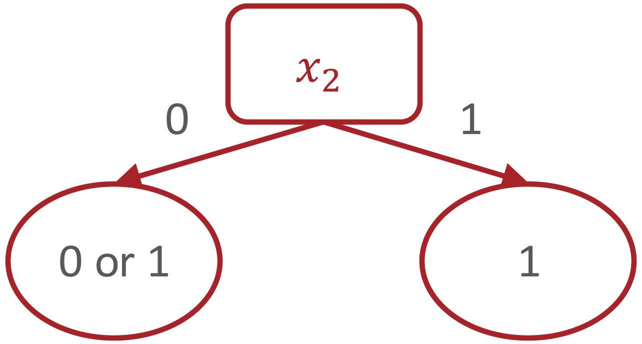
$$= 1 - \frac{1}{2} H\left(Y_{x_d=0}\right) - \frac{1}{2} H\left(Y_{x_d=1}\right)$$

$$= 1 - \frac{1}{2}(1) - \frac{1}{2}(1) = 0$$

Mutual Information as a Splitting Criterion

x_1	x_2	y
1	0	0
1	0	0
1	0	1
1	0	1
1	1	1
1	1	1
1	1	1
1	1	1

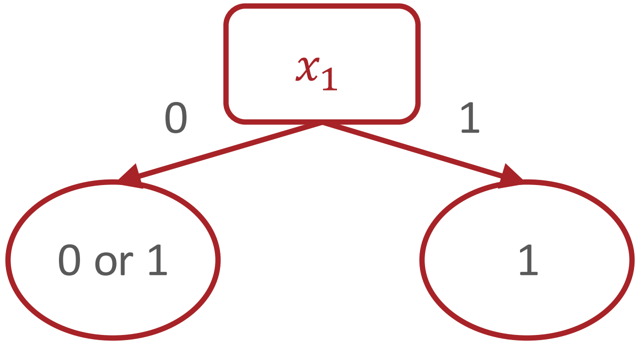




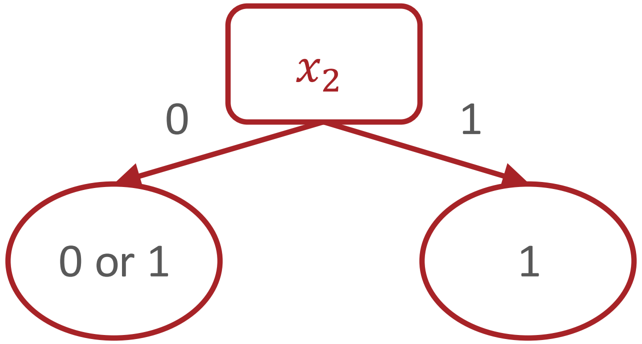
Mutual Information: $H(Y) - \frac{1}{2}H\left(Y_{x_2=0}\right) - \frac{1}{2}H\left(Y_{x_2=1}\right)$

Mutual Information as a Splitting Criterion

x_1	x_2	y
1	0	0
1	0	0
1	0	1
1	0	1
1	1	1
1	1	1
1	1	1
1	1	1



Mutual Information: 0



Mutual Information: $\left(-\frac{2}{8}\log_2\frac{2}{8}-\frac{6}{8}\log_2\frac{6}{8}\right)-\frac{1}{2}(1)-\frac{1}{2}(0)\approx 0.31$

Key Takeaways

- Memorization as a form of learning
- Generalization
- Notation for datasets and evaluation
- Mutual information as a splitting criterion for decision stumps/trees