CS480/680: Introduction to Machine Learning

Lec 13: Decision Trees

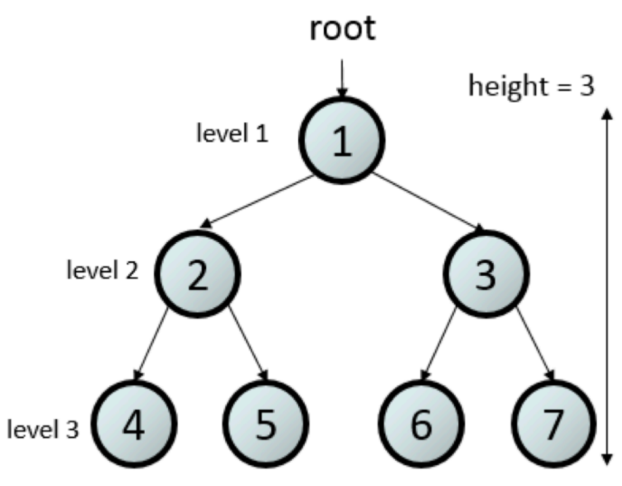
Yaoliang Yu

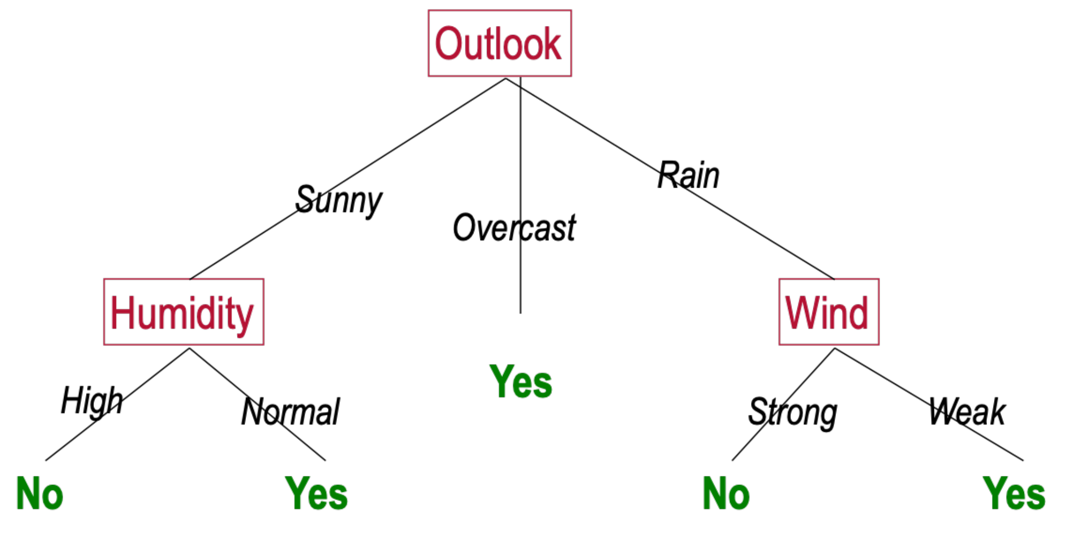
UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS
DAVID R. CHERITON SCHOOL OF COMPUTER SCIENCE

June 19, 2024

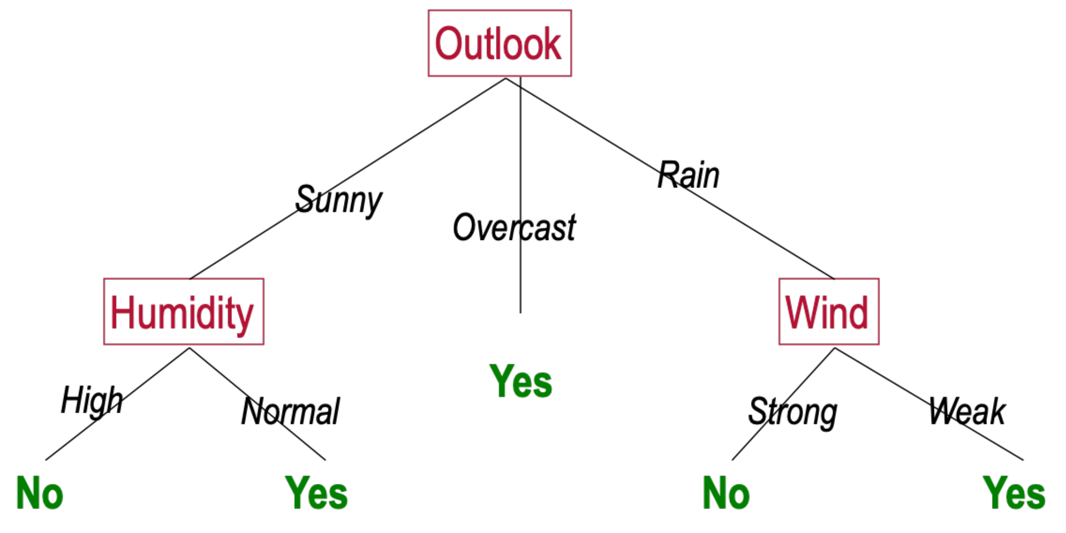
Trees Recalled

Trees Recalled



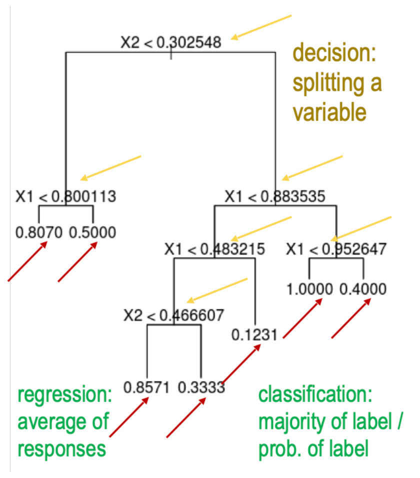


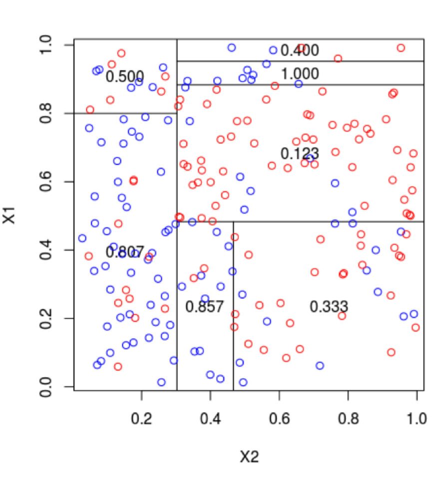
- Decision trees can represent any boolean function



- Decision trees can represent any boolean function

Classification And Regression Tree





L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. "Classification and Regression Trees". CRC, 1984.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization: e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

Learning Decision Trees

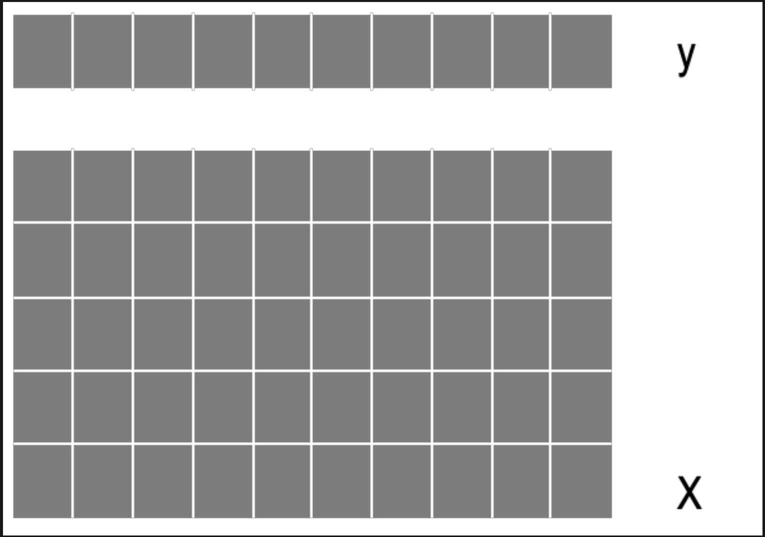
- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.

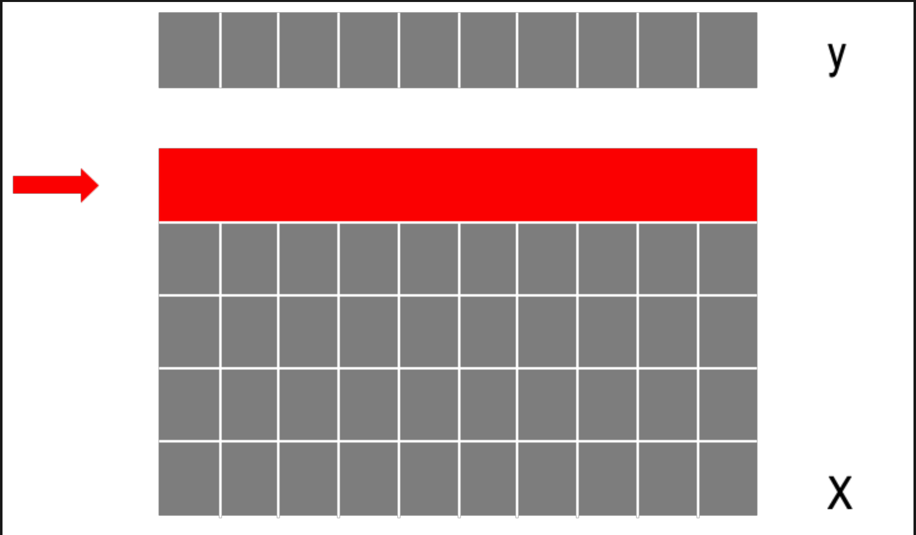
Learning Decision Trees

- Which variables to split at each stage?
- What threshold to use?
- When to stop?
  - regularization, e.g. early stopping
  - pruning
- What to put at the leaves?
  - classification: majority / probability
  - regression: average
  - can also simply store the training set

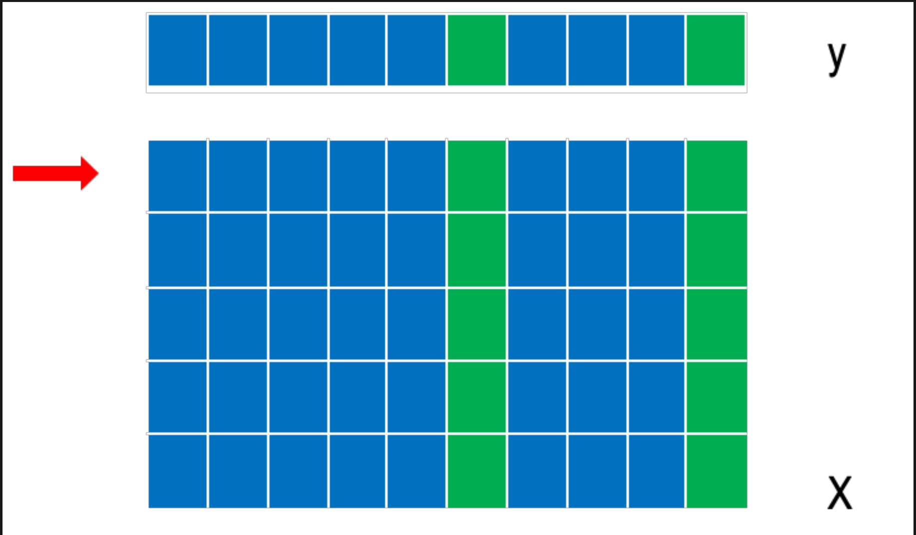
L. Hyafil and R. L. Rivest. "Constructing optimal binary decision trees is NP-complete". *Information Processing Letters*, vol. 5, no. 1 (1976), pp. 15–17.



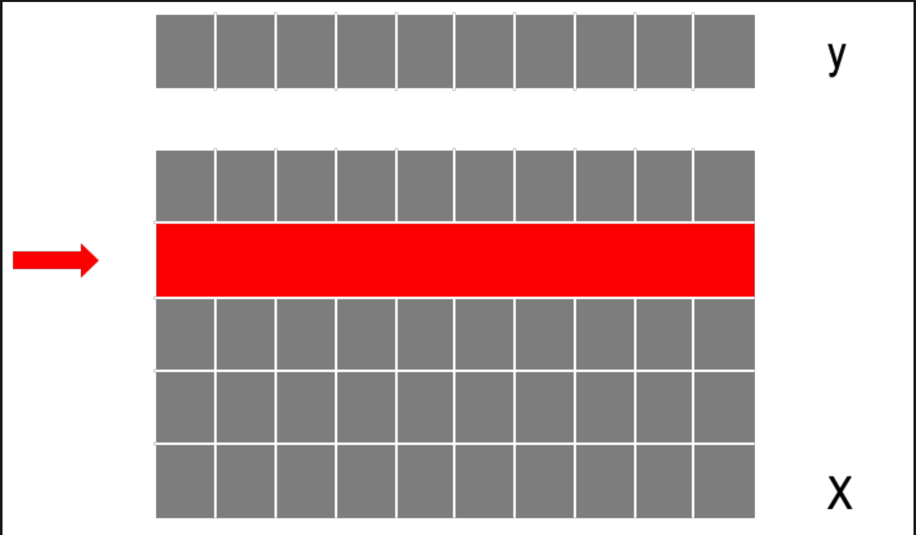
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



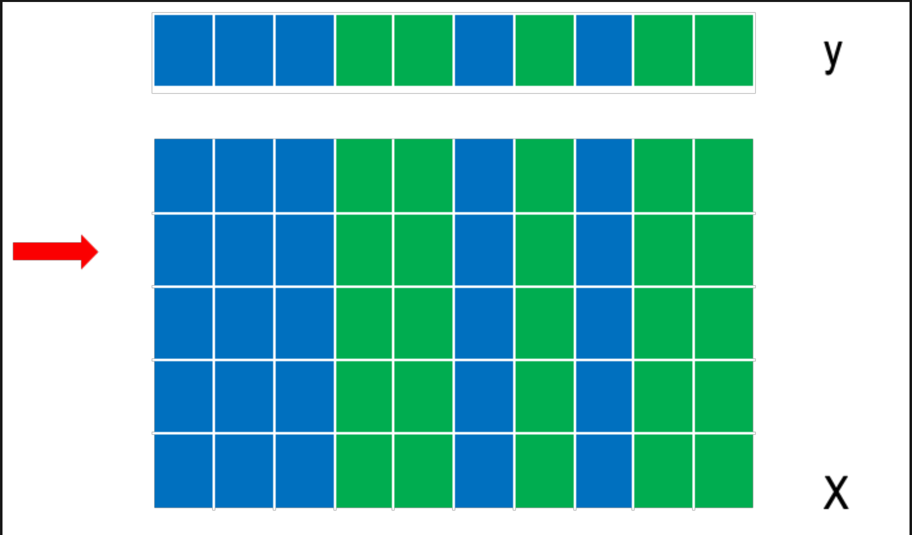
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



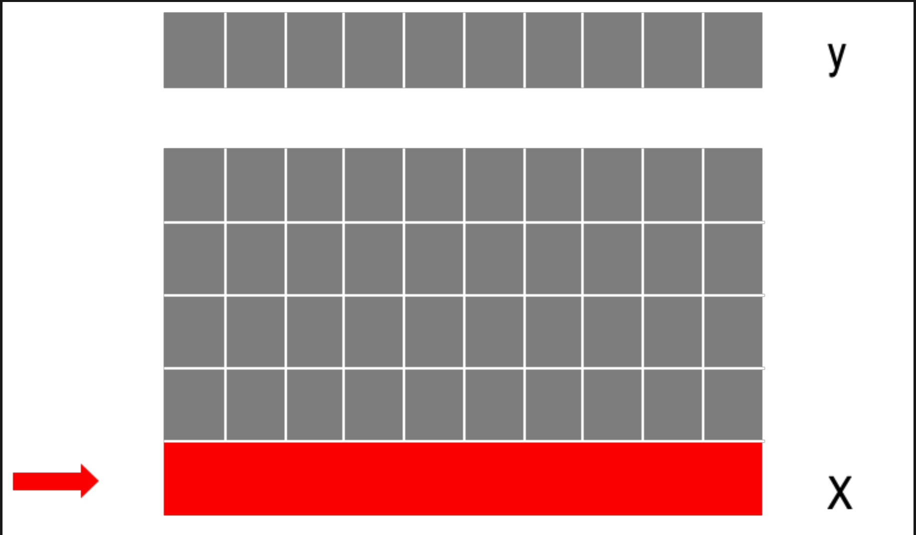
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



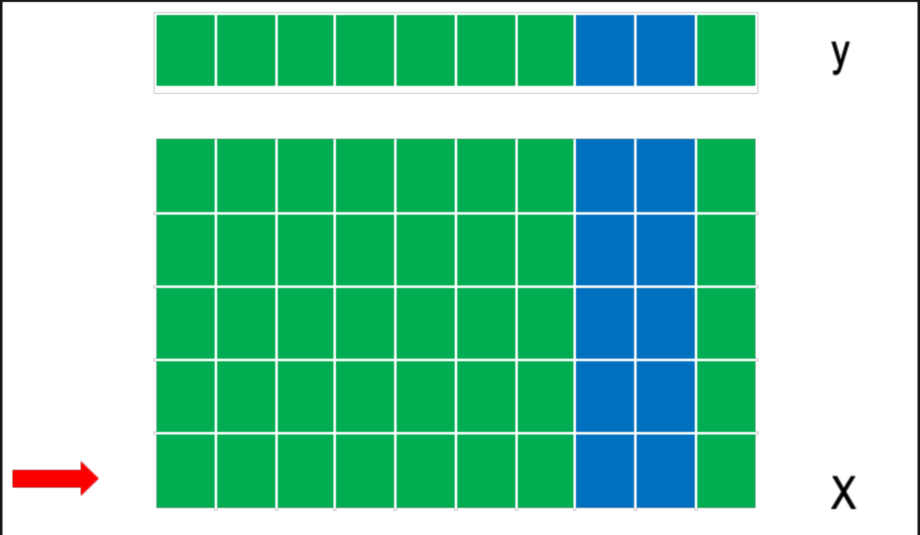
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



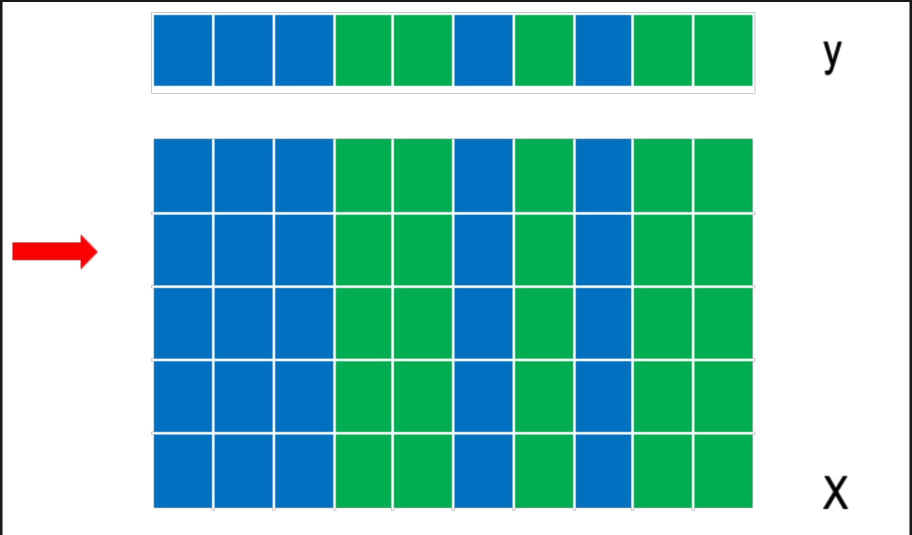
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



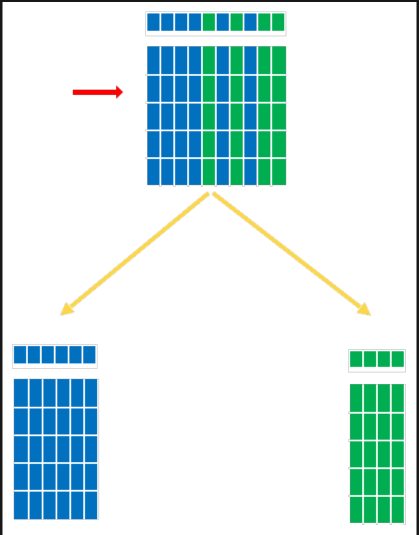
- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well



- Splitting can only be based on $\mathbf{x}$
- Evaluation can be based on y as well

Splitting by Minimizing

$$(j^*, t^*) = \operatorname*{argmin}_{j=1,\ldots,d} \min_{t \in T_j} \ \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \leq t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \leq t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Splitting by Minimizing

$$(j^*, t^*) = \underset{j=1,\dots,d}{\operatorname{argmin}} \min_{t \in T_j} \ \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \leq t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \leq t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Splitting by Minimizing

$$(j^*, t^*) = \operatorname*{argmin}_{j=1,\dots,d} \min_{t \in T_j} \ \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \le t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \le t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Splitting by Minimizing

$$(j^*, t^*) = \operatorname*{argmin}_{j=1,\ldots,d} \min_{t \in T_j} \ \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \le t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \le t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Splitting by Minimizing

$$(j^*, t^*) = \operatorname*{argmin}_{j=1,\ldots,d} \min_{t \in T_j} \ \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \le t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \le t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Splitting by Minimizing

$$(j^*, t^*) = \operatorname*{argmin}_{j=1,\ldots,d} \min_{t \in T_j} \; \ell(\{\mathbf{x}_i, y_i\} : x_{ij} \leq t) + \ell(\{\mathbf{x}_i, y_i\} : x_{ij} > t)$$

- Greedily choose the j-th feature to split
- Greedily choose a threshold $t \in T_j$ to split
  - what should T_j be? For categorical features?
- Partition training data into two disjoint parts: $x_{ij} \leq t$ vs. $x_{ij} > t$
- Evaluate the resulting cost (objective)

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R)\right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R) \right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R)\right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R)\right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R)\right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R) \right)$$

- Cross-validation

Stopping Criterion

- Maximum depth exceeded
- Maximum runtime exceeded
- All childen nodes are (sufficiently) homogeneous
- All children nodes have too few training examples
- Reduction in cost stagnates:

$$\Delta := \ell(D) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \cdot \ell(\mathcal{D}_R) \right)$$

- Cross-validation

Regression Cost

$$\ell(\mathcal{D}) := \left[\min_y \sum_{y_i \in \mathcal{D}} (y_i - y)^2\right] = \sum_{y_i \in \mathcal{D}} (y_i - \bar{y})^2, \quad \text{where} \quad \bar{y} = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} y_i$$

- Can use any reasonable loss (other than the square loss)
- Can even fit a regression model on $\mathcal{D}$

Regression Cost

$$\ell(\mathcal{D}) := \left[\min_{y} \sum_{y_i \in \mathcal{D}} (y_i - y)^2\right] = \sum_{y_i \in \mathcal{D}} (y_i - \bar{y})^2, \quad \text{where} \quad \bar{y} = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} y_i$$

- Can use any reasonable loss (other than the square loss)
- Can even fit a regression model on $\mathcal{D}$

Regression Cost

$$\ell(\mathcal{D}) := \left[\min_{y} \sum_{y_i \in \mathcal{D}} (y_i - y)^2\right] = \sum_{y_i \in \mathcal{D}} (y_i - \bar{y})^2, \quad \text{where} \quad \bar{y} = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} y_i$$

- Can use any reasonable loss (other than the square loss)
- Can even fit a regression model on $\mathcal{D}$

Classification Cost

$$\hat{p}_k = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} [\![y_i \in k]\!], \qquad \hat{y} := \underset{k=1,\ldots,c}{\operatorname{argmax}} \ \hat{p}_k$$

- Misclassification error: $\ell(\mathcal{D}) := 1 - \hat{p}_{\hat{y}}$, reduces to $\hat{p} \wedge (1 - \hat{p})$ if $c = 2$
- Gini index: $\ell(\mathcal{D}) := \sum_{k=1}^{c} \hat{p}_k (1 - \hat{p}_k) = 1 - \sum_{k=1}^{c} \hat{p}_k^2$, reduces to $2\hat{p}(1 - \hat{p})$ if $c = 2$
- Entropy: $\ell(\mathcal{D}) := -\sum_{k=1}^{c} \hat{p}_k \log \hat{p}_k$, reduces to $-\hat{p} \log \hat{p} - (1 - \hat{p}) \log(1 - \hat{p})$ if $c = 2$

Classification Cost

$$\hat{p}_k = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} [\![y_i \in k]\!], \qquad \hat{y} := \operatorname*{argmax}_{k=1,\dots,c} \ \hat{p}_k$$

- Misclassification error: $\ell(\mathcal{D}) := 1 - \hat{p}_{\hat{y}}$, reduces to $\hat{p} \wedge (1 - \hat{p})$ if $c = 2$
- Gini index: $\ell(\mathcal{D}) := \sum_{k=1}^{c} \hat{p}_k (1 - \hat{p}_k) = 1 - \sum_{k=1}^{c} \hat{p}_k^2$, reduces to $2\hat{p}(1-\hat{p})$ if $c = 2$
- Entropy: $\ell(\mathcal{D}) := -\sum_{k=1}^{c} \hat{p}_k \log \hat{p}_k$, reduces to $-\hat{p} \log \hat{p} - (1 - \hat{p}) \log(1 - \hat{p})$ if $c = 2$

Classification Cost

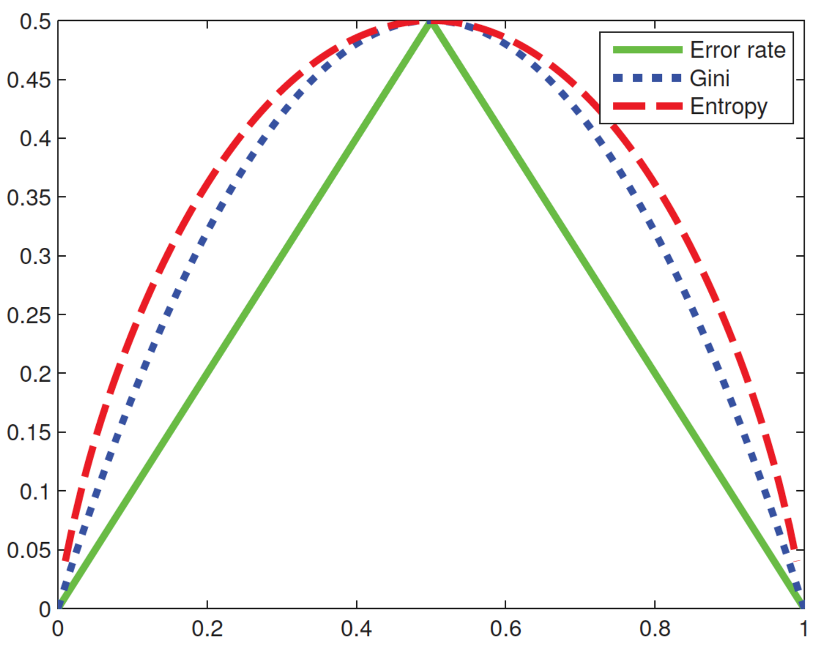
$$\hat{p}_k = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} [\![y_i \in k]\!], \qquad \hat{y} := \operatorname*{argmax}_{k=1,\ldots,c} \; \hat{p}_k$$

- Misclassification error: $\ell(\mathcal{D}) := 1 - \hat{p}_{\hat{y}}$, reduces to $\hat{p} \wedge (1 - \hat{p})$ if $c = 2$
- Gini index: $\ell(\mathcal{D}) := \sum_{k=1}^{c} \hat{p}_k (1 - \hat{p}_k) = 1 - \sum_{k=1}^{c} \hat{p}_k^2$, reduces to $2\hat{p}(1 - \hat{p})$ if $c = 2$
- Entropy: $\ell(\mathcal{D}) := -\sum_{k=1}^{c} \hat{p}_k \log \hat{p}_k$, reduces to $-\hat{p} \log \hat{p} - (1 - \hat{p}) \log(1 - \hat{p})$ if $c = 2$

Classification Cost

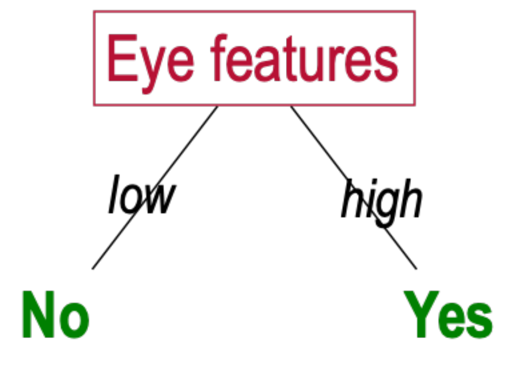
$$\hat{p}_k = \frac{1}{|\mathcal{D}|} \sum_{y_i \in \mathcal{D}} [\![y_i \in k]\!], \qquad \hat{y} := \underset{k=1,\dots,c}{\operatorname{argmax}} \ \hat{p}_k$$

- Misclassification error: $\ell(\mathcal{D}) := 1 - \hat{p}_{\hat{y}}$, reduces to $\hat{p} \wedge (1 - \hat{p})$ if $c = 2$
- Gini index: $\ell(\mathcal{D}) := \sum_{k=1}^{c} \hat{p}_k (1 - \hat{p}_k) = 1 - \sum_{k=1}^{c} \hat{p}_k^2$, reduces to $2\hat{p}(1 - \hat{p})$ if $c = 2$
- Entropy: $\ell(\mathcal{D}) := -\sum_{k=1}^{c} \hat{p}_k \log \hat{p}_k$, reduces to $-\hat{p} \log \hat{p} - (1 - \hat{p}) \log(1 - \hat{p})$ if $c = 2$



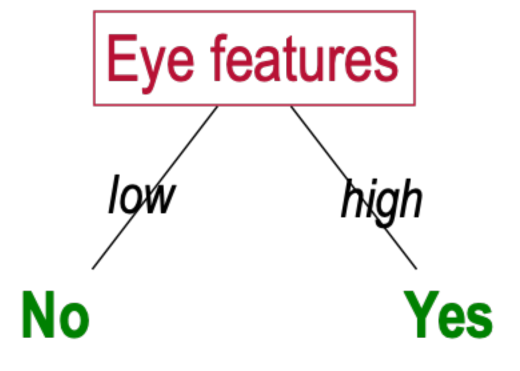
Error rate
Gini
Entropy
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.2
0.4
0.6
0.8
1

Decision Stump



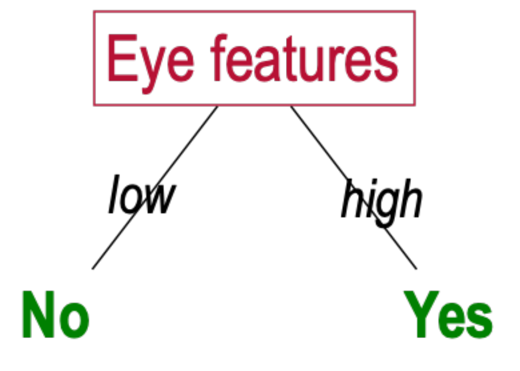
- A binary tree with depth 1
- Performs classification based on 1 feature
- Easy to train, interprettable, but underfits (addressed in next lecture)

Decision Stump



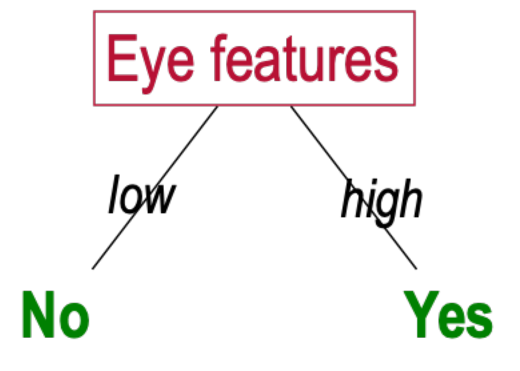
- A binary tree with depth 1
- Performs classification based on 1 feature
- Easy to train, interprettable, but underfits (addressed in next lecture)

Decision Stump



- A binary tree with depth 1
- Performs classification based on 1 feature
- Easy to train, interprettable, but underfits (addressed in next lecture)

Decision Stump



- A binary tree with depth 1
- Performs classification based on 1 feature
- Easy to train, interprettable, but underfits (addressed in next lecture)