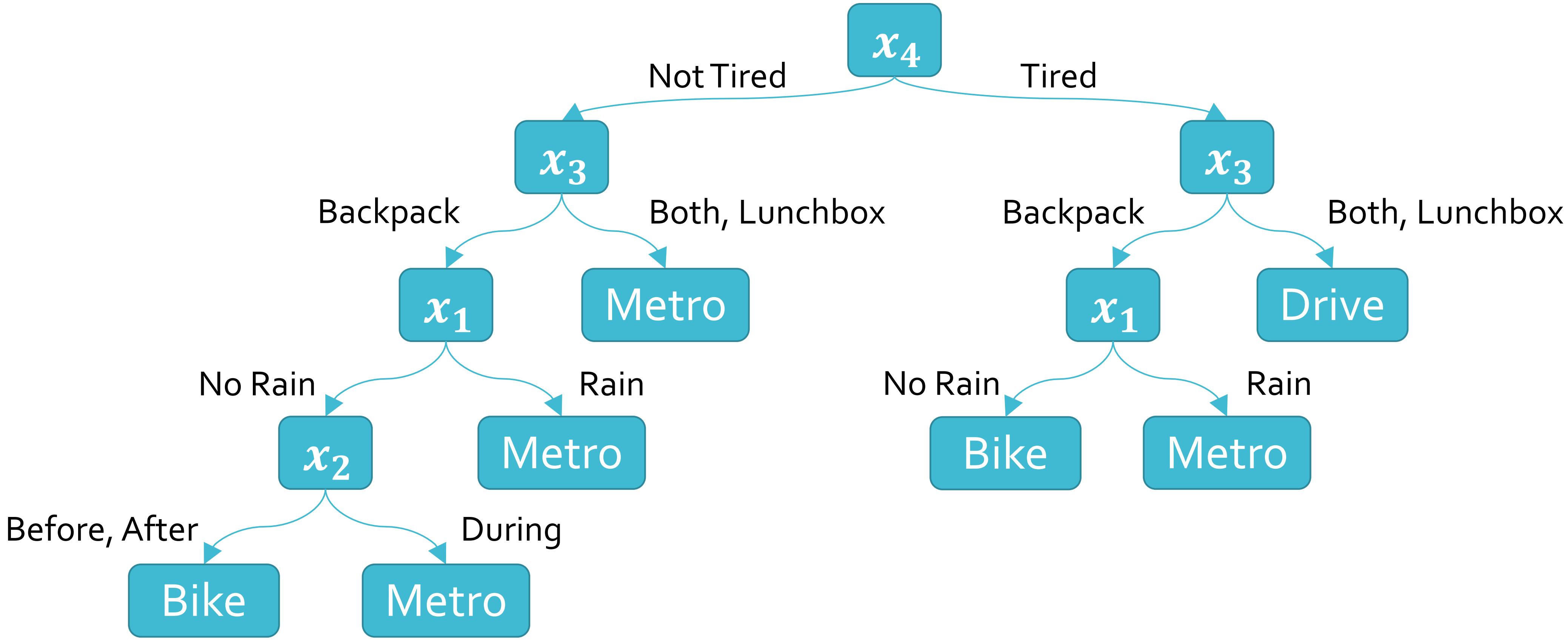
CSE 417T: Introduction to Machine Learning

Lecture 16: Bagging

Henry Chai

10/25/18



Decision Tree: Example

Decision Tree / ID3 Pros

- Intuitive / explainable
- Can handle categorical and real-valued features
- Automatically performs feature selection
- The ID3 algorithm has a preference for shorter trees (simpler hypotheses)

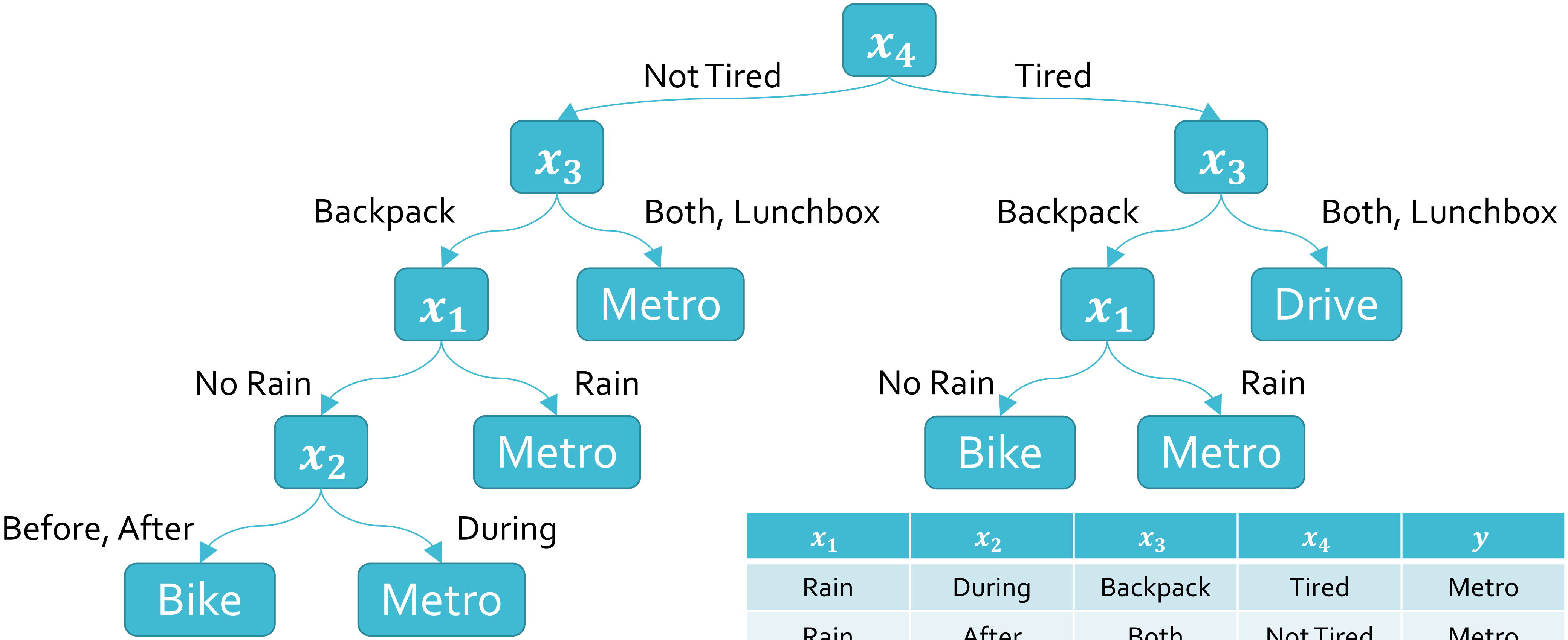
Decision Tree / ID3 Cons

- The ID3 algorithm is greedy (it selects the feature w/ the highest information gain at every step) so no optimality guarantee

- Overfitting
  - Can be addressed via heuristics ("regularization") or pruning ("validation"):

Addressing Overfitting

- Heuristics (”regularization”):
  - Do not split leaves past a fixed depth δ
  - Do not split leaves with fewer than c labels
  - Do not split leaves where the maximal information gain is less than τ
  - Predict the most common label at each leaf
- Pruning (“validation”):
  - Evaluate each split using a validation set
  - Compare the validation error with and without that split (replacing it with the most common label at that point)

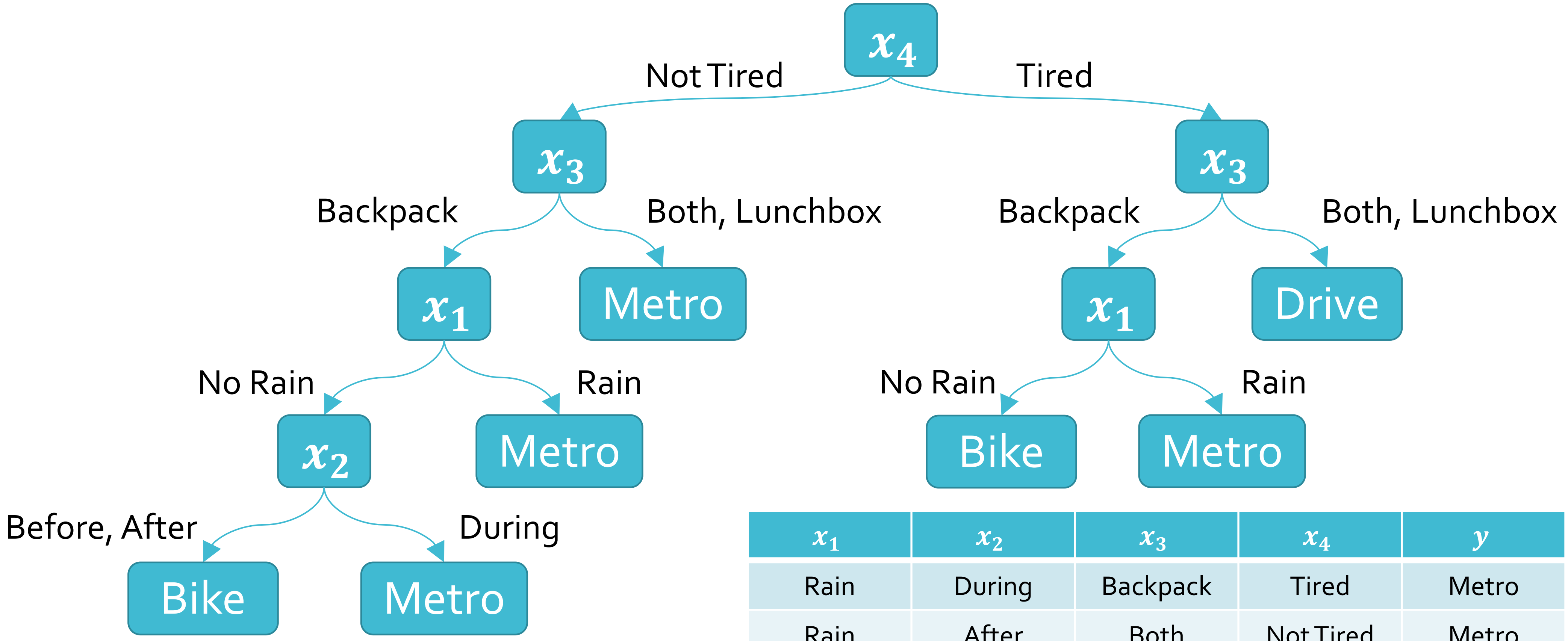


$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

Pruning: Example

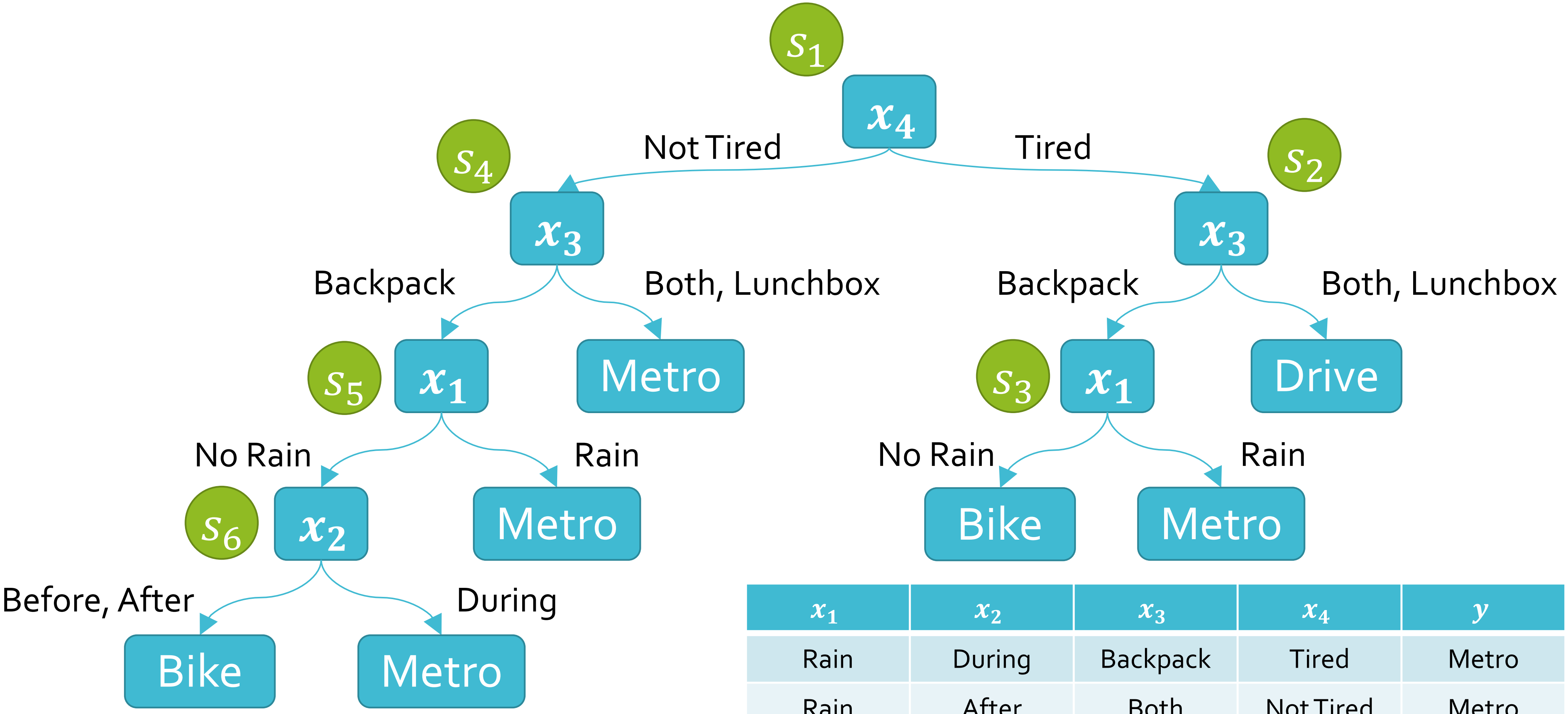
- Input: a decision tree, t and a validation dataset, $\mathcal{D}_{val}$
- Compute the validation error of t, $E_{val}(t)$
- For each split, $s \in t$
  - Compute $E_{val}(t \backslash s)$ = the validation error of t with s replaced by a leaf using the most common label at s
- If ∃ a split $s \in t$ s.t. $E_{val}(t \backslash s) \leq E_{val}(t)$, repeat the pruning process with $t \backslash s^*$ where $t \backslash s^*$ is the pruned tree with minimal validation error (shorter trees win ties)
- Output: a pruned decision tree $t \backslash s^*$



$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

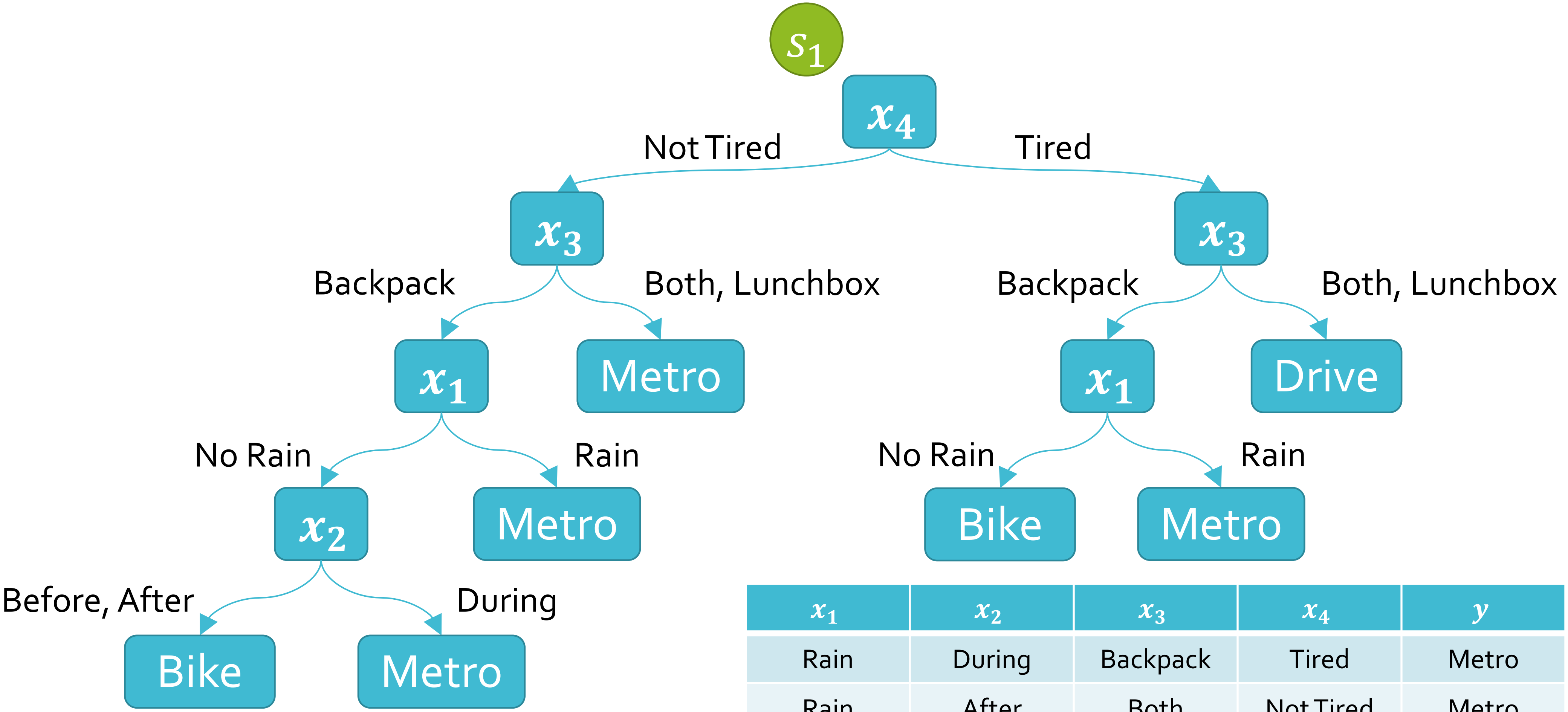
$E_{val}(t) = 0.2$



$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

$E_{val}(t) = 0.2$



$$\mathcal{D}_{val} =$$

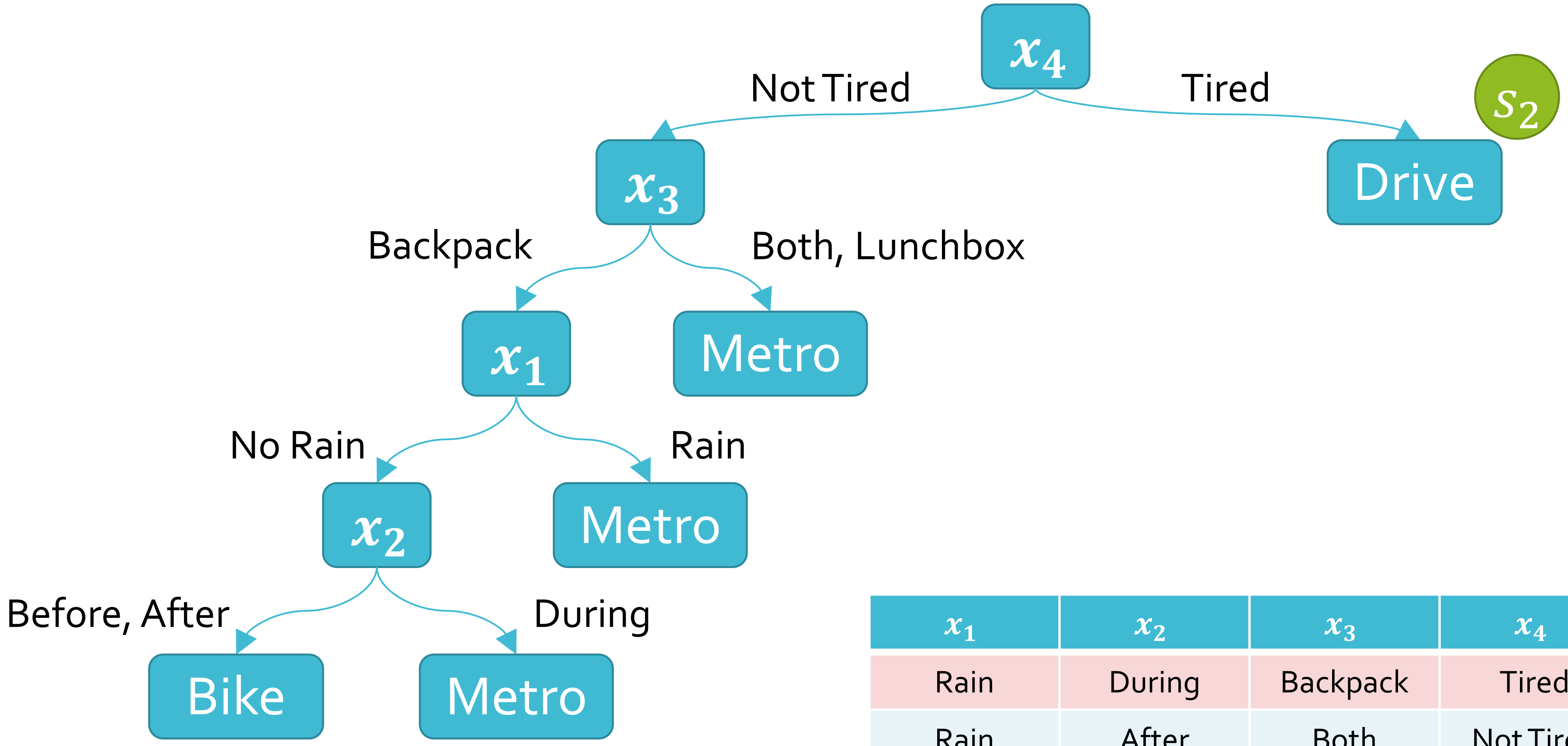
x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

$$E_{val}(t \backslash s_1)$$

$$\mathcal{D}_{val} =$$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

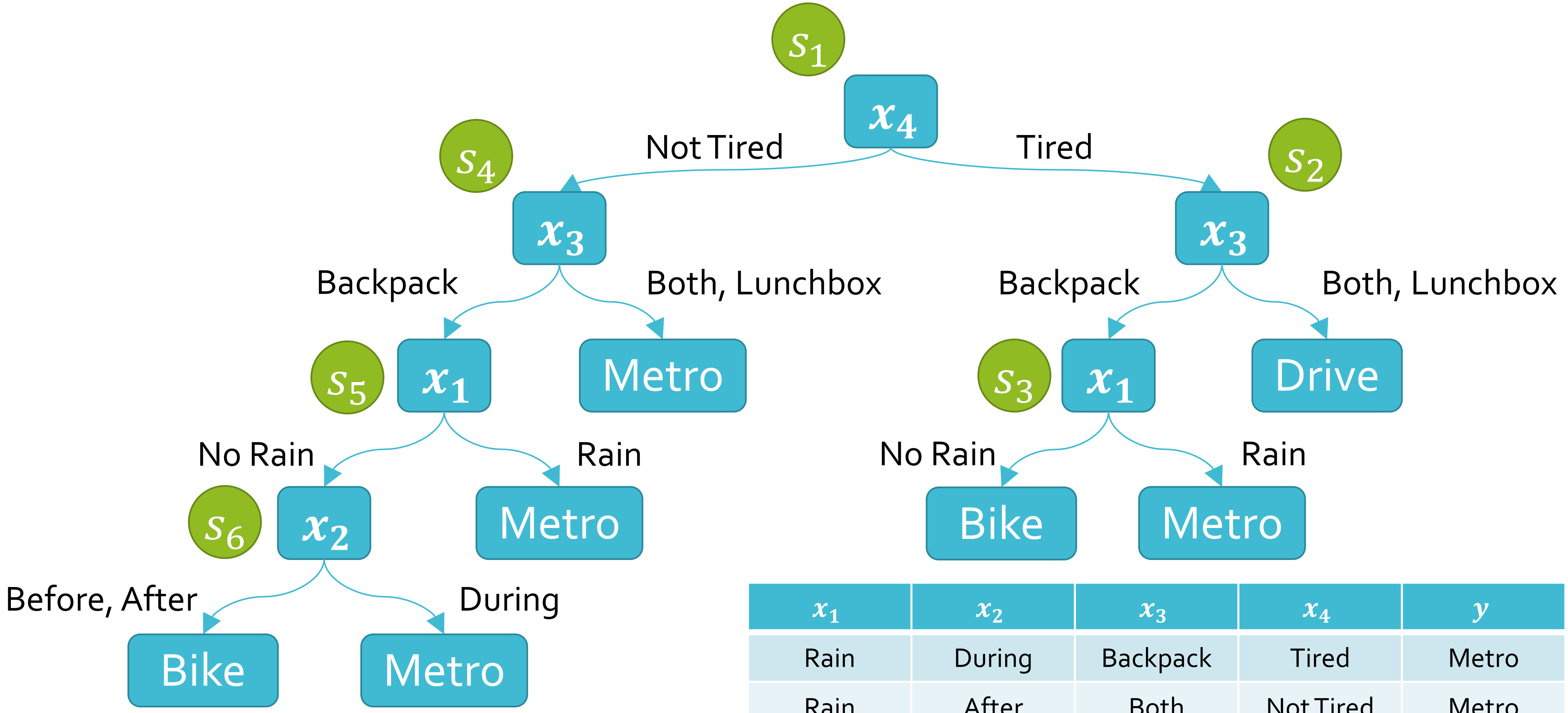
$$E_{val}(t \backslash s_1) = 0.4$$



$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

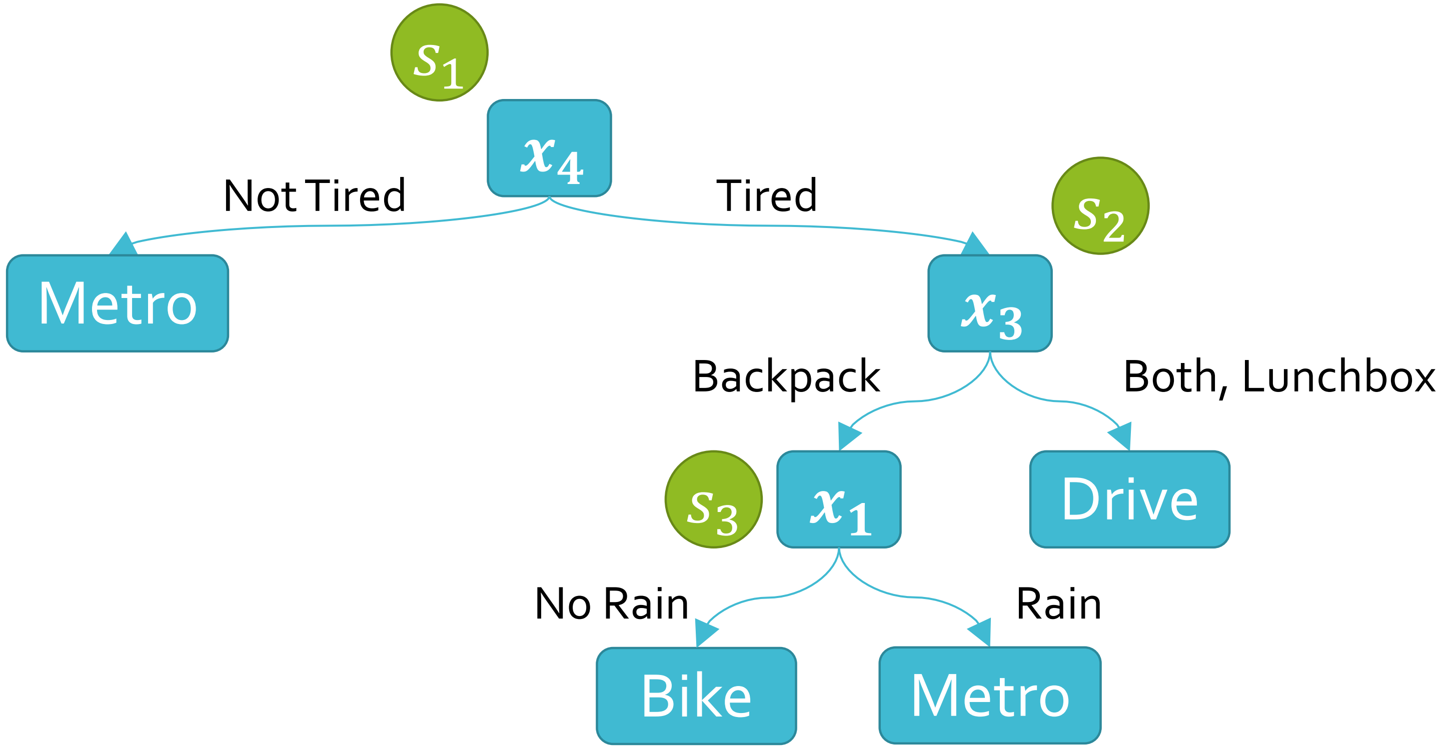
$E_{val}(t \backslash s_2) = 0.4$



	s_1	s_2	s_3	s_4	s_5	s_6
$E_{val}(t\backslash s)$	0.4	0.4	0.4	**0**	0	0.2

$\mathcal{D}_{val} =$

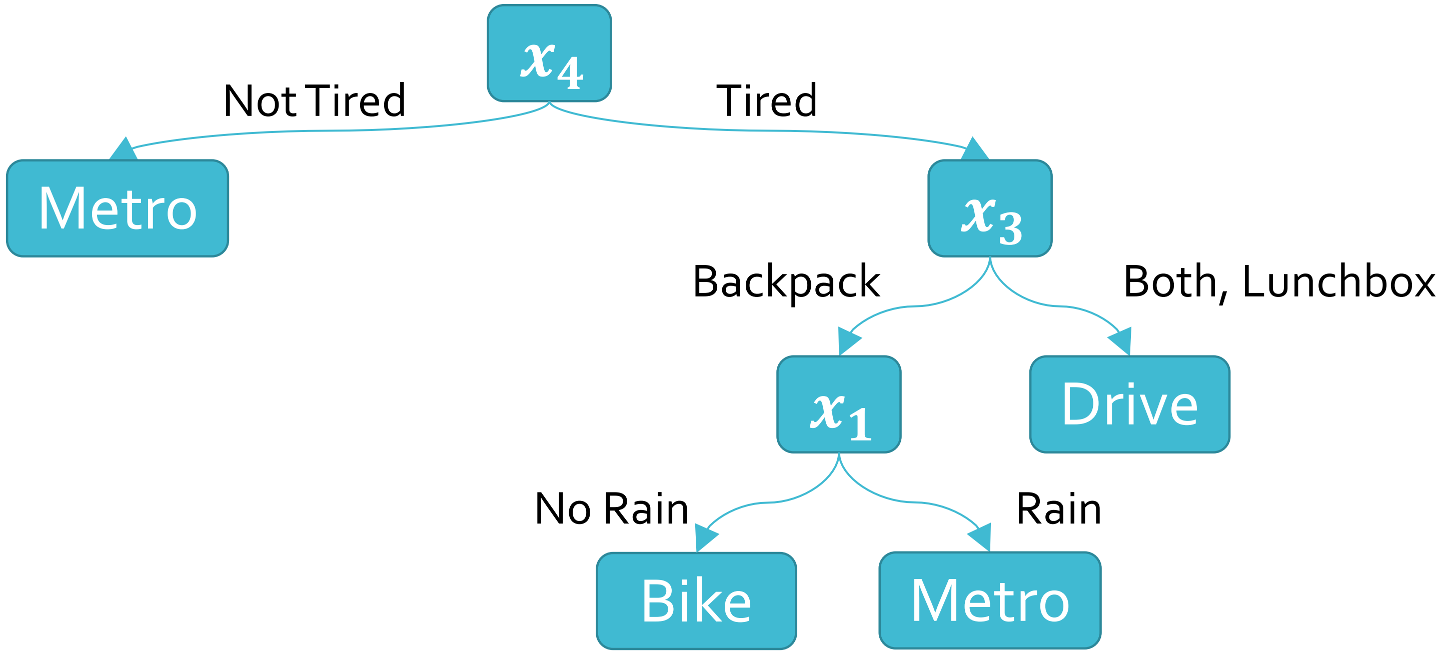
x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive



	s_1	s_2	s_3
$E_{val}(t\backslash s)$	0.4	0.2	0.2

$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive



$\mathcal{D}_{val} =$

x_1	x_2	x_3	x_4	y
Rain	During	Backpack	Tired	Metro
Rain	After	Both	Not Tired	Metro
No Rain	Before	Backpack	Not Tired	Metro
No Rain	During	Lunchbox	Tired	Drive
No Rain	After	Lunchbox	Tired	Drive

$E_{val}(t) = 0$

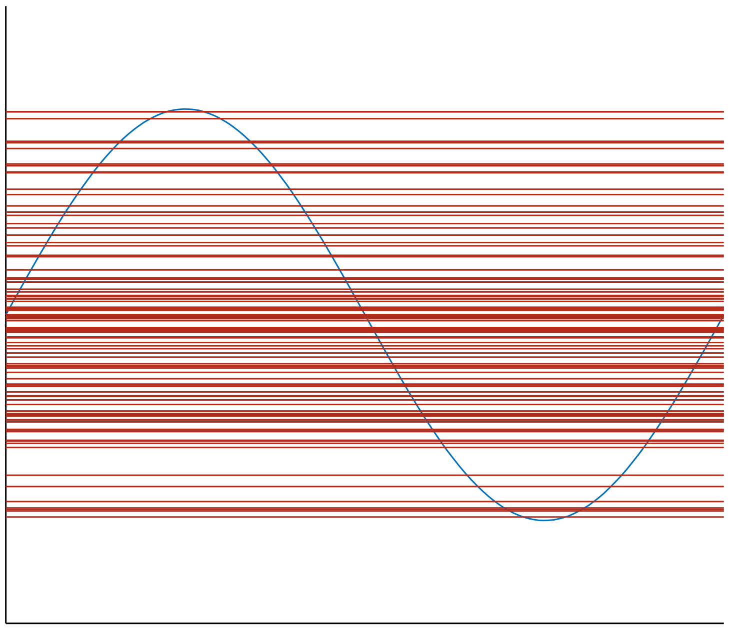
Decision Tree / ID3 Cons

- The ID3 algorithm is greedy (it selects the feature w/ the highest information gain at every step) so no optimality guarantee

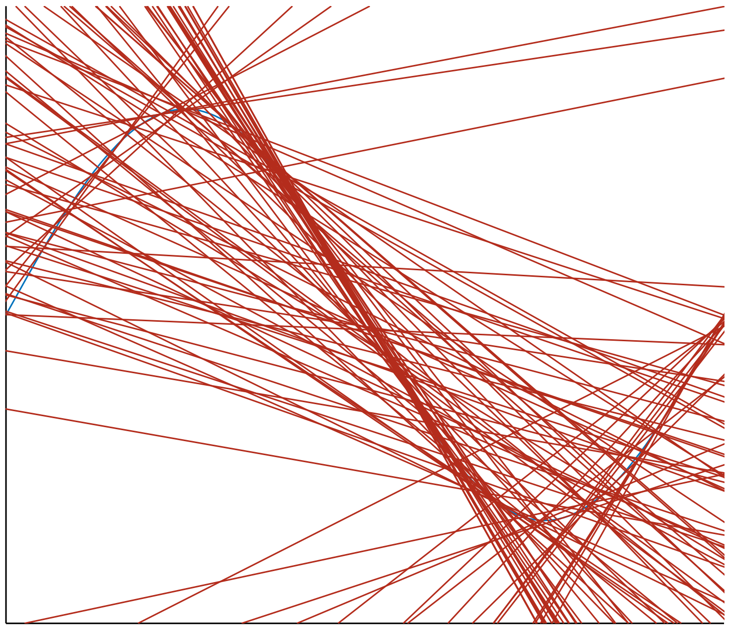
- Overfitting
  - Can be addressed via heuristics ("regularization") or pruning ("validation"):

- **High variance**

Bias-Variance Tradeoff (Example)

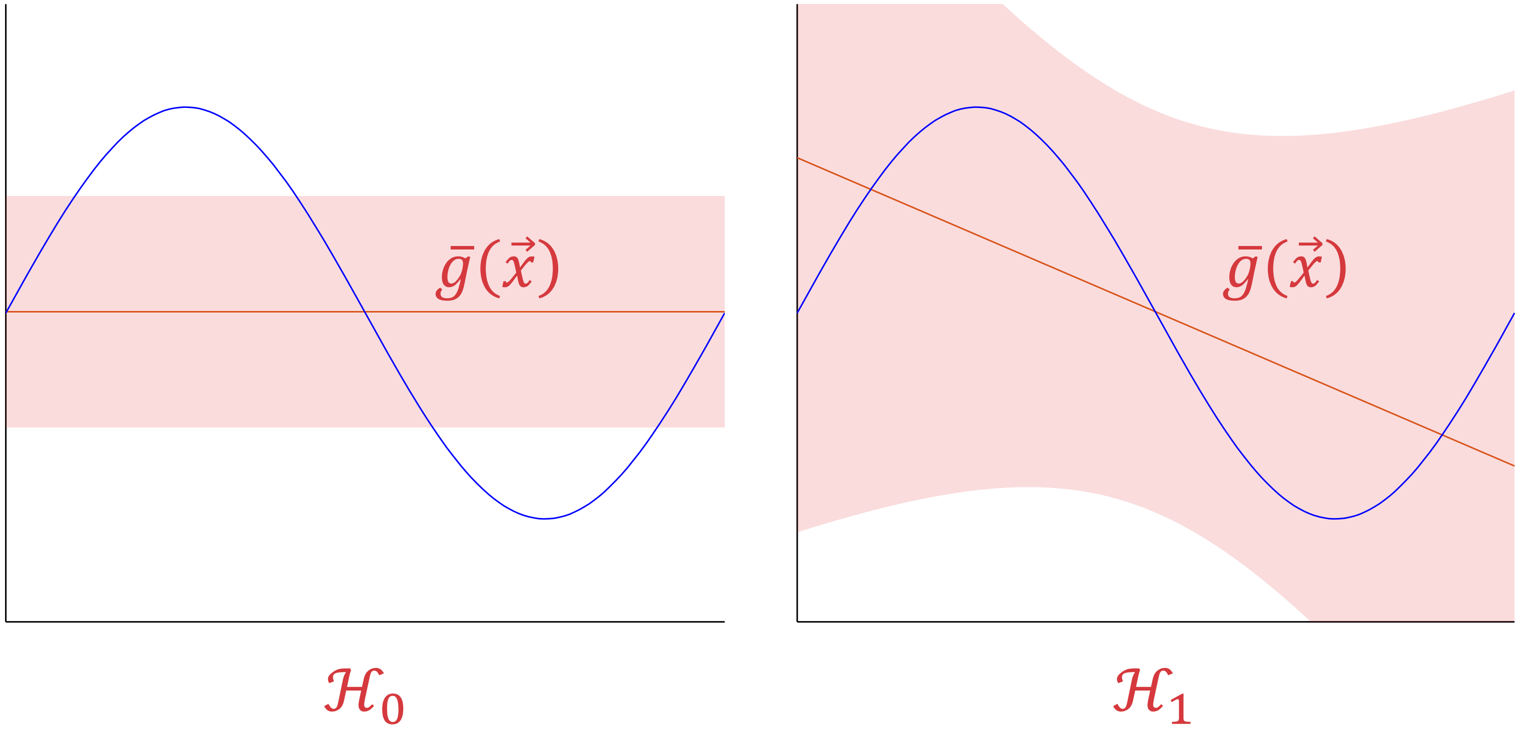


$\mathcal{H}_0$



$\mathcal{H}_1$

Bias-Variance Tradeoff (Example)



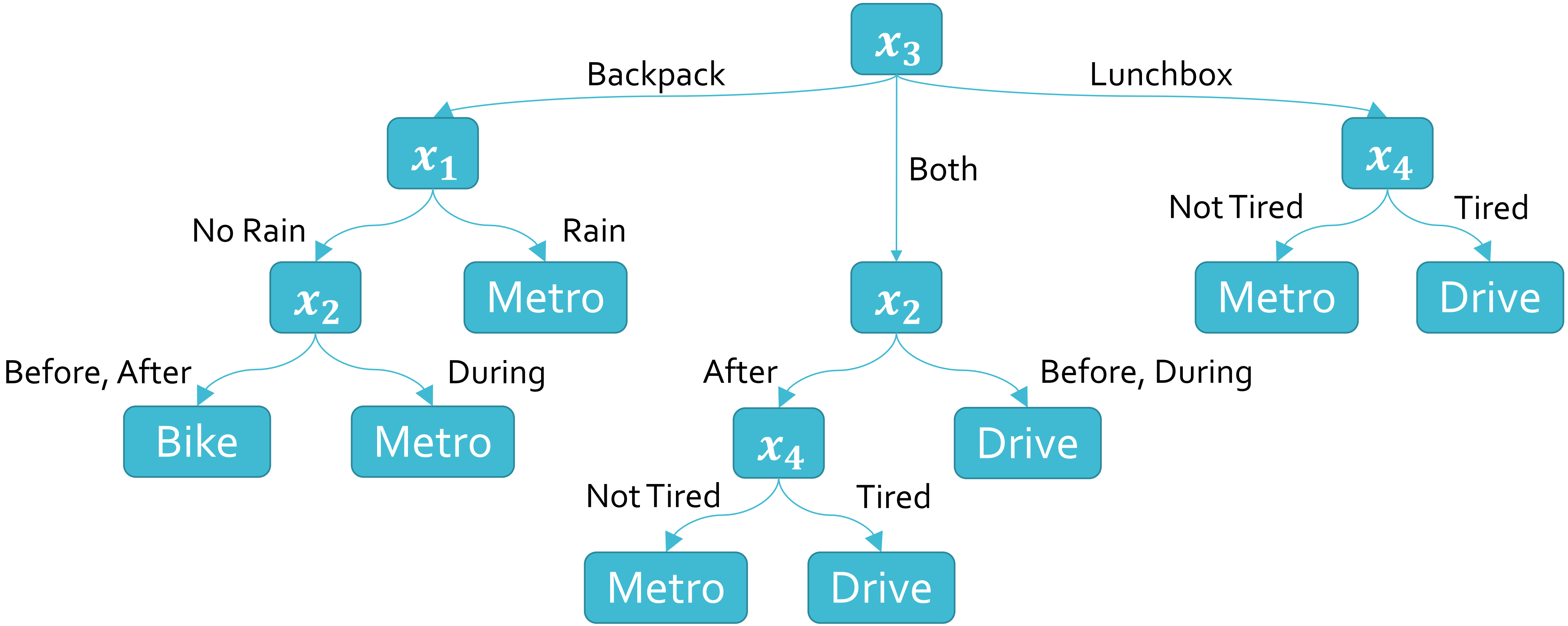
$$\mathbb{E}_{\mathcal{D}}[E_{out}(g_{\mathcal{D}})] = \mathbb{E}_{\vec{x}}[\text{Variance of } g_{\mathcal{D}}(\vec{x}) + \text{Bias of } \bar{g}(\vec{x})]$$

Data

x_1	x_2	x_3	x_4	y
Rain	Before	Both	Tired	Drive
Rain	During	Both	Not Tired	Metro
Rain	During	Both	Tired	Drive
Rain	After	Backpack	Not Tired	Metro
Rain	After	Backpack	Tired	Metro
Rain	After	Lunchbox	Tired	Drive
No Rain	Before	Backpack	Tired	Bike
No Rain	Before	Lunchbox	Not Tired	Metro
No Rain	Before	Lunchbox	Tired	Drive
No Rain	During	Backpack	Not Tired	Metro
No Rain	During	Both	Tired	Drive
No Rain	After	Backpack	Not Tired	Bike
No Rain	After	Backpack	Tired	Bike
No Rain	After	Both	Not Tired	Metro
No Rain	After	Both	Tired	Drive
No Rain	After	Lunchbox	Not Tired	Metro

Data

x_1	x_2	x_3	x_4	y
Rain	Before	Both	Tired	Drive
Rain	During	Both	Not Tired	***Drive***
Rain	During	Both	Tired	Drive
Rain	After	Backpack	Not Tired	Metro
Rain	After	Backpack	Tired	Metro
Rain	After	Lunchbox	Tired	Drive
No Rain	Before	Backpack	Tired	Bike
No Rain	Before	Lunchbox	Not Tired	Metro
No Rain	Before	Lunchbox	Tired	Drive
No Rain	During	Backpack	Not Tired	Metro
No Rain	During	Both	Tired	Drive
No Rain	After	Backpack	Not Tired	Bike
No Rain	After	Backpack	Tired	Bike
No Rain	After	Both	Not Tired	Metro
No Rain	After	Both	Tired	Drive
No Rain	After	Lunchbox	Not Tired	Metro



Decision Tree: Example

Bagging

- Short for **B**ootstrap **agg**regat**ing**

- Combines the prediction of many hypotheses to reduce variance

- If n independent random variables $x_1, x_2, \ldots, x_n$ all have variance σ^2, then the variance of $\frac{1}{n}\sum_{i=1}^{n} x_i$ is $\frac{\sigma^2}{n}$

Bootstrapping

- A statistical method for estimating properties of a distribution, given (potentially a small number of) samples from that distribution
- Relies on resampling the samples ***with replacement*** many, many times

Bootstrapping: Example

- Suppose you want to know the mean of a distribution so you draw 8 samples from that distribution: $\mathcal{D} =$
$$\{1.70, -0.23, 0.54, -0.38, -1.53, 0.84, 0.60, 1.84\}$$

- Resample 8 values (with replacement) from $\mathcal{D}$ 1000 times:
$$\{-0.23, 0.54, -0.38, -1.53, -0.23, -0.38, -0.23, -1.53\}$$
$$\{-0.38, 0.60, -0.38, 1.85, -0.38, -0.38, 1.84, -0.23\}$$
$$\vdots$$
$$\{1.84, 0.84, 1.84, -1.53, 1.84, 1.84, 1.84, -0.23\}$$

Bootstrapping: Example

- Suppose you want to know the mean of a distribution so you draw 8 samples from that distribution: $\mathcal{D} =$
$$\{1.70, -0.23, 0.54, -0.38, -1.53, 0.84, 0.60, 1.84\}$$

- Resample 8 values (with replacement) from $\mathcal{D}$ 1000 times

- Compute the mean of each new resampled set

- Use these means to build point estimates (e.g. 0.43) or confidence intervals (e.g. [-0.31, 1.12])

Aggregating

- Combining multiple hypotheses, $\{h_1, h_2, \ldots, h_m\}$, to arrive at a single hypothesis
- Regression: average the predictions $\left(\bar{h}(\vec{x}) = \frac{1}{m}\sum_{i=1}^{m} h_i(\vec{x})\right)$
- Classification: find the category that the most hypotheses predict (plurality vote)

Bagging Decision Trees

- Input: $\mathcal{D} = \{(\overrightarrow{x_1}, y_1), (\overrightarrow{x_2}, y_2), \ldots, (\overrightarrow{x_n}, y_n)\}$, B
- For $b = 1, 2, \ldots, B$
  - Create a dataset, $\mathcal{D}_b$, by sampling n points from $\mathcal{D}$ with replacement
  - Learn a decision tree, t_b, using $\mathcal{D}_b$ and the ID3 algorithm
- Output: $\bar{t}$, the aggregated hypothesis

Bagging

- Short for **B**ootstrap **agg**regat**ing**

- Combines the prediction of many hypotheses to reduce variance

- If n ***independent*** random variables $x_1, x_2, \dots, x_n$ all have variance σ^2, then the variance of $\frac{1}{n}\sum_{i=1}^{n} x_i$ is $\frac{\sigma^2}{n}$

Split-Feature Randomization

- Predictions made by trees trained on similar datasets are highly correlated
- To decorrelate these predictions, randomly limit the features available at each iteration of the ID3 algorithm
- Every time the ID3 algorithm goes to split an impure leaf, randomly select $m < d$ features and only allow the algorithm to use one of those m features.
  - For classification, a common choice is $m = \sqrt{d}$
  - For regression, a common choice is $m = \frac{d}{3}$

Random Forests

- Input: $\mathcal{D} = \{(\overrightarrow{x_1}, y_1), (\overrightarrow{x_2}, y_2), \dots, (\overrightarrow{x_n}, y_n)\}$, B, m
- For $b = 1, 2, \dots, B$
  - Create a dataset, $\mathcal{D}_b$, by sampling n points from $\mathcal{D}$ with replacement
  - Learn a decision tree, t_b, using $\mathcal{D}_b$ and the ID3 algorithm ***with split-feature randomization***
- Output: $\bar{t}$, the aggregated hypothesis

Random Forests and Validation

- For each training point, $\vec{x_i}$, there are some trees which $\vec{x_i}$ was not used to train (roughly B/e); let these trees be $t_i^- = \{t_{i,1}^-, t_{i,2}^-, \dots, t_{i,n_i}^-\}$
- Compute an aggregated prediction for each $\vec{x_i}$ using t_i^-:

$$\text{(e.g. for regression) } \bar{t}_i^-(\vec{x_i}) = \frac{1}{n_i}\sum_{j=1}^{n_i} t_{i,j}^-(\vec{x_i})$$

- Compute the out-of-bag (OOB) error:

$$E_{OOB} = \frac{1}{n}\sum_{i=1}^{n} e\left(y_i, \bar{t}_i^-(\vec{x_i})\right)$$

- E_{OOB} is almost an unbiased estimator of E_{out}

Random Forests and Feature Selection

- The interpretability of decision trees gets lost when we switch to random forests
- Random forests allow for the computation of "variable importance", a way of ranking features based on how useful they are at predicting the output
- Initialize each feature's importance to zero
- Each time a feature is chosen by the ID3 algorithm (with split-feature randomization), add that feature's information gain (relative to the split) to its importance