CS5228: Knowledge Discovery and Data Mining

Lecture 6 — Classification & Regression II

Course Logistics — Update

- ## Assignment 2
  - Submission deadline: Oct 03, 11.59 pm
  - Don't forget to check the Discussion and Errata page on Canvas

- ## Midterm
  - Check new Canvas page for midterm exam
  - Report any issues with the practice exam early enough
  - If needed, 2nd practice exam during Recess Week
  - Coming up: survey regarding request for loaner laptop

- ## Project
  - Submission deadline for progress report: Oct 10, 11.59 om

Quick Recap — Classification & Regression

- Pattern of interest
  - Matching or function between input features and output
  - Goal: use matching to predict outputs for unseed samples
  - Categorical output ➜ classification
  - Numerical output ➜ regression

- Important: Evaluation of predictions
  - Straightforward for regression
  - Series of metrics for classification
    (accuracy, recall, precision, f1, AUC-ROC)

Age	Edu-cation	Marital Status	Income Level	Credit Approval	Credit Limit
23	Masters	Single	Mid	No	$S5,000
35	College	Married	High	Yes	$S7,000
26	Masters	Single	High	No	$S9,000
41	PhD	Single	Mid	Yes	$S5,000
18	Poly	Single	Low	No	$S6,000
55	Poly	Married	High	Yes	$S10,000
30	College	Single	High	Yes	$S8,000
35	PhD	Married	High	Yes	$S10,000
28	Masters	Married	Mid	Yes	$S5,000
45	Masters	Married	Mid	???	???

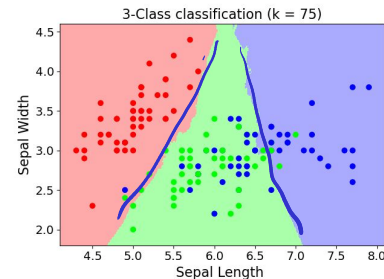# Quick Recap — KNN Algorithm
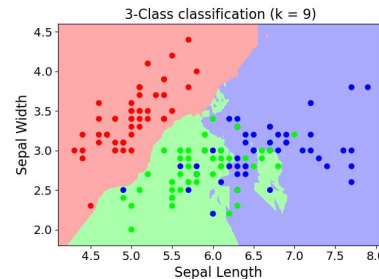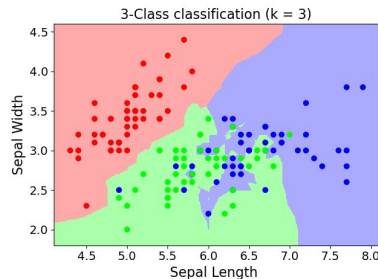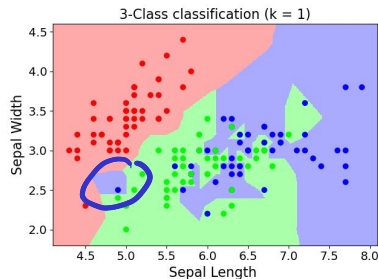
- Intuition behind KNN:
  - Label of an unseen data point *x* derives from the labels of the k-nearest neighbors of *x*
  - Similar data points ➜ similar labels
  - Caveats due to reliance of similarity metric
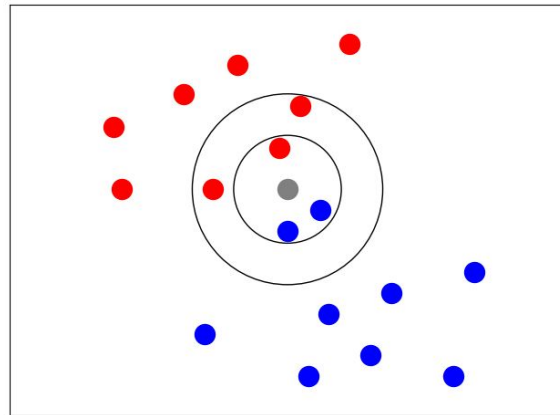
- Effects of hyperparameter k
  - Tradeoff between (risks of) underfitting and overfitting

Outline

- **Decision Trees**
  - **Overview**
 - Training Decision Trees
  - Overfitting

- Tree Ensembles
  - Bagging
  - Random Forest
  - Boosting

Decision Tree

- Example: Decision Tree for classification

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Masters	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes
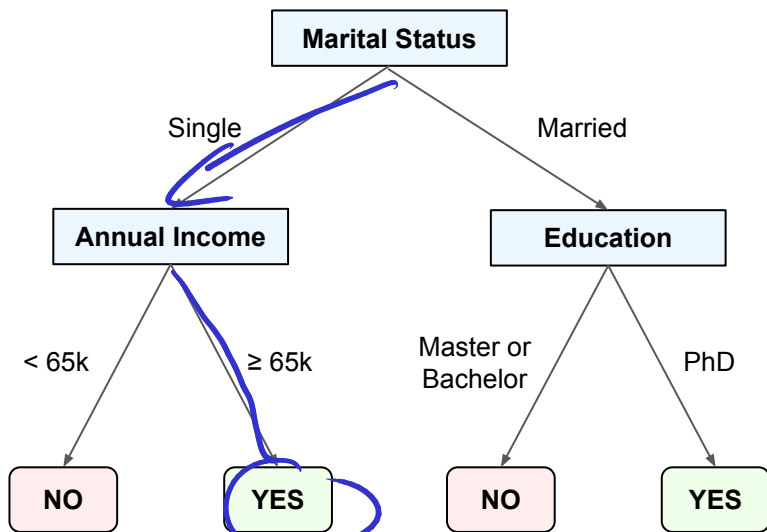
Decision Tree
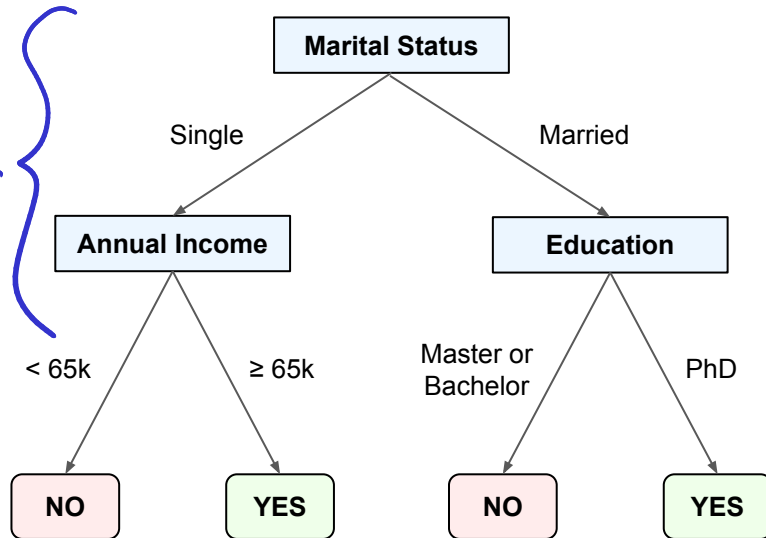
- Decision Tree — idea
  - Represent mapping between features and label/value as flowchart-like structure

- Components (a bit simplified at the moment)
  - (Inner) node — test on a single feature

  - Branch — outcome of a test; corresponds to a feature values or range of values

  - Leaf — label (classification) or real value (regression)

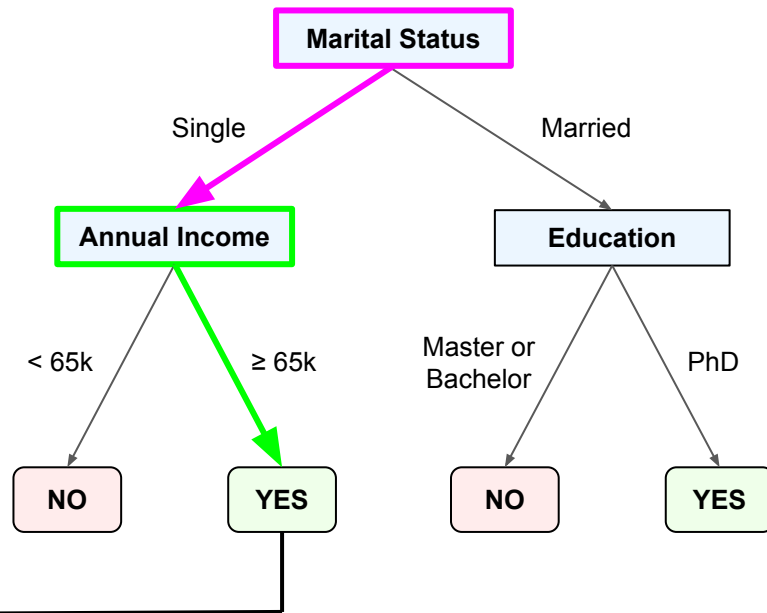# Decision Tree — Application to Unseen Data

Decision Tree

- Same dataset, different Decision Tree
  - In general, there are multiple trees that match a dataset

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
2	35	Bachelor	Married	50k	No
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
5	18	Bachelor	Single	40k	No
6	55	Masters	Married	85k	No
7	30	Bachelor	Single	60k	No
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

Education

Bachelor → NO

Master, PhD → Age

Age: < 50 → YES

Age: ≥ 50 → NO

Which Decision Tree is Better?
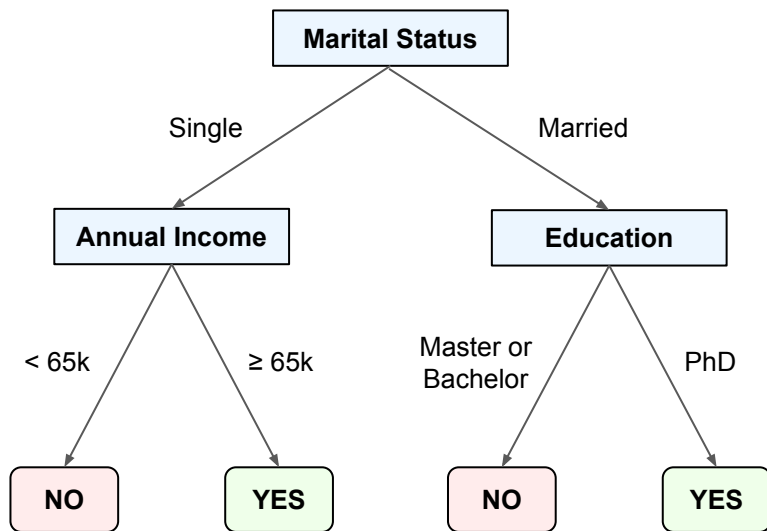
Left tree:

Marital Status
- Single → Annual Income
  - < 65k → NO
  - ≥ 65k → YES
- Married → Education
  - Master or Bachelor → NO
  - PhD → YES

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
50	PhD	Single	70k	**Yes**

Right tree:

Education
- Bachelor → NO
- Master, PhD → Age
  - < 50 → YES
  - ≥ 50 → NO
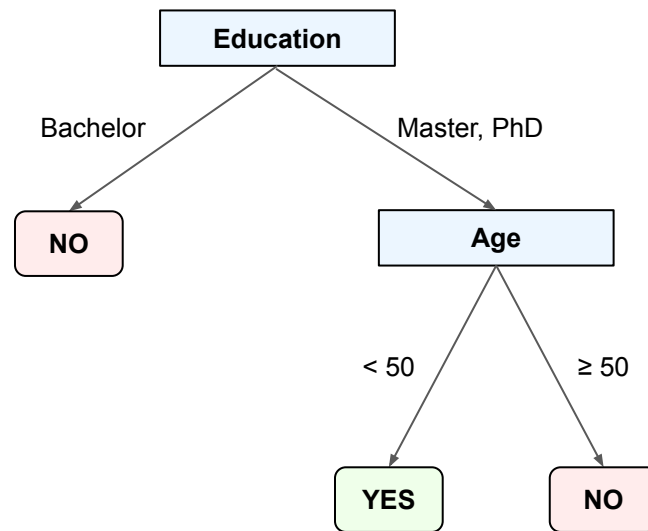
Age	Edu-cation	Marital Status	Annual Income	Credit Approval
50	PhD	Single	70k	**NO**

Quick Quiz

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
50	PhD	Married	70k	???

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
50	Poly	Single	70k	???

What to do in case
of **unknown values**
for a feature?
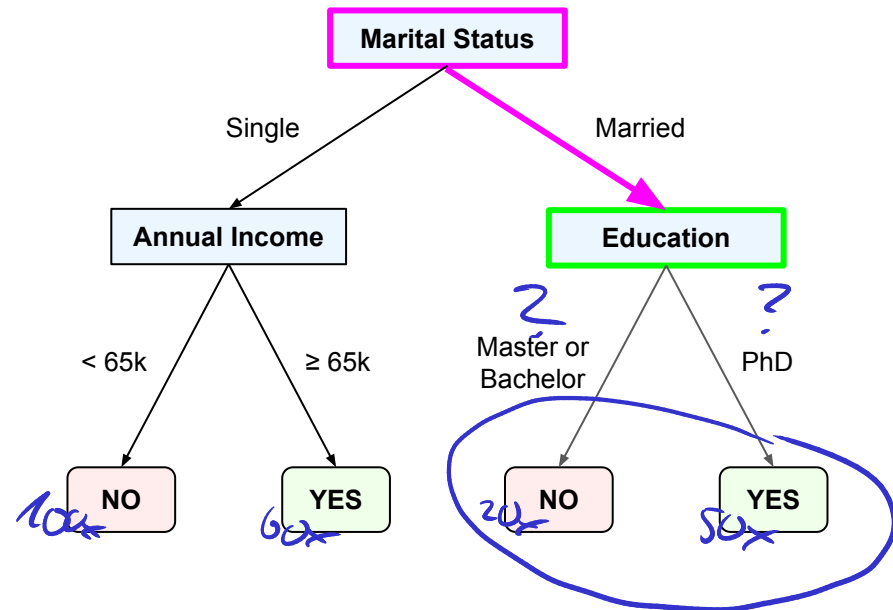
Decision Tree — Diversity (Sneak Preview)

- Different branching factors
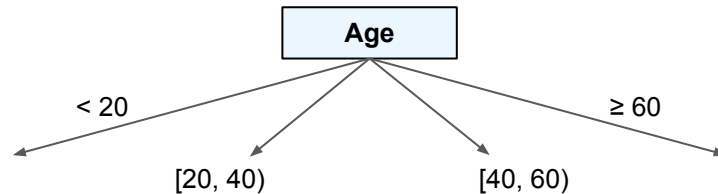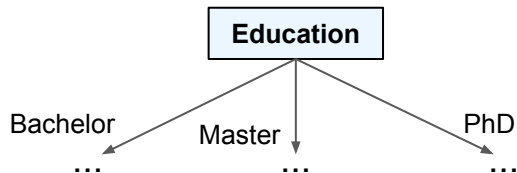
- Different depth
  - Leaves can have more than one label or real value

  - Required based on dataset or based on choice (➜ Pruning)

  - Final output: majority label (classification) or mean of values (regression)
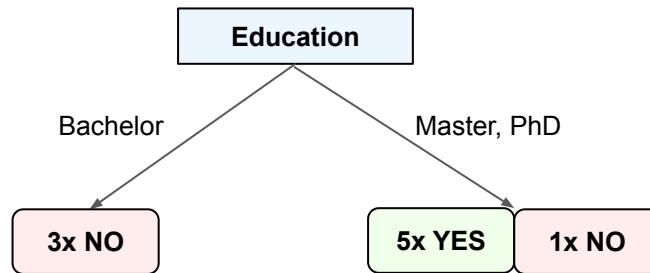
Outline

- **Decision Trees**
  - Overview
  - **Training Decision Trees**
  - Overfitting

- Tree Ensembles
  - Bagging
  - Random Forest
  - Boosting

Building a Decision Tree

- **Notations**
  - D_t — set of records that reach node t
  - D_0 — set of all records at root node

- **General procedure**
  - If $|D_t| = 1$ or all records in D_t have the same class or value ➜ t is leaf node
  - Otherwise, choose test (feature + conditions) to split D_t into smaller subsets (i.e., subtrees)
  - Recursively apply procedure to each subtree

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
2	35	Bachelor	Married	50k	No
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
5	18	Bachelor	Single	40k	No
6	55	Masters	Married	85k	No
7	30	Bachelor	Single	60k	No
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

all samples

1,2,3,4,6,7,8,9

Education

Bachelor → 2,5,7

Master, PhD → 1,3,4,6,8,9

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
2	35	Bachelor	Married	50k	No
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
5	18	Bachelor	Single	40k	No
6	55	Masters	Married	85k	No
7	30	Bachelor	Single	60k	No
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

2,5,7 ?   1,3,4,6,8,9

split further

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
2	35	Bachelor	Married	50k	No
5	18	Bachelor	Single	40k	No
7	30	Bachelor	Single	60k	No

1,2,3,4,6,7,8,9

Education

Bachelor → 2,5,7

Master, PhD → 1,3,4,6,8,9

All the same labels ➜ leaf node

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
2	35	Bachelor	Married	50k	No
5	18	Bachelor	Single	40k	No
7	30	Bachelor	Single	60k	No

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

2,5,7   1,3,4,6,8,9 **?**

NO

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
6	55	Masters	Married	85k	No
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

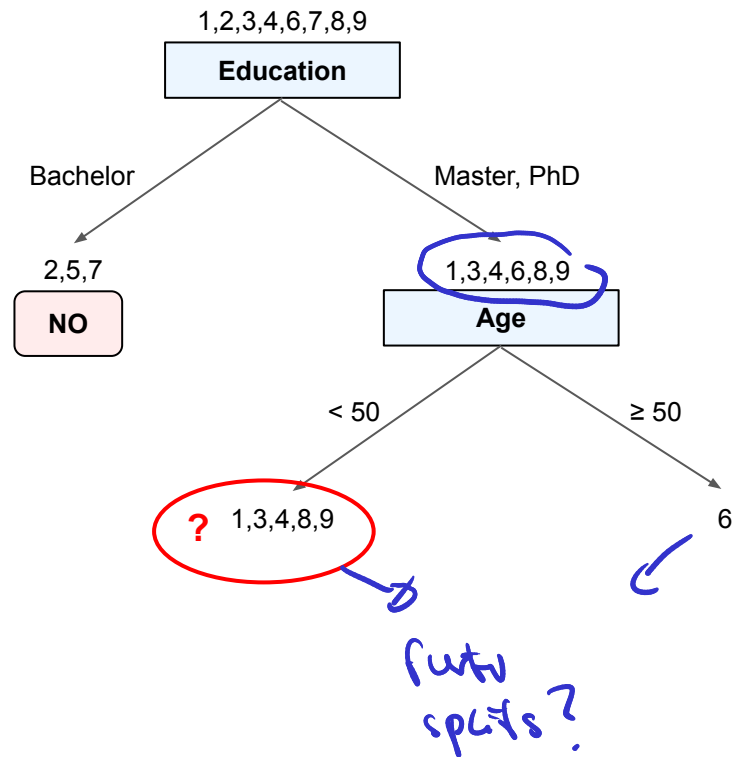2,5,7   1,3,4,6,8,9 **?**

NO

Different labels ➜ split set of records

Here: select feature "age" and threshold "50"
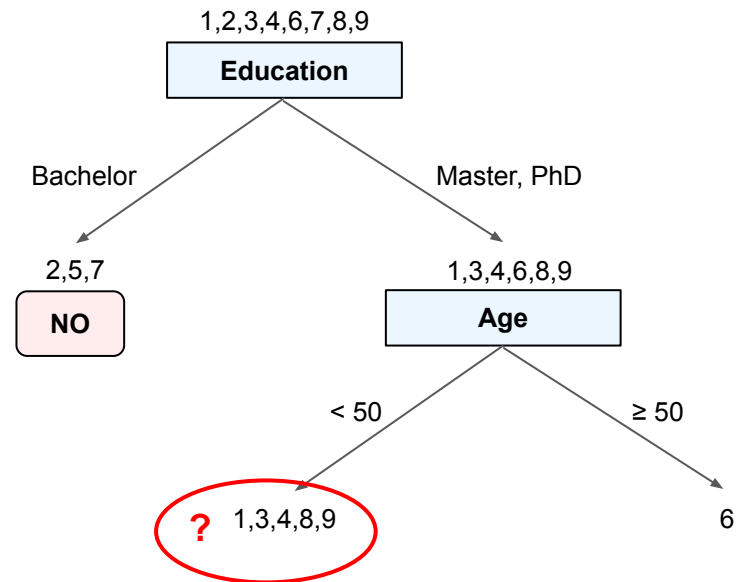(this selection process is the core of DTL and will be defined later)

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
6	55	Masters	Married	85k	No
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

All the same labels ➜ leaf node

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

2,5,7   1,3,4,6,8,9

NO   **Age**

< 50   ≥ 50

? 1,3,4,8,9   6

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
1	23	Masters	Single	75k	Yes
3	26	Masters	Single	70k	Yes
4	41	PhD	Single	95k	Yes
8	35	PhD	Married	60k	Yes
9	28	PhD	Married	65k	Yes

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

2,5,7   1,3,4,6,8,9

NO   **Age**

< 50   ≥ 50

1,3,4,8,9   ? 6

YES

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval
6	55	Masters	Married	85k	No

Only one record ➜ leaf node

1,2,3,4,6,7,8,9

Education

Bachelor   Master, PhD

2,5,7   1,3,4,6,8,9

NO   **Age**

                    < 50   ≥ 50

              1,3,4,8,9   **?** 6

              YES

Building a Decision Tree — Step-by-Step Example

ID	Age	Edu-cation	Marital Status	Annual Income	Credit Approval

All records covered ➜ Done!

1,2,3,4,6,7,8,9

Education

Bachelor

Master, PhD

2,5,7

NO

1,3,4,6,8,9

Age

< 50

≥ 50

1,3,4,8,9

YES

6

NO
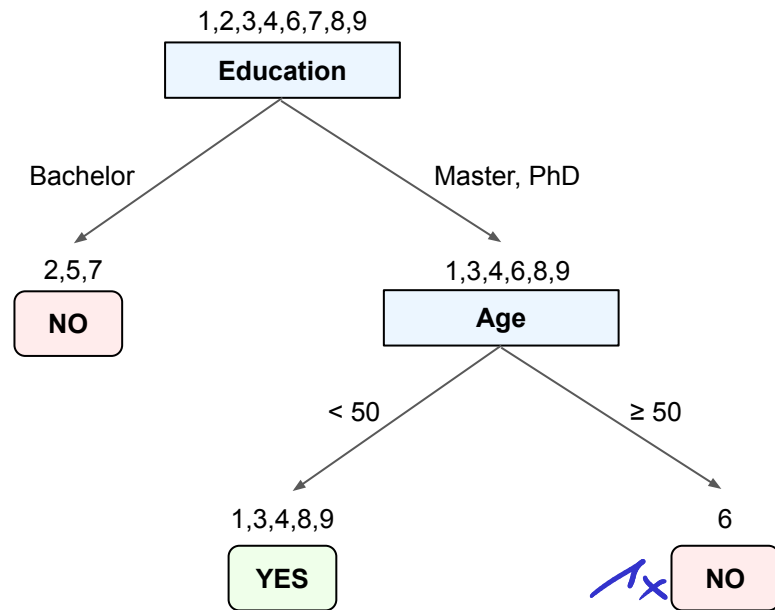
How to Split? — Nominal Attributes

- ● Binary split
  - ■ Partition all *d* values into two subsets
  - ■ $\dfrac{2^d - 2}{2}$ possible splits

Education — Bachelor, Master | PhD — ... ...

OR

Education — Bachelor, PhD | Master — ... ...

OR

Education — Bachelor | Master, PhD — ... ...
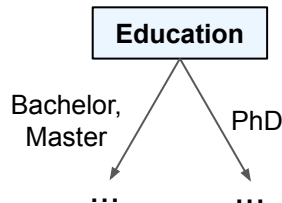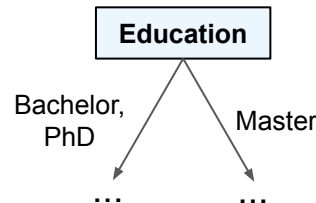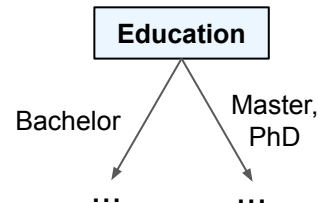
- ● Multiway split
  - ■ Each value yields on subtree
  - ■ In principle, arbitrary splits into $2 \leq s \leq d$ subtrees possible, but number of possible splits explodes

Education — Bachelor | Master | PhD — ... ...
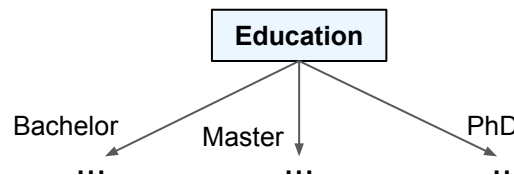
How to Split? — Ordinal Attributes

- **Binary split**
  - Partition all *d* values into two subsets
  - Partitions must preserve natural order of values
  - $d - 1$ possible splits
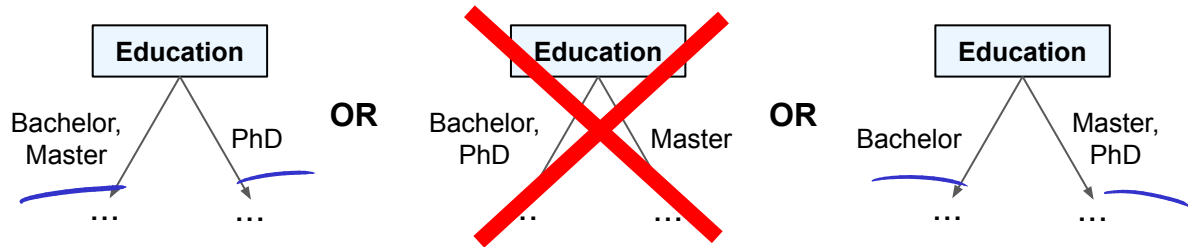
- **Multiway split**
  - Each value yields on subtree
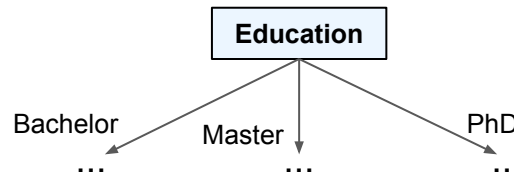
> **Note:** Whether "Education" is treated as nominal or ordinal feature is up to interpretation and a design choice of the user
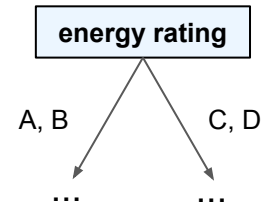
How to Split? — Numerical Values

- Binary split

Age

< 50   ≥ 50

…   …
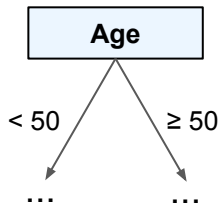
- Multiway split

Age

< 25   [25, 38)   ≥ 38

…   …   …
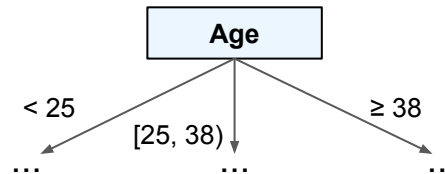
Age
23
35
26
41
18
55
30
35
28

Possible thresholds

18   23   26   28   30   35   41   55
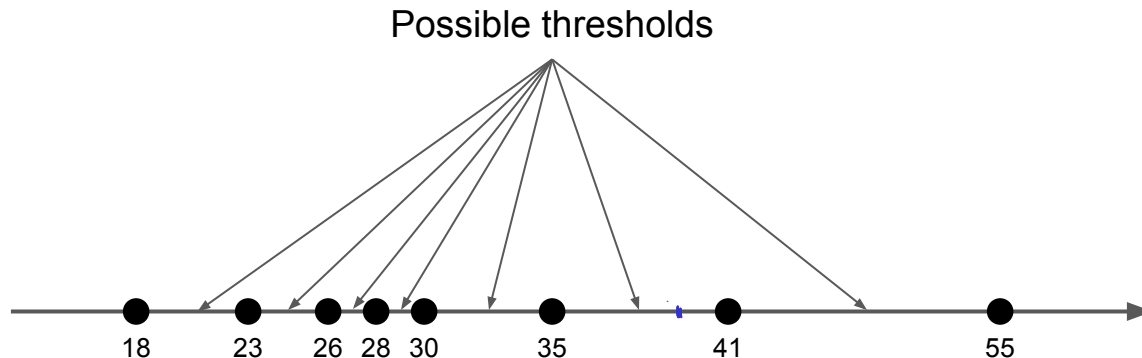
Quick Quiz

For the **energy rating** of a building,
what is generally **not** a suitable split?

A

energy rating

A, B | C, D
… | …

B ✓

energy rating

A, D | B, C
... | ...

C

energy rating

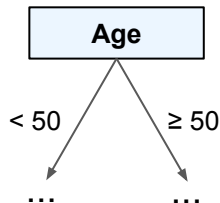A | B, C, D
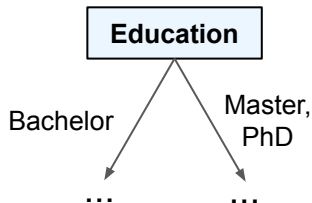… | …

D

energy rating
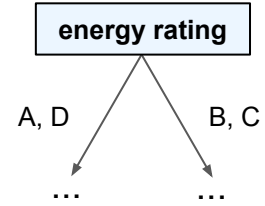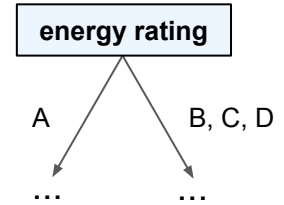
A, B, C | D
… | …
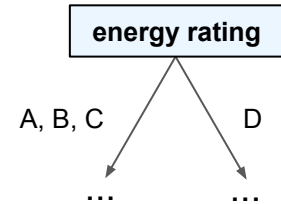
Finding the Best Splits?

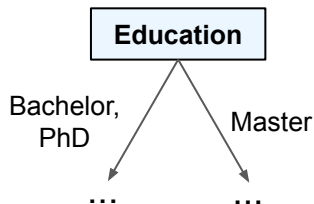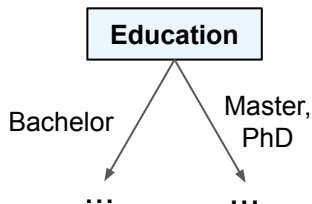- Which feature to use for splitting the training records

vs.

Why choose **Age** over **Education**, or vice versa?

- How to split the w.r.t. a selected feature?

vs.

What makes **{B,M}/{P}** a better split then **{B}/{MP}**, or vice versa — or any other alternative way?
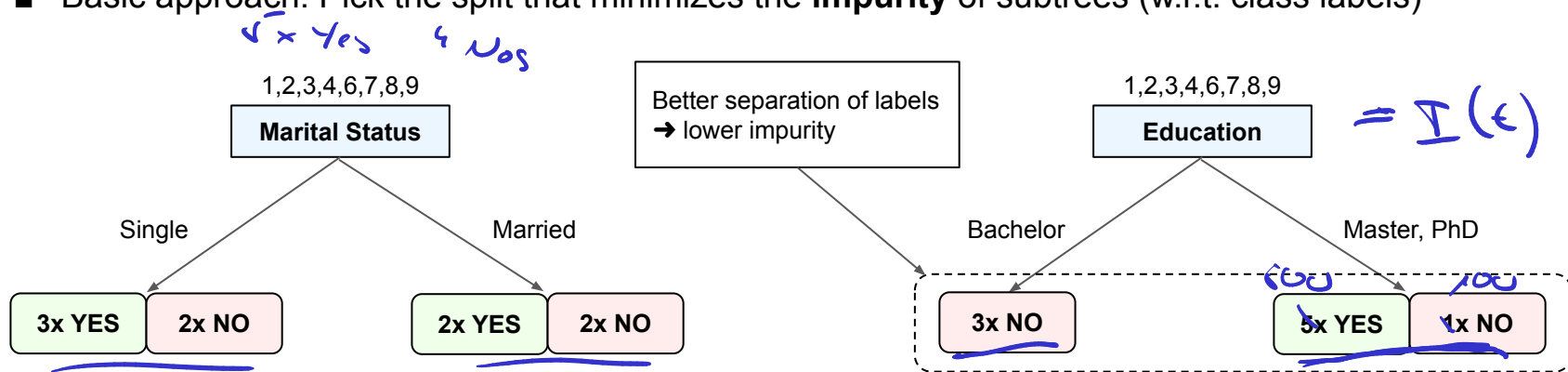
Finding the Best Splits

- ## Global optimum
  - Best splits = splits that result in a Decision Tree with the highest accuracy
  - Problem: Finding the optimal tree is NP-complete ➜ not practical for large datasets

- ## Greedy approach
  - Fast(er) heuristics but no guarantees for optimal results
  - Basic approach: Pick the split that minimizes the **impurity** of subtrees (w.r.t. class labels)

Finding the Best Splits

- **General procedure**
  - Calculate impurity $I(t)$ of node t before splitting

  - For each possible / considered split, calculate impurity of split I_{split}
    (weighted average of impurities of resulting child nodes)

  - Select split with lowest impurity I_{split}

  - Perform split if $I_{split} < I(t)$ — (not necessarily always the case)

Impurity of a Node (Classification)

Gini Index

Entropy

$$Gini(t) = 1 - \sum_{c \in C} P(c|t)^2 \qquad\qquad Entropy(t) = - \sum_{c \in C} P(c|t) \log P(c|t)$$

perfectly pure

0x YES	6x NO

$$1 - (1.0^2 + 0^2) = 0 \qquad\qquad -(0 \log_2 0 + 1 \log_2 1) = 0$$

4x YES	2x NO

$$1 - ((4/6)^2 + (2/6)^2) = 0.44 \qquad\qquad -(4/6 \log_2 4/6 + 2/6 \log_2 2/6) = 0.92$$

$P(c|t)$ = relative frequency of class c in node t

Impurity of a Node (Classification)

- Gini Index vs. Entropy for 2-class problem
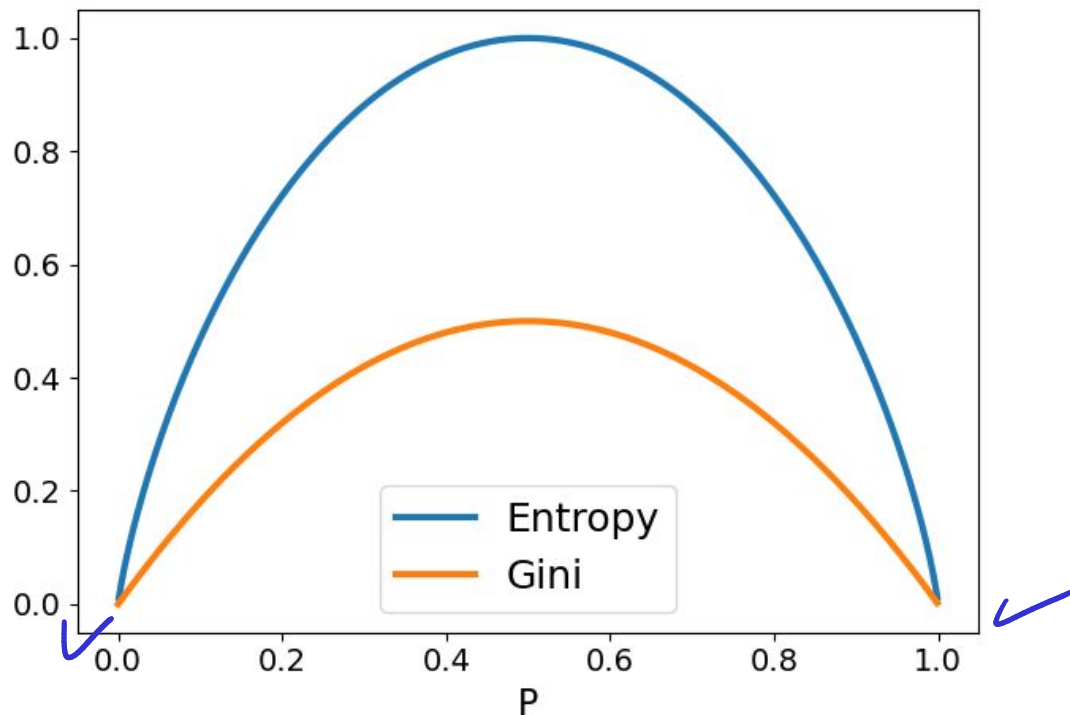
Impurity of a Split (Classification)

- Assume node t is split into k children
  - n_i — number of records at i-th child
  - n — number of records at node t
  - $I(i)$ — impurity of node (e.g., Gini, Entropy)
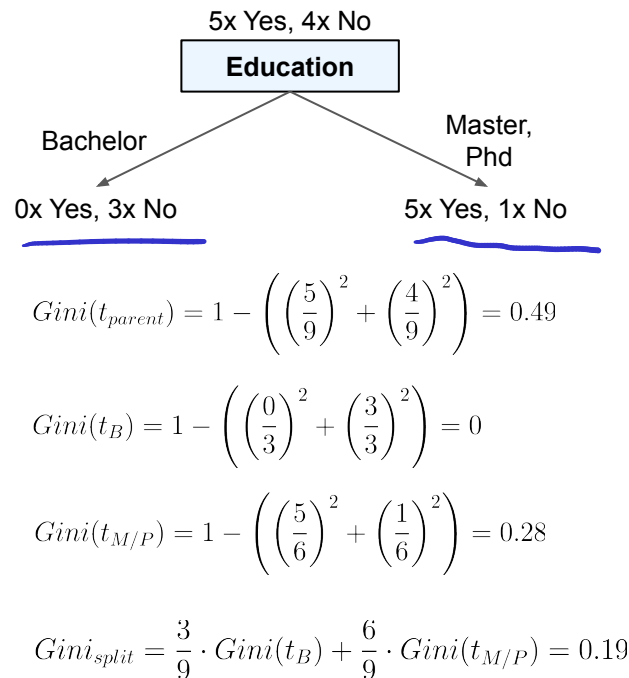
- **Information Gain** IG
  - Difference between $I(t)$ and I_{split}
  - Choose split that minimizes I_{split} = split that maximizes IG
  - Required condition: $IG > 0$

Sum of impurity values of all children, weighted by the number of records at each child.

$$I_{split} = \sum_{i}^{k} \frac{n_i}{n} I(i)$$

$$IG = I(t) - I_{split}$$

Impurity of Split (Classification) — Example

5x Yes, 4x No

Marital Status

Single   Married

3x Yes, 2x No   2x Yes, 2x No

$Gini(t_{parent}) = 1 - \left(\left(\frac{5}{9} \right)^2 + \left(\frac{4}{9} \right)^2 \right) = 0.49$

$Gini(t_{single}) = 1 - \left(\left(\frac{3}{5} \right)^2 + \left(\frac{2}{5} \right)^2 \right) = 0.48$

$Gini(t_{married}) = 1 - \left(\left(\frac{2}{4} \right)^2 + \left(\frac{2}{4} \right)^2 \right) = 0.5$

$Gini_{split} = \frac{5}{9} \cdot Gini(t_{single}) + \frac{4}{9} \cdot Gini(t_{married}) = 0.49$

5x Yes, 4x No

Education

Bachelor   Master, Phd

0x Yes, 3x No   5x Yes, 1x No

$Gini(t_{parent}) = 1 - \left(\left(\frac{5}{9} \right)^2 + \left(\frac{4}{9} \right)^2 \right) = 0.49$

$Gini(t_B) = 1 - \left(\left(\frac{0}{3} \right)^2 + \left(\frac{3}{3} \right)^2 \right) = 0$

$Gini(t_{M/P}) = 1 - \left(\left(\frac{5}{6} \right)^2 + \left(\frac{1}{6} \right)^2 \right) = 0.28$

$Gini_{split} = \frac{3}{9} \cdot Gini(t_B) + \frac{6}{9} \cdot Gini(t_{M/P}) = 0.19$
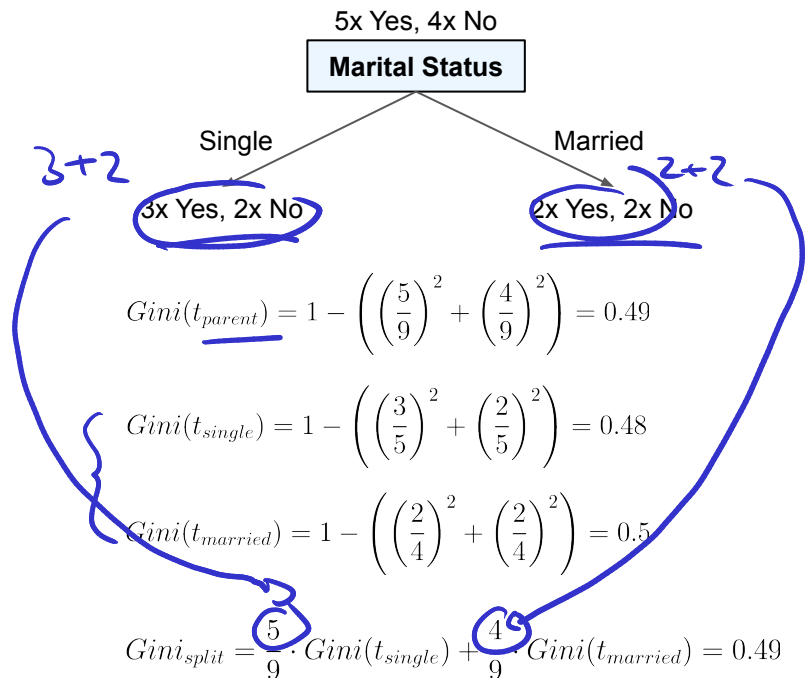
$IG = Gini(t_{parent}) - Gini_{split} = 0$
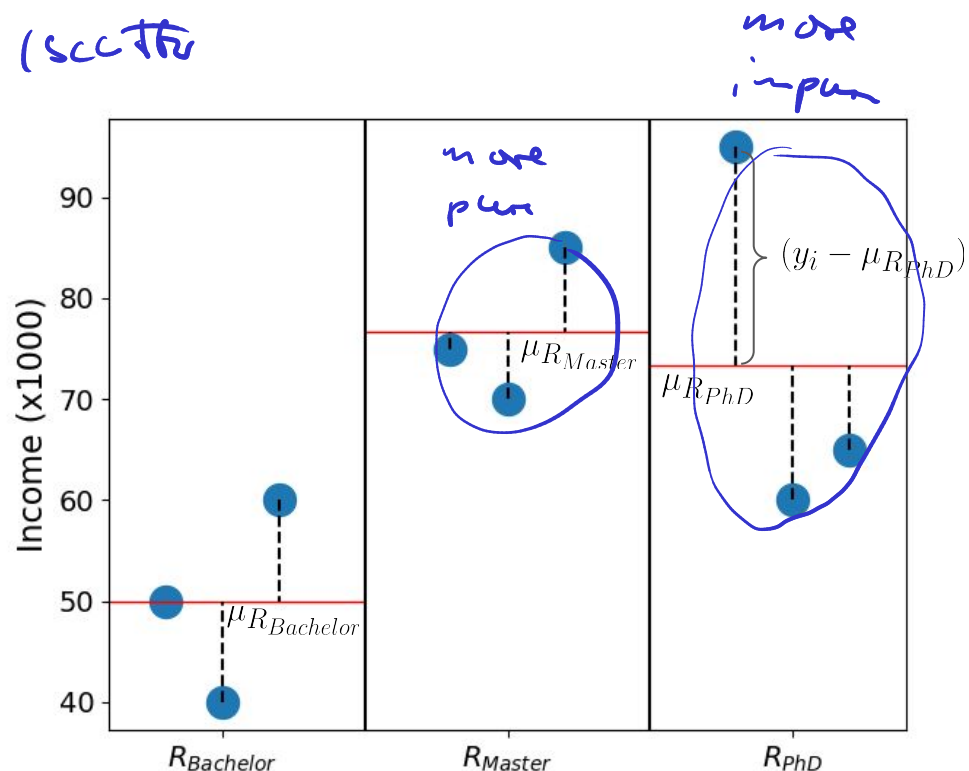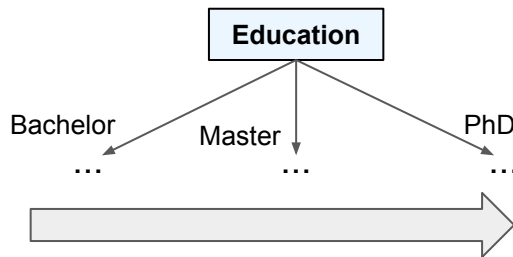
$IG = Gini(t_{parent}) - Gini_{split} = 0.3$

Impurity of a Split (Regression)

Residual Sum of Squares (RSS)

$$RSS_{split} = \sum_{k=1}^{K} \sum_{i \in R_k} (y_i - \mu_{R_k})^2$$

variance (scatter) *more impure*

Edu-cation	Annual Income
Masters	75k
Bachelor	50k
Masters	70k
PhD	95k
Bachelor	40k
Master	85k
Bachelor	60k
PhD	60k
PhD	65k

Education

Bachelor   Master   PhD
...   .

35

Impurity of a Split (Regression)

$$\mu_{R_{Bachelor}} = 50$$

$$\mu_{R_{Master}} = 76.67$$

$$\mu_{R_{PhD}} = 73.33$$

Education

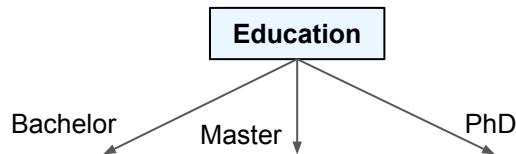Bachelor   Master   PhD

$$RSS_{split} = \sum_{k=1}^{K} \sum_{i \in R_k} (y_i - \mu_{R_k})^2$$

$$= \sum_{i \in R_{Bachelor}} (y_i - \mu_{R_{Bachelor}})^2 + \sum_{i \in R_{Master}} (y_i - \mu_{R_{Master}})^2 + \sum_{i \in R_{PhD}} (y_i - \mu_{R_{PhD}})^2$$

$$= \quad (50 - 50)^2 + (40 - 50)^2 + (60 - 50)^2$$

$$+ (75 - 76.67)^2 + (70 - 76.67)^2 + (85 - 76.67)^2$$

$$+ (95 - 73.33)^2 + (60 - 73.33)^2 + (65 - 73.33)^2$$

$$= 1033.34$$

Edu-cation	Annual Income
Masters	75k
Bachelor	50k
Masters	70k
PhD	95k
Bachelor	40k
Master	85k
Bachelor	60k
PhD	60k
PhD	65k

# Decision Trees — Pros & Cons

- Pros
  - Inexpensive to train and test

  - Easy to interpret (if tree is not too large)

 - Can handle categorical and numerical data

- Cons
  - Sensitive to small changes in the training data
    (Hierarchical structure: errors early on propagate down)

  - Greedy approach does not guarantee optimal tree

  - Each decision involves only a single feature

  - Does not take interactions between features into account

Decision Trees — Interaction between Features

Optimal decision boundary
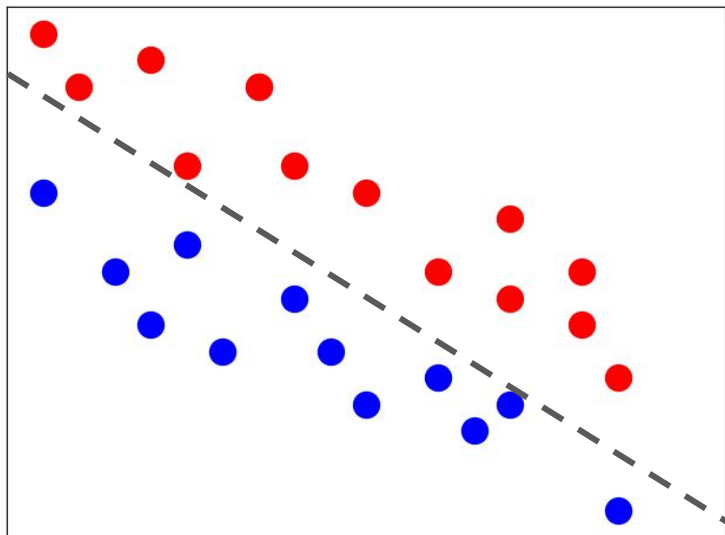
Decision Tree boundaries
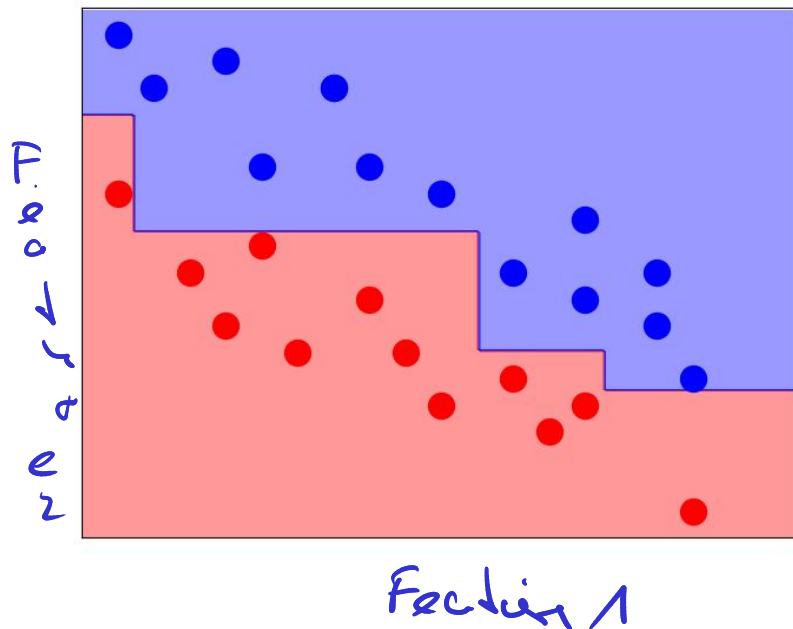
Feature 2
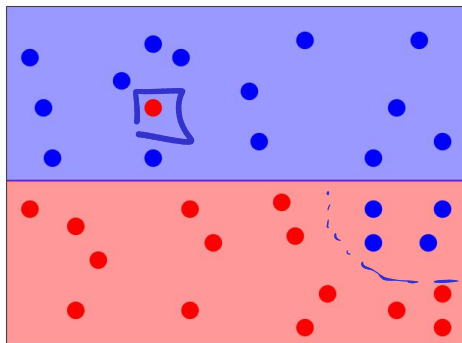
Feature 1

Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - **Overfitting**

- Tree Ensembles
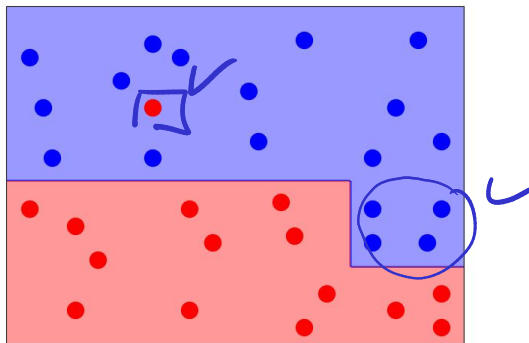 - Bagging
  - Random Forest
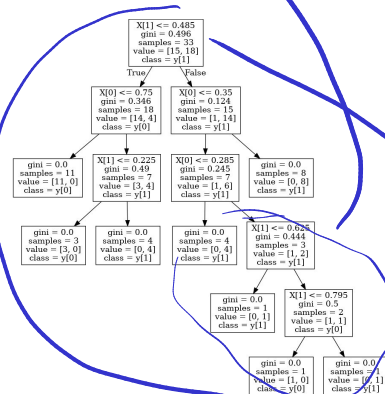 - Boosting

Decision Trees — Underfitting & Overfitting

Underfitting

Good fit

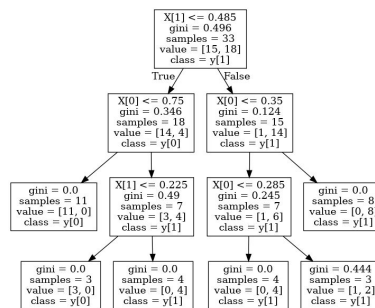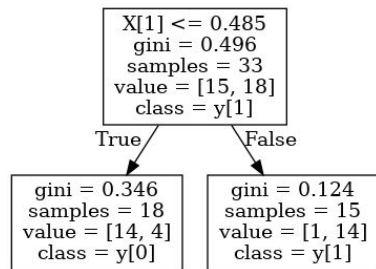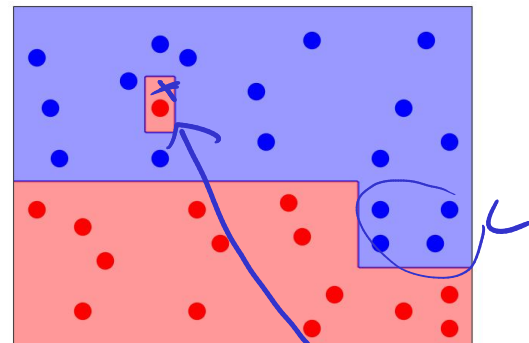Overfitting

Decision Trees — Underfitting & Overfitting

Decision Trees — Underfitting & Overfitting

"Sweet Spot"

Overfitting
Increasing training score,
decreasing test score
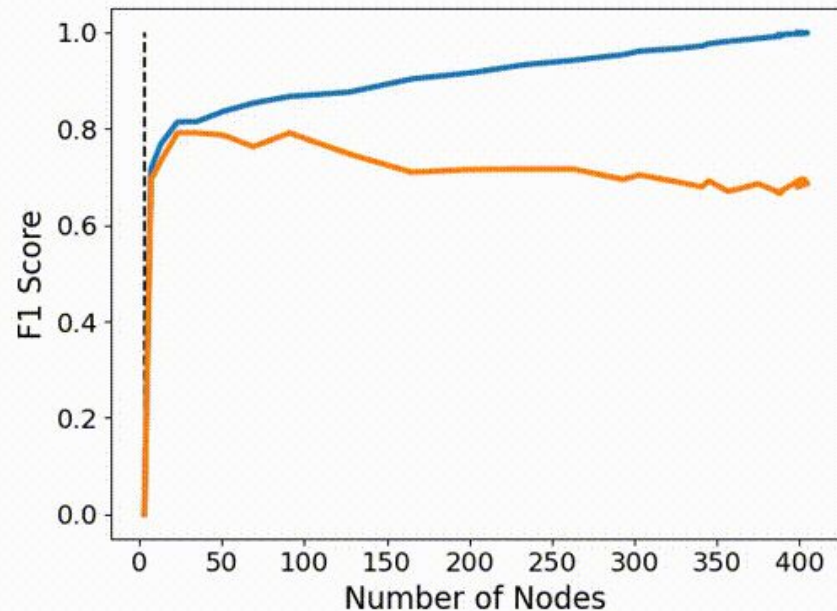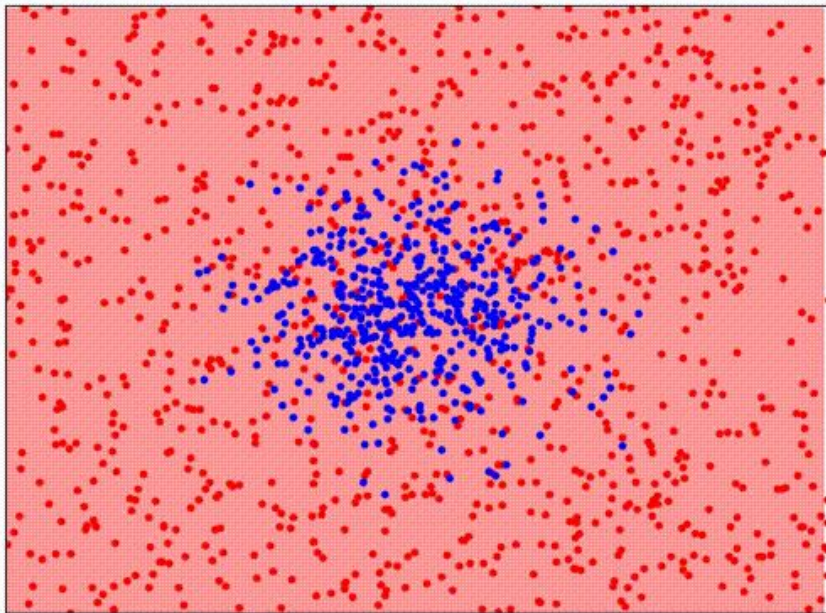
Underfitting
Both training and test
score (very) low
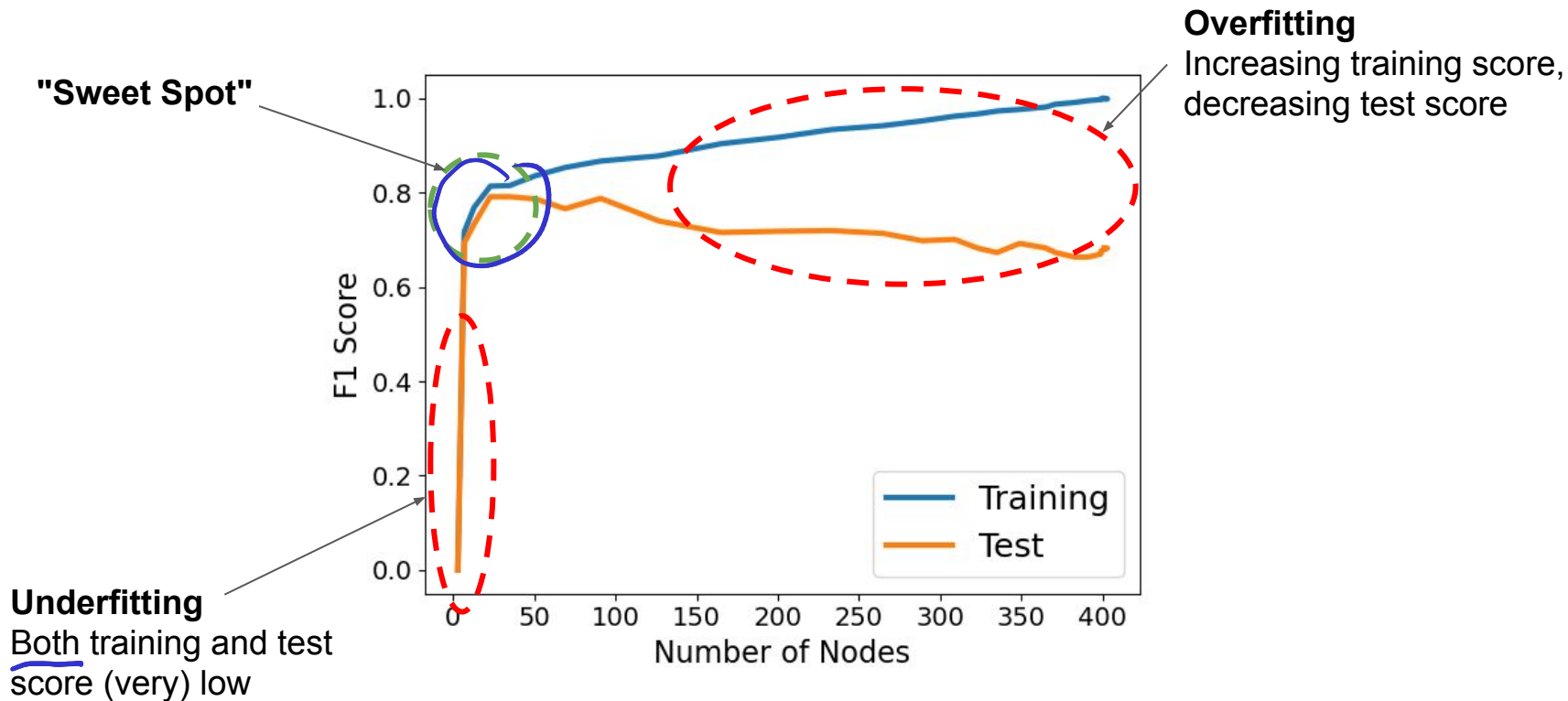
Decision Trees — Overfitting

- Decision Tree algorithm can always split the training data perfectly*
  - Keep splitting (i.e., increase height of tree) until each leaf contains only one data sample
  - One data sample ➜ 100% pure

- Solution: Limit size/height of Decision Tree ➜ Pruning
  - Pre-pruning: Stop splitting nodes ahead of time
  - Post-pruning: Build full tree, but then remove leaves/splits if beneficial
  - … combination of multiple approaches

*assuming not duplicate data points with different target labels/values
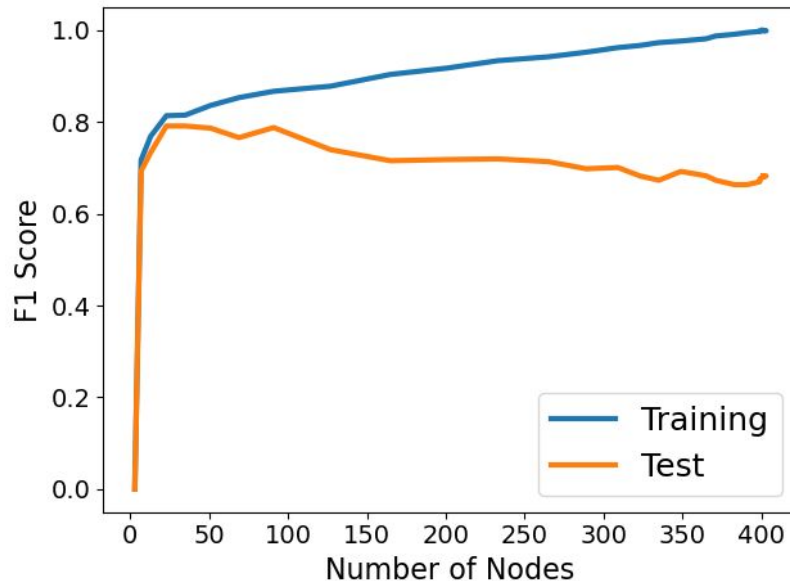
Pre-Pruning: Maximum Depth

- Define maximum depth/height of tree
    - Stop splitting if maximum depth is reached

- Example: maximum depth = 3

Depth=3

Full tree

Tree with max depth = 3

Those nodes are never generated!

44
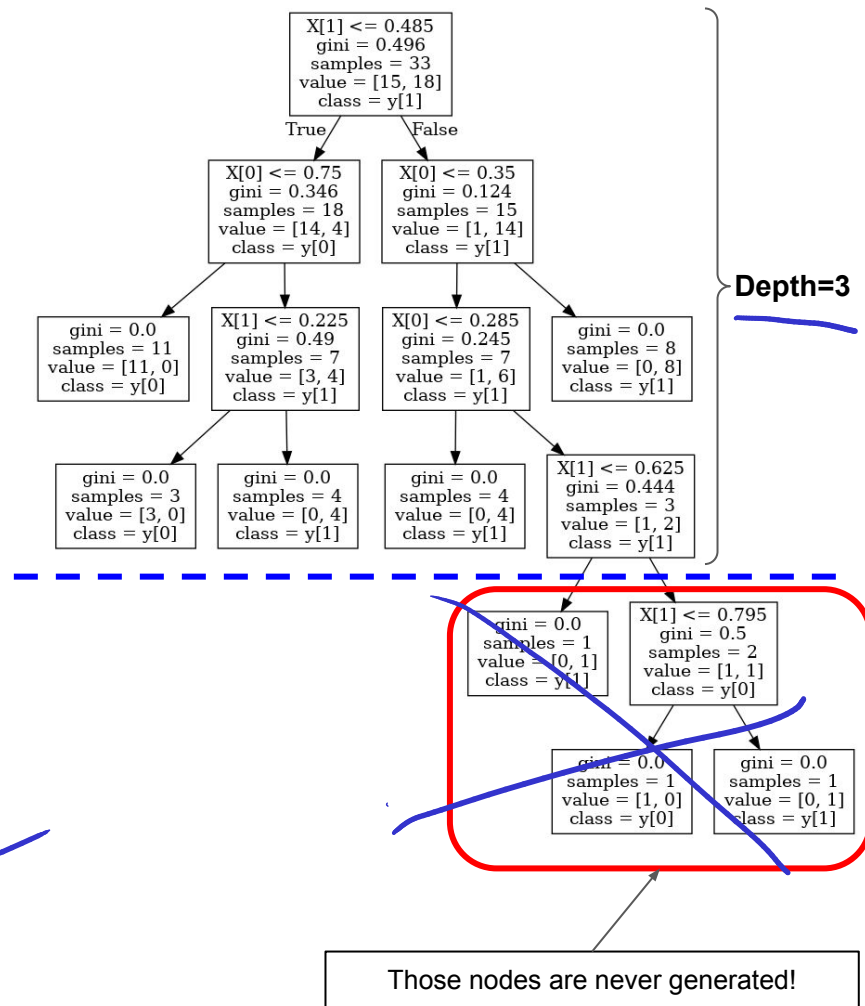
Pre-Pruning: Minimum Sample Count

- Define minimum number of samples each node must have
  - Stop splitting if node has less than the minimum number of samples

- Example: minimum sample count = 16
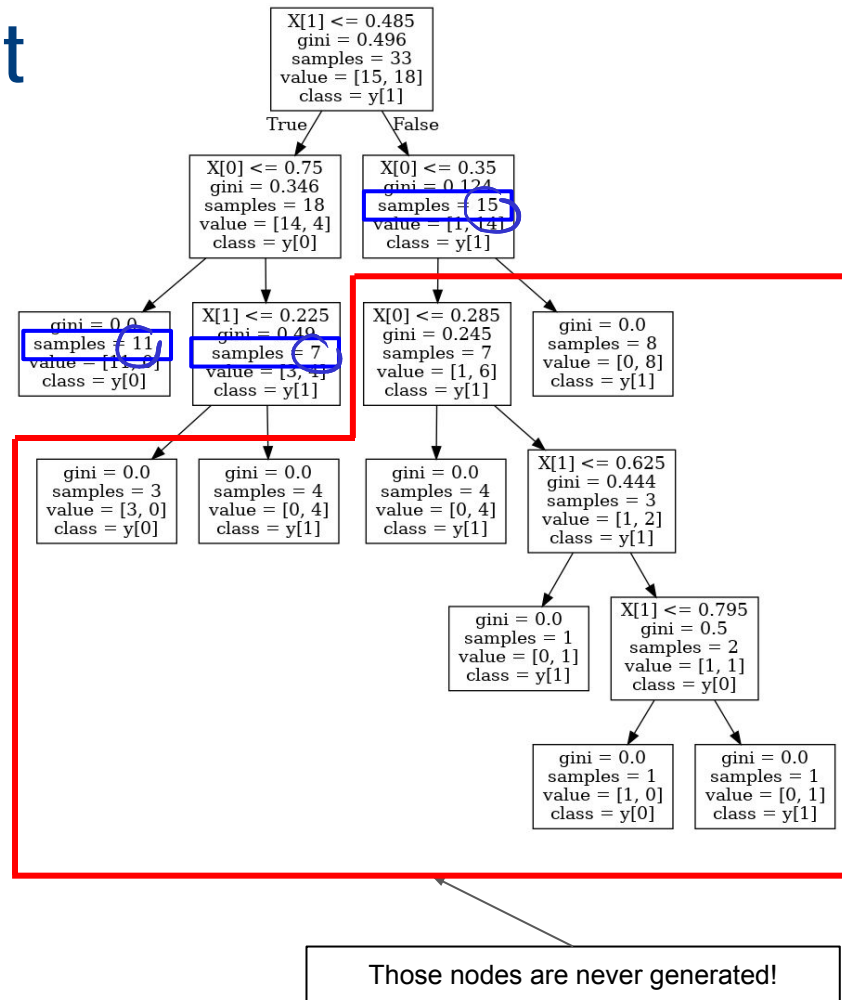
Full tree
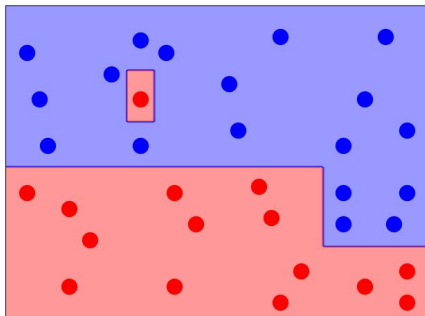
Tree with min sample count = 16

Those nodes are never generated!
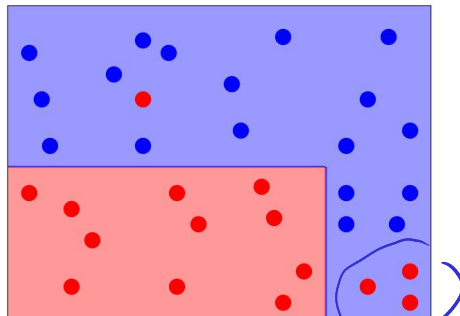
Post-Pruning: Prune Leaves/Splits using Validation
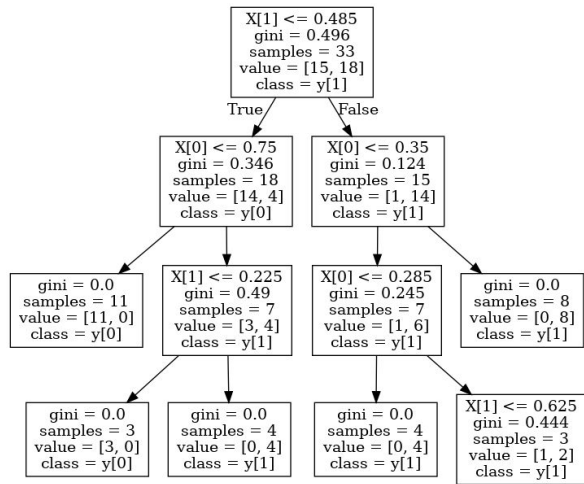
$$F1_{input} = 0.78$$
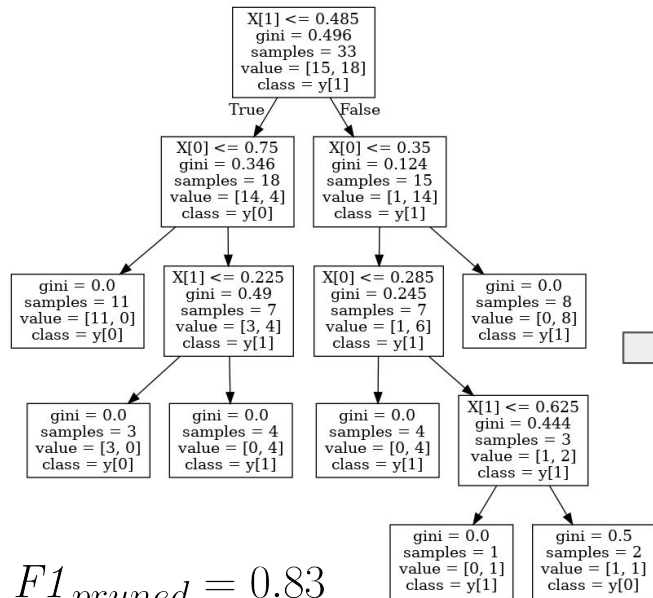
$$F1_{pruned} = 0.83$$

$$F1_{pruned} > F1_{input}$$ ➔ Remove split from Decision Tree (and continue checking next split)

Quick Quiz

What is the **maximum** possible **depth** of a Decision Tree given a dataset with N data points?

A $O(N)$

B $O(\log N)$

C $O(N \log N)$

D $O(\log \log N)$

Quick Quiz

N = 8
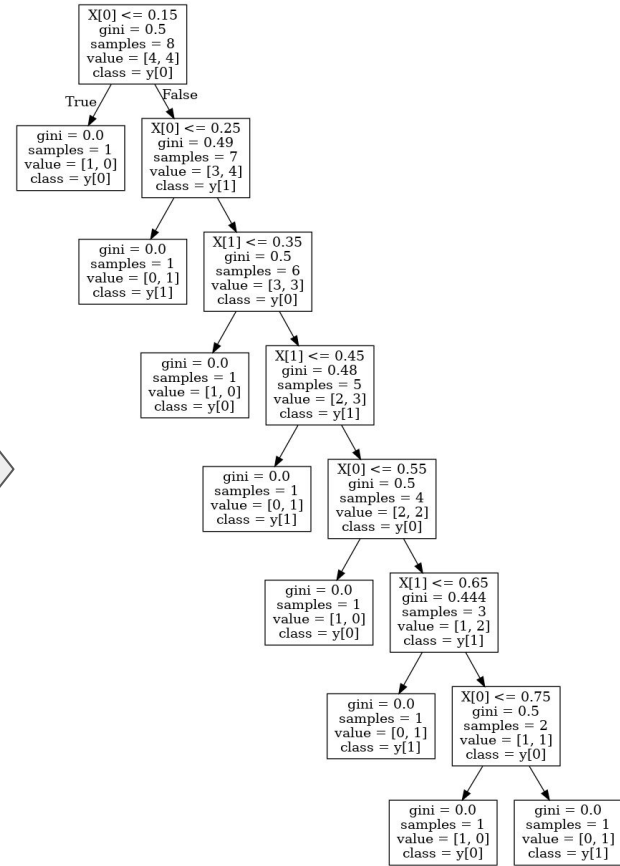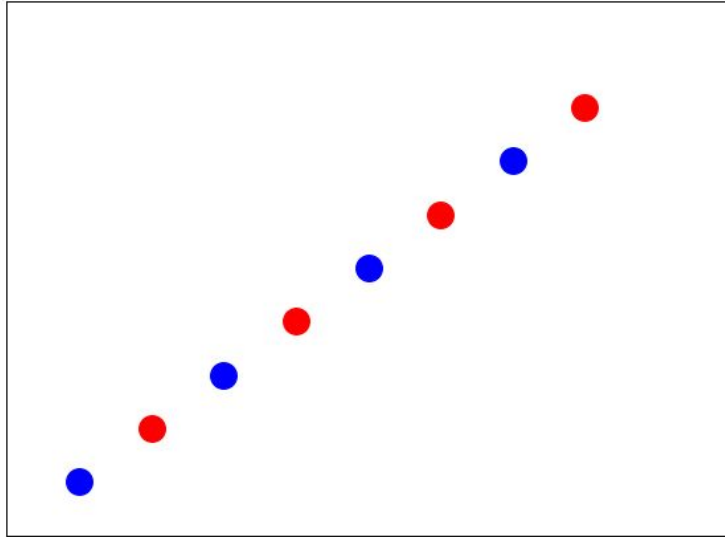
depth = N-1

Outline

- Decision Trees
    - Overview
    - Training Decision Trees
    - Overfitting

- **Tree Ensembles**
    - Bagging
    - Random Forest
    - Boosting

Tree Ensembles

- Aim to address limitations of (single) Decision Trees
  - High variance — i.e., sensitivity to small changes in training data

  - Typically not the same accuracy as other approaches

- Countermeasure: Tree Ensembles
  Construct many decision trees and combine their predictions
  - Bagging

  - Random Forests

  - Boosting

Basic trade-off of ensemble methods:
Higher accuracy, lower variance
vs.
Lower interpretability, longer training time

Outline

Bagging — Bootstrap Aggregation

- Bagging — basic idea (not limited to Decision Trees)
  - Train many models (classifiers/regressors) of different training data

  - Combine predictions of each models for final prediction

  - Increases accuracy and lowers variance

- Where to get more training data from? ➜ **Bootstrap Sampling**
  - Take repeated samples from a single training dataset D

  - Bootstrap sample D_i sampled from D, **uniformly** and **with replacement** ($|D_i| = |D|$)

  - Train a model over each bootstrap dataset D_i

Bagging — Bootstrap Aggregation

Bagging — Bootstrap Aggregation

- Limitations
  - Assume original dataset D has one or more strong predictors — features that yield splits with a (very) high information gain

  - Bootstrap samples D_i are also likely to have those strong predictors

→ Consequences
  - Most bagged trees will use strong predictors on top

  - Most bagged trees will look very similar

  - Predictions of bagges trees will be highly correlated

→ Only limited reduction in variance!

Outline

Random Forests

- Random Forest = bootstrap sampling (bagging) + **feature sampling**
  - Create bootstrap samples D_i like for bagging

  - Feature sampling: For each D_i, consider only a random subset of features of size m

d features

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes

bootstrap sampling
+ feature sampling
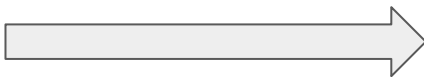
typically

$$m \approx \sqrt{d}$$

m features

Edu-cation	Annual Income	Credit Approval
Masters	75k	Yes
Bachelor	50k	No
Masters	70k	Yes
PhD	95k	Yes
Bachelor	40k	No
Masters	70k	Yes
Masters	75k	Yes
Bachelor	40k	No
Bachelor	40k	No

Random Forests

- Effects of feature sampling
  - Strong predictors in D are often absent in D_i
  - Resulting trees often look very different
  - Predictions of trees much less correlated

➜ Higher reduction in variance + typically higher accuracy
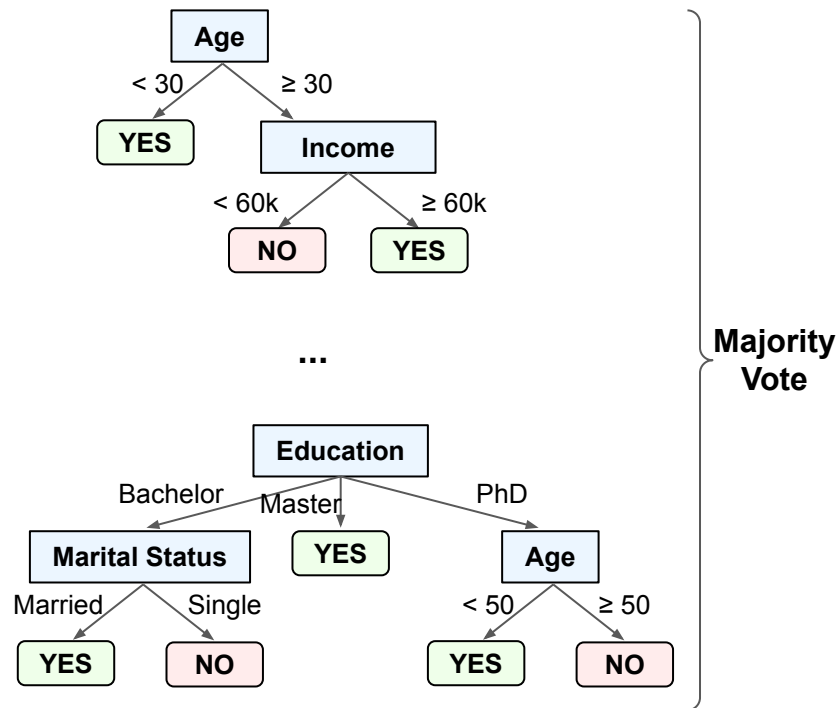
Random Forests — Pros & Cons (Compared to Decision Trees)

- Pros
  - High accuracy — fairly close to state of the art
  - Sampling and training independent across D_i ➜ parallelizable!
  - Not much tuning required

- Cons
  - Less Interpretable
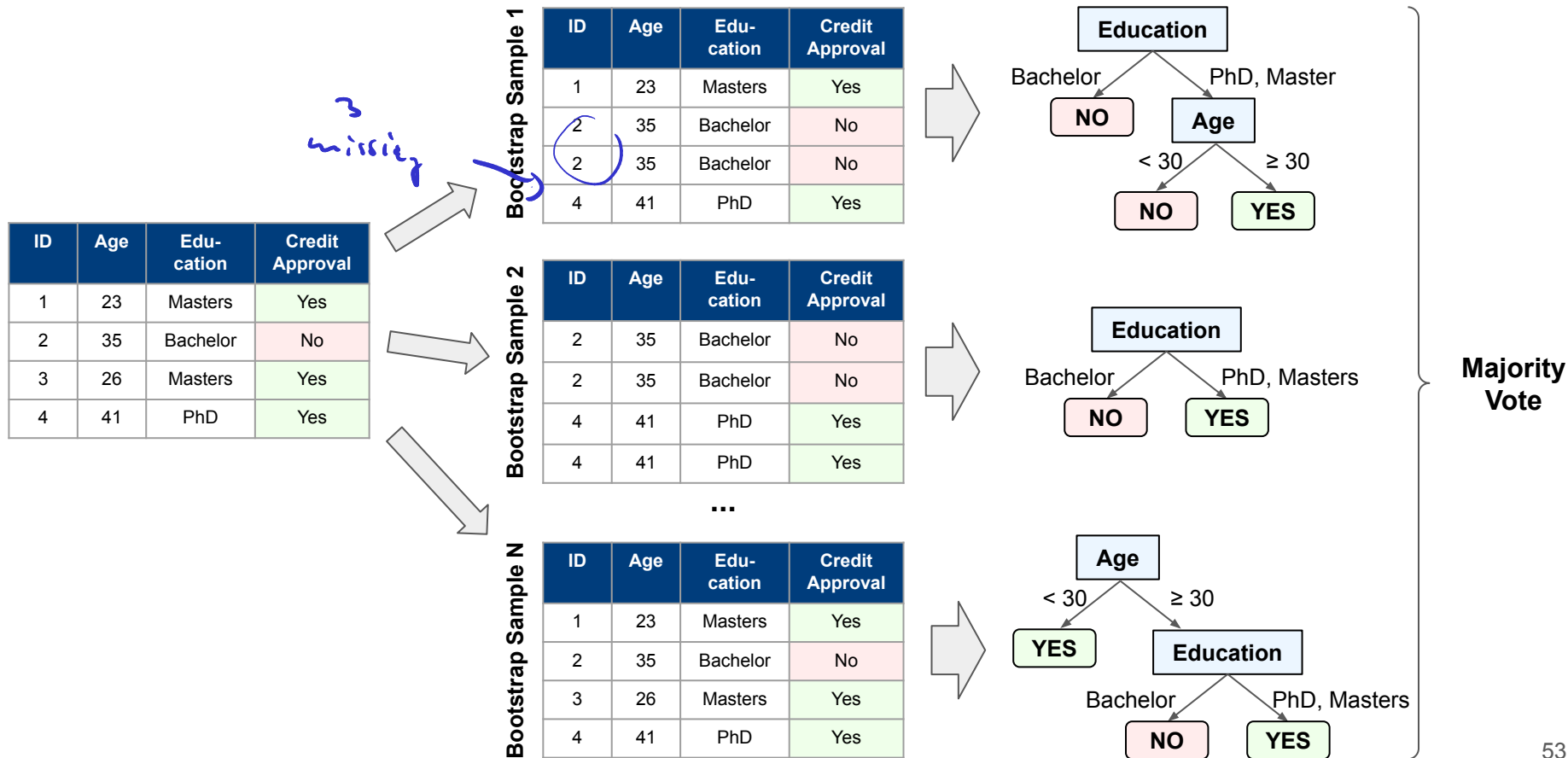 - Slower training and prediction

Outline

- **Decision Trees**
    - Overview
    - Training Decision Trees
    - Overfitting

- **Tree Ensembles**
    - Bagging
    - Random Forest
    - **Boosting**

Boosting

- Like bagging, boosting combines multiple trees (in general, multiple models)

- So what are the key differences?

	Bagging	**Boosting**
Training	Trees are trained independently (and can be done in parallel)	Trees are trained in sequence; (the accuracy of the last tree affects the training of the next tree)
Prediction	All trees have the same amount of say in the final prediction	Trees have different amount of say in the final prediction (depending on their individual accuracy)

Boosting & Weak Learners

- So far, all models discussed are **Strong Learners**
  - Goal: perform as best as possible on a given classification or regression task

- **Weak Learner**
  - Goal: perform (slightly) better than guessing

  - Very common weak learner: **Decision Stump**
    (e.g., decision tree of height 1, i.e., only one split)

  - Very simple model ➜ very fast training

Education

Bachelor   Master, PhD

3x NO   **5x YES** **1x NO**

0 incorrect   **1 incorrect**
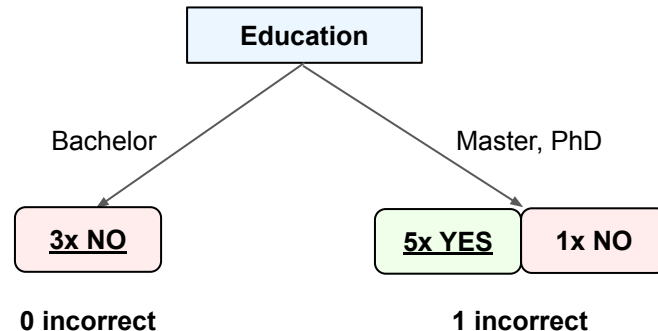
- **Boosting**: Combine many weak classifiers into a single strong learner
  - Basic idea: subsequent models try to improve the errors of previous models

Theoretical underpinnings: The Strength of Weak Learnability (Schapire, 1990)

AdaBoost — Adaptive Boosting (for Decision Trees)

- ## AdaBoost
  - Applicable to many classification/regression algorithms to improve performance
  - Very commonly combined with Decision Trees

- ## Basic training algorithm
  - Train a Weak Learner over D_i (e.g., Decision Stump)
  - Identify all misclassified samples
  - Calculate error rate of learner to quantify its amount of say
  - Sample D_{i+1} such that misclassified samples are more likely to be picked than correctly classified samples
  - Repeat...

AdaBoost Training — Step-by-Step Example (in the *m*-th iteration)

- ## Step 1:

  1a) Train Decision Stump h_m over sampled dataset D_m
  (original dataset D in the beginning)

  1b) Identify all misclassified training samples in D

Sample Weight w
1/9
1/9
1/9
1/9
1/9
1/9
1/9
1/9
1/9

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes

Education

Bachelor → **3x NO** — 0 incorrect

Master, PhD → **5x YES** | **1x NO** — 1 incorrect

Initial step: assign each data sample in D with a weight $w_i = 1/|D|$

AdaBoost Training — Step-by-Step Example (in the m-th iteration)

- Step 2

  2a) Calculate total error ϵ_m

$$\epsilon_m = \sum_i^N w_i \cdot \underbrace{\delta(h_m(x_i) \neq y_i)}_{\substack{\text{0, if } x_i \text{ is correctly classified} \\ \text{1, if } x_i \text{ is misclassified}}}$$

$$\epsilon_m = 1/9$$

  2b) Calculate "amount of say" α_m of h_m

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \epsilon_m}{\epsilon_m}$$

$$\alpha_m = 1.04$$
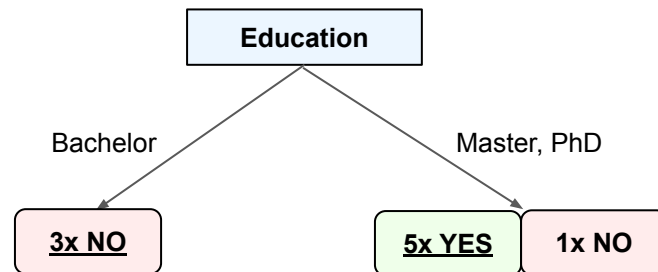
AdaBoost Training — Step-by-Step Example (in the m-th iteration)

- Step 3

  3a) Update sample weights

  $$w_i = w_i \cdot \begin{cases} e^{\alpha_m}, \text{if } x_i \text{ was misclassified} \\ e^{-\alpha_m}, \text{if } x_i \text{ was correctly classified} \end{cases}$$

  3b) Normalize sample weights

  $$w_i = \frac{w_i}{\sum_i^N w_i}$$

Age	Edu-cation	Marital Status	Annual Income	Credit Approval	
23	Masters	Single	75k	Yes	✔
35	Bachelor	Married	50k	No	✔
26	Masters	Single	70k	Yes	✔
41	PhD	Single	95k	Yes	✔
18	Bachelor	Single	40k	No	✔
55	Master	Married	85k	No	✘
30	Bachelor	Single	60k	No	✔
35	PhD	Married	60k	Yes	✔
28	PhD	Married	65k	Yes	✔

Sum up to 1

Sample Weight w	3a)	3b)
1/9	0.04	0.0635
1/9	0.04	0.0635
1/9	0.04	0.0635
1/9	0.04	0.0635
1/9	0.04	0.0635
1/9	**0.31**	**0.492**
1/9	0.04	0.0635
1/9	0.04	0.0635
1/9	0.04	0.0635

Sample Weight w
0.0635
0.0635
0.0635
0.0635
0.0635
0.492
0.0635
0.0635
0.0635

AdaBoost Training — Step-by-Step Example (in the *m*-th iteration)

- Step 4

  4a) Generate new D_i based on sample weights
      (misclassified samples are much more likely to be picked)

  4b) With new D_i, go to Step 1 and continue

Sample Weight w
0.0635
0.0635
0.0635
0.0635
0.0635
0.492
0.0635
0.0635
0.0635

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes

New input for Step 1

Age	Edu-cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
55	Master	Married	85k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
55	Master	Married	85k	No
55	Master	Married	85k	No
26	Masters	Single	70k	Yes
35	PhD	Married	60k	Yes
55	Master	Married	85k	No

66

AdaBoost Training — Basic Algorithm

Initialization: Dataset D, $|D|=N$, with initial sample weights $\quad w_i = \dfrac{1}{N}$

for m = 1 to M **do:**

Generate D_m by sampling from D w.r.t. sampling weights w

Train Decision Stump h_m over D_m

Apply h_m to all samples in D and identify misclassified samples

Calculate total error $\qquad \epsilon_m = \sum_i^N w_i \cdot \delta(h_m(x_i) \neq y_i)$

Calculate amount of say $\qquad \alpha_m = \dfrac{1}{2} \ln \dfrac{1 - \epsilon_m}{\epsilon_m}$

Update sample weights $\qquad w_i = w_i \cdot \begin{cases} e^{\alpha_m}, \text{if } x_i \text{ was misclassified} \\ e^{-\alpha_m}, \text{if } x_i \text{ was correctly classified} \end{cases}$ & $\quad w_i = \dfrac{w_i}{\sum_i^N w_i}$

end for

AdaBoost Prediction

- Assume 8 boosted Decision Stumps $h_1, \ldots, h_8$
  - Each tree has an "amount of say" α_m

  - Let h_1, h_3, h_8 say "Yes"; all other trees say "No"

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \epsilon_m}{\epsilon_m}$$

h_1	α_1 = 0.34
h_3	α_3 = 1.20
h_8	α_8 = 0.97
h_2	α_2 = 0.14
h_4	α_4 = 0.58
h_5	α_5 = 0.09
h_6	α_6 = 0.62
h_7	α_7 = 0.45

0.34 + 1.20 + 0.97 = **2.51** ⟹ Final prediction: **"Yes"**

0.14 + 0.58 + 0.09 + 0.62 + 0.45 = **1.88**

Gradient Boosted Trees

- ## Gradient Boosting
  - Mainly applied to regression algorithms to improve performance

  - Very commonly combined with Decision Trees (for regression)

- ## Basic training algorithm
  - Start with a initial prediction (e.g., mean over all values)

  - Calculate residuals = error between true value and current prediction

  - Train Decision Stump to predict residuals

  - Update predictions based on predicted residuals

  - Repeat...

GB Training — Step-by-Step Example (in the *m*-th iteration)

- ## Step 1:

  1a) Calculate residuals $r_{i,m} = y_i - f_{m-1}(x_i)$

  1b) Fit Decision Stump h_m to residuals $r_{i,m}$

Assume m = 1 $f_0(x_i) = 1000$

target   *initial prediction*

Age	Edu-cation	Marital Status	Annual Income	Credit Limit y	$f_{m-1}(x)$	$r_m(x)$
23	Master	Single	75k	1,400	1,000	400
35	Bachelor	Married	50k	900	1,000	100
26	Master	Single	70k	1,300	1,000	300
41	PhD	Single	95k	1,500	1,000	500
18	Bachelor	Single	40k	500	1,000	-500
55	Master	Married	85k	1,200	1,000	200
30	Bachelor	Single	60k	800	1,000	-200
35	PhD	Married	60k	800	1,000	-200
28	PhD	Married	65k	600	1,000	-400

h_m

70
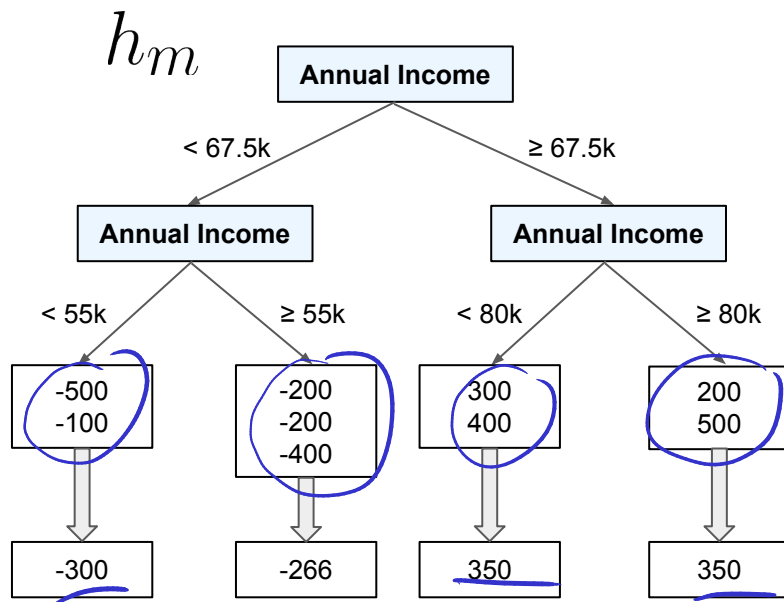
GB Training — Step-by-Step Example (in the *m*-th iteration)

- Step 2:

  1a) Calculate predicted residuals $h_m(x_i)$ for all training samples

  1b) Calculate new predictions $f_m(x_i) = f_{m-1} + \eta \cdot h_m(x_i)$ (here: $\eta = 0.1$)

  1c) Set m = m+1, go to Step 1

learning
d

Age	Edu-cation	Marital Status	Annual Income	Credit Limit y	$f_{m-1}(x)$	$r_m(x)$	$h_m(x)$	$f_m(x)$
23	Master	Single	75k	1,400	1,000	400	350	1,035
35	Bachelor	Married	50k	900	1,000	-100	-300	970
26	Master	Single	70k	1,300	1,000	300	350	1,035
41	PhD	Single	95k	1,500	1,000	500	350	1,035
18	Bachelor	Single	40k	500	1,000	-500	-300	970
55	Master	Married	85k	1,200	1,000	200	350	1,035
30	Bachelor	Single	60k	800	1,000	-200	-266	973
35	PhD	Married	60k	800	1,000	-200	-266	973
28	PhD	Married	65k	600	1,000	-400	-266	973

Note: long-term trend

- The residuals r_m go towards 0

- The predicted values f_m are closer to the true values y

GB Training — Step-by-Step Example (in the *m*-th iteration)

- Output for after Step 1 & 2 for m+1

Includes building new Decision Stump h_{m+1}

$1,400 - 1,035 = 365$

Step 1 **Step 2**

Age	Edu-cation	Marital Status	Annual Income	Credit Limit y	$f_{m-1}(x)$	$r_m(x)$	$h_m(x)$	$f_m(x)$	$r_{m+1}(x)$	$h_{m+1}(x)$	$f_{m+1}(x)$
23	Master	Single	75k	1,400	1,000	400	350	1,035	365	315	1,067
35	Bachelor	Married	50k	900	1,000	-100	-300	970	-70	-270	943
26	Master	Single	70k	1,300	1,000	300	350	1,035	265	315	1.067
41	PhD	Single	95k	1,500	1,000	500	350	1,035	465	315	1.067
18	Bachelor	Single	40k	500	1,000	-500	-300	970	-470	-270	943
55	Master	Married	85k	1,200	1,000	200	350	1,035	165	315	1,067
30	Bachelor	Single	60k	800	1,000	-200	-266	973	-173	-240	949
35	PhD	Married	60k	800	1,000	-200	-266	973	-173	-240	949
28	PhD	Married	65k	600	1,000	-400	-266	973	-373	-240	949

72

Gradient Boosting Training — Basic Algorithm

Initialization: Dataset *D*, $f_0(x_i) = mean(y)$, $\eta = 0.1$

for *m* = 1 to *M* **do:**

   Calculate residuals $r_{i,m} = y_i - f_{m-1}(x_i)$

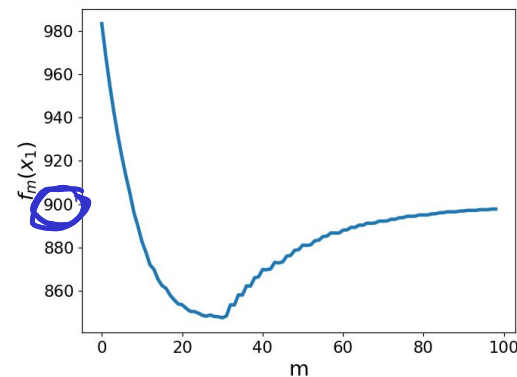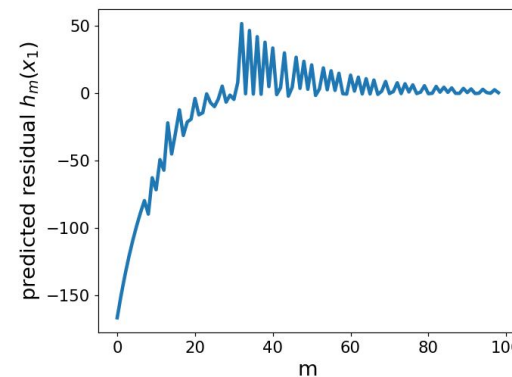   Train Decision Stump *h*_m over *D* with with $r_{i,m}$ as targets
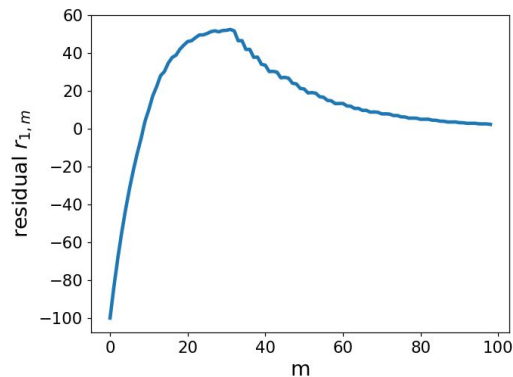
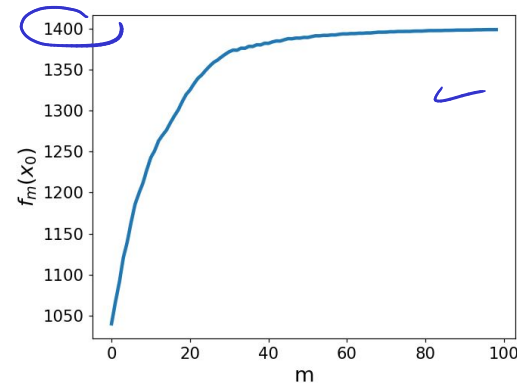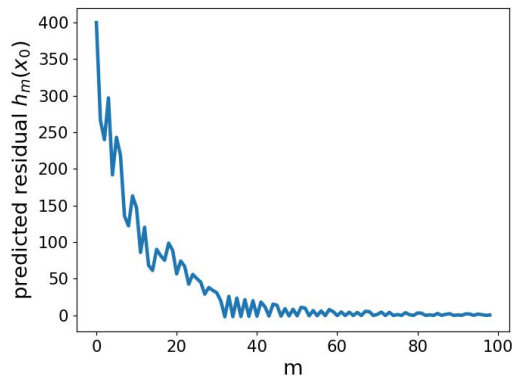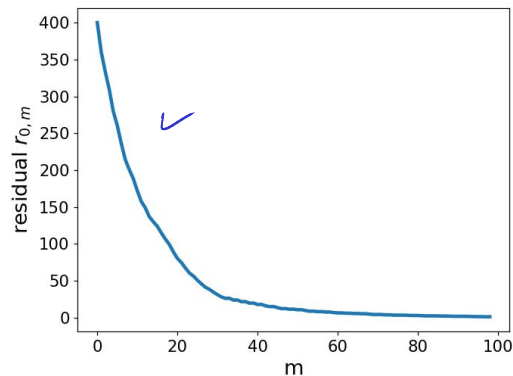   Predicted residuals $h_m(x_i)$ for all training samples

   Calculate new predictions $f_m(x_i) = f_{m-1} + \eta \cdot h_m(x_i)$

   end for

Output: *M* Decision Stumps *h*₁, *h*₂, …, *h*_M

Gradient Boosting Training — Convergence for x_0 and x_1

Gradient Boosting Prediction

Age	Edu-cation	Marital Status	Annual Income	Credit Limit
50	PhD	Single	70k	**???**

$$h(x) = h_0(x) + \eta h_1(x) + \eta h_2(x) + ... + \eta h_M(x)$$

Initial prediction $f_0(x)$

(e.g., $f_0(x) = 1000$)

Boosting Methods — Pros & Cons (Compared to Decision Trees)

- Pros
  - High accuracy — often state of the art

- Cons
  - Less Interpretable (arguably even less compared to Random Forests)
  - Slower training and prediction ➜ sequential processing ➜ not parallelizable

Summary

- ## Decision Trees
  - Intuitive model for classification and regression ➜ interpretable!

  - Can handle categorical and numerical data (although tricky in practice)

  - Typically good but not great results

- ## Tree Ensembles
  - Aim to address limitations of single decision trees (particularly high variance)

  - Ensembles of independent models: Bagging, Random Forests

  - Ensembles of dependent models: AdaBoost, Gradient Boosted Trees

  - State of the art in many application contexts

Solutions to Quick Quizzes

- Slide 11: Throw error (safe default), use majority label of subtree

- Slide 27: B ("natural" ranking of energy labels is broken)

- Slide 47: A