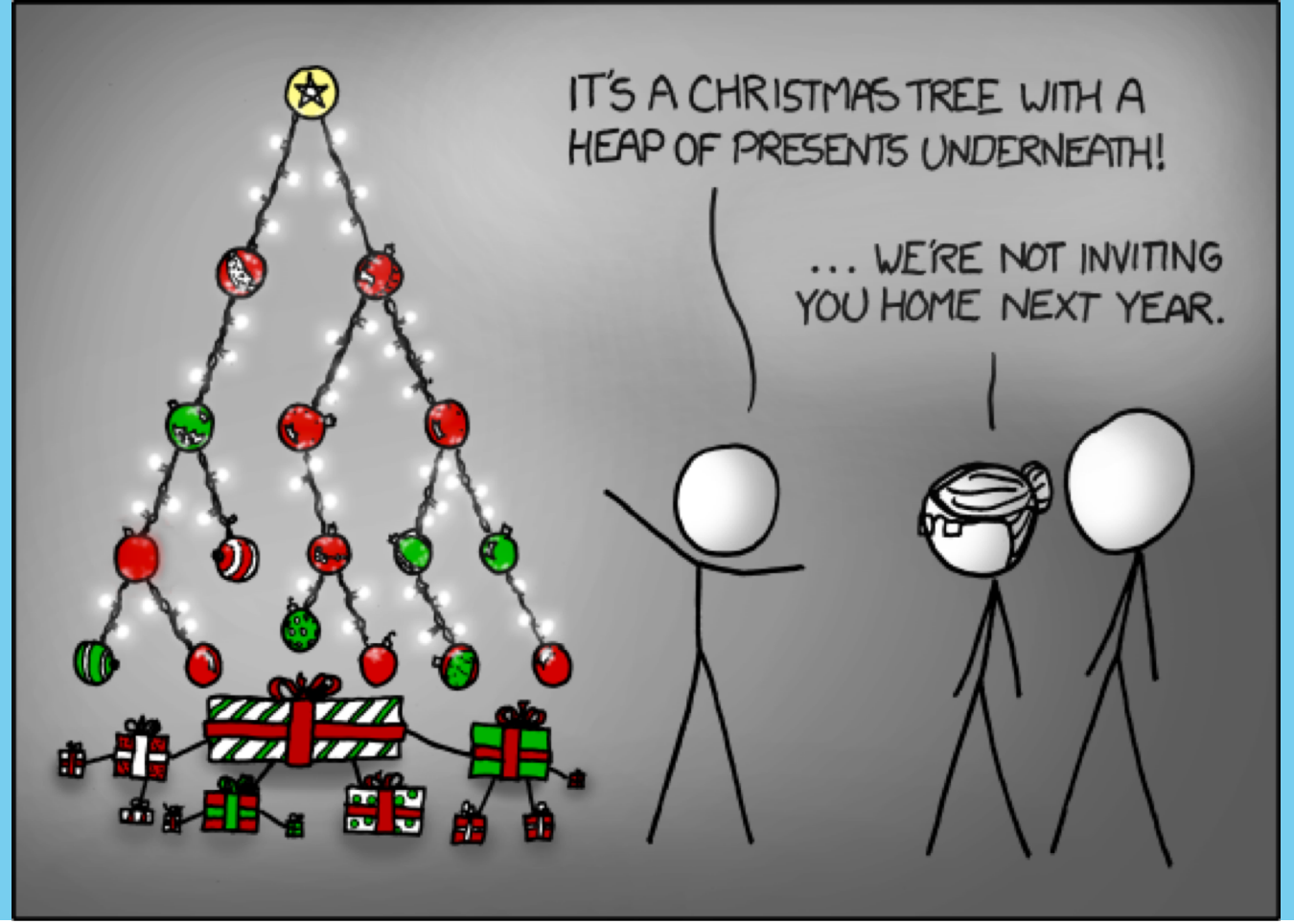


CS5228 LECTURE 6: CLASSIFICATION II: TREES & ENSEMBLES

Bryan Hooi

School of Computing

National University of Singapore

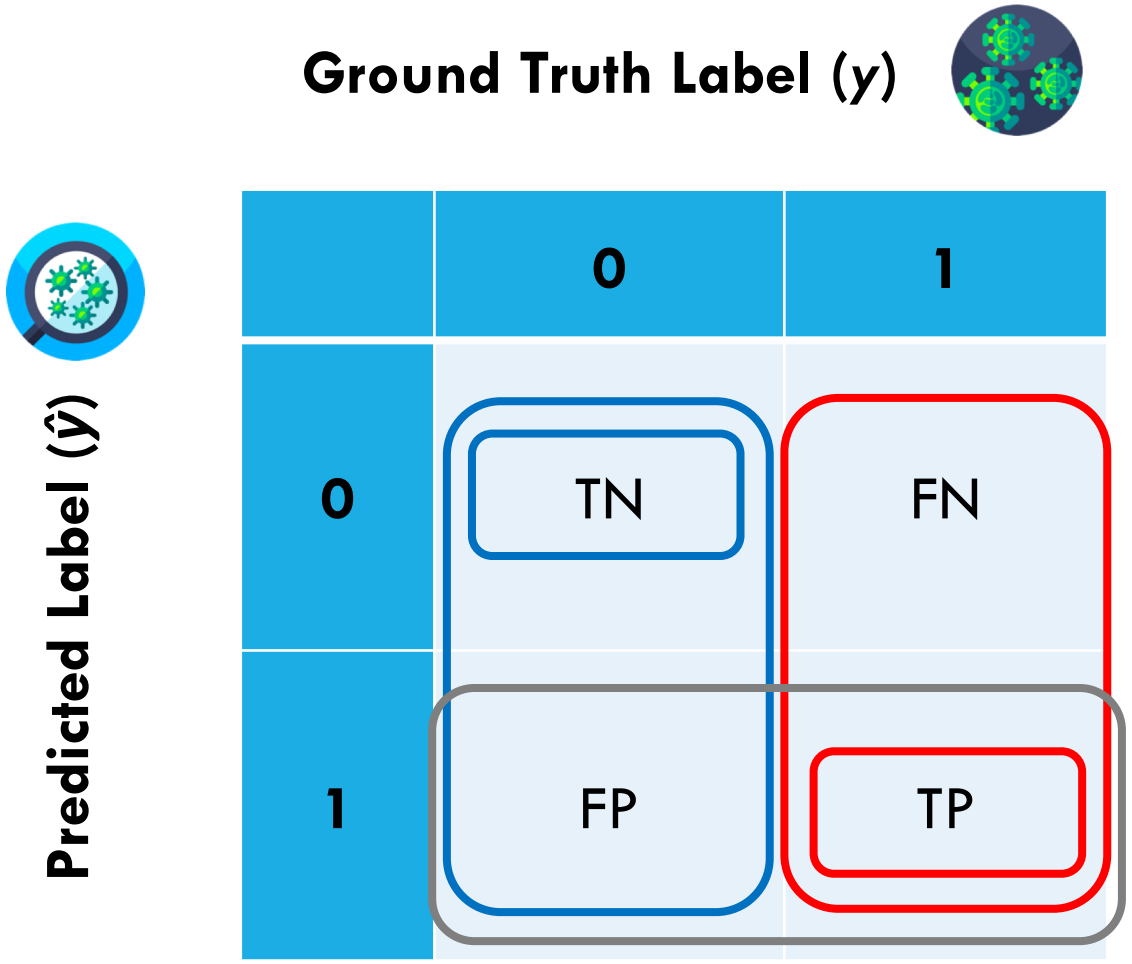
Slide Credit: Wang Wei, Ng See Kiong, Wynne Hsu, Chris von der Weth

ANNOUNCEMENTS

- Assignment 1 is due this Sunday 11.59pm (unless you use late days). You have 8 late days; see the recent announcement for details.
- Assignment 2 will be released this weekend.
- Please finalize your project group in Canvas if you have not done so (deadline is next Friday; email course staff for any difficulties)

Week	Date	Topics	Tutorials	Important Dates
1	13 Jan	Introduction		
2	20 Jan	No class (public holiday)		
3	27 Jan	Clustering I	Tutorial 1	
4	3 Feb	Clustering II		Release A1 + project
5	10 Feb	Association Rules	Tutorial 2	
6	17 Feb	Regression & Classification I		
Recess		No class		
7	3 Mar	Regression & Classification II	Tutorial 3	A1 due (Sunday 11.59pm), release A2
8	10 Mar	Regression & Classification III		
9	17 Mar	Recommender Systems	Tutorial 4	
10	24 Mar	Graph Mining		
11	31 Mar	Data Stream Mining	Tutorial 5	A2 due (Sunday 11.59pm)
12	7 Apr	No class (public holiday)		
13	14 Apr	Review & Outlook		Project due (Sunday 11.59pm)

BINARY CLASSIFICATION METRICS



Sensitivity: same as recall (fraction of actual positives that are correctly identified)

= TP / (TP + FN)

Specificity: fraction of actual negatives that are correctly identified

= TN / (TN + FP)

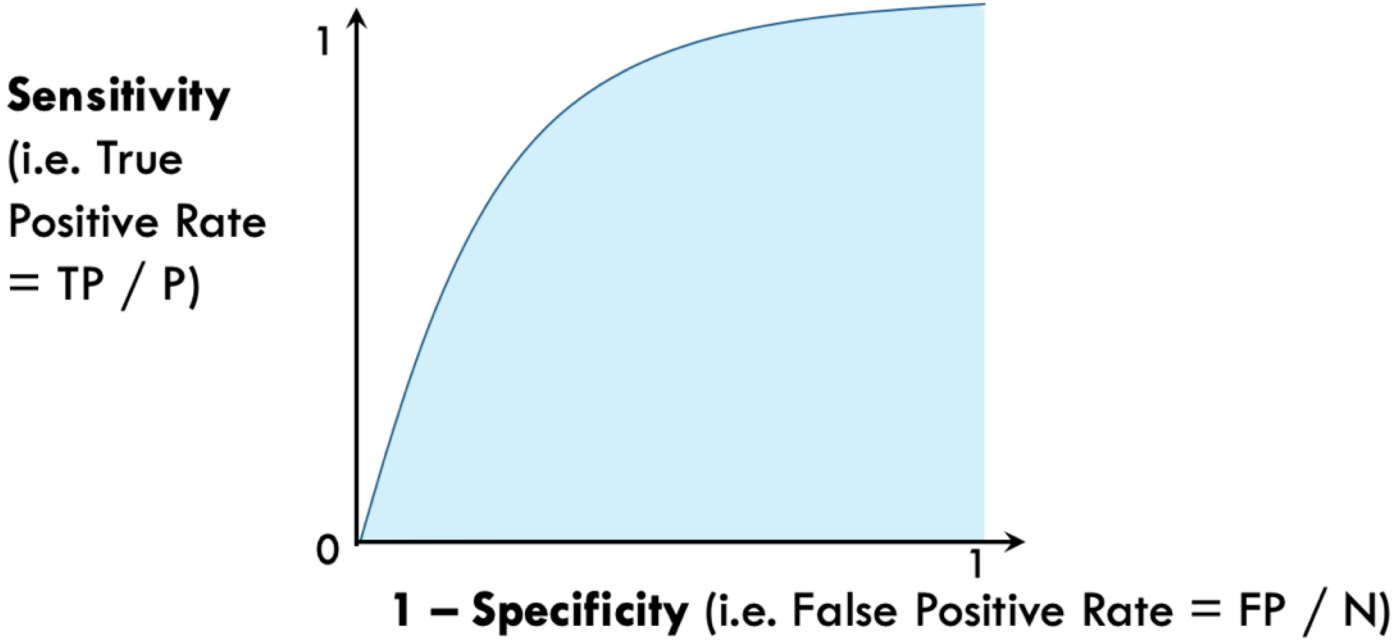
Interpretation:

Sensitivity = accuracy among positives (y=1)

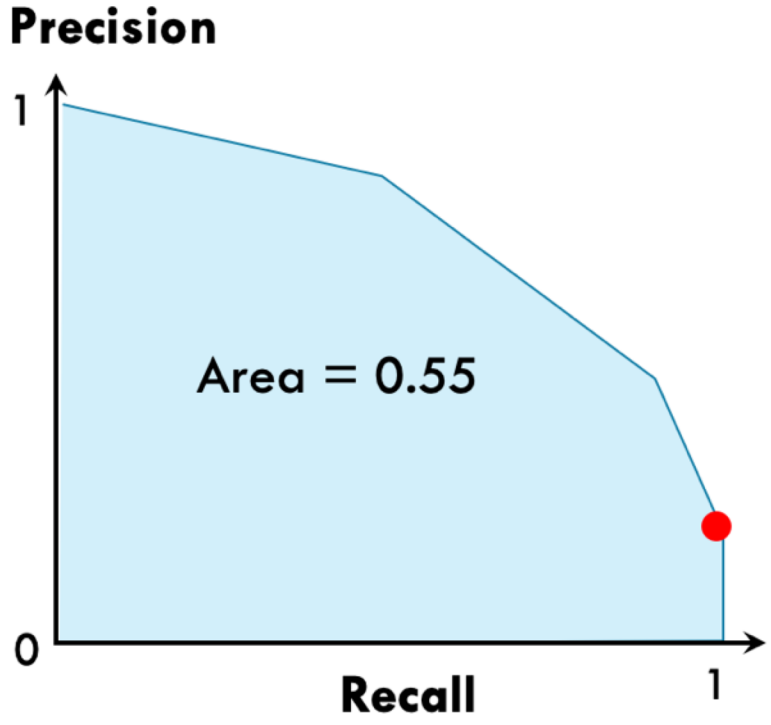
Specificity = accuracy among negatives (y=0)

Precision = accuracy among detected (ŷ=1)

REVIEW: AUC, AUPRC



Area Under ROC Curve (AUC)



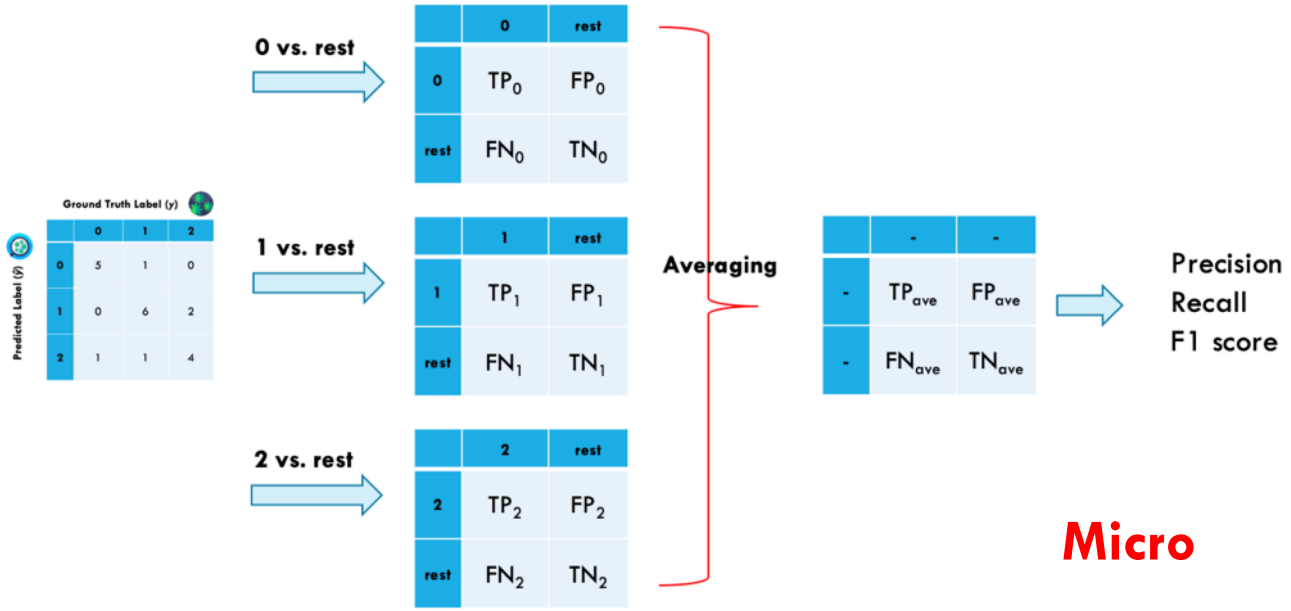
Area Under Precision Recall Curve (AUPRC)

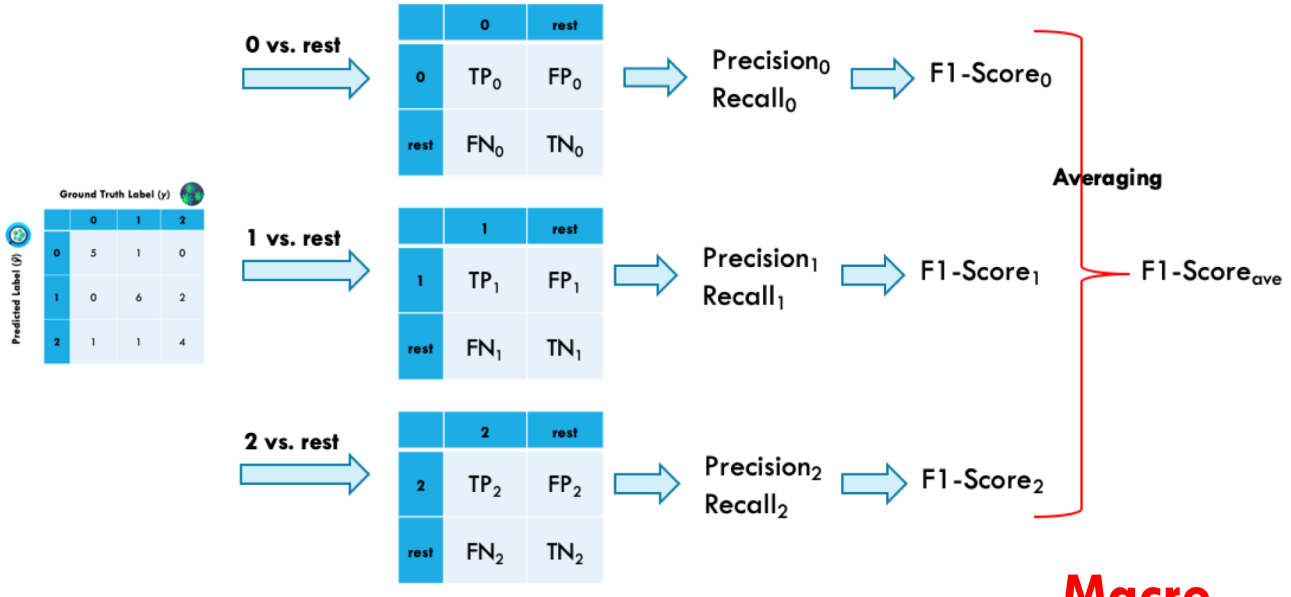
REVIEW: MICRO VS MACRO PRECISION / RECALL / F1

Both based on **one vs rest confusion matrices**

Micro: averaging takes place over (TP, TN, FP, FN)

Macro: averaging takes place over the statistic itself (e.g. F1-score)

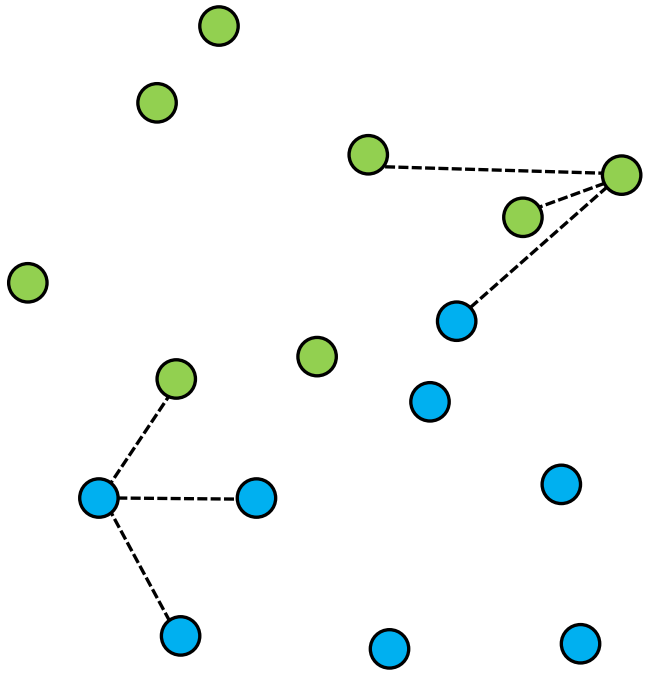




REVIEW: K NEAREST NEIGHBORS ALGORITHM

For each test point x:

- Compute its distances to all training instances: $d_1, \ldots, d_n$
- **Majority Vote:** Output most frequent class among the k nearest neighbors of x



$k = 3$

CLASSIFICATION OVERVIEW

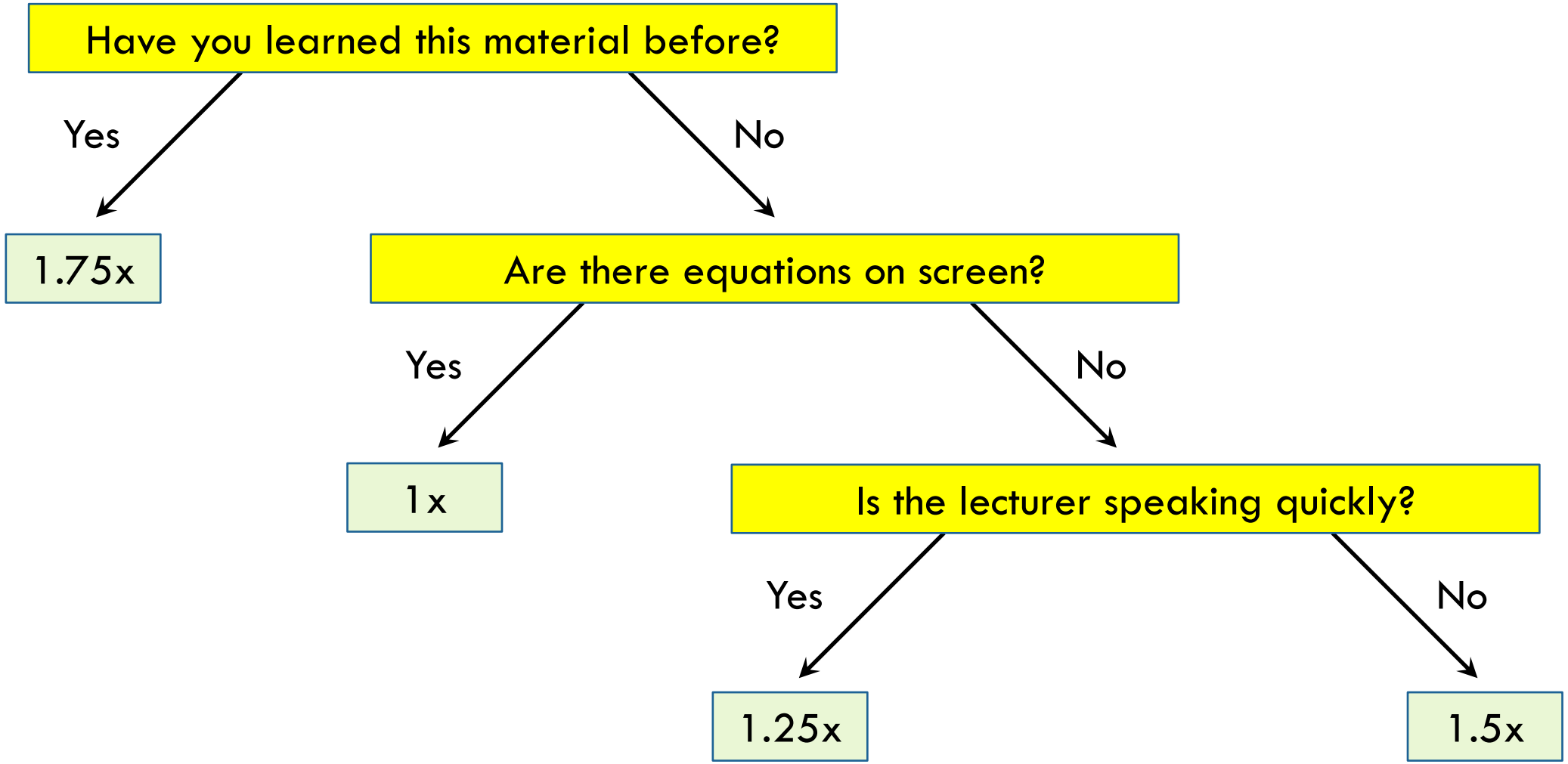
1. Problem Setup
2. Evaluating Classifiers
3. Nearest Neighbor Methods
4. **Trees and Ensembles**
   a) Decision Trees
   b) Bagging and Random Forests
   c) Boosting and Gradient Boosting Machines
5. Logistic Regression
6. Deep Learning

Today's Lecture

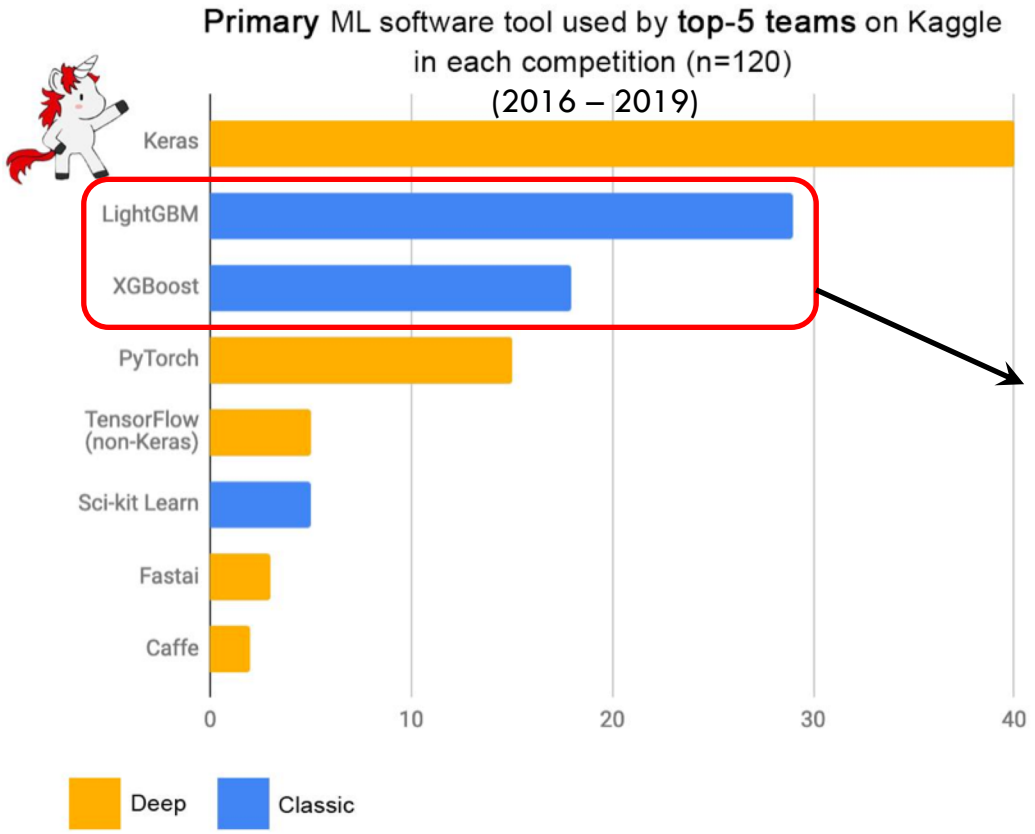
CLASSIFICATION OVERVIEW

1. Problem Setup
2. Evaluating Classifiers
3. Nearest Neighbor Methods
4. **Trees and Ensembles**
   a) Decision Trees
   b) Bagging and Random Forests
   c) Boosting and Gradient Boosting Machines
5. Logistic Regression
6. Deep Learning

AT WHAT SPEED SHOULD I WATCH THE LECTURE VIDEO?



TREE-BASED METHODS ACHIEVE STATE OF THE ART RESULTS



Gradient boosted tree methods

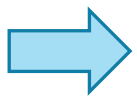
Source: Francois Chollet

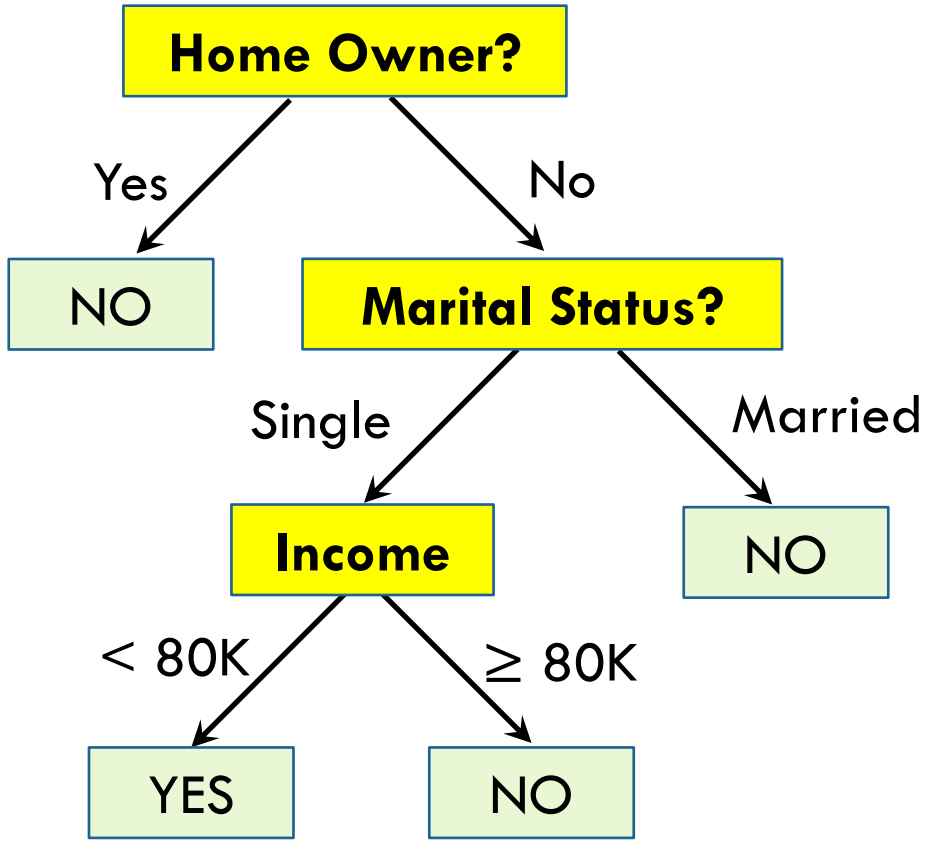
Ke G, et al.. **Lightgbm**: A highly efficient gradient boosting decision tree. *NeurIPS 2017*

Chen, Tianqi, and Carlos Guestrin. "**Xgboost**: A scalable tree boosting system." *KDD 2016*

WHAT IS A DECISION TREE?

	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

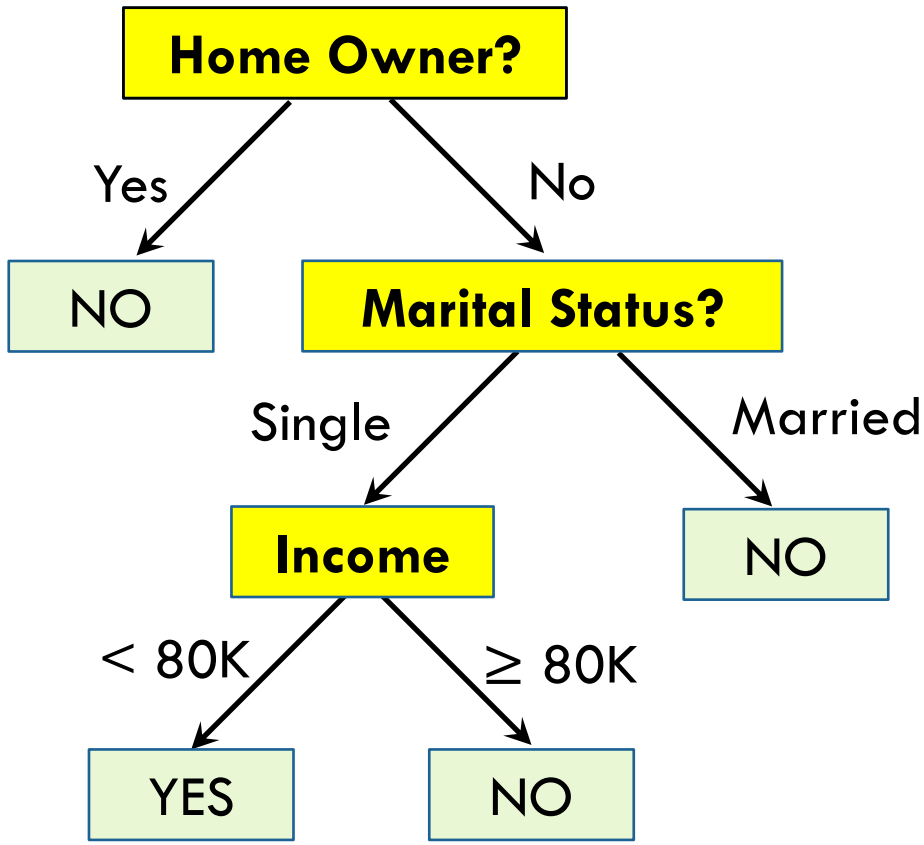




APPLYING MODEL TO TEST DATA

	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower?**
1	No	Married	205K	?

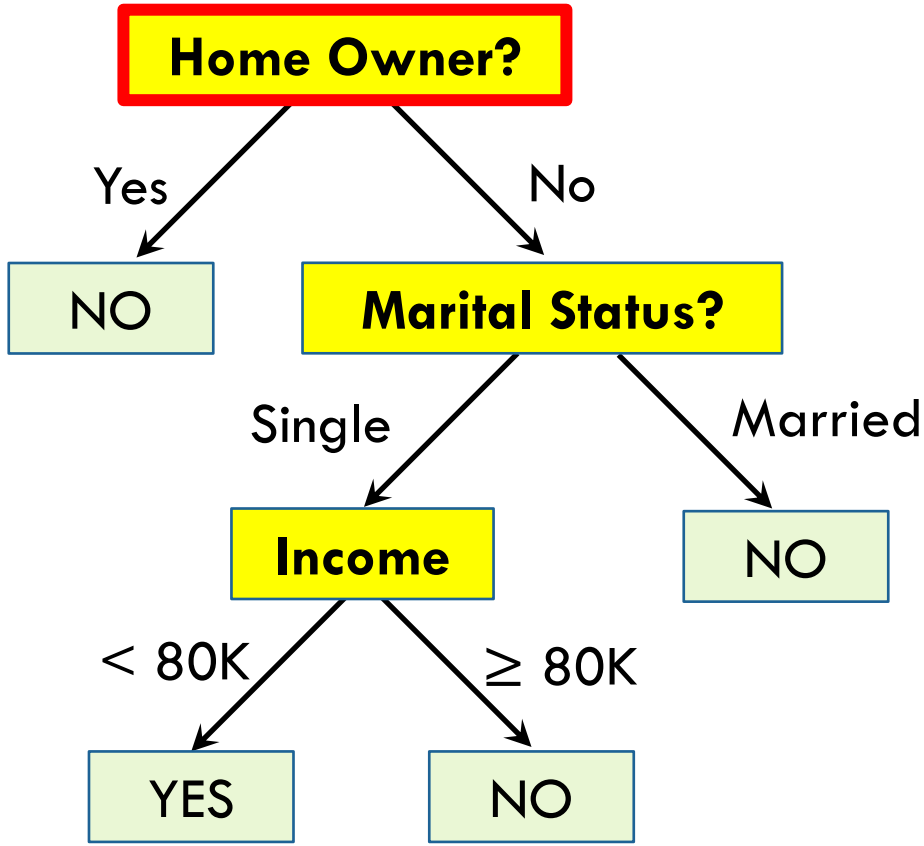
Test Record



APPLYING MODEL TO TEST DATA

	categorical	categorical	continuous	response
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower?
1	No	Married	205K	?

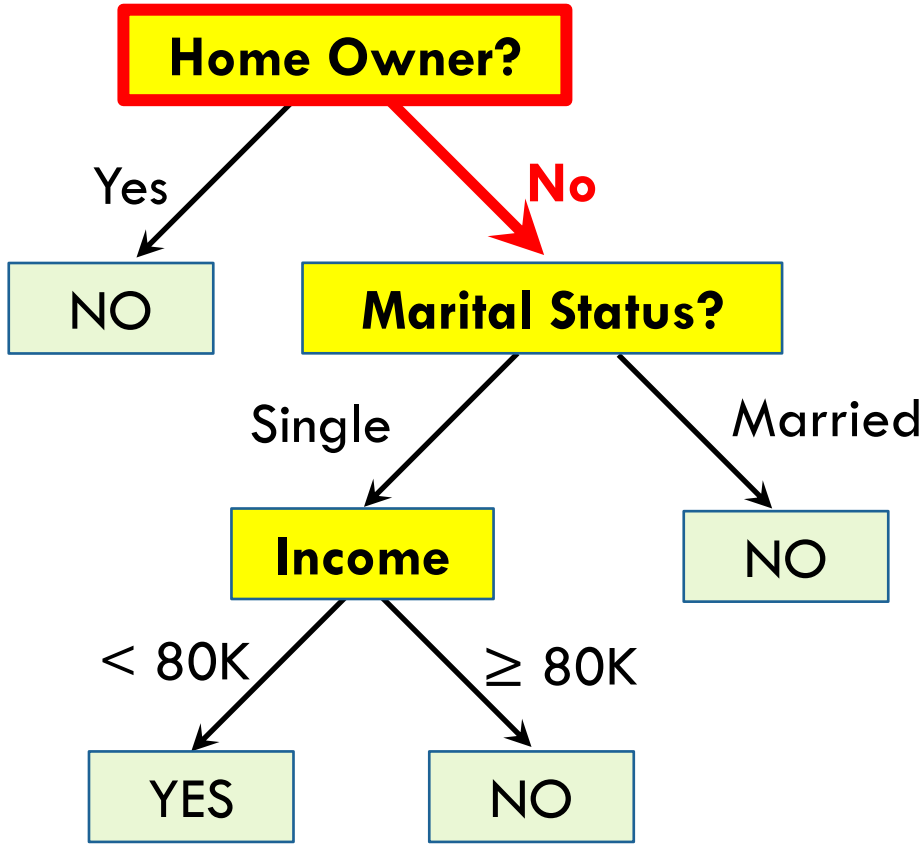
Test Record



APPLYING MODEL TO TEST DATA

	categorical	categorical	continuous	response
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower?
1	No	Married	205K	?

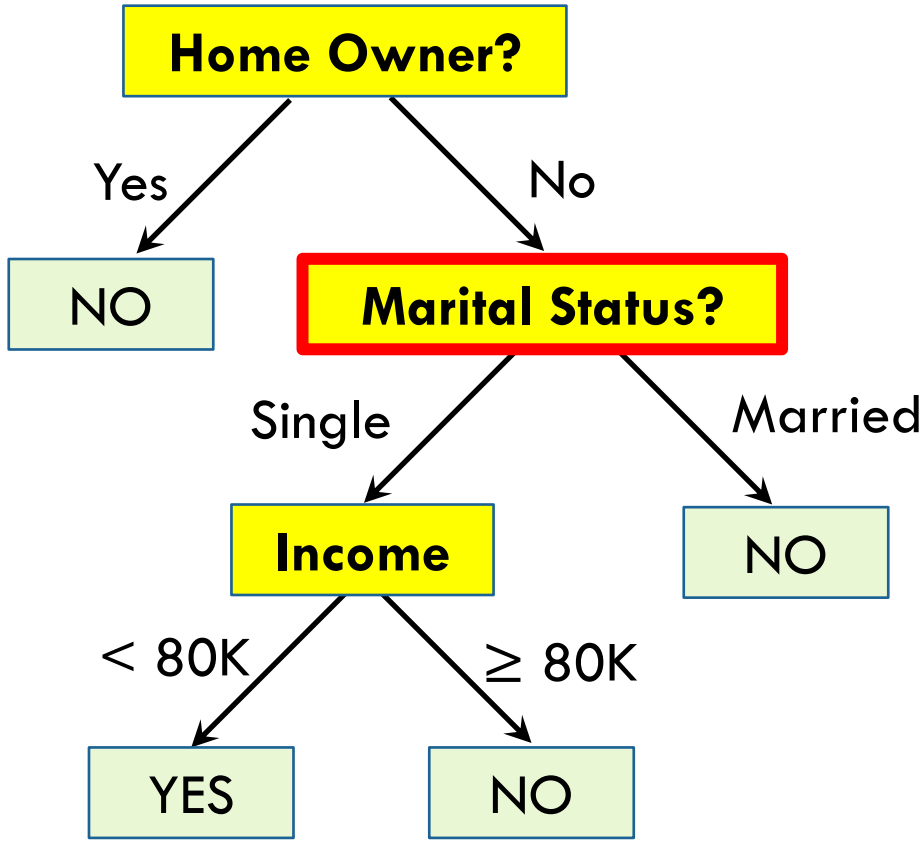
Test Record



APPLYING MODEL TO TEST DATA

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower?
	categorical	categorical	continuous	response
1	No	Married	205K	?

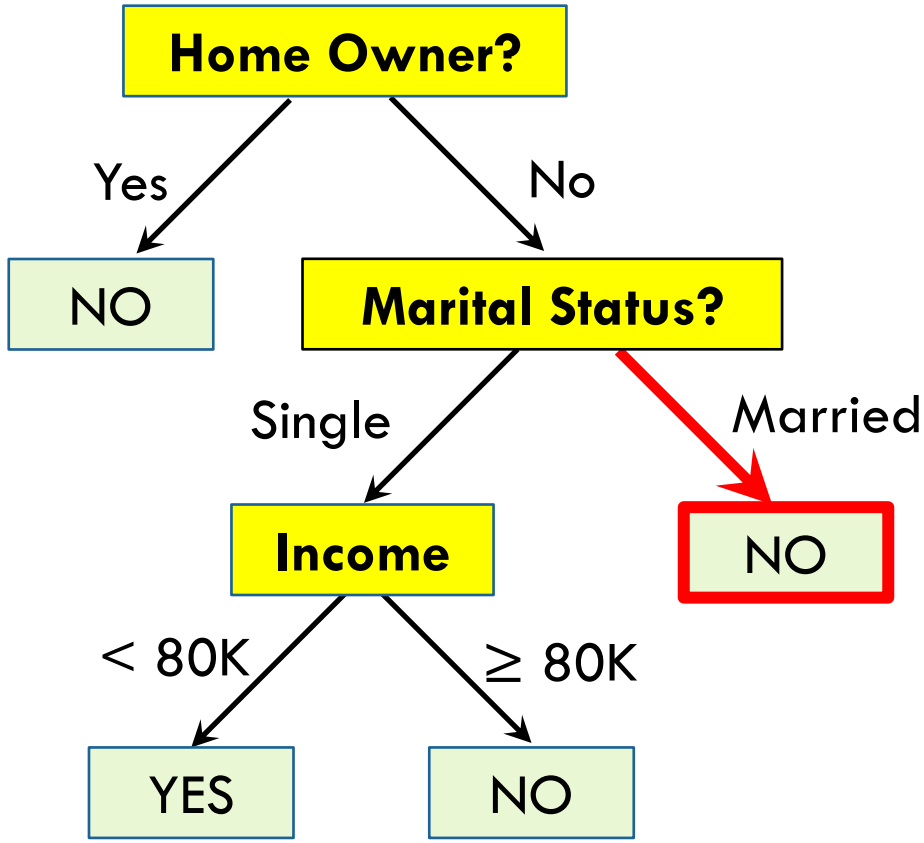
Test Record



APPLYING MODEL TO TEST DATA

	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower?**
1	No	Married	205K	?

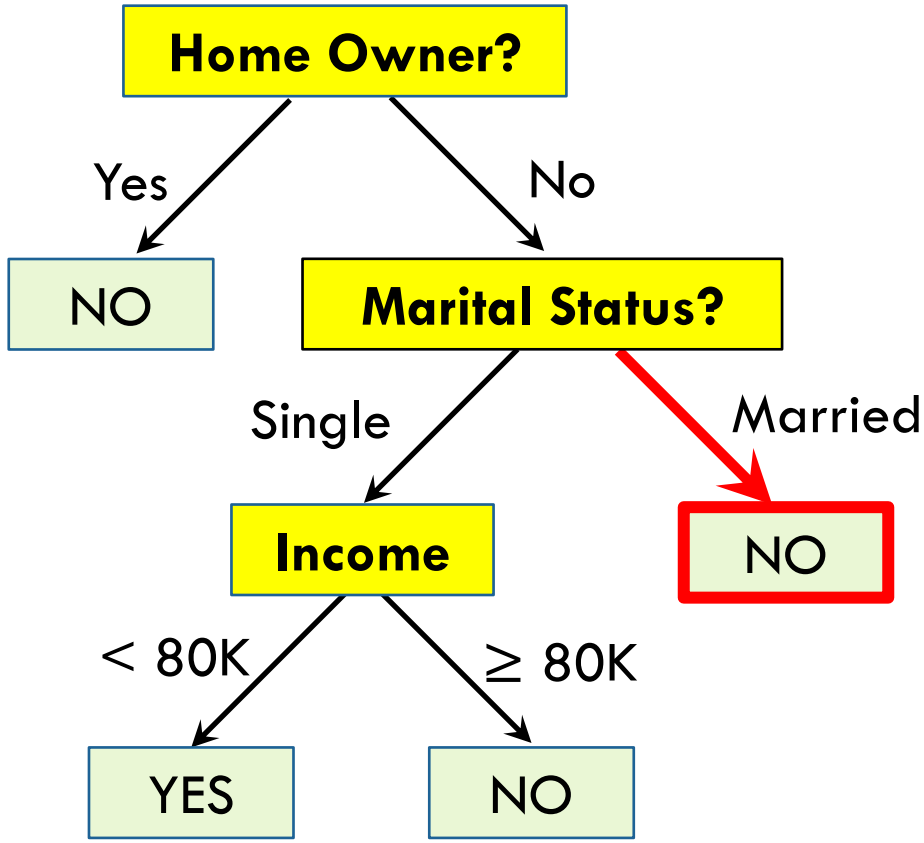
Test Record



APPLYING MODEL TO TEST DATA

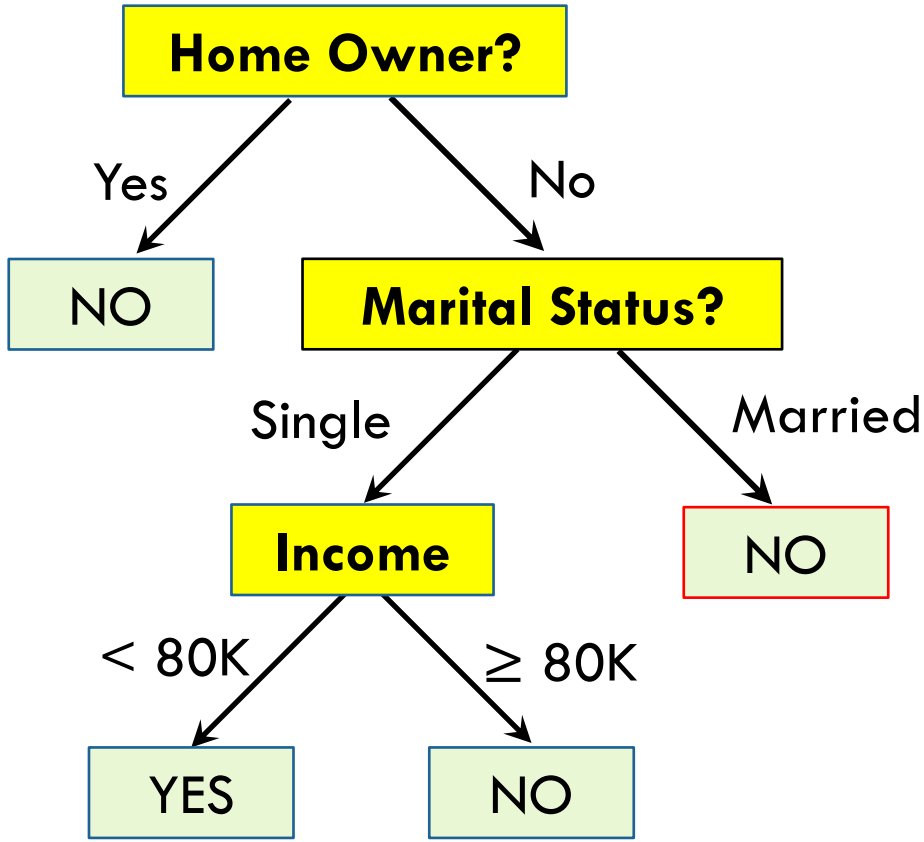
	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower?**
1	No	Married	205K	No

Test Record



APPLYING MODEL TO TEST DATA

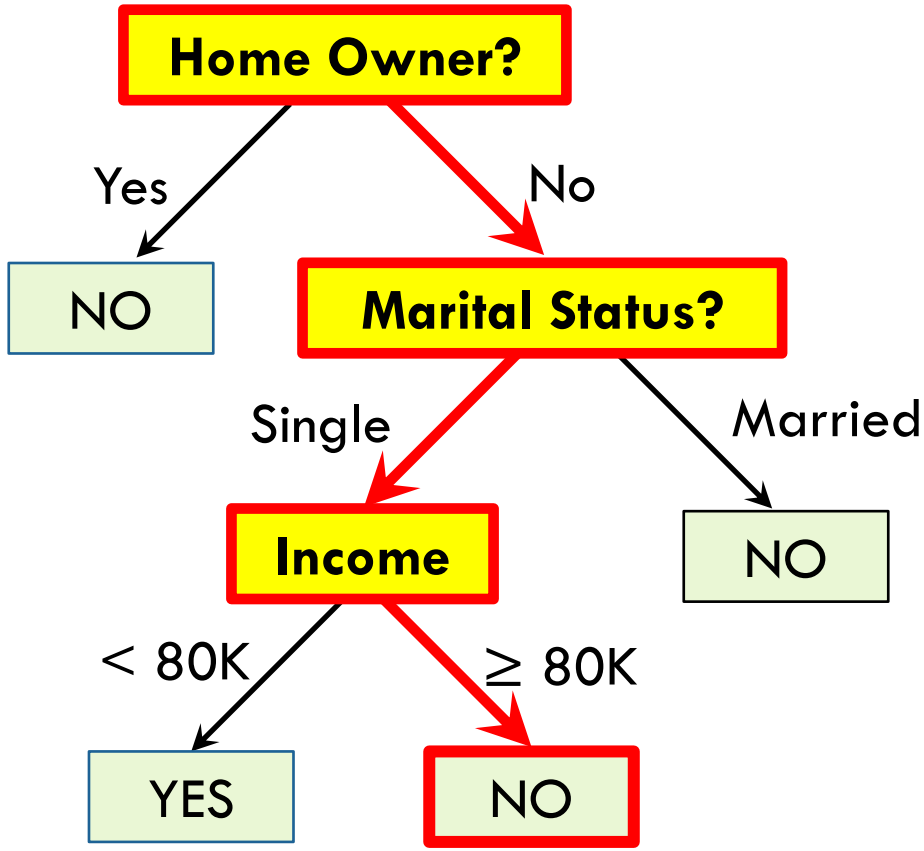
	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower?**
1	No	Single	80K	?



Q: What is the output of the decision tree on this new test record?

APPLYING MODEL TO TEST DATA

	categorical	categorical	continuous	response
ID	**Home Owner**	**Marital Status**	**Annual Income**	**Defaulted Borrower?**
1	No	Single	80K	?



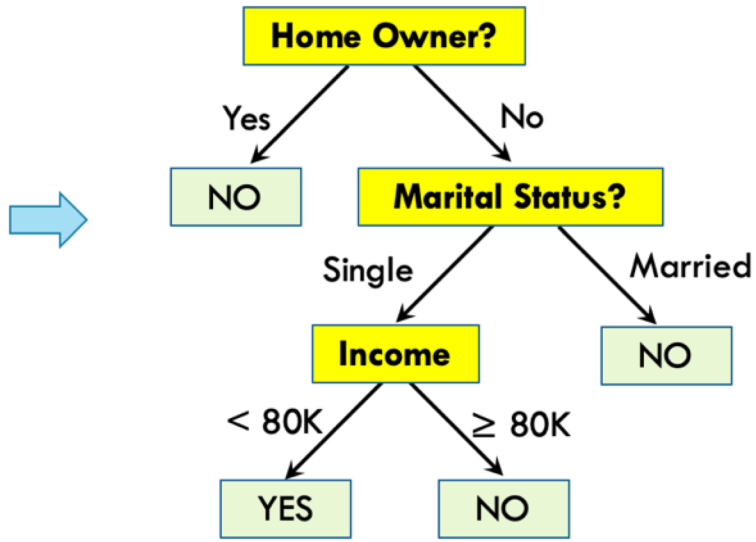
A: NO

HOW TO TRAIN DECISION TREE?

Finding the optimal decision tree (for certain notions of optimality) is NP-complete!

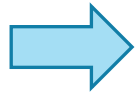
Instead, most algorithms use heuristic (i.e. sub-optimal) approaches

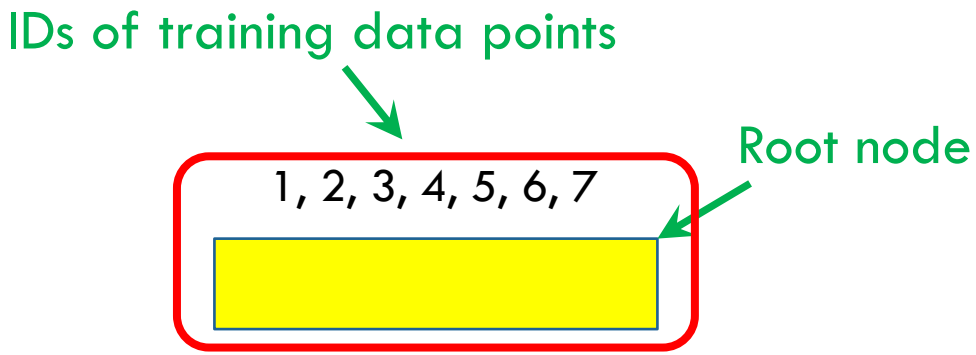
	categorical	categorical	continuous	response
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

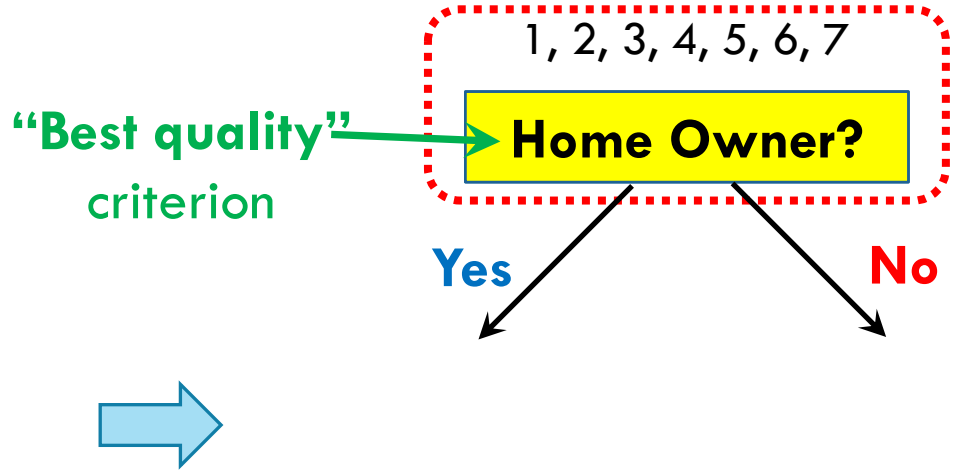




1. Initially, the **root node** contains all the training points

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

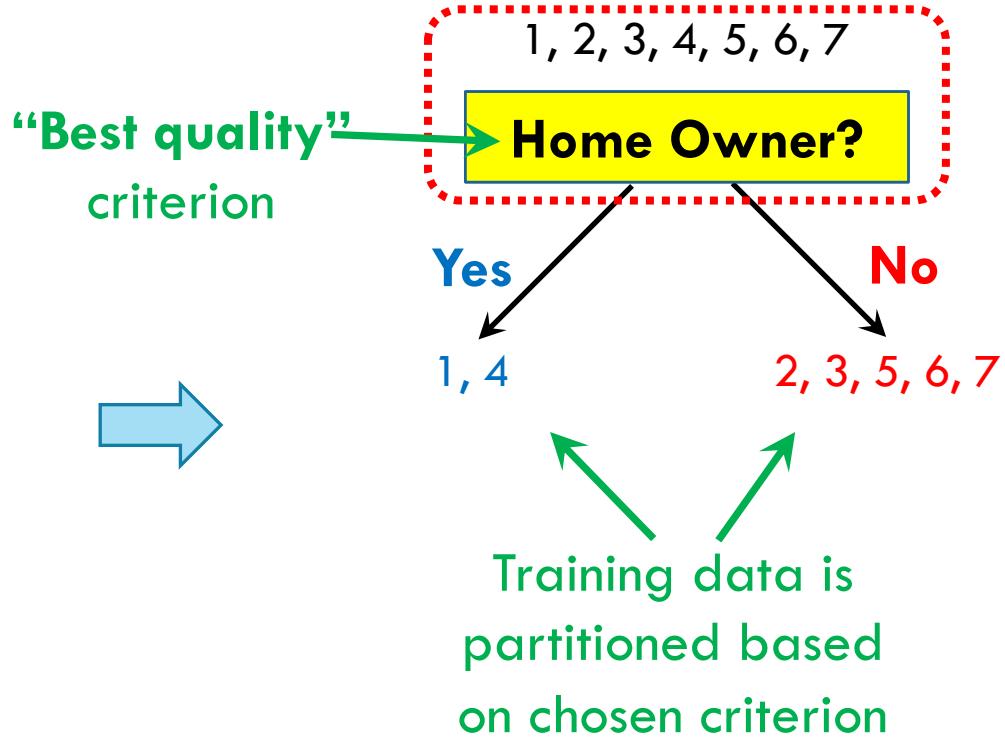


To be defined later

2. Find the **"best quality"** criterion, which partitions the data into 2 subsets

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

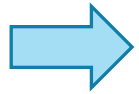


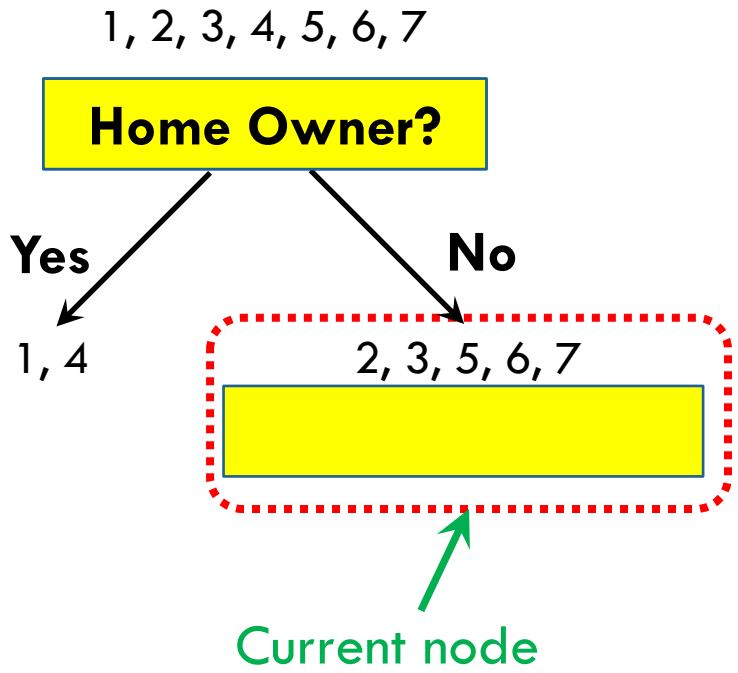
To be defined later

2. Find the **"best quality"** criterion, which partitions the data into 2 subsets

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
2	No	Married	100K	No
3	No	Single	170K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

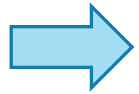


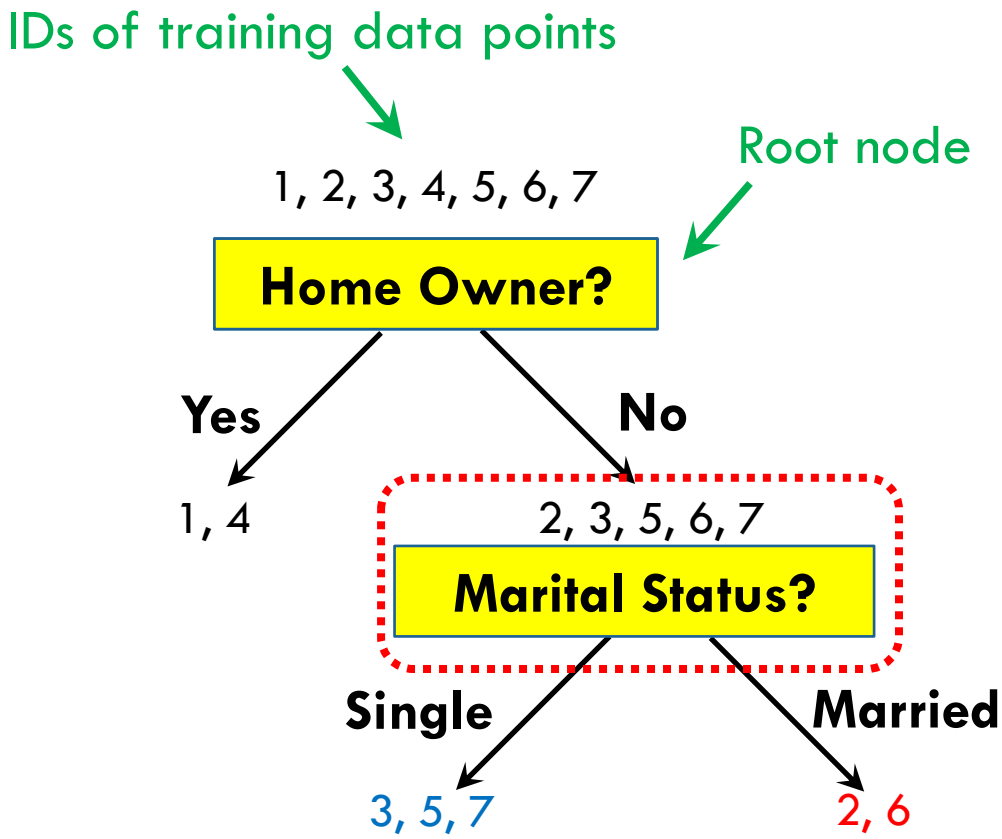


3. Continue recursively onto the resulting subsets (i.e. splitting them further)

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
2	No	Married	100K	No
3	No	Single	170K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

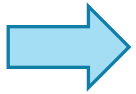


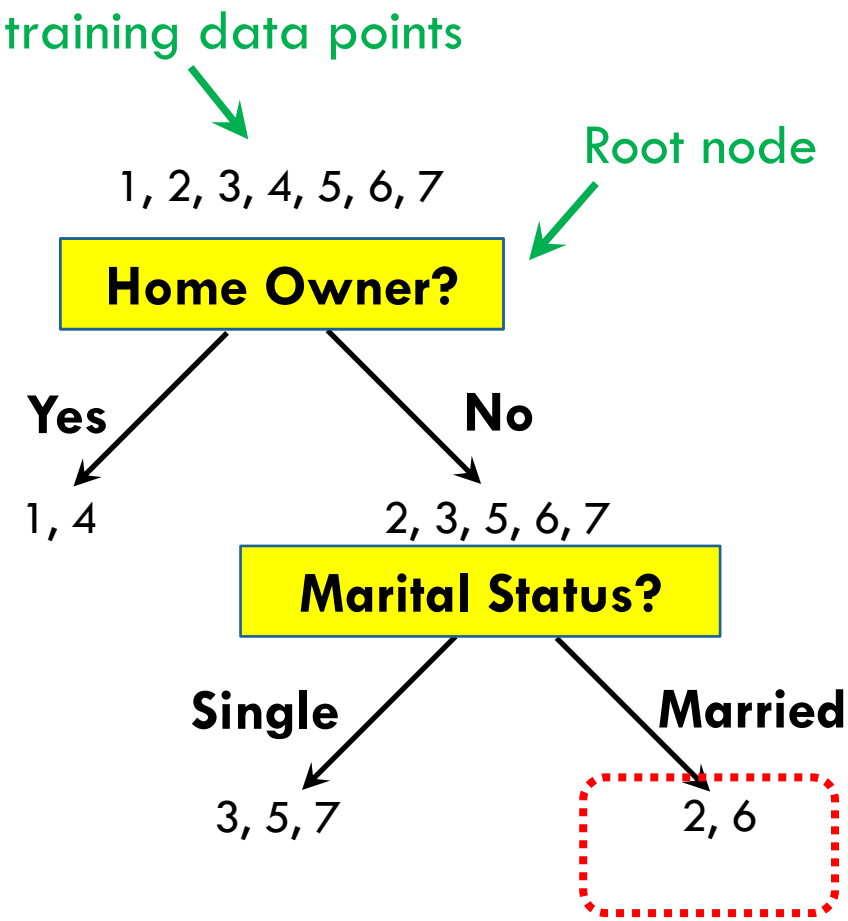


3. Continue recursively onto the resulting subsets (i.e. splitting them further)

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
2	No	Married	100K	No
6	No	Married	160K	No

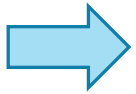


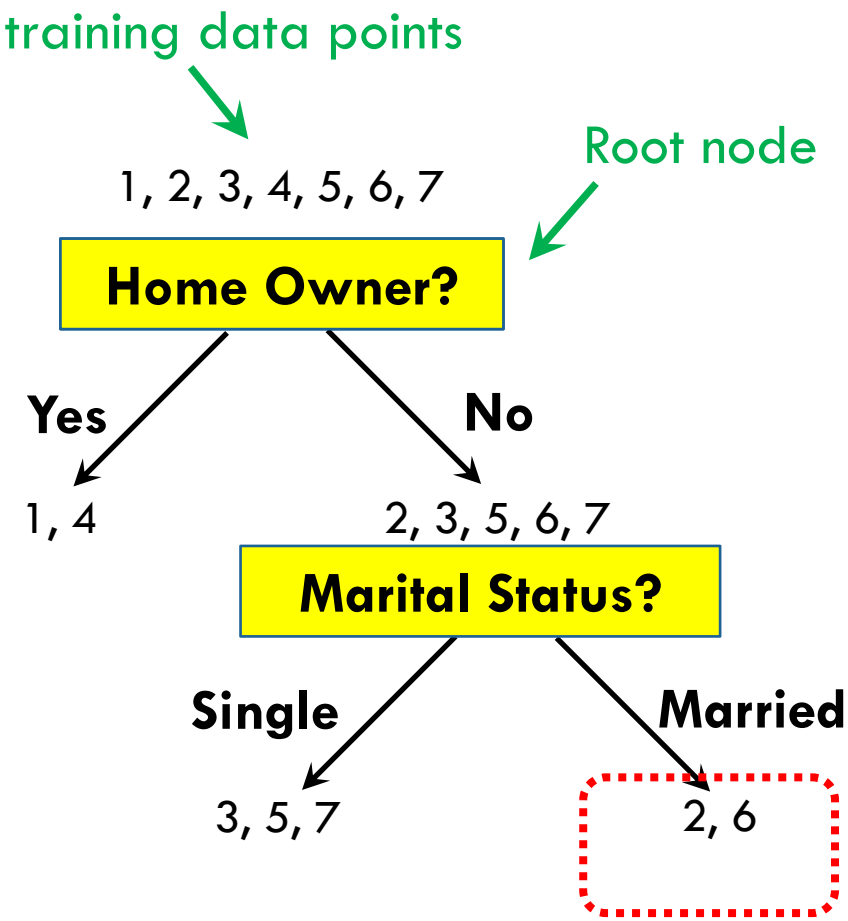


3. Continue recursively onto the resulting subsets (i.e. splitting them further)

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
2	No	Married	100K	No
6	No	Married	160K	No



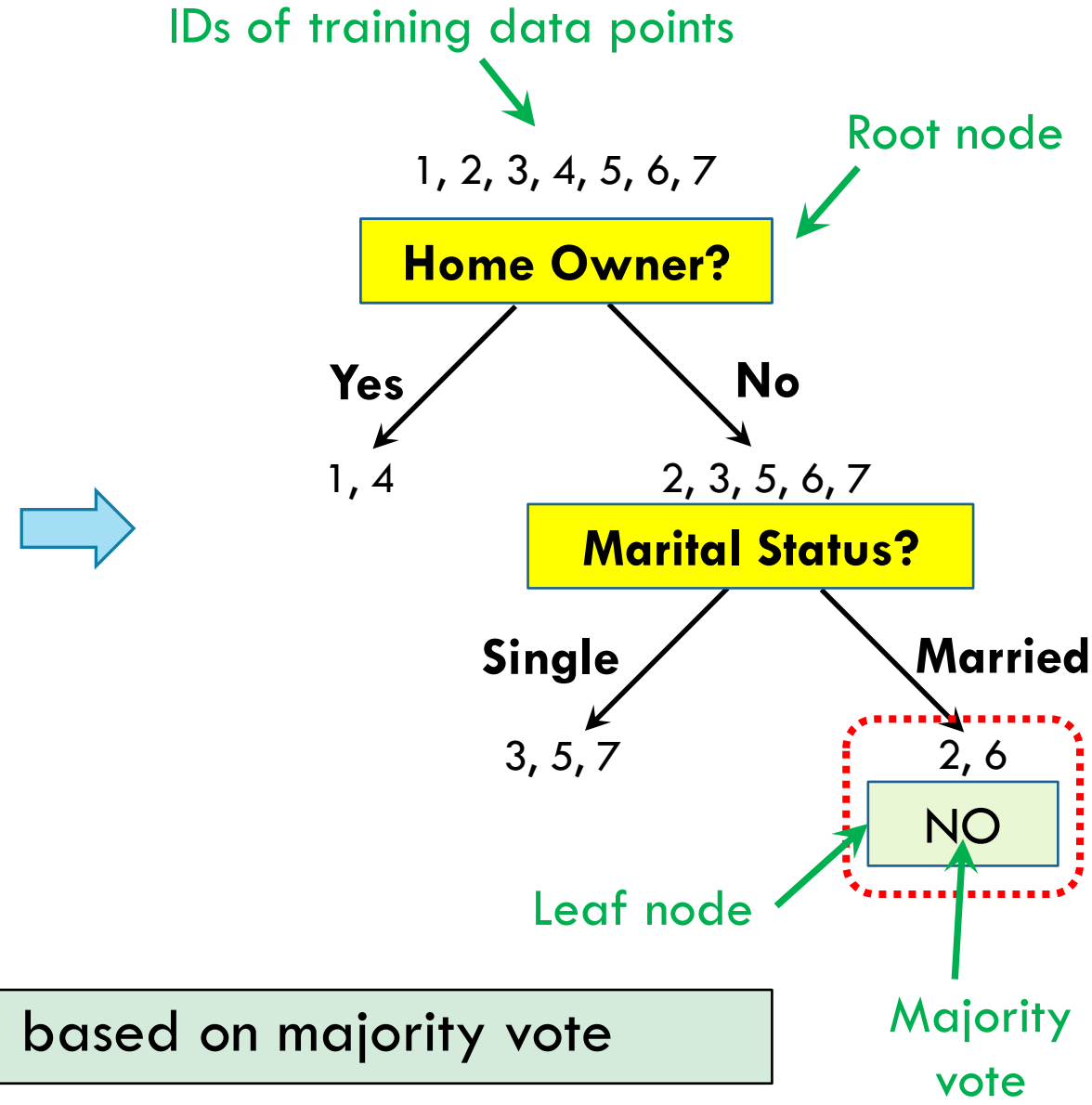


Depth limit = 10

4. Continue until all points have the same response value, or the tree has reached a **depth limit**

TRAINING DECISION TREE

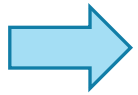
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
2	No	Married	100K	No
6	No	Married	160K	No

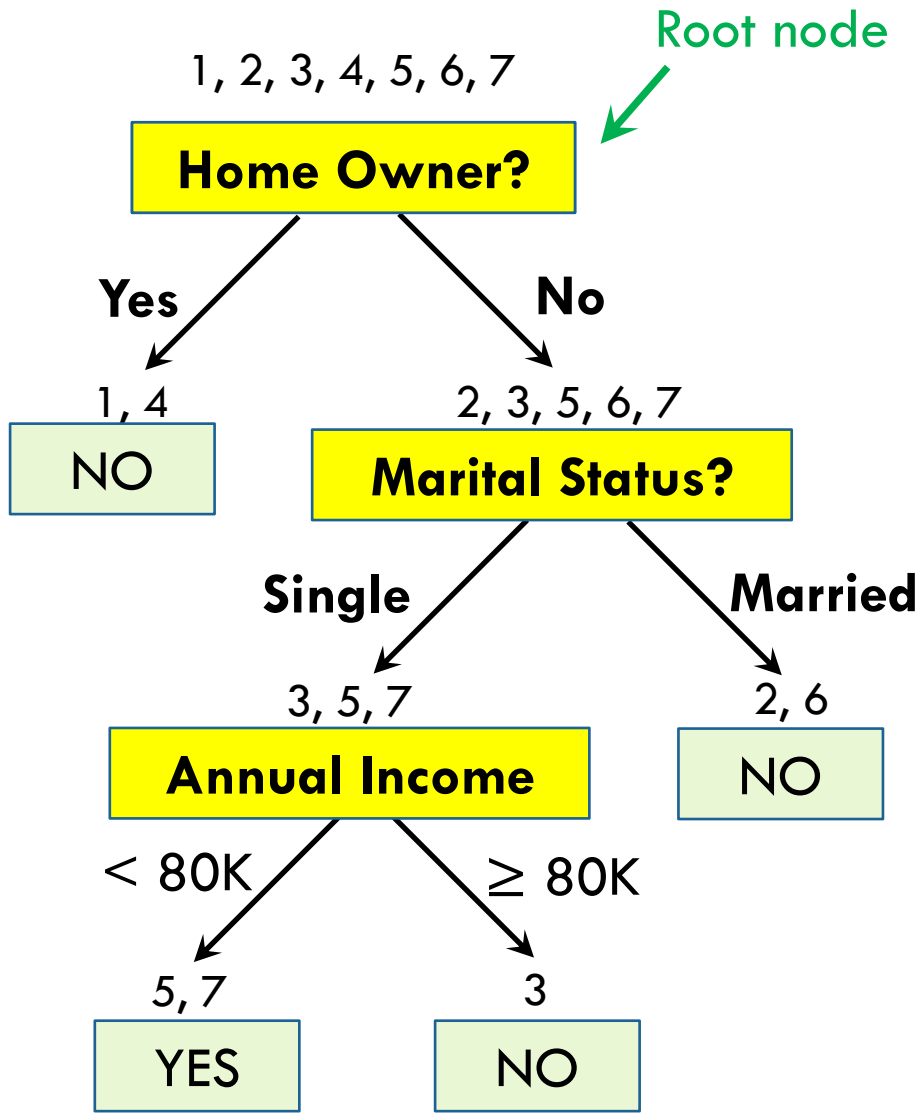


4. On termination, leaf nodes classify the data based on majority vote

TRAINING DECISION TREE

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



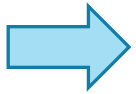


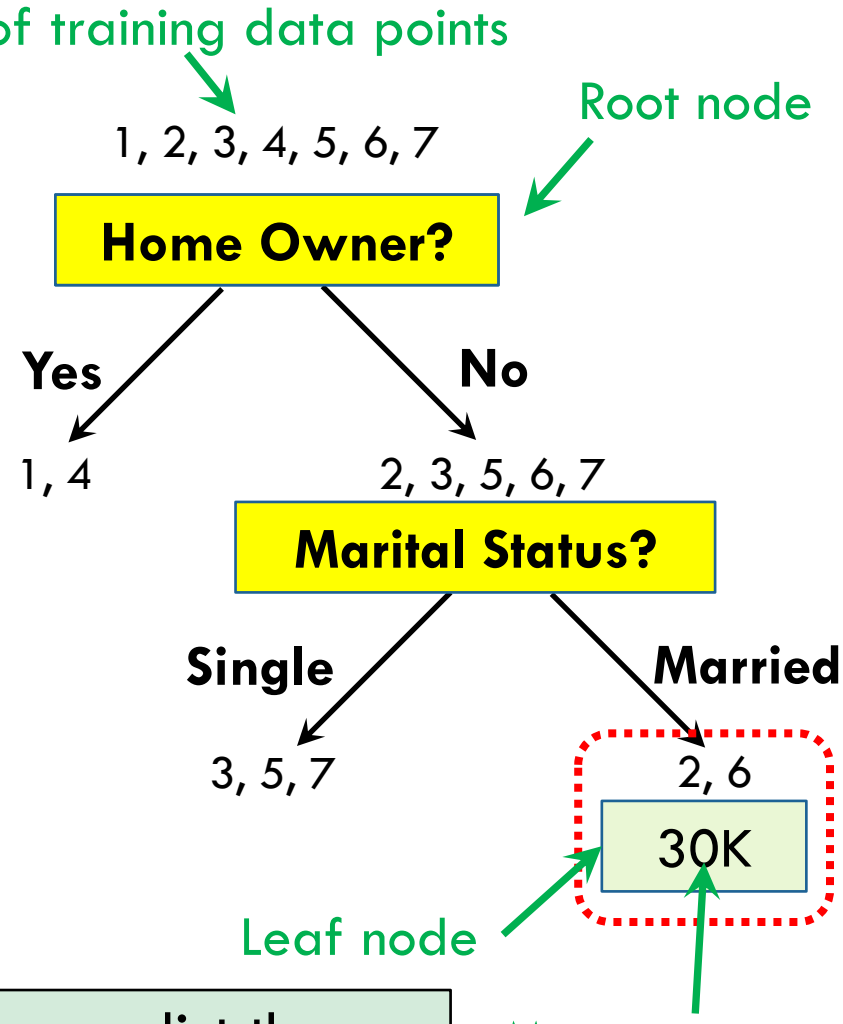
Recursion continues until all leaf nodes are generated

TRAINING REGRESSION TREES (FOR NUMERIC RESPONSE VARIABLE)

Numeric response variable

ID	Home Owner	Marital Status	Annual Income	Defaulted Amount
2	No	Married	100K	20K
6	No	Married	160K	40K

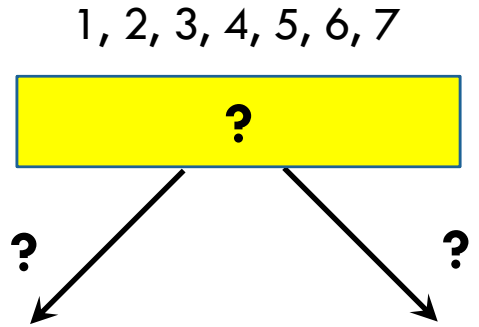




Regression trees: instead of using majority vote at leaf nodes, we predict the *mean* of the response variable for training points assigned to the leaf nodes

HOW TO FIND THE BEST SPLIT?

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

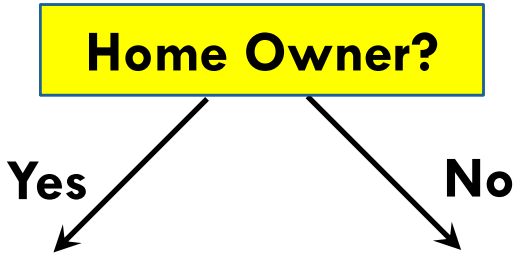


Steps:

1. Try all **valid splits** of each feature
2. Measure the **quality** of each such split
3. Return the highest quality split

SPLITTING BINARY VARIABLES

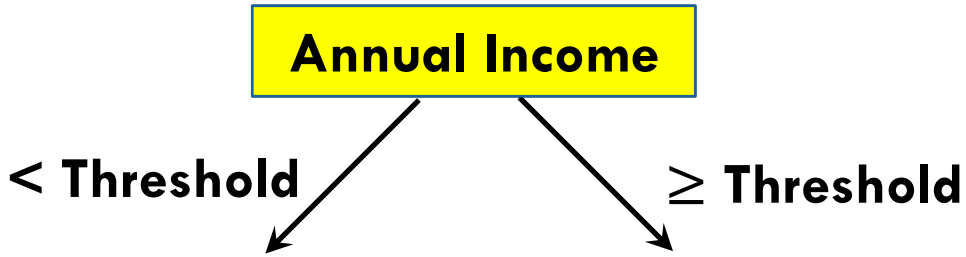
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



Binary Variables: partition the records into 2 sets based on this variable

SPLITTING NUMERIC VARIABLES

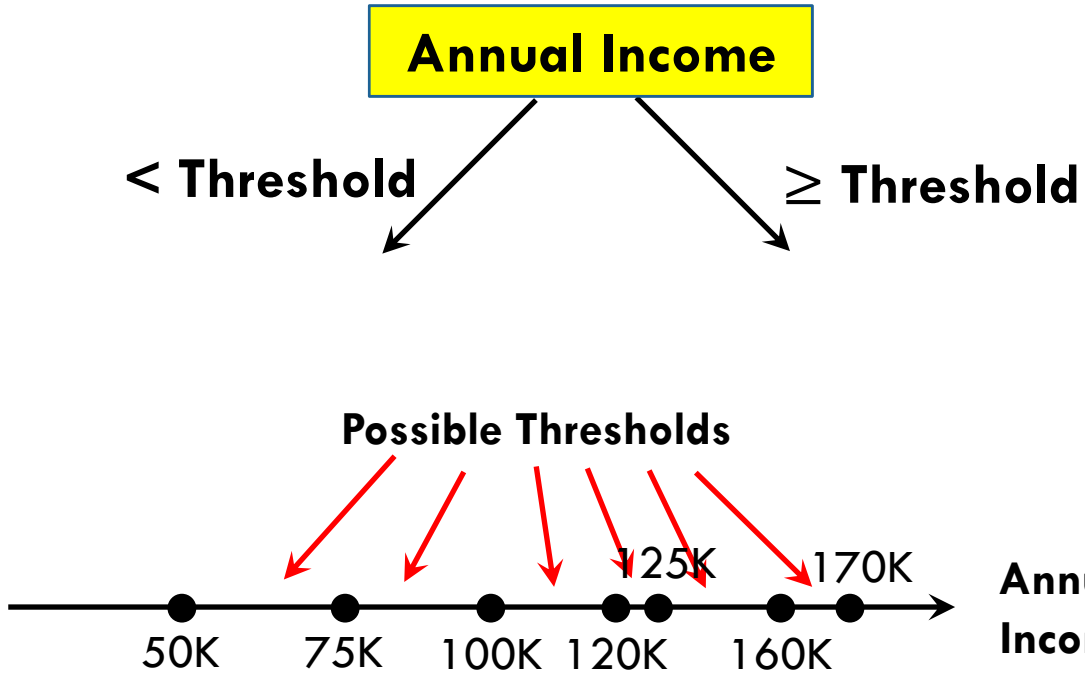
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



Numeric Variables: consider all possible partitions into below and above a threshold

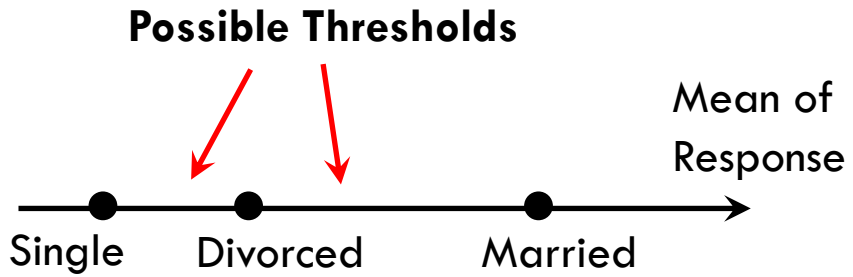
SPLITTING NUMERIC VARIABLES

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



Numeric Variables: consider all possible partitions into below and above a threshold

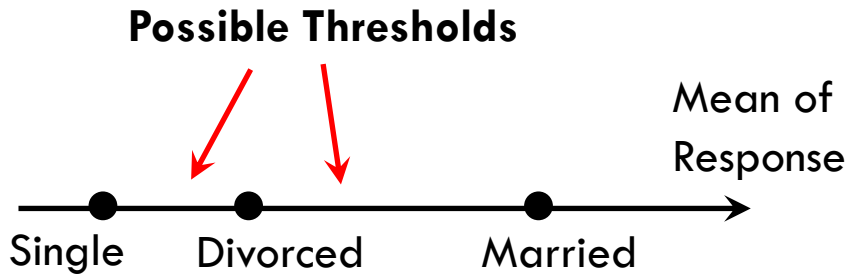
SPLITTING CATEGORICAL VARIABLES



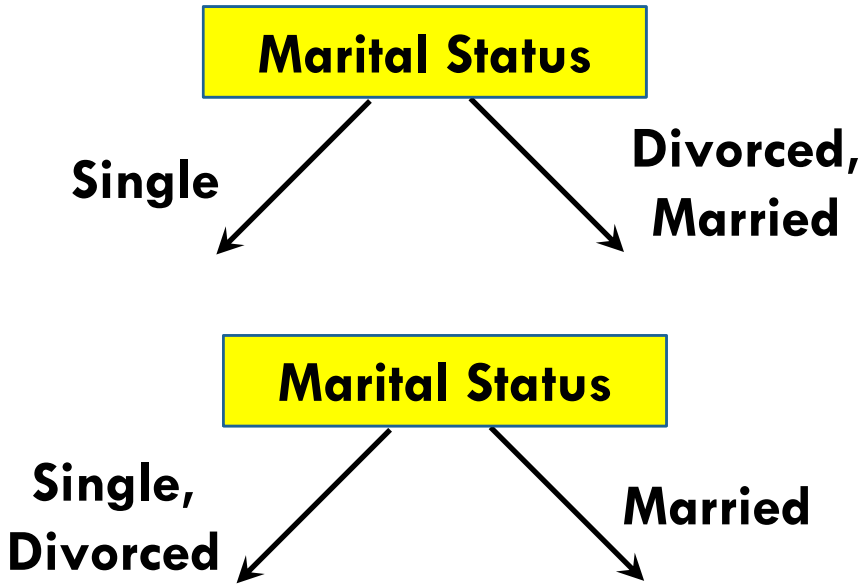
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

Categorical Variables: sort the categories by mean response, and treat as continuous variable

SPLITTING CATEGORICAL VARIABLES



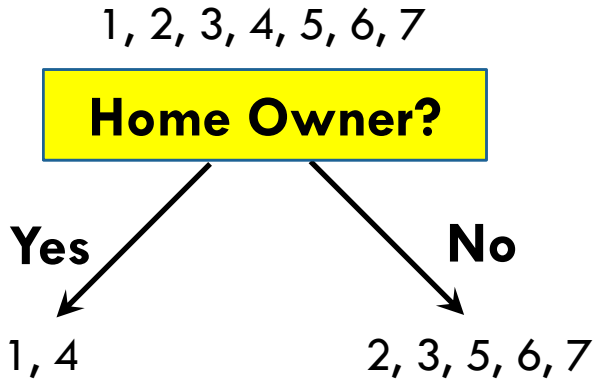
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



Categorical Variables: sort the categories by *mean response*, and treat as continuous variable

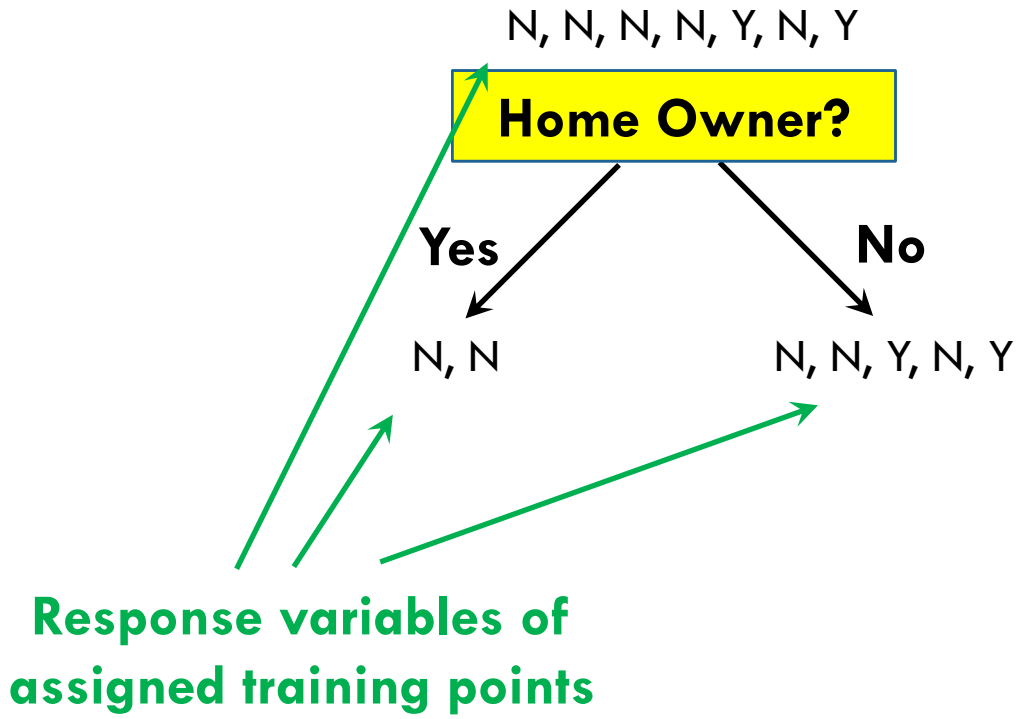
HOW TO EVALUATE THE QUALITY OF A SPLIT?

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



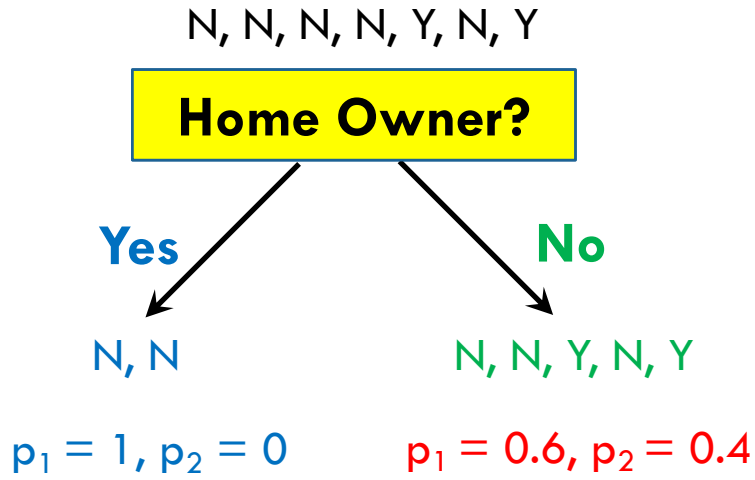
EVALUATING SPLIT QUALITY: GINI INDEX

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



EVALUATING SPLIT QUALITY: GINI INDEX

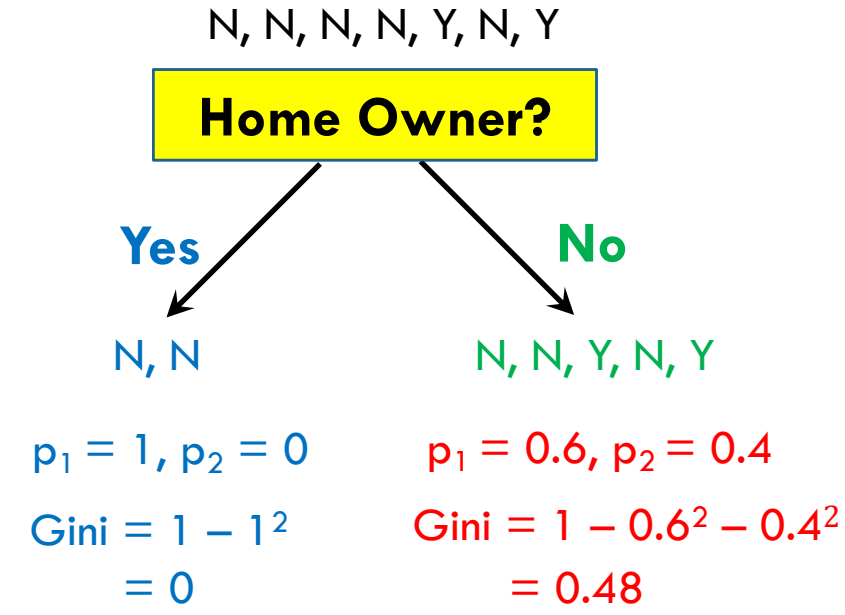
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



$$\text{Gini Index} = 1 - \sum_{i=1}^{J} {p_i}^2$$

EVALUATING SPLIT QUALITY: GINI INDEX

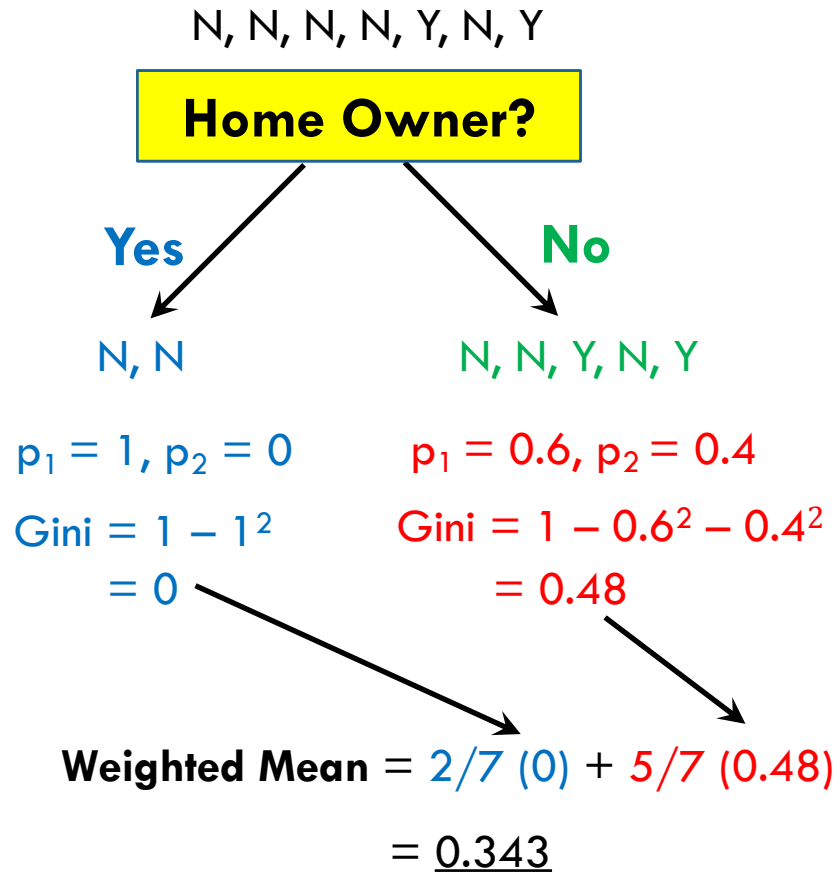
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



$$\text{Gini Index} = 1 - \sum_{i=1}^{J} {p_i}^2$$

EVALUATING SPLIT QUALITY: GINI INDEX

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

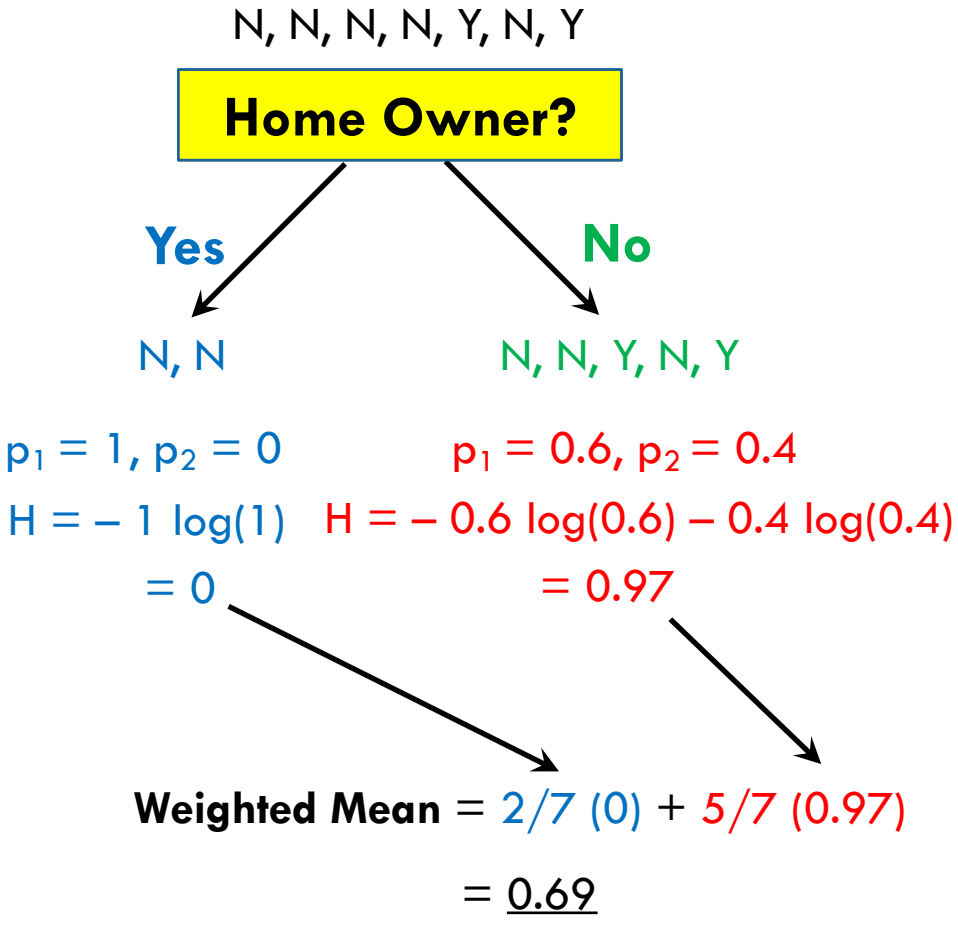


$$\text{Gini Index} = 1 - \sum_{i=1}^{J} {p_i}^2$$

(Lower Gini = Purer distribution = Better split)

EVALUATING SPLIT QUALITY: ENTROPY

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes



Entropy: H $= -\sum_{i=1}^{J} p_i \log_2 p_i$

(Lower Entropy = Purer distribution = Better split)

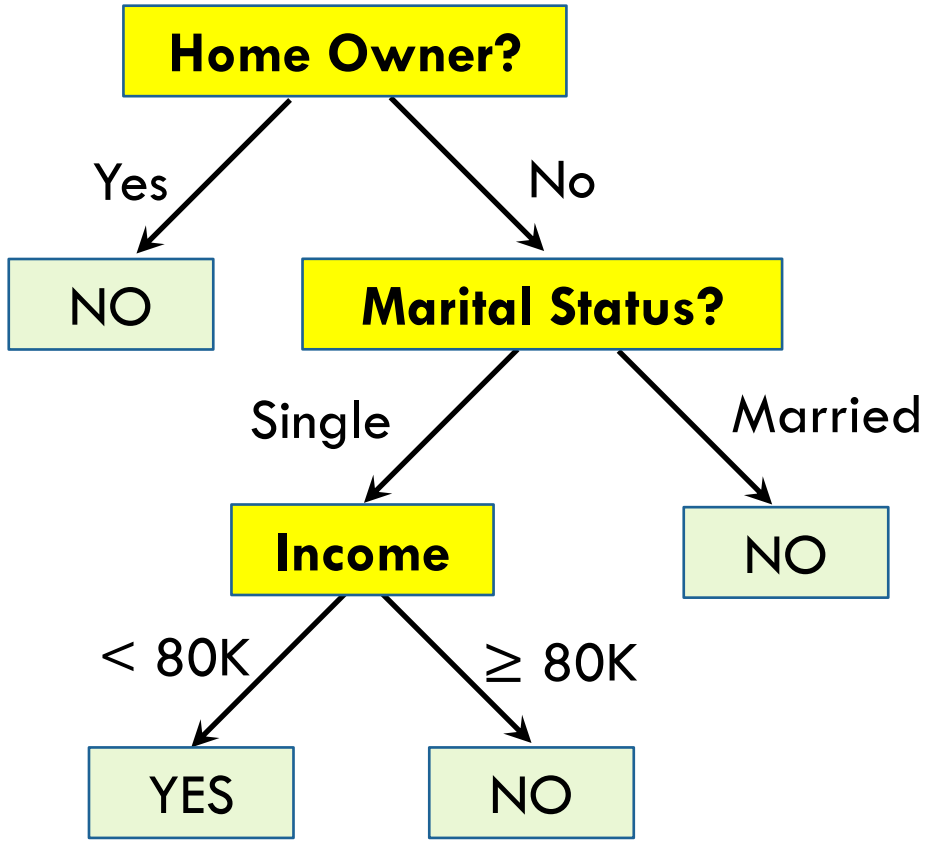
PROS AND CONS

Pros

- Fast to train / test
- Interpretable (mimic human reasoning)
- Can handle categorical and continuous features

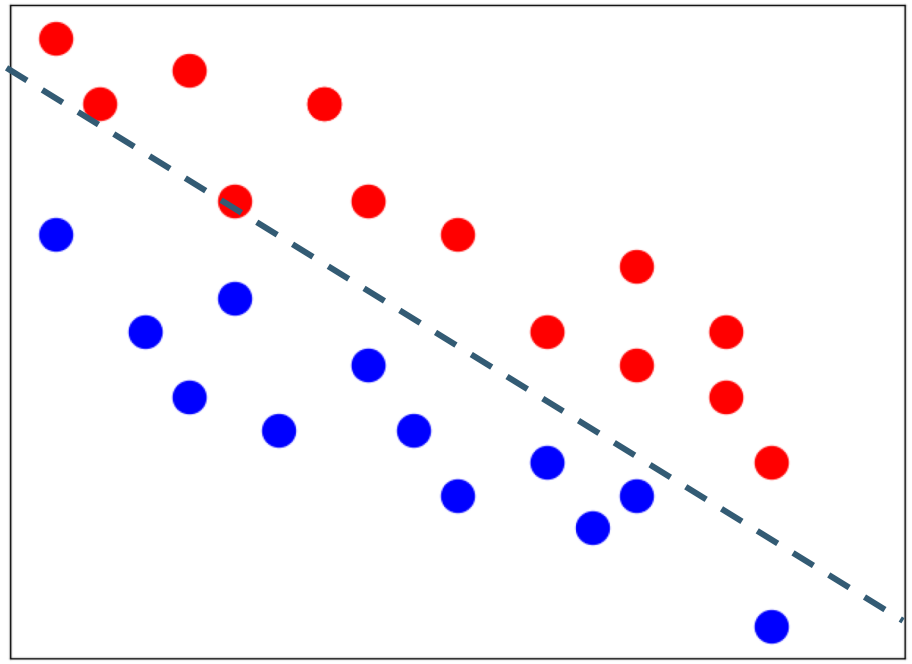
Cons

- Not robust: small changes in data can result in completely different trees
- Greedy approach may be suboptimal
- Relatively poor accuracy

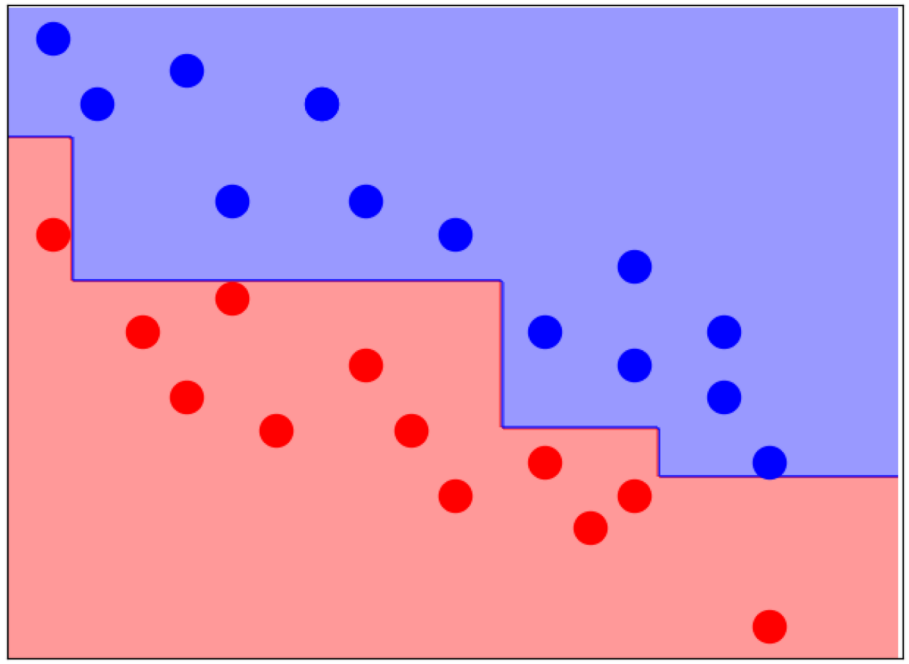


PROPERTIES – MODELLING INTERACTIONS

Optimal decision boundary

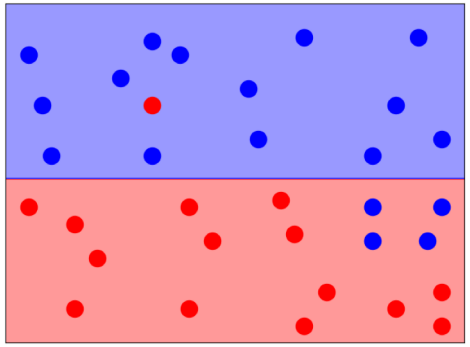


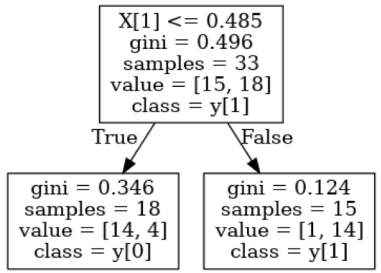
Decision Tree boundaries



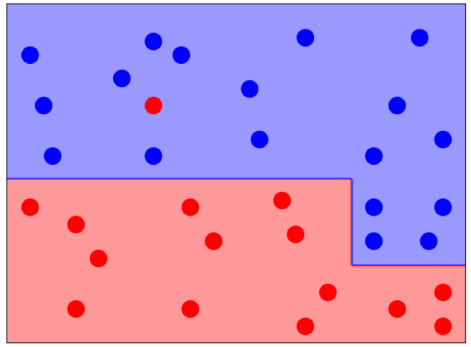
PROPERTIES – UNDERFITTING VS OVERFITTING

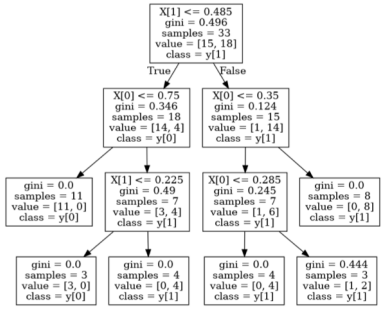
Underfitting



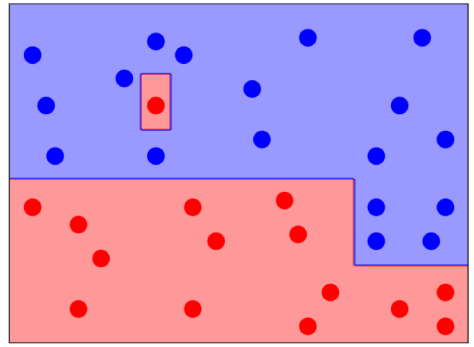


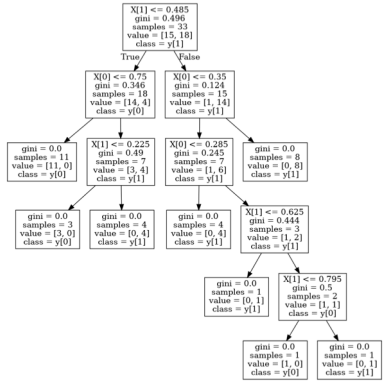
Good fit



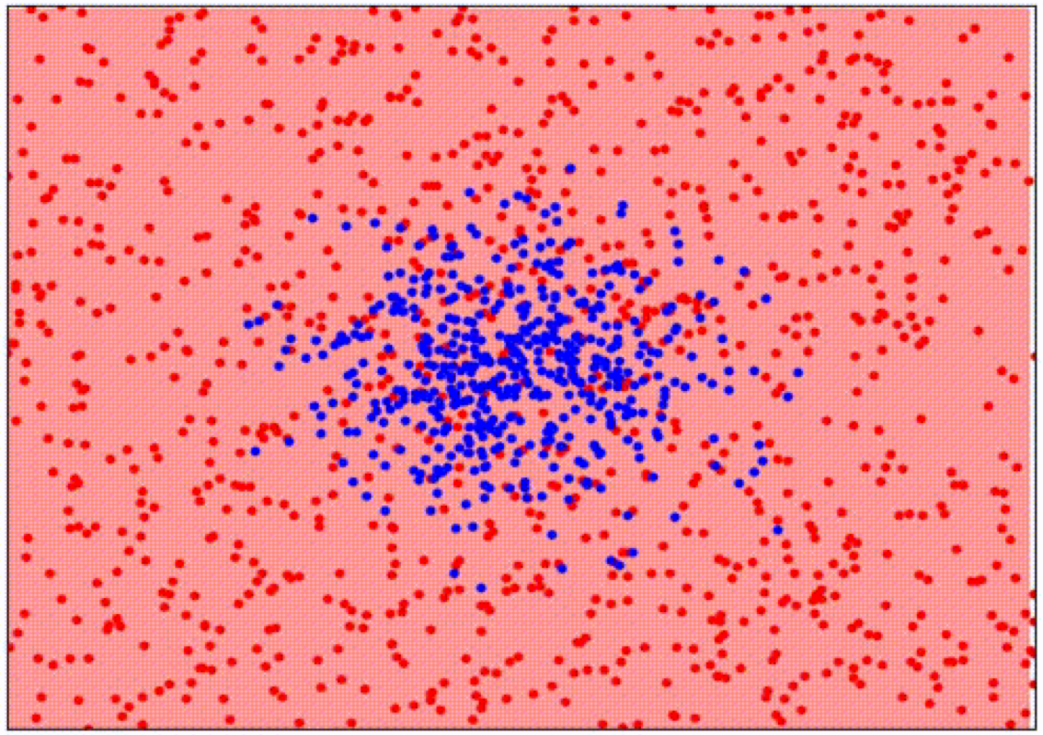


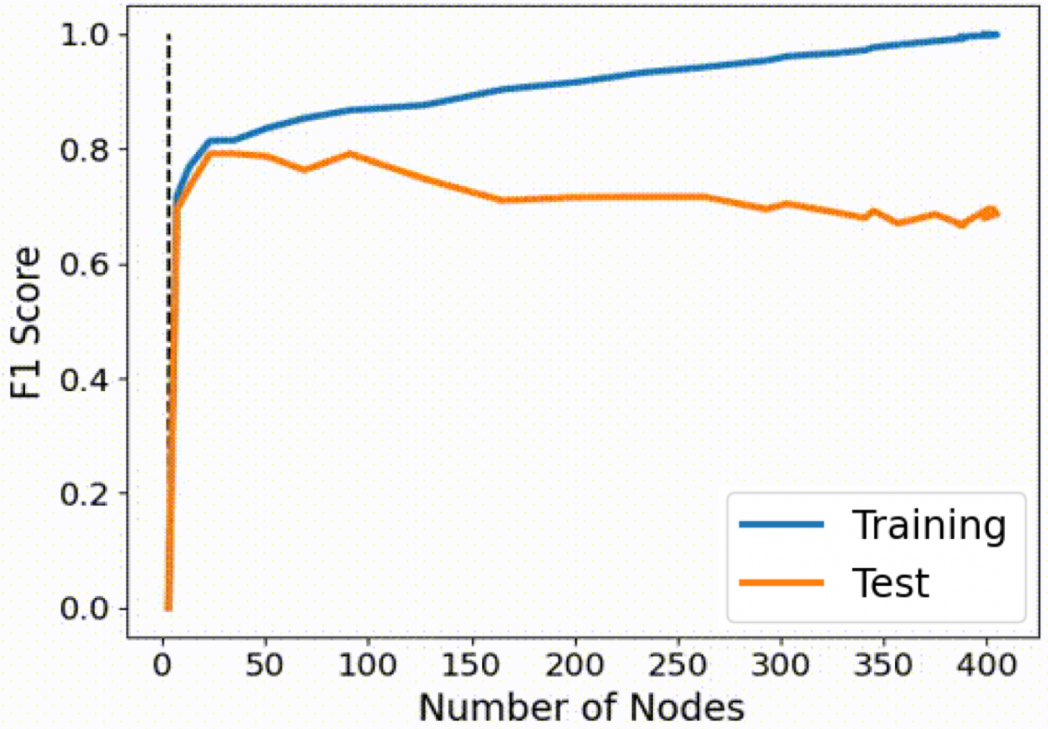
Overfitting





PROPERTIES – UNDERFITTING VS OVERFITTING

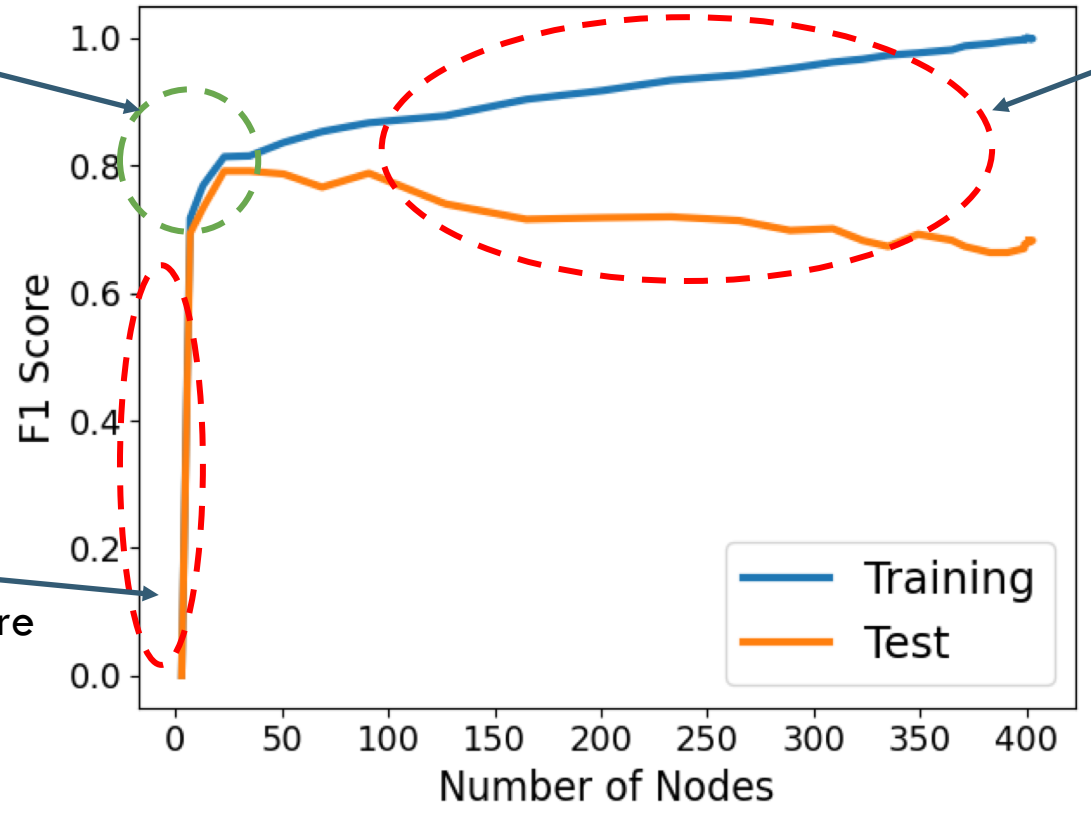




Credit: https://www.comp.nus.edu.sg/~chris/downloads/images/decision-tree-overfitting.gif

PROPERTIES – UNDERFITTING VS OVERFITTING

"Sweet Spot"



Overfitting
Increasing training score, decreasing test score

Underfitting
Both training and test score (very) low

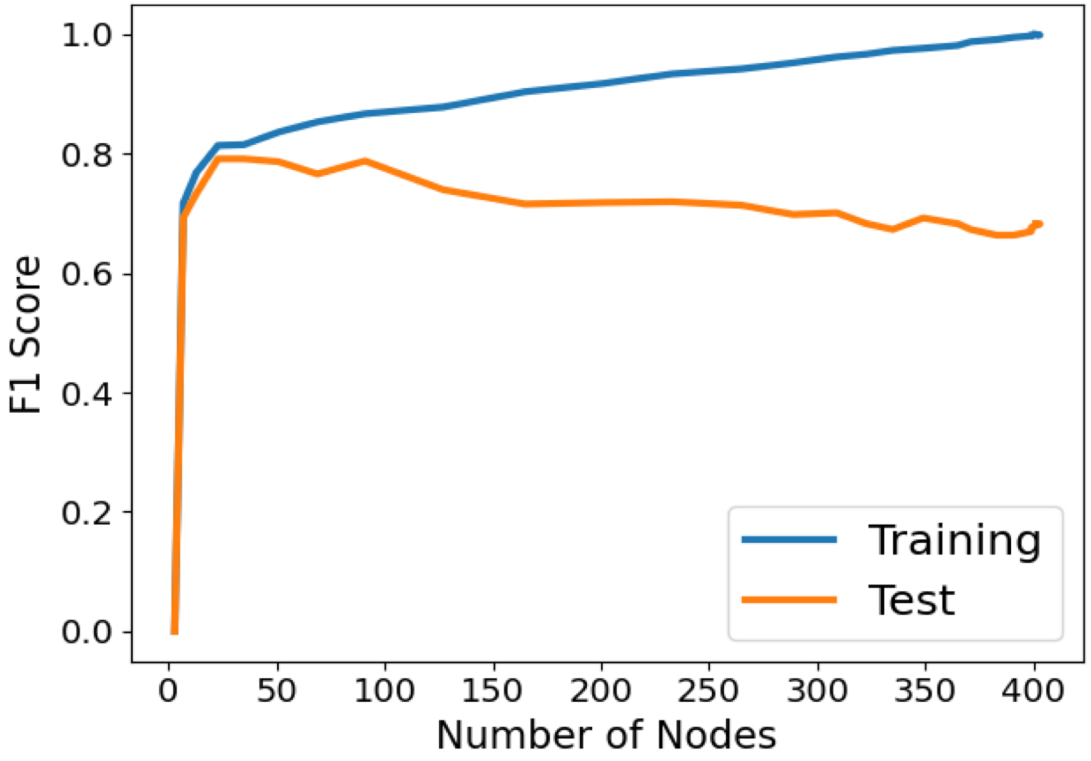
AVOIDING OVERFITTING

Issue: Decision Tree algorithm can always split the training data perfectly (assuming no duplicates)

- It can keep splitting (i.e., increase height of tree) until each leaf contains only one data sample
- One data sample at each leaf → 100% pure

Solution: Limit size/height of Decision Tree → Pruning

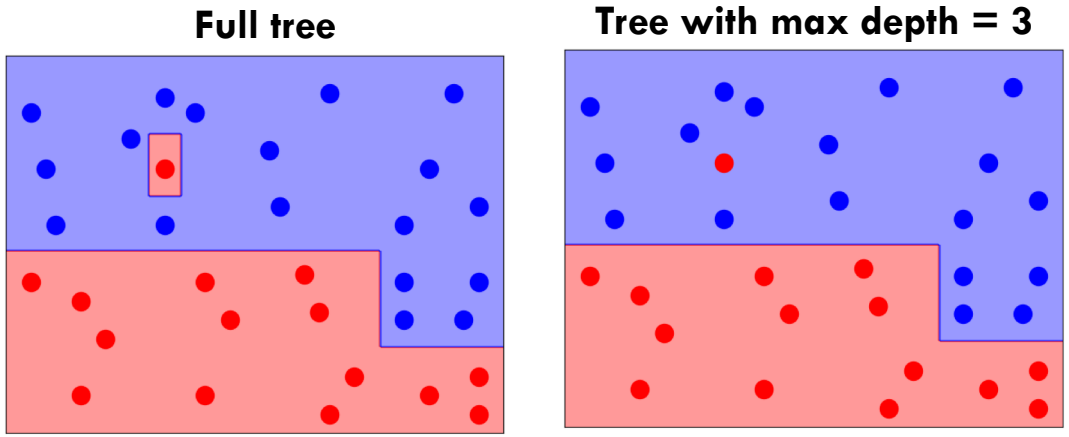
- Pre-pruning: Stop splitting nodes ahead of time
- Post-pruning: Build full tree, but then remove leaves/splits if beneficial
- … combination of multiple approaches

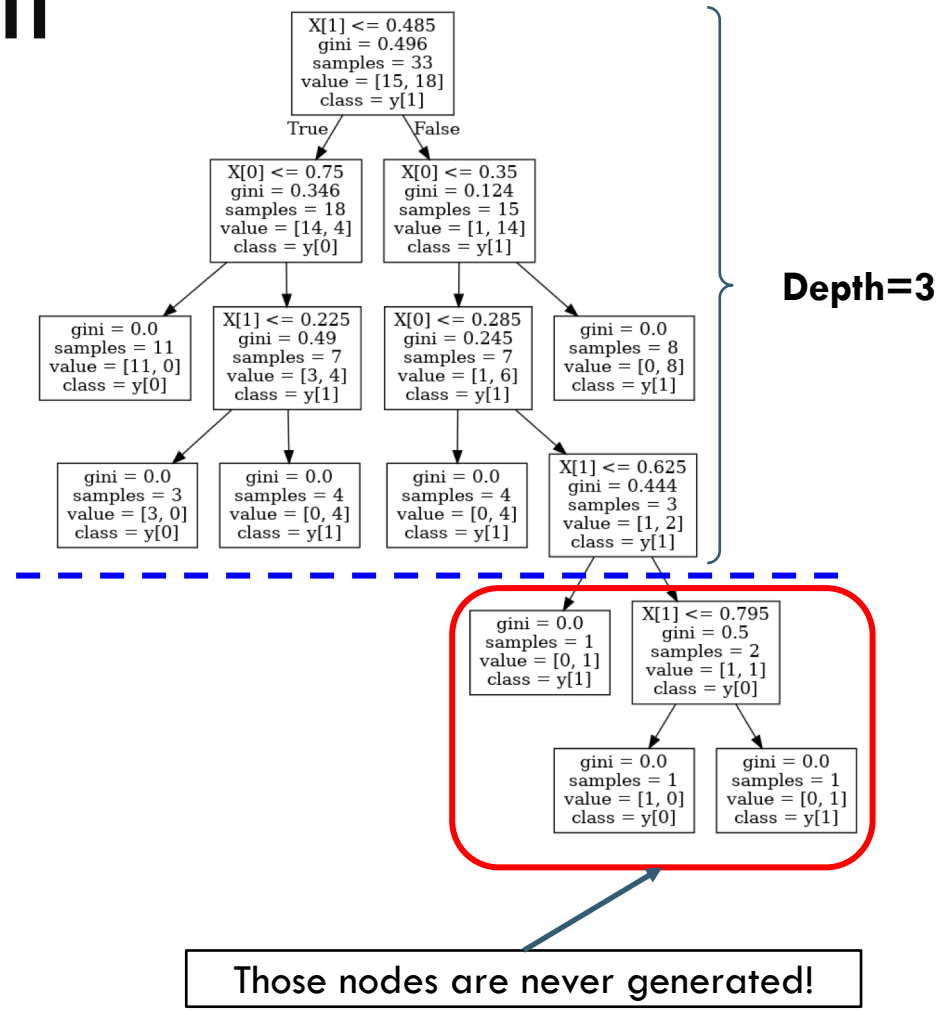


PRE-PRUNING: MAXIMUM DEPTH

Stop splitting if maximum depth (set by user) is reached, e.g., depth=3

- Corresponds to `max_depth` in scikit-learn

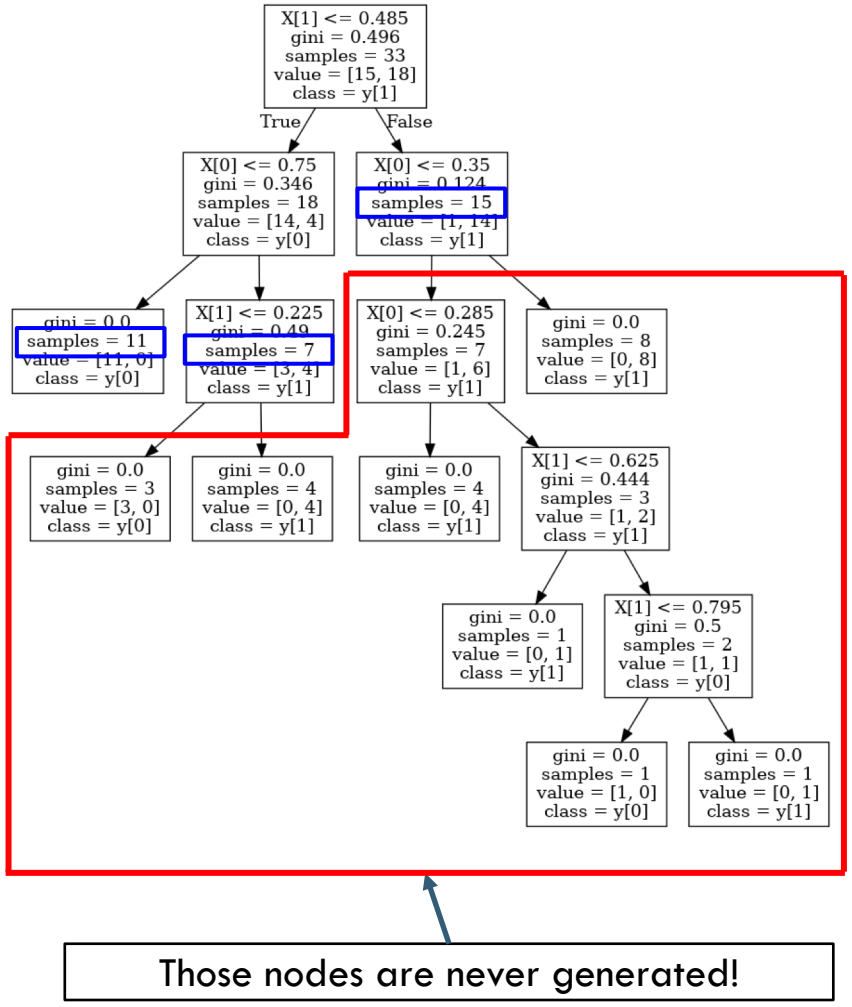




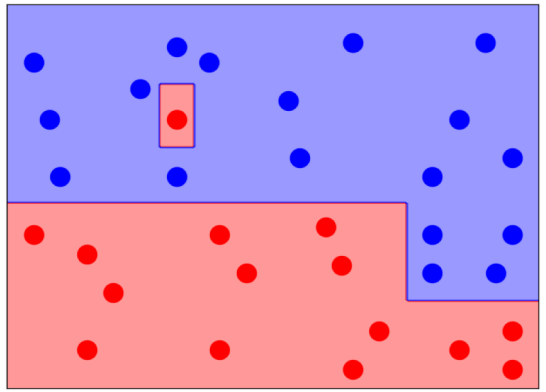
PRE-PRUNING: MINIMUM SAMPLE COUNT

Stop splitting if a node has less than the minimum number of samples: e.g. minimum sample count = 16

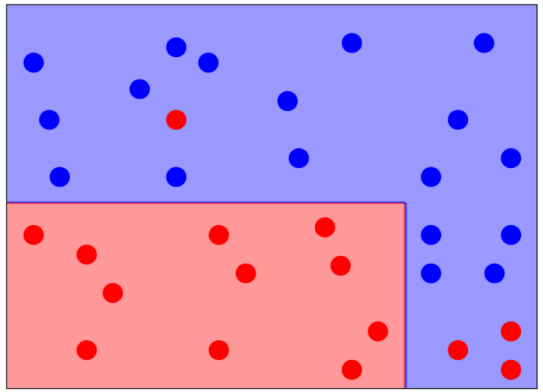
- Corresponds to `min_samples_split` in scikit-learn



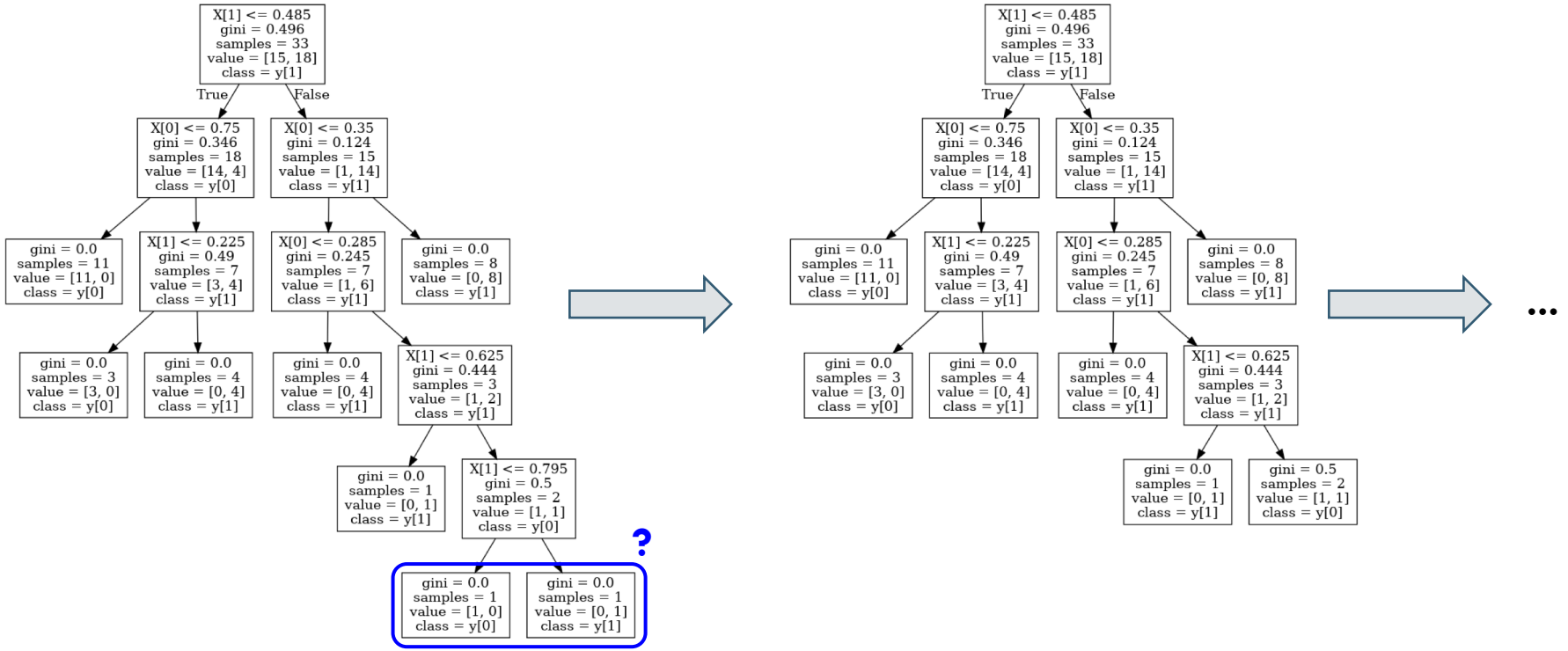
Full tree



Tree with min sample count = 16



POST-PRUNING: COST-COMPLEXITY PRUNING



$$Cost(T) = Error(T) + \alpha \cdot |T| \qquad Cost(T') = Error(T') + \alpha \cdot |T'|$$

Complexity hyperparameter

Number of leaves

$Cost(T) > Cost(T')$ → Remove split from Decision Tree (and continue checking next split)

DECISION TREES WHEN RESCALING VARIABLES

Q: Is there any change to a trained decision tree's predictions if we shift or rescale any feature (e.g. use annual income in US$ instead of S$)?

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

DECISION TREES WHEN RESCALING VARIABLES

Q: Is there any change to a trained decision tree's predictions if we shift or rescale any feature (e.g. use annual income in US$ instead of S$)?

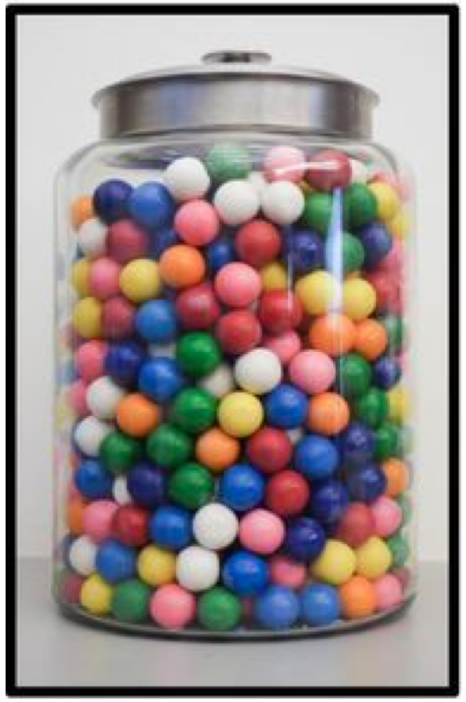
A: No, the algorithm does the exact same: e.g. for each cut in US$, there is a corresponding cut in S$ that has the same effect. This applies to any order-preserving transformations: e.g. normalization, standardization

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

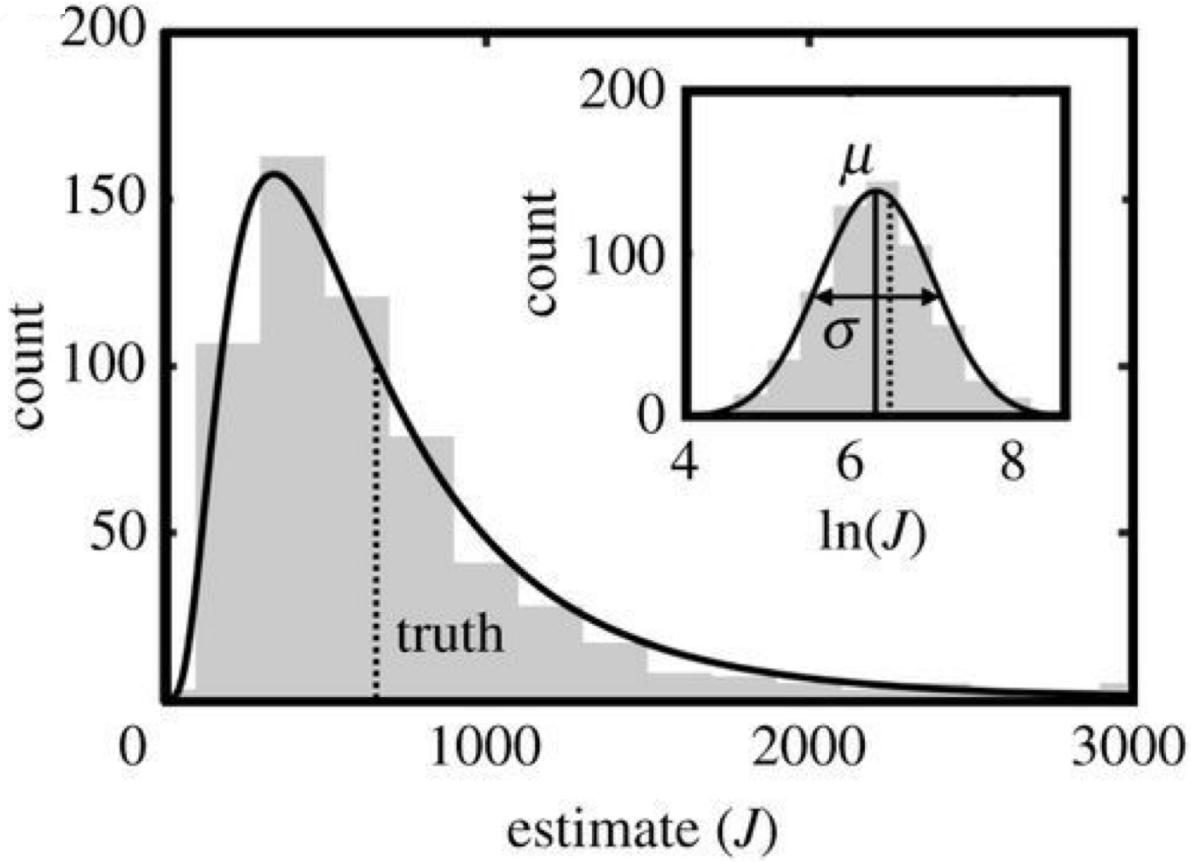
CLASSIFICATION OVERVIEW

1. Problem Setup
2. Evaluating Classifiers
3. Nearest Neighbor Methods
4. **Trees and Ensembles**
   a) Decision Trees
   b) Bagging and Random Forests
   c) Boosting and Gradient Boosting Machines
5. Logistic Regression
6. Deep Learning

BAGGING: MOTIVATION ('WISDOM OF THE CROWDS')



$J = 659$ objects
$\ln(J) = 6.5$



Kao et al., "Counteracting estimation bias and social influence to improve the wisdom of crowds"

BOOTSTRAP SAMPLING

Goal: generate randomly sampled "versions" of the dataset

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

Original Data

Resample same-sized dataset, **with replacement**

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
3	No	Single	170K	No
1	Yes	Single	125K	No
7	No	Single	50K	Yes
1	Yes	Single	125K	No
4	Yes	Married	120K	No
7	No	Single	50K	Yes
2	No	Married	100K	No

A Bootstrap Sample

BOOTSTRAP AGGREGATION ('BAGGING')

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	170K	No
4	Yes	Married	120K	No
5	No	Single	75K	Yes
6	No	Married	160K	No
7	No	Single	50K	Yes

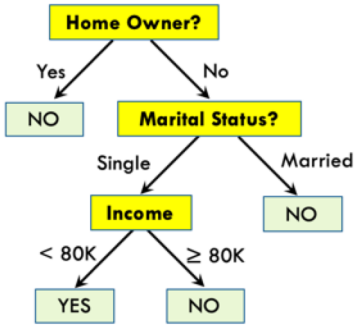
Original Data

Resampling

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
3	No	Single	170K	No
1	Yes	Single	125K	No
7	No	Single	50K	Yes
1	Yes	Single	125K	No
4	Yes	Married	120K	No
7	No	Single	50K	Yes
2	No	Married	100K	No

Bootstrap Sample 1

Training



Model 1

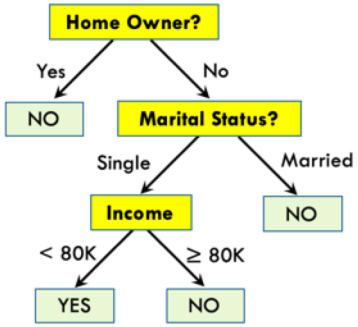
• • •

• • •

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
4	Yes	Married	120K	No
7	No	Single	50K	Yes
5	No	Single	75K	Yes
6	No	Married	160K	No
1	Yes	Single	125K	No
7	No	Single	50K	Yes
1	Yes	Single	125K	No

Bootstrap Sample N

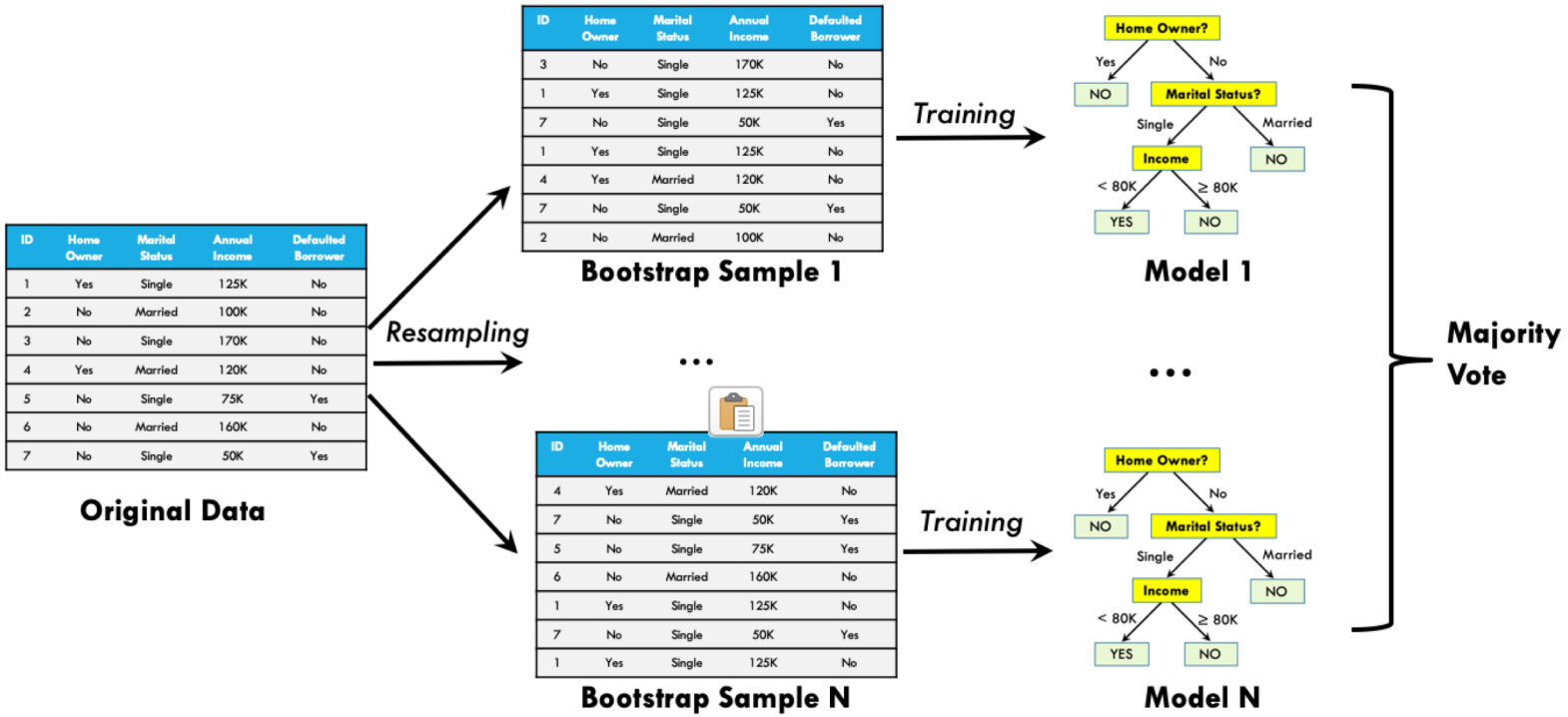
Training



Model N

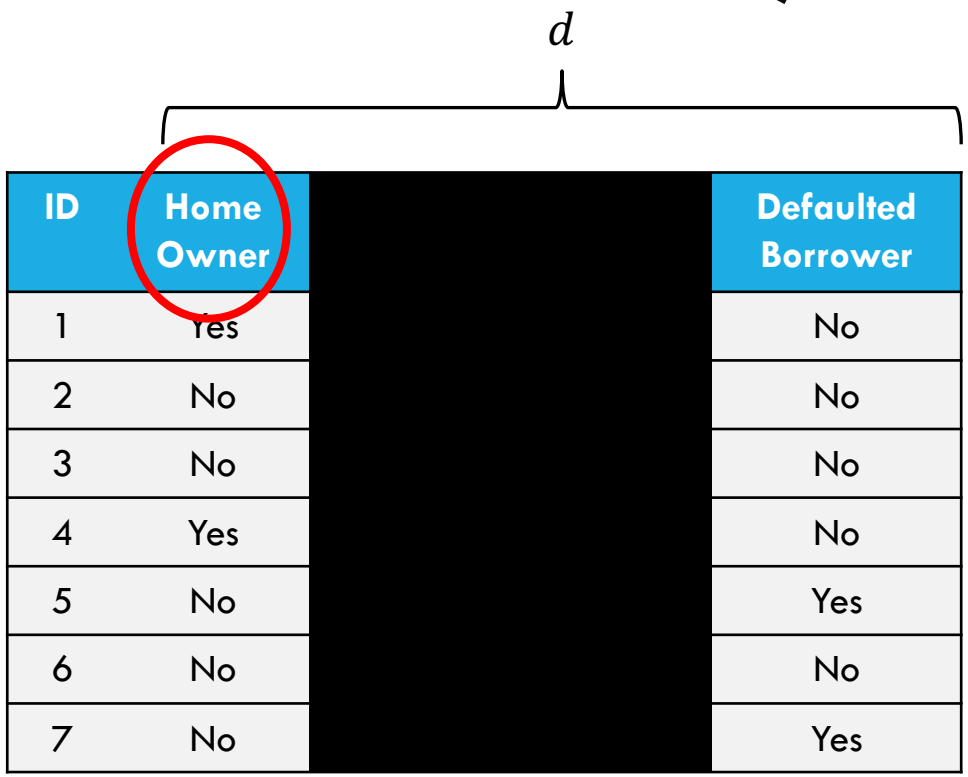
Majority Vote

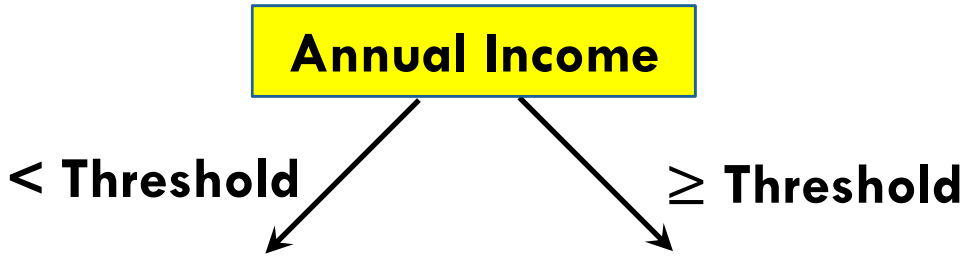
RANDOM FORESTS: BOOTSTRAP RESAMPLING ("ROW SAMPLING")



Bootstrap resampling: Random forests train an ensemble of decision trees on N bootstrap samples. (Sometimes, we take bootstrap samples of size smaller than the original dataset size, e.g., based on a user-specified parameter)

RANDOM FORESTS: FEATURE SAMPLING ("COLUMN SAMPLING")

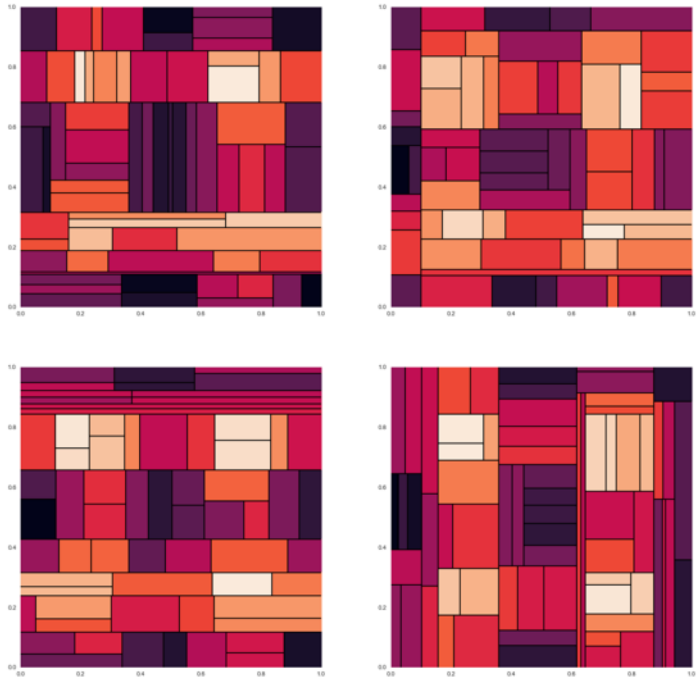




Feature sampling: During each split, instead of considering **all** d features, we only consider a **random subset** of features, usually of size $\sqrt{d}$

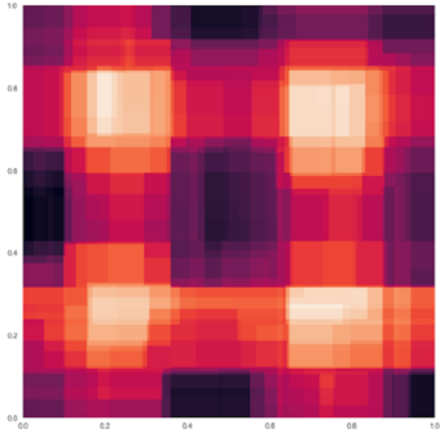
WHY DO RANDOM FORESTS WORK WELL?

Individual decision tree predictions



Averaging

Random Forest predictions



Variance Reduction: individual trees can overfit and give highly variable output. But when averaging them, the predictions are smoother and perform better on test data.

Randomization is important in making the decision trees **decorrelated**

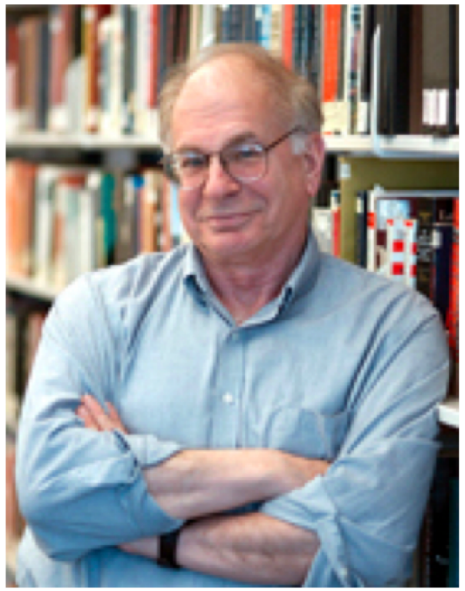
Credit: Tyler White, http://structuringtheunstructured.blogspot.com/2017/11/coloring-with-random-forests.html

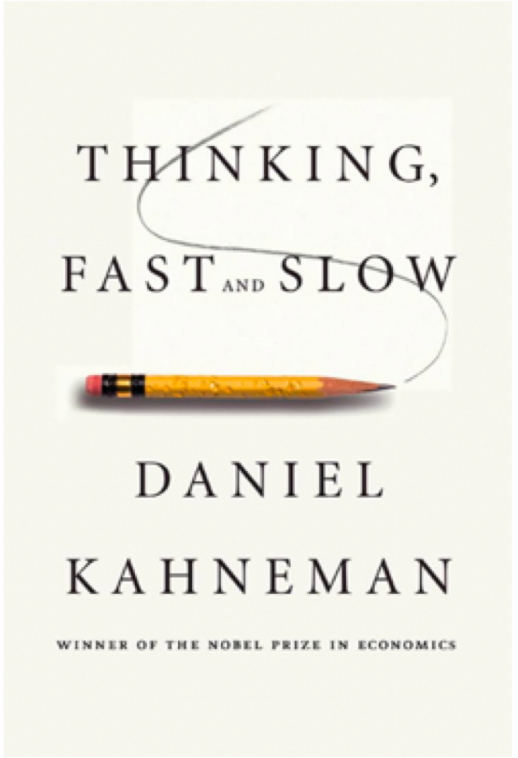
IMPORTANCE OF DECORRELATION

"To derive the most useful information from multiple sources of evidence, you should always try to make these sources independent of each other."

"A simple rule can help: before an issue is discussed, all members of the committee should be asked to write a very brief summary of their position. This procedure makes good use of the value of the **diversity of knowledge and opinion in the group**. The standard practice of open discussion gives too much weight to the opinions of those who speak early and assertively, causing others to line up behind them."

Daniel Kahnemann, *Thinking Fast and Slow*





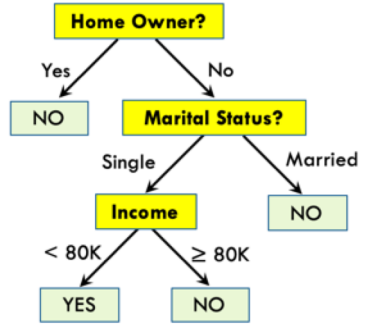
PROS AND CONS (VS. DECISION TREES)

Pros

- *Variance Reduction:* Ensembling many decision trees leads to more stable and accurate predictions
- Accuracy is fairly close to state of the art
- Parallelizable
- Not much tuning required

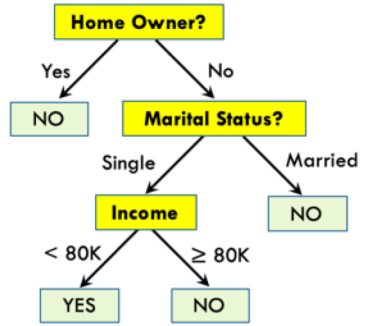
Cons

- Less interpretable than decision trees
- Slower than decision trees



Model 1

...



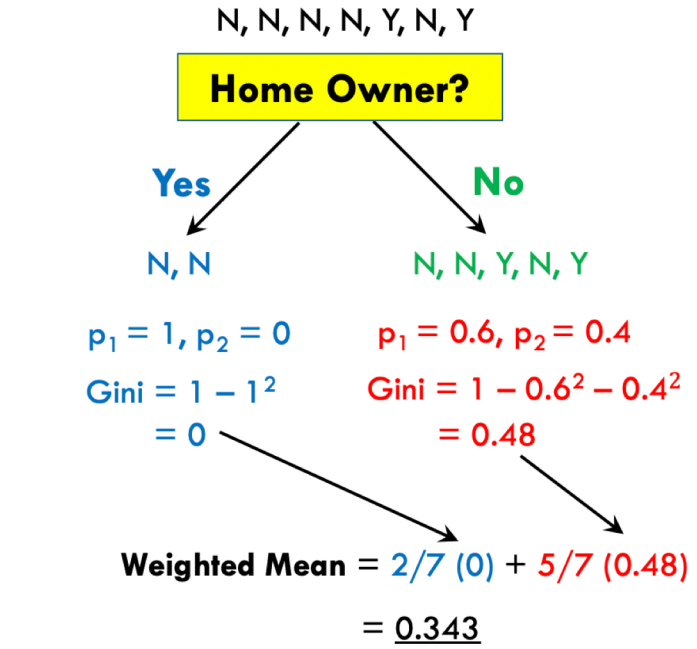
Model N

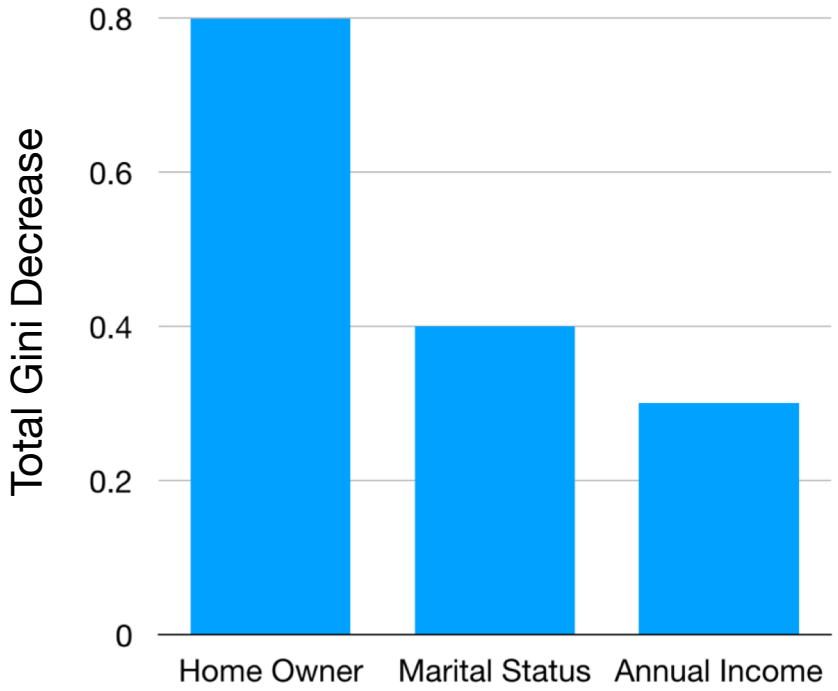
VARIABLE IMPORTANCE PLOTS

The importance of each variable is measured by the **total reduction in Gini index** brought about by that feature.

More important variables result in greater decrease in Gini index on average.

Computing variable importance is useful for **feature selection**.

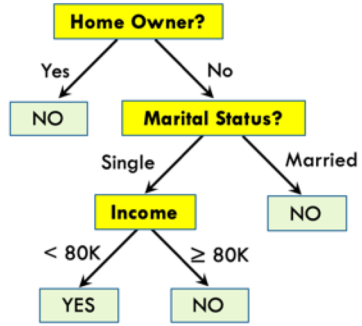




QUIZ: RANDOM FOREST HYPERPARAMETERS

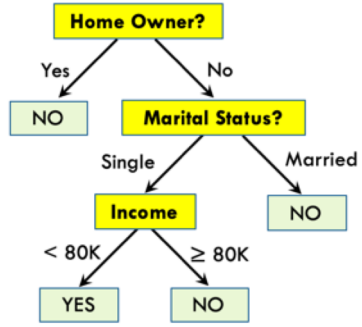
Q: Which of the following tends to reduce overfitting?

1. Increasing number of trees
2. Increasing depth of trees
3. Decreasing the number of features considered when selecting each split



Model 1

…

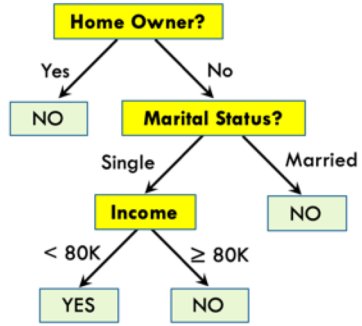


Model N

QUIZ: RANDOM FOREST HYPERPARAMETERS

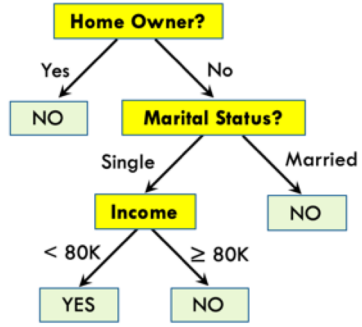
Q: Which of the following tends to reduce overfitting?

1. Increasing number of trees 
2. Increasing depth of trees 
3. Decreasing the number of features considered when selecting each split 



Model 1

...



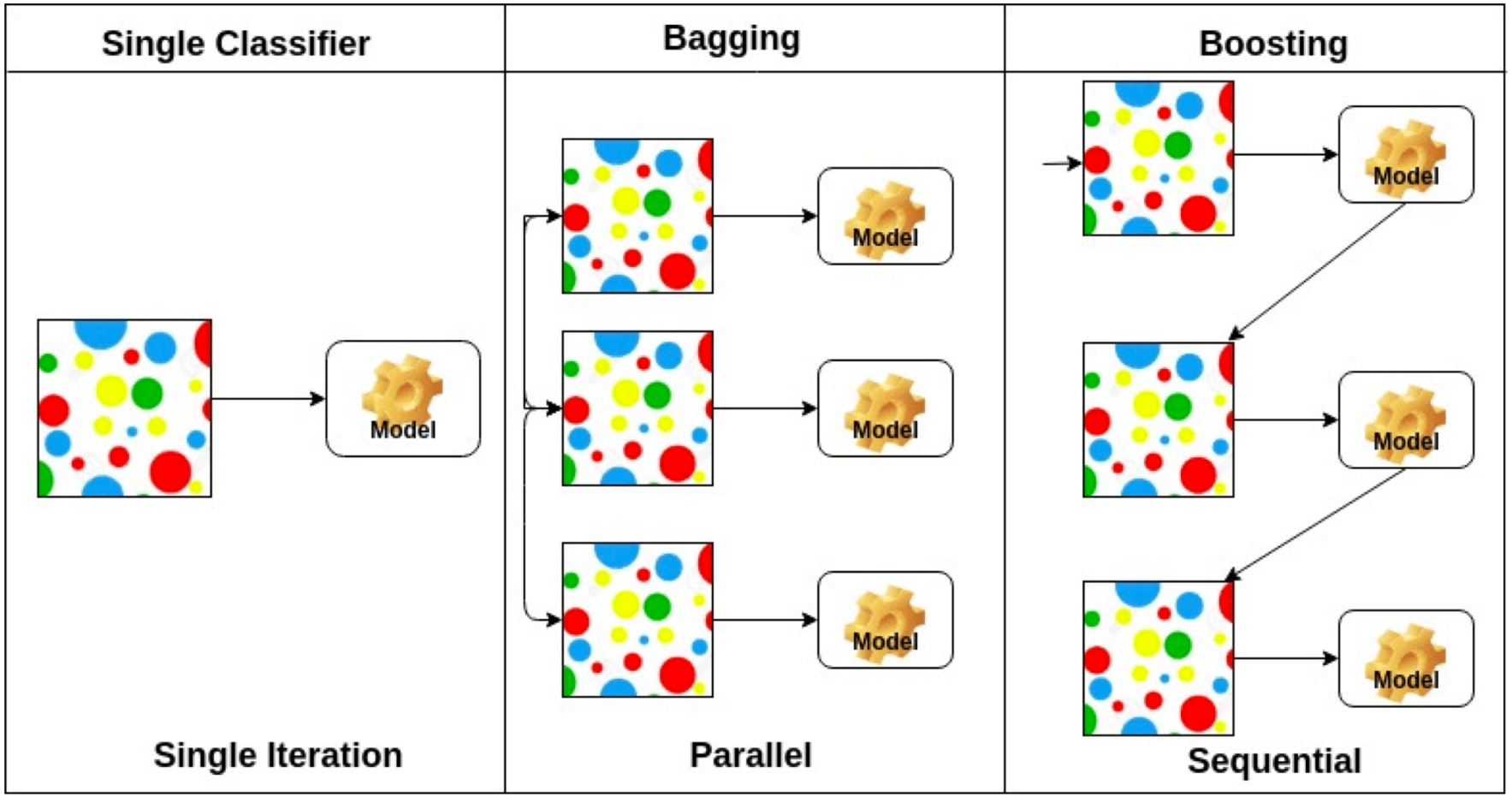
Model N

CLASSIFICATION OVERVIEW

1. Problem Setup
2. Evaluating Classifiers
3. Nearest Neighbor Methods
4. **Trees and Ensembles**
   a) Decision Trees
   b) Bagging and Random Forests
   c) Boosting and Gradient Boosting Machines
5. Logistic Regression
6. Deep Learning

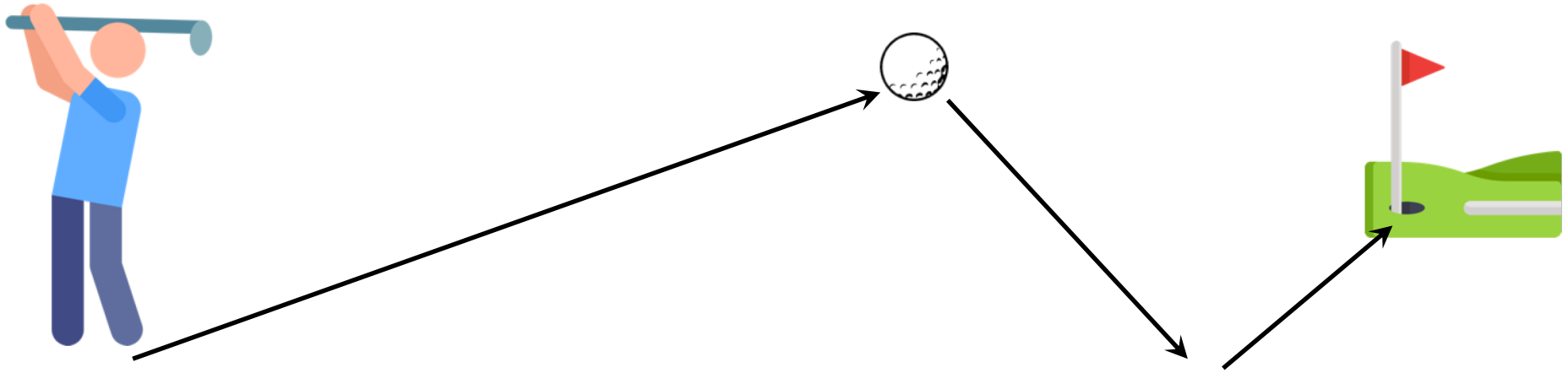
BAGGING VS BOOSTING

Bagging operates on bootstrap samples **independently**, but boosting is **sequential**: each new classifier "corrects for the mistakes" of the previous classifiers



https://www.datacamp.com/community/tutorials/adaboost-classifier-python

BOOSTING: ITERATIVE PROCESS



Each new classifier is trained to “correct for the mistakes” of the previous set of classifiers, gradually getting closer to the ideal classifier

GRADIENT BOOSTED DECISION TREES

Current Fitted Model

ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

1. Initial residuals are the original response data

GRADIENT BOOSTED DECISION TREES

Current Fitted Model

ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data (X)

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

Fit tree with response r_0

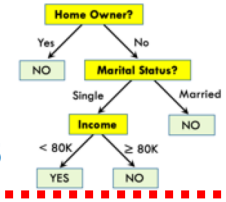
f_1
0.2
0.3
0.3
1
1
0.4
0.3

2. Fit a decision tree to predict residuals r_0

GRADIENT BOOSTED DECISION TREES

Learning rate → λ = 0.5

Current Fitted Model



ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data (X)

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

Fit tree with response r_0 →

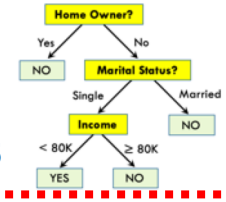
f_1	λf_1
0.2	0.1
0.3	0.15
0.3	0.15
1	0.5
1	0.5
0.4	0.2
0.3	0.15

3. Compute 'damped predictions', i.e. λ times decision tree output

GRADIENT BOOSTED DECISION TREES

Current Fitted Model

Learning rate → $\lambda = 0.5$



ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data (X)

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

Fit tree with response r_0 →

f_1	λf_1
0.2	0.1
0.3	0.15
0.3	0.15
1	0.5
1	0.5
0.4	0.2
0.3	0.15

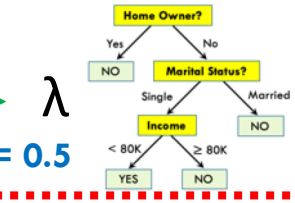
subtract

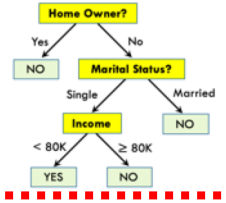
r_1
-0.1
0.35
0.15
0.5
0.4
0.2
0.15

Residual

4. Compute new residual: $r_1 = r_0 - \lambda f_1$

GRADIENT BOOSTED DECISION TREES

Learning rate → $\lambda = 0.5$ × 

+ λ × 

Current Fitted Model

ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data (X)

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

Fit tree with response r_0 →

f_1	λf_1	r_1
0.2	0.1	-0.1
0.3	0.15	0.35
0.3	0.15	0.15
1	0.5	0.5
1	0.5	0.4
0.4	0.2	0.2
0.3	0.15	0.15

Residual

Fit tree with response r_1 →

f_2	λf_2	r_2
-0.2	-0.1	0
0.6	0.3	0.05
0.2	0.1	0.05
0.6	0.3	0.2
0.4	0.2	0.2
0.2	0.1	0.1
0.2	0.1	0.05

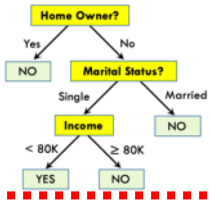
Residual

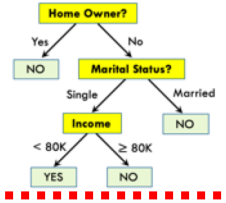
• • •

5. Repeat the same process (fitting decision tree, etc.) on the new residuals r_1

GRADIENT BOOSTED DECISION TREES

Current Fitted Model

λ = 0.5 

+ λ 

+ • • •

ID	Home Owner	Marital Status	Annual Income	y
1	Yes	Single	125K	0
2	No	Married	100K	0.5
3	No	Single	170K	0.3
4	Yes	Married	120K	1
5	No	Single	75K	0.9
6	No	Married	160K	0.4
7	No	Single	50K	0.3

Original Data (X)

r_0
0
0.5
0.3
1
0.9
0.4
0.3

Residual (Initial)

Fit tree with response r_0 →

f_1	λf_1	r_1
0.2	0.1	-0.1
0.3	0.15	0.35
0.3	0.15	0.15
1	0.5	0.5
1	0.5	0.4
0.4	0.2	0.2
0.3	0.15	0.15

Residual

Fit tree with response r_1 →

f_2	λf_2	r_2
-0.2	-0.1	0
0.6	0.3	0.05
0.2	0.1	0.05
0.6	0.3	0.2
0.4	0.2	0.2
0.2	0.1	0.1
0.2	0.1	0.05

Residual

• • •

6. After fitting B trees, the final prediction is the sum of damped trees

SUMMARY: GRADIENT TREES BOOSTING (REGRESSION CASE)

Model Predictions

Initialization: $\hat{f}(x) = 0$, and $r_i = y_i$ for all i

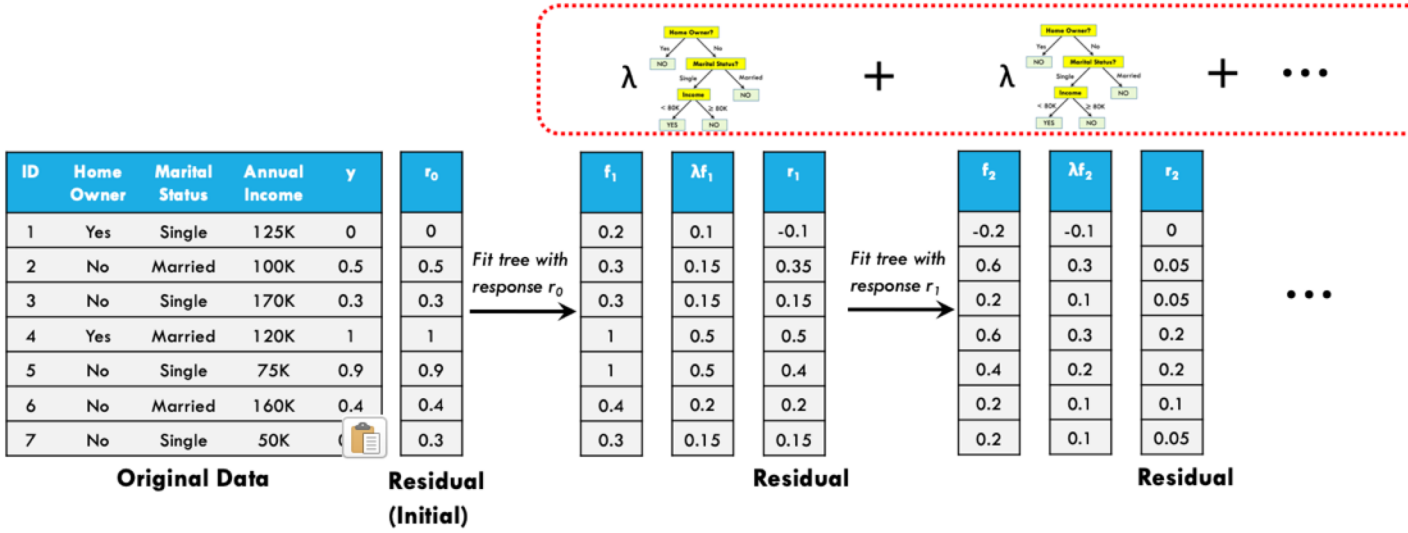
For $b = 1, \ldots, B$:

- **Fit Decision Tree** $\hat{f}^b$ to residuals $r_1, \ldots, r_n$.
- **Update Residual**

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$$

Output:

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$$

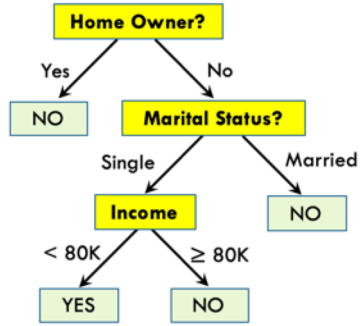


ID	Home Owner	Marital Status	Annual Income	y	r_0	f_1	λf_1	r_1	f_2	λf_2	r_2
1	Yes	Single	125K	0	0	0.2	0.1	-0.1	-0.2	-0.1	0
2	No	Married	100K	0.5	0.5	0.3	0.15	0.35	0.6	0.3	0.05
3	No	Single	170K	0.3	0.3	0.3	0.15	0.15	0.2	0.1	0.05
4	Yes	Married	120K	1	1	1	0.5	0.5	0.6	0.3	0.2
5	No	Single	75K	0.9	0.9	1	0.5	0.4	0.4	0.2	0.2
6	No	Married	160K	0.4	0.4	0.4	0.2	0.2	0.2	0.1	0.1
7	No	Single	50K	(	0.3	0.3	0.15	0.15	0.2	0.1	0.05

QUIZ: GRADIENT BOOSTING HYPERPARAMETERS

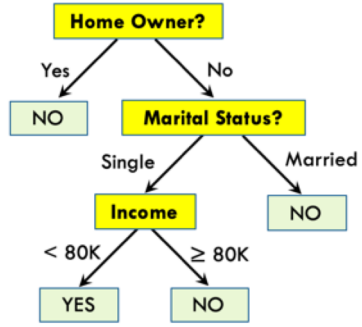
Q: While tuning a gradient boosting tree model, you decide to decrease the value of λ. How does this affect the number of trees you should use?

1. Increase
2. Decrease



Model 1

...

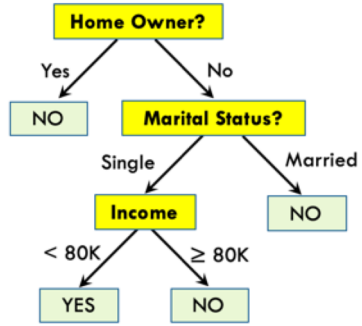


Model N

QUIZ: GRADIENT BOOSTING HYPERPARAMETERS

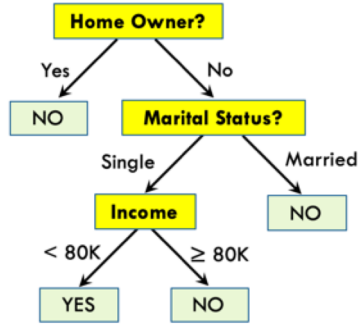
Q: While tuning a gradient boosting tree model, you decide to decrease the value of λ. How does this affect the number of trees you should use?

1. Increase
2. Decrease



Model 1

...



Model N

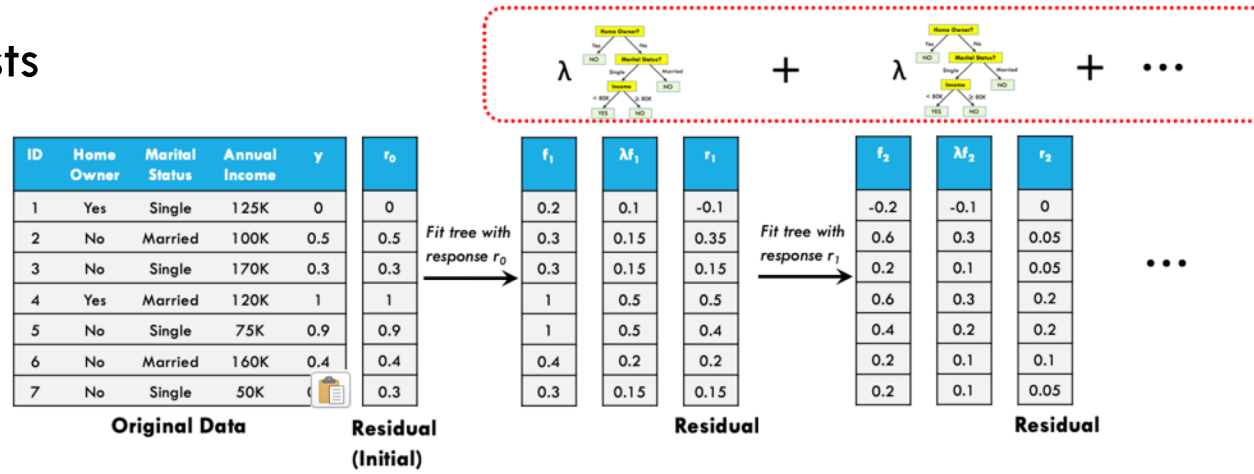
PROS AND CONS

Pros

- *Variance Reduction:* Ensembling many decision trees leads to more accurate predictions
- Generally more accurate than random forests

Cons

- Less interpretable than decision trees
- Slower than decision trees
- Significant tuning required[1] (for tree depth, learning rate, number of trees)



1: **General tuning guidelines**: tree depth is usually between 1 to 10, and is important for controlling overfitting. Tuning it by selecting from {2, 4, 6, 8, 10} based on cross validation is generally fine. Learning rate and no. of trees have to be tuned together: learning rate is usually around 0.01 to 0.1. Smaller learning rate requires more trees (and thus longer training time). If speed is not an issue, a common approach is to set a low learning rate (e.g. 0.01), and train it while selecting the number of trees using early stopping (most gradient boosting tree packages have early stopping already implemented). If this is too slow, it may be advisable to start with a higher learning rate (e.g. 0.1) and later assess whether lowering the learning rate provides improvement in accuracy. See https://machinelearningmastery.com/configure-gradient-boosting-algorithm/ (and other guides that exist online) for more details (including other tuning parameters)