



Decision Trees

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington

©2015-2016 Emily Fox & Carlos Guestrin

Predicting potential loan defaults

What makes a loan risky?



Credit
★★★

Income
★★

Term
★★★★

Personal Info
★★

Did I paid my previous bills on time?
↗

→ e.g.: Around \$100k

→ How soon I have to repay the loan?
e.g. 2 year / 3 year

Age, Martial Status,
Machine Learning Specialization

Given all this info, we have to predict if it's risky or not

Credit history explained

Did I pay previous
loans on time?



Credit History
★★★

Example: excellent,
good, or fair

Income
★★

Term
★★★★★

Personal Info
★★

Income

What's my income?

Example: \$80K per year

Credit History
★★★☆

Income
★★☆

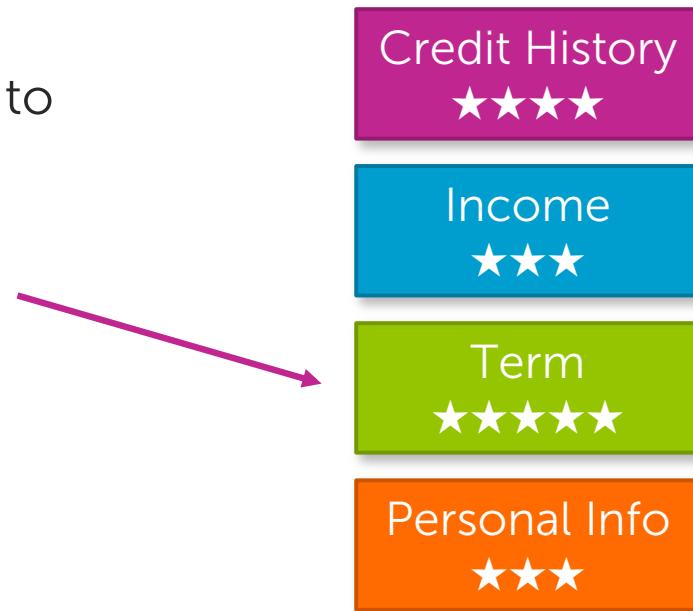
Term
★★★★★

Personal Info
★★★

Loan terms

How soon do I need to pay the loan?

Example: 3 years,
5 years,...



Personal information

Age, reason for the
loan, marital status,...

Example: Home loan
for a married couple

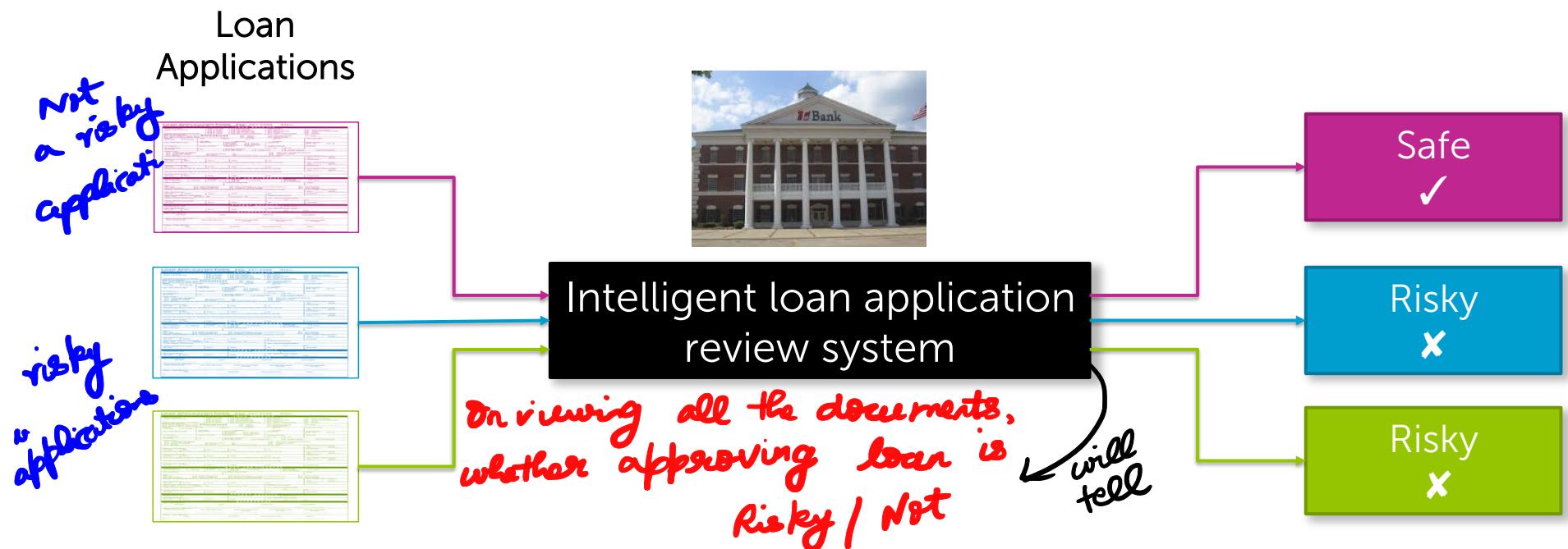
Credit History
★★★

Income
★★

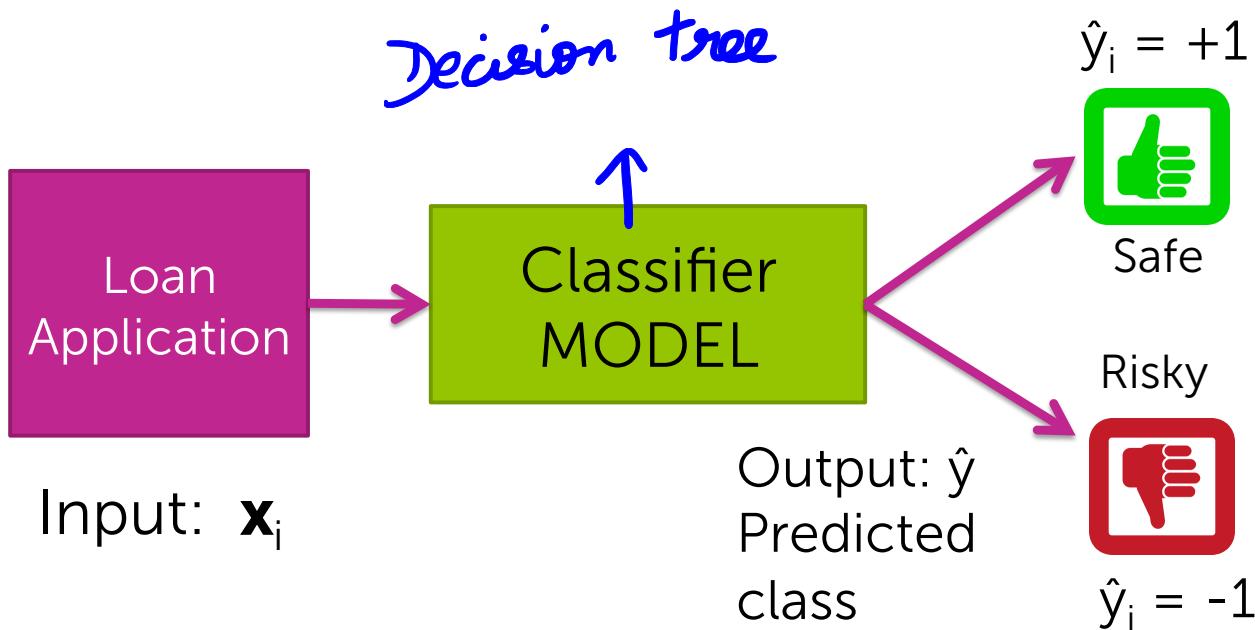
Term
★★★★★

Personal Info
★★

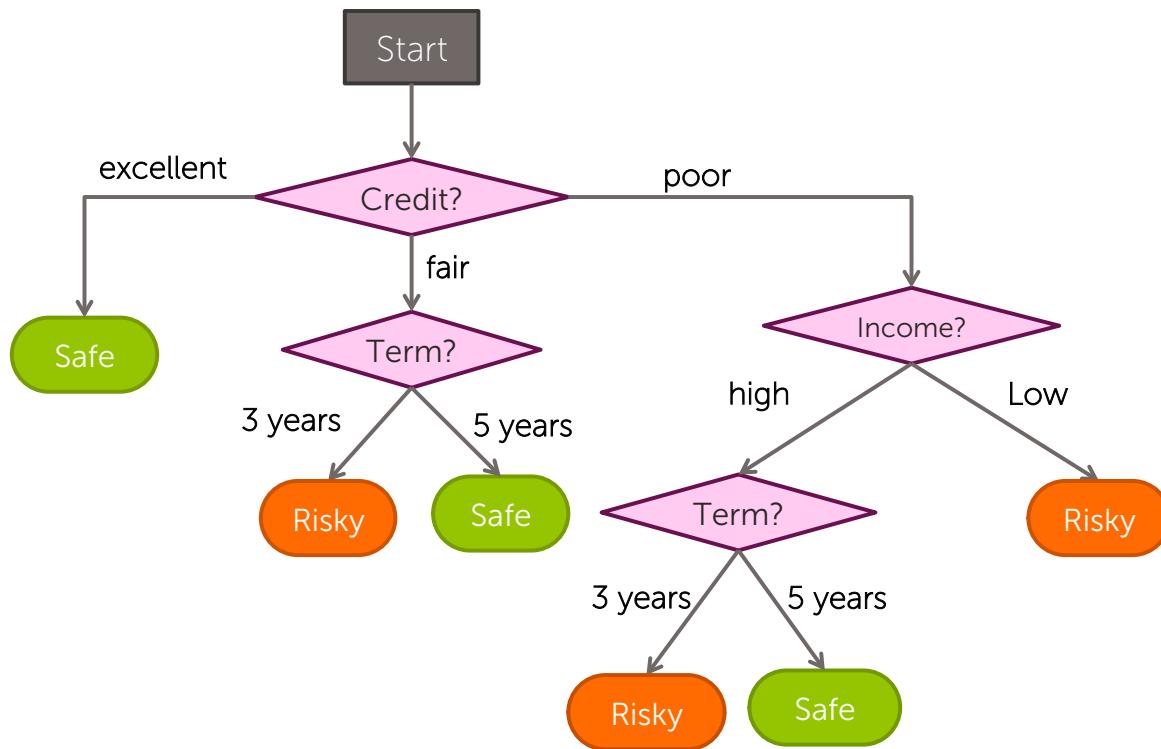
Intelligent application



Classifier review

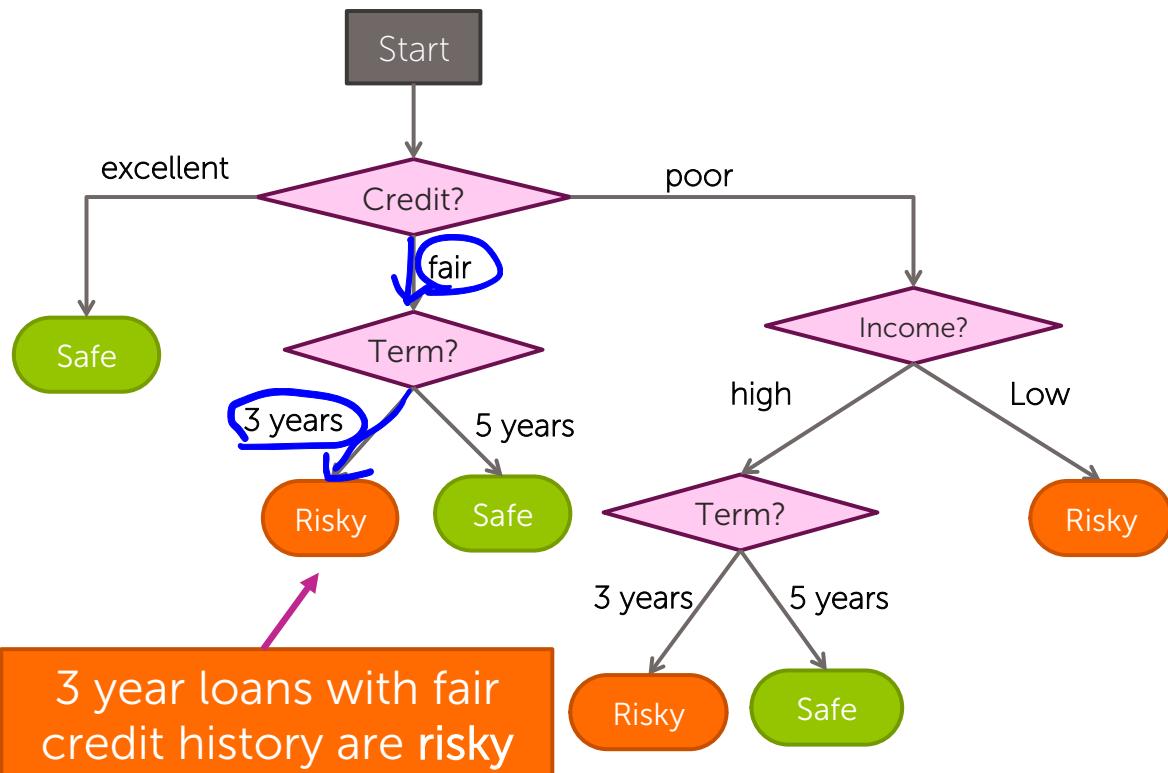


This module ... decision trees

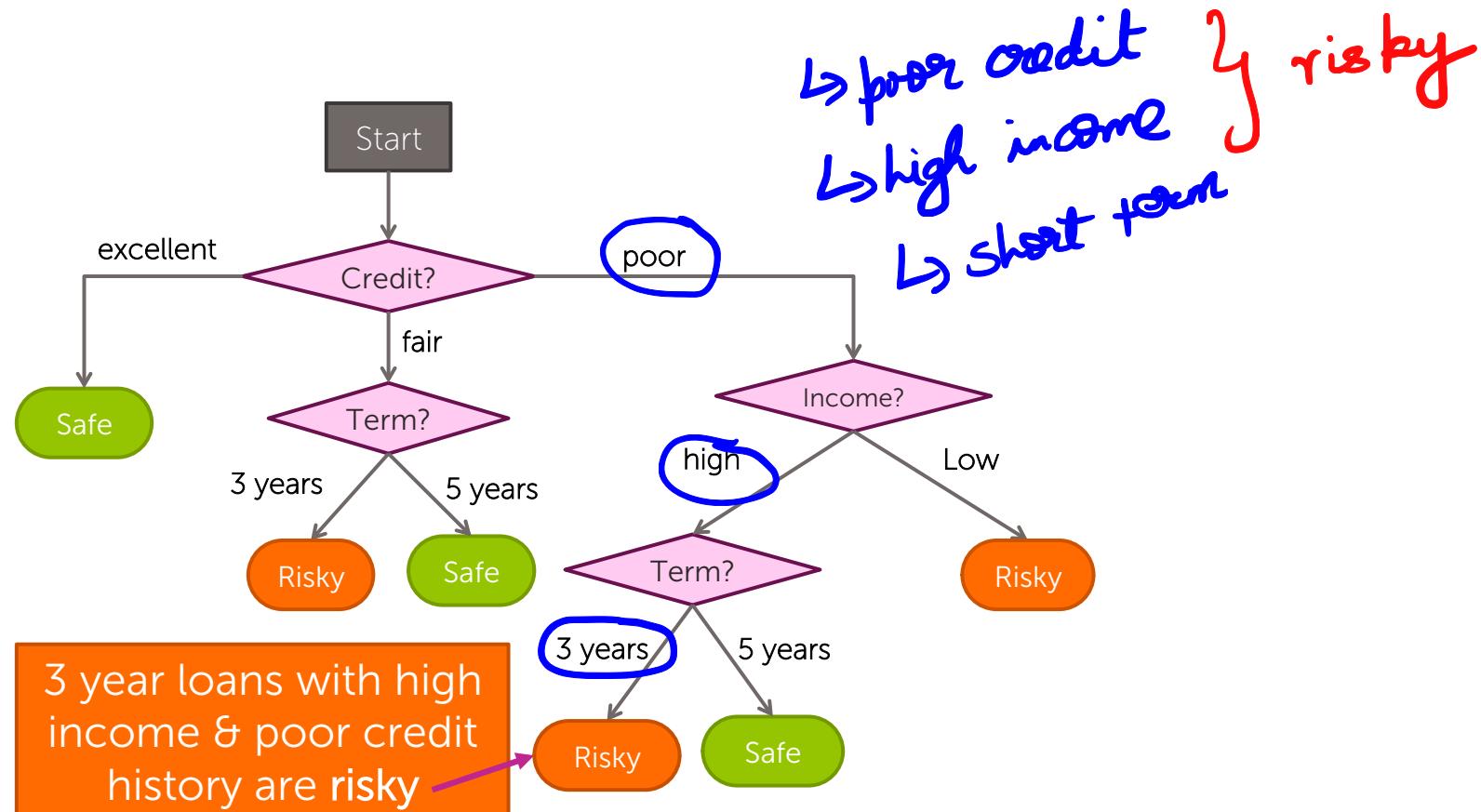


Decision trees: *Intuition*

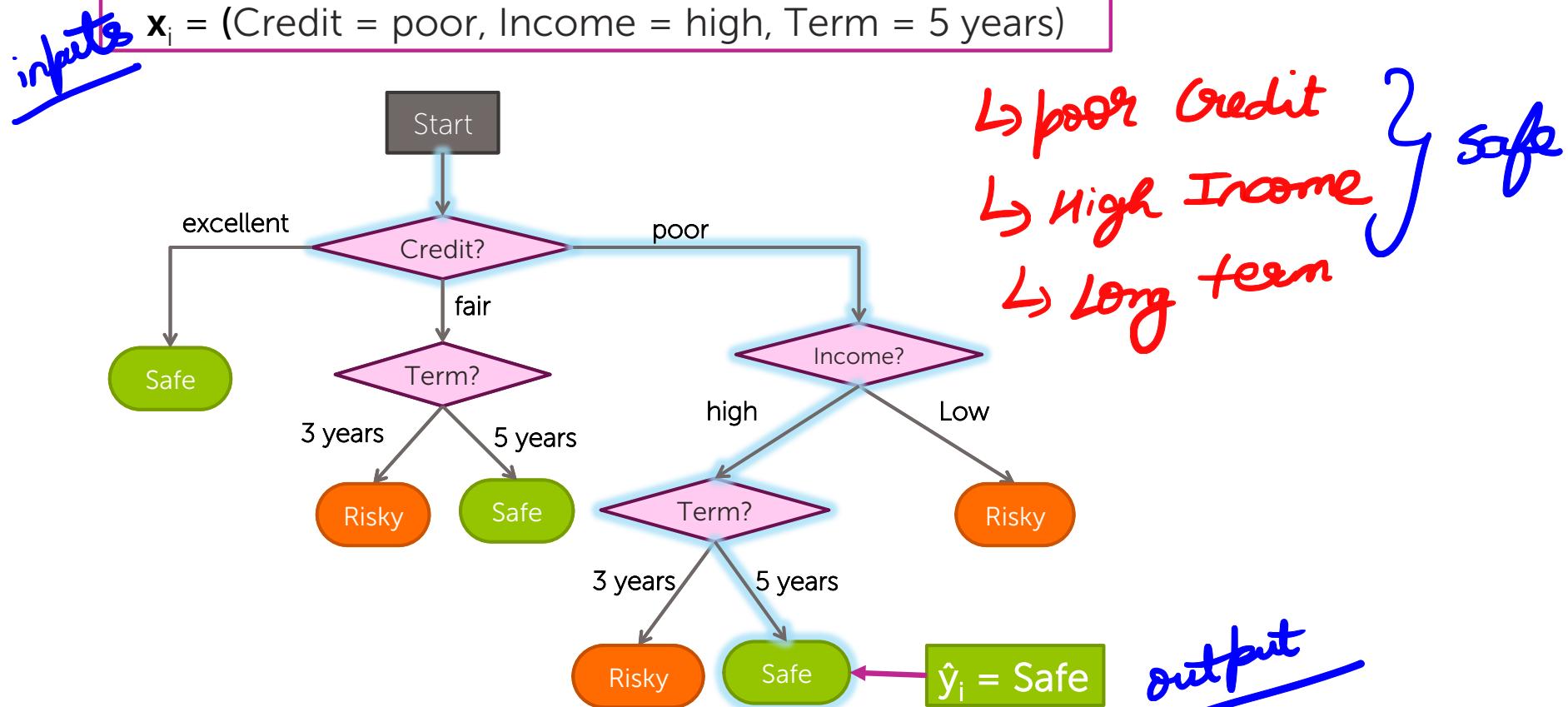
What does a decision tree represent?



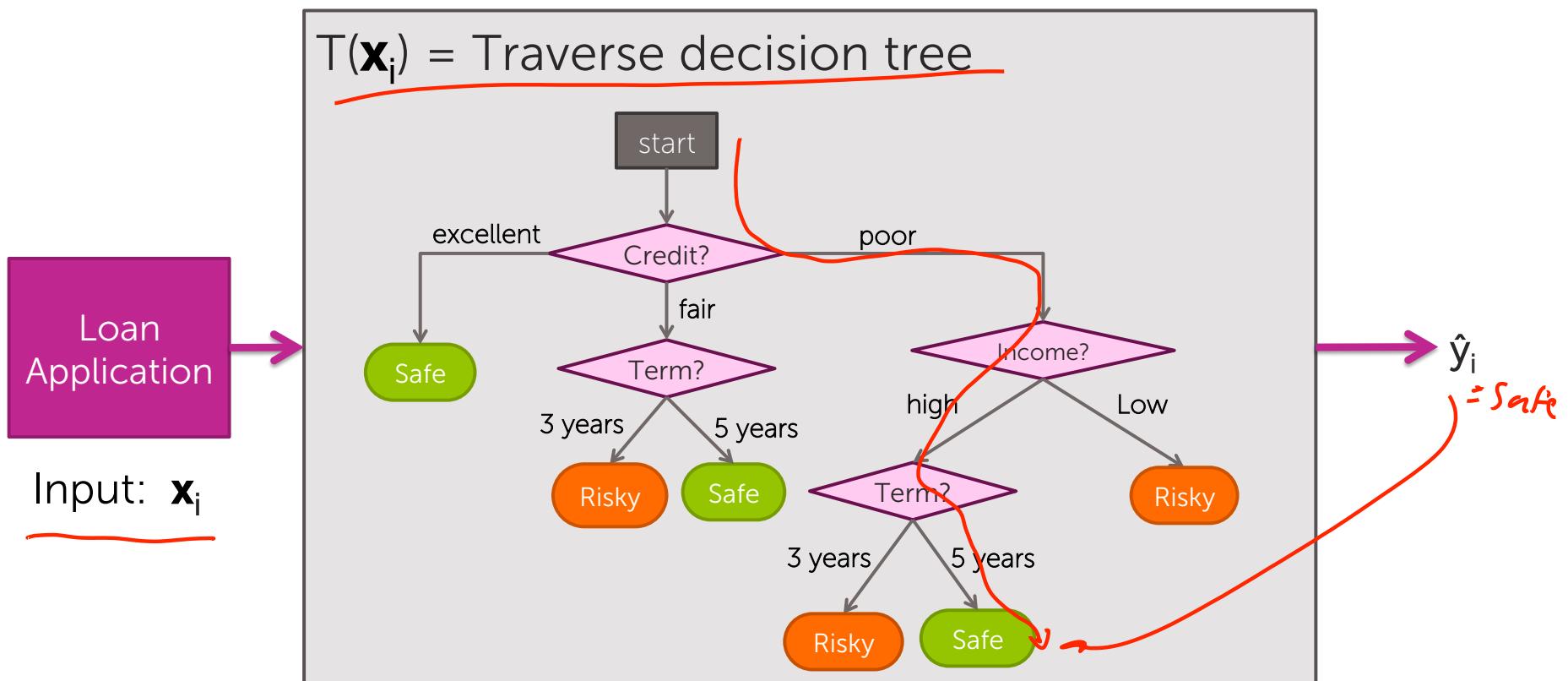
What does a decision tree represent?



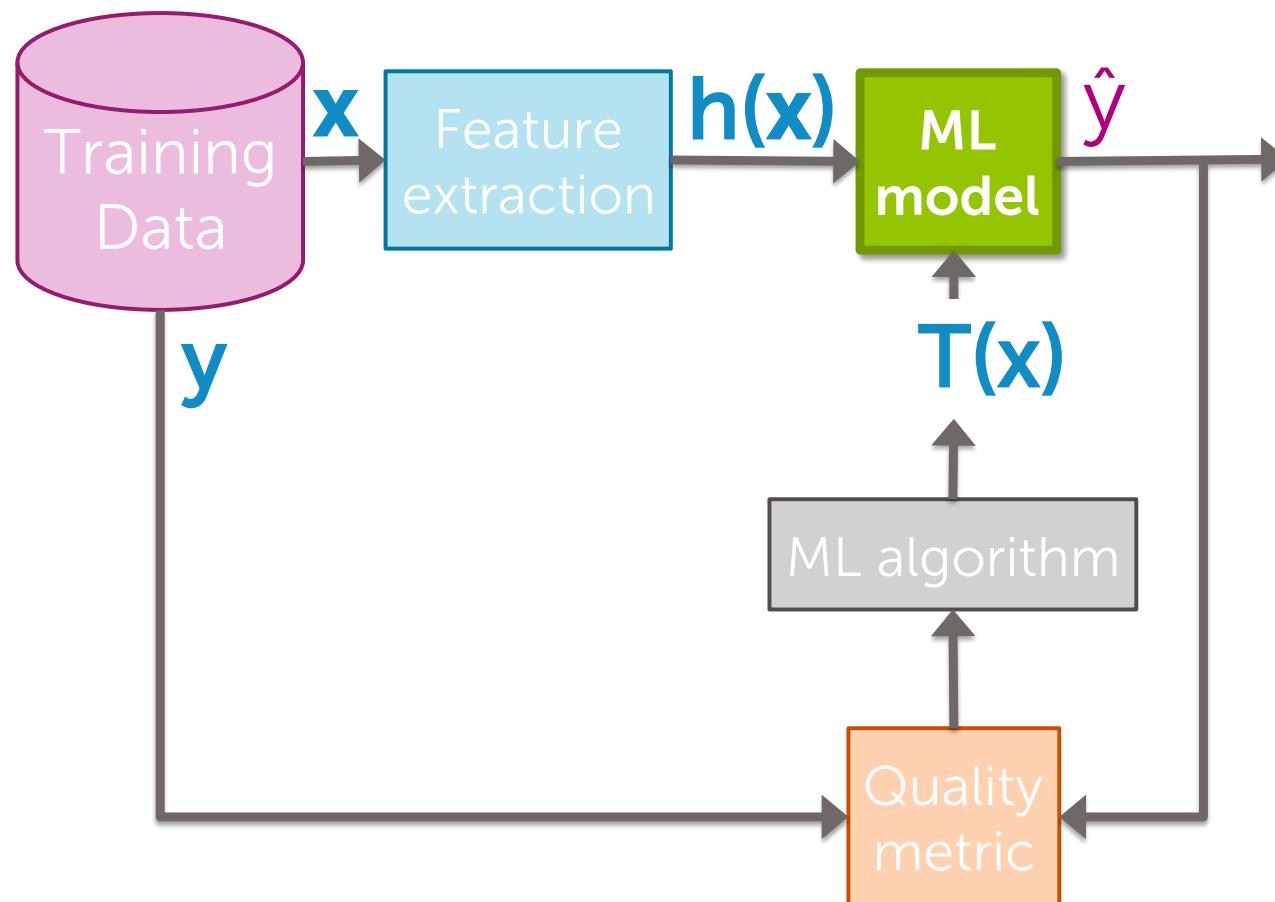
Scoring a loan application



Decision tree model

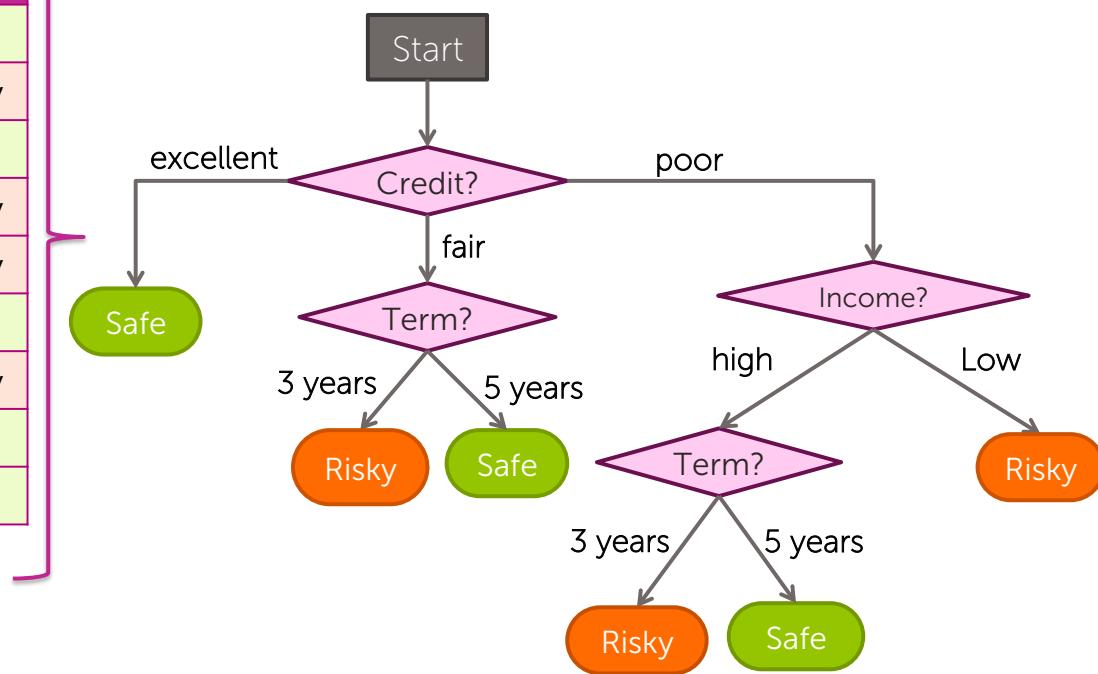


Decision tree learning task



Learn decision tree from data?

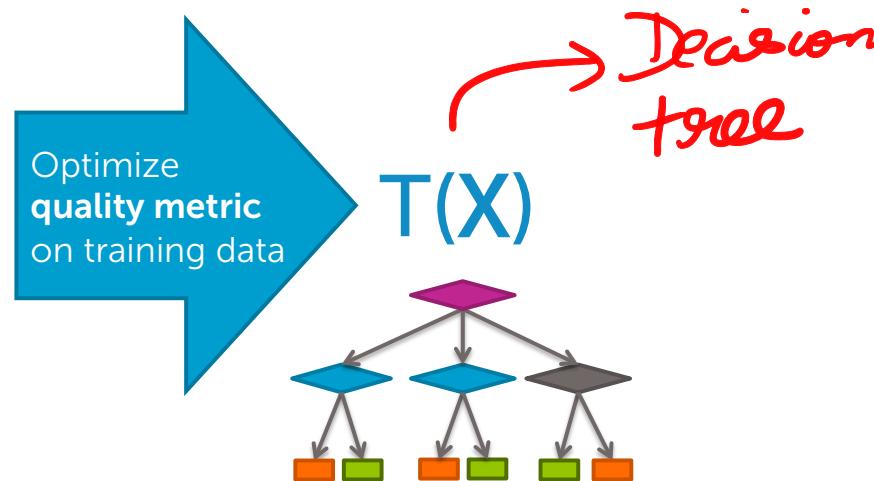
	$h_1(x)$	$h_2(x)$	$h_3(x)$	Loan Status
Credit	Term	Income	y	
excellent	3 yrs	high	safe	
fair	5 yrs	low	risky	
fair	3 yrs	high	safe	
poor	5 yrs	high	risky	
excellent	3 yrs	low	risky	
fair	5 yrs	low	safe	
poor	3 yrs	high	risky	
poor	5 yrs	low	safe	
fair	3 yrs	high	safe	



Decision tree learning problem

Training data: N observations (\mathbf{x}_i, y_i)

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



Quality metric: Classification error

- Error measures fraction of mistakes

Number of samples $\rightarrow 35$

$$\text{Error} = \frac{\# \text{ incorrect predictions}}{\# \text{ examples}}$$

- Best possible value : 0.0
- Worst possible value: 1.0

\rightarrow All the predictions are correct
There are no errors at all.

Try go to close to 0 and far from 1

↓

All the predictions are wrong

Number of wrongly predicted samples = Number of the examples in the data-set

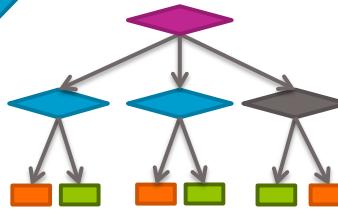
$$\frac{35}{35} \Rightarrow 1$$

Find the tree with lowest classification error

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Minimize
classification error
on training data

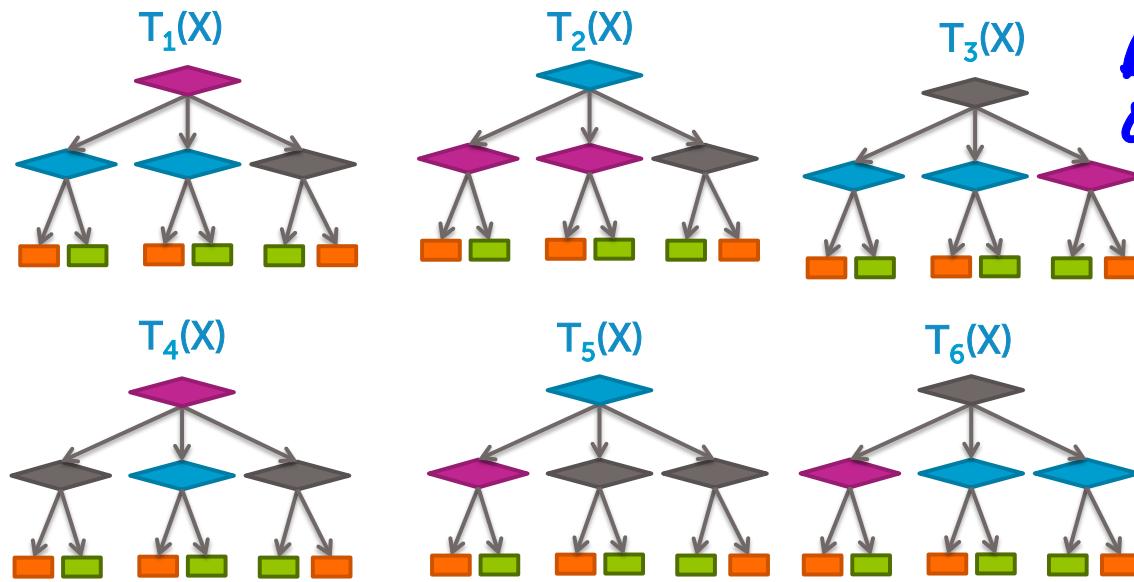
$T(X)$



How do we find the best tree?

Exponentially large number of possible trees makes decision tree learning hard!
(NP-hard problem)

NP-hard problem
↳ no approximation algorithm



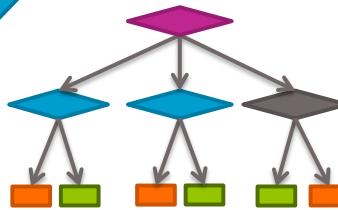
But there are some ideas of heuristics that tends to perform extremely well

idea → Greedy Algorithm

Simple (greedy) algorithm finds “good” tree

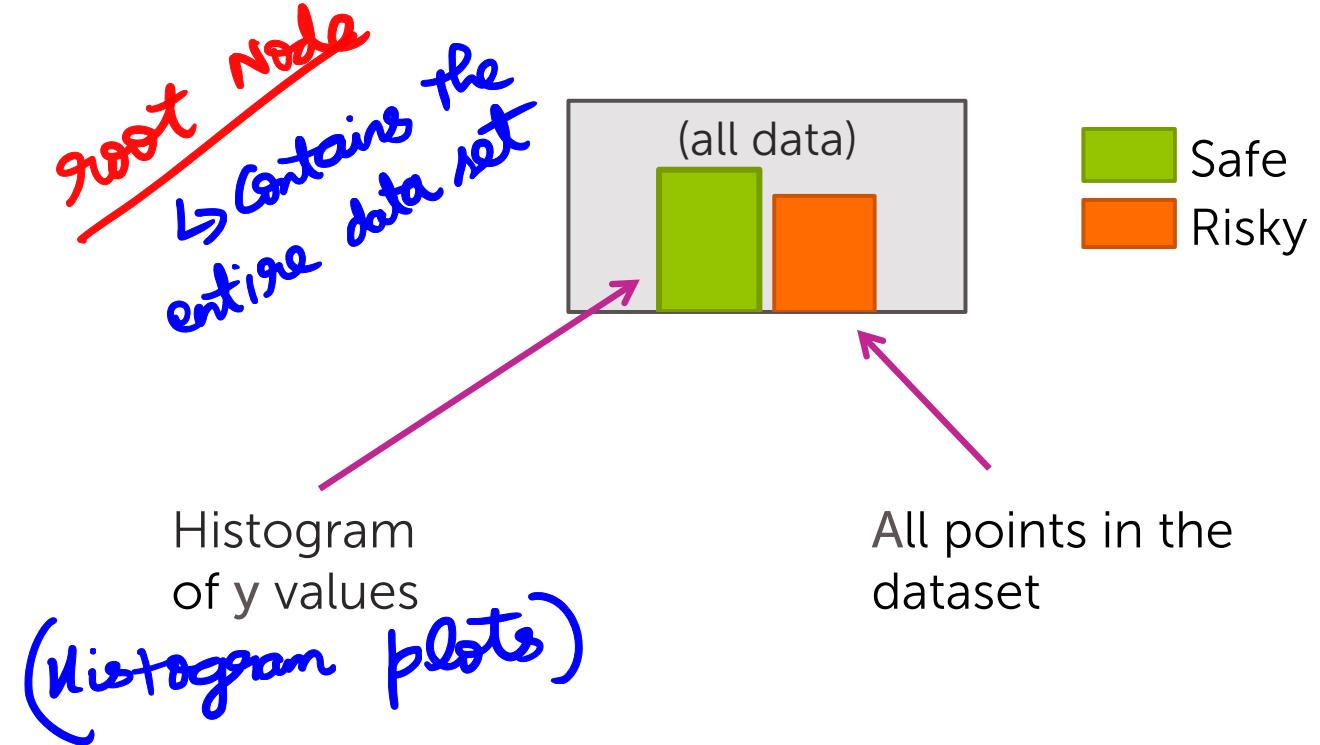
Going to incrementally build a tree one layer at a time, in order to get best possible classification error at each step.

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



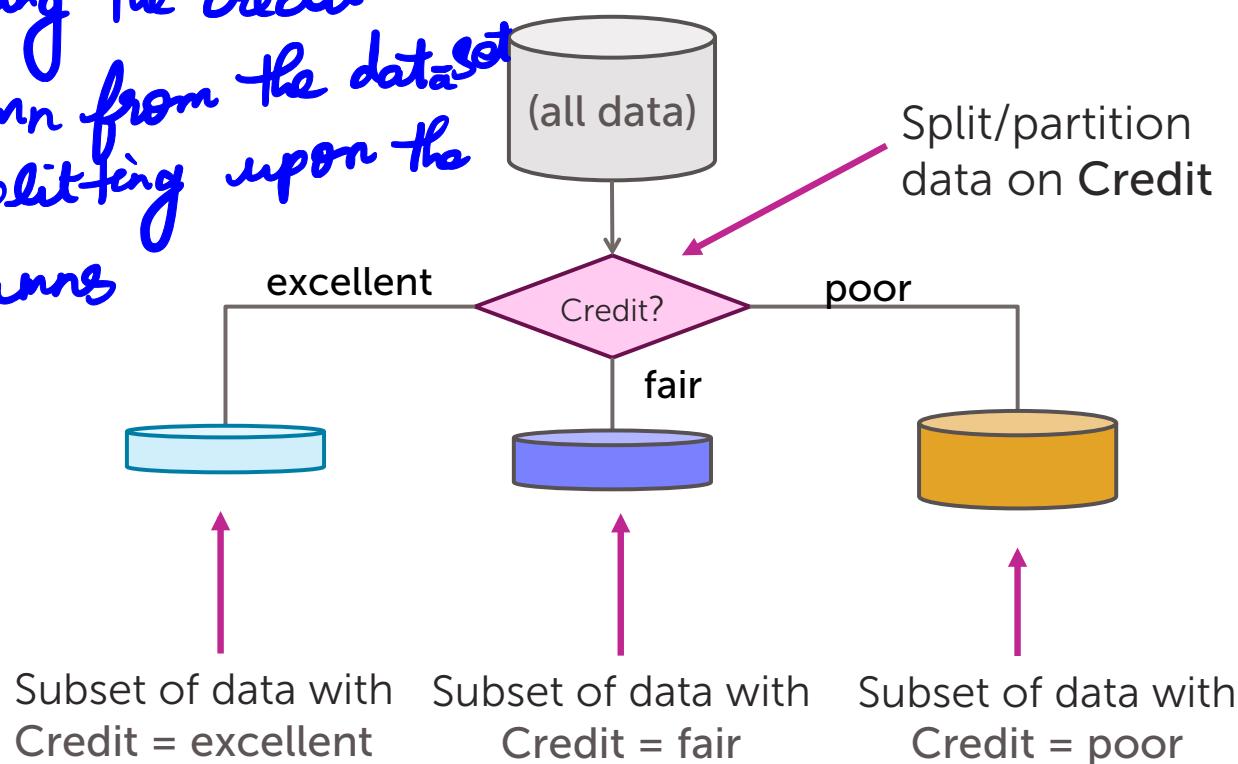
Greedy decision tree learning: *Algorithm outline*

Step 1: Start with an empty tree



Step 2: Split on a feature

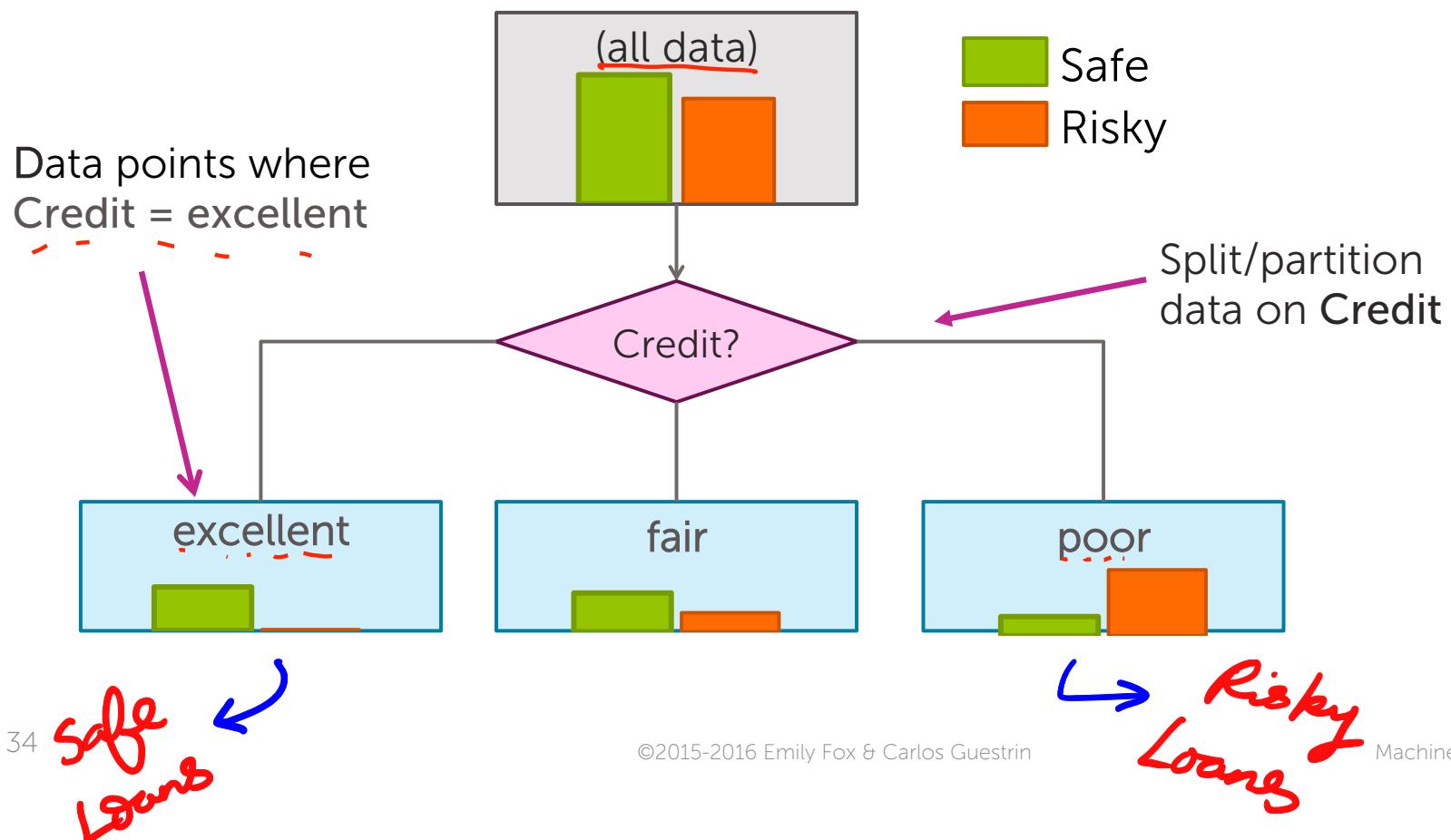
1) Taking the credit column from the dataset
2) Splitting upon the columns



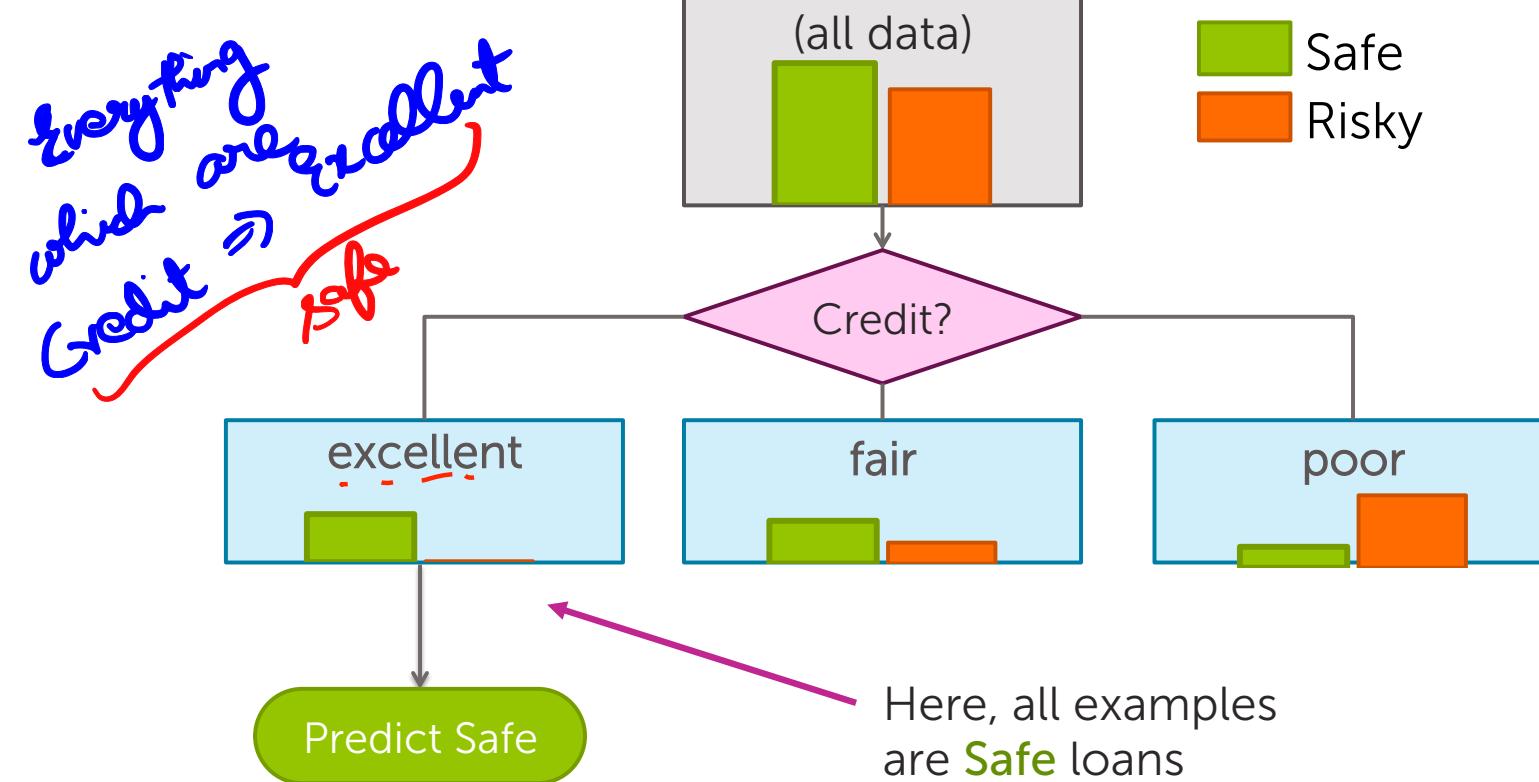
Credit	Income	Tens	Personal

Credit
↳ excellent
↳ fair
↳ poor

Feature split explained



Step 3: Making predictions



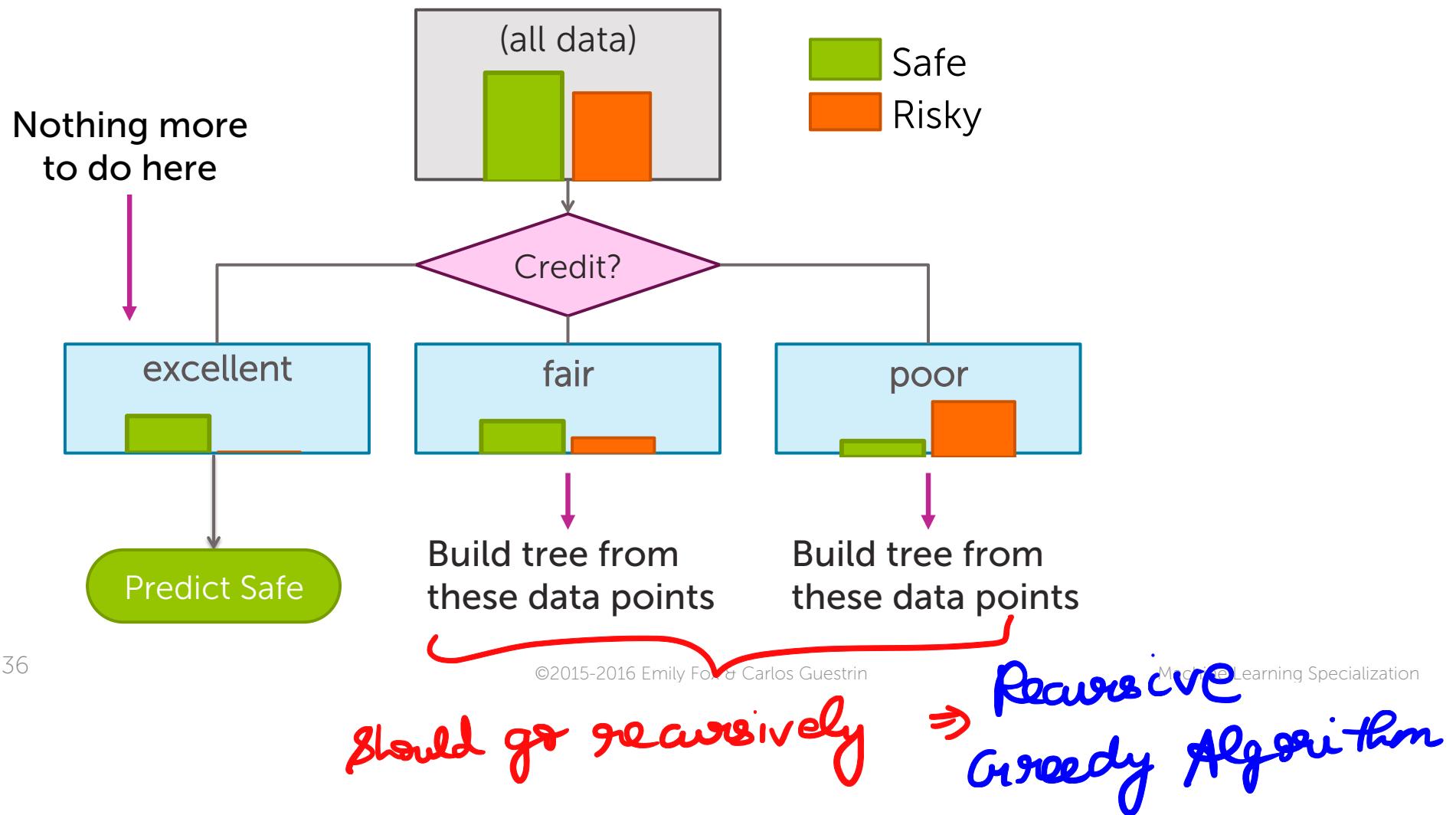
35

Predicted safe, don't
disturb go for next.

©2015-2016 Emily Fox & Carlos Guestrin

Machine Learning Specialization

Step 4: Recursion



Greedy decision tree learning

features {	Credit	Income	Team	Personal

- Step 1: Start with an empty tree

- Step 2: Select a feature to split data

- For each split of the tree:

- Step 3: If nothing more to, make predictions

- Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Problem 1: Feature split selection

Problem 2: Stopping condition

Recursion

Credit

↳ excellent

37
↳ fair

↳ poor

Taking excellent

©2015-2016 Emily Fox & Carlos Guestrin

Again do
recurs iC
until classified

Machine Learning Specialization



Feature split learning

=

Decision stump learning

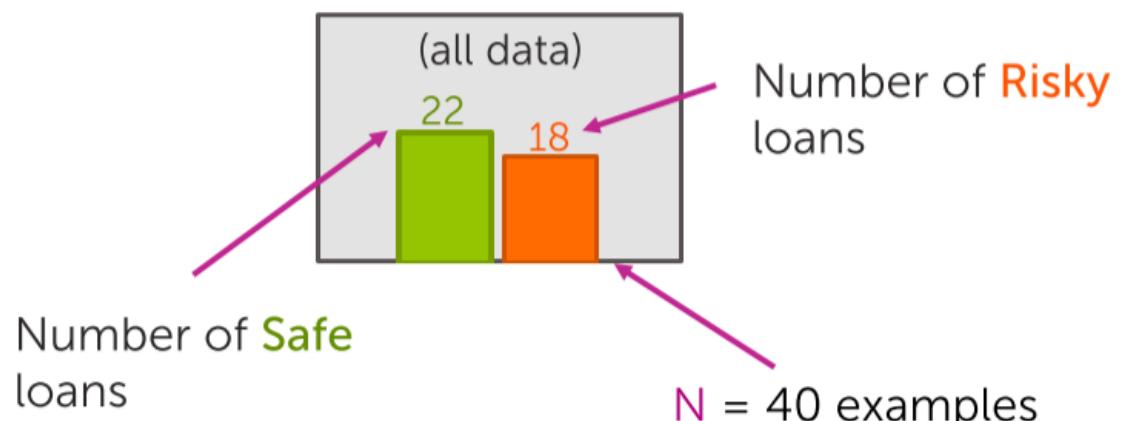
↳ one level decision tree

Start with the data

Assume $N = 40$, 3 features

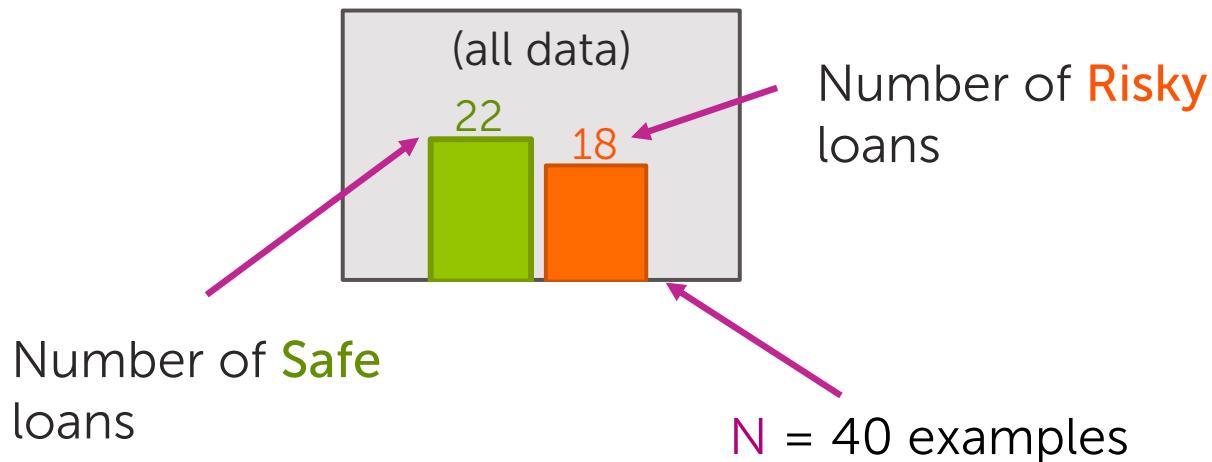
Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Loan status: **Safe** **Risky**



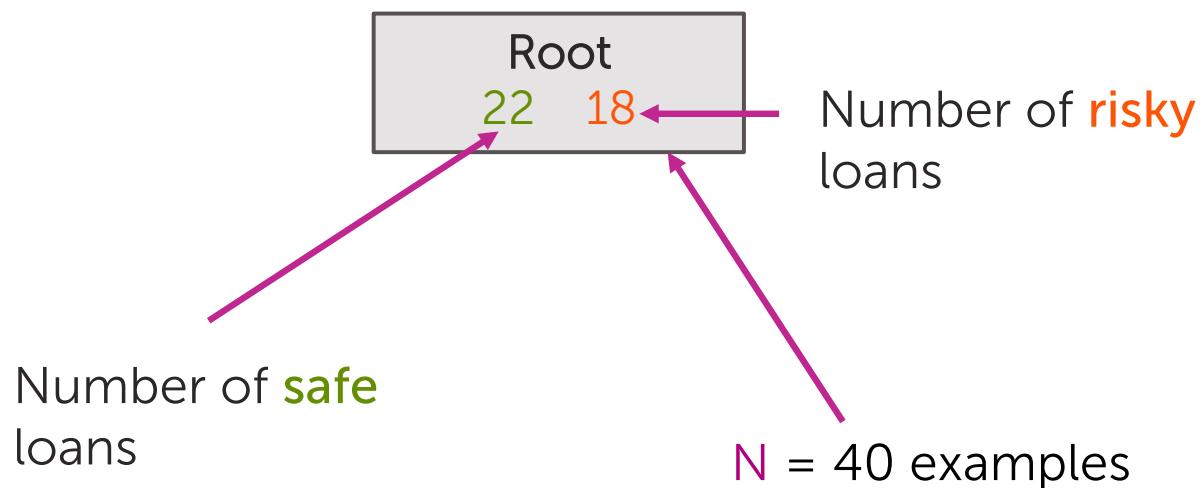
Start with all the data

Loan status: **Safe** **Risky**

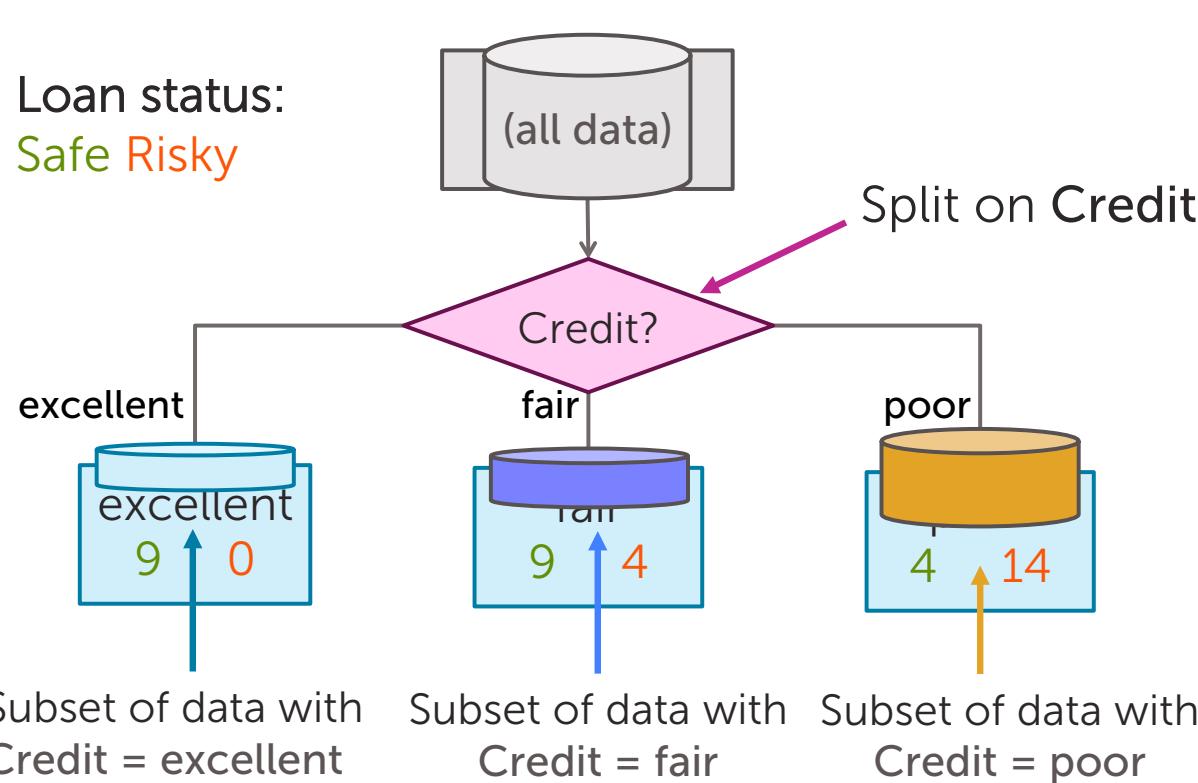


Compact visual notation: Root node

Loan status: **Safe** **Risky**

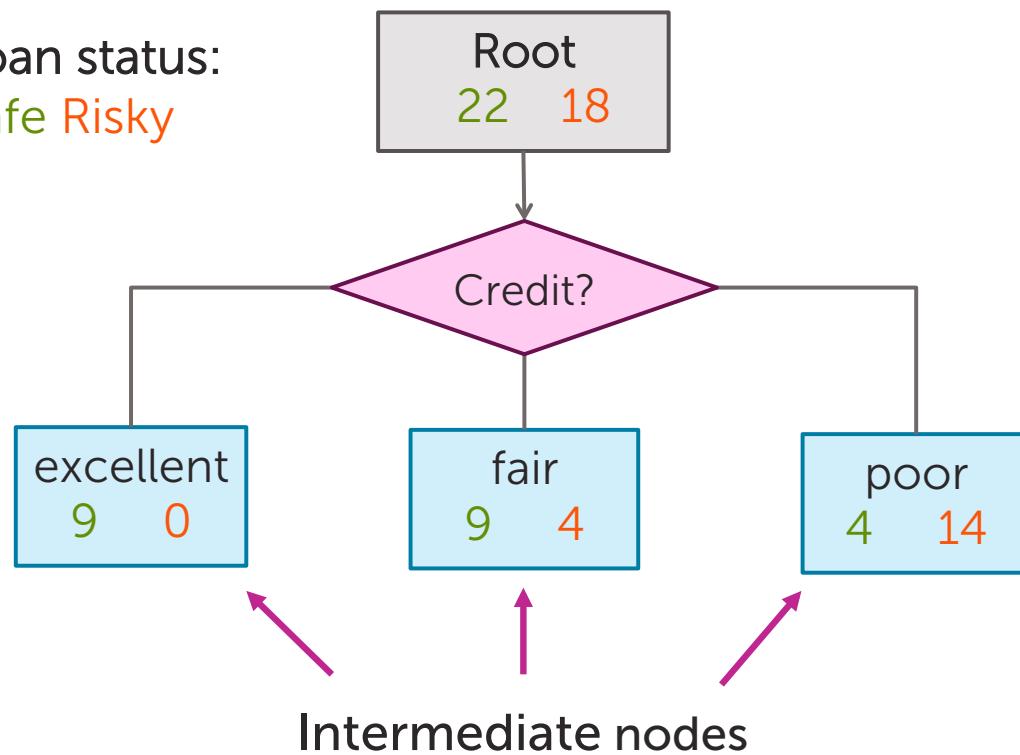


Decision stump: Single level tree

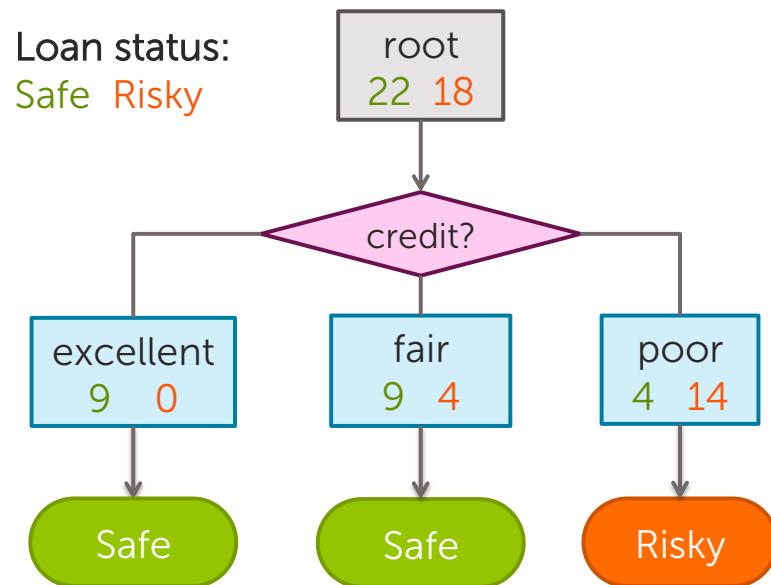


Visual Notation: Intermediate nodes

Loan status:
Safe Risky



Making predictions with a decision stump



↳ For each node, we are looking into majority value for the prediction

y^{1st Decision S}

For each intermediate node,
set $\hat{y} = \text{majority value}$

Selecting best feature to split on

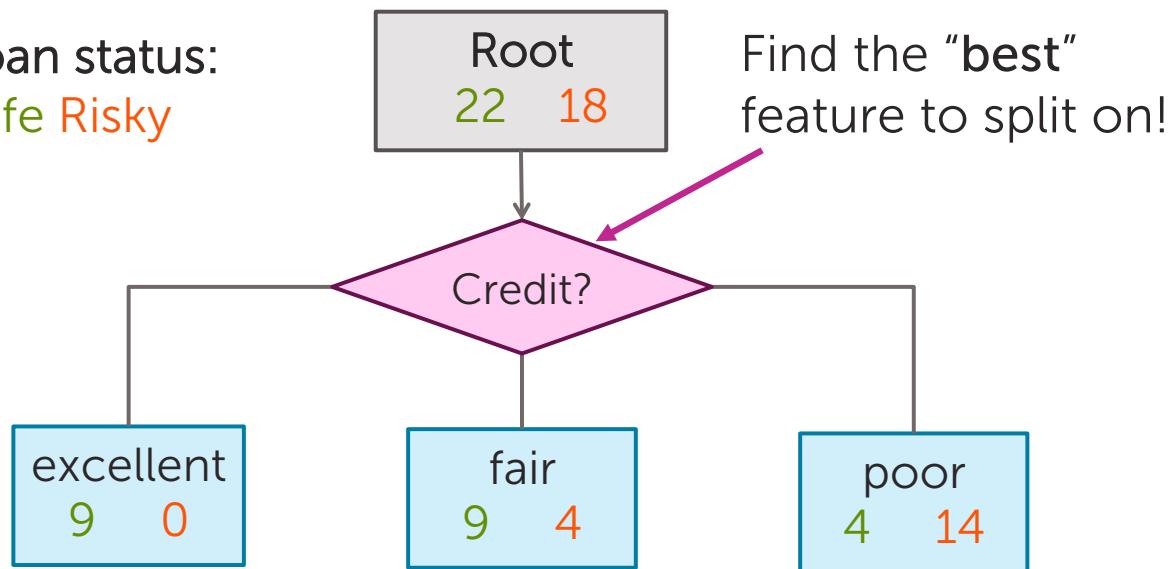
To get better predictions

↳ Going to split further

↳ Before split, we are going to discuss
Why we took CREDIT?

How do we learn a decision stump?

Loan status:
Safe Risky

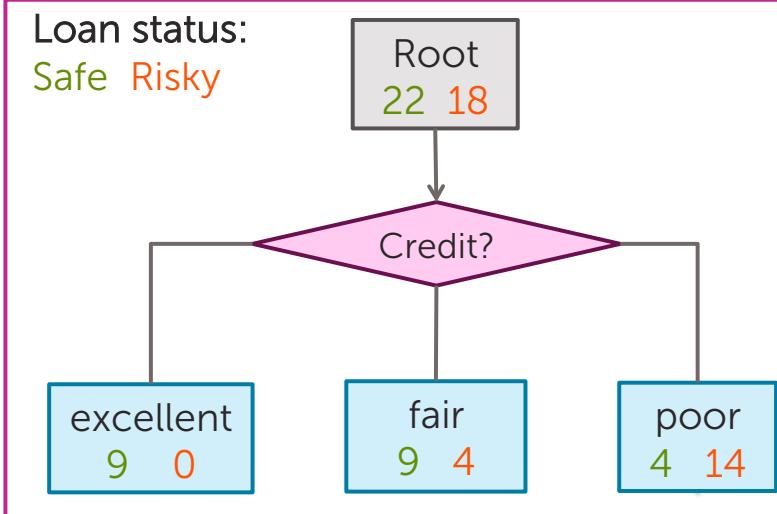


How do we select the best feature?

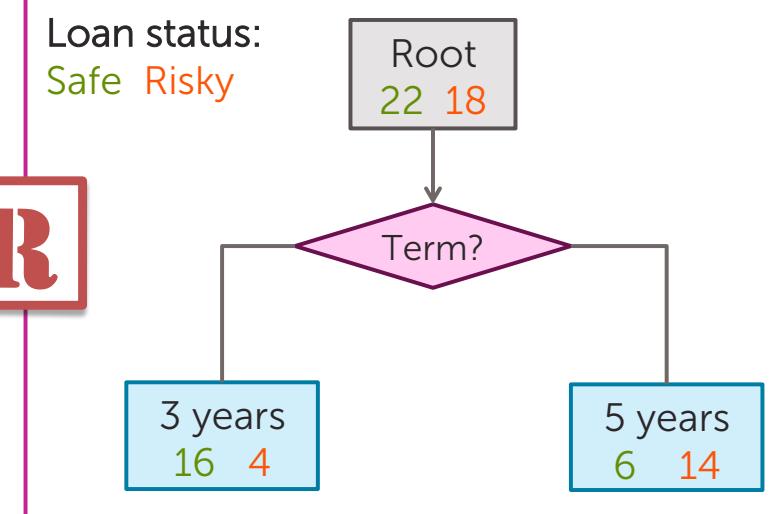
Better split → Gives lowest classification error.

↙ Better?

Choice 1: Split on Credit

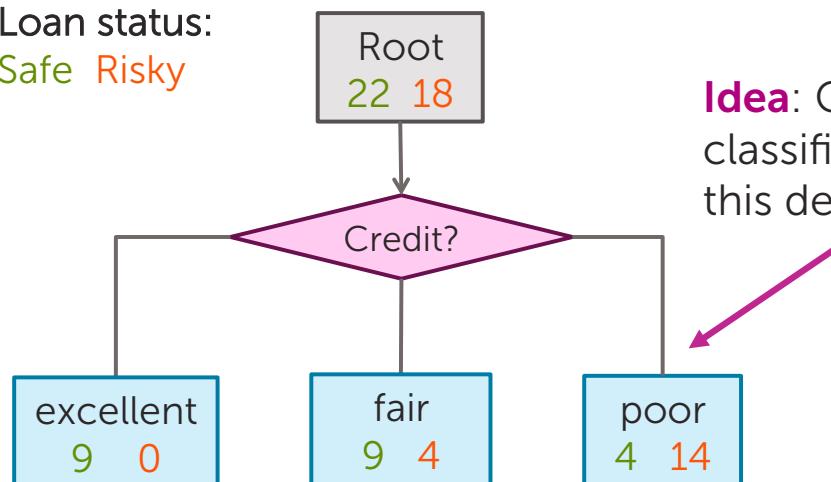


Choice 2: Split on Term



How do we measure effectiveness of a split?

Loan status:
Safe Risky



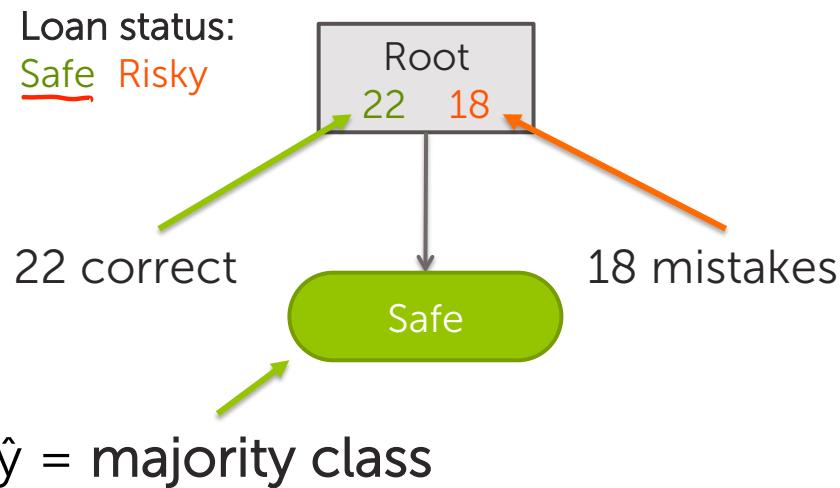
Idea: Calculate classification error of this decision stump

$$\text{Error} = \frac{\# \text{ mistakes}}{\# \text{ data points}}$$

Calculating classification error

- Step 1: \hat{y} = class of majority of data in node
- Step 2: Calculate classification error of predicting \hat{y} for this data

$$9+9+4 \Rightarrow 22$$
$$4+14 \Rightarrow 18$$



$$\text{Error} = \frac{18}{22+18}$$
$$= 0.45$$

Tree	Classification error
(root)	0.45

Error on a
binary classification

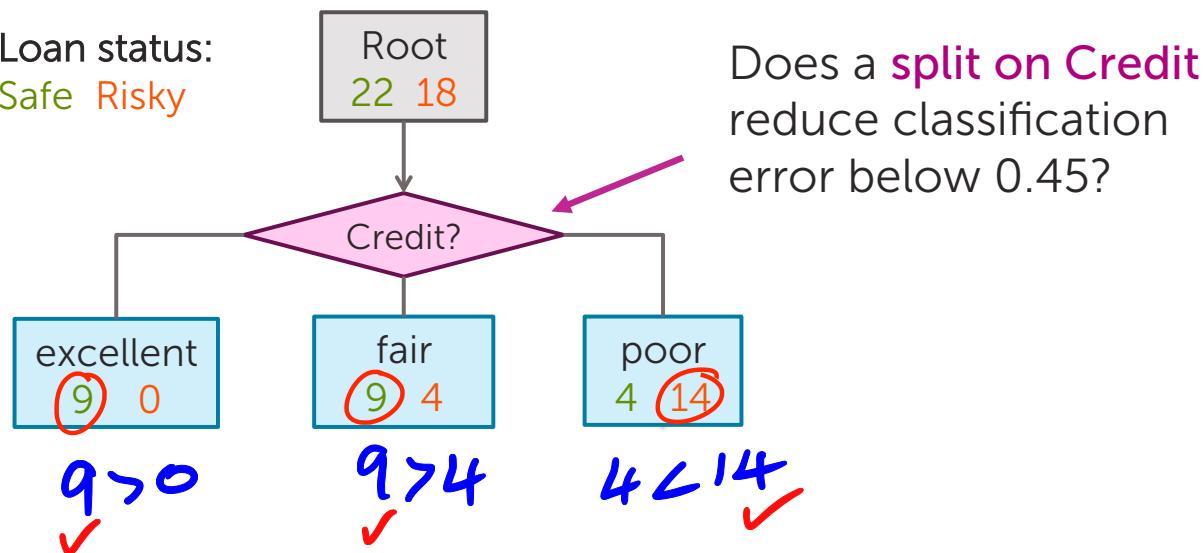
Not splitting on anything \Rightarrow Gives a bad result

How good is the decision stump that splits on credit?
And how does it compare to not split on anything, which has a classification error of 0.45

Choice 1: Split on credit history?

Choice 1: Split on Credit

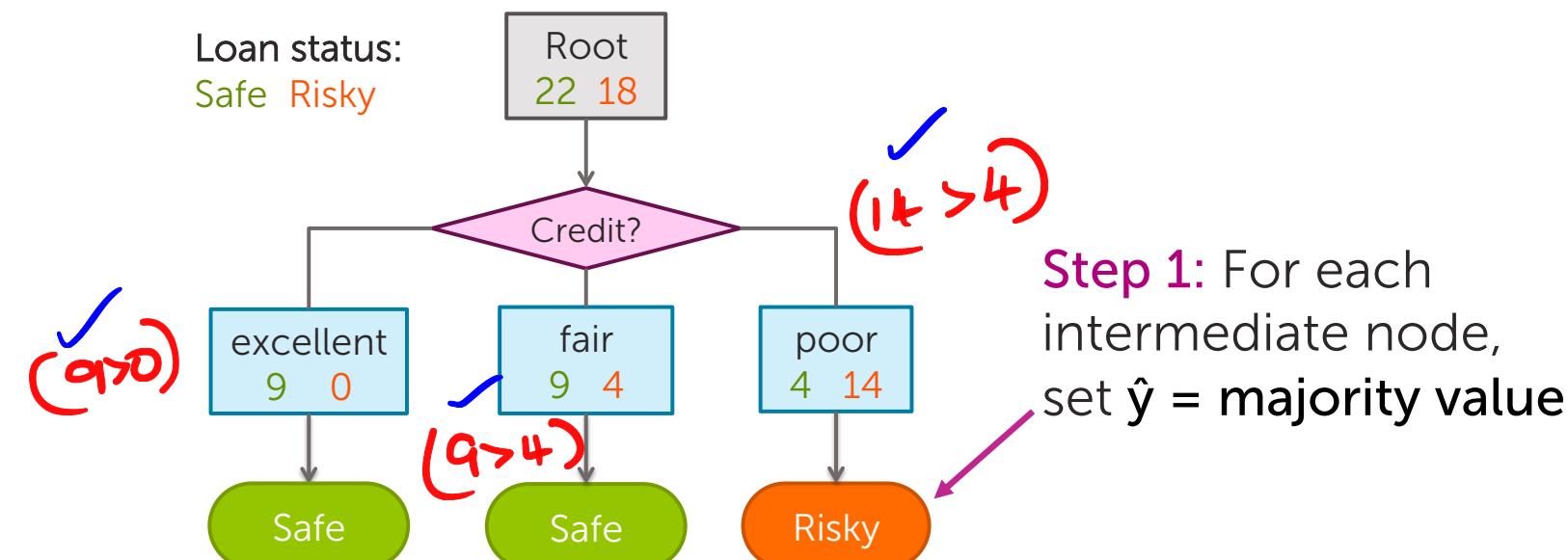
Loan status:
Safe Risky



How good is the split on Credit?

Choice 1: Split on Credit

Loan status:
Safe Risky



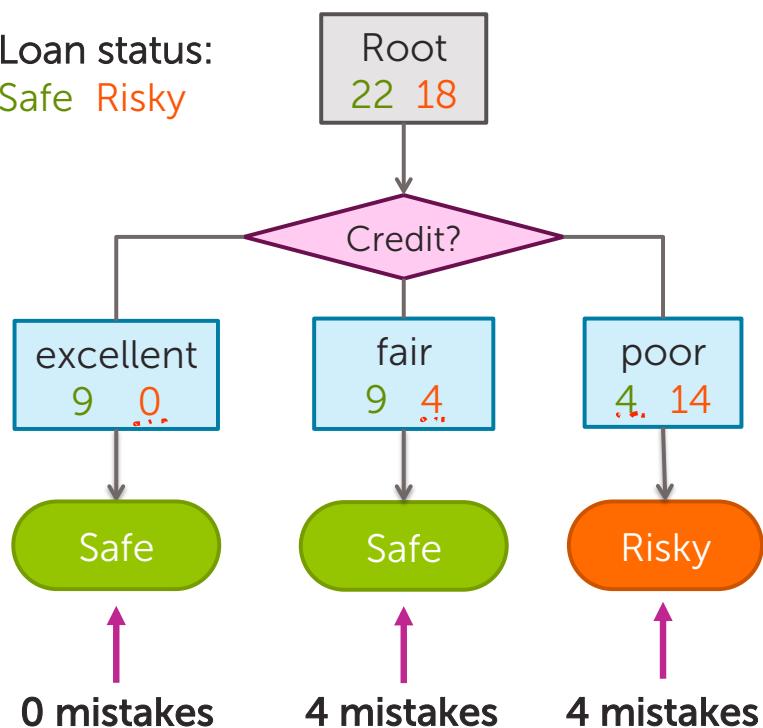
Step 1: For each intermediate node, set $\hat{y} = \text{majority value}$

Majority plays a Major Role

Split on Credit: Classification error

Choice 1: Split on Credit

Loan status:
Safe Risky



$$\text{Error} = \frac{4+4}{40} = 0.20$$

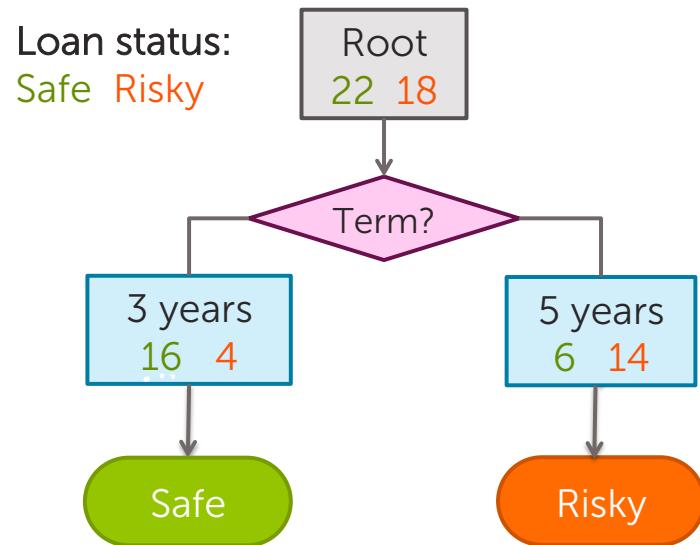
Tree	Classification error
(root)	0.45
Split on credit	0.2

$$\frac{0 + 4 + 4}{9 + 13 + 18}$$

$$\Rightarrow \frac{8}{40} \rightarrow 0.20$$

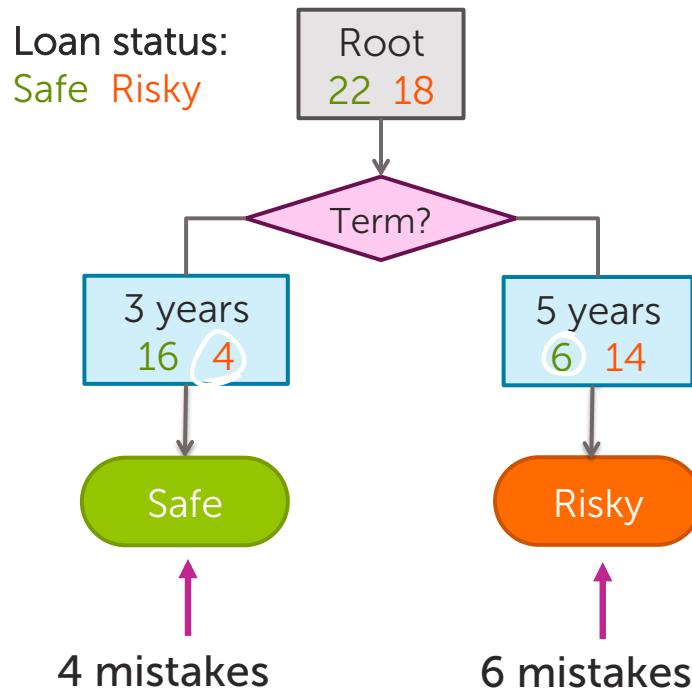
Choice 2: Split on Term?

Choice 2: Split on Term



Choice 2: Split on Term?

Choice 2: Split on Term



$$\text{Error} = \frac{4+6}{40} = 0.25$$

Tree	Classification error
(root)	0.45
Split on credit	0.2
Split on term	0.25

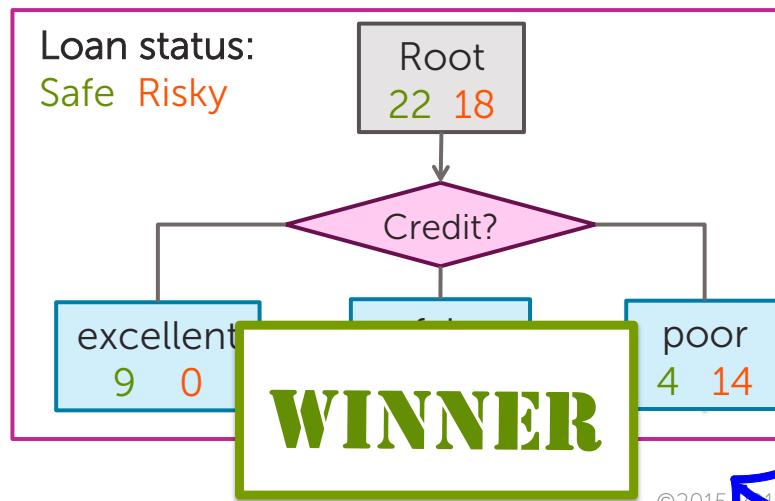
$$\frac{4+6}{20+20} \Rightarrow \frac{10}{40} \Rightarrow 0.25$$

Choice 1 vs Choice 2

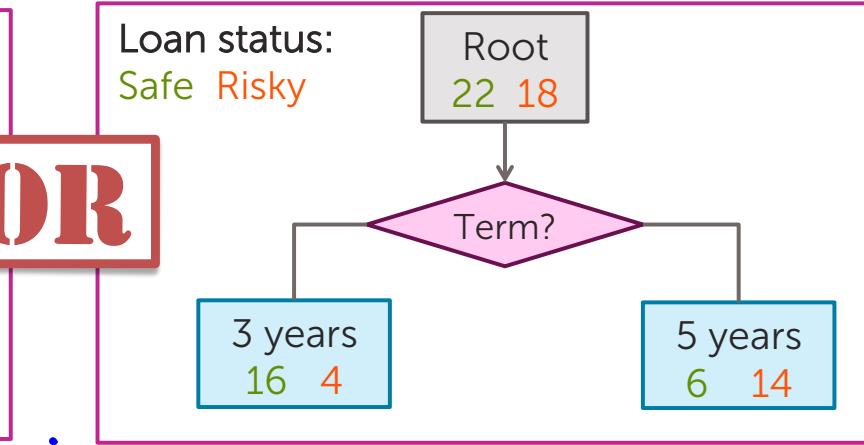
Tree	Classification error
(root)	0.45
split on <u>credit</u>	0.2
split on loan term	0.25

First split!

Choice 1: Split on Credit



Choice 2: Split on Term



57

©2015-2016 Emily Fox & Carlos Guestrin

In Greedy Algorithm perspective

winner

loser

Machine Learning Specialization

Feature split selection algorithm

- Given a subset of data M (a node in a tree)
- For each feature $h_i(x)$: \leftarrow ^{credit, term, income}
- 1. Split data of M according to feature $h_i(x)$
- 2. Compute classification error split
- Chose feature $h^*(x)$ with lowest classification error \nwarrow
credit

Credit is the winner

Greedy decision tree learning

- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
 - Step 3: If nothing more to, make predictions
 - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Among the *these features*
are picked up the *Credit
feature*

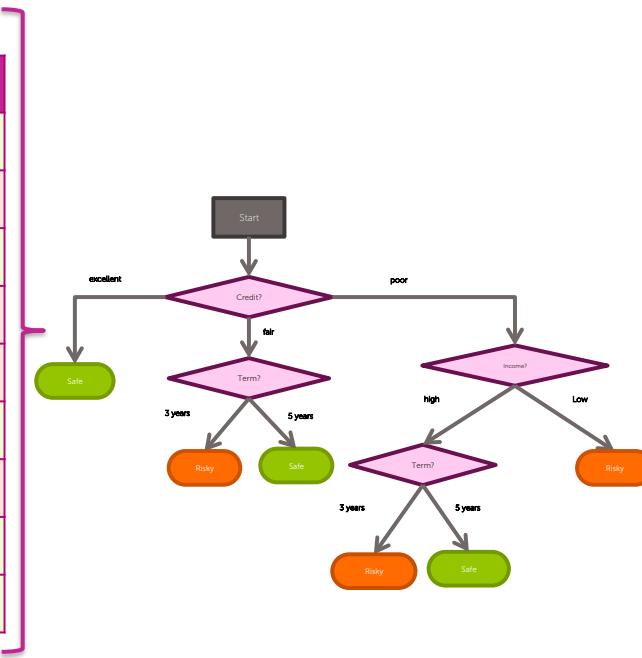
Pick feature split
leading to lowest
classification error



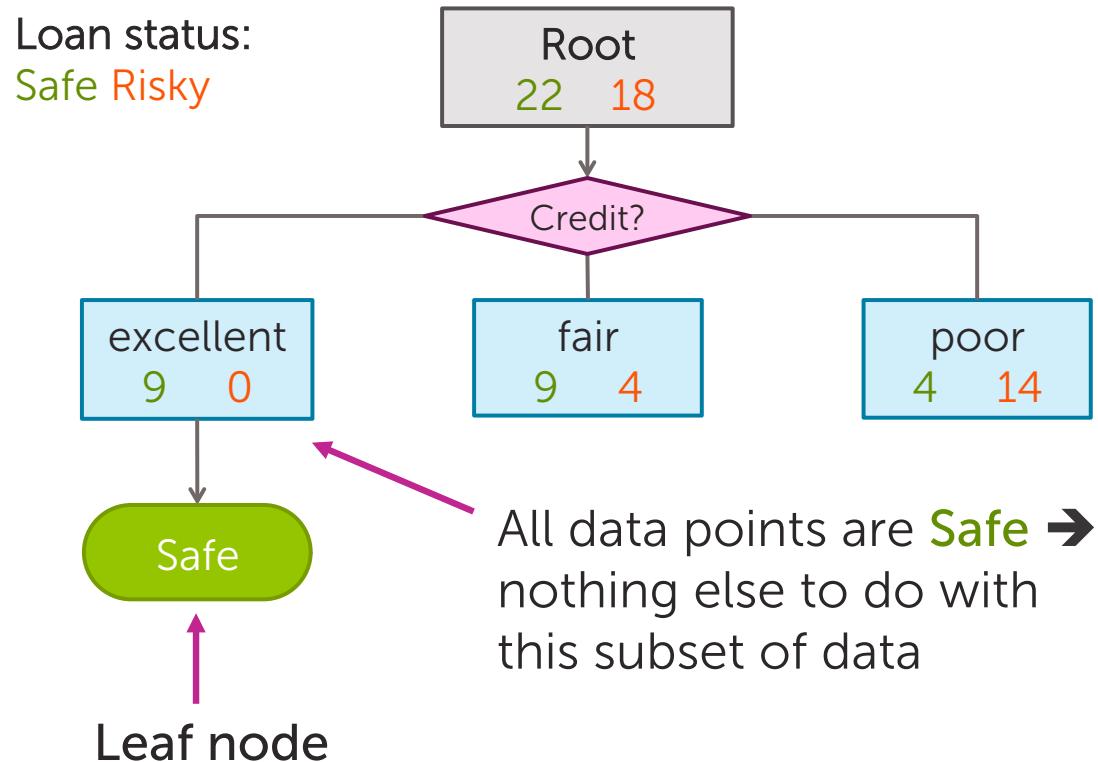
Decision Tree Learning: *Recursion & Stopping conditions*

Learn decision tree from data?

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

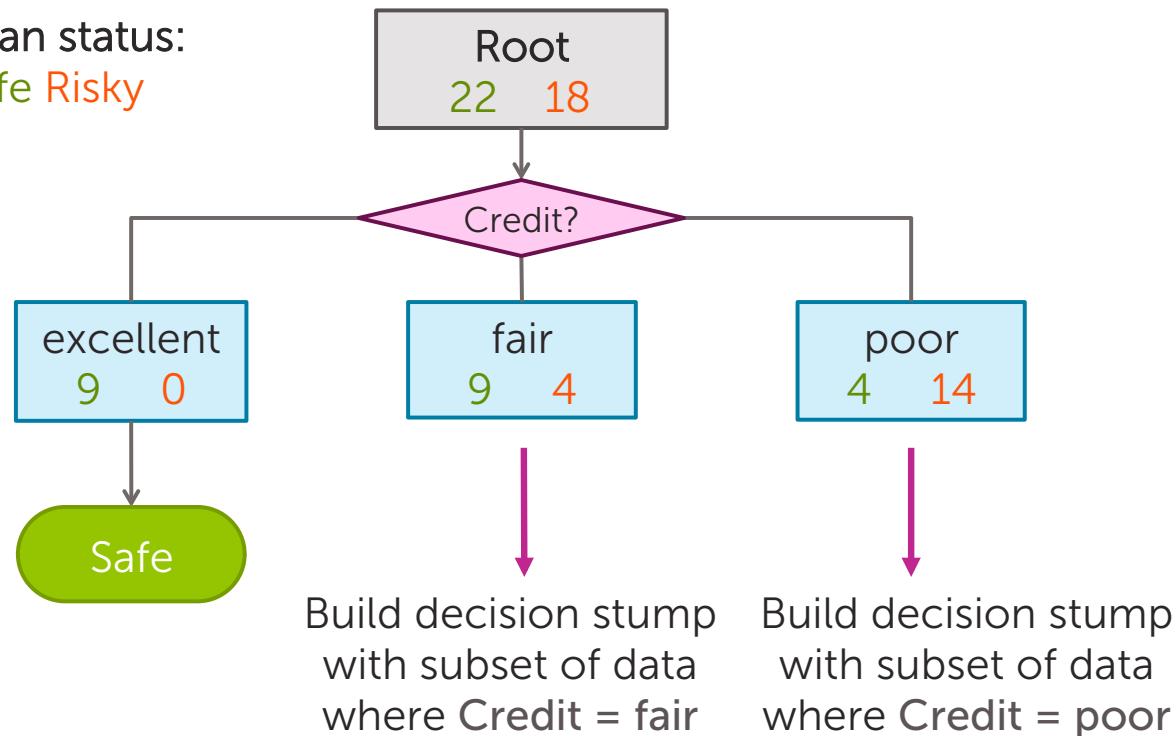


We've learned a decision stump, what next?

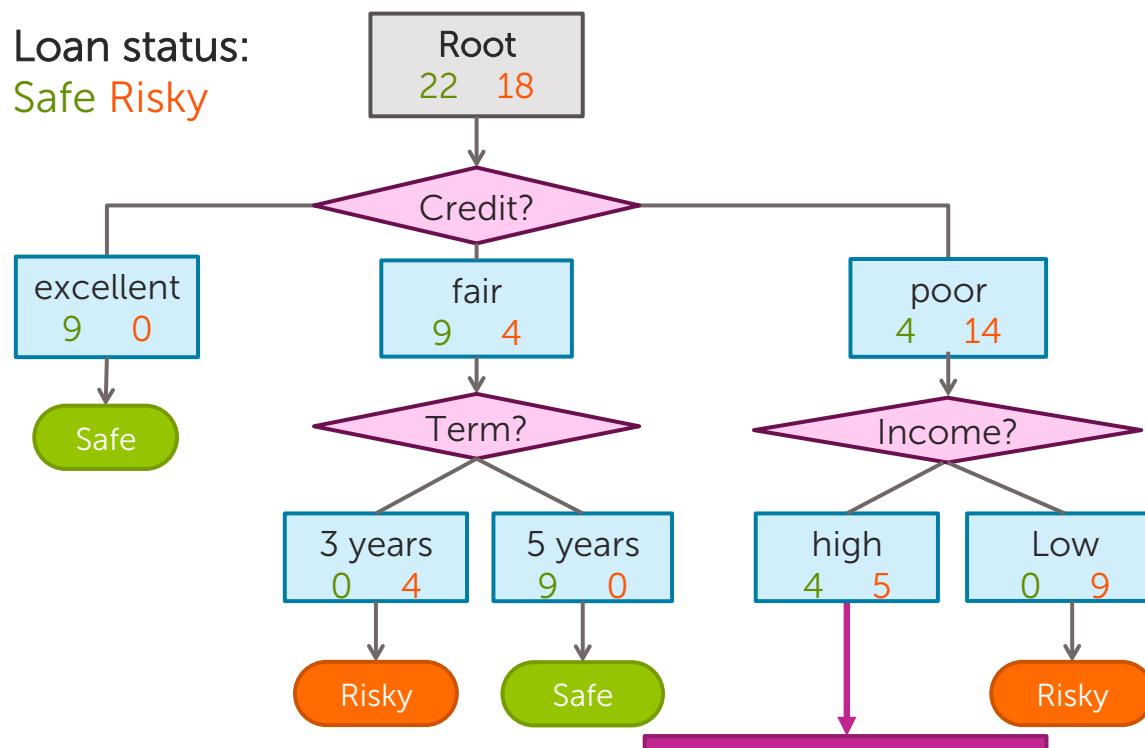


Tree learning = Recursive stump learning

Loan status:
Safe Risky

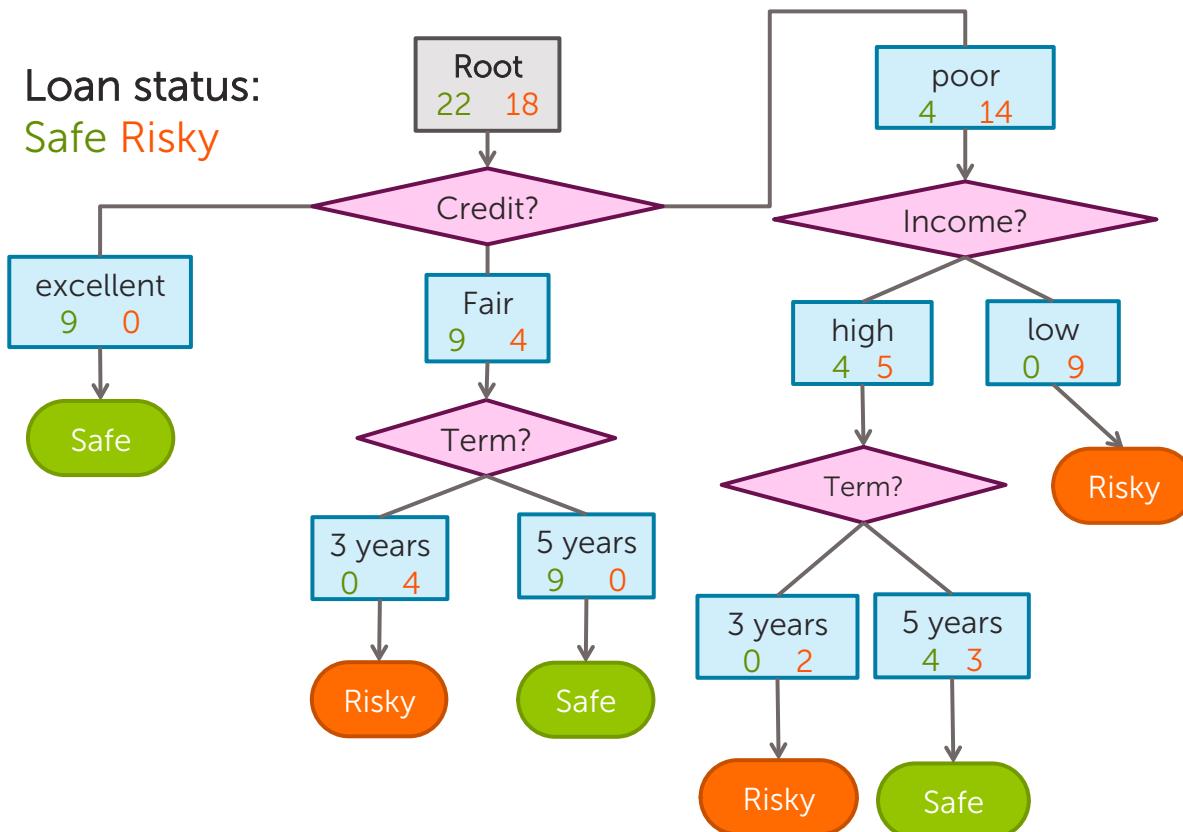


Second level

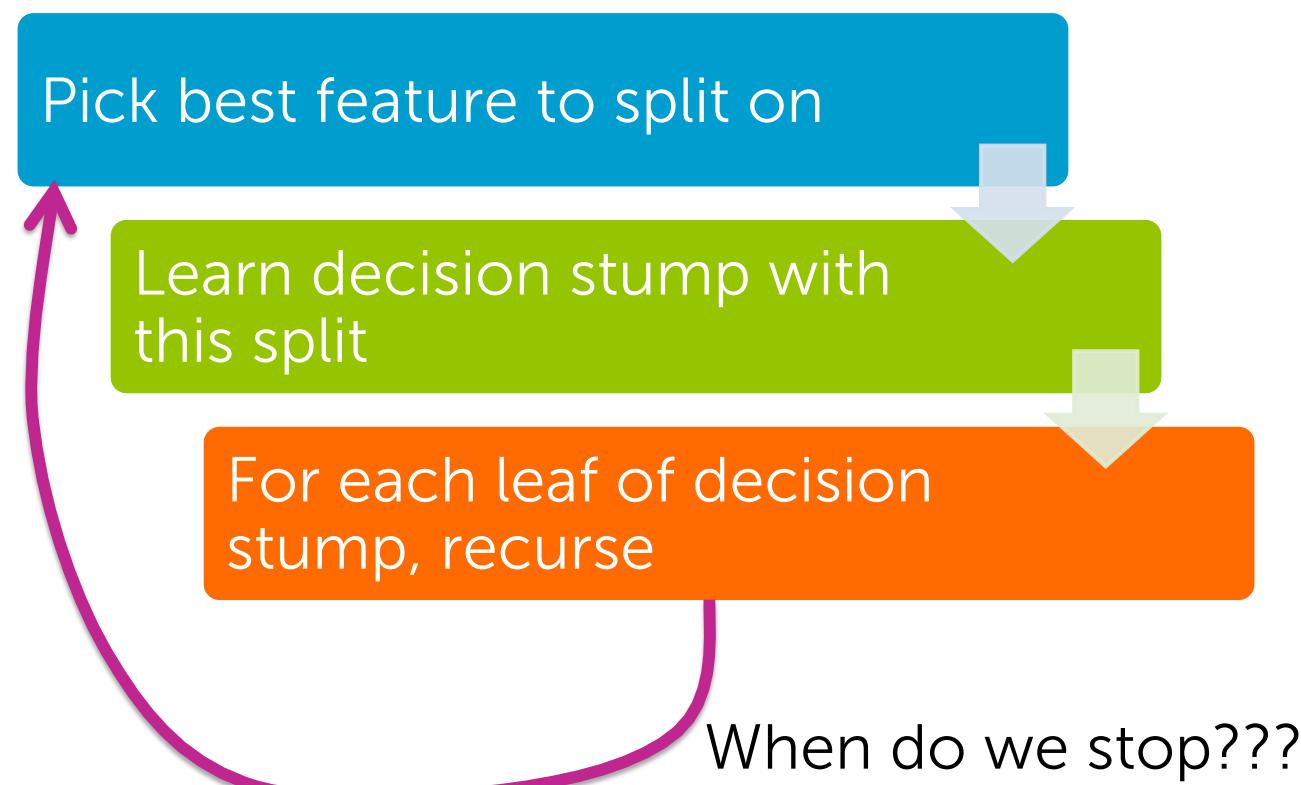


Build another stump
these data points

Final decision tree

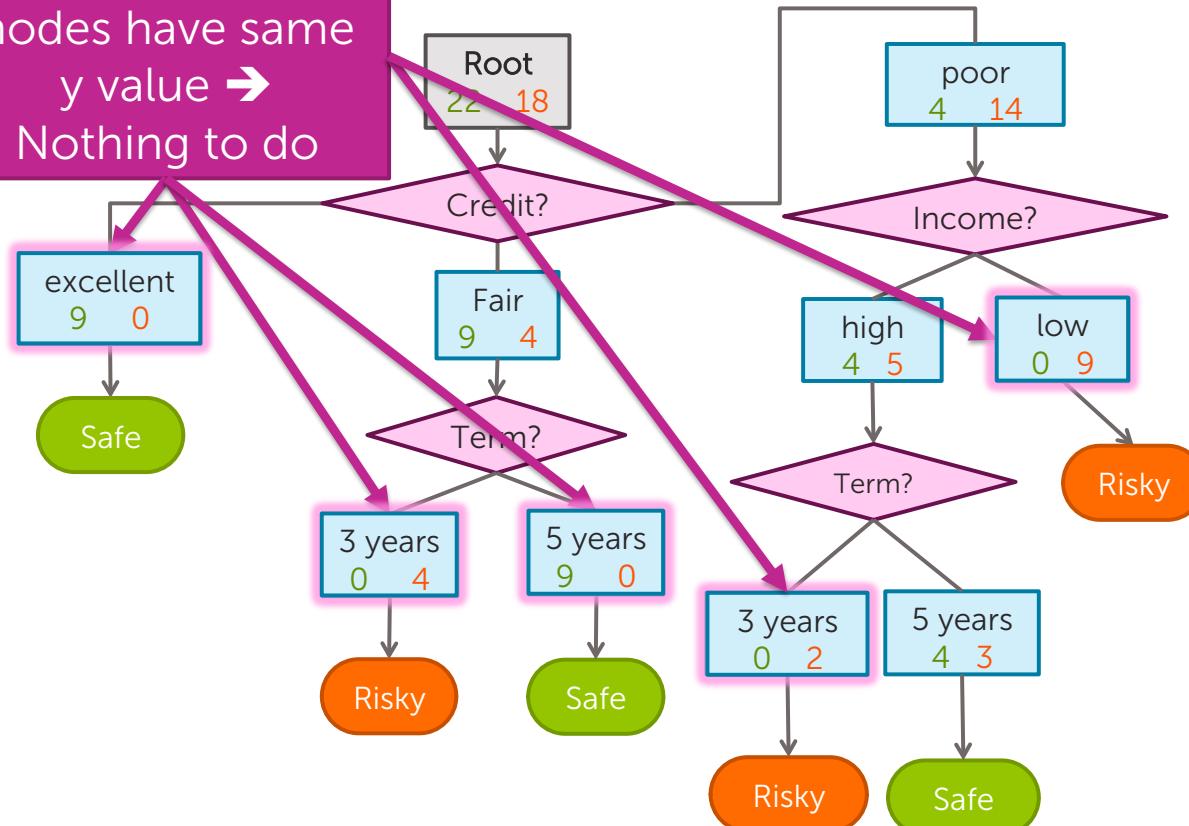


Simple greedy decision tree learning



Stopping condition 1: All data agrees on y

All data in these nodes have same y value →
Nothing to do

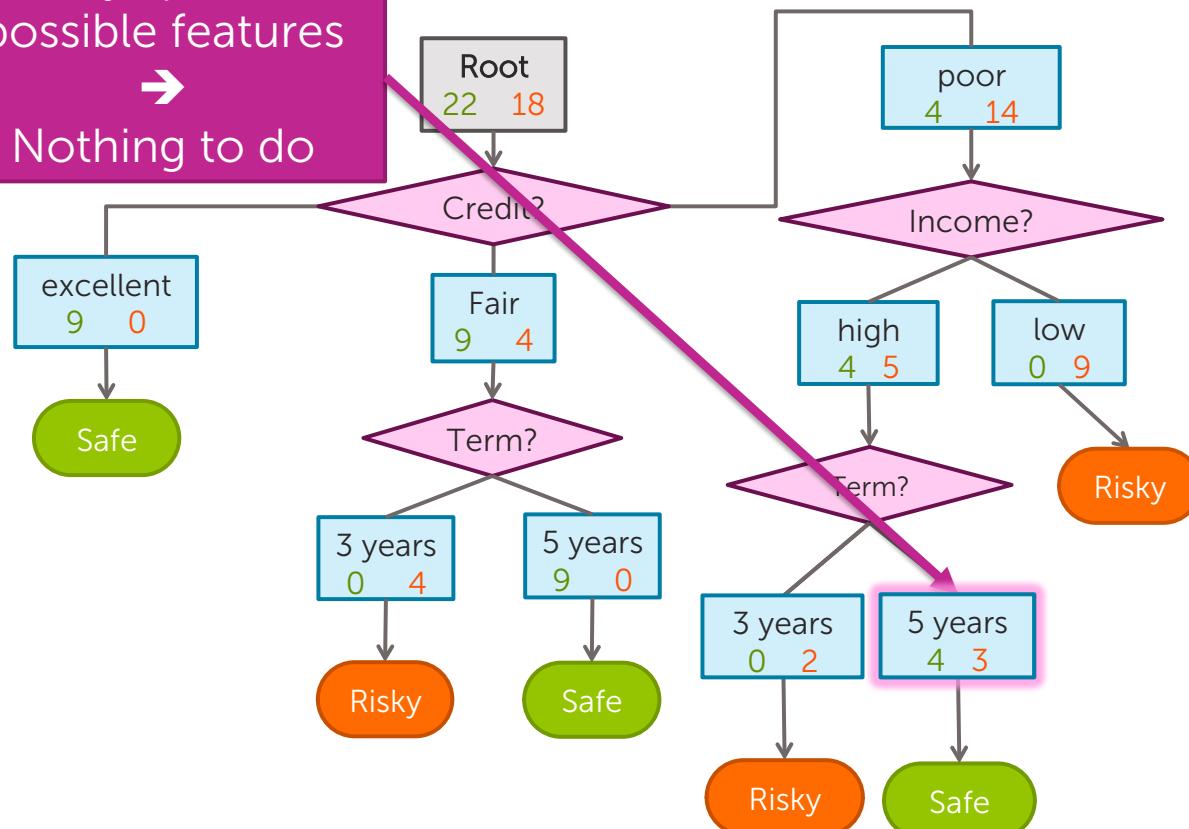


Stopping condition 2: Already split on all features

Already split on all possible features



Nothing to do



Greedy decision tree learning

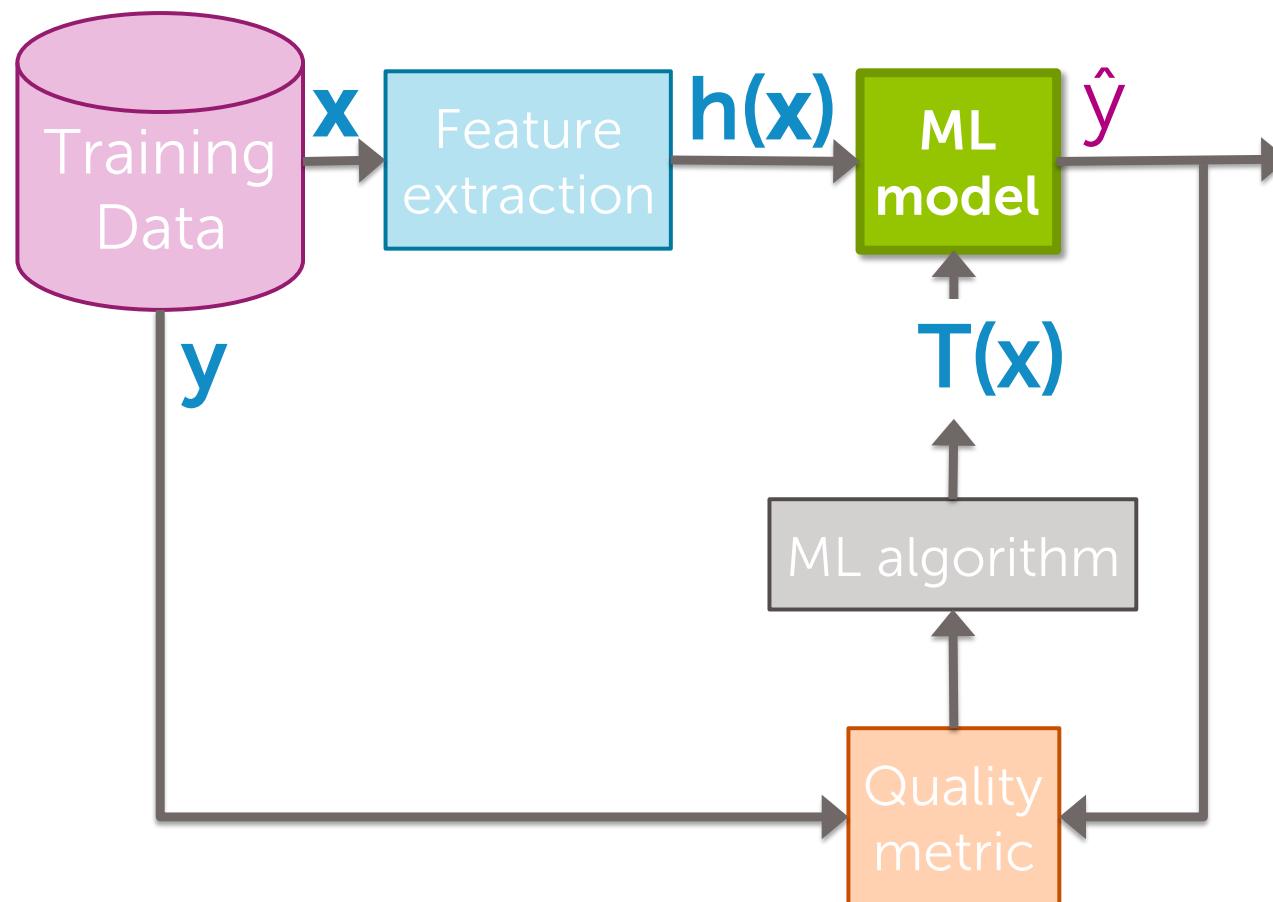
- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
 - Step 3: If nothing more to, make predictions
 - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Pick feature split leading to lowest classification error

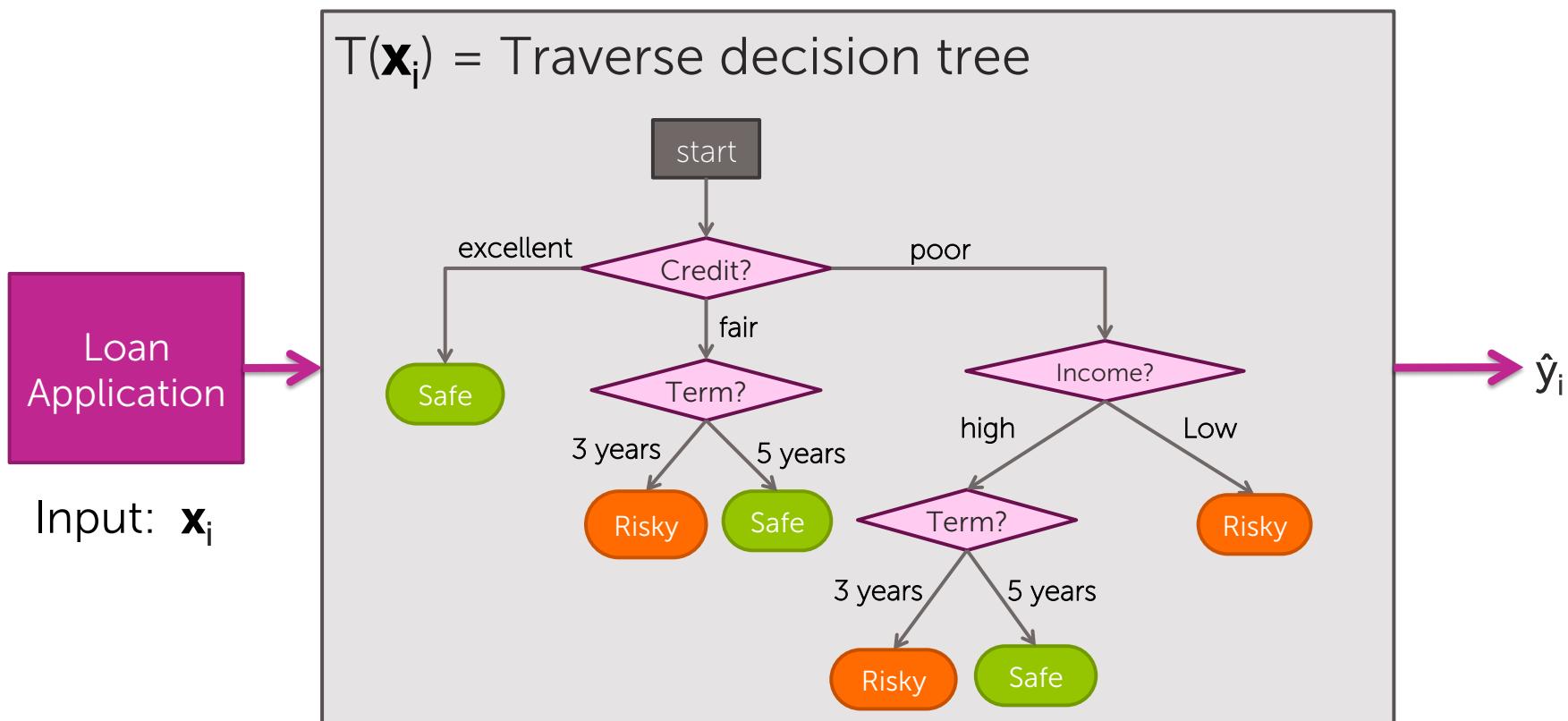
Stopping conditions 1 & 2

Recursion

Predictions with decision trees

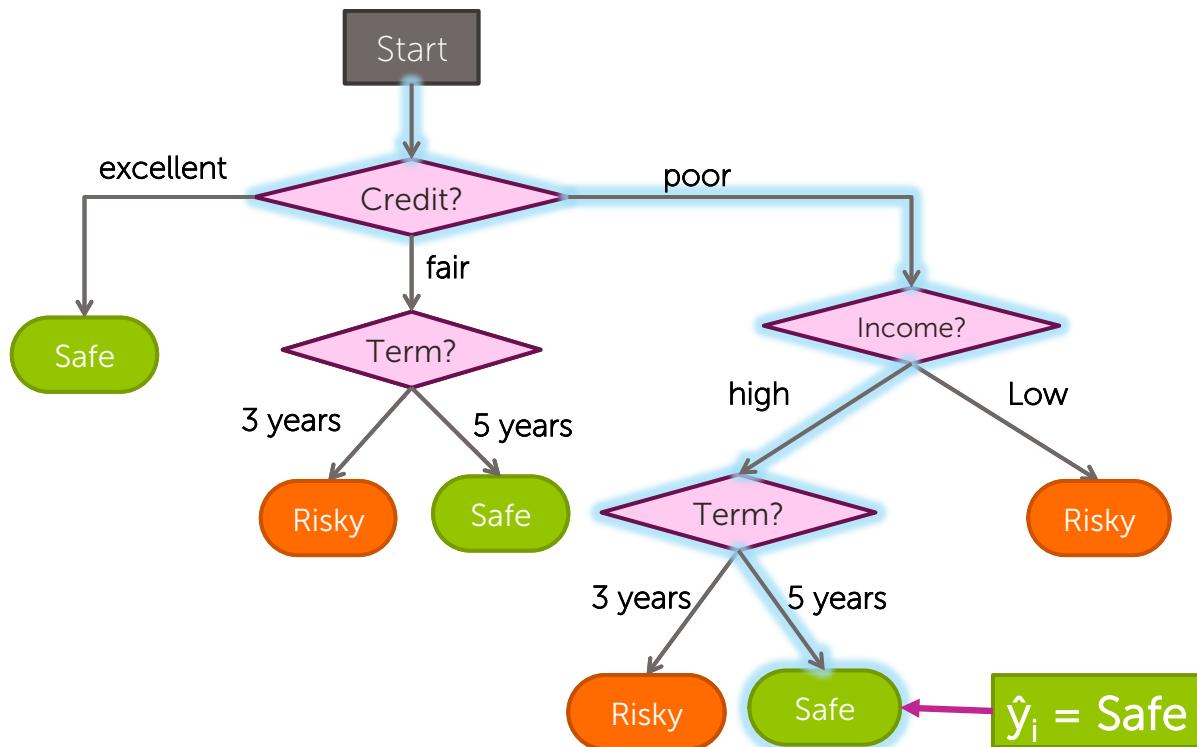


Decision tree model



Traversing a decision tree

$x_i = (\text{Credit} = \text{poor}, \text{Income} = \text{high}, \text{Term} = 5 \text{ years})$



Decision tree prediction algorithm

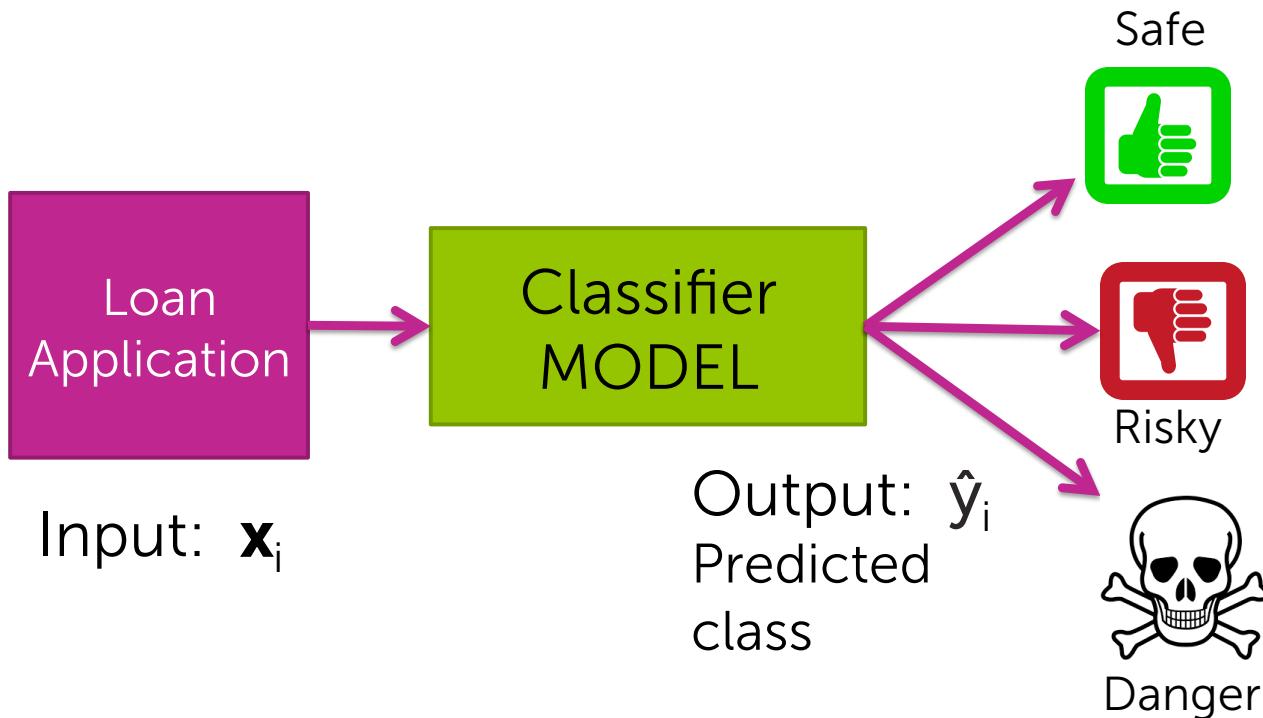
predict(tree_node, input)

- If current `tree_node` is a leaf:
 - `return` majority class of data points in leaf
- else:
 - `next_note` = child node of `tree_node` whose feature value agrees with `input`
 - `return predict(next_note, input)`



Multiclass classification & predicting probabilities

Multiclass prediction



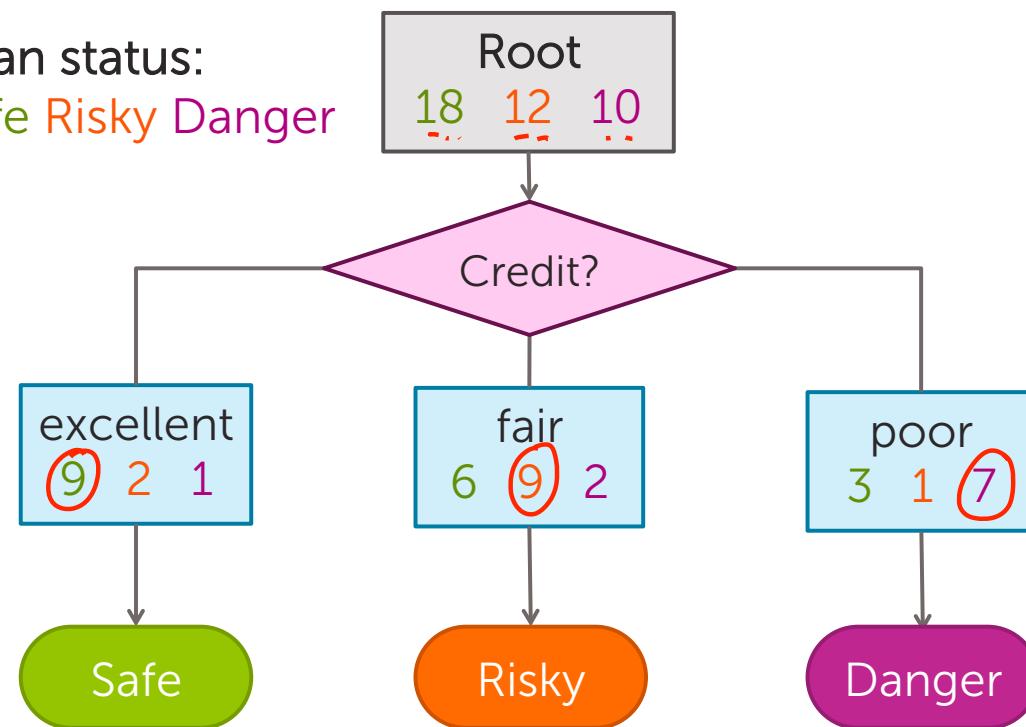
Multiclass decision stump

$N = 40$,
1 feature,
3 classes

Credit	y
excellent	safe
fair	risky
fair	safe
poor	danger
excellent	risky
fair	safe
poor	danger
poor	safe
fair	safe
...	...

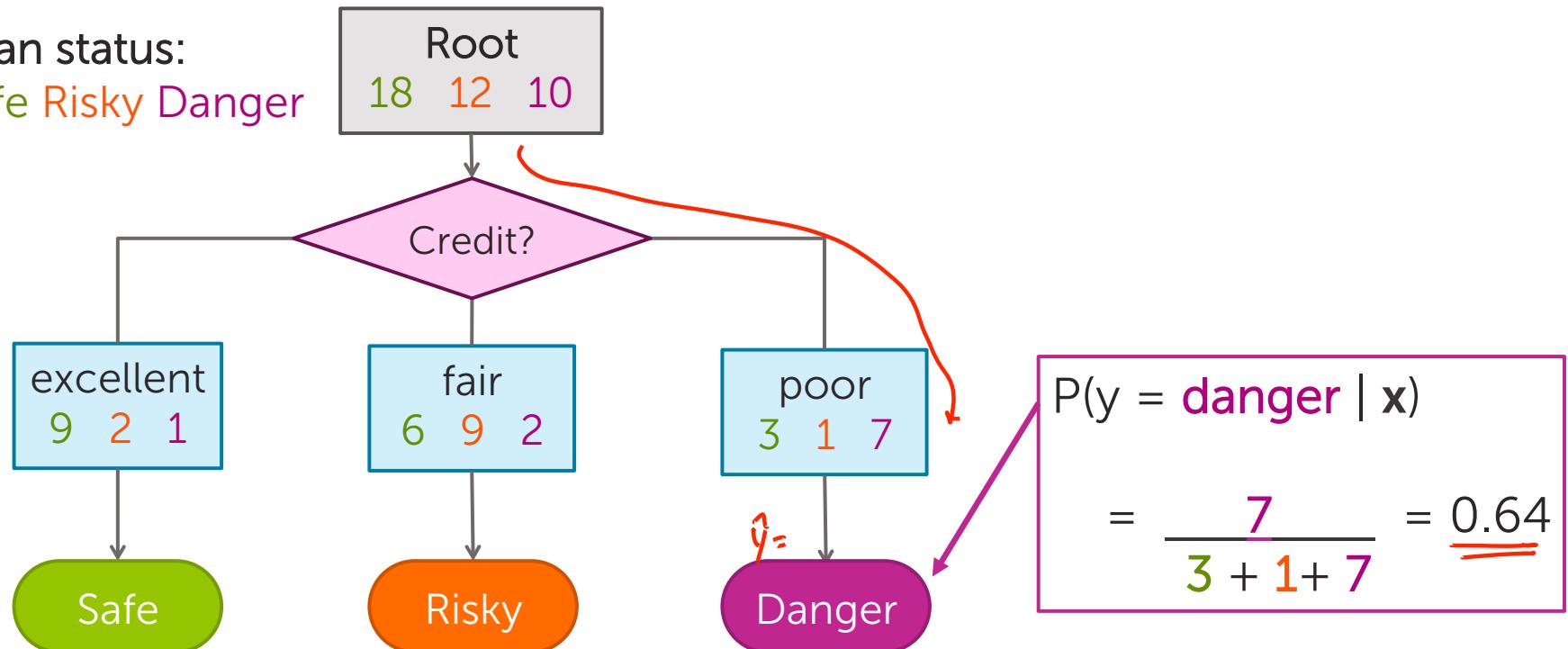


Loan status:
Safe Risky Danger



Predicting probabilities with decision trees

Loan status:
Safe Risky Danger



Decision tree learning: *Real valued features*

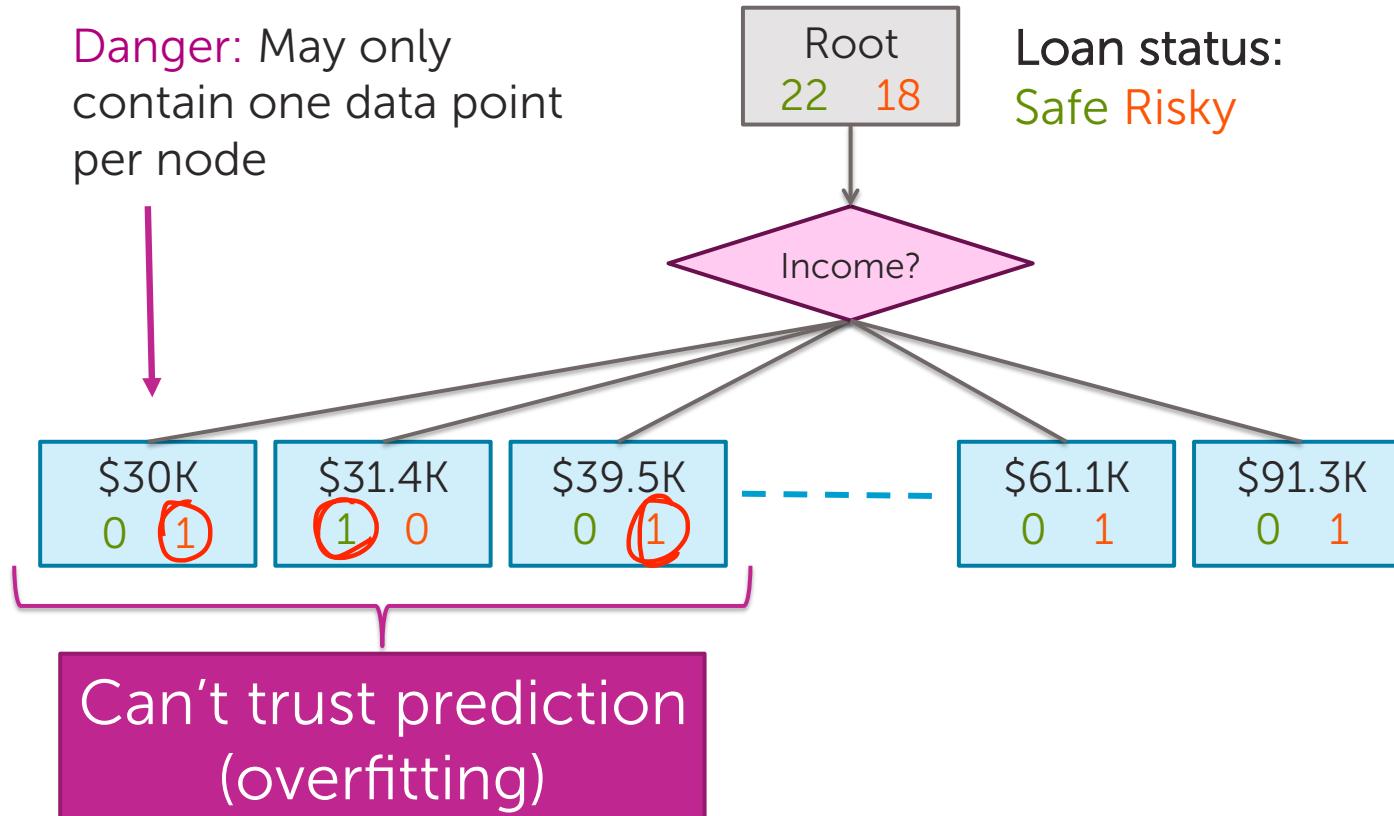
How do we use real values inputs?

Income	Credit	Term	y
\$105 K	excellent	3 yrs	Safe
\$112 K	good	5 yrs	Risky
\$73 K	fair	3 yrs	Safe
\$69 K	excellent	5 yrs	Safe
\$217 K	excellent	3 yrs	Risky
\$120 K	good	5 yrs	Safe
\$64 K	fair	3 yrs	Risky
\$340 K	excellent	5 yrs	Safe
\$60 K	good	3 yrs	Risky

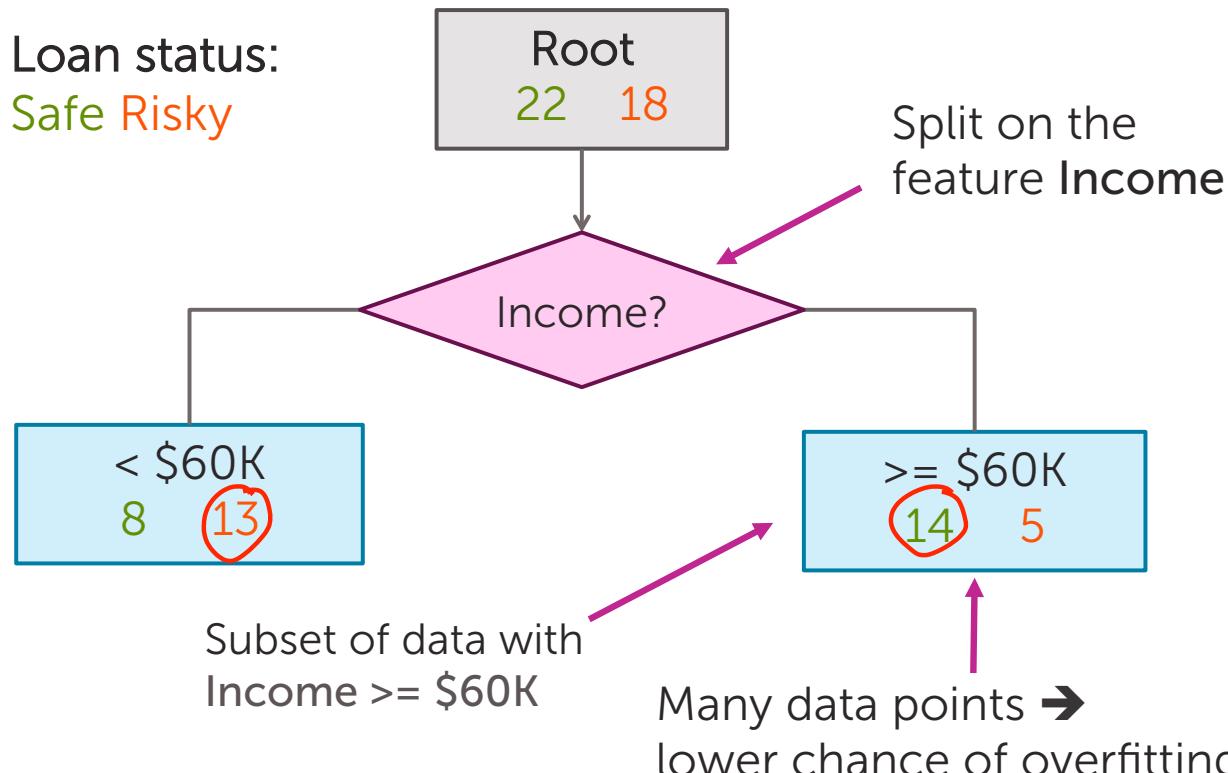
Split on each numeric value?

Danger: May only contain one data point per node

Loan status:
Safe Risky

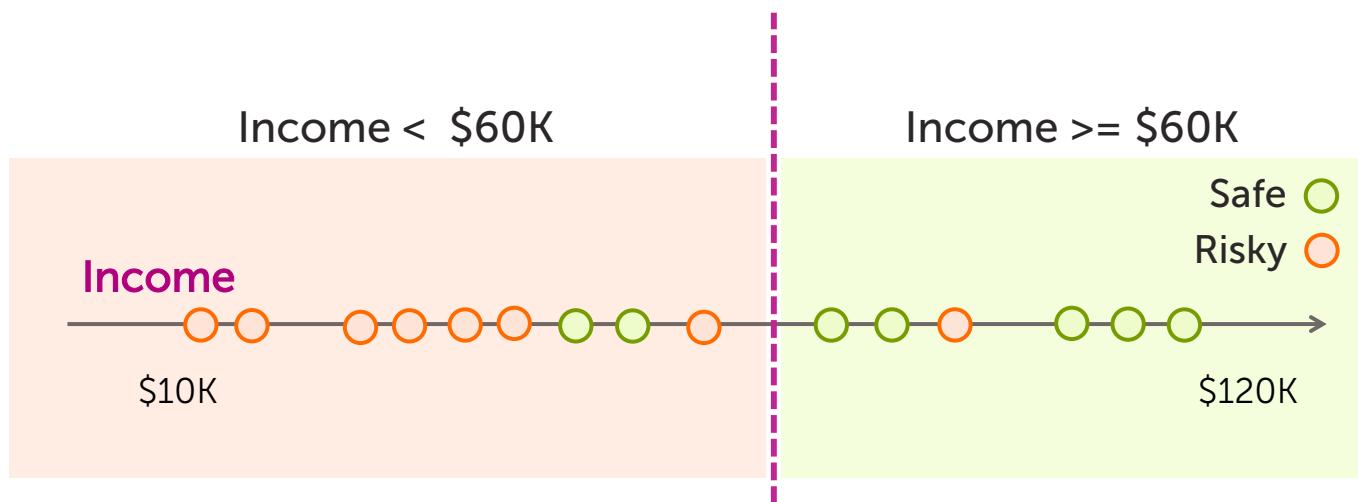


Alternative: Threshold split

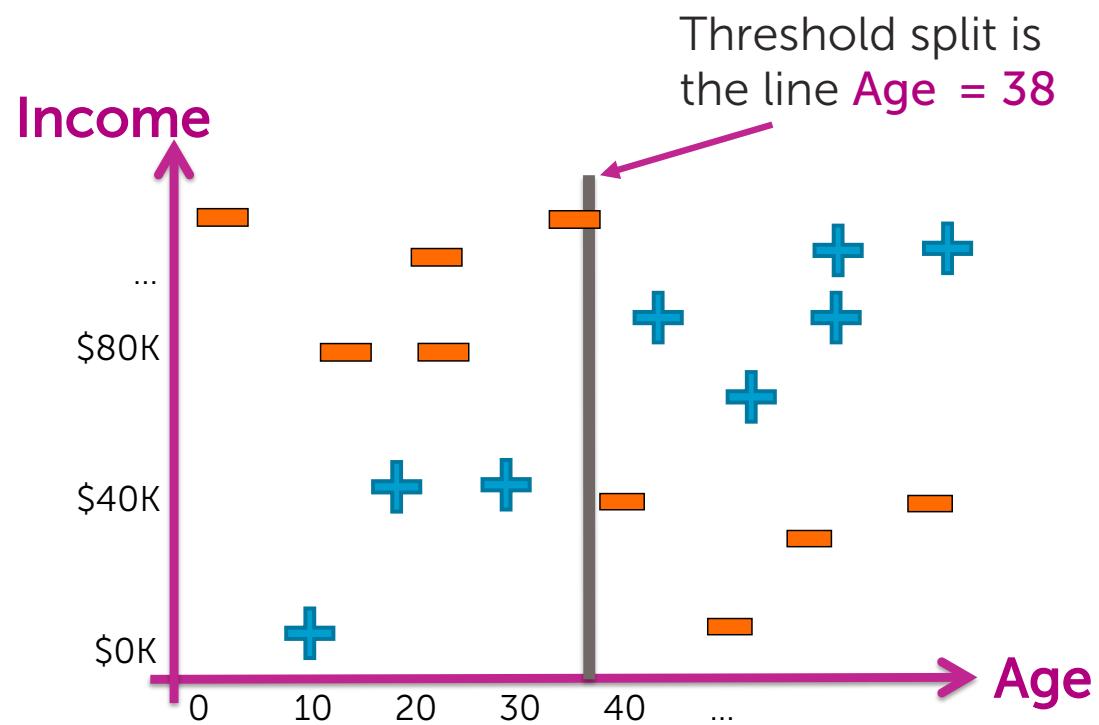


Threshold splits in 1-D

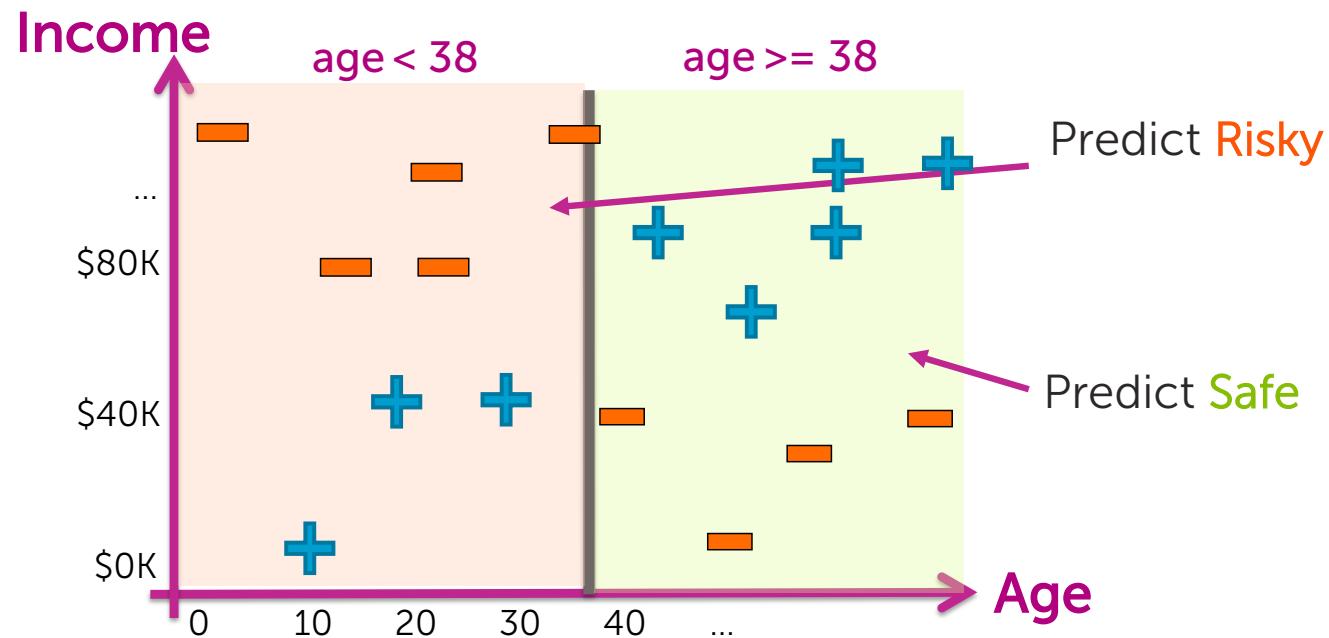
Threshold split is the line
 $\text{Income} = \$60K$



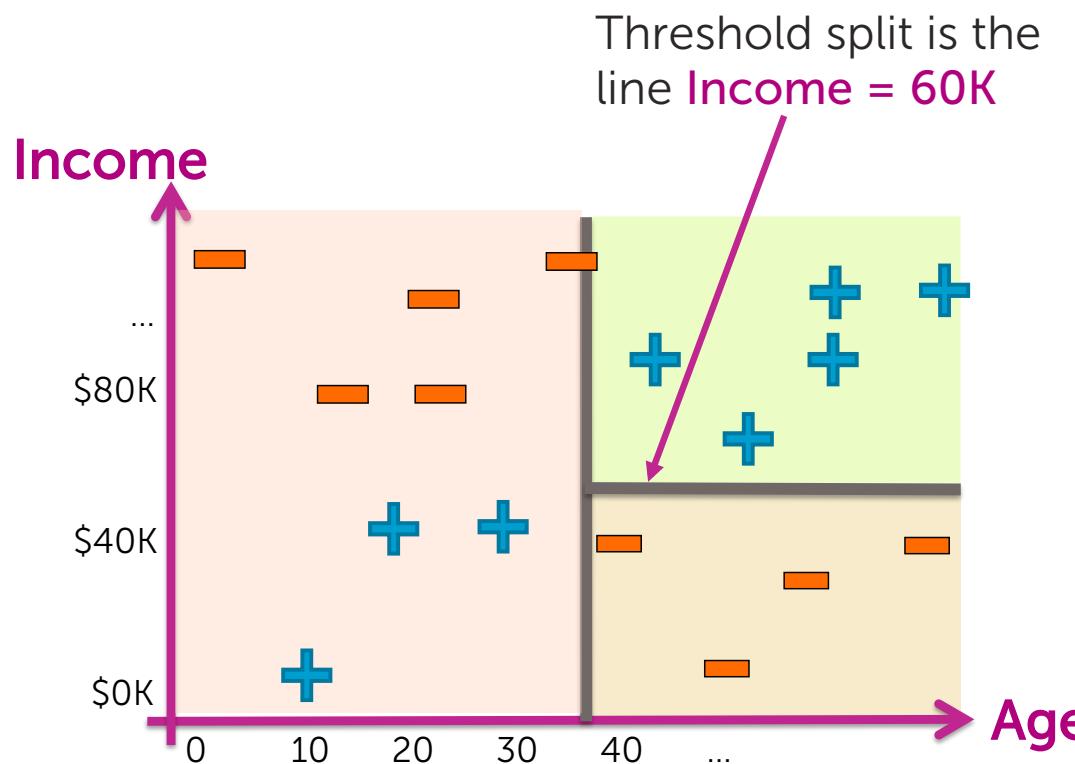
Visualizing the threshold split



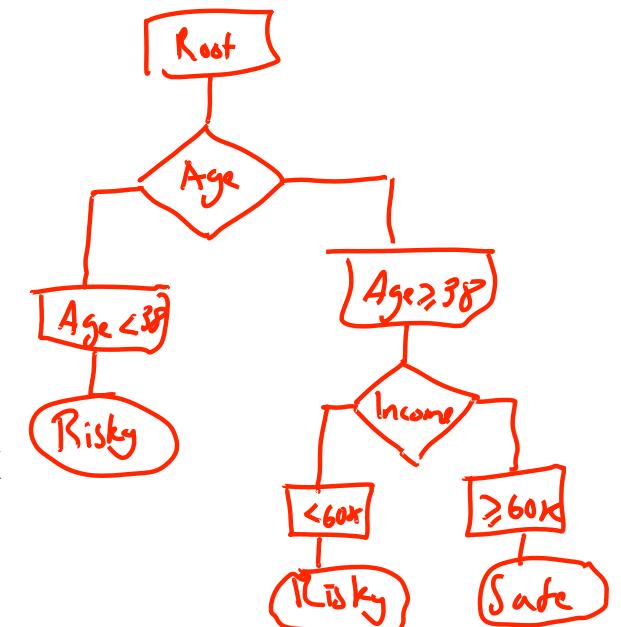
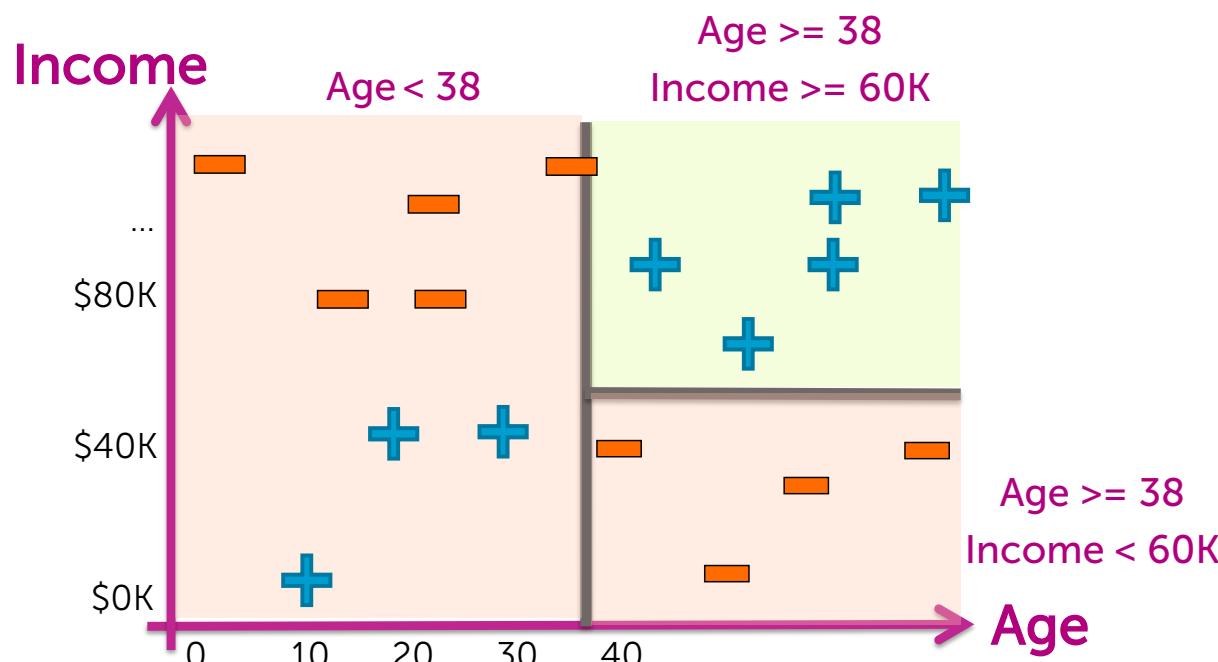
Split on Age ≥ 38



Depth 2: Split on Income $\geq \$60K$



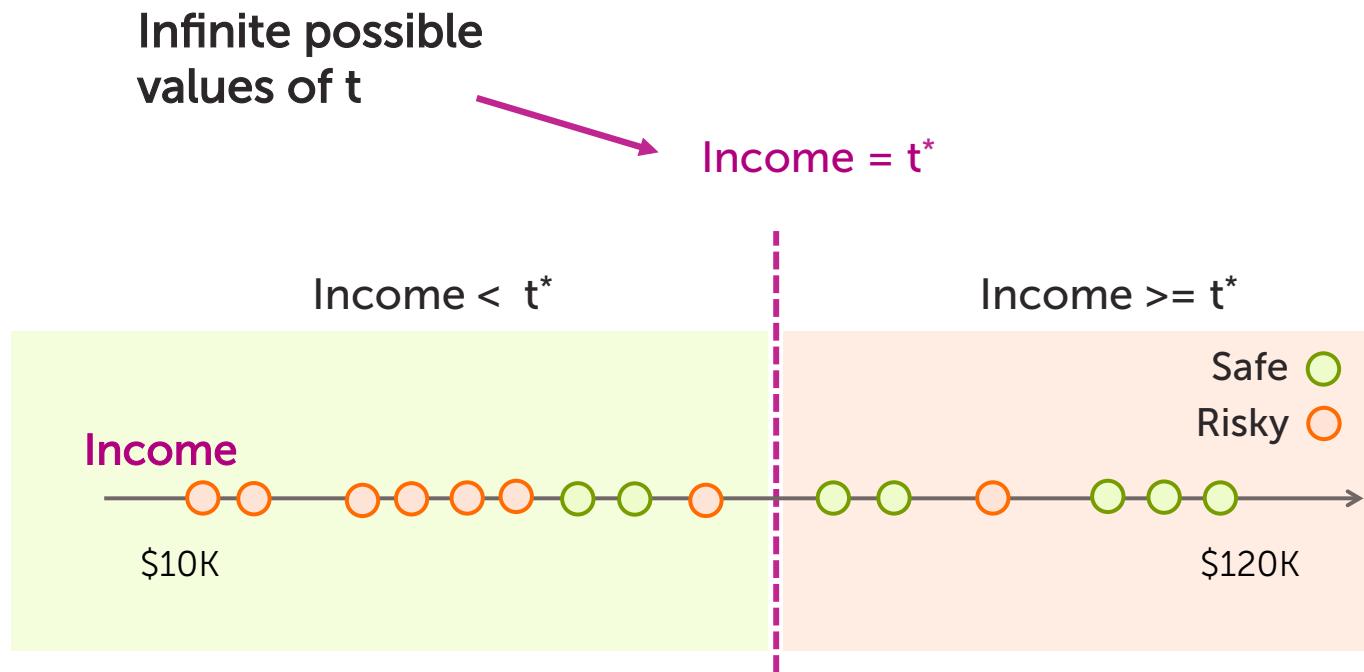
Each split partitions the 2-D space



Finding the best threshold split

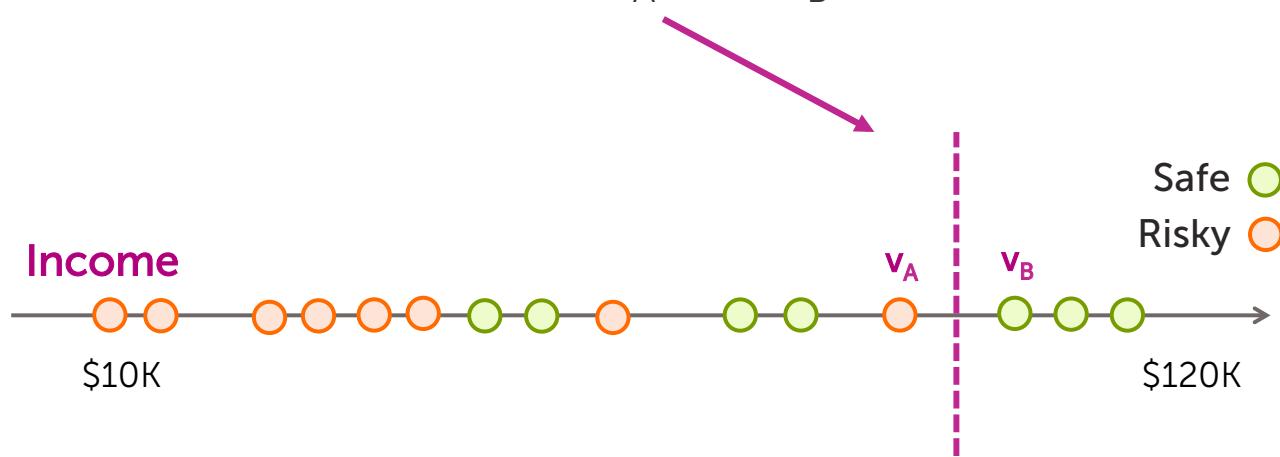
OPTIONAL

Finding the best threshold split



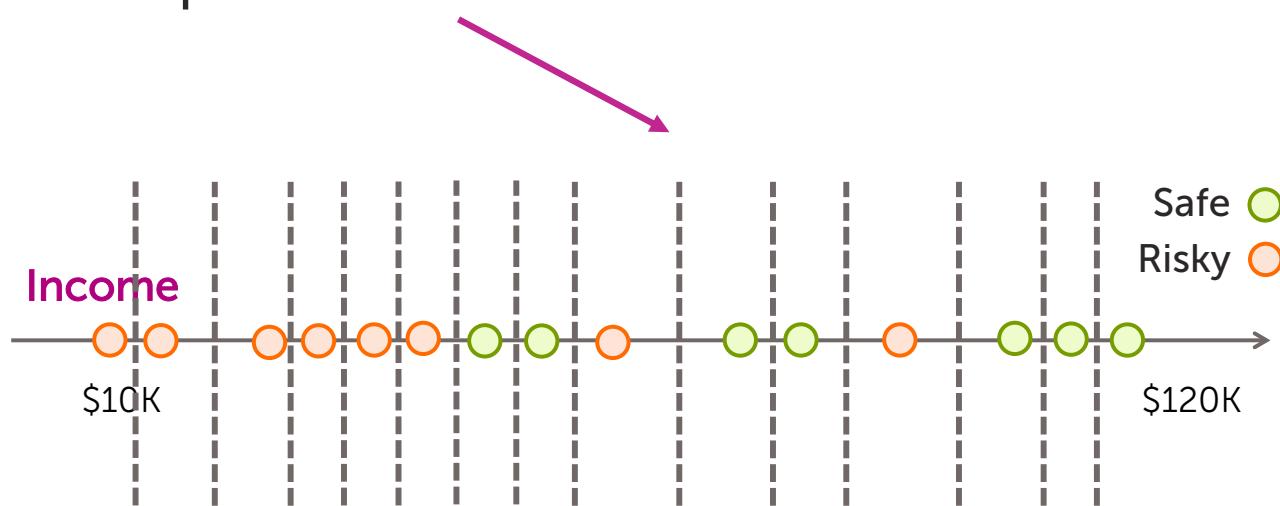
Consider a threshold between points

Same classification error for any threshold split between v_A and v_B



Only need to consider mid-points

Finite number of
splits to consider



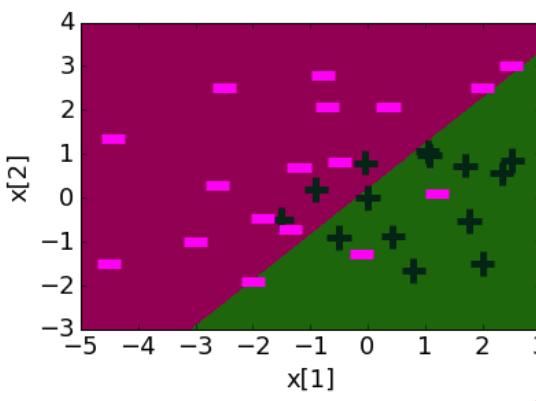
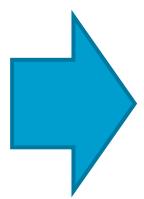
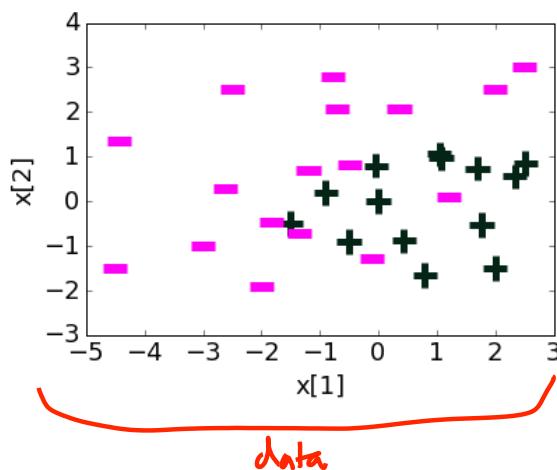
Threshold split selection algorithm

- Step 1: Sort the values of a feature $h_j(\mathbf{x})$:
Let $\{v_1, v_2, v_3, \dots, v_N\}$ denote sorted values
- Step 2:
 - For $i = 1 \dots N-1$
 - Consider split $t_i = (v_i + v_{i+1}) / 2$
 - Compute classification error for threshold
 $h_j(\mathbf{x}) \geq t_i$
 - Choose the t^* with the lowest classification error

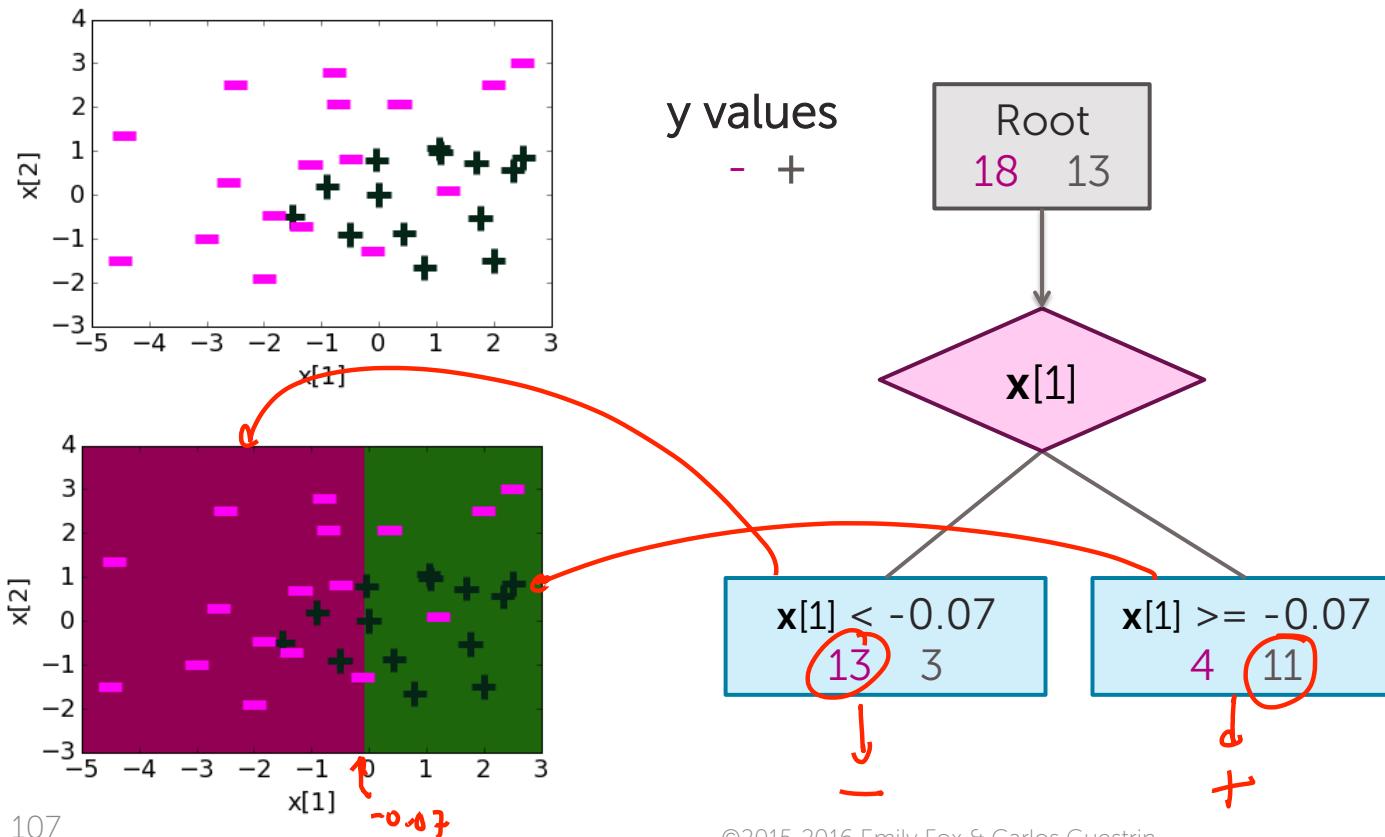
Decision trees vs logistic regression: *Example*

Logistic regression

Feature	Value	Weight Learned
$h_0(\mathbf{x})$	1	0.22
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	1.12
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	-1.07



Depth 1: Split on $x[1]$

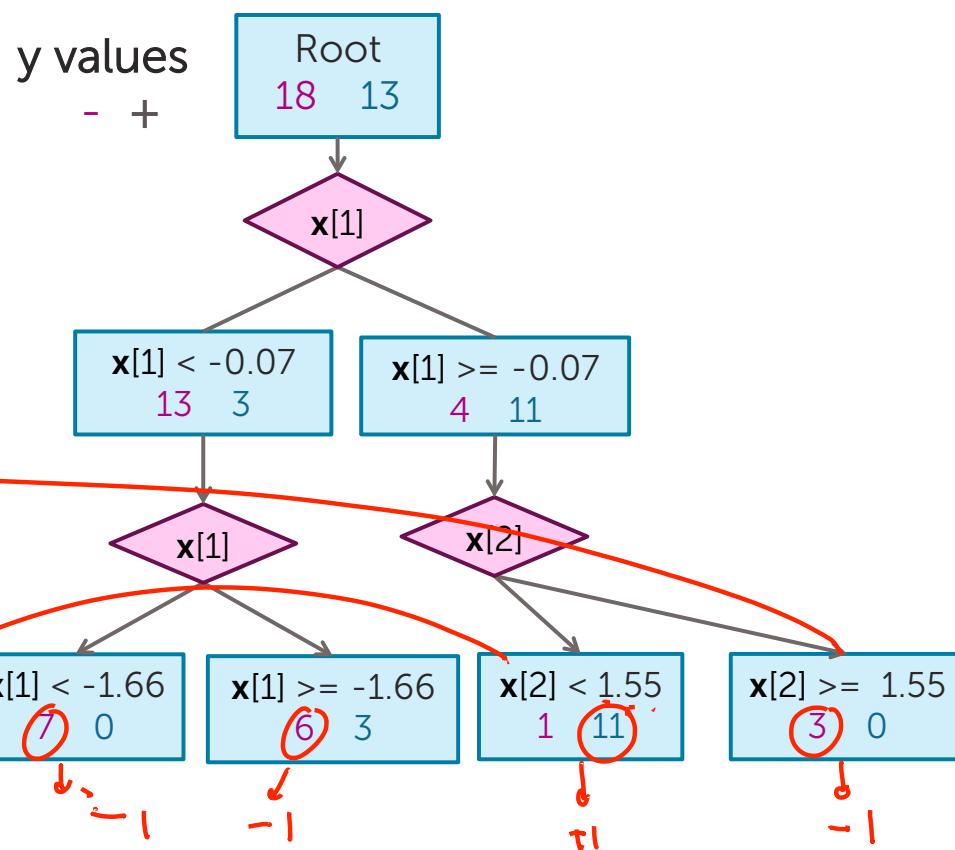
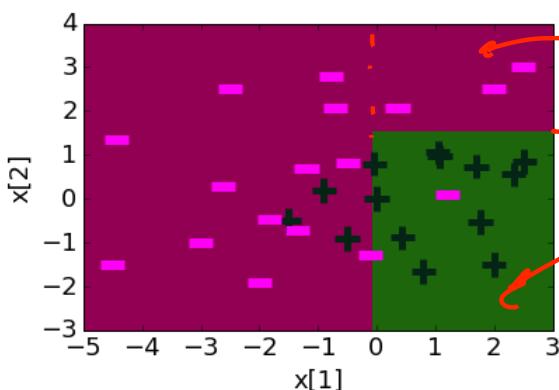
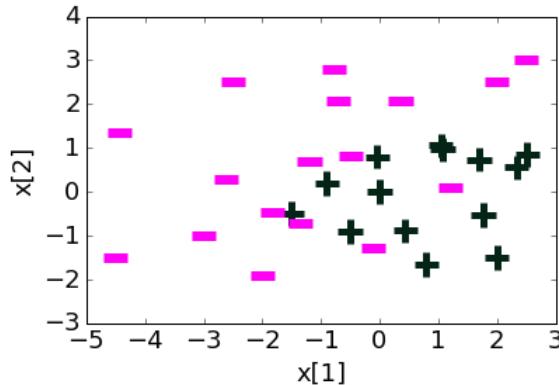


107

©2015-2016 Emily Fox & Carlos Guestrin

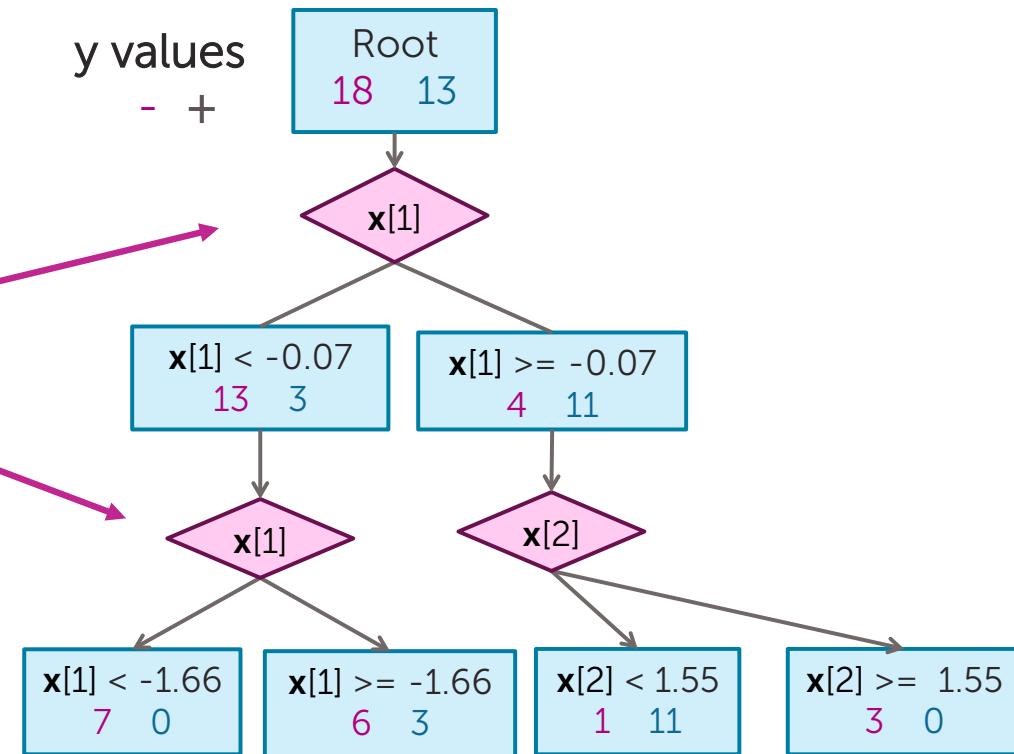
Machine Learning Specialization

Depth 2

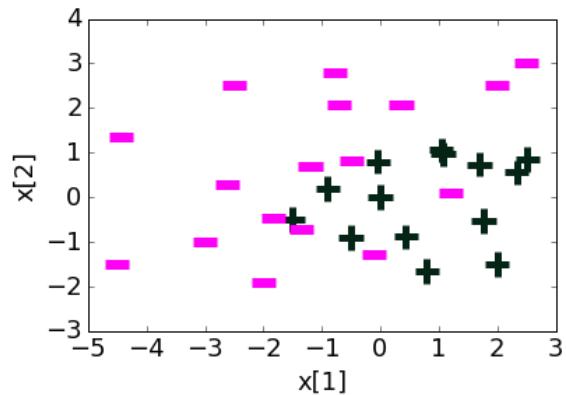


Threshold split caveat

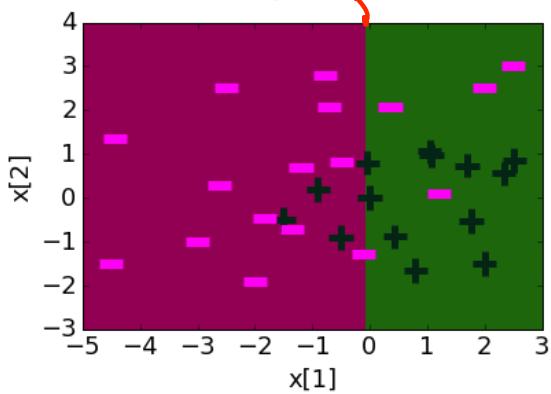
For threshold splits,
same feature can be
used multiple times



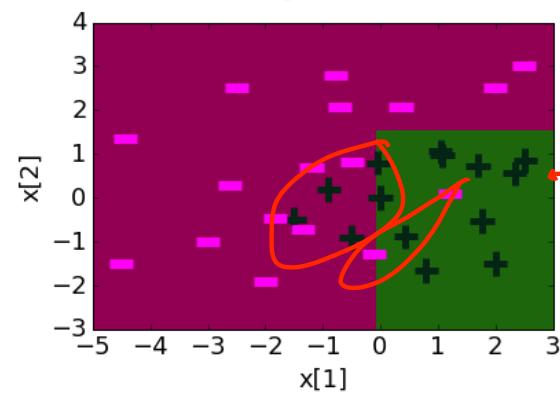
Decision boundaries



Depth 1



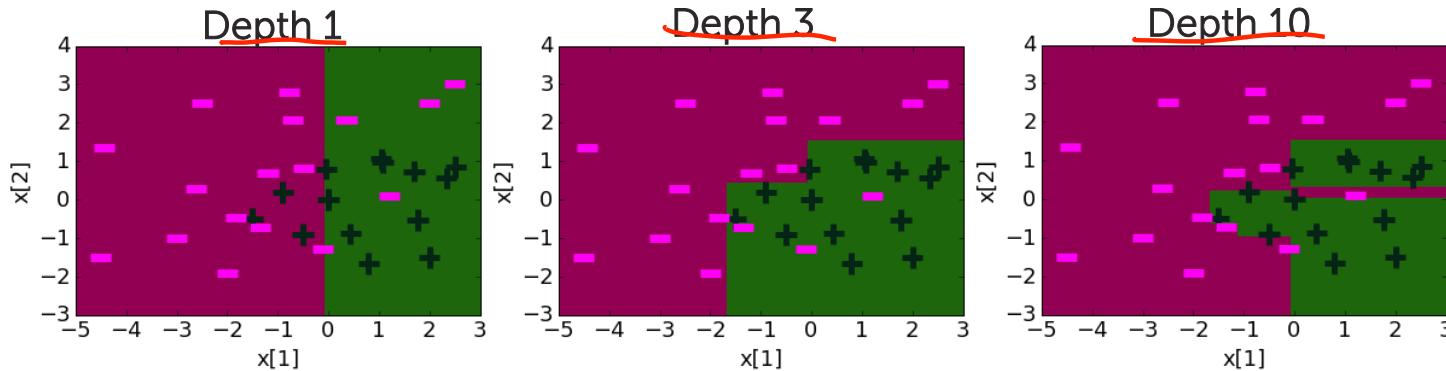
Depth 2



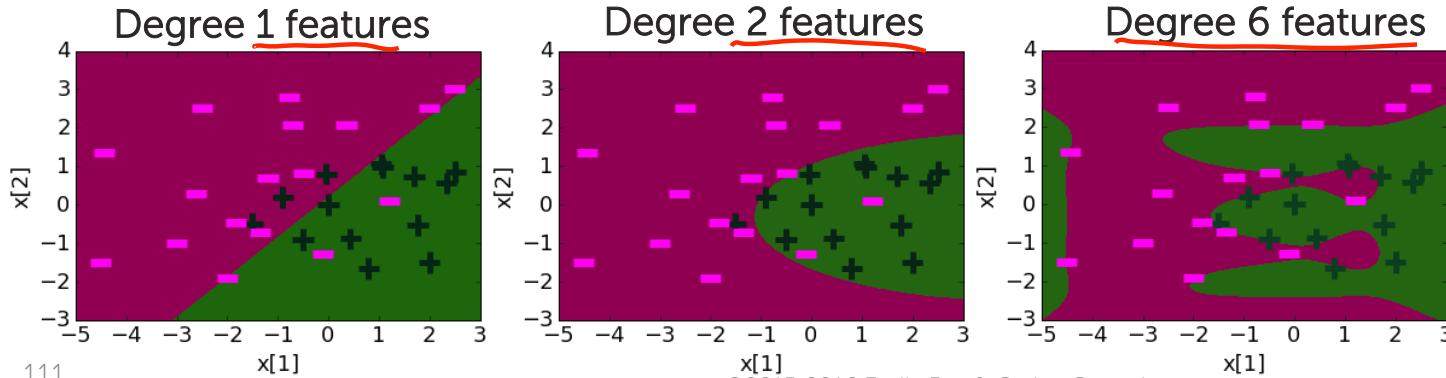
Depth 10

Comparing decision boundaries

Decision Tree



Logistic Regression



Summary of decision trees



What you can do now

- Define a decision tree classifier
- Interpret the output of a decision trees
- Learn a decision tree classifier using greedy algorithm
- Traverse a decision tree to make predictions
 - Majority class predictions
 - Probability predictions
 - Multiclass classification

Thank you to Dr. Krishna Sridhar



Dr. Krishna Sridhar
Staff Data Scientist, Dato, Inc.