



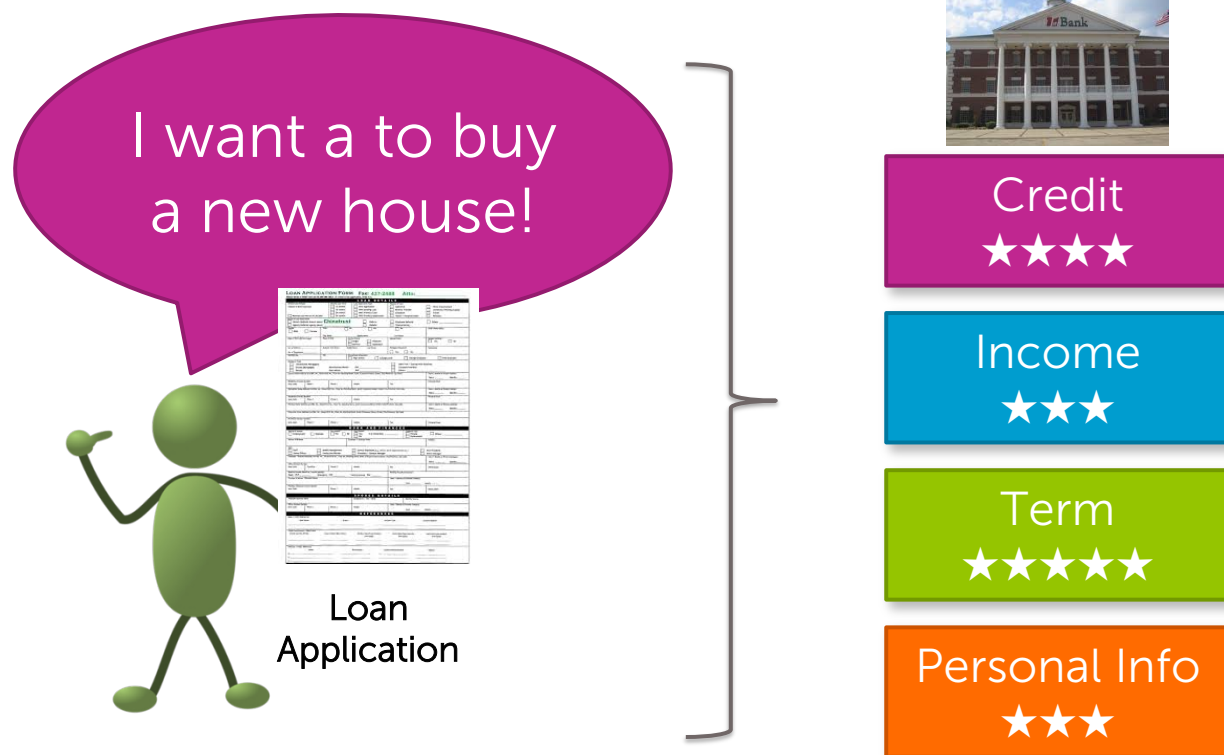
# Decision Trees

Emily Fox & Carlos Guestrin  
Machine Learning Specialization  
University of Washington



# Predicting potential loan defaults

# What makes a loan risky?

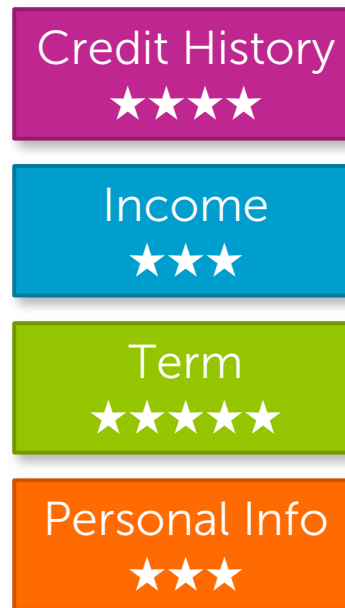


# Credit history explained

Did I pay previous  
loans on time?



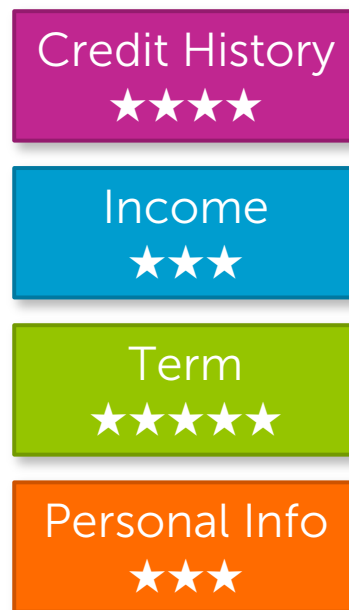
**Example:** excellent,  
good, or fair



# Income

What's my income?

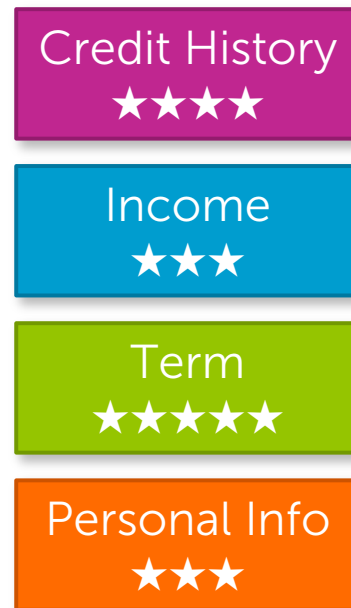
**Example:**  
\$80K per year



# Loan terms

How soon do I need to pay the loan?

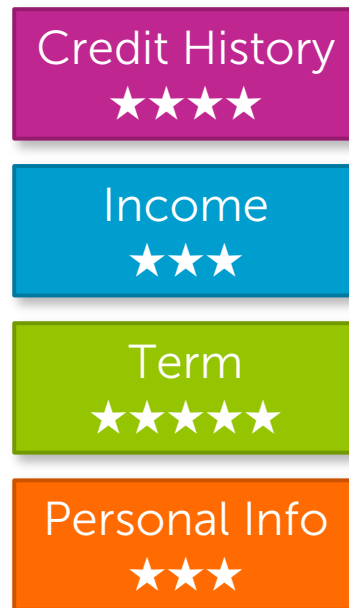
**Example:** 3 years,  
5 years,...



# Personal information

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

**Example:** Home loan  
for a married couple



# Intelligent application

Loan  
Applications

A pink loan application form with various fields and checkboxes.A blue loan application form with various fields and checkboxes.A green loan application form with various fields and checkboxes.

Intelligent loan application  
review system

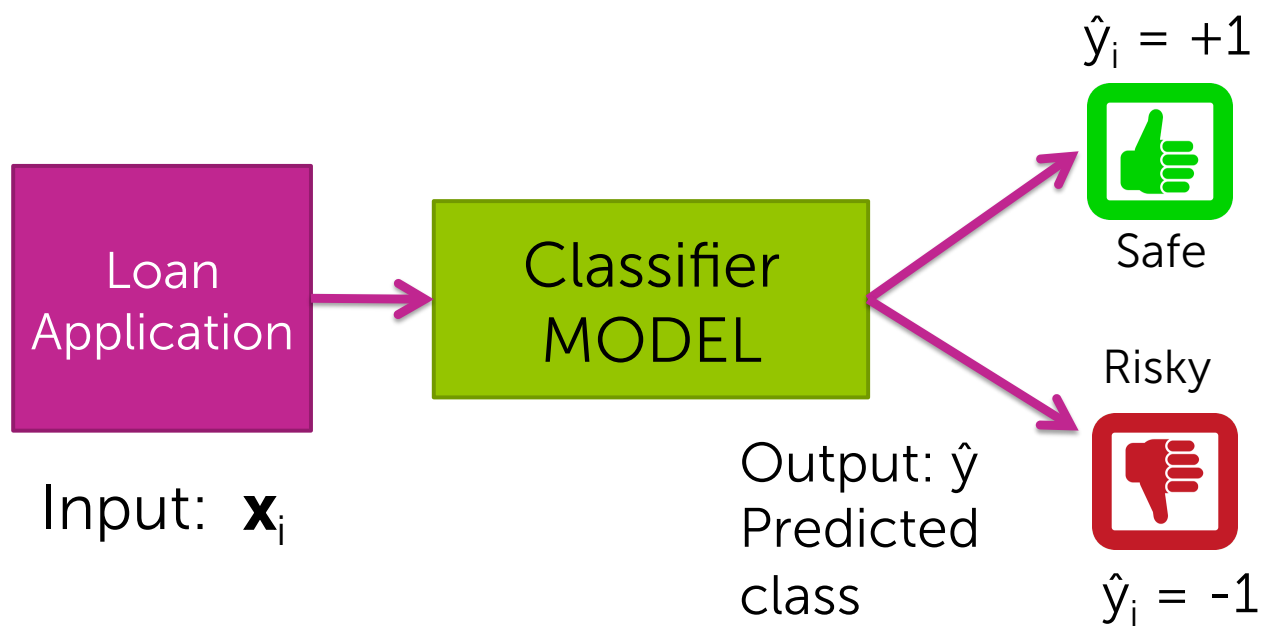
Safe  
✓

Risky  
✗

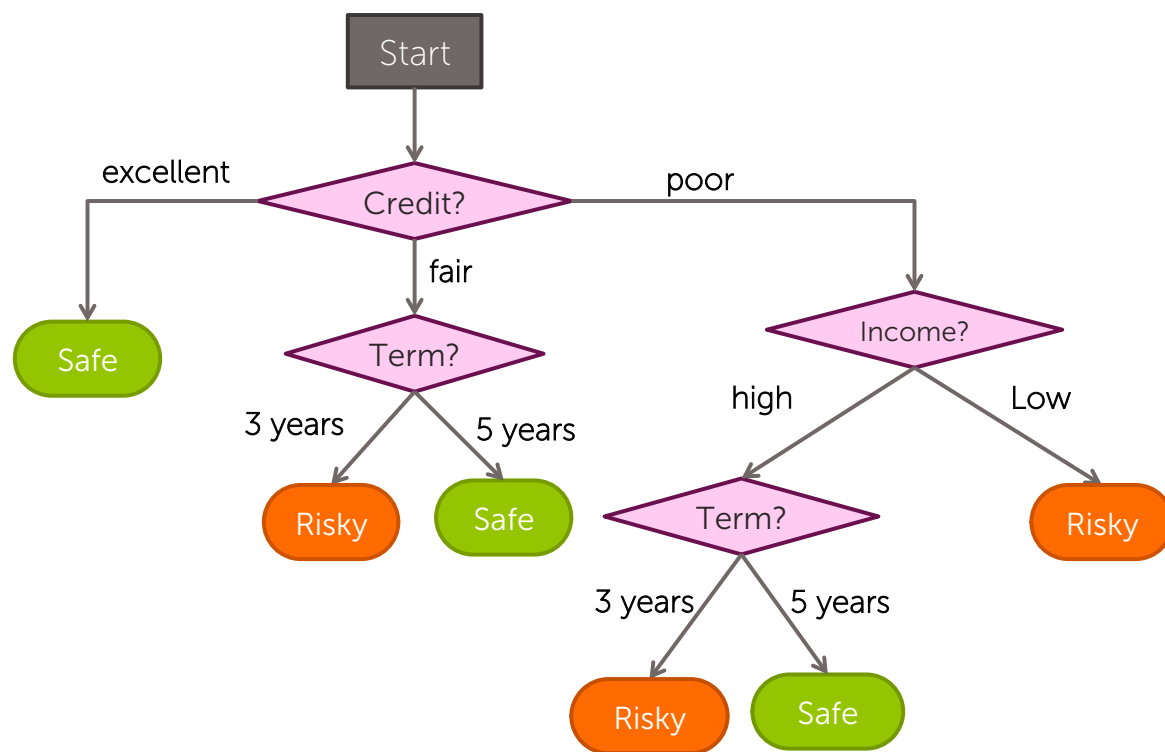
Risky  
✗



# Classifier review

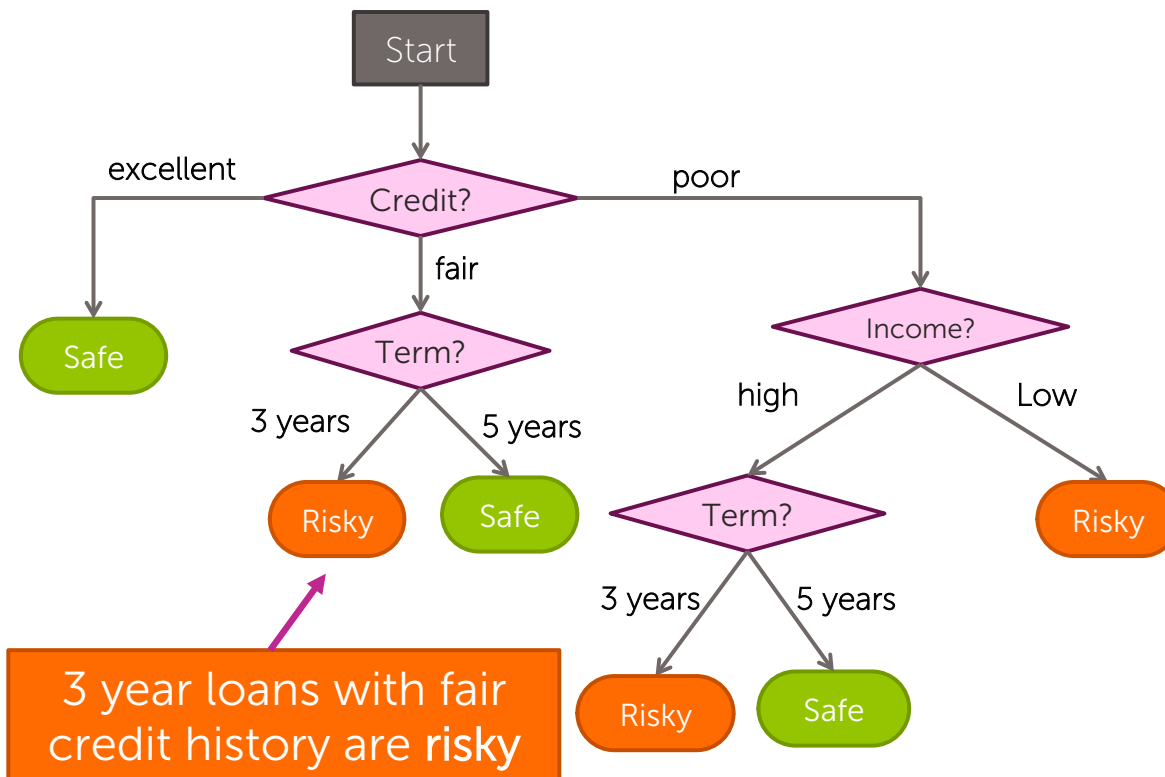


# This module ... decision trees

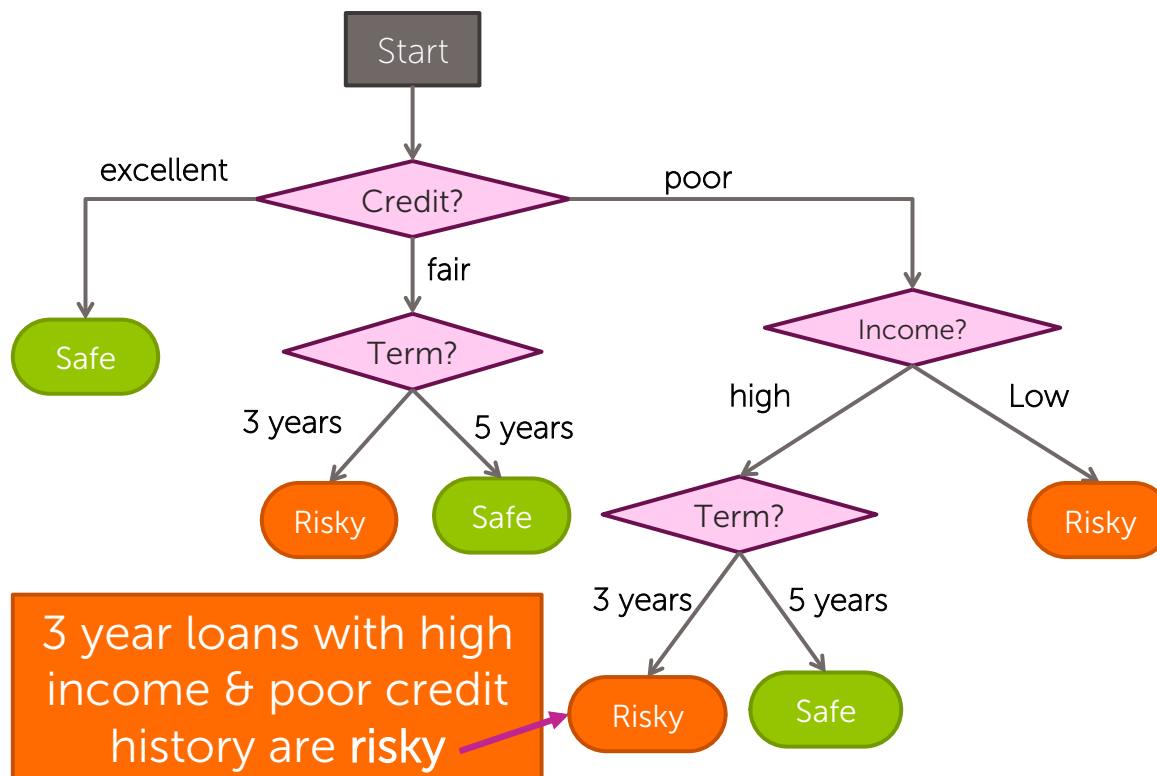


# Decision trees: *Intuition*

# What does a decision tree represent?

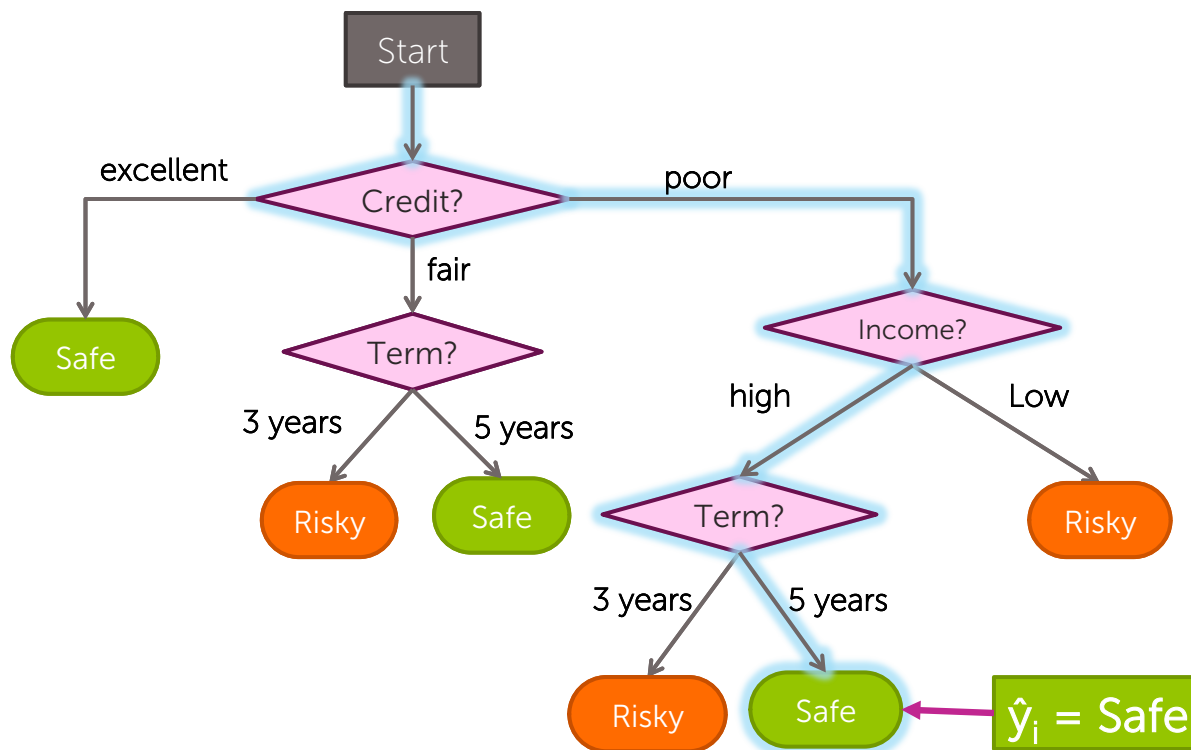


# What does a decision tree represent?

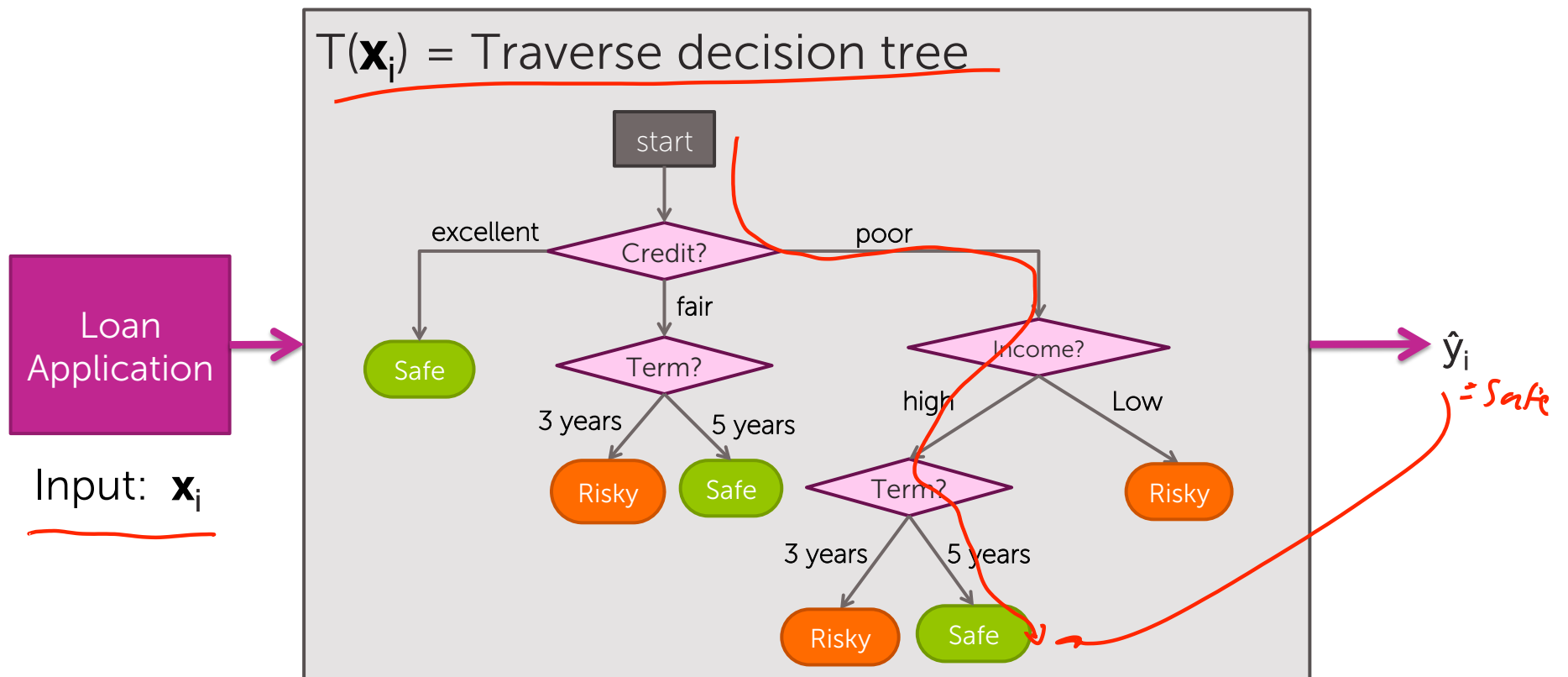


# Scoring a loan application

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

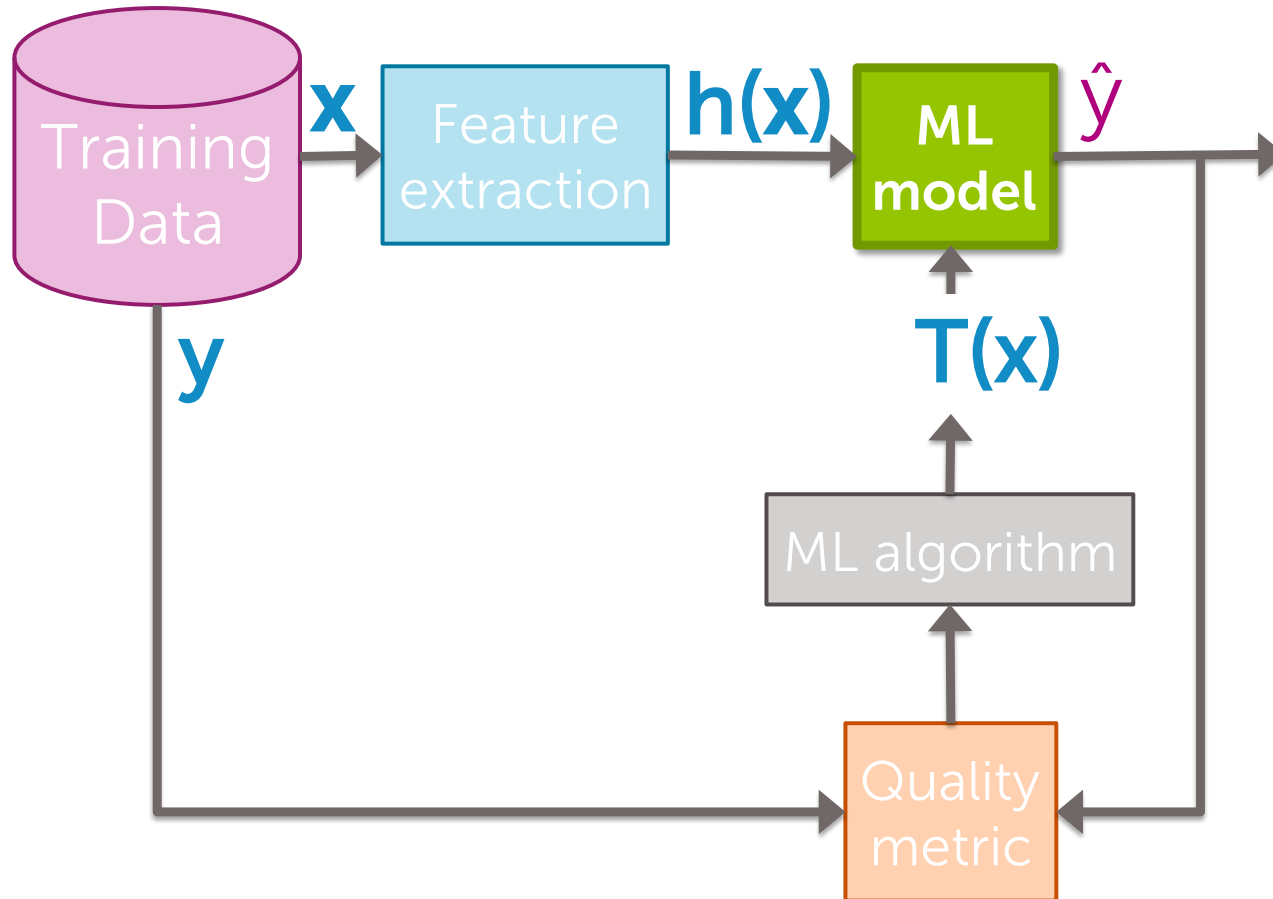


# Decision tree model



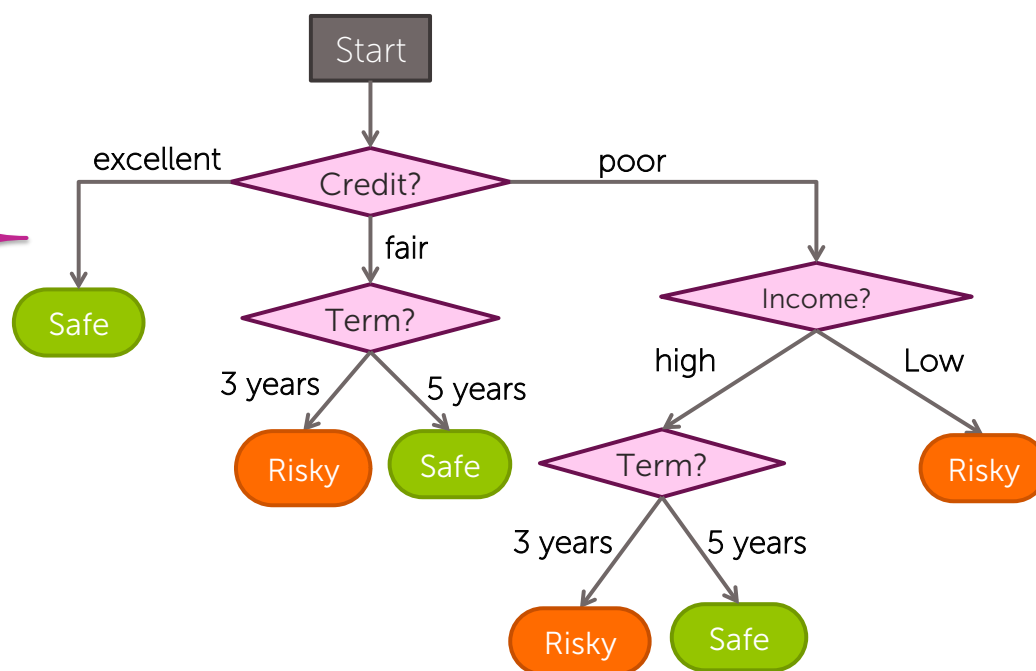
# Decision tree learning task





# Learn decision tree from data?

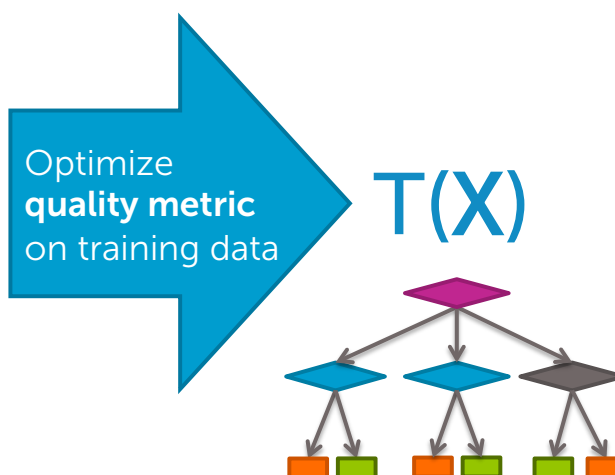
$h_1(x)$	$h_2(x)$	$h_3(x)$	Loan Status
Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



# 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

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

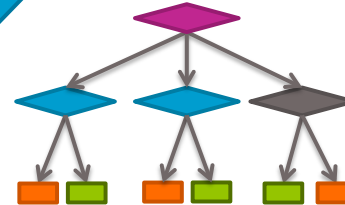
- Best possible value : 0.0
- Worst possible value: 1.0

# 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

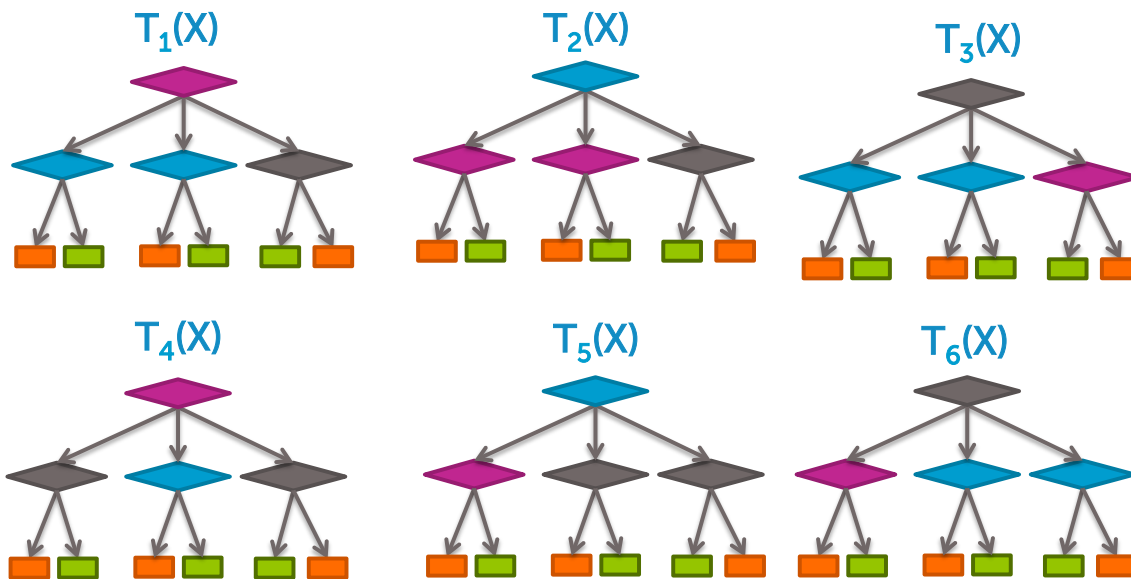
Minimize  
**classification error**  
on training data

$T(X)$



# How do we find the best tree?

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

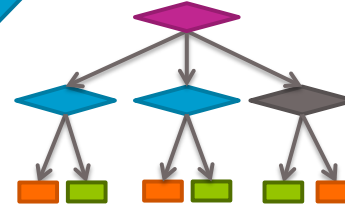


# Simple (greedy) algorithm finds “good” tree

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

Approximately  
minimize  
**classification error**  
on training data

$T(X)$

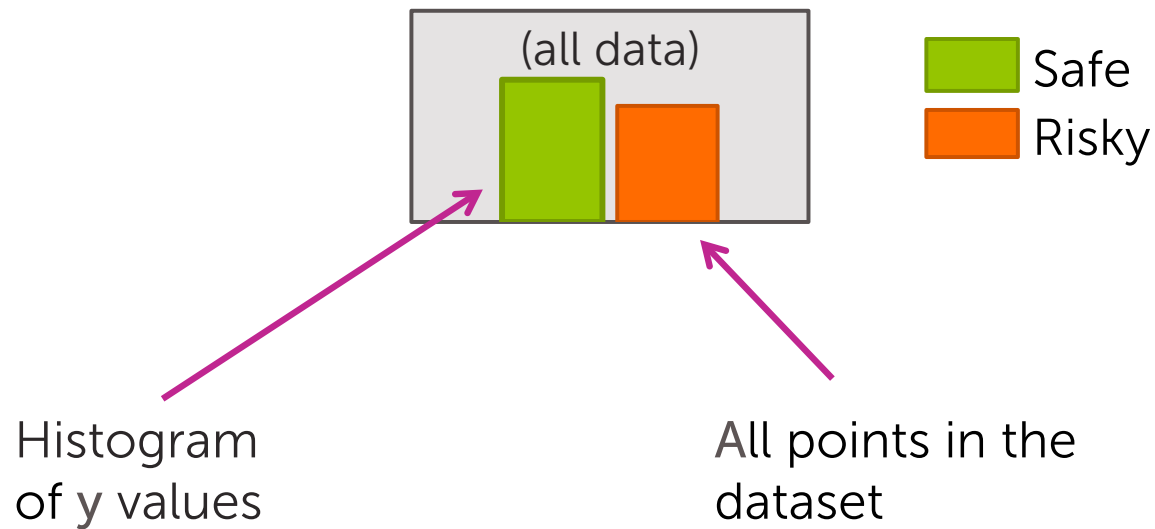


# Greedy decision tree learning:

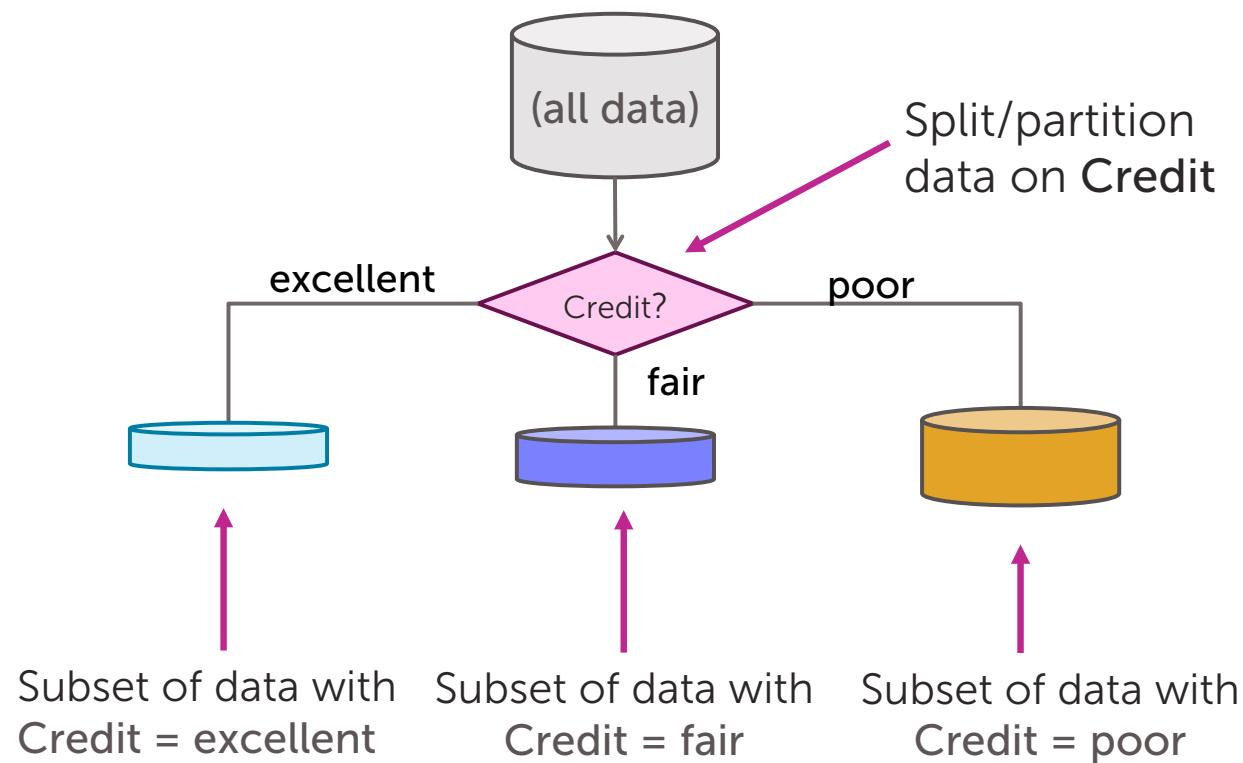
## *Algorithm outline*



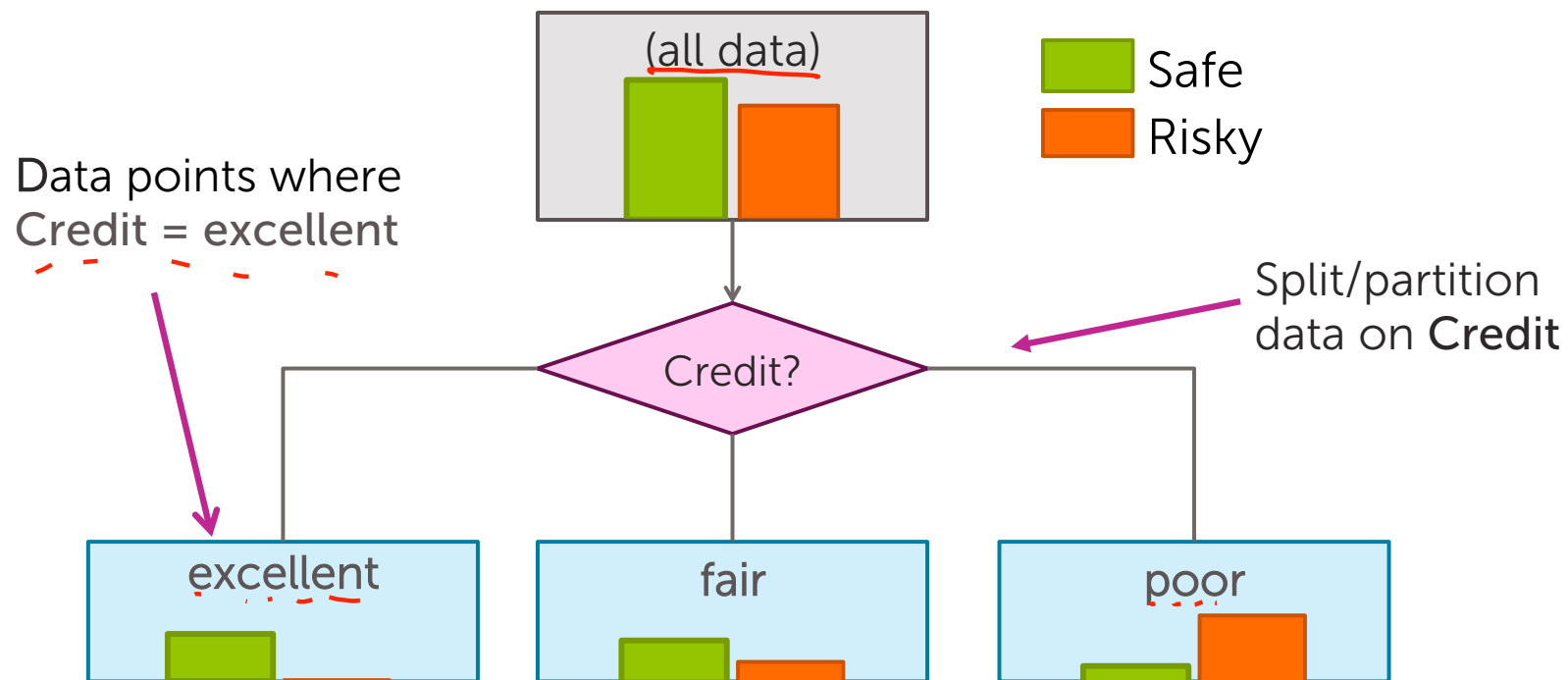
# Step 1: Start with an empty tree



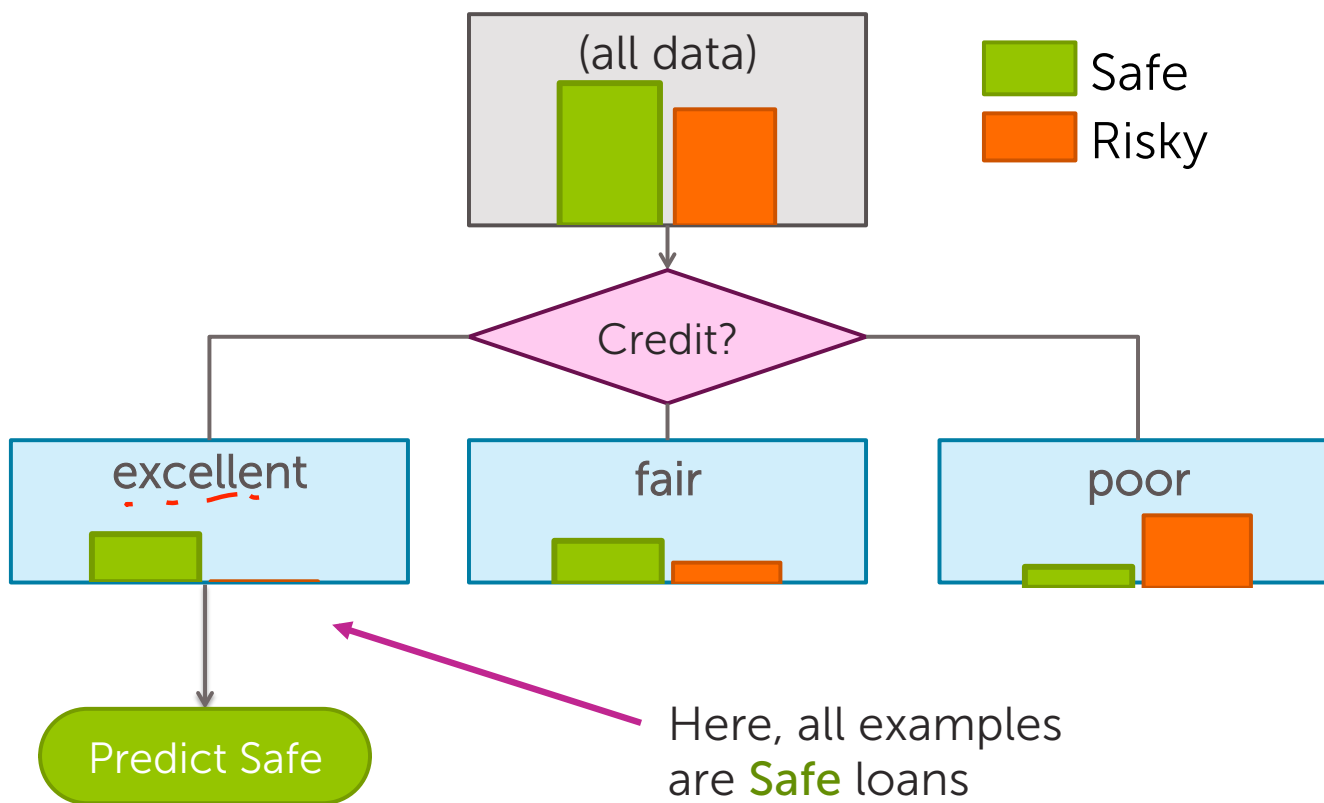
## Step 2: Split on a feature



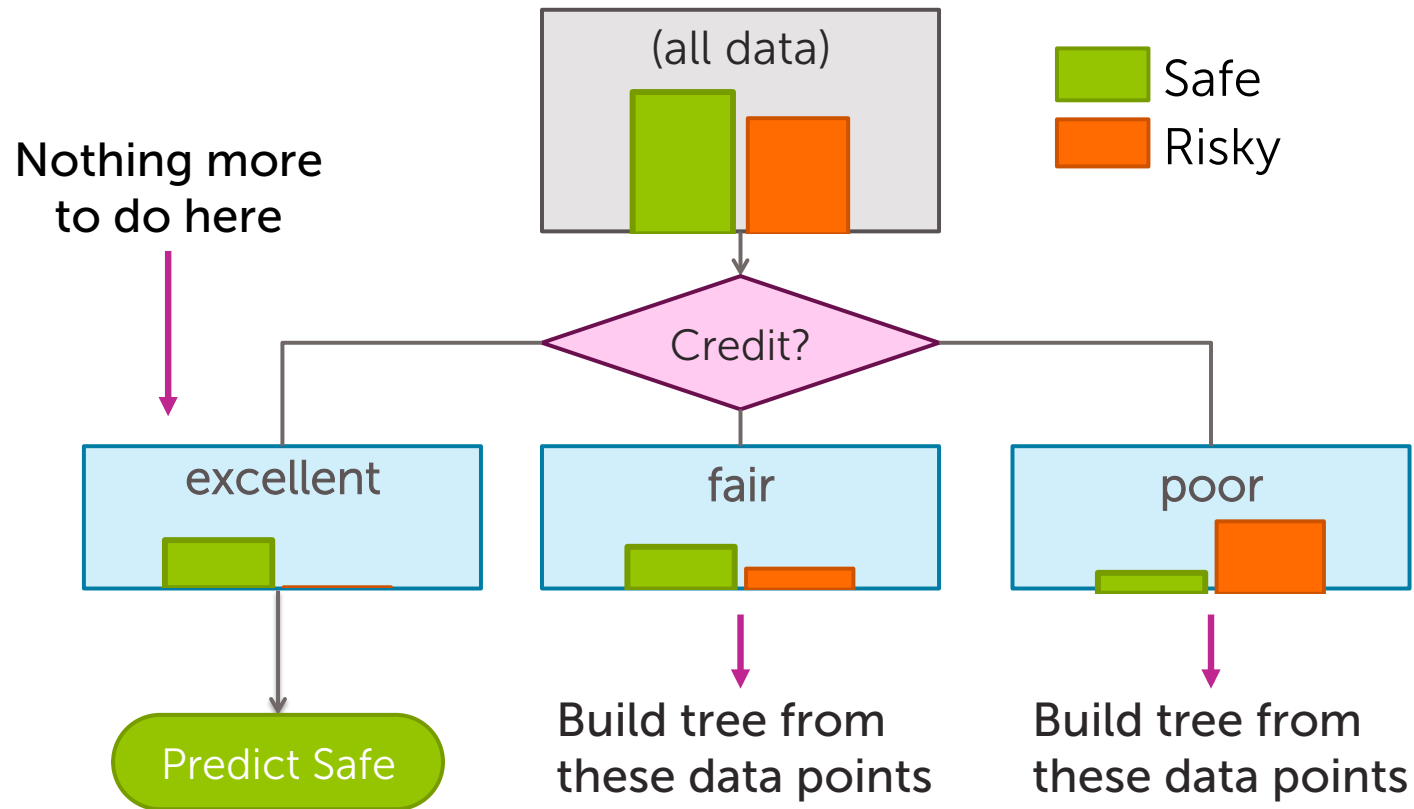
# Feature split explained



## Step 3: Making predictions



## Step 4: Recursion



# 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

**Problem 1:** Feature split selection

**Problem 2:** Stopping condition

Recursion



Feature split learning

=

Decision stump learning



# Start with the data

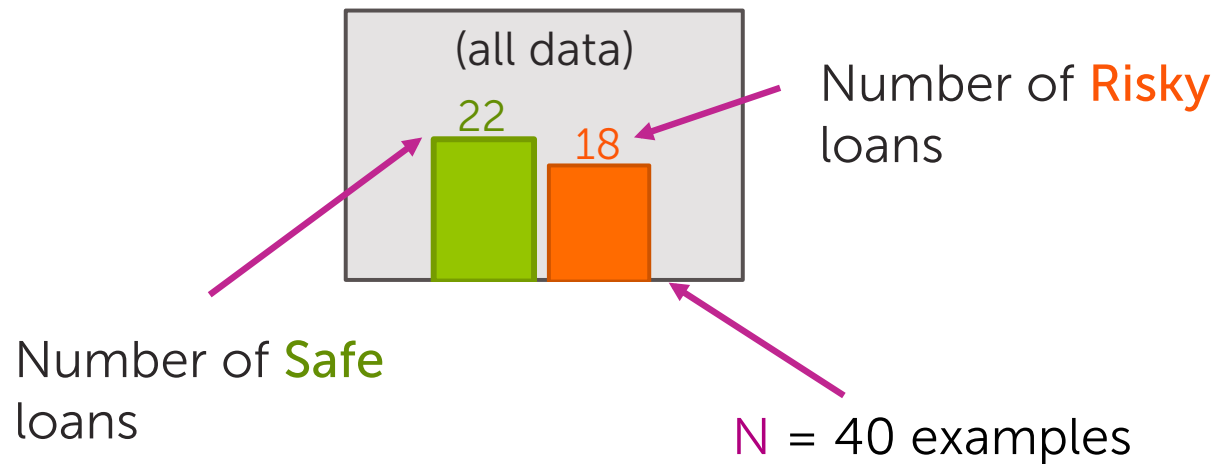
Assume  $N = 40$ , 3 features

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

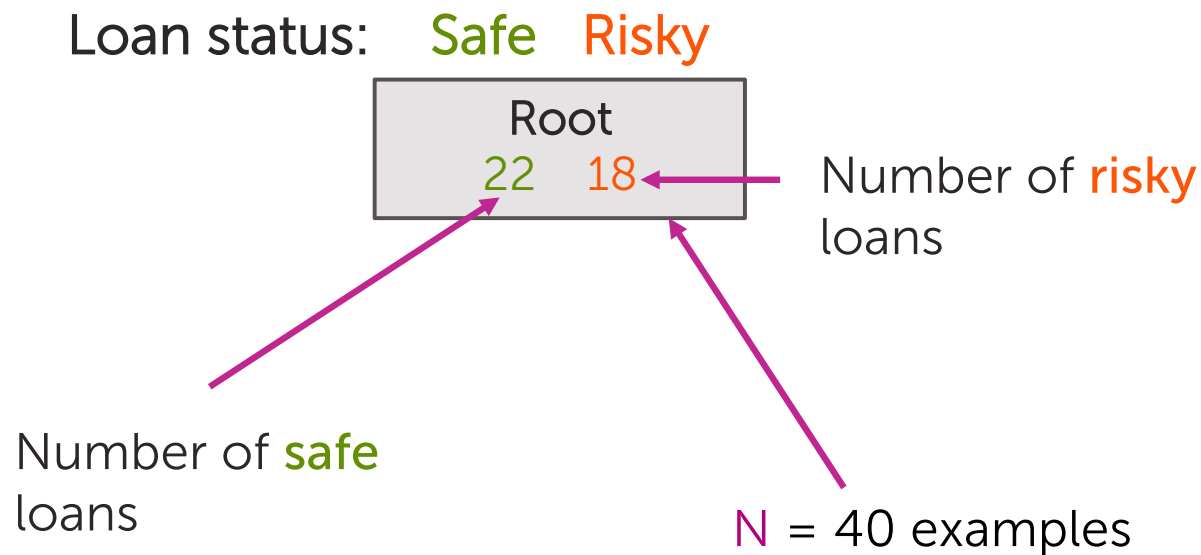


# Start with all the data

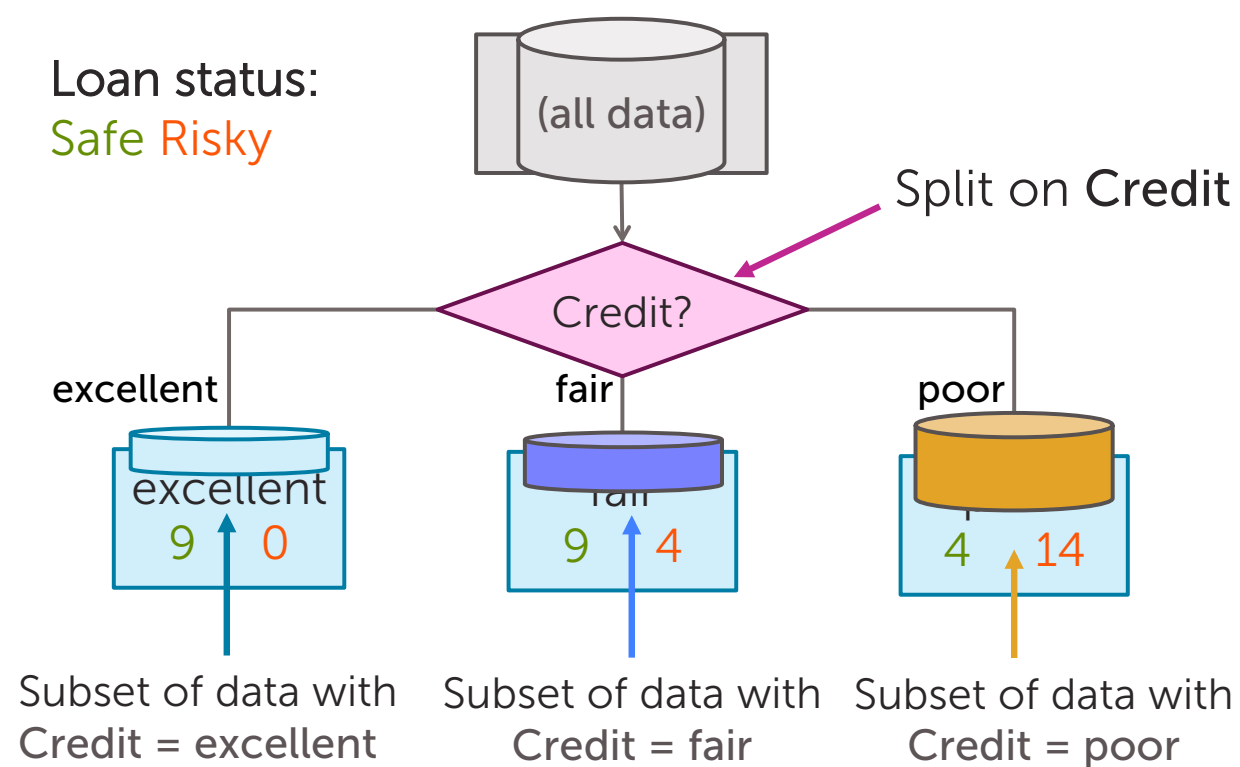
Loan status: **Safe** **Risky**



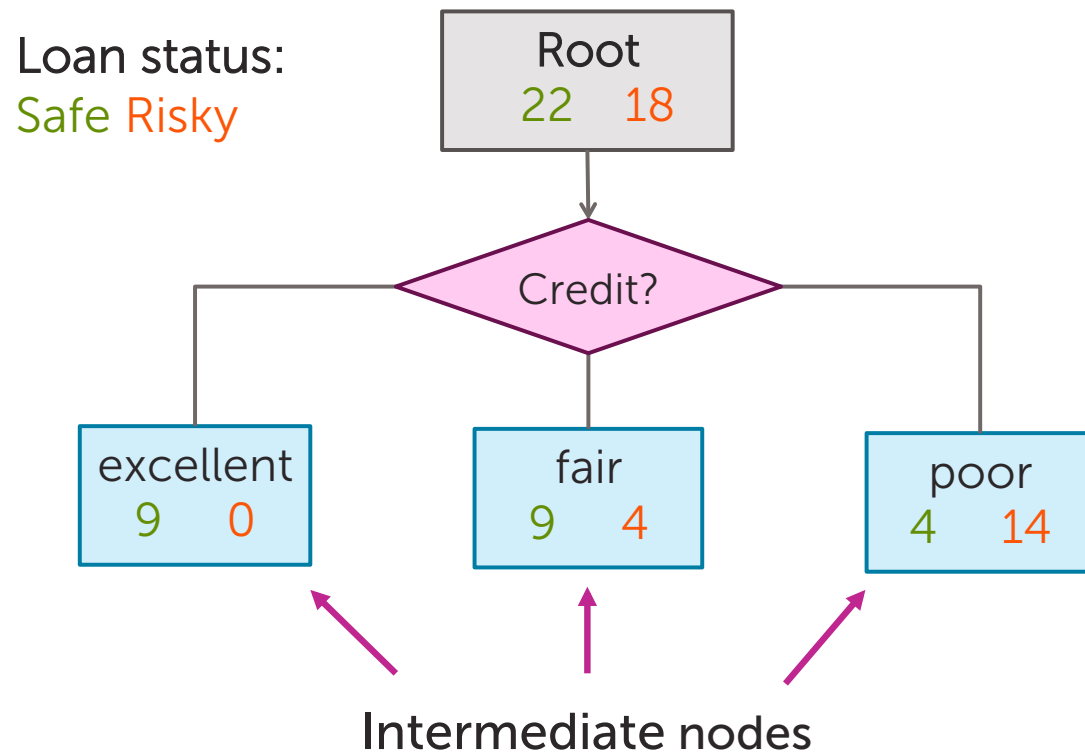
# Compact visual notation: Root node



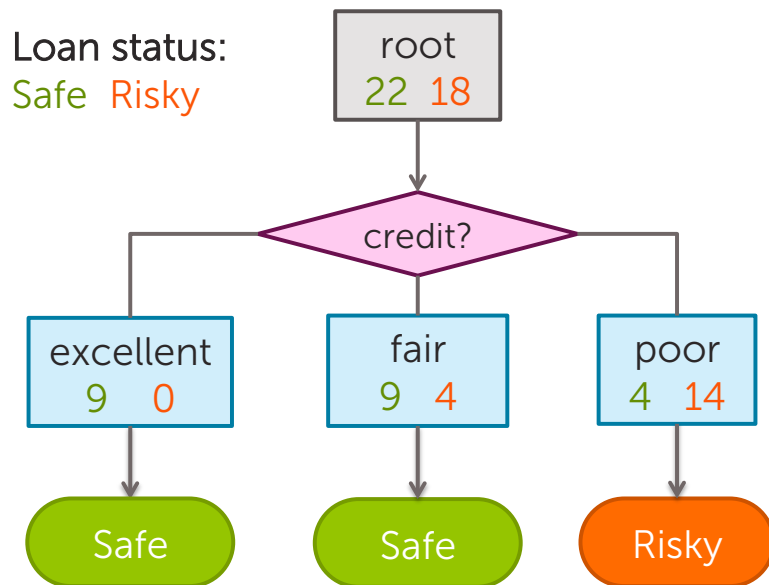
# Decision stump: Single level tree



# Visual Notation: Intermediate nodes



# Making predictions with a decision stump

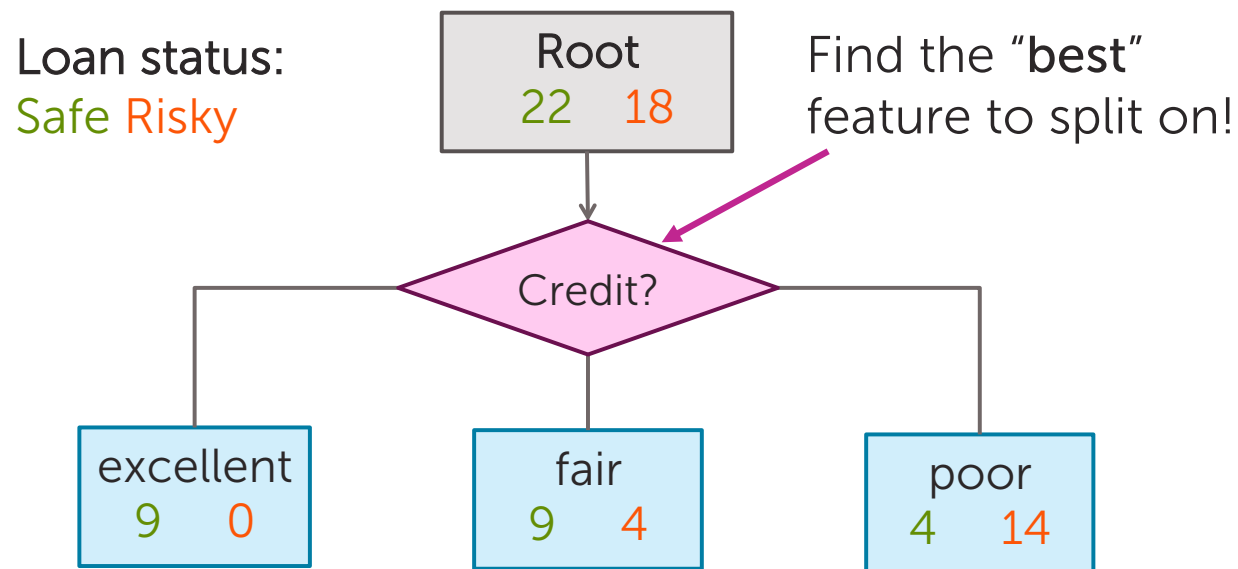


For each intermediate node,  
set  $\hat{y}$  = majority value

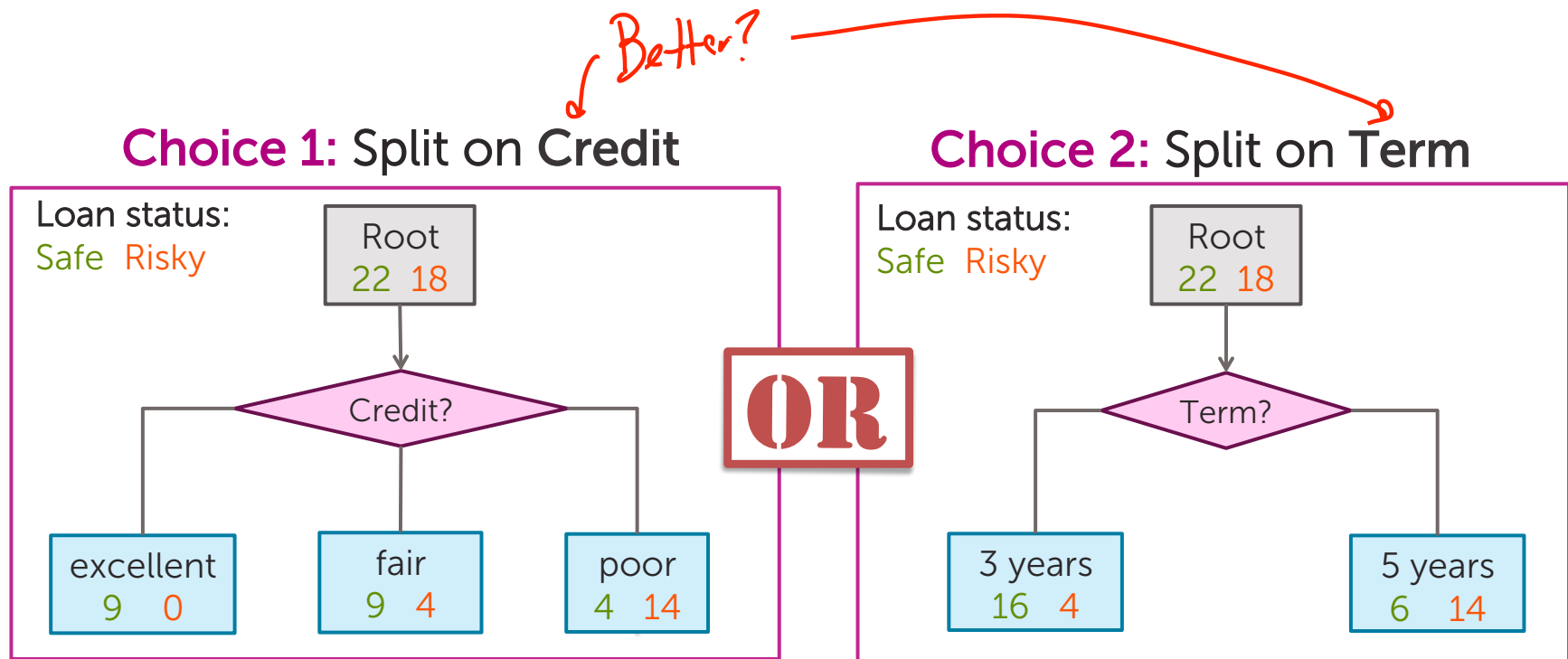


## Selecting best feature to split on

# How do we learn a decision stump?

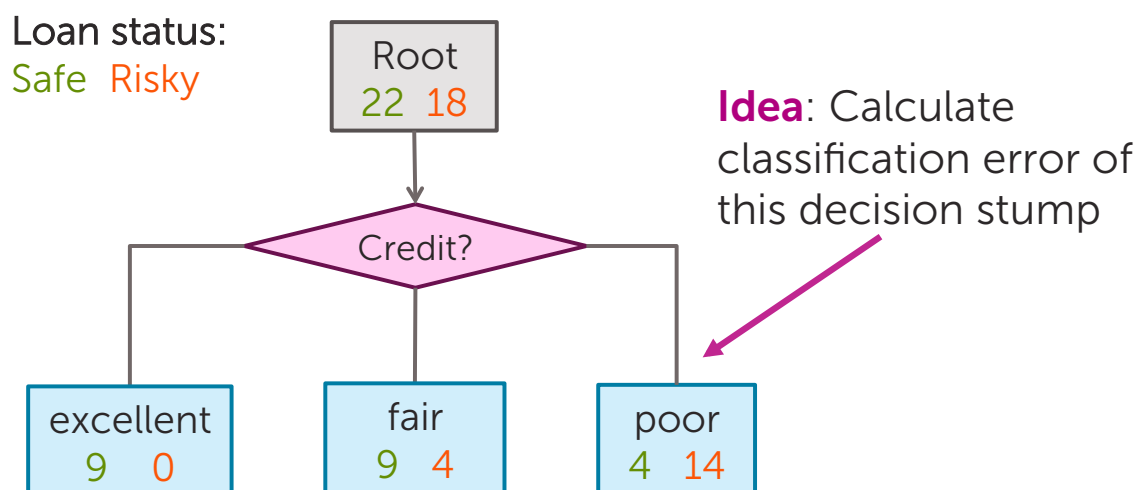


# How do we select the best feature?





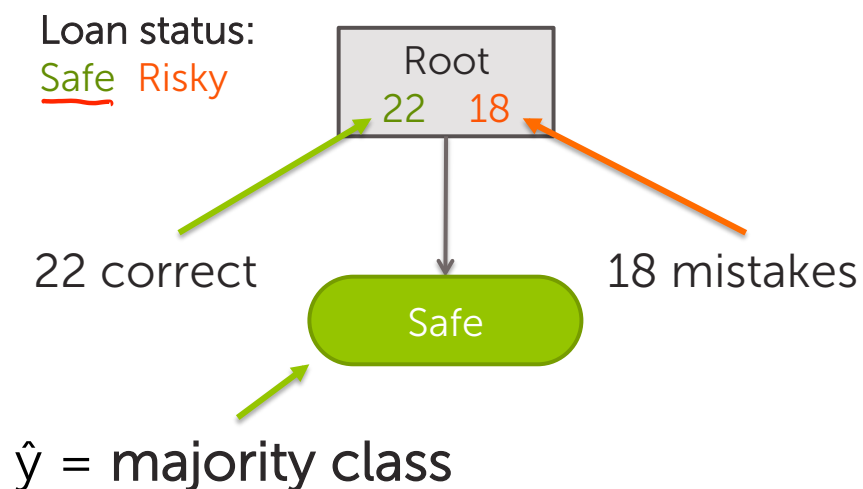
# How do we measure effectiveness of a split?



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

# 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

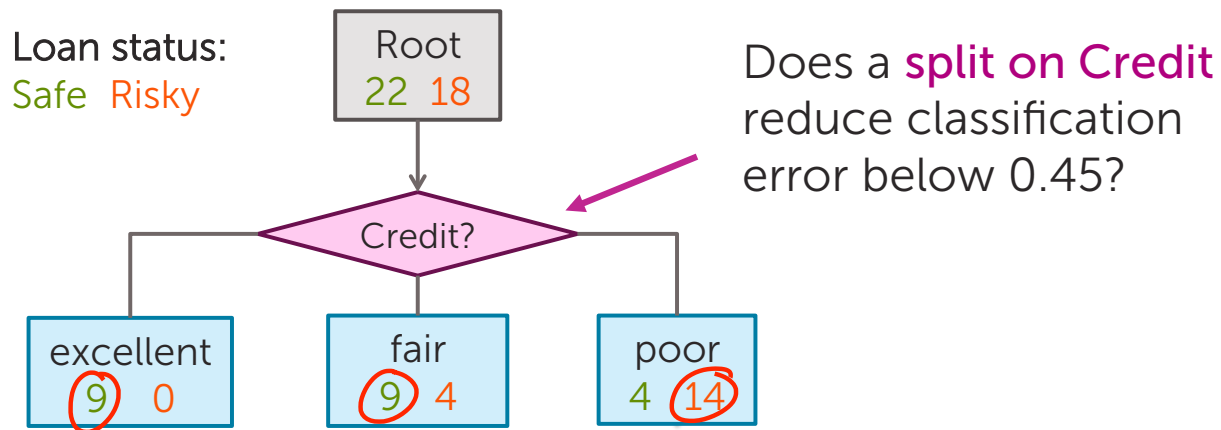


$$\text{Error} = \frac{18}{22+18}$$
$$= 0.45$$

Tree	Classification error
(root)	0.45

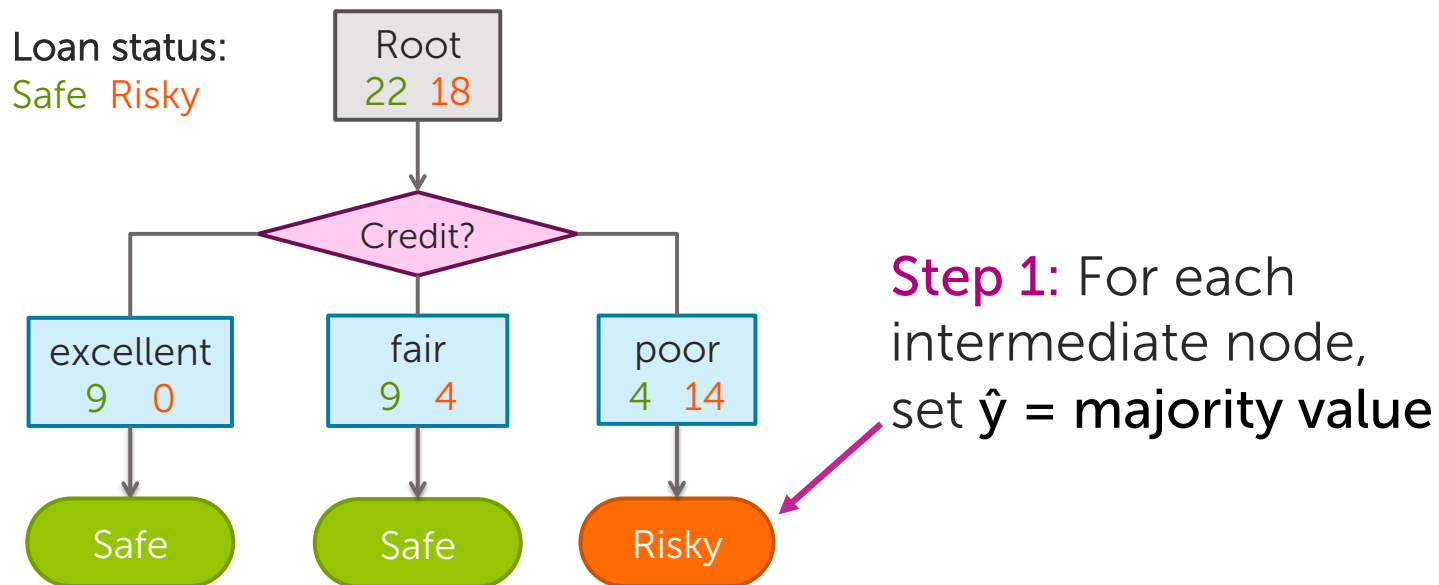
# Choice 1: Split on credit history?

## Choice 1: Split on Credit



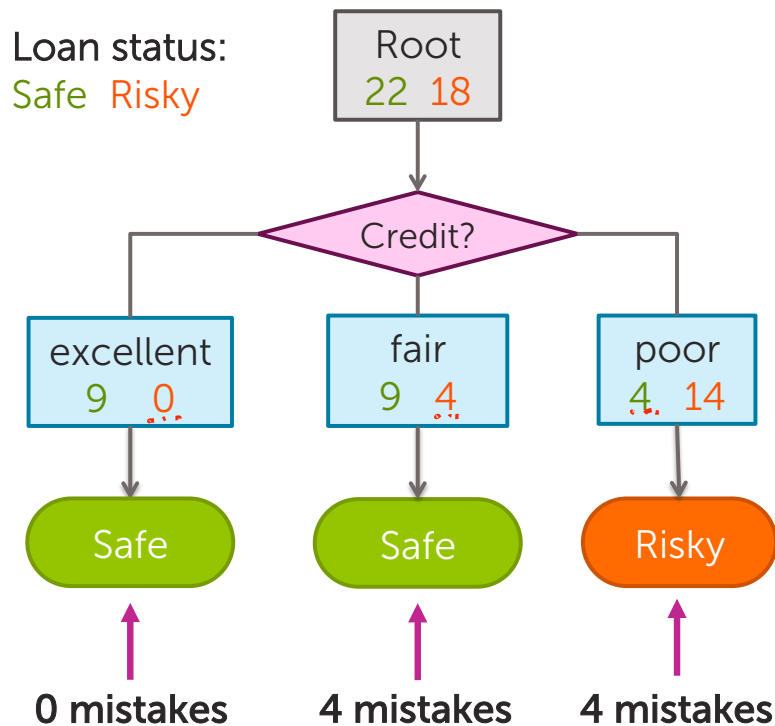
# How good is the split on Credit?

## Choice 1: Split on Credit



# Split on Credit: Classification error

## Choice 1: Split on Credit

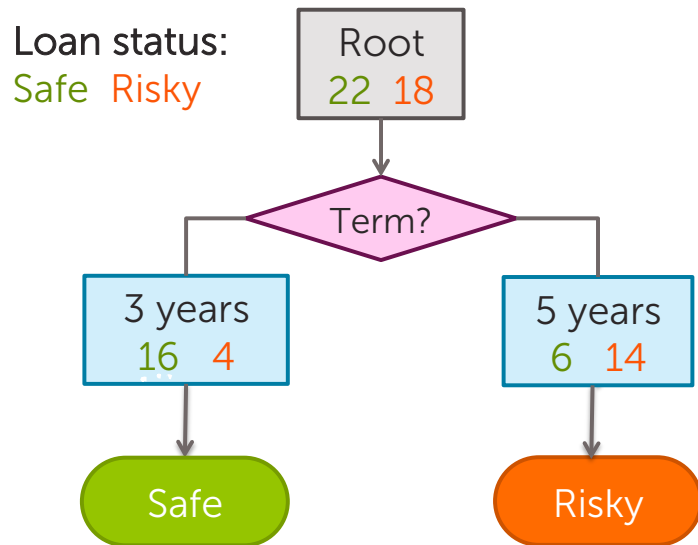


$$\text{Error} = \frac{4 + 4}{40} = 0.20$$

Tree	Classification error
(root)	0.45
Split on credit	0.2

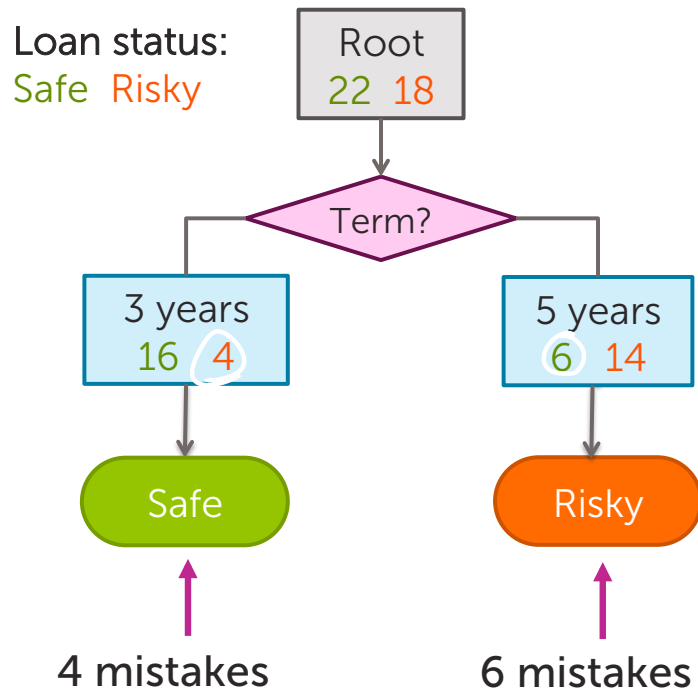
# Choice 2: Split on Term?

## Choice 2: Split on Term



# Evaluating the split on Term

## Choice 2: Split on Term



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

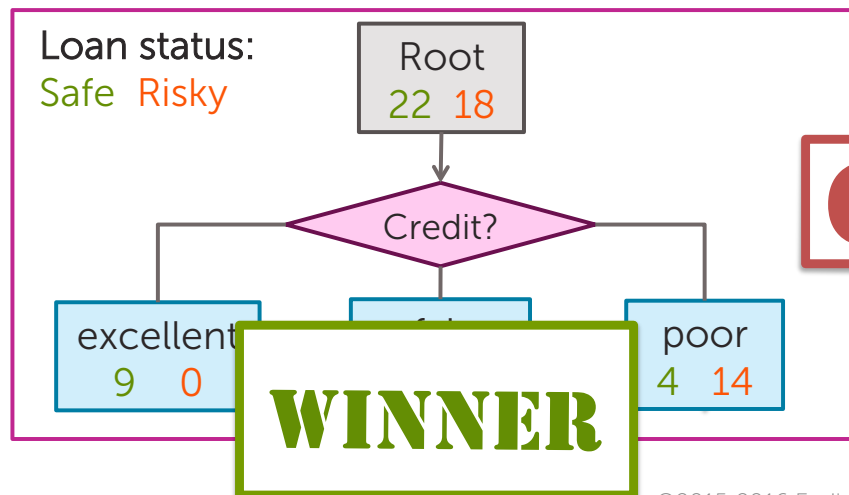
Tree	Classification error
(root)	0.45
Split on credit	0.2
Split on term	0.25

# Choice 1 vs Choice 2

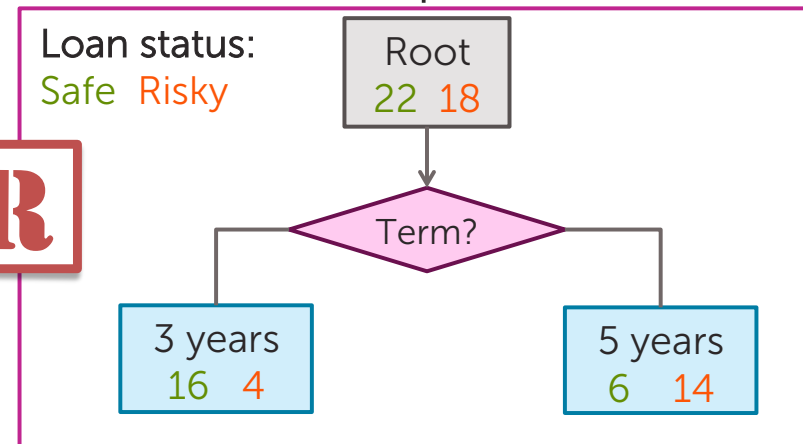
Tree	Classification error
(root)	0.45
split on <u>credit</u>	0.2
split on <b>loan term</b>	0.25

← First split!

## Choice 1: Split on Credit



## Choice 2: Split on Term



**OR**




# Feature split selection algorithm

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

# 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

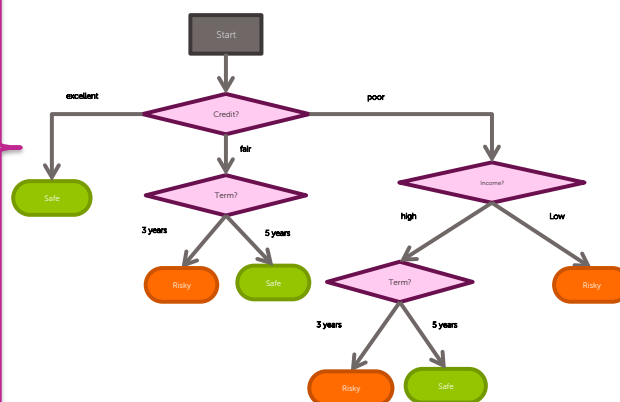


# Decision Tree Learning:

## *Recursion & Stopping conditions*

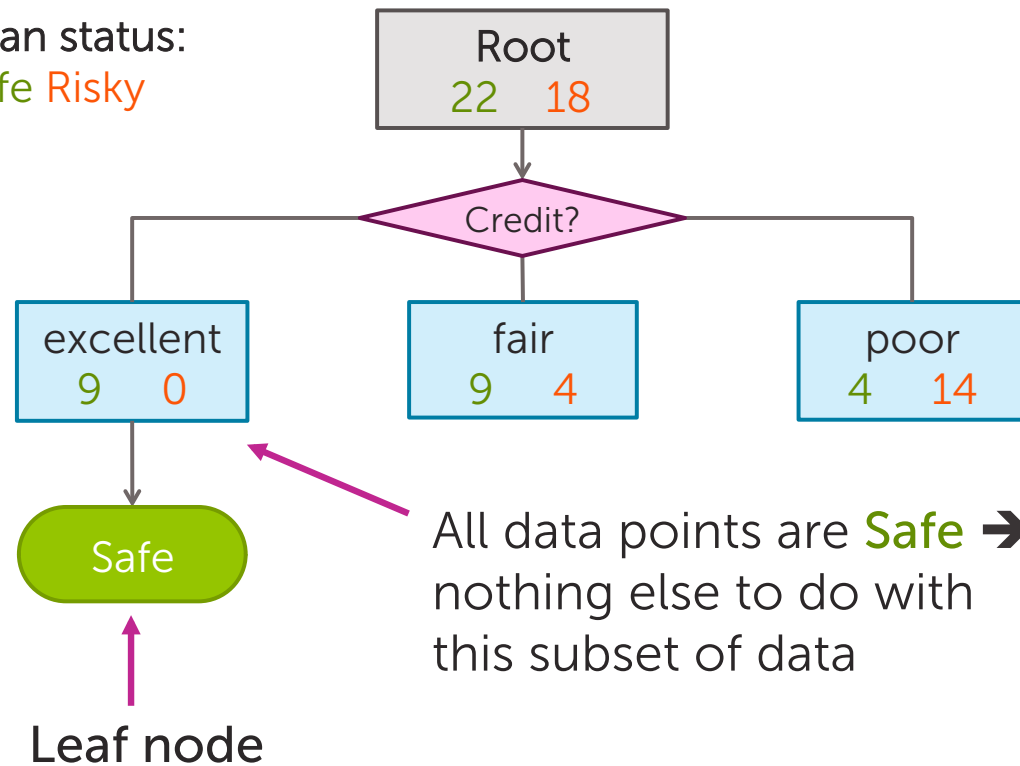
# Learn decision tree from data?

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



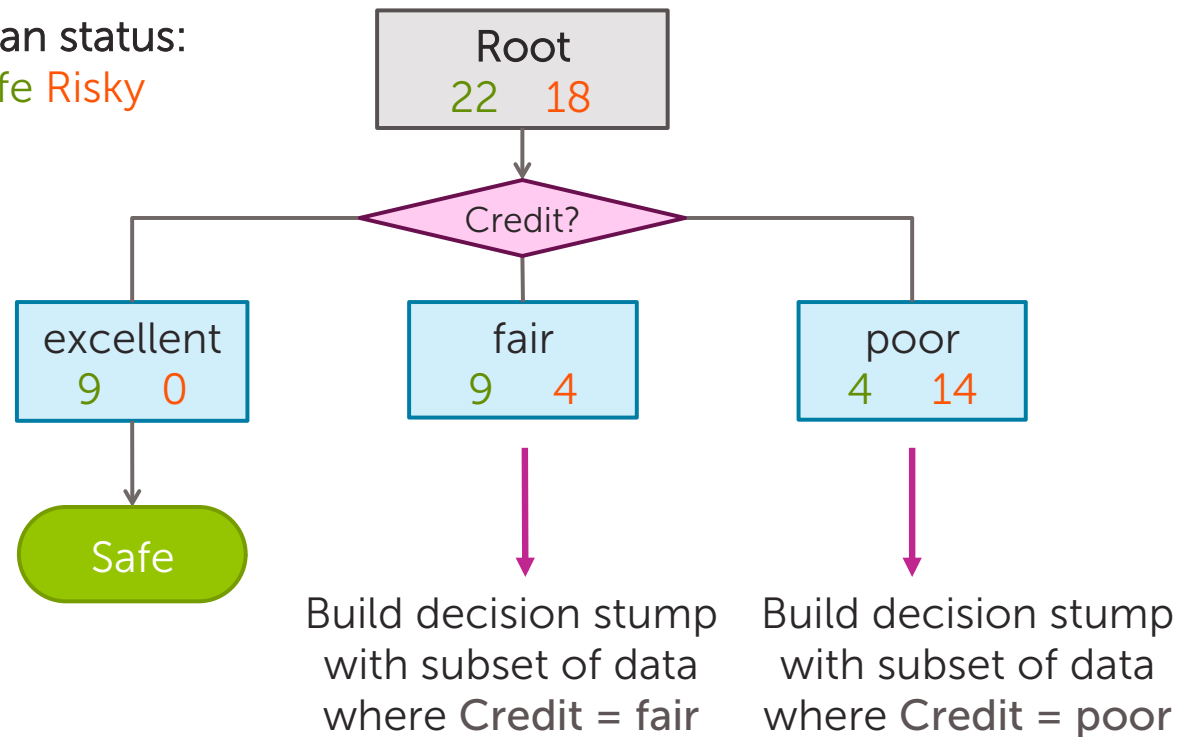
# We've learned a decision stump, what next?

Loan status:  
Safe Risky

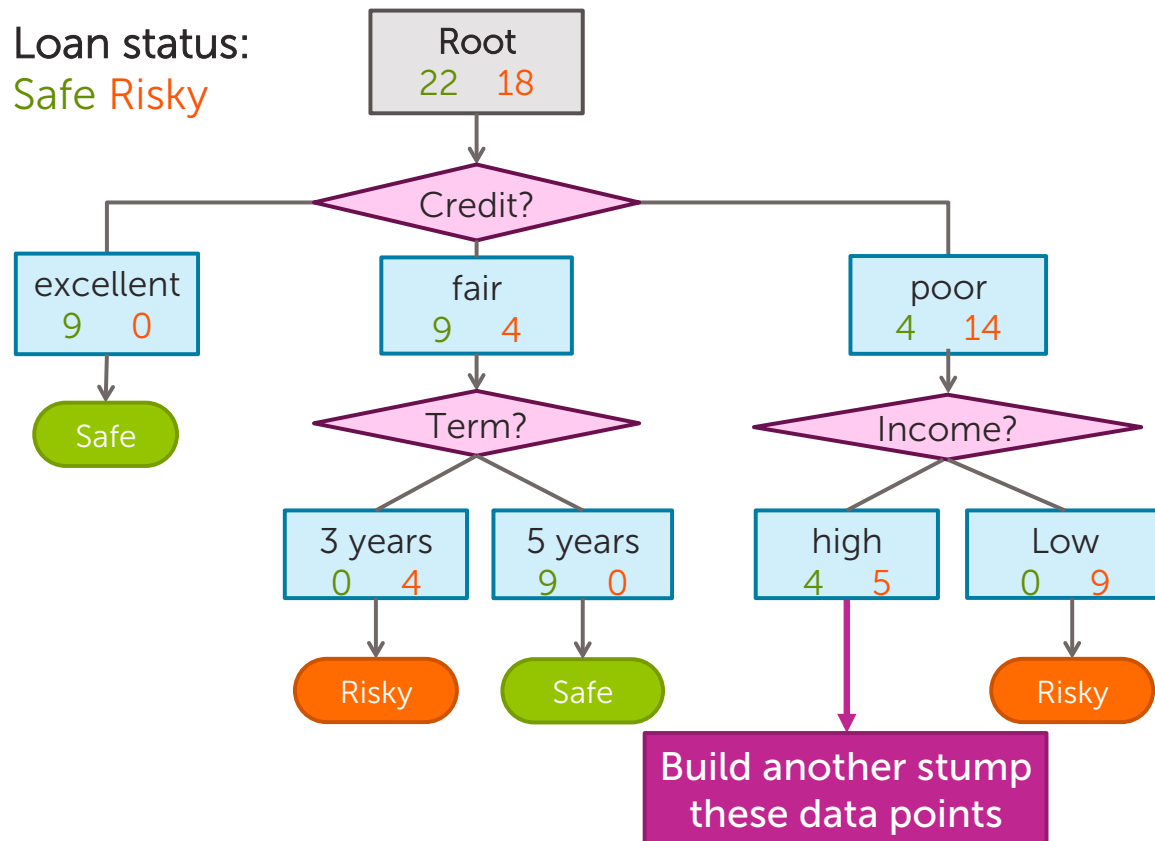


# Tree learning = Recursive stump learning

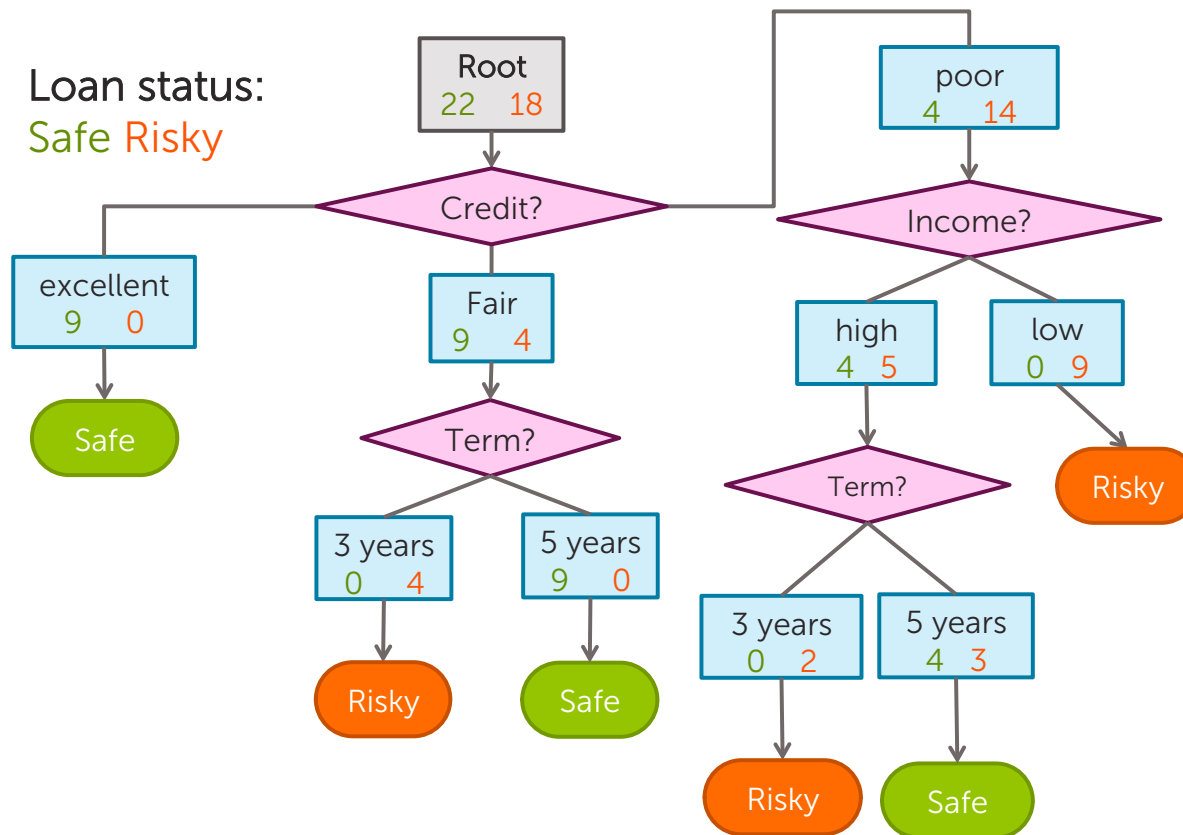
Loan status:  
Safe Risky



## Second level

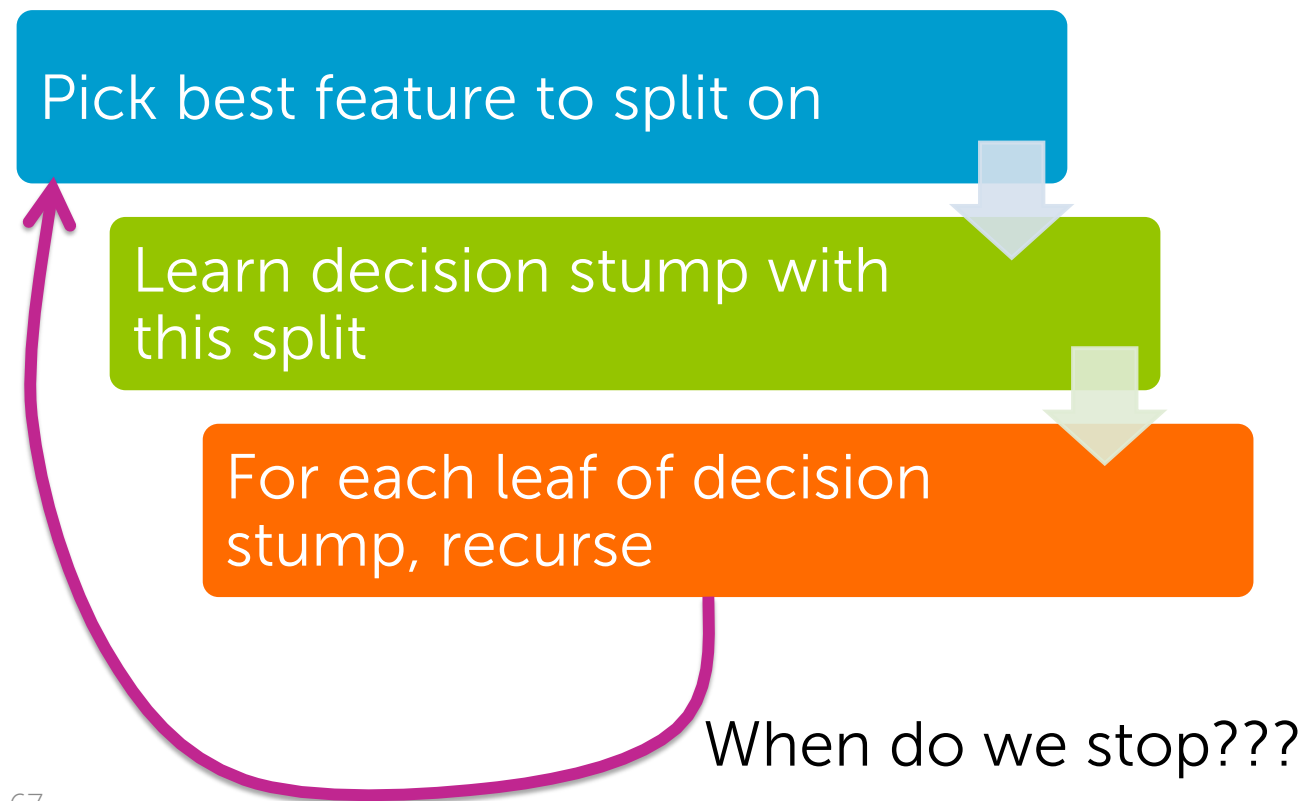


# Final decision tree



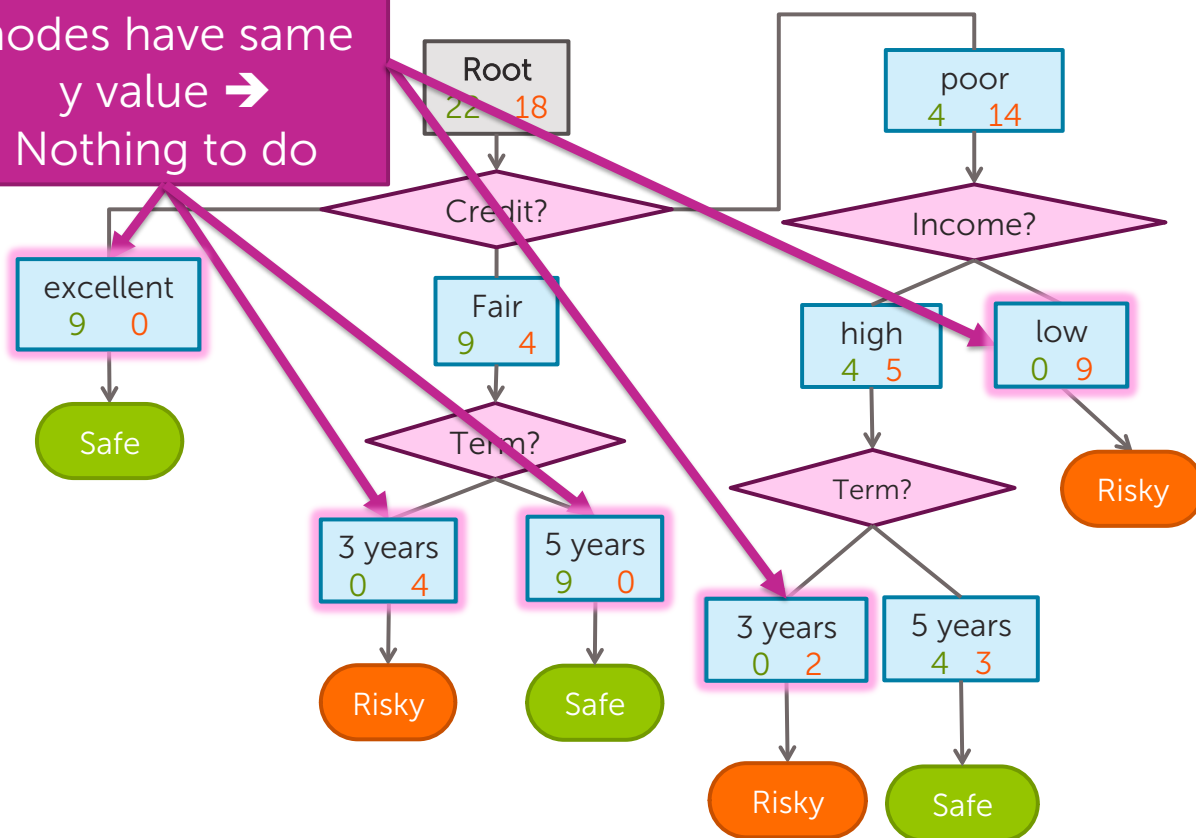


# Simple greedy decision tree learning



# Stopping condition 1: All data agrees on y

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

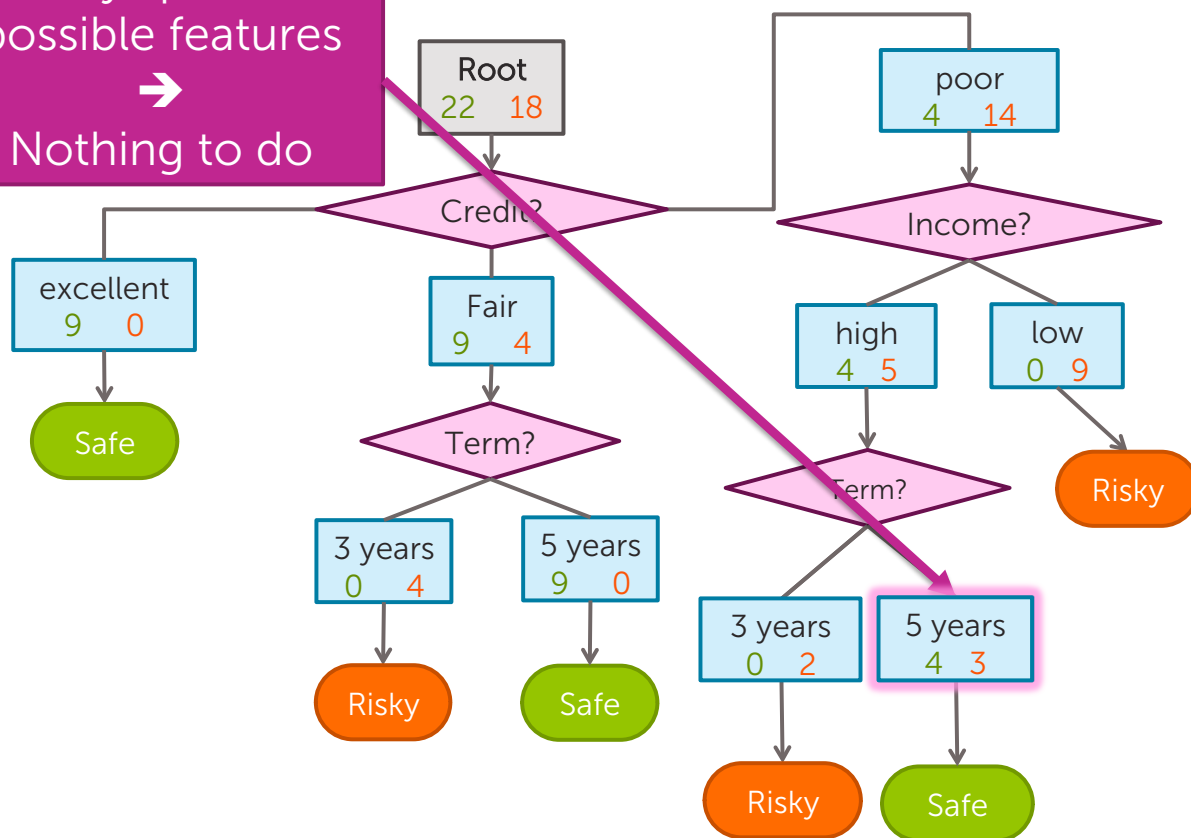


## Stopping condition 2: Already split on all features

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

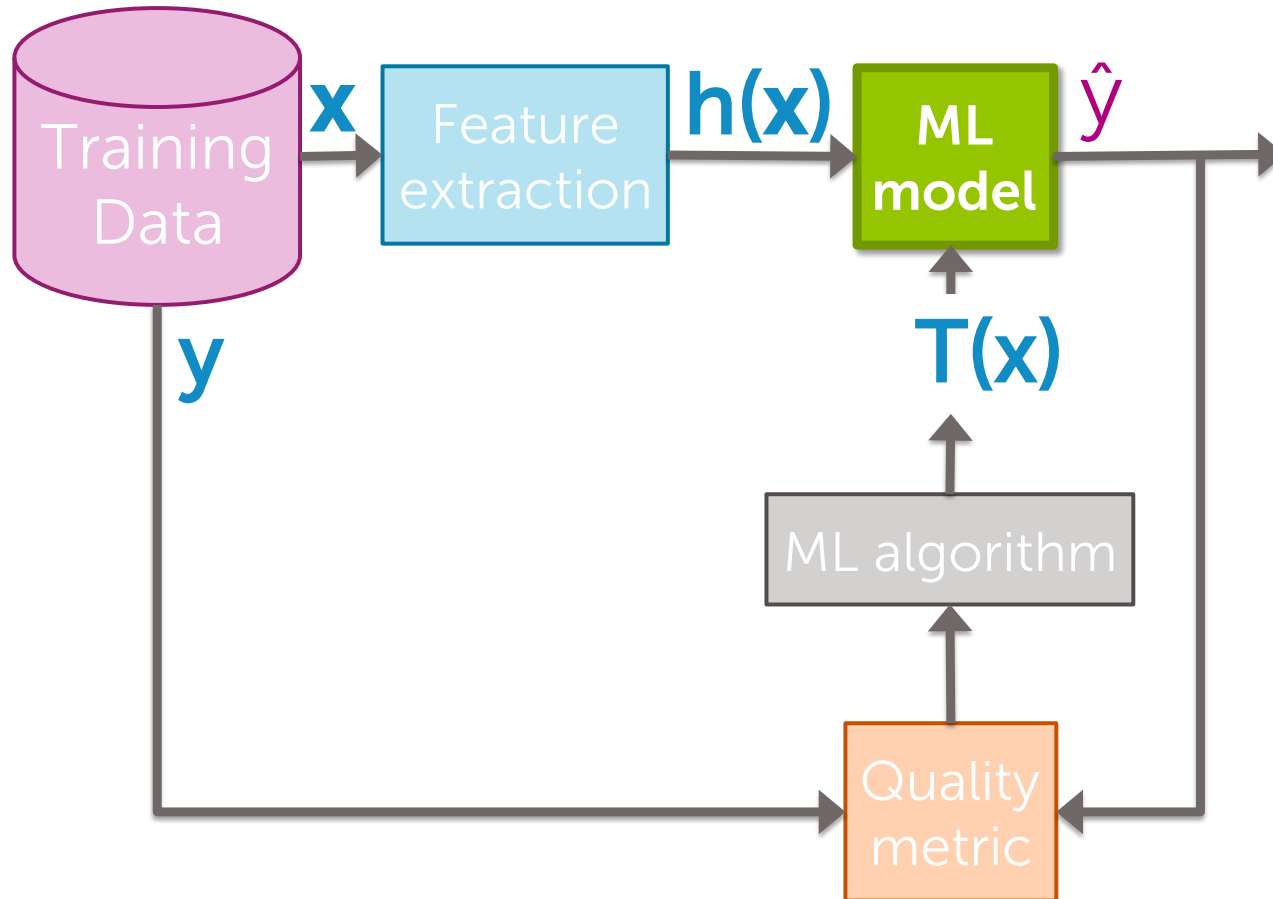
- **Step 4:** Otherwise, go to **Step 2** & continue (recurse) on this split

Pick feature split leading to lowest classification error

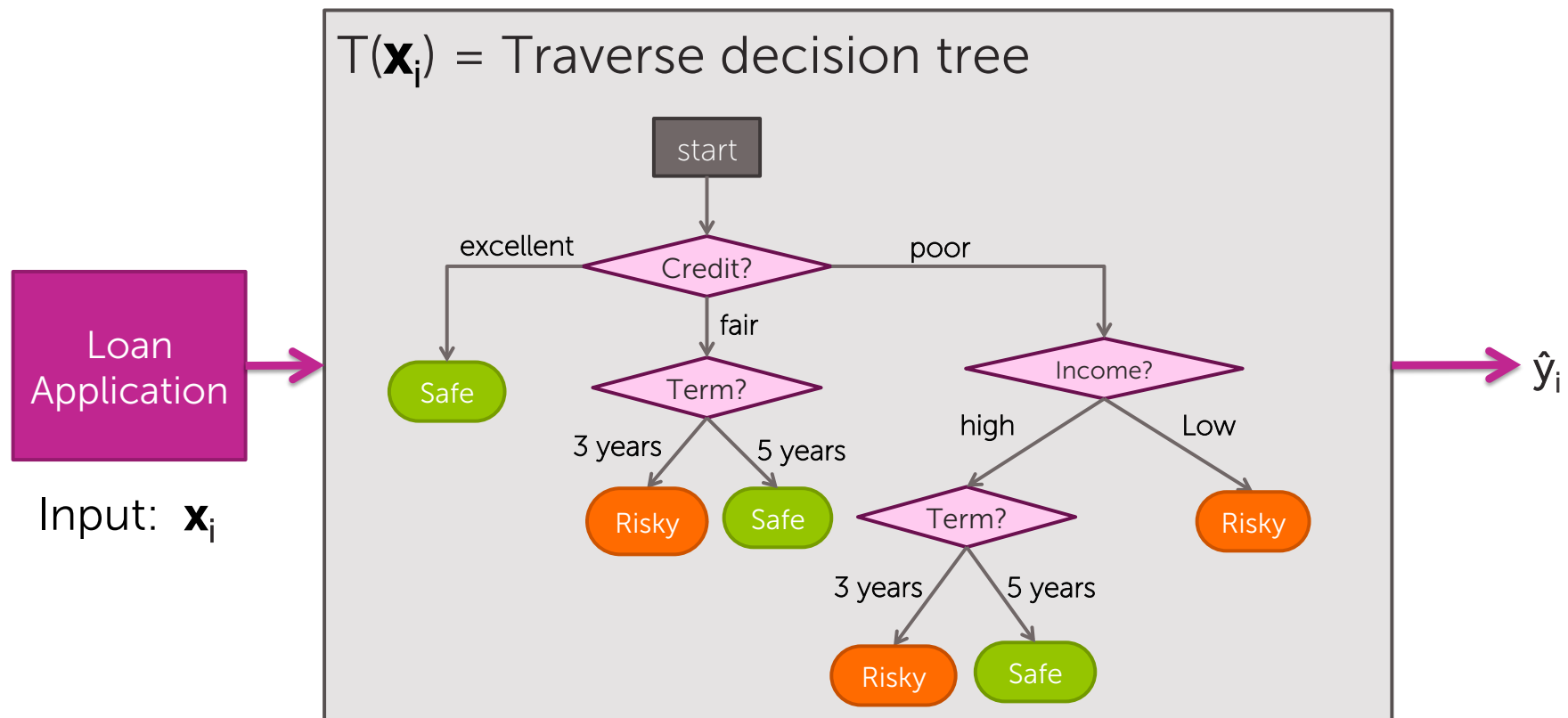
Stopping conditions 1 & 2

Recursion

# Predictions with decision trees

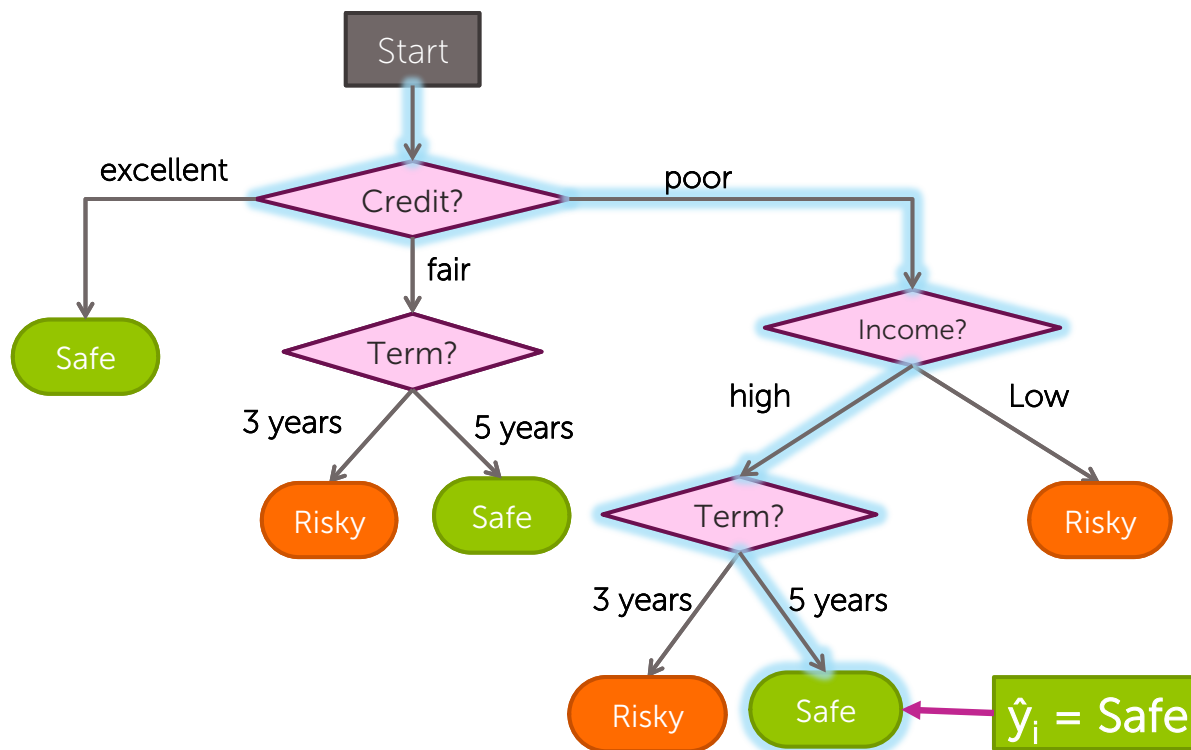


# Decision tree model



# Traversing a decision tree

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







# Decision tree prediction algorithm

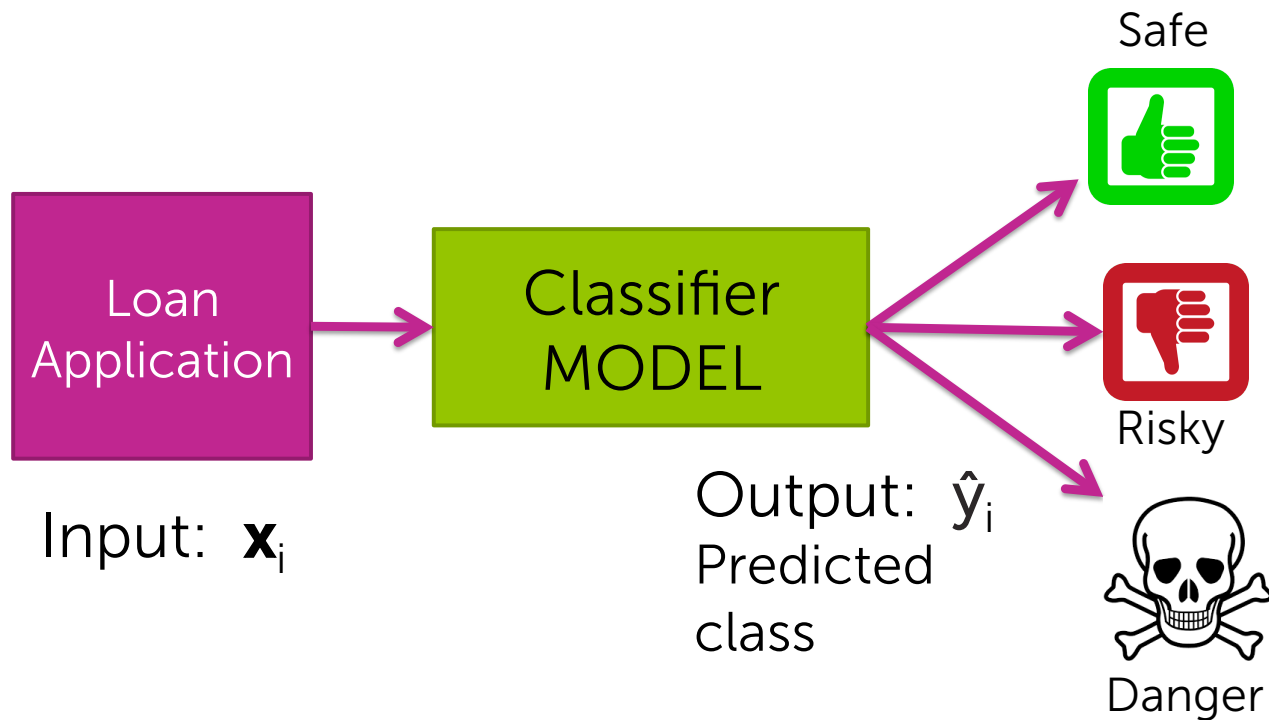
**predict**(tree\_node, input)

- 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

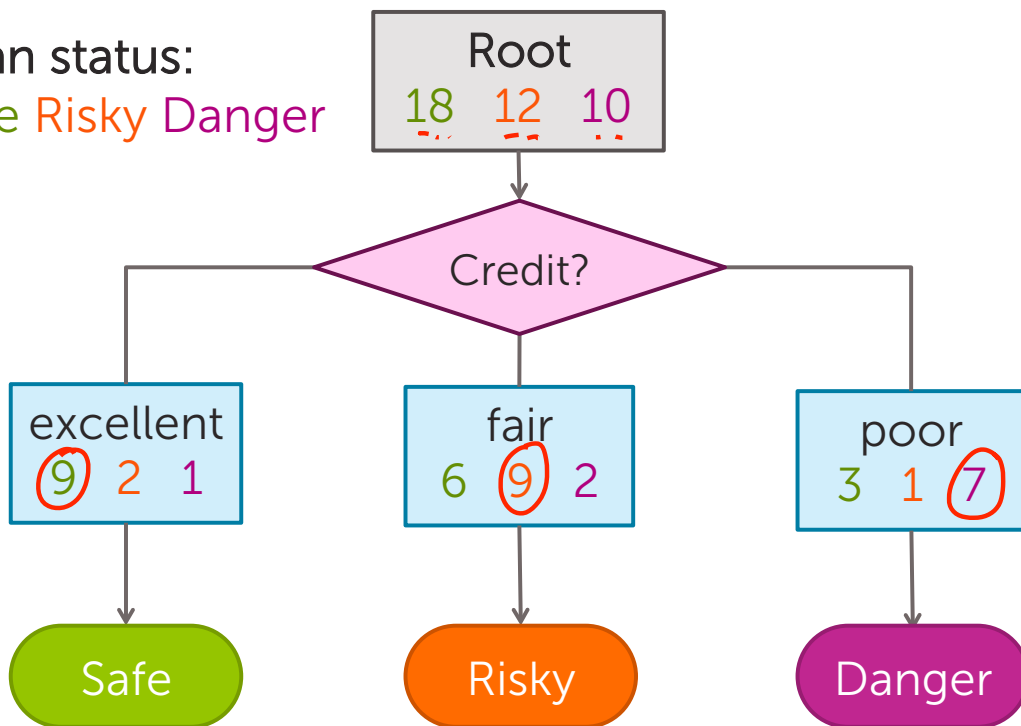


# Multiclass decision stump

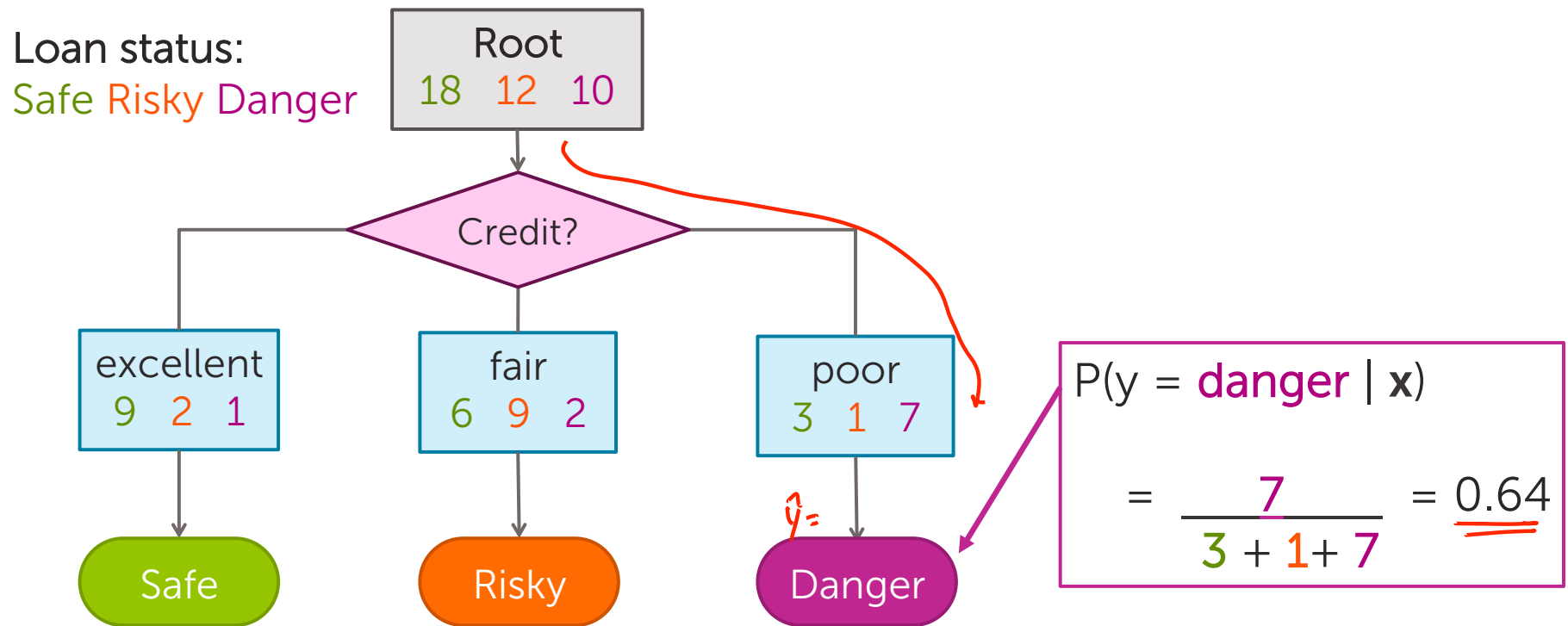
N = 40,  
1 feature,  
3 classes

Credit	y
excellent	<u>safe</u>
fair	<u>risky</u>
fair	safe
poor	<u>danger</u>
excellent	risky
fair	safe
poor	danger
poor	safe
fair	safe
...	...

Loan status:  
Safe Risky Danger



# Predicting probabilities with decision trees





# Decision tree learning: *Real valued features*

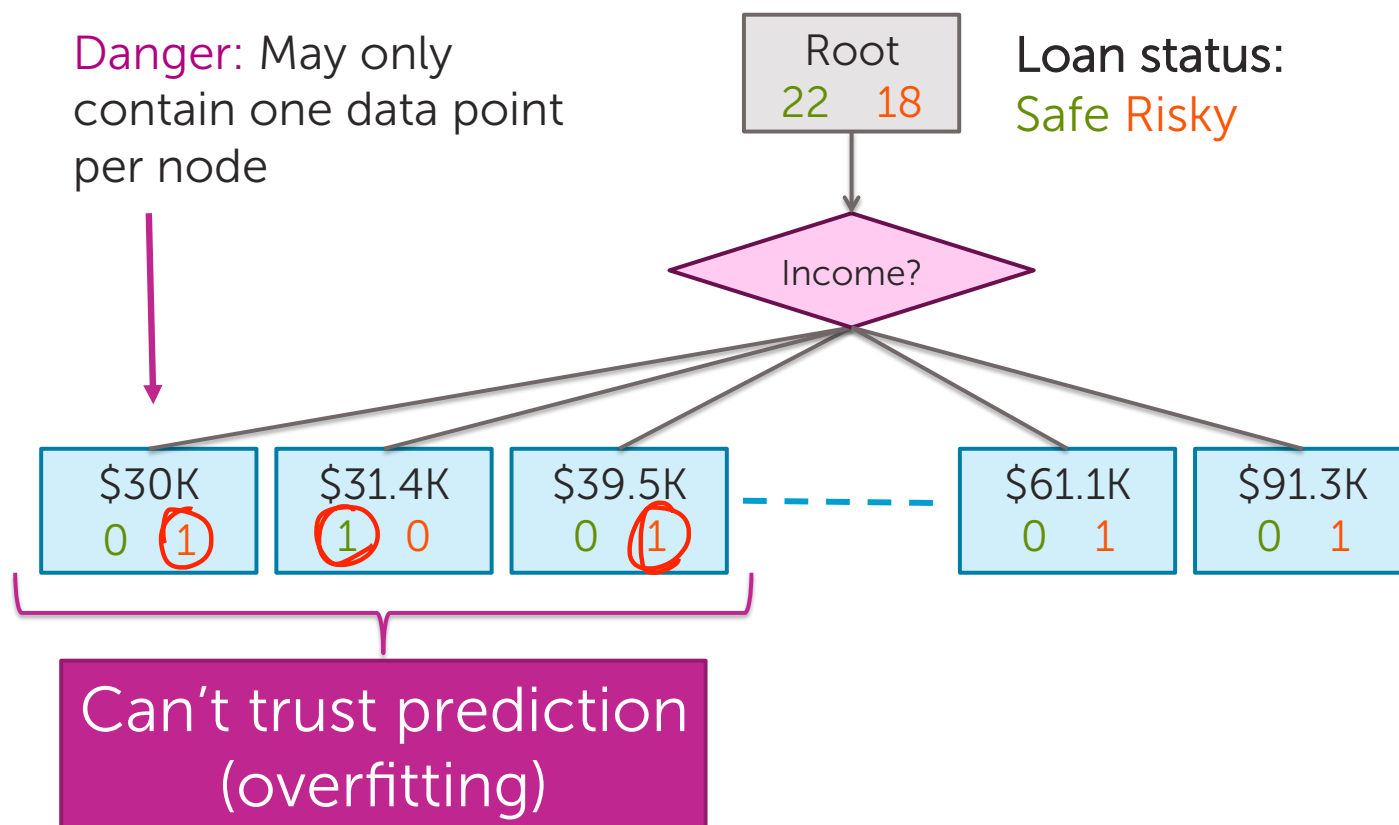
# How do we use real values inputs?

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?

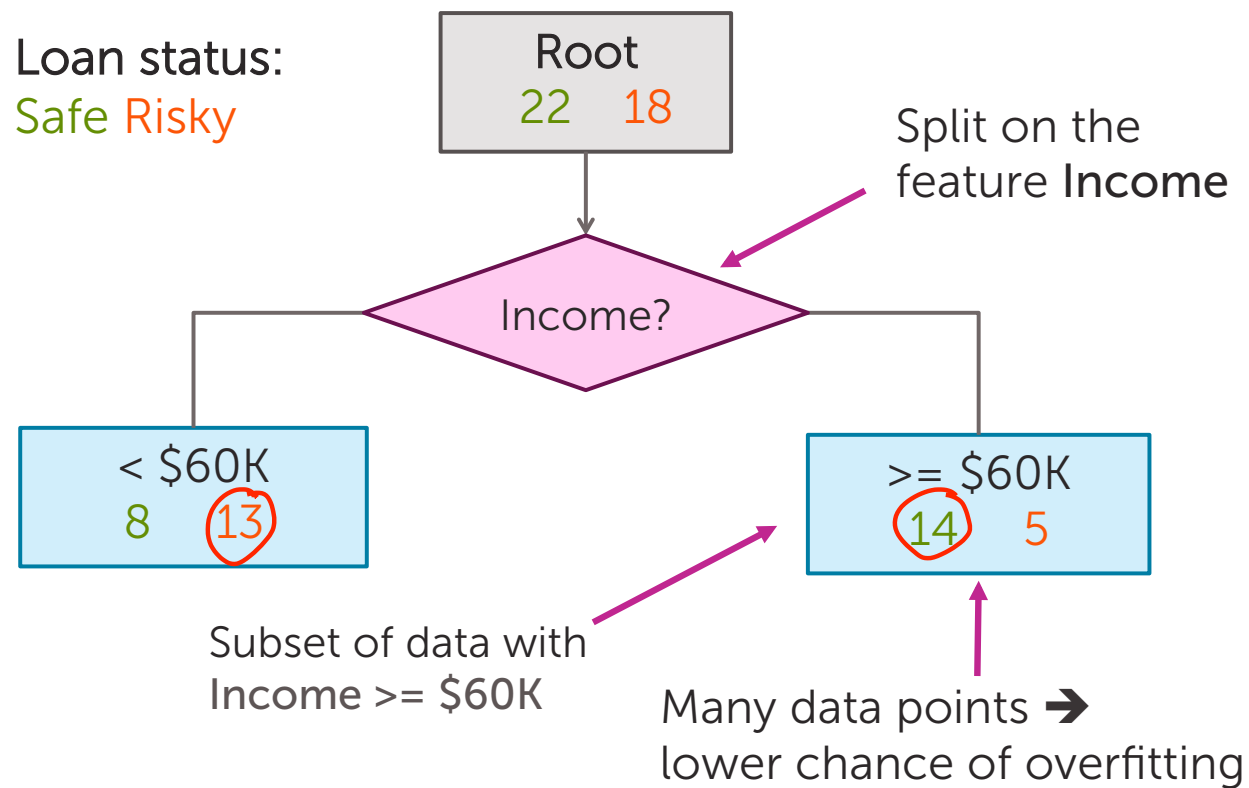
**Danger:** May only contain one data point per node

Loan status:  
Safe Risky



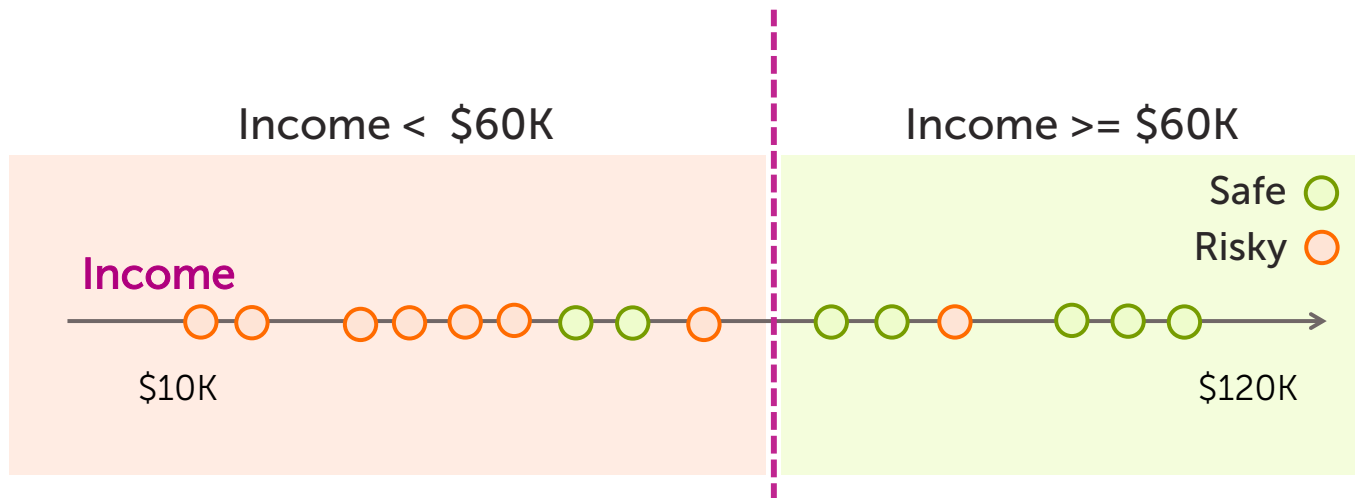


# Alternative: Threshold split

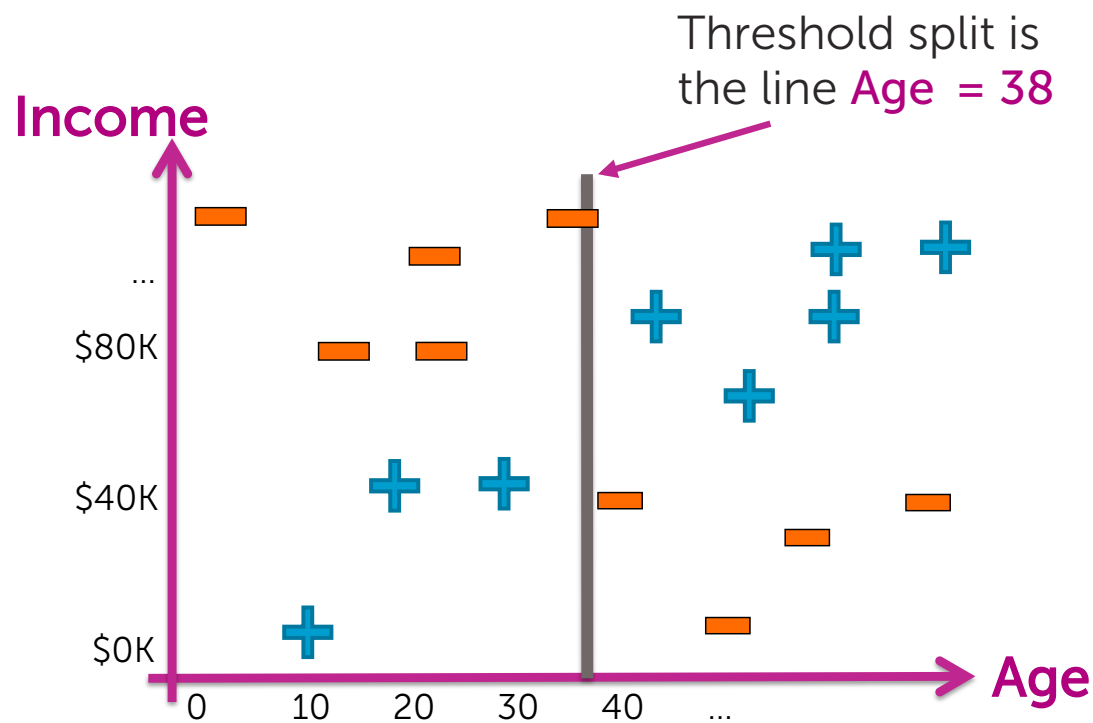


# Threshold splits in 1-D

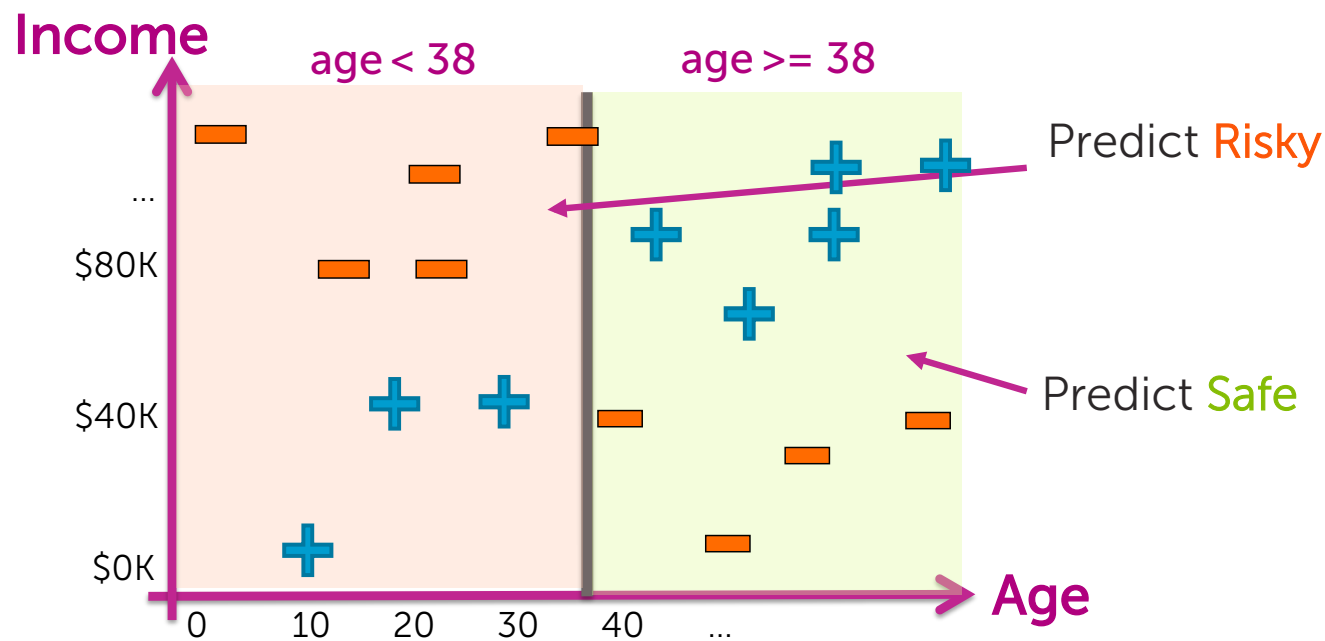
Threshold split is the line  
 $\text{Income} = \$60\text{K}$



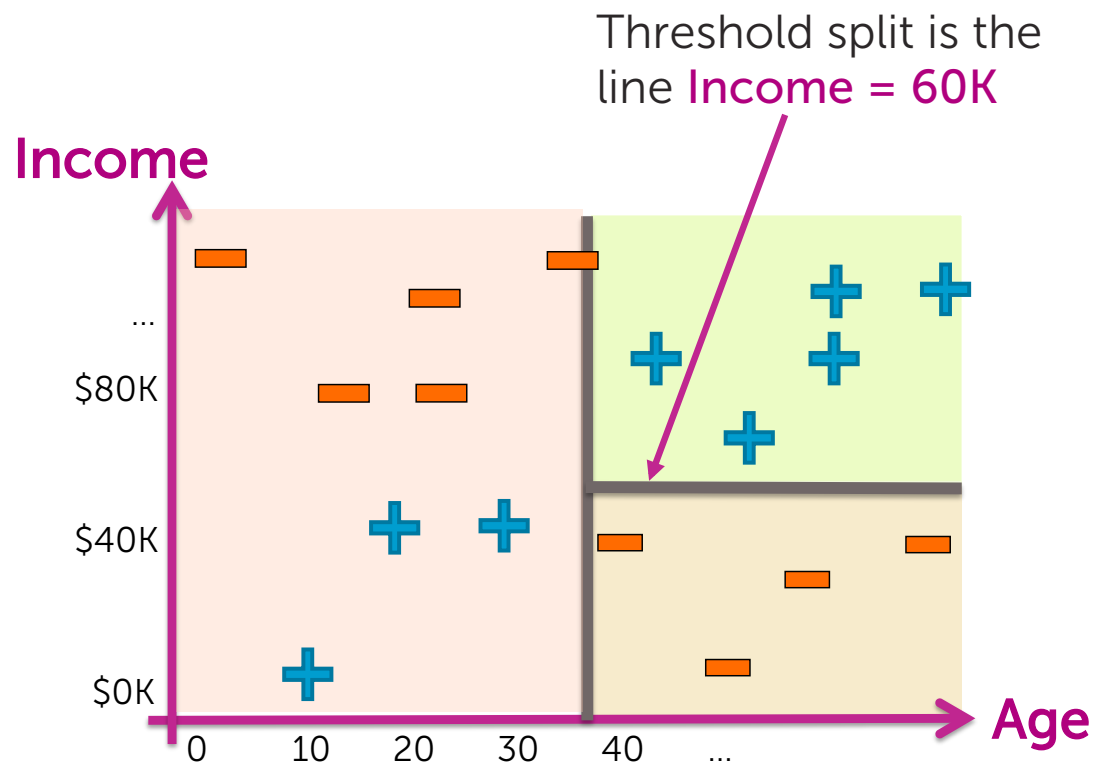
# Visualizing the threshold split



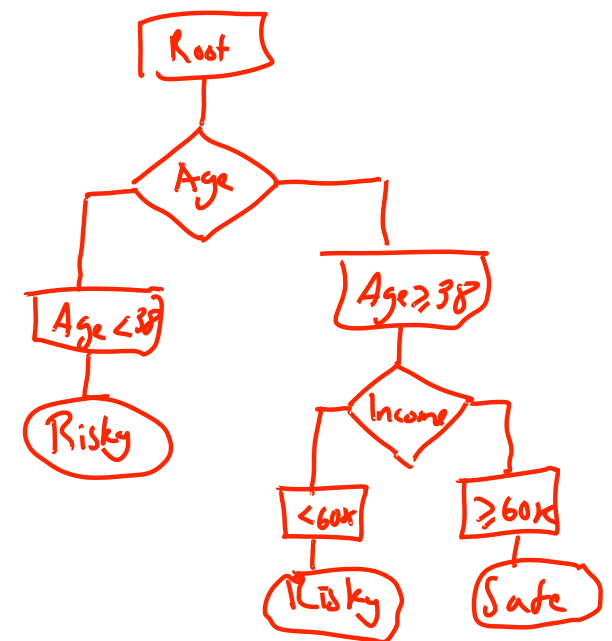
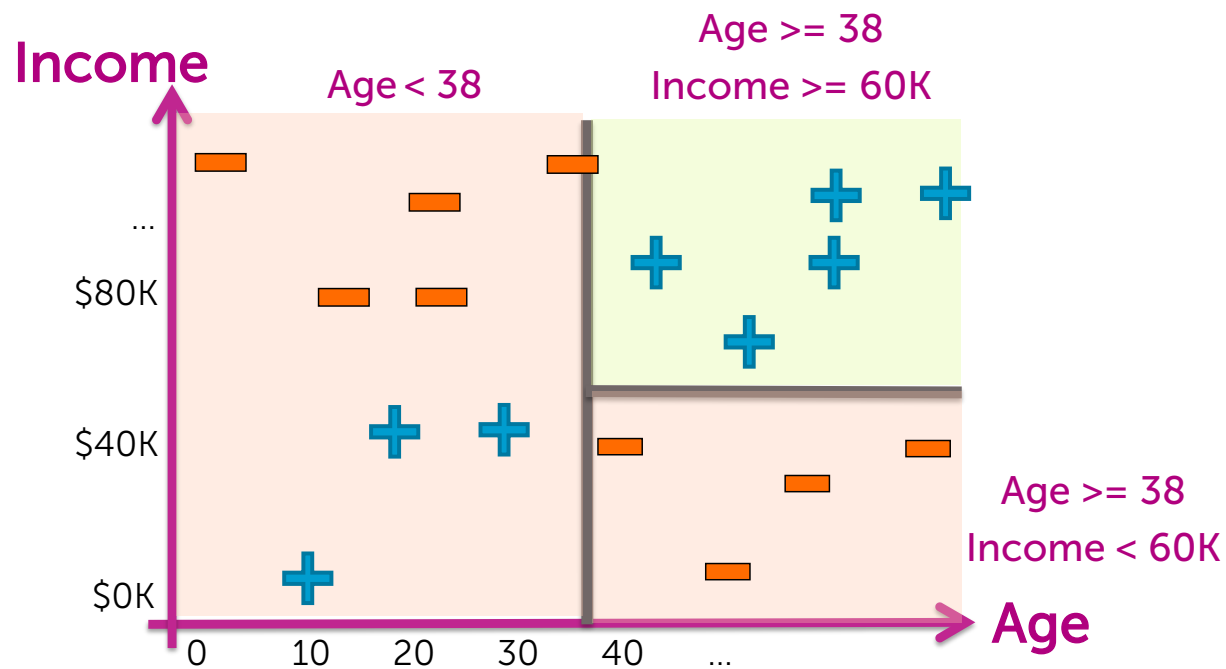
# Split on Age $\geq 38$



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



# Each split partitions the 2-D space





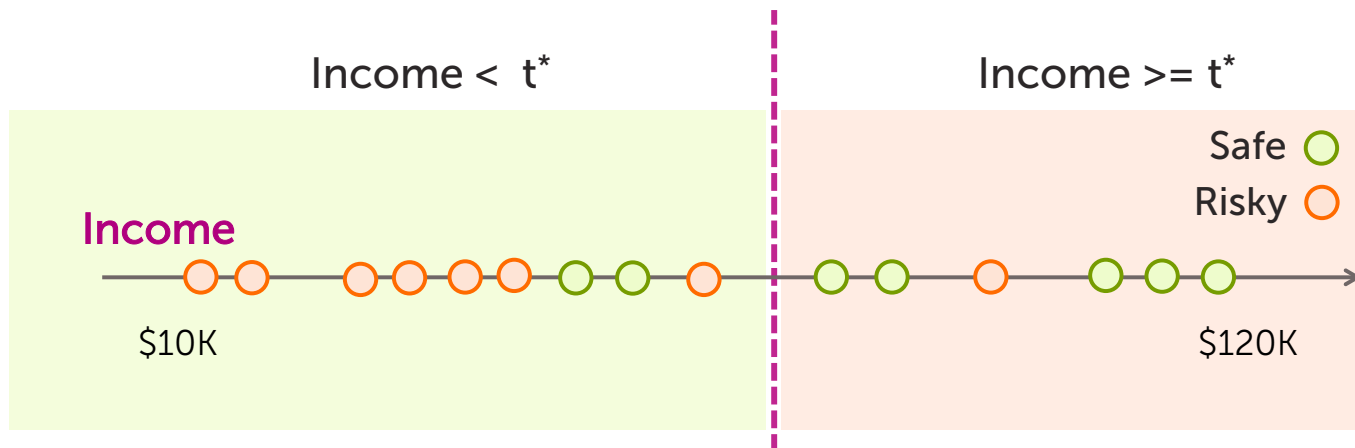
## Finding the best threshold split

**OPTIONAL**

# Finding the best threshold split

Infinite possible  
values of  $t$

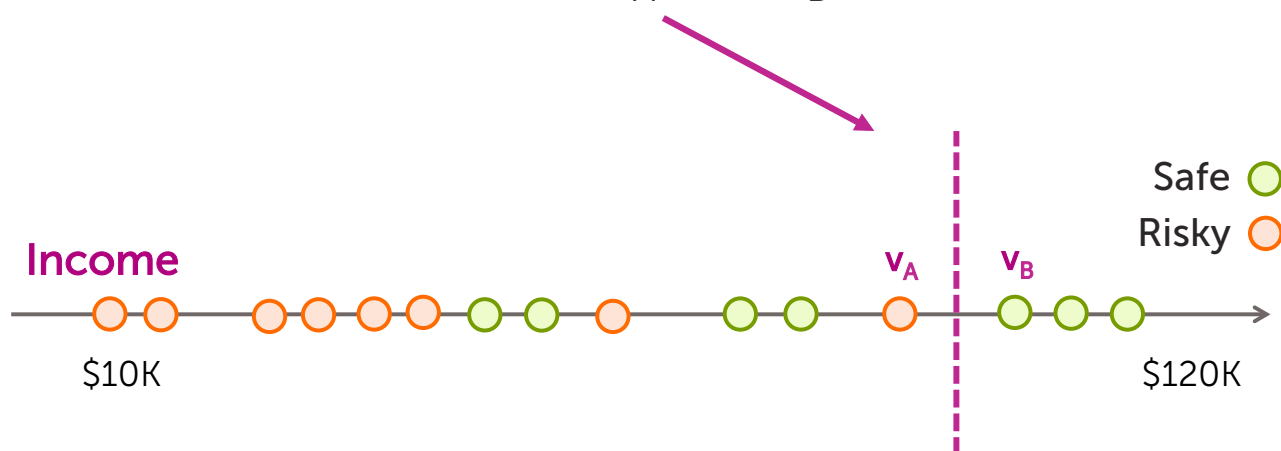
Income =  $t^*$





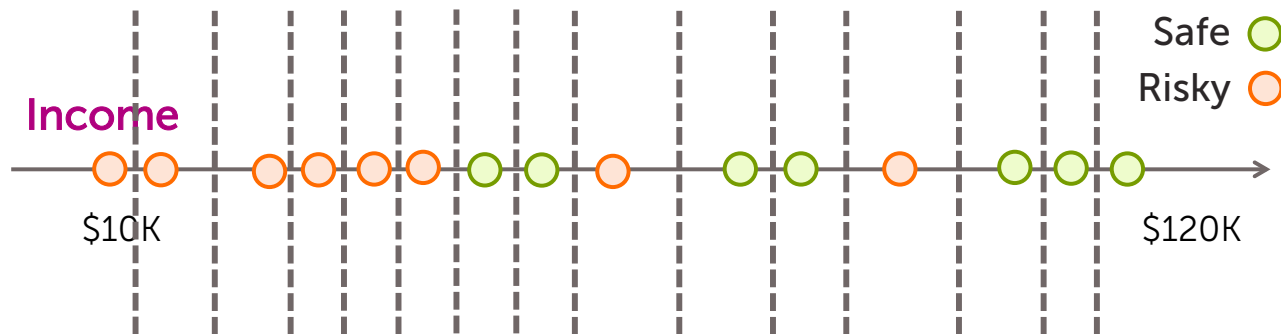
# Consider a threshold between points

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  
split  $h_j(\mathbf{x}) \geq t_i$
  - Chose the  $t^*$  with the lowest classification error

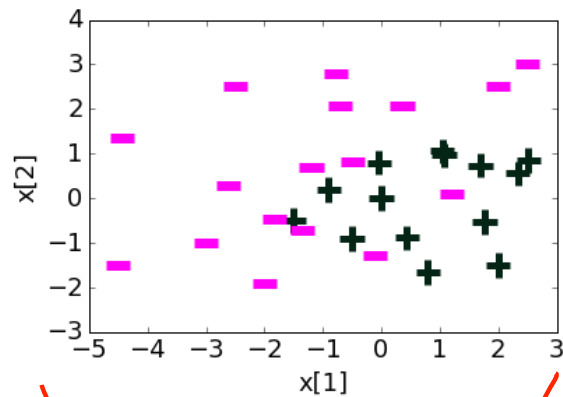
✓ income

# Decision trees vs logistic regression:

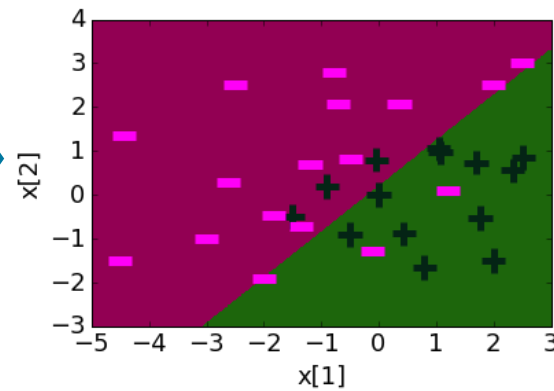
## *Example*

# Logistic regression

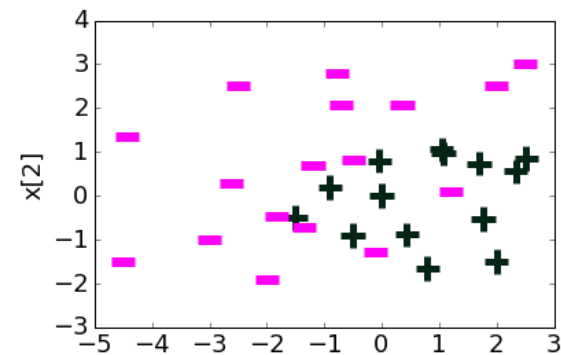
Feature	Value	Weight Learned
$h_0(\mathbf{x})$	1	0.22
$h_1(\mathbf{x})$	$x[1]$	1.12
$h_2(\mathbf{x})$	$x[2]$	-1.07



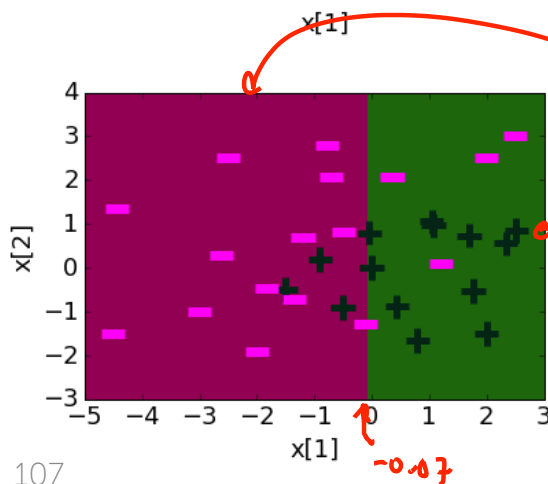
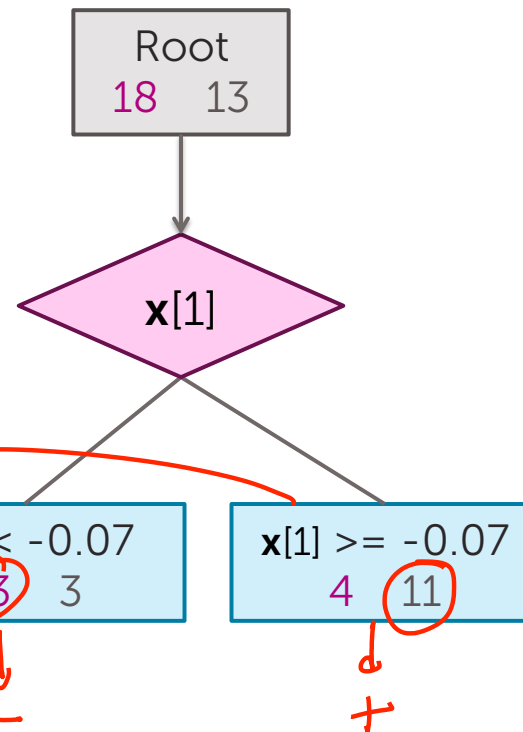
*data*



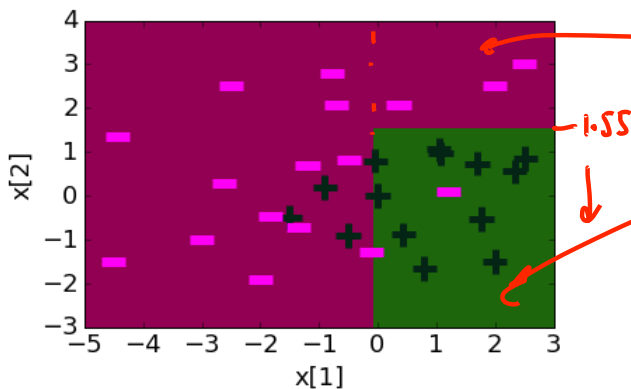
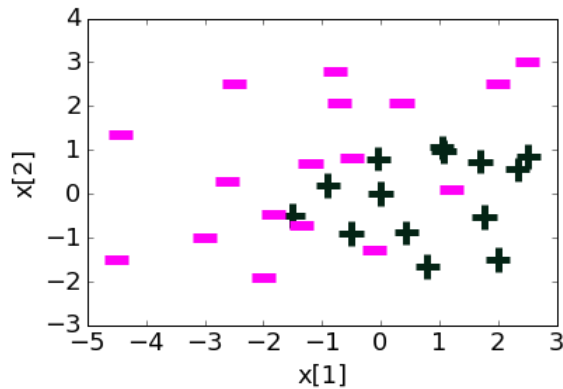
# Depth 1: Split on $x[1]$



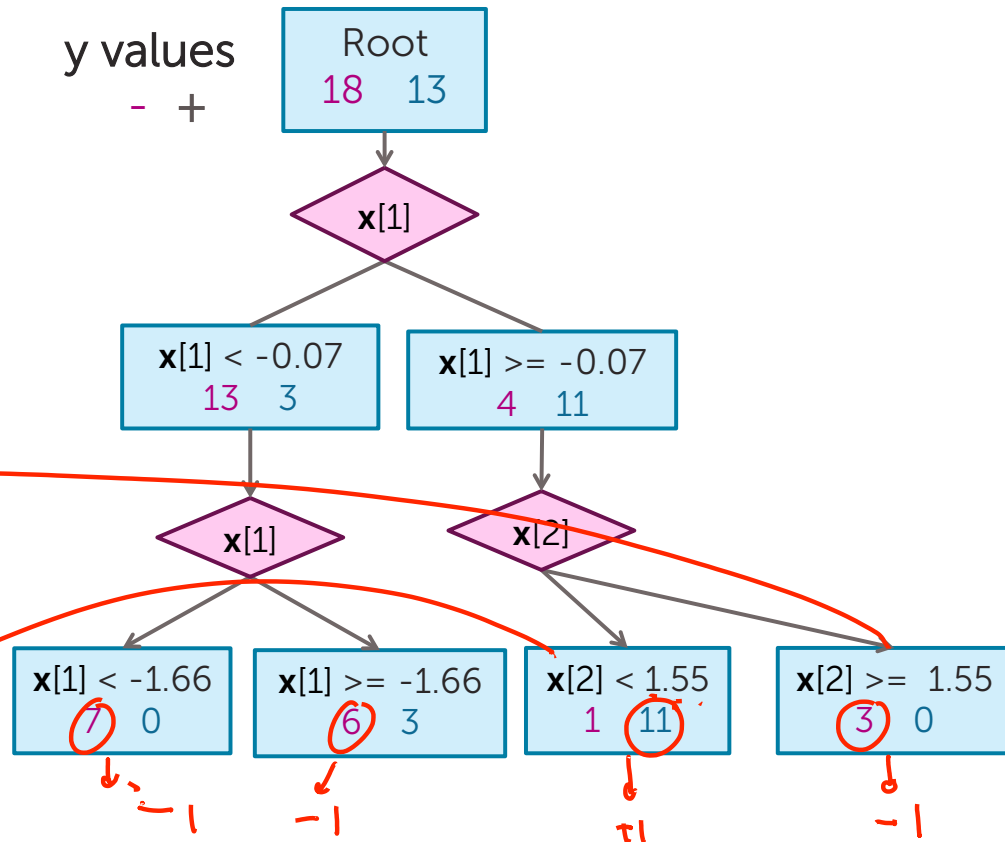
y values  
- +



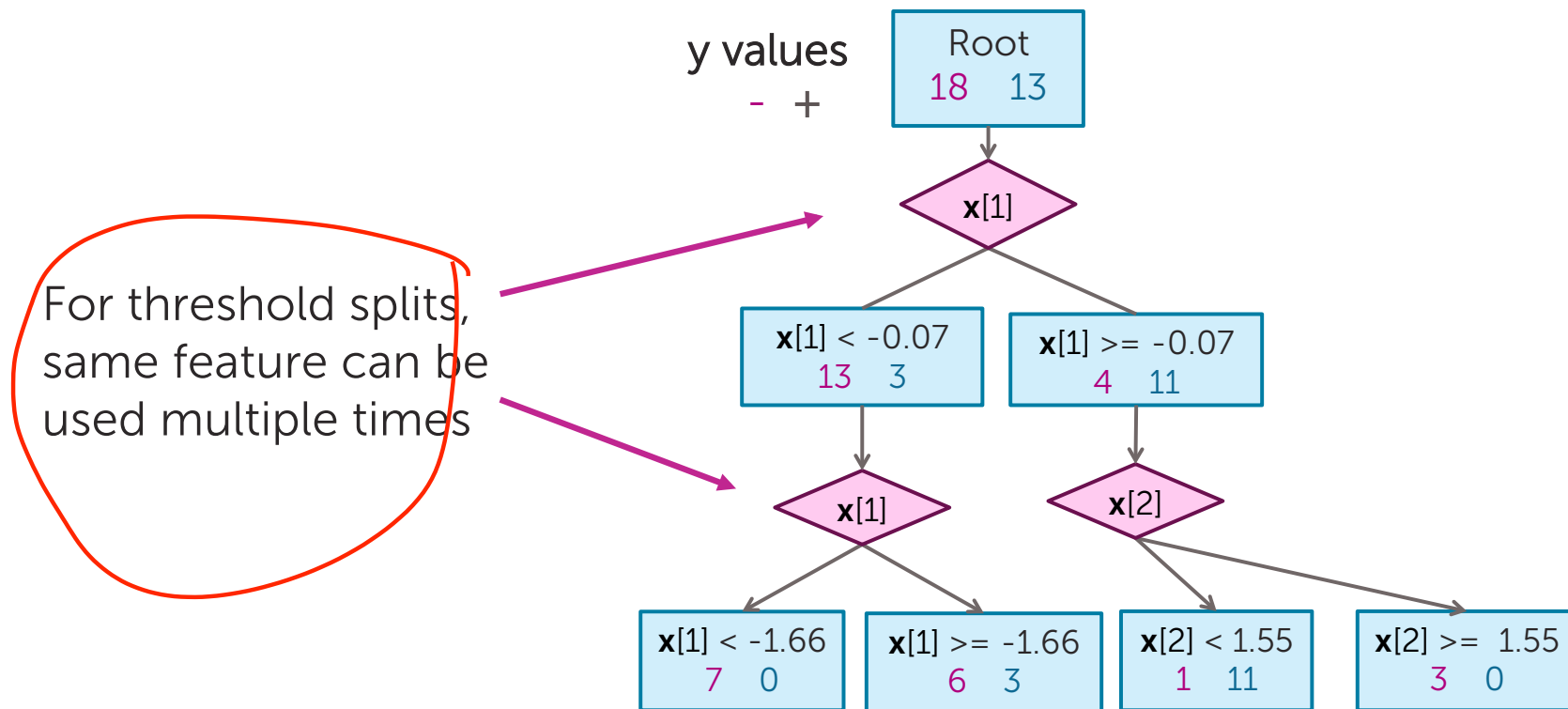
# Depth 2



y values  
- +

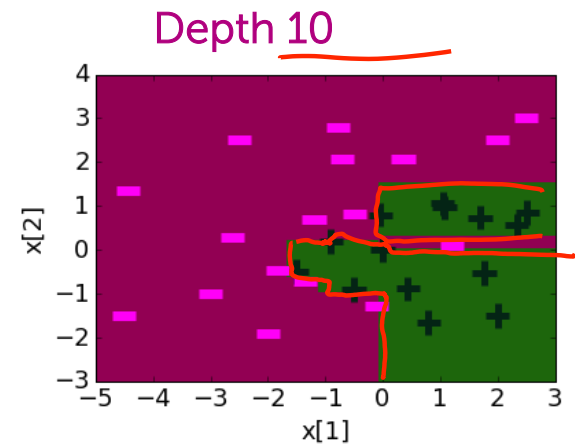
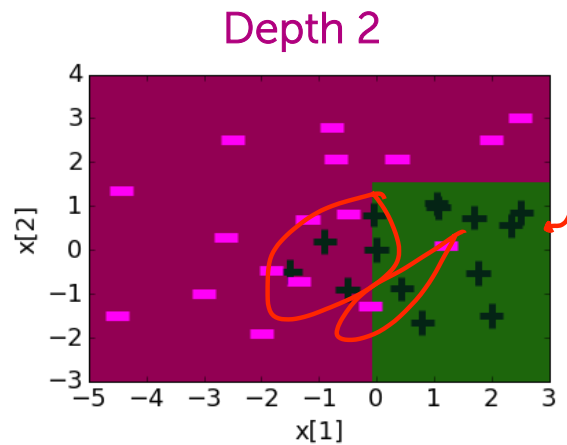
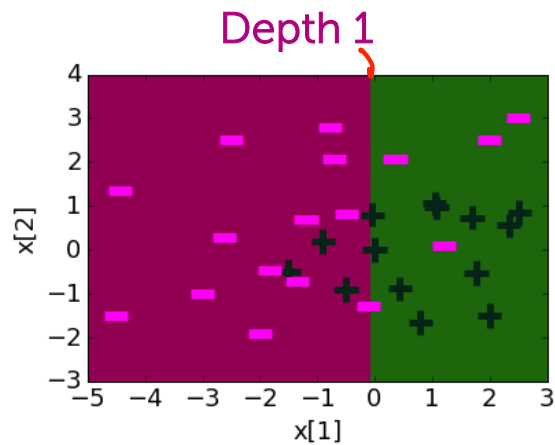
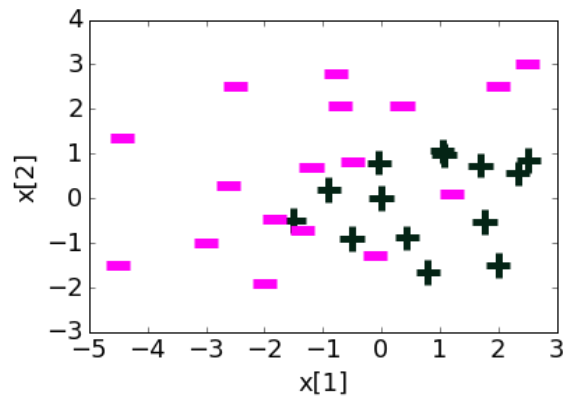


# Threshold split caveat



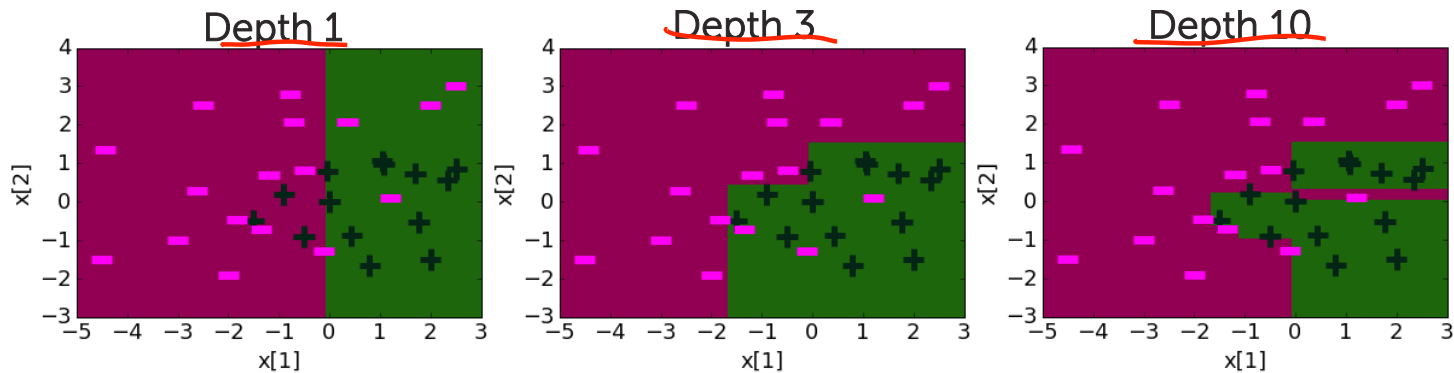


# Decision boundaries

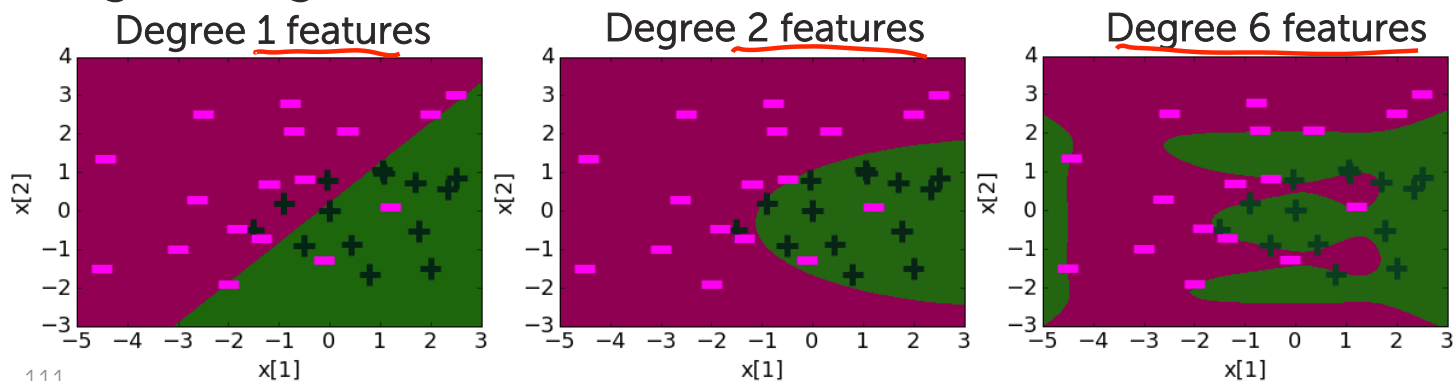


# Comparing decision boundaries

## 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.