Decision Trees

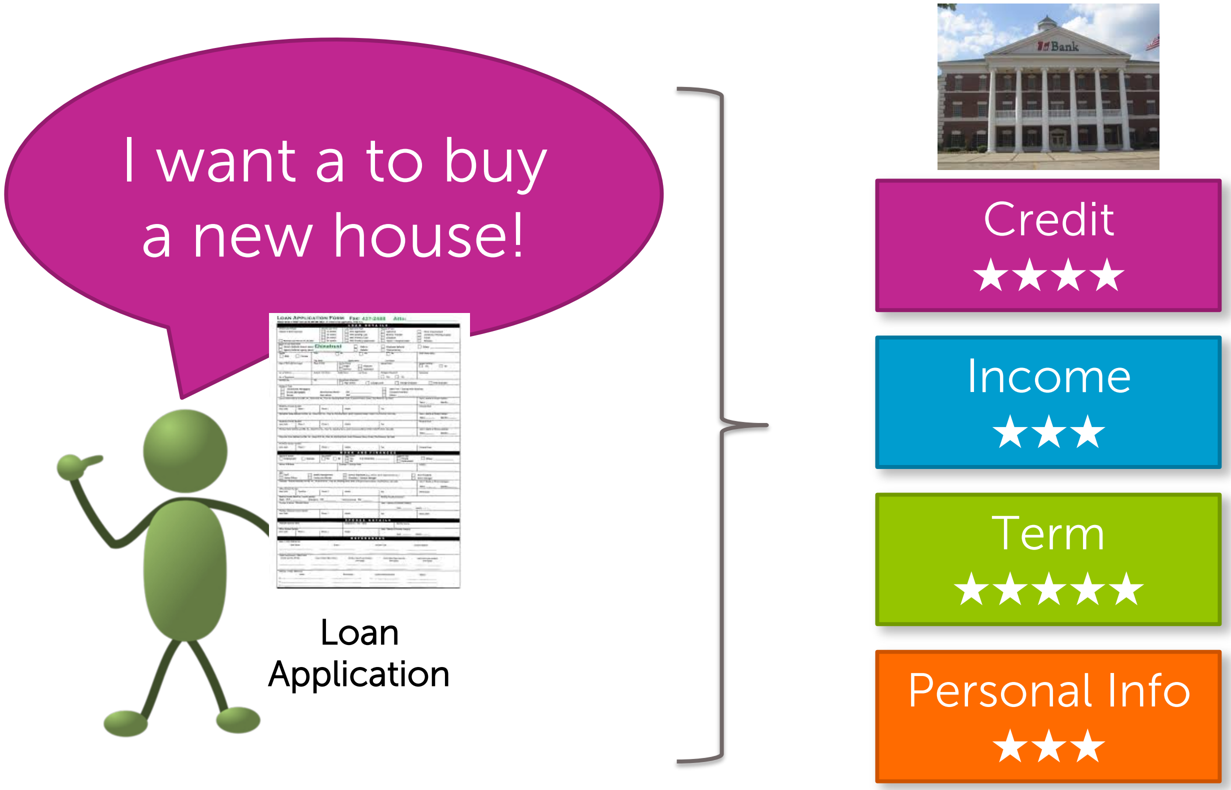
Emily Fox & Carlos Guestrin

Machine Learning Specialization

University of Washington

Predicting potential loan defaults

What makes a loan risky?



Credit ★★★★

Income ★★★

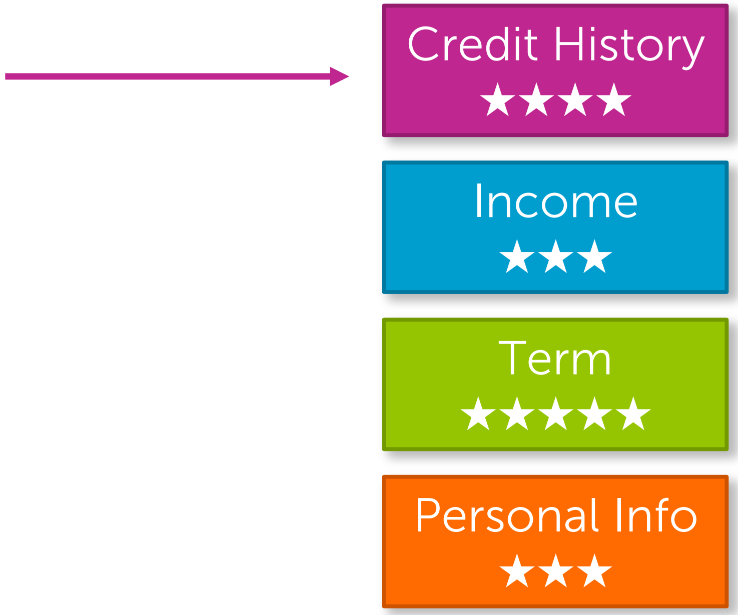
Term ★★★★★

Personal Info ★★★

Credit history explained

Did I pay previous loans on time?

Example: excellent, good, or fair



Credit History
★★★★

Income
★★★

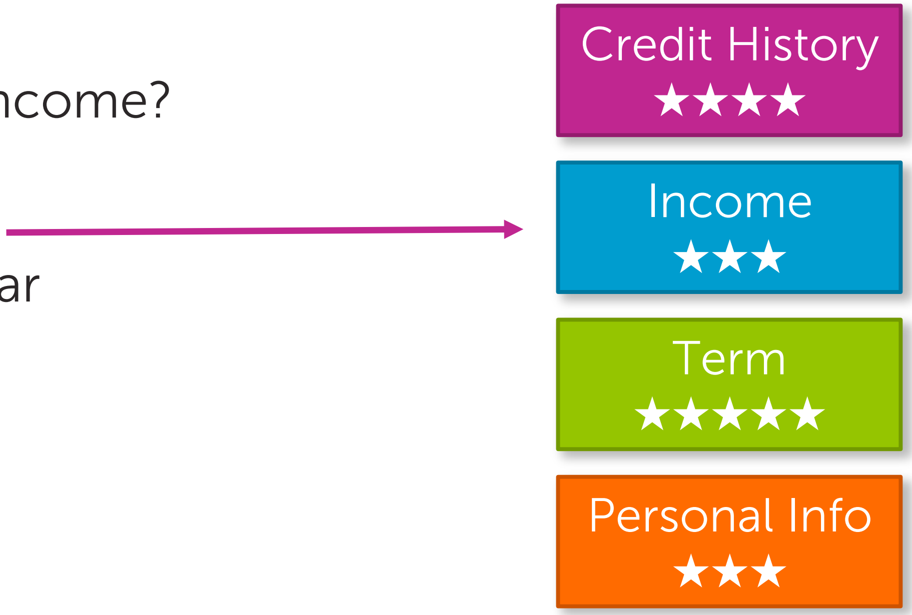
Term
★★★★★

Personal Info
★★★

Income

What's my income?

Example:
$80K per year



Credit History
★★★★

Income
★★★

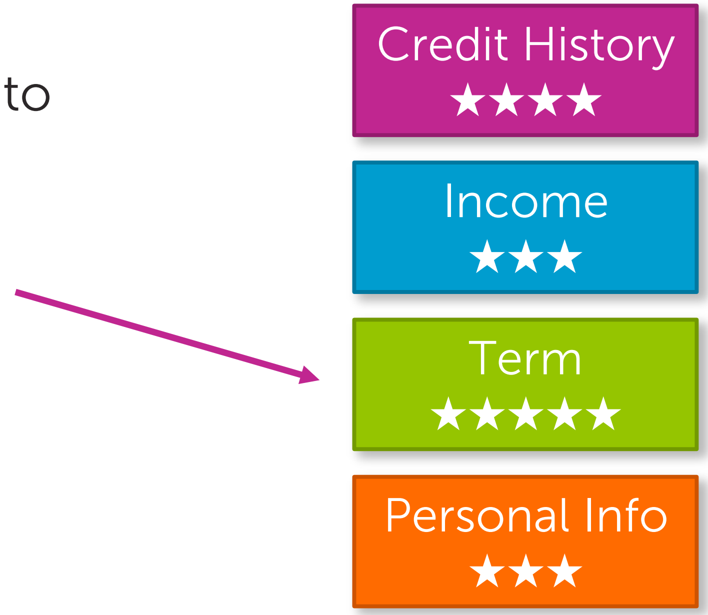
Term
★★★★★

Personal Info
★★★

Loan terms

How soon do I need to pay the loan?

Example: 3 years, 5 years,…



Credit History ★★★★

Income ★★★

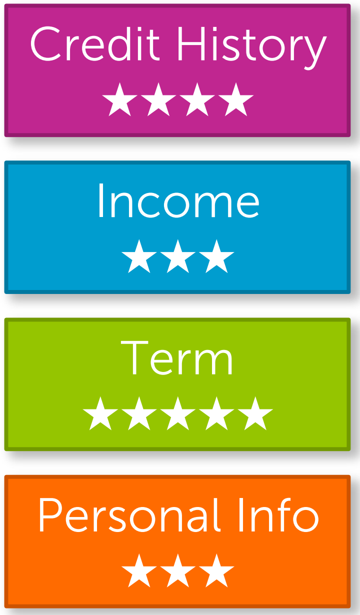
Term ★★★★★

Personal Info ★★★

Personal information

Age, reason for the loan, marital status,…

Example: Home loan for a married couple



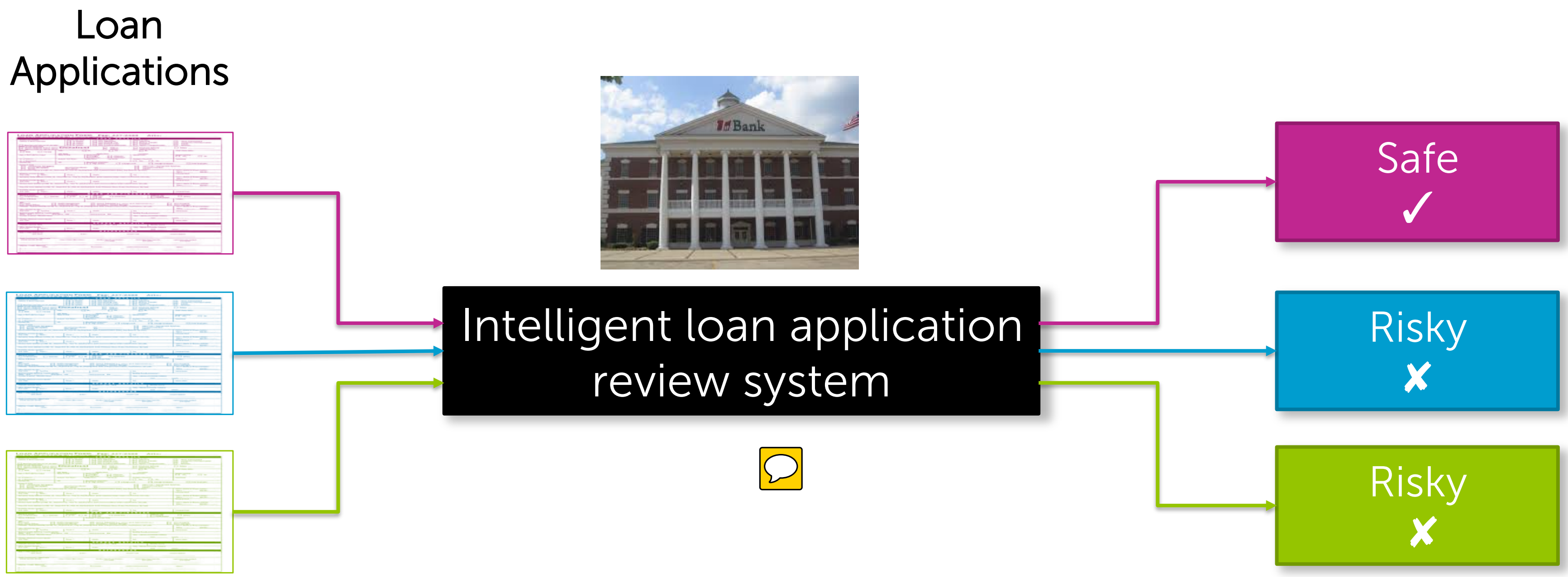
Credit History ★★★★

Income ★★★

Term ★★★★★

Personal Info ★★★

Intelligent application



Loan Applications

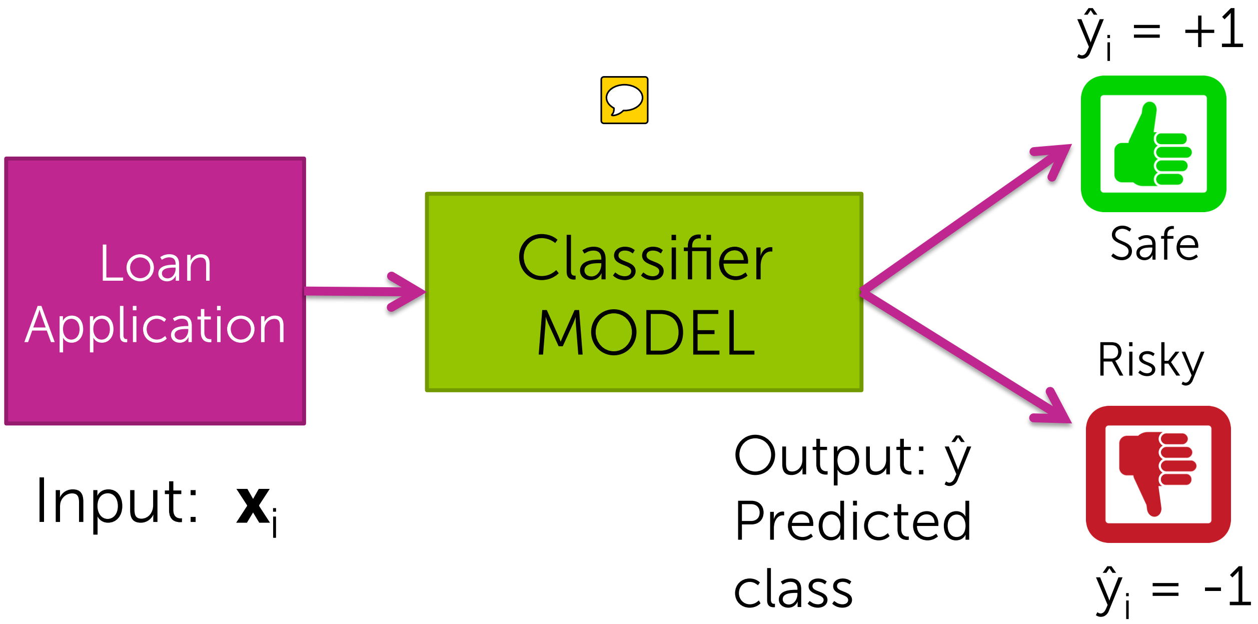
Intelligent loan application review system

Safe ✓

Risky ✘

Risky ✘

Classifier review



Loan Application

Input: $\mathbf{x}_i$

Classifier MODEL

Output: $\hat{y}$ Predicted class

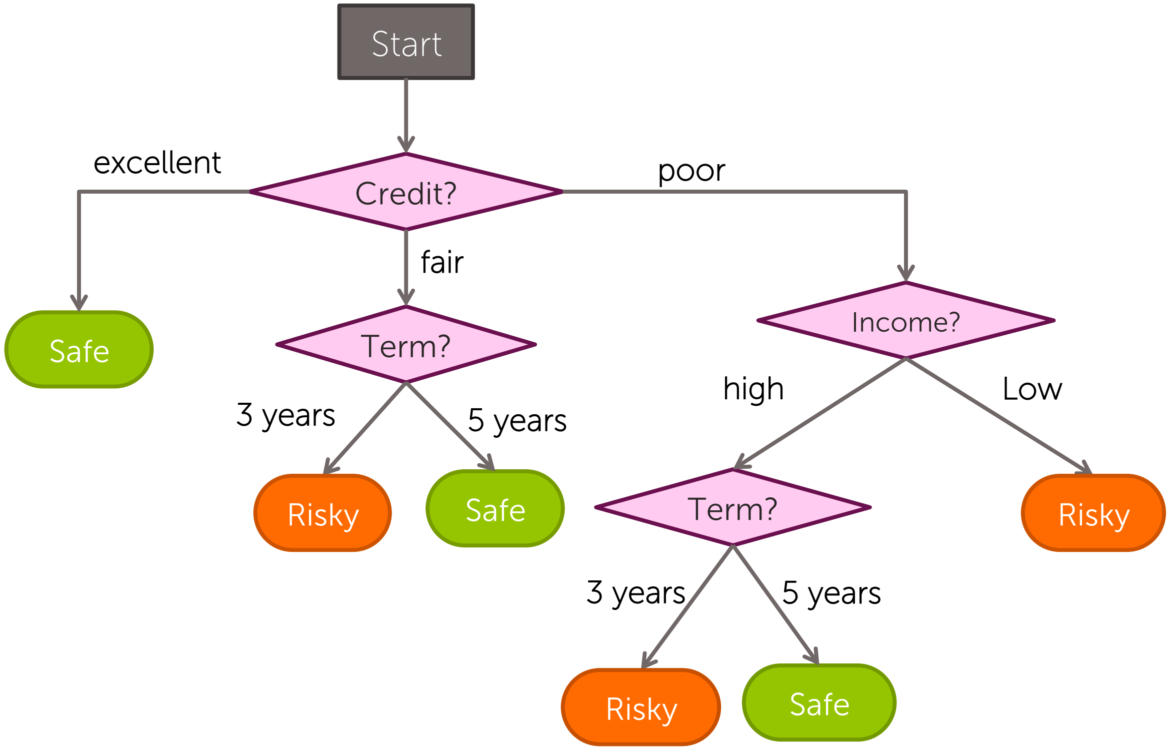
$\hat{y}_i = +1$

Safe

Risky

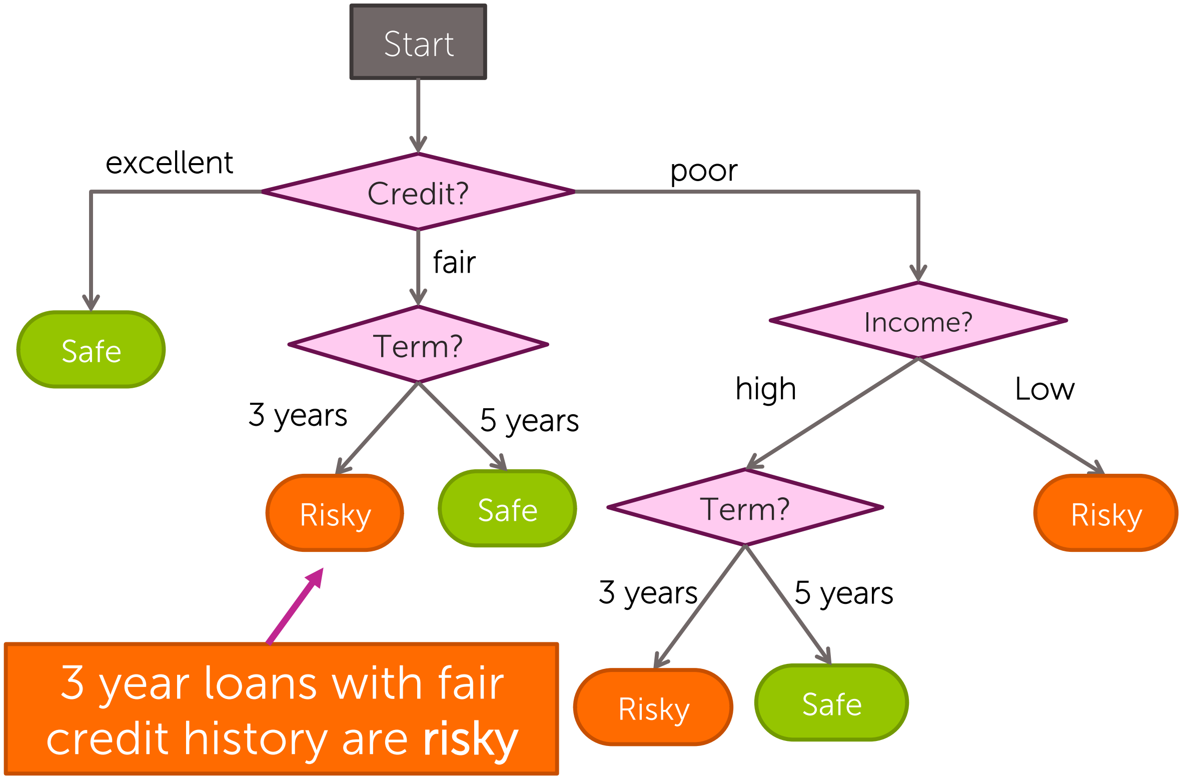
$\hat{y}_i = -1$

This module ... decision trees



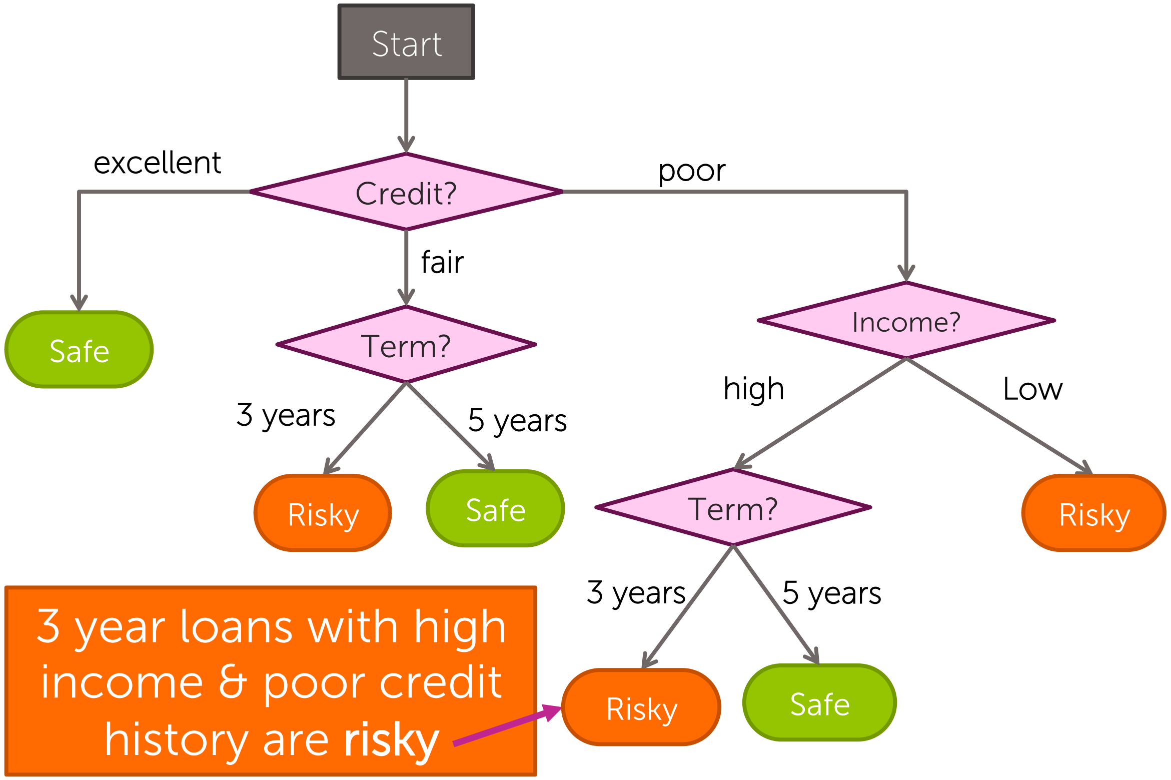
Decision trees: *Intuition*

What does a decision tree represent?



3 year loans with fair credit history are **risky**

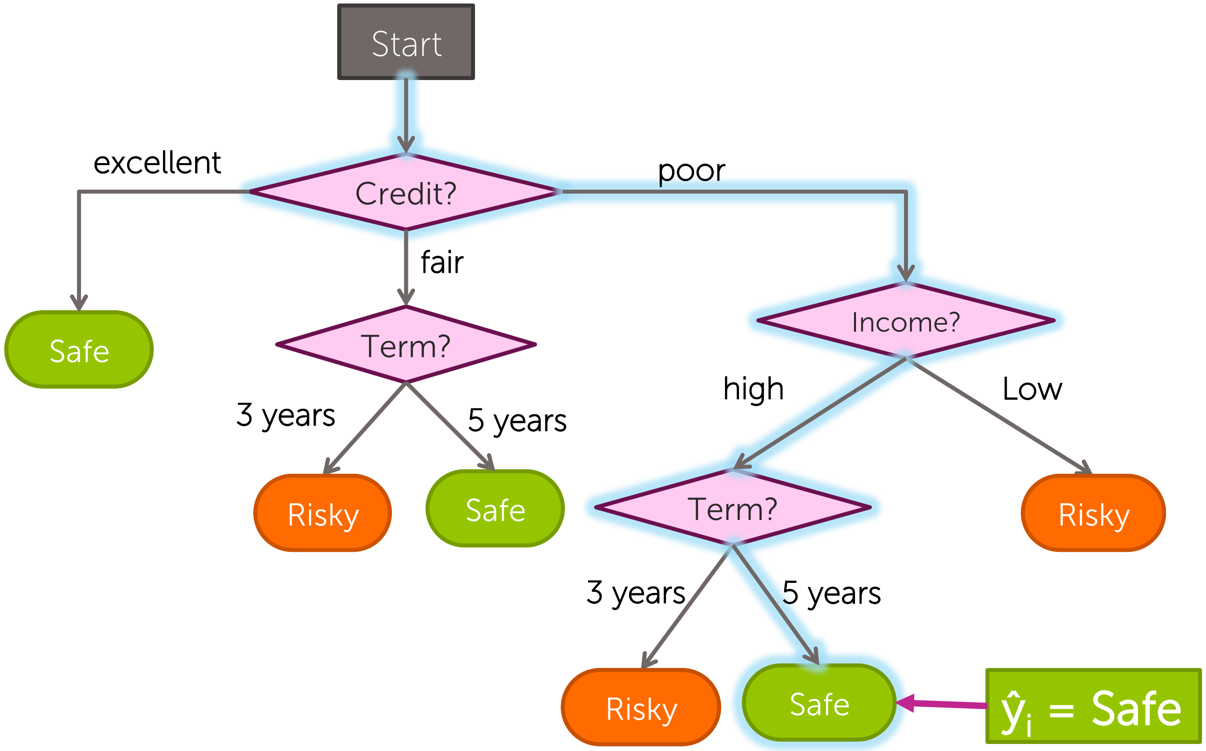
What does a decision tree represent?



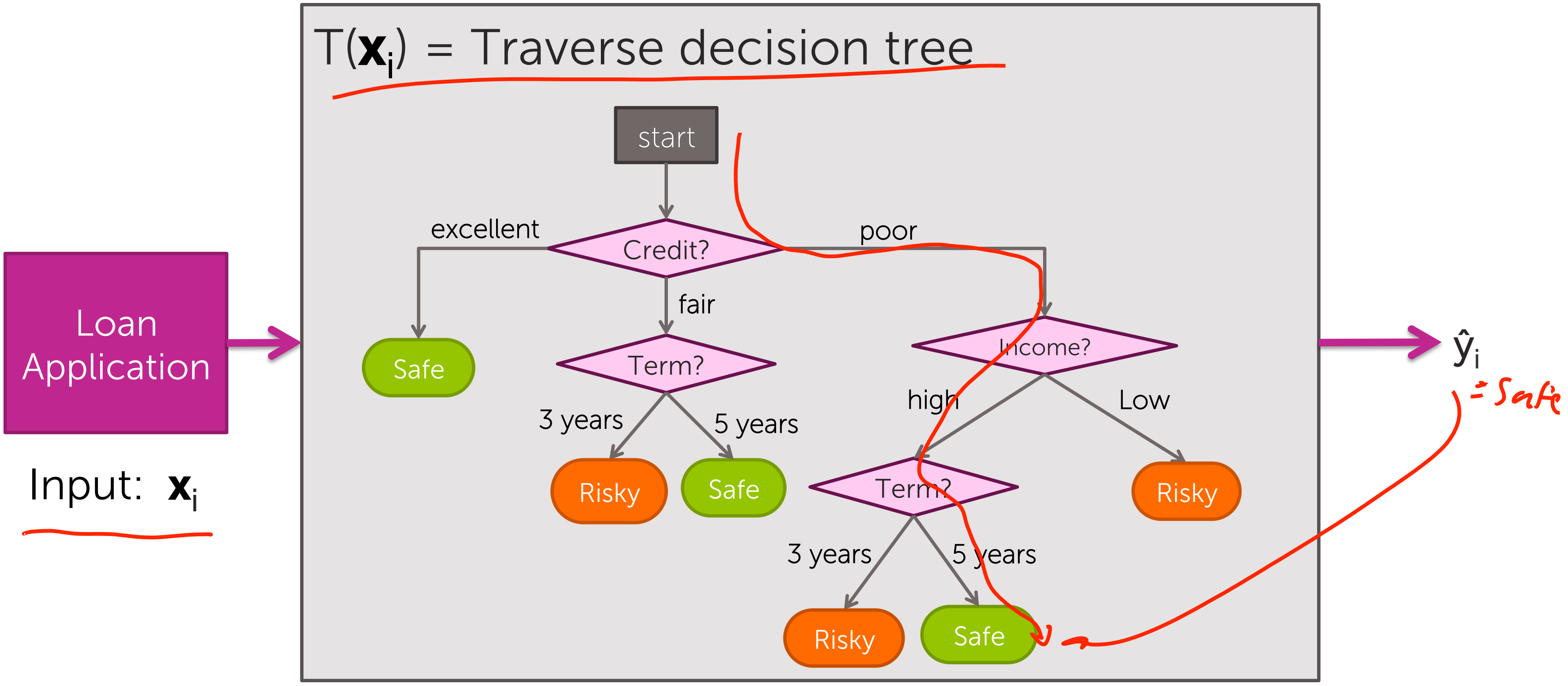
3 year loans with high income & poor credit history are **risky**

Scoring a loan application

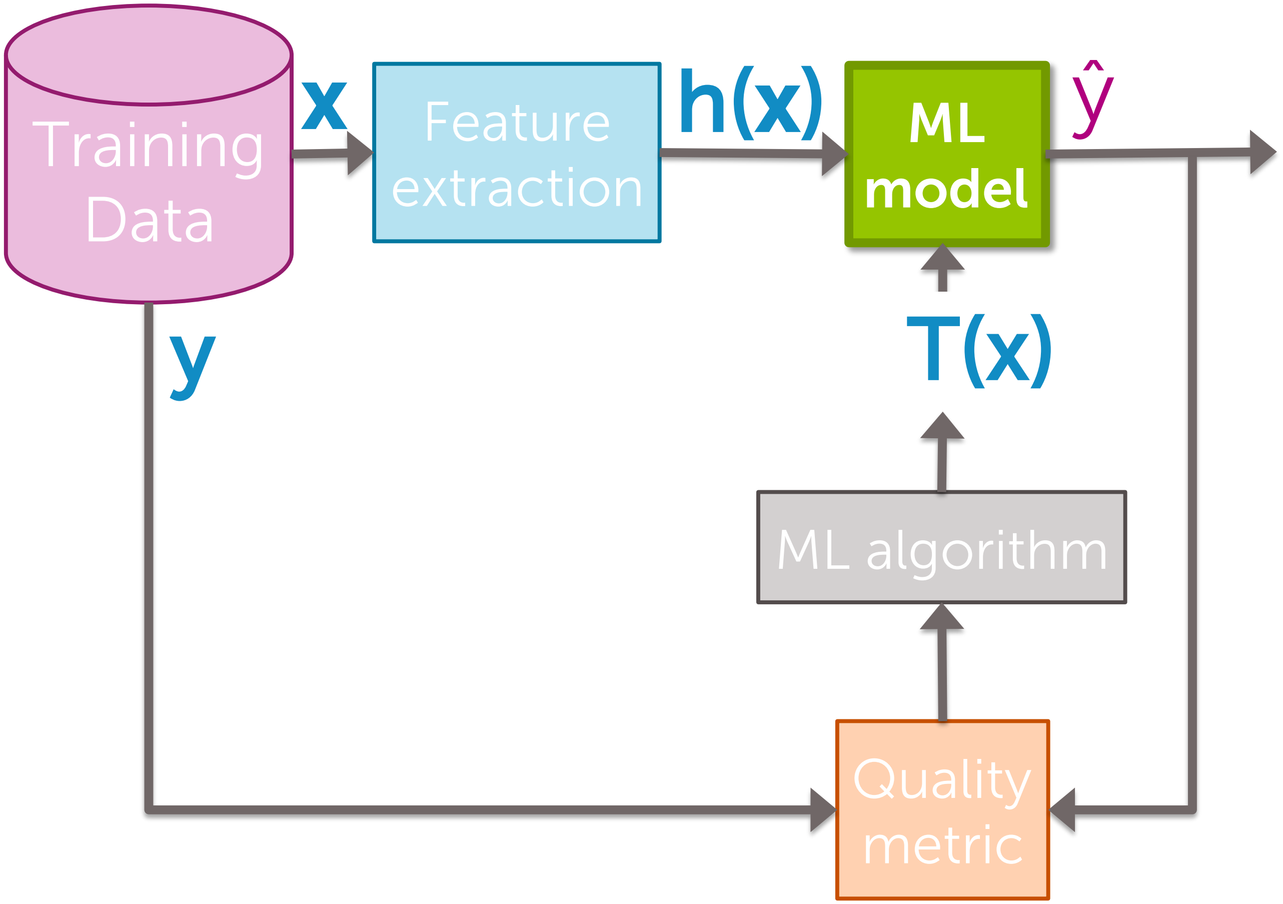
$\mathbf{x}_i$ = (Credit = poor, Income = high, Term = 5 years)



Decision tree model

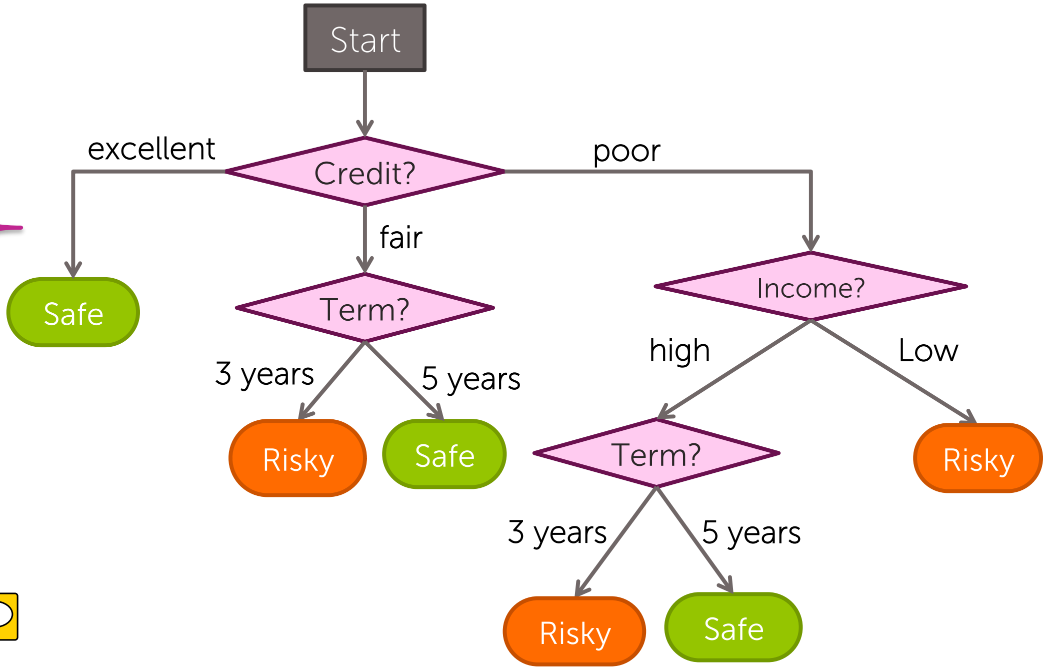


Decision tree learning task


Training Data
x
Feature extraction
h(x)
ML model
ŷ
y
T(x)
ML algorithm
Quality metric

Learn decision tree from data?

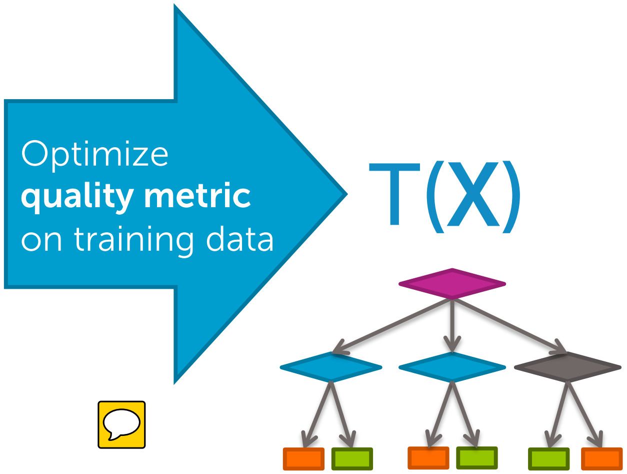
$h_1(x)$	$h_2(x)$	$h_3(x)$	Loan Status
Credit	**Term**	**Income**	**y**
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



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



Optimize **quality metric** on training data

Quality metric: Classification error

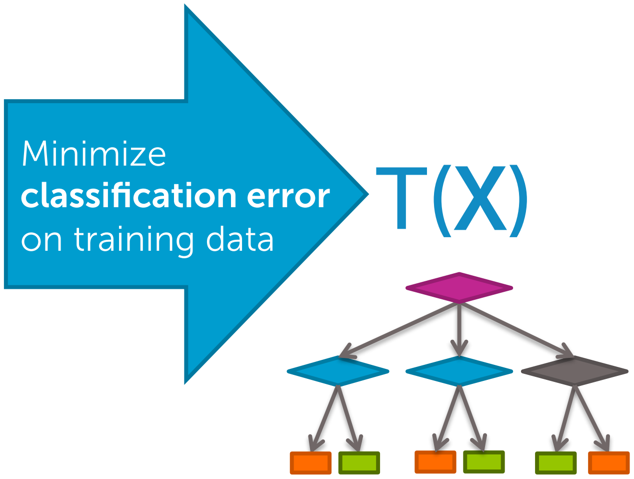
- Error measures fraction of mistakes

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

  - 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

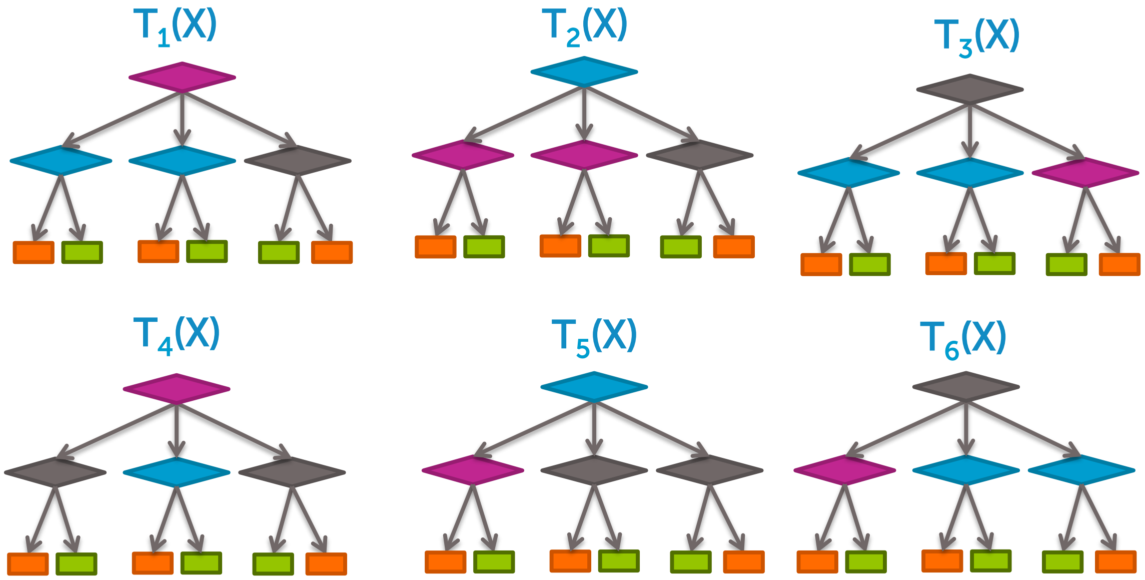


Minimize **classification error** on training data

$T(X)$

How do we find the best tree?

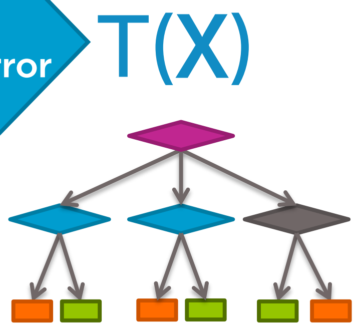
Exponentially large number of possible trees makes decision tree learning hard! *(NP-hard problem)*



Simple (greedy) algorithm finds "good" tree

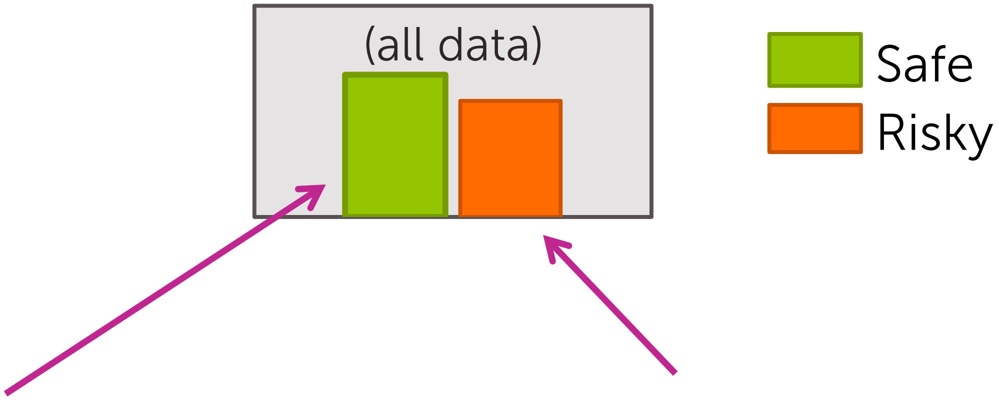
Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Approximately minimize **classification error** on training data



Greedy decision tree learning:
Algorithm outline

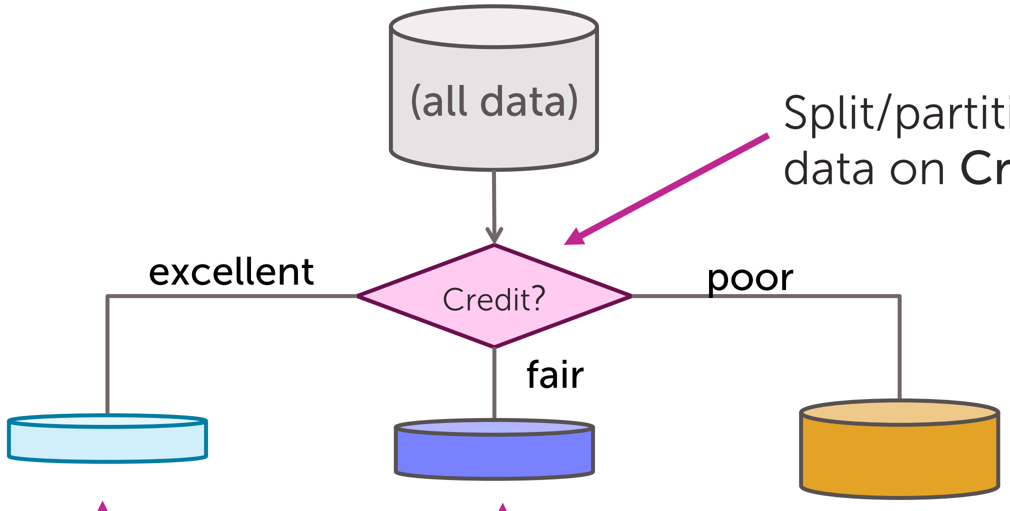
Step 1: Start with an empty tree



Histogram
of y values

All points in the
dataset

Step 2: Split on a feature



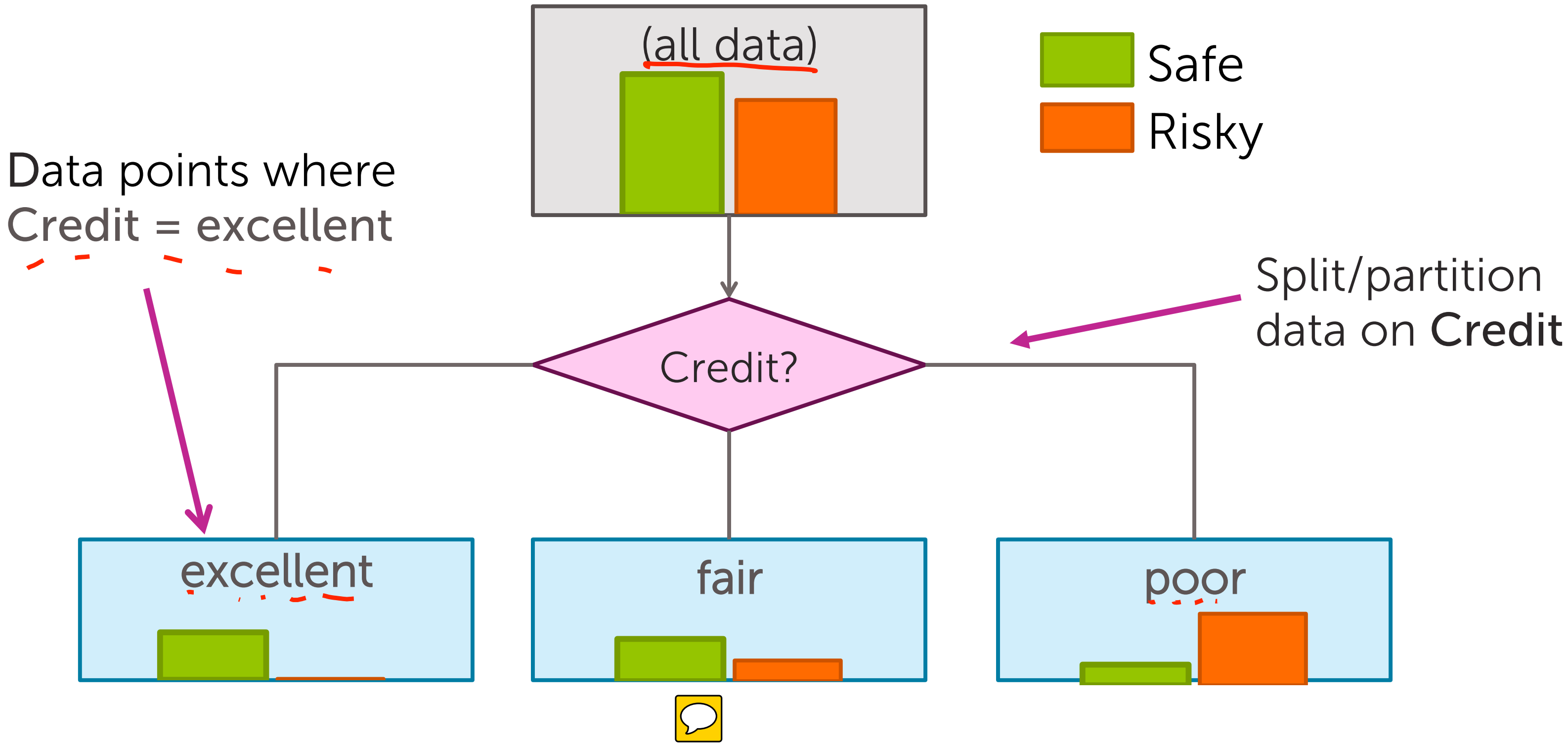
Split/partition data on **Credit**

Subset of data with Credit = excellent

Subset of data with Credit = fair

Subset of data with Credit = poor

Feature split explained



(all data)

🟩 Safe

🟥 Risky

Data points where Credit = excellent

Split/partition data on **Credit**

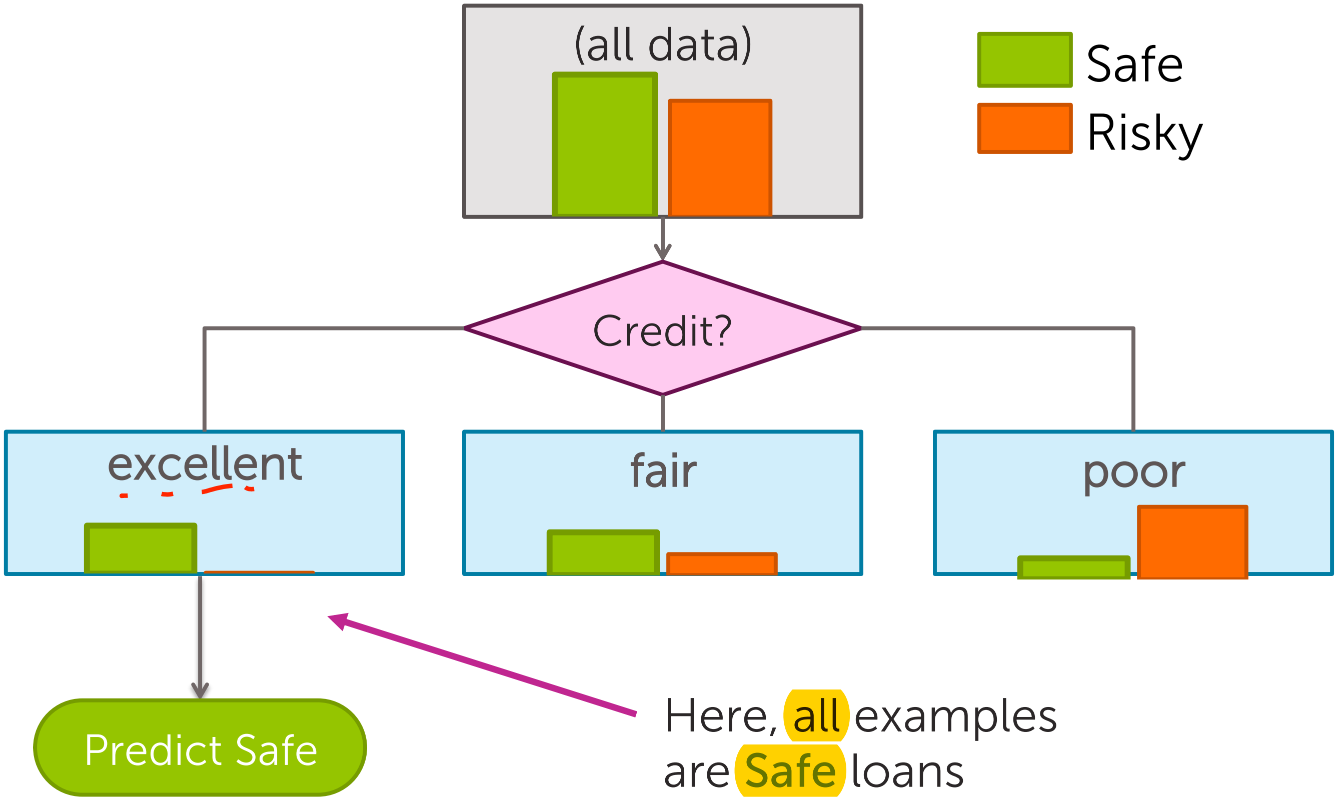
Credit?

excellent

fair

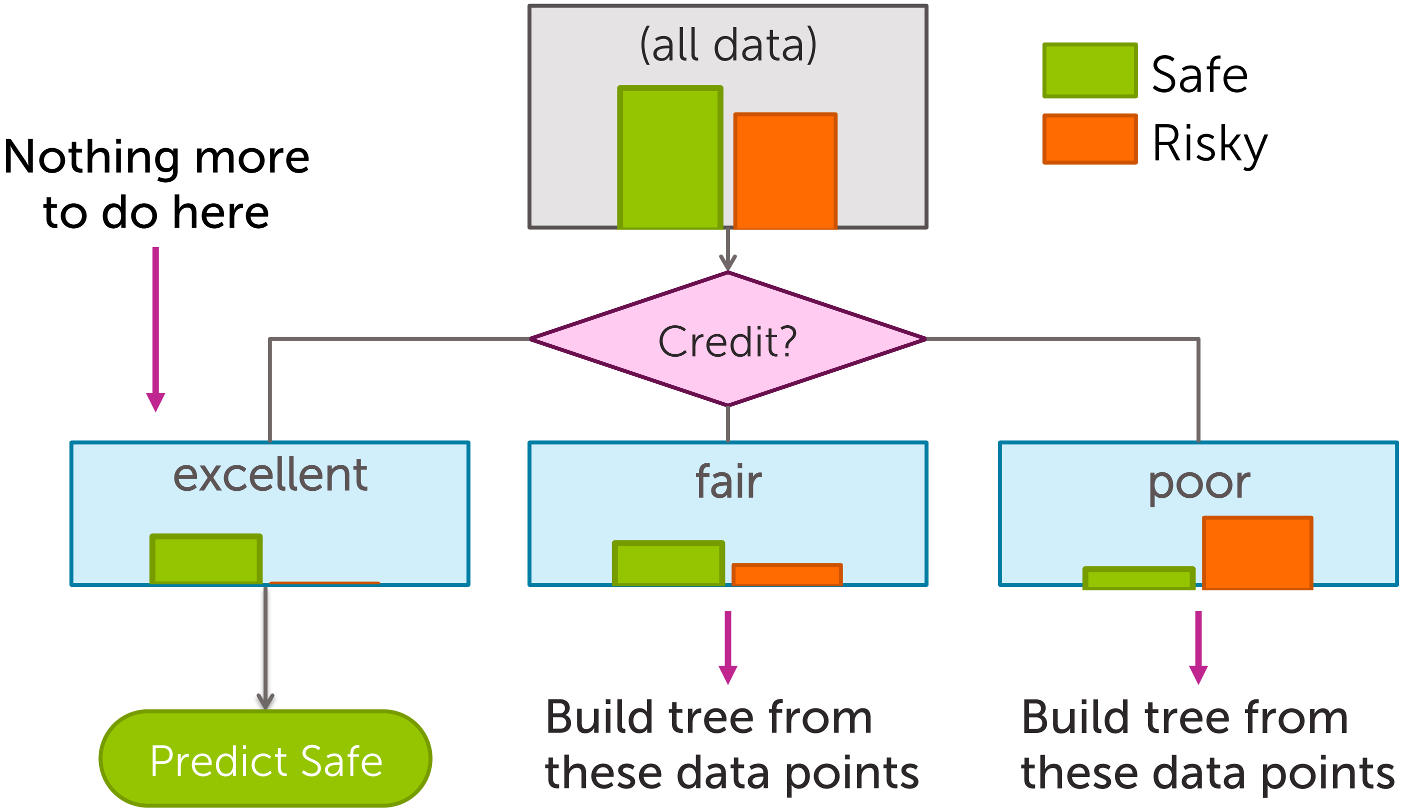
poor

Step 3: Making predictions

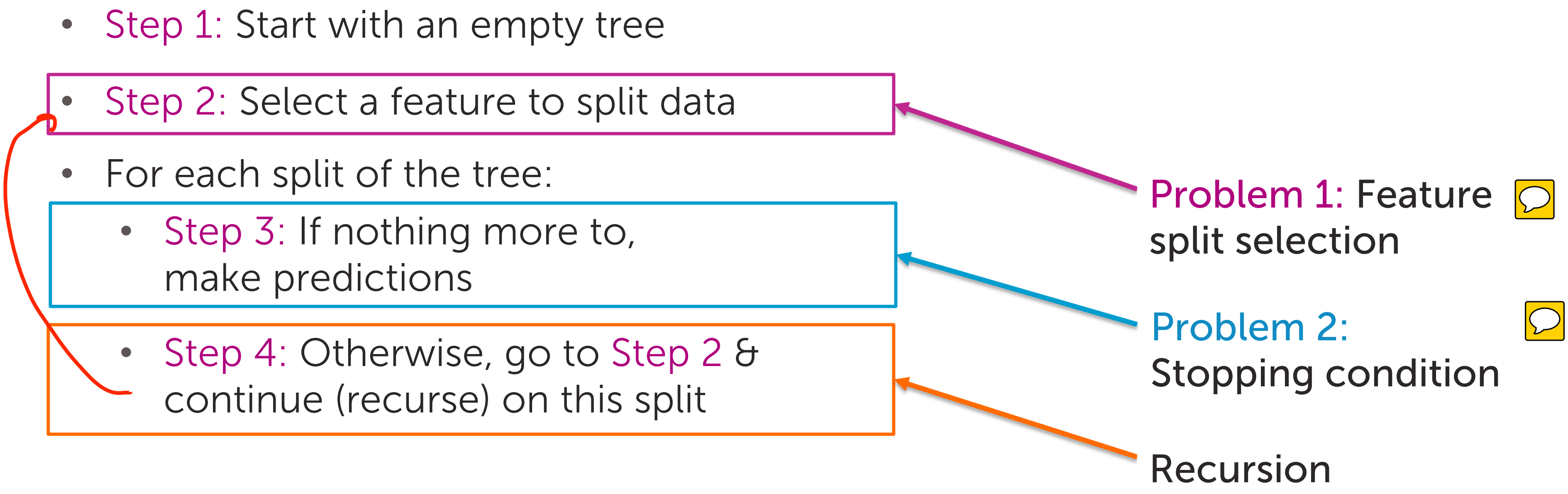


Here, all examples are Safe loans

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

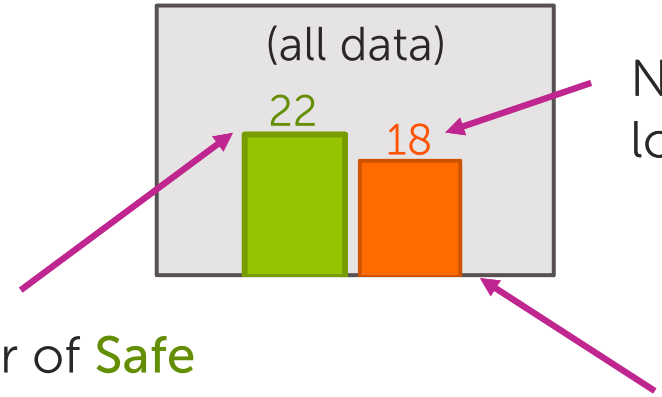
Start with the data

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

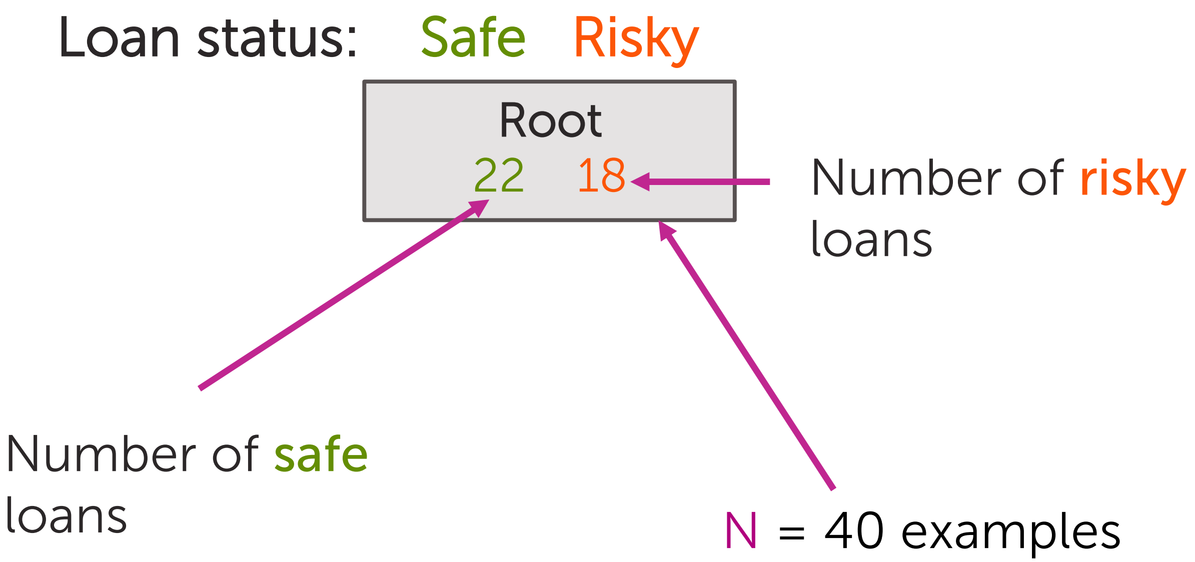


Number of Risky loans

Number of Safe loans

N = 40 examples

Compact visual notation: Root node



Loan status: Safe Risky

Root

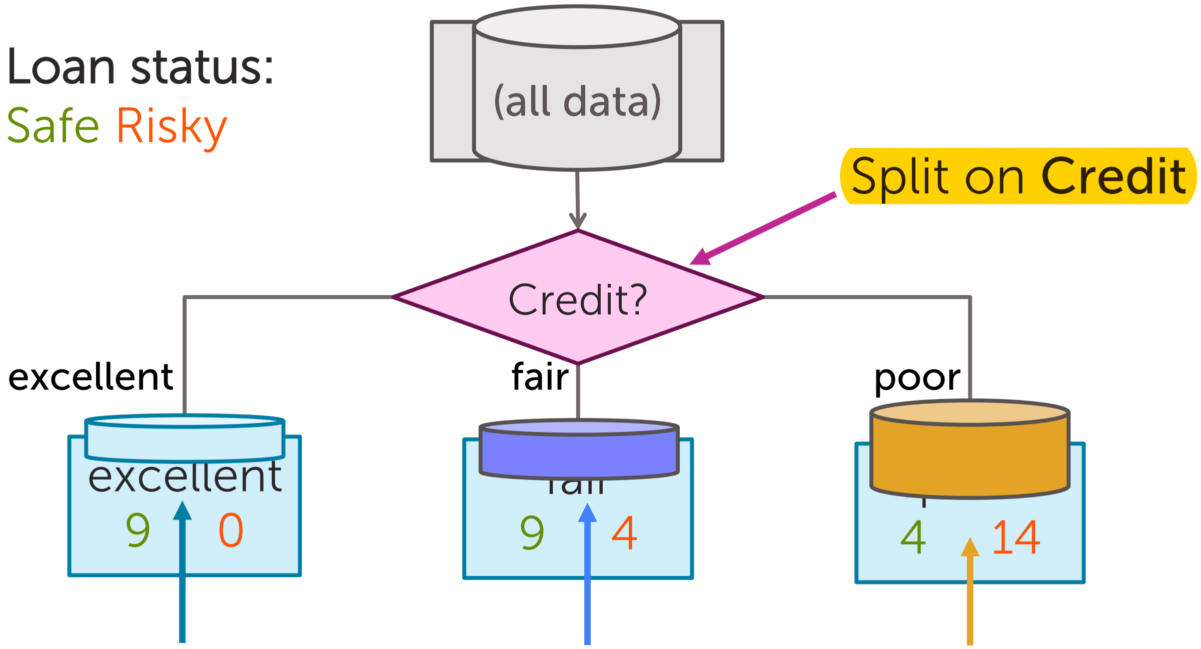
22 18

Number of **risky** loans

Number of **safe** loans

N = 40 examples

Decision stump: Single level tree



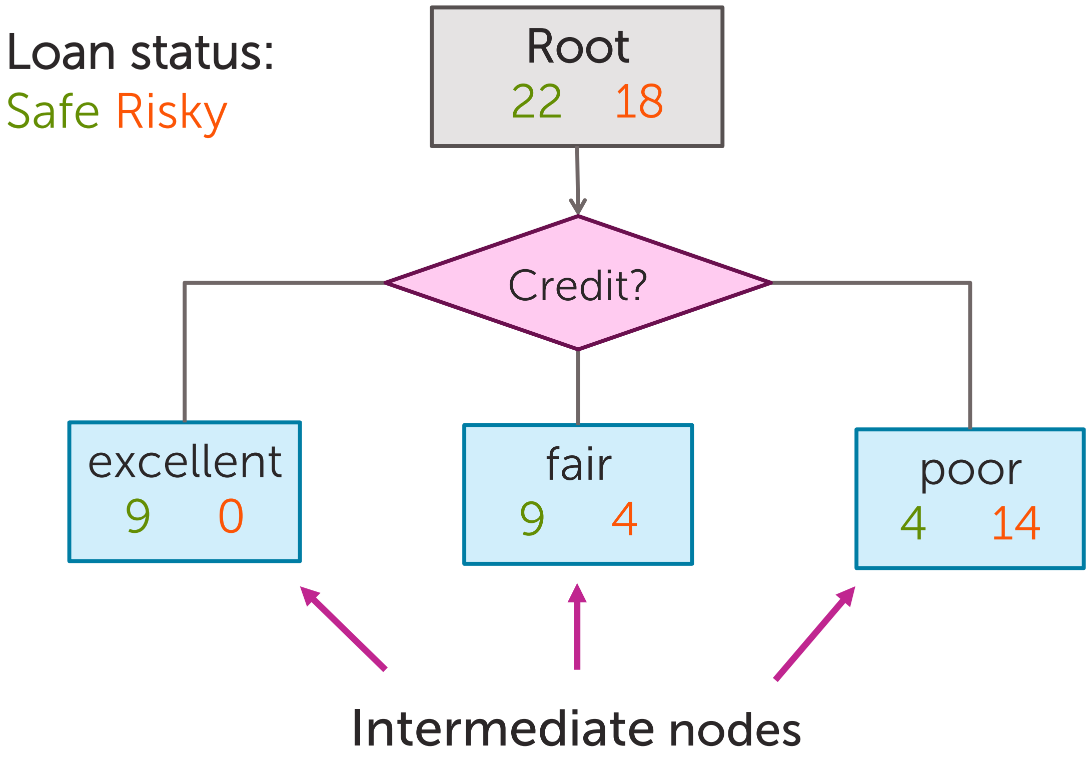
Split on Credit

Subset of data with Credit = excellent

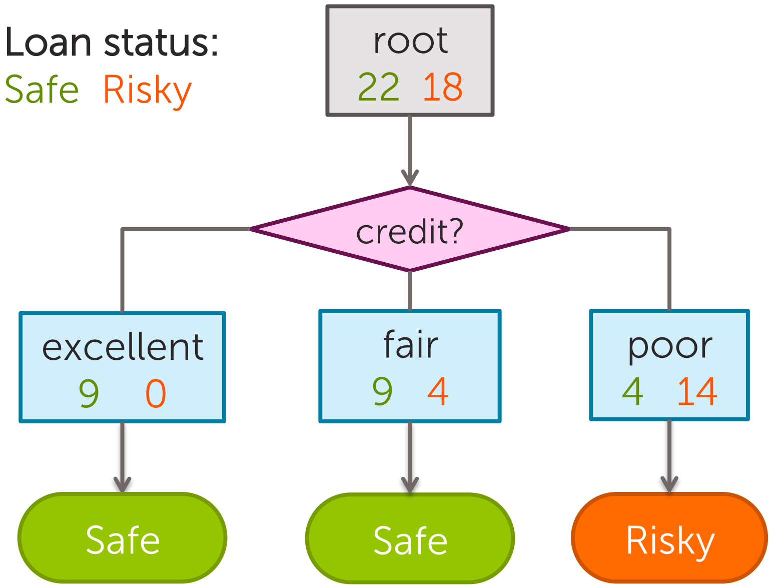
Subset of data with Credit = fair

Subset of data with Credit = poor

Visual Notation: Intermediate nodes



Making predictions with a decision stump

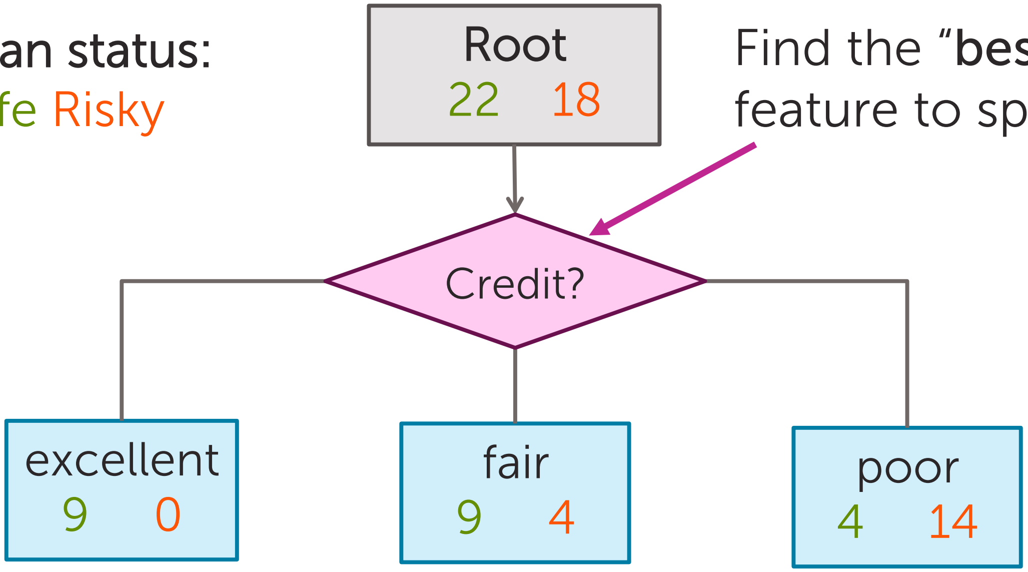


For each intermediate node, set ŷ = **majority value**

Selecting best feature to split on

How do we learn a decision stump?

Loan status:
Safe Risky

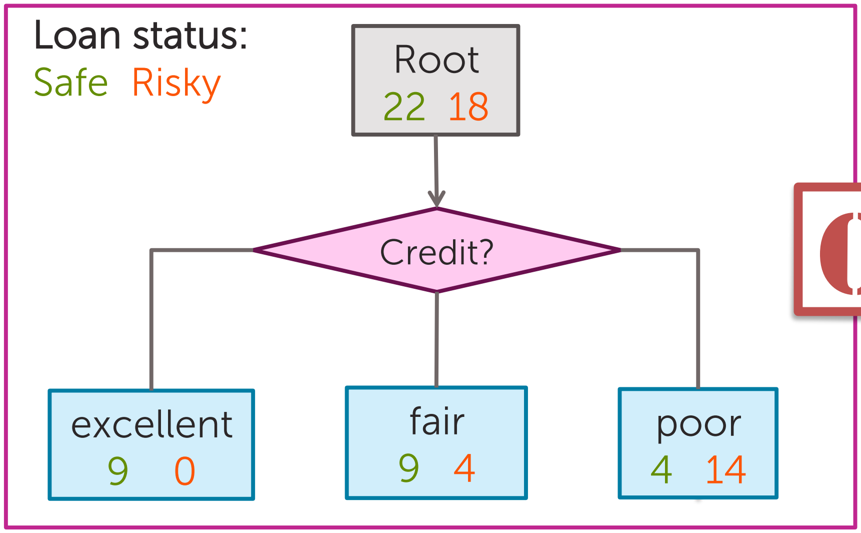


Find the “**best**” feature to split on!

How do we select the best feature?

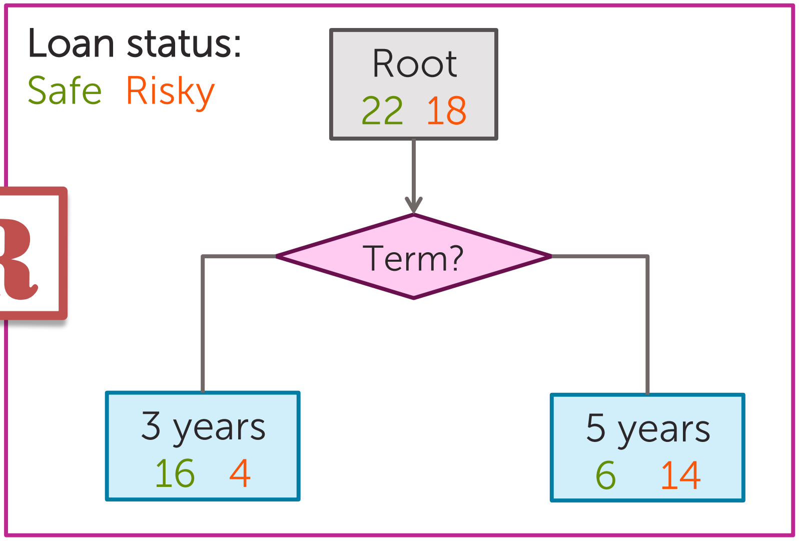
Better?

Choice 1: Split on Credit

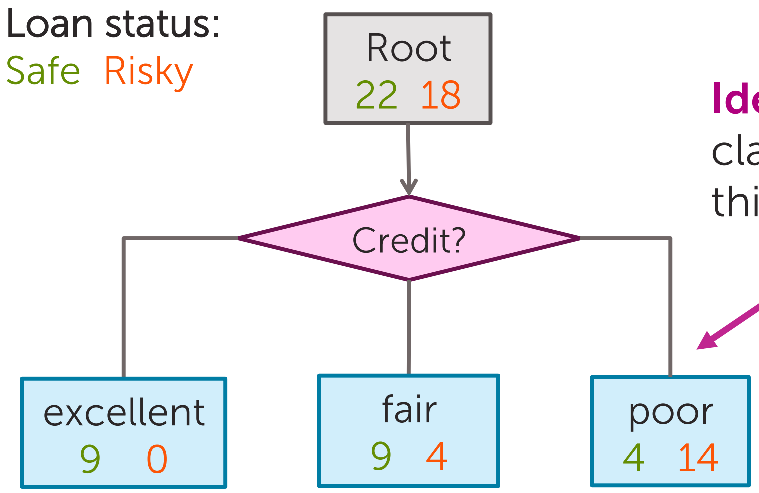


OR

Choice 2: Split on Term



How do we measure effectiveness of a split?

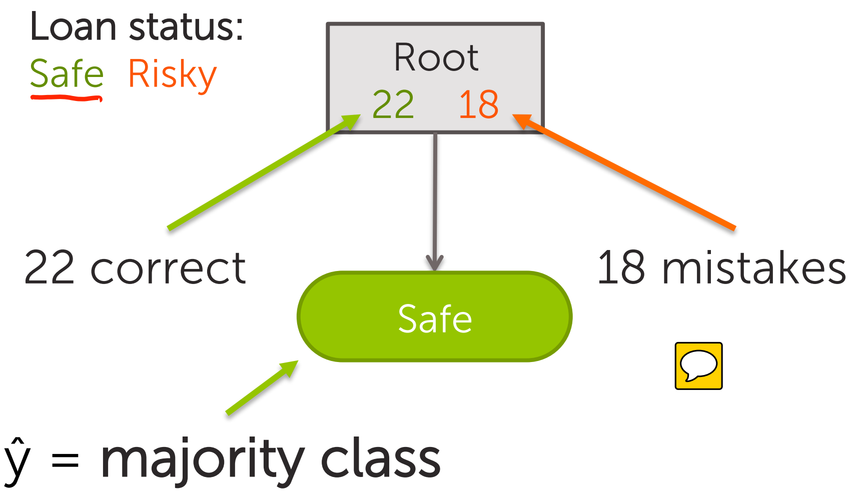


Idea: Calculate classification error of this decision stump

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

Calculating classification error

- **Step 1:** ŷ = class of majority of data in node
- **Step 2:** Calculate classification error of predicting ŷ for this data

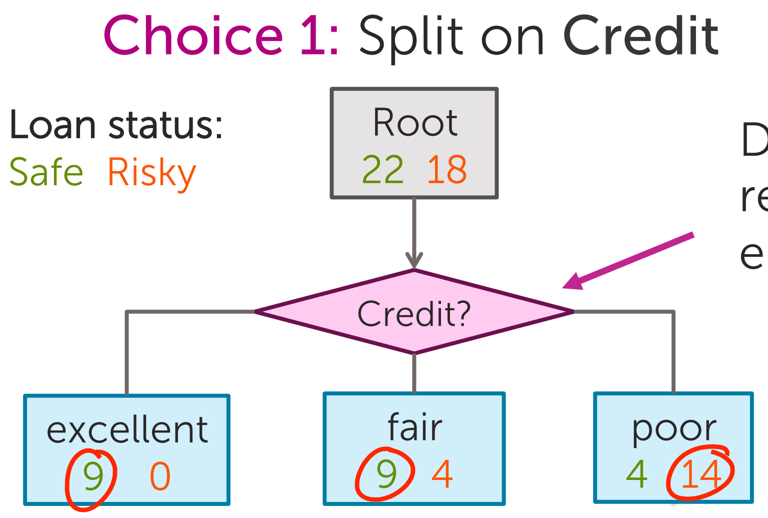


$$\text{Error} = \frac{18}{22+18}$$

$$= 0.45$$

Tree	Classification error
(root)	0.45

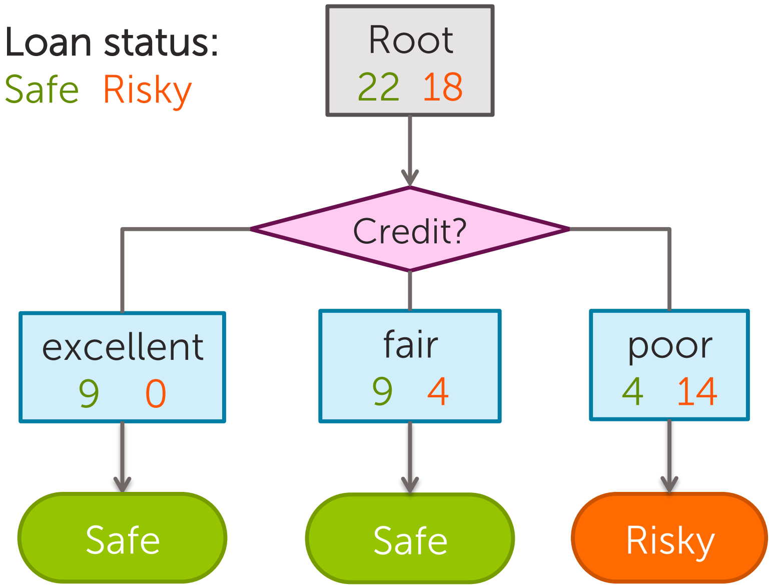
Choice 1: Split on **credit history?**



Does a split on Credit reduce classification error below 0.45?

How good is the split on **Credit**?

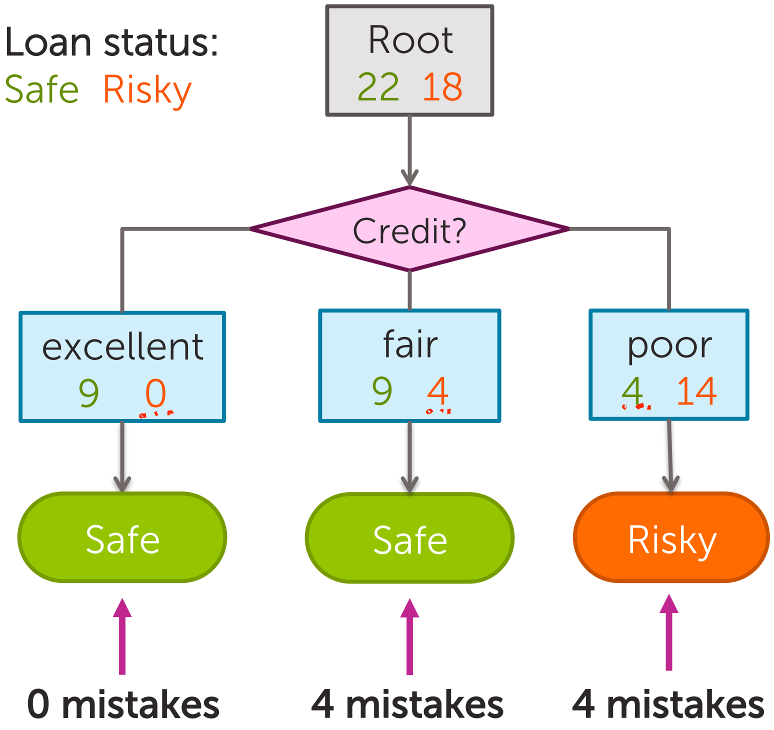
Choice 1: Split on **Credit**



Step 1: For each intermediate node, set ŷ = **majority value**

Split on **Credit**: Classification error

Choice 1: Split on **Credit**

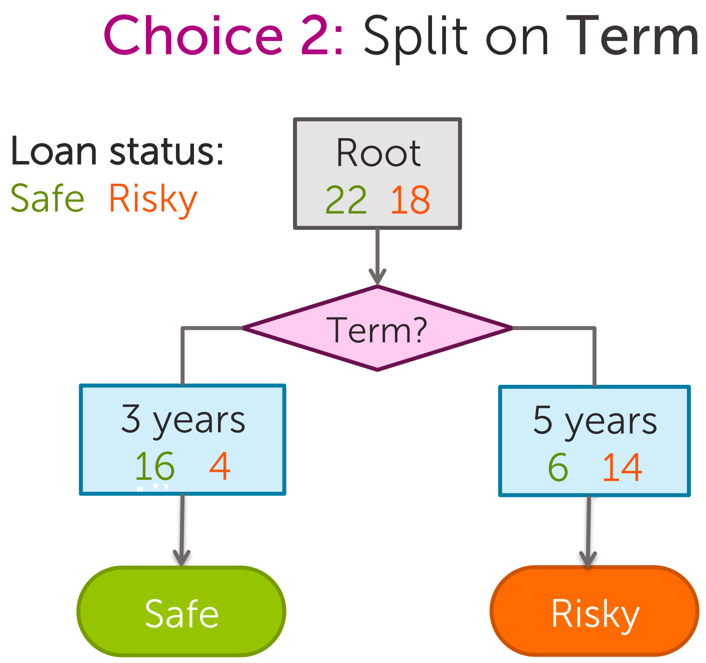


$$\text{Error} = \frac{4+4}{40}$$

$$= 0.20$$

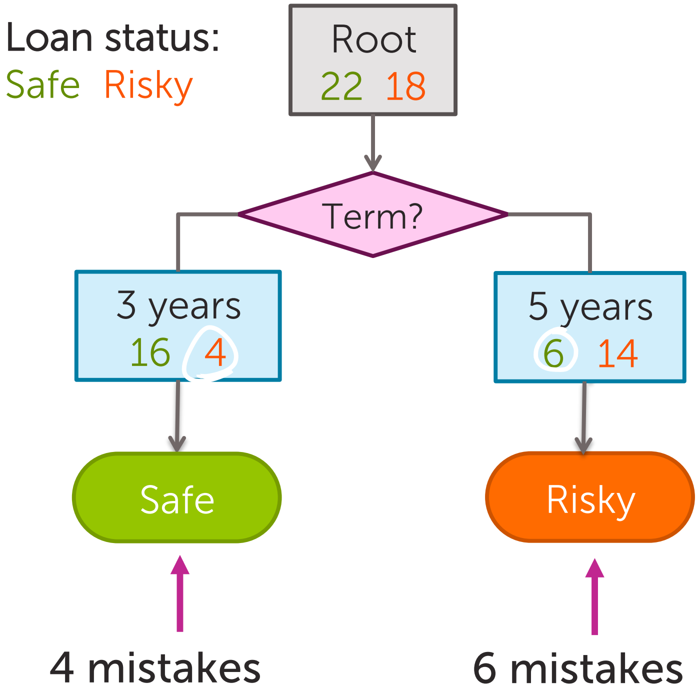
Tree	Classification error
(root)	0.45
Split on **credit**	0.2

Choice 2: Split on Term?



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

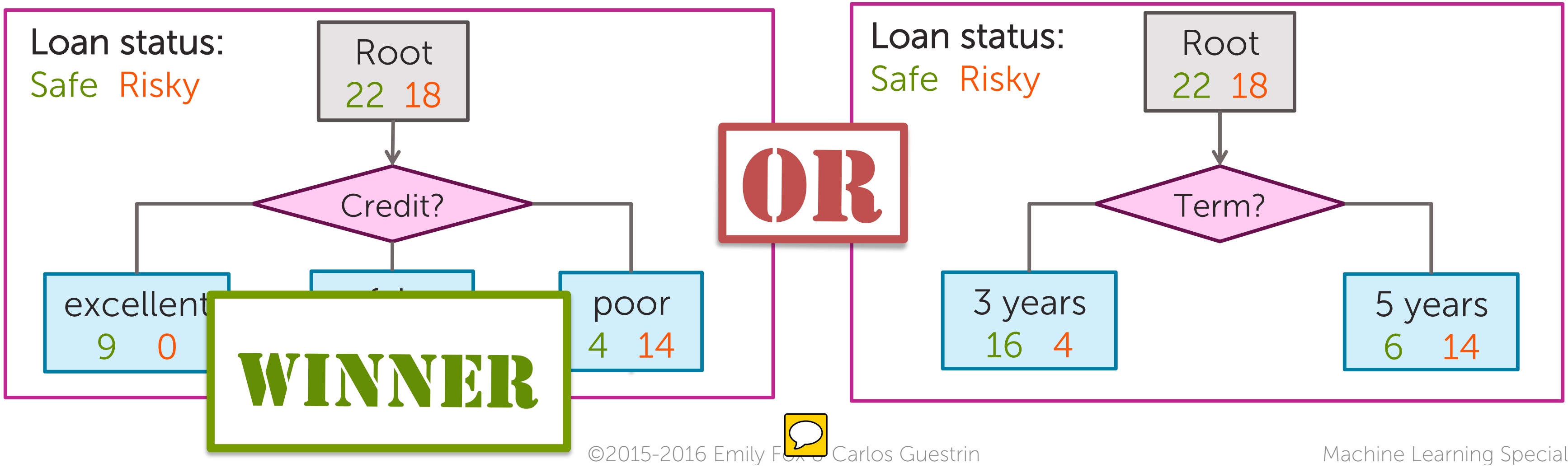
Choice 1 vs Choice 2

Tree	Classification error
(root)	0.45
split on **credit**	0.2
split on **loan term**	0.25

← First split!

Choice 1: Split on **Credit**

Choice 2: Split on **Term**



Feature split selection algorithm

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

credit

Greedy decision tree learning

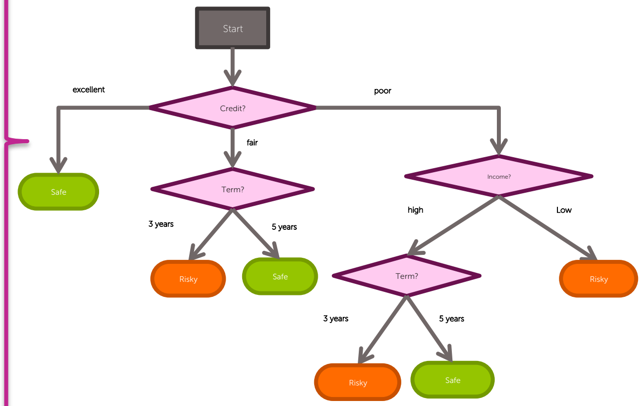
- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
  - Step 3: If nothing more to, make predictions
  - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Pick feature split leading to lowest classification error

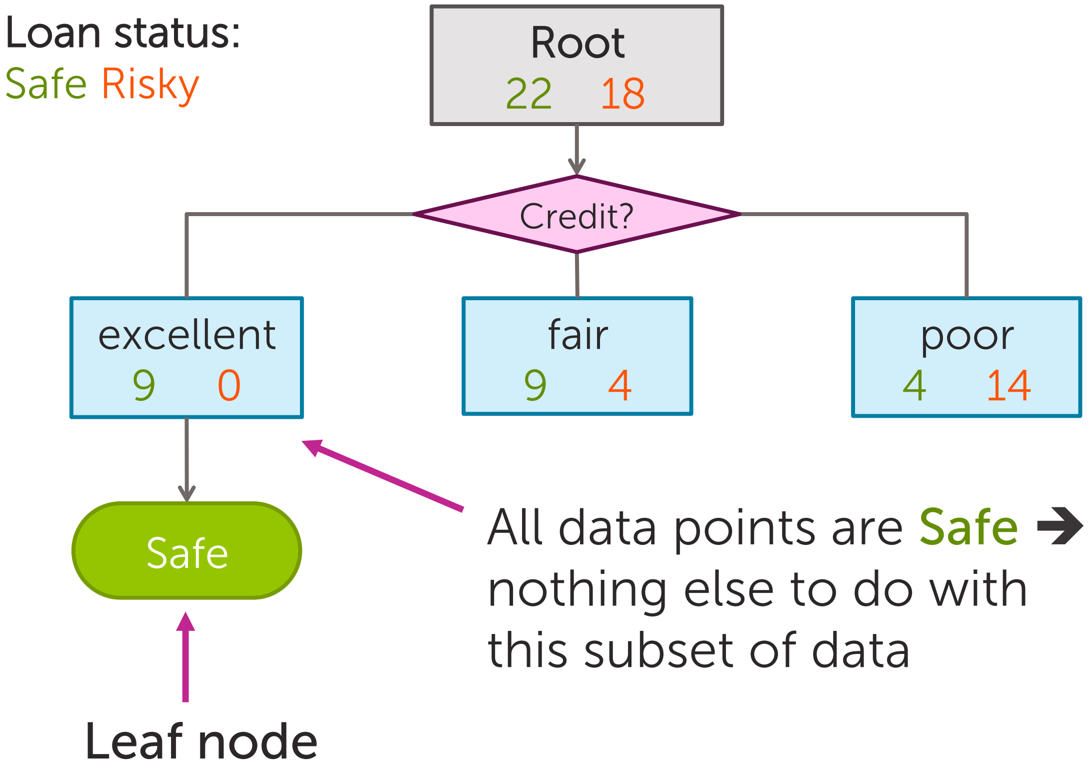
Decision Tree Learning: *Recursion & Stopping conditions*

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

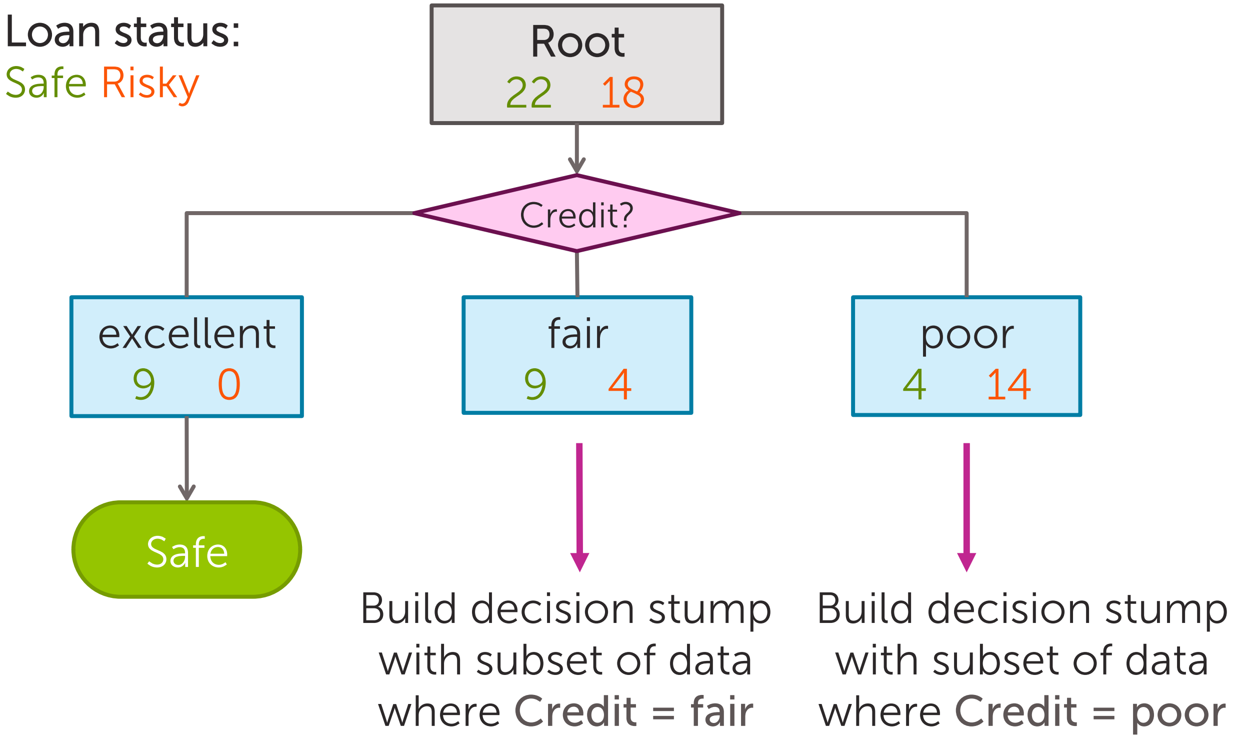


We've learned a decision stump, what next?



All data points are Safe → nothing else to do with this subset of data

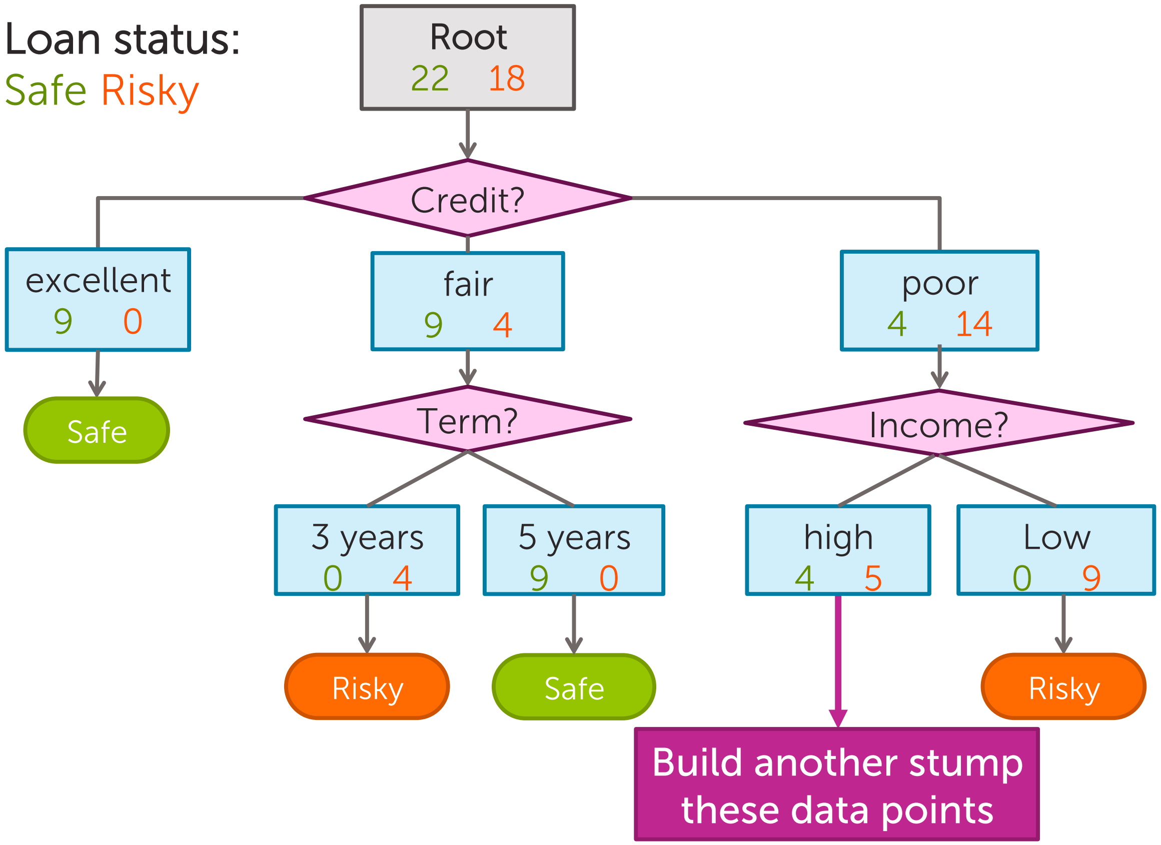
Tree learning = Recursive stump learning



Build decision stump with subset of data where Credit = fair

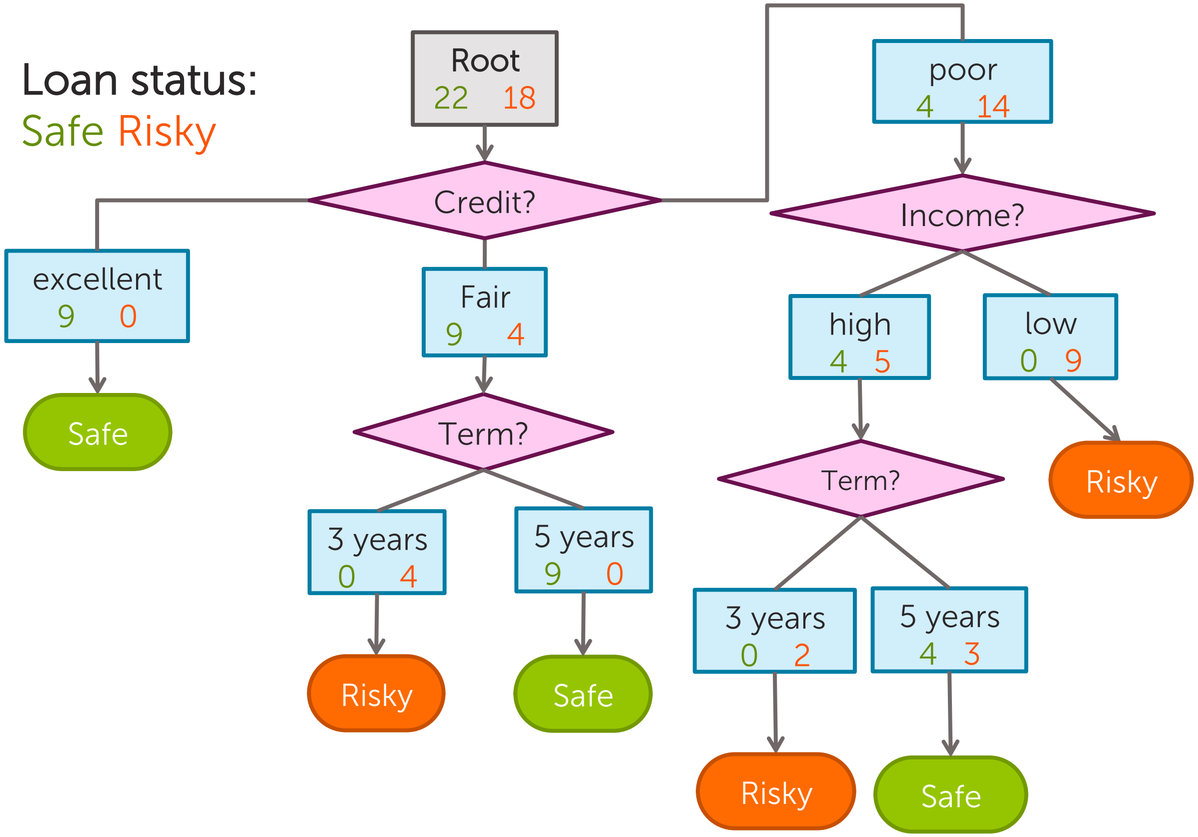
Build decision stump with subset of data where Credit = poor

Second level

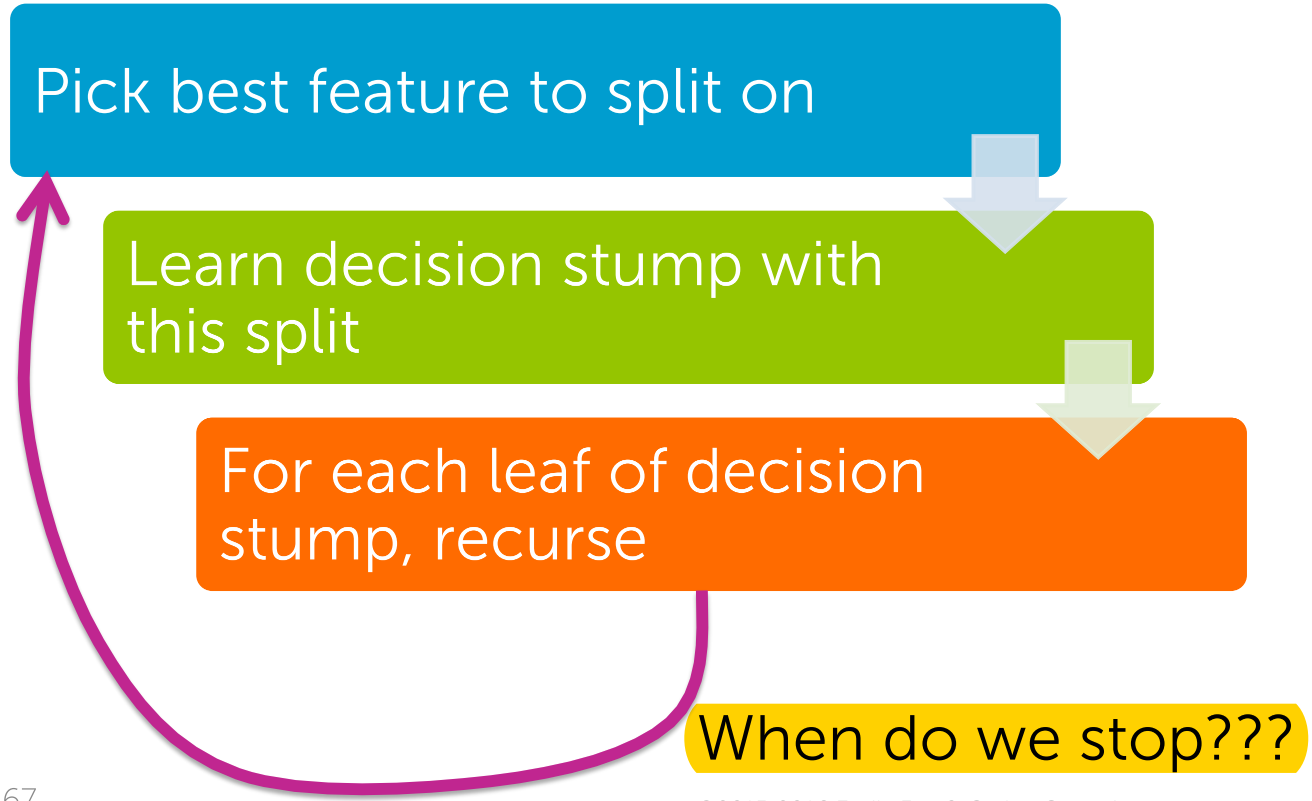


Final decision tree

Loan status:
Safe Risky



Simple greedy decision tree learning



Pick best feature to split on

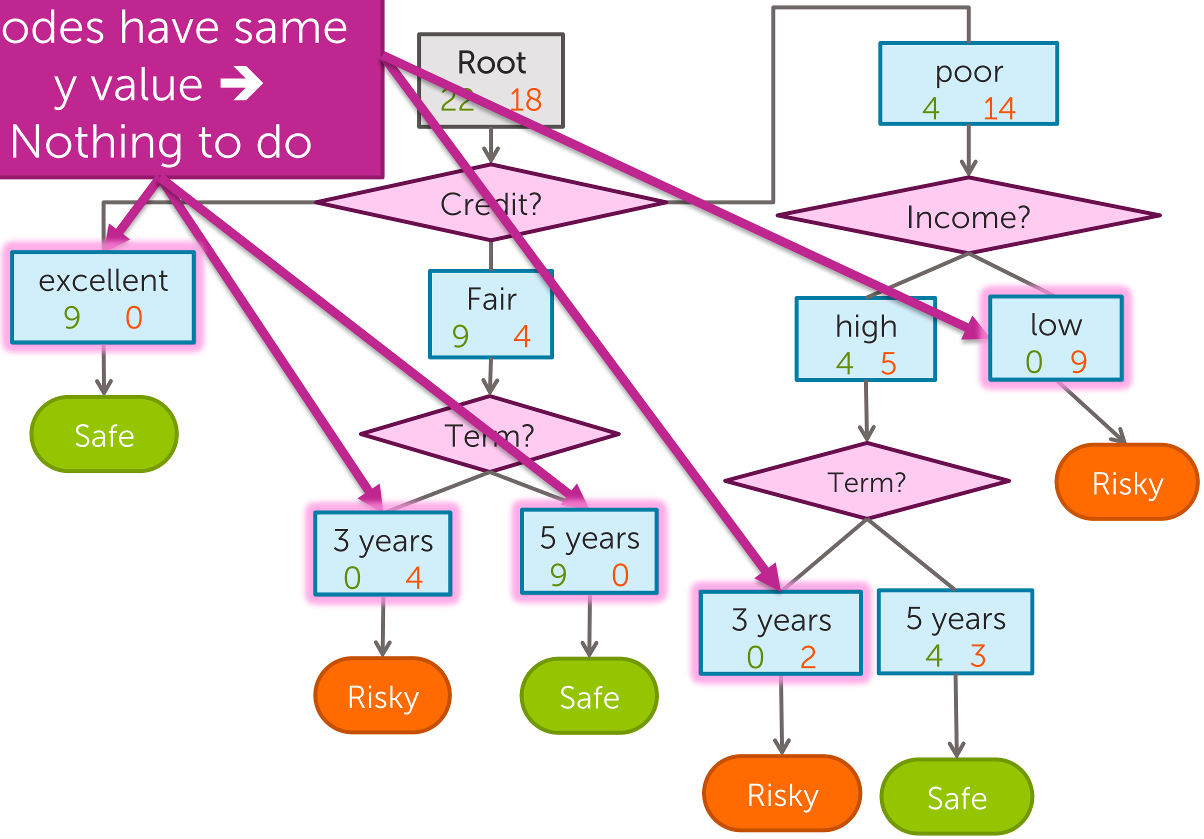
Learn decision stump with this split

For each leaf of decision stump, recurse

When do we stop???

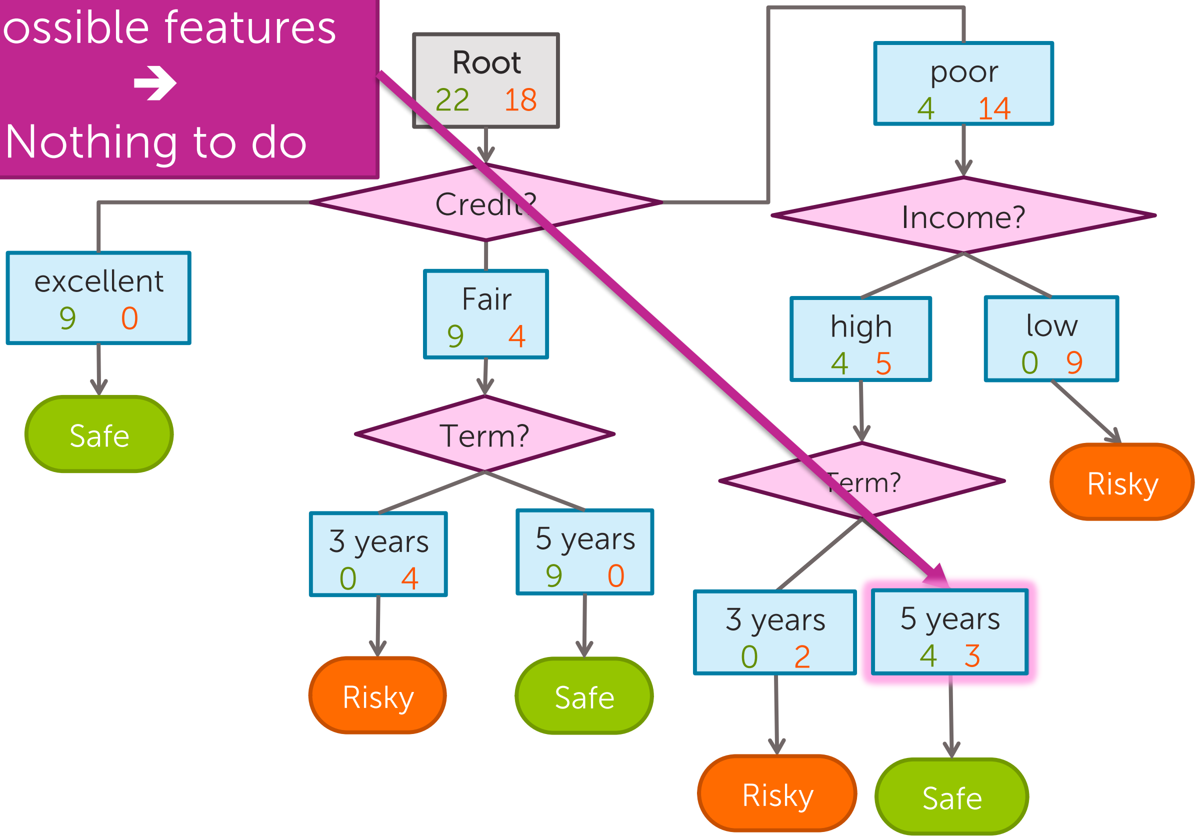
Stopping condition 1: All data agrees on y

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

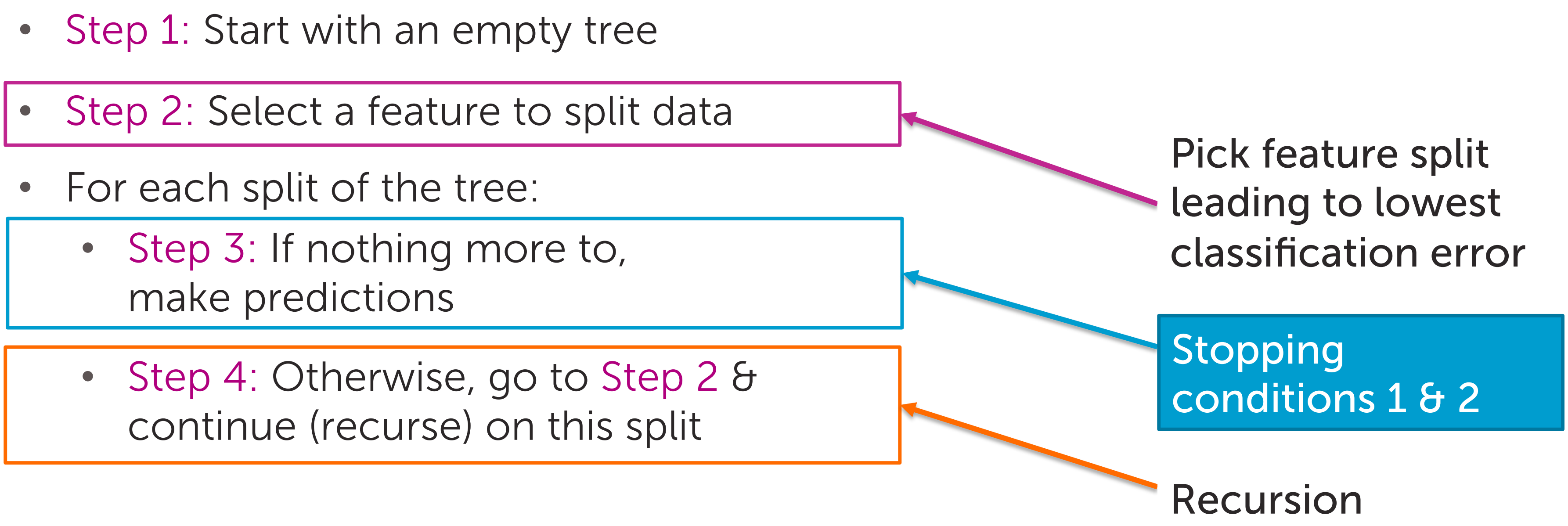


Stopping condition 2: Already split on all features

Already split on all possible features
➔
Nothing to do



Greedy decision tree learning



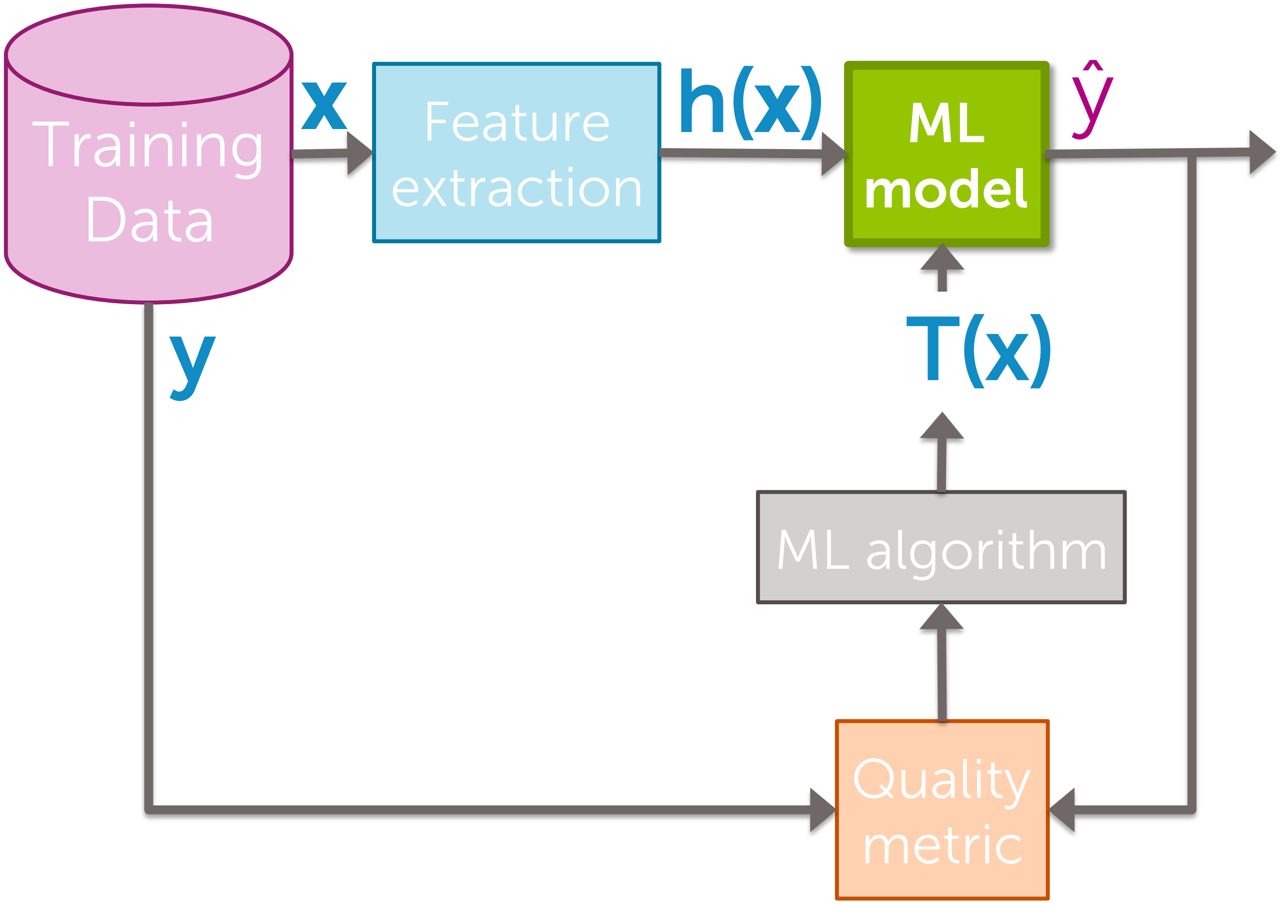
- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
  - Step 3: If nothing more to, make predictions
  - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Pick feature split leading to lowest classification error

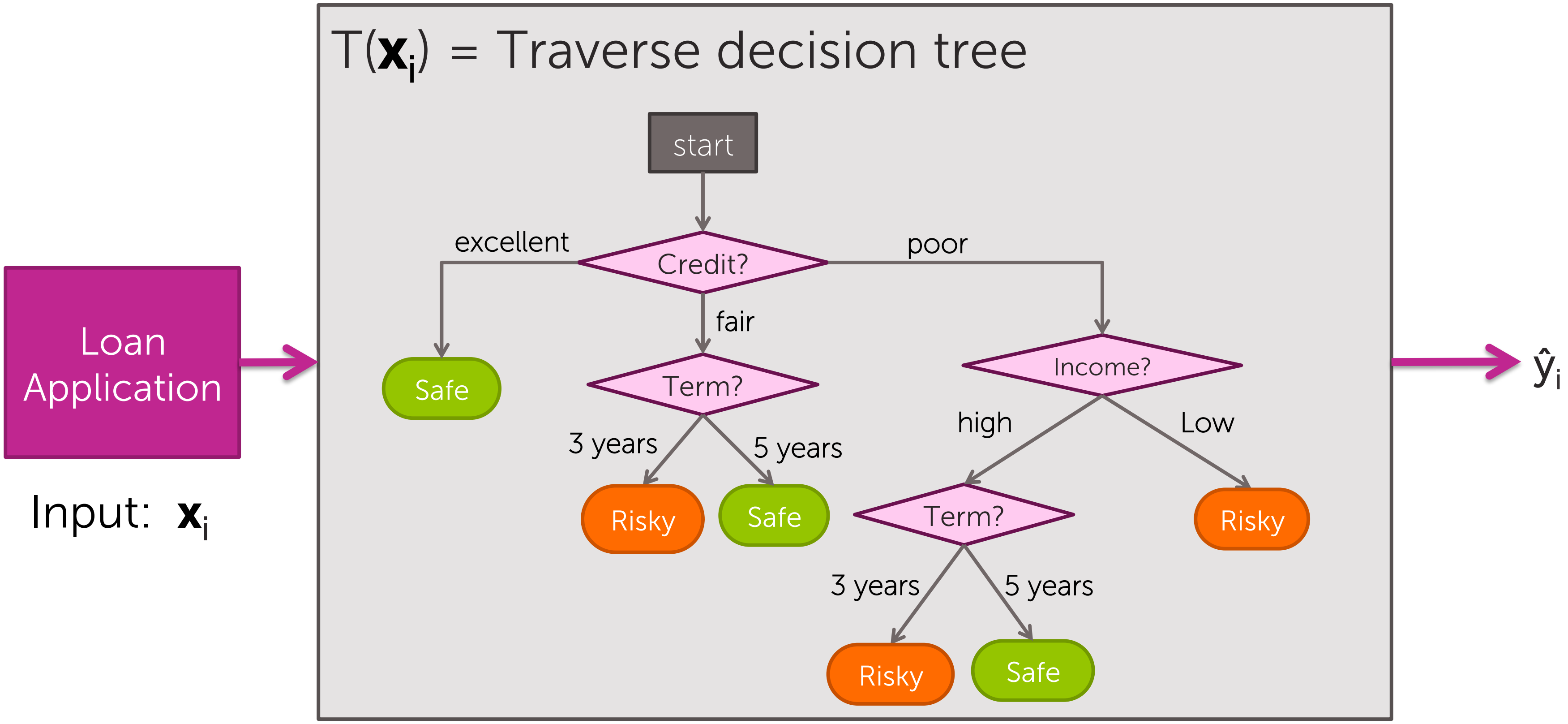
Stopping conditions 1 & 2

Recursion

Predictions with decision trees


Training Data
x
Feature extraction
h(x)
ML model
ŷ
y
T(x)
ML algorithm
Quality metric

Decision tree model



Loan Application

Input: $\mathbf{x}_i$

$T(\mathbf{x}_i)$ = Traverse decision tree

start

Credit?

excellent

Safe

fair

Term?

3 years

Risky

5 years

Safe

poor

Income?

high

Term?

3 years

Risky

5 years

Safe

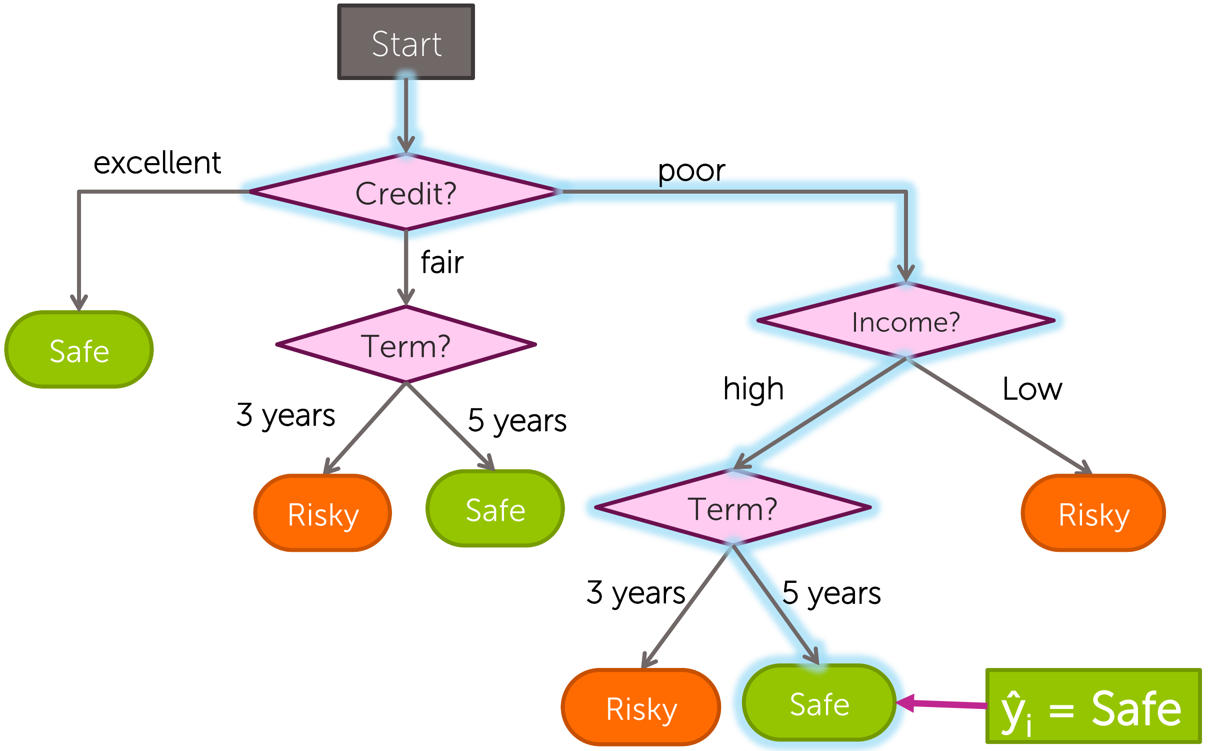
Low

Risky

$\hat{y}_i$

Traversing a decision tree

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



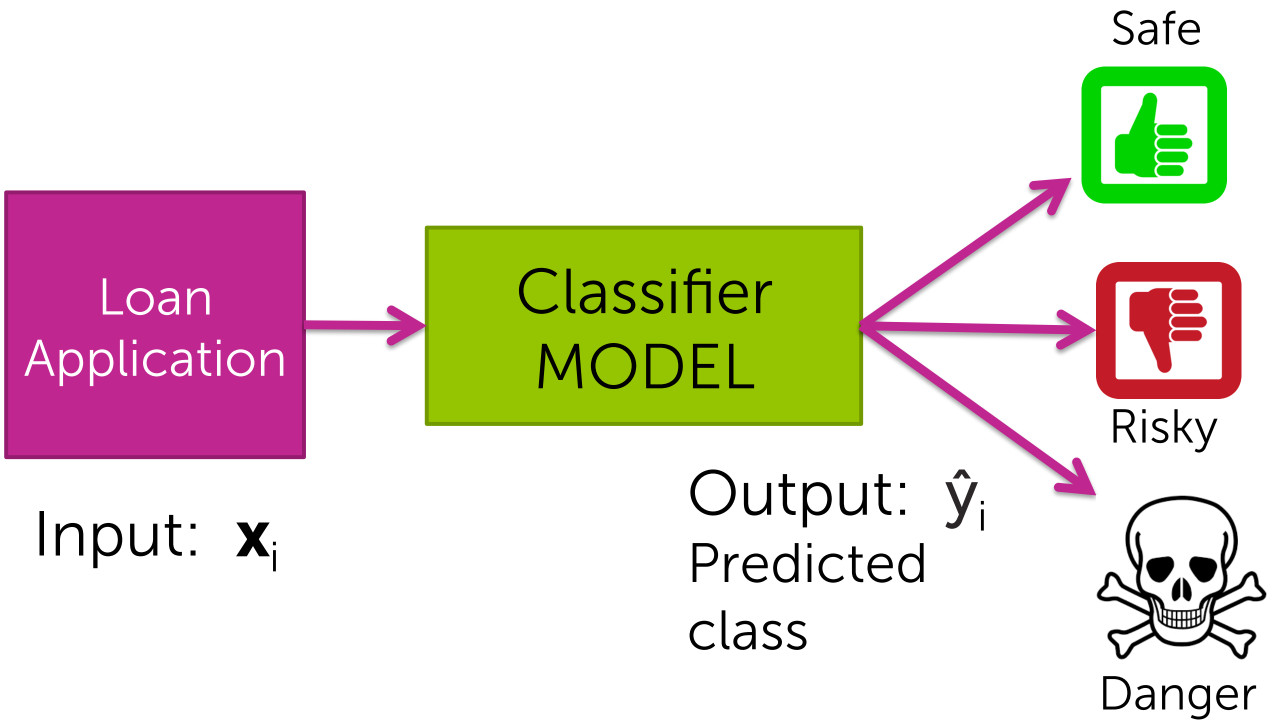
Decision tree prediction algorithm

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



Input: $\mathbf{x}_i$

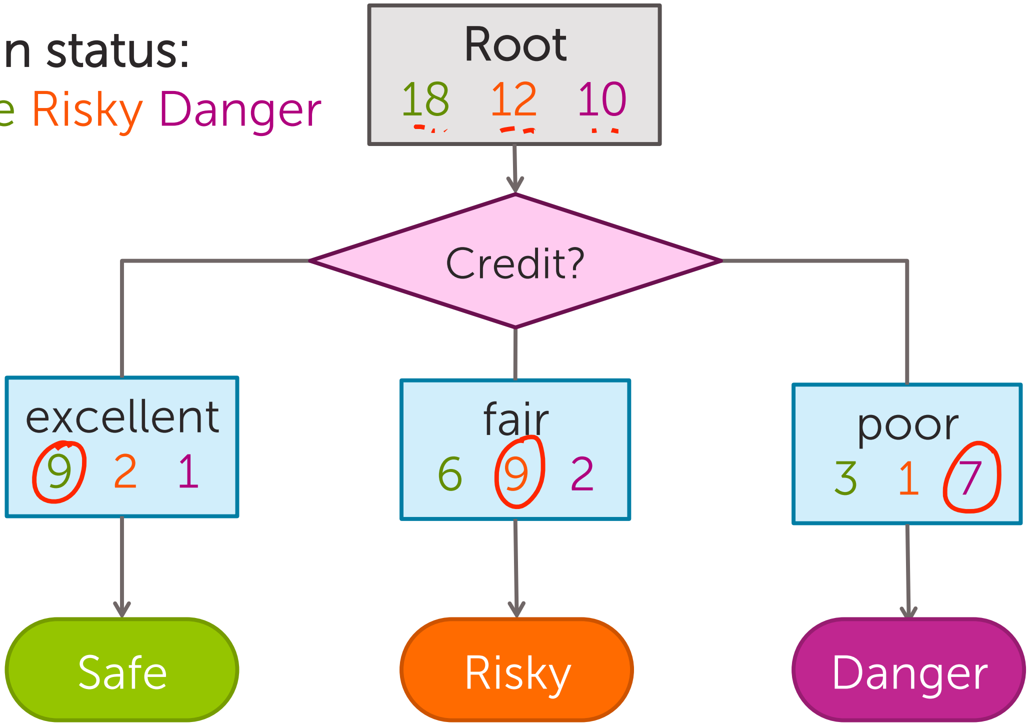
Output: $\hat{y}_i$
Predicted class

Multiclass decision stump

N = 40,
1 feature,
3 classes

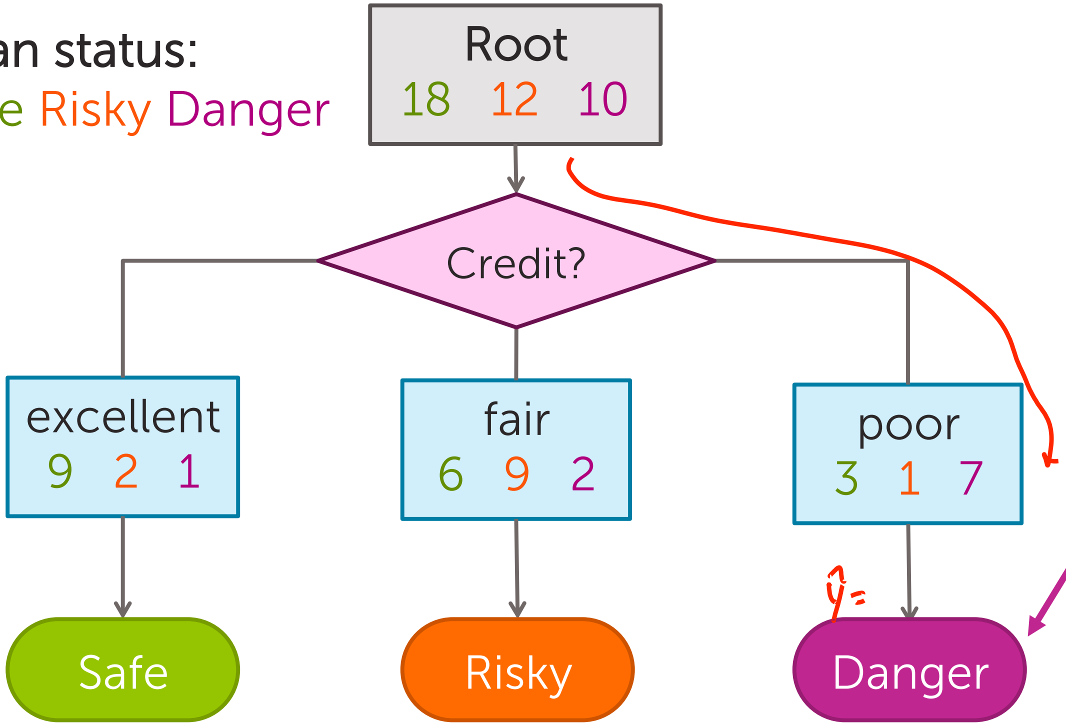
Credit	y
excellent	safe
fair	risky
fair	safe
poor	danger
excellent	risky
fair	safe
poor	danger
poor	safe
fair	safe
…	…

Loan status:
Safe Risky Danger



Predicting probabilities with decision trees

Loan status:
Safe Risky Danger



$$P(y = \textbf{danger} \mid \mathbf{x}) = \frac{7}{3+1+7} = \underline{0.64}$$

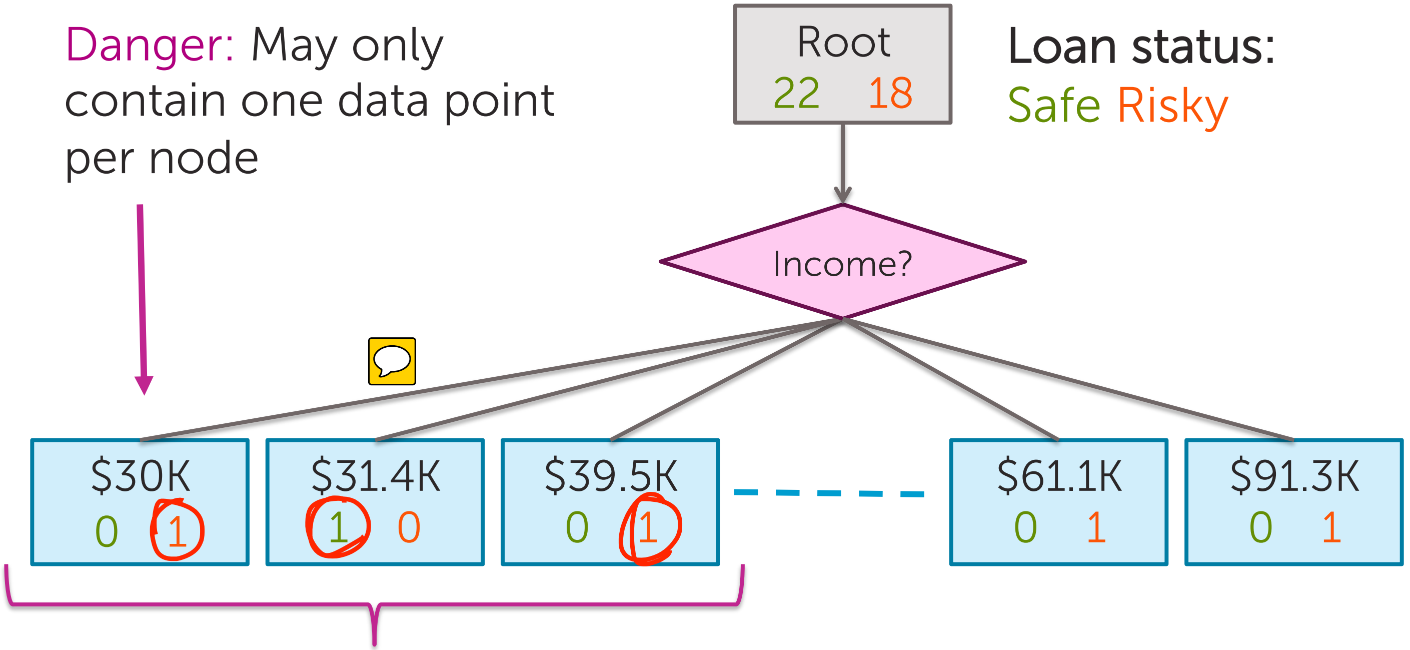
Decision tree learning:
Real valued features

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



Root
22 18

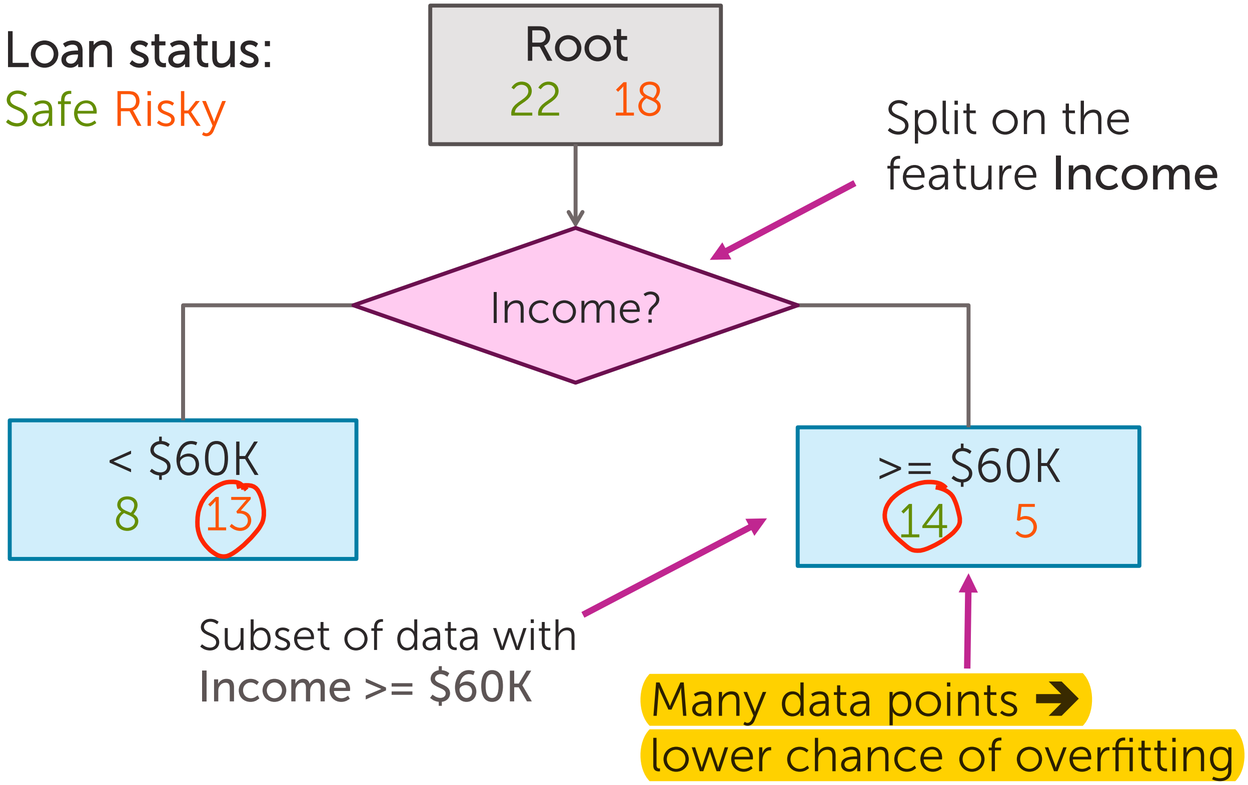
Loan status:
Safe Risky

Income?

$30K	$31.4K	$39.5K	$61.1K	$91.3K
0 1	1 0	0 1	0 1	0 1

Can't trust prediction (overfitting)

Alternative: Threshold split

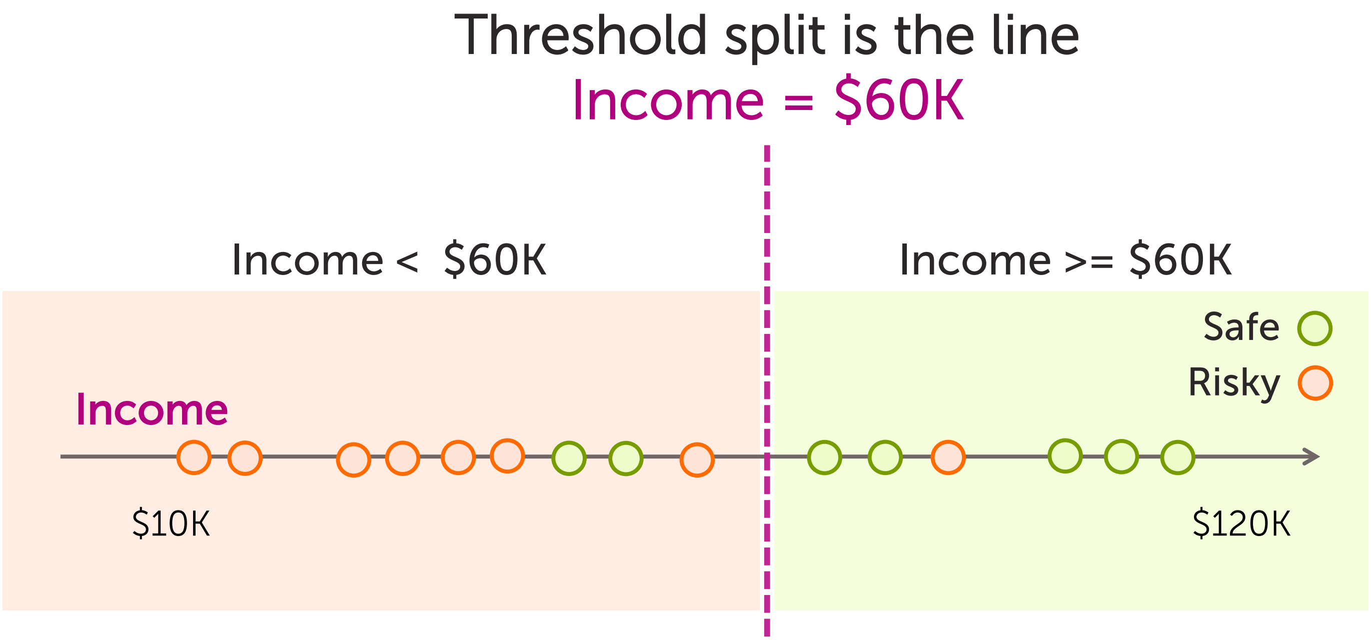


Split on the feature **Income**

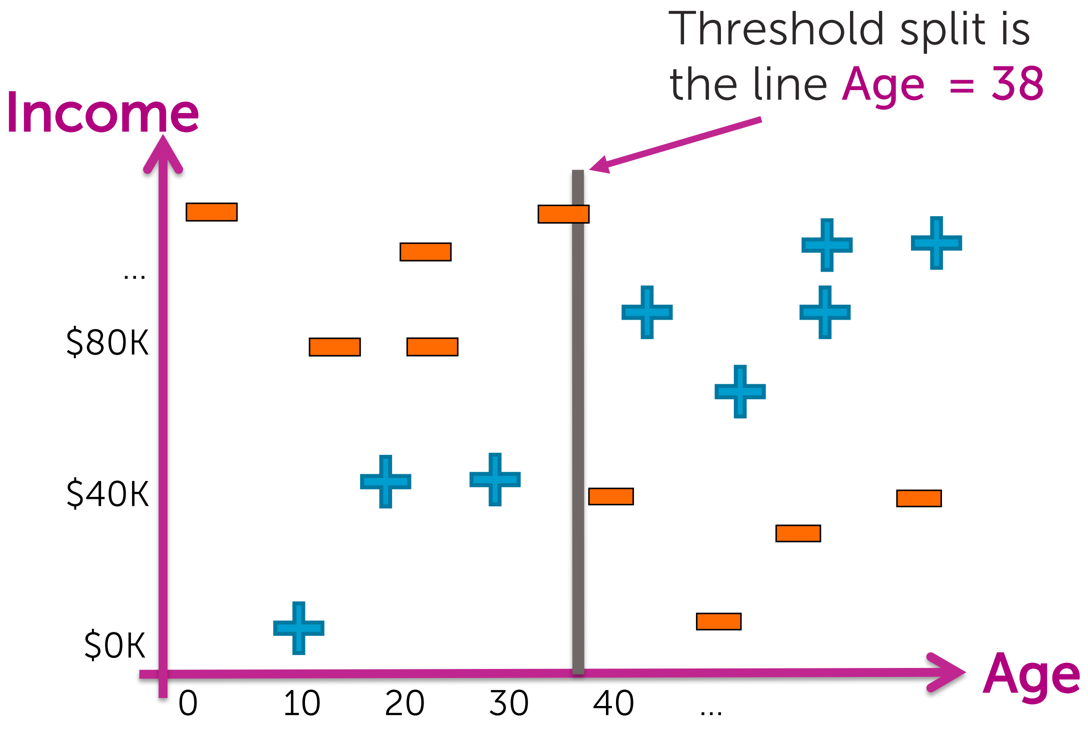
Subset of data with Income >= $60K

Many data points → lower chance of overfitting

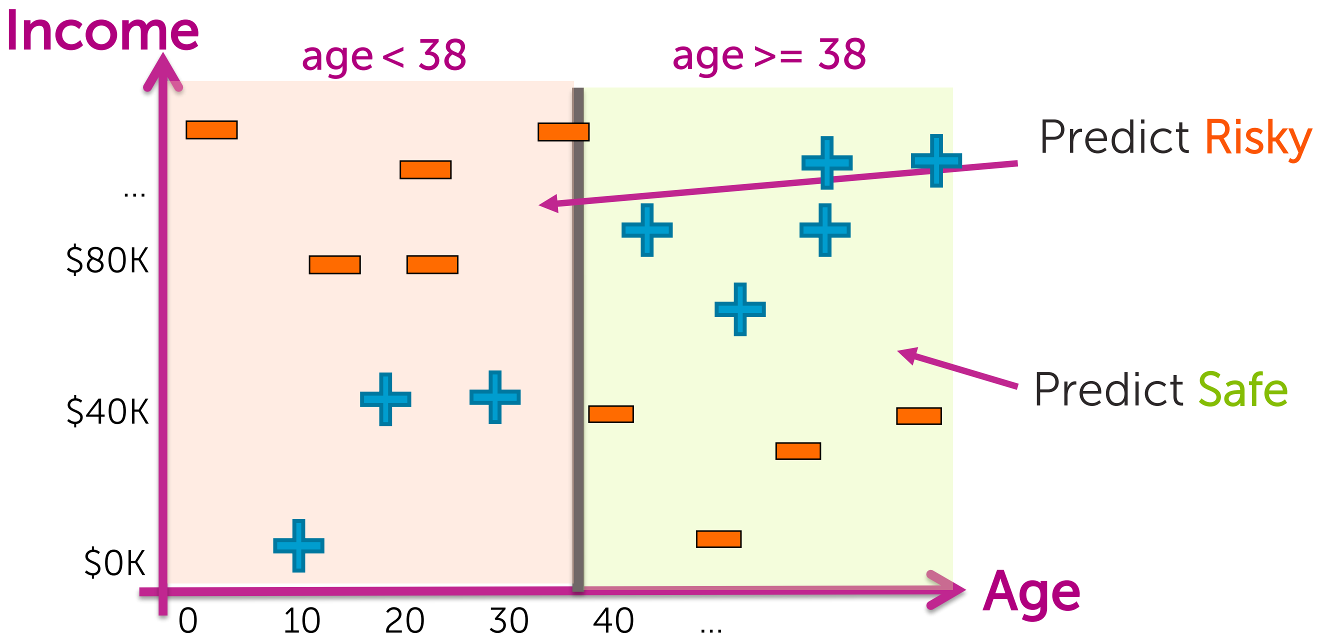
Threshold splits in 1-D



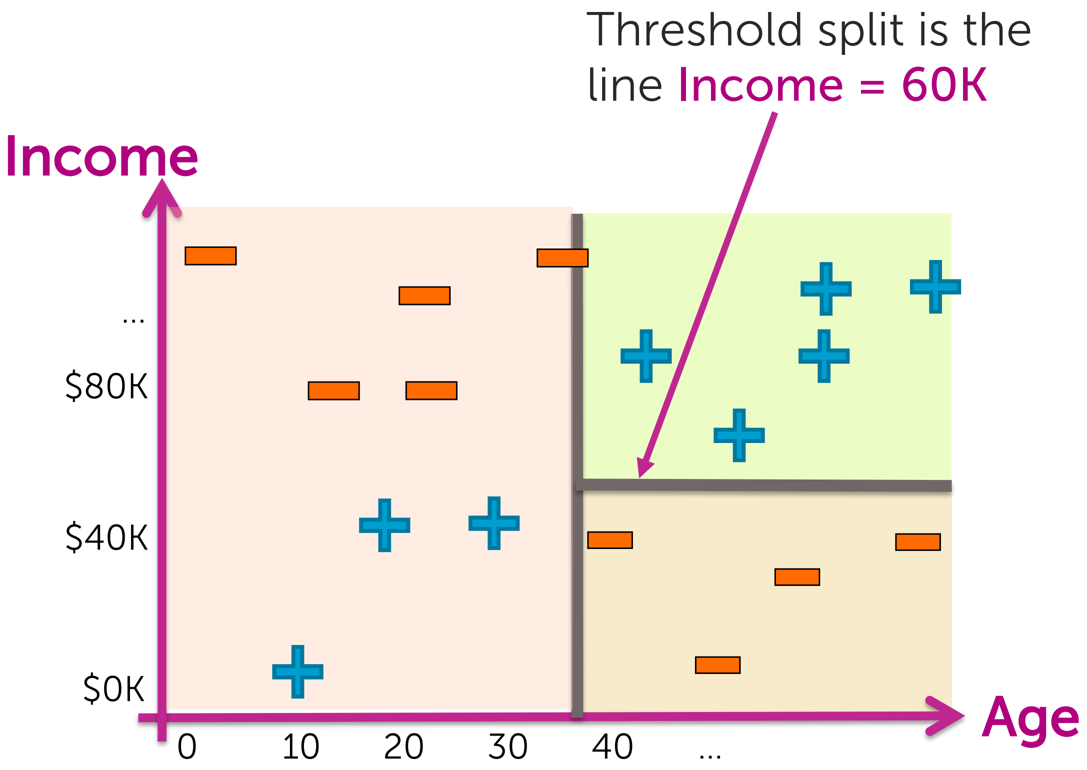
Visualizing the threshold split



Split on Age >= 38

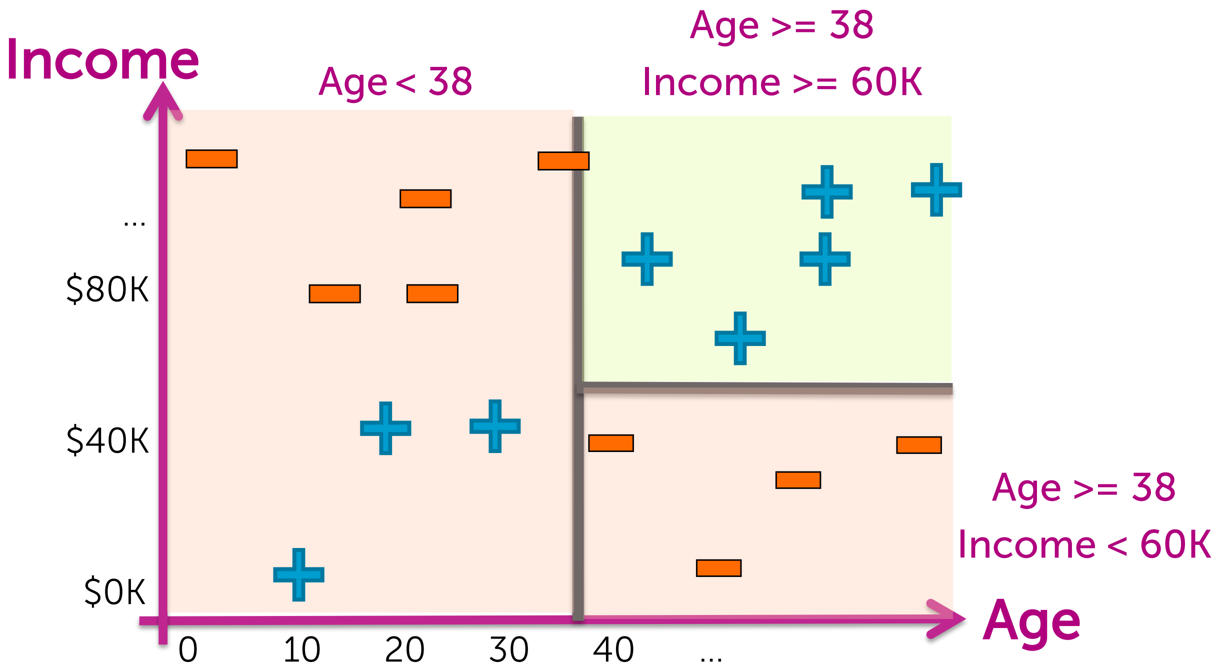


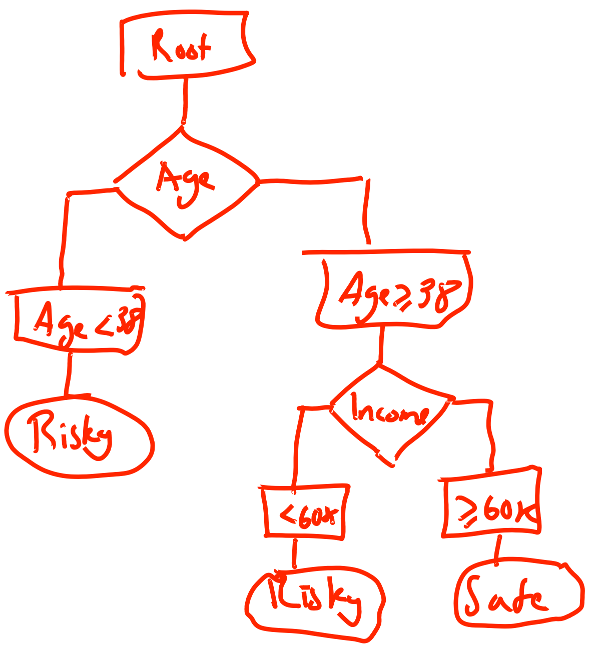
Depth 2: Split on Income >= $60K



Threshold split is the line Income = 60K

Each split partitions the 2-D space

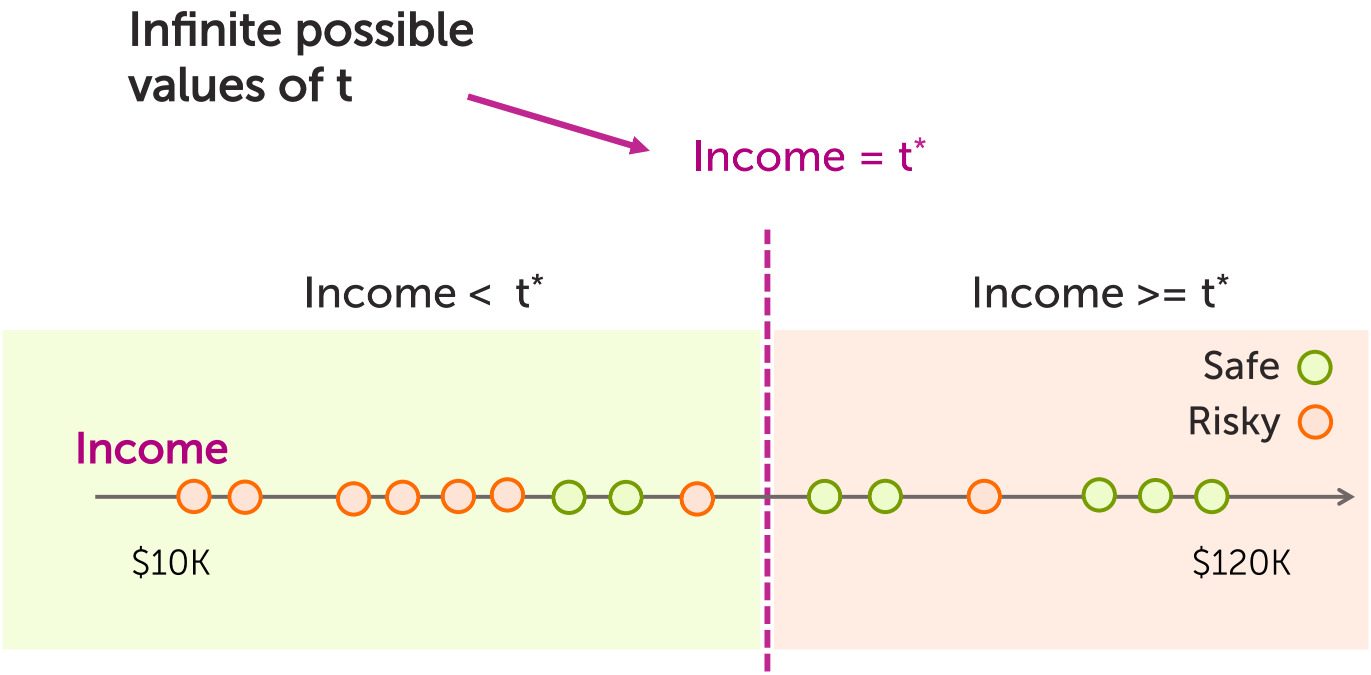




Finding the best threshold split

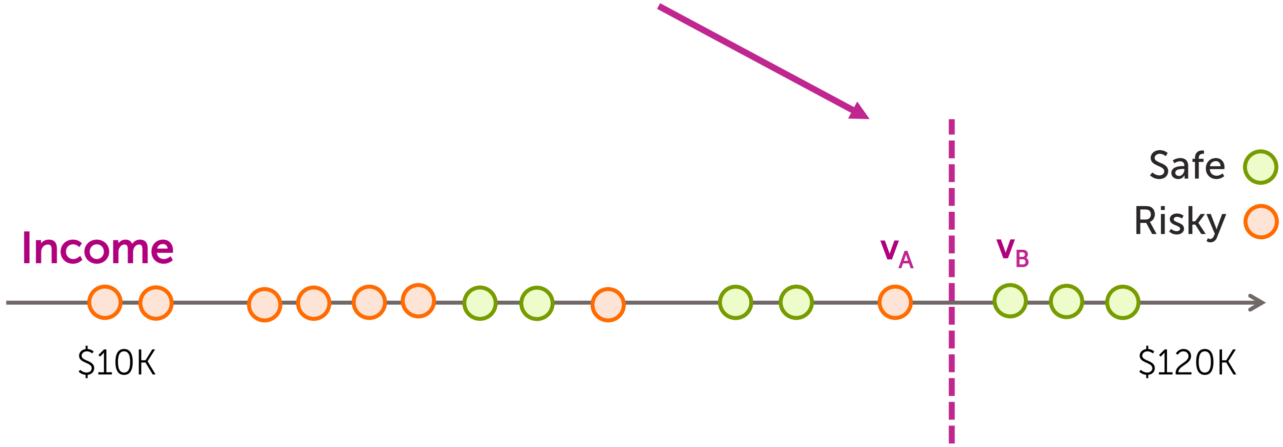
OPTIONAL

Finding the best threshold split



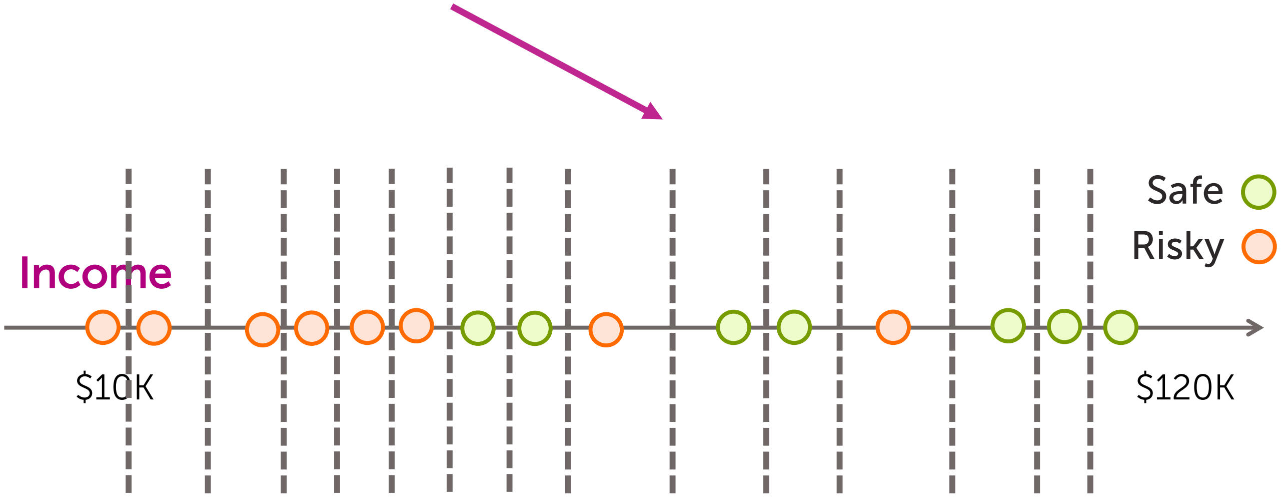
Consider a threshold between points

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

Income

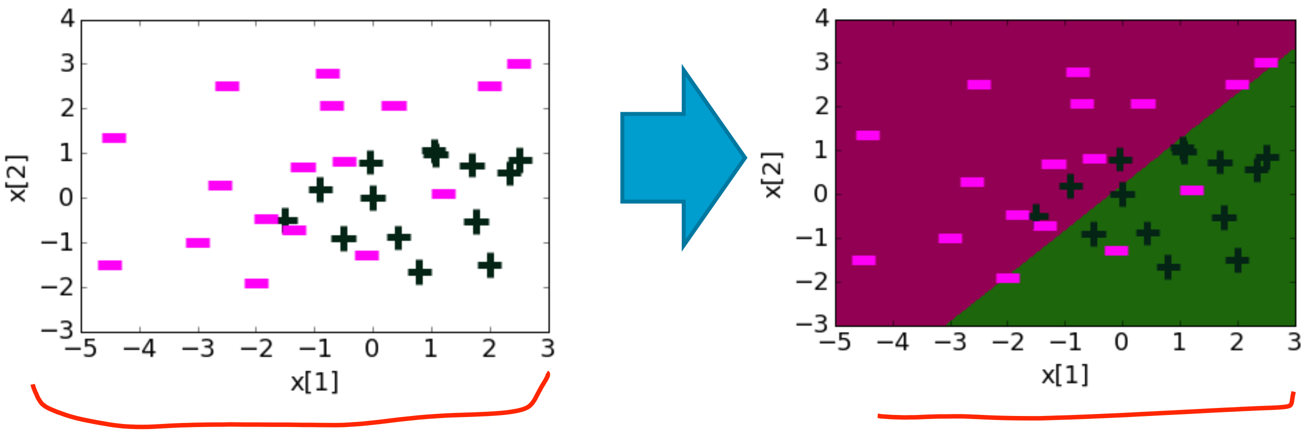
- Step 1: Sort the values of a feature $h_j(\mathbf{x})$:

  Let $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots \mathbf{v}_N\}$ denote sorted values
- Step 2:
  - For i = 1 … N-1
    - Consider split $t_i = (\mathbf{v}_i + \mathbf{v}_{i+1}) / 2$
    - Compute classification error for treshold split $h_j(\mathbf{x}) >= t_i$
  - Chose the $\mathbf{t}^*$ with the lowest classification error

Decision trees vs logistic regression: *Example* 💬

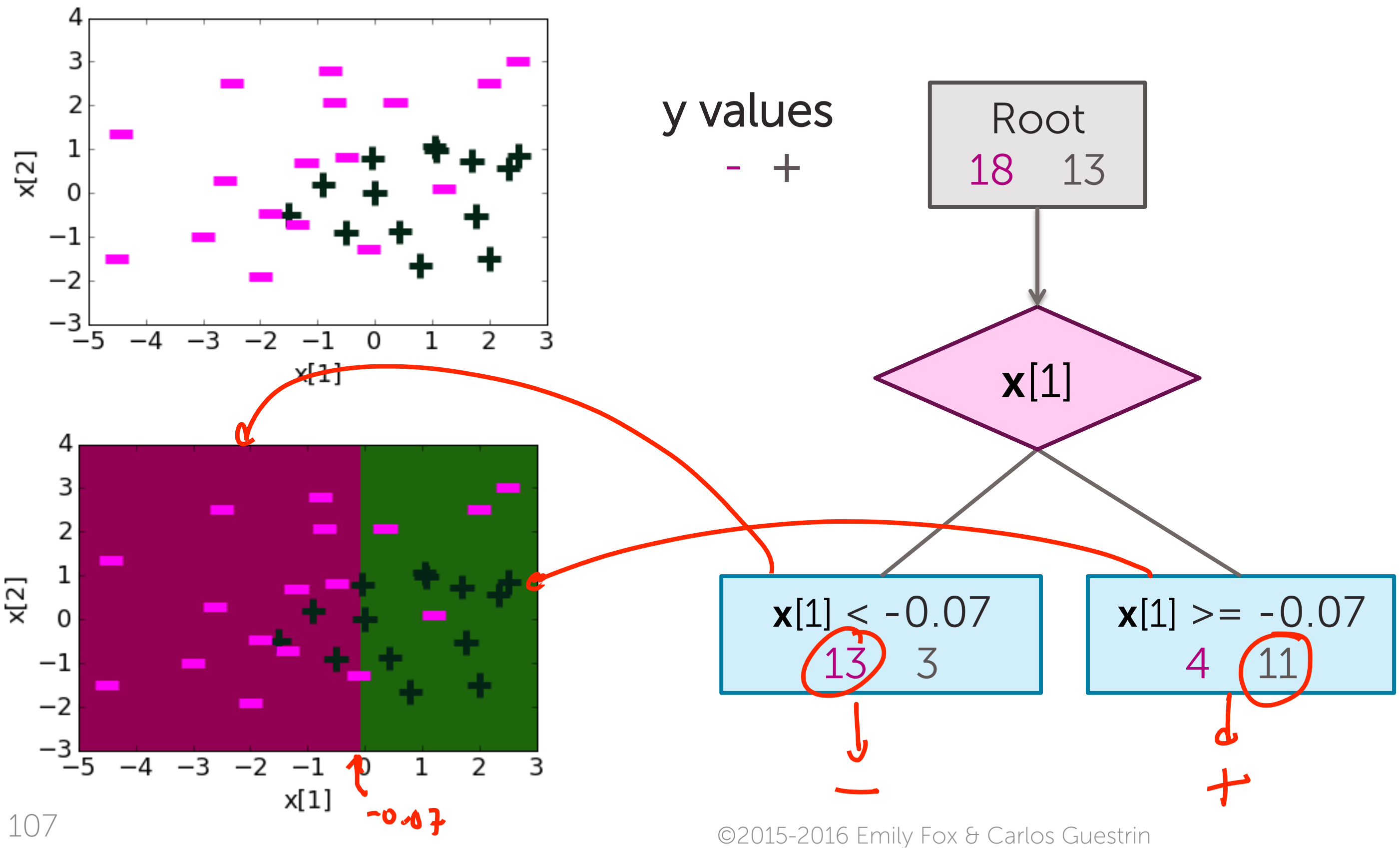
Logistic regression

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



data

Depth 1: Split on **x**[1]



y values

- +

Root

18 13

x[1]

x[1] < -0.07

13 3

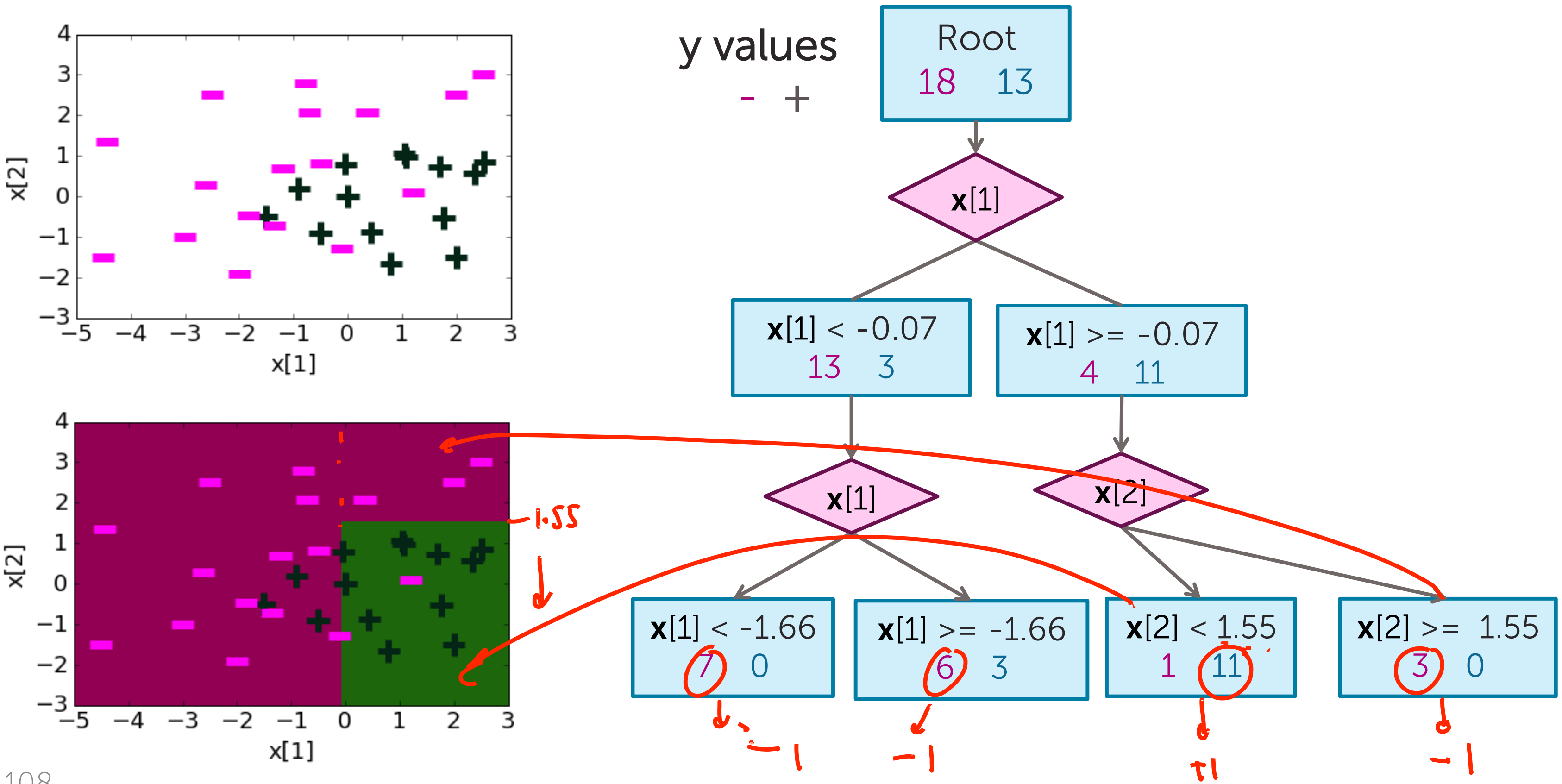
x[1] >= -0.07

4 11

−

+

Depth 2



y values

- +

Root
18 13

x[1]

x[1] < -0.07
13 3

x[1] >= -0.07
4 11

x[1]

x[2]

x[1] < -1.66
7 0

x[1] >= -1.66
6 3

x[2] < 1.55
1 11

x[2] >= 1.55
3 0

-1

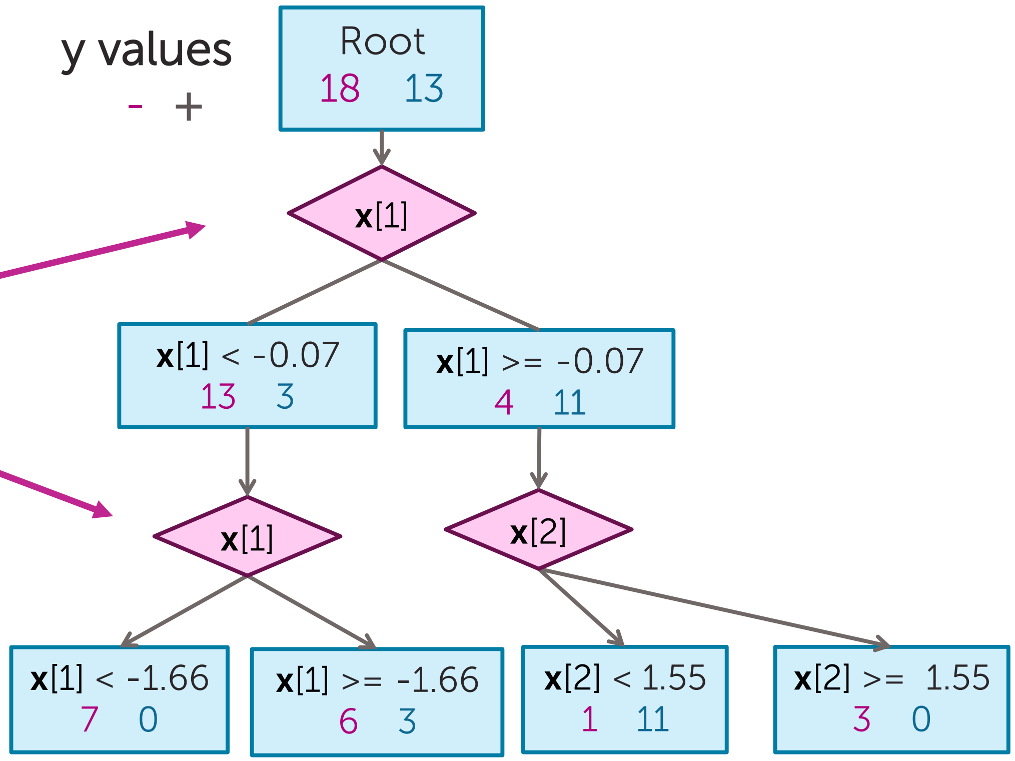
-1

+1

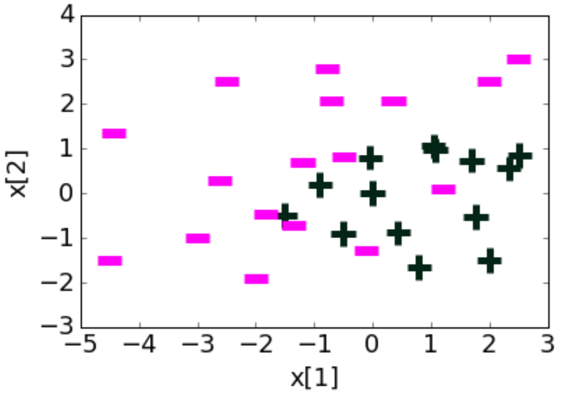
-1

Threshold split caveat

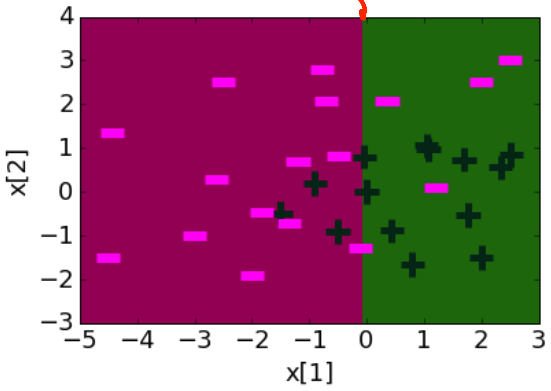
For threshold splits, same feature can be used multiple times



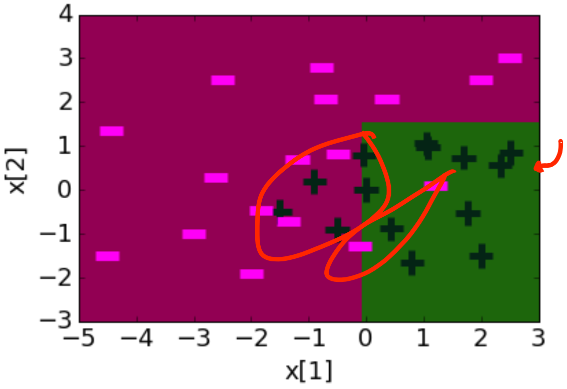
Decision boundaries



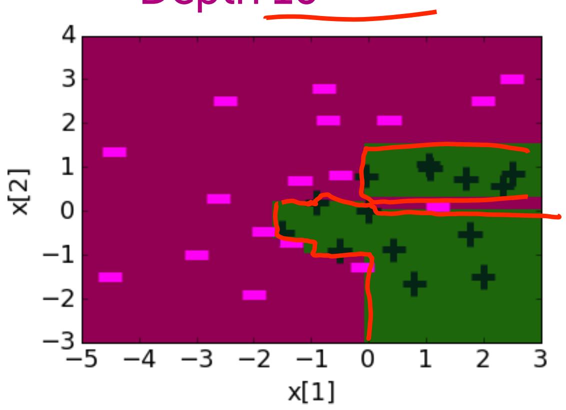
Depth 1



Depth 2

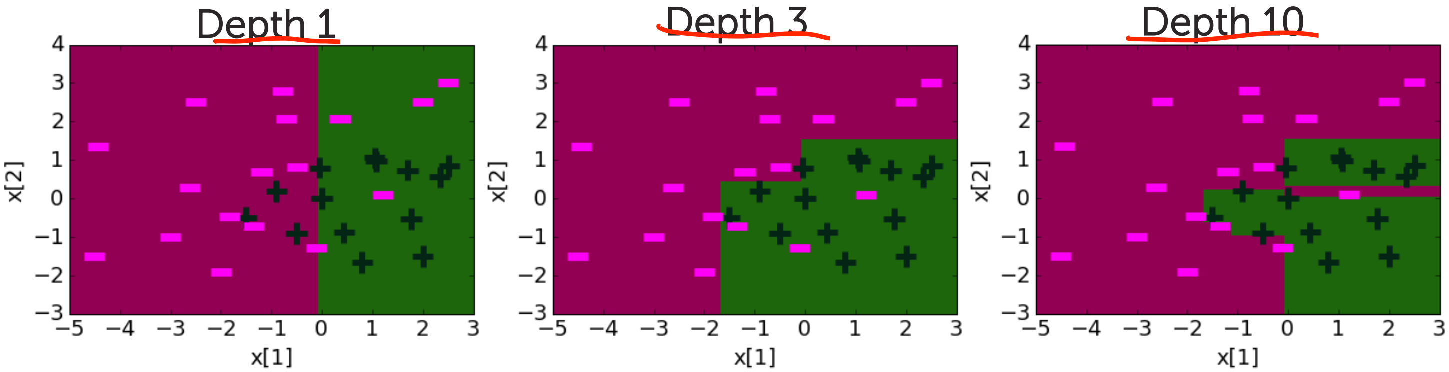


Depth 10

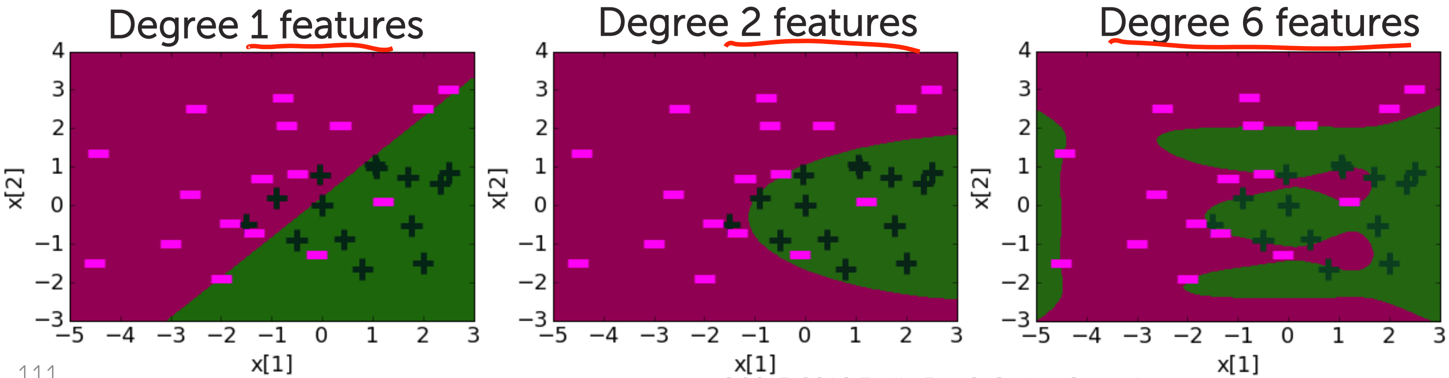


Comparing decision boundaries

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.