CSE/STAT 416

Decision Trees

Vinitra Swamy
University of Washington
July 13, 2020

* Content built on the work of Hunter Schafer and Emily Fox.

Logistics

- If you have a question, there is a high chance somebody else in the class the same question too
- Homework 3 –
  - Extension until Friday
  - Concept question #11 has been removed

Today:

- Naïve Bayes
- Decision Trees

Probability Classifier

Idea: Estimate probabilities $\hat{P}(y|x)$ and use those for prediction

Probability Classifier

Input x: Sentence from review

- Estimate class probability $\hat{P}(y = +1|x)$
- If $\hat{P}(y = +1|x) > 0.5$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$

Notes:

- Estimating the probability improves **interpretability**

Interpreting Score

$$Score(x_i) = w^T h(x_i)$$

$-\infty$ | $\hat{y}_i = -1$ | 0 | $\hat{y}_i = +1$ | ∞

Very sure $\hat{y}_i = -1$

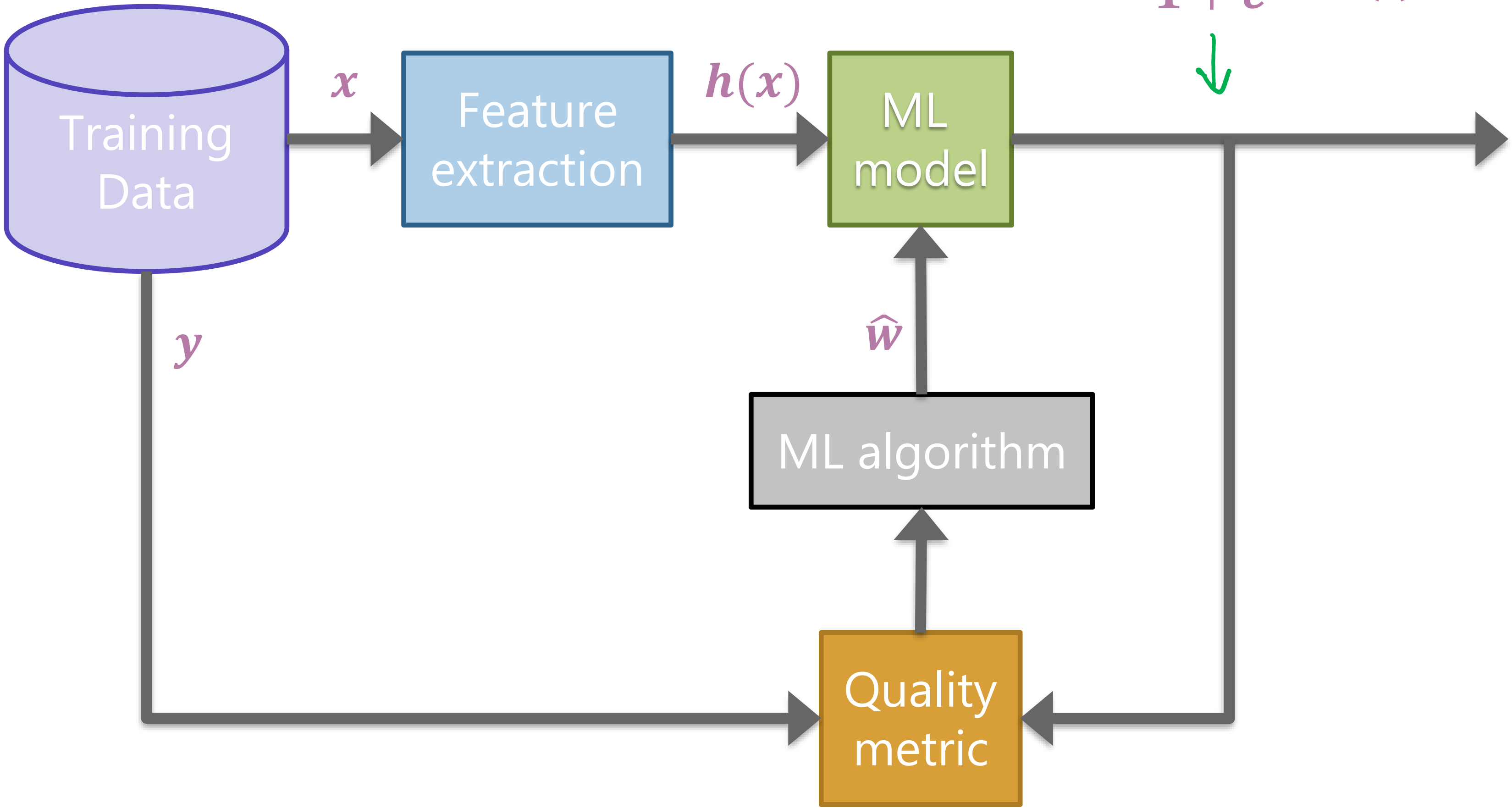
Not sure if $\hat{y}_i = -1$ *or* $\hat{y}_i = +1$

Very sure $\hat{y}_i = +1$

$\hat{P}(y_i = +1|x_i) = 0$ $\quad$ $\hat{P}(y_i = +1|x_i) = 0.5$ $\quad$ $\hat{P}(y_i = +1|x_i) = 1$

0 $\quad$ $\hat{P}(y = +1\,|x)$ $\quad$ 1

$$\hat{P}(y=+1|x,\hat{w}) = sigmoid\left(\hat{w}^T h(x)\right) = \frac{1}{1+e^{-\hat{w}^T h(x)}}$$



Naïve Bayes

Idea: Naïve Bayes

$$x = \text{“}The\ sushi\ \&\ everything\ else\ was\ awesome!\text{”}$$

$$P\left(y = +1 \,\middle|\, x = \text{“}The\ sushi\ \&\ everything\ else\ was\ awesome!\text{”}\right)?$$

$$P\left(y = -1 \,\middle|\, x = \text{“}The\ sushi\ \&\ everything\ else\ was\ awesome!\text{”}\right)?$$

Idea: Select the class with the highest probability!

Bayes Rule: $P(y = +1|x) = \frac{P(x|y = +1)P(y=+1)}{P(x)}$

$$\frac{P\left(\text{“}The\ sushi\ \&everything\ else\ was\ awesome!\text{”} \,\middle|\, +1\right) \boxed{P(+1)}}{\cancel{P(\text{“}The\ sushi\ \&\ everything\ else\ was\ awesome!\text{”})}}$$

Since we're just trying to find out which class has the greater probability, we can discard the divisor.

Problem

Idea: Select the class with the highest probability!

Problem: We have not seen the sentence before.

Assumption: Words are independent from each other.

$$\frac{\#+\text{reviews}}{\#\text{reviews}}$$

$x =$ *"The sushi & everything else was awesome!"*

$$\frac{P(\text{"The sushi \& everything else was awesome!"}|+1)\, P(+1)}{\cancel{P(\text{"The sushi \& everything else was awesome!"})}}$$

$$P(\text{"The sushi \&everything else was awesome!"} \mid +1)$$
$$= P(\text{The} \mid +1) * P(sushi \mid +1) * P(\&| + 1) * P(everything| + 1)$$
$$* P(else| + 1) * P(was| + 1) * P(awesome| + 1)$$

$$P(\text{"awesome"} \mid +1)?$$

$$= \frac{\#\text{ of times "awesome" appears in + reviews}}{\#\text{ of words in + reviews}}$$

Zeros

If a feature is missing in a class everything becomes zero.

$$P(\text{“}The\ sushi\ \&everything\ else\ was\ awesome!\text{”} \mid +1)$$

$$= P(\mathrm{The} \mid +1) * \mathrm{P}(sushi \mid +1) * P(\& \mid +1) \ * P(everything \mid +1)$$

$$* P(else \mid +1) * P(was \mid +1) * P(awesome \mid +1)$$

$= 0$

Solutions?

- Take the log (product becomes a sum: linear classifier)
- Laplacian Smoothing (adding a constant to avoid multiplying by zero)

Naïve Bayes vs Logistic Regression

Naïve Bayes vs Logistic Regression

Logistic Regression:

$$P(y = +1|x, w) = \frac{1}{1 + e^{-w^T h(x)}}$$

Naïve Bayes:

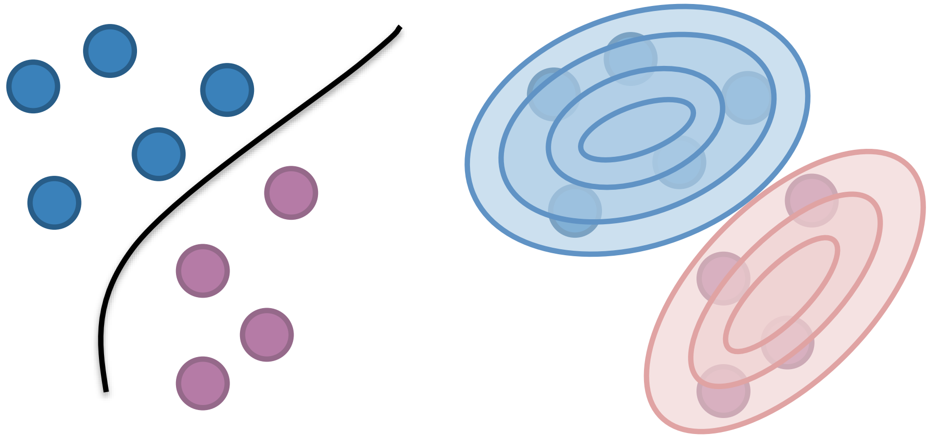
$$P(y|\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_d) = \prod_{j=1}^{d} P(x_j|y)\ P(y)$$

Naïve Bayes vs Logistic Regression

Generative vs Discriminative Classifiers $P(x|y)\ P(y)$

Generative: defines a model for generating x (e.g. Naïve Bayes)

Discriminative: only cares about defining and optimizing a decision boundary (e.g. Logistic Regression) $P(y|x) \rightarrow Ws$

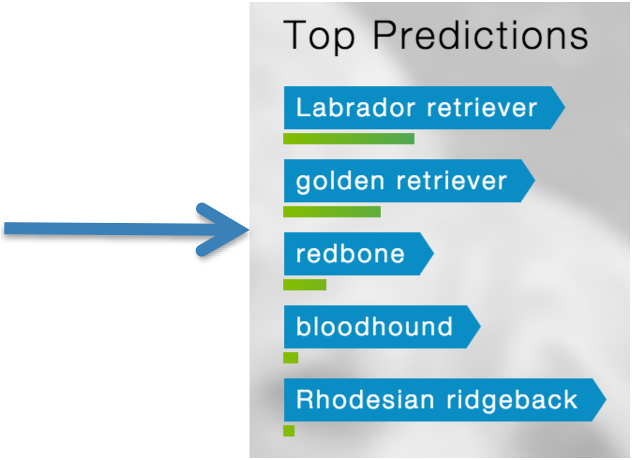


Properties

- Linear Classifier for discrete values
- Continuous Variables - Gaussian Naïve Bayes
- Gaussian Naïve Bayes is equivalent to a Logistic Regression!
- Naïve Bayes very efficient for discrete data: only counts
- Naïve Bayes works well for big datasets

Multiclass Classification

- Everything works with multiple classes!



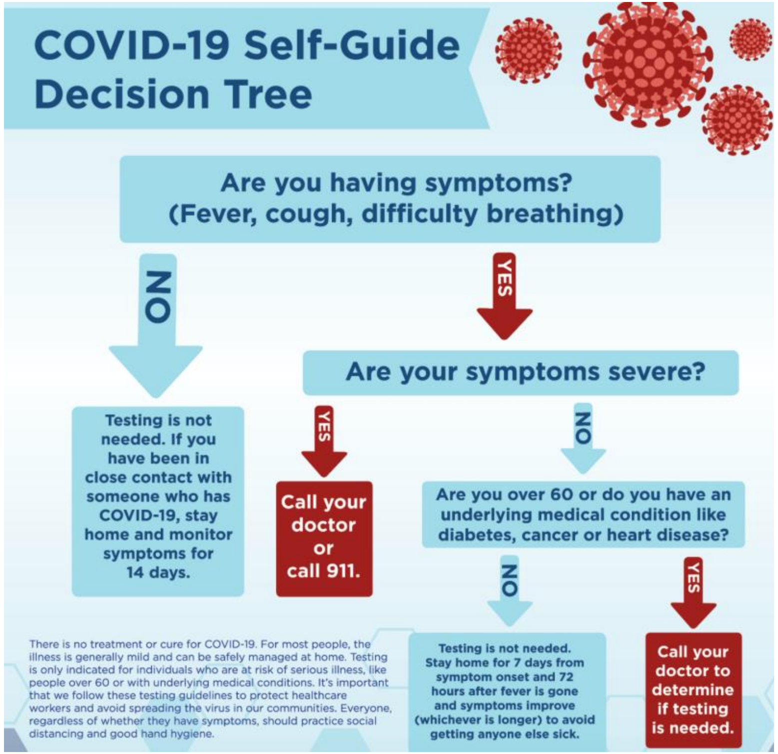
Input: **x**
Image pixels

Output: y
Object in image

Take max of:
P(Labrador retriever|x), P(golden retriever|x), P(redbone|x), P(bloodhound|x), P(Rhodesian ridgeback|x)

Decision Trees

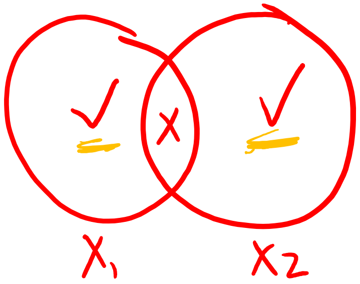
How do we make decisions?



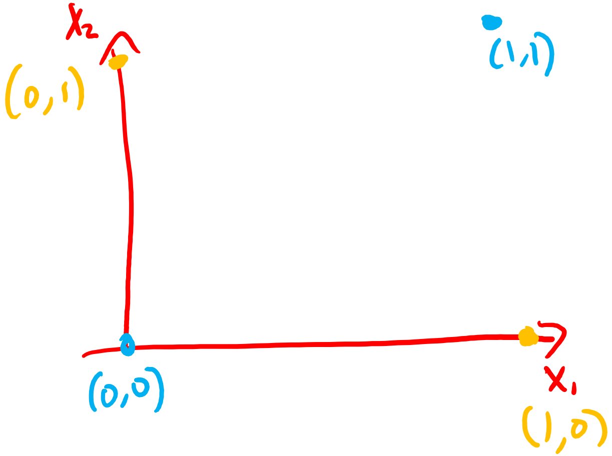
https://www.holzer.org/coronavirus-covid-19-updates/

XOR
(Exclusive Or)

- A line might not always support our decisions.



x_1	0	1	0	1
x_2	1	0	0	1
y	1	1	0	0



What makes a loan risky?

I want to buy a new house!

Loan Application

Credit History ★★★★

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
★★★

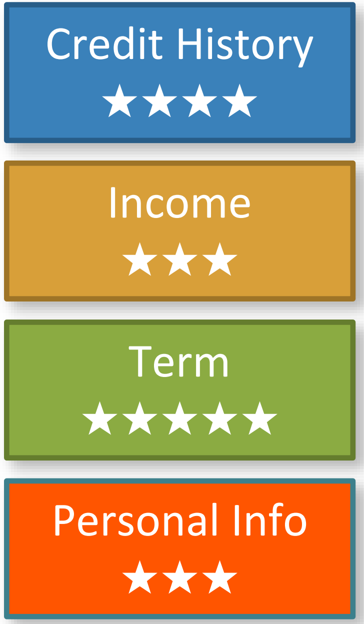
Term
★★★★★

Personal Info
★★★

Loan terms

How soon do I need to pay the loan?

Example: 3 years, 5 years,…



Personal information

Age, reason for the loan, marital status,…

Example: Home loan for a married couple

Credit History
★★★★

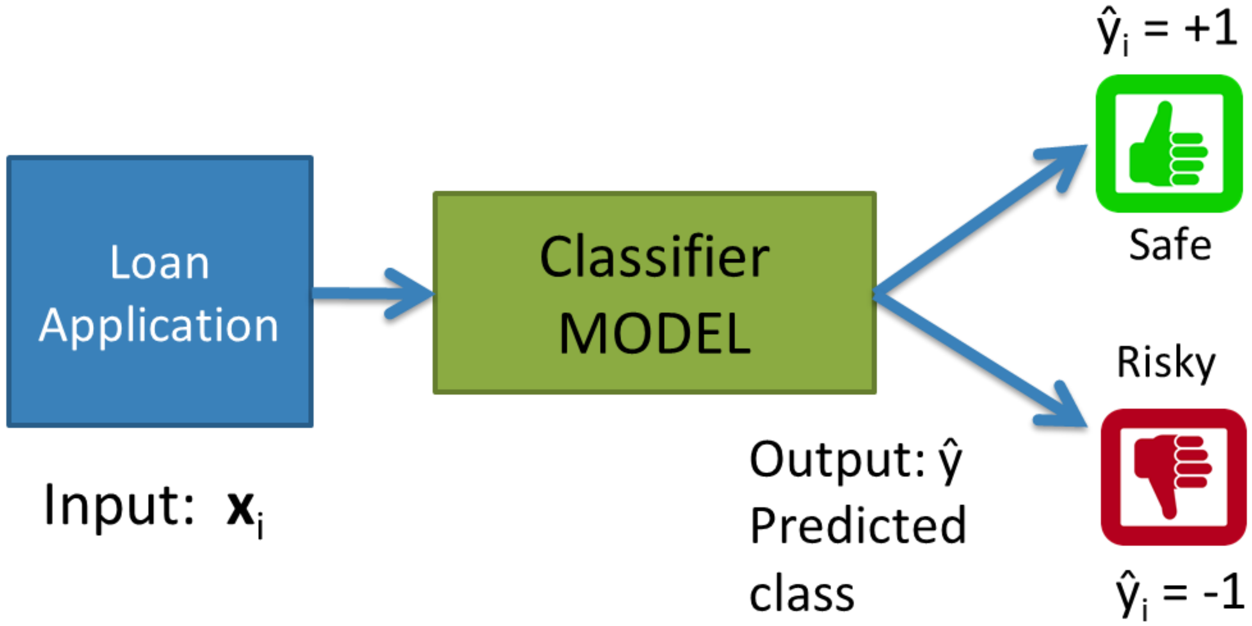
Income
★★★

Term
★★★★★

Personal Info
★★★

Intelligent application

Classifier review



Setup

N= 9

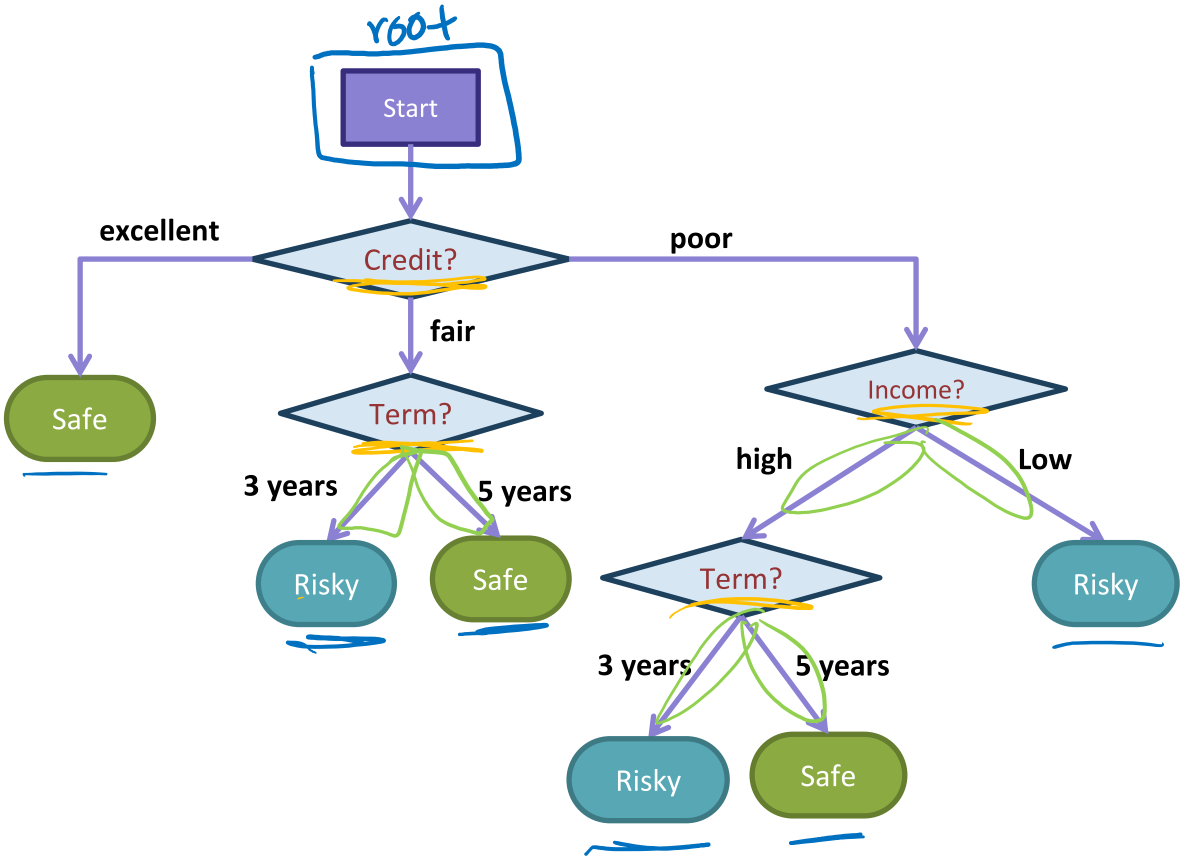
Data (N observations, 3 features)

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	safe
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Evaluation: classification error

Many possible decisions: number of trees grows exponentially!

Decision Trees



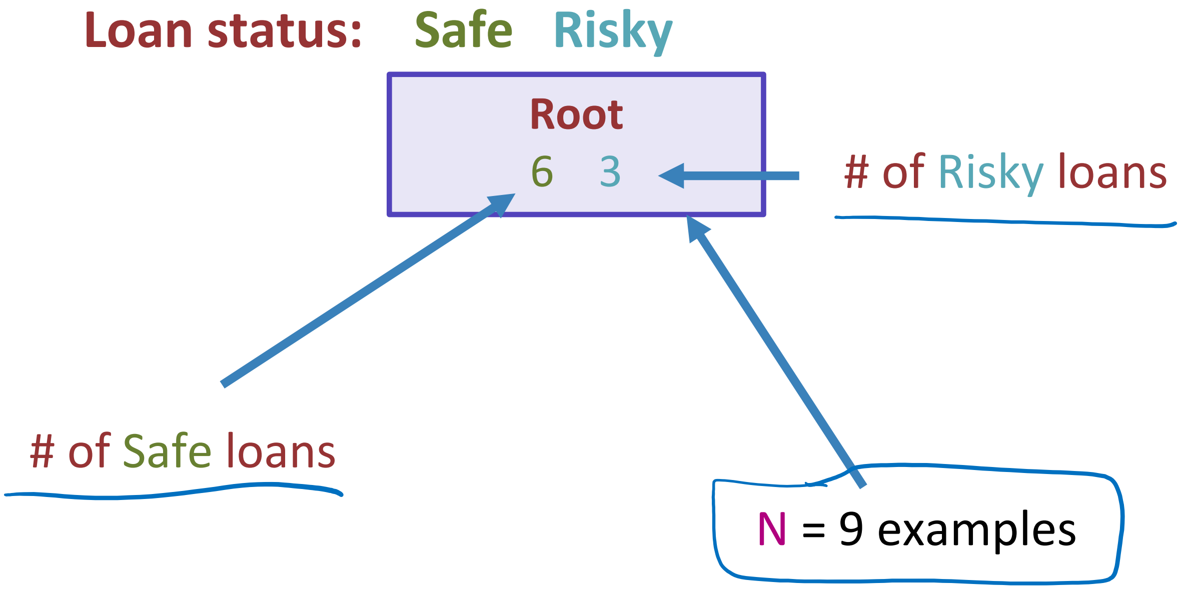
- **internal node:** testing a feature
- **branch:** splits into possible values of a feature
- **leaf:** final decision (the class value)

Growing Trees

- Grow the trees using a greedy approach
- What do we need?

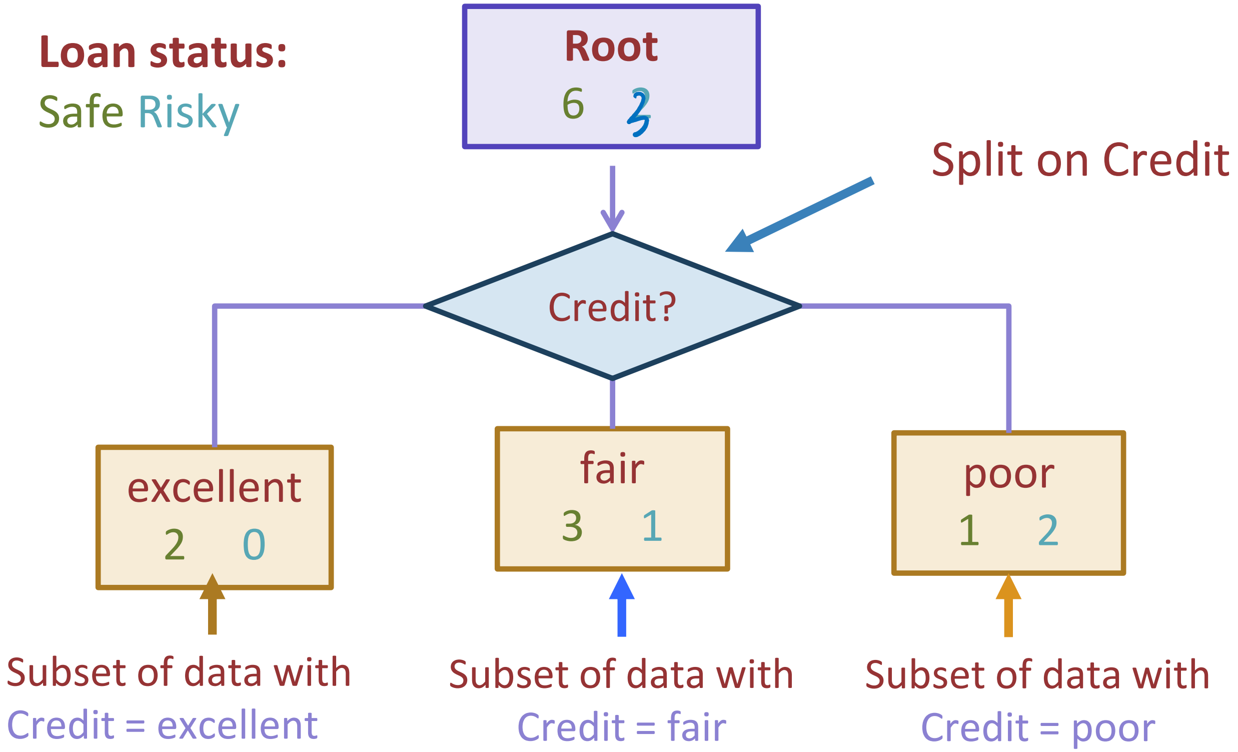
- feature importance to cut off possibilities early
  - → weight
  - → order
- stopping

Visual Notation



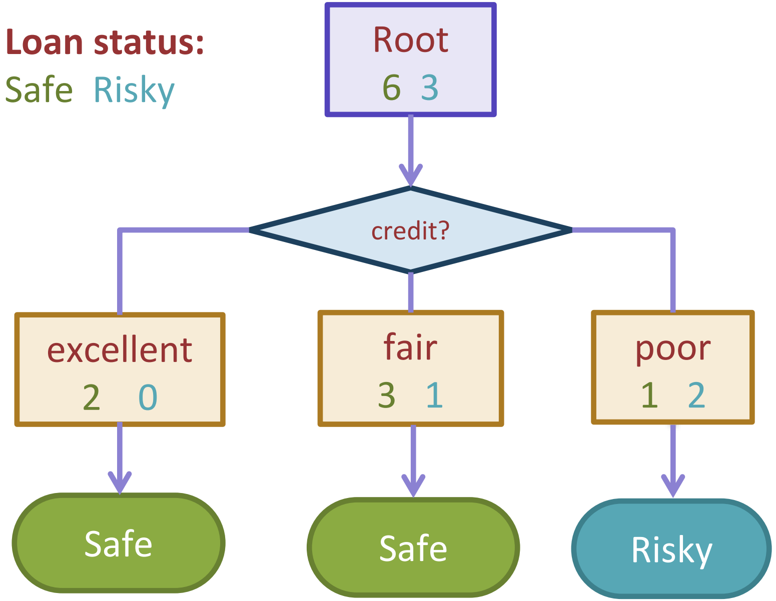
Decision stump: 1 level

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	safe
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



Making predictions

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

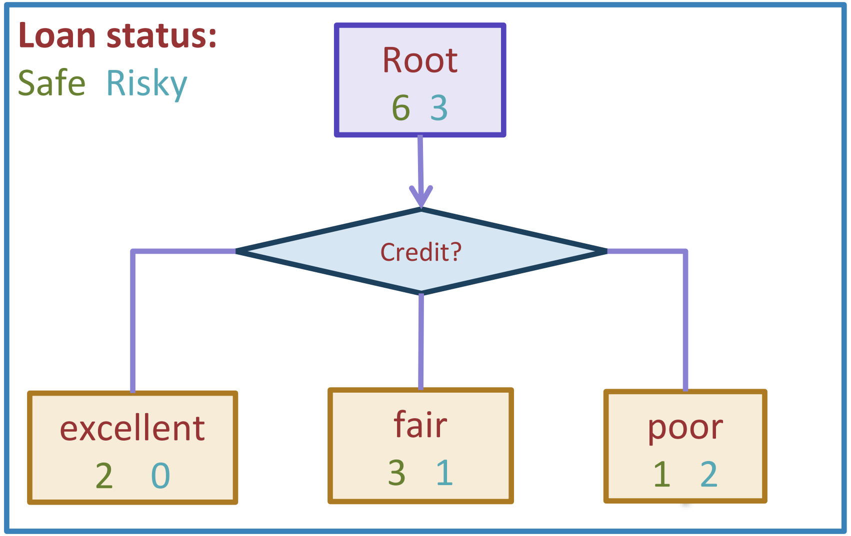


How do we select the best feature?

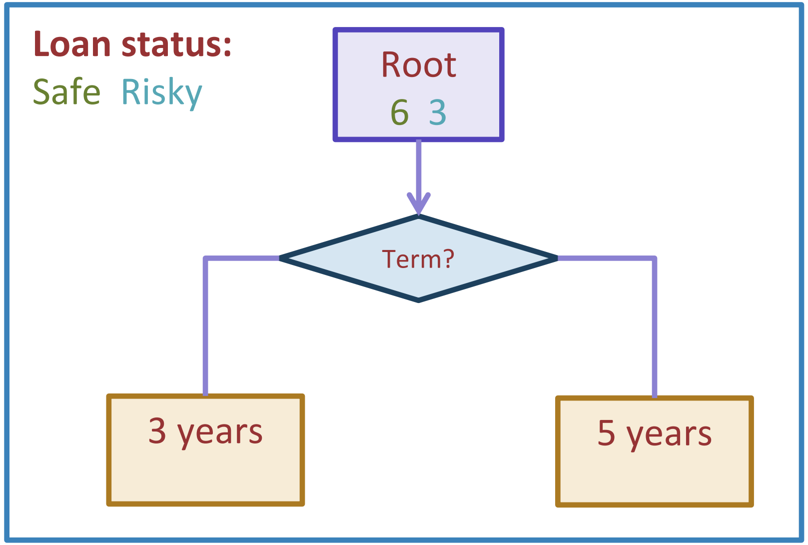
.

* Select the split with lowest classification error

Choice 1: Split on Credit



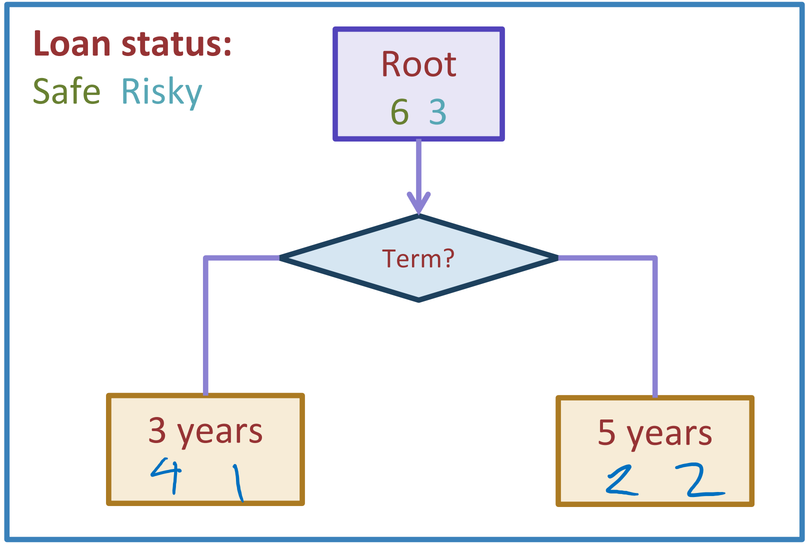
Choice 2: Split on Term



Calculate the node values.

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	safe
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Choice 2: Split on Term

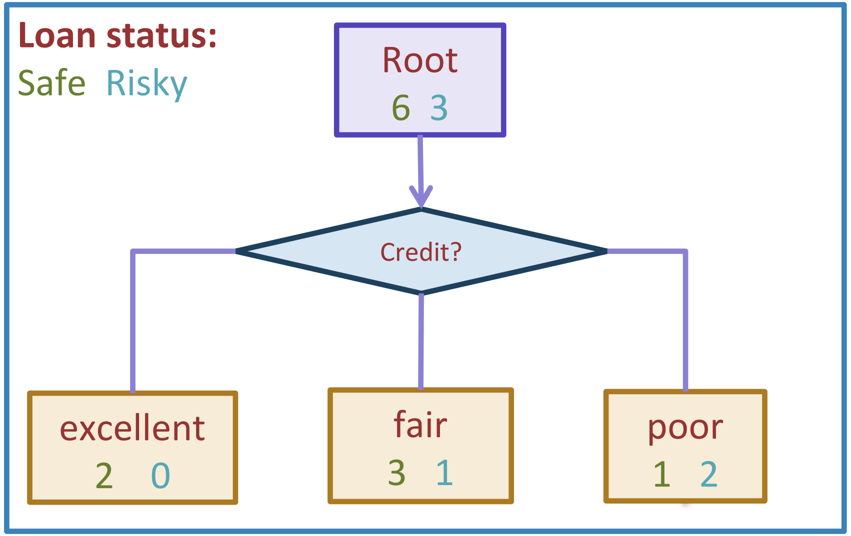


pollev.com/cse416

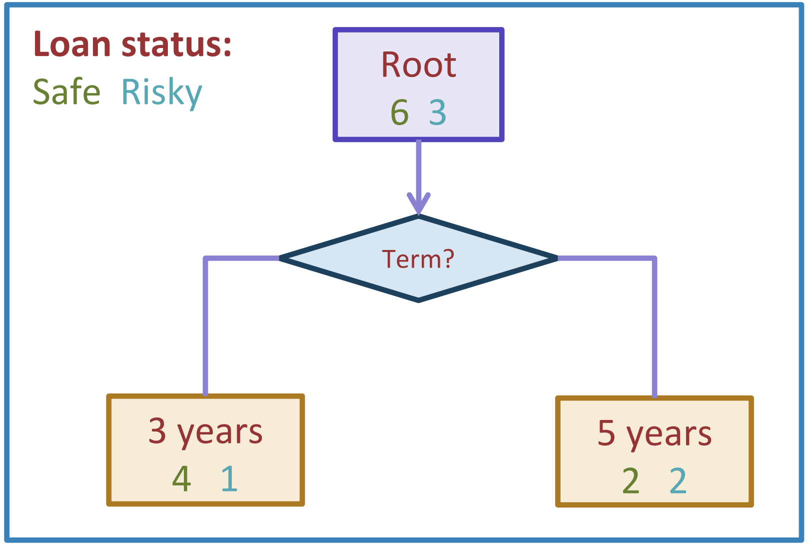
How do we select the best feature?

* Select the split with lowest classification error

Choice 1: Split on Credit



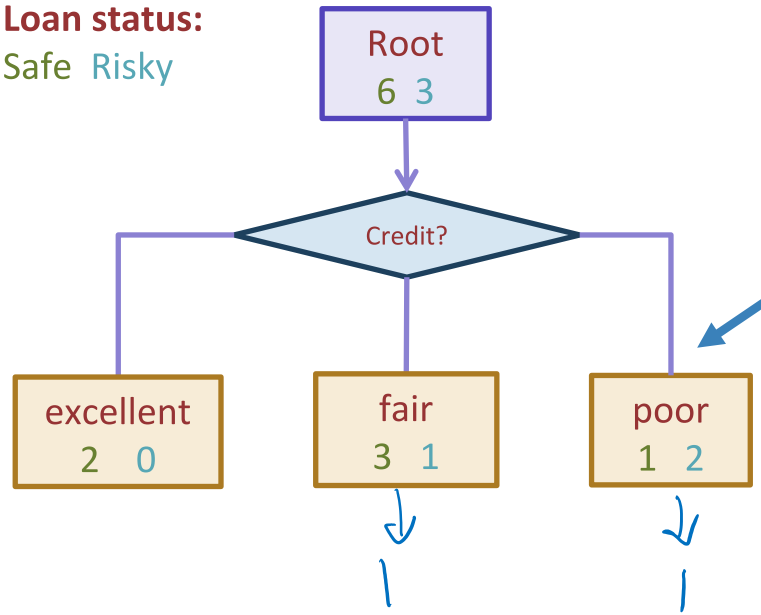
Choice 2: Split on Term



Brain Break

9:45

How do we measure effectiveness of a split?



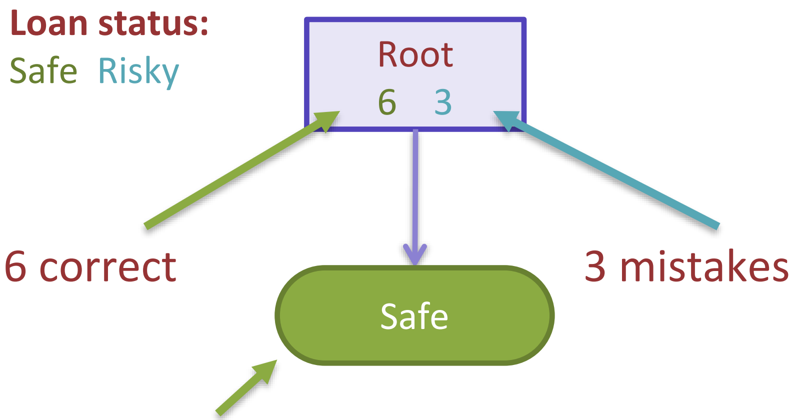
Idea: Calculate classification error of this decision stump

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

Calculating classification error

Step 1: $\hat{y}$ = class of majority of data in node

Step 2: Calculate classification error of predicting $\hat{y}$ for this data



$\hat{y}$ = **majority class**

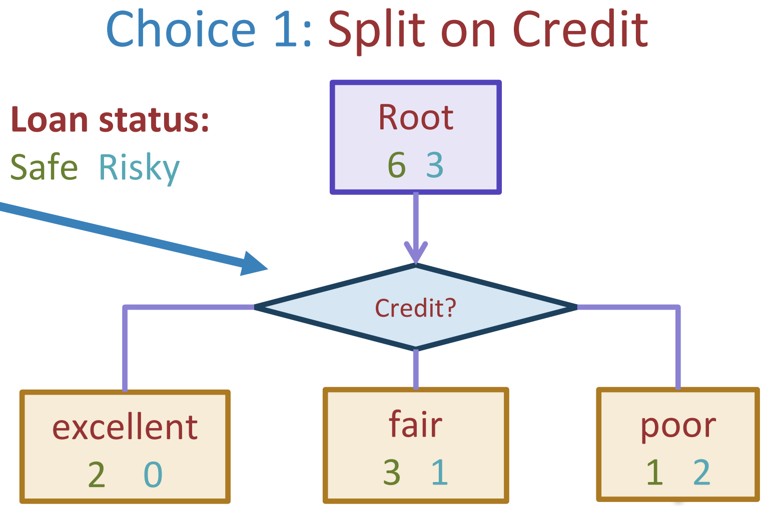
$$\text{Error} = \frac{3}{9}$$

$$= 0.333$$

Tree	Classification error
(root)	0.33

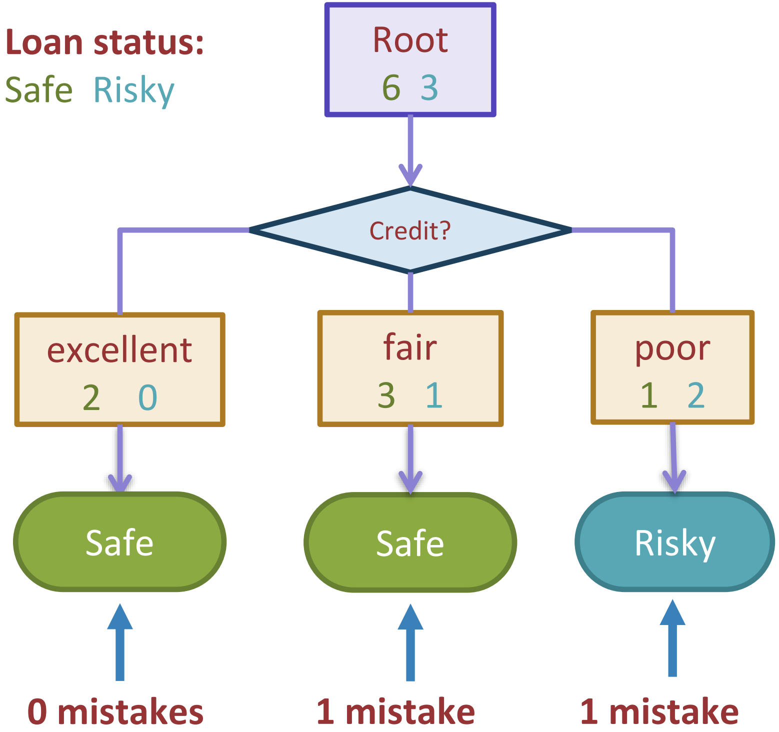
Choice 1: Split on Credit history?

Does a split on Credit reduce classification error below 0.33?



Split on Credit: Classification error

Choice 1: Split on Credit

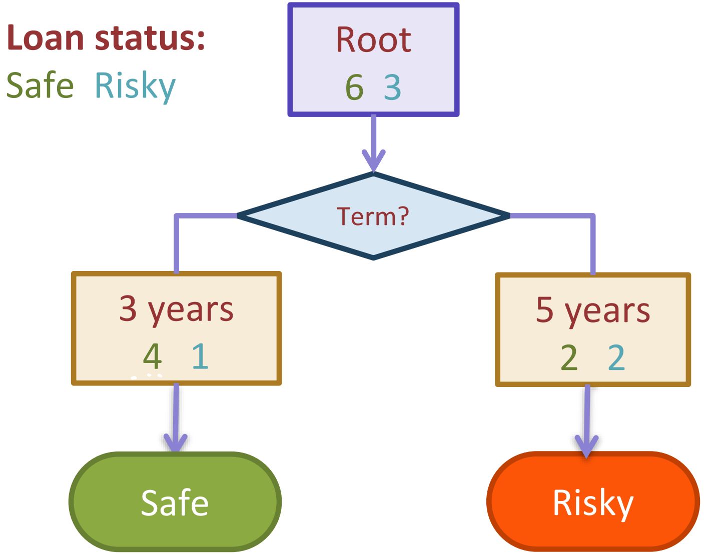


$$\text{Error} = \frac{2}{9} = 0.22$$

Tree	Classification error
(root)	0.33
Split on credit	0.22

Choice 2: Split on Term?

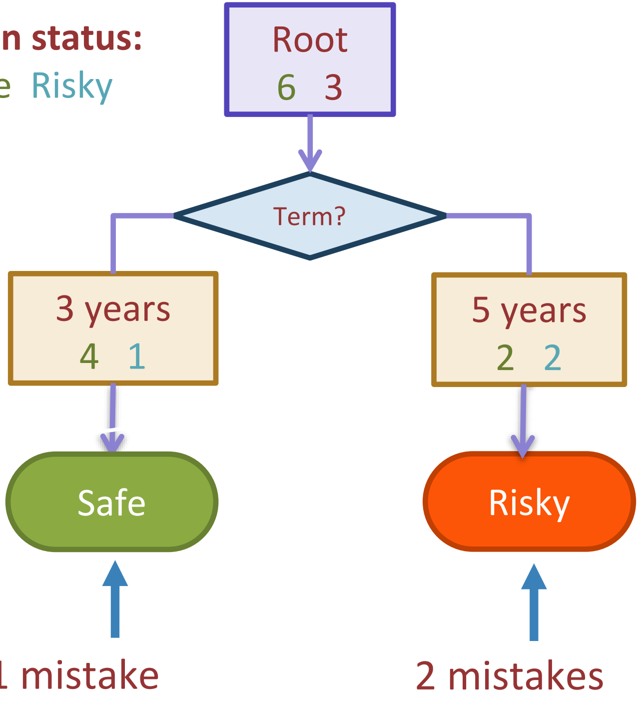
Choice 2: Split on Term



Evaluating the split on Term

Choice 2: Split on Term

Loan status:
Safe Risky



$$\text{Error} = \frac{3}{9}$$

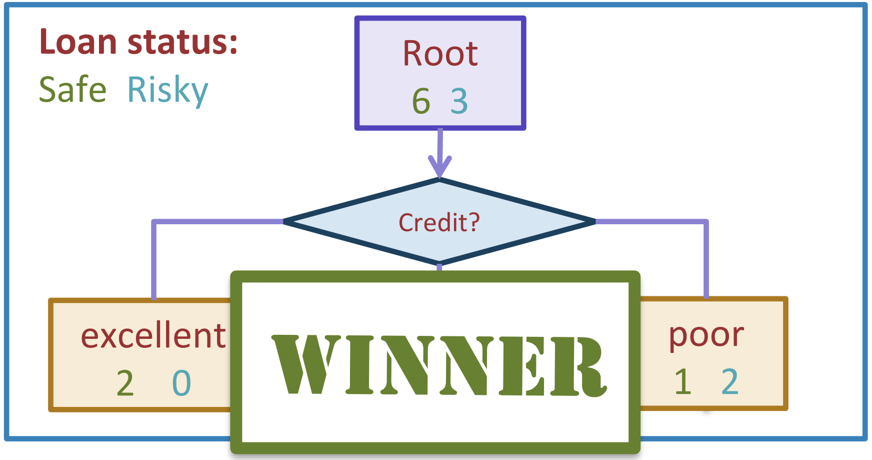
$$= 0.333$$

Tree	Classification error
(root)	0.33
Split on credit	0.22
Split on term	0.33

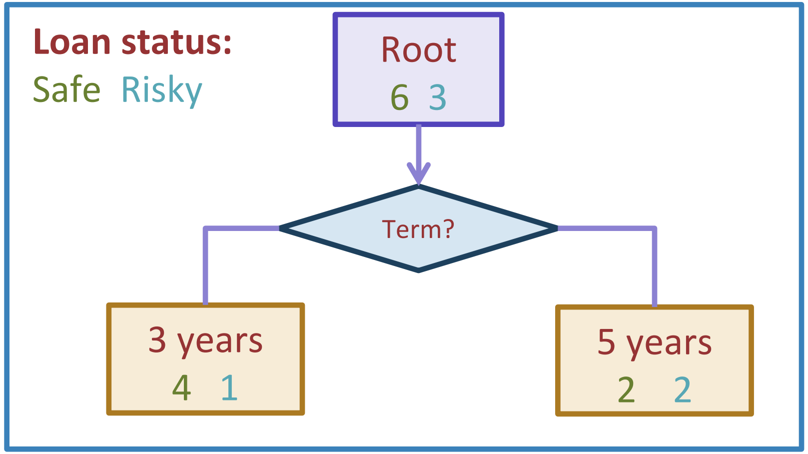
Choice 1 vs Choice 2: Comparing split on credit vs term

Tree	Classification error
(root)	0.33
split on credit	0.22
split on loan term	0.33

Choice 1: Split on Credit



Choice 2: Split on Term



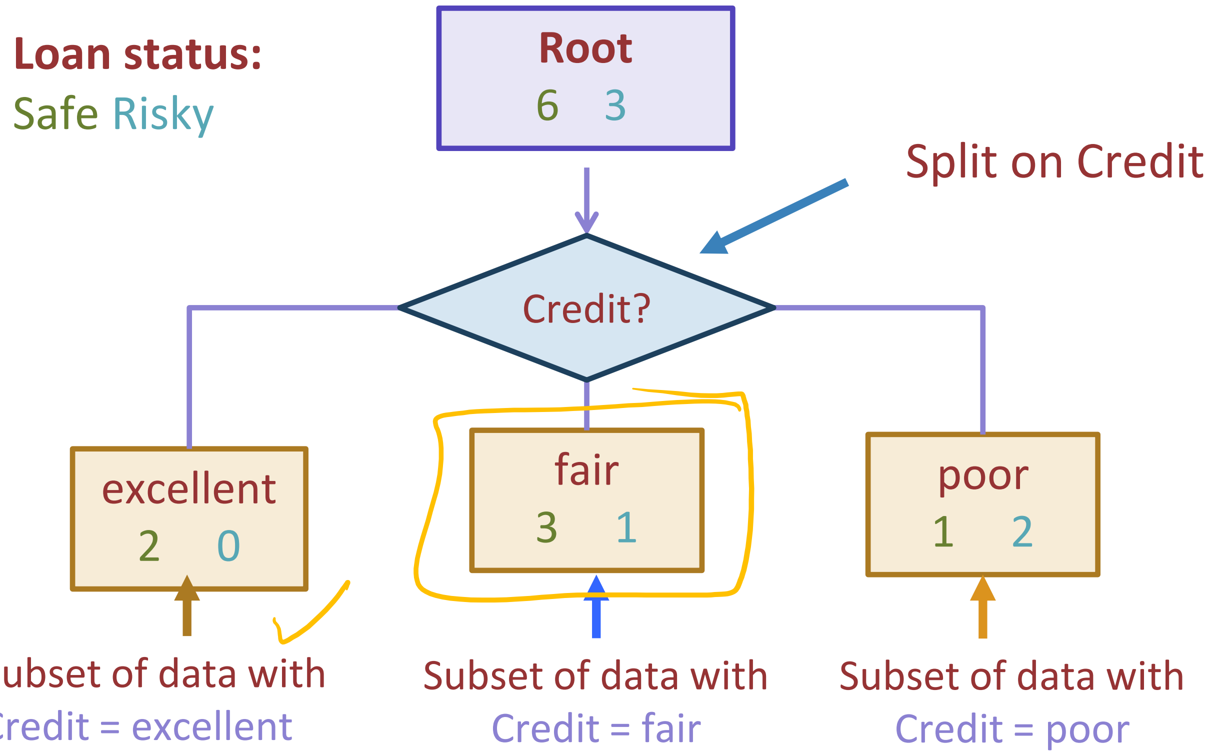
Split Selection Summary

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

Greedy Algorithm

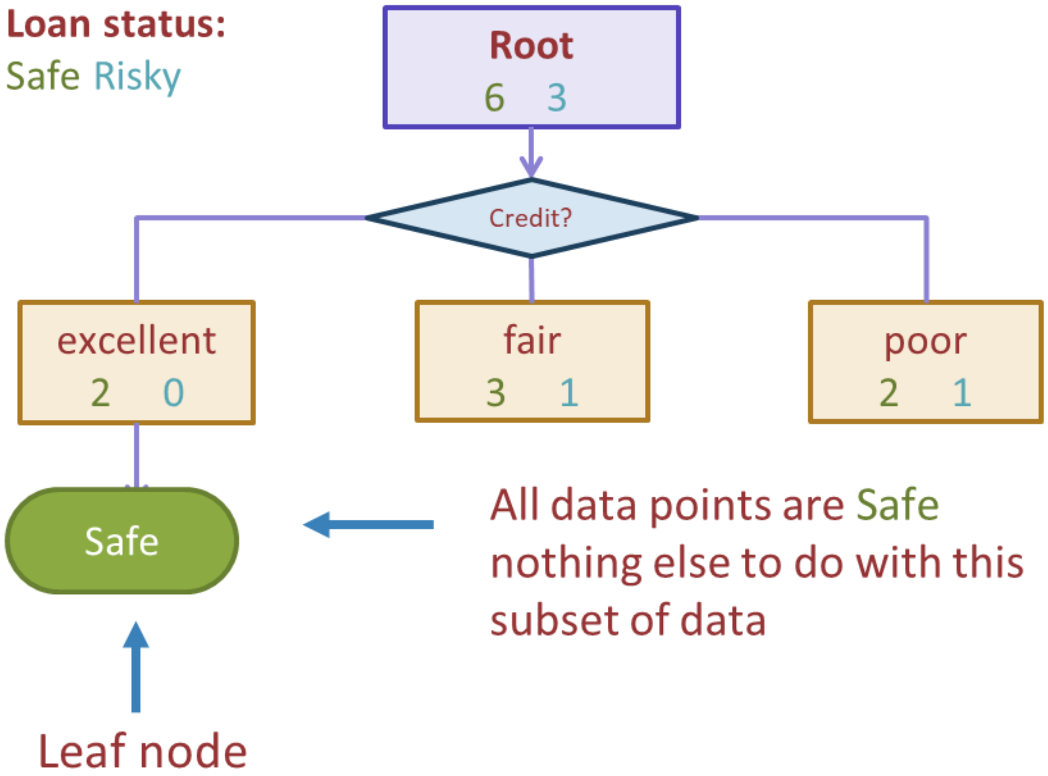
- If split is perfect (classification error = 0) or out of features:
  - Stop
- Else:
  - repeat split selection with next stump

Decision stump: 1 level

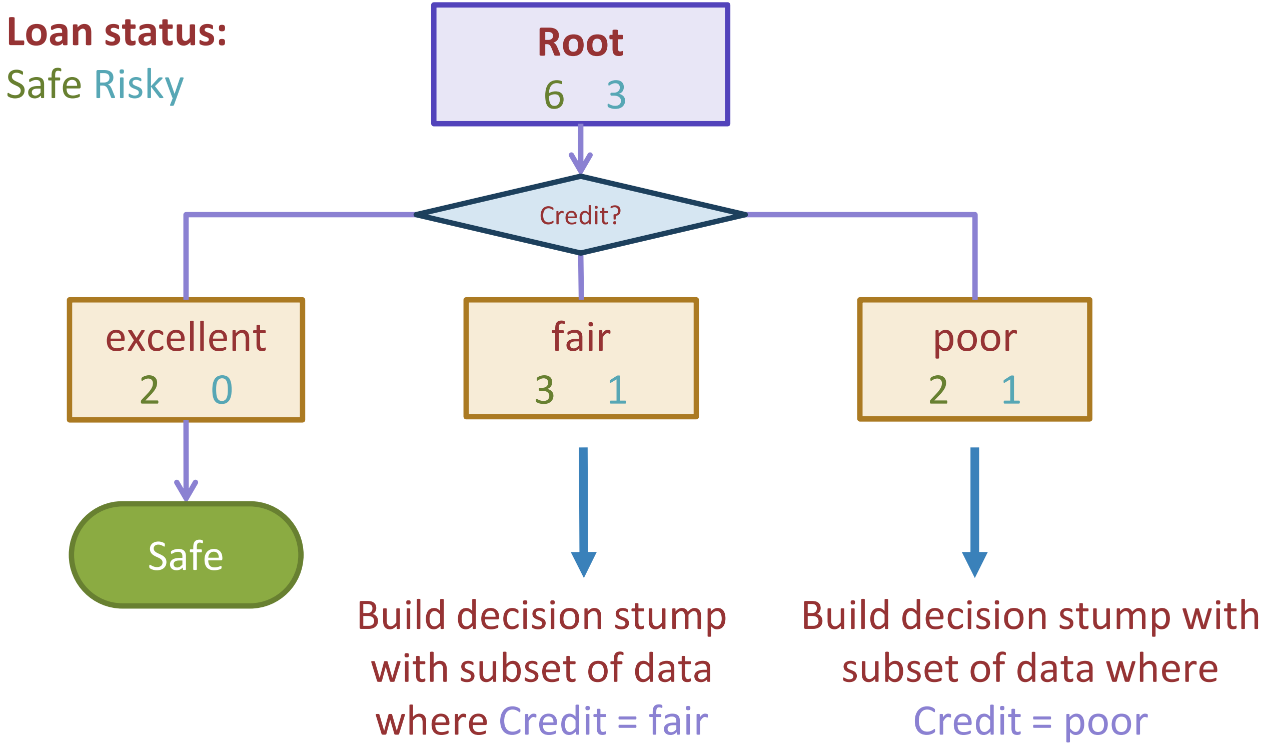


Stopping

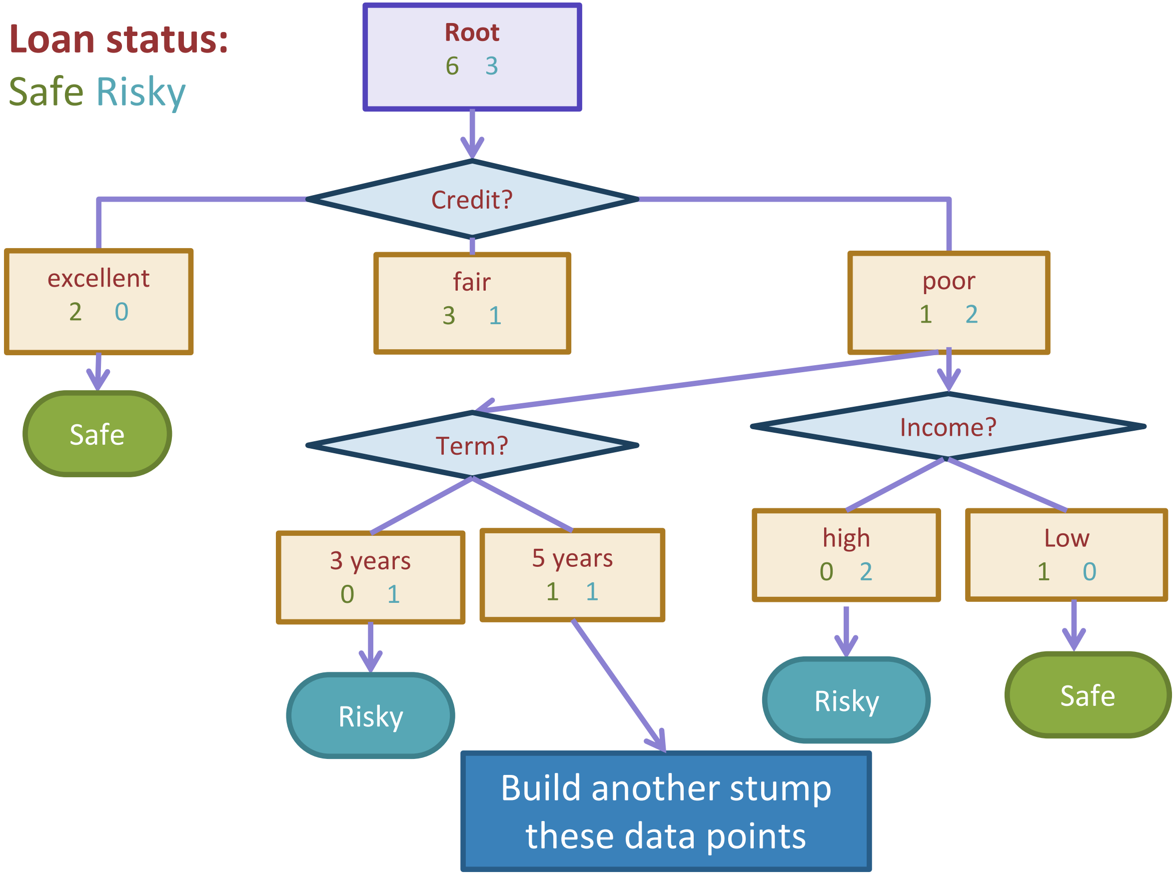
- Stop if all points are in one class



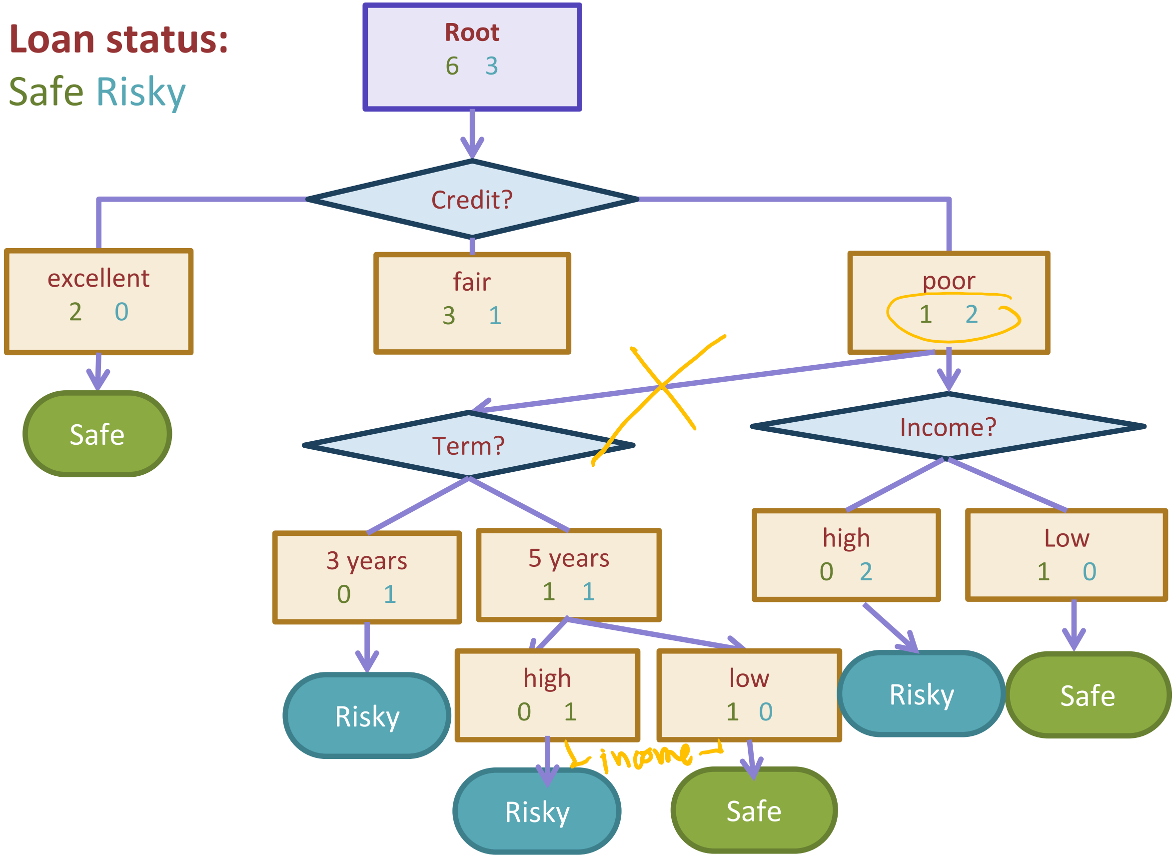
Tree learning = Recursive stump learning



Second level



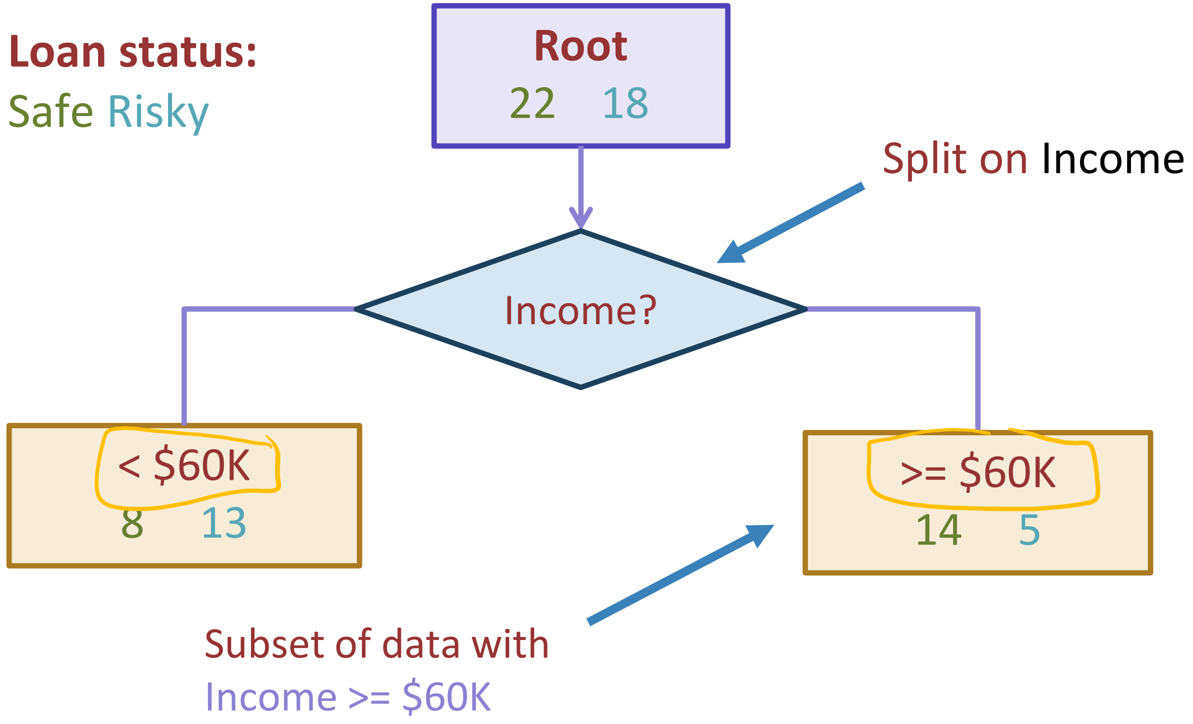
Next Step Tree



Real valued features

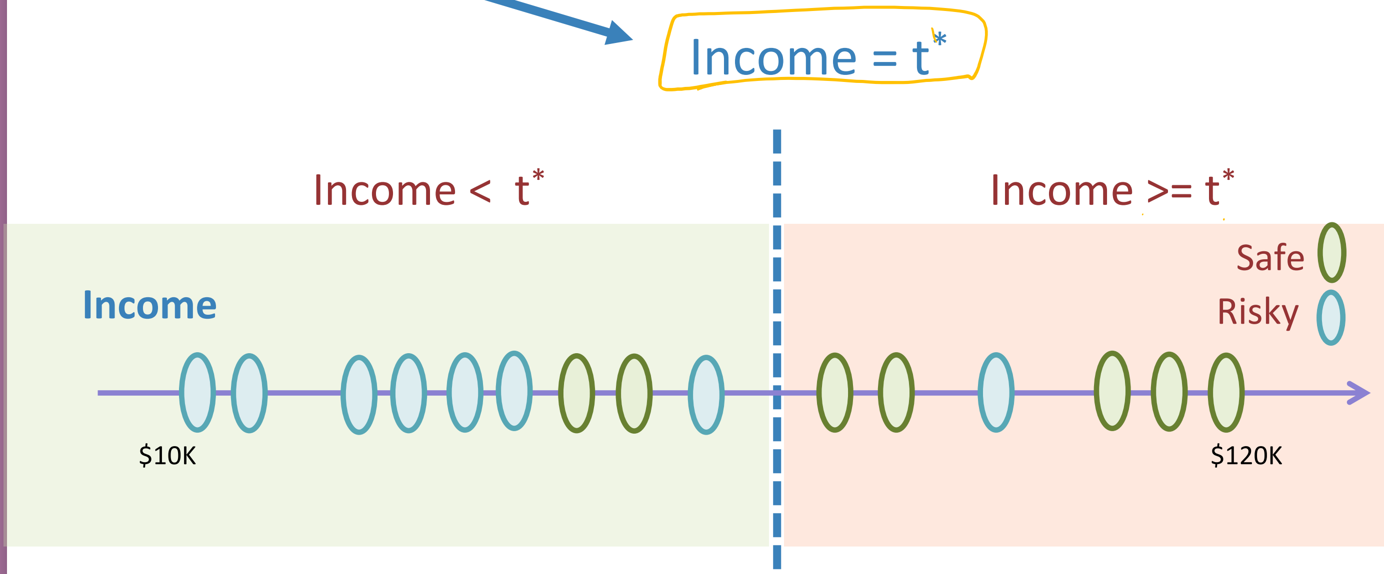
Income	Credit	Term	y
$105 K	excellent	3 yrs	Safe
$112 K	good	5 yrs	Risky
$73 K	fair	3 yrs	Safe
$69 K	excellent	5 yrs	Safe
$217 K	excellent	3 yrs	Risky
$120 K	good	5 yrs	Safe
$64 K	fair	3 yrs	Risky
$340 K	excellent	5 yrs	Safe
$60 K	good	3 yrs	Risky

Threshold split



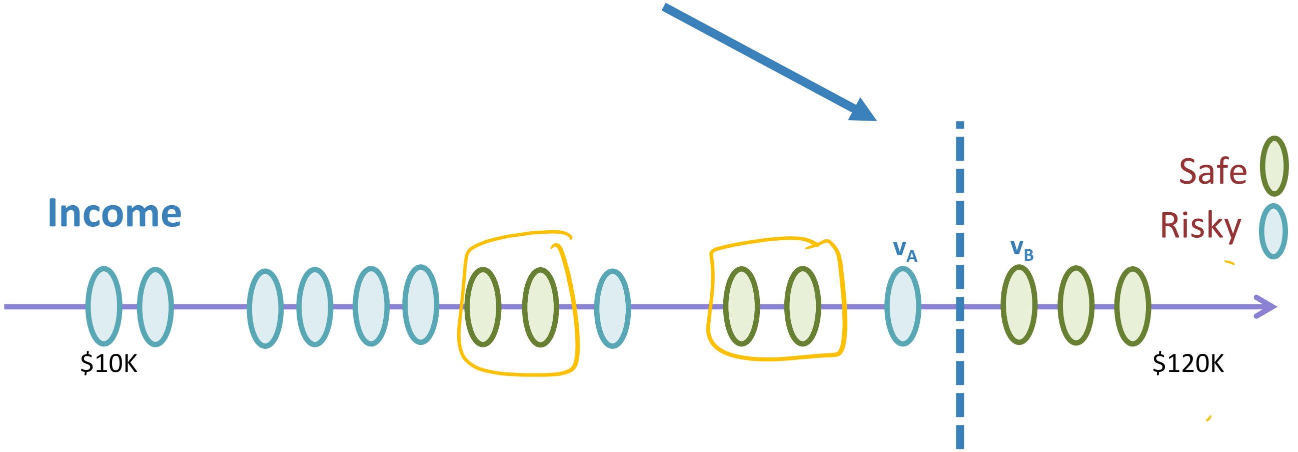
Best threshold?

Infinite possible values of t



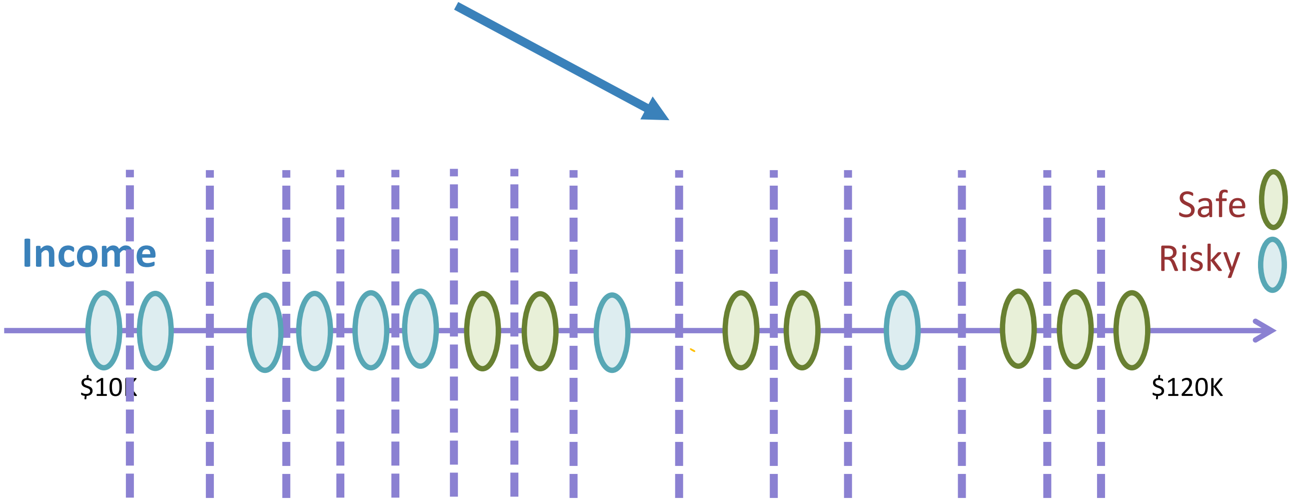
Threshold between points

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



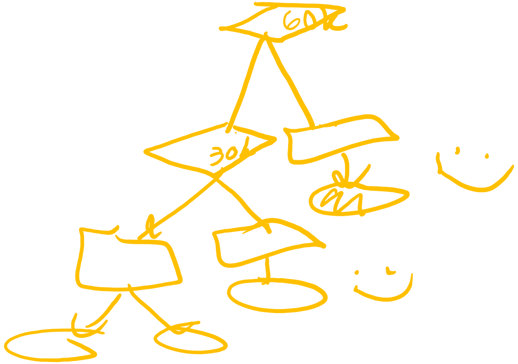
Threshold split selection algorithm

- **Step 1:** Sort the values of a feature $h_j(x)$:

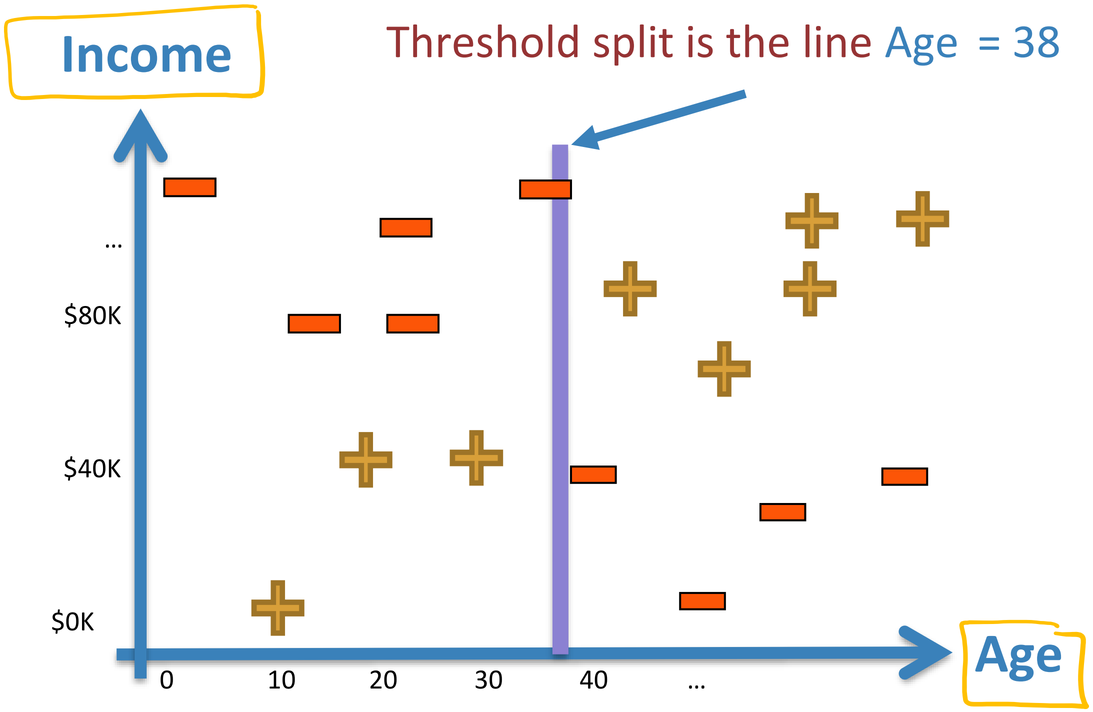
  Let $\{\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}, \ldots \mathbf{v_N}\}$ denote sorted values

- **Step 2:**
  - For i = 1 … N-1
    - Consider split $t_i = (\mathbf{v_i} + \mathbf{v_{i+1}}) / 2$
    - Compute classification error for threshold split $h_j(x) >= \mathbf{t_i}$
  - Chose the $\mathbf{t}^*$ with the lowest classification error

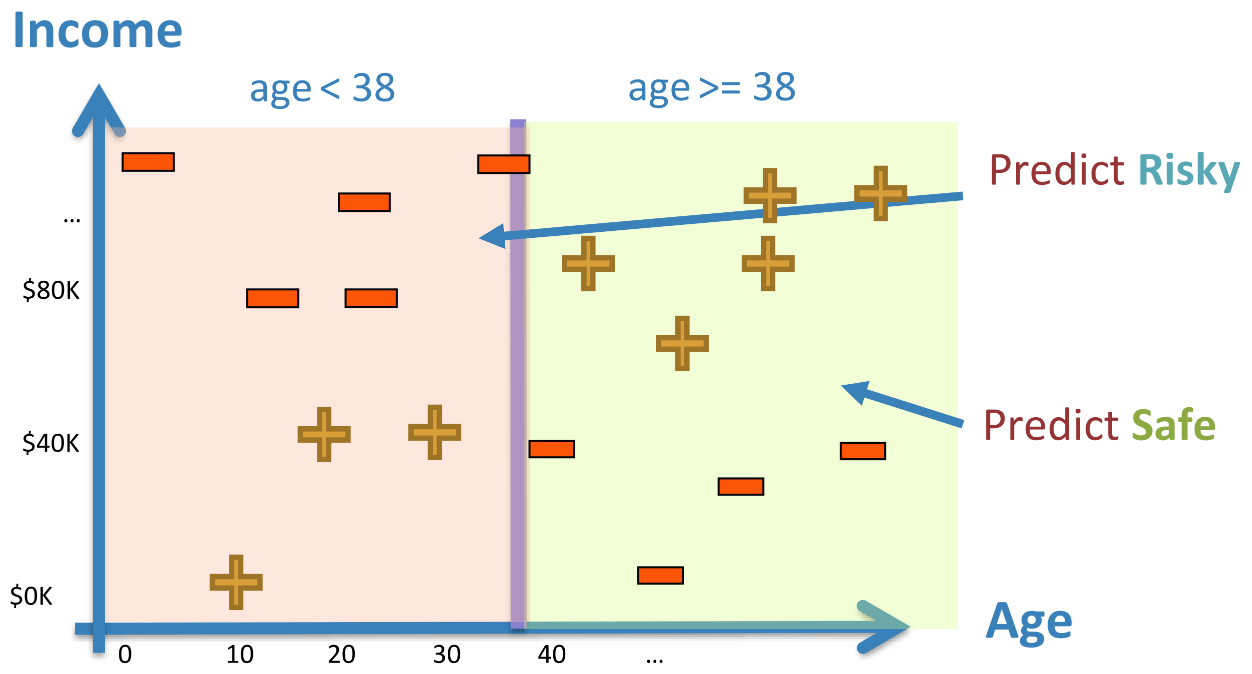
$\frac{\#\text{mistakes}}{\#\text{pts}}$



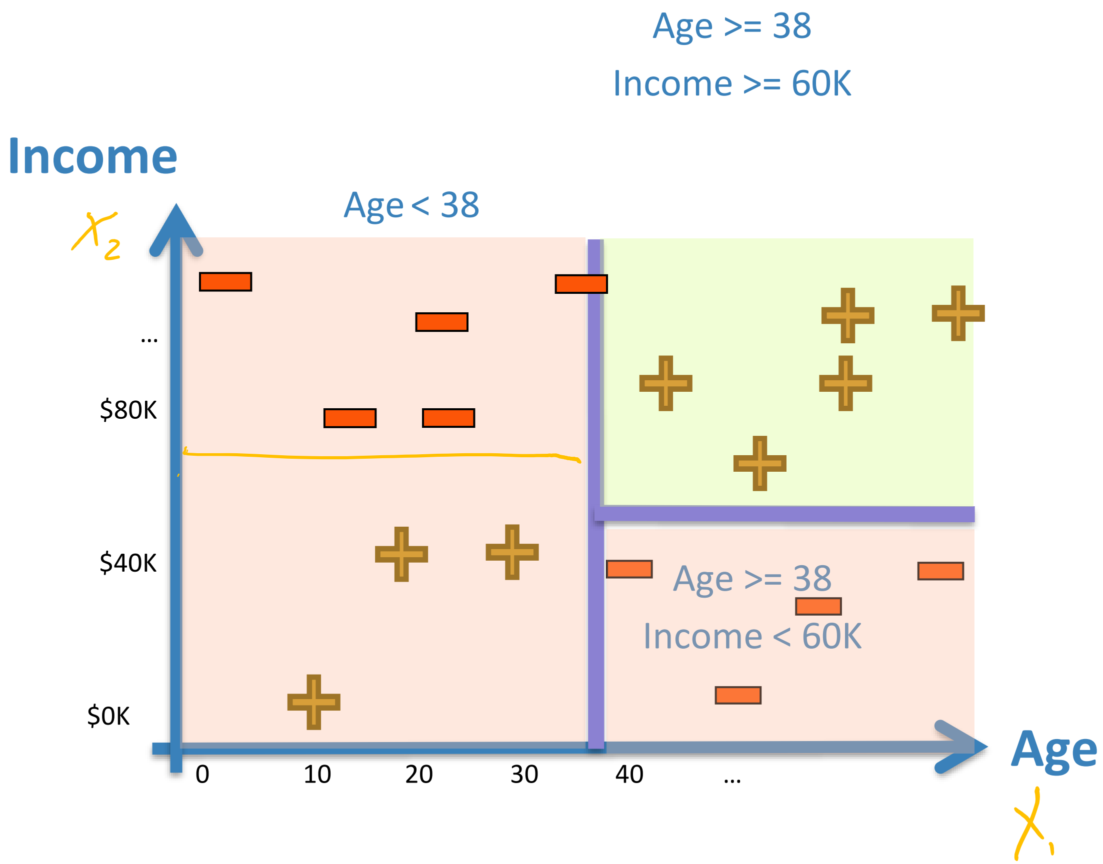
Visualizing the threshold split



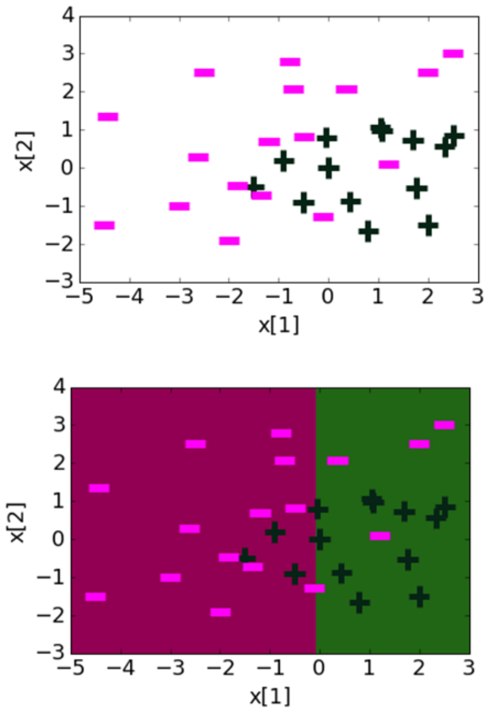
Split on Age >= 38

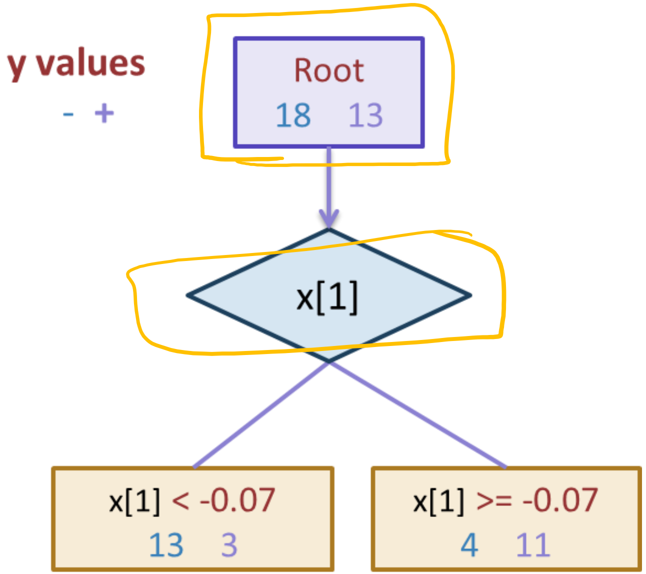


Each split partitions the 2-D space

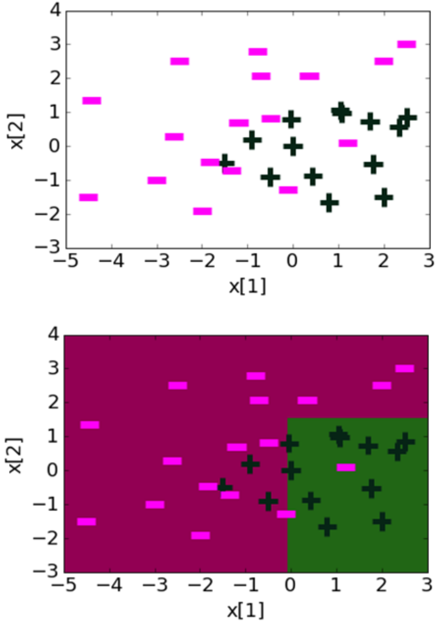


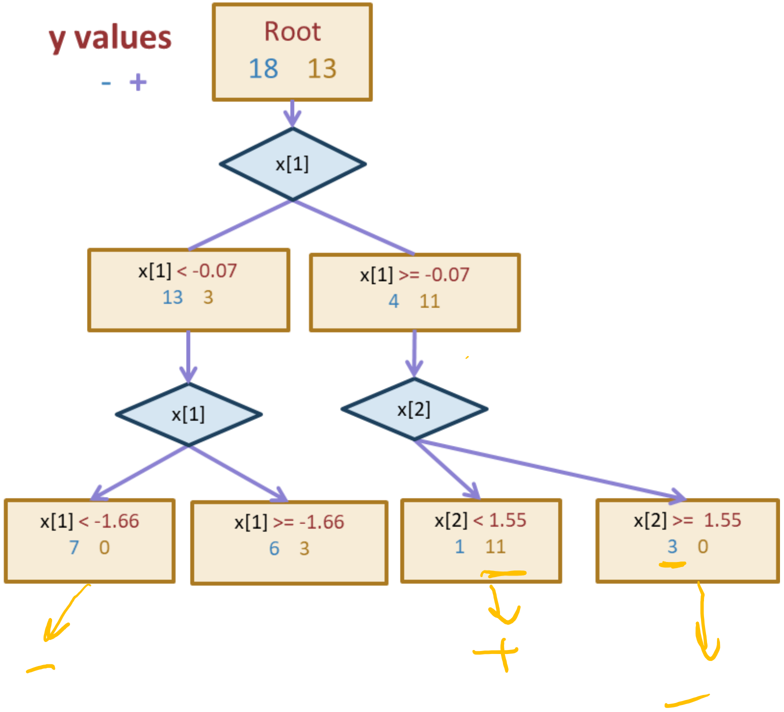
Depth 1: Split on x[1]





Depth 2

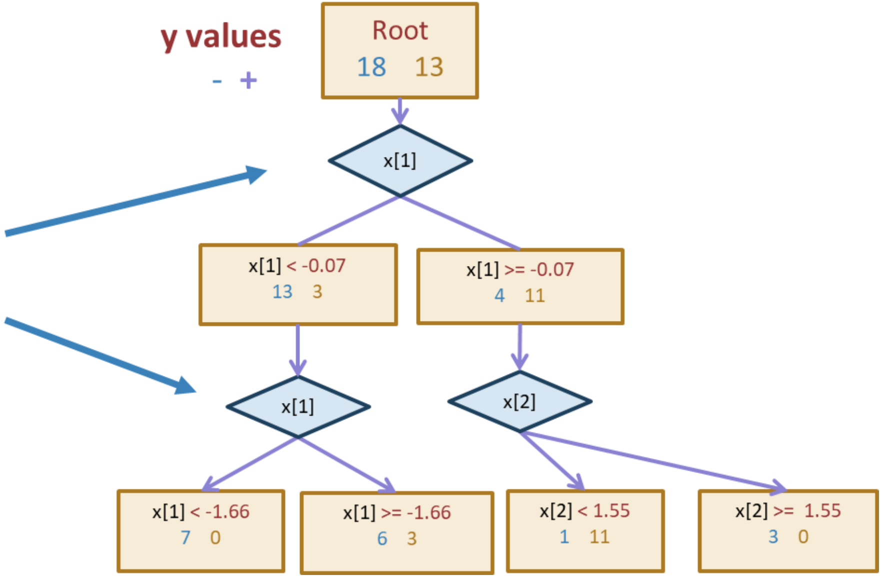




y values
- +
Root
18 13
x[1]
x[1] < -0.07
13 3
x[1] >= -0.07
4 11
x[1]
x[2]
x[1] < -1.66
7 0
x[1] >= -1.66
6 3
x[2] < 1.55
1 11
x[2] >= 1.55
3 0
-
+
-

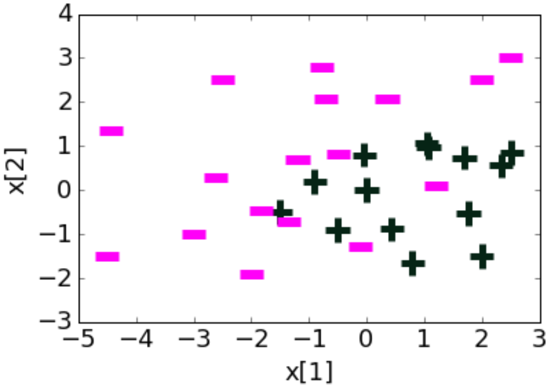
Threshold split caveat

For threshold splits, same feature can be used multiple times

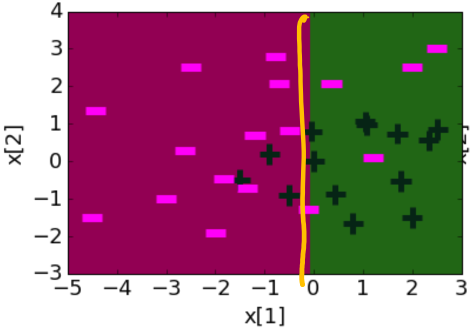


Decision boundaries

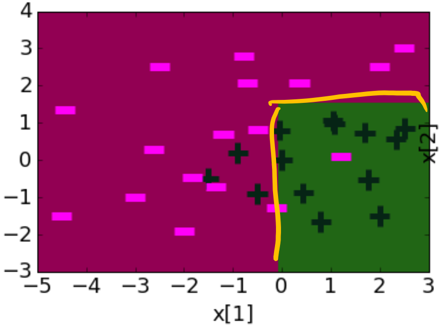
- Decision boundaries can be complex!



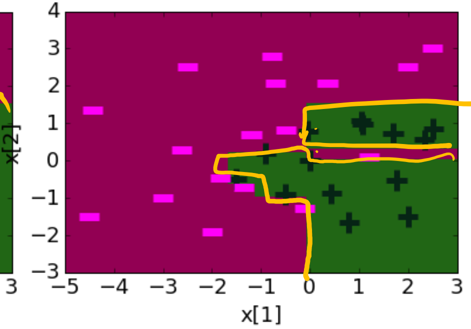
Depth 1



Depth 2

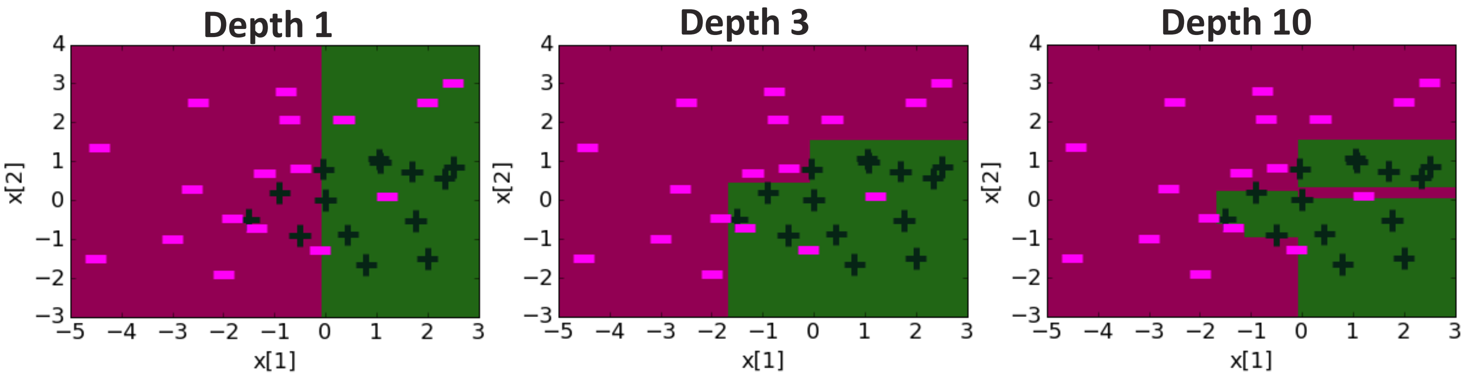


Depth 10

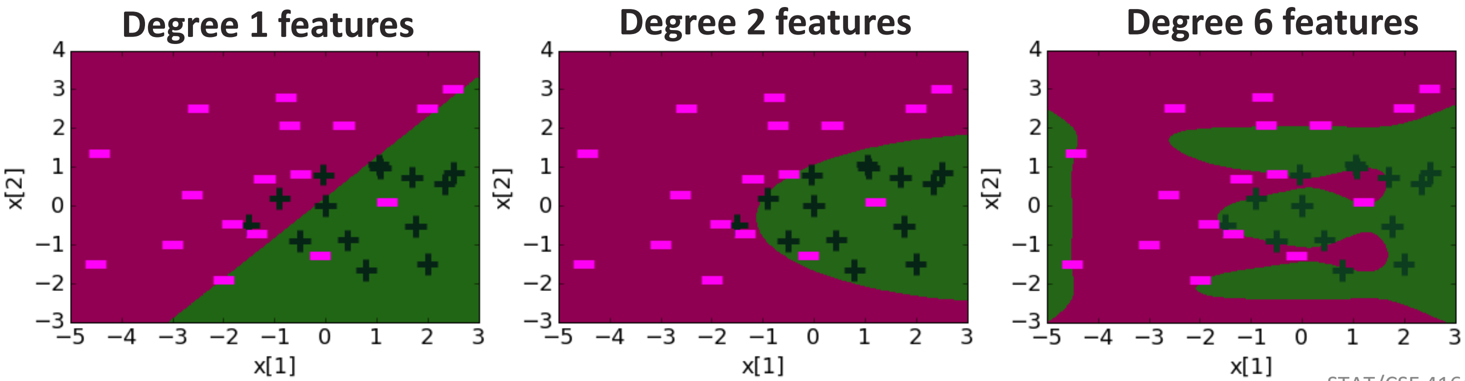


Comparing decision boundaries

Decision Tree



Logistic Regression



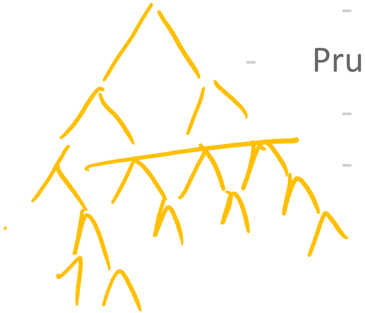
Overfitting

original stopping criteria:

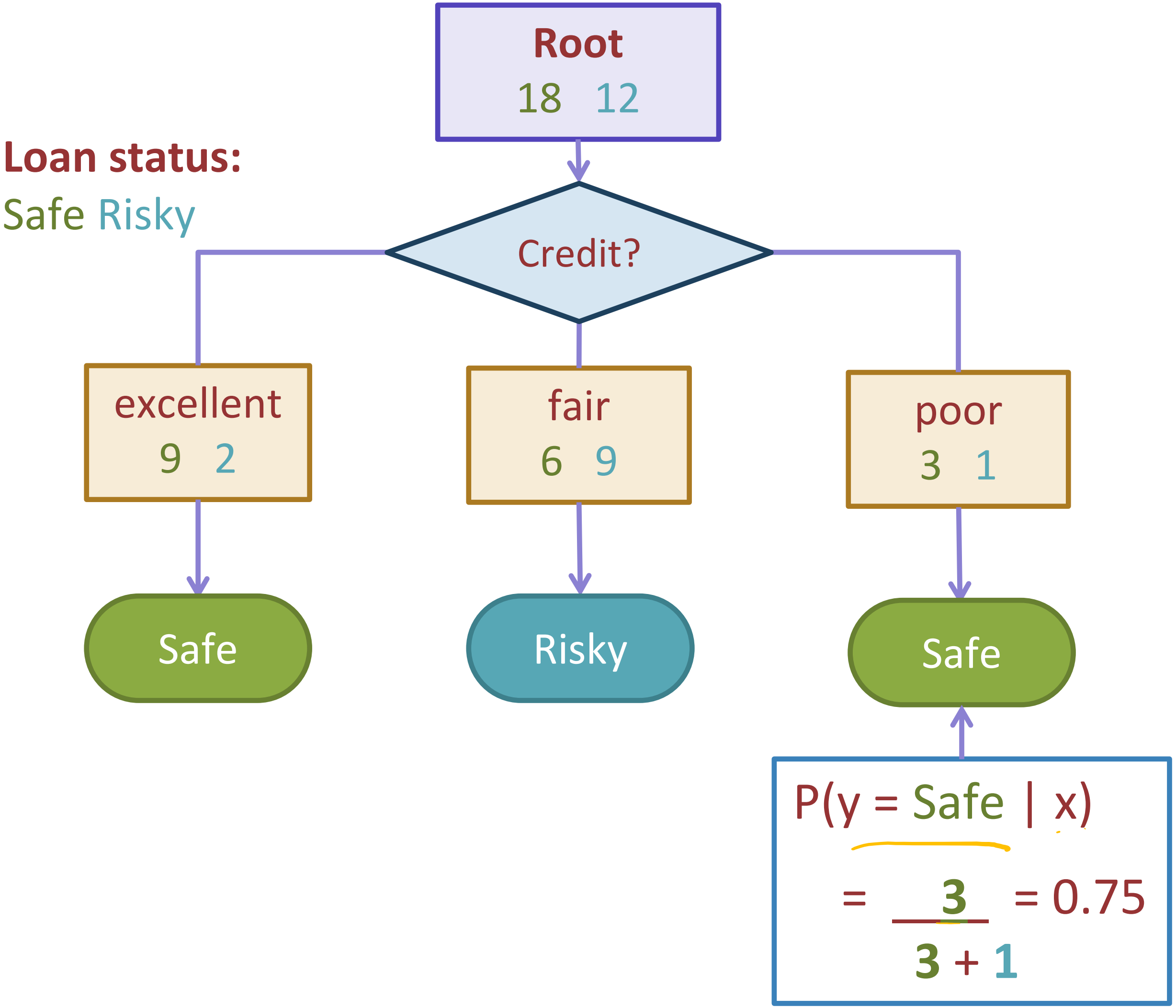
1. 0 classification error
2. no more features left

- Deep decision trees are prone to overfitting
  - Decision boundaries are interpretable but not stable
  - Small change in the dataset leads to big difference in the outcome
- Overcoming Overfitting:
  - Early stopping
    - Fixed length depth
    - Stop if error does not considerably decrease
  - Pruning
    - Grow full length trees
    - Prune nodes to balance a complexity penalty

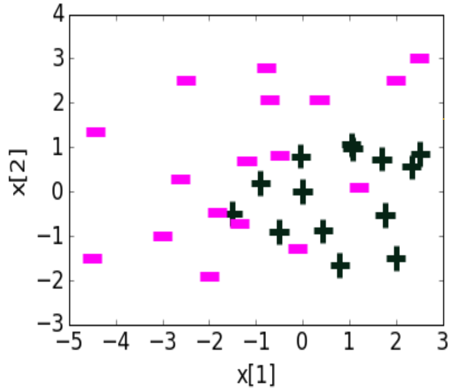
K-D trees

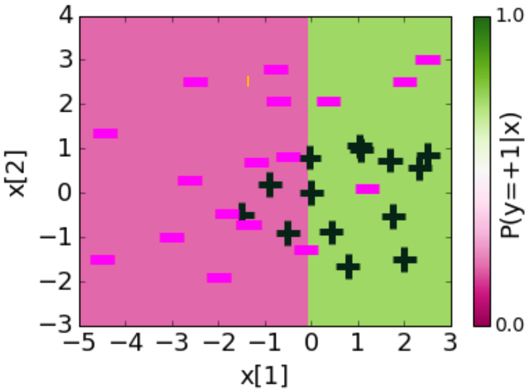


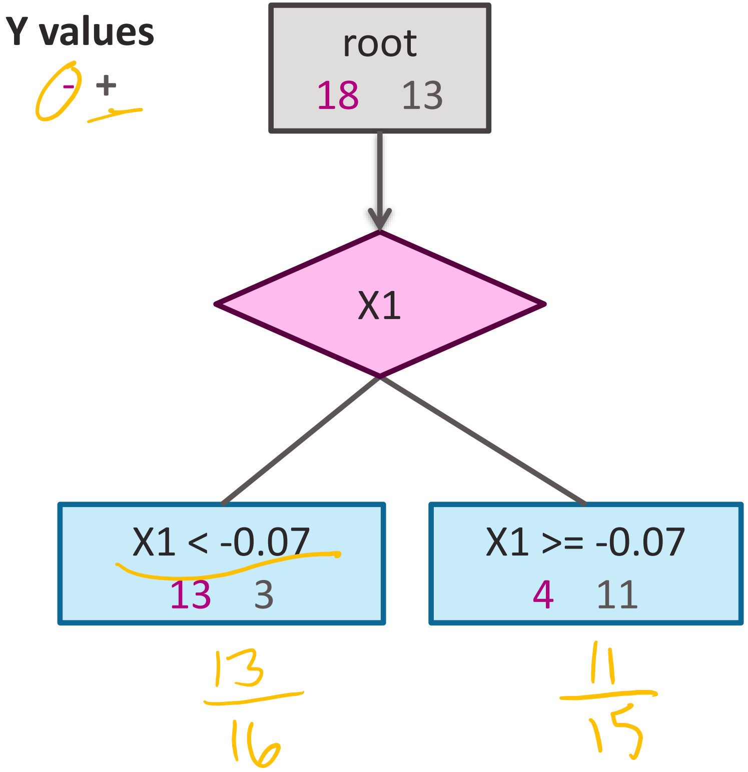
Predicting probabilities



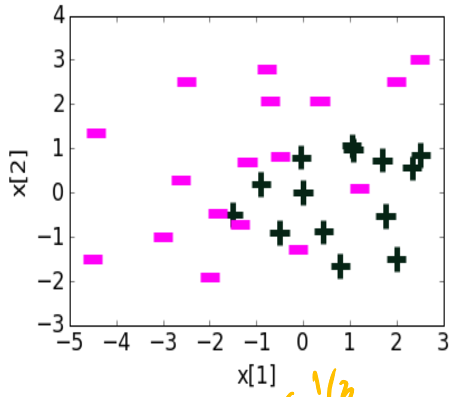
Depth 1 probabilities

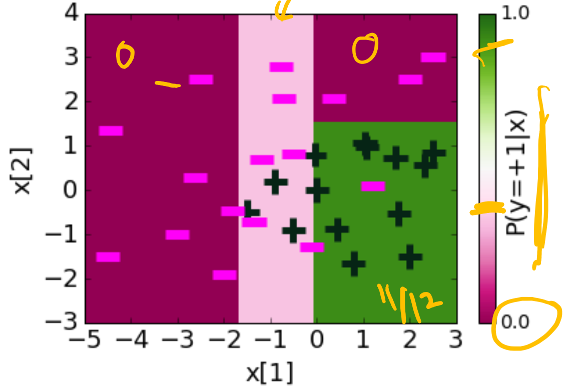






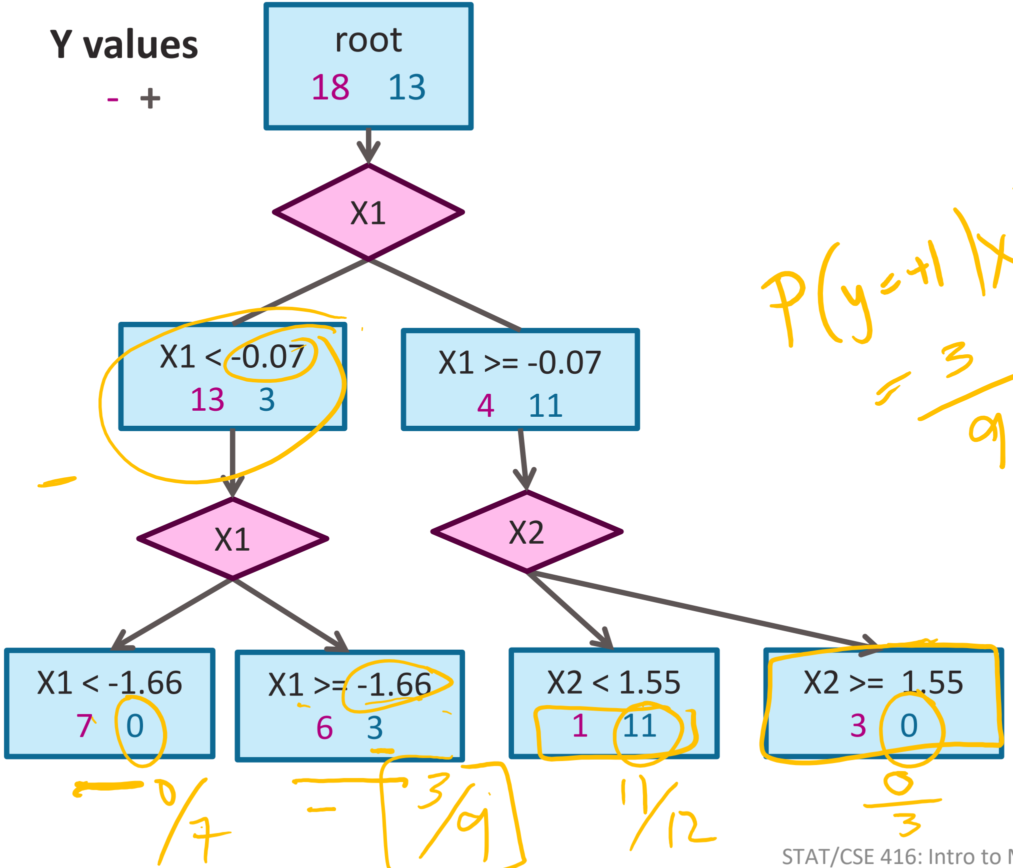
Depth 2 probabilities





Y values

- +



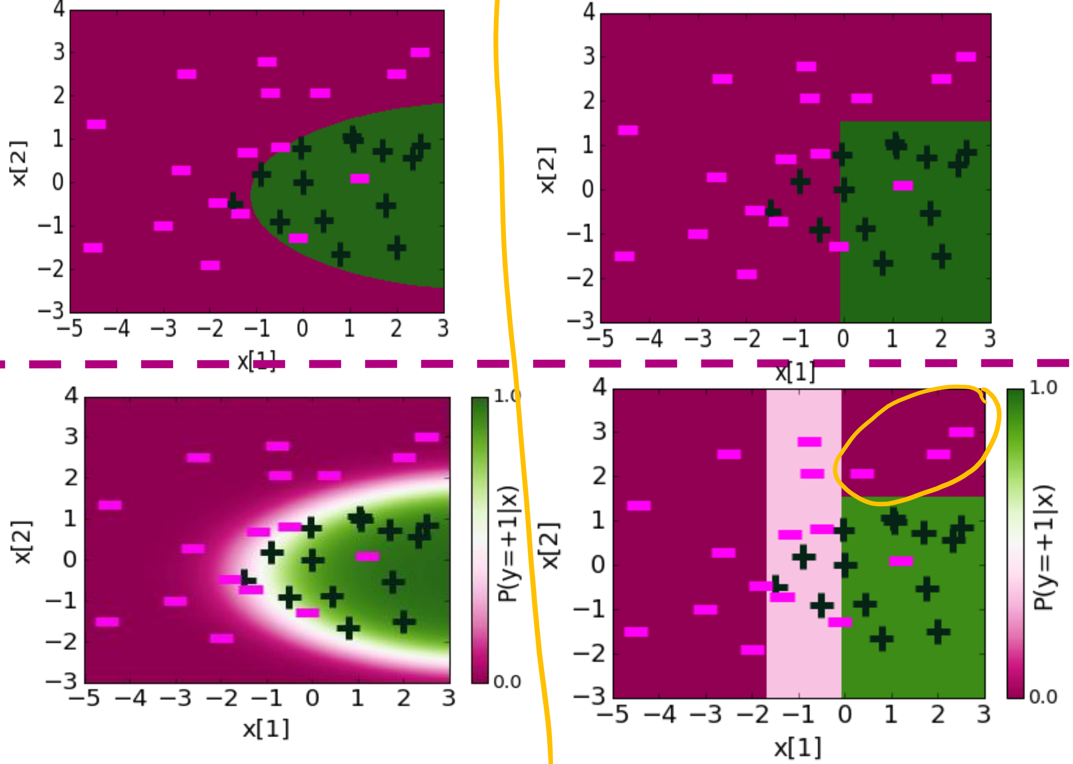
$P(y=+1|x) = \frac{3}{9} = \frac{1}{3}$

0.33

Comparison with logistic regression

Logistic Regression (Degree 2)

Decision Trees (Depth 2)



Class

Probability

Recap

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