Decision Trees and Random Forests

Ivan Corneillet

Data Scientist

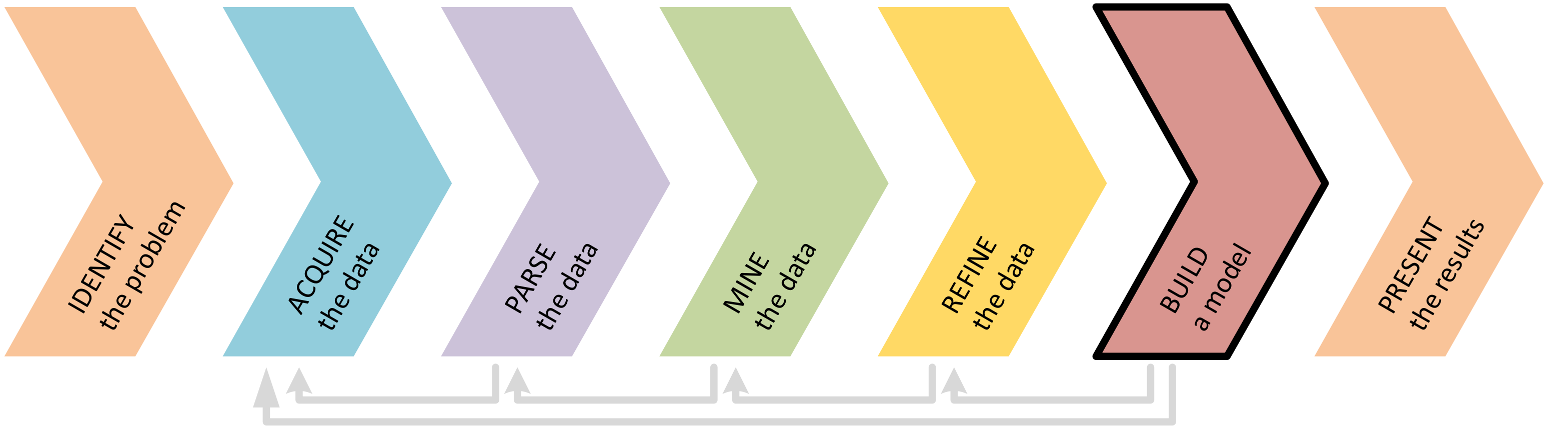
Learning Objectives

After this lesson, you should be able to:

- Understand and build decision tree models for classification and regression
- Understand the differences between linear and non-linear models
- Understand and build random forest models for classification and regression
- Know how to extract the most important predictors in a random forest model

Today, we will mine a new dataset and build a new type of model (decision tree) that we will then refine (random forests) to get the best possible predictive ability

Unit 1 – Research Design and Data Analysis	*Research Design*	*Data Visualization in Pandas*	*Statistics*	*Exploratory Data Analysis in Pandas*
Unit 2 – Foundations of Modeling	Linear Regression	Classification Models	Evaluating Model Fit	Presenting Insights from Data Models
Unit 3 – Data Science in the Real World	*Decision Trees and Random Forests*	*Time Series Data*	*Natural Language Processing*	*Databases*



Outline

- Review
- Our first classifiers revisited
- Decision Trees
  - The 2008 democratic primaries + Codealong
  - Entropy
  - Training a classification decision tree
  - Codealong: Building the 2008 democratic primaries decision tree by hand
  - Training a regression decision tree
  - Overfitting
- Random Forests
  - Training and prediction
- Unit Project 4's Presentations (cont.)
- Lab (Decision trees and random forests with *scikit-learn*)
- Review
- In-flight
  - Final Project 2 (due in 1 week)

Review

Activity: Knowledge Check

EXERCISE

ANSWER THE FOLLOWING QUESTIONS (5 minutes)

1. Define the difference between the precision and recall of a model
2. What are some common components and use cases for logistic regression?
3. When finished, share your answers with your table

DELIVERABLE

Answers to the above questions

Precision ($precision = \frac{TP}{TP+FP}$) and recall ($recall = \frac{TP}{P} = TPR$)

- With *precision*, we're interested in producing a high amount of relevancy instead of irrelevancy
- Precision asks, "Out of all of our positive predictions (both true positive and false positive), how many were correct?"
- With *recall*, we're interesting in seeing how well a model returns specific data (literally, checking whether the model can recall what a class label looked like)
- Recall asks, "Out of all of our positive class labels, how many were correct?"

Q & A

Pre-Work

Pre-Work

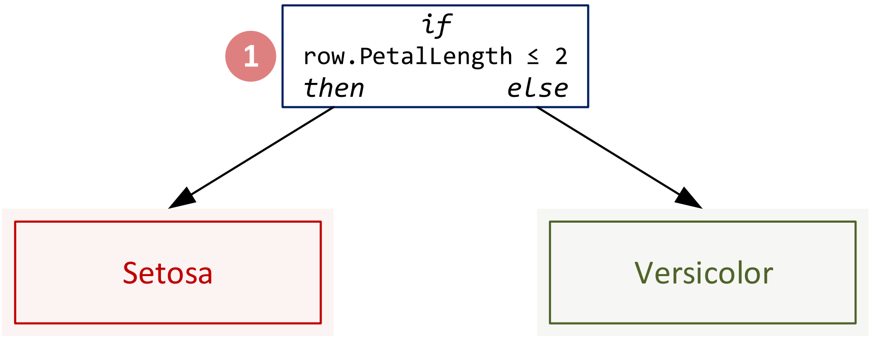
Before this lesson, you should already be able to:

- Explain the concepts of cross-validation, logistic regression, and overfitting
- Know how to build and evaluate some classification model in *scikit-learn* using cross-validation and AUC

Introduction to Decision Trees

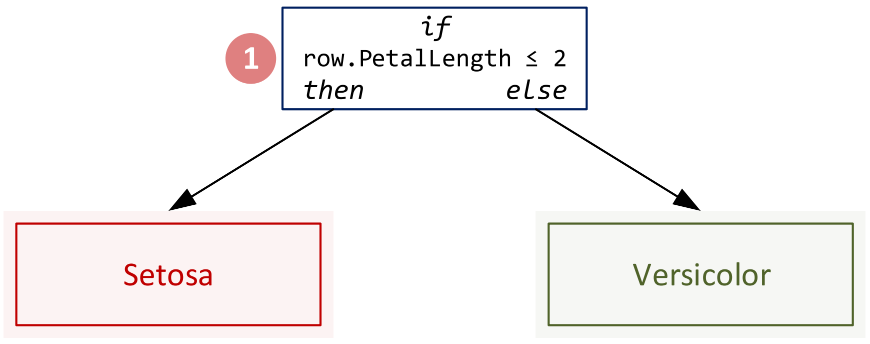
Our first classifier (session 8) can be understood as a decision tree

```
def my_first_classifier(row):
  if row.PetalLength < 2:
    return 'Setosa '
  else:
    return 'Versicolor'
```

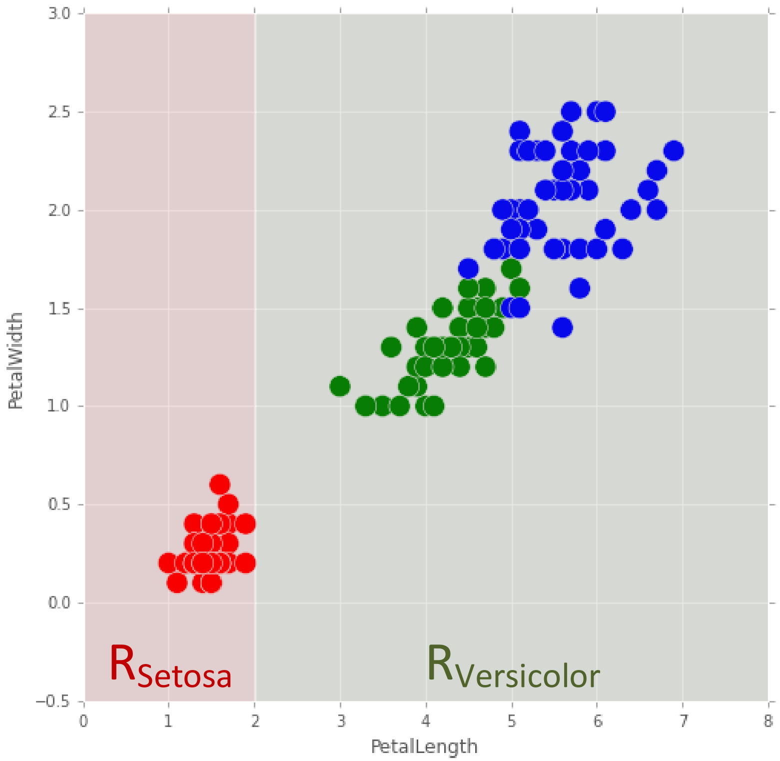


Our first classifier (session 8) can be understood as a decision tree (cont.)

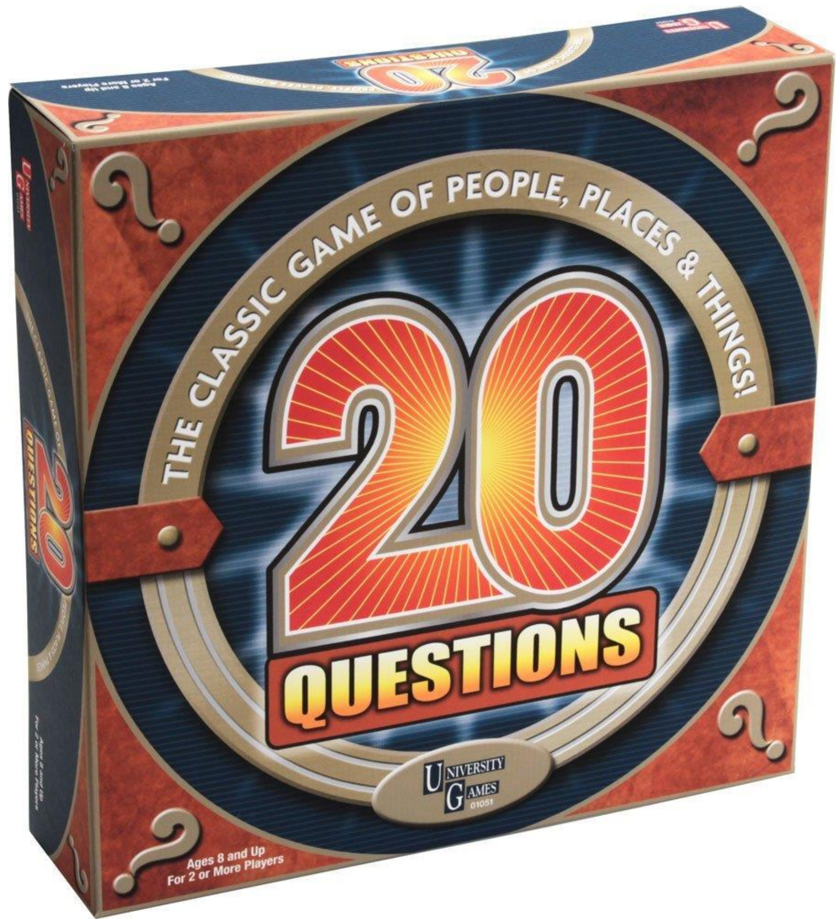
Decision Tree

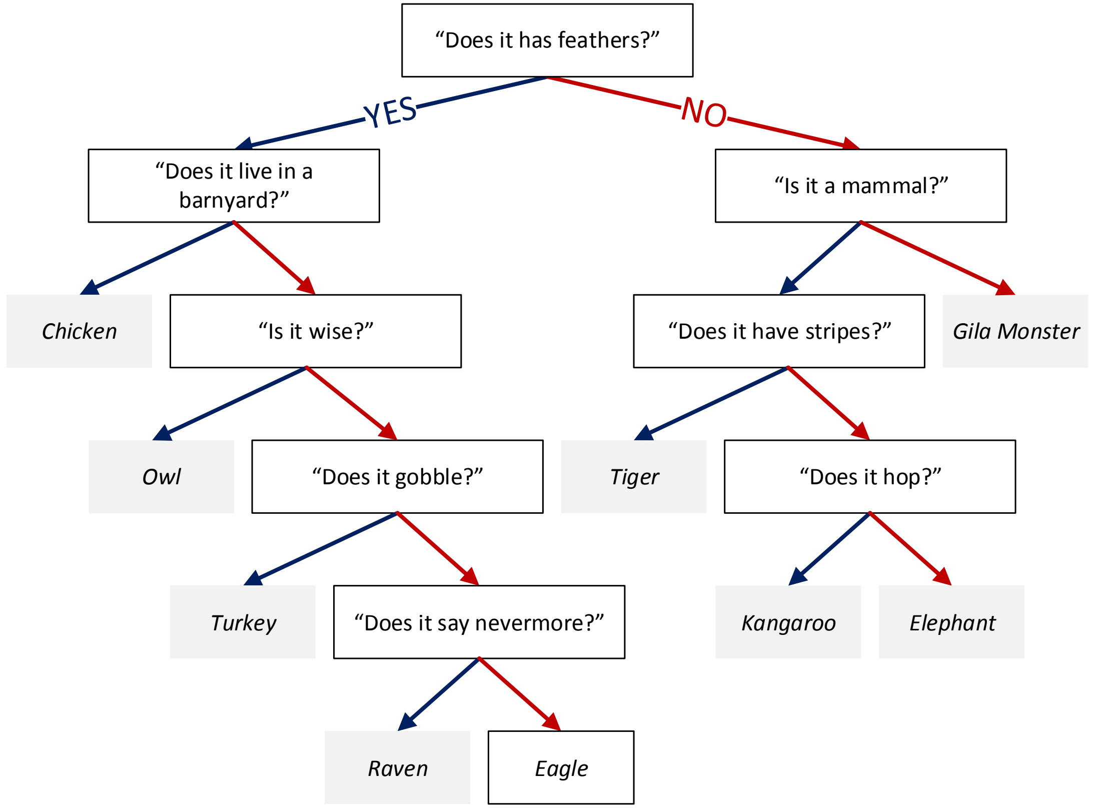


Feature Space

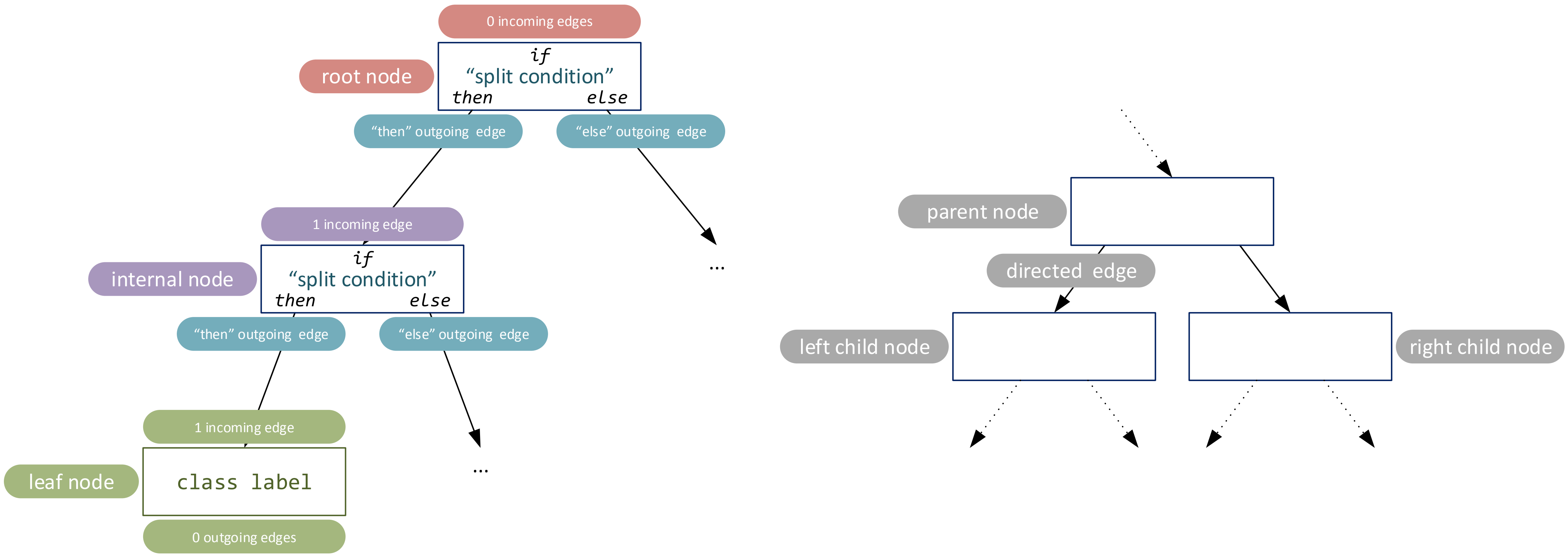


Decision trees are like the game “20 questions”. They make decision by answering a series of questions, most often binary questions. (yes or no)





Structure of decision trees



Activity: `my_second_classifier` as a decision tree

EXERCISE

ANSWER THE FOLLOWING QUESTION (5 minutes)

1. Turn our second hand-coded classifier (`my_second_classifier` from session 8) into a decision tree

2. When finished, share your answer with your table

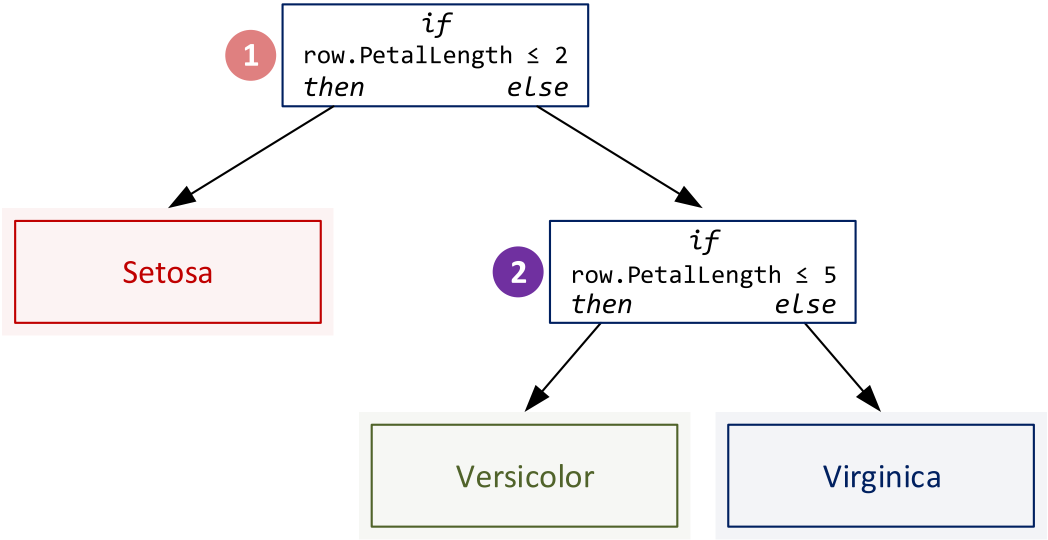
DELIVERABLE

Answers to the above question

```
def my_second_classifier(row):
  if row.PetalLength < 2:
    return 'Setosa '
  elsif row.PetalLength < 5:
    return 'Versicolor'
  else:
    return 'Virginica'
```

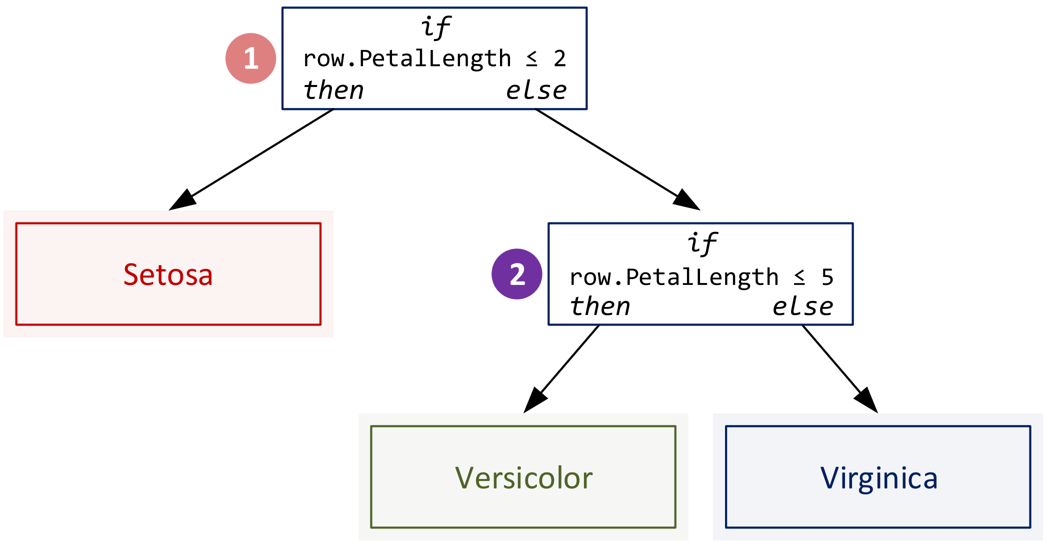
Activity: `my_second_classifier` as a decision tree (cont.)

```
def my_second_classifier(row):
  if row.PetalLength < 2:
    return 'Setosa '
  elsif row.PetalLength < 5:
    return 'Versicolor'
  else:
    return 'Virginica'
```

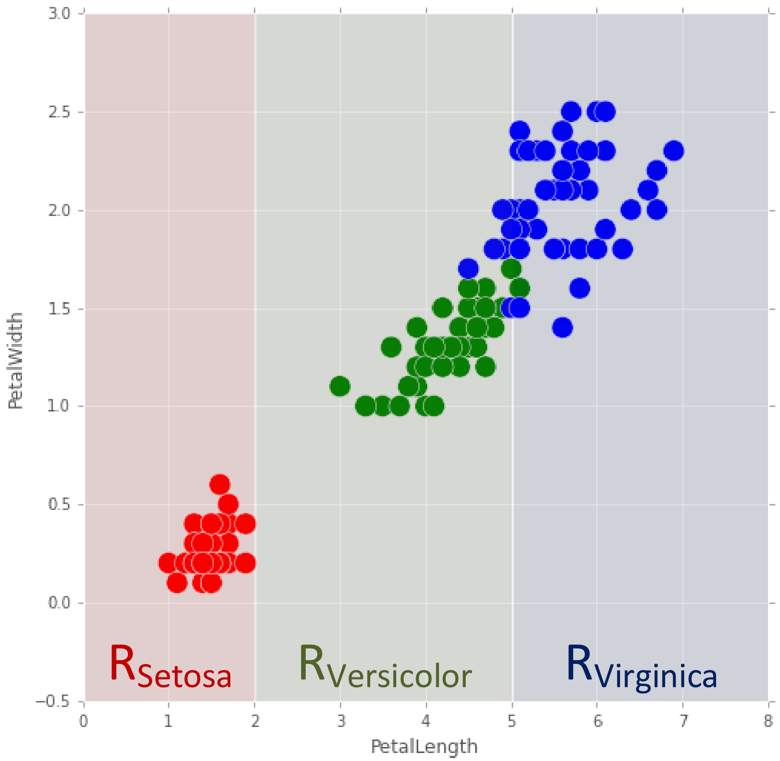


Activity: `my_second_classifier` as a decision tree (cont.)

Decision Tree



Feature Space



How to use decision trees to make predictions?

```
predict(x):
        # i. start from the root node at the top
        set the root as the current node

        # ii. traverse the tree top-down
        while the current node is not a leaf:
                if the current node's split condition (using x) is true:
                        set the left child as current node
                otherwise:
                        set the right child as current node

        # iii. stop when reaching a leaf node
        return the class pointed by the current (leaf) node
```

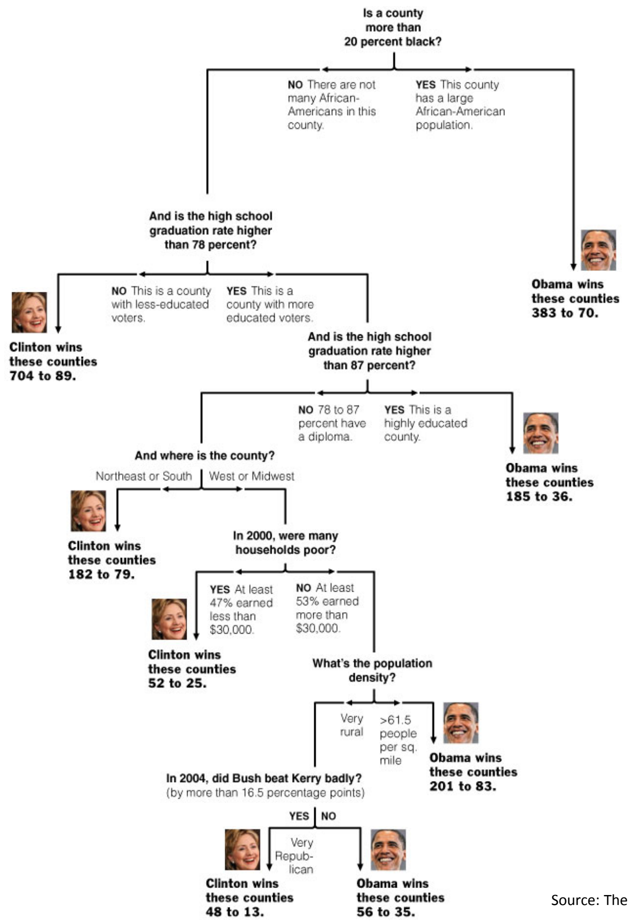
How to build Decision Trees?

The 2008 Democratic Primaries

Motivating Example: The 2008 Democratic Primaries

EXAMPLE

- Decision Tree: The Obama-Clinton Divide
  - Published in April 16, 2008 while the Democratic Primaries were still running



Source: The New York Times

Activity: The 2008 Democratic Primaries

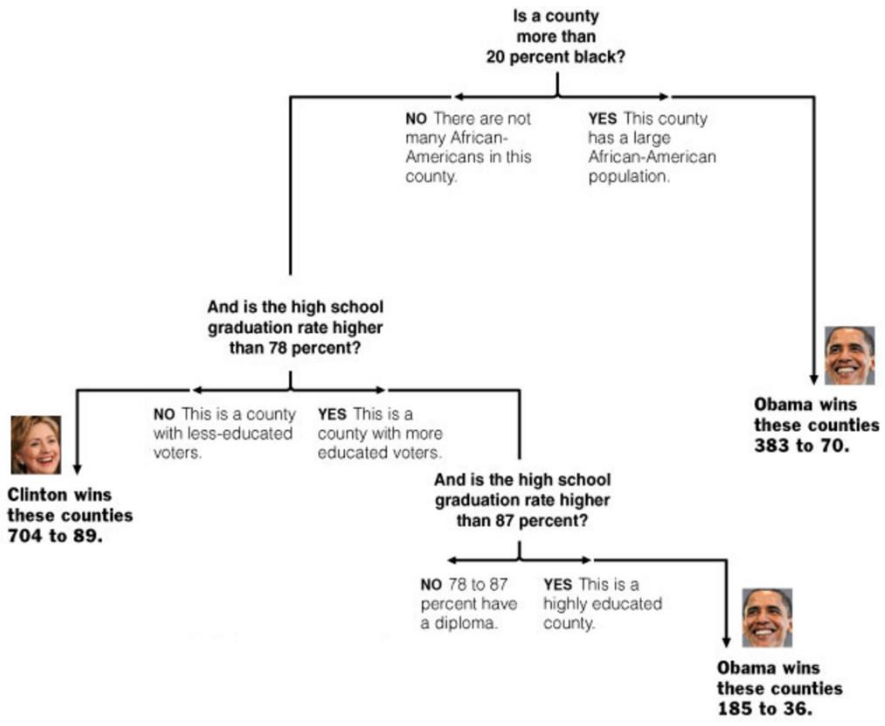
EXERCISE

ANSWER THE FOLLOWING QUESTIONS (10 minutes)

1. In a couple of sentences, describe what counties senators Obama and Clinton have won so far in the nomination contest
2. What questions do you have for the data scientist on how she created this decision tree?
3. When finished, share your answers with your table

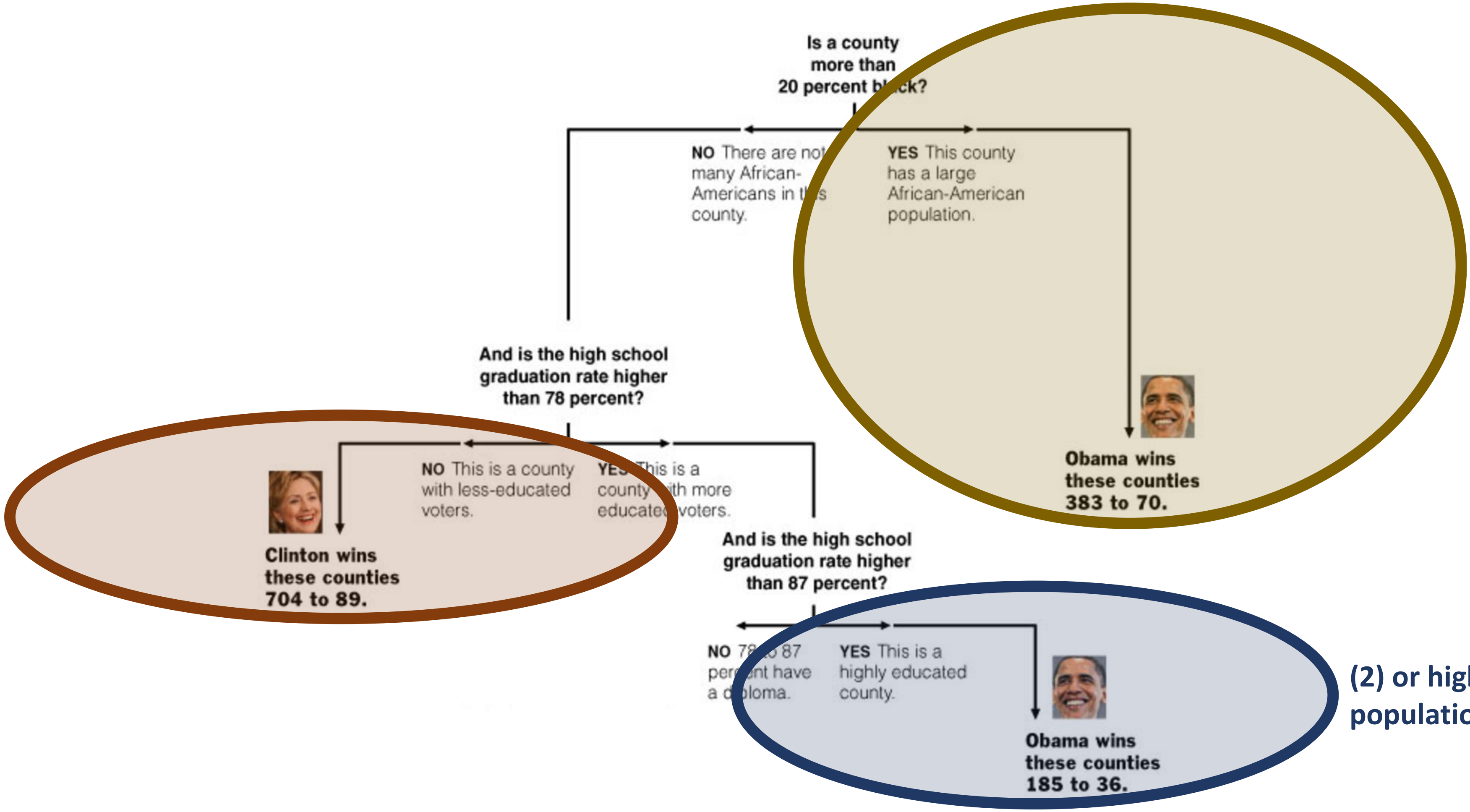
DELIVERABLE

Answers to the above questions



Activity: “In the nominating contests so far, …

(3) Senator Hillary Rodham Clinton has a commanding lead in less-educated counties dominated by whites”



(1) Senator Barack Obama has won the vast majority of counties with large black populations

(2) or highly educated populations.

Activity (cont.)

- “In the nominating contests so far, Senator Barack Obama has won the vast majority of counties with large black or highly educated populations. Senator Hillary Rodham Clinton has a commanding lead in less-educated counties dominated by whites”

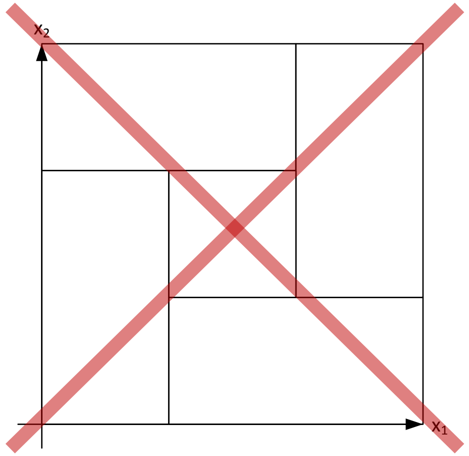
- Surely an over-simplification but it is easy to display, interpret, and explain

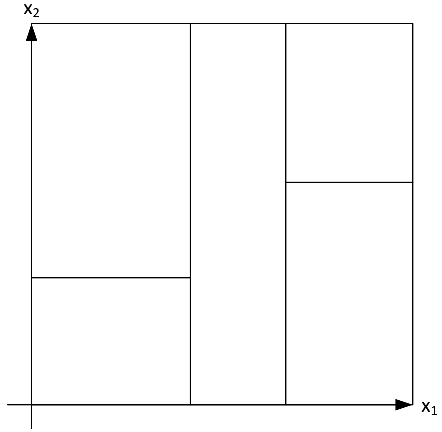
Codealong – Part A
The 2008 Democratic Primaries

The 2008 Democratic Primaries (cont.)

Details of the training process

- We divide the feature space $x = (x_1, \dots, x_k)$ into distinct and non-overlapping regions
    - This is also called stratification or segmentation
- In theory, the regions could have any shape
    - However, we choose to divide the feature space into high dimensional rectangles, or boxes, for simplicity and for ease of interpretation of the resulting predictive model





Activity: Open questions

- ❶ How to choose the split conditions? (variables and threshold values)
  - E.g., why is the threshold for African-American population set at 20%?
- ❷ How to choose the order of the conditions?
  - E.g., why is the first split on African-American population vs. the voters' education level?
- ❸ When do we stop?

Because it isn't computationally feasible to consider every possible partition of the feature space, we take a *top-down*, *greedy* approach known as recursive binary splitting

Top-Down

- The approach begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree

Greedy

- At each step of the tree-building process, the *best* split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step

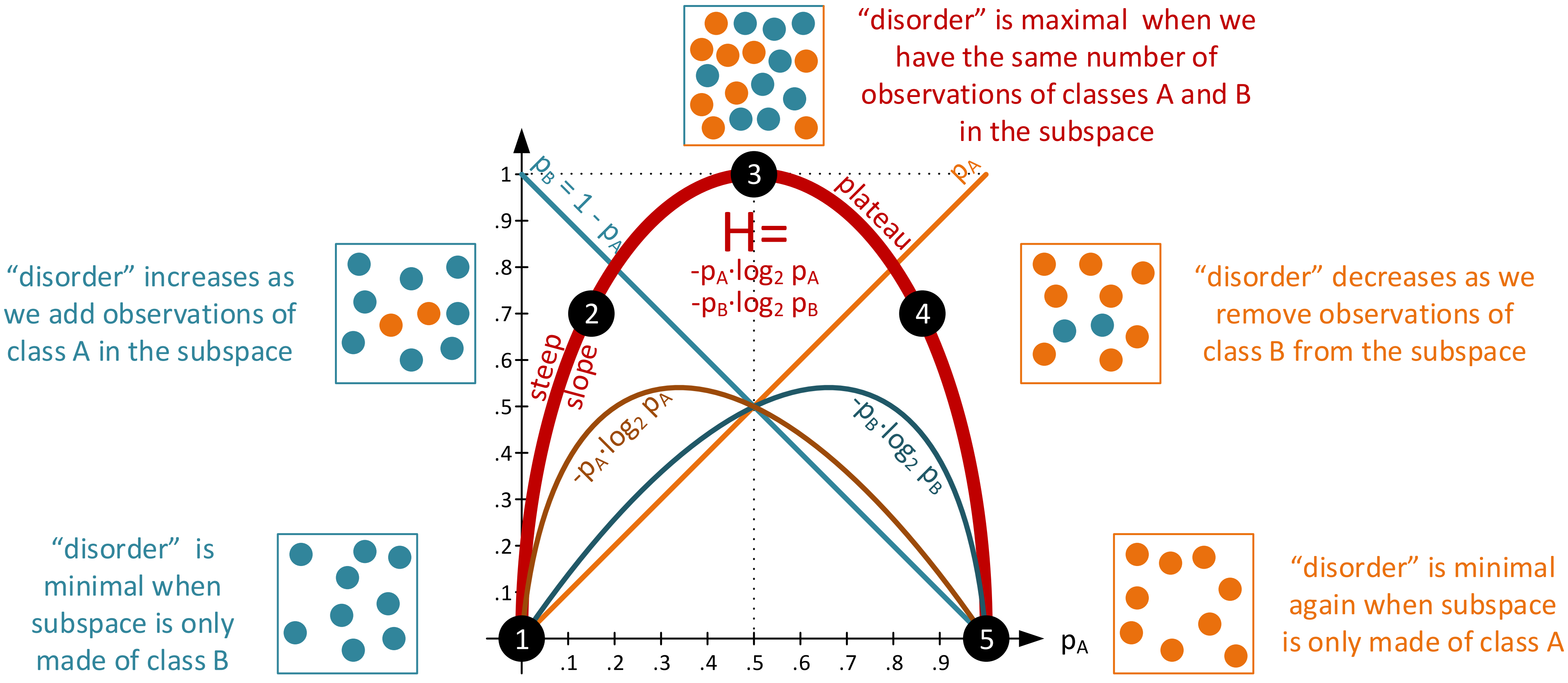
Decision trees can be applied to both classification and regression problems

- We first consider classification problems to address ❷ (How to choose the order of the conditions?)

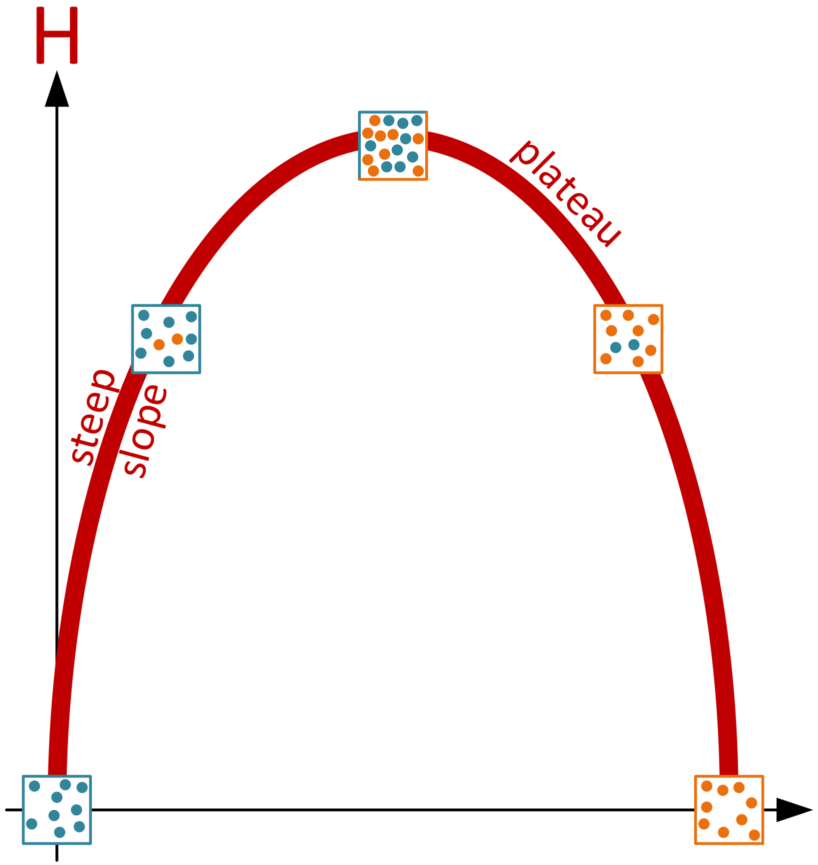
- We'll then move on to regression problems when addressing ❶ (How to choose the split conditions?)

Entropy

Entropy (H) is a measure of disorder



Entropy: What to remember



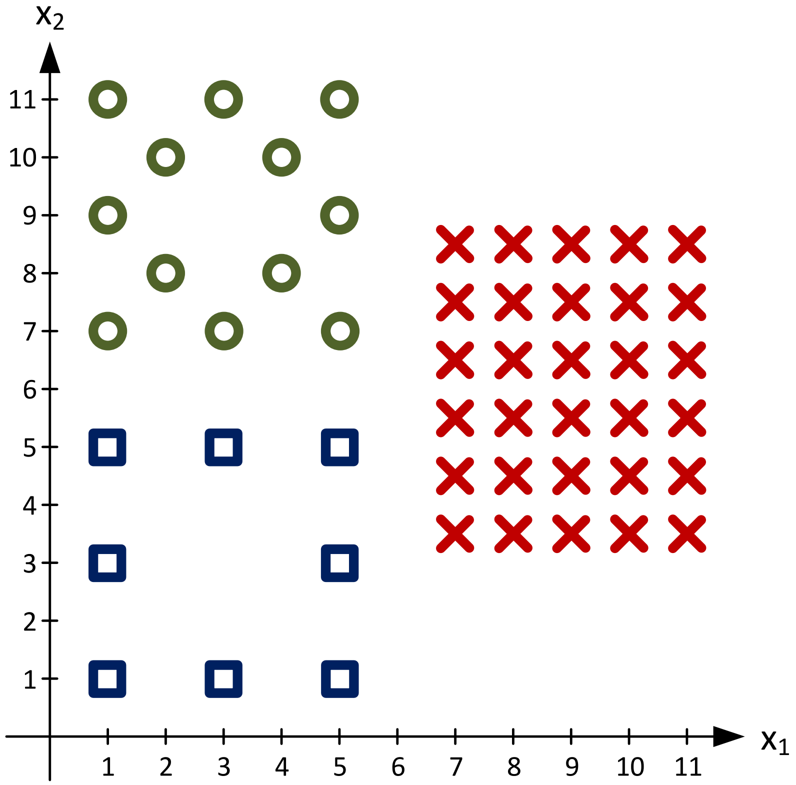
$$H = -\sum_{i=1}^{k} p_i \cdot log_2(p_i)$$

(p_i represents the proportion of observations in the region that are from the i^{th} class)

Training a Classification Decision Tree ❷ How to choose the order of the conditions?

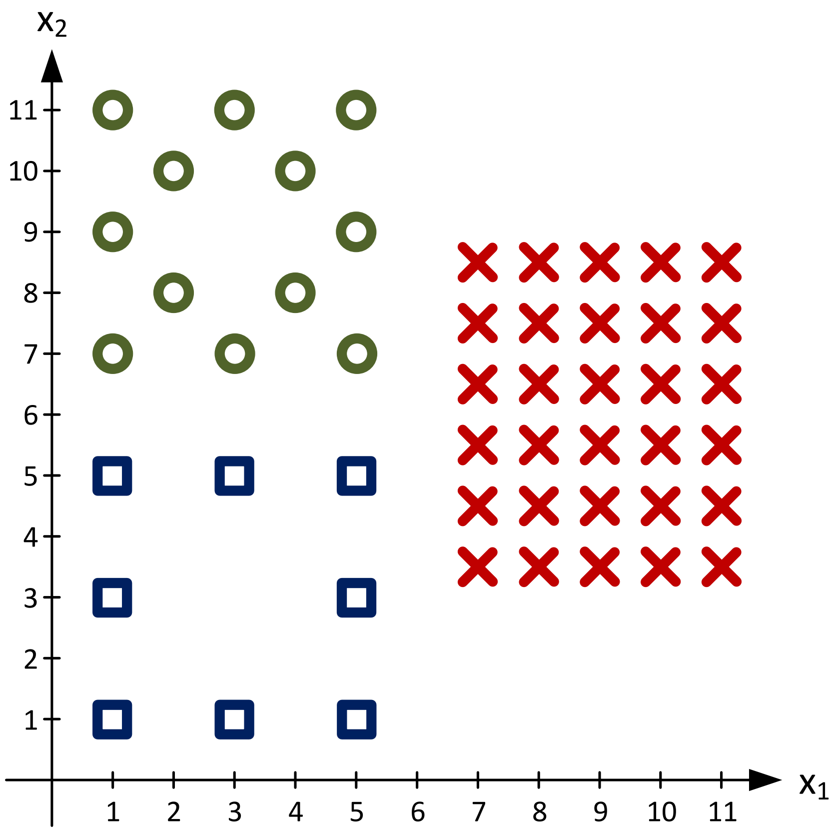
Let's learn a decision tree from the following training set

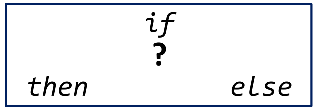
EXAMPLE



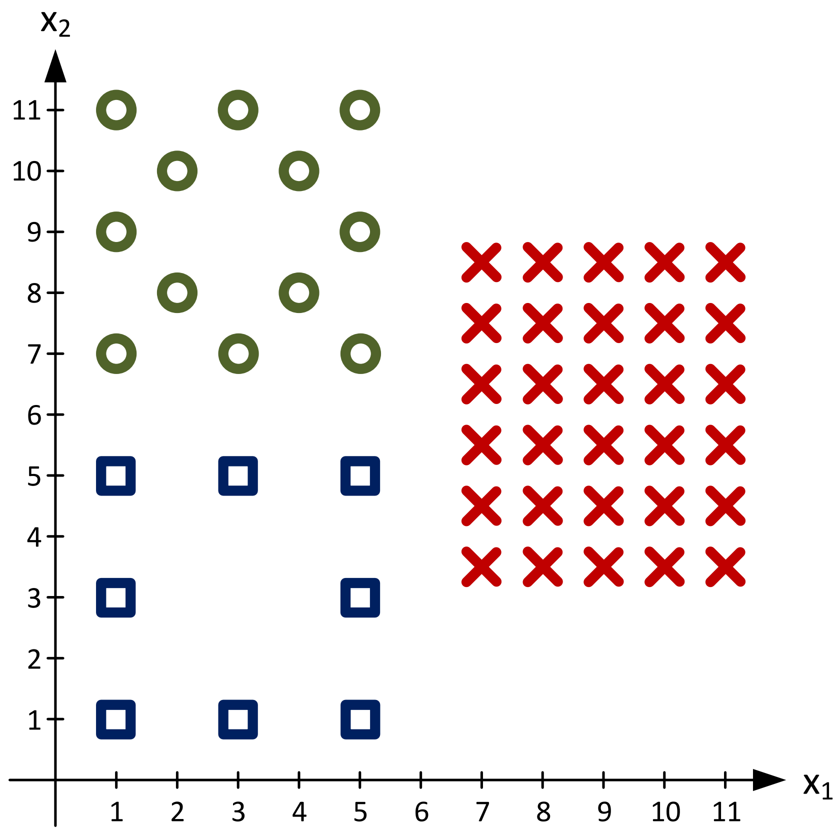
- 3 classes
  - "red", "green", and "blue"
- 50 data points
  - 30 "red"
  - 12 "green"
  - 8 "blue"

What's the first cut?





❶ What's the entropy of the data set/root node? (before the first split) What's your intuition of its level? ("null", "high", "~max")



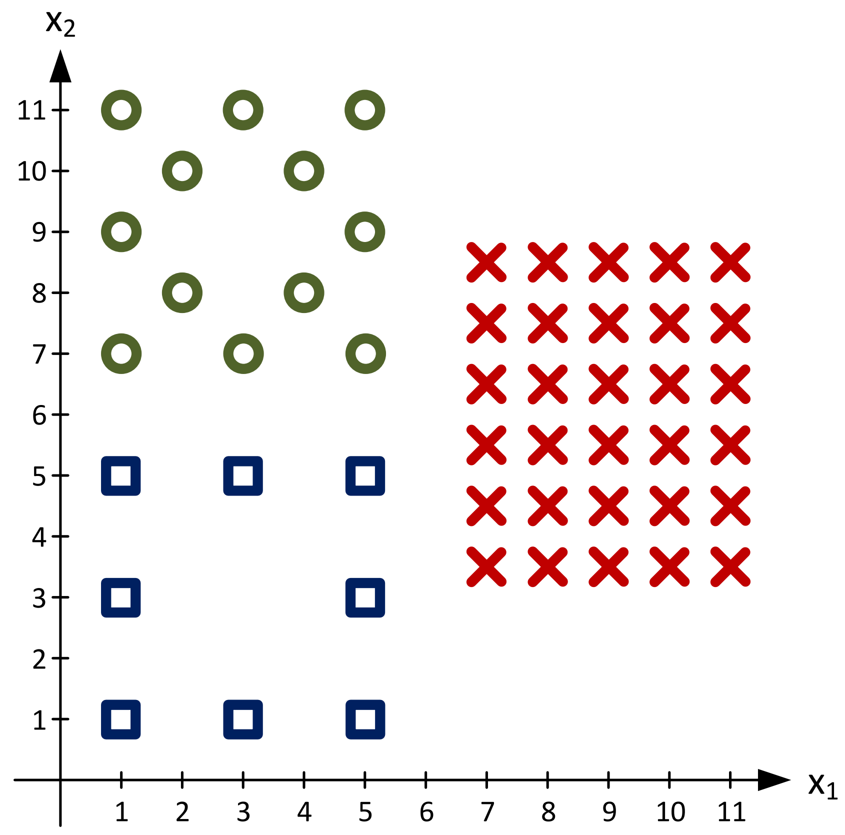
root node				
	red	green	blue	overall
c_i	?	?	?	?
p_i	?	?	?	?
H		?		

$$n = \sum_{i=1}^{k} c_i$$

$$p_i = \frac{c_i}{n}$$

$$H = -\sum_{i=1}^{k} p_i \cdot log_2(p_i)$$

❶ The entropy of the data set/root node is "high" at 1.36. The intuition is that we have three classes in significant proportion



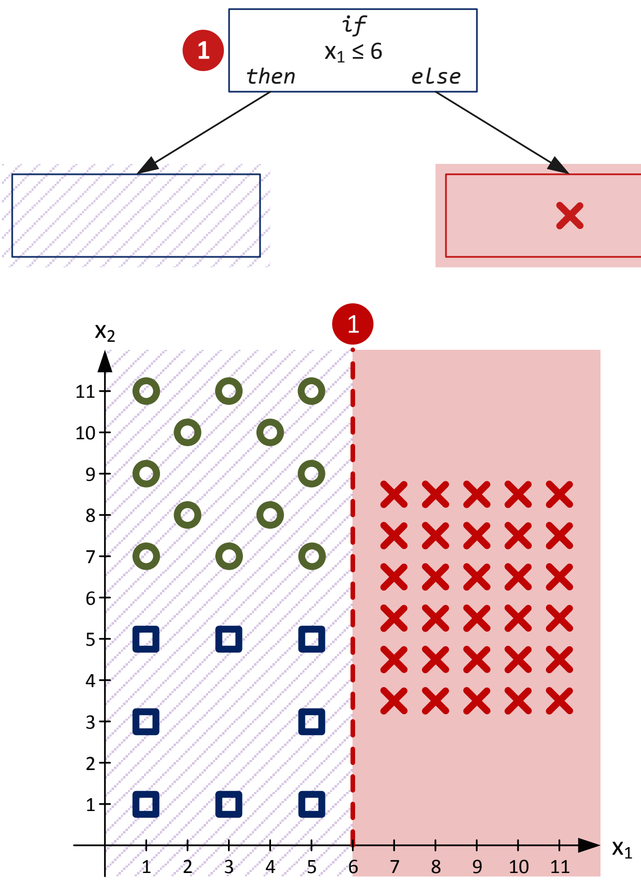
root node				
	red	green	blue	overall
c_i	30	12	8	50 (n)
p_i	$\frac{30}{50} = .6$	$\frac{12}{50} = .24$	$\frac{8}{50} = .16$	1
H	$.6 \times log_2(.6) + .24 \times log_2(.24) + .16 \times log_2(.16)$ $= 1.35933083224$			

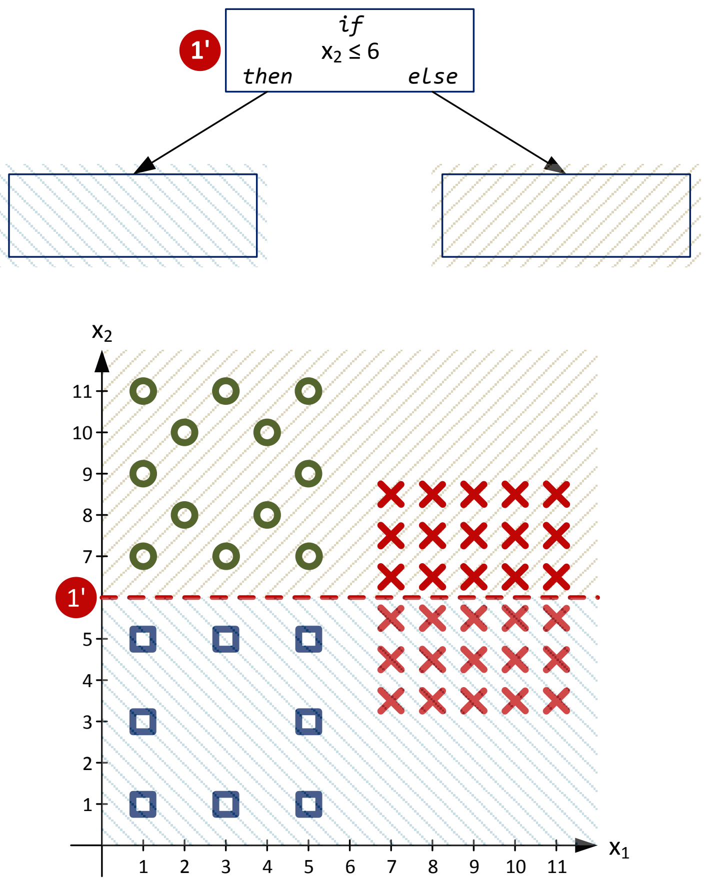
$$n = \sum_{i=1}^{k} c_i$$

$$p_i = \frac{c_i}{n}$$

$$H = -\sum_{i=1}^{k} p_i \cdot log_2(p_i)$$

We have two main alternatives to partition this data set into two



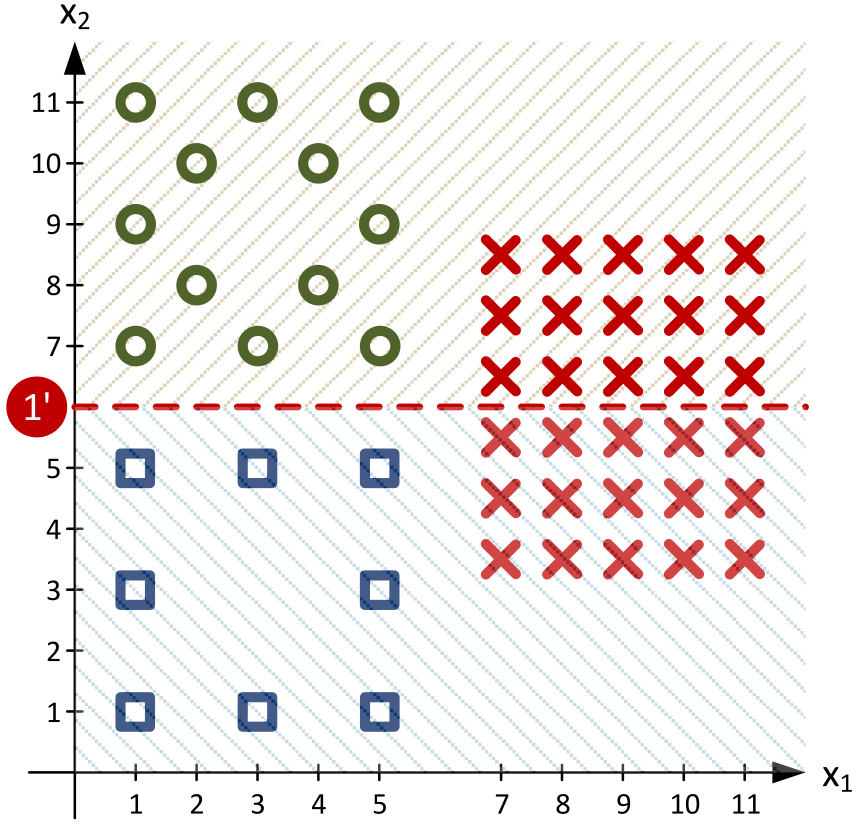


❷ What's the entropy after split 1'? What's the intuition?

left node/lower subspace after split 1'				
	red	green	blue	overall
c_i				
p_i				
H_{left}				

right node/higher subspace after split 1'				
	red	green	blue	overall
c_i				
p_i				
H_{right}				

overall after split 1'	
H_{after}	
IG	

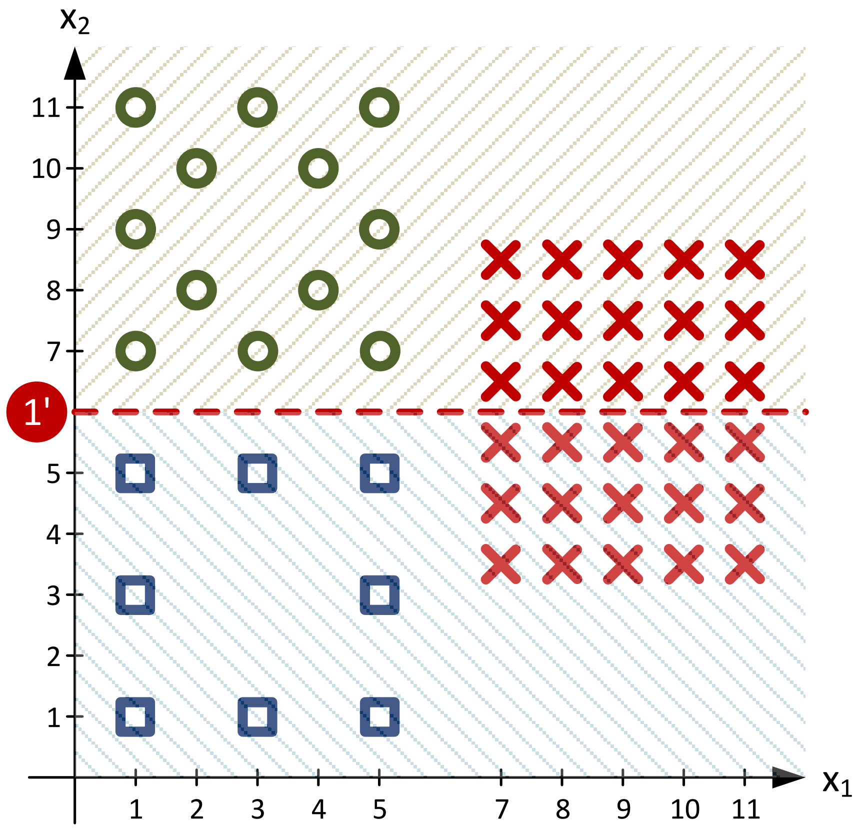


❷ Split 1’ information gain for is “modest” at .395. The intuition is that each subspace has two classes in similar proportions

left node/lower subspace after split 1’				
	red	green	blue	overall
c_i	15	0	8	23
p_i	15/23	0	8/23	1
H_{left}	$15/23 \times log_2(15/23) + 8/23 \times log_2(8/23)$ $= .932111567617$			

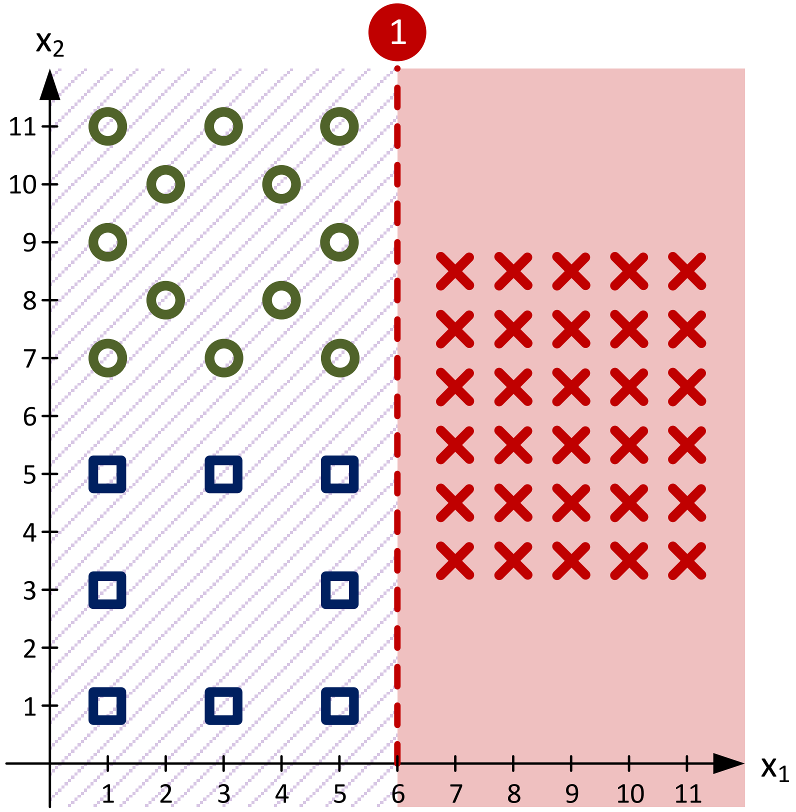
right node/higher subspace after split 1’				
	red	green	blue	overall
c_i	15	12	0	27
p_i	5/9	4/9	0	1
H_{right}	$5/9 \times log_2(5/9) + 4/9 \times log_2(4/9)$ $= .991076059838$			

overall after split 1’	
H_{after}	$23/50 \times .932111567617$ $+27/50 \times .991076059838$ $= .963952393416$
IG	$1.35933083224 - .963952393416$ $= .39537843882$



Activity: ❸ What’s the entropy after split 1? What’s the intuition?

EXERCISE

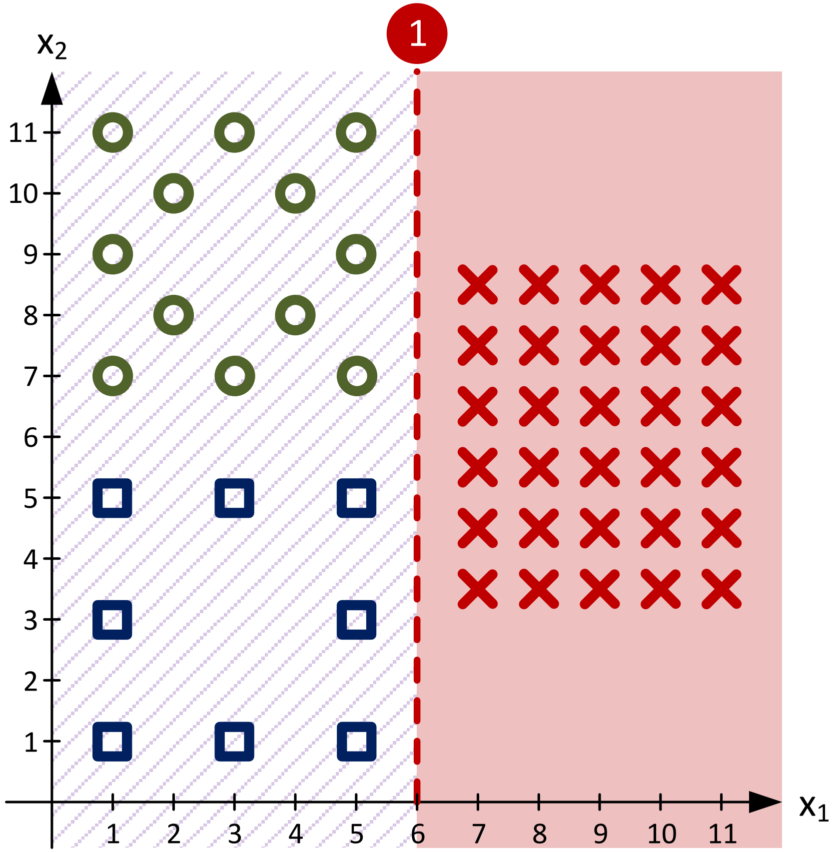


left node after split 1				
	red	green	blue	overall
c_i				
p_i				
H_{left}				

right node after split 1				
	red	green	blue	overall
c_i				
p_i				
H_{right}				

overall after split 1	
H_{after}	
IG	

The information gain for split 1 is "significant" at .971

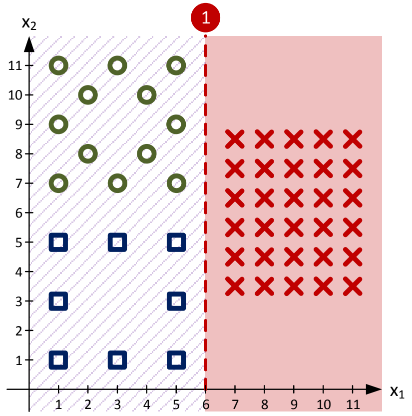


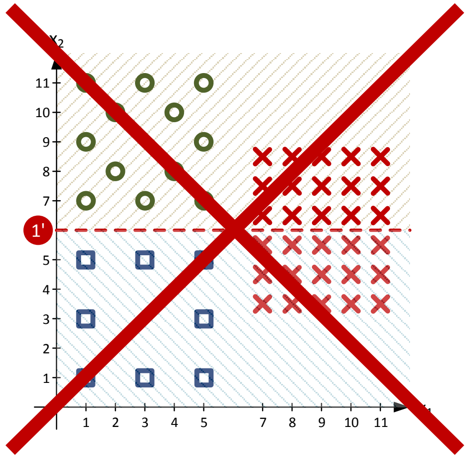
left node after split 1				
	red	green	blue	overall
c_i	0	12	8	20
p_i	0	$\frac{12}{20} = .6$	$\frac{8}{20} = .4$	1
H_{left}	$12/20 \times log_2(12/20) + 8/20 \times log_2(8/20)$ $= .970950594455$			

right node after split 1				
	red	green	blue	overall
c_i	30	0	0	30
p_i	1	0	0	1
H_{right}	$1 \times \log_2(1) = 0$			

overall after split 1	
H_{after}	$20/50 \times .970950594455$ $+30/50 \times 0$ $= .388380237782$
IG	$1.35933083224 - .388380237782$ $= .970950594455$

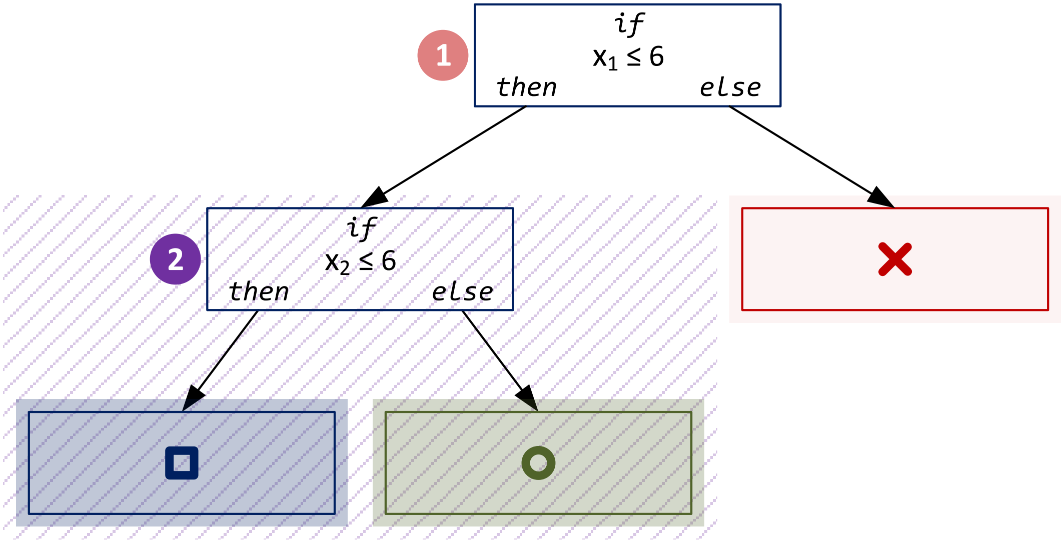
❹ Split 1 (with IG of .971) wins over split 1’ (with IG of .395)

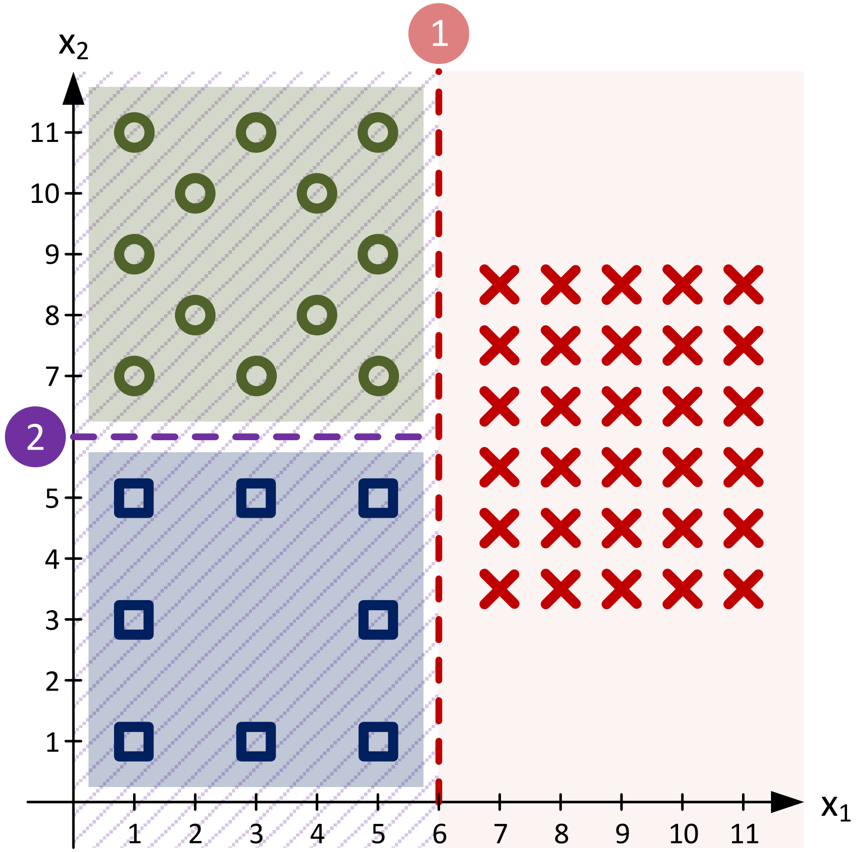




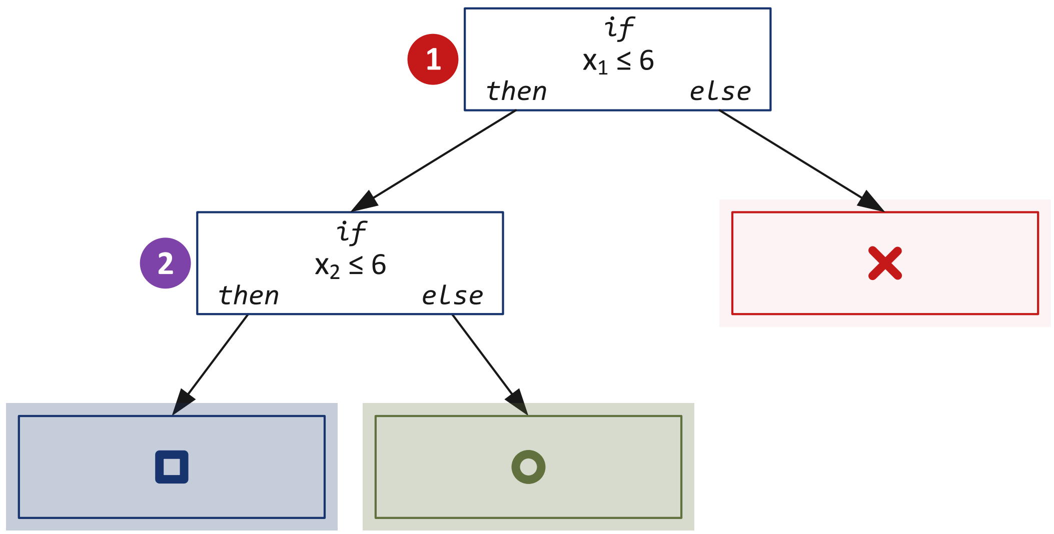
- Intuition: the subspace on the left has two classes in similar proportions but the other one is pure, bringing down significantly the weighted average entropy after the split

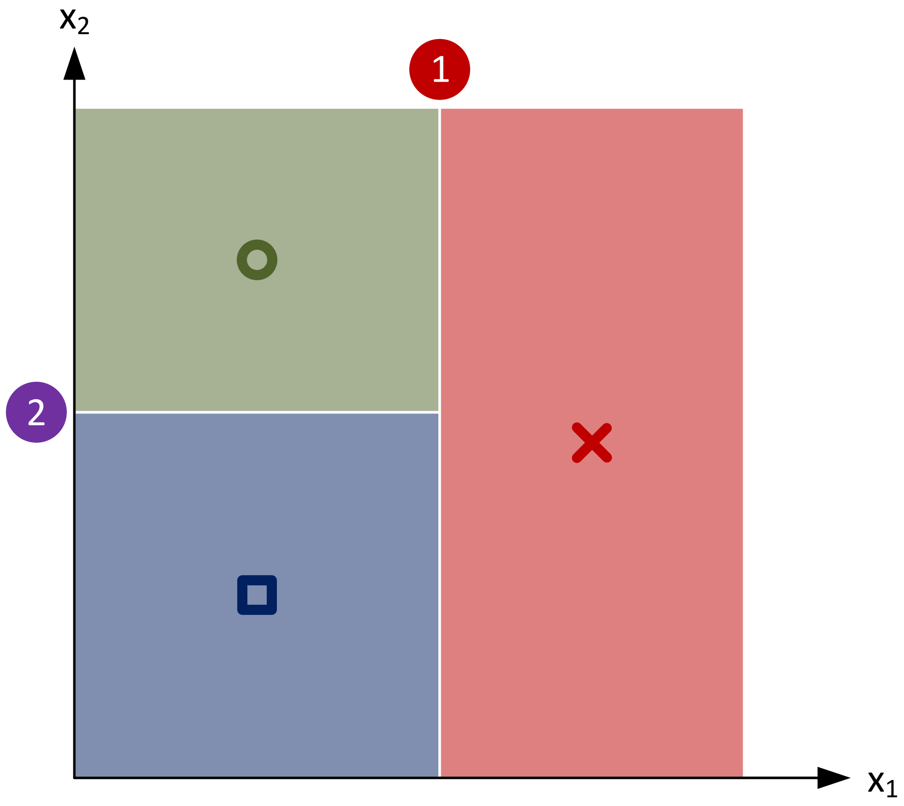
Second cut: $x_2 \leq 6$





The decision tree once training is complete





Most commonly occurring class

- In practice, we don’t expect each terminal region to hold a single class

- Instead, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs

Codealong – Part B
Building the 2008 Democratic Primaries Decision Tree by Hand

Training a Regression Tree
❶ How to choose the split conditions

Commonalities and differences between classification and regression decision trees

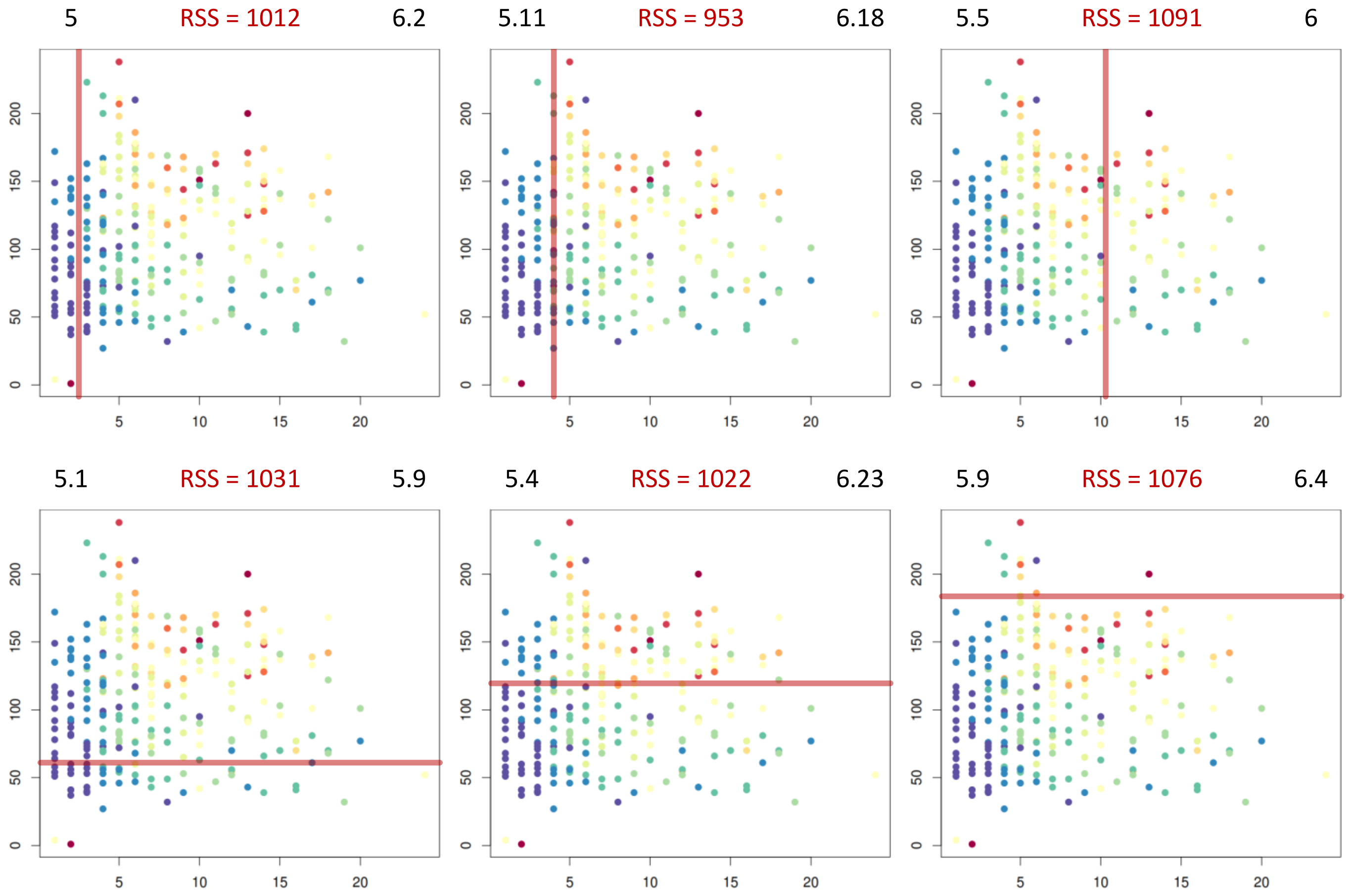
- Just as in the classification setting, we use recursive binary splitting to grow a regression tree

- For every observation that falls into the region R_j, we make the prediction $\hat{y}_{R_j}$, which is the mean of the response values for the training observations in R_j

- In the regression setting, we cannot use entropy for making the binary splits. A natural alternative to H is RSS (residual sum of squares)

$$\sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2$$

We first select the feature and the cutpoint such that splitting the feature space into the regions $\{x \mid feature \leq cutpoint\}$ and $\{x \mid feature > cutpoint\}$ leads to the greatest possible reduction in RSS



Top-down greedy approach (a.k.a., recursive binary splitting)

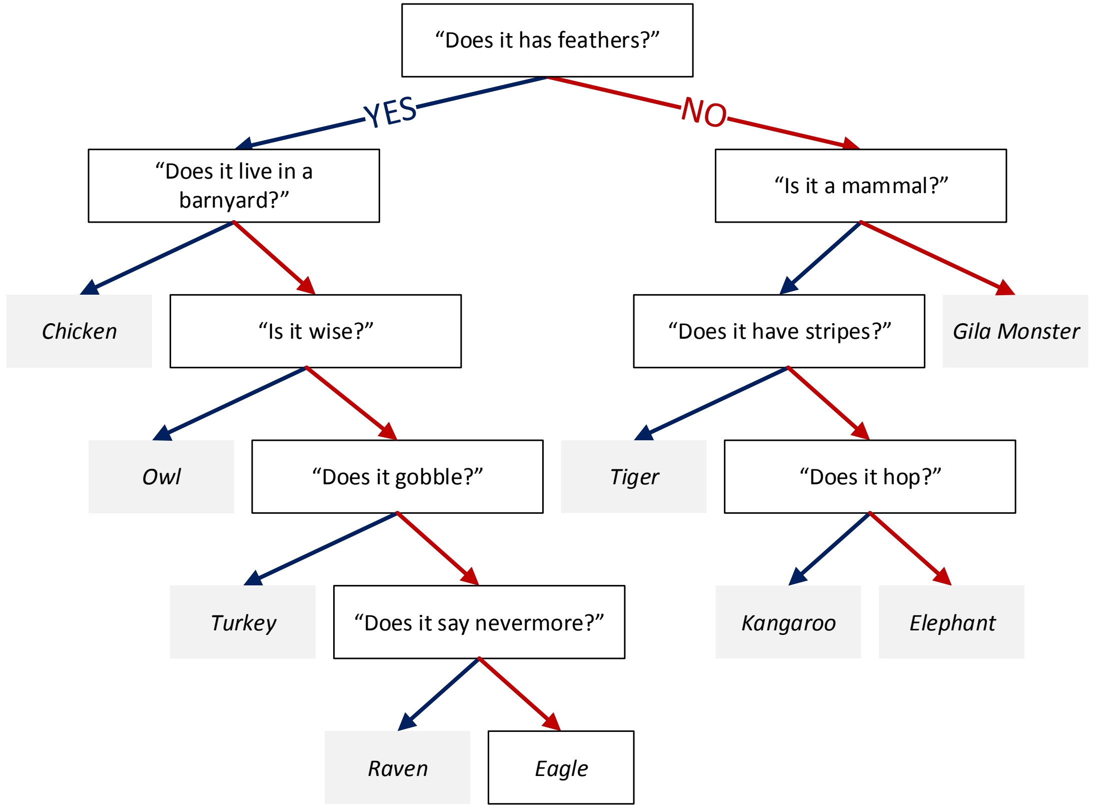
- Once the first cut is made, we recursively repeat the process in the two previously identified regions

- For *regression* trees, we would be looking for the best predictor and the *best cutpoint* in order to split the data further so as to minimize the *RSS* within each of the resulting regions

- For classification trees, we would be looking for the best predictor and the *highest* information gain in order to split the data further so as to minimize the *entropy* within each of the resulting regions

Decision trees are like the game “20 questions” (cont.)

- We want the smallest set of questions to get to the right answer
- Each questions should reduce the search space as much as possible



Training decision trees (cont.)

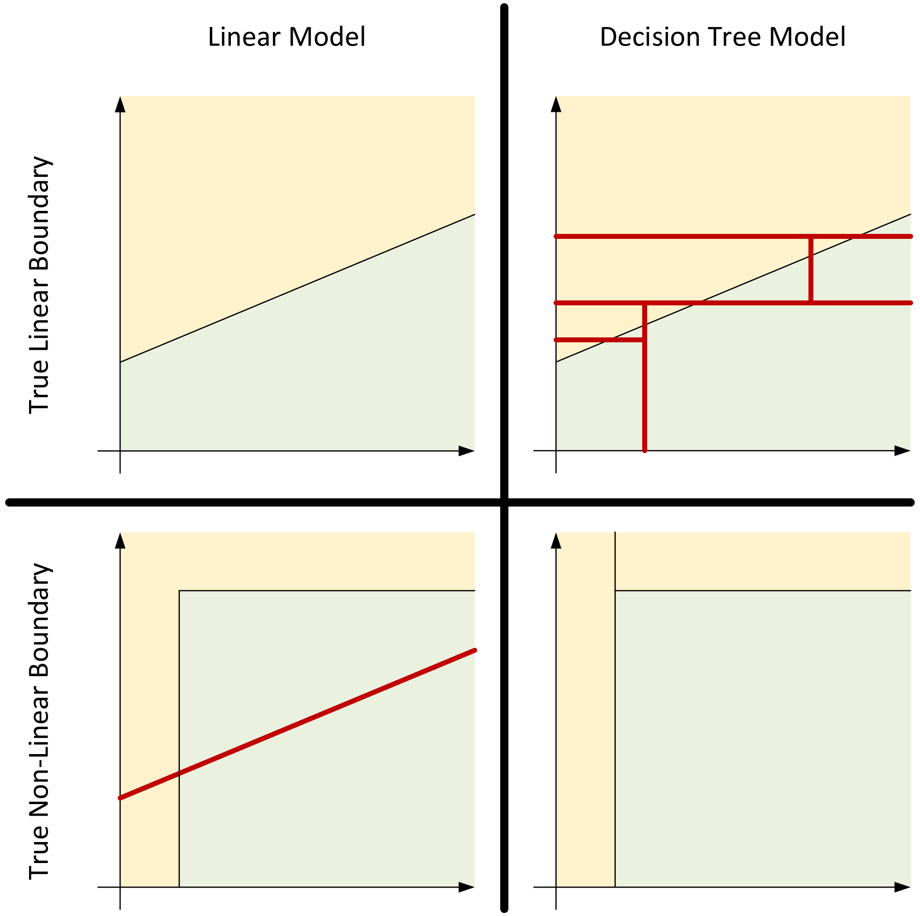
- Training a decision model is deciding the best set of questions to ask

- A good question will be one that best segregates the positive group from the negative group and then narrows in on the correct answer

- Trees automatically contain interaction of features, since each question is dependent on the last

Comparison to previous models

- Decision trees are non-linear, an advantage over linear regression and logistic regression
- A linear model is one in which a change in an input variable has a constant change on the output variable



Activity: Linear or non-linear models?

EXERCISE

ANSWER THE FOLLOWING QUESTIONS (5 minutes)

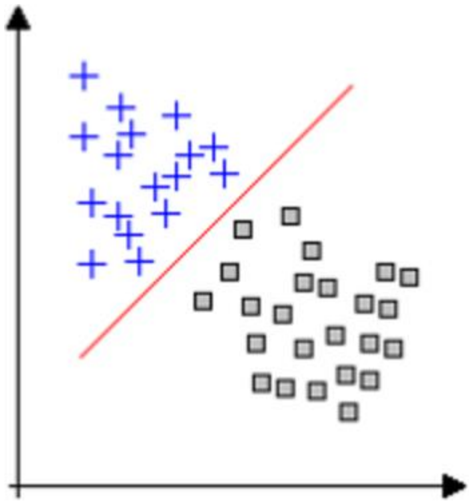
1. Check the models in the handout. Are they linear or non-linear? Why? Why not?
2. When finished, share your answers with your table

DELIVERABLE

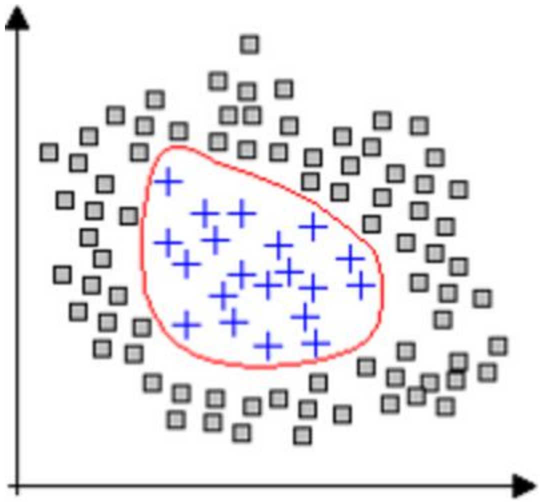
Answers to the above questions

Activity: Linear or non-linear models? (cont.)

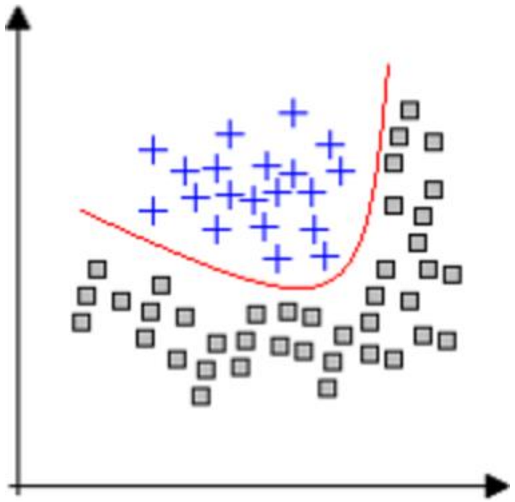
Linear



Non-linear



Linear



Pros and cons of decision trees

☺ Trees are very easy to explain to people. They are even easier to explain than linear regression

☺ Decision trees more closely mirror human decision-making than do the regression and classification methods seen so far

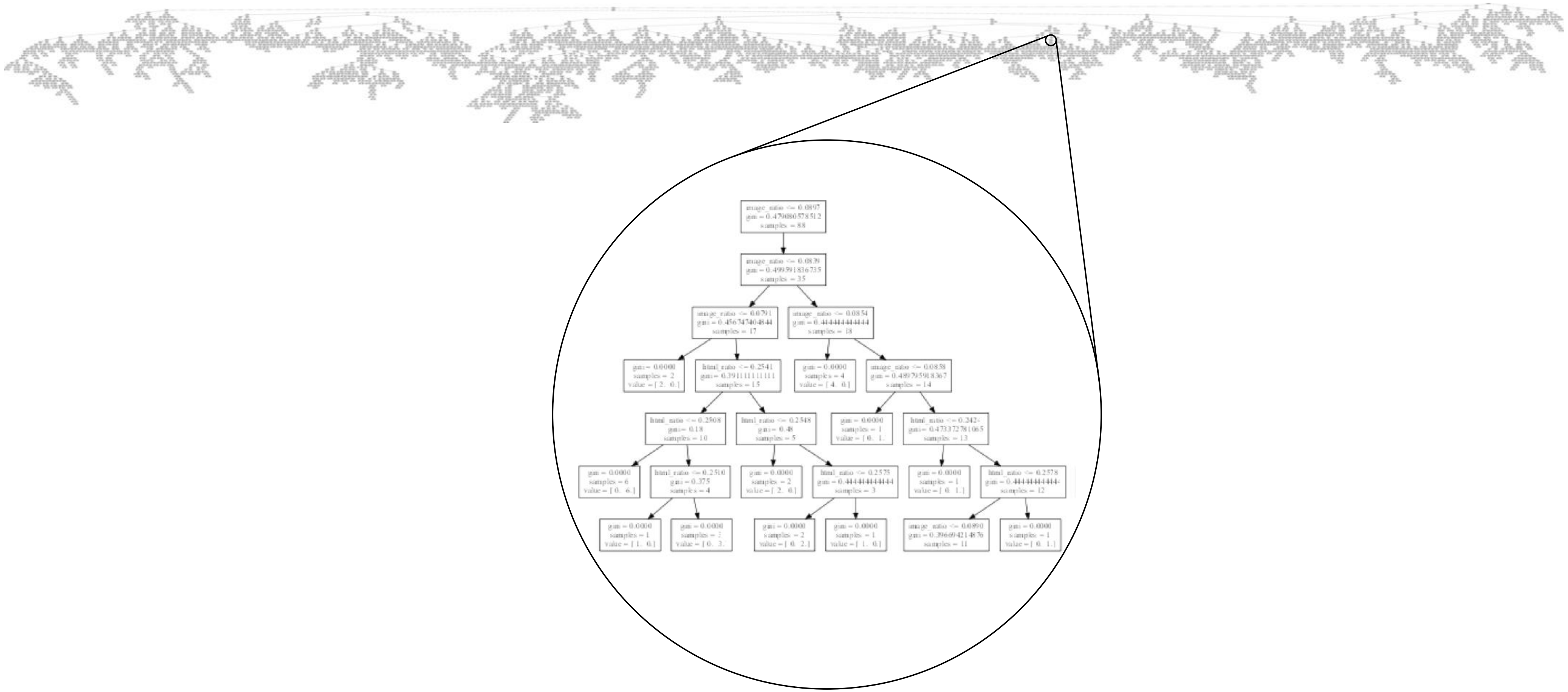
☺ Trees can be displayed graphically and are easily interpreted even by non-experts

☺ Trees can easily handle qualitative predictors without the need to create dummy variables

☹ Trees do not generally have the same level of predictive accuracy as some of the other regression and classification methods seen so far. However, by aggregating many decision trees, the predictive performance of trees can be substantially improved

Overfitting

An unconstrained decision tree can learn an extreme tree (e.g., below)



Overfitting

- Decision trees tend to be weak models because they can easily memorize or overfit to a dataset
  - A model is overfit when it memorizes or bends to a few specific data points rather than picking up general trends in the data

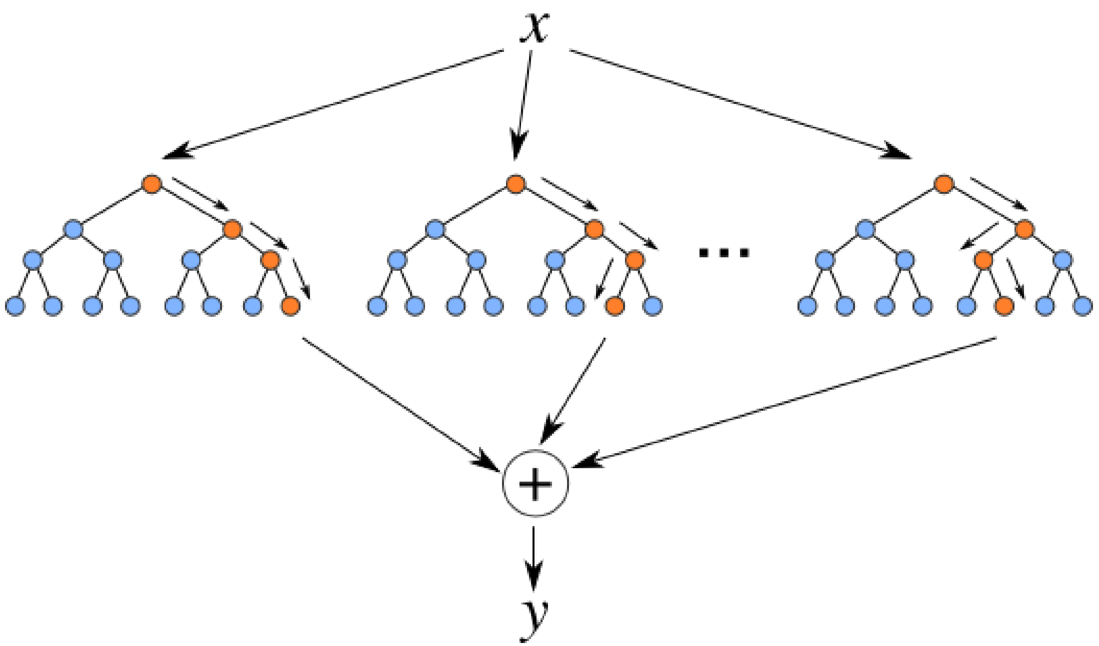
- We can limit our decision trees using a few methods.
  - Limit the number of questions (nodes) a tree can have
  - Limit the number of samples in the leaf nodes

Next: How can we avoid overfitting and increase predictability?

Random Forests

Random forests are an *ensemble* or collection of individual decision trees

- Random forest models are one of the most widespread classifiers used
- They are relatively simple to use and help avoid overfitting



Pros and cons of random forests

Advantages

- Easy to tune
- Built-in protection against overfitting
- Non-linear
- Built-in interaction effects

Disadvantages

- Slow
- No "coefficients"
- Black-box
- Harder to explain

Training a Random Forest

Training a Random Forest

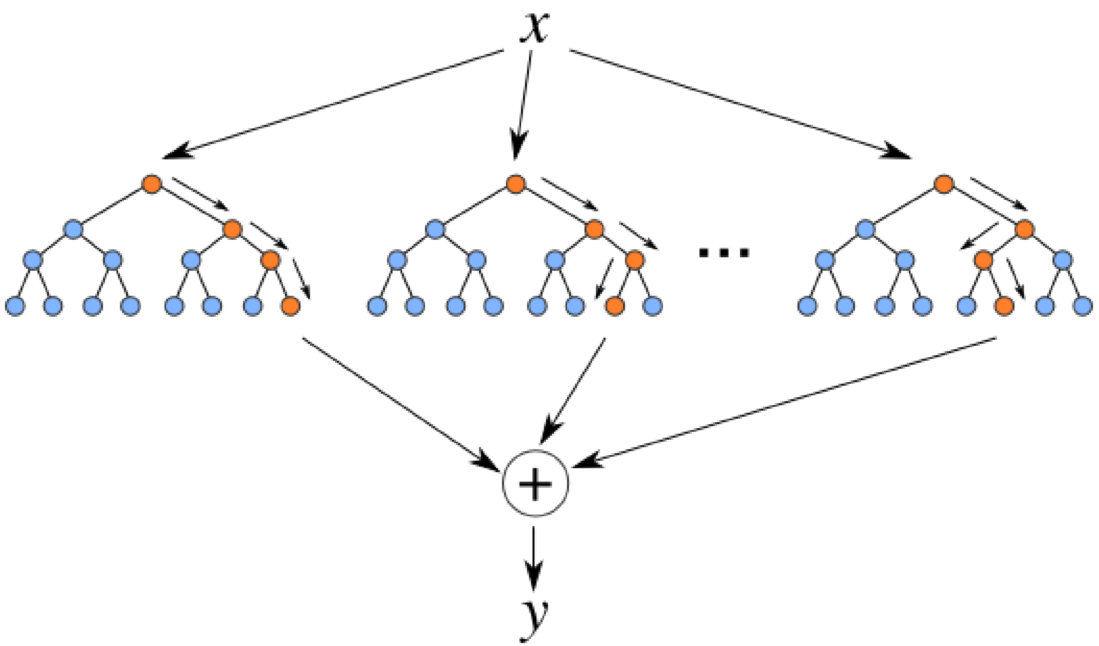
- Training a random forest model involves training many decision tree models
- Since decision trees easily overfit, we use many decision trees together and randomize the way they are created

- Random Forest Training Algorithm
  - Take a bootstrap sample (random sample) of the dataset
  - Train a decision tree on the bootstrap sample
  - For each split/feature selection, only evaluate a *limited* number of features to find the best one
  - Repeat this for a number of trees

Predicting using a Random Forest

Predicting using a Random Forest

- Predictions for a random forest model come from each decision tree
- Make an individual prediction with each decision tree
- Combine the individual predictions and take the majority vote



Lab
Decision Trees and Random Forests with *scikit-learn*

Review

Review

- What are decision trees?
- What does training involve?
- What are some common problems with decision trees?
- What are random forests?
- What are some common problems with random forests?

Q & A

Pre-Work

Pre-Work

Before the next lesson, you should already be able to:

- Experience with *scikit-learn* classifiers, specifically random forests and decision trees
- Install the Python package spacy with `pip install spacy`
- Run the spacy download data command with `python -m spacy.en.download --force all`

Exit Ticket

Don't forget to fill out your exit ticket here

Sources

- "The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition"