# Machine Learning: Decision Trees and Information, Evaluating ML Models

(Ch. 18.1–18.3)

1

## Bookkeeping

- Midterm—see next slide
- HW3 **now due 10/25—please see schedule**
- Today
  - Back to ML 2—more about decision trees; all about information gain
  - Measuring model quality
- Next time
  - Knowledge-based agents
  - Propositional logics

2

# Midterm

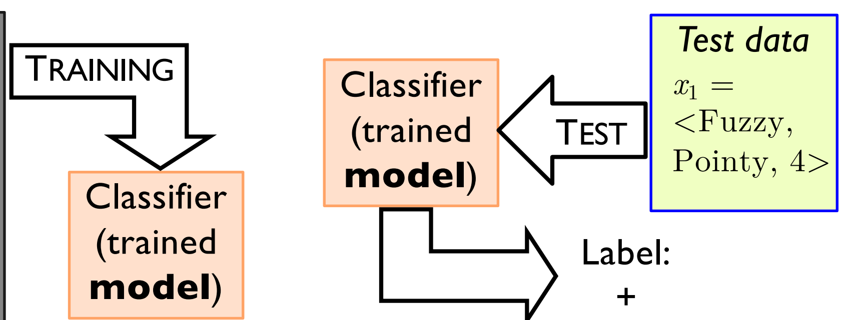
- Returned at end of class today
- **Reminder: take time to try to work out the correct answers**
  - 24 hours after return until we'll answer questions
- Next class we'll take time to go over some sticking points
- Average was about 50; maximum was 88
- **Approximate** grade cutoffs: **A = 55+**; **B = 30-54**
- 20% of total grade

3

# Inductive Learning Pipeline

*Training data, $X$*

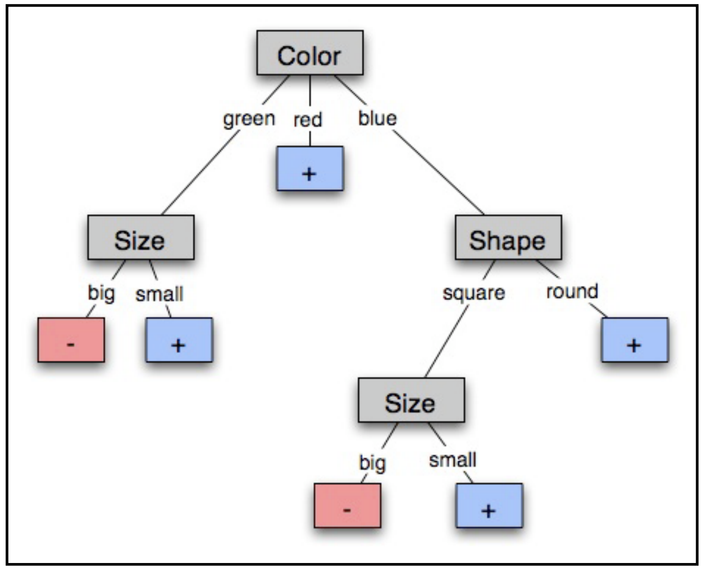
| *Text-ure* | *Ears* | *Legs* | *Class* |
|---|---|---|---|
| Fuzzy | Round | 4 | + |
| Slimy | Missing | 4 | - |
| Fuzzy | Pointy | 4 | - |
| Fuzzy | Round | 4 | + |
| Fuzzy | Pointy | 4 | + |
| … | | | |




4

# Learning Decision Trees

- Each **non-leaf** node is an attribute (feature)
- Each **arc** is one value of the attribute at the node it comes from
- Each **leaf** node is a classification (+ or -)



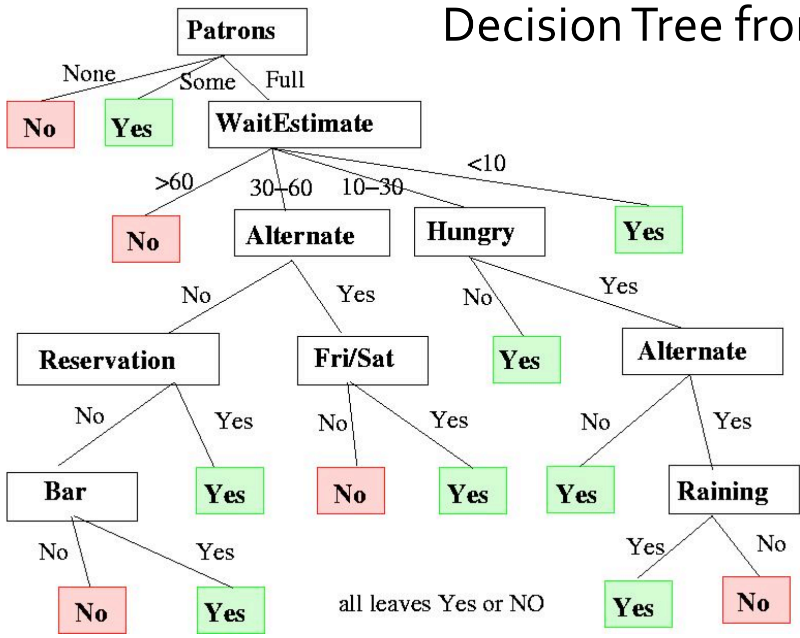

5

# A Training Set

| Datum | Attributes | | | | | | | | | | Outcome (Label) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | altern-atives | bar | Friday | hungry | people | $ | rain | reser-vation | type | wait time | Wait? |
| $X_1$ | Yes | No | No | Yes | Some | $$$ | No | Yes | French | 0-10 | Yes |
| $X_2$ | Yes | No | No | Yes | Full | $ | No | No | Thai | 30-60 | No |
| $X_3$ | No | Yes | No | No | Some | $ | No | No | Burger | 0-10 | Yes |
| $X_4$ | Yes | No | Yes | Yes | Full | $ | Yes | No | Thai | 10-30 | Yes |
| $X_5$ | Yes | No | Yes | No | Full | $$$ | No | Yes | French | >60 | No |
| $X_6$ | No | Yes | No | Yes | Some | $$ | Yes | Yes | Italian | 0-10 | Yes |
| $X_7$ | No | Yes | No | No | None | $ | Yes | No | Burger | 0-10 | No |
| $X_8$ | No | No | No | Yes | Some | $$ | Yes | Yes | Thai | 0-10 | Yes |
| $X_9$ | No | Yes | Yes | No | Full | $ | Yes | No | Burger | >60 | No |
| $X_{10}$ | Yes | Yes | Yes | Yes | Full | $$$ | No | Yes | Italian | 0-30 | No |
| $X_{11}$ | No | No | No | No | None | $ | No | No | Thai | 0-10 | No |
| $X_{12}$ | Yes | Yes | Yes | Yes | Full | $ | No | No | Burger | 30-60 | Yes |

6

## Decision Tree from Inspection



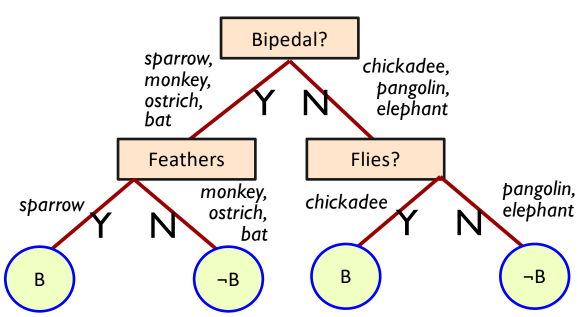

*Problem from R&N, table from Dr. Manfred Kerber @ Birmingham, with thanks – www.cs.bham.ac.uk/~mmk/Teaching/AI/l3.html*

7

## Bird or Not-Bird?

1.
2.
3.
4.
5.

**But… we should have split on feathers first**




| Examples (training data) | Attributes | | | Outcome |
|---|---|---|---|---|
| | Bipedal | Flies | Feathers | |
| Sparrow | Y | Y | Y | B |
| Monkey | Y | N | N | ¬B |
| Ostrich | Y | N | Y | B |
| Pangolin | N | N | N | ¬B |
| Bat | Y | Y | N | ¬B |
| Elephant | N | N | N | ¬B |
| Chickadee | N | Y | Y | B |

<u>Test</u>

mouse: <B:N, Fl:N, Fe:N>

8

# ID3/C4.5

- A **greedy** algorithm for decision tree construction
  - Ross Quinlan, 1987
- Construct decision tree top-down by recursively selecting the "best attribute" to use at current node
  - Select best attribute for current node → how?
  - Generate child nodes (one for each possible value of attribute)
  - Partition training data using attribute values
  - Assign subsets of examples to the appropriate child node
  - Repeat for each child node until all examples associated with a node are either all positive or all negative

9

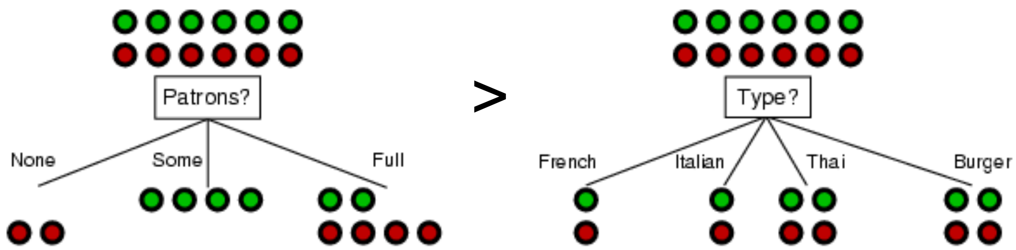
# Choosing the Best Attribute

- **Key problem**: what attribute to split on?
- Some possibilities are:
  - Random: Select any attribute at random
  - Least-Values: Choose attribute with smallest number of values
  - Most-Values: Choose attribute with largest number of values
  - **Max-Gain**: Choose attribute that has the largest expected **information gain**—the attribute that will result in the smallest expected size of the subtrees rooted at its children
- ID3 uses Max-Gain to select the best attribute

10

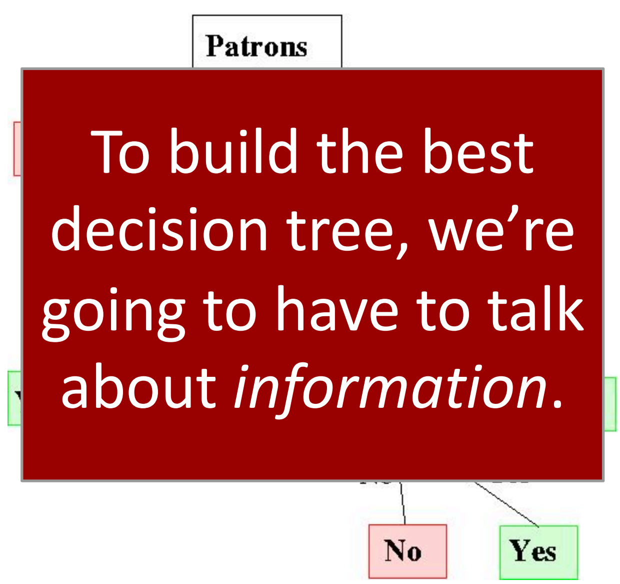
## Choosing an Attribute

- Core idea: a good attribute splits the examples into subsets that are (ideally) “all positive” or “all negative” – that is, we want *pure* groups




11

## ID3-induced Decision Tree




To build the best decision tree, we’re going to have to talk about *information*.

12

## Information Theory 101

- **Information**: the **minimum number of bits** needed to store or send some information
  - Wikipedia: "The measure of data, known as information entropy, is usually expressed by the average number of bits needed for storage or communication"
- Intuition: minimize effort to communicate/store
  - Common words (a, the, dog) are shorter than less common ones (parliamentarian, foreshadowing)
  - In Morse code, common (probable) letters have shorter encodings

*"A Mathematical Theory of Communication," Bell System Technical Journal, 1948, Claude E. Shannon, Bell Labs*

13

## Information Theory 102

- Information is measured in **bits.**
- Information in a message depends on its probability.
- Given *n* equally probable possible messages, what is probability $p_n$ of each one?

  $1/n$

- Information conveyed by a message is:

  $\log_2(n) = -\log_2(p_n)$

- Example: with 16 possible messages, $\log_2(16) = 4$, and we need 4 bits to identify/send each message

14

# Information Theory 102.b

- Information conveyed by a message is $\log_2(n) = -\log_2(p)$
- Given a probability distribution for n messages:

  $P = (p_1, p_2 \ldots p_n)$

- The information conveyed by that distribution is:

  $I(P) = -(p_1 * \log_2(p_1) + p_2 * \log_2(p_2) + .. + p_n * \log_2(p_n))$

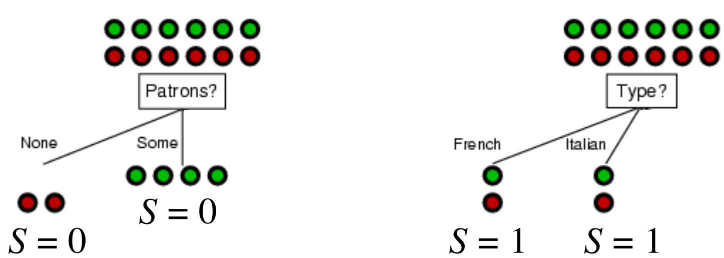
- This is the **entropy** of P.

n = messages
$p_n$ = probability of n occurring

15

# Entropy Interlude

- Entropy (*S*): the homogeneity (purity) of a sample
  - If everything is the same, *S* = 0
  - If differences are even *S* = 1




16

## Information Theory 103

- Entropy: **average** number of bits (per message) needed to represent a stream of messages

  $I(P) = -(p_1*\log_2(p_1) + p_2*\log_2(p_2) + .. + p_n*\log_2(p_n))$

- Examples:
  - $P = (0.5, 0.5)$ : $I(P) = 1$ → entropy of a fair coin flip
  - $P = (0.67, 0.33)$ : $I(P) = 0.92$
  - $P = (0.99, 0.01)$ : $I(P) = 0.08$
  - $P = (1, 0)$ : $I(P) = 0$
- **As the distribution becomes *more skewed*, the amount of information *decreases.* Why?**
- **Because I can just predict the most likely element, and usually be right**

17

## Entropy as Measure of **Homogeneity of Examples**

- Entropy can be used to characterize the (im)purity of an arbitrary collection of examples
- **Low entropy** implies **high homogeneity**
  - Given a collection *S* (like the table of 12 examples for the restaurant domain), containing positive and negative examples of some target concept, the entropy of *S* relative to its Boolean classification is:

$$I(S) = -(p_+*\log_2(p_+) + p_-*\log_2(p_-))$$

Entropy([6+, 6-]) = 1

Entropy([9+, 5-]) = 0.940

18

## Information Gain

- **Information gain: how much entropy decreases (homogeneity increases) when a dataset is split on an attribute.**
  - High homogeneity → high likelihood samples will have the same class
- Information Gain is the expected reduction in entropy of target variable Y for data sample S
- Constructing a decision tree is all about finding the attribute that returns the highest information gain (i.e., the most homogeneous branches)

19

## Information Gain, cont.

- Use to rank attributes and build decision tree!
- **Choose nodes using attribute with greatest info gain**
  - Meaning least information remaining after split
  - I.e., subsets are all **as skewed as possible**
- Why?
  - Create small decision trees: predictions can be made with few attribute tests
  - Try to find a minimal process that still captures the data (Occam's Razor)

20

## Information Theory 103b

- Entropy over a dataset
- Consider a dataset with 1 blue, 2 greens, and 3 reds: ●●●●●●
- I(●●●●●●) $= -\Sigma_i (p_i \log_2(p_i))$

$= -(p_b \log_2(p_b) + (p_g \log_2(p_g)) + (p_r \log_2(p_r))$

$= -(⅙ \log_2(⅙) + (⅓ \log_2(⅓)) + (½ \log_2(½))$

$= 1.46$

Entropy is between 0 and 1 only in binary cases—with > than 2 outcomes you can need >1 bit of information!

21

## Information Gain: Using Information

- A chosen attribute A divides the training set S into subsets $S_1, \ldots, S_v$ according to their values for A, where A has *v* distinct values.
- The information gain IG(S,A) (or just IG(S)) of an attribute A relative to a collection of examples S is defined as:

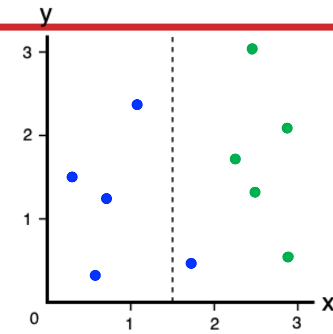
$$IG(S, A) = I(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \times I(S_v)$$

- This is the gain in information due to attribute A
  - Expected reduction in entropy (≡ increase in homogeneity)
- This represents the difference between
  - I(S)—the entropy of the original collection S
  - Remainder(A)—expected value of the entropy after S is partitioned using attribute A

22

# Information Gain: Example

- First we calculate the entropy *before* the split, $I(S)$
  - I(●●●●●●●●●●) = **1** (perfectly balanced)
- Split, then calculate the entropy of each branch
  - $I_{left}$(●●●●) = **0** (pure)
  - $I_{right}$(●●●●●●) = – (⅙ $\log_2$(⅙) + ⅚ $\log_2$(⅚)) = **0.65**




- Then we calculate the entropy of the split by weighting each branch's entropy by how many data points that branch covers
  - *Left* has 4 data points: 4/10 of the data, 0.4. *Right* has 0.6 of the data.
  - $I_{split} = (0.4*0) + (0.6*0.65) = \mathbf{0.39}$
- Information gain = **1 – 0.39 = 0.61**

$$IG(S, A) = I(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \times I(S_v)$$

example from victorzhou.com/blog/information-gain/

23

# ID3/C4.5

- A **greedy** algorithm for decision tree construction
  - Ross Quinlan, 1987
- Construct decision tree top-down by recursively selecting the "best attribute" to use at current node
  1. Select best attribute for current node → Using best information gain
  2. Generate child nodes (one for each possible value of attribute)
  3. Partition training data using attribute values
  4. Assign subsets of examples to the appropriate child node
  5. Repeat for each child node until all examples associated with a node are either all positive or all negative

24

## Extensions of the Decision Tree Learning Algorithm

- Real-valued data
- Noisy data and overfitting
- Generation of rules
- Pruning decision trees
- Cross-validation for experimental validation of performance
- C4.5 is a (more applicable) extension of ID3 that accounts for real-world problems: unavailable values, continuous attributes, pruning decision trees, rule derivation, …

25

## Extensions: Real-Valued Data

- Select thresholds defining intervals so each becomes a discrete value of attribute
- Use heuristics, e.g. always divide into quartiles
- Use domain knowledge, e.g. divide age into infant (0-2), toddler (3-5), school-aged (5-8)
- Or treat this as another learning problem
  - Try different ways to discretize continuous variable; see which yield better results w.r.t. some metric
  - E.g., try midpoint between every pair of values

31

# Converting Decision Trees to Rules

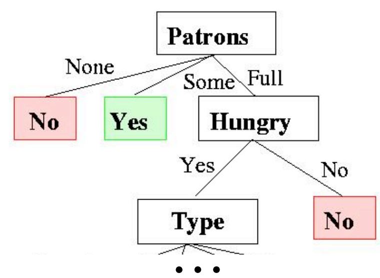
- 1 rule for each path in tree (from root to a leaf)
- Left-hand side: labels of nodes and arcs

  Patrons=None → Don't wait

  Patrons=Some → Wait

  Patrons=Full ∧ Hungry=No → Don't wait

  etc…

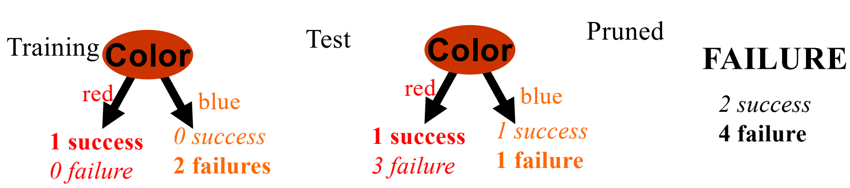



- Resulting rules can be simplified and reasoned over

32

# Pruning Decision Trees

- Replace a whole subtree by a leaf node
- If: a decision rule establishes that he expected error rate in the subtree is greater than in the single leaf. E.g.,
  - Training: one training red success and two training blue failures
  - Test: three red failures and one blue success
  - Consider replacing this subtree by a single Failure node. (leaf)
- After replacement we will have only two errors instead of five:




33

## Summary: Decision Tree Learning

- A widely used learning methods in practice
- Can out-perform human experts in many problems
  - Strengths:
    - Fast
    - Simple to implement
    - Can convert to a set of easily interpretable rules
    - Empirically valid in many commercial products
    - Handles noisy data
  - Weaknesses:
    - Univariate splits/Partitioning using only one attribute at a time (limits types of possible trees)
    - Large trees hard to understand
    - Requires fixed-length feature vectors
    - Non-incremental (i.e., batch method)

34

## How Well Does it Work?

- At least as accurate as human experts (sometimes)
  - Diagnosing breast cancer: humans correct 65% of the time; decision tree classified 72% correct
  - BP designed a decision tree for gas-oil separation for offshore oil platforms; replaced an earlier rule-based expert system
  - Cessna designed an airplane flight controller using 90,000 examples and 20 attributes per example
  - SKICAT (Sky Image Cataloging and Analysis Tool) used a DT to classify sky objects **an order of magnitude** fainter than was previously possible, with an accuracy of over 90%.

35

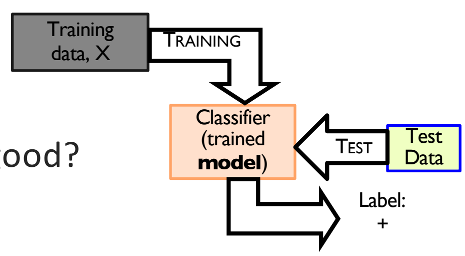
# Measuring Model Quality

- So we went through a bunch of training data and made a decision tree (or any other ML model).
- Is that model any good?

36

## ML: Measuring Model Quality

- So we have training data, and we have learned a model
  - A learned decision tree is one such model
- We have some set of test data we have held out
- How do we evaluate whether the model is good?
- How can this process fail?




37

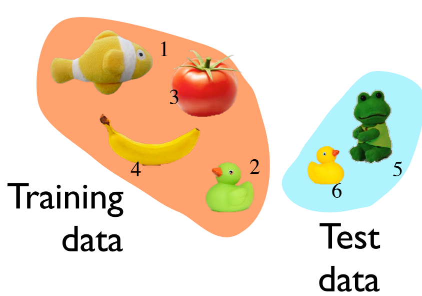
# Measuring Model Quality

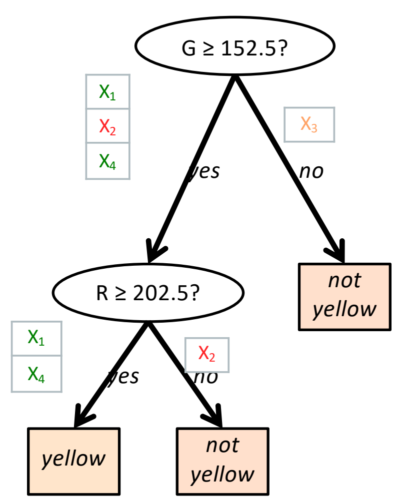
- **How good is a model?**
- Predictive accuracy
- False positives / false negatives for a given cutoff threshold
  - Loss function (accounts for cost of different types of errors)
- Area under the curve
- Minimizing loss can lead to problems with overfitting

38

# One Possible Decision Tree

| *sample* | *attributes* | | | | *label* |
|---|---|---|---|---|---|
| | *R* | *G* | *B* | *Fuzzy?* | *Yellow?* |
| $X_1$ | 205 | 200 | 40 | Y | yes |
| $X_2$ | 90 | 250 | 90 | N | no |
| $X_3$ | 220 | 10 | 22 | N | no |
| $X_4$ | 205 | 210 | 10 | N | yes |




39

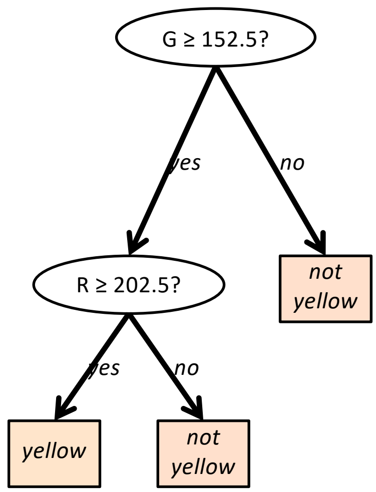
# One Possible Decision Tree

- Predictions

7

8

| | *R* | *G* | *B* | *Fuzzy?* | *Prediction: Is it yellow?* |
|---|---|---|---|---|---|
| $X_7$ | 215 | 45 | 190 | N | |




So what went wrong?

40

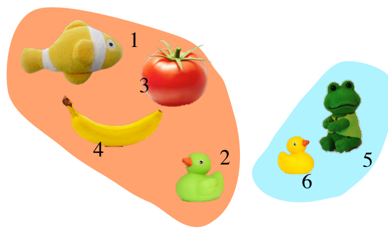
# Measuring Model Quality

- Training error
  - Train on all data; measure error on all data
  - Subject to overfitting (of course we'll make good predictions on the data on which we trained!)
- Regularization
  - Attempt to avoid overfitting
  - Explicitly minimize the complexity of the function while minimizing loss
  - Tradeoff is modeled with a regularization parameter

41

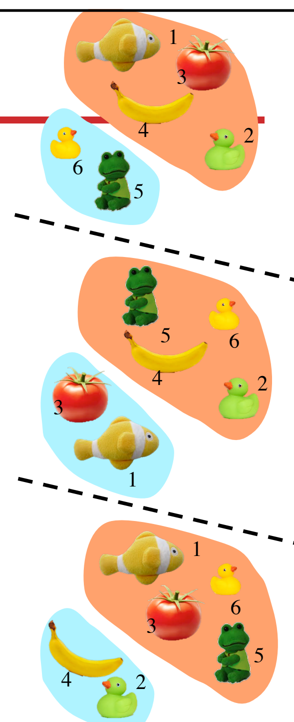
# Cross-Validation

- Holdout cross-validation:
  - Divide data into training set and test set
  - Train on training set; measure error on test set
  - Better than training error, since we are measuring generalization to new data
  - To get a good estimate, we need a reasonably large test set
  - But this gives less data to train on, reducing our model quality!




42

# Cross-Validation, cont.

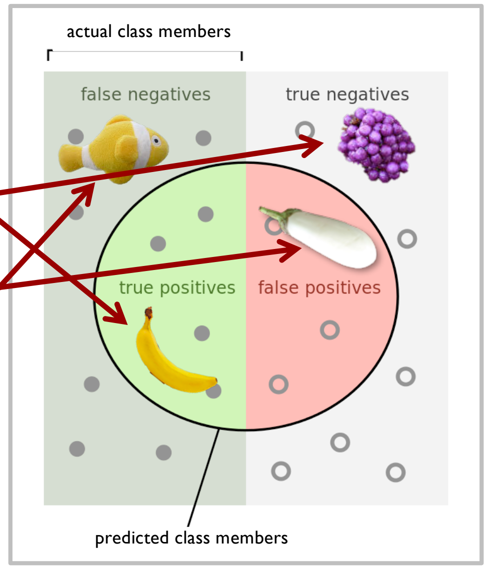



- *k*-fold cross-validation:
  - Divide data into *k* folds
  - Train on *k*-1 folds, use the $k^{th}$ fold to measure error
  - Repeat *k* times; use average error to measure generalization accuracy
  - Statistically valid and gives good accuracy estimates
  - 5 and 10 are common values for *k*
- Leave-one-out cross-validation (LOOCV)
  - *k*-fold cross validation where *k*=N (test data = 1 instance!)
  - Quite accurate, but also quite expensive, since it requires building N models

43

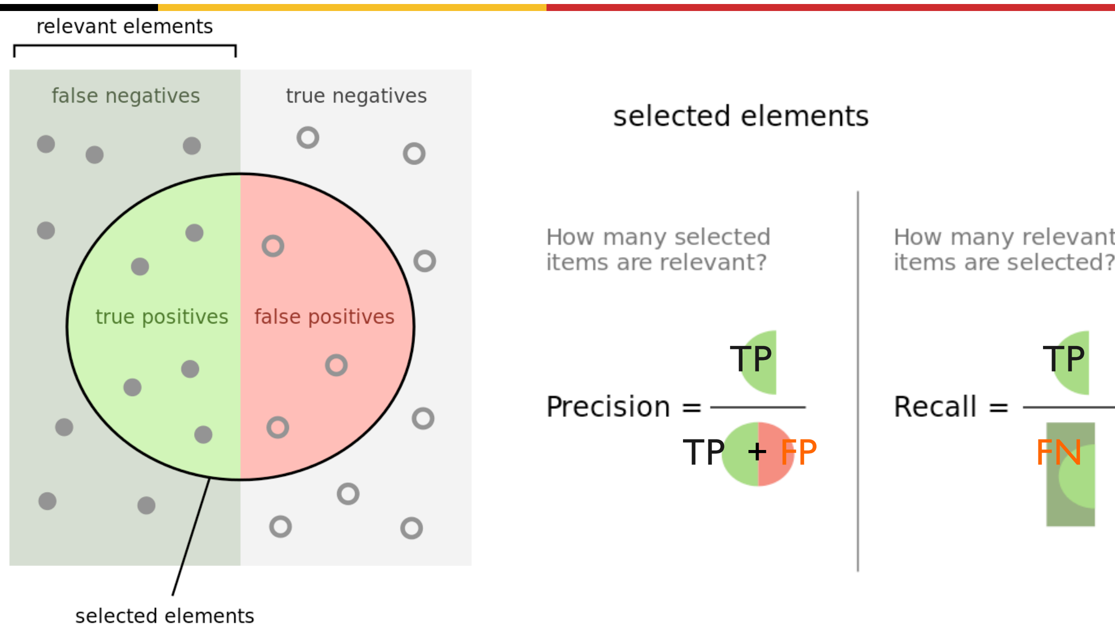
# Correctness

- True positive
  - I predict it's yellow, and it is yellow
- True negative
  - I predict it's not yellow, and it's not
- False positive
  - I predict it's yellow, but it's not
- False negative
  - I predict it's not yellow, but it is




44

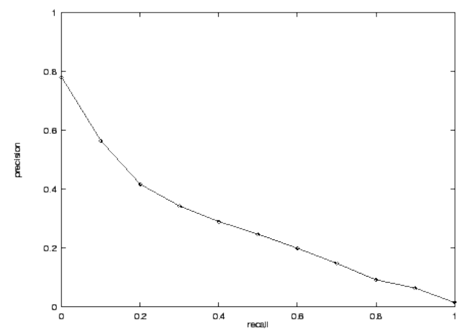
# Precision/Recall




45

## Precision, or Recall?

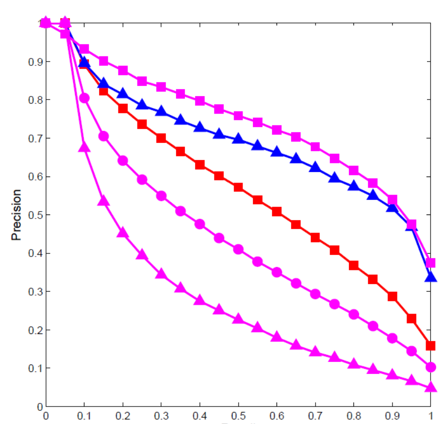
- Precision (specificity) and recall (sensitivity) are in tension
- In general, increasing one causes the other to decrease
  - The more *precise* you are, the more things you will miss
  - The more you guarantee you will catch everything, the more you will return some incorrect things (casting a wide net)
- So… which is better?
  - Recall our cancer example
- Studying the precision/recall curve is informative




46

## Precision and Recall

- If one system's curve is always above the other, it's strictly better




47

## F measure

- The F1 measure combines both into a useful single metric

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$= \frac{\text{TP}}{\text{TP} + 1/2\ (\text{FP} + \text{FN})}$$

- Idea: both precision and recall need to be reasonably good
- Heavily penalizes small precision or small recall

48

## Confusion Matrix (1)

- A confusion matrix can be a better way to show results
- For binary classifiers it's simple and is related to type I *and* type II errors (i.e., false positives and false negatives)
- There may be different costs for each kind of error
- So we need to understand their frequencies

predicted

| actual | | C | ¬C |
|---|---|---|---|
| | C | True positive | False negative |
| | ¬C | False positive | True negative |

49

# Confusion Matrix (2)

- For multi-way classifiers, a confusion matrix is even more useful
- It lets you focus in on where the errors are

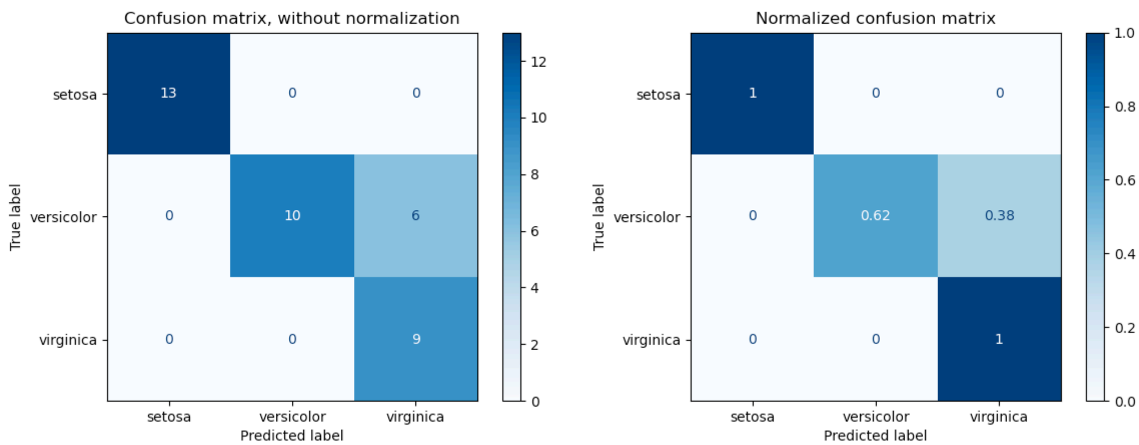
predicted

| actual | Cat | Dog | rabbit |
|---|---|---|---|
| Cat | 5 | 3 | 0 |
| Dog | 2 | 3 | 1 |
| Rabbit | 0 | 2 | 11 |

50

# Confusion Matrix (2)

- For multi-way classifiers, a confusion matrix is even more useful
- It lets you focus in on where the errors are




Figures: scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html

51

# Overfitting

- Sometimes, model fits training data well but doesn't do well on test data
- Can be it "overfit" to the training data
  - Model is too specific to training data
  - Doesn't generalize to new information well
- Learned model: (Y∧Y∧Y→B ∨ Y∧N∧N→ ¬B ∨ ...)

| Examples (training data) | Attributes | | | Outcome |
|---|---|---|---|---|
| | Bipedal | Flies | Feathers | |
| Sparrow | Y | Y | Y | B |
| Monkey | Y | N | N | ¬B |
| Ostrich | Y | N | Y | B |
| Bat | Y | Y | N | ¬B |
| Elephant | N | N | N | ¬B |

52

# Overfitting 2

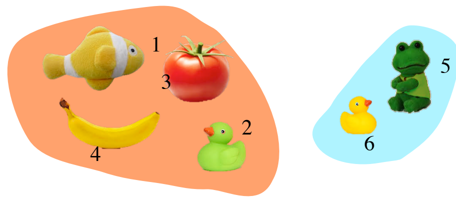
- Irrelevant attributes can also lead to overfitting
- If hypothesis space has many dimensions (many attributes), may find meaningless regularity
  - Ex: Name starts with [A-M] → ¬Bird

| Examples (training data) | Attributes | | | Outcome |
|---|---|---|---|---|
| | Bipedal | Flies | Feathers | |
| Sparrow | Y | Y | Y | B |
| Monkey | Y | N | N | ¬B |
| Ostrich | Y | N | Y | B |
| Bat | Y | Y | N | ¬B |
| Elephant | N | N | N | ¬B |

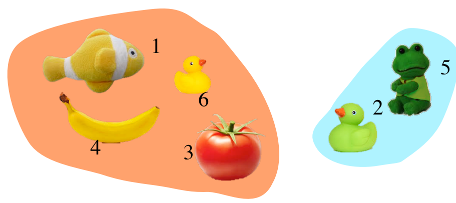
53

# Overfitting 3

- Incomplete training data → overfitting

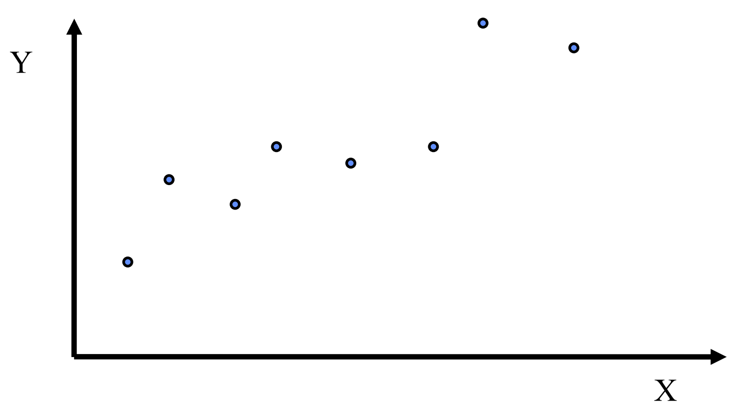



- Bad training/test split → overfitting




54

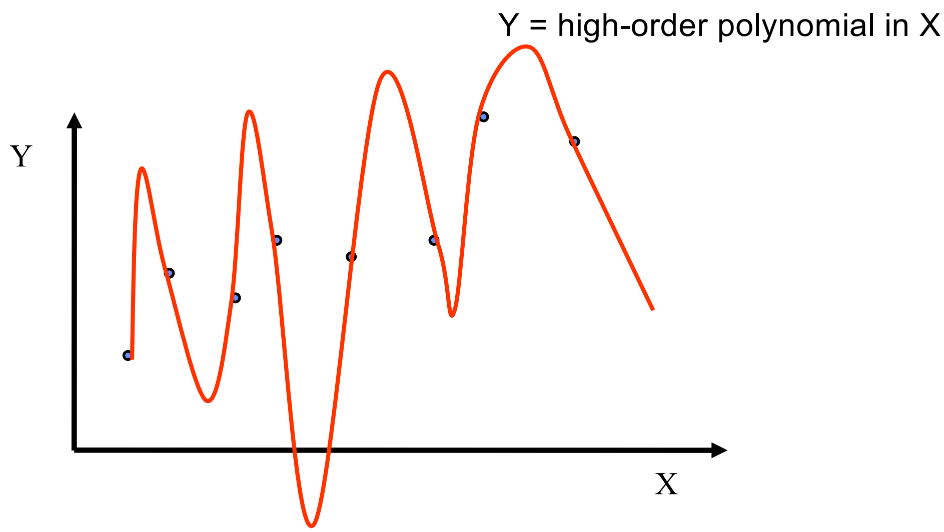
# Overfitting and Underfitting




Slide credit Richard H. Lathrop

55

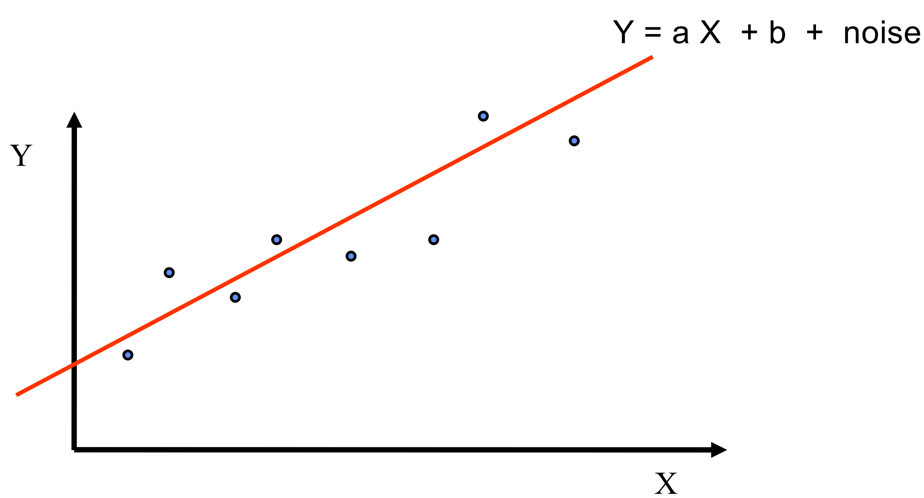
## A Complex Model




*Slide credit Richard H. Lathrop*

56

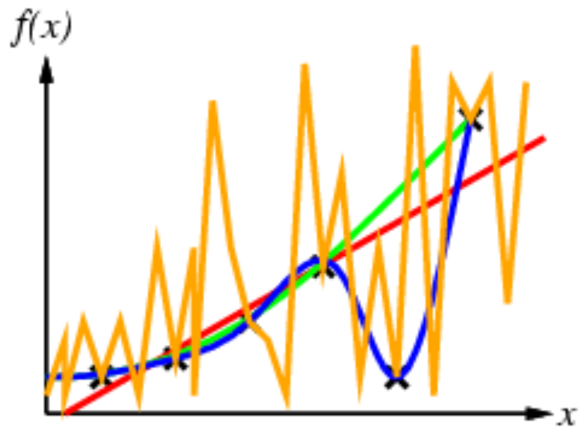
## A Much Simpler Model




*Slide credit Richard H. Lathrop*

57

## Another example




Slide credit Richard H. Lathrop

58

## Overfitting

- Fix by…
  - Getting more training data
  - Removing irrelevant features (e.g., remove 'first letter' from bird/mammal feature vector)
  - In decision trees, pruning low nodes (e.g., if improvement from best attribute at a node is below a threshold, stop and make this node a leaf rather than generating child nodes)
- Regularization
- Lots of other choices…

59

## Noisy Data

- Many kinds of "noise" can occur in the examples:
  - Two examples have same attribute/value pairs, but different classifications
  - Some values of attributes are incorrect
    - Errors in the data acquisition process, the preprocessing phase, …
  - Classification is wrong (e.g., + instead of -) because of some error
  - Some attributes are irrelevant to the decision-making process, e.g., color of a die is irrelevant to its outcome
  - Some attributes are missing (are pangolins bipedal?)

60

## Summary: Measuring Model Quality

- Performance on training, test, and deployment data
- Multiple failure modes: **false positive** vs. **false negative**
  - Which one is more important depends on your use case
- Precision and Recall tradeoff: do we want to be more **precise** or more **complete**? Or both?
  - F1 combines precision and recall
- Confusion matrices capture overall confusions
- One major type of failure: overfitting
  - Doing well on training data vs. actual deployment cases

61