CS440/ECE448 Lecture 8: Learning

Mark Hasegawa-Johnson, 1/2024

Public domain image: Classes at the University of Bologna. From *Liber ethicorum des Henricus de Alemannia, Laurentius a Voltolina, 14th century, scanned by* The Yorck Project , 2002

Outline

- Biological inspiration
- Parametric learning example: Decision tree
- A mathematical definition of learning
- Overtraining
- Early stopping

Biological inspiration: Hebbian learning

“Neurons that fire together, wire together.

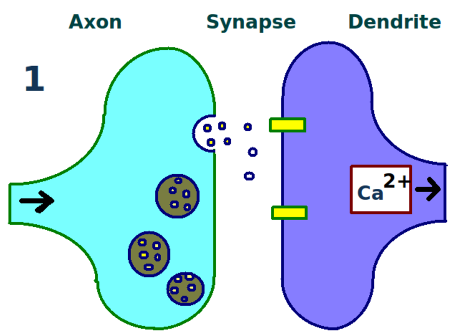
…

The general idea is an old one, that any two cells or systems of cells that are repeatedly active at the same time will tend to become `associated’ so that activity in one facilitates activity in the other.”

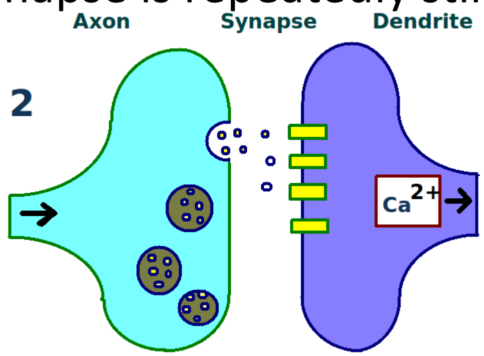
- D.O. Hebb, 1949

Biological inspiration: Long-term potentiation

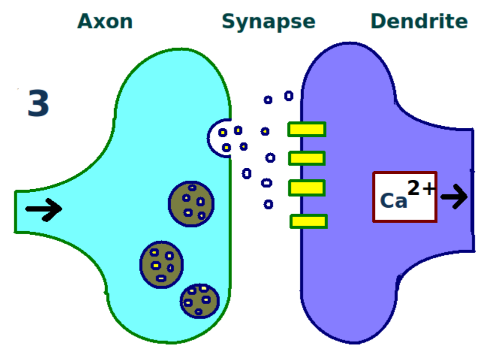
Figures this page are public domain, by Thomas W. Sulcer, 2011



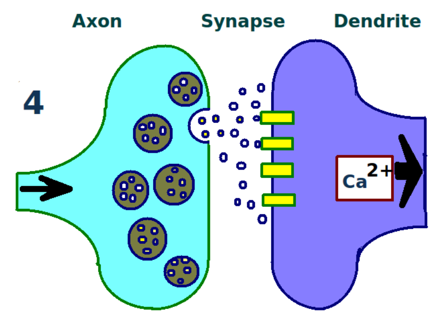
1. A synapse is repeatedly stimulated



2. More dendritic receptors



3. More neurotransmitters

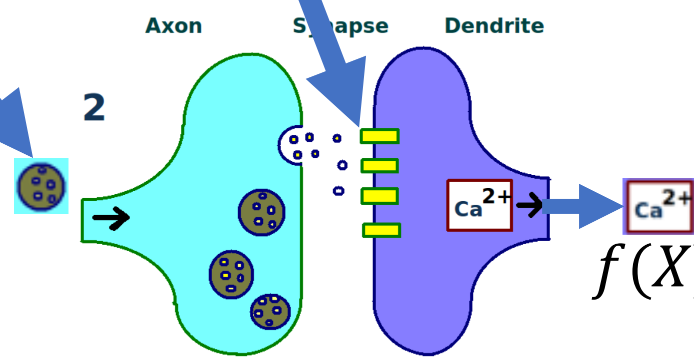


4. A stronger link between neurons

Mathematical model: Learning

Parameters of the learning machine: how many dendritic receptors exist? What types of neurotransmitter do they respond to?

X =input signal

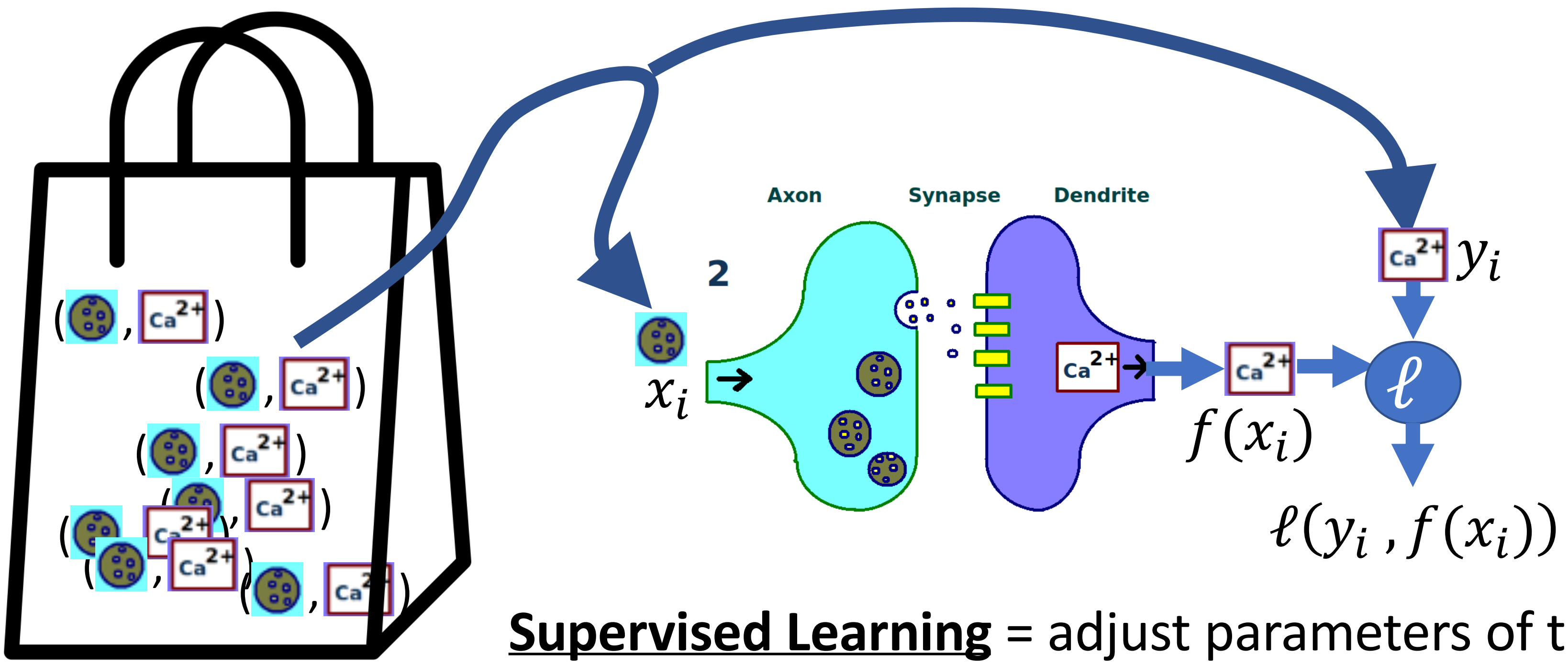


$f(X)$ =output signal

<u>Learning</u> = adjust the parameters of the learning machine so that $f(X)$ becomes the function we want

Mathematical model: Supervised Learning

Supervision: $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ = training dataset containing pairs of (example signal x_i, desired system output y_i)



Supervised Learning = adjust parameters of the learner to minimize $\mathrm{E}[\ell(Y, f(X))]$

Outline

- Biological inspiration
- Parametric learning example: Decision tree
- A mathematical definition of learning
- Overtraining
- Early stopping

Decision tree learning: An example

- The Titanic sank.
- You were rescued.
- You want to know if your friend was also rescued.
- You can't find them.
- Can you use machine learning methods to estimate the probability that your friend survived?

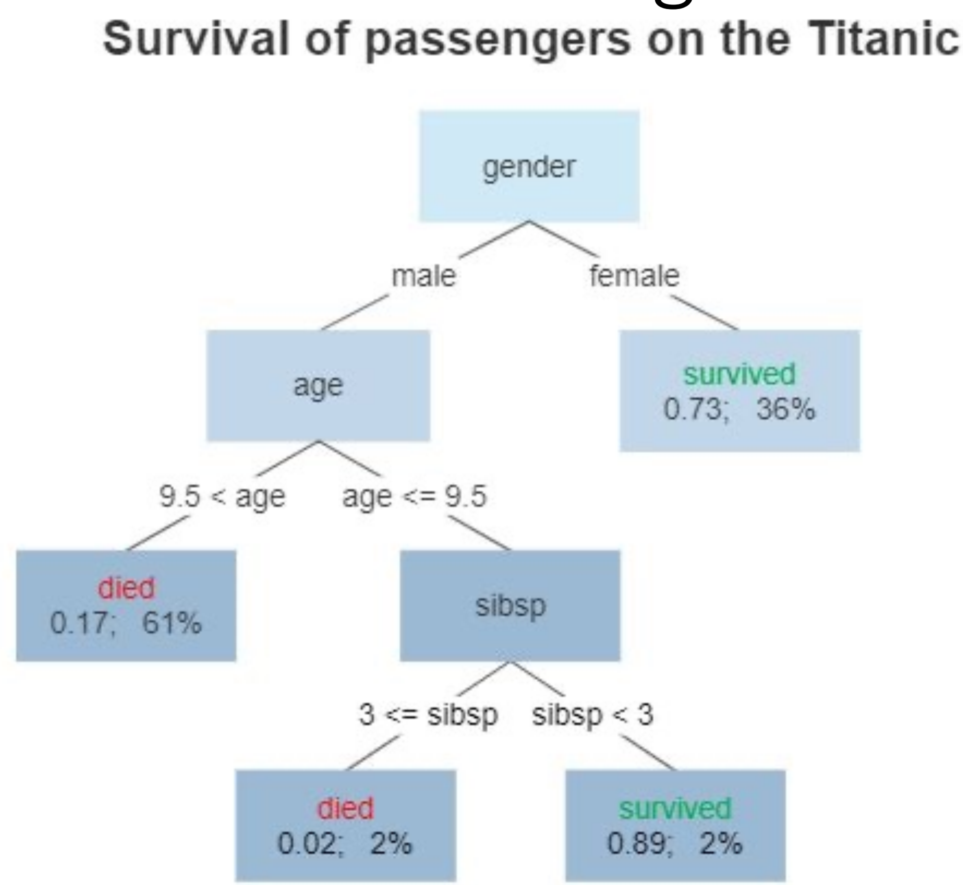
Survival of the Titanic: A machine learning approach

1. Gather data about as many of the passengers as you can.
   - X = variables that describe the passenger, e.g., age, gender, number of siblings on board.
   - Y = 1 if the person is known to have survived
2. Learn a function, f(X), that matches the known data as well as possible
3. Apply f(x) to your friend's facts, to estimate their probability of survival

Survival of the Titanic: A machine learning approach

Decision-tree learning:

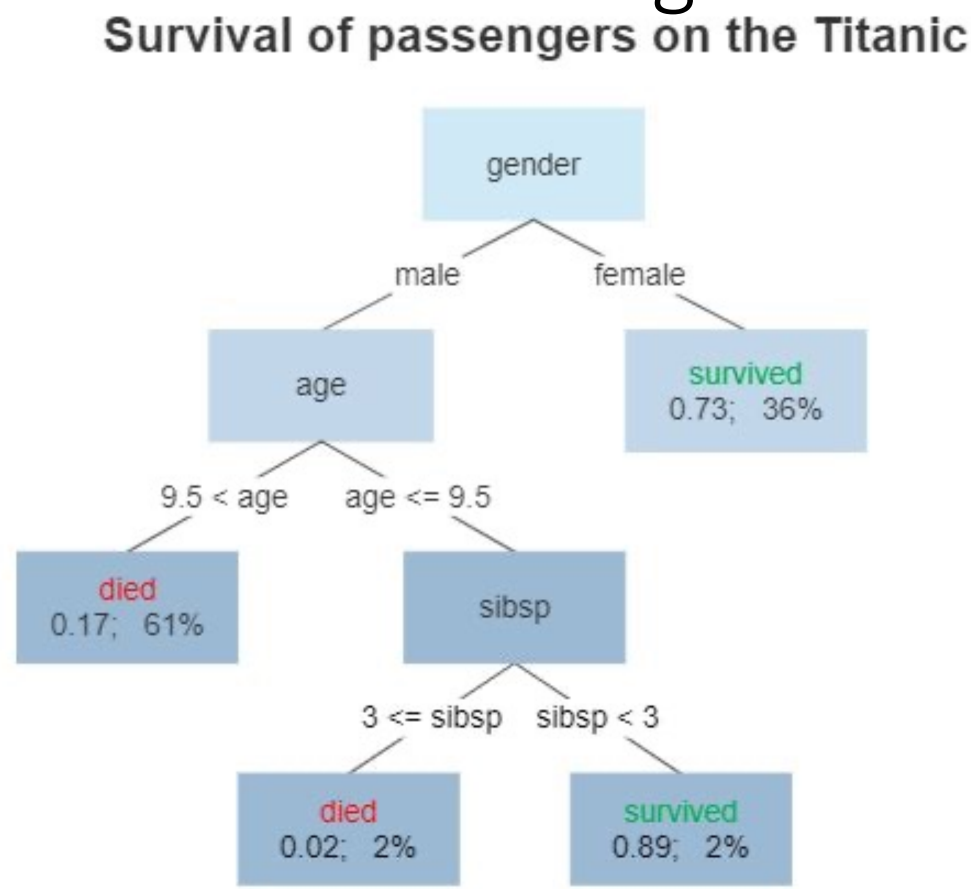
- 1st branch = variable that best distinguishes between groups with higher vs. lower survival rates (e.g., gender)
- 2nd branch = variable that best subdivides the remaining group
- Quit when all people in a group have the same outcome, or when the group is too small to be reliably subdivided.



Survival of the Titanic: A machine learning approach

In each leaf node of this tree:

- Number on the left = probability of survival
- Number on the right = percentage of all known cases that are explained by this node



Parametric Learner

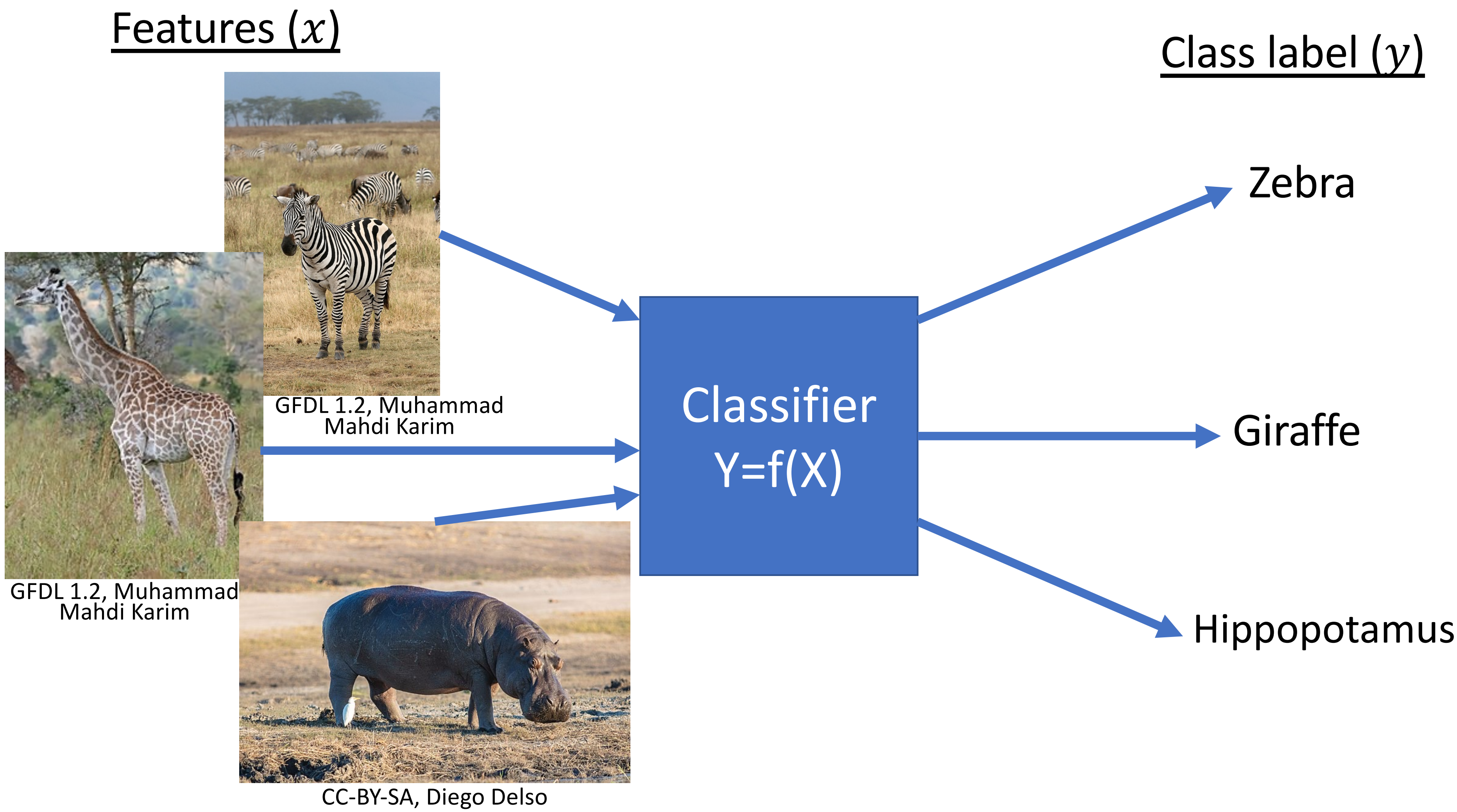
- A decision tree is an example of a parametric learner
- The function f(x) is determined by some learned parameters
- In this case, the parameters are:
  - Should this node split, or not?
  - If so, which tokens go to the right-hand child?
  - If not, what is $f(x)$ at the current node?
- Titanic shipwreck example:

$$\theta = [Y, \text{female}, Y, \text{age} \leq 9.5, N, f(x) = 0.73, \ldots]$$

Outline

- Biological inspiration
- Parametric learning example: Decision tree
- A mathematical definition of learning
- Overtraining
- Early stopping

Learning: learn a function $f(x)$, where x=features, y=true label, $f(x)$=estimated label



A mathematical definition of learning

- **Environment:** there are two random variables, X and Y, that are jointly distributed according to
$$P(X, Y)$$
- **Data:** $P(X, Y)$ is unknown, but we have a sample of training data
$$\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$$
- **Objective:** We would like a function f that minimizes the expected value of some loss function, $\ell(Y, f(X))$:
$$\mathcal{R} = \mathrm{E}[\ell(Y, f(X))]$$
- **Definition of learning:** Learning is the task of estimating the function f, given knowledge of $\mathcal{D}$.

Outline

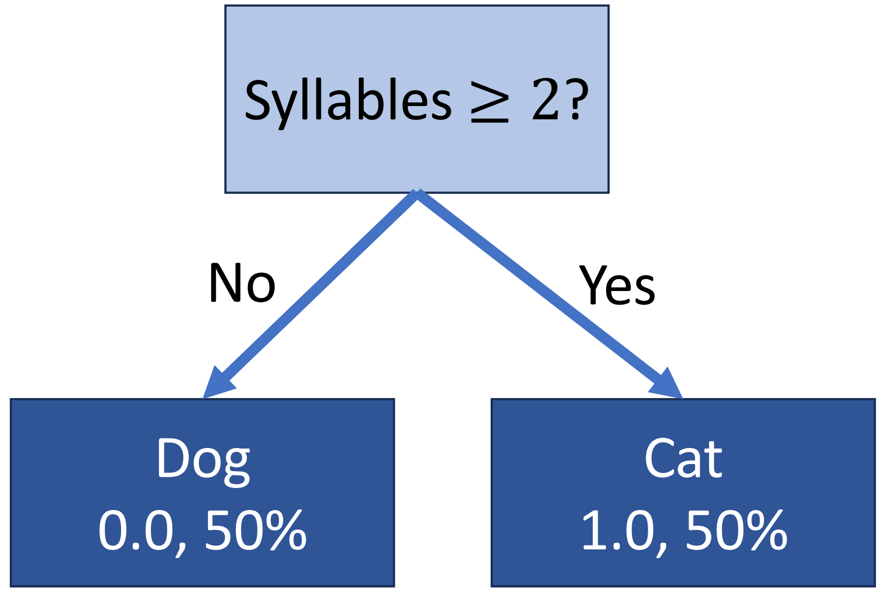
- Biological inspiration
- Parametric learning example: Decision tree
- A mathematical definition of learning
- Overtraining
- Early stopping

Overtraining

Consider the following experiment: among all of your friends' pets, there are 4 dogs and 4 cats.

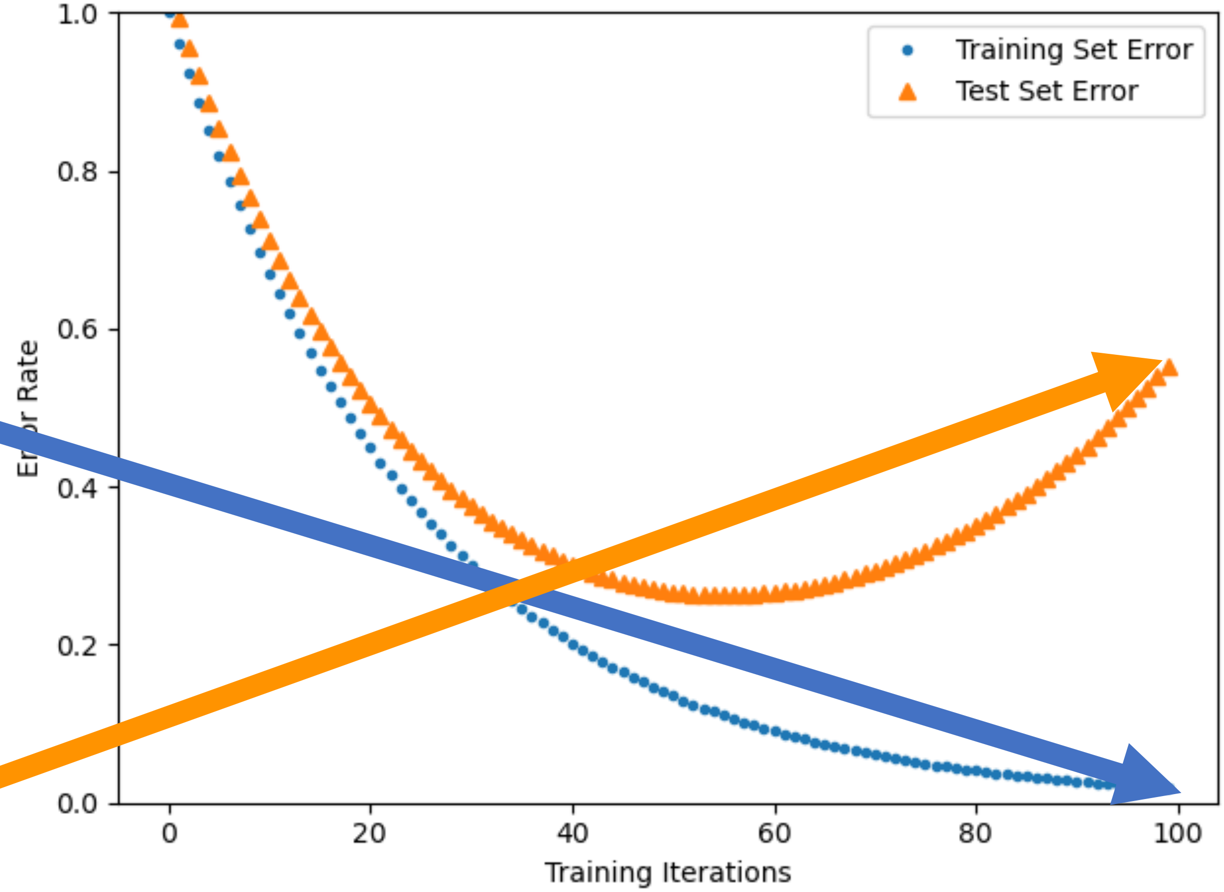
1. Measure several attributes of each animal: weight, height, color, number of letters in its name…
2. You discover that, among your friends' pets, all dogs have 1-syllable names, while the names of all cats have 2+ syllables.

Is it correct to say that this classifier has 100%? Is it useful to say so?



Training vs. Test Corpora

- Suppose you need 100 branch-nodes to reach zero training error
- … Then what is the training error after you find the best 100 questions?
- … and what is the error on a different “test” set then?



Training vs. Test Corpora

Training Corpus = a set of data that you use in order to optimize the parameters of your classifier (for example, optimize which features you measure, how you use those features to make decisions, and so on).

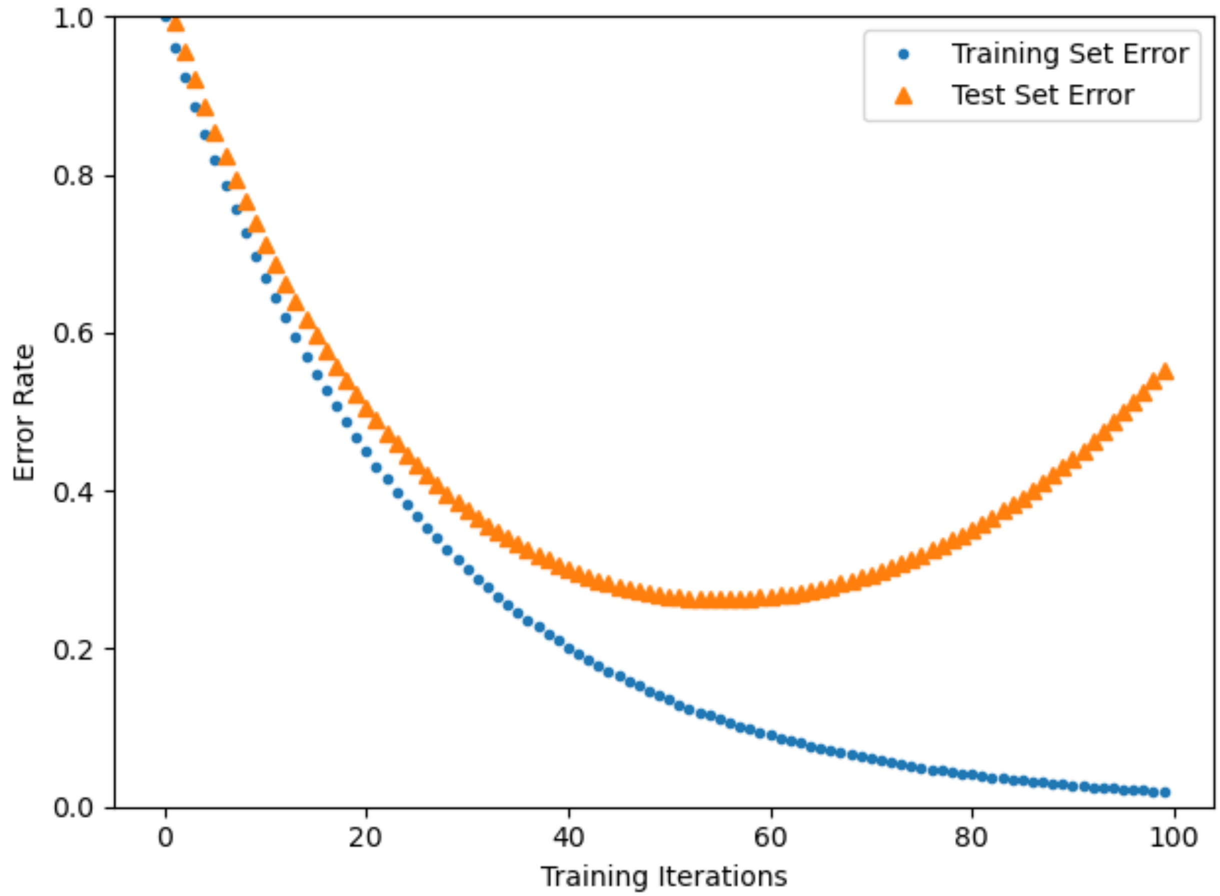
- Measuring the training corpus accuracy is important for debugging: if your training algorithm is working, then training corpus error rate should always go down.

Test Corpus = a set of data that is non-overlapping with the training set (none of the test tokens are also in the training dataset) that you can use to measure the error rate.

- Measuring the test corpus error rate is the only way to estimate how your classifier will work on new data (data that you’ve never yet seen).

Training vs. Test Corpora

- Training error is sometimes called "optimization error." It happens because you haven't finished optimizing your parameters.
- Test error = ***optimization error*** + ***generalization error***



Training corpus error vs. Test corpus error

- **Learning:** Given $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, find the function $f(X)$ that minimizes some measure of risk.
- **Empirical risk**, a.k.a. training corpus error:

$$\mathcal{R}_{\text{emp}} = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i))$$

- **True risk,** a.k.a. expected test corpus error:

$$\mathcal{R} = \mathrm{E}[\ell(Y, f(X))] = \mathcal{R}_{\text{emp}} + \mathcal{R}_{\text{generalization}}$$

Try the quiz!

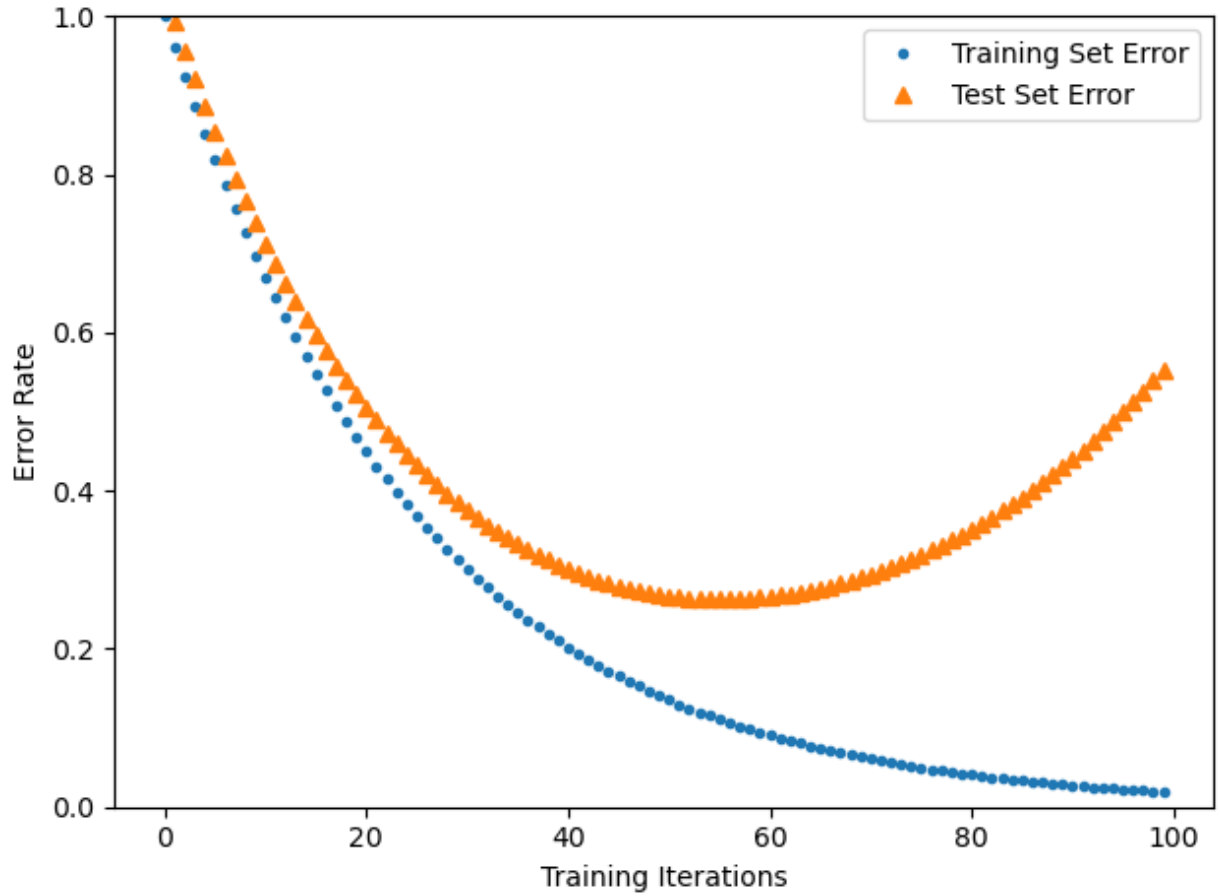
https://us.prairielearn.com/pl/course_instance/147925/assessment/2395191

Outline

- Biological inspiration
- Parametric learning example: Decision tree
- A mathematical definition of learning
- Overtraining
- Early stopping

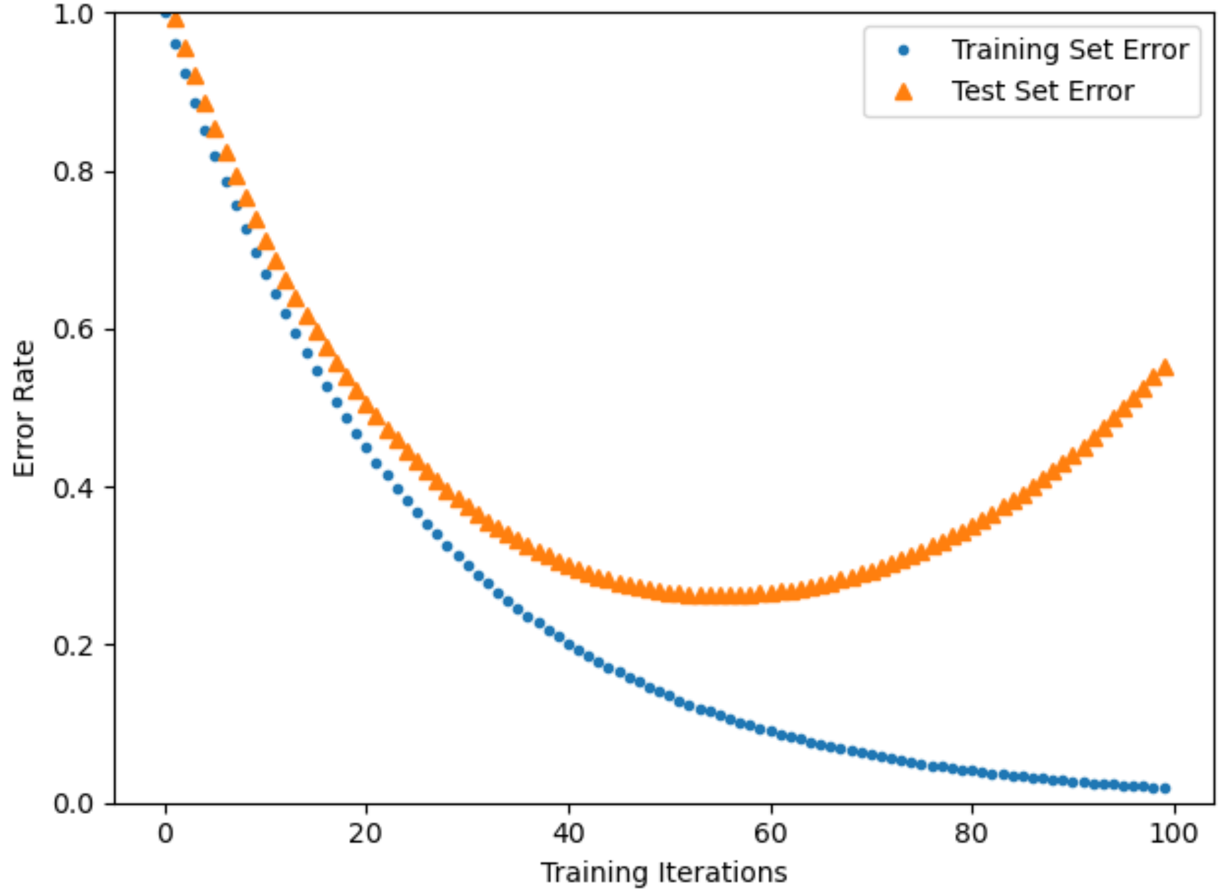
Training vs. Test Corpora

- As you iterate training, error on the training set should go to 0
- When should you stop training?



Cheaters always win

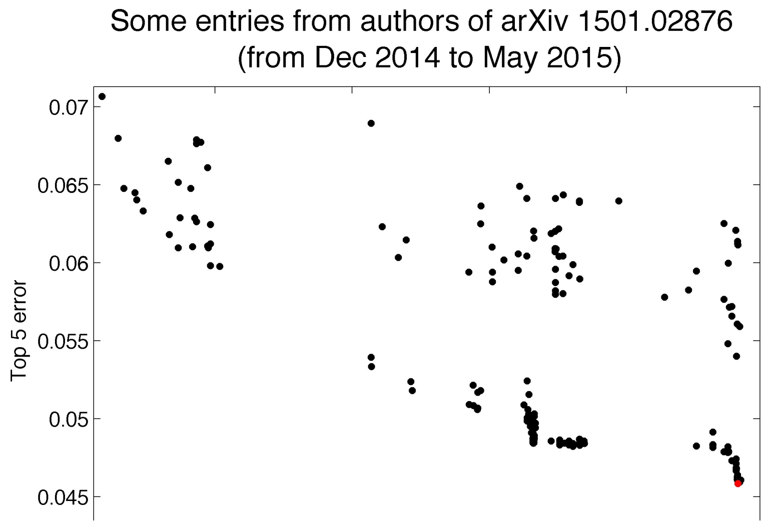
Why not just stop training when test set error reaches a minimum?

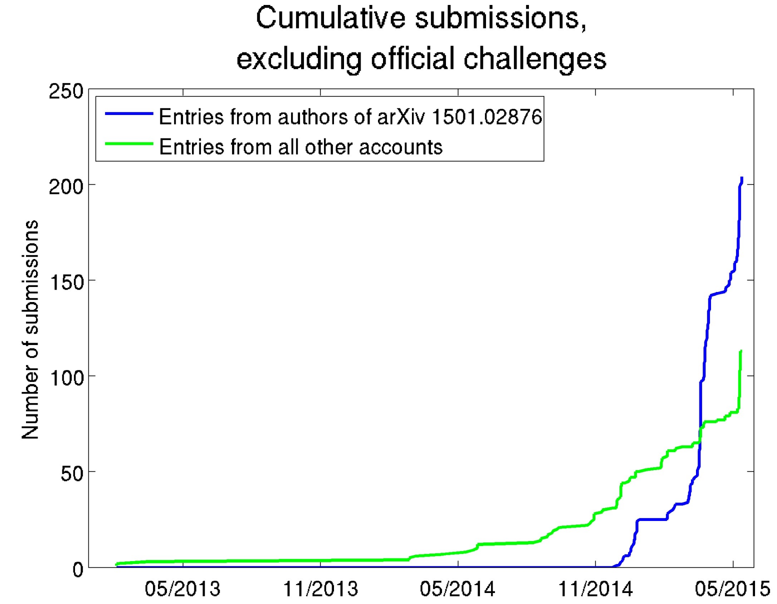


Accuracy on which corpus?

This happened:

- Large Scale Visual Recognition Challenge 2015: Each competing institution was allowed to test up to 2 different fully-trained classifiers per week.
- One institution used 30 different e-mail addresses so that they could test a lot more classifiers (200, total). One of their systems achieved <46% error rate – the competition's best, at that time.
- Is it correct to say that that institution's algorithm was the best?





Training vs. development test vs. evaluation test corpora

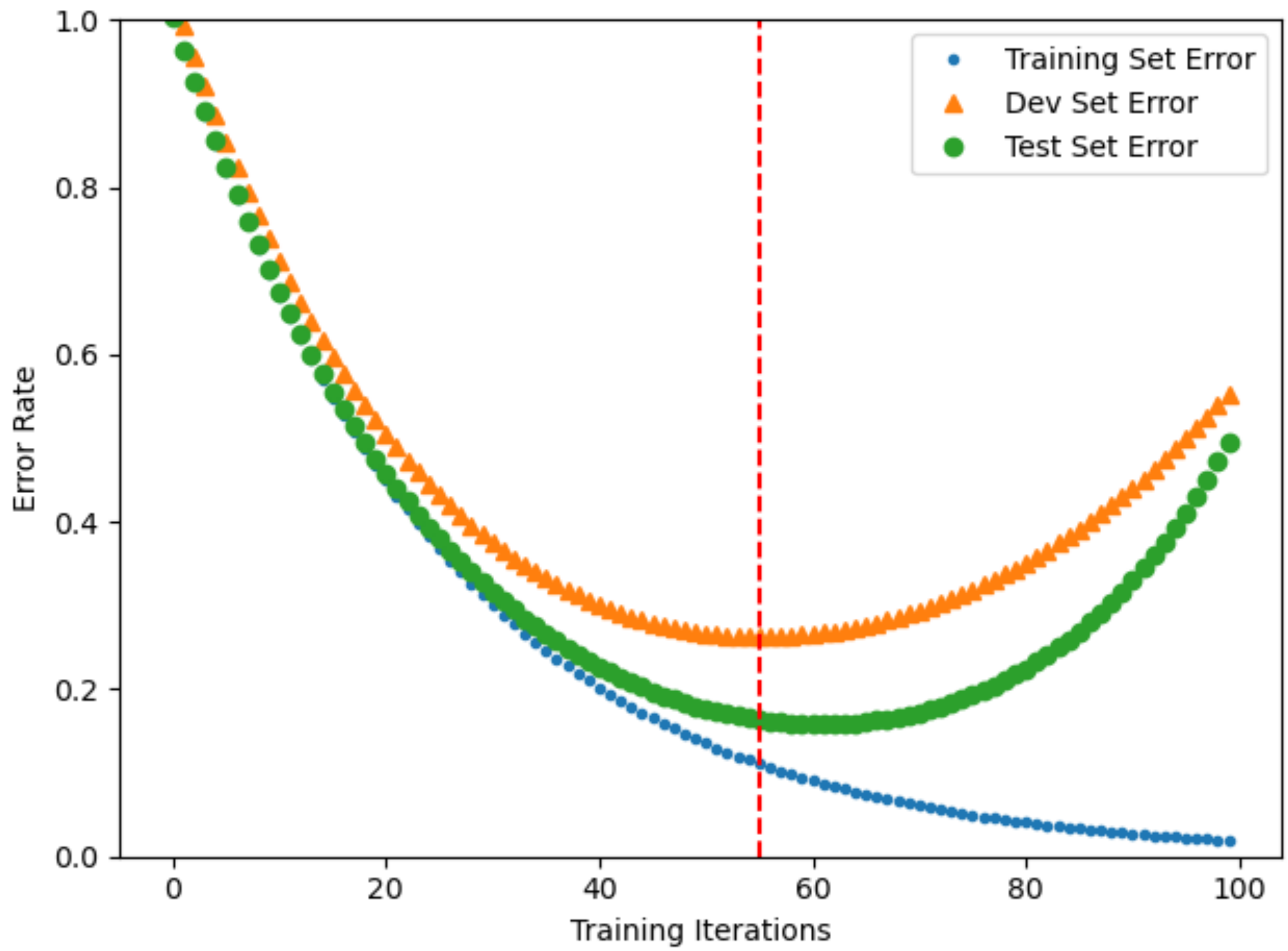
Training Corpus = a set of data that you use in order to optimize the parameters of your classifier (for example, optimize which features you measure, what are the weights of those features, what are the thresholds, and so on).

Development Test (DevTest or Validation) Corpus = a dataset, separate from the training dataset, on which you test 200 different fully-trained classifiers (trained, e.g., using different training algorithms, or different features) to find the best.

Evaluation Test Corpus = a dataset that is used only to test the ONE classifier that does best on DevTest. From this corpus, you learn how well your classifier will perform in the real world.

Train, Dev, Test

- Usually, minimum test error and minimum dev error don’t occur at the same time
- … but early stopping based on the test set is cheating,
- … so early stopping based on the dev set is the best we can do w/o cheating.



Summary

- **Biological inspiration:** Neurons that fire together wire together. Given enough training examples (x_i, y_i), can we learn a desired function so that $f(x) \approx y$?
- **Classification tree:** Learn a sequence of if-then statements that computes $f(x) \approx y$
- **Mathematical definition of supervised learning:** Given a training dataset, $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, find a function f that minimizes the risk, $\mathcal{R} = \mathrm{E}[\ell(Y, f(X))]$.
- **Overtraining:** $\mathcal{R}_{\text{emp}} = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i))$ reaches zero if you train long enough.
- **Early Stopping**: Stop when error rate on the dev set reaches a minimum