Tufts

# Class #02: Types of Learning; Information Theory

Machine Learning (COMP 135): M. Allen, 09 Sept. 19

## Defining a Learning Problem

- Suppose we have three basic components:
  1. Set of tasks, *T*
  2. A performance measure, *P*
  3. Data describing some experience, *E*

> A computer program ***learns*** if its performance at tasks in *T*, as measured by *P*, improves based on *E*.
>
> From: Tom M. Mitchell, *Machine Learning* (1997)

## An Example Problem

- Suppose we want to build a system, like Siri or Alexa, that responds to voice commands
- What are our components?
  1. Tasks, *T*
  2. Performance measure, *P*
  3. Experience, *E*

***Task:***

Take system actions, based upon speech

***Performance:***

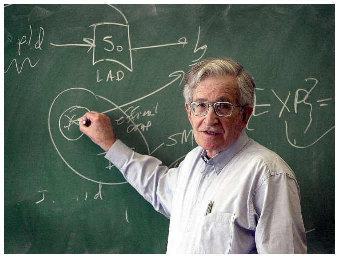
How often correct action is taken during testing

***Experience?***

This is the tricky part!

## The Expert Systems Approach

- One (older) approach used ***expert-generated rules***:
  1. Find someone with advanced knowledge of linguistics
  2. Get them to devise the structural rules of language's grammar and semantics
  3. Encode those rules in program for parsing written language
  4. Build another program to translate speech into written language, and tie that to ***another*** program for taking actions based upon the parsing



## Another Approach: Supervised Learning

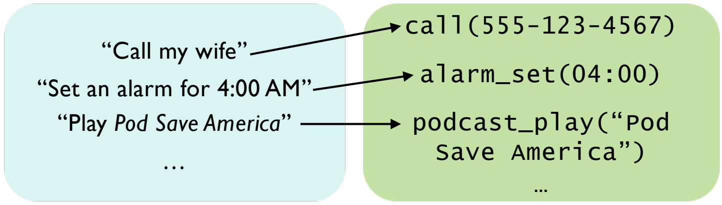
- In supervised learning, we:

1. Provide a set of ***correct answers*** to a problem
2. Use algorithms to find (mostly) correct answers to ***similar problems***

- We can still use experts, but their job is different:
  - ***Don't need*** to devise complex rules for understanding speech
  - ***Instead***, they just have to be able to tell what the ***correct results*** of understanding look like

## Another Approach: Supervised Learning

- Collect a large set of sample things a set of test users say to our system
- For each, map it to a correct outcome action the system should take



"Call my wife" → `call(555-123-4567)`

"Set an alarm for 4:00 AM" → `alarm_set(04:00)`

"Play *Pod Save America*" → `podcast_play("Pod Save America")`

… …

- A large set of such (*speech, action*) pairs can be created
- This can then form the experience, *E*, the system needs

## Inductive Learning

- In its simplest form, induction is the task of learning a ***function*** on some inputs from ***examples*** of its outputs
- For a function, $f$, that we want to learn, each of these training examples is a pair

$$(x,\ f(x))$$

  - We assume that we do not yet know the actual form of the function $f$ (if we did, we don't need to learn)

- ***Learning problem***: find a hypothesis function, $h$, such that $h(x) = f(x)$ (at least ***most*** of the time), based on a training set of example input-output pairs

## Decisions to Make

- When collecting our training example pairs, $(x,\ f(x))$, we still have some decisions to make
- ***Example***: Medical Informatics
  - We have some genetic information about patients
  - Some get sick with a disease and some don't
  - Patients live for a number of years (sick or not)
- ***Question***: what do we want to learn from this data?
- Depending upon what we decide, we may use:
  - Different models of the data
  - Different machine learning approaches
  - Different measurements of successful learning

## One Approach: Regression

- We decide that we want to try to learn to predict how long patients will live
- We base this upon information about the degree to which they express a specific gene
- A regression problem: the function we learn is the "best (linear) fit" to the data we have

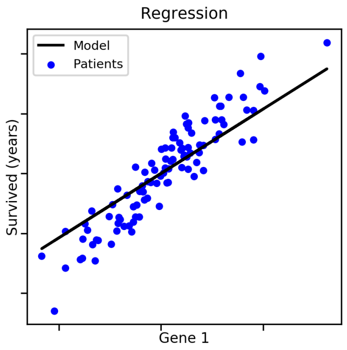


Image source: https://aldro61.github.io/microbiome-summer-school-2017/sections/basics/

## Another Approach: Classification

- We decide instead that we simply want to decide whether a patient will get the disease or not
- We base this upon information about expression of two genes
- A classification problem: learned function separates individuals into 2 groups (binary classes)

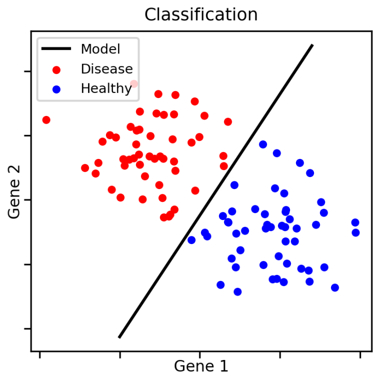
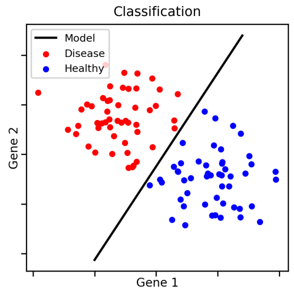
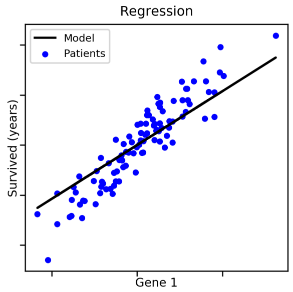


Image source: https://aldro61.github.io/microbiome-summer-school-2017/sections/basics/

## Which is the Correct Approach?





- The approach we use depends upon what we want to achieve, and what works best based upon the data we have
- Much machine learning involves investigating different approaches

## Uncertainty and Learning

- Often, when learning, we deal with uncertainty:
  - Incomplete data sets, with ***missing*** information
  - Noisy data sets, with ***unreliable*** information
  - Stochasticity: causes and effects related ***non-deterministically***
  - And many more…
- Probability theory gives us mathematics for such cases
  - A precise mathematical theory of chance and causality

## Basic Elements of Probability

- Suppose we have some event, $e$: some fact about the world that may be true or false
- We write $P(e)$ for the probability that $e$ occurs:

$$0 \leq P(e) \leq 1$$

- We can understand this value as:

1. $P(e) = 1$: $e$ will certainly happen
2. $P(e) = 0$: $e$ will certainly ***not*** happen
3. $P(e) = k$, $0 < k < 1$: over an arbitrarily long stretch of time, we will observe the fraction

$$\frac{\text{Event } e \text{ occurs}}{\text{Total } \# \text{ of events}} = k$$

## Properties of Probability

- Every event must either ***occur***, or ***not occur***:

$$P(e \vee \neg e) = 1$$

$$P(e) = 1 - p(\neg e)$$

- Furthermore, suppose that we have a set of ***all possible*** events, each with its own probability:

$$\mathcal{E} = \{e_1, e_2, \ldots, e_k\}$$

$$\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$$

- This set of probabilities is called a probability distribution, and it must have the following property:

$$\sum_i p_i = 1$$

## Probability Distributions

- A uniform distribution is one in which every event occurs with equal probability, which means that we have:

$$\mathcal{P} = \{p_1, p_2, \ldots, p_k\} \quad \wedge \quad \forall i, \, p_i = \frac{1}{k}$$

- Such distributions are common in games of chance, e.g. where we have a fair coin-toss:

$$\mathcal{E} = \{Heads, Tails\}$$

$$\mathcal{P}_1 = \{0.5, 0.5\}$$

- ***Not every*** distribution is uniform, and we might have a coin that comes up tails ***more often*** than heads (or even ***always***!)

$$\mathcal{P}_2 = \{0.25, 0.75\}$$

$$\mathcal{P}_3 = \{0.0, 1.0\}$$

## Information Theory

- Claude Shannon created information theory in his 1948 paper, "A mathematical theory of communication"
- A theory of the amount of information that can be carried by communication channels
- Has implications in networks, encryption, compression, and many other areas
- Also the source of the term "bit" (credited to John Tukey)

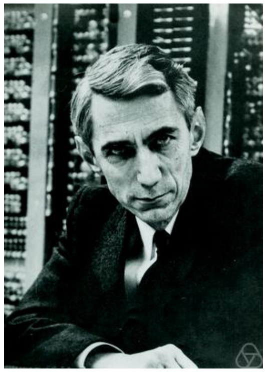
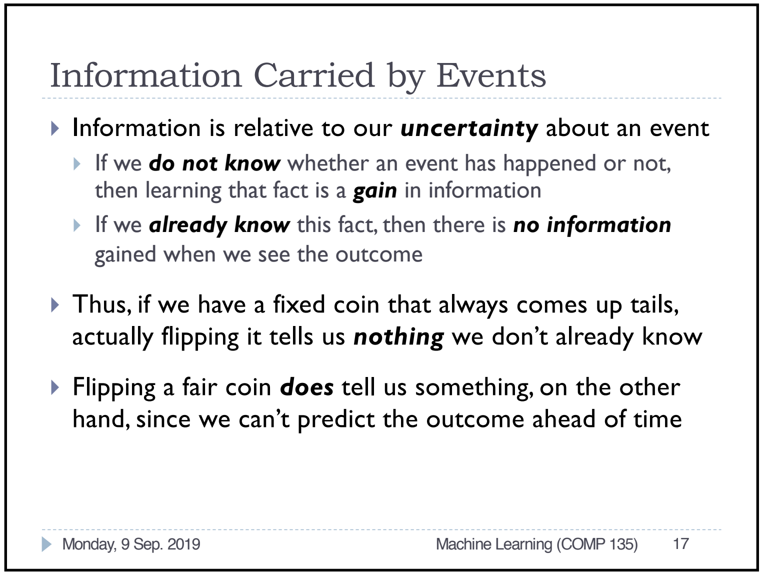
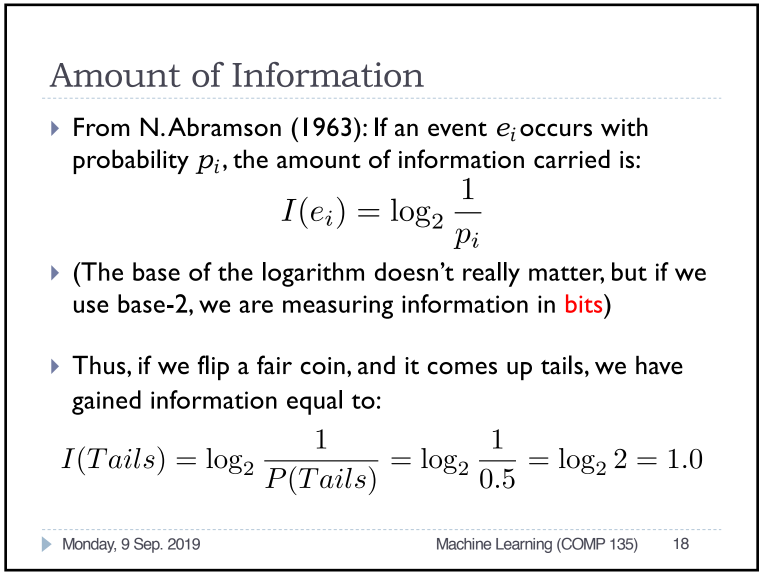


Photo source: Konrad Jacobs (https://opc.mfo.de/detail?photo_id=3807)



## Information Carried by Events

- Information is relative to our ***uncertainty*** about an event
  - If we ***do not know*** whether an event has happened or not, then learning that fact is a ***gain*** in information
  - If we ***already know*** this fact, then there is ***no information*** gained when we see the outcome
- Thus, if we have a fixed coin that always comes up tails, actually flipping it tells us ***nothing*** we don't already know
- Flipping a fair coin ***does*** tell us something, on the other hand, since we can't predict the outcome ahead of time



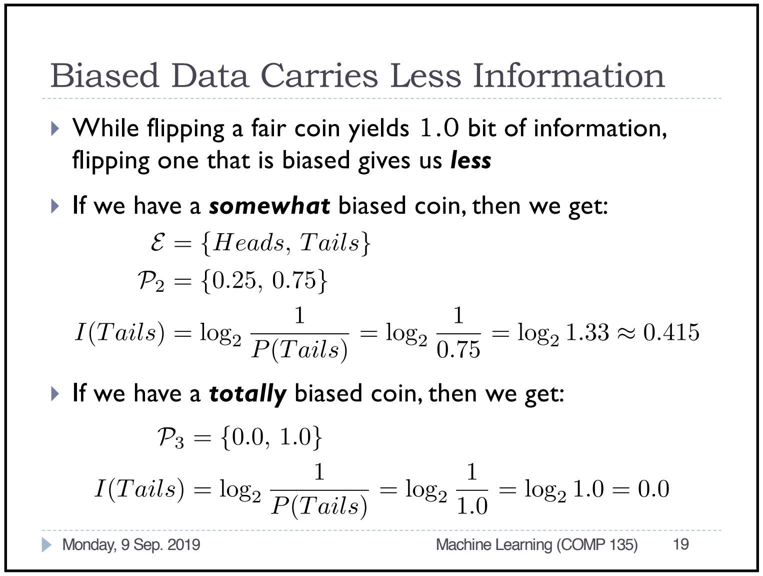
## Amount of Information

- From N. Abramson (1963): If an event $e_i$ occurs with probability $p_i$, the amount of information carried is:

$$I(e_i) = \log_2 \frac{1}{p_i}$$

- (The base of the logarithm doesn't really matter, but if we use base-2, we are measuring information in bits)
- Thus, if we flip a fair coin, and it comes up tails, we have gained information equal to:

$$I(Tails) = \log_2 \frac{1}{P(Tails)} = \log_2 \frac{1}{0.5} = \log_2 2 = 1.0$$



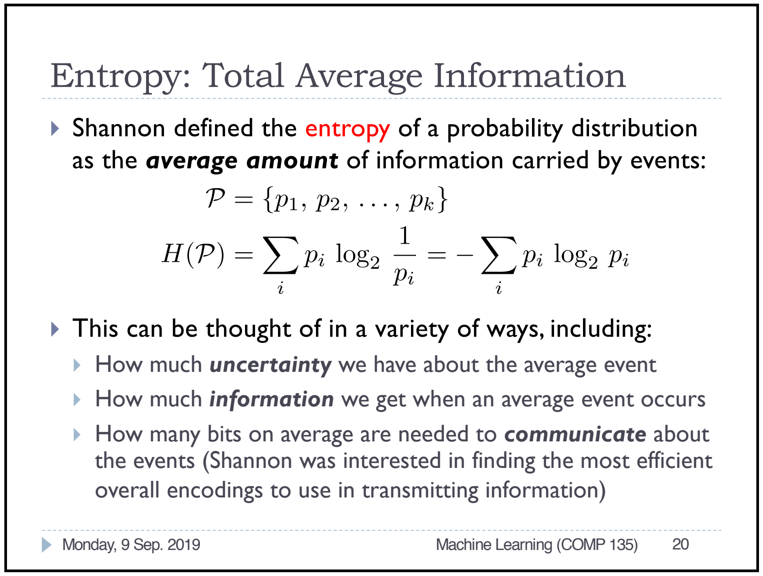
## Biased Data Carries Less Information

- While flipping a fair coin yields $1.0$ bit of information, flipping one that is biased gives us ***less***
- If we have a ***somewhat*** biased coin, then we get:

$$\mathcal{E} = \{Heads, Tails\}$$
$$\mathcal{P}_2 = \{0.25, 0.75\}$$
$$I(Tails) = \log_2 \frac{1}{P(Tails)} = \log_2 \frac{1}{0.75} = \log_2 1.33 \approx 0.415$$

- If we have a ***totally*** biased coin, then we get:

$$\mathcal{P}_3 = \{0.0, 1.0\}$$
$$I(Tails) = \log_2 \frac{1}{P(Tails)} = \log_2 \frac{1}{1.0} = \log_2 1.0 = 0.0$$



## Entropy: Total Average Information

- Shannon defined the entropy of a probability distribution as the ***average amount*** of information carried by events:

$$\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$$
$$H(\mathcal{P}) = \sum_i p_i \log_2 \frac{1}{p_i} = -\sum_i p_i \log_2 p_i$$

- This can be thought of in a variety of ways, including:
  - How much ***uncertainty*** we have about the average event
  - How much ***information*** we get when an average event occurs
  - How many bits on average are needed to ***communicate*** about the events (Shannon was interested in finding the most efficient overall encodings to use in transmitting information)

## Entropy: Total Average Information

- For a coin, *C*, the formula for entropy becomes:

$$H(C) \;=\; -(P(Heads)\,\log_2\,P(Heads) + P(Tails)\,\log_2\,P(Tails))$$

- A fair coin, {0.5, 0.5}, has ***maximum*** entropy:

$$H(C) \;=\; -(0.5\,\log_2\,0.5 + 0.5\,\log_2\,0.5) = 1.0$$

- A somewhat biased coin, {0.25, 0.75}, has ***less***:

$$H(C) = -(0.25\log_2 0.25 + 0.75\log_2 0.75) \approx 0.81$$

- And a fixed coin, {0.0, 1.0}, has ***none***:

$$H(C) \;=\; -(1.0\,\log_2\,1.0 + 0.0\,\log_2\,0.0) = 0.0$$

## A Mathematical Definition

$$H(\mathcal{P}) = -\sum_i p_i\,\log_2\,p_i$$

- It is easy to show that for any distribution, entropy is always greater than or equal to 0 (***never negative***)
- ***Maximum*** entropy occurs with a uniform distribution
  - In such cases, entropy is $\log_2\,k$, where $k$ is the number of different probabilistic outcomes
- Thus, for any distribution possible, we have:

$$\mathcal{P} = \{p_1,\,p_2,\,\ldots,\,p_k\}$$
$$0 \leq H(\mathcal{P}) \leq \log_2 k$$

## This Week

- Information Theory & Decision Trees
  - Some material in these slides drawn from Russel & Norvig, *Artificial Intelligence: A Modern Approach* (Prentice Hal, 2010)
- Readings:
  - Blog post on Information Theory (linked from class schedule)
  - Chapter 1 of the Daumé text (linked from class schedule)
- Office Hours: 237 Halligan
  - Tuesday, 11:00 AM – 1:00 PM