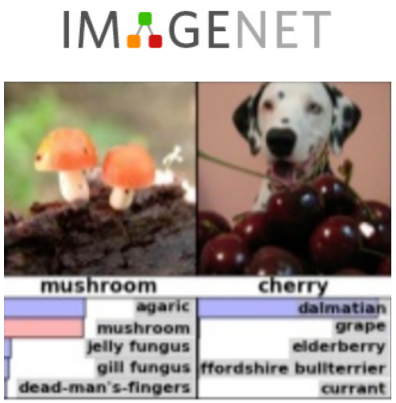
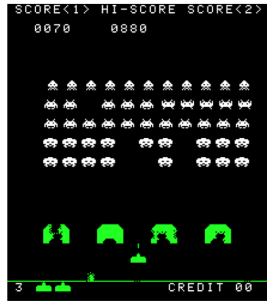
CSC2541 Lecture 1
Introduction

Roger Grosse

Motivation

- Recent success stories of machine learning, and neural nets in particular





Google Translate

- But our algorithms still struggle with a decades-old problem: knowing what they don't know

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting
  - **Ensembling:** smooth your predictions by averaging them over multiple possible models

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting
  - **Ensembling:** smooth your predictions by averaging them over multiple possible models
  - **Model selection:** decide which of multiple plausible models best describes the data

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting
  - **Ensembling:** smooth your predictions by averaging them over multiple possible models
  - **Model selection:** decide which of multiple plausible models best describes the data
  - **Sparsification:** drop connections, encode them with fewer bits

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting
  - **Ensembling:** smooth your predictions by averaging them over multiple possible models
  - **Model selection:** decide which of multiple plausible models best describes the data
  - **Sparsification:** drop connections, encode them with fewer bits
  - **Exploration**
    - **Active learning:** decide which training examples are worth labeling
    - **Bandits:** improve the performance of a system where the feedback actually counts (e.g. ad targeting)
    - **Bayesian optimization:** optimize an expensive black-box function
    - **Model-based reinforcement learning** (potential orders-of-magnitude gain in sample efficiency!)

Motivation

- Why model uncertainty?
  - **Confidence calibration:** know how reliable a prediction is (e.g. so it can ask a human for clarification)
  - **Regularization:** prevent your model from overfitting
  - **Ensembling:** smooth your predictions by averaging them over multiple possible models
  - **Model selection:** decide which of multiple plausible models best describes the data
  - **Sparsification:** drop connections, encode them with fewer bits
  - **Exploration**
    - **Active learning:** decide which training examples are worth labeling
    - **Bandits:** improve the performance of a system where the feedback actually counts (e.g. ad targeting)
    - **Bayesian optimization:** optimize an expensive black-box function
    - **Model-based reinforcement learning** (potential orders-of-magnitude gain in sample efficiency!)
  - **Adversarial robustness:** make good predictions when the data might have been perturbed by an adversary

Course Overview

- Weeks 2–3: Bayesian function approximation
  - Bayesian neural nets
  - Gaussian processes
- Weeks 4–5: variational inference
- Weeks 6–8: using uncertainty to drive exploration
- Weeks 9–10: other topics (adversarial robustness, optimization)
- Weeks 11–12: project presentations

What we Don't Cover

- Uncertainty in ML is way too big a topic for one course.
- Focus on uncertainty in function approximation, and its use in directing exploration and improving generalization.
- How this differs from other courses
  - No generative models or discrete Bayesian models (covered in other iterations of 2541)
  - CSC412, STA414, and ECE521 are core undergrad courses giving broad coverage of probabilistic modeling.
    - We cover fewer topics in more depth, and more cutting edge research.
  - This is an ML course, not a stats course.
    - Lots of overlap, but problems are motivated by use in AI systems rather than human interpretability.

Adminis-trivia: Presentations

- 10 lectures
  - Each one covers about 4–6 papers.
- I will give 3 (including this one).
- The remaining 7 will be student presentations.
  - 8–12 presenters per lecture (signup procedure to be announced soon)
  - Divide lecture into sub-topics on an ad-hoc basis
  - Aim for a total of about 75 minutes plus questions/discussion
  - I will send you advice roughly 2 weeks in advance
  - **Bring a draft presentation to office hours.**

Adminis-trivia: Projects

- Goal: write a workshop-quality paper related to the course topics
- Work in groups of 3–5
- Types of projects
  - Tutorial/review article.
    - Must have clear value-added: explain the relationship between different algorithms, come up with illustrative examples, run experiments on toy problems, etc.
  - Apply an existing algorithm in a new setting.
  - Invent a new algorithm.
- You're welcome to do something related to your research (see handout for detailed policies)
- Full information:
  `https://csc2541-f17.github.io/project-handout.pdf`

Adminis-trivia: Projects

- Project proposal (due Oct. 12)
  - about 2 pages
  - describe motivation, related work
- Presentations (Nov. 24 and Dec. 1)
  - Each group has 5 minutes + 2 minutes for questions.
- Final report (due Dec. 10)
  - about 8 pages plus references (not strictly enforced)
  - submit code also
- See handout for specific policies.

Adminis-trivia: Marks

- Class presentations — 20%
- Project Proposal — 20%
- Projects — 60%
  - 85% (A-/A) for meeting requirements, last 15% for going above and beyond
  - See handout for specific requirements and breakdown

History of Bayesian Modeling

- 1763 — Bayes' Rule published (further developed by Laplace in 1774)
- 1953 — Metropolis algorithm (extended by Hastings in 1970)
- 1984 — Stuart and Donald Geman invent Gibbs sampling (more general statistical formulation by Gelfand and Smith in 1990)
- 1990s — Hamiltonian Monte Carlo
- 1990s — Bayesian neural nets and Gaussian processes
- 1990s — probabilistic graphical models
- 1990s — sequential Monte Carlo
- 1990s — variational inference
- 1997 — BUGS probabilistic programming language
- 2000s — Bayesian nonparametrics
- 2010 — stochastic variational inference
- 2012 — Stan probabilistic programming language

History of Neural Networks

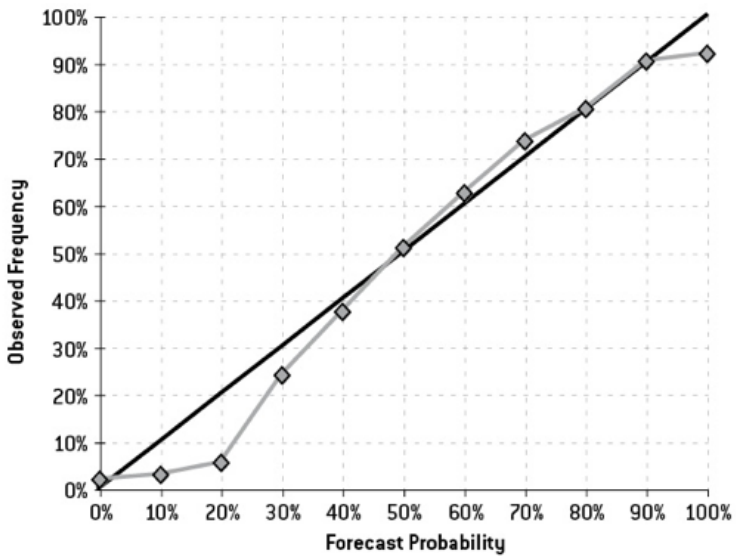
- 1949 — Hebbian learning ("fire together, wire together")
- 1957 — perceptron algorithm
- 1969 — Minsky and Papert's book *Perceptrons* (limitations of linear models)
- 1982 — Hopfield networks (model of associative memory)
- 1988 — backpropagation
- 1989 — convolutional networks
- 1990s — neural net winter
- 1997 — long-term short-term memory (LSTM) (not appreciated until last few years)
- 2006 — "deep learning"
- 2010s — GPUs
- 2012 — AlexNet smashes the ImageNet object recognition benchmark, leading to the current deep learning boom
- 2016 — AlphaGo defeats human Go champion

This Lecture

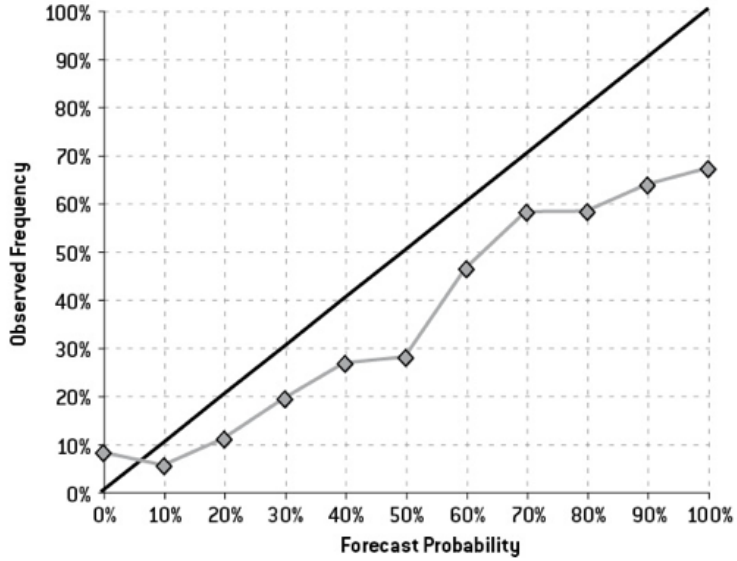
- confidence calibration
- intro to Bayesian modeling: coin flip example
- n-armed bandits and exploration
- Bayesian linear regression

Calibration

- Calibration: of the times your model predicts something with 90% confidence, is it right 90% of the time?
- From Nate Silver's book, "The Signal and the Noise": calibration of weather forecasts



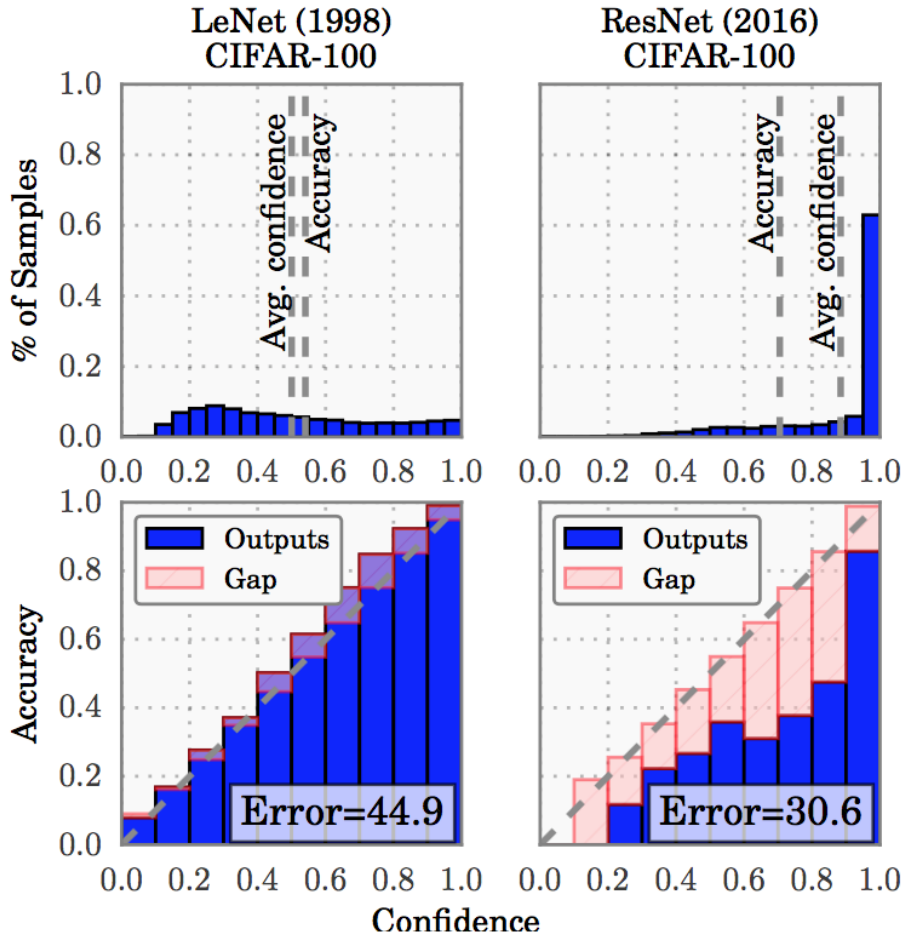
The Weather Channel



local weather station

Calibration

- Most of our neural nets output probability distributions, e.g. over object categories. Are these calibrated?
- From Guo et al. (2017):



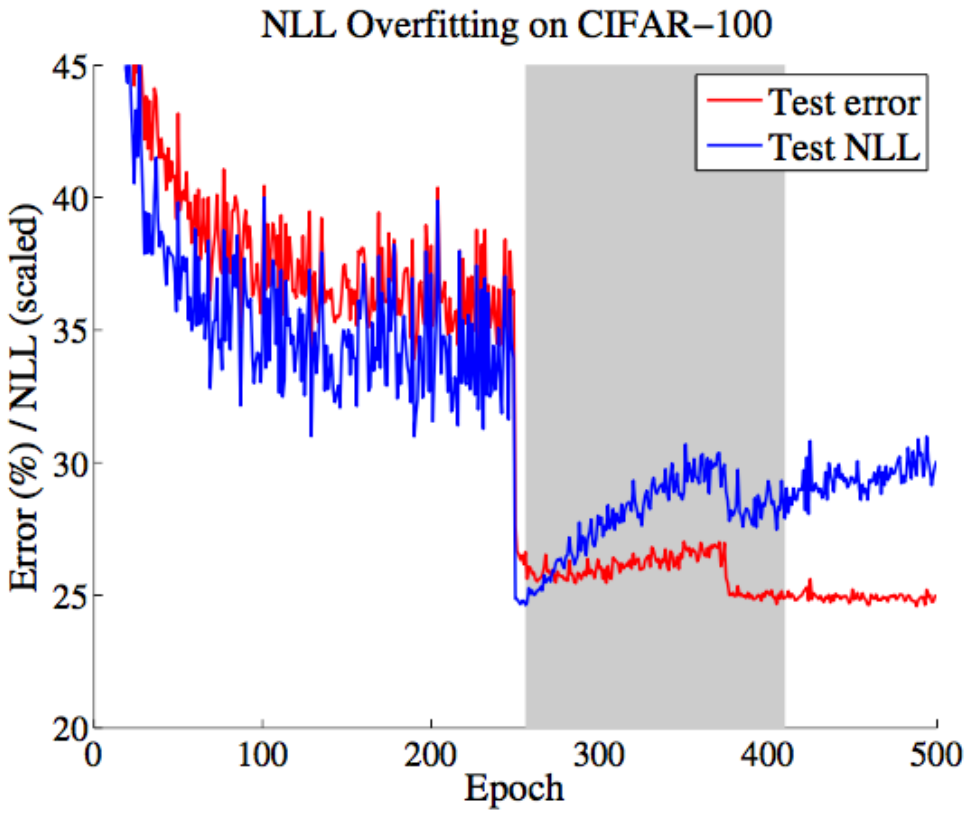
Calibration

- Suppose an algorithm outputs a probability distribution over targets, and gets a loss based on this distribution and the true target.
- A proper scoring rule is a scoring rule where the algorithm's best strategy is to output the true distribution.
- The canonical example is negative log-likelihood (NLL). If k is the category label, $\mathbf{t}$ is the indicator vector for the label, and $\mathbf{y}$ are the predicted probabilities,

$$L(\mathbf{y}, \mathbf{t}) = -\log y_k = -\mathbf{t}^\top (\log \mathbf{y})$$

Calibration

- Calibration failures show up in the test NLL scores:



— Guo et al., 2017, On calibration of modern neural networks

Calibration

- Guo et al. explored 7 different calibration methods, but the one that worked the best was also the simplest: temperature scaling.
- A classification network typically predicts $\sigma(\mathbf{z})$, where σ is the softmax function

$$\sigma(\mathbf{z})_k = \frac{\exp(z_k)}{\sum_{k'} \exp(z_{k'})}$$

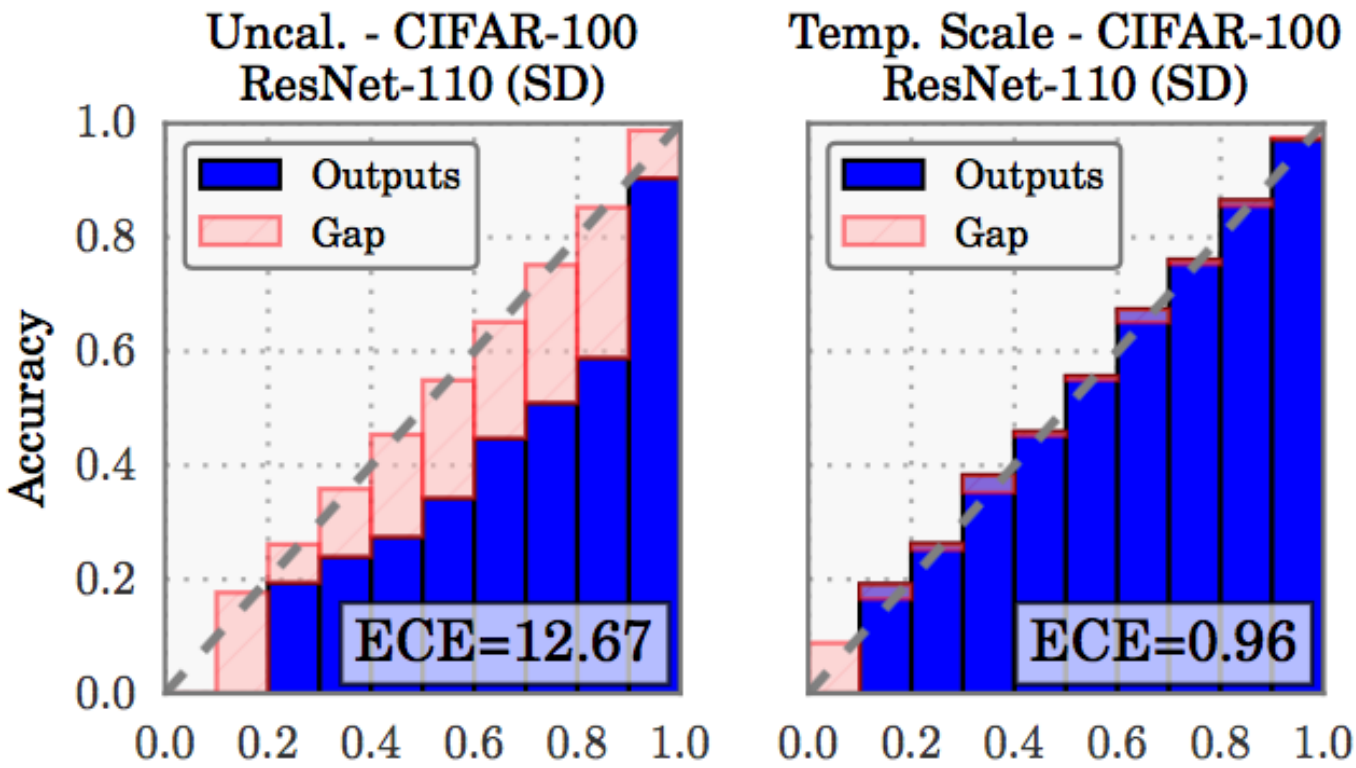
  and $\mathbf{z}$ are called the logits.
- They replace this with

$$\sigma(\mathbf{z}/T),$$

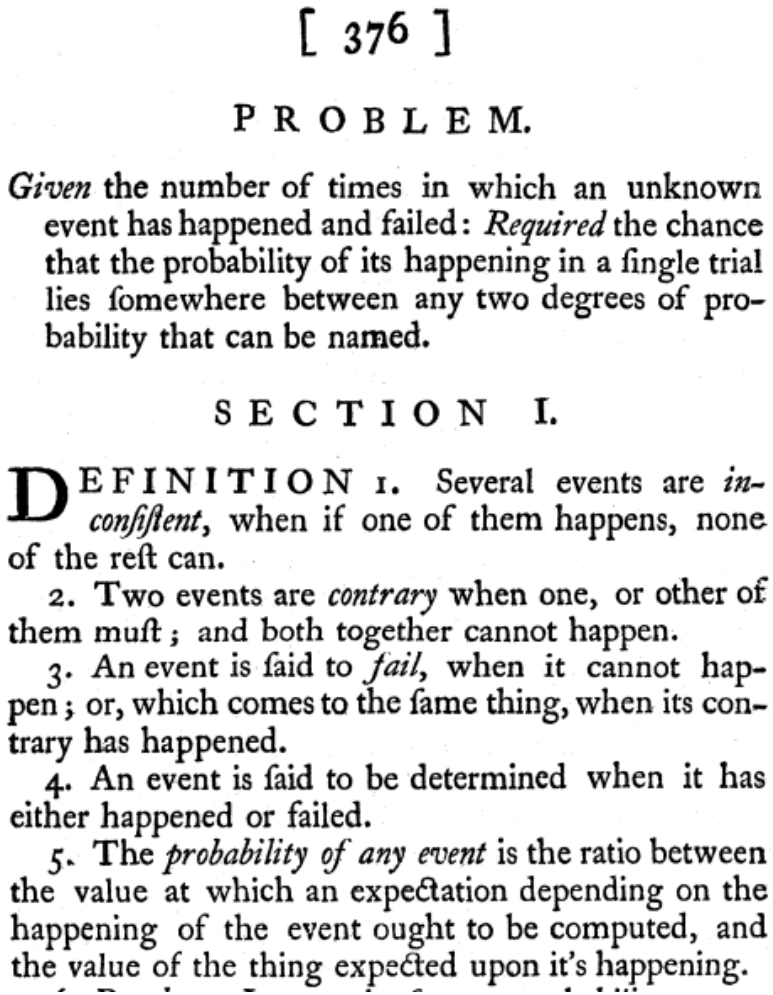
  where T is a scalar called the temperature.
- T is tuned to minimize the NLL on a validation set.
- Intuitively, because NLL is a proper scoring rule, the algorithm is incentivized to match the true probabilities as closely as possible.

Calibration

- Before and after temperature scaling:



A Toy Example



[376]

PROBLEM.

Given the number of times in which an unknown event has happened and failed: *Required* the chance that the probability of its happening in a ſingle trial lies ſomewhere between any two degrees of probability that can be named.

SECTION I.

DEFINITION 1. Several events are *inconſiſtent*, when if one of them happens, none of the reſt can.

2. Two events are *contrary* when one, or other of them muſt; and both together cannot happen.

3. An event is ſaid to *fail*, when it cannot happen; or, which comes to the ſame thing, when its contrary has happened.

4. An event is ſaid to be determined when it has either happened or failed.

5. The *probability of any event* is the ratio between the value at which an expectation depending on the happening of the event ought to be computed, and the value of the thing expected upon it's happening.

Thomas Bayes, "An Essay towards Solving a Problem in the Doctrine of Chances." Philosophical Transactions of the Royal Society, 1763.

A Toy Example

- Motivating example: estimating the parameter of a biased coin
  - You flip a coin 100 times. It lands heads $N_H = 55$ times and tails $N_T = 45$ times.
  - What is the probability it will come up heads if we flip again?
- Model: observations x_i are independent and identically distributed (i.i.d.) Bernoulli random variables with parameter θ.
- The likelihood function is the probability of the observed data (the entire sequence of H's and T's) as a function of θ:

$$\begin{aligned} L(\theta) = p(\mathcal{D}) &= \prod_{i=1}^{N} \theta^{x_i}(1-\theta)^{1-x_i} \\ &= \theta^{N_H}(1-\theta)^{N_T} \end{aligned}$$

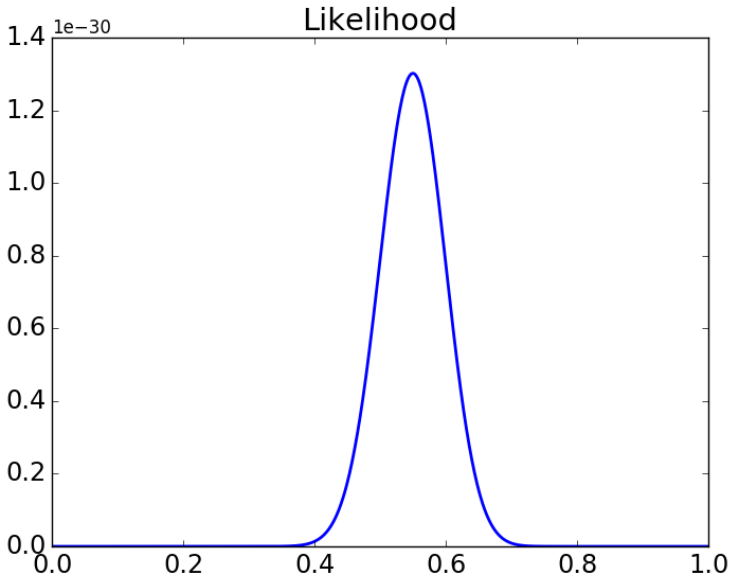
- N_H and N_T are sufficient statistics.

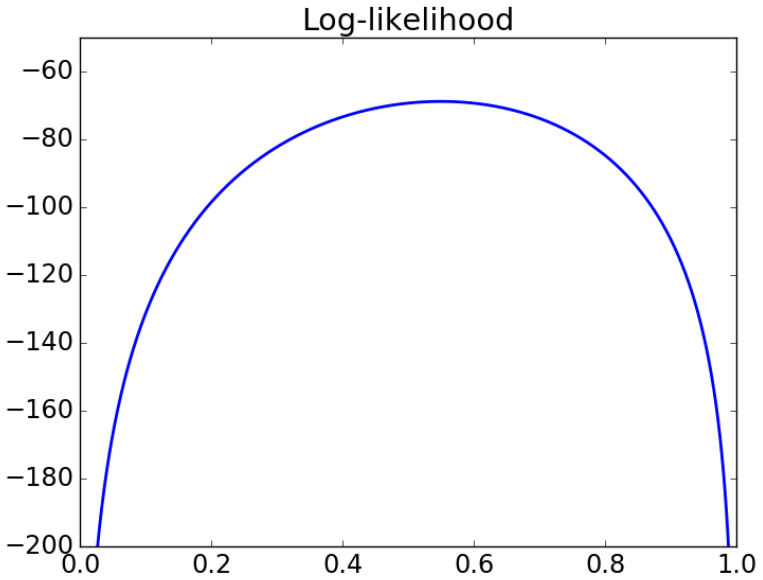
A Toy Example

- The likelihood is generally very small, so it's often convenient to work with log-likelihoods.

$$L(\theta) = \theta^{N_H}(1-\theta)^{N_T} \approx 7.9 \times 10^{-31}$$

$$\ell(\theta) = \log L(\theta) = N_H \log \theta + N_T \log(1-\theta) \approx -69.31$$





A Toy Example

- Good values of θ should assign high probability to the observed data. This motivates the maximum likelihood criterion.
- Solve by setting derivatives to zero:

$$\begin{aligned}\frac{\mathrm{d}\ell}{\mathrm{d}\theta} &= \frac{\mathrm{d}}{\mathrm{d}\theta}\left(N_H \log\theta + N_T \log(1-\theta)\right) \\ &= \frac{N_H}{\theta} - \frac{N_T}{1-\theta}\end{aligned}$$

- Setting this to zero gives the maximum likelihood estimate:

$$\hat{\theta}_{\mathrm{ML}} = \frac{N_H}{N_H + N_T},$$

- Normally there's no analytic solution, and we need to solve an optimization problem (e.g. using gradient descent).

A Toy Example

- Maximum likelihood has a pitfall: if you have too little data, it can overfit.
- E.g., what if you flip the coin twice and get H both times?

$$\theta_{\mathrm{ML}} = \frac{N_H}{N_H + N_T} = \frac{2}{2+0} = 1$$

- Because it never observed T, it assigns this outcome probability 0. This problem is known as data sparsity.
- If you observe a single T in the test set, the likelihood is $-\infty$.

A Toy Example

- In maximum likelihood, the observations are treated as random variables, but the parameters are not.
- The Bayesian approach treats the parameters as random variables as well.
- To define a Bayesian model, we need to specify two distributions:
  - The prior distribution $p(\boldsymbol{\theta})$, which encodes our beliefs about the parameters *before* we observe the data
  - The likelihood $p(\mathcal{D} \mid \boldsymbol{\theta})$, same as in maximum likelihood
- When we update our beliefs based on the observations, we compute the posterior distribution using Bayes' Rule:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\boldsymbol{\theta})p(\mathcal{D} \mid \boldsymbol{\theta})}{\int p(\boldsymbol{\theta}')p(\mathcal{D} \mid \boldsymbol{\theta}') \, \mathrm{d}\boldsymbol{\theta}'}.$$

- We rarely ever compute the denominator explicitly.

A Toy Example

- Let's revisit the coin example. We already know the likelihood:

$$L(\theta) = p(\mathcal{D}) = \theta^{N_H}(1-\theta)^{N_T}$$

- It remains to specify the prior $p(\theta)$.
  - We can choose an uninformative prior, which assumes as little as possible. A reasonable choice is the uniform prior.
  - But our experience tells us 0.5 is more likely than 0.99. One particularly useful prior that lets us specify this is the beta distribution:

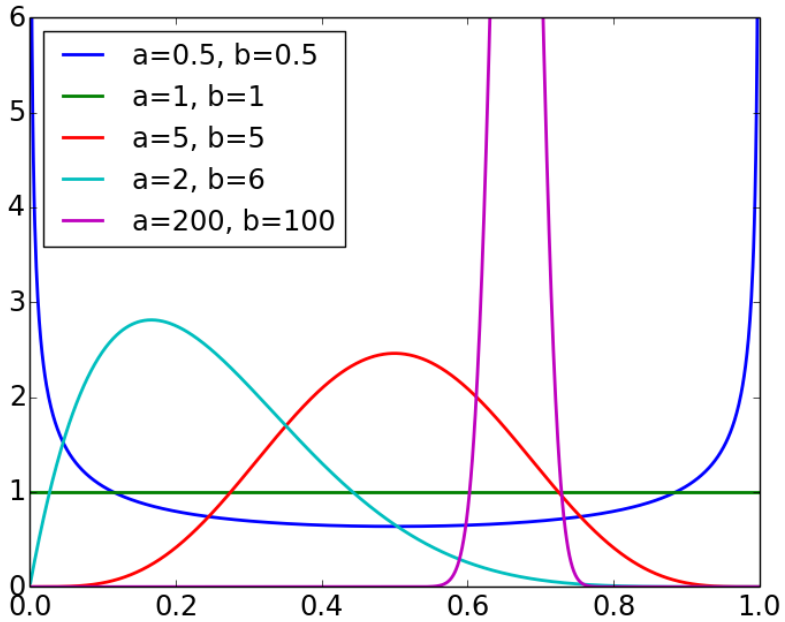
$$p(\theta; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\,\theta^{a-1}(1-\theta)^{b-1}.$$

  - This notation for proportionality lets us ignore the normalization constant:

$$p(\theta; a, b) \propto \theta^{a-1}(1-\theta)^{b-1}.$$

A Toy Example

- Beta distribution for various values of a, b:



- Some observations:
  - The expectation $\mathbb{E}[\theta] = a/(a+b)$.
  - The distribution gets more peaked when a and b are large.
  - The uniform distribution is the special case where $a = b = 1$.
- The main thing the beta distribution is used for is as a prior for the Bernoulli distribution.

A Toy Example

- Computing the posterior distribution:

$$\begin{aligned} p(\boldsymbol{\theta} \mid \mathcal{D}) &\propto p(\boldsymbol{\theta}) p(\mathcal{D} \mid \boldsymbol{\theta}) \\ &\propto \left[\theta^{a-1}(1-\theta)^{b-1}\right]\left[\theta^{N_H}(1-\theta)^{N_T}\right] \\ &= \theta^{a-1+N_H}(1-\theta)^{b-1+N_T}. \end{aligned}$$

- This is just a beta distribution with parameters $N_H + a$ and $N_T + b$.
- The posterior expectation of θ is:

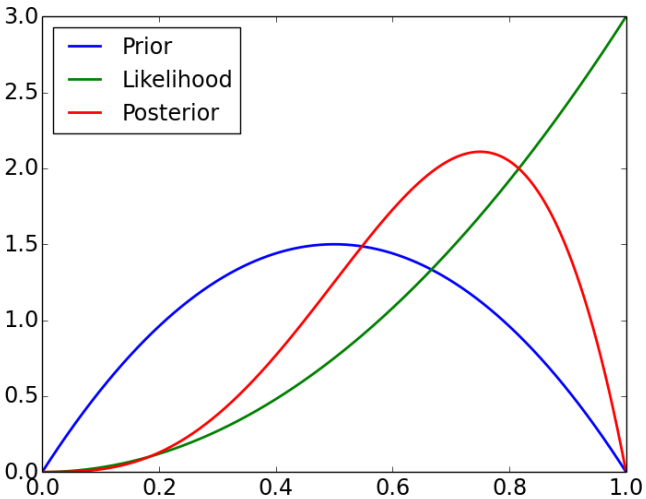
$$\mathbb{E}[\theta \mid \mathcal{D}] = \frac{N_H + a}{N_H + N_T + a + b}$$

- The parameters a and b of the prior can be thought of as pseudo-counts.
  - The reason this works is that the prior and likelihood have the same functional form. This phenomenon is known as conjugacy, and it's very useful.

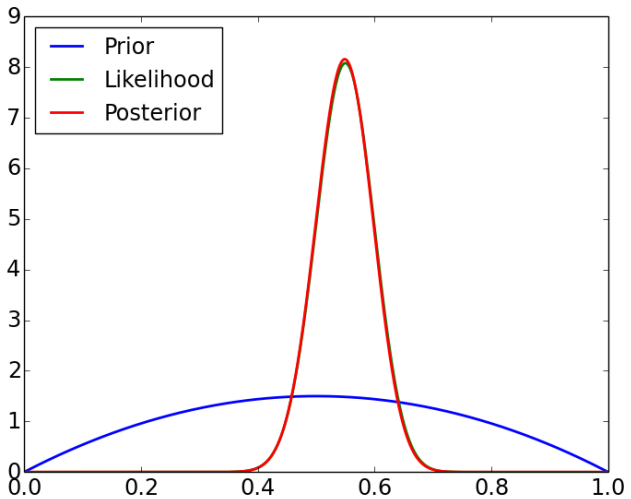
A Toy Example

Bayesian inference for the coin flip example:

Small data setting
$N_H = 2,\ N_T = 0$



Large data setting
$N_H = 55,\ N_T = 45$



When you have enough observations, the data overwhelm the prior.

A Toy Example

- What do we actually do with the posterior?
- The posterior predictive distribution is the distribution over future observables given the past observations. We compute this by marginalizing out the parameter(s):

$$p(\mathcal{D}' \mid \mathcal{D}) = \int p(\boldsymbol{\theta} \mid \mathcal{D}) p(\mathcal{D}' \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}. \tag{1}$$

- For the coin flip example:

$$\begin{aligned} \theta_{\text{pred}} &= \Pr(\mathbf{x}' = H \mid \mathcal{D}) \\ &= \int p(\theta \mid \mathcal{D}) \Pr(\mathbf{x}' = H \mid \theta) \, \mathrm{d}\theta \\ &= \int \mathrm{Beta}(\theta; N_H + a, N_T + b) \cdot \theta \, \mathrm{d}\theta \\ &= \mathbb{E}_{\mathrm{Beta}(\theta; N_H + a, N_T + b)}[\theta] \\ &= \frac{N_H + a}{N_H + N_T + a + b}, \end{aligned} \tag{2}$$

A Toy Example

- Maximum a-posteriori (MAP) estimation: find the most likely parameter settings under the posterior
- This converts the Bayesian parameter estimation problem into a maximization problem

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta} \mid \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta}, \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta})\, p(\mathcal{D} \mid \boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \; \log p(\boldsymbol{\theta}) + \log p(\mathcal{D} \mid \boldsymbol{\theta})
\end{aligned}$$

A Toy Example

- Joint probability in the coin flip example:

$$\begin{aligned}\log p(\theta, \mathcal{D}) &= \log p(\theta) + \log p(\mathcal{D} \mid \theta) \\ &= \text{const} + (a-1)\log\theta + (b-1)\log(1-\theta) + N_H \log\theta + N_T \log(1-\theta) \\ &= \text{const} + (N_H + a - 1)\log\theta + (N_T + b - 1)\log(1-\theta)\end{aligned}$$

- Maximize by finding a critical point

$$0 = \frac{\mathrm{d}}{\mathrm{d}\theta} \log p(\theta, \mathcal{D}) = \frac{N_H + a - 1}{\theta} - \frac{N_T + b - 1}{1 - \theta}$$

- Solving for θ,

$$\hat{\theta}_{\text{MAP}} = \frac{N_H + a - 1}{N_H + N_T + a + b - 2}$$

A Toy Example

Comparison of estimates in the coin flip example:

	Formula	$N_H = 2, N_T = 0$	$N_H = 55, N_T = 45$
$\hat{\theta}_{\text{ML}}$	$\frac{N_H}{N_H+N_T}$	1	$\frac{55}{100} = 0.55$
θ_{pred}	$\frac{N_H+a}{N_H+N_T+a+b}$	$\frac{4}{6} \approx 0.67$	$\frac{57}{104} \approx 0.548$
$\hat{\theta}_{\text{MAP}}$	$\frac{N_H+a-1}{N_H+N_T+a+b-2}$	$\frac{3}{4} = 0.75$	$\frac{56}{102} \approx 0.549$

$\hat{\theta}_{\text{MAP}}$ assigns nonzero probabilities as long as $a, b > 1$.

A Toy Example

- Lessons learned
  - Bayesian parameter estimation is more robust to data sparsity.
    - Not the most spectacular selling point. But stay tuned.

A Toy Example

- Lessons learned
  - Bayesian parameter estimation is more robust to data sparsity.
    - Not the most spectacular selling point. But stay tuned.
  - Maximum likelihood is about optimization, while Bayesian parameter estimation is about integration.
    - Which one is easier?

A Toy Example

- Lessons learned
  - Bayesian parameter estimation is more robust to data sparsity.
    - Not the most spectacular selling point. But stay tuned.
  - Maximum likelihood is about optimization, while Bayesian parameter estimation is about integration.
    - Which one is easier?
  - It's not (just) about priors.
    - The Bayesian solution with a uniform prior is robust to data sparsity. Why?

A Toy Example

- Lessons learned
  - Bayesian parameter estimation is more robust to data sparsity.
    - Not the most spectacular selling point. But stay tuned.
  - Maximum likelihood is about optimization, while Bayesian parameter estimation is about integration.
    - Which one is easier?
  - It's not (just) about priors.
    - The Bayesian solution with a uniform prior is robust to data sparsity. Why?
  - The Bayesian solution converges to the maximum likelihood solution as you observe more data.
    - Does this mean Bayesian methods are only useful on small datasets?

Preview: Bandits

- Despite its simplicity, the coin flip example is already useful.
- n-armed bandit problem: you have n slot machine arms in front of you, and each one pays out \$1 with an unknown probability θ_i. You get T tries, and you'd like to maximize your total winnings.

Preview: Bandits

- Despite its simplicity, the coin flip example is already useful.
- n-armed bandit problem: you have n slot machine arms in front of you, and each one pays out \$1 with an unknown probability θ_i. You get T tries, and you'd like to maximize your total winnings.
- Consider some possible strategies:
  - greedy: pick whichever one has paid out the most frequently so far

Preview: Bandits

- Despite its simplicity, the coin flip example is already useful.
- n-armed bandit problem: you have n slot machine arms in front of you, and each one pays out \$1 with an unknown probability θ_i. You get T tries, and you'd like to maximize your total winnings.
- Consider some possible strategies:
  - greedy: pick whichever one has paid out the most frequently so far
  - pick the arm whose parameter we are most uncertain about

Preview: Bandits

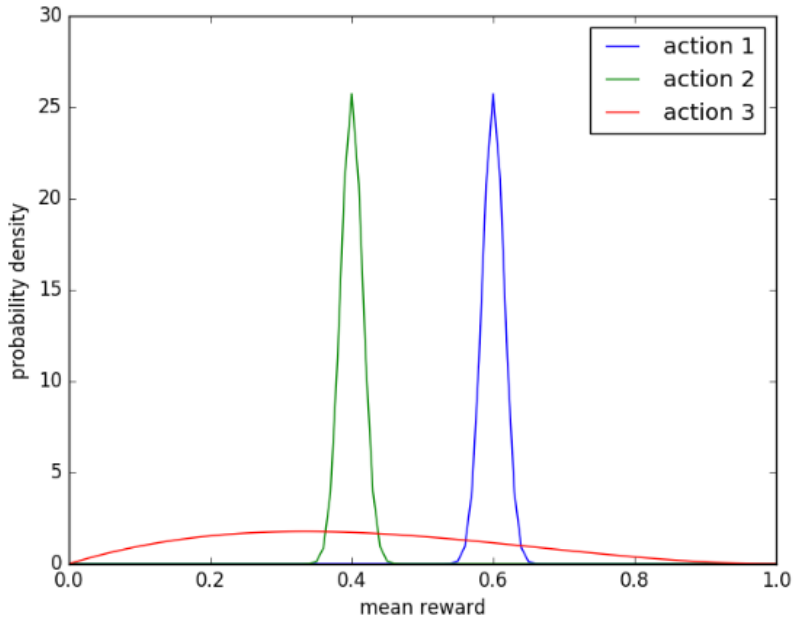
- Despite its simplicity, the coin flip example is already useful.
- n-armed bandit problem: you have n slot machine arms in front of you, and each one pays out \$1 with an unknown probability θ_i. You get T tries, and you'd like to maximize your total winnings.
- Consider some possible strategies:
  - greedy: pick whichever one has paid out the most frequently so far
  - pick the arm whose parameter we are most uncertain about
  - ε-greedy: do the greedy strategy with probability $1 - \varepsilon$, but pick a random arm with probability ε

Preview: Bandits

- Despite its simplicity, the coin flip example is already useful.
- n-armed bandit problem: you have n slot machine arms in front of you, and each one pays out \$1 with an unknown probability θ_i. You get T tries, and you'd like to maximize your total winnings.
- Consider some possible strategies:
  - greedy: pick whichever one has paid out the most frequently so far
  - pick the arm whose parameter we are most uncertain about
  - ε-greedy: do the greedy strategy with probability $1 - \varepsilon$, but pick a random arm with probability ε
- We'd like to balance exploration and exploitation.
  - Optimism in the face of uncertainty
  - Bandits are a good model of exploration/exploitation for more complex settings we'll cover in this course (e.g. Bayesian optimization, reinforcement learning)

Preview: Bandits

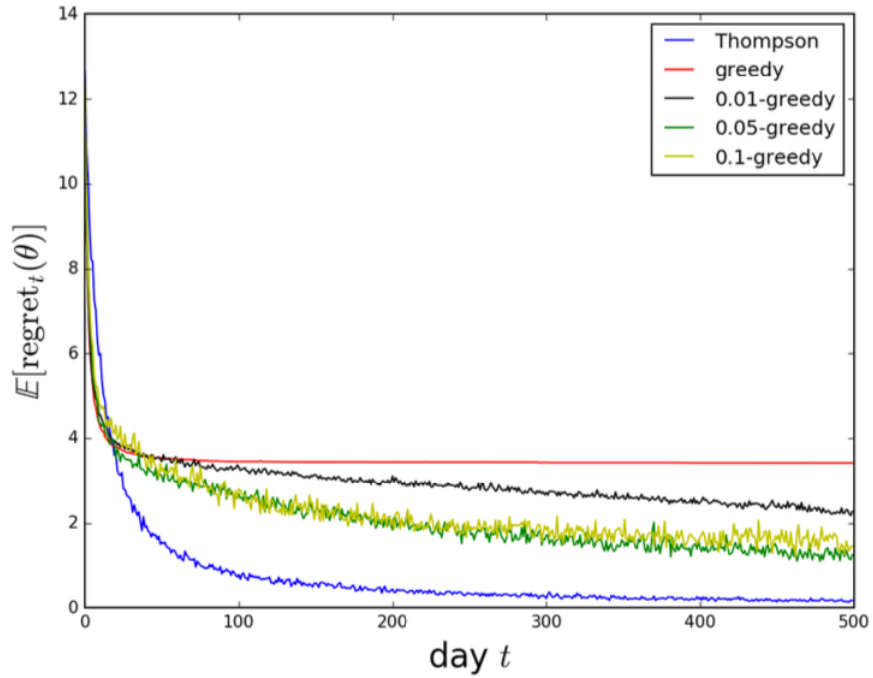
- One elegant solution: Thompson sampling (invented in 1933, ignored in AI until 1990s)
- Sample each $\theta_i \sim p(\theta_i \mid \mathcal{D})$, and pick the max.
- If these are the current posteriors over three arms, which one will it pick next?



— Russo et al., 2017, A tutorial on Thompson sampling

Preview: Bandits

- Why does this:
  - encourage exploration?
  - stop trying really bad actions?
  - emphasize exploration first and exploitation later?
- Comparison of exploration methods on a more structured bandit problem



(a) regret

— Russo et al., 2017, A tutorial on Thompson sampling