Language and Statistics II

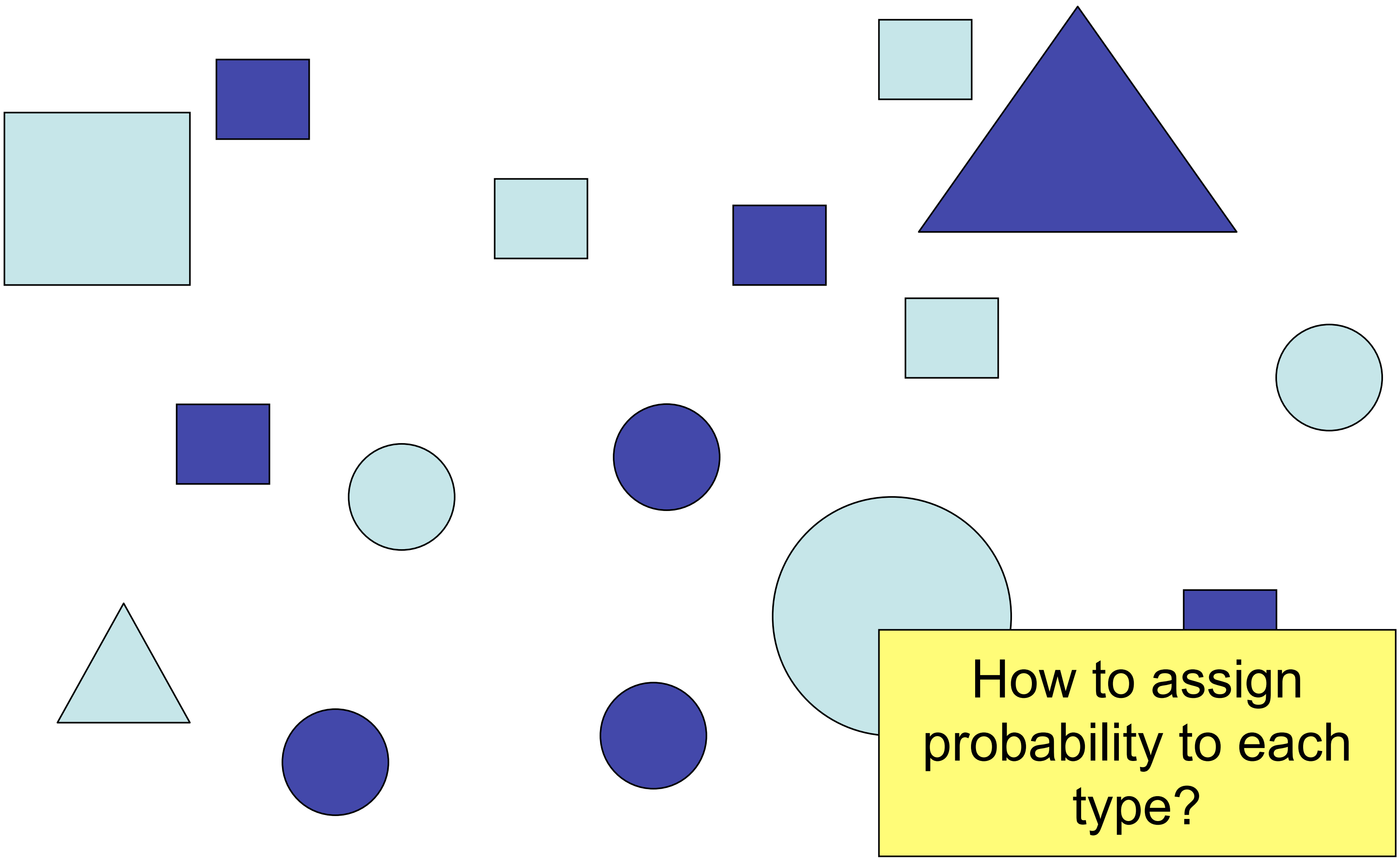
Lecture 5: Log-Linear Models
(The Details)

Noah Smith

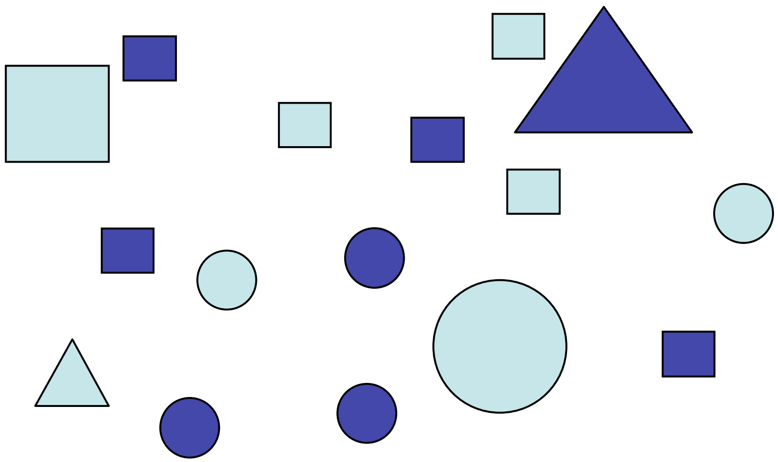
Today's Plan

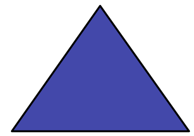
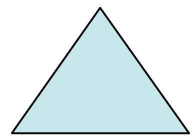
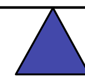
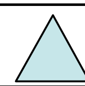
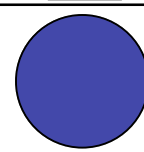
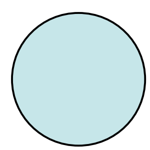
- (Anonymous) pop quiz
- Maximum Entropy modeling
- Relationship to log-linear models
- How to do it!
- Feature selection
- Regularization
- Conditional estimation

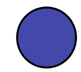
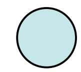
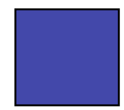
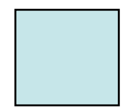
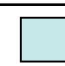
Data



Maximum Likelihood (Multinomial)



	1	.06
	0	0
	0	0
	1	.06
	0	0
	1	.06

	3	.19
	2	.12
	0	0
	1	.06
	4	.25
	3	.19

Overfitting?

11 df

Maximum Likelihood Estimation

- Given a model family, pick the parameters to maximize

  p(data | model)

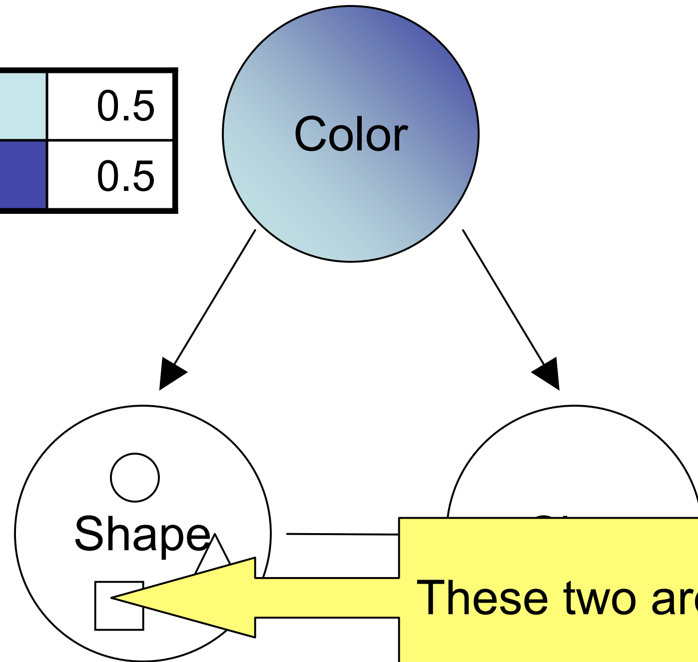
- Examples:
  - Gaussian: $\hat{\mu} = \bar{x}, \hat{\sigma} = \sqrt{\frac{\sum_i (x_i - \hat{\mu})^2}{n}}$
  - Bernoulli: $\hat{p} = \frac{n_{\text{success}}}{n}$
  - Multinomial: $\forall i, \hat{p}_i = \frac{n_i}{n}$
  - *n*-gram model?
  - HMM?

closed form solution

Using the Chain Rule

11 df

	0.5
	0.5



	△	0.125
	○	0.375
	□	0.500
	△	0.125
	○	0.375
	□	0.500

	△	large	0.000
		small	1.000
	○	large	0.333
		small	0.667
	□	large	0.250
		small	0.750
		large	1.000
		small	0.000
	○	large	0.000
		small	1.000
		large	0.000
		small	1.000

These two are the same!

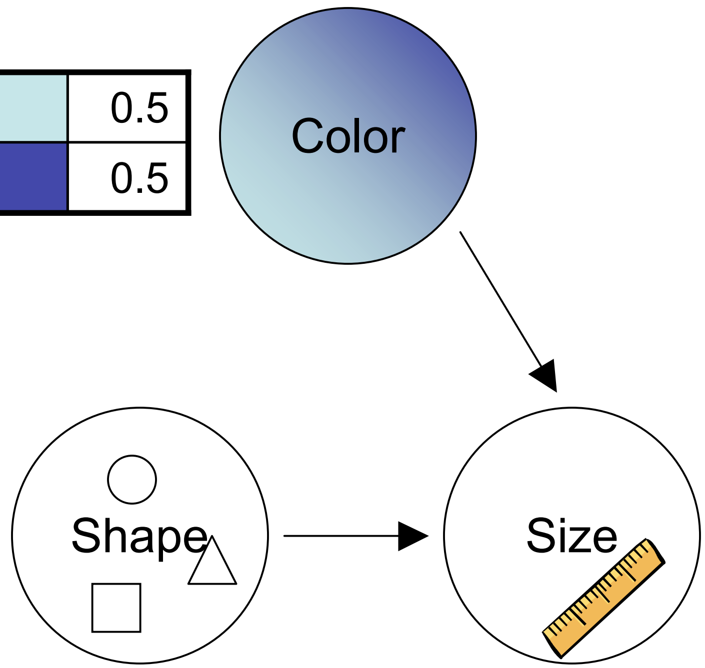
These two are the same!

Pr(Color, Shape, Size) = Pr(Color) • Pr(Shape | Color) • Pr(Size | Color, Shape)

Add an Independence Assumption?

9 df

	0.5
	0.5



△	0.125
○	0.375
□	0.500

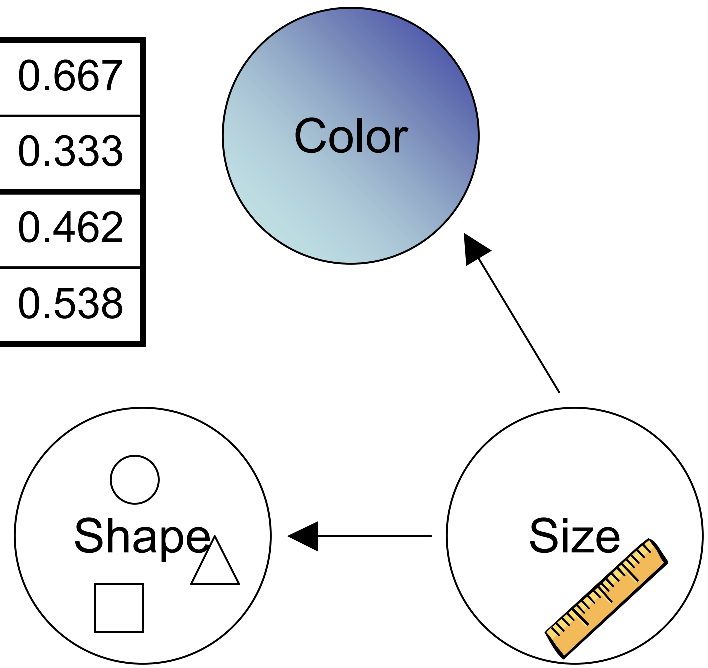
	△	large	0.000
		small	1.000
	○	large	0.333
		small	0.667
	□	large	0.250
		small	0.750
	△	large	1.000
		small	0.000
	○	large	0.000
		small	1.000
	□	large	0.000
		small	1.000

Pr(Color, Shape, Size) = Pr(Color) • Pr(Shape) • Pr(Size | Color, Shape)

Reverse Arrows?

7 df

large		0.667
		0.333
small		0.462
		0.538



large	0.375
small	0.625

large	△	0.333
	○	0.333
	□	0.333
small	△	0.077
	○	0.385
	□	0.538

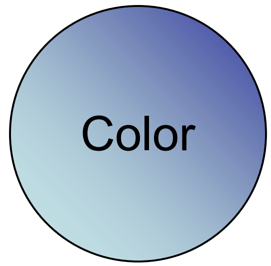
What's my prob.?

Pr(Color, Shape, Size) = Pr(Size) • Pr(Shape | Size) • Pr(Color | Size)

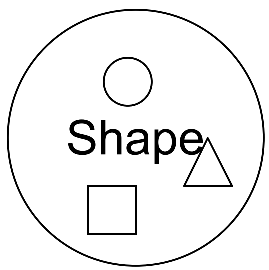
Strong Independence?

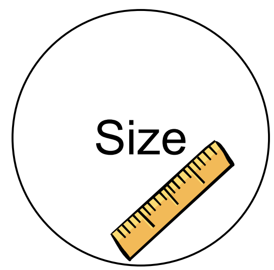
4 df

	0.5
	0.5



△	0.125
○	0.375
□	0.500





large	0.375
small	0.625

Pr(Color, Shape, Size) = Pr(Size) • Pr(Shape) • Pr(Color)

This Is Hard!

- Different **factorizations** affect
  - Model size (e.g., number of parameters or df)
  - Complexity of inference
  - “Interpretability”
  - Goodness of fit to the data
  - Generalization
  - Smoothing methods
- How would it change if we used **log-linear** models?
- Arguable: some major “innovations” in NLP involved really good choices about independence assumptions, directionality, and smoothing!

A Log-Linear Shape Model

How do we pick the features?

How do we set the weights?

$$p(\text{shape}) = \frac{\exp \sum_i f_i(\text{shape}) \cdot \theta_i}{Z(\vec{\theta})}$$

Desideratum: after we pick features, picking the weights should be the computer's job!

Some Intuitions

- Simpler models are better
  - (E.g., fewer degrees of freedom)
  - Why?
- Want to fit the data
- Don’t want to assume that an unobserved event has probability 0

Occam’s Razor

One should not increase, beyond what is necessary, the number of entities required to explain anything.

Uniform model

	△	○	□
small	0.083	0.083	0.083
small	0.083	0.083	0.083
large	0.083	0.083	0.083
large	0.083	0.083	0.083

Constraint: Pr(small) = 0.625

	△	○	□
small	0.104	0.104	0.104
small	0.104	0.104	0.104
large	0.063	0.063	0.063
large	0.063	0.063	0.063

} 0.625

Where did the constraint come from?

Pr(△, small) = 0.048

0.048

	△	○	□
small	0.024	0.144	0.144
small	0.024	0.144	0.144
large	0.063	0.063	0.063
large	0.063	0.063	0.063

0.625

Pr(large,) = 0.125

0.048

	△	○	□
small	0.024	0.144	0.144
small	0.024	0.144	0.144
large	0.063	0.063	0.063
large	0.063	0.063	0.063

0.625

?

Maximum Entropy

$$\max_p H(p) \equiv \max_p \sum_x -p(x)\log p(x)$$

subject to

$$\sum_x p(x) = 1, \quad \forall x, p(x) \geq 0$$

$$\forall j \in \{1,2,\ldots,m\}, \quad \mathbf{E}_p\left[f_j(X)\right] = \alpha_j$$

$$\sum_x p(x) f_j(x) = \alpha_j$$

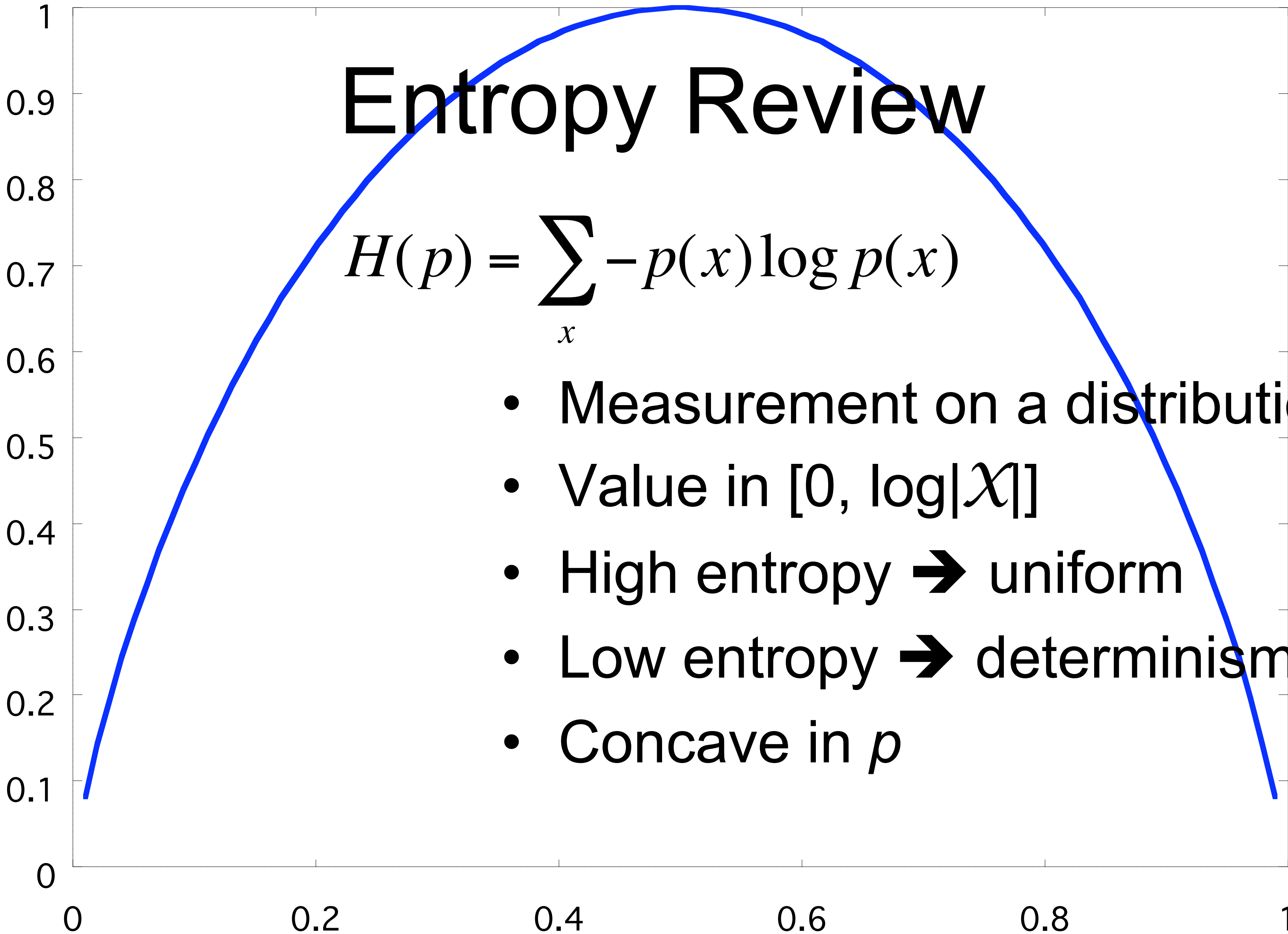
Questions Worth Asking

- Does a solution always exist?
  - What to do if it doesn’t?
- How to find the solution?

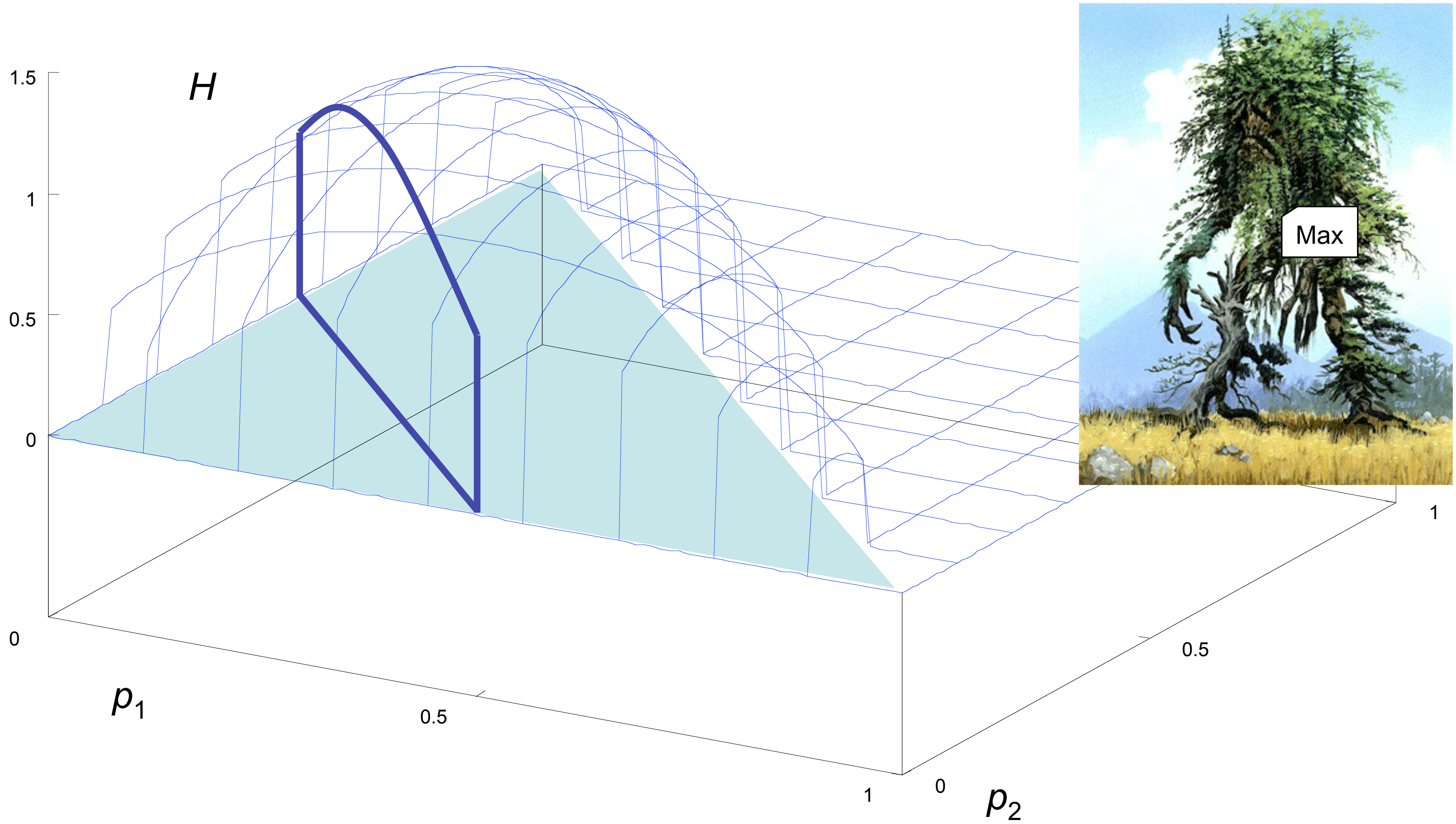
Entropy Review

$$H(p) = \sum_x -p(x) \log p(x)$$

- Measurement on a distribution
- Value in $[0, \log|\mathcal{X}|]$
- High entropy ➔ uniform
- Low entropy ➔ determinism
- Concave in *p*



Max Ent



Maximum Entropy

$$\max_p H(p) \equiv \max_p \sum_x -p(x)\log p(x)$$

subject to

$$\sum_x p(x) = 1, \quad \forall x, p(x) \geq 0$$

$$\forall j \in \{1,2,\ldots,m\}, \quad \mathbf{E}_p\left[f_j(X)\right] = \alpha_j$$

$$\sum_x p(x) f_j(x) = \alpha_j$$

Marginal Constraints

$$\sum_x p(x) f_j(x) = \alpha_j$$

$$\sum_x p(x) f_j(x) = \frac{1}{D} \sum_{i=1}^{D} f_j(\tilde{x}_i)$$

Example:

$$\sum_x p(x) \begin{cases} 1 \text{ if } x \text{ is square} \\ 0 \text{ otherwise} \end{cases} = \frac{1}{D} \sum_{i=1}^{D} \begin{cases} 1 \text{ if } \tilde{x}_i \text{ is square} \\ 0 \text{ otherwise} \end{cases} = \frac{\text{count}(\text{square})}{D}$$

Let $\mathcal{P}$ represent the set of distributions *p* that meet the constraints.

Claim 1

The unique solution to the maximum entropy problem

$$\arg\max_{p \in \mathcal{P}} H(p)$$

is a **log-linear** model on the **same** features as $\mathcal{P}$.

Claim 2

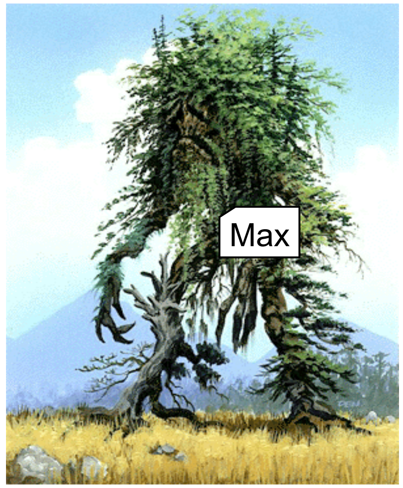
The unique solution to the maximum entropy problem

$$\arg\max_{p \in \mathcal{P}} H(p)$$

is **the** log-linear model on the **same** features as $\mathcal{P}$ that also solves

$$\arg\max_{p \in \text{Loglinear}} p\left(\vec{\tilde{x}}\right)$$

Mathematical Magic



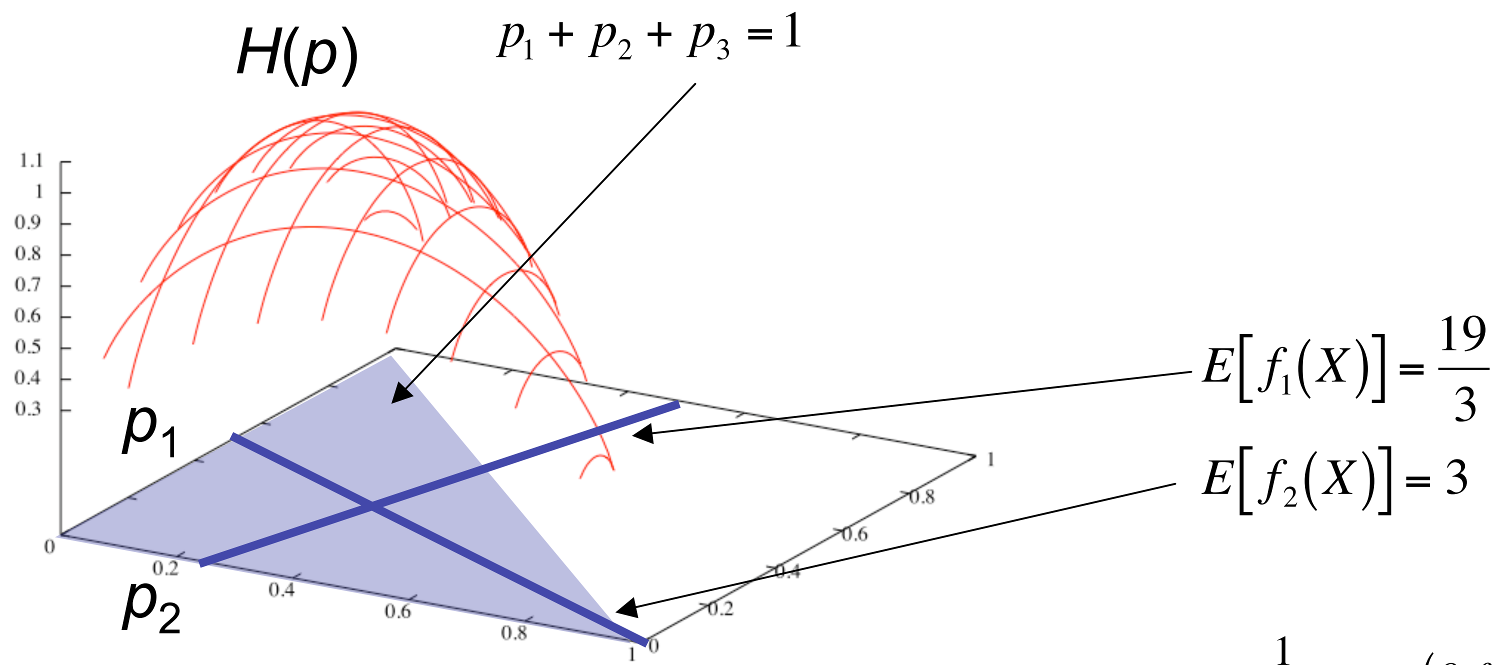
constrained
$|\mathcal{X}|$ variables (p)
concave in p

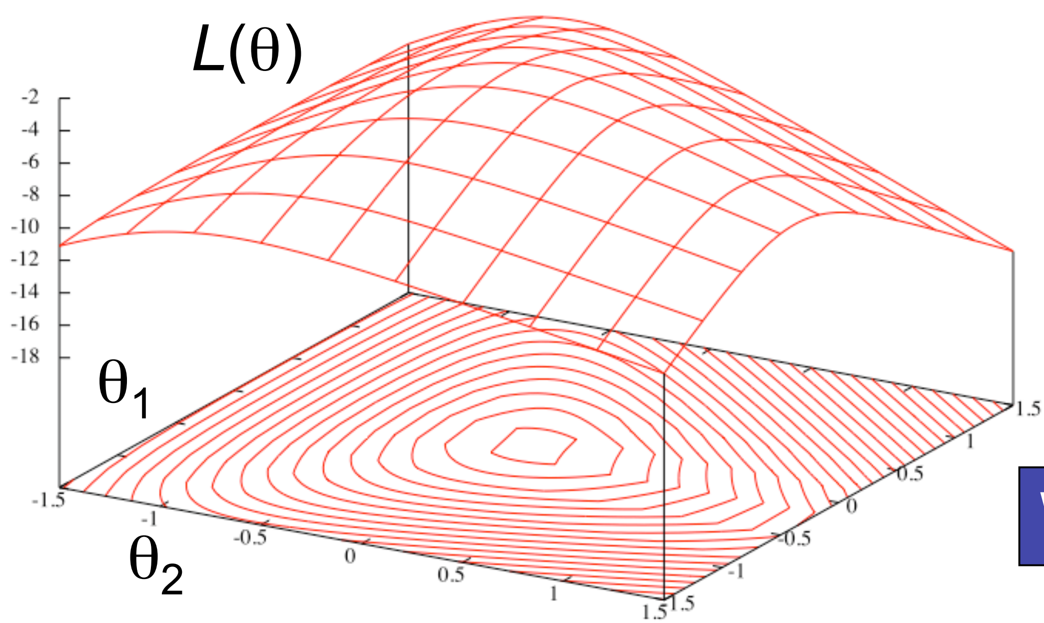
*un*constrained
m variables (θ)
concave in θ

Mathematical Magic

For details: see handout on course page.

1. Use Lagrangean multipliers (one per constraint).
2. Take the gradient, set equal to zero.
3. Algebra …
4. Voilà! Maximum likelihood problem!





$$p_1 = \frac{1}{Z(\theta_1,\theta_2)} \exp\left(\theta_1 f_1(x_1) + \theta_2 f_2(x_1)\right)$$

$$p_2 = \frac{1}{Z(\theta_1,\theta_2)} \exp\left(\theta_1 f_1(x_2) + \theta_2 f_2(x_2)\right)$$

$$p_3 = \frac{1}{Z(\theta_1,\theta_2)} \exp\left(\theta_1 f_1(x_3) + \theta_2 f_2(x_3)\right)$$

What if we took out f_2?

Additional Point

- If the constraints are empirical, then they are satisfiable (solution exists).

- So there is a **unique** solution to:

  Max Ent = Log-linear MLE

Slightly More General View

- Instead of “maximize entropy,” can describe this as “minimize divergence” to a **base** distribution *q* (which happens so far to be uniform, but needn’t have been).

$$D\left(p \| q\right) = \sum_{x} p\left(x\right) \log \frac{p(x)}{q(x)}$$

- Everything goes through pretty much the same.

Training the Weights

- Old answer: "iterative scaling"
  - Specialized method for this problem
  - Later versions: Generalized IS (Darroch and Ratliff, 1972) and Improved IS (Della Pietra, Della Pietra, and Lafferty, 1995)
- More recent answer:
  - It's unconstrained, convex optimization!
  - See Malouf (2002) for comparison.

Improved Iterative Scaling (Della Pietra et al., 1997)

- Initialize each θ_j arbitrarily.
- Let: $f_{\#}(x) = \sum_j f_j(x)$
- Repeat until convergence:
  - Solve for each δ_j: $$\sum_x \tilde{p}(x) f_j(x) = \sum_x \frac{\exp f(x) \cdot \vec{\theta}}{Z(\vec{\theta})} f_j(x) e^{\delta_j f_{\#}(x)}$$
  - Update: $$\theta_j \leftarrow \theta_j + \delta_j$$

Berger's IIS tutorial gives a derivation.

Gradient Ascent

- Initialize each θ_j arbitrarily.
- Repeat until convergence:
  - Line search for step size:

$$\hat{\alpha} \leftarrow \arg\max_{\alpha} f\left(\vec{\theta} + \alpha \nabla f\left(\vec{\theta}\right)\right)$$

  - Gradient step:

$$\vec{\theta} \leftarrow \vec{\theta} + \hat{\alpha} \nabla f\left(\vec{\theta}\right)$$

Quasi-Newton Methods

- Use the same information as gradient ascent: function value and gradient.
- Build up an approximate Hessian matrix (second derivatives) over time.
- Converge **much** faster.
- There are existing implentations: you provide a function that computes f and ∇f.

- (Could use true Hessian, but $n \times n$ second derivatives to compute!)

- Common examples: conjugate gradient, L-BFGS.

What are the Function and Gradient?

$$L(\theta) = \frac{1}{D} \sum_{j} \theta_j \sum_{i=1}^{D} f_j(\tilde{x}_i) - \log \underbrace{\sum_{x} \exp \sum_{j} f_j(x) \cdot \theta_j}_{Z(\vec{\theta})}$$

$$\frac{\partial L}{\partial \theta_j} = \frac{1}{D} \sum_{i=1}^{D} f_j(\tilde{x}_i) - \mathbf{E}_{p_{\vec{\theta}}(X)}\left[f_j(X) \right]$$

Should remind you of Max Ent constraints!