ETH zürich

Log-Linear Modeling (Meet the Softmax)

Ryan Cotterell, Lucas Torroba Hennigen and Niklas Stoehr

ETH zürich

Administrivia

Course Assignment / Project Update

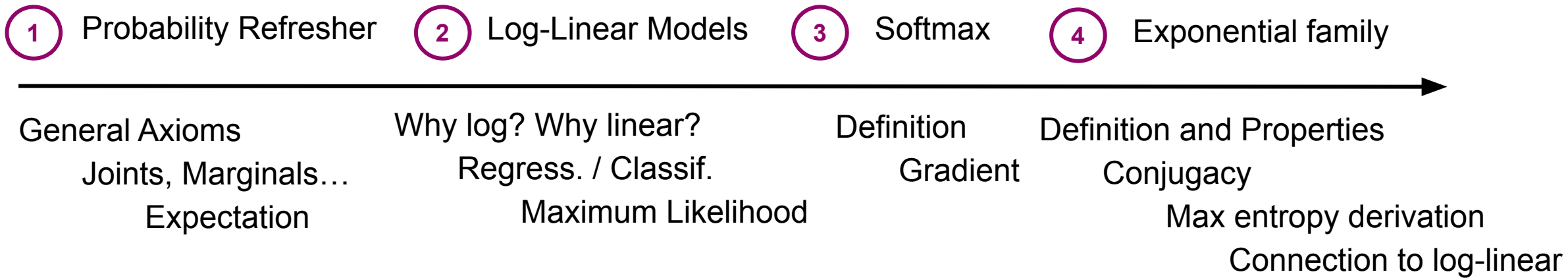
- **Course Assignment**
  - We are working on the course assignment as we speak! Stay tuned. It's gonna be a lot of fun :P
- **Course Project**
  - Please start forming your groups now. We are going to require that you submit a project proposal by **November 1st 2020**.
  - The proposal is 1 to 2 pages using the ACL sty file (https://www.overleaf.com/latex/templates/acl-2020-proceedings-template/zsrkcwjptpcd). It should explain what you plan to do and provide a brief literature overview
  - **The proposal is not graded**; its purpose is to get you early feedback to ensure you get a good grade.
  - Please post on Piazza if you are having trouble finding a group

Course Outline

- This is the second more technical lecture
  - The softmax, along with backpropagation, is a fundamental technique in modern NLP

- Starting next week, our lectures will be ½ NLP applications and ½ technical content
  - So, language is coming back!

Structure of this Lecture



Supplementary Material

Reading: Eisenstein Ch. 2

Supplementary Reading: Ferraro and Eisner (2013) Jason Eisner's list of further resources on log-linear modeling

ETH zürich

Probability Refresher

(Much of this class is about developing probabilistic models of language)

1 What is Probability?

- **The Frequentist Interpretation** considers the relative frequencies of events of interest to the total number of events that occurred. The probability of an event is defined as the relative frequency of the event in the limit when one has infinite data.
- **The Bayesian Interpretation** uses probability to specify the degree of uncertainty that the user has about an event. It is sometimes referred to as “subjective probability” or “degree of belief”.

- **A Random Variable X** is a function that maps outcomes of random experiments to a set of properties. (We will dig more into this later.)
- **A Probability Distribution p(X=x)** is a function that measures the probability that outcomes with the particular property *x* will occur

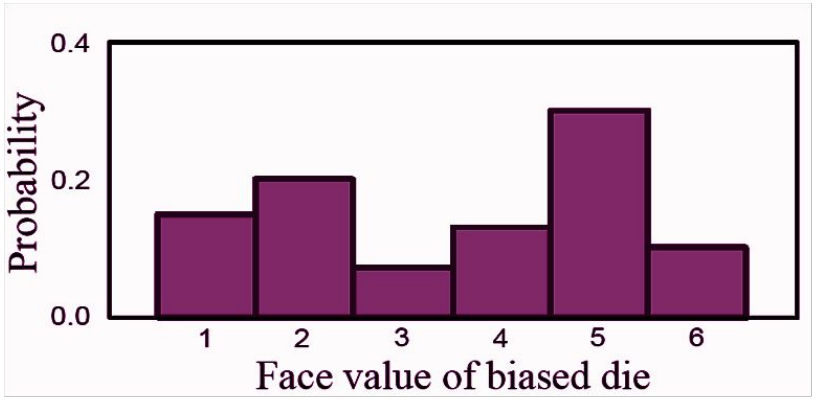
adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth Mathematics for Machine Learning

1 Probability Refresher

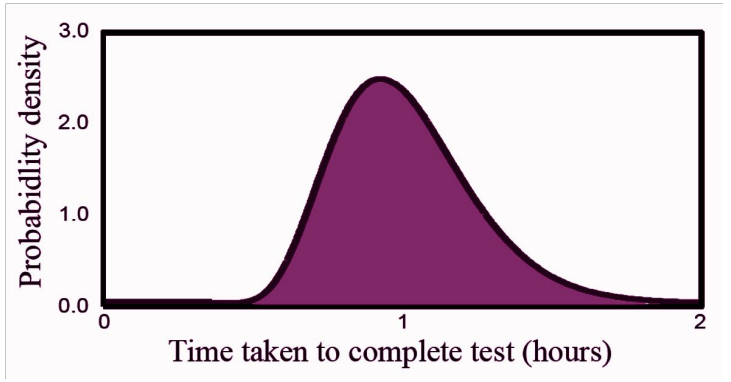
Probability Distributions

Type	"Point Probability"	"Interval Probability"
Discrete	p(X = x) (Probability Mass Function)	not applicable
Continuous	p(x) (Probability Density Function)	p(X≤x) (Cumulative Distribution Function)

Discrete



Continuous



adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

1 Axioms of Probability (in the Discrete Case)

- Let $\mathcal{X}$ be a countable set. A discrete probability distribution a must satisfies three axioms:

1. **Non-negativity:** $p(X = x) \geq 0$
2. **Sums to 1:** $\sum_{x \in \mathcal{X}} p(X = x) = 1$
3. **Countable Additivity:** $p(x \textbf{ or } y) = p(x) + p(y) - p(x \textbf{ and } y)$

- Note that 3 is basically always true for discrete probability distributions. Mostly important for continuous densities

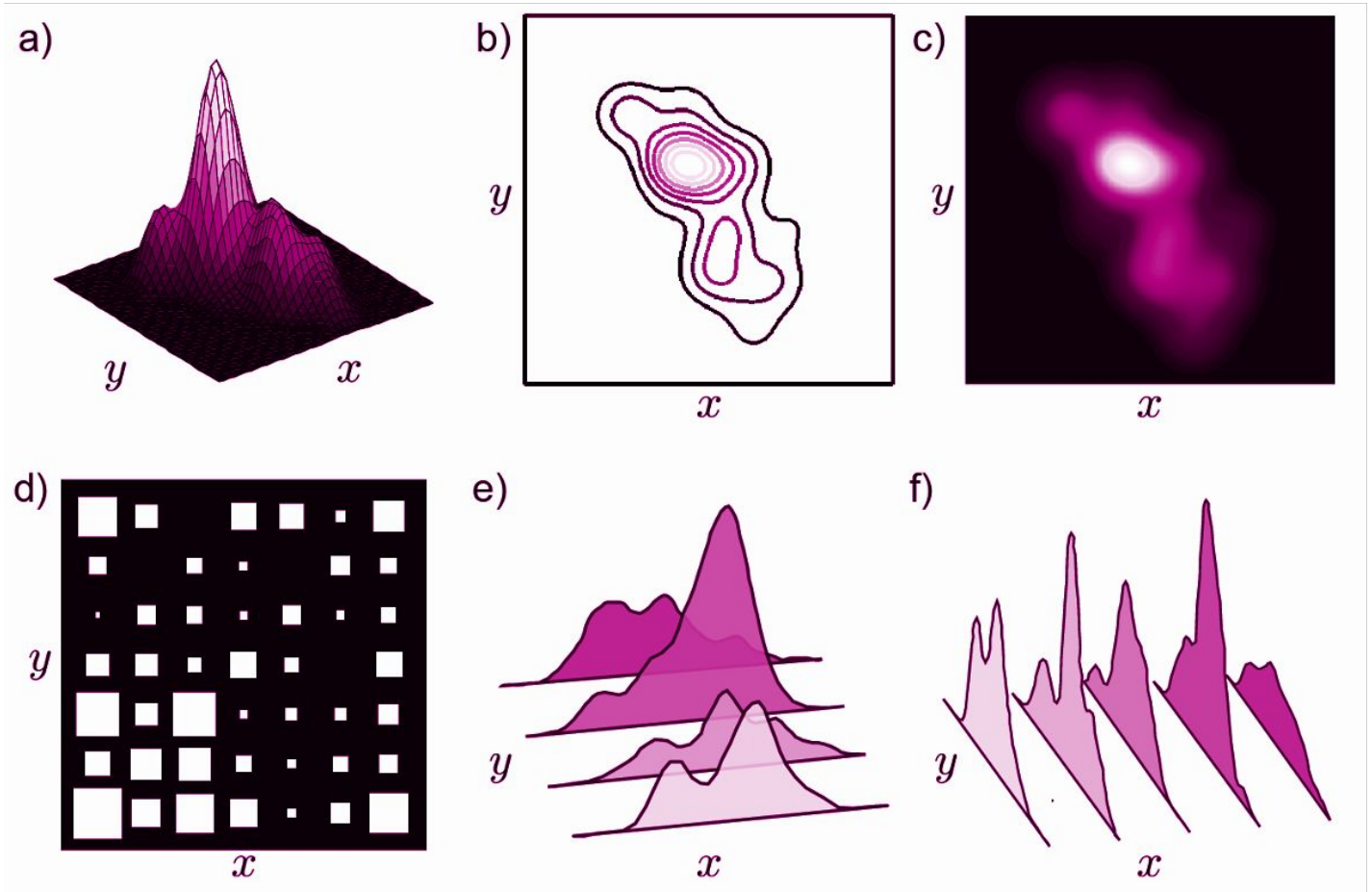
- Continuous case is very similar, but requires measure theory

1 Probability Refresher

Joint Probability

$$p(x, y)$$

- Consider two random variables x and y. If we observe multiple paired instances, then some combinations of outcomes are more likely than others



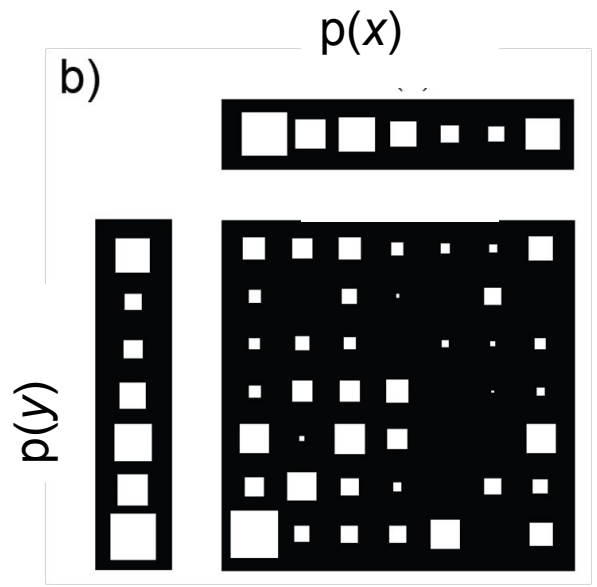
adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

1 Probability Refresher

Marginalization

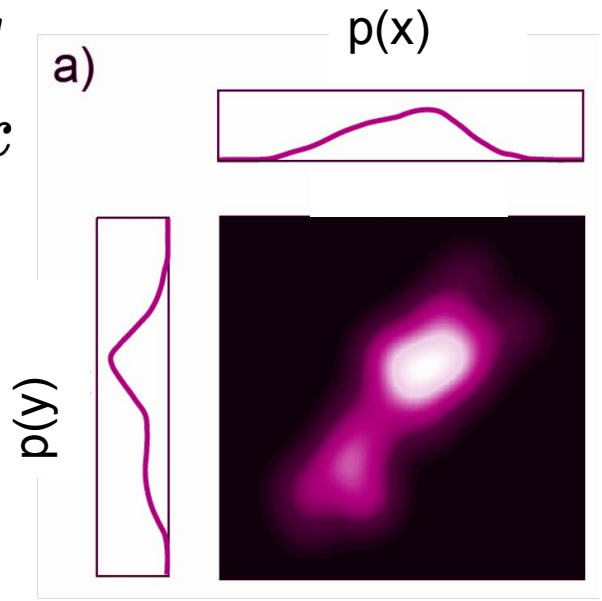
Discrete

$$p(x) = \sum_y p(x, y)$$
$$p(y) = \sum_x p(x, y)$$



Continuous

$$p(x) = \int p(x, y) \mathrm{d}y$$
$$p(y) = \int p(x, y) \mathrm{d}x$$



- We can recover the probability distribution of any variable in a joint distribution by integrating (or summing) over the other variables

adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

1 Probability Refresher

Conditional Probability

$$p(x \mid y) = \frac{p(x,y)}{p(y)}$$

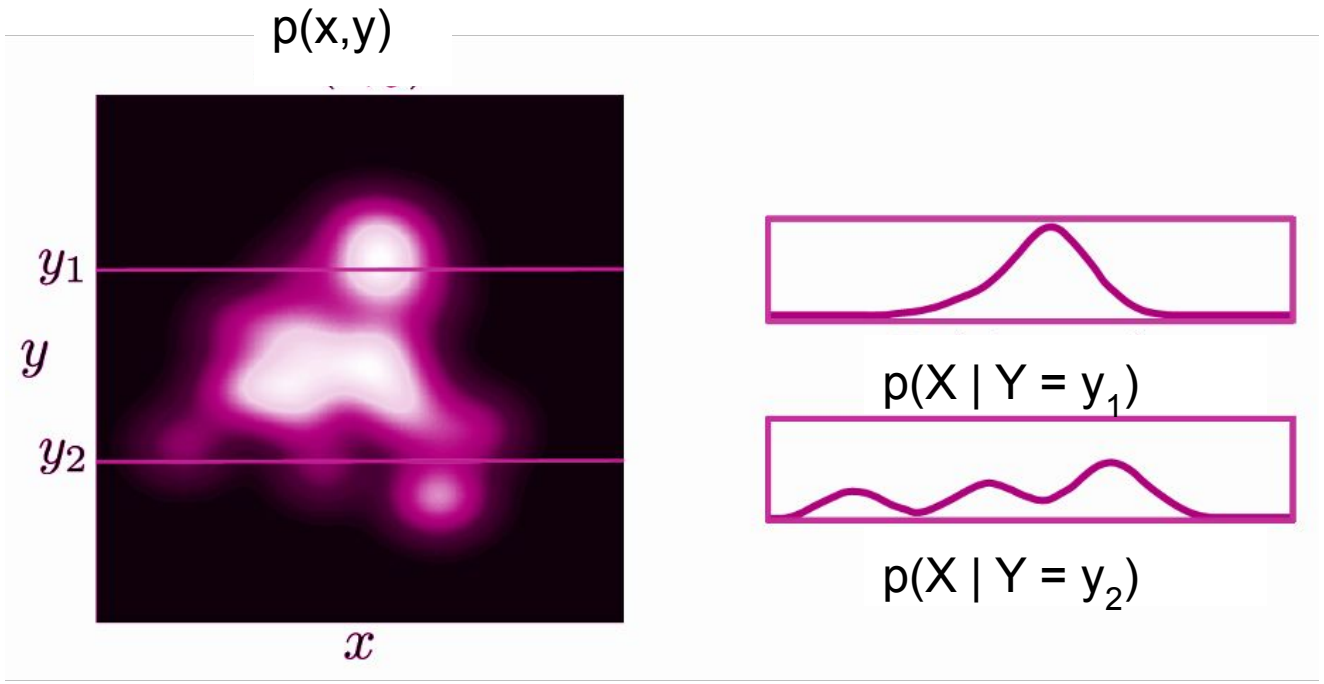
Bayes' Rule

Posterior

Likelihood Prior

$$p(y \mid x) = \frac{p(x|y)p(y)}{\int p(x|y)p(y)\mathrm{d}y}$$

Evidence



- Conditional probability of x given that Y=y_1

adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

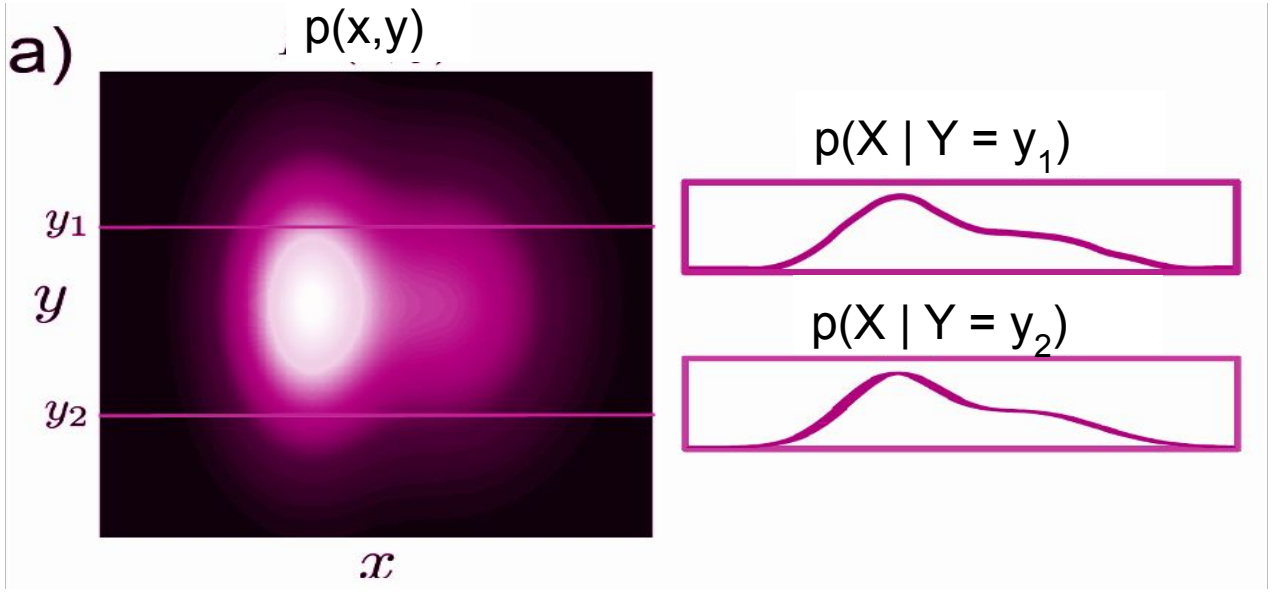
1 Probability Refresher

Independence

$$p(y \mid x) = p(y)$$

$$p(x \mid y) = p(x)$$

$$x \perp\!\!\!\perp y$$



- When variables are independent, the joint factorizes into a product of the marginals:

$$p(x, y) = p(x)p(y)$$

- If two variables x and y are independent then variable x tells us nothing about variable y (and vice-versa)

adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

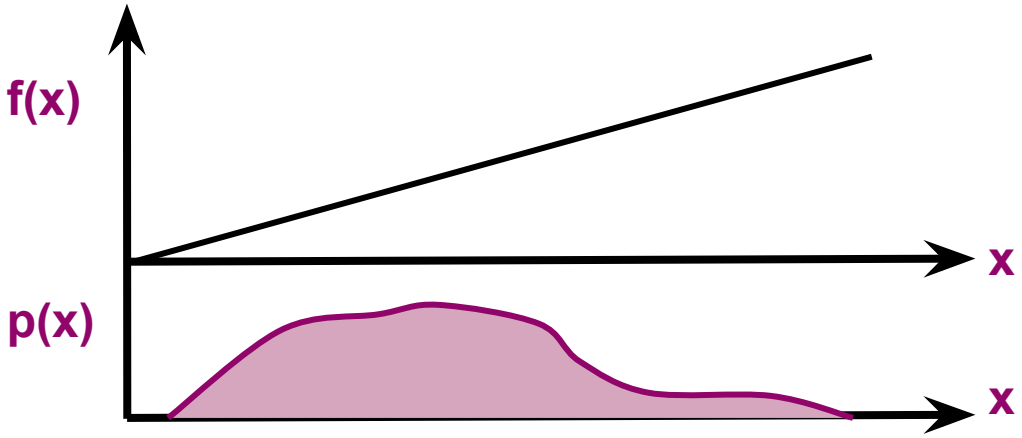
1 Probability Refresher

Expectation

- Expectation tells us the expected or average value of some function f[x], taking into account the distribution of x.

Discrete $E[f(x)] = \sum_x f(x)p(x)$

Continuous $E[f(x)] = \int f(x)p(x)\mathrm{d}x$



Function f[▪]	Expectation
x	mean μ_x
x^k	k^{th} moment about zero
$(x-\mu_x)^k$	k^{th} moment about mean
$(x-\mu_x)^2$	variance
$(x-\mu_x)^3$	skew
$(x-\mu_x)^4$	kurtosis
$(x-\mu_x)(y-\mu_y)$	covariance of x and y

adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth, Mathematics for Machine Learning and Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

1 Probability Refresher

Expectation

Rule 1 (Linearity): Expected value of a constant is the constant

$$E[k] = k$$

Rule 2 (Linearity): Expected value of constant times function = constant times expected value function

$$E[kf(x)] = kE[f(x)]$$

Rule 3 (Linearity): Expectation of sum of functions is sum of expectation of functions

$$E[f(x) + g(x)] = E[f(x)] + E[g(x)]$$

Rule 4: Expectation of product of functions in variables x and y is product of expectations of functions if x and y are independent.

$$E[f(x)g(x)] = E[f(x)]E[g(x)] \qquad \text{iff} \quad x \perp\!\!\!\perp y$$

1 What is a Probability Space?

- **Sample Space Ω**
  - The set of all possible outcomes of an experiment.
  - For example, two successive coin tosses have a sample space of {hh, tt, ht, th}, where "h" denotes "heads" and "t" denotes "tails".

- **Event Space A**
  - The space of potential results of the experiment. *E* is a set of subsets of Ω. In other words, it is a subset of the power set of Ω.
  - In the discrete case, we generally take *E* = powerset(Ω).

- **Probability function *p***
  - For each event $e \in E$, we associate a number $p(e) \in [0,1]$ that measures the probability or degree of belief that the outcome will occur.

adapted from A. Aldo Faisal, Cheng Soon Ong, and Marc Peter Deisenroth Mathematics for Machine Learning, page 175

1 Why do we need Probability Spaces?

- In the discrete case, it may seem like overkill to have a probability space
  - Why all the notation? Scary!

- The reason the machinery is necessary comes up in the continuous case where we can't simply define E to be the powerset of Ω

- In the continuous case, we need E to be a set of **measurable** subsets of Ω
  - It's okay if you don't know what measurable means! Just replace it with "nice" in your mind.
  - The Borel sets is one such set of measurable subsets

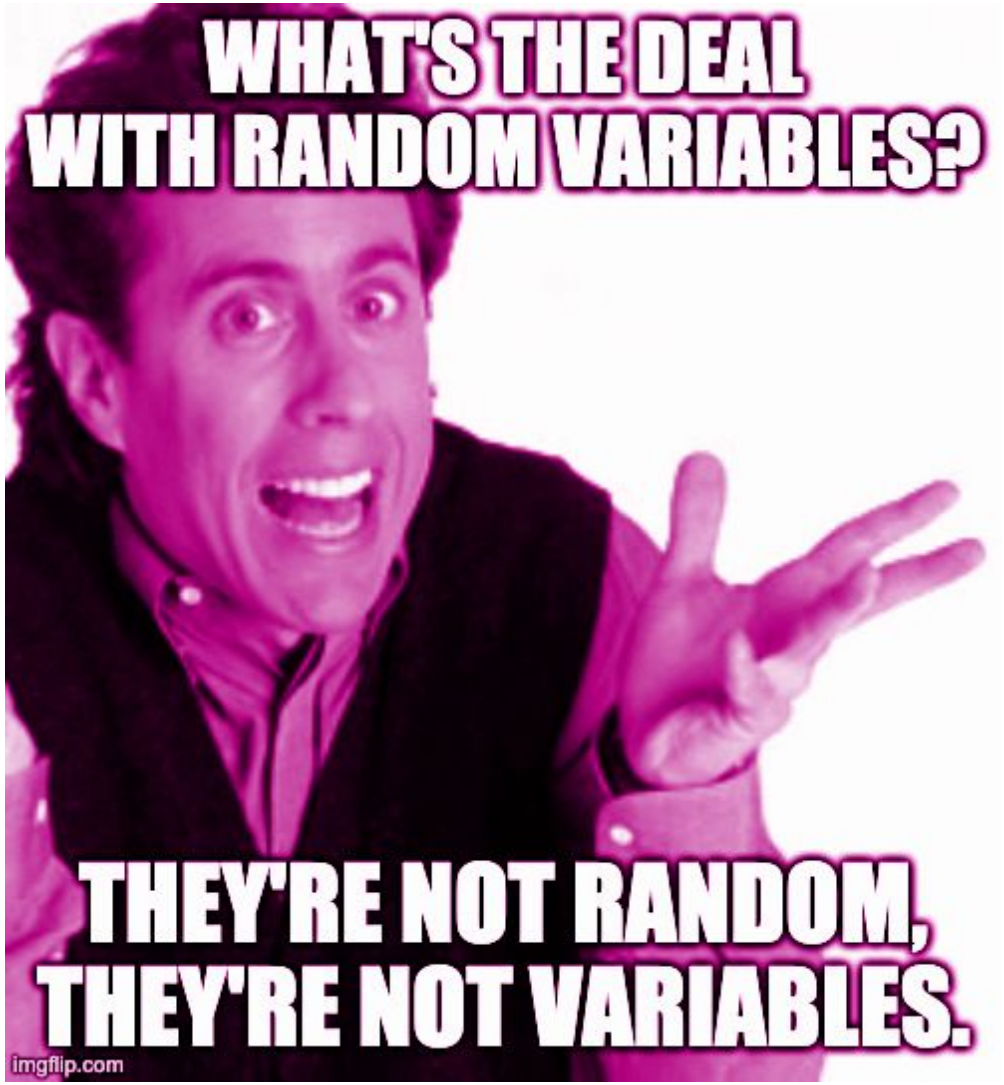
- Measure theory is not necessary for this course, so you can safely ignore it, but it's good to know why we have the added complexity

1 What is a Random Variable?

- Given a probability space (Ω, *E*, *p*), we may want to transform an outcome space Ω into a target space *T*
- We often avoid explicitly referring to the probability space Ω, but instead refer to probabilities on properties of interest. This is how we did it on slide 6
- **A Random Variable** is a function $X : \Omega \rightarrow T$ that takes an element of Ω (an outcome) and returns a value in *T*.
- **Example**: In the case of tossing two coins and counting the number of heads, a random variable *X* maps to the three possible outcomes: *X*(hh) = 2, *X*(ht) = 1, *X*(th) = 1, and *X*(tt) = 0, where we have a sample space Ω: {hh, tt, ht, th} and a target space *T*: {0, 1, 2}
- To preserve the structure of the event space *E*, *X* must be a **measurable function**
  - Measurable functions map measurable sets to measurable sets!

→ the name "random variable" is a great source of misunderstanding as it is neither random nor is it a variable. It is a function! (Functions are deterministic!)

1 What is a Random Variable?



1 Random Variable Example

- Let's define a probability space!
  - $\Omega = \{1, 2, 3, 4, 5, 6\}$
  - A = powerset(Ω); note that $|A| = 2^6$
  - Define $p(x) = \frac{1}{6}$ for every $x \in \Omega$

- What do the individual events in A mean?
  - Every element of A is a subset of powerset(Ω)
  - Consider $a = \{1, 5, 6\} \subseteq \Omega$
  - $p(a) = p(\{1, 5, 6\}) = p(1) + p(5) + p(6) = \frac{1}{2}$
    - What does it *mean*? The probability rolling a 1 or a 5 or a 6

- Let's define the even-roll random variable X.
  - $X(x)$ = "even" if x is even and $X(x)$ = "odd" if x is odd

- Now, $p(X = \text{"even"}) = \frac{1}{2}$ and $p(X = \text{"odd"}) = \frac{1}{2}$

- Random-variable notation makes it very easy to talk about properties of the roll

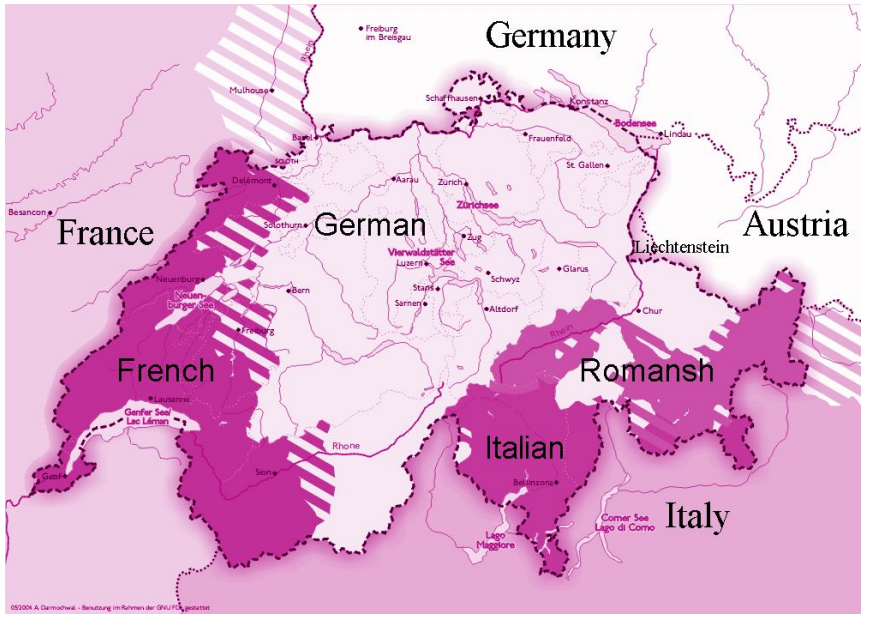
1 Why do we need Random Variables?

- Random variables are also scary! You didn't need a fancy measurable function to tell you the probability that a dice lands on an even number…
  - So, why bother with all the complexity?

- To see why, let's define a new probability space
  - Let Ω = set of all people in the world
  - Let E = powerset(Ω)
  - p(x) = 1/|Ω| for all $x \in \Omega$

- Define a Swiss German random variable
  - S(x) = **true** if x speaks Swiss German (**false** otherwise)

- Define a Romansch random variable
  - R(x) = **true** if x speaks Romansch (**false** otherwise)



1 Why do we need Random Variables?

- We have a single source of randomness (over Ω) but we want to ask questions:
  - p(x speaks Swiss German, x speaks Romasch) = p(S = **true**, R = **false**)
  - p(x speaks Swiss German if we know x speaks Romansch) = p(S = **true** | R = **true**)
  - p(x speaks Swiss German if we know x does not speak Romansch) = p(G = **true** | R = **false**)

- This would be very hard without random variables!
  - Random variables are fundamentally about interactions between different ***properties*** of elements of the sample space
  - Without random variables, it is hard to talk about independence and correlation, etc...
- **Remember:** independence and correlation are properties of random variables and not of the probability spaces

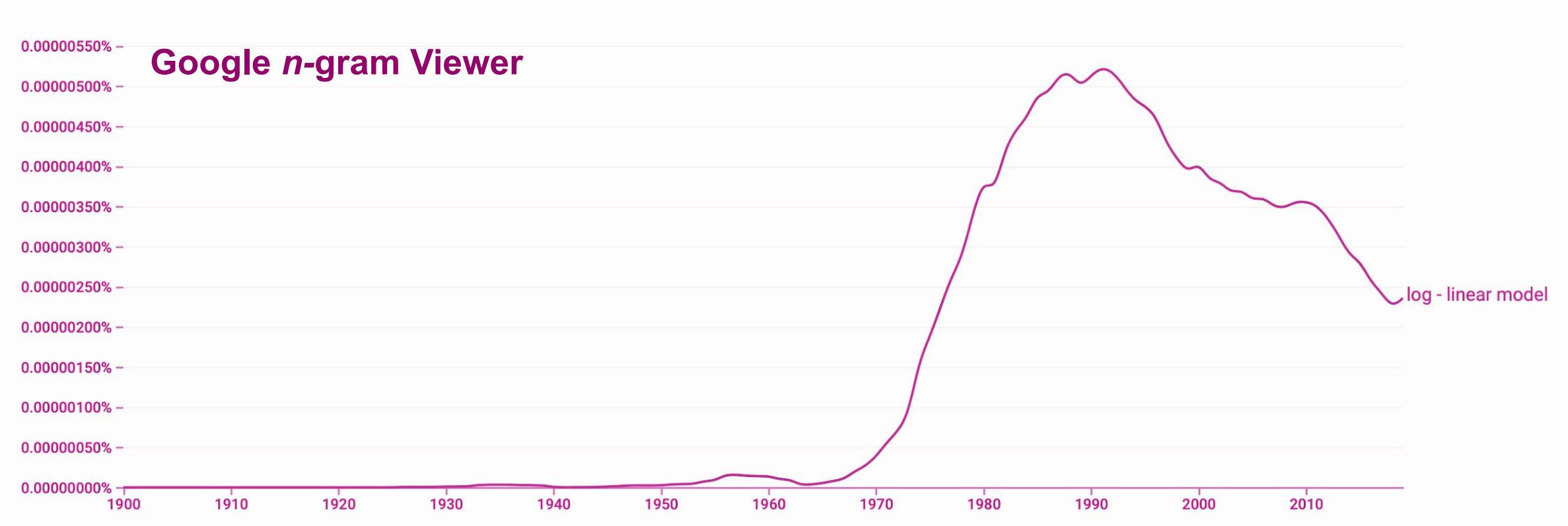
ETH zürich

Log-Linear Modeling

Log-linear modeling was the default way of doing probabilistic NLP from 1995 to 2015

② Why log-linear modeling?

Log-linear modeling was the default way of doing probabilistic NLP from 1995 to 2015



https://books.google.com/ngrams/graph?smoothing=3&corpus=26&year_end=2019&year_start=1900&content=log-linear+model&direct_url=t1%3B%2Clog%20-%20linear%20model%3B%2Cc0#t1%3B%2Clog%20-%20linear%20model%3B%2Cc0

2 Modeling Discrete Distributions

- Suppose we wish to model p(Y = y | X = x), i.e. the conditional probability that an event will turn out to be y when the context happens to be x.

Simple procedure: $$p(y \mid x) := \frac{\text{count}(x,y)}{\text{count}(x)}$$

- Is this a good procedure?

- **Important first step**: Throughout the course, we will later build model of more complicated items, e.g. sequences, tranductions and trees

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2
and Jason Eisner (Johns Hopkins University) https://www.cs.jhu.edu/~jason/tutorials/loglin/formulas.pdf

2 Modeling Discrete Distributions

X (car company)	Tesla	BMW	Toyota	BMW	Tesla	Tesla
Y (color)	black	white	red	black	white	black

- **Quick review:**
  - What's the outcome space?
    - Ω = {set of all cars}
    - These are car tokens not types so there can be duplicates
  - What's the event space?
    - E = powerset(Ω)
  - What's the probability function?
    - $p(x) = 1/|\Omega|, x \in \Omega$
  - What are the random variables?
    - $X : \Omega \rightarrow$ {set of companies}; X(x) = "Telsa" if x is made by Telsa, etc..
    - $Y : \Omega \rightarrow$ {set of colors}; X(x) = "red" if x is colored red, etc..

2 Modeling Discrete Distributions

X (car company)	Tesla	BMW	Toyota	BMW	Tesla	Tesla
Y (colour)	black	white	red	black	white	black

$$p(y \mid x) := \frac{\text{count}(x,y)}{\text{count}(x)} \quad \Rightarrow \quad p(Y = \text{"black"} \mid X = \text{"Tesla"}) = \frac{2}{3}$$

Estimating parameters: this model is a unique Categorical over X for every element y in Y

- There is no smoothing
- Also, there is no way to look at finer-grained aspects of the *x*.
  - Maybe German cars are related?
- Suppose the **numerator** count(x,y) = 0. Then, the estimate will be 0, leading to a model that says *y* is impossible in context *x*.
- We need a more general framework for modeling conditional distributions!

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2
and Jason Eisner (Johns Hopkins University) https://www.cs.jhu.edu/~jason/tutorials/loglin/formulas.pdf

2 Consider Simple Exponential Models

- A very simple manner to define a probability distribution is to exponentiate some scoring function
  - The exponentiation ensures non-negativity

- Thus, we may define a conditional probability distribution as

$$p(y \mid x) \propto \exp \operatorname{score}(x, y) \geq 0$$

- We construct an **arbitrary** score(x, y) that tells us how good x and y are together; higher is better
  - This allows us to construct more complicated distributions
- In future lectures, score will be all sorts of crazy things, e.g. convolutional or recurrent neural network; for now, it's just a black box

② Linear Scoring Functions

Feature Function

- Define a feature function **f** that extracts descriptions. If we decide that our model will have *K* features, then **f**(*x*,*y*) denotes a vector of *K* real numbers ($f_1(x, y)$, $f_2(x, y)$, . . . , $f_K(x, y)$) that describe event y in context x.

Linear Scoring Function

$$\exp \text{score}(x, y) = \exp \sum_{k=1}^{K} (\theta_k \cdot f_k(x, y))$$
$$= \exp(\boldsymbol{\theta} \cdot \mathbf{f}(x, y)) > 0$$

Vector of Feature Weights

$$\boldsymbol{\theta} \in \mathbb{R}^K$$

Quiz: Why can't this equal zero here? (What is new assumption on this slide?)

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2
and Jason Eisner (Johns Hopkins University) https://www.cs.jhu.edu/~jason/tutorials/loglin/formulas

2 Log-linear Models

- **Log-linear models** are a very general family of probability distributions.

Inputs $x \in \mathcal{X}$

Output label $y \in \mathcal{Y}$

Feature function $\mathbf{f} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^K$

Parameters $\boldsymbol{\theta} \in \mathbb{R}^K$

$$p(y \mid x, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp(\boldsymbol{\theta} \cdot \mathbf{f}(x, y))$$

where $$Z(\boldsymbol{\theta}) := \sum_{y' \in \mathcal{Y}} \exp \boldsymbol{\theta} \cdot \mathbf{f}(x, y')$$

- The denominator Z is also called the **partition function**, and normalizes everything to one. From the German Zustandsumme for sum over all states (from statistical mechanics).
- Visualisation of log-linear modeling: https://www.cs.jhu.edu/~jason/tutorials/loglin/#1
  Accompanying Blog: https://www.cs.jhu.edu/~jason/papers/ferraro+eisner.tnlp13.pdf

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

2 Closer Look at the Numerator

$$p(y \mid x, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta} \cdot \mathbf{f}(x,y))}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{\theta} \cdot \mathbf{f}(x,y'))}$$

Numerator

$\boldsymbol{\theta} \in \mathbb{R}^K$ → a real vector

$\mathbf{f}(x,y) \in \mathbb{R}^K$ → a (sparse) real vector

$\boldsymbol{\theta} \cdot \mathbf{f}(x,y) = \sum_{k=1}^{K} \theta_k$ → the sum of weight values for non-zero features, a real number, can be positive or negative

$\exp \boldsymbol{\theta} \cdot \mathbf{f}(x,y)$ → a positive real number

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

2 Closer Look at the Denominator

$$p(y \mid \mathbf{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{x}, y))}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{\theta} \cdot \mathbf{f}(x, y'))}$$

Denominator

$$\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{x}, y'))$$

→ Sum over all possible y values. Runs in $\mathcal{O}(|\mathcal{Y}|)$ time

→ Normalisation constant for the probability distribution

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

2 Why are they Called Log-linear Models?

- Recall the formula for a log-linear model:

$$p(y \mid x, \boldsymbol{\theta}) \propto \exp \boldsymbol{\theta} \cdot \mathbf{f}(x, y)$$

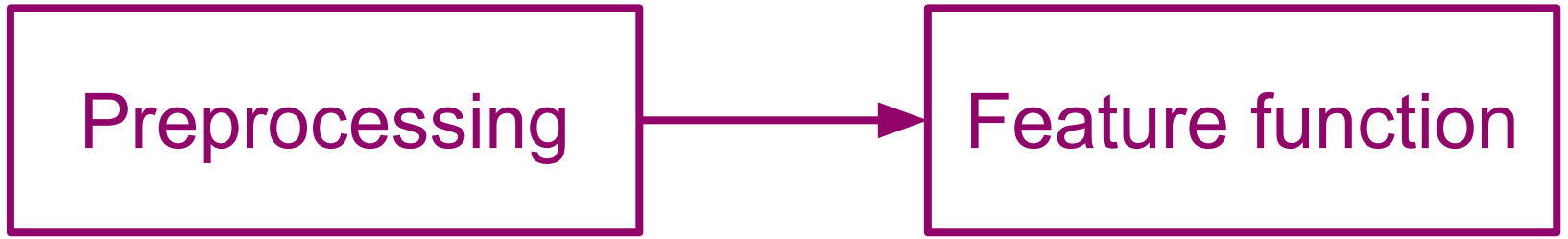
- If we take the log of the above, we end up with a linear function

$$\log p(y \mid x, \boldsymbol{\theta}) = \boldsymbol{\theta} \cdot \mathbf{f}(x, y) + \text{const.}$$

- It's that simple! It's the exponential of a linear function!

② Designing the Feature Function

- The feature function is a crucial part of a log-linear model
- Designing a good feature function is called **feature engineering**.
- Back in the good ol' days, this used to be a **big** portion of machine learning for NLP!
- In this lecture, we talk about feature engineering in two parts



see https://www.slideshare.net/HJvanVeen/feature-engineering-72376750 for more

2 Feature Engineering: Preprocessing

- In preprocessing, our goal is to convert raw text into a form that is more amenable to feature design
  - Tokenization
    - “João’s quadricycle is completely BROKEN!” → [João, ‘s, quadricycle, is, completely, broken, !]
  - Lower casing
    - [joão, …, broken, !]
  - Stemming
    - [joão, ‘s, quadricycle, is, completely, broken, !] → [.., complete, broke, ...]
  - Stop word removal
    - [joão, ‘s, quadricycle, is, complete, broke, !] → [joão, quadricycle, complete, broke]
  - Reducing vocabulary
    - [UNK, ‘s, UNK, complete, broke, !]

see https://www.slideshare.net/HJvanVeen/feature-engineering-72376750 for more

2 Feature Engineering: Feature design

Once we have conducted all our preprocessing, we can obtain our **features**.

- *n*-grams
  - [(BOS, BOS, UNK), (BOS, UNK, ‘s), …, (complete, broke, !)]
- One-hot encoding
  - [0, 0, 1, 0, ...]
- Bag-of-words (*see spam classification example*)
  - [0, 1, 0, 2, …]
- Word embeddings
  - Convert words into continuous representations
  - Can be contextual (e.g., ELMo, BERT) or not (e.g., Glove, fastText)
  - More on this in future lectures!
- Bag-of-embeddings
  - Average embeddings for all words in a sequence
- Domain-specific features *(next slide)*

see https://www.slideshare.net/HJvanVeen/feature-engineering-72376750 for more

② Domain-Specific Feature Engineering

- Domain knowledge could also help us design better functions

- Using 3-gram features, *(thought, of)* → *herself/himself* should be equally probable (exc. data biases)

- But the reflexive pronoun must agree with the subject's grammatical gender

  → We can amend our feature function to track the grammatical gender of the subject

$$p(w_3 = \text{herself} \mid \text{SubjGend} = \text{FEM}, w_1 = \text{thought}, w_2 = \text{of})$$

>

$$p(w_3 = \text{himself} \mid \text{SubjGend} = \text{FEM}, w_1 = \text{thought}, w_2 = \text{of})$$

② Example: Spam Email Classification

- The following would be a bag-of-words feature function for spam email classification

$$f(x,y) = \begin{pmatrix} \texttt{CountOf("money", x)} \wedge \texttt{y = 1} \\ \texttt{CountOf("bank", x)} \wedge \texttt{y = 1} \\ \texttt{CountOf("\$", x)} \wedge \texttt{y = 1} \\ \texttt{CountOf("newsletter", x)} \wedge \texttt{y = 1} \\ \texttt{...} \wedge \texttt{y = 1} \\ \texttt{CountOf("money", x)} \wedge \texttt{y = 0} \\ \texttt{CountOf("bank", x)} \wedge \texttt{y = 0} \\ \texttt{CountOf("\$", x)} \wedge \texttt{y = 0} \\ \texttt{CountOf("newsletter", x)} \wedge \texttt{y = 0} \\ \texttt{...} \wedge \texttt{y = 0} \end{pmatrix}$$

Features for log-linear models can be defined in the form of class-agnostic **templates**

- **TIP**: In this case, we can leverage the sparsity of the feature vector and ignore all $f_k(x,y) = 0$ However, for dense feature functions (e.g., embeddings), we cannot do this.

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

② Example: Spam Email Classification

Different e-mails would get different feature encodings

Welcome to the newsletter highlighting *The Economist*'s best writing on the pandemic.

Our cover this week examines the huge impact that the virus has had on the office and working life. Around the world employees, governments and firms are trying to work out if the office is obsolete—and are coming to radically different conclusions. Before the pandemic only 3% of Americans worked from home regularly; now vast numbers do. How much of this change will stick when a vaccine arrives? The emerging picture is of an "optional office", which people attend, but less frequently. That will have huge economic costs, from the collapse of city-centre cafés to the $16bn budget shortfall that New York subway's system faces. Still, rather than turning the clock back, governments and firms need to adapt.

…

source: *The Economist* newsletter, 12/09/2020

$$f(x,1) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ \cdots \\ 0 \\ 0 \\ 0 \\ 0 \\ \cdots \end{bmatrix} \qquad f(x,0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \cdots \\ 0 \\ 0 \\ 1 \\ 1 \\ \cdots \end{bmatrix}$$

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

2 Example: Spam Email Classification

- Different e-mails would get different feature encodings

```
Greetings to you,
My name is Daichi Haruki Chiyoko I am a credit account
officer at Bank SinoPac here in Hong Kong, The reason
why I contacted you was to discuss a business proposal
and handle it with you. I want to discuss it with you
privately when you reply. Hence, the transaction worth
($11.3 million dollars ) that relates to your name in
our bank. Text me your phone number or you can simply
write me via:Email for more details.
Yours Sincerely,
Mr. Daichi Haruki Chiyoko.
```

source: """Mr. Daichi Haruki Chiyoko"""

$$f(x,1) = \begin{bmatrix} 0 \\ 2 \\ 1 \\ 0 \\ \cdots \\ 0 \\ 0 \\ 0 \\ 0 \\ \cdots \end{bmatrix} \qquad f(x,0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \cdots \\ 0 \\ 2 \\ 1 \\ 0 \\ \cdots \end{bmatrix}$$

adapted from Eisenstein, Introduction to Natural Language Processing, Ch. 2

② Estimating the Parameters Log-linear Models

MLE – Maximum Likelihood Estimation

- Find the parameters that maximize the log-likelihood of the training data

Training data: $\{(x_n, y_n)\}_{n=1}^{N}$

Log-likelihood $L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log p(y_n \mid x_n, \boldsymbol{\theta})$

MLE estimation: $\boldsymbol{\theta}_{\text{MLE}} = \underset{\boldsymbol{\theta} \in \Theta}{\operatorname{argmax}}\, L(\boldsymbol{\theta})$

where Θ is a compact subset of $\mathbb{R}^K$

adapted from Noah A. Smith: https://homes.cs.washington.edu/~nasmith/papers/smith.tut04.pdf

2 Estimating the Parameters Log-linear Models

- **MLE estimation:** $\boldsymbol{\theta}_{\text{MLE}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, L(\boldsymbol{\theta})$

  where Θ is a compact subset of $\mathbb{R}^K$

- **Why is it not** $\boldsymbol{\theta} \in \mathbb{R}^K$ **?**

- What would happen if we have only one training point? Then, likely, some weights are going to zoom off to infinity, but infinity is not in $\mathbb{R}^K$
- **Recall**: The supremum of the set (0, 1) is 1, but the max is undefined because for an number, say 0.9999, we can always find a bigger one in (0,1), say 0.99999!
- The MLE is, in general, a **biased** estimator. With enough assumptions, i.e. compactness assumption and some others, we can prove it is **consistent**!
- The Wikipedia page on MLE has pretty good coverage of the technicalities

adapted from Noah A. Smith: https://homes.cs.washington.edu/~nasmith/papers/smith.tut04.pdf

2 Log-linear Models

How to find the optimal parameters?

- The objective function is convex

- **Convexity:** A real-valued function defined on an *n*-dimensional interval is called **convex** if the line segment between any two points on the graph of the function lies above the graph between the two points.
- **Importantly**, any local minimum of a convex function is a global minimum

adapted from Noah A. Smith: https://homes.cs.washington.edu/~nasmith/papers/smith.tut04.pdf
definition from https://en.wikipedia.org/wiki/Convex_function

2 Log-linear Models

How to estimate the parameters?

- The objective function is convex
- We typically use gradient-based methods, which converge to a global optimum due to convexity
- We typically we minimize the negative log-likelihood (minimization is “standard form” for optimization problems)

$$L(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p(y_n \mid x_n, \boldsymbol{\theta})$$

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta}\in\mathbb{R}^K}{\operatorname{argmax}}\, L(\boldsymbol{\theta}) = \underset{\boldsymbol{\theta}\in\mathbb{R}^K}{\operatorname{argmax}} \prod_{n=1}^{N} p(y_n \mid x_n, \boldsymbol{\theta}) = \underset{\theta\in\mathbb{R}^K}{\operatorname{argmin}} - \sum_{i=1}^{N} \log p(y_n \mid x_n, \boldsymbol{\theta})$$

② The Gradient of a Log-Linear Model

Finding the partial derivative

$$L(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p(y_n \mid x_n, \boldsymbol{\theta}) \qquad p(y \mid x, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta} \cdot \mathbf{f}(x,y))}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{\theta} \cdot \mathbf{f}(x,y'))}$$

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \theta_k} = \sum_{n=1}^{N} f_k(x_n, y_n) - \sum_{n=1}^{N} \sum_{y' \in \mathcal{Y}} p(y' \mid y_n; \boldsymbol{\theta}) f_k(x_n, y')$$

observed feature "counts" **expected feature "count"**

- Recall that the partial derivatives with respect to to each of the parameters θ_k are the components of the gradient

② When is the gradient of a log-linear model zero?

What happens at the (global) optimum?

- At the local optimum we know that the gradient is zero. Hence:

$$\sum_{n=1}^{N} \mathbf{f}(x_n, y_n) = \sum_{n=1}^{N} \sum_{y' \in \mathcal{Y}} p(y' \mid y_n; \boldsymbol{\theta}) \mathbf{f}(x_n, y')$$

observed feature "counts" **expected feature "count"**

- Therefore, the optimum is where the observed feature counts look like the expected feature counts. In other words, our training data looks exactly like what our model predicts through the eyes of our features function **f**.
- This is referred to as **expectation matching**

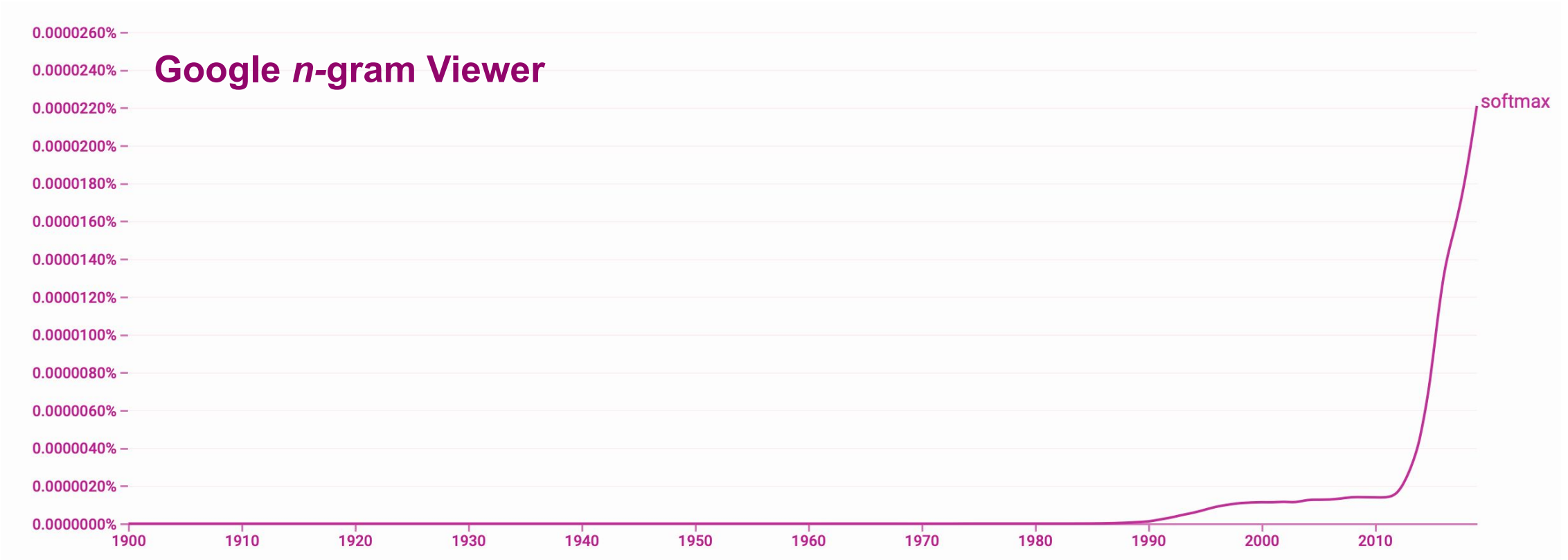
ETH zürich

Softmax

Softmax is the default way of building probabilistic models using neural networks, and it's basically the same as log-linear modeling

2 Why softmax?

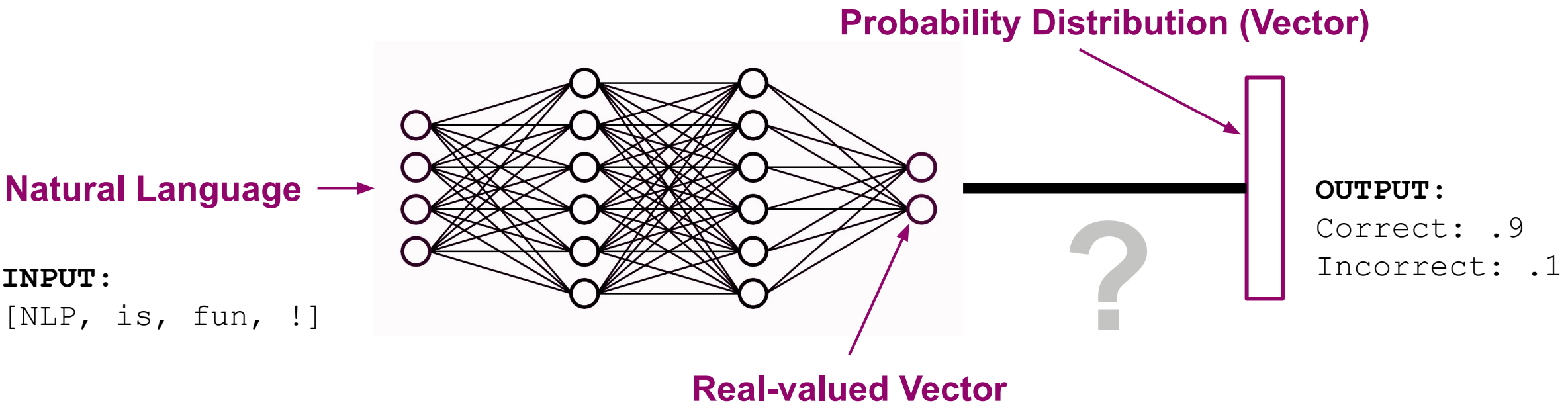
Softmax is the default way of building probabilistic models using neural networks, and it's basically the same as log-linear modeling



https://books.google.com/ngrams/graph?smoothing=3&corpus=26&year_end=2019&year_start=1900&content=log-linear+model&direct_url=t1%3B%2Clog%20-%20linear%20model%3B%2Cc0#t1%3B%2Clog%20-%20linear%20model%3B%2Cc0

③ Why are we learning about the softmax?

- This class is about building probabilistic models of natural language!
- Many of the models discussed will involve neural-network architectures
- Softmax is the default way of building prob. models using neural networks when output is discrete
  - often the case in NLP, e.g., we output a word, token, tree, tag, class, etc.



3 Softmax function

- We can also think of the log-linear model as a **dot product** followed by a **softmax** function

$$\text{softmax}(\mathbf{h}, y, T) = \frac{\exp(h_y/T)}{\sum_{y' \in \mathcal{Y}} \exp(h_{y'}/T)}$$

where

$$h_y = \boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{x}, y)$$

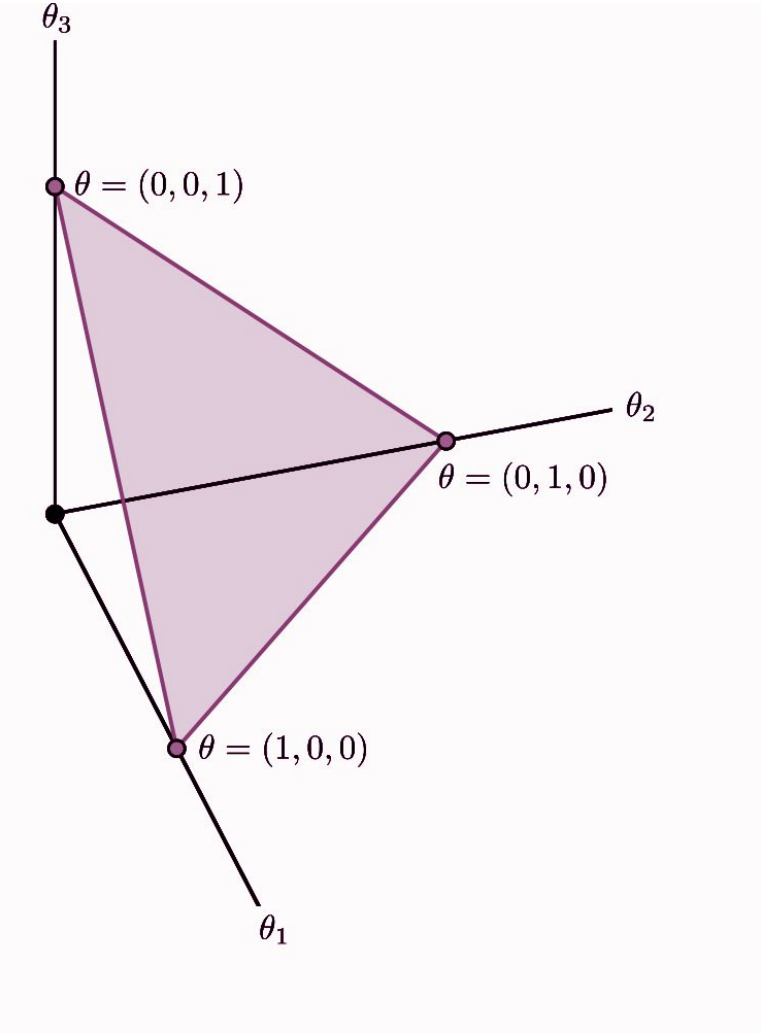
- **Temperature** $T \in \mathbb{R}^K$ is a hyperparameter (often omitted)
- The temperature parameter is called the annealing parameter
  - Why? Wait for next slide!
- The softmax can be thought of as a map from a vector $\in \mathbb{R}^K$ to a point on the simplex Δ^{K-1}

3 Background: What is Annealing?

- Annealing is a technique in metallurgy
- It involves heating and cooling metal to reduce its hardness and increase its ductility
  - I don’t know more about it than that!
- The term annealing and temperature parameter are used as a metaphor to this physical process

3 Background: What is the Simplex?

- When talking about categorical distributions, it is useful to talk about the **simplex** Δ^{K-1}
- The *K*-dimensional simplex $\mathbb{R}^K_{\geq 0}$ is a region of space in such that the sum of the components is 1
- The two dimensional "triangle" living in three dimensional space is the simplex Δ^2
- This means that any point in a *(K-1)*-simplex defines a unique categorical distribution, where *K* is the number of categories.
  - This makes sense since a Categorical only has *K-1* degrees of freedom!



3 Sneak Preview: A Distribution over the Simplex?

- We could go beyond this and assign a probability distribution on the simplex, and sample different categorical distributions from it.
- The **Dirichlet** distribution, which we will see later in the course, does this
- Some examples are shown below

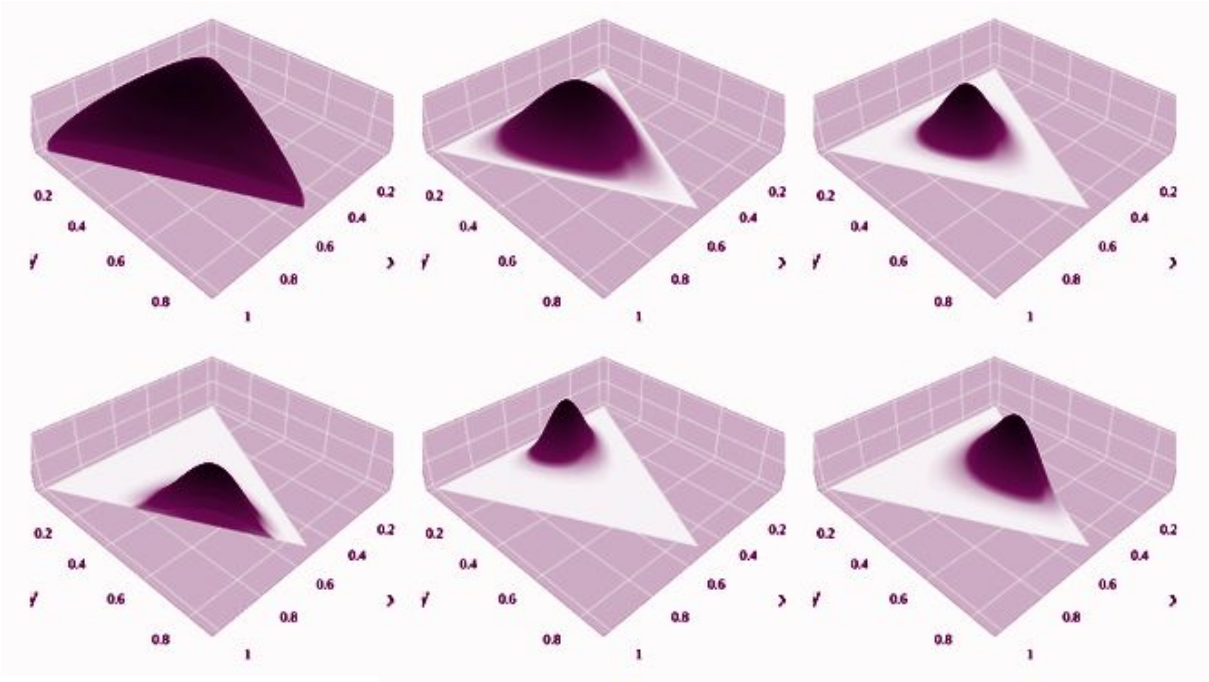
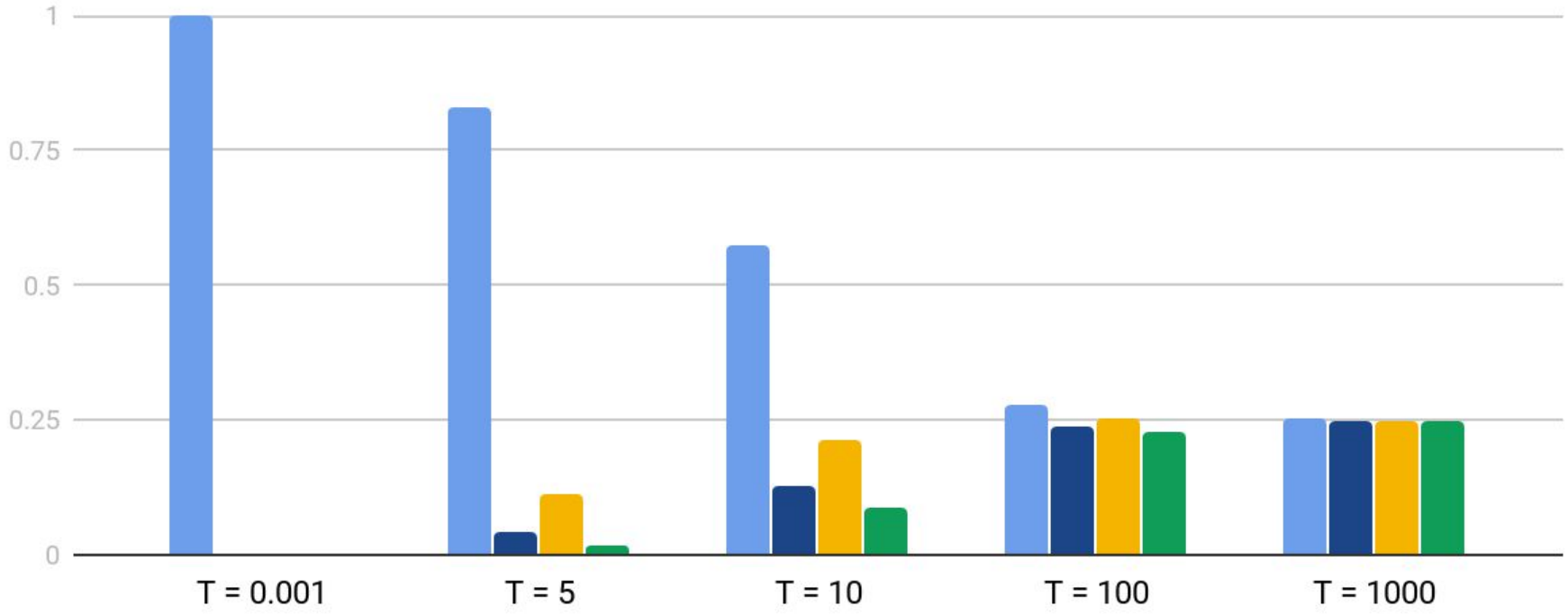


image from https://en.wikipedia.org/wiki/Dirichlet_distribution

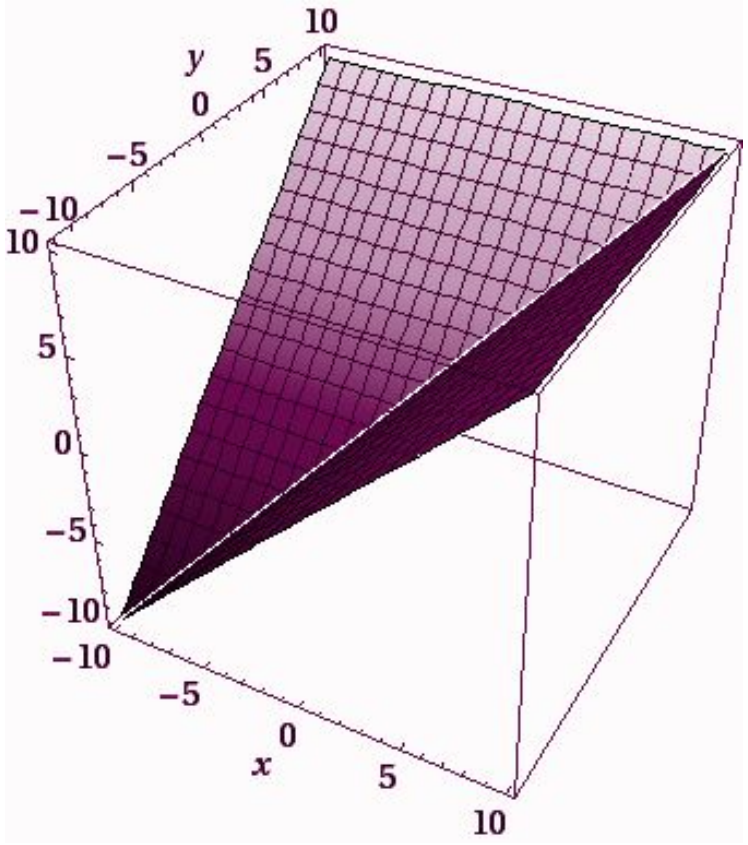
3 Why is it called the Soft*max*?

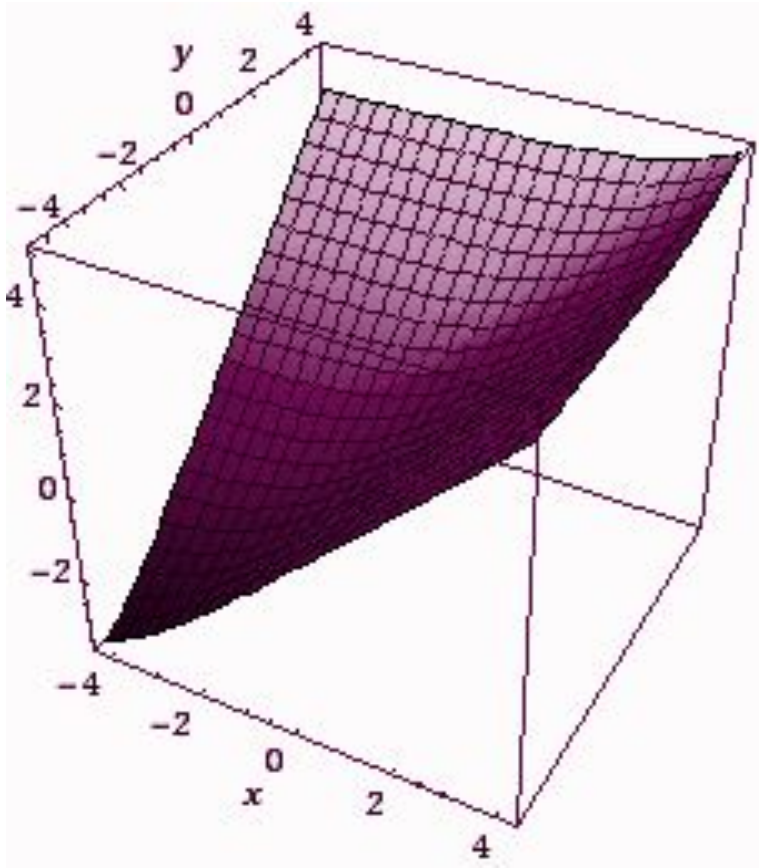
- Let T be a non-negative temperature parameter.
- As $T \rightarrow \infty$, we approach a uniform categorical distribution (maximum entropy).
- As $T \rightarrow 0$ (aka. **annealing**) all mass is placed on the maximum (minimum entropy).
- Hence the name soft**max**



3 Visualizing the Limit

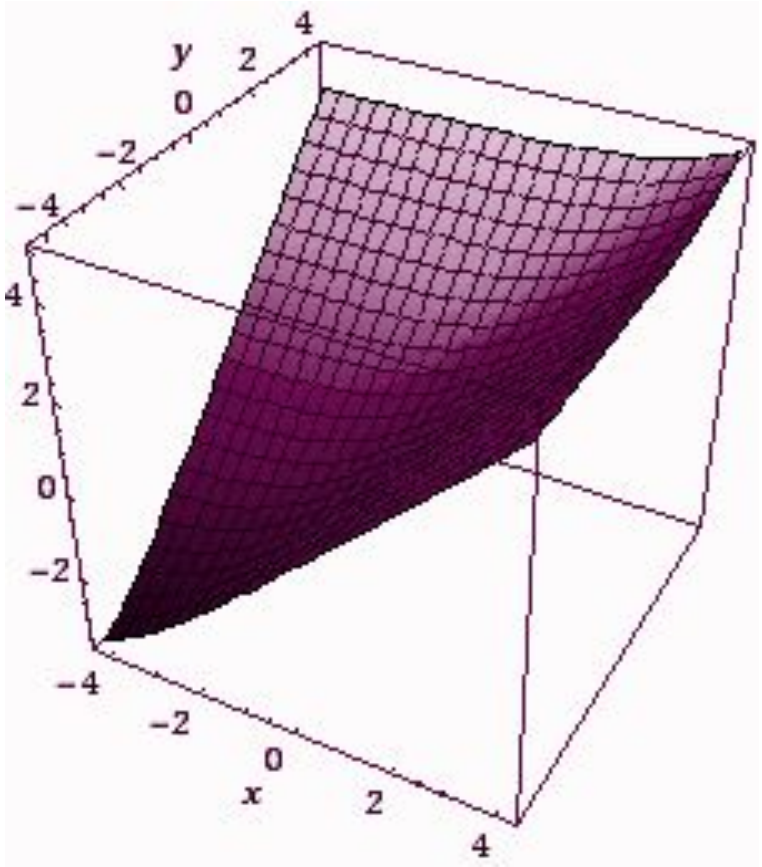
- Let’s Plot max and softmax!





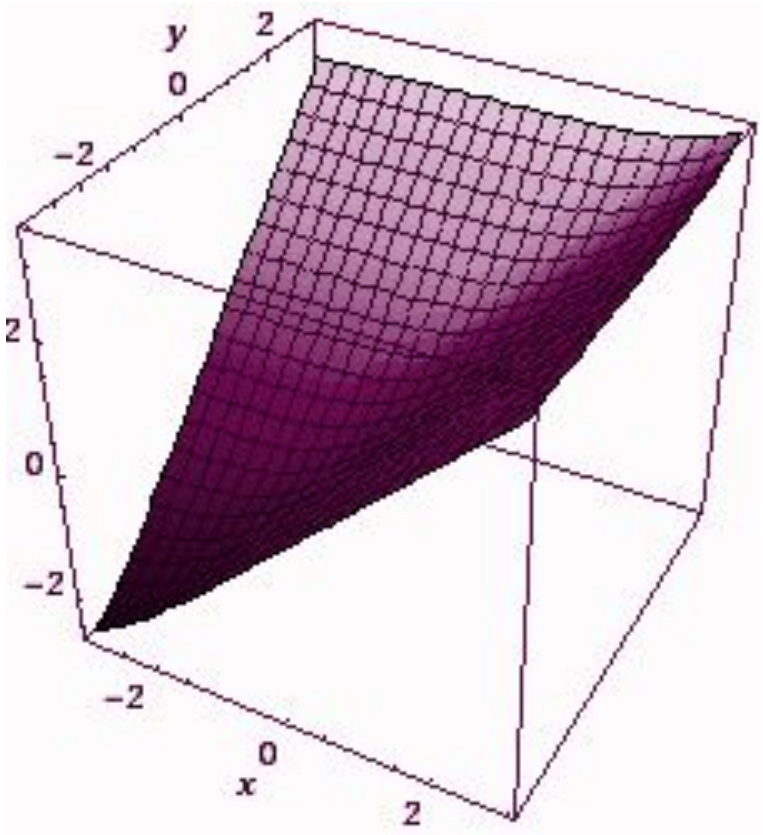
3 Visualizing the Limit

$T = 1$



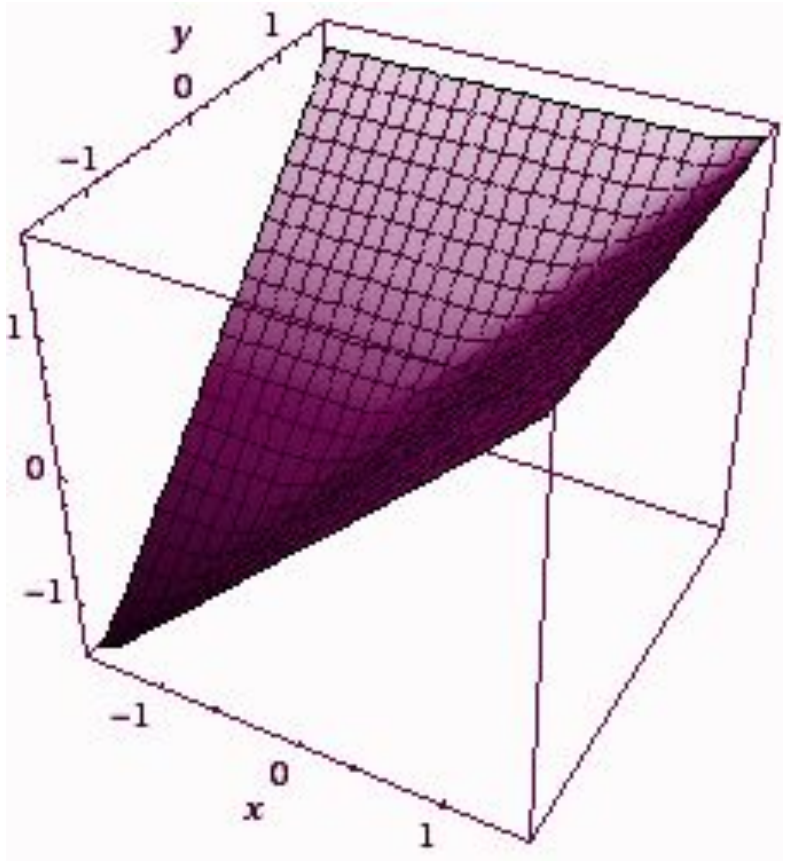
3 Visualizing the Limit

$T = 2$



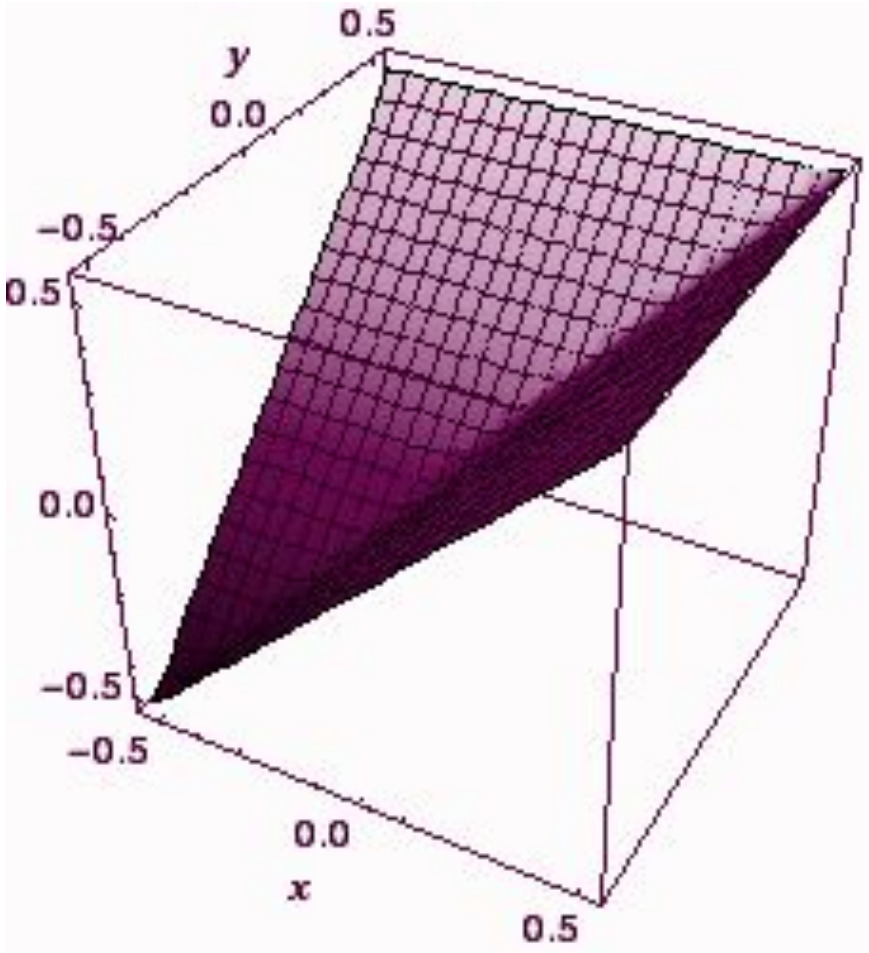
3 Visualizing the Limit

$T = 8$



3 Visualizing the Limit

$T = 64$



3 A more formal relationship

1. Recall we introduced an annealing parameter T that allows us to smoothing interpolate between softmax and max

$$T \log \sum_{i=1}^{n} \exp\left(\frac{x_i}{T}\right)$$

2. I will show that the limit as $T \to 0$ is the max function

3. Annealed softmax is differentiable for any $T > 0$; max is not differentiable at 0

4. An interesting case for analysis geeks
   - We may describe an infinite sequence of differentiable functions whose limit is not differentiable!

3 Why is it called soft***max*** (formally)?

- Suppose wlog $x = \max(x, y)$; assume $x \neq y$ for simplicity

$$\lim_{T\to 0} T \log\left[\exp\left(\frac{x}{T}\right) + \exp\left(\frac{y}{T}\right)\right] = \lim_{T\to 0} T \log\left[\exp\left(\frac{x}{T}\right)\left(\exp\left(\frac{y-x}{T}\right) + 1\right)\right]$$

$$= \lim_{T\to 0} T \log\exp\left(\frac{x}{T}\right) + \lim_{T\to 0} T \log\left(\exp\left(\frac{y-x}{T}\right) + 1\right)$$

$$= x + \lim_{T\to 0} T \cdot \lim_{T\to 0} \log\left(\exp\left(\frac{y-x}{T}\right) + 1\right)$$

$$= x + \lim_{T\to 0} T \cdot \log \lim_{T\to 0}\left(\exp\left(\frac{y-x}{T}\right) + 1\right)$$

$$= x + 0 \cdot 0$$

$$= x$$

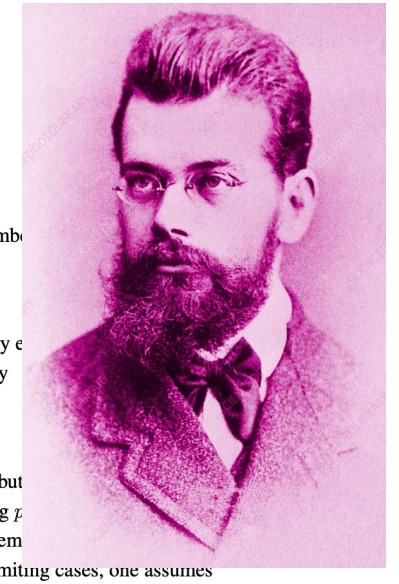
$$= \max(x, y)$$

Argument also works for *n* variables!

Always negative since $x = \max(x, y)$!

$$\lim_{z\to\infty} \exp(-z) = 0$$

3 Historical note on the temperature



CHAPTER IV.

ON THE DISTRIBUTION IN PHASE CALLED CANONICAL, IN WHICH THE INDEX OF PROBABILITY IS A LINEAR FUNCTION OF THE ENERGY.

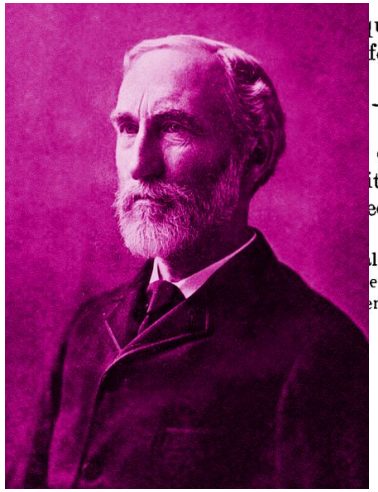
Let us now give our attention to the statistical equilibrium of ensembles of conservation systems, especially to those cases and properties which promise to throw light on the phenomena of thermodynamics.

The condition of statistical equilibrium may be expressed in the form *

$$\Sigma\left(\frac{dP}{dp_1}\dot{p}_1 + \frac{dP}{dq_1}\dot{q}_1\right) = 0, \qquad (88)$$

where P is the coefficient of probability, or the quotient of the density-in-phase by the whole number of systems. To satisfy this condition, it is necessary and sufficient that P should be a function of the p's and q's (the momenta and coördinates) which does not vary with the time in a moving system. In all cases which we are now considering, the energy, or any function of the energy, is such a function.

$$P = \text{func.}\ (\epsilon)$$



…uation, as indeed appears identi-
…form

$$\left(-\frac{dP}{dp_1}\frac{d\epsilon}{dq_1}\right) = 0.$$

… conditions to which P is subject,
…itions of statistical equilibrium, as
…ed in the definition of the coeffi-

…lso the paragraph following equation (20).
…es which can affect the systems are here
…ems and constant in time.

cient of probability, whether the case is one of equilibrium or not. These are: that P should be single-valued, and neither negative nor imaginary for any phase, and that expressed by equation (46), viz.,

$$\int_{\text{phases}}^{\text{all}} \cdots \int P\, dp_1 \ldots dq_n = 1. \qquad (89)$$

These considerations exclude

$$P = \epsilon \times \text{constant},$$

as well as

$$P = \text{constant},$$

as cases to be considered.

The distribution represented by

$$\eta = \log P = \frac{\psi - \epsilon}{\Theta}, \qquad (90)$$

or

$$P = e^{\frac{\psi-\epsilon}{\Theta}}, \qquad (91)$$

I. Kinetic Energy Has Discrete Values

We assume initially, each molecule is only capable of assuming a finite numb…

$$0, \frac{1}{q}, \frac{2}{q}, \frac{3}{q}, \ldots \frac{p}{q},$$

where p and q are arbitrary finite numbers. Upon colliding, two molecules may e… after the collision both molecules still have one of the above velocities, namely

$$0, \text{or } \frac{1}{q}, \text{or } \frac{2}{q}, \textit{ etc. till } \frac{p}{q},$$

This assumption does not correspond to any realistic mechanical model, but… mathematically, and the actual problem to be solved is re-established by letting p…

Even if, at first sight, this seems a very abstract way of treating the problem… desired objective, and when you consider that in nature all infinities are but limiting cases, one assumes each molecule can behave in this fashion only in the limiting case where each molecule can assume more and more values of the velocity.

To continue, however, we will consider the kinetic energy, rather than the velocity of the molecules. Each molecule can have only a finite number of values for its kinetic energy. As a further simplification, we assume that the kinetic energies of each molecule form an arithmetic progression, such as the following:

$$0, \epsilon, 2\epsilon, 3\epsilon, \ldots p\epsilon$$

- The temperature parameter gets its name from the **Boltzmann** distribution (aka. the **Gibbs** distribution) in physics
- It was studied by both Boltzmann and Gibbs in the context of statistical mechanics
- It describes the probability that a physical system is on a specific state given the state's energy and the system's temperature

images from Kim Sharp and Franz Matschinsky, *Translation of Ludwig Boltzmann's Paper "On the Relationship between the Second Fundamental Theorem of the Mechanical Theory of Heat and Probability Calculations Regarding the Conditions for Thermal Equilibrium"*; Josiah Willard Gibbs, Elementary Principles in Statistical Mechanics

ETH zürich

③ Gradient of the Softmax

- Let *T* = *1*. We can derive the gradient of the log-softmax with the chain rule

$$\log \text{softmax}(\mathbf{h}, y) = h_y - \log \sum_{y' \in \mathcal{Y}} \exp(h_{y'})$$

- Therefore

$$\frac{\partial \log \text{softmax}(\mathbf{h}, y)}{\partial h_i}$$

$$= \frac{\partial}{\partial h_i}\left[h_y - \log \sum_{y' \in \mathcal{Y}} \exp(h_{y'})\right] = \delta_{yi} - \frac{\partial}{\partial h_i}\left[\log \sum_{y' \in \mathcal{Y}} \exp(h_{y'})\right]$$

$$= \delta_{yi} - \frac{1}{\sum_{y' \in \mathcal{Y}} \exp(h_{y'})} \frac{\partial}{\partial h_i}[\exp(h_i)] = \delta_{yi} - \frac{1}{\sum_{y' \in \mathcal{Y}} \exp(h_{y'})} \exp(h_i)$$

$$= \delta_{yi} - \text{softmax}(\mathbf{h}, i)$$

③ Gradient of the Log-linear Model Rederived

- Since the log-linear model is just a softmaxed dot product, it easy to (re-)derive its gradient

$$\log \text{softmax}(\mathbf{h}, y) = h_y - \log \sum_{y' \in \mathcal{Y}} \exp(h_{y'})$$

$$\text{where } \mathbf{h} = \boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{x}, y)$$

- So…

$$\frac{\partial \log \text{softmax}(\boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x},y),y)}{\partial \boldsymbol{\theta}} = \frac{\partial \log \text{softmax}(\mathbf{h},y)}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}} = \sum_i \boxed{\frac{\partial \log \text{softmax}(\mathbf{h},y)}{\partial h_i}} \frac{\partial h_i}{\partial \boldsymbol{\theta}}$$

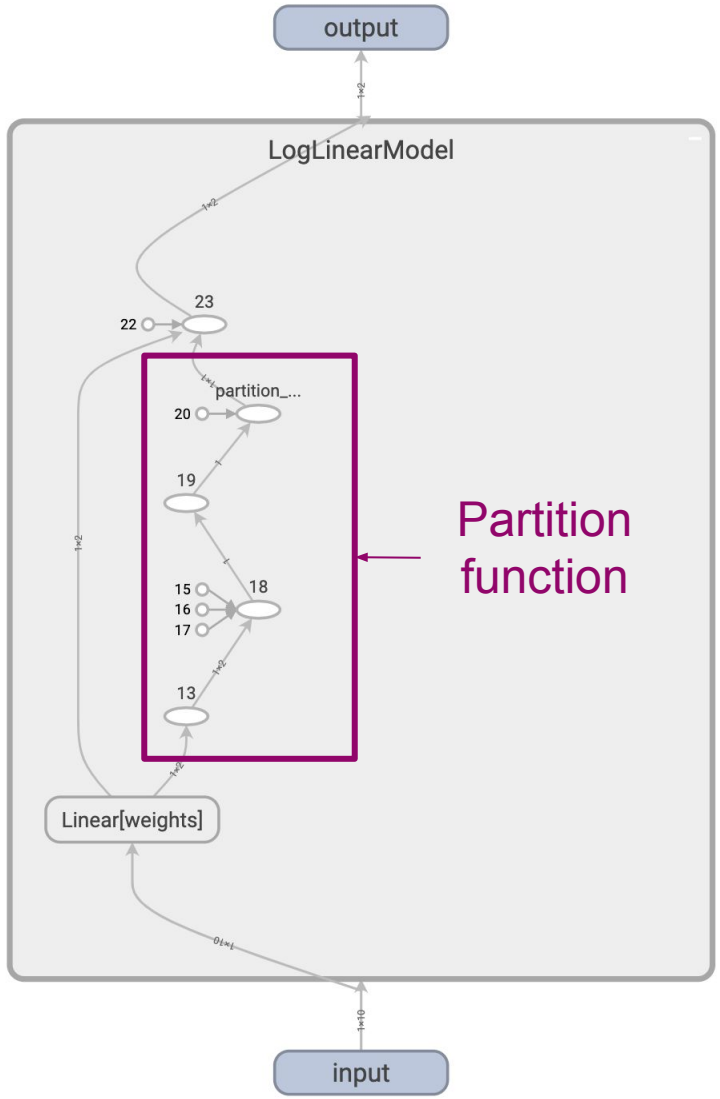
We just did this!

where $\frac{\partial h_i}{\partial \boldsymbol{\theta}} = \frac{\partial}{\partial \boldsymbol{\theta}} \boldsymbol{\theta} \cdot \mathbf{f}(\mathbf{x}, i) = \mathbf{f}(\mathbf{x}, i)$

so $$\frac{\partial \log \text{softmax}(\boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x},y),y)}{\partial \boldsymbol{\theta}} = \mathbf{f}(\mathbf{x}, y) - \sum_i \text{softmax}(\mathbf{h}, i)\mathbf{f}(\mathbf{x}, i)$$

$$= \mathbf{f}(\mathbf{x}, y) - \mathbb{E}_{y' \sim \text{softmax}(\mathbf{h},\cdot,T)} [\mathbf{f}(\mathbf{x}, y')]$$

3 Quick Digression: Softmax and Backpropagation

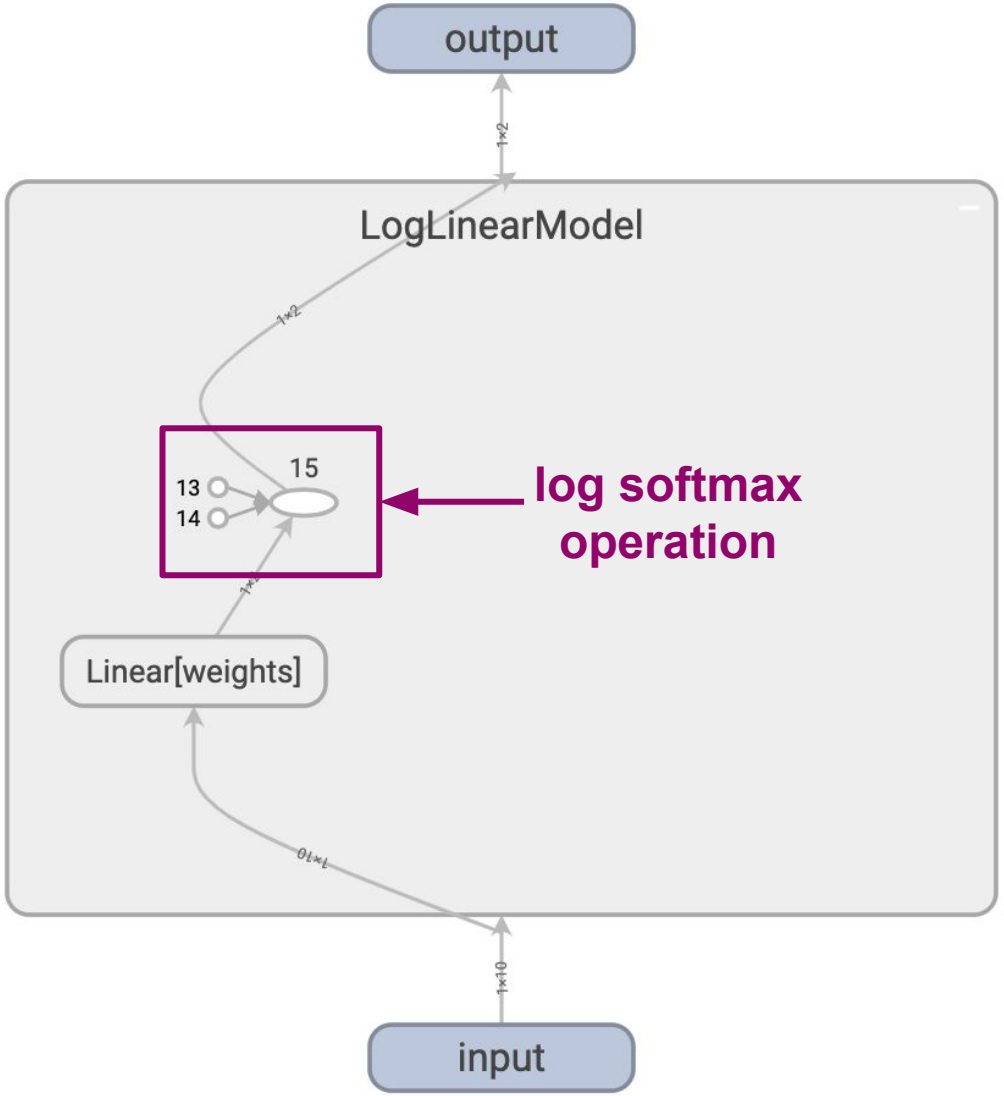


Partition function

If we implement this from scratch on pyTorch, we would get a computational graph somewhat like this

It works, but can we do better…?

3 Quick Digression: Softmax and Backpropagation



Many of these frameworks have these layers as **primitives**, just like +, -, exp, etc.

Their forward and backward passes may be optimized! **So always try to use them**

③ Relationship to Neural Modeling

- How do log-linear models relate to neural networks?
  - You design your own “hidden representation”
- Log-linear models were the standard tool for NLP from 1995-2015
  - NLPers hand-crafted their own feature functions for their modeling
- When the deep learning wave hit, log-linear models became known as the softmax
- NLPers now let neural networks extract the automatically
- Remember how we defined the softmax as

$$\text{softmax}(\mathbf{h}, y, T) = \frac{\exp(h_y / T)}{\sum_{y' \in \mathcal{Y}} \exp(h_{y'} / T)}$$

- In the context of the softmax, ***h*** can be **whatever you want!**

$$h_y = \text{blackbox}(y)$$

- In upcoming lectures we will see all sorts of ways of designing ***h***

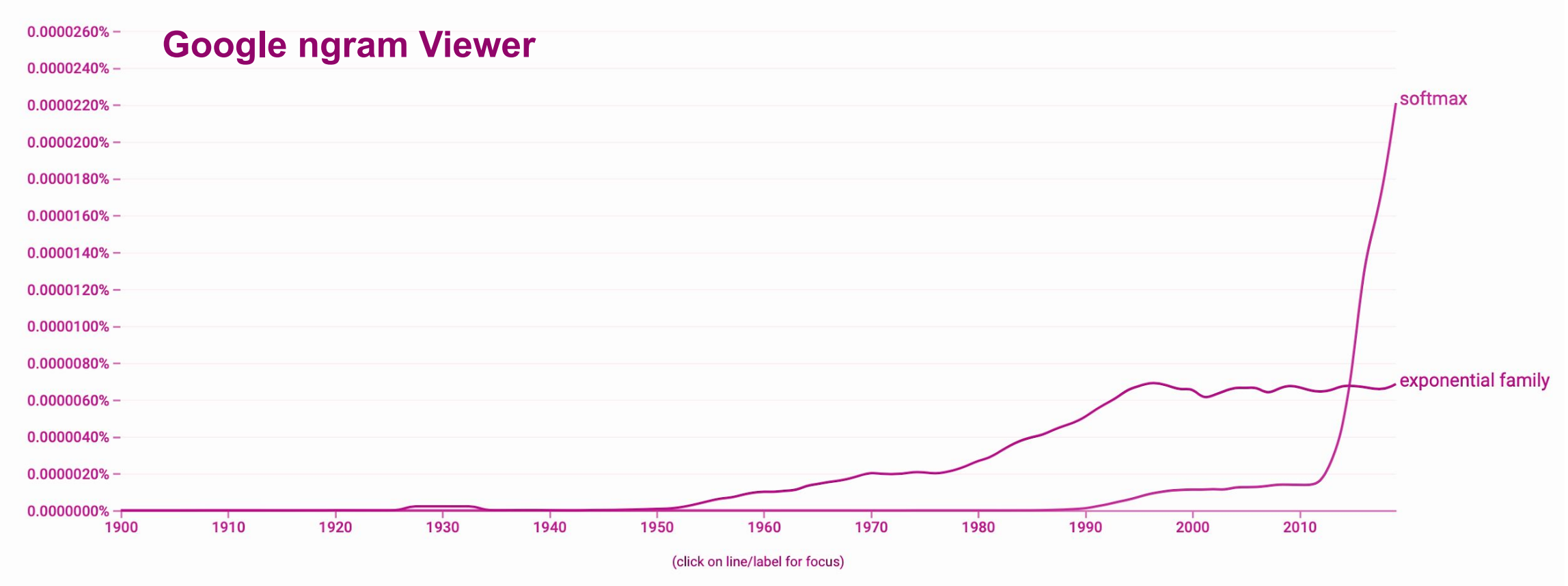
ETH zürich

Generalizing the Softmax: The Exponential Family

Exponential families generalize the softmax

4 Introduction to the Exponential Family

Exponential families generalize the softmax



https://books.google.com/ngrams/graph?content=softmax%2Cexponential+family&year_start=1900&year_end=2019&corpus=26&smoothing=3

4 Why “The Exponential Family”?

- The **exponential family** is a family of probability distributions over x ϵ *X*, parameterized by some ***θ***, of the form

$$p(x \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(x))$$

where

- *Z(**θ**)* is the partition function
- *h(**x**)* determines the support (exact zeros in the model)
- *θ* are the **canonical parameters**
- *Φ(x)* are the **sufficient statistics**
  - This is the same as a feature function! Just different terminology between statistics and NLP!

adapted from Philippe Rigolett, Statistics for Applications, Lecture 21: Generalized Linear Models

4 Introduction to the Exponential Family

More informally

- We have two variables, ***x*** and ***θ***. There are many ways of defining $p(x, \boldsymbol{\theta})$
- Exponential families are those distributions that can be written like this

1. The interaction between the terms must be an **exponentiated dot product**

$$p(x \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(x))$$

2. We allow each variable to individually **scale** the distribution

adapted from Philippe Rigolett, Statistics for Applications, Lecture 21: Generalized Linear Models

4 Why care about the Exponential Family?

- Exponential families are the most general form of the softmax and we will see various exponential families throughout the class

- The terms "distribution" and "family" are a bit confusing
- **An exponential family:** *set* of distributions, where the specific distribution varies with the parameter.

- **A parametric family of distributions:** often referred to as "*a* distribution" (like "the normal distribution", meaning "the family of normal distributions")

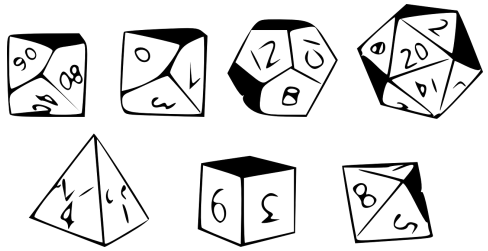
- **The set of all exponential families:** sometimes loosely referred to as "the" exponential family.

$$p(x \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(x))$$

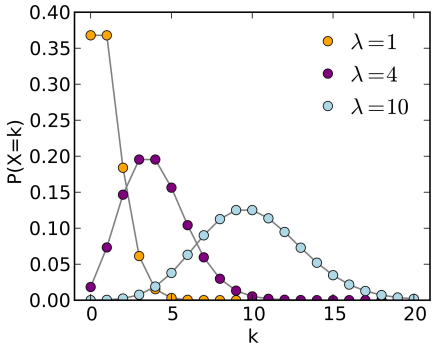
4 Why care about the Exponential Family?

- This is just *one* of the many ways to define the joint distribution between $\boldsymbol{x}$ and $\boldsymbol{\theta}$, why should you care?
- If you prove something about the exponential family, you've proven it about a lot of distributions at once!

Discrete



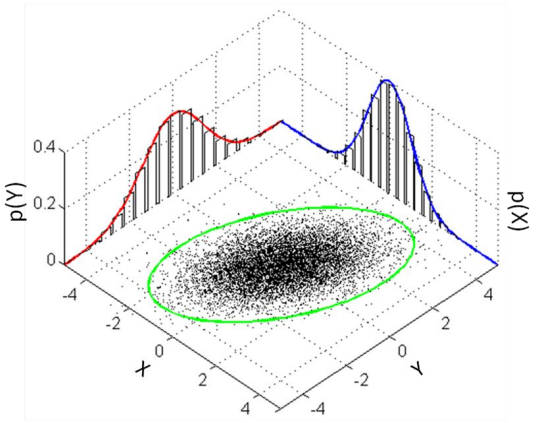
Bernoulli/Categorical



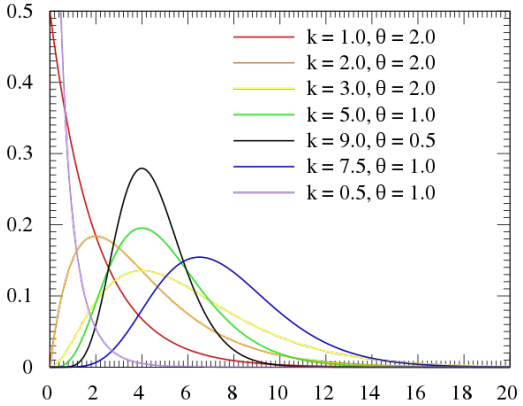
Poisson

...and many more

Continuous



Gaussian



Gamma

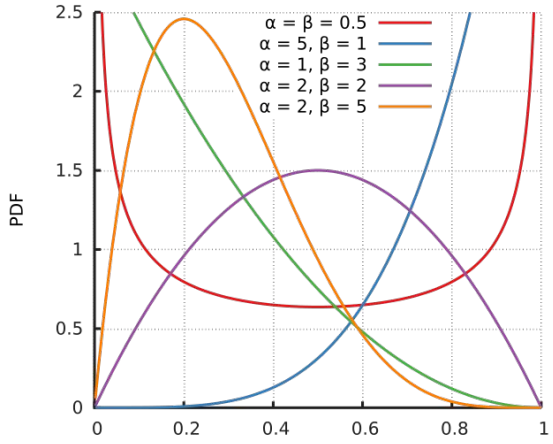


image credit: Wikipedia

Beta

4 Why care about the Exponential Family?

Data Type	Domain	Distribution
univariate, discrete, binary	$x \in \{0, 1\}$	Bernoulli
univariate, discrete, multi-valued	$x \in \{1, 2, \ldots, K\}$	categorical
univariate, continuous, unbounded	$x \in \mathbb{R}$	univariate normal
univariate, continuous, bounded	$x \in [0, 1]$	beta
multivariate, continuous, unbounded	$\mathbf{x} \in \mathbb{R}^K$	multivariate normal
multivariate, continuous, bounded, sums to one	$\mathbf{x} = [x_1, x_2, \ldots, x_K]^T$ $x_k \in [0, 1], \sum_{k=1}^{K} x_k = 1$	Dirichlet
bivariate, continuous, x_1 unbounded, x_2 bounded below	$\mathbf{x} = [x_1, x_2]$ $x_1 \in \mathbb{R}$ $x_2 \in \mathbb{R}^+$	normal-scaled inverse gamma
multivariate vector $\mathbf{x}$ and matrix $\mathbf{X}$ $\mathbf{x}$ unbounded, $\mathbf{X}$ square, positive definite	$\mathbf{x} \in \mathbb{R}^K$ $\mathbf{X} \in \mathbb{R}^{K \times K}$ $\mathbf{z}^T \mathbf{X} \mathbf{z} > 0 \quad \forall\, \mathbf{z} \in \mathbb{R}^K$	normal inverse Wishart

Table 2.1: Common probability distributions: the choice of distribution depends on the type/domain of data to be modeled.

adapted from Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

4 Why care about the Exponential Family?

The exponential family has many interesting properties

- Only family with **finite sufficient statistics**

Given data drawn from a member of the exponential family, we can compress all of it into a finite vector without any loss of information

- Only family with **conjugate priors**

In Bayesian statistics, conjugate priors make our life easier

- Corresponds to **maximum entropy distributions** subject to certain constraints
  - Namely, that the sufficient statistic matches the training data
  - Generalization of how log-linear models are maximum entropy!

4 Background: Bayesian Machine Learning

Earlier we spoke of the Bayesian interpretation of probability--here's how it relates to ML

- Let $\mathcal{D}$ be some dataset, and $\boldsymbol{\theta}$ be the parameters of our model
- According to Bayes theorem

$$\underbrace{p(\boldsymbol{\theta} \mid \mathcal{D})}_{\text{posterior}} = \frac{\overbrace{p(\mathcal{D}|\boldsymbol{\theta})}^{\text{likelihood}}\overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\mathcal{D})}_{\substack{\text{model evidence} \\ \text{(aka. marginal likelihood)}}}}$$

- In most recent NLP work, people usually only care about maximizing the likelihood
  - Formally this means solving $\boldsymbol{\theta}^\star = \text{argmax}_{\boldsymbol{\theta}}\, p(\mathcal{D} \mid \boldsymbol{\theta})$
  - This is known as **maximum likelihood estimation (MLE)**

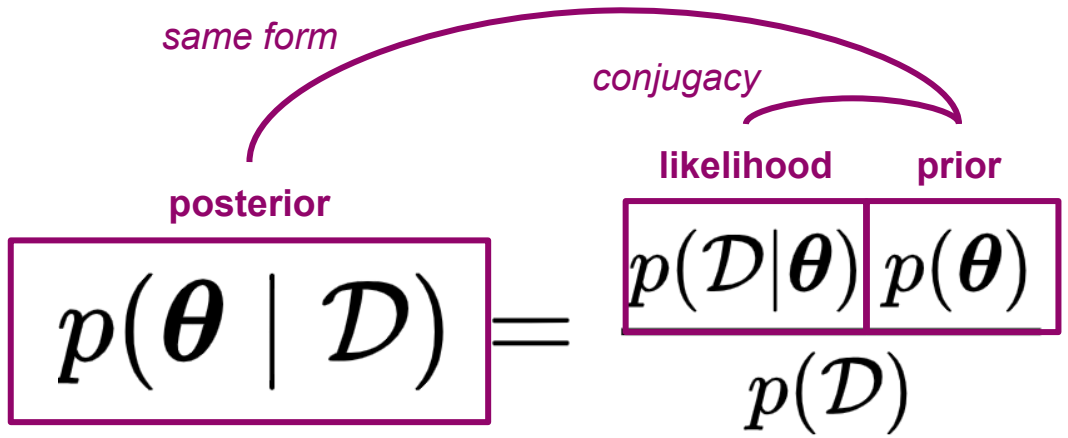
4 Background: Bayesian Machine Learning

$$\underbrace{p(\boldsymbol{\theta} \mid \mathcal{D})}_{\text{posterior}} = \frac{p(\mathcal{D}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{D})}$$

- In a Bayesian setting we tend to care about the **posterior**
  - This is a distribution over the parameters of our model after we have seen all our data
  - It quantifies the **uncertainty** we have about our parameters
- This idea can also be extended to prediction (aka. the posterior predictive)
- In some fields (e.g., medicine) quantifying uncertainty can be very important!

4 Background: Bayesian Machine Learning

- Unfortunately, a lot of the time being Bayesian comes at the cost of a lot more work
  - e.g., we might have to deal with intractable integrals
- One exception to this is when we use **conjugate priors**
- The idea is that, for certain kinds of **likelihood**, we can pick a **prior** distribution such that the **posterior** distribution is in the same form



- Which kinds of likelihoods can we do this for?

4 Conjugacy

- Likelihoods in the **exponential family**!

- Intuition as to why: is the posterior must be of the same form as the prior, then we must be able to summarize the data into a finite vector (recall finite sufficient statistics)

Distribution	Domain	Parameters modeled by
Bernoulli	$x \in \{0, 1\}$	beta
categorical	$x \in \{1, 2, \ldots, K\}$	Dirichlet
univariate normal	$x \in \mathbb{R}$	normal inverse gamma
multivariate normal	$\mathbf{x} \in \mathbb{R}^k$	normal inverse Wishart

adapted from Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

4 Conjugacy

Example: Univariate Normal Distribution

univariate, continuous, unbounded	$x \in \mathbb{R}$	univariate normal

… is modelled by Normal-Scaled Inverse Gamma

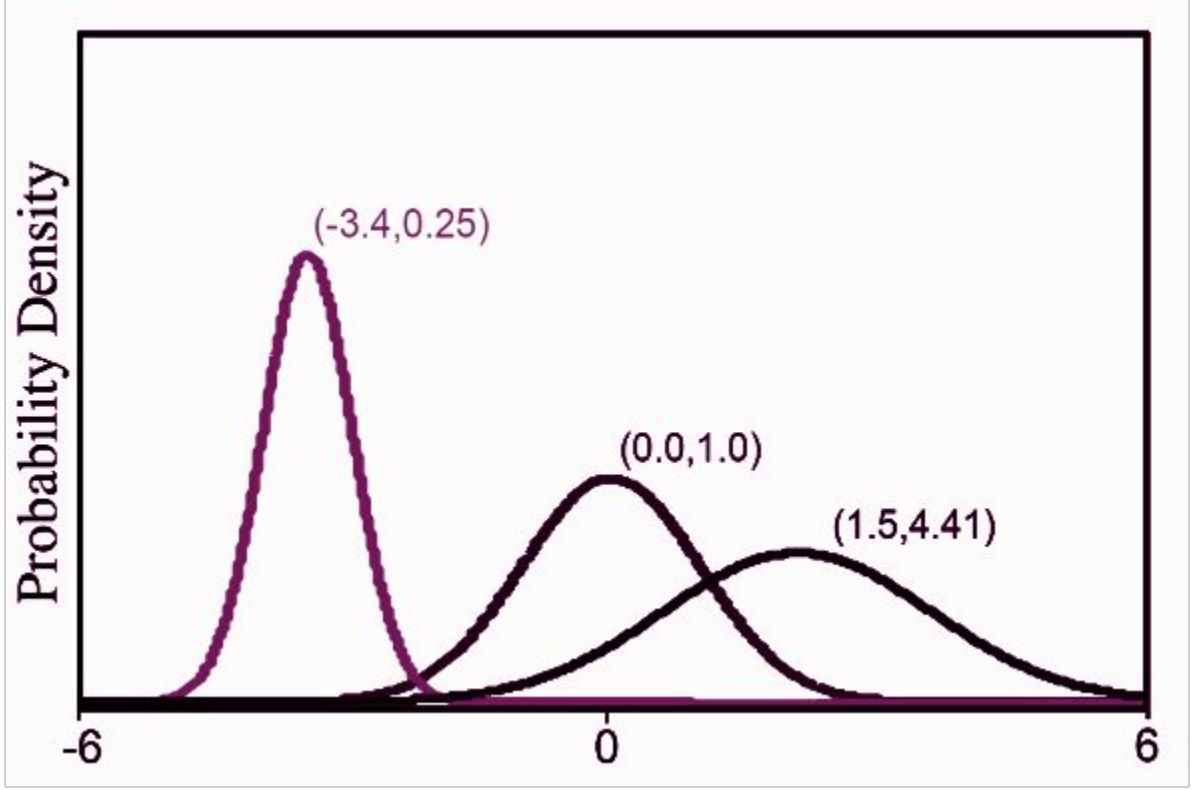
bivariate, continuous, x_1 unbounded, x_2 bounded below	$\mathbf{x} = [x_1, x_2]$ $x_1 \in \mathbb{R}$ $x_2 \in \mathbb{R}^+$	normal-scaled inverse gamma

controls **mean** of univariate normal distribution

controls **variance** of univariate normal distribution

adapted from Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

4 The Gaussian is an Exponential Family Distribution



$$p(x \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(x))$$

$$\text{Let } x \in \mathbb{R}$$

$$\phi(x) = [x, x^2]^\top$$

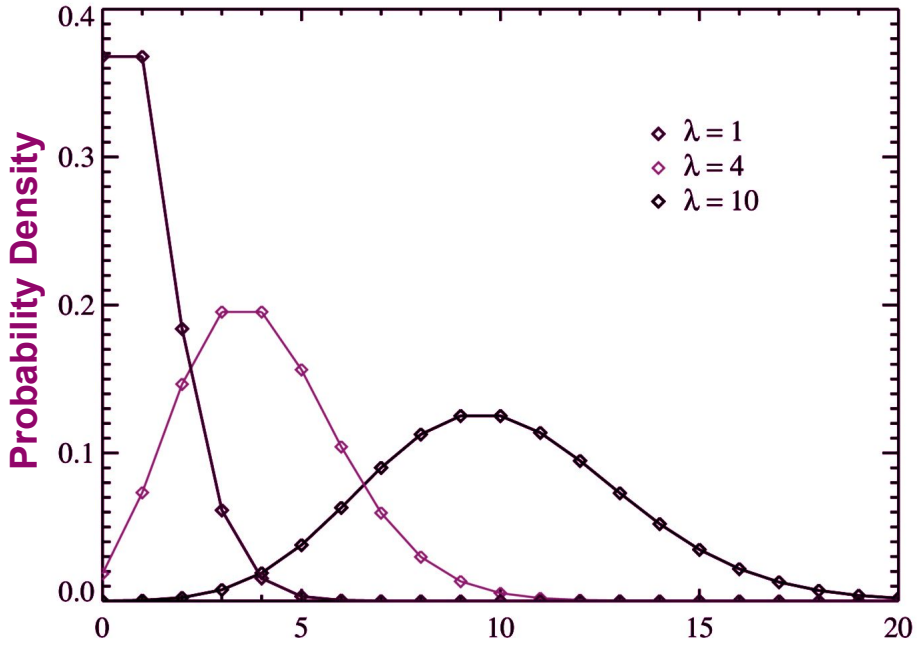
$$Z(\theta) = \sqrt{2\pi\sigma^2} \exp\left(\frac{\mu^2}{2\sigma^2}\right)$$

$$h(x) = 1$$

$$\theta = \left[\frac{1}{2\sigma^2}, \frac{\mu}{\sigma^2}\right]^\top$$

- Univariate normal distribution is an exponential family. Takes 2 parameters μ and $\sigma^2>0$.
- Not standard formulation of the Gaussian!

4 The Poisson in an Exponential Family Distribution



$$p(x \mid \lambda) = \frac{\lambda^x \exp(-\lambda)}{x!}$$

$$p(x \mid \theta) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\theta \cdot \phi(x))$$

$$p(x \mid \lambda) = \frac{1}{\exp(\exp(\log \lambda))} \frac{1}{x!} \exp(x \log \lambda)$$

$$\text{Let } x \in \mathbb{N}$$

$$\phi(x) = x$$

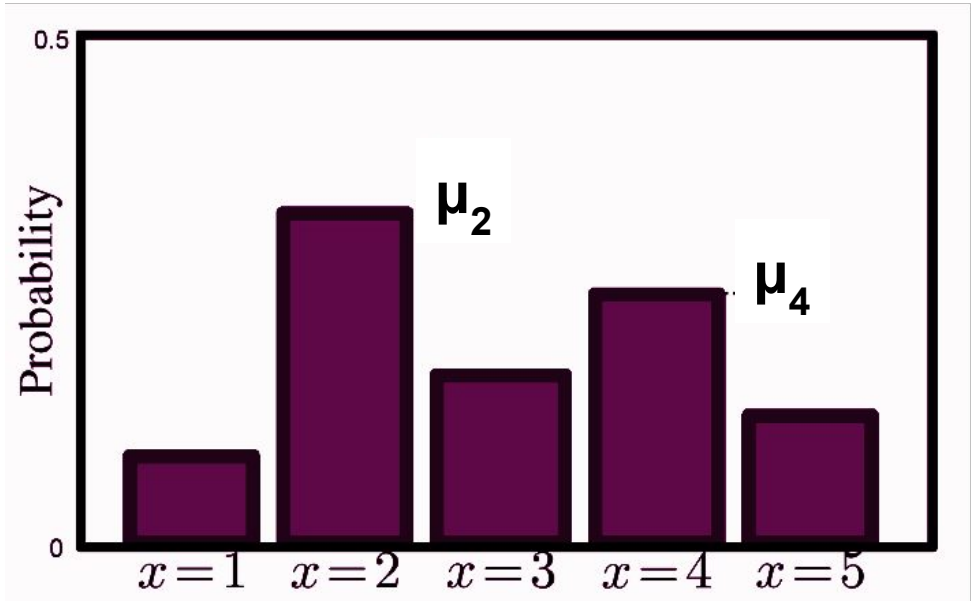
$$Z(\theta) = \exp(\exp(\theta))$$

$$h(x) = \frac{1}{x!}$$

$$\theta = \log \lambda$$

adapted from Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

④ The Categorical in an Exponential Family Distribution



$$X \sim \text{Categorical}(\boldsymbol{\mu}) \Rightarrow p(x \mid \boldsymbol{\mu}) = \mu_x$$

$$p(x \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(x) \exp(\boldsymbol{\theta} \cdot \phi(x))$$

Looks just like the log-linear model formula!

(In this case, h(x) = 1 unless some element x has probability 0, then h(x) = 0; h encodes structural zeros)

- Categorical distribution describes situation with *K* possible outcomes *y=1,…, y=k*.

adapted from Simon J.D. Prince, Computer Vision: Models, Learning, and Inference

4 Maximum Entropy: Motivation

- Let's take a detour which will make sense in a moment
- Let *p* be a discrete distribution over $X = \{1, \ldots, C\}$
- Let $f : X \to \mathbb{R}$
- Suppose we know $\mathbb{E}[f(x)] = \sum_{x \in X} p(x) f(x) = F$
- That is, we know the expectation, but not the underlying distribution
- **What is $\mathbb{E}[g(x)]$ for some $g : X \to \mathbb{R}$?**

adapted from Edwin T. Jaynes, "Information theory and statistical mechanics"

4 Maximum Entropy: Motivation

- If $C = 2$ we have

$$F = P(X = 1)f(1) + P(X = 2)f(2)$$
$$1 = P(X = 1) + P(X = 2)$$

- We can solve this linear system to fully determine p(C)
- But for $C > 2$ there is no unique solution; in fact, there are an infinite number of solutions!

And yet this problem is ubiquitous

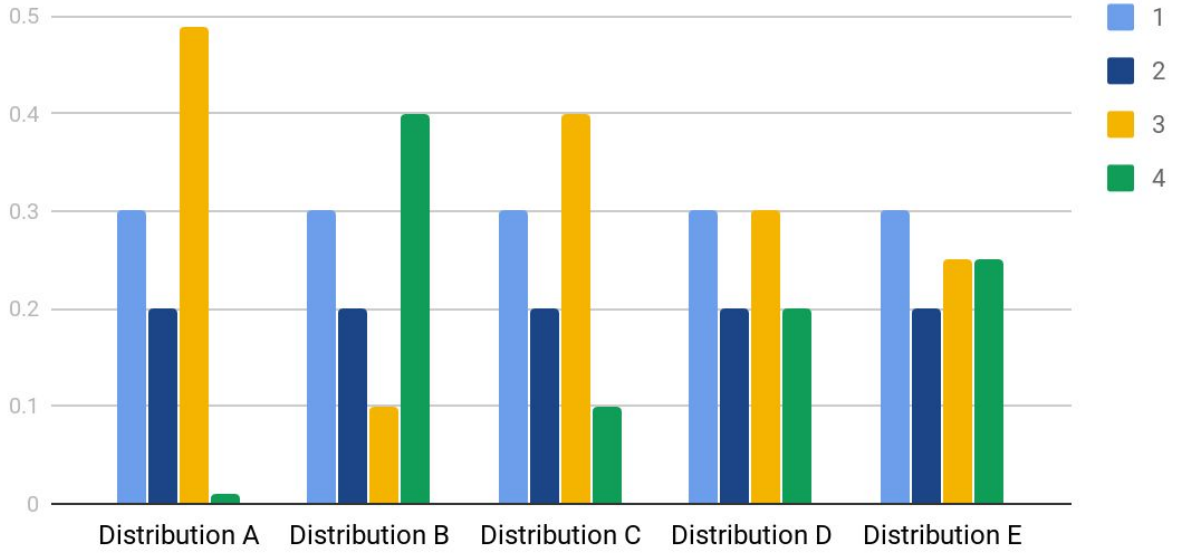
- We observe the expected counts of some features for each datapoint, and how they are classified, *but what is the true conditional probability p(Y|X)?*
- We observe the mean and covariance of an r.v. X, and estimate p(X), b*ut what is the true probability p(X)?*
- During estimation, which *p* do we choose?

adapted from Edwin T. Jaynes, “Information theory and statistical mechanics”

4 Maximum Entropy: Motivation

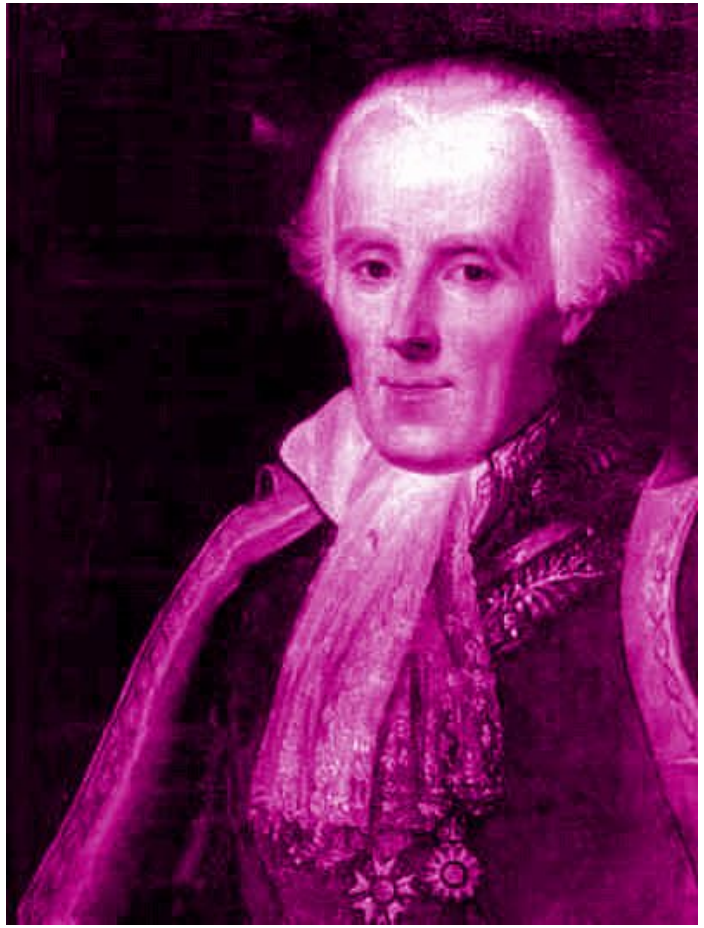
- Consider the case where *C* = *4*, let $X = \{1, 2, 3, 4\}$
- Assume we know $\mathbb{E}[\mathbb{I}[X = 1]] = 0.3, \mathbb{E}[\mathbb{I}[X = 2]] = 0.15$

- **Which distribution would you pick?** (Note how only *p(X = 3)* and *p(X = 4)* change)



adapted from Edwin T. Jaynes, “Information theory and statistical mechanics”

4 Maximum Entropy: Motivation



- Laplace's "**Principle of Insufficient Reason**" stipulates that if we have no reason to believe otherwise, we should assign **equal probability** to each event
- This may make sense intuitively, but is it *justified*?
- It also poses some issues, e.g.
  - What does "equal probability" mean for a continuous random variable?

adapted from Edwin T. Jaynes, "Information theory and statistical mechanics"

4 Maximum Entropy: Motivation

- Jaynes argued strongly in favor of the **maximum entropy principle**

"[we need] a probability assignment which avoids bias, while agreeing with whatever information is given" — Jaynes (1957)

- The crux of the argument is that the information-theoretic **entropy** quantifies the uncertainty of a distribution

$$\mathrm{H}(p) = -\sum_{x \in X} p(x) \log p(x)$$

- We should opt for the distribution that maximizes the entropy, given whatever information we have available (e.g., mean, covariance, expected counts, etc.)

adapted from Edwin T. Jaynes, "Information theory and statistical mechanics"

4 Maximum Entropy: A bit of history

- Jaynes developed this idea in the context of statistical mechanics, establishing a link to information theory

> *"Previously, one constructed a theory based on the equations of motion, supplemented by additional hypotheses of ergodicity, [etc.] and the identification of entropy was made only at the end, by comparison of the resulting equations with the laws of phenomenological thermodynamics. Now, however,* ***we can take entropy as our starting concept, and the fact that a probability distribution maximizes the entropy subject to certain constraints becomes the essential fact which justifies use of that distribution for inference****."*
>
> — Jaynes (1957)

adapted from Edwin T. Jaynes, "Information theory and statistical mechanics"

4 Maximum Entropy: Discrete case

- We require a distribution over $X = \{1, \ldots, C\}$
- Suppose we know the value of $\mathbb{E}[\boldsymbol{\phi}(x)] = \mathbf{F}$ under the **true** probability distribution for some

$$\boldsymbol{\phi}(x) = [\phi_1(x), \phi_2(x), \ldots, \phi_k(x)]$$

- For example, we may have a lot of data, and can therefore estimate **F** reliably
- By the **maximum entropy principle**, we want to find a

$$J(p) = -\sum_{x \in X} p(x) \log p(x)$$

subject to the following constraints

$$p(x) \geq 0, \quad \forall x \in \mathcal{X} \qquad \textbf{Non-negativity}$$

$$\sum_{x \in \mathcal{X}} p(x) = 1 \qquad \textbf{Sum-to-one}$$

$$\mathbf{F} = \sum_{x \in \mathcal{X}} p(x) \phi_k(x) \qquad \textbf{User-defined constraint}$$

adapted from Kevin P. Murphy, Machine Learning: A Probabilistic Perspective

4 Maximum Entropy: Discrete case

- We can maximize our constrained objective using **Lagrange multipliers**
- We can simplify our problem by treating **p** as a fixed length vector (since our distribution is discrete)

(The non-negativity constraint is implicit)

$$J(p,\lambda) = -\boxed{\sum_x p(x)\log p(x)} + \boxed{\lambda_0\left(1-\sum_x p(x)\right)} + \boxed{\sum_k \lambda_k\left(F_k - \sum_x p(x)\phi_k(x)\right)}$$

entropy (original objective) | **sum-to-one constraint (Lagrange multiplier)** | **user-defined constraint (Lagrange multiplier)**

- Hence $\frac{\partial}{\partial p(x)} J(p,\lambda) = -1 - \log p(x) - \lambda_0 - \sum_x \lambda_k \phi_k(x)$

- Setting this to zero we obtain

$$\log p(x) = -1 - \lambda_0 - \sum_k \lambda_k \phi_k(x) = \frac{1}{Z}\exp\left(\sum_k \lambda_k \phi_k(x)\right)$$

- where $Z = \exp(1+\lambda_0)$

adapted from Kevin P. Murphy, Machine Learning: A Probabilistic Perspective

4 Maximum Entropy: Discrete case

- Enforcing the sum-to-one constraint, we find that $Z = \sum_x \exp(-\sum_k \lambda_k \phi_k(x))$

- so the discrete distribution that maximizes the entropy w.r.t. known $\mathbb{E}[\boldsymbol{\phi}(x)]$ is

$$p(x) = \frac{1}{\sum_x \exp(-\boldsymbol{\lambda}^\top \boldsymbol{\phi}(x))} \exp\left(\boldsymbol{\lambda}^\top \boldsymbol{\phi}(x)\right)$$

  which is in the exponential family!

- Notice that this is agnostic to $\boldsymbol{\phi}(x)$, so if we have

$$\phi(x) = [\mathbb{I}[x = 1], \mathbb{I}[x = 2] \ldots \mathbb{I}[x = C - 1]]^\top$$

- **We recover the exponential family form of the categorical distribution**

adapted from Kevin P. Murphy, Machine Learning: A Probabilistic Perspective

ETH zürich

Sneak Preview of Next Lecture

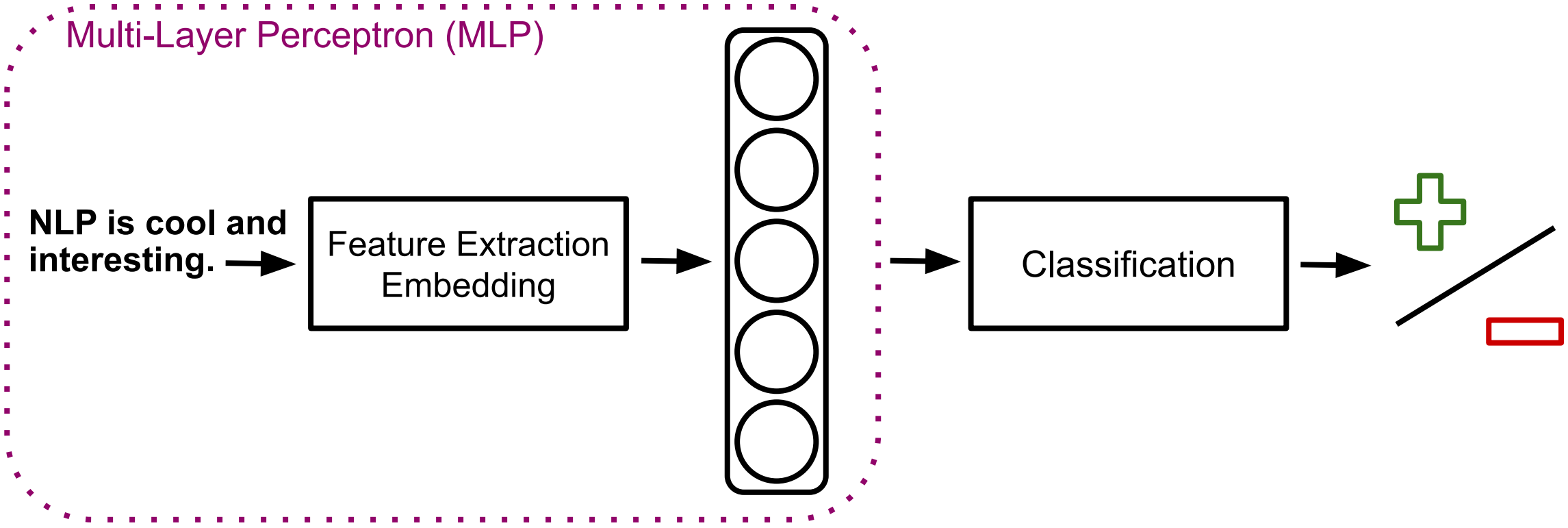
Sentiment Analysis with Multi-layer Perceptrons

How would rate the following sentences and how do you come up with your rating?

NLP is cool.
NLP is not cool.
NLP is cool and interesting.

→ **Fool a Sentiment Analyzer:**

http://text-processing.com/demo/sentiment/



ETH zürich

Fin