The entropy of a distribution $P(x)$ is

$$H[P] = - \sum_x P(x) \log P(x).$$

It is a measure of the information obtained by observing a sample $x$ from $P(x)$.

**Examples:** Let $x \in \{x_1, \ldots, x_N\}$

ie. $x$ has a finite set of values.

**Case (i):** Suppose $P(x=x_i) = 1$, $P(x=x_j) = 0$, $i \neq j$

There is no uncertainty, we know that the sample from $P(x)$ will be $x_i$ before we observe it.

Entropy

$$H[P] = - \sum_{i=1}^N P(x=x_i) \log P(x=x_i)$$

$$= - \left\{ 1 \log 1 + (N-1) 0 \log 0 \right\}$$

$$= 0.$$ 

no information is gained by making the observation.

(Note: $0 \log 0$ is defined by $\lim_{x \to 0^+} x \log x = 0$.)

**Case (ii):** $P(x=x_i) = \frac{1}{N}$, $i = 1, \ldots, n$.

All samples are equally likely.

$$H[P] = - N \left( \frac{1}{N} \log \left( \frac{1}{N} \right) \right) = \log N.$$ 

we gain $\log N$ bits of information by making the observation.
(2) **Shannon** proposed that we encode information for transmission so that events \( x \) will be encoded by \(- \log p(x)\) bits

(i.e. high probability \( x \) have short codes,
low probability \( x \) have long codes).

Then \(- \frac{1}{x} \sum x p(x) \log p(x)\) is the expected coding length of event \( x \) from distribution \( p(x) \).

**Shannon** considered the task of modeling the **English language**

\[ \rightarrow \text{sequences of letters}. \]

**What is the Entropy of English?**

How does entropy relate to the learning ideas we discussed last time?

\[ p(x|\lambda) = \frac{1}{Z(\lambda)} e^{-\phi(x)} \]

Entropy = \(- \frac{1}{x} \sum x p(x|\lambda) \log p(x|\lambda)\)

\[ = - \frac{1}{x} \sum x \phi(x)p(x) + \log Z(\lambda), \]

Suppose we select \( \lambda \) by minimizing the entropy

such that \( \frac{1}{x} \sum x p(x) = \hat{y} \) = observed state

Then \( \hat{\lambda} \) is exactly the **ML estimate** \( \hat{\lambda}_M \).
(3) **Maximum Entropy Principle:**

Given statistics $f(x)$ with observed value $\Psi$, choose the distribution $p(x)$ to maximize the entropy subject to constraints by Lagrange multipliers:

$$-\sum_x p(x) \log p(x) + \mu \left( \sum_x p(x) - 1 \right) + \lambda \left( \sum_x p(x) f(x) - \Psi \right)$$

$$\frac{d}{dp(x)} \left( -\log p(x) - 1 + \mu + \lambda \cdot f(x) \right) = 0$$

Solution: $p(x|\mathcal{Z}) = \frac{e^{\lambda f(x)}}{Z_{\mathcal{Z}}}$

where $\lambda, Z_{\mathcal{Z}}$ are chosen to satisfy the constraints:

$$\sum_x p(x|\mathcal{Z}) = 1$$

$$\sum_x p(x|\mathcal{Z}) f(x) = \Psi$$

The maximum entropy principle recovers an exponential distribution.
4. An alternative viewpoint on ML learning of distributions. This gives deeper understanding.

Suppose the data is generated by a distribution \( f(x) \).

Define the Kullback-Leibler divergence between \( f(x) \) and the model \( p(x|\theta) \).

\[
D(f \| p) = \sum_x f(x) \log \frac{f(x)}{p(x|\theta)}
\]

KL has the property that

\[
D(f \| p) \geq 0 \quad \forall f, p
\]

\[
D(f \| p) = 0, \quad \text{if and only if } f(x) = p(x|\theta).
\]

So \( D(f \| p) \) is a measure of the similarity between \( f(x) \) and \( p(x|\theta) \).

We can write

\[
D(f \| p) = \sum_x f(x) \log f(x) - \sum_x f(x) \log p(x|\theta)
\]

\(
\text{space of all distributions } p(x|\theta)
\)

\( \text{independent of } \theta \)

\( \text{depends on } \theta \)
Now suppose we have samples (i.i.d.) \( x_1, \ldots, x_n \) from \( f(x) \).

\[ f(x) = \frac{1}{N} \sum_{i=1}^{N} S_{x_i} x_i \]

This gives us an empirical distribution.

The KL divergence between \( f_{emp}(x) \) and \( p(x|\theta) \) can be written as:

\[ J(\theta) = -\sum_x f_{emp}(x) \log p(x|\theta) + K \]

\[ J(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \log p(x_i|\theta) + K \]

Minimizing \( J(\theta) \) w.r.t. \( \theta \) finds the distribution \( p(x|\theta) \) which is closest to \( f_{emp}(x) \).

But minimizing \( J(\theta) \) w.r.t. \( \theta \) is exactly ML.

\[ \theta = \arg \max_{\theta} \sum_{i=1}^{N} \log p(x_i|\theta) \]

So ML has meaning even if best fit.

To the model. Every model is only an approximation.
Two important papers (Della Pietra et al., Zhu et al.) show how this principle is useful for learning complicated models.

(Della Pietra paper is available on the website.)

For a real problem, we do not know which statistics to use. (E.g., stats of letters, letter pairs, etc.)

But we can define a large class of possible statistics:

\[ \phi_1(x), \ldots, \phi_c(x) \]

where \( c \) is very large.

Two problems:

(i) Selection: which of these statistics to use.

(ii) Parameter estimation: how to determine the 2 parameters of the model.
Example: Della Pietra et al. (Zhu, Wu, Mumphrey)

Want a model to generate strings of

text \( w = (w_1, w_2, \ldots, w_n) \)

Each \( w_i \) can be a letter, a space, or commas, etc.

First order model:

Simplest model: (lower case letters only)

\[
p(w_1^n) = \frac{1}{Z} \exp \left( \sum_{i=1}^n \lambda_{[a-z]}(w_i) \right)
\]

\( Z = \sum_{w} \exp \left( \sum_{i=1}^n \lambda_{[a-z]}(w_i) \right) \)

\( \lambda_{[a-z]}(w_i) \) is the potential of \( w_i \)

Length of string is generated by another process

Represent as a graph.

Distribution is independent on letter

\[
p(w_1^n) = \prod_{i=1}^n p(w_i)
\]

Learning the parameters corresponds to

Learning the frequencies of letters.
(3) This model does not fit the data (typical words). Because the letters are not independent — i.e. "t" is often followed by "h", "q" is almost followed by "u".

So second-order model using a new statistic: 

\[ X \sum_{i=2}^{n} \mathbb{I}_{a_i \neq a_{i-1}}(w_{ij}) \] — indicator function of adjacent letters.

But there are many other features we could use:

Stochastic sampling from the model is a generative way to check the model. Samples from first-order model xevo, iijjir, samples from second order model was, rease, in,

Graphical representation

\[
P(w_i) = \frac{1}{Z[]} e^{\sum \phi_i(w)} \quad \text{with} \quad \phi_i(w) = \sum_{\omega_i} \delta_{\omega_i}(w_i)
\]

\[
\sum_{M=1}^{M} \mathbb{I}_{\mu} \phi_{\mu}(w) = \sum_{M=1}^{M} \mathbb{I}_{\mu} \delta_{\mu}(w) = \sum_{M=1}^{M} \mathbb{I}_{\mu} \phi_{\mu}(w)
\]

\[
P(w) = \frac{1}{Z[]} e^{\sum_{i=1}^{N-1} \mathbb{I}_{X_i}(w_i)}
\]

Pairwise connection.
The task of learning is to minimize
\[ \log \mathbb{E}[\mathcal{Z}] - \mathcal{L} \] w.r.t. \( \mathcal{Z} \).

Equivalently, minimize
\[ G = \mathbb{E} \mathcal{Z} e^{-\mathcal{L}} \]
Write:
\[ G = \sum_x \mathbb{E}_{\mathcal{Z}} \sum_{\phi_x} \prod_{\lambda} \mathbb{E}_{\phi_x} (x) \mathbb{E}_{\phi_x} (y) \]
Initialize this by setting \( \sum_{\mu} \phi_{\mu} = 0, \quad \forall \mu \)
(i.e. no statistics are selected)

Minimize \( G \) by a "feature pursuit" strategy. (This will be used later in the course also).

At turn \( t \), we have state \( (\mathcal{Z}^t, (\phi^t_{\mu}, \mu = 0)) \)

\( (A) \) Minimize w.r.t. each \( \mathcal{Z}^t_{\mu} \) separately to solve for \( \mathcal{Z}^{t+1} \)
by solving \( \frac{\partial G}{\partial \mathcal{Z}^t_{\mu}} = 0 \) keep other \( \mathcal{Z}^t_{\mu} \) fixed.

\( (B) \) Select the feature \( \mu \), that

\[ \text{minimize } G \left( \mathcal{Z}^t_{\mu}, \mathcal{Z}^{t+1}, \mu \neq \mu \right) \]

\( (C) \) Update \( \mathcal{Z}^{t+1}_{\mu} \rightarrow \mathcal{Z}^{t+1}_{\mu} \).

Problem: both stages (A) & (B) are difficult.
New Page: Simply steps (A) & (B)

Minimize \( G[\phi, \psi] \) w.r.t. \( \Delta u \), with \( \lambda \mu \) fixed, \( \mu \neq 0 \).
\[ \Delta u = \Delta_{\mu}^+ + \Delta_{-\mu}^- \]
Write as
\[ \frac{\partial G}{\partial \Delta u} = \frac{\partial}{\partial \Delta u} \sum_x \left( \phi_x \psi_{-\mu} \right) e^{\Delta u \cdot \phi_x} e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} e_{\lambda \mu}^{-\frac{1}{\lambda \mu}} \phi_x \psi_{-\mu} \]

Hence \( \frac{\partial G}{\partial \Delta u} = 0 \), is equivalent to solving

(A) \[ \sum_x \left( \phi_x \psi_{-\mu} \right) e^{\Delta u \cdot \phi_x} e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} p(x | \lambda \mu) = 0 \], for \( \Delta u \)

We can use a similar technique to represent
\[ G[\phi, \psi] = \sum_x e^{\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} e^{\Delta u \cdot \phi_x} e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} e_{\lambda \mu}^{-\frac{1}{\lambda \mu}} \phi_x \psi_{-\mu} \]

where \( \Delta u = \Delta_{\mu}^+ \), for all \( \mu \neq 0 \), \( \Delta_0 = \Delta_{\mu}^+ + \Delta_{-\mu}^- \).

Hence \( G[\phi, \psi] = \left( G[\phi, \psi] e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} \right) e^{-\Delta u \cdot \phi_x} p(x | \lambda \mu) \)

Indep of \( \lambda \mu \).

(B) To minimize \( G[\phi, \psi] \) w.r.t. \( \mu \), need to minimize
\[ \sum_x e^{\Delta u \cdot \phi_x} e^{-\frac{1}{\lambda \mu} \phi_x \psi_{-\mu}} p(x | \lambda \mu) \]
But there are some situations where this is not difficult. (See Della Pietra et al.)

Let the features \( \phi \mu(x) \) be binary-valued.

Define: \( X^+_\nu = \{ x : \phi \nu(x) = +1 \} \)
\( X^-_\nu = \{ x : \phi \nu(x) = -1 \} \).

Express (1) as:

\[
\sum_{x \in X^+_\nu} (1 - \psi \nu) e^{\Delta \nu} p(x | \mathcal{Z}^{x-\mu}) + \sum_{x \in X^-_\nu} (-1 - \psi \nu) e^{-\Delta \nu} p(x | \mathcal{Z}^{x+\mu})
\]

\[
\Delta \nu = -\frac{1}{2} \log \left\{ \frac{(1 + \psi \nu) \sum_{x \in X^-_\nu} p(x | \mathcal{Z}^{x+\mu})}{(1 - \psi \nu) \sum_{x \in X^+_\nu} p(x | \mathcal{Z}^{x-\mu})} \right\}
\]

Express (2) as:

Select \( \nu \) to minimize:

\[
\sum_{x \in X^+_\nu} e^{\Delta \nu} p(x | \mathcal{Z}^{x-\mu}) + e^{-\Delta \nu} \sum_{x \in X^-_\nu} p(x | \mathcal{Z}^{x+\mu})
\]

So we need to evaluate:

\[
\sum_{x \in X^+_\nu} p(x | \mathcal{Z}^{x-\mu}) \quad \text{and} \quad \sum_{x \in X^-_\nu} p(x | \mathcal{Z}^{x+\mu})
\]
(12) It remains to compute:

\[
\sum_{x \in \mathbf{X}_n^+} P(x \mid \mathbf{z}_n^+) \quad \text{for every } n.
\]

(The sum of both of them is 1)

If the distribution \( P(x \mid \mathbf{z}_n^+) \) is defined over a finite set of states (Della Pietra et al. then these can be computed easily.)

Otherwise: Stochastic Samplng (e.g. Jun Liu's Book, Stat 202C)

Key Point: Stochastic Sampling is very effective at estimating summation integrals in high dimensions.

Also, for this problem, we only need to sample from \( P(x \mid \mathbf{z}_n^+) \) once to get some \( \{x_1, \ldots, x_m\} \quad m \gg 1 \)

Then we can compute \( \mathbf{y} \) from these samples without needing to sample separately for each \( n \).
Penalizing Additional Features.

If our algorithm selects a feature $v$ that is already non-zero, then we are merely adjusting the parameter $\beta_v$ of that feature.

But if it selects a feature $v$ with $\beta_v = 0$, then we are activating a feature that has not been used yet.

It can be good to penalize the number of features used. There are a variety of criteria used to do this, AIC, BIC, $K N$ or $K \log N$. $K$ - constant.
(14) Model Selection

To compare two models:

Model 1:

\[ P(\mathbf{x} | \theta, \text{Model 1}), \quad P(\theta | \text{Model 1}) \]

\[ P(\mathbf{x} | \phi, \text{Model 2}), \quad P(\phi | \text{Model 2}) \]

\[ P(\mathbf{x} | \text{Model 1}) = \sum_{\theta} \frac{P(\mathbf{x} | \theta, \text{Model 1})}{P(\theta | \text{Model 1})} \]

\[ P(\mathbf{x} | \text{Model 2}) = \sum_{\phi} \frac{P(\mathbf{x} | \phi, \text{Model 2})}{P(\phi | \text{Model 2})} \]

Then use the log-likelihood ratio test:

\[ \log \frac{P(\mathbf{x} | \text{Model 1})}{P(\mathbf{x} | \text{Model 2})} \geq T \leq \text{Threshold} \]

Observe that this takes into account all the different parameter values that could generate the data.

Grind an Occam's razor - penalizes model with unnecessary large no. of parameters.
Minimax versus Model Selection.

Zhu et al. proposed learning distributions by minimax.

(1) Use maximum entropy to learn a set of distributions using different statistics.

\[ p(x | z) = \frac{1}{Z(z)} \exp \sum \phi_i(x) \]

\( \phi_i(x) \) are chosen such that

\[ \psi_i = \sum \phi_i(x) p(x | z) \]

\( \psi_i \) are observed statistics.

(2) Use minimum entropy to select between different distributions — i.e. to compute

\[ p(x | z_1) = \frac{1}{Z(z_1)} \exp Z z_1 \phi_1(x) \]

\[ p(x | z_2) = \frac{1}{Z(z_2)} \exp Z z_2 \phi_2(x) \]

Entrophy is

\[ -\sum p(x | z) \log p(x | z) \]

\[ = \log Z(z) - E. \psi \]
(10) Hence using the minimum entropy principle is like maximum likelihood.

Both involve minimizing

$$\log Z[A] - 2 \times$$

Consider:

$$P(x | \hat{X}_1, \hat{X}_2) = \frac{e^{\zeta \cdot \phi_1(x)}}{Z[A]}$$

Using minimum entropy is choosing

$$\left( \hat{X}_1 = \hat{X}_2 = 0 \right) \text{ over } \left( \hat{X}_1 = 0, \hat{X}_2 = 0 \right)$$

provided $$P(x | \hat{X}_1, 0) > P(x | 10, \hat{X}_2)$$.

This differs from model selection. When you should compare

$$\log \frac{\sum_{\hat{X}_1} P(x | \hat{X}_1) P(\hat{X}_1)}{\sum_{\hat{X}_2} P(x | \hat{X}_2) P(\hat{X}_2)}$$
Occam Factor: Plackett.

Automatic selection of complexity of the model.

\[ P(D | \text{Model 1}) \]

\[ P(D | \text{Model 2}) \]

Suppose Model 1 is flexible \( \Rightarrow \) i.e., it can model a lot of datasets.

But Model 2 is more specific

Then model selection will automatically favor the more specific model.

The Occam factor: