

CS195-5 : Introduction to Machine Learning

Lecture 22

Greg Shakhnarovich

November 6, 2006

Announcements

- When is $\mathbf{X}^T \mathbf{X}$ invertible?..

Review: the EM algorithm

- Observed data X , hidden variables Z .
 - E.g., *missing data*.
- Complete data log-likelihood: $\ell(X, Z; \theta)$
- Initialize θ^{old} , and iterate until convergence:

E-step: Compute the expected likelihood as a function of θ .

$$Q(\theta; \theta^{old}) = E_{p(Z | X, \theta^{old})} [\ell(X, Z; \theta) | X, \theta^{old}]$$

M-step (regularized): Compute

$$\theta^{new} = \operatorname{argmax}_{\theta} \{ Q(\theta; \theta^{old}) + \log p(\theta) \}$$

Model selection with BIC

- Let $\pi(\mathcal{M})$ be the number of free parameters in the model \mathcal{M} .
 - For a MoG model with k components in \mathbb{R}^d :

$$\pi(\mathcal{M}) = k \cdot (d + d(d + 1)/2) + k - 1.$$

- For each model, we find ML (or MAP) estimates of the parameters on $X_N = [\mathbf{x}_1, \dots, \mathbf{x}_N]$:

$$L^*(\mathcal{M}) \triangleq \max_{\theta_{\mathcal{M}}} \log p(X_N | \mathcal{M}; \theta_{\mathcal{M}}).$$

- The BIC score for the model \mathcal{M} on data X_N :

$$BIC(\mathcal{M}) = L^*(\mathcal{M}) - \frac{\pi(\mathcal{M})}{2} \log N.$$

Plan for today

- Model selection and connection to coding and information.
 - Minimum description length principle

Learning as communication

- Suppose we want to *communicate* the data set X_N .
- The receiver knows the model class $\mathcal{M}(\theta)$.
- We need to communicate: $\hat{\theta}$ and the prediction *errors*.
- The goal of learning: find the most efficient way of communicating this information.
 - A thought experiment: suppose we have a N -component mixture, with zero covariance Gaussians centered on each data point.
 - No prediction errors!

Learning as communication

- Suppose we want to *communicate* the data set X_N .
- The receiver knows the model class $\mathcal{M}(\theta)$.
- We need to communicate: $\hat{\theta}$ and the prediction *errors*.
- The goal of learning: find the most efficient way of communicating this information.
 - A thought experiment: suppose we have a N -component mixture, with zero covariance Gaussians centered on each data point.
 - No prediction errors! But need to send the N means and – do not gain anything

Learning as communication

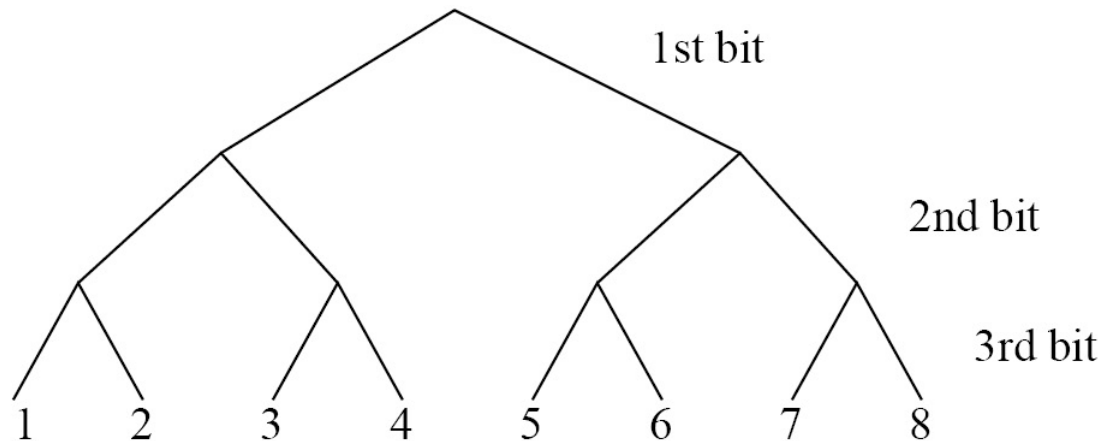
- Suppose we want to *communicate* the data set X_N .
- The receiver knows the model class $\mathcal{M}(\theta)$.
- We need to communicate: $\hat{\theta}$ and the prediction *errors*.
- The goal of learning: find the most efficient way of communicating this information.
 - A thought experiment: suppose we have a N -component mixture, with zero covariance Gaussians centered on each data point.
 - No prediction errors! But need to send the N means and – do not gain anything \Rightarrow haven't learned anything.

Coding: example

- Suppose we had an alphabet with 8 letters; each letter appears with probability $1/8$.
- How many bits do we need to code an n -letter message?

Coding: example

- Suppose we had an alphabet with 8 letters; each letter appears with probability $1/8$.
- How many bits do we need to code an n -letter message?



- three bits per letter $\Rightarrow 3n$ bits total.

Optimal coding

- Suppose we had an alphabet with m letters a_1, \dots, a_m
- Probabilistic model of the language: for a letter A , $p(A = a_i) = p_i$.
- Need to encode n -letter message;
 - General idea for *optimal* code: encode most *frequent* words with *shortest* keywords.
- Example: Huffman's code. Suppose $p(a) = 1/2, p(b) = p(c) = 1/4$.
 - Code words: $a \rightarrow 0, b \rightarrow 10, c \rightarrow 11$.

Optimal codelength

- Shannon's optimal code: codelengt for letter a is has length $l(a) = \log 1/p(a)$.
 - If the probabilities are not powers of two, we will incur some cost: $\lceil \log 1/p_i \rceil$
- Asymptotically, as $n \rightarrow \infty$, the expected code length is

$$\frac{1}{n} E_{a \sim p} \left[\sum_{i=1}^n l(a_i) \right] = \sum_{i=1}^m p_i \log \frac{1}{p_i}$$

bits per letter, assuming i.i.d. letters.

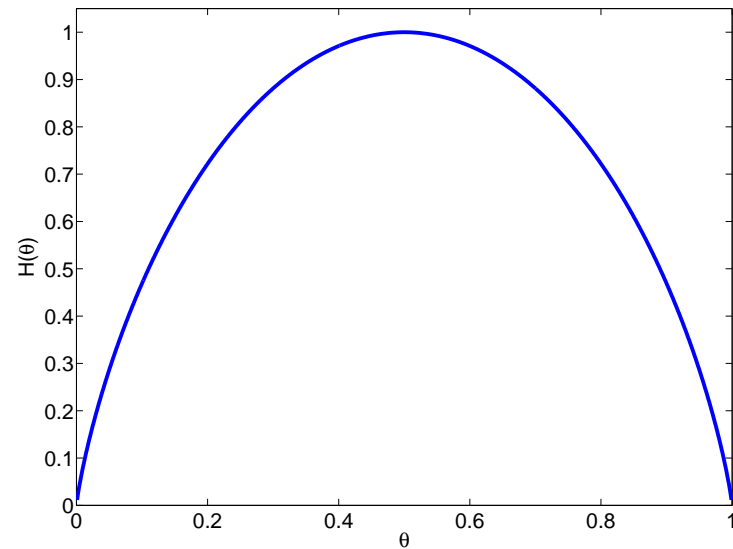
- The quantity

$$H(A) \triangleq \sum_{i=1}^m p_i \log \frac{1}{p_i} = - \sum_{i=1}^m p_i \log p_i$$

is the *entropy* of the random variable A .

Entropy as a measure of uncertainty

- Entropy $H(A) = -\sum_{i=1}^m \log p(A = a_i)$ gives the amount of information *gained* from observing an instance of A .
 - Measured in *bits* (if using \log_2) or *nats* if \log_e
- Example: Bernoulli, $A \in \{0, 1\}$, $p(A = 1) = \theta$.
 - Highest entropy: fair coin.
 - Lowest entropy: fully biased coin.



Coding of real numbers

- With real-valued messages the code length depends on precision.
 - Precision means “the number of values we want to distinguish”;
 - Precision m means that we can discretize the real-valued A into m bins;
 - Will need $\log m$ nats to encode one value, assuming all values are equal.
- A more precise treatment:
 - Compute p_i as the probability that the real value A falls into the i -th bin.
 - The expected optimal code length is again

$$- \sum_{i=1}^m p_i \log p_i.$$

Description length: data

- We have a generative model $p(\mathbf{x}|\theta)$, that for the given data set produces log-likelihood

$$\ell(X_N|\theta) = \sum_{i=1}^N \log p(\mathbf{x}_i|\theta).$$

- We can build the code for the data, assuming $p(\mathbf{x}|\theta)$ is the *true* distribution.
 - If the receiver knows θ , this is all we need to send!
- The *description length* of the message containing the data:

$$\sum_{i=1}^N \log \frac{1}{p(\mathbf{x}_i|\theta)} = - \sum_{i=1}^N \log p(\mathbf{x}_i|\theta) = -\ell(X_N|\theta).$$

Description length: data and parameters

- However the receiver can't know θ - we need to also transmit $\hat{\theta}$.
- We will discretize θ into $1/\sqrt{N}$ distinct values
 - Intuitive argument: with N data points, the estimation error for $\hat{\theta}$ is roughly $1/\sqrt{N}$.
 - Will need $\log \sqrt{N}$ nats to encode *each component* of $\hat{\theta}$
- Total description length with k parameters:

$$DL(X_N, \hat{\theta}) \approx - \sum_{i=1}^N \log p(\mathbf{x}_i | \hat{\theta}) + k \log \sqrt{N}$$

- minimizing MDL \Rightarrow maximizing BIC score.

BIC for classification

- A similar communication setup:
 - The receiver knows the model class and the inputs $\mathbf{x}_1, \dots, \mathbf{x}_N$.
 - We need to send $\hat{\theta}$ and the classification errors $\hat{y} - y$.
- Again, a trivial solution: memorize the data (NN classifier)
 - (almost) zero DL for errors, but high DL for the parameters.
- Here the code length is given by *conditional entropy*

$$H(y | \mathbf{x}) = - \sum_{c=1}^C p(y = c | \mathbf{x}) \log p(y = c | \mathbf{x})$$

- The BIC score:

$$\sum_{i=1}^N \log p(y_i | \mathbf{x}_i, \hat{\theta}) - \frac{\pi(\mathcal{M})}{2} \log N.$$

Learning and coding

- Suppose we have a random discrete variable X with distribution p , $p_i \triangleq \Pr(X = i)$, $i = 1, \dots, m$.
- Optimal code (knowing p) has expected length per observation

$$L(p) = - \sum_{i=1}^m p_i \log p_i.$$

- Suppose now we *think* (estimate) the distribution is $\hat{p} = q$.
 - We build code with codeword lengths $-\log q_i$;
 - The expected length is

$$L(q) = - \sum_{i=1}^m p_i \log q_i.$$

KL divergence

- The cost of estimating p by q :

$$\begin{aligned} D_{KL}(p||q) &\triangleq L(q) - L(p) = -\sum_{i=1}^m p_i \log q_i + \sum_{i=1}^m p_i \log p_i \\ &= \sum_{i=1}^m p_i (\log p_i - \log q_i) \\ &= \sum_{i=1}^m p_i \log \frac{p_i}{q_i}. \end{aligned}$$

called the *Kullback-Leibler divergence* between p and q

- A result from information theory:
 - For any p, q , $D_{KL}(p||q) \geq 0$, with $D_{KL}(p||p) = 0$.
 - I.e., the cost is lowest (zero) when we estimate $\hat{p} = p$.

Next time

Clustering.