10-704: Information Processing and Learning **Spring 2015**

Lecture 4: Jan 22

Lecturer: Akshay Krishnamurthy *Scribes: Akshay Krishnamurthy, Yipei Wang*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

4.1 Some Applications

4.1.1 Sensor Placement

Last time we looked at the maximum mutual information subset selection problem and its application to sensor placement.

$$\max_{S \subset [p]} I(X_S; X_{[p] \setminus S})$$

We maximized mutual information between the sensor locations (treated as random variables) to find a near-optimal choice of k sensors. In the original paper, the authors used a gaussian process to model the random variables and they therefore had analytical expressions for the mutual informations. Given data $X_1, \ldots, X_n \in \mathbb{R}^p$ (we have p possible sensor locations and n samples), they fit a gaussian process to the data (I will not discuss how to do this) and then use that GP for the sensor placement problem.

Another alternative is to simply estimate the mutual information from the data. We will see several techniques for how to estimate this quantity here. The greedy algorithm does not change, we just use estimates for the mutual information instead of the analytical expression.

4.1.2 Nonparametric Forest Density Estimation

Liu, Lafferty and Wasserman [4] apply their entropy estimation results to the task of estimating the structure of tree of forest graphical model under nonparametric assumptions. Here we have a collection of random variables $X^1, \ldots, X^p$, and we have n samples from the joint distribution on these random variables. We assume that the random variables form a tree-structured (or forest-structured) graphical model, where two random variables are conditionally independent given the other variables whenever they do not share an edge. The task is to learn the structure of this tree or forest from the samples. Note that we are making no parametric assumptions about the data.

A tree-structured graphical model is a way to encode independence assertions for a probability distribution. In particular a tree $T = (V, E)$ where the nodes corresponds to the random variables encodes that the distribution factors along its edges.

$$p(x_1, \ldots, x_p) = \prod_{(i,j) \in E} p_{ij}(x_i, x_j)$$

From this, it is easy to see that two nodes are conditionally independent given any node on the path between them.

The point is that if you learn the structure of this tree, it is very easy to estimate the density. You just have to fit several two-dimensional densities. This makes it much easier to do high-dimensional density estimation.

A workhorse algorithm for learning the structure of a tree/forest is the Chow-Liu algorithm, which constructs a maximum-weight spanning tree where the weight on the edge (i, j) is the mutual information $I(X^i; X^j)$. This algorithm relies on the data processing inequality, since in an undirected tree-structured graphical model, every set of three nodes forms a "markov chain" (it is undirected). If the chain looks like $X^i - X^j - X^k$, then the data processing inequality implies that $I(X^i; X^j) \geq I(X^i; X^k)$, so the edge (i, j) will appear in the tree over the edge (i, k).

We will consistently estimate the tree structure provided that we have consistent estimates of the mutual information. Using the identity:

$$I(X; Y) = H(X, Y) - H(X) - H(Y),$$

we just have to estimate this bi-variate and uni-variate entropies, and we will build such an estimator later in the course. With a union bound, we have the following theorem:

Theorem 1. *Let $\hat{F}$ be the estimated s-edge forest graph from the Chow-Liu algorithm. If we choose $h \asymp n^{-1/4}$ and with appropriate smoothness assumptions, then:*

$$\mathbb{P}\left(\hat{F} \neq F_s^\star\right) = O\left(\sqrt{\frac{s}{n}}\right) \text{ whenever } \frac{\log p}{n} \to 0$$

The last condition ensures that we can take a union bound.

Remark 2. *Here we only estimate the entropy of a single variable or a pair of variables. We will see that this becomes much harder for more variables (i.e. the high-dimensional setting), and these high dimensional conditional mutual information estimates are needed to learn more general graph structures. Decide if one set of variables is conditionally independent with another set through mutual information is still an open problem in the high-dimensional setting.*

4.1.3 ML on Distributions

Lastly, there has been a recent line of work referred to as "Machine Learning on Distributions." Here we treat the data points as distributions, and the goal is to do clustering, classification, or regression where the data points are distributions, and we only access them through their samples.

Many ML algorithms can be written just with distances or similarities (via the kernel trick) and so we can use a divergence to specify a distance or a similarity. Two examples are kernel-SVM and spectral clustering, where we just need a measure of similarity between each example (which is a distribution). We can plug an estimate of KL-divergence, L_2-divergence, or anything else, into the heat kernel to define a measure of similarity. Then we can simply run these algorithms.

One caveat is that we may not be preserving positive semidefiniteness of the Gram matrix. One workaround is to simply project the similarity matrix we have onto the PSD cone.

4.2 Discrete Setting

We have a distribution P supported on a finite alphabet $\{1, \ldots, d\}$ with $P(X = j) = p_j$ ($\sum_{j=1}^{d} p_j = 1$). We observe independent samples $\{X\}_{i=1}^{n} \sim P$ and would like to estimate some functional of P, say the entropy:

$$H(P) = -\sum_{j=1}^{d} p_j \log_2(p_j) \tag{4.1}$$

4.2.1 Plugin Estimator

The classical estimator here is the **plugin** estimator. We use the sample to estimate the frequencies, $\hat{p}_j = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[X_i = j]$ and simply plug in these frequencies to obtain:

$$\hat{H}_n = -\sum_{j=1}^{d} \hat{p}_j \log_2(\hat{p}_j) \tag{4.2}$$

Some of the relevant papers here are [1, 2]

Theorem 3.

$$\mathbb{E}\left(\hat{H}_n - H\right)^2 = O\left(\frac{1}{n}\right) \tag{4.3}$$

$$\sqrt{n}(\hat{H}_n - H) \rightsquigarrow \mathcal{N}(0, \sigma^2) \tag{4.4}$$

where $\sigma^2 = \mathrm{Var}(-\log_2 p(X))$. *In particular the plugin estimator achieves the parametric rate.*

Proof. For the whole proof, we'll actually work with the entropy defined using the natural log. At the end, multiply everything through by $\log_2(e)$ to change the base back to base two.

To bound the mean-squared error, we use the bias-variance decomposition.

$$\mathbb{E}\left(\hat{H}_n - H\right)^2 = \mathbb{E}[(\hat{H}_n - \mathbb{E}\hat{H}_n)^2] + \left[\mathbb{E}\hat{H}_n - H_n\right]^2$$

We can show that:

$$\mathbb{E}\hat{H}_n = H - \frac{d-1}{2n}\log_2 e + O(\frac{1}{n^2}) \qquad \mathrm{Var}(\hat{H}_n) = \frac{\sigma^2}{n} + O(\frac{1}{n^2})$$

Both of these bounds are obtained by Taylor expanding $H(\hat{p})$ around $H(p)$ and using standard facts about the moments of binomial random variables (the p_is are binomial draws). The important facts are:

$$\begin{aligned}
\mathbb{E}\hat{p}_j - p_j &= 0 \\
\mathbb{E}\left(\hat{p}_j - p_j\right)^2 &= \frac{p_j(1-p_j)}{n} \\
\mathbb{E}\left(\hat{p}_j - p_j\right)^3 &= \frac{p_j - 3p_j^2 + 3p_j^3}{n^2}
\end{aligned}$$

The fourth moment is also $O(1/n^2)$.

$$H(\hat{p}) = H(p) - \sum_{j=1}^{d}(\hat{p}_j - p_j)(1+\log p_j) - \frac{1}{2}\sum_{j=1}^{d}\frac{(\hat{p}_j-p_j)^2}{p_j} + \frac{1}{6}\sum_{j=1}^{d}\frac{(\hat{p}_j-p_j)^3}{p_j^2}$$
$$- \frac{1}{12}\sum_{j=1}^{d}\frac{(\hat{p}_j-p_j)^4}{((1-\theta)p_j+\theta\hat{p}_j)^3}$$

where $\theta \in [0,1]$. For the bias, the first order term is annihilated. The second and third order terms are:

$$\frac{1}{2}\sum_{j=1}^{d}\mathbb{E}\frac{(\hat{p}_j-p_j)^2}{p_j} = \frac{1}{2}\sum_{j=1}^{d}\frac{(1-p_j)}{n} = \frac{d-1}{2n}$$
$$\frac{1}{6}\sum_{j=1}^{d}\mathbb{E}\frac{(\hat{p}_j-p_j)^3}{p_j^2} = \frac{1}{6N^2}\sum_{j=1}^{d}2p_j - 3 + \frac{1}{p_j} = O(\frac{d}{n^2})$$

By the fact that $\frac{1}{((1-\theta)p_j+\theta\hat{p}_j)^3} \leq \frac{1}{(1-\theta)^3 p_j^3}$, the remainder term is also $O(1/n^2)$.

The variance analysis is quite tedious but it follows along similar lines. Write:

$$\mathbb{E}\left[(\hat{H}_n - \mathbb{E}H)^2\right] = \mathbb{E}\left[(\hat{H}_n - H + \frac{d-1}{2n} + O(1/n^2))^2\right]$$

and then take a third order taylor expansion of $\hat{H}_n$ around H to get:

$$\mathbb{E}\left[\left(-\sum_{j=1}^{d}(\hat{p}_j-p_j)(1+\log p_j) + \frac{d-1}{2n} - \frac{1}{2}\sum_{j=1}^{d}\frac{(\hat{p}_j-p_j)^2}{p_j} + \frac{1}{6}\sum_{j=1}^{d}\frac{(\hat{p}_j-p_j)^3}{(p_j(1-\theta)+\theta\hat{p}_j)^2} + O(1/n^2)\right)^2\right]$$

When you square, the term that matters is the first one:

$$\mathbb{E}\left[\left(-\sum_{j=1}^{d}(\hat{p}_j-p_j)(1+\log p_j)\right)^2\right] = \sum_{j,k=1}^{d}(1+\log p_j)(1+\log p_k)\mathbb{E}(\hat{p}_j-p_j)(\hat{p}_k-p_k)$$
$$= \frac{1}{n}\sum_{j=1}^{d}(1+\log p_j)^2 p_j(1-p_j) - \frac{1}{n}\sum_{j\neq k}(1+\log p_j)(1+\log p_j)p_j p_k$$
$$= \frac{1}{n}\sum_{j=1}^{d}p_j\log^2 p_j - H^2 = \frac{1}{n}\sigma^2$$

All of the other terms are $O(1/n^2)$. □

4.2.2 LP based estimator

A recent line of work by Greg and Paul Valiant shows that you can actually get away with using $O(d/\log(d))$ samples [3]. This is interesting because with this sample complexity it is extremely unlikely that you even see all of the items.

We will sketch some of the main ideas. We transform the sample $\{X_i\}_{i=1}^n$ into a **fingerprint**, which is a vector $f = (f_1, f_2, \ldots)$ for which f_i is the number of elements in the domain that occured i times in the

sample. A fingerprint of a distribution P is a mapping $h_P : (0,1] \to \mathbb{N} \cup \{0\}$ for which $h_P(x)$ is the number of domain elements that occur with probability exactly x. More formally, $h_P(x) = |\{\alpha : p_\alpha = x\}|)$. Notice that the entropy can be written only in terms of the fingerprint:

$$H(P) = -\sum_{x:h_P(x)\neq 0} h_P(x) x \log_2 x \tag{4.5}$$

Example 1. *Consider the following sample from a distribution* $\{a, a, b, c, d, e, e, b, b, b\}$. *This sample has fingerprint* $f = (2, 2, 0, 1)$ *because two symbols occur once, two symbols occur twice and one symbol occurs four times. For a distribution with* $P(a) = 1/2, P(b) = 1/16, P(c) = 1/8, P(d) = 1/4, P(e) = 1/16$ *the histogram has:*

$$h_P(1/16) = 2, h_P(1/18) = h_P(1/4) = h_P(1/12) = 1$$

Consider the element j, which occurs with probability p_j. If we draw n samples from the population, the number of occurences of j is distributed as Binomial(n, p_j). Letting $B(n, p, i) = \mathbb{P}[X = i]$ when $X \sim$ Binomial(n, p), by linearity of expectation, we have:

$$\mathbb{E}[f_i] = \sum_{j=1}^{d} B(n, p_j, i) = \sum_{x:h_P(x)\neq 0} h_P(x) B(n, x, i)$$

The basic idea is to invert these linear equations to solve for $h_P(x)$, from which we can compute the entropy. If we discretize $(0, 1]$ to a grid $x_1, \ldots, x_k$ and let h_j be the variable associated with $h_P(x_j)$ we get the following program.

$$\text{minimize} \sum_i \frac{1}{\sqrt{1+f_i}} \left| f_i - \sum_{j=1}^{k} h_j \text{poi}(kx_j, i) \right| \text{ s.t. } \sum_{j=1}^{k} x_j h_j = \sum_i f_i/n, \forall j. h_j \geq 0$$

There are a couple of things going on here. First, we used poisson approximation to the binomial $B(n, p, i) \approx$ Poisson(np, i). The objective is making sure we fit the fingerprint well. The normalization by $\sqrt{1+f_i}$ is an estimate for the standard deviation of f_i, so that normalization puts all of the terms on the same footing. The constraint is just saying that h_j has to correspond to a valid fingerprint.

To resolve ambiguity (there are many distributions that can generate very similar fingerprints), we next aim to find the simplest distribution that could have generated the observe fingerprint. To do this we use the value of the solution to the previous linear program, which we call OPT_1. In particular, we look for a fingerprint with low ℓ_1 norm that has objective value not too far from the best fit in the previous program:

$$\begin{aligned} &\text{minimize} \sum_{j=1}^{j} h_j \\ &\text{subject to} \sum_i \frac{1}{\sqrt{1+f_i}} \left| f_i - \sum_{j=1}^{k} h_j \text{poi}(kx_j, i) \right| \leq OPT_1 + \alpha \\ &\qquad \sum_{j=1}^{k} x_j h_j = \sum_i f_i/n, \forall j. h_j \geq 0 \end{aligned}$$

Analyzing this program, you can show:

Theorem 4. *There exists a constant $C_0 > 0$ and an appropriate setting for α such that given a sample of size $n = c\frac{d}{\log d}$ of independent draws from P over a domain of size at most d, with probability $1 - e^{-n^{\Omega(1)}}$, the two-LPs return a histogram $\hat{h}$ such that $|H(P) - H(\hat{h})| \leq \frac{C_0}{\sqrt{c}}$.*

The proof of this result is quite technical but at a high level, it decomposes into the following steps:

1. Use deviation bounds to argue that the observed fingerprint is not too anomalous.
2. Analyze the linear program and show that the true histogram is a feasible point for the first LP (even when you round to the discretization).
3. Argue that any two feasible points that have low objective value (for the second program) must be close together. Since the true histogram is feasible and has low objective value, this means that whatever you find must be close to the truth.

I will put up references to some of the relevant papers if you are interested. I should also point out that this is the optimal sample complexity for this problem. This lower bound was also shown by Valiant and Valiant, and again the proof is quite interesting.

4.3 Continuous Setting

Here, we assume the distribution P is continuous with density $p = dP/d\mu$. As before we obtain a sample $\{X_i\}_{i=1}^n \sim P$. We would like to estimate something like the (negative) differential entropy:

$$H(p) = \int p(x) \log p(x) d\mu(x)$$

The main assumption is a **smoothness assumption** on the density.

Definition 5. *The class of functions $\Sigma(\beta, L)$ is called the **Holder class**. A function $f : [0,1]^d \to \mathbb{R}$ is in $\Sigma(\beta, L)$ if for all tuples $(r_1, \ldots, r_d)$ with $\sum_i r_i \leq \lfloor \beta \rfloor$ (the largest integer strictly smaller than β) we have:*

$$|D^r f(x+u) - D^r f(x)| \leq L\|u\|^{\beta - |r|}$$

where $|r| = \sum_j r_j$.

For example the class $\Sigma(2, L)$ is the class of functions with Lipschitz continuous derivatives.

There are several approaches here

4.3.1 Plugin Estimator

The plugin estimator is conceptually straightforward: estimate the density $\hat{p}$ using something like a kernel density estimator, nearest-neighbor graph, or histogram and plug this into the functional. Specifically with an estimator $\hat{p}$ we use:

$$\hat{H} = \int \hat{p}(x) \log \hat{p}(x) d\mu(x)$$

An interesting aspect of these estimators is that typically we *undersmooth* the density estimate. For kernel density estimation, this means that we choose a bandwidth h that is smaller than the optimal bandwidth for the density estimation problem.

We give some more details for the KDE based entropy estimator [4]. Recall that the density estimator $\hat{p}_h$ using a kernel K and bandwidth h is given by (Think of $K(x) = \exp(-x^2)$.):

$$\hat{p}_h(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\|x - X_i\|}{h}\right)$$

It is a classical result that if p has smoothness β (i.e. belongs to Holder, or Sobolev classes) then:

$$\mathbb{E}\hat{p}_h(x) - p(x) \asymp h^\beta \qquad \mathrm{Var}(\hat{p}_h(x)) \asymp \frac{1}{nh^d}$$

And the mean squared error is equal to the squared bias plus the variance. The MSE is minimized when we choose $h \asymp n^{\frac{-1}{2\beta+d}}$ in which case the MSE is $O(n^{\frac{-2\beta}{2\beta+d}})$.

Liu, Lafferty, and Wasserman [4] show instead that for the entropy functional:

$$|H(\hat{p}_h) - H(p)| \leq \underbrace{|H(\hat{p}_h) - \mathbb{E}H(\hat{p}_h)|}_{\text{Variance}} + \underbrace{|\mathbb{E}H(\hat{p}_h) - H(p)|}_{\text{bias}}$$

$$\mathbb{P}(|H(\hat{p}_h) - \mathbb{E}H(\hat{p}_h)| > \epsilon) \leq 2\exp\left(\frac{n\epsilon^2}{32\kappa^2}\right)$$

$$|\mathbb{E}H(\hat{p}_h) - H(p)| \leq c_1 h^\beta + \frac{c_3}{nh^d}$$

and the bias term is optimized by setting $h \asymp n^{\frac{-1}{\beta+d}}$, so that the bias is $O(n^{\frac{-\beta}{\beta+d}})$. The concentration bound on the variance shows that variance-like term is $O(1/\sqrt{n})$ and putting these together gives the rate:

$$\mathbb{E}[|H(\hat{p}_h) - H(p)|] \leq O_P\left(\frac{1}{\sqrt{n}} + n^{\frac{-\beta}{\beta+d}}\right)$$

Notice that when $\beta > d$, the second term is asymptotically dominated by the first, leading to a parametric convergence rate in this smooth regime.

The main point here is that we chose the bandwidth $h \asymp n^{\frac{-1}{\beta+d}}$ which is *undersmoothing* the density estimate. Typically undersmoothing makes the bias lower and the variance higher, but integration in the entropy functional removes some of this variance.

Some comments:

1. You can also use other forms of plugin estimators. For example, Poczos and Schneider use k-NN based estimators for renyi and other divergences. Interestingly, they show that you can consistently estimate these divergences by keeping k fixed with n. This is essentially undersmoothing, when k is fixed, you have lower bias than usual but higher variance. The k-NN estimator is:

   $$\hat{p}_k(x) = \frac{1}{n} \frac{k}{\mathrm{vol}(B_k(x))}$$

   where B_k is the Euclidean ball centered at x that contains k samples.

2. One downside is that tuning the hyperparameter is tough here. It is not clear how to do something like cross-validation. The problem is that the hyperparameter is not the same one you would use for density estimation.

3. Another downside is that the existing analyses are highly specialized. The proofs for the kNN method are quite different from the techniques outlined here. This contrasts with below, where we get to borrow a lot from existing density estimation results.

4.3.2 Von Mises Estimator

Another approach again uses the taylor expansion but is based on the observation that the linear, quadratic, and cubic terms involve simple functionals of the unknown density p (See [5] and references therein for details). There are some technical details here (you have to define derivatives on functions appropriately) but you can write:

$$\begin{aligned} H(p) &= H(q) + \int (\log(q(x)+1)(q(x)-p(x))dx + O(\|q-p\|_2^2) \\ &= -\int p(x)\log(q(x)) + O(\|q-p\|_2^2) \end{aligned}$$

This is just a functional taylor expansion, but it known as the Von Mises Expansion (since the functionals are defined over the space of distributions).

This expansion motivates the following estimator. Split the data into half, use the first half to construct a density estimate $\hat{p}_h$ and then replace the integral (which is just an expectation) with a sample mean:

$$\hat{H}(p) = \frac{2}{n}\sum_{i=n/2+1}^{n} -\log(\hat{p}_h(X_i))$$

The expansion immediately gives:

$$\hat{H}(p) - H(p) = \frac{2}{n}\sum_{i=n/2+1}^{n} -\log(\hat{p}_h(X_i)) + \int p(x)\log\hat{p}_h(x)dx + O(\|p-\hat{p}_h\|_2^2)$$

The first two terms are just the difference between a sample average and its mean, so it will have squared error of order $O(1/n)$. The second term is just the MSE of the density estimator, which we know is $O(n^{\frac{-2\beta}{2\beta+d}})$. Therefore, if $\beta > d/2$, this first-order corrected estimator has a $n^{-1/2}$ rate. Notice that we set the bandwidth $h \asymp n^{\frac{-1}{2\beta+d}}$ which is the optimal bandwidth for density estimation. This contrasts with the plugin method.

It turns out that the minimax rate for these sorts of problems is typically $n^{-\min\{1/2, \frac{4\beta}{4\beta+d}\}}$. This means that the critical smoothness is $d/4$; if $\beta > d/4$ you can achieve the parametric convergence rate. The estimators that achieve this require higher order corrections. If you make a second order expansion with a third order remainder, you achieve the critical smoothness index, but are suboptimal in the $\beta < d/4$ regime. To be minimax optimal everywhere you have to make a third order expansion with fourth order remainder.

Some other comments:

1. This framework is particularly appealing since you can use whatever density estimator you like. You can use kNN-based, histogram, KDE etc.
2. This is also great because you can tune the hyperparameter (in this case the bandwidth) by cross-validating the density estimator.
3. You can also show other properties for these first-order estimators in some special cases. For example, provided the first order term at the true distribution is not identically zero (for entropy this means p is uniform) you can show asymptotic normality. In this case you can also build confidence intervals by using a plugin estimator for the asymptotic variance.

References

[1] Basharin, GP (1959). On a statistical estimate for the entropy of a sequence of independent random variables. *Theory of Probability & its Applications*.

[2] Antos, András & and Kontoyiannis, Ioannis (2001). Convergence properties of functional estimates for discrete distributions. *Random Structures & Algorithms*.

[3] Valiant, Paul & Valiant, Greg (2013). Estimating the Unseen: Improved Estimators for Entropy and other Properties. *Advances in Neural Information Processing Systems*.

[4] Liu, Han and Wasserman, Larry and Lafferty, John D (2012). Exponential concentration for mutual information estimation with application to forests. *Advances in Neural Information Processing Systems*.

[5] Krishnamurthy, Akshay and Kandasamy, Kirthevasan and Poczos, Barnabas and Wasserman, Larry (2014). Nonparametric Estimation of Renyi Divergence and Friends. *International Conference on Machine Learning (ICML)*.