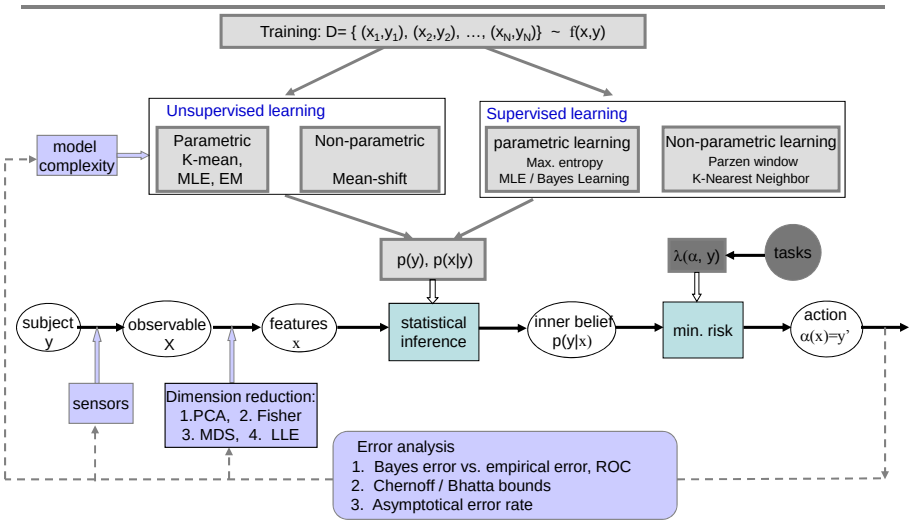
Diagram of Bayesian Method for Pattern Classification



Lecture 16: MLE, Bayes Learning, and Maximum Entropy

Objective : Learning the prior and class models, both the *parameters* and the *formulation (forms)*, from training data for classification.

1. Introduction to some general concepts.
2. Maximum likelihood estimation (MLE)
3. Recursive Bayes learning
4. Maximum entropy principle

Terminology clarification

1. Supervised vs unsupervised learning:

   In supervised learning, the data are labeled manually.

   In unsupervised learning, the computer will have to discover the number of classes, and to label the data and estimate the class models in an iterative way.

2. Parametric methods vs non-parametric methods:

   In a parametric method, the probability model is specified by a number of parameter with more or less fixed length. For example, Gaussian distribution.

   In a non-parametric method, the probability model is often specified by the samples themselves. If we treat them as parameters, the number of parameters often increases linearly with the size of the training data set $|D|$.

3. Frequency vs probability (model):

   For a learning problem, we always assume that there exists an underlying frequency f(x) which is objective and intrinsic to the problem domain. For example the fish length distribution for salmon in Alaska. But it is not directly observable and we can only draw finite set of samples from it.

   In contrast, what we have in practice is a probability p(x) estimation to f(x) based on the finite data. This is called a "model". A model is subjective and approximately true, as it depends on our experience (data), purpose, and choice of models. "*All models are wrong, but some are useful*".

Learning by Maximum Likelihood Estimate (MLE)

In Bayesian decision theory, we construct an optimal decision rule with the assumption that the *prior* and *class conditional probabilities* are known. In this lecture, we move one step further and study how we may learn these probabilities from training data.

Given: a set of training data with labels $D=\{ (x_1,c_1), (x_2,c_2), \ldots, (x_N, c_N) \}$

Goal: to estimate (learn) the prior p(y=i) and conditional probabilities p(x|y=i), i=1,2,…,k.

Basic assumption here:

1). There is an underlying frequency f(x,y) for variables x and w jointly.
   the training data are independent samples from f(x,y).

2). We assume that we know the probability family for p(x|y=i), i=1,2,…,k. Each family is specified by a vector valued parameter θ. --- parametric method.

3). The different class of models can be learned independently. E.g. no correlation between salmon and sea bass in the training data.

Problem formulation

For clarity of notation, we remove the class label, and estimate each class model separately.

Given: A set of training data D={ $x_1, x_2, \ldots, x_N$} as independent samples from f(x) for a class w_i

Objective: Learning a model p(x) from D as an estimation of f(x).

Assumption: p(x) is from a probability family specified by parameters θ.
Denote p(x) by p(x; θ), and the family by Ω_θ Thus the objective is to estimate θ.

Formulation: We choose θ to minimize a "distance measure" between f(x) and p(x; θ),

$$\theta^* = \arg\min_{\theta\in\Omega_\theta} \int f(x) \log \frac{f(x)}{p(x;\theta)} dx$$

This is called the *Kullback-Leibler divergence* in information theory. You may choose other distance measure, such as,

$$\int | f(x) - p(x;\theta) |^2 \, dx$$

But the KL divergence has many interpretations and is easy to compute, so people fall in love with it.

Maximum Likelihood Estimate (MLE)

The above formulation basically gives us an explanation for the popular MLE

$$\begin{aligned}
\theta^* &= \arg\min_{\theta\in\Omega_\theta} \int f(x) \log \frac{f(x)}{p(x;\theta)} dx \\
&= \arg\min_{\theta\in\Omega_\theta} E_f[\log f(x)] - E_f[\log p(x;\theta)] \\
&= \arg\max_{\theta\in\Omega_\theta} E_f[\log p(x;\theta)] \\
&= \arg\max_{\theta\in\Omega_\theta} \sum_{i=1}^{N} \log p(x_i;\theta) + O(\tfrac{\varepsilon}{N})
\end{aligned}$$

In the last step, we replace the expectation (mean) by a sample mean.
The MLE is to find the "best" parameter to maximize the likelihood of the data:

$$\theta^* = \arg\max_{\theta\in\Omega_\theta} \sum_{i=1}^{N} \log p(x_i;\theta)$$

In fact, you should remember that nearly all learning problems start from this formulation !

MLE example

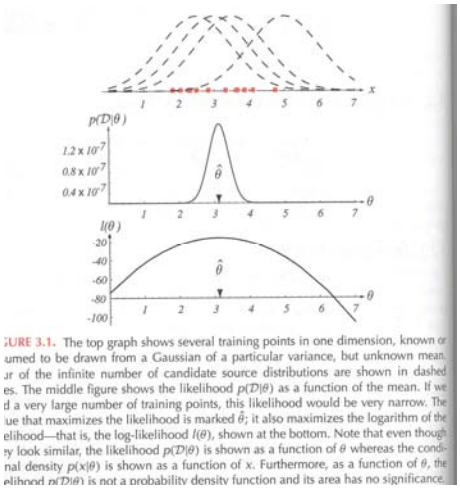
We denote the log-likelihood as a function of θ

$$l(\theta) = \sum_{i=1}^{N} \log p(x_i;\theta)$$

θ∗ is computed by solving equations

$$\frac{dl(\theta)}{d\theta} = 0$$

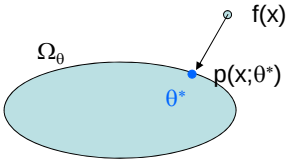
For example, the Gaussian family gives close form solution.



URE 3.1. The top graph shows several training points in one dimension, known or umed to be drawn from a Gaussian of a particular variance, but unknown mean. ır of the infinite number of candidate source distributions are shown in dashed es. The middle figure shows the likelihood $p(\mathcal{D}|\theta)$ as a function of the mean. If we d a very large number of training points, this likelihood would be very narrow. The ue that maximizes the likelihood is marked $\hat{\theta}$; it also maximizes the logarithm of the elihood—that is, the log-likelihood $l(\theta)$, shown at the bottom. Note that even though y look similar, the likelihood $p(\mathcal{D}|\theta)$ is shown as a function of θ whereas the condi- nal density $p(x|\theta)$ is shown as a function of x. Furthermore, as a function of θ, the elihood $p(\mathcal{D}|\theta)$ is not a probability density function and its area has no significance.

MLE Summary

The MLE computes one point in the probability family Ω_θ



It treats q as a quantity. The problems are:

1). It does not preserve the full uncertainty (for example, see the figure in previous page) in estimating q.

2). It is difficult to integrate with new data incrementally. For example, if new data arrive after the MLE, how do we update q?

Bayes Learning

The Bayes learning method takes a different view from MLE. It views θ as a random variable, and thus it estimates a probability distribution of θ. Now, we denote the class probability as $p(x \mid \theta)$, in contrast to $p(x; \theta)$. Instead of computing a single θ^*, we compute the posterior probability from the data set D. As the samples in D are independent, we have

$$p(\theta|D) = p(D|\theta)p(\theta)/p(D) = \prod_{i=1}^{N} p(x_i|\theta)p(\theta)/p(D)$$

In the above equation, $p(\theta)$ is a prior distribution for q. In the absence of *a priori* knowledge on q, we can set it to be a uniform distribution. It is trivial to show that the MLE θ^* is also a maximum posterior estimation.

Recursive Bayes Learning

Suppose that we observe new data set $D^{new} = \{x_{n+1}, \ldots, x_{n+m}\}$ after learning the posterior $p(\theta|D)$, we can treat $p(\theta|D)$ as our prior model and compute

$$\begin{aligned} p(\theta|D^{new}, D) &= p(D^{new}|\theta, D)\, p(\theta \mid D) \ / \ p(D^{new}) \\ &= \prod_{i=N+1}^{m} p(x_i|\theta)\, p(\theta \mid D) \ / \ p(D^{new}) \\ &= \prod_{i=1}^{N+m} p(x_i|\theta)\, p(\theta) \ / \ p(D^{new}, D) \end{aligned}$$

In the above equations, *p(D)* and *$p(D^{new})$* are treated as constants.

Clearly, it is equivalent to MLE by pooling the two datasets D and D^{new}. Therefore when the data come in batches, we can recursively apply the Bayes rule to learning the posterior probability on θ. Obviously the posterior becomes sharper and shaper when the number the samples increases.

Recursive Bayes Learning

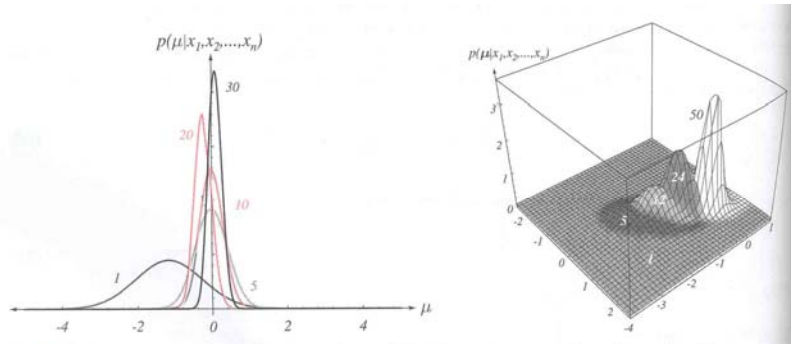


FIGURE 3.2. Bayesian learning of the mean of normal distributions in one and two dimensions. The posterior distribution estimates are labeled by the number of training samples used in the estimation.

Bayes Learning

As we are not very certain on the value of θ, we have to pass the uncertainty of θ to the class model p(x). This is to take an expectation with respect to the probability distribution $p(\theta \mid D)$.

$$p(x \mid D) = \int p(x \mid \theta, D)\, p(\theta \mid D) d\theta$$

This causes a smoothness effect of the class model. When the dataset goes to Infinity, $p(\theta \mid D)$ becomes a a Delta function $p(\theta \mid D) = \delta(\theta - \theta^*)$. Then the Bayes Learning and MLE are equivalent.

The main problem with Bayes learning is that it is difficult to compute and remember a whole distribution, especially when the dimension of θ is high.

Learning the class prior model

So far, we discussed the learning of the class model p(x|y=i) (we put the class label back here). The learning of the class prior probability becomes straight forward.

p(y=1) + p(y=2) + …+ p(y=k) =1

We assume p(w) follows a multi-nomial distribution,

$$\theta = (\theta_1, \theta_2, ..., \theta_k), \qquad \theta_1 + \theta_2 + \cdots + \theta_k = 1$$

Suppose the training set is divided into K subsets and the samples have the same label in each subset

$$D = D_1 \cup D_2 \cup \cdots \cup D_k$$

Then the ML-estimation for θ is,

$$\theta_i = \frac{|D_i|}{|D|}$$

Sufficient statistics and maximum entropy principle

Sufficient statistics and maximum entropy principle

In Bayes decision theory, we assumed that the prior and class models are given.

In MLE and Bayes learning, we learned these models from a labeled training set D, but we still assumed that the probability families are given and only the parameters q are to be computed.

Now we take another step further, and show how we may create new classes of probability models through a maximum entropy principle. For example, how was the Gaussian distribution derived at the first place?

1). Statistic and sufficient statistic
2). Maximum entropy principle
3). Exponential family of models.

Statistic

Given a set of samples D={x_1, x_2, …, x_N}, a statistic s of D is a function of the D, denoted by

$$s = \varphi(D) = \varphi(x_1, x_2 ..., x_N)$$

For example, the mean and variance

$$s_1 = \mu = \frac{1}{N}\sum_{i=1}^{N} x_i \ , \qquad s_2 = \sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2$$

A statistic summarizes important information in the training sample, and in some cases it is sufficient to just remember such statistics for estimating some models, and thus we don't have to store the Large set of samples. But what statistics to good to keep?

Sufficient Statistics

In the context of learning the parameters θ^* from a training set D by MLE, a statistic s (may be vector) is said to be *sufficient* if s contains all the information needed for computing θ^*. In a formal term,

$$p(\theta \mid s, D) = p(\theta \mid s)$$

The book shows many examples for sufficient statistics for the exponential families.

In fact, these exponential families have sufficient statistics, because they are created from these statistics in the first place.

Creating probability families

We revisit the learning problem:

Suppose we are given a set of training examples which are samples from a underlying frequency f(x).

D={x1, x2, …, xN} ~ f(x),

Our goal is to learn a probability p(x) from D so that p(x) is close to f(x). But this time we don't know the form of p(x). i.e. we don't know which family p(x) is from.

We start with computing a number of *n* statistics from D

$$s_j = \frac{1}{N}\sum_{i=1}^{N}\phi_j(x_i), \quad j = 1,2,...,n$$

For example,

$$\phi_1(x) = 1, \qquad \phi_2(x) = x,$$
$$\phi_3(x) = (x-u)^2, \qquad \phi_4(x) = \ln x$$

Creating probability families

As N increases, we know that the sample mean will approach the true expectation,

$$s_j = \frac{1}{N}\sum_{i=1}^{N}\phi_j(x_i) \approx \int f(x)\phi_j(x)dx = E_f[\phi_j(x)], \qquad N \to \infty, j = 1,2,...,n.$$

As our goal is to compute p(x), it is fair to let our model p(x) produces the same expectations, That is, p(x) should satisfy the following constraints,

$$\int p(x)\phi_j(x)dx = E_p[\phi_j(x)] = s_j \approx E_f[\phi_j(x)], \qquad j = 1,2,...,n.$$

Of course, p(x) has to satisfy another constraint,

$$\int p(x)\,dx = 1.$$

Creating probability families

The n+1 constraints are still not enough to define a probability model p(x) as p(x) has a huge number Of degrees of freedom. Thus we choose a model that has maximum entropy among all distributions that satisfy the n+1 constraints.

This poses a constrained optimization problem,

$$p^* = \arg\max - \int p(x) \log p(x)\, dx$$

Subject to:

$$\int p(x)\, \phi_j(x) dx = s_j\ , \qquad j = 1,2,...,n.$$

$$\int p(x)\, dx = 1$$

Lagrange multipliers

We can solve the constrained optimization problem by Lagrange multiplier (you must have studied this in calculus).

The problem becomes to find p that maximizes the following functional,

$$E[p] = -\int p(x) \log p(x)\, dx - \sum_{j=1}^{n} \lambda_j (\int p(x) \phi_j(x) dx - s_j) - \lambda_0 (\int p(x) dx - 1)$$

By calculus of variation, we set $\frac{\delta E[p]}{\delta p} = 0$, and have

$$-\log p - 1 - \sum_{j=1}^{n} \lambda_j \phi_j(x) - \lambda_0 = 0$$

Exponential family

Therefore we obtain a probability model –the exponential family

$$p(x;\,\theta) = e^{-1-\sum_{j=1}^{n}\lambda_j\phi_j(x)-\lambda_0} = \frac{1}{Z}e^{-\sum_{j=1}^{n}\lambda_j\phi_j(x)}$$

Where Z is a normalization constant which makes sure that the probability sums to one.
The parameters are

$$\theta = (\lambda_1, \lambda_2, \ldots, \lambda_n)$$

These parameters can be solved from the constraint equations, or equivalently by MLE.
The more statistics we choose, the model p(x) is closer to f(x). But given the finite data in D, we at least should choose n < N otherwise it is overfitting. In general, n =O(logN)

For example

If we choose two statistics,

$$\phi_1(\mathrm{x}) = x,\ \phi_2(\mathrm{x}) = x^2$$

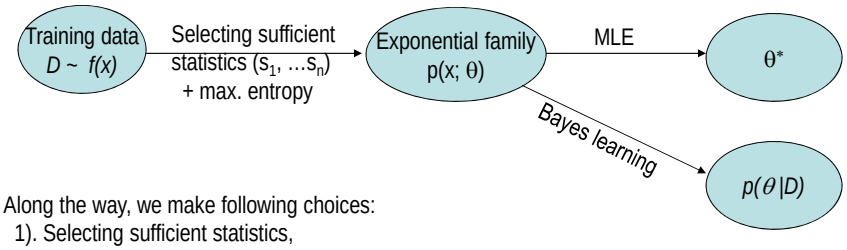
We obtain a Gaussian distribution

$$p(x;\,\theta) = \frac{1}{Z}e^{-\sum_{j=1}^{n}\lambda_j\phi_j(x)} = \frac{1}{Z}e^{-\lambda_1 x-\lambda_2 x^2}$$

Summary

The exponential families are derived by a maximum entropy principle (Jaynes, 1957) under the constraint of sample statistics. Obviously these statistics are the sufficient statistics for The family of model it helps to construct.

The supervised learning procedure is,



Along the way, we make following choices:

1). Selecting sufficient statistics,

2). Using the maximum entropy principle, → exponential families

3). Using the Kullback-Leibler divergence →MLE