Learning with Fully Observed Variables

Handout Notes - Generative and Deep Learning (GDL)

Davide Bacciu - University of Pisa

---

1 Scope and assumptions

We discuss *parameter learning* in probabilistic models under a particularly friendly regime:

> *All variables of interest are observed in the training set (complete data), and the model structure is fixed.*

This is the setting in which many classical learning rules look deceptively simple: for discrete models, maximum likelihood learning typically reduces to *counting occurrences and normalizing*.

The same principles (likelihood, priors, posterior) underpin learning in more complex situations (e.g., missing data or latent variables), but complete-data learning is the best place to build intuition because we can often derive closed-form estimators.

2 Learning as inference over hypotheses and parameters

A probabilistic model can be viewed as a *hypothesis* about how data are generated. We will write a hypothesis as h, and often we will consider a family of hypotheses indexed by parameters θ:

$$h = h_\theta, \qquad \theta \in \Theta.$$

Given observed data $\mathcal{D}$, learning can be understood as inference over θ (equivalently over h_θ). Three common approaches are:

2.1 Bayesian prediction (posterior averaging)

Bayesian learning maintains a *posterior* distribution over parameters and predicts by averaging over that posterior:

$$P(X \mid \mathcal{D}) \;=\; \int P(X \mid h_\theta)\, P(h_\theta \mid \mathcal{D})\, d\theta.$$

This approach explicitly represents parameter uncertainty. It can be very robust with limited data, but the integral is often analytically intractable for realistic models and requires approximation.

2.2 Maximum a posteriori (MAP)

MAP selects the single hypothesis (or parameter value) with highest posterior probability:

$$h_{\text{MAP}} \;=\; \arg\max_{h \in H} P(h \mid \mathcal{D}) \;=\; \arg\max_{h \in H} P(\mathcal{D} \mid h)\, P(h).$$

MAP is often the most useful "bridge" between Bayesian modeling and optimization: it looks like likelihood maximization, but tempered by a prior $P(h)$ which functions as a regularizer, especially important in sparse-data regimes.

2.3 Maximum likelihood (ML)

ML discards the prior and chooses parameters that maximize only the likelihood of the observed data:

$$h_{\mathrm{ML}} \;=\; \arg\max_{h\in H} P(\mathcal{D} \mid h), \qquad \text{equivalently} \qquad \theta_{\mathrm{ML}} \;=\; \arg\max_{\theta\in\Theta} P(\mathcal{D} \mid \theta).$$

ML is widely used as a practical point estimate, particularly when datasets are large. However, ML can produce degenerate estimates when data are scarce.

3 MAP as regularization and compression

A useful way to understand MAP is to look at the log of the objective. Using base-2 logarithms (so values can be interpreted in bits),

$$\arg\max_{h_\theta} P(\mathcal{D} \mid h_\theta)\, P(h_\theta) \iff \arg\min_{h_\theta} \Big[-\log_2 P(\mathcal{D} \mid h_\theta) \;-\; \log_2 P(h_\theta) \Big].$$

The term $-\log_2 P(\mathcal{D} \mid h_\theta)$ measures *lack of fit* to data: models that assign low probability to observed data incur a larger penalty. The term $-\log_2 P(h_\theta)$ penalizes hypotheses that are unlikely under the prior; it can be interpreted as a *description length* for the model itself. This connects MAP to the Minimum Description Length (MDL) principle: a good model balances explanatory power and simplicity.

4 ML learning with i.i.d. complete data

Assume the dataset is $\mathcal{D} = \{x_1, \ldots, x_N\}$ and samples are i.i.d. under $P(\cdot \mid \theta)$. Then the likelihood factorizes:

$$P(\mathcal{D} \mid \theta) \;=\; \prod_{i=1}^{N} P(x_i \mid \theta).$$

Because products are awkward numerically and algebraically, we maximize the log-likelihood

$$\ell(\theta) \;=\; \log P(\mathcal{D} \mid \theta) \;=\; \sum_{i=1}^{N} \log P(x_i \mid \theta).$$

For smooth objectives, analytical solutions follow from solving $\frac{\partial \ell(\theta)}{\partial \theta} = 0$ (and checking boundary conditions).

5 The biased coin: a warm-up in Bernoulli ML

We start with the simplest generative model: a Bernoulli (coin flip). Let $C \in \{\text{head}, \text{tail}\}$ and parameterize the model by

$$P(C = \text{head}) = \theta, \qquad P(C = \text{tail}) = 1 - \theta, \qquad \theta \in [0, 1].$$

Suppose we observe n tosses, with n_H heads and n_T tails (so $n = n_H + n_T$).

Worked example — MLE for a Bernoulli parameter

The likelihood is

$$P(\mathcal{D} \mid \theta) = \theta^{n_H}(1-\theta)^{n_T}.$$

The log-likelihood is

$$\ell(\theta) = n_H \log\theta + n_T \log(1-\theta).$$

Differentiate:

$$\frac{d\ell}{d\theta} = \frac{n_H}{\theta} - \frac{n_T}{1-\theta}.$$

Set to zero and solve:

$$\frac{n_H}{\theta} = \frac{n_T}{1-\theta} \implies n_H(1-\theta) = n_T\theta \implies \boxed{\theta_{\text{ML}} = \frac{n_H}{n_H+n_T} = \frac{n_H}{n}}.$$

This estimator is just the empirical frequency of heads. It is a first example of the "count then normalize" pattern that will reappear for Naive Bayes.

6 MAP learning for the coin: Beta prior

ML cannot encode prior beliefs (e.g., "coins are usually close to fair"). To incorporate prior beliefs we treat θ as a random variable and place a Beta prior:

$$P(\theta) = \text{Beta}(\alpha,\beta) \;\propto\; \theta^{\alpha-1}(1-\theta)^{\beta-1}.$$

The Beta distribution is conjugate to the Bernoulli likelihood, so the posterior remains Beta:

$$P(\theta \mid \mathcal{D}) = \text{Beta}(\alpha+n_H,\ \beta+n_T).$$

Intuitively, α and β act like pseudo-counts: they behave as if we had already observed $\alpha-1$ heads and $\beta-1$ tails before seeing the data.

Worked example — MAP estimate under a Beta prior (posterior mode)

Ignoring constants that do not depend on θ, the log posterior is

$$\log P(\theta \mid \mathcal{D}) = (n_H+\alpha-1)\log\theta + (n_T+\beta-1)\log(1-\theta) + \text{const}.$$

Differentiate and set to zero:

$$\frac{n_H+\alpha-1}{\theta} - \frac{n_T+\beta-1}{1-\theta} = 0.$$

Solving gives the posterior mode (valid for $\alpha,\beta>1$):

$$\boxed{\theta_{\text{MAP}} = \frac{n_H+\alpha-1}{n_H+n_T+\alpha+\beta-2}}.$$

A note on Bayesian prediction. A Bayesian predictor would integrate over θ rather than plug in a single estimate:

$$P(x' \mid \mathcal{D}) = \int P(x' \mid \theta)\, P(\theta \mid \mathcal{D})\, d\theta.$$

In conjugate cases this can be computed in closed form; in many real-world models the corresponding integral is the main computational challenge.

7 Naive Bayes classifier: a generative model for classification

Now we move from a single Bernoulli parameter to a full generative classifier.

7.1 Problem setting and representation

Each instance is represented by a vector of discrete attributes

$$x = \langle a_1, \ldots, a_L \rangle,$$

and the class label is $c \in C$. Naive Bayes predicts the most probable class given the attributes:

$$c_{\mathrm{NB}} = \arg\max_{c \in C} P(c \mid a_1, \ldots, a_L).$$

7.2 From Bayes' rule to the naive assumption

Bayes' rule yields

$$P(c \mid a_1, \ldots, a_L) \propto P(a_1, \ldots, a_L \mid c)\, P(c),$$

where $P(c)$ is the class prior and $P(a_1, \ldots, a_L \mid c)$ is the class-conditional likelihood of the feature vector.

The defining assumption of Naive Bayes is *conditional independence* of attributes given the class:

$$P(a_1, \ldots, a_L \mid c) = \prod_{l=1}^{L} P(a_l \mid c).$$

Under this assumption, the decision rule becomes

$$c_{\mathrm{NB}} = \arg\max_{c \in C} \Big[P(c) \prod_{l=1}^{L} P(a_l \mid c) \Big].$$

Computationally, we typically work in log space to avoid underflow:

$$c_{\mathrm{NB}} = \arg\max_{c \in C} \Big[\log P(c) + \sum_{l=1}^{L} \log P(a_l \mid c) \Big].$$

Even though the conditional-independence assumption is often violated, Naive Bayes can perform very well in high-dimensional sparse problems (notably in text classification), partly because its parameters are easy to estimate and smoothing is straightforward.

8 ML parameter estimation for Naive Bayes: counting and normalization

Assume we have a labeled dataset of size D:

$$\mathcal{D} = \{(x_1, c_1), \ldots, (x_D, c_D)\}, \quad x_j = \langle a_1^{(j)}, \ldots, a_L^{(j)} \rangle.$$

Let there be K classes. For each attribute a_l, let it take values in $\{1, \ldots, S_l\}$. Naive Bayes requires estimating:

$$P(c = k) \text{ for } k = 1, \ldots, K, \qquad \text{and} \qquad P(a_l = s \mid c = k) \text{ for each } (l, k, s).$$

8.1 Deriving Naive Bayes updates

Worked example — ML estimates for Naive Bayes parameters

Assume a labeled dataset $\mathcal{D} = \{(x_i, c_i)\}_{i=1}^N$ where each instance is $x_i = \langle a_1^{(i)}, \ldots, a_L^{(i)} \rangle$ and $c_i \in \{1, \ldots, K\}$. For each attribute a_l, let its values lie in $\{1, \ldots, S_l\}$.

Indicator variables and parameters. Define indicators

$$z_{ik} = \mathbb{I}[c_i = k], \qquad t_{ls}^{(i)} = \mathbb{I}[a_l^{(i)} = s],$$

and parameters

$$\pi_k \equiv p(C = k), \qquad \phi_{lsk} \equiv p(A_l = s \mid C = k).$$

Constraints:

$$\sum_{k=1}^{K} \pi_k = 1, \ \pi_k \geq 0, \qquad \forall (l, k) : \ \sum_{s=1}^{S_l} \phi_{lsk} = 1, \ \phi_{lsk} \geq 0.$$

Likelihood and log-likelihood. Under the Naive Bayes factorization,

$$p(x_i, c_i \mid \theta) = p(c_i) \prod_{l=1}^{L} p(a_l^{(i)} \mid c_i),$$

so the dataset likelihood can be written using indicators as

$$p(\mathcal{D} \mid \theta) = \prod_{i=1}^{N} \prod_{k=1}^{K} \left[\pi_k \prod_{l=1}^{L} \prod_{s=1}^{S_l} \phi_{lsk}^{t_{ls}^{(i)}} \right]^{z_{ik}}.$$

Taking logs gives

$$\ell(\theta) = \log p(\mathcal{D} \mid \theta) = \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \log \pi_k + \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \sum_{l=1}^{L} \sum_{s=1}^{S_l} t_{ls}^{(i)} \log \phi_{lsk}.$$

Introduce the counts

$$N_k = \sum_{i=1}^{N} z_{ik}, \qquad N_{lsk} = \sum_{i=1}^{N} z_{ik} \, t_{ls}^{(i)},$$

so that

$$\ell(\theta) = \sum_{k=1}^{K} N_k \log \pi_k + \sum_{l=1}^{L} \sum_{k=1}^{K} \sum_{s=1}^{S_l} N_{lsk} \log \phi_{lsk}.$$

Worked example — ML estimates for NB parameters (cont'd)

Step 1: maximize w.r.t. class priors π_k. We maximize $\sum_k N_k \log \pi_k$ subject to $\sum_k \pi_k = 1$ using a Lagrange multiplier λ:

$$\mathcal{J}(\pi, \lambda) = \sum_{k=1}^{K} N_k \log \pi_k + \lambda\Big(\sum_{k=1}^{K} \pi_k - 1\Big).$$

Differentiate and set to zero:

$$\frac{\partial \mathcal{J}}{\partial \pi_k} = \frac{N_k}{\pi_k} + \lambda = 0 \quad \Longrightarrow \quad \pi_k = -\frac{N_k}{\lambda}.$$

Enforce $\sum_k \pi_k = 1$:

$$1 = \sum_{k=1}^{K} \pi_k = -\frac{1}{\lambda} \sum_{k=1}^{K} N_k = -\frac{1}{\lambda} N \quad \Longrightarrow \quad \lambda = -N.$$

Hence

$$\boxed{\pi_k = p(C = k) = \frac{N_k}{N}.}$$

Step 2: maximize w.r.t. conditional probabilities ϕ_{lsk}. For each fixed pair (l, k), maximize

$$\sum_{s=1}^{S_l} N_{lsk} \log \phi_{lsk} \quad \text{subject to} \quad \sum_{s=1}^{S_l} \phi_{lsk} = 1.$$

Introduce a Lagrange multiplier γ_{lk} for the constraint:

$$\mathcal{J}_{lk}(\phi, \gamma_{lk}) = \sum_{s=1}^{S_l} N_{lsk} \log \phi_{lsk} + \gamma_{lk}\Big(\sum_{s=1}^{S_l} \phi_{lsk} - 1\Big).$$

Differentiate w.r.t. ϕ_{lsk} and set to zero:

$$\frac{\partial \mathcal{J}_{lk}}{\partial \phi_{lsk}} = \frac{N_{lsk}}{\phi_{lsk}} + \gamma_{lk} = 0 \quad \Longrightarrow \quad \phi_{lsk} = -\frac{N_{lsk}}{\gamma_{lk}}.$$

Enforce normalization:

$$1 = \sum_{s=1}^{S_l} \phi_{lsk} = -\frac{1}{\gamma_{lk}} \sum_{s=1}^{S_l} N_{lsk} \quad \Longrightarrow \quad \gamma_{lk} = -\sum_{s=1}^{S_l} N_{lsk}.$$

Therefore

$$\boxed{\phi_{lsk} = p(A_l = s \mid C = k) = \frac{N_{lsk}}{\sum_{s'=1}^{S_l} N_{ls'k}}.}$$

Since for fixed (l, k) we have $\sum_{s'=1}^{S_l} N_{ls'k} = N_k$ (each example in class k contributes to exactly one value of attribute l), this can also be written as

$$\boxed{p(A_l = s \mid C = k) = \frac{N_{lsk}}{N_k}.}$$

9 Sparse data and the zero-frequency problem

A practical issue appears immediately in discrete models. If an attribute value s is never observed with class k, then $N_{ls}(k) = 0$ and ML assigns

$$P(a_l = s \mid c = k) = 0.$$

Because Naive Bayes multiplies conditional probabilities across features, a single zero factor can force the whole class score to zero for any test point containing that unseen attribute value. This is usually undesirable: it confuses "unobserved" with "impossible."

10 Smoothing as MAP estimation: Dirichlet priors

A standard remedy is additive (Laplace / add-α) smoothing. Conceptually, smoothing corresponds to a MAP/Bayesian treatment: place a *Dirichlet prior* over each multinomial row $\{P(a_l = s \mid c = k)\}_{s=1}^{S_l}$, which adds pseudo-counts and prevents zeros.

Worked example — Add-α smoothing for Naive Bayes

For an attribute a_l with S_l possible values:

$$\boxed{P(a_l = s \mid c = k) = \frac{N_{ls}(k) + \alpha}{\sum_{s'=1}^{S_l} (N_{ls'}(k) + \alpha)}} = \boxed{\frac{N_{ls}(k) + \alpha}{N(k) + \alpha S_l}}.$$

Common choices are $\alpha = 1$ (Laplace smoothing) or smaller α for milder smoothing.

Smoothing introduces a small bias but can greatly reduce variance, often improving generalization when data are limited or classes are imbalanced.