# Week 3: Naïve Bayes

Instructor: Sergey Levine

## 1 Generative modeling

In the classification setting, we have discrete labels $y \in \{0, \ldots, L_y - 1\}$ (let's assume for now that $L_y = 2$, so we are just doing binary classification), and attributes $\{x_1, \ldots, x_K\}$, where each $x_k$ can take on one of $L_k$ labels $x_k \in \{0, \ldots, L_k - 1\}$. In general, $x_k$ could also be real-valued, and we'll discuss this later, but for now let's again assume that $x_k$ is binary, so $L_k = 2$. We'll assume we have $N$ records. For clarity of notation, superscripts will index records, and subscripts will index attributes, so $y^i$ denotes the label of the $i^{\text{th}}$ record, $\mathbf{x}^i$ denotes all of the attributes of the $i^{\text{th}}$ record, and $x_k^i$ denotes the $k^{\text{th}}$ attribute of the $i^{\text{th}}$ record. Note that there is some abuse of notation here, since $x_k$ is a *random variable*, while $x_k^i$ is the value assigned to that random variable in the $i^{\text{th}}$ record (in this case, an integer between 0 and $L_k - 1$).

If we would like to build a probabilistic model for classification, we could use the conditional likelihood, just like we did with linear regression, which is given by $p(y|\mathbf{x}, \theta)$. In fact, this is what decision trees do, since the distribution over labels at each leaf can be treated as a probability distribution. However, the algorithm for constructing decision trees does not actually maximize $\sum_{i=1}^{N} \log p(y^i|\mathbf{x}^i, \theta)$, because optimally constructing decision trees is intractable. Instead, we use a greedy heuristic, which often works well in practice, but introduces complexity and requires some ad-hoc tricks, such as pruning, in order to work well.

If we wish to construct a probabilistic classification algorithm that actually optimizes a likelihood, we could use $p(\mathbf{x}, y|\theta)$ instead. The difference here is a bit subtle, but modeling such a likelihood is often simpler because we can decompose it into a conditional term and a prior:

$$p(\mathbf{x}, y|\theta) = p(\mathbf{x}|y, \theta)p(y|\theta).$$

Note that the prior now is $p(y|\theta)$: it's a prior on $y$ (we could also have a prior on $\theta$, more on that later). The prior is very easy to estimate: just count the number of times $y = 0$ in the data, count the number of times $y = 1$, and fit the binomial distribution just like we did last week. So that leaves $p(\mathbf{x}|y, \theta)$.

In general, learning $p(\mathbf{x}|y, \theta)$ might be very difficult. We usually can't just "count" the number of times each value of $\mathbf{x}$ occurs in the dataset for $y = 0$, count the number of times each occurs for $y = 1$, and estimate the probabilities that way, because $\mathbf{x}$ consists of $K$ features, and even if each feature is only

binary, we have $2^K$ possible values of $\mathbf{x}$: we'll never get a dataset big enough to see each value of $\mathbf{x}$ even once as $K$ gets large! So we'll use an approximation.

## 2 Naïve Bayes

The approximation consists of exploiting conditional independence. First, let's try to understand conditional independence with a simple example. Let's say that we are trying determine whether there is a rain storm outside, so our label $y$ is 1 if there is a storm, and 0 otherwise. We have two features: rain and lightening, both of which are binary, so $\mathbf{x} = \{1_{\text{rain}}, 1_{\text{lightening}}\}$. If want to model the full joint distribution $p(\mathbf{x}, y)$, we need to represent all 8 possible outcomes ($2^3$). If we want to model $p(\mathbf{x})$, we need to represent all 4 possible outcomes ($2^2$). However, if we just want to represent the conditional $p(\mathbf{x}|y)$, we observe an interesting independence property: if we already know that there is a storm, then rain and lightening are independent of one another. Put another way, if we know there is a storm, and someone tells us that it's raining, that does not tell us anything about the probability of lightening. But if we don't know whether there is a storm or not, then knowing that there is rain makes the probability of lightening higher. Mathematically, this means that:

$$p(\mathbf{x}) = p(x_1, x_2) \neq p(x_1)p(x_2) \quad \text{and} \quad p(x_1, x_2|y) = p(x_1|y)p(x_2|y)$$

We say in this case that rain is *conditionally independent* of lightening – they are independent, but only when conditioned on $y$. Note that as the number of features increases, the total number of parameters in the full joint $p(\mathbf{x}|y)$ increases exponentially, since there are exponentially many values of $\mathbf{x}$. However, the number of parameters in the conditionally independent distribution $\prod_{k=1}^{K} p(x_k|y)$ increases linearly: if the features are binary, each new feature adds just two parameters: the probability of the feature being 1 when $y = 0$ and its probability of being 1 when $y = 1$.

The main idea behind naïve Bayes is to exploit the efficiency of the conditional independence assumption. In naïve Bayes, we assume that *all* of the features are conditionally independent. This allows us to efficiently estimate $p(\mathbf{x}|y)$.

**Question.** What is the data?

**Answer.** The data is defined as $\mathcal{D} = \{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}$, where $y$ is categorical, and $\mathbf{x}$ is a vector of features which may be binary, multivariate, or, as we will see later, continuous.

**Question.** What is the hypothesis space?

**Answer.** The hypothesis space is the space of all distributions that factorize according to

$$p(y) \prod_{k=1}^{K} p(x_k|y).$$

If we assume (for now) that $y$ and each $x_k$ are binary, then we have $2K + 1$ different binomial distributions that we need to estimate. Since each of these distributions has one parameter, we have $\theta \in [0, 1]^{2K+1}$.

**Question.** What is the objective?

**Answer.** The MLE objective for naïve Bayes is

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \log p(\mathbf{x}^i, y^i|\theta).$$

Later, we'll also see that we can formulate a Bayesian objective of the form $\log p(\theta|\mathcal{D})$.

**Question.** What is the algorithm?

**Answer.** In order to optimize the objective, we simply need to estimate each of the distributions $p(x_k|y)$ and the prior $p(y)$. Each of these can be treated as a separate MLE problem. To estimate the prior $p(y)$, we simply estimate

$$p(y = j) = \frac{\text{Count}(y = j)}{\sum_{j'} \text{Count}(y = j')},$$

where $\sum_{j'}^{L_y} \text{Count}(y = j') = N$,[1] the size of the dataset. For each feature $x_k$, if $x_k$ is multinomial (or binomial), we estimate

$$p(x_k = \ell|y = j) = \frac{\text{Count}(x_k = \ell \text{ and } y = j)}{\sum_{\ell'} \text{Count}(x_k = \ell' \text{ and } y = j)},$$

where $\sum_{\ell'} \text{Count}(x_k = \ell' \text{ and } y = j) = \text{Count}(y = j)$, the number of records for which $y = j$. It's easy to check that this estimate of the parameters maximizes the likelihood, and this is left as an exercise.

Now, a natural question to ask is: when we observe a new record with features $\mathbf{x}^\star$, how do we predict the corresponding label $y^\star$? This is referred to as the *inference* problem: given our model of $p(\mathbf{x}, y)$, we have to determine the $y^\star$ that makes the observed $\mathbf{x}^\star$ most probable. That is, we have to find

$$y^\star = \arg\max_y p(\mathbf{x}^\star, y)$$

Fortunately, the number of labels $y$ is quite small, so we can simply evaluate the probability of each label $j$. So, given a set of features $\mathbf{x}^\star$, we simply test $p(\mathbf{x}^\star, y = j)$ for all $j$, and take the label $j$ that gives the highest probability.

[1] We can express this more formally in set notation: $\text{Count}(y = j) = |\{y^i \in \mathcal{D}|y^i = j\}|$.