CS281 Section #2 Notes, v 1.0

Due: never

1 Naive Bayes Classification

In this section, we cover the basics of classification, introduce the Naive Bayes classifier, and show how the softmax function falls out of the Naive Bayes classifier.

1.1 Classification

We aim to sort N samples, each with J features, into C categories. Let's consider the emaple of email spam detection, where N is the number of emails in our dataset and C is the two-element set $\{spam, notspam\}$. We can generate the J features for each email in a variety of ways. For instance, we count the number of occurrences of 500 different words in each email and get $J = 500$. Alternatively, we could use just the average number of words in each email's sentences and total number of words in the email and get $J = 2$. Feature selection is an important step in real-life classification problems, but for the purpose of this section we assume that our features have already been generated.

1.2 Bayes Classification

To classify the samples, we need to infer the category of each email given the features. Formally, the label of the n-th sample y_n can be written as

$$y_n = argmax_{y_n} p(y_n|\mathbf{x_n}) = argmax_{y_n} \frac{p(\mathbf{x_n}|y_n)p(y_n)}{Z}, \tag{1}$$

where Z is a normalizer. We call this a "Bayes" classifier because the second expression is given by Bayes' theorem.

1.2.1 Naive Bayes

The Naive Bayes classifier (NBC) is "naive" because it assume a sample's features are independent given its label:

$$p(\mathbf{x_n}|y_n) = p(x_{n1}, x_{n2}, ..., x_{nJ}|y_n) \xrightarrow{naive} \prod_j p(x_{nj}|y_n) \tag{2}$$

The final step is our naive assumption, which is by no means guaranteed.

Q: What is the graphical model for the NBC?

1.2.2 Supplying the distributions needed

For an NBC to work, it must be supplied with these distributions: $p(x_{nj}|y_n = c)$ for all j, c, and $p(y_n = c)$ for all c. While the latter is usually simply given as a categorical distribution (let's denote the probability of the category c as π_c), the former can take on various forms. In particular, if $p(x_{nj}|y_n = c)$ is assumed to be from a multivariate normal (MVN) distribution, then we can write the MVN distribution as

$$p(\mathbf{x}|y = c) = \mathcal{N}\left(\mu_c, \Sigma^{(c)}_{diag.}\right). \tag{3}$$

The covariance matrix is diagonal (it is only non-zero on the diagonal) because we assumed that different elements have no correlation. The parameters that describe these distributions generally need to be learned from the data.

1.3 Learning the distributions

Denote all the parameters in an NBC as θ. If we have N samples to learn the distributions from, we denote $\mathbf{X}$ as the N-by-J data matrix and $\mathbf{Y}$ as the N sized vector with labels for each sample, then we can write the model likelihood as

$$p\left(\mathbf{X}, \mathbf{Y}|\theta\right) = \prod_c \pi_c^{N(\mathbf{Y}=c)} \prod_j \prod_c \prod_{n, y_n=c} p(x_{nj}|y_n). \tag{4}$$

This seems complicated, so let's explain it. The equation has two parts: first, for each category c, we count how many samples are classified into this category (that's $N(\mathbf{Y}=c)$), and take the product of the probability π_c of this category. Second, for each feature (j) of each sample ($n, y_n = c$) in this category (c), we take the product of the conditional $p(x_{nj}|y_n)$. This entire product yields the joint distribution $p(X, Y|\theta)$.

Next, we take the logarithm to yield the log-likelihood:

$$\log \mathcal{L}(\theta) = \sum_c N(\mathbf{Y}=c) \log(\pi_c) + \sum_j \sum_c \sum_{n, y_n=c} \log p(x_{nj}|y_n). \tag{5}$$

We can learn the parameters by performing maximum likelihood estimation (MLE).

1.4 Classifying

Say we trained our NBC on some data $\mathbf{X}$ to obtain MLE-fitted parameters θ. Now let's try to predict a new sample $x_{n'}$. The probability that it belongs to the category c is given by

$$p(y_{n'} = c|\mathbf{X}, \theta) \propto \pi_c \prod_j p(x_{n'j}|y_{n'}). \tag{6}$$

Again, take the logarithm to obtain

$$\log\left[p(y_{n'} = c|\mathbf{X}, \theta)\right] = \log \pi_c + \sum_j \log p(x_{n'j}|y_{n'}) + Const. \tag{7}$$

1.5 A special consideration for multinoulli models

To continue with the discussion in lecture, let's consider a specific classification problem. Here, each feature is categorical, and there are only two categories. That is, instead of scalar values, $x_{nj} \in \{0, 1\}$. If this is the case, then we can simply characterize the model with a mean rate μ_{jc}, which corresponds to the probability of each event of the feature j taking place in the category c. For example, if the feature in question of an email is whether the tone is positive (0) or negative (1), then μ_{j0} would be the probability that a spam email is positive. We then have (from eq.6):

$$p(y_{n'} = c|x_{n'j}) \propto \pi_c \prod_j \mu_{jc}^{N(x_j=1)} (1-\mu_{jc})^{N(x_j=0)}. \tag{8}$$

This is the "informal parameterization" in class. Likewise, we can rewrite eq.7 as

$$\log\left[p(y_{n'} = c|\mathbf{X}, \theta)\right] = \log \pi_c + \sum_j \log(1-\mu_{jc}) + \sum_j x_j \log \frac{\mu_{jc}}{1-\mu_{jc}} + Const. \tag{9}$$

For the purpose of being lazy, we define $\theta_{jc} = \log \frac{\mu_{jc}}{1-\mu_{jc}}$. To be even lazier, combine all the θ_{jc} and write θ_c. Thus,

$$\sum_j x_j \log \frac{\mu_{jc}}{1-\mu_{jc}} = \theta_{\mathbf{c}}^{\mathbf{T}} \mathbf{X}. \tag{10}$$

We also denote $b_c = \log \pi_c + \sum_j \log(1-\mu_{jc})$, which does not depend on the data at all. This is how to obtain the final form:

$$p(y_{n'} = c|x_{n'j}) \propto \exp\left(b_c + \theta_{\mathbf{c}}^{\mathbf{T}} \mathbf{X}\right). \tag{11}$$