COMP9417: Naive Bayes Classifiers

Omar Ghattas

July 2, 2019

1 Probabalistic Classification

In a classification problem, we are given a set of classes, $C = \{c_1, \ldots, c_K\}$, a set of examples $D = \{(\mathbf{x}_i, c_i) : i = 1, \ldots, n\}$, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ is a vector of features for the i-th subject. The goal of classification is, given a new datum, $\mathbf{x}_\star$, to predict the 'best' class, $c_\star$. The definition of 'best' varies across methods, for example, under kNN, 'best' meant the class in agreement with the k neighbours of $\mathbf{x}_\star$. Under the probabalistic approach 'best' is taken to mean 'most probable'. In order to talk about probabilities, we need to talk about probability distributions. In particular, we are most interested in the conditional probability distribution of a class given the data:

$$p(c_k|\mathbf{x}_\star).$$

We would like to compute this probability for each of the k classes, and return our prediction as

$$c_\star = \arg\max_k p(c_k|\mathbf{x}_\star).$$

There are a couple of approaches we could take to learning this conditional distribution. One approach, called the discriminative approach, is to learn the distribution $p(c_k|\mathbf{x})$ directly - this is what Logistic regression does for example. An alternative approach that we will focus on here would be the generative approach. To understand this second approach, recall Bayes rule:

$$p(c_k|\mathbf{x}) = \frac{p(\mathbf{x}|c_k)p(c_k)}{p(\mathbf{x})},$$

which tells us that instead of focusing on $P(c_k|\mathbf{x})$, we could instead model $p(\mathbf{x}|c_k)$ and $p(c_k)$, for each k - this is called a generative approach. $p(\mathbf{x}|c_k)$ is termed the class conditional distribution, and $p(c_k)$ is termed the class prior distribution. Note that we do not need to model $p(\mathbf{x})$ since it can be computed from knowledge of the class conditional and prior distributions, i.e.

$$p(\mathbf{x}) = \sum_{k=1}^{K} p(\mathbf{x}|c_k)p(c_k).$$

Now, what we need to do under a generative approach is to choose the form of the class conditional distribution (e.g. continuous, multinomial, multivariate bernoulli, etc.) and class prior distribution (e.g. discrete uniform, categorical, etc). The prior is usually taken to be a discrete distribution since the number of classes is finite, whilst the class conditional distribution can be either continuous or discrete. One example of this approach is Gaussian (linear) discriminant analysis.

2 Naive Bayes

In the previous section we saw that in order to do generative classification, we must learn a class conditional distribution for each of the k classes in our data. Now, if our observations live in a high-dimensional space (the number of features p is very large), then learning these distributions can be very difficult. For example, if $p = 1000$, and we assume that $x|c_k \sim N(\mu_k, \sigma_k^2)$, then we would have to learn a 1000-dimensional Gaussian for each class. Learning high dimensional distributions can be very tricky, and we often need a large amount of data (relative to p) to be able to do it well. Naive Bayes gives us a way out of this problem. Before describing it, let's go back and rewrite the quantity we are trying to estimate:

$$\begin{aligned}
p(\mathbf{x}|c_k)p(c_k) &= p(\mathbf{x}, c_k) \\
&= p(x_1, x_2, \ldots, x_p, c_k) \\
&= p(x_1|x_2, \ldots, x_p, c_k)p(x_2, x_3, \ldots, x_p, c_k) \\
&= p(x_1|x_2, \ldots, x_p, c_k)p(x_2|x_3, \ldots, x_p, c_k)p(x_3, x_4 \ldots, x_p, c_k) \\
&\quad \vdots \\
&= p(x_1|x_2, \ldots, x_p, c_k)p(x_2|x_3, \ldots, x_p, c_k)p(x_3|x_4 \ldots, x_p, c_k) \times \cdots \times p(x_p|c_k)p(c_k).
\end{aligned}$$

These computations are valid for any probability distribution, and we have so far made zero assumptions. Naive Bayes introduces the assumption that the features are conditionally independent, this is written as

$$x_i \perp x_j | c_k \quad \text{for all } j \neq i.$$

Now, let's interpret this assumption intuitively. Let's assume that we are trying to classify whether a patient has or doesn't have diabetes, and I tell you that a particular patient, $\mathbf{x}$, in our dataset belongs to the class diabetes. Let's also assume that x_1 describes the blood pressure of the patient, and x_2 describes the weight of the patient. Now, the naive bayes assumption tells us that if we know the patient is diabetic, then also knowing that the patient has high blood pressure tells us nothing about their weight. Realistically, we expect that for patients with diabetes, there exists a positive correlation between blood pressure and weight. Under Naive Bayes however, we ignore this correlation for the sake of mathematical tractability.

Mathematically, the Naive Bayes assumption means that $p(x_i|x_j, c_k) = p(x_i|c_k)$. Using this, we can rewrite:

$$\begin{aligned} p(\mathbf{x}|c_k)p(c_k) &= p(\mathbf{x}, c_k) \\ &= p(x_1|x_2, \ldots, x_p, c_k)p(x_2|x_3, \ldots, x_p, c_k)p(x_3|x_4 \ldots, x_p, c_k) \times \cdots \times p(x_p|c_k)p(c_k) \\ &= p(x_1|c_k)p(x_2|c_k)p(x_3|c_k) \times \cdots \times p(x_p|c_k)p(c_k) \\ &= p(c_k) \prod_{i=1}^{p} p(x_i|c_k). \end{aligned}$$

This means that instead of estimating a p-dimensional distribution, we now just have to estimate p different 1-dimensional distributions (for each of the k classes) instead - since estimating 1-dimensional distributions is much more striaght forward, this is a much easier task!

3 spam or ham

We have seen that Naive Bayes allows us to write

$$p(\mathbf{x}|c_k) = \prod_{i=1}^{p} p(x_i|c_k),$$

but we still need to decide the functional form of the $p(x_i|c_k)$. First, let's consider the problem of classifying emails as spam or ham, we consider the following example from tutorial:

e_1	b	d	e	b	b	d	e			
e_2	b	c	e	b	b	d	d	e	c	c
e_3	a	d	a	d	e	a	e	e		
e_4	b	a	d	b	e	d	a	b		
e_5	a	b	a	b	a	b	a	e	d	
e_6	a	c	a	c	a	c	a	e	d	
e_7	e	a	e	d	a	e	a			
e_8	d	e	d	e	d					

Each row represents an email, and each email is a combination of words taken from the set $\{a, b, c, d, e\}$. We treat the words d, e as stop words - these are words that are not useful for classification purposes, for example, the word 'the' is too common to be useful for classifying documents as spam or ham. We therefore define our vocabulary as $V = \{a, b, c\}$. Note that in this case we have two classes, so $k = 2$, and we will assume a uniform prior, that is:

$$p(c_+) = p(c_-) = \frac{1}{2},$$

where $c_+ =$ spam, $c_- =$ ham. We now focus on two ways of encoding the observations, and what this means for the class conditional distribution.

3.1 Multivariate Bernoulli Naive Bayes

First, we encode the first email (e_1) as $\mathbf{x}_1 = (x_{1a} = 0, x_{1b} = 1, x_{1c} = 0)$, so that each of the features are binary and represent whether a word is present or not in a given email. Carrying this out for all emails in our dataset results in:

	x_{ia}	x_{ib}	x_{ic}
e_1	0	1	0
e_2	0	1	1
e_3	1	0	0
e_4	1	1	0
e_5	1	1	0
e_6	1	0	1
e_7	1	0	0
e_8	0	0	0

Under this model, we are choosing to ignore the frequency of words in the email and just consider whether a word appears or not in an email. This is equivalent to modelling the class conditional distribution as

$$p(\mathbf{x}|c_k) = \prod_{j \in V} p(x_j|c_k),$$

where $x_j|c_k \sim \text{Bernoulli}(p_j^k)$. In other words:

$$p(x_j|c_k) = (p_j^k)^{x_j}(1 - p_j^k)^{1-x_j}.$$

Note that p_j^k is the probability that a document of class k contains the word j or not. So, under this approach, we must estimate the following parameters:

$$p_a^+, p_b^+, p_c^+, p_a^-, p_b^-, p_c^-.$$

We will not go into detail here, but under the hood we are using maximum likelihood estimation to estimate these probabilities. It turns out that maximum likelihood gives us an intuitive estimate

$$p_j^k = \frac{\text{number of documents of class } k \text{ that contain the word } j}{\text{number of documents in class } k},$$

so, based on our data, this gives us:

$$p_a^+ = \frac{2}{4}, \quad p_b^+ = \frac{3}{4}, \quad p_c^+ = \frac{1}{4},$$
$$p_a^- = \frac{3}{4}, \quad p_b^- = \frac{1}{4}, \quad p_c^- = \frac{1}{4}.$$

Now, we have built our classifier! Let's assume we get a new email that we want to classify: $e_\star = abbdebb$, then encoding this under a multivariate bernoulli model gives $\mathbf{x}_\star = (1, 1, 0)$. Now, we classify according the formula

$$c_\star = \arg \max_{k \in \{+,-\}} p(c_k) p(\mathbf{x}_\star | c_k).$$

First, for $k = +$, we have:

$$\begin{aligned}
p(c_+)p(\mathbf{x}_\star|c_+) &\overset{\text{(NB)}}{=} p(c_+) \prod_{j \in V} p(x_j = x_{\star j}|c_+) \\
&= p(c_+) \times p(x_a = x_{\star a}|c_+) \times p(x_b = x_{\star b}|c_+) \times p(x_c = x_{\star c}|c_+) \\
&= p(c_+) \times p(x_a = 1|c_+) \times p(x_b = 1|c_+) \times p(x_c = 0|c_+) \\
&= p(c_+) \times p_a^+ \times p_b^+ \times (1 - p_c^+) \\
&= \frac{1}{2} \times \frac{2}{4} \times \frac{3}{4} \times \left(1 - \frac{1}{4}\right) \\
&= \frac{9}{64},
\end{aligned}$$

where (NB) is to signify that we are making the Naive Bayes assumption. A similar computation gives us:

$$\begin{aligned}
p(c_-)p(\mathbf{x}_\star|c_-) &= p(c_-) \times p_a^- \times p_b^- \times (1 - p_c^-) \\
&= \frac{1}{2} \times \frac{3}{4} \times \frac{1}{4} \times \left(1 - \frac{1}{4}\right) \\
&= \frac{9}{128}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
c_\star &= \arg \max_{k \in \{+,-\}} p(c_k) p(\mathbf{x}_\star | c_k) \\
&= \arg\max \left\{ \frac{9}{64}, \frac{9}{128} \right\} \\
&= c_+.
\end{aligned}$$

3.1.1 Multivariate Bernoulli Smoothing

In the previous section, we saw how to make predictions under the Naive Bayes Multivariate Bernoulli model. Note that for each prediction, we have to multiply various probabilities with each other. These probabilities are of the form p_j^k or $(1-p_j^k)$. Since we estimate these probabilities based on occurences in the data, this can be problematic if we have not seen particular observations before. For example, assume that we never observe the word a in any spam emails, so that:

	x_{ia}	x_{ib}	x_{ic}
e_1	0	1	0
e_2	0	1	1
e_3	0	0	0
e_4	0	1	0
e_5	1	1	0
e_6	1	0	1
e_7	1	0	0
e_8	0	0	0

Now, our estimate of $p_a^+ = 0$. Therefore, any new data point, $x_\star$, which has $x_{\star a} = 1$, regardless of the values of the other features, gets $p(\mathbf{x}_\star|+) = 0$. Further note that if $p_a^+ = 1$, which means every spam email has the word a, we get similar behaviour, since any new email that does not have the word a gets probability of spam equal to zero, as $(1-p_a^+) = (1-1) = 0$. This is not ideal behaviour, since we allow one feature to override the entire model, and in the case of document classification, the model is completely unable to deal with new words not before seen in our vocabulary. Therefore, the modification is often made to Naive Bayes in which we smooth our probability estimates. This amounts to altering our estimating equation slightly to

$$p_j^k = \frac{\text{number of documents of class } k \text{ that contain the word } j \,+\, 1}{\text{number of documents in class } k + \text{number of classes}}.$$

This modification ensures that $p(\mathbf{x}|c_k)$ is never equal to zero, even if the number of times a word j appears in a document of class k is actually zero. So, our smoothed estimates now would be:

$$p_a^+ = \frac{2+1}{4+2} = \frac{3}{6}, \quad p_b^+ = \frac{3+1}{4+2} = \frac{4}{6}, \quad p_c^+ = \frac{1+1}{4+2} = \frac{2}{6},$$
$$p_a^- = \frac{3+1}{4+2} = \frac{4}{6}, \quad p_b^- = \frac{1+1}{4+2} = \frac{2}{6}, \quad p_c^- = \frac{1+1}{4+2} = \frac{2}{6}.$$

Estimation is done identically as before, except with our new estimates.

Note that smoothing can be thought of in a different way. We want our probability estimates to never be zero, which is equivalent to observing every possible combination of words and classes. In other words, we need to observe documents in each class that contain each word in our vocabulary, as well as documents in each class that do not contain each word in our vocabulary. We can therefore introduce 'pseudo-documents', that ensure we have seen every combination of outcomes at least once. This amounts to adding $e_{p1}, e_{p2}, e_{p3}, e_{p4}$ as follows

	x_{ia}	x_{ib}	x_{ic}
e_1	0	1	0
e_2	0	1	1
e_3	0	0	0
e_4	0	1	0
e_{p1}	1	1	1
e_{p2}	0	0	0
e_5	1	1	0
e_6	1	0	1
e_7	1	0	0
e_8	0	0	0
e_{p1}	1	1	1
e_{p2}	0	0	0

If this was our dataset, then none of our probability estimates based on the original estimation procedure would be zero.

3.2 Multinomial Naive Bayes

The multvariate bernoulli model ignores frequencies, which can throw away useful information. For example, it would be important to know that the word 'casino' appeared multiple times in an email, instead of just once. Before jumping into the Naive Bayes model, let's review the Multinomial distribution.

3.2.1 Multinomial Distribution

Consider a k-sided die, with the probability of each side being θ_i, so that $\sum_{i=1}^{k} \theta_i = 1$. Now, if we throw the die n times (the n throws are independent) and record the frequency of each face of the die as a vector $X = (X_1, X_2, \ldots, X_k)$, where $X_i =$ number of times face i came up, then we say that X follows a multinomial distribution with parameters $\theta_1, \ldots, \theta_k$ and n. We summarise this as:

$$X \sim \text{Multinomial}(n, \theta_1, \ldots, \theta_k), \quad \sum_{i=1}^{k} \theta_i = 1, \ \theta_i > 0 \forall i.$$

The probability mass function of the multinomial is:

$$P(X = (x_1, x_2, \ldots, x_k)) = \frac{n!}{x_1! \times x_2! \times \cdots \times x_k!} \theta_1^{x_1} \times \cdots \times \theta_k^{x_k}$$
$$= \frac{n!}{\prod_{i=1}^{k} x_i!} \prod_{i=1}^{k} \theta_i^{x_i}$$

3.2.2 Multinomial Naive Bayes

Now, under this model, we choose to encode emails as count vectors,

	x_{ia}	x_{ib}	x_{ic}
e_1	0	3	0
e_2	0	3	3
e_3	3	0	0
e_4	2	3	0
e_5	4	3	0
e_6	4	0	3
e_7	3	0	0
e_8	0	0	0

where now the features represent the number of occurences of a word, rather than whether or not they were present. We now model the class conditional distribution as a multinomial, which implicitly uses the Naive Bayes assumption since the multinomial distribution assumes independence across trials.

$$p(\mathbf{x}|c_k) = \frac{n!}{\prod_{j \in V} x_j!} \prod_{j \in V} p(x_j|c_k)^{x_j},$$

and we write $\theta_j^k = p(x_j|c_k)$ to denote the probability that a word in document of class k takes the value j. Note the subtle difference to the bernoulli model. Previously, the probabilities were to do with the probability that a particular word existed in a given class. The probabilities here are to do with how likely a randomly chosen word in the class is a particular word. Now, we need to estimate the following parameters for our model:

$$\theta_a^+, \theta_b^+, \theta_c^+, \theta_a^-, \theta_b^-, \theta_b^-.$$

We once more omit the details of how to derive the estimates of these parameters, and as one might expect, the estimates are:

$$\theta_j^k = \frac{\text{number of times word } j \text{ appears in class } k}{\text{number of words that appear in class } k}$$

so, based on our data, this gives us:

$$\begin{aligned} \theta_a^+ &= \frac{5}{17}, \quad \theta_b^+ = \frac{9}{17}, \quad \theta_c^+ = \frac{3}{17}, \\ \theta_a^- &= \frac{11}{17}, \quad \theta_b^- = \frac{3}{17}, \quad \theta_c^- = \frac{3}{17}. \end{aligned}$$

Now, given $\mathbf{x}_\star = (4, 3, 1)$, we can compute

$$\begin{aligned} p(c_+)p(x_\star|c_+) &= p(c_+)\frac{8!}{4! \times 3! \times 1!} \times \left(\theta_a^4 \times \theta_b^3 \times \theta_c^1\right) \\ &= \frac{1}{2} \times \frac{8!}{4! \times 3! \times 1!} \times \left(\left(\frac{5}{17}\right)^4 \times \left(\frac{9}{17}\right)^3 \times \left(\frac{3}{17}\right)^1\right) \end{aligned}$$

and similarly for $p(c_-)p(\mathbf{x}_\star|c_-)$.

3.2.3 Multinomial Smoothing

In our predictions here, just as in the multivariate bernoulli case, we must multiply various probabilities. Therefore, this model suffers the same potential problems when probabilities take the value zero. The difference here is that this only occurs when the *frequency* of a word in a particular class is zero. To get around this, we apply smoothing once more, which involves adding 1 to each of the numerators of our estimates. Note that for each class k, this means adding 1 to the numerator $|V|$ times, and so we add $|V|$ to the denominator to account for the $|V|$ extra (pseudo-documents) that we are artifically adding. Our estimates therefore become

$$\theta_j^k = \frac{\text{number of times word } j \text{ appears in class } k+1}{\text{number of words that appear in class } k + |V|}.$$

Now, our estimates for the spam problem become

$$\begin{aligned} \theta_a^+ &= \frac{5+1}{17+3} = \frac{6}{20}, \quad \theta_b^+ = \frac{9+1}{17+3} = \frac{10}{20}, \quad \theta_c^+ = \frac{3+1}{17+3} = \frac{4}{20}, \\ \theta_a^- &= \frac{11+1}{17+3} = \frac{12}{20}, \quad \theta_b^- = \frac{3+1}{17+3} = \frac{4}{20}, \quad \theta_c^- = \frac{3+1}{17+3} = \frac{4}{20}. \end{aligned}$$