Naive Bayes

1 Hypothesis

Consider the domain $\mathcal{X} = \{0,1\}^D$ and the codomain $\mathcal{Y} = \{1,\ldots,C\}$. In Naive Bayes classification, we use the hypothesis $h : \mathcal{X} \to \mathcal{Y}$ defined by

$$h(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} \hat{p}(y = c) \prod_{i=1}^{D} \hat{p}(x^{(i)} \mid y = c)$$

$\hat{p}(y = c)$ is the estimate of the probability of class label c obtained by dividing the number of training samples with that label by the total number of training samples. $\hat{p}(x^{(i)} \mid y = c)$ is the estimate of the conditional probability of the value of feature $x^{(i)}$ (which could be 0 or 1) given whether the class label is c.

The hypothesis comes from applying Bayes rule to $p(y = c \mid x^{(1)}, x^{(2)}, \ldots, x^{(D)})$, which is precisely what we want find: the probability that a data point is of class c given its many features. Applying Bayes rule, we obtain

$$p(y = c \mid x^{(1)}, x^{(2)}, \ldots, x^{(D)}) = \frac{p(x^{(1)}, x^{(2)}, \ldots, x^{(D)} \mid y = c)p(y = c)}{p(x^{(1)}, x^{(2)}, \ldots, x^{(D)})}$$

We can drop $p(x^{(1)}, x^{(2)}, \ldots, x^{(D)})$ because it is not relevant to our analysis (if we are interested in the actual probabilities and not just the greatest "probability", we can simply normalize; this is further discussed in the next section). We can estimate $p(y = c)$ as described previously (obtaining $\hat{p}(y = c)$). The $p(x^{(1)}, x^{(2)}, \ldots, x^{(D)} \mid y = c)$ term is where the "Naive" in Naive Bayes comes from: by assuming that the probability of each feature given the class label is conditionally independent, the term could be factored as such

$$p(x^{(1)}, x^{(2)}, \ldots, x^{(D)} \mid y = c) = \prod_{i=1}^{D} p(x^{(i)} \mid y = c)$$

Each of the terms in the product can then be estimated as described previously (obtaining $\hat{p}(x^{(i)} \mid y = c)$).

Although this assumption is very likely not true in most situations, we are still able to obtain an approximation of the probability which at the very least let's us perform classification.

The Naive Bayes hypothesis can also be viewed as a simple Bayes Net, with one node corresponding to the class label, which is connected to D many nodes each representing a feature.

2 Normalization

Usually, since Naive Bayes is used for classification, only the greatest class "probability" is relevant (as can be observed in the hypothesis). However, the numbers obtained (namely $\hat{p}(y = c) \prod_{i=1}^{D} \hat{p}(x^{(i)} \mid y = c)$ for each $c \in \mathcal{Y}$) do not represent actual probabilities, due to the fact that they do not add up to 1. To obtain actual probabilities, it may seem necessary to calculate the term $p(x^{(1)}, x^{(2)}, \ldots, x^{(D)})$. However, we can instead simply normalize the numbers obtained so that they add up to one.

To accomplish this, we can introduce a scalar α such that

$$1 = \alpha \sum_{c \in \mathcal{Y}} \hat{p}(y = c) \prod_{i=1}^{D} \hat{p}(x^{(i)} \mid y = c) \iff \alpha = \frac{1}{\sum_{c \in \mathcal{Y}} \hat{p}(y = c) \prod_{i=1}^{D} \hat{p}(x^{(i)} \mid y = c)}$$

3 Laplace smoothing

An issue with the $\hat{p}(x^{(i)} \mid y = c)$ estimates arises when no training samples of class label c contain a certain feature. This is because, letting $x^{(k)}$ be such a feature, the estimate will be

$$\hat{p}(x^{(k)} \mid y = c) = \frac{\text{Number of samples of class } c \text{ with feature } x^{(k)}}{\text{Total number of samples}} = \frac{0}{N} = 0$$

As such, when calculating $\prod_{i=1}^{D} \hat{p}(x^{(i)} \mid y = c)$, we will obtain 0, since one of the terms in the product will be 0. However, just because we happen to have some data where no sample in some class contains some feature, we don't want the probability of belonging to that class to be 0 if a new sample contains said feature. To address this, a common technique, called Laplace smoothing, is to add "pseudo counts" so that the numerator is never 0. Thus, using some small value ε, we have that the new estimate is

$$\hat{p}(x^{(k)} \mid y = c) = \frac{\text{Number of samples of class } c \text{ with feature } x^{(k)} + \varepsilon}{\text{Total number of samples} + \varepsilon}$$

4 Alternative input spaces

Naive Bayes classification is widely used in detecting spam. In this context, we have that features are the absence or presence of individual words, and $\mathcal{Y}$ can be interpreted as either spam or not spam. In this or other similar applications, alternative inputs exist which may offer advantages over just looking at individual words. These input spaces are discussed in the subsections below.

4.1 n-grams

In practically all natural languages, the order of words is important for deriving meaning. Thus, instead of using single words as features, pairs of words or small phrases can be used instead. A feature that represents n concatenated words is called an n-gram.

4.2 Stemming

Instead of having different features for each tense or variation of a word, we can reduce words to their "root" form. For example, "retrieve", "retrieving", "retrieval", "retrieved" and "retrieves" could all be mapped to *retriev*.

4.3 Stop words

Ignore common words that don't convey much information, such as "and" or "the".

4.4 TF-IDF

Binary features of whether or not a word appears in a piece of text (usually called a "document") is a relatively coarse feature. A more nuanced metric is the frequency that a word t appears within a document d. This is called "term frequency", and can be calculated by

$$\text{tf}(t, d) = \frac{\text{Number of times word } t \text{ appears in } d}{\text{Number of words in } d}$$

However, some words are more common than others (for example "and" or "the"), so we can scale by the log of the inverse of the frequency of emails containing the word t in the training data D. This is called the "inverse document frequency", and can be calculated by

$$\text{idf}(t, D) = \frac{\text{Number of emails in training data } D}{\text{Number of emails containing word } t}$$