Naïve Bayes Notes – Aron Culotta – CS429

Given two random variables x and y, this is Bayes' Rule:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

There are no assumptions being made in this rule – it's simply an application of the laws of probability. In particular, it just applies two other rules of probability:

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

and

$$p(x,y) = p(x|y)p(y)$$

If we again let $y_i \in \{-1, 1\}$ be our binary classification label, and x_i be our feature vector, then we can use Bayes' rule to do classification for one instance x_i:

$$h(x_i, y_i) = p(y_i|x_i) = \frac{p(x_i|y_i)p(y_i)}{p(x_i)}$$

The Naïve part comes from how we define $p(x_i|y_i)$. Recall that x_i is a vector of feature values. For example, x_i may have three binary feature values, x_{i1}, x_{i2}, x_{i3}. The table corresponding to $p(x_i|y_i)$ might look like this:

x_{i1}	x_{i2}	x_{i3}	$p(x_{ij}\|y_i = 1)$
0	0	0	0.1
0	0	1	0.2
0	1	0	0.15
0	1	1	0.15
1	0	0	0.05
1	0	1	0.05
1	1	0	0.10
1	1	1	0.2

So, if $y = 1$ means a person is attractive, and the features correspond to "has blonde hair", "is tall", and "has green eyes", then given that someone is attractive, the probability that they possess all those features is 0.2.

The problem with this table is that it grows exponentially with the number of features (i.e., if there are k features, there are 2^k rows in this table). Since many problems we care about (e.g., text classification) have millions of features, this is not tractable.

So, to make this tractable, we need to become Naïve. We do this by making a **conditional independence assumption**. In general, conditional independence means that $p(a, b|c) = p(a|c)p(b|c)$. Here, we assume each feature value is independent of others given the class label y.

$$p(x_{i1} \dots x_{ik}|y_i) = p(x_{i1}|y_i) \dots p(x_{ik}|y_i) = \prod_j p(x_{ij}|y_i)$$

Thus, our exponentially large table now becomes a list of three much smaller tables, e.g.:

x_{i1}	$p(x_{i1}\|y = 1)$	x_{i2}	$p(x_{i2}\|y = 1)$	x_{i3}	$p(x_{i3}\|y = 1$
0	0.3	0	0.4	0	0.9
1	0.7	1	0.6	1	0.1

This will scale much better (linearly) to millions of features.

Plugging this assumption into our classifier, we get:

$$p(y_i|x_i) = \frac{p(y_i)\prod_j p(x_{ij}|y_i)}{p(x_i)}$$

Now, given some training set D, we need to estimate the three probabilities $p(x_{ij}|y_i), p(y_i), p(x_i)$. We do this not by any fancy gradient descent methods, but simply by counting.

Bernoulli Naive Bayes

Let $\#(y_i = 1)$ be the number of instances in D with label 1, e.g., the number of attractive people. Let $\#(x_{ij} = 1, y_i = 1)$ be the number of instances in D with label $y_i = 1$ that have $x_{ij} = 1$, e.g., the number of attractive people with green eyes. Then

$$p(x_{ij} = 1|y_i = 1) = \frac{\#(x_{ij} = 1, y_i = 1)}{\#y_i = 1}$$

E.g, the probability of a person having green eyes given that they are attractive is simply the proportion of attractive people that have green eyes. We can perform similar computations for, e.g., $p(x_{ij} = 0|y_i = 1)$ and $p(x_{ij} = 1|y_i = 0)$.

As for $p(y_i)$, this again is computed by counting:

$$p(y_i = 1) = \frac{\#(y_i = 1)}{\#(y_i = 1) + \#(y_i = -1)}$$

e.g., the probability of a person being attractive is simply the proportion of people in D that are labeled as attractive.

Finally, $p(x_i)$ is found by simply summing over the numerator for each setting of y_i.

$$p(x_i) = p(y_i = 1) \prod_j p(x_{ij}|y_i = 1) + p(y_i = -1) \prod_j p(x_{ij}|y_i = -1)$$

Using the above, given some training instances D, you should be able to compute all the probabilities above needed to compute $p(y_i|x_i)$.

Laplacian smoothing:

To avoid 0-probabilities in the calculations above, we can simply add a small value to each count, and likewise update the normalizer so terms sum to 1. For

$$p(x_{ij} = 1|y_i = 1) = \frac{\#(x_{ij} = 1, y_i = 1) + \epsilon}{\#(y_i = 1) + 2\epsilon}$$

Commonly, $\epsilon = 1$ is used ("plus one" smoothing).

Multinomial Naive Bayes

The preceeding assumes a binary event model; that is, $x_{ij} \in \{0, 1\}$. Alternatively, we can use term frequencies; i.e. $x_{ij} \in \mathcal{N}_+$. The term probabilities become:

$$p(x_{ij} = 1|y_i = 1) = \frac{T_{1j}}{\sum_k T_{1k}}$$

where T_{cj} is the number occurrences of term j in documents where $y = c$. E.g., count all the occurrences of the term j in documents where the true class label is c. You can use the analogous equation for class -1, $p(x_{ij} = 1|y_i = -1)$.

Smoothing operates differently for Multinomial than Bernoulli Naive Bayes:

$$p(x_{ij} = 1|y_i = 1) = \frac{T_{1j} + \epsilon}{|V|\epsilon + \sum_k T_{1k}}$$

where $|V|$ is the number of unique terms in the vocabulary.

Note that in Multinomial Naive Bayes, to classify a new document, we only multiply terms that occur in the document:

$$p(y_i|x_i) \propto p(y_i) \prod_{j \in n_i} p(x_{ij}|y_i)$$

where n_i iterates over **tokens**, rather then **terms**. That is, if the term *dog* occurs twice in document i, its corresponding probability $p(\text{dog}|y_i)$ will appear twice in the product above.

See your book (Ch13[1]) for more details.

[1] http://nlp.stanford.edu/IR-book/pdf/13bayes.pdf