**Naive Bayes text classification [2]**

This is one of the first and time optimal supervised learning method. We describe the *multinomial Naive Bayes* or *multinomial NB* model, a probabilistic learning method. The probability of a document D being in class C is computed as

|  |  |  |  |
| --- | --- | --- | --- |
| $\displaystyle P(c\vert d) \propto P(c) \prod_{1 \leq \tcposindex \leq n_d} P(\tcword_\tcposindex\vert c)$ |  |  | (1) |

Where P (t k |c) the conditional probability of term tk occurring in a document of class c . We interpret P (t k |c) as a measure of how much evidence tk contributes that c is the correct class. P( c ) Is the prior probability of a document occurring in class C If a document's terms do not provide clear evidence for one class versus another, we choose the one that has a higher prior probability. {t1 t2 t3… tn­d} Are the tokens in d that are part of the vocabulary we use for classification and nd is the number of such tokens in d. For example {t1 t2 t3… tn­d} for the one-sentence document “India, Brazil are BRIC Nations” might be {India , Brazil ,Bric , Nations }, with nd = 4 if we treat the terms and the as stop words.

In text classification, our goal is to find the *best* class for the document. The best class in NB classification is the most likely or *maximum a posteriori* (MAP) class Cmap

|  |  |
| --- | --- |
| \begin{displaymath}c_{map} = \argmax_{\tcjclass \in \mathbb{C}} \hat{P}(\tcjc... ...posindex \leq n_d} \hat{P}(\tcword_\tcposindex\vert\tcjclass). \end{displaymath} | (2) |

We write P` for P because we do not know the true values of the parameters

P(c) and P (tk|c), but estimate them from the training set as we will see in a moment. In Equation 2 , many conditional probabilities are multiplied, one for each position  . This can result in a floating point underflow. It is therefore better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. The class with the highest log probability score is still the most probable; *log (xy) =log(x) + log(y)* and the logarithm function is monotonic. Hence, the maximization that is actually done in most implementations of NB is:

|  |  |  |  |
| --- | --- | --- | --- |
| $\displaystyle c_{map} = \argmax_{\tcjclass \in \mathbb{C}} \ [ \log \hat{P}(\tc... ...{1 \leq \tcposindex \leq n_d} \log \hat{P}(\tcword_\tcposindex\vert\tcjclass)].$ |  |  | (3) |

Equation 3 has a simple interpretation. Each conditional parameter log P` (tk|c) is a weight that indicates how good an indicator tk is for c. Similarly, the prior log P`(c ) is a weight that indicates the relative frequency of c more frequent classes are more likely to be the correct class than infrequent classes. The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class, and Equation 3 selects the class for which we have the most evidence. *Maximum likelihood estimate* (MLE; probtheory), which is simply the relative frequency and corresponds to the most likely value of each parameter given the training data.

For the priors this estimate is:

|  |  |  |  |
| --- | --- | --- | --- |
| $\displaystyle \hat{P}(\tcjclass) = \frac{N_c}{N},$ |  |  | (4) |

Where Nc is the number of documents in class C and Nis the total number of documents. We estimate the conditional probability P` (t|c) as the relative frequency of term t in documents belonging to class c

|  |  |
| --- | --- |
| \begin{displaymath} \hat{P}(\tcword\vert c) = \frac{T_{c\tcword}}{\sum_{\tcword' \in V} T_{c\tcword'}}, \end{displaymath} | (5) |

Where Tct is the number of occurrences of *t* in training documents from class *C* including multiple occurrences of a term in a document. The problem with the MLE estimate is that it is zero for a term-class combination that did not occur in the training data. If the term WTO in the training data only occurred in China documents, then the MLE estimates for the other classes, for example UK, will be zero: To eliminate zeros, we use *add-one* or *Laplace* *smoothing*, which simply adds one to each count

|  |  |
| --- | --- |
| \begin{displaymath} \hat{P}(\tcword\vert c) = \frac{T_{c\tcword}+1}{\sum_{\tcwor... ...frac{T_{c\tcword}+1}{(\sum_{\tcword' \in V} T_{c\tcword'})+B}, \end{displaymath} | (6) |

Where B = | V | is the number of terms in the vocabulary. Add-one smoothing can be interpreted as a uniform prior (each term occurs once for each class) that is then updated as evidence from the training data comes in. Note that this is a prior probability for the occurrence of a *term* as opposed to the prior probability of a *class* which we estimate in Equation 4 on the document level.