This is the algorithm which I use in naïve bayes for text classification

```
TrainMultinomialNB(ℂ, 𝔻)
V ← ExtractVocabulary(𝔻)
N ← CountDocs(𝔻)
for each c ∈ ℂ
do N_c ← CountDocsInClass(𝔻, c)
    prior[c] ← N_c / N
    text_c ← ConcatenateTextOfAllDocsInClass(𝔻, c)
    for each t ∈ V
    do T_ct ← CountTokensOfTerm(text_c, t)
    for each t ∈ V
    do condprob[t][c] ← (T_ct + 1) / Σ_t' (T_ct' + 1)
return V, prior, condprob
```

$$c_{\text{map}} = \underset{c \in \mathbb{C}}{\arg\max}\, \hat{P}(c|d) = \underset{c \in \mathbb{C}}{\arg\max}\ \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}(t_k|c).$$

$$c_{\text{map}} = \underset{c \in \mathbb{C}}{\arg\max}\, [\log \hat{P}(c) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|c)].$$

To eliminate zeros, we use *add-one* or *Laplace smoothing*, which simply adds one to each count (cf. Section 11.3.2):

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B'},$$

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 (13.5) on the document level.