a decision tree that detects spam. Finding the right features is paramount for this task, so design your feature set carefully.

Exercise 16.4 [⋆⋆]

Another important application of text categorization is the detection of 'adult' content, that is, content that is not appropriate for children because it is sexually explicit. Collect training and test sets of adult and non-adult material from the World Wide Web and build a decision tree that can block access to adult material.

Exercise 16.5 [⋆⋆]

Collect a reasonable amount of text written by yourself and by a friend. You may want to break up individual texts (e.g., term papers) into smaller pieces to get a large enough set. Build a decision tree that automatically determines whether you are the author of a piece of text. Note that it is often the 'little' words that give an author away (for example, the relative frequencies of words like *because* or *though*).

Exercise 16.6 [⋆⋆]

Download a set of English and non-English texts from the World Wide Web or use some other multilingual source. Build a decision tree that can distinguish between English and non-English texts. (See also exercise 6.10.)

16.2 Maximum Entropy Modeling

Maximum entropy modeling is a framework for integrating information from many heterogeneous information sources for classification. The data for a classification problem is described as a (potentially large) number of features. These features can be quite complex and allow the experimenter to make use of prior knowledge about what types of information are expected to be important for classification. Each feature corresponds to a constraint on the model. We then compute the *maximum entropy model*, the model with maximum entropy of all the models that satisfy the constraints. This term may initially seem perverse, since we have spent most of the book trying to minimize the (cross) entropy of data according to models, but the idea is that we do not want to go beyond the data. If we chose a model with less entropy, we would add 'information' to the model that is not justified by the empirical evidence available to us. Choosing the maximum entropy model is motivated by the desire to preserve as much uncertainty as possible.

We have simplified matters in this chapter by neglecting the problem of feature selection (we use the same 20 features throughout). In maximum entropy modeling, feature selection and training are usually integrated.

Ideally, this enables us to specify all potentially relevant information at the beginning, and then to let the training procedure worry about how to come up with the best model for classification. We will only introduce the basic method here and refer the reader to the Further Reading for feature selection.

The features f_i are binary functions that can be used to characterize any property of a pair $(\vec{x}, c)$, where $\vec{x}$ is a vector representing an input element (in our case the 20-dimensional vector of word weights representing an article as in table 16.3), and c is the class label (1 if the article is in the "earnings" category, 0 otherwise). For text categorization, we define features as follows:

$$f_i(\vec{x}_j, c) = \begin{cases} 1 & \text{if } s_{ij} > 0 \text{ and } c = 1 \\ 0 & \text{otherwise} \end{cases} \tag{16.3}$$

Recall that s_{ij} is the term weight for word i in Reuters article j. Note that the use of binary features is different from the rest of this chapter: The other classifiers use the magnitude of the weight, not just the presence or absence of a word.[5]

For a given set of features, we first compute the *expectation* of each feature based on the training set. Each feature then defines the constraint that the expectation of the feature in our final maximum entropy model must be the same as this *empirical expectation*. Of all probability distributions that obey these constraints, we attempt to find the *maximum entropy distribution*, the one with the highest entropy. One can show that there exists a unique such maximum entropy distribution and there exists an algorithm, generalized iterative scaling, which is guaranteed to converge to it.

EMPIRICAL EXPECTATION

MAXIMUM ENTROPY DISTRIBUTION

The model class for the particular variety of maximum entropy modeling that we introduce here is *loglinear models* of the following form:

LOGLINEAR MODELS

$$\mathrm{p}(\vec{x}, c) = \frac{1}{Z} \prod_{i=1}^{K} \alpha_i^{f_i(\vec{x}, c)} \tag{16.4}$$

where K is the number of features, α_i is the weight for feature f_i and Z is a normalizing constant (commonly called "the partition function"), used

5. The maximum entropy approach is not in principle limited to binary features. Generalized iterative scaling, which we introduce below, requires merely that weights are non-negative and that their sum is bounded. However, binary features have generally been employed because they improve the efficiency of the computationally intensive reestimation procedure.

to ensure that a probability distribution results. To use the model for text categorization, we compute $p(\vec{x}, 0)$ and $p(\vec{x}, 1)$ and, in the simplest case, choose the class label with the greater probability.

FEATURES

Note that, in this section, *features* contain information about the *class* of the object in addition to the *'measurements'* of the object we want to classify. Here, we are following most publications on maximum entropy modeling in defining feature in this sense. The more common use of the term "feature" (which we have adopted for the rest of the book) is that it only refers to some characteristic of the object, independent of the class the object is a member of.

Equation (16.4) defines a loglinear model because, if we take logs on both sides, then $\log p$ is a linear combination of the logs of the weights:

$$\log p(\vec{x}, c) = -\log Z + \sum_{i=1}^{K} f_i(\vec{x}, c) \log \alpha_i \qquad (16.5)$$

Loglinear models are a general and very important class of models for classification with categorical variables. Other examples of the class are logistic regression (McCullagh and Nelder 1989), decomposable models (Bruce and Wiebe 1999), and the HMMs and PCFGs of earlier chapters. We introduce the maximum entropy modeling approach here because maximum entropy models have recently been widely used in Statistical NLP and because it is an application of the important maximum entropy principle.

16.2.1 Generalized iterative scaling

GENERALIZED ITERATIVE SCALING

Generalized iterative scaling is a procedure for finding the maximum entropy distribution p^* of form (16.4) that obeys the following set of constraints:

$$E_{p^*} f_i = E_{\tilde{p}} f_i \qquad (16.6)$$

In other words, the expected value of f_i for p^* is the same as the expected value for the empirical distribution $\tilde{p}$ (in other words, for the training set).

The algorithm requires that the sum of the features for each possible $(\vec{x}, c)$ be equal to a constant C:[6]

$$\forall \vec{x}, c \quad \sum_i f_i(\vec{x}, c) = C \qquad (16.7)$$

6. See Berger et al. (1996) for *Improved Iterative Scaling*, a variant of generalized iterative scaling that does not impose this constraint.

In order to fulfill this requirement, we define C as the greatest possible feature sum (over all possible data, not just observed data):

$$C \overset{\text{def}}{=} \max_{\vec{x},c} \sum_{i=1}^{K} f_i(\vec{x}, c)$$

and add a feature f_{K+1} that is defined as follows:

$$f_{K+1}(\vec{x}, c) = C - \sum_{i=1}^{K} f_i(\vec{x}, c)$$

Note that this feature is not binary, in contrast to the others.

$E_\text{p} f_i$ is defined as (section 2.1.5):

$$E_\text{p} f_i = \sum_{\vec{x},c} \text{p}(\vec{x}, c) f_i(\vec{x}, c) \tag{16.8}$$

where the sum is over the event space, that is, all possible vectors $\vec{x}$ and class labels c. The empirical expectation is easy to compute:

$$E_{\tilde{\text{p}}} f_i = \sum_{\vec{x},c} \tilde{\text{p}}(\vec{x}, c) f_i(\vec{x}, c) = \frac{1}{N} \sum_{j=1}^{N} f_i(\vec{x}_j, c) \tag{16.9}$$

where N is the number of elements in the training set and we use the fact that the empirical probability for a pair that doesn't occur in the training set is 0.

In general, the maximum entropy distribution $E_\text{p} f_i$ cannot be computed efficiently since it would involve summing over all possible combinations of $\vec{x}$ and c, a huge or infinite set. Instead, people use the following approximation where only empirically observed $\vec{x}$ are considered (Lau 1994: 25):

$$E_\text{p} f_i \approx \sum_{\vec{x},c} \tilde{\text{p}}(\vec{x})\, \text{p}(c|\vec{x}) f_i(\vec{x}, c) = \frac{1}{N} \sum_{j=1}^{N} \sum_{c} \text{p}(c|\vec{x}_j) f_i(\vec{x}_j, c) \tag{16.10}$$

where c still ranges over all possible classes, in our case $c \in \{0, 1\}$.

Now we have all the pieces to state the generalized iterative scaling algorithm:

1. Initialize $\{\alpha_i^{(1)}\}$. Any initialization will do, but usually we choose $\alpha_i^{(1)} = 1, \forall 1 \le j \le K + 1$. Compute $E_{\tilde{\text{p}}} f_i$ as shown above. Set $n = 1$.

2. Compute $\text{p}^{(n)}(\vec{x}, c)$ for the distribution $\text{p}^{(n)}$ given by the $\{\alpha_i^{(n)}\}$ for each element $(\vec{x}, c)$ in the training set:

$$\text{p}^{(n)}(\vec{x}, c) = \frac{1}{Z} \prod_{i=1}^{K+1} (\alpha_i^{(n)})^{f_i(\vec{x},c)} \quad \text{where} \quad Z = \sum_{\vec{x},c} \prod_{i=1}^{K+1} (\alpha_i^{(n)})^{f_i(\vec{x},c)} \tag{16.11}$$

$\vec{x}$ *profit*	c "earnings"	f_1	f_2	$\beta = f_1 \log \alpha_1 + f_2 \log \alpha_2$	2^β
(0)	0	0	1	1	2
(0)	1	0	1	1	2
(1)	0	0	1	1	2
(1)	1	1	0	2	4

Table 16.6 An example of a maximum entropy distribution in the form of equation (16.4). The vector $\vec{x}$ consists of a single element, indicating the presence or absence of the word *profit* in the article. There are two classes (member of "earnings" or not). Feature f_1 is 1 if and only if the article is in "earnings" and *profit* occurs. f_2 is the "filler" feature f_{K+1}. For one particular choice of the parameters, namely $\log \alpha_1 = 2.0$ and $\log \alpha_2 = 1.0$, we get after normalization ($Z = 2 + 2 + 2 + 4 = 10$) the following maximum entropy distribution: $\mathrm{p}(0,0) = \mathrm{p}(0,1) = \mathrm{p}(1,0) = 2/Z = 0.2$ and $\mathrm{p}(1,1) = 4/Z = 0.4$. An example of a data set with the same empirical distribution is $((0,0),(0,1),(1,0),(1,1),(1,1))$.

3. Compute $E_{\mathrm{p}^{(n)}} f_i$ for all $1 \le i \le K+1$ according to equation (16.10).

4. Update the parameters α_i:

$$\alpha_i^{(n+1)} = \alpha_i^{(n)} \left(\frac{E_{\tilde{\mathrm{p}}} f_i}{E_{\mathrm{p}^{(n)}} f_i} \right)^{\frac{1}{C}} \tag{16.12}$$

5. If the parameters of the procedure have converged, stop, otherwise increment n and go to 2.

We present the algorithm in this form for readability. In an actual implementation, it is more convenient to do the computations using logarithms.

One can show that this procedure converges to a distribution p^* that obeys the constraints (16.6), and that of all such distributions it is the one that maximizes the entropy $H(\mathrm{p})$ and the likelihood of the data. Darroch and Ratcliff (1972) show that this distribution always exists and is unique.

A toy example of a maximum entropy distribution that generalized iterative scaling will converge to is shown in table 16.6.

Exercise 16.7 [⋆]

What are the classification decisions for the distribution in table 16.6? Compute $P(\text{"earnings"}|\textit{profit})$ and $P(\text{"earnings"}|\neg\textit{profit})$.

Does *profit*	Is topic "earnings"?	
occur?	YES	NO
YES	20	9
NO	8	13

Table 16.7 An empirical distribution whose corresponding maximum entropy distribution is the one in table 16.6.

Exercise 16.8 [⋆]

Show that the distribution in table 16.6 is a fixed point for generalized iterative scaling. That is, computing one iteration should leave the distribution unchanged.

Exercise 16.9 [⋆]

Consider the distribution in table 16.7. Show that for the features defined in table 16.6, this distribution has the same feature expectations E_{p} as the one in table 16.6.

Exercise 16.10 [⋆]

Compute a number of iterations of generalized iterative scaling for the data in table 16.7 (using the features defined in table 16.6). The procedure should converge towards the distribution in table 16.6.

Exercise 16.11 [⋆⋆]

Select one of exercises 16.3 through 16.6 and build a maximum entropy model for the corresponding text categorization task.

16.2.2 Application to text categorization

We have already suggested how to define appropriate features for text categorization in equation (16.3). For the task of identifying Reuters "earnings" articles we end up with 20 features, each corresponding to one of the selected words, and the f_{K+1} feature introduced at the start of the last subsection, defined so that the features added to $C = 20$.

We trained on the 9603 training set articles. Table 16.8 shows the weights found by generalized iterative scaling after convergence (500 iterations). The features with the highest weights are *cts*, *profit*, *net* and *losst*. If we use $P(\text{"earnings"}|\vec{x}) > P(\neg\text{"earnings"}|\vec{x})$ as our decision rule, we get the classification results in table 16.9. Classification accuracy is 96.2%.

An important question in an implementation is when to stop the iteration. One way to test for convergence is to compute the log difference

Word	Feature weight	
w^i	α_i	$\log_e \alpha_i$
vs	2.696	0.992
mln	1.079	0.076
cts	12.303	2.510
;	0.448	−0.803
&	0.450	−0.798
000	0.756	−0.280
loss	4.032	1.394
’	0.993	−0.007
"	1.502	0.407
3	0.435	−0.832
profit	9.701	2.272
dlrs	0.678	−0.388
1	1.193	0.177
pct	0.590	−0.528
is	0.418	−0.871
s	0.359	−1.025
that	0.703	−0.352
net	6.155	1.817
lt	3.566	1.271
at	0.490	−0.713
f_{K+1}	0.967	−0.034

Table 16.8 Feature weights in maximum entropy modeling for the category “earnings” in Reuters.

“earnings”	“earnings” correct?	
assigned?	YES	NO
YES	1014	53
NO	73	2159

Table 16.9 Classification results for the distribution corresponding to table 16.8 on the test set. Classification accuracy is 96.2%.

between empirical and estimated feature expectations ($\log E_{\tilde{p}} - \log E_{p^{(n)}}$), which should approach zero. Ristad (1996) recommends to also look at the largest α when doing iterative scaling. If the largest weight becomes too large, then this indicates a problem with either the data representation or the implementation.

When is the maximum entropy framework presented in this section appropriate as a classification method? A characteristic of the maximum entropy systems that are currently practical for large data sets is the restriction to binary features. This is a shortcoming in some situations. In text categorization, we often need a notion of "strength of evidence" which goes beyond simply recording presence or absence of evidence. But this does not appear to have hurt us much here. (This perhaps partly reflects how easy this classification problem is: Simply classifying based on whether the *cts* feature is non-zero yields an accuracy of 91.2%.)

Generalized iterative scaling can also be computationally expensive due to slow convergence (but see (Lau 1994) for suggestions for speeding up convergence). For binary classification, the loglinear model defines a linear separator that is in principle no more powerful than *Naive Bayes* or *linear regression*, classifiers that can be trained more efficiently. However, it is important to stress that, apart from the theoretical power of a classification method, the training procedure is crucial. Unlike Naive Bayes, generalized iterative scaling takes dependence between features into account: if one duplicated a feature, then the weight of each instance of it would be halved. If feature dependence is not expected to a be a problem, then Naive Bayes is a better choice than maximum entropy modeling.

NAIVE BAYES
LINEAR REGRESSION

Finally, the lack of smoothing can also cause problems. For example, if we have a feature that always predicts a certain class, then this feature may get an excessively high weight. One way to deal with this is to 'smooth' the empirical data by adding events that did not occur. In practice, features that occur less than five times are usually eliminated.

One of the strengths of maximum entropy modeling is that it offers a framework for specifying all possibly relevant information. The attraction of the method lies in the fact that arbitrarily complex features can be defined if the experimenter believes that these features may contribute useful information for the classification decision. For example, Berger et al. (1996: 57) define a feature for the translation of the preposition *in* from English to French that is 1 if and only if *in* is translated as *pendant* and *in* is followed by the word *weeks* within three words. There is also no need to worry about heterogeneity of features or weighting fea-