Shrinking Targets for Regularization of Logistic Regression

Rob Abbott
June 14, 2013

1 Introduction

In machine learning, regularization serves to counteract a base algorithm’s tendency to overfit. This paper specifically explores the technique of target shrinking for logistic regression. We show that this is equivalent to adding label noise. A variation on the technique is proposed and evaluated on the spam dataset.

Figure 1: Overfitting is characterized by a wide gap in performance on test and training sets. Often improving performance on the training set corresponds to declining performance on the test set as seen here. Overfitting is especially a problem when there are many sparse features relative to the number of training examples.
For logistic regression, we can regularize by shrinking the objective’s target value. Logistic regression predicts class labels with a probability and incurs loss based on the difference between this prediction and the target (conventionally the target is $\in \{0, 1\}$). By shrinking the targets from $\{0, 1\}$ to $\{0 + \epsilon_1, 1 - \epsilon_2\}$ the algorithm is punished for making absolute claims about predicted labels. This encourages the algorithm to choose a softer model which hopefully generalizes better. This paper proposes a technique for varying $\epsilon$ on a per example basis by considering density of training examples nearby.

The more widely known L1 and L2 regularization are members of a class of regularization techniques which accomplish their goal by adding a term to the objective function. This leads to altered weight update rules which prioritize less complex or less extreme weight vectors.

Another method of regularization is to add noise to training examples. Doing so can discourage overfitting. This works by preventing the model from zeroing in on specific examples. Noise may be added to either the features or the labels and this might be used in conjunction with other regularization techniques such as early stopping.

2 Background

2.1 Logistic Regression

Logistic regression is a discriminative learning algorithm for binary classification which uses a logistic function to map weighted feature values into class probabilities ($\text{class } y \in \{0, 1\}$). Weight, Features $w, x \in \mathbb{R}^n$).

Logistic regression predicts with a Bernoulli distribution where $p = \frac{1}{1+e^{-w \cdot x}}$. That is, $P(Y = y|x; w) = \left(\frac{1}{1+e^{-w \cdot x}}\right)^y \left(1 - \frac{1}{1+e^{-w \cdot x}}\right)^{(1-y)}$

Note that As $w \cdot x \to \infty$, then $P(Y = 1|x; w) \to 1$ and $P(Y = 0|x; w) \to 0$

3 Shrinking Targets

Shrinking targets opts to target $y - \epsilon$ instead of $y$.

So, instead of using: $w = \arg \min_w - \sum_j y_j \log \frac{1}{1+e^{-w \cdot x_j}} + (1 - y_j) \log \left(1 - \frac{1}{1+e^{-w \cdot x_j}}\right)$

We use: $w = \arg \min_w - \sum_j (y_j - \epsilon_j) \log \frac{1}{1+e^{-w \cdot x_j}} + (1 - (y_j - \epsilon_j)) \log \left(1 - \frac{1}{1+e^{-w \cdot x_j}}\right)$

In the simplest case $\epsilon_j = \epsilon$ for all $j$ and no distinction is made between positive and negative labels.
3.1 Equivalence to Adding Label Noise

I can be shown that shrinking the targets is equivalent to adding a small amount of noise to each label. (see presentation)

3.2 Alternate Shrinking Targets

One concern with shrinking target regularization is that it applies the same modification to all training examples, while some are more in need of it.
Figure 4: This plot assumes the correct label is $y = 1$. By shrinking the targets we alter the loss function leading to singularities on both ends instead of just the *incorrect* $y = 0$ end. In the graph, 20% shrinking is used to exaggerate the effect.

than others. I considered this a problem of data sparsity as the instantiated training examples had excessive influence on the final model. To ameliorate this I made the amount of shrinking correspond to the density of other examples near any given example.

First I calculated the cosine similarity between training examples using their tf-idf feature vectors. I then made a very crude measure of local density by calculating the median of these cosine similarities for each example. Next I rescaled these from $[0,1]$ to $[0, \text{shrink parameter}]$ (e.g. $[0,1] \rightarrow [0, 0.01]$). Finally I centered this so that the mean across all training examples was the shrink parameter. This vector over training examples became my new per-example shrink parameter.

The motivation for rescaling and centering as I did was to directly compare with the fixed shrink and to not let this regularization overwhelm the true data. The method I used also ensured that the new shrink values had a fixed range equal in size to and centered by mass on the shift parameter - that meant they were $\geq 0$ and $\leq 2 \times \text{shrink parameter}$.

It should be noted that this is a very crude method and there are some obvious ways to improve it. Taking into account the class distribution both locally and globally is likely to yield better results. Median of similarity measures is an exceedingly crude method of capturing nearby example density - better methods are likely to yield better results. The scaling and
Figure 5: The distribution of shrinking parameters for the spam dataset’s training examples when $\mu = 0.01$. This is the amount by which the target class $y$ is pushed away from the edge ($\{0,1\}$). High parameter values correspond to few similar examples, low values correspond to many similar examples.

shifting was arbitrary. Lastly, this is not on any statistically sound footing, it is an arbitrary heuristic.

4 Data

For this paper I have only explored the spam dataset consisting of 2000 examples and 2000 sparse features. The features are binary and represent whether a particular word occurred in the example. The original set of features underwent information gain feature selection and only the top 2000 were maintained. Because of this I elected to not run SVD on the features though that may be worth trying. I randomly split the dataset into 75% training and 25% test (1500 and 500 examples respectively).

5 Procedure

I used a variable learning rate $\eta$ which increased by 20% when the loss was found to decrease, and halved and tried again when the loss increased.
Although I experimented with mini-batch/stochastic gradient descent and Newton’s method, the figures presented here used simple batch gradient descent. Weights presented do not include the bias term. L1 & L2 regularization did not include the bias term.

6 Results

<table>
<thead>
<tr>
<th>Shrink</th>
<th>Fixed Training</th>
<th>Training</th>
<th>Test</th>
<th>Test</th>
<th>Variable Training</th>
<th>Training</th>
<th>Test</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0001</td>
<td>0.00057</td>
<td>0.18726</td>
<td>0.00053</td>
<td>0.18593</td>
<td>0.00094</td>
<td>0.12854</td>
<td>0.00279</td>
<td>0.14243</td>
</tr>
<tr>
<td>0.001</td>
<td>0.00301</td>
<td>0.14573</td>
<td>0.00279</td>
<td>0.14243</td>
<td>0.01937</td>
<td>0.12260</td>
<td>0.01863</td>
<td>0.12118</td>
</tr>
<tr>
<td>0.005</td>
<td>0.01094</td>
<td>0.12854</td>
<td>0.01639</td>
<td>0.12118</td>
<td>0.015</td>
<td>0.12713</td>
<td>0.02620</td>
<td>0.12416</td>
</tr>
<tr>
<td>0.01</td>
<td>0.01937</td>
<td>0.12600</td>
<td>0.01863</td>
<td>0.12102</td>
<td>0.02</td>
<td>0.03417</td>
<td>0.12911</td>
<td>0.03272</td>
</tr>
<tr>
<td>0.015</td>
<td>0.02713</td>
<td>0.12574</td>
<td>0.02620</td>
<td>0.12416</td>
<td>0.03</td>
<td>0.04778</td>
<td>0.13548</td>
<td>0.03645</td>
</tr>
<tr>
<td>0.02</td>
<td>0.03417</td>
<td>0.12911</td>
<td>0.03272</td>
<td>0.12631</td>
<td>0.034</td>
<td>0.07165</td>
<td>0.15001</td>
<td>0.05026</td>
</tr>
</tbody>
</table>

Table 1: Log loss with varying shrink parameter. The variable algorithm always wins albeit barely! L2 regularization was able to achieve testing loss as low as 0.11904.

Figure 6: Test and training loss. These require far fewer epochs when using stochastic gradient descent. (shrink parameter = 0.01, lambda = 0.5)
Figure 7: Weights after 400 epochs for the various regularization methods. Note how L1 goes off the chart in both the x and y dimensions.

7 Discussion

L2 still beat out shrinking but the variable shrinking beat the fixed shrinking.