# Feature selection, $L_1$ vs. $L_2$ regularization, and rotational invariance

**Andrew Y. Ng** ANG@CS.STANFORD.EDU
Computer Science Department, Stanford University, Stanford, CA 94305, USA

## Abstract

We consider supervised learning in the presence of very many irrelevant features, and study two different regularization methods for preventing overfitting. Focusing on logistic regression, we show that using $L_1$ regularization of the parameters, the sample complexity (i.e., the number of training examples required to learn "well,") grows only *logarithmically* in the number of irrelevant features. This logarithmic rate matches the best known bounds for feature selection, and indicates that $L_1$ regularized logistic regression can be effective even if there are exponentially many irrelevant features as there are training examples. We also give a lower-bound showing that any rotationally invariant algorithm—including logistic regression with $L_2$ regularization, SVMs, and neural networks trained by backpropagation—has a worst case sample complexity that grows at least *linearly* in the number of irrelevant features.

## 1. Introduction

We consider supervised learning in settings where there are many input features, but where there is a small subset of the features that is sufficient to approximate the target concept well.

In supervised learning settings with many input features, overfitting is usually a potential problem unless there is ample training data. For example, it is well-known that for unregularized discriminative models fit via training-error minimization, sample complexity (i.e., the number of training examples needed to learn "well") grows linearly with the VC dimension. Further, the VC dimension for most models grows about linearly in the number of parameters (Vapnik, 1982), which typically grows at least linearly in the number of input features. Thus, unless the training set size is large relative to the dimension of the input, some special mechanism—such as regularization, which encourages the fitted parameters to be small—is usually needed to prevent overfitting.

In this paper, we focus on logistic regression, and study the behavior of two standard regularization methods when they are applied to problems with many irrelevant features. The first, $L_1$ regularization, uses a penalty term which encourages the sum of the absolute values of the parameters to be small. The second, $L_2$ regularization, encourages the sum of the squares of the parameters to be small. It has frequently been observed that $L_1$ regularization in many models causes many parameters to equal zero, so that the parameter vector is sparse. This makes it a natural candidate in feature selection settings, where we believe that many features should be ignored. For example, linear least-squares regression with $L_1$ regularization is called the Lasso algorithm (Tibshirani, 1996), which is known to generally give sparse feature vectors. Another example of learning using $L_1$ regularization is found in (Zheng et al., 2004).

In this paper, we prove that for logistic regression with $L_1$ regularization, sample complexity grows only *logarithmically* in the number of irrelevant features (and at most polynomially in all other quantities of interest). Logistic regression with $L_1$ regularization is an appealing algorithm since it requires solving only a convex optimization problem. Further, the logarithmic dependence on the input dimension matches the best known bounds proved in various feature selection contexts (e.g., Ng, 1998; Ng & Jordan, 2001; Littlestone, 1988; Helmbold et al., 1996; Kivinen & Warmuth, 1994).

We also consider logistic regression with $L_2$ regularization. (E.g., Nigam et al., 1999). We show that this gives a rotationally invariant algorithm, and that any

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004. Copyright 2004 by the first author.

rotationally invariant algorithm—which also includes SVMs, neural networks, and many other algorithms—has a worst case sample complexity that grows at least *linearly* in the number of irrelevant features, even if only a single feature is relevant. This suggests that these algorithms may not be effective in settings where only a few features are relevant, and the number of training examples is significantly smaller than the input dimension.

## 2. Preliminaries

We consider a supervised learning problem where we are given a set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ of $m$ training examples drawn i.i.d. from some distribution $\mathcal{D}$. Here, $x^{(i)} \in [-1,1]^n$ are the $n$-dimensional inputs, and $y^{(i)} \in \{0,1\}$ are the labels. For notational convenience, we assume that the last coordinate of the input vectors $x_n^{(i)} = 1$ always, so that the intercept term needs not be treated separately. We will focus on logistic regression, so our model will be

$$p(y = 1|x;\theta) = \frac{1}{1+\exp(-\theta^T x)}, \qquad (1)$$

where $\theta \in \mathbb{R}^n$ are the parameters of our model.

One way to describe regularized logistic regression is as the finding the parameters $\theta$ that solve following optimization problem:

$$\arg\max_\theta \sum_{i=1}^m \log p(y^{(i)}|x^{(i)};\theta) - \alpha R(\theta), \qquad (2)$$

where $R(\theta)$ is a regularization term that is used to penalize large weights/parameters. If $R(\theta) \equiv 0$, then this model is the standard, unregularized, logistic regression model with its parameters fit using the maximum likelihood criteria. If $R(\theta) = ||\theta||_1 = \sum_{i=1}^n |\theta_i|$, then this is $L_1$ regularized logistic regression. If $R(\theta) = ||\theta||_2^2 = \sum_{i=1}^n \theta_i^2$, this is $L_2$ regularized logistic regression.

In the optimization problem in Equation (2), the parameter $\alpha \geq 0$ controls a tradeoff between fitting the data well, and having well-regularized/small parameters. In this paper, it will sometimes be useful to consider an alternative way of parameterizing this tradeoff. Specifically, we will also consider the constrained optimization problem:

$$\max_\theta \quad \sum_{i=1}^m \log p(y^{(i)}|x^{(i)};\theta) \qquad (3)$$

$$\text{subject to} \quad R(\theta) \leq B. \qquad (4)$$

For every solution $\theta$ to Equation (2) found using some particular value of $\alpha$, there is some corresponding value of $B$ in the optimization problem (3-4) that will give the same $\theta$. Thus, these are two equivalent reparameterizations of the same problem. Readers familiar with convex analysis (Rockafellar, 1970) may also verify the equivalence between the two problems by noting that the Lagrangian for the constrained optimization (3-4) is exactly the objective in the optimization (2) (plus a constant that does not depend on $\theta$), where here $\alpha$ is the Lagrange multiplier. Thus, (3-4) may be solved by solving (2) for an appropriate $\alpha$.

Because our logistic regression model is fit via (regularized) maximum likelihood, one natural metric for a fitted model's error is its negative loglikelihood (also called the "logloss") on test data:

$$\varepsilon^l(\theta) = \mathrm{E}_{(x,y)\sim\mathcal{D}}[-\log p(y|x;\theta)]. \qquad (5)$$

Here, the subscript "$(x,y) \sim \mathcal{D}$" indicates that the expectation is with respect to a test example $(x,y)$ drawn from $\mathcal{D}$. Our main theoretical results regarding $L_1$ regularization will use this error metric. Given a dataset $S$, we also define the empirical logloss on $S$ to be

$$\hat{\varepsilon}^l(\theta) = \hat{\varepsilon}^l_S(\theta) = \frac{1}{m}\sum_{i=1}^m -\log p(y^{(i)}|x^{(i)};\theta). \qquad (6)$$

Sometimes, we will also be interested in the 0/1 misclassification error of our algorithm. We define

$$\varepsilon^m(\theta) = \mathrm{P}_{(x,y)\sim\mathcal{D}}[t(1/(1+e^{-\theta^T x})) \neq y], \qquad (7)$$

where $t$ is a threshold function ($t(z) = 1$ if $z \geq 0.5$, $t(z) = 0$ otherwise). The empirical 0/1 misclassification error $\hat{\varepsilon}^m(\theta) = \hat{\varepsilon}^m_S(\theta)$ is also defined analogously to be the fraction of examples in $S$ that a model using parameter $\theta$ misclassifies.

It is straightforward to verify that, for the logistic regression model, we have $\varepsilon^l(\theta) \geq (\log 2)\cdot\varepsilon^m(\theta)$. Thus, an upper-bound on logloss also implies an upper-bound on misclassification error, and a lower-bound on misclassification error (such as given in Section 4) also implies a lower-bound on logloss.

## 3. $L_1$ regularized logistic regression

We are interested in supervised learning problems where there is a very large number $n$ of input features, but where there may be a small subset—say $r \ll n$ of them—that is sufficient to learn the target concept well. We will consider the following implementation of $L_1$ regularized logistic regression:

1. Split the data $S$ into a training set $S_1$ consisting of the first $(1-\gamma)m$ examples, and a hold-out cross

validation set $S_2$ consisting of the remaining $\gamma m$ examples.

2. For $B = 0, 1, 2, 4, \ldots, C$,

   Fit a logistic regression model using the training set $S_1$ only, by solving the optimization problem (3-4) with the specified value of $B$. Call the resulting parameter vector $\theta_B$.

3. Among the $\theta_B$'s from Step 2, select and output the one with the lowest hold-out error on $S_2$. I.e., pick $\theta = \arg\min_{i \in \{0,1,2,\ldots,C\}} \hat{\varepsilon}_{S_2}(\theta_i)$

Thus, this algorithm uses uses hold-out cross validation to select the regularization parameter $B$ used in (3-4). In Step 3 of the algorithm, we did not exactly specify the error metric $\hat{\varepsilon}_{S_2}$. If the goal is to minimize our expected logloss on the test data, it would make sense to use $\hat{\varepsilon}_{S_2}(\theta) = \hat{\varepsilon}^l_{S_2}(\theta)$ here. It will be this minimum logloss setting to which the theoretical results in this section apply. However, if the goal is to minimize $0/1$ misclassification error, then it would also make sense to pick the $\theta_i$ with the smallest misclassification error on the hold-out test set, and use $\hat{\varepsilon}_{S_2}(\theta) = \hat{\varepsilon}^m_{S_2}(\theta)$.

We want to show that if there is some hypothesis that attains low generalization error using only a small number $r$ of features, then $L_1$ regularized logistic regression will attain performance that is (nearly) as good as that of this hypothesis, even if the training set is small.

**Theorem 3.1:** *Let any $\epsilon > 0, \delta > 0, C > 0, 0 < \gamma < 1, K \geq 1$, and $m$ be fixed. Suppose there exists $r$ indices $1 \leq i_1, i_2, \ldots, i_r \leq n$, and a parameter vector $\theta^* \in \mathbb{R}^n$ such that only the $r$ corresponding components of $\theta^*$ are non-zero, and $|\theta_{i_j}| \leq K$ $(j = 1, \ldots, r)$. Suppose further that $C \geq rK$. Then, in order to guarantee that, with probability at least $1 - \delta$, the parameters $\hat{\theta}$ output by our learning algorithm does nearly as well as $\theta^*$, i.e., that*

$$\varepsilon^l(\hat{\theta}) \leq \varepsilon^l(\theta^*) + \epsilon, \tag{8}$$

*it suffices that*

$$m = \Omega\left((\log n) \cdot \mathrm{poly}(\mathrm{r}, \mathrm{K}, \log(1/\delta), 1/\epsilon, \mathrm{C})\right). \tag{9}$$

The main tools used to show this result are certain covering number bounds shown by (Bartlett, 1998; Zhang, 2002). The proof is given in Appendix A. This result shows that the sample complexity of our algorithm—that is, the number of training examples needed to learn "well"—grows only *logarithmically* in the number of irrelevant features. Thus, logistic regression with $L_1$ regularization is capable of learning in problems even where the number of irrelevant features may be far larger than the training set size.

Space constraints preclude a full discussion, but we also note that $C$ can be chosen automatically (as a function of $m$) so that the same bound as stated above holds, but with the dependence on $C$ removed. Further, by modifying the definition of $p(y|x;\theta)$, it is straightforward to generalize this result to $L_1$ regularized versions of other models from the generalized linear model family (McCullagh & Nelder, 1989), such as linear least squares regression.

## 4. Rotational invariance and $L_2$ regularization

Let $\mathcal{M} = \{M \in \mathbb{R}^{n \times n} | MM^T = M^TM = I, |M| = 1\}$ be the class of rotational matrices.[1] Thus, if $x \in \mathbb{R}^n$ and $M \in \mathcal{M}$, then $Mx$ is $x$ rotated through some angle around the origin.[2]

Given a training set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^m$, we let $MS = \{(Mx^{(i)}, y^{(i)})\}_{i=1}^m$ denote the training set with all the inputs rotated according to $M$. Given a learning algorithm $L$, we let $L[S](x)$ denote the predicted label resulting from using the learning algorithm to train on a dataset $S$, and using the resulting hypothesis/classifier to make a prediction on $x$.

**Definition 4.1:** Given a (deterministic) learning algorithm $L$, we say that it is **rotationally invariant** if, for any training set $S$, rotational matrix $M \in \mathcal{M}$, and test example $x$, we have that $L[S](x) = L[S'](x')$, where $S' = MS$, $x' = Mx$. More generally, if $L$ is a stochastic learning algorithm so that its predictions are random, we say that it is rotationally invariant if, for any $S, M, x$, the predictions $L[S](x)$ and $L[S'](x')$ have the same distribution.

Some readers familiar with logistic regression may already recognize that its $L_2$ regularized version is rotationally invariant. But for the sake of completeness, we will state and formally prove this here.

**Proposition 4.2:** *$L_2$ regularized logistic regression (Equation 2, with $\alpha > 0$) is rotationally invariant.*

**Proof.** Let any $S, M, x$ be given, and let $S' = MS$, $x' = Mx$. Because $M^TM = I$, we have $\frac{1}{1+\exp(-\theta^T x)} =$

[1]If we drop the condition that the determinant is $|M| = 1$, then we obtain the class of all orthogonal matrices, which may include a reflection as well as a rotation. (Strang, 1988) Using the more restrictive set as we do here leads to a slightly stronger theoretical result.

[2]If we are using the convention (mentioned earlier) that $x_n = 1$ always to handle the intercept term, then we may restrict attention to matrices $M$ where $M_{nn} = 1, M_{jn} = 0$ $(j < n)$, so that the final coordinate is not changed by $M$. This makes no difference to our results.

$\frac{1}{1+\exp(-(M\theta)^T(Mx))}$, and thus $p(y|x;\theta) = p(y|Mx;M\theta)$. Further, $R(\theta) = \theta^T\theta = (M\theta)^T(M\theta) = R(M\theta)$. Define $J(\theta) = \sum_{i=1}^m \log p(y^{(i)}|x^{(i)};\theta) - \alpha R(\theta)$, and $J'(\theta) = \sum_{i=1}^m \log p(y^{(i)}|Mx^{(i)};\theta) - \alpha R(\theta)$. Let $\hat{\theta} = \arg\max_\theta J(\theta)$ be the parameters resulting from fitting $L_2$ regularized logistic regression to $S$. (Because $\alpha > 0$, the Hessian of $J$ can be shown to be negative definite, and thus $J$ has a unique maximum.) Similarly, let $\hat{\theta}' = \arg\max_\theta J'(\theta)$ be the parameters resulting from fitting to $S'$. By our previous argument, clearly $J(\theta) = J'(M\theta)$ for all $\theta$. Thus, $\hat{\theta} = \arg\max_\theta J(\theta) = M^{-1}\arg\max_\theta J'(\theta) = M^{-1}\hat{\theta}'$, which implies $\hat{\theta}' = M\hat{\theta}$. Hence, $L[S](x) = 1/(1+\exp(-\hat{\theta}^T x)) = 1/(1+\exp(-(M\hat{\theta})^T(Mx))) = 1/(1+\exp(-(\hat{\theta}')^T x')) = L[S'](x')$. $\square$

We also give, without proof, additional examples of rotationally invariant algorithms:

- SVMs using most kernels.[3]
- Multilayer neural networks trained using back-propagation.[4]
- Unregularized logistic regression.[5]
- The perceptron algorithm.
- Any algorithm that uses PCA or ICA as a pre-processing step, by first re-representing the data in the basis formed by the top $k$ principle components/independent components.[6]
- Gaussian discriminant analysis (a generative learning algorithm which models $p(x|y)$ with a multivariate normal distribution).[7]

Examples of non-rotationally invariant algorithms include logistic regression with $L_1$ regularization, naive Bayes, decision trees that make only axis-aligned splits, Winnow (Littlestone, 1988), EG (Kivinen & Warmuth, 1994), and most feature selection algorithms (Blum & Langley, 1997; Kohavi & John, 1997; Ng & Jordan, 2001; Ng, 1998).

We now give a lower-bound on the worst-case sample complexity of feature selection using any rotationally invariant algorithm.

**Theorem 4.3:** *Let $L$ be any rotationally invariant learning algorithm, and let any $0 < \epsilon < 1/8$, $0 < \delta < 1/100$ be fixed. Then there exists a learning problem $\mathcal{D}$ so that: (i) The labels are deterministically related to the inputs according to $y = 1$ if $x_1 \geq t$, $y = 0$ otherwise for some $t$, and (ii) In order for $L$ to attain $\epsilon$ or lower 0/1 misclassification error with probability at least $1-\delta$, it is necessary that the training set size be at least*

$$m = \Omega(n/\epsilon).$$

Thus, for any rotationally invariant algorithm $L$, there exists at least one problem that should have been "easy" in the sense that there is only one relevant feature ($x_1$) and the labels are simply obtained by thresholding $x_1$, but $L$ requires a large number of training examples to learn it. Note that a good feature selection algorithm should be able to learn any target concept of this form using only $O(\log n)$ training examples. (E.g., Ng, 1998, Littlestone, 1988.) However, $L$ requires a number of training examples that's at least *linear* in the dimension of the input.

This suggests that rotationally invariant algorithms are unlikely to be effective feature selection algorithms, particularly in settings where only a small subset of the features are relevant, and the dimension of the input $n$ is significantly larger than the training set size $m$.

The proof of this result is given in Appendix B, and uses ideas from the lower-bounds originally proved by (Ehrenfeucht et al., 1989; Vapnik, 1982). A related result was also shown by (Kivinen et al., 1995) for the perceptron learning algorithm. They point out that the perceptron is rotationally invariant, and that an adversary choosing the sequence of training examples can force it (or, more generally, any "additive linear online prediction algorithm") to make $\Omega(n)$ mistakes.

**Remark.** Support Vectors Machines have been proved to work well in extremely high dimensional input spaces, even infinite-dimensional ones, as long as the data is separated with a large margin $\gamma$. (Vapnik, 1998) Thus, it may seem surprising that we can show that SVMs perform poorly in the presence of high dimensional inputs (with many irrelevant features). To reconcile this, we note that while the margin does not

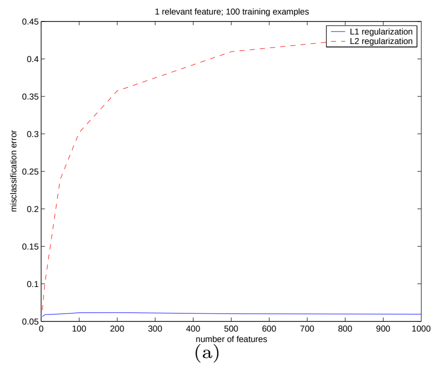
[3]Including the linear $K(x,z) = x^T z$, polynomial $K(x,z) = (x^T z + c)^d$, or RBF (Gaussian) $K(x,z) = \exp(-||x-z||^2/\sigma^2)$ kernels, or any other kernel $K(x,z)$ that can be written as a function of only $x^T x$, $x^T z$ and $z^T z$. Note also that the "$L_1$ norm soft margin" formulation of SVMs uses a different, per-training example, $L_1$ penalty on the slack variables, and rotational invariance still holds.

[4]Under the technical assumption that the weights are initialized, say, using independent samples from a Normal(0,$\epsilon$) distribution (or any other spherically symmetric distribution).

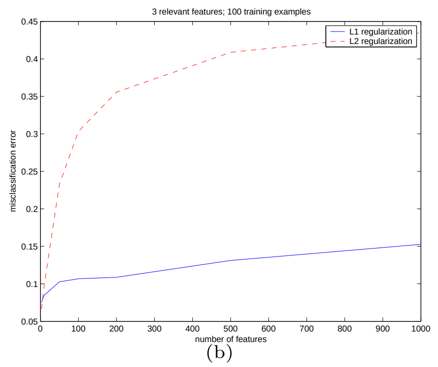
[5]Here, we restrict attention to training sets where the maximization (2) has a unique optimum with $\alpha = 0$. (If not, one can also use the limiting solution from $\alpha \to 0^+$ with $R(\theta) = ||\theta||_2^2$, if the limit exists).

[6]Assuming we do not preprocess the data for PCA by rescaling each input feature to have the same variance.

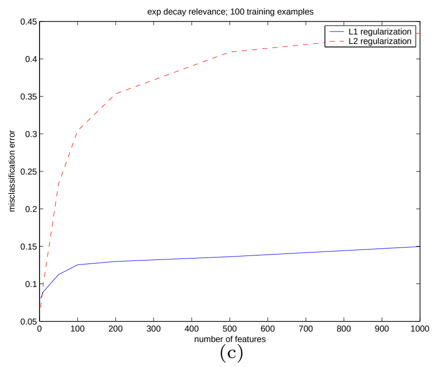
[7]Assuming the model uses a full covariance matrix, so that the off-diagonal entries are allowed to be non-zero.



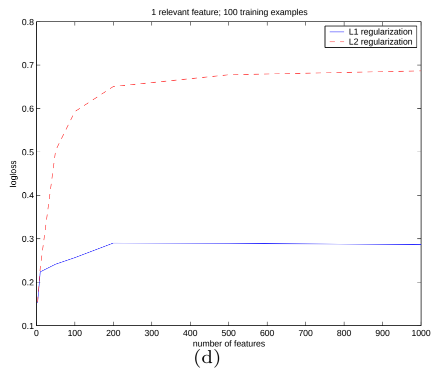
(a)



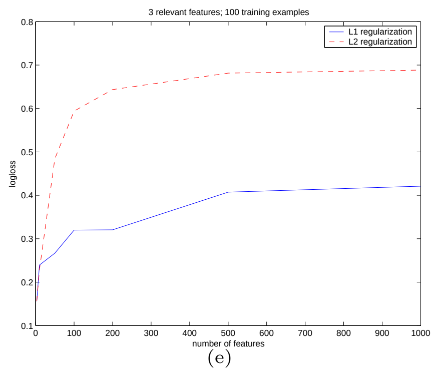
(b)



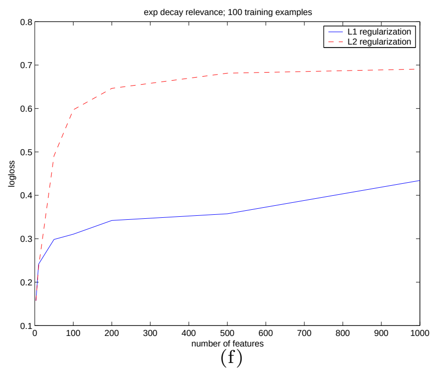
(c)



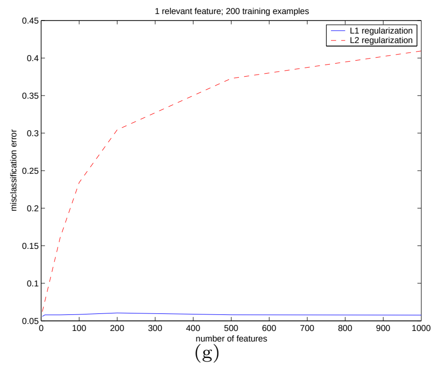
(d)



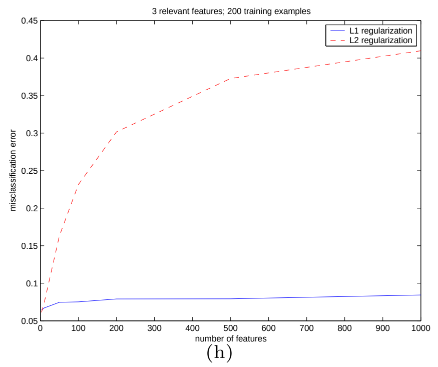
(e)



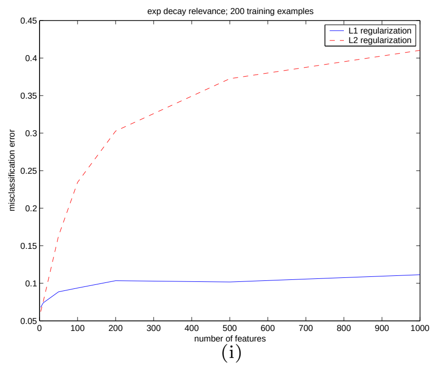
(f)



(g)



(h)



(i)

*Figure 1.* Experiment comparing logistic regression with $L_1$ regularization (blue solid lines; colors where available) vs. logistic regression with $L_2$ regularization (red dashed lines). Left column: One relevant feature. Middle column: Three relevant features. Right column: Exponentially-decaying relevance. Top row: Misclassification error with m=100. Middle row: Logloss error with m=100. Bottom row: Misclassification error with m=200.

shrink as extra irrelevant features are added, the *diameter* of the data (e.g., maximum distance between any two points measured in the $L_2$-norm) grows with the number of irrelevant features, and it is actually the margin *divided by the diameter* that governs generalization performance. (Vapnik, 1998)

## 5. Experiments

We now present some empirical results comparing logistic regression using $L_1$ and $L_2$ regularization. All results reported here are averages over at least 100 independent trials, and in each experiment, 30% of the data was used as hold-out data for selecting the regularization parameter. (Very similar results are obtained if the regularization parameters are tuned on test data.)

In the first experiment, we let the total number of features vary and let just a single feature be relevant.[8] Figure 1a shows the misclassification error of the two

[8]Experimental details: Inputs were drawn from a multivariate normal distribution. For one relevant feature, the labels were generated using a logistic model with $\theta_1 = 10$ (and all other $\theta_i = 0$). For three relevant features, we used $\theta_1 = \theta_2 = \theta_3 = 10c_1$. For the third problem, we used $\theta_i = (1/2)^{i-1}c_2$ $(i \geq 1)$, (The constants were $c_1 = 1/\sqrt{3}$, $c_2 = \sqrt{75}$, which preserve the scaling of the problem so that Bayes error remains the same.) Results reporting logloss and 0/1 misclassification error used respectively $\hat{\varepsilon}_{S_2} = \hat{\varepsilon}^l_{S_2}$ and $\hat{\varepsilon}_{S_2} = \hat{\varepsilon}^m_{S_2}$ in the hold-out cross validation step.

methods, when trained using 100 training examples. As we see, the results are dramatically different. Using $L_1$ regularization, logistic regression is extremely insensitive to the presence of irrelevant features. Note the scale on the horizontal axis: Even learning with just 100 examples in a 1000-dimensional input space, it is able to attain very low generalization error. In contrast, the error of logistic regression with $L_2$ regularization rapidly approaches 0.5.

Figure 1b shows the same experiment repeated with three relevant features. Figure 1c shows results from a third experiment where all the features contain some information about the output, but where the degree to which feature $i$ is relevant decreases exponentially with $i$. Only the first few features have a significant effect on the output label, and to model the data well, it is sufficient to use only a very small number of features. Again, $L_1$ regularization is clearly superior as $n$ becomes large.

Figures 1d-f repeat the same experiments, but here the logloss is plotted instead of misclassification error. Figures 1g-i show the same experiments repeated using 200 training examples. In all cases, logistic regression with $L_1$ regularization, as predicted by the theoretical results, exhibits a significantly higher tolerance to the presence of many irrelevant features.

## Acknowledgments

I give warm thanks to Pieter Abbeel, Chris Manning, Rajat Raina, Yoram Singer and Kristina Toutanova for helpful conversations. This work was supported by the Department of the Interior/DARPA under contract number NBCHD030010.

## References

Anthony, M., & Bartlett, P. (1999). *Neural network learning: Theoretical foundations.* Cambridge University Press.

Bartlett, P. (1998). The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, *2*, 525–536.

Blum, A., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*, 245–271.

Ehrenfeucht, A., Haussler, D., Kearns, M., & Valiant, L. (1989). A general lower bound on the number of examples needed for learning. *Information and Computation*, *82*, 247–261.

Haussler, D. (1992). Decision-theoretic generalizations of the PAC model for neural networks and other applications. *Information and Computation*, *100*, 78–150.

Kivinen, J., Warmuth, M., & Auer, P. (1995). The perceptron vs. winnow: Linear vs. logarithmic mistake bounds when few input variables are relevant. *Proc. 8th Annual Conference on Computational Learning Theory* (pp. 289–296).

Kivinen, J., & Warmuth, M. K. (1994). *Exponentiated gradient versus gradient descent for linear predictors* (Technical Report UCSC-CRL-94-16). Univ. of California Santa Cruz, Computer Research Laboratory.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*, 273–324.

Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, *2*, 285–318.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models (second edition).* Chapman and Hall.

Ng, A. Y. (1998). On feature selection: Learning with exponentially many irrelevant features as training examples. *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 404–412). Morgan Kaufmann.

Ng, A. Y., & Jordan, M. I. (2001). Convergence rates of the voting gibbs classifier, with application to bayesian feature selection. *Proceedings of the Eighteenth International Conference on Machine Learning.* Morgan Kaufmann.

Nigam, K., Lafferty, J., & McCallum, A. (1999). Using maximum entropy for text classification. *IJCAI-99 Workshop on ML for Information Filtering.*

Pollard, D. (1984). *Empirical processes: Theory and applications.* Springer-Verlag.

Rockafellar, R. (1970). *Convex analysis.* Princeton Univ. Press.

Strang, G. (1988). *Linear algebra and its applications, 3rd ed.* International Thomas Publishing.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, *58*, 267–288.

Vapnik, V. (1982). *Estimation of dependences based on empirical data.* Springer-Verlag.

Vapnik, V. N. (1998). *Statistical learning theory.* John Wiley & Sons.

Zhang, T. (2002). Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 527–550.

Zheng, A. X., Jordan, M. I., Liblit, B., & Aiken, A. (2004). Statistical debugging of sampled programs. *Neural Information Processing Systems 16.*

## Appendix A: Proof of Theorem 3.1

Our proof of Theorem 3.1 is based on bounding the covering numbers of certain function classes. Due to space constraints, our proof is necessarily brief, but for highly readable introductions to covering numbers, see, e.g., (Anthony & Bartlett, 1999; Haussler, 1992).

Let there be a class of functions $\mathcal{F}$ with some domain $U$ and range $[-M, M] \subset \mathbb{R}$. Given some set of points $z^{(1)}, \ldots, z^{(m)} \in U$, we let $\mathcal{F}_{|z^{(1)},\ldots,z^{(m)}} = \{[f(z^{(1)}), \ldots, f(z^{(m)})]; f \in \mathcal{F}\} \subseteq [-M, M]^m$. We say that a set $\{v^{(1)}, \ldots, v^{(k)}\} \subseteq \mathbb{R}^m$ $\epsilon$-covers $\mathcal{F}_{|z^{(1)},\ldots,z^{(m)}}$ in the $p$-norm if, for every $u \in \mathcal{F}_{|z^{(1)},\ldots,z^{(m)}}$, there is some $v^{(i)}$ so that $||u - v^{(i)}||_p \leq m^{1/p}\epsilon$. Here, $||t||_p = (\sum_{i=1}^m |t_i|^p)^{1/p}$. Define $\mathcal{N}_p(\mathcal{F}, \epsilon, [z_1, \ldots, z_m])$ to be the size of the smallest set that $\epsilon$-covers $\mathcal{F}_{|z_1,\ldots,z_m}$ in the $p$-norm. Also, let $\mathcal{N}_p(\mathcal{F}, \epsilon, m) = \sup_{z_1,\ldots,z_m} \mathcal{N}_p(\mathcal{F}, \epsilon, [z_1, \ldots, z_m])$.

Let there be some distribution $D$ over $U$, and define $\varepsilon(f) = \mathrm{E}_{z\sim D}[f(z)]$. If $z^{(1)}, \ldots, z^{(m)} \sim_{\text{iid}} D$, then (Pollard, 1984) showed that

$$P\left[\exists f \in \mathcal{F} : \left|\frac{1}{m}\sum_{i=1}^m f(z^{(i)}) - \mathrm{E}_{z\sim D}[f(z)]\right| > \epsilon\right]$$
$$\leq 8E[\mathcal{N}_1(\mathcal{F}, \epsilon/8, [z^{(1)}, \ldots, z^{(m)}])] \exp\left(\frac{-m\epsilon^2}{512M^2}\right) \quad (10)$$

Further, (Zhang, 2002) shows that if $\mathcal{G} = \{g : g(x) = \theta^T x, x \in \mathbb{R}^n, ||\theta||_q \leq a\}$ is a class of linear functions parameterized by weights $\theta$ with $q$-norm bounded by $a$, and if the inputs $x \in \mathbb{R}^n$ are also norm-bounded so that $||x||_p \leq b$, and further $1/p + 1/q = 1$ (so the $p$- and $q$-norms and dual) with $2 \leq p \leq \infty$, then

$$\log_2 \mathcal{N}_2(\mathcal{G}, \epsilon, m) \leq \left\lceil \frac{a^2 b^2}{\epsilon^2} \right\rceil \log_2(2n + 1). \quad (11)$$

(A special case of this is also found in Bartlett, 1998.)

Some other well-known properties of covering numbers (e.g., Anthony and Bartlett, 1999; Zhang, 2002; Haussler, 1992) include that

$$\mathcal{N}_1 \leq \mathcal{N}_2, \quad (12)$$

and that given a class of functions $\mathcal{G}$ with domain $\mathbb{R}$, if $\mathcal{F}$ is a class of functions $\mathbb{R} \times Y \mapsto \mathbb{R}$ defined according to $\mathcal{F} = \{f_g(x, y) = \ell(g(x), y) : g \in \mathcal{G}, y \in \{0, 1\}\}$, where $\ell(\cdot, y)$ (for any fixed $y$ and viewed a function of the first parameter only) is Lipschitz continuous with Lipschitz constant $L$, then

$$\mathcal{N}_1(\mathcal{F}, \epsilon, m) \leq \mathcal{N}_1(\mathcal{G}, \epsilon/L, m). \quad (13)$$

We now give the main part of the proof. First, notice that the algorithm uses hold-out cross validation to select amongst the values $B = 0, 1, 2, 4, \ldots$. Let $\hat{B}$ be the smallest value in $\{0, 1, 2, 4, \ldots\}$ that is greater than or equal to $rK$. Notice therefore that $rK \leq \hat{B} \leq 2rK$. We will begin by considering the step in the algorithm where logistic regression was fit using the regularization parameter $\hat{B}$. Specifically, let $\hat{\theta}$ denote the parameter vector resulting from solving the optimization problem given by Equations (3-4) with $B = \hat{B}$.

Let $\mathcal{G} = \{g_\theta : [-1, 1]^n \mapsto \mathbb{R} : g_\theta(x) = \theta^T x, ||\theta||_1 \leq \hat{B}\}$ be a class of linear functions parameterized by $\theta$ with $L_1$-norm bounded by $\hat{B}$. Using Equations (12,11), we have that

$$\begin{aligned} \log_2 \mathcal{N}_1(\mathcal{G}, \epsilon, m) &\leq \log_2 \mathcal{N}_2(\mathcal{G}, \epsilon, m) \\ &\leq \left\lceil \frac{\hat{B}^2}{\epsilon^2} \right\rceil \log_2(2n + 1). \end{aligned} \quad (14)$$

(Recall our assumption in Section 2 that $x \in [-1, 1]^n$, which implies $||x||_\infty \leq 1$.) From Holder's inequality, we also have

$$|g_\theta(x)| = |\theta^T x| \leq ||\theta||_1 \cdot ||x||_\infty \leq \hat{B}. \quad (15)$$

Now, let $\mathcal{F}$ be a class of functions $f : \mathbb{R} \times Y \mapsto \mathbb{R}$ defined according to $\mathcal{F} = \{f_\theta(x, y) = \ell(g_\theta(x), y) : g_\theta \in \mathcal{G}, y \in \{0, 1\}\}$, where $\ell(g(x), 1) = -\log 1/(1 + \exp(-g(x)))$, and $\ell(g(x), 0) = -\log(1 - 1/(1 + \exp(-g(x))))$. Thus, $\ell(g(x), y)$ is the logloss suffered by the logistic regression model on an example where it predicts $p(y = 1|x) = 1/(1 + \exp(-g(x))$, and the correct label was $y$. It is straightforward to show that $|\frac{d}{dt}\ell(t, y)| \leq 1$ for any $y \in \{0, 1\}$. Thus, $\ell(\cdot, y)$ is Lipschitz continuous with Lipschitz constant $L = 1$. Hence, combining Equations (13,14), we get

$$\log_2 \mathcal{N}_1(\mathcal{F}, \epsilon, m) \leq \left\lceil \frac{\hat{B}^2}{\epsilon^2} \right\rceil \log_2(2n + 1). \quad (16)$$

It is also straightforward to show that $|\ell(t, 1)| = |\log 1/(1+\exp(-t))| \leq |t|+1$ (and similarly for $\ell(t, 0)$). Together with Equation (15), this implies that

$$|f_\theta(x, y)| = |\ell(g_\theta(x), y)| = |\ell(\theta^T x, y)| \leq \hat{B} + 1. \quad (17)$$

Let $m_1 = (1 - \gamma)m$ be the number of examples the parameters $\hat{\theta}$ were trained on in the inner-loop of the algorithm. (The remaining $m_2 = \gamma m$ examples were used for hold-out cross validation.) Recalling that $\mathcal{N}_1(\mathcal{F}, \epsilon, [z^{(1)}, \ldots, z^{(m)}])] \leq \mathcal{N}_1(\mathcal{F}, \epsilon, m)$ by definition, and putting together Equations (16,10,17) with $M = \hat{B} + 1$, we find that

$$P\left[\exists f \in \mathcal{F} : \left|\frac{1}{m_1}\sum_{i=1}^{m_1} f(x^{(i)}, y^{(i)}) - \mathrm{E}_{(x,y)\sim D}[f(x, y)]\right| > \epsilon\right]$$
$$\leq 8 \cdot 2^{64\hat{B}^2/\epsilon^2+1}(2n + 1) \cdot \exp\left(\frac{-m_1\epsilon^2}{512(\hat{B} + 1)^2}\right). \quad (18)$$

We would like for this probability to be small. By setting the right hand side to $\delta$ and solving for $m_1$, we find that in order for the probability above to be upper-bounded by $\delta$, it suffices that $m_1 = \Omega((\log n) \cdot \mathrm{poly}(\hat{B}, \frac{1}{\epsilon}, \log \frac{1}{\delta})) = \Omega((\log n) \cdot \mathrm{poly}(r, K, \frac{1}{\epsilon}, \log \frac{1}{\delta}))$, where to obtain the second equality we used the fact (shown earlier) that $rK \le \hat{B} \le 2rK$. Since $m_1 = (1-\gamma)m$, if we treat $(1-\gamma)$ as a constant that can be absorbed into the big-$\Omega$ notation, then to ensure the above holds, it suffices that

$$m = \Omega((\log n) \cdot \mathrm{poly}(r, K, \tfrac{1}{\epsilon}, \log \tfrac{1}{\delta})). \tag{19}$$

To summarize, we have shown that if $m$ satisfies Equation (19), then with probability $1-\delta$, it will hold true that for all $f \in \mathcal{F}$, we have that

$$\left| \frac{1}{m_1} \sum_{i=1}^{m_1} f(x^{(i)}, y^{(i)}) - \mathrm{E}_{(x,y)\sim D}[f(x,y)] \right| \le \epsilon. \tag{20}$$

By referring to the definitions of $\mathcal{F}$ and $\mathcal{G}$, we see this would imply that for all $\theta : ||\theta||_1 \le \hat{B}$, we have

$$\left| \frac{1}{m_1} \sum_{i=1}^{m_1} -\log p(y^{(i)}|x^{(i)};\theta) - \mathrm{E}_{(x,y)\sim D}[-\log p(y|x;\theta)] \right| \le \epsilon \tag{21}$$

and therefore that for all $\theta : ||\theta||_1 \le \hat{B}$,

$$\left| \hat{\varepsilon}^l_{S_1}(\theta) - \varepsilon^l(\theta) \right| \le \epsilon. \tag{22}$$

In summary, we have shown that with a training set whose size has to be at most logarithmic in $n$ and polynomial in all quantities of interest, with probability $1-\delta$, we will have that $\hat{\varepsilon}^l_{S_1}(\theta)$ is a uniformly good estimate for $\varepsilon^l(\theta)$. Now, recall that the parameter vector $\hat{\theta}$ was found by solving $\hat{\theta} = \arg\min_{\theta:||\theta||_1 \le \hat{B}} \hat{\varepsilon}^l_{S_1}(\theta)$. Using Equation (22), a standard uniform convergence result[9] (e.g., Vapnik, 1982; Anthony and Bartlett, 1999) shows that minimizing $\hat{\varepsilon}^l$ is nearly as good as minimizing $\varepsilon^l$, and that in particular, Equation (22) implies

$$\begin{aligned} \varepsilon^l(\hat{\theta}) &\le \min_{\theta:||\theta||_1 \le \hat{B}} \varepsilon^l(\theta) + 2\epsilon \\ &\le \varepsilon^l(\theta^*) + 2\epsilon, \end{aligned} \tag{23}$$

where the second step used the fact that $||\theta^*||_1 \le \hat{B}$.

Hence, we have shown that in Step 2 of the algorithm, we will find at least one parameter vector $\hat{\theta}$ whose performance (generalization error as measured according to $\varepsilon^l$) is nearly as good as that of $\theta^*$. In Step 3 of the algorithm, we use hold-out cross validation to select from the set of $\theta_B$'s found in Step 2. Using another entirely standard argument, it is straightforward to show that, with only a "small" (at most polynomially large, and independent of $n$) number of examples used in the hold-out set, we can ensure that with probability $1-\delta$, the selected parameter vector will have performance at most $\epsilon$ worse that that of the best parameter vector in the set. The details of this step are omitted due to space, but is entirely standard and may be found in, e.g., (Vapnik, 1982; Anthony & Bartlett, 1999). Putting this together with (23), we have shown that, with probability $1-2\delta$, the output $\theta$ satisfies

$$\varepsilon^l(\theta) \quad \le \quad \varepsilon^l(\theta^*) + 3\epsilon. \tag{24}$$

Finally, replacing $\delta$ with $\delta/2$ and $\epsilon$ with $\epsilon/3$ everywhere in the proof shows the theorem. □

[9]Specifically, that if $|f(\theta) - \hat{f}(\theta)| \le \epsilon$ for all $\theta \in \Theta$, then $f(\arg\min_{\theta\in\Theta} \hat{f}(\theta)) \le \min_{\theta\in\Theta} f(\theta) + 2\epsilon$.

## Appendix B: Proof of Theorem 4.3

Let $L$ and $\epsilon, \delta$ be as given in the statement of the theorem. Consider the concept class of all linear separators in $n$ dimensions, $\mathcal{C} = \{h_\theta : h_\theta(x) = 1\{\theta^T x \ge \beta\}, \theta \ne 0\}$. (Here, we do not use the convention adopted previously that necessarily $x_n = 1$. Also, $1\{\cdot\}$ is the indicator function, so that $1\{\text{True}\} = 1$, and $1\{\text{False}\} = 0$.) It is well-known that $\mathrm{VC}(\mathcal{C}) = n+1$. (Vapnik, 1982)

From a standard PAC lower bound, there must therefore exist a distribution $D_X$ over the inputs, and a target concept $h^* \in \mathcal{C}$, so that if $D_X$ is the input distribution, and the labels are given by $y = h^*(x)$, then for $L$ to attain $\epsilon$ or lower 0/1 misclassification error with probability at least $1-\delta$, it is necessary that the training set size be at least $m = \Omega(n/\epsilon)$. (Results of this type have been proved by Vapnik, 1982 and Ehrenfeucht et al., 1989. The particular result stated here is also given in, e.g., Theorem 5.3 of Anthony and Bartlett, 1999).

Since $h^* \in \mathcal{C}$ is a linear target concept, it can be written $h^*(x) = 1\{\theta^{*T} x \ge \beta^*\}$ for some $\theta^* \in \mathbb{R}^n$ and $\beta^* \in \mathbb{R}$. Because replacing $(\theta^*, \beta^*)$ with $(c\theta^*, c\beta^*)$ for any positive constant $c$ does not change anything, we may assume without loss of generality that $||\theta^*||_2 = 1$.

Let $M$ be any orthogonal matrix whose first row is $\theta^{*T}$. Such a matrix must exist. (Strang, 1988) Thus, $M\theta^* = [1, 0, \ldots, 0]^T = e_1$ (the first basis vector). Further, by flipping the signs of any single row (other than the first row) of $M$ if necessary, we may ensure that $|M| = 1$, and hence $M \in \mathcal{M}$. Now, consider a learning problem where the input distribution is induced by sampling $x \sim D_X$, and then computing $x' = Mx$. Further, let the labels be given by $y' = 1\{x'_1 \ge \beta^*\}$. Because $1\{x'_1 \ge \beta^*\} = 1\{e_1^T x' \ge \beta^*\} = 1\{(M\theta^*)^T(Mx) \ge \beta^*\} = 1\{(\theta^*)^T x \ge \beta^*\} = y$, we therefore see that a learning problem with examples drawn from the $(x', y')$ distribution is simply a rotated version of the problem with examples $(x, y)$. But since $L$ is rotationally invariant, its predictions on test sets under the original and rotated problems will be identical, and thus its generalization error, and sample complexity, must also be the same under either problem. □