Making Logistic Regression A Core Data Mining Tool With TR-IRLS

Paul Komarek
Carnegie Mellon University
School of Computer Science
Pittsburgh, PA 15213
komarek@cmu.edu

Andrew W. Moore
Carnegie Mellon University
School of Computer Science
Pittsburgh, PA 15213
awm@cs.cmu.edu

Abstract

Binary classification is a core data mining task. For large datasets or real-time applications, desirable classifiers are accurate, fast, and need no parameter tuning. We present a simple implementation of logistic regression that meets these requirements. A combination of regularization, truncated Newton methods, and iteratively re-weighted least squares make it faster and more accurate than modern SVM implementations, and relatively insensitive to parameters. It is robust to linear dependencies and some scaling problems, making most data preprocessing unnecessary.

1 Motivation and Terminology

This article is motivated by the success of a fast, simple logistic regression (LR) algorithm in several high-dimensional data mining engagements, including life sciences data mining [10, 7], threat classification and temporal link analysis [16], collaborative filtering [11], and text processing [7]. The rise of support vector machines (SVMs) for binary classification has renewed interest in LR, due to similar loss functions [20, 21]. Many recent papers propose new methods of estimating the LR model parameters, see [8] and references therein. Many of the new LR implementations include instructions for tuning parameters or data preprocessing. These steps can be distracting on small datasets, and impractical or impossible on large datasets. Running many cross-validations to tune a parameter wastes researchers' and practitioners' time.

Our LR fitting procedure is a simple modification of iteratively re-weighted least squares (IRLS) that mimics truncated Newton methods and adds regularization. We claim it is simple and does not need parameter tuning or data preprocessing. We compare our algorithm, TR-IRLS, to other LR implementations and linear and radial basis function (RBF) SVMs. See [8] for further comparison and analysis. Our software, source, and most of our datasets are available at `http://www.autonlab.org` and `http://komarix.org`.

Our primary contribution is the demonstration that TR-IRLS can make LR simple, effective, and parameter-free. By parameter-free, we mean that tuning is generally unnecessary. Though not discussed in this short article, TR-IRLS applies to kernelized LR, as well as any generalized linear model. We provide a complete description of our algorithm.

In this paper we are concerned with binary classification, and hence a data point belongs to the positive or negative class. A sufficiently fast binary classifier can be reasonably applied to multiclass problems through a linear-time (in the number of classes) transformation [11]. A dataset is a matrix of inputs $\mathbf{X}$, and a binary vector of outputs $\mathbf{y}$. When $y_i = 1$, the i^{th} row of $\mathbf{X}$ belongs to the positive class. There are M features and R records. For a sparse binary matrix, F is the sparsity, and MRF is the number of nonzero elements.

Conjugate gradient (CG) is an iterative minimization algorithm. CG only requires computation of matrix-vector products. When applied to a quadratic form, CG simplifies to *linear CG*. Otherwise, it is called *nonlinear CG* and requires heuristic direction updates, line searches, and restarts. Because the Hessian of a quadratic form is a matrix, linear CG can be used to solve systems of linear equations. Further explanation can be found in [19, 17].

2 Logistic regression

LR models the relation of each row $\mathbf{x_i}$ of $\mathbf{X}$ to the expected value $E(y_i)$, with the logistic function $\mu(\mathbf{x}, \beta) = \exp(\beta^T \mathbf{x})/(1 + \exp(\beta^T \mathbf{x}))$, where β is the vector of parameters. We assume $x_0 = 1$ so that β_0 is a constant term. Our regression model is $y = \mu(\mathbf{x}, \beta) + \varepsilon$, where ε is a binomial error term. Let $\mu_i = \mu(\mathbf{x_i}, \beta)$. The log-likelihood is

$$\ln \mathbb{L}(\mathbf{X}, \mathbf{y}, \beta) = \sum_{i=1}^{R} y_i \ln(\mu_i, \beta)) + (1 - y_i) \ln(1 - \mu_i, \beta)) \quad (1)$$

The loss function is the *deviance* (DEV), which for binary outputs is $-2 \ln \mathbb{L}$ [13, 4]. LR is a linear classifier and

```
input   : $\mathbf{X}$, $\mathbf{y}$, initial LR parameter estimate $\hat{\beta}_0$
output  : final LR parameter estimate $\hat{\beta}$
Set $i = 0$
repeat
    Compute $\mu_j = \mu(\hat{\beta}_i, \mathbf{x_j})$ for $j = 1 \ldots R$
    Weights: $w_j = \mu_j(1 - \mu_j)$, $\mathbf{W} = \mathrm{diag}(w_1, \ldots, w_R)$
    Adjusted dependent covariates:
        $z_j = \hat{\beta}_i^T \mathbf{x_j} + (y_j - \mu_j)/w_j$
    Compute $\hat{\beta}_{i+1}$ via WLS:
        $(\mathbf{X}^T\mathbf{W}\mathbf{X})\hat{\beta}_{i+1} = \mathbf{X}^T\mathbf{W}\mathbf{z}$
    $i = i + 1$
until $\hat{\beta}_{i+1}$ converges
```

Alg. 1: Finding the LR MLE with IRLS.

```
input   : $\mathbf{A} = (\mathbf{X}^T\mathbf{W}\mathbf{X})$, $\mathbf{b} = (\mathbf{X}^T\mathbf{W}\mathbf{z})$, $\mathbf{v_0}$
output  : $\mathbf{v}$ such that $\mathbf{Av} = \mathbf{b}$
Set $i = 0$, initialize residual $\mathbf{r_0}$ to $\mathbf{b} - \mathbf{Av_0}$
repeat
    Update the search direction mixing parameter $\tau_i$:
        On the zeroth iteration, let $\tau_i = 0$
        Otherwise, let $\tau_i = \mathbf{r_i}^T\mathbf{r_i}/(\mathbf{r_{i-1}}^T\mathbf{r_{i-1}})$
    Update search direction: $\mathbf{d_i} = \mathbf{r_i} + \tau_i\mathbf{d_{i-1}}$
    Compute optimal step:   $\alpha_i = -\mathbf{d_i}^T\mathbf{r_0}/(\mathbf{d_i}^T\mathbf{Ad_i})$
    Update approx solution: $\mathbf{v_{i+1}} = \mathbf{v_i} - \alpha_i\mathbf{d_i}$
    Update residual:        $\mathbf{r_{i+1}} = \mathbf{b} - \mathbf{Av_i}$
    $i = i + 1$
until $|\mathrm{DEV}_{i-1} - \mathrm{DEV}_i|/|\mathrm{DEV}_i|$ converges
```

Alg. 2: Linear CG solving $(\mathbf{X}^T\mathbf{W}\mathbf{X})\mathbf{v} = \mathbf{X}^T\mathbf{W}\mathbf{z}$

might classify low-dimensional data with nonlinear boundaries poorly. However, in high-dimensional spaces, linear boundaries are often adequate. Kernelizing LR may overcome this low-dimensional limitation [21].

IRLS Iteratively re-weighted least squares (IRLS) is a nonlinear optimization algorithm that uses a series of weighted least squares (WLS) subproblems to search for the MLE. [13, 2]. IRLS is a special case of Fisher's scoring method, a quasi-Newton algorithm that replaces the objective function's Hessian with the Fisher information. For LR, IRLS is a special form of Newton's method [13, 3, 2]. We summarize IRLS in Algorithm 1. There is no step length to compute, in contrast with most Newton variants. The difficult part is solving the WLS subproblem $(\mathbf{X}^T\mathbf{W}\mathbf{X})\hat{\beta}_{i+1} = \mathbf{X}^T\mathbf{W}\mathbf{z}$, a linear system with M equations and variables. The $(i, j)^{\text{th}}$ entry of $\mathbf{X}^T\mathbf{W}\mathbf{X}$ is the weighted dot-product of the i^{th} and j^{th} input columns. The weights change every iteration, forcing recomputation. Newton's method converges quadratically, but each iteration is expensive. For a detailed description of our early work on IRLS, see [10].

TR-IRLS Because the WLS subproblems are linear systems, they can be solved using linear CG as shown in Algorithm 2. This is simpler than likelihood optimization with nonlinear CG, since linear CG has an optimal direction update formula and no line searches or restarts. Linear CG is $\mathrm{O}(MRF)$, if one ignores the condition number of $\mathbf{X}^T\mathbf{W}\mathbf{X}$ [19]. We can stop CG iterations early to approximate the WLS solution, thus creating a *truncated Newton method* with accompanying convergence guarantees [17]. CG only requires matrix-vector products, eliminating computation of $\mathbf{X}^T\mathbf{W}\mathbf{X}$ and simplifying sparse computations.

Correlated attributes can cause scaling problems, which we address using ridge regression to regularize the WLS subproblems [18, 3]. This only requires perturbing $\mathbf{X}^T\mathbf{W}\mathbf{X}$ by $\lambda\mathbf{I}$. We call the combination of IRLS, linear CG, and ridge regression, *truncated regularized IRLS* or TR-IRLS.

We stop IRLS when the relative difference of the deviance $|\mathrm{DEV}_{i-1} - \mathrm{DEV}_i|/|\mathrm{DEV}_i|$ is less than ε_1. The same test can be used for CG iterations, with bound ε_2. Sometimes CG iterations stray and increase the deviance, and we limit the number of consecutive non-improving iterations with the *CG window* parameter. Our final specialization is to initialize $\mathbf{v}$ in Algorithm 2 to $\hat{\beta}_i$ when solving for $\hat{\beta}_{i+1}$.

During a large empirical evaluation of λ, ε_1, ε_2, and the CG window [7], we observed that accuracy was relatively insensitive to these parameter values. Therefore we use default values for all experiments in this paper, with $\lambda = 10$, $\varepsilon_1 = 1/100$, $\varepsilon_2 = 1/200$, and a CG window of 3 iterations.

CG-MLE Besides IRLS, other nonlinear methods can be used to optimize the LR likelihood. Nonlinear CG was among the better methods in Minka's survey [15], and was among the earliest methods used for computerized LR [14]. We call this combination CG-MLE. See [8] for discussion of quasi-newton alternatives to nonlinear CG. Nonlinear CG is $\mathrm{O}(MRF)$ if one ignores the condition number of the input matrix [19]. The actual run-time depends on many factors including the direction update formula, line search, and restarting criterion. Nonlinear CG has many direction update formulas. We compared the top four in [7], and chose the modified Polak-Ribiére formula. It combines a Polak-Ribiére update with Powell restarts [17], and may be written $\tau_i = \max(0, \mathbf{r_i}^T(\mathbf{r_i} - \mathbf{r_{i-1}})/\mathbf{r_{i-1}}^T\mathbf{r_{i-1}})$ where τ_i corresponds to the same symbol in Algorithm 2.

Regularization is important for CG-MLE [20, 15]. Like ridge regression, the likelihood is penalized by $\lambda(\hat{\beta}^T\hat{\beta})$. Any reasonable λ works well [7], and we use 10. It has been reported that initializing β_0 to the mean of $\mathbf{y}$ improves stability [14], but we only observed a speed improvement.

Table 1. Dataset summary, see Section 3.

Name	Columns	Rows	Nonzero	Pos
citeseer	105,354	181,395	512,267	299
imdb	685,569	167,773	2,442,721	824
ds2	1,143,054	88,358	29,861,146	423
ds1	6,348	26,733	3,732,607	804
ds1.100	100	26,733	NA	804
ds1.10	10	26,733	NA	804
modapte.sub	26,299	7,769	423,025	495

The TR-IRLS CG window technique also helps CG-MLE. We terminate CG-MLE iterations using the same test as TR-IRLS, with an epsilon of 1/200 [7].

3 Experiments

All experiments are ten-fold cross-validations scored with the Area Under Curve (AUC) metric [3]. AUC measures the ability to rank-order classifications, with 1.0 for perfection and 0.5 for random guessing. We describe AUC further in [7]. Our conclusions hold with other metrics, such as precision, recall, and F1. All times are "real" seconds but do not include I/O or time spent tuning the SVMs. We used an AMD Opteron 242, and less than 4GB of RAM.

In this short paper, we compare LR to per-dataset tuned linear and radial basis function (RBF) SVMs from SVM^{light} version 5 [6]. We briefly compare TR-IRLS to SAS' proc logistic. In previous work, Naive Bayes and C4.5 decision trees routinely scored worse than LR [7].

Table 1 summarizes the seven datasets of this paper. Details for the link analysis datasets citeseer and imdb and the life sciences datasets ds2, ds1, ds1.100, and ds1.10 are in [7]. The last two are PCA projections of ds1. We also use a text classification dataset modapte.sub, a subset of the multiclass Reuters-21578 ModApte training corpus. We kept classes with 100 or more positive rows: acq, coffee, corn, crude, dlr, earn, gnp, grain, interest, money-fx, money-supply, oilseed, ship, sugar, trade, and wheat, denoted a_1 through a_{16}. The column "Nonzero" shows the number of nonzero inputs in sparse datasets. "Pos" shows the number of positive rows, except for modapte.sub where it is averaged over the output attributes. All datasets except ds1 and ds2 are publicly available [9].

4 Results and analysis

Table 2 shows AUC and time results on the life sciences and link datasets. The LR methods have nearly identical AUC scores. To eliminate any AUC versus speed bias held by the authors, we ran two sets of SVM experiments. The first optimizes the AUC through extensive tuning, and the second set optimizes speed such that the AUC is within 10% of the best SVM AUC . Still, TR-IRLS was faster in every case. The considerable time spent tuning linear and RBF SVMs is not included in any of our timings.

Table 3. Times in seconds for LR and SVMs on the first ten attributes of modapte.sub.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
TR-IRLS	14	16	16	15	15	14	16	14	15	14
SVM LIN	26	18	18	24	17	28	17	24	22	25
SVM RBF	131	68	70	99	58	127	57	97	84	102

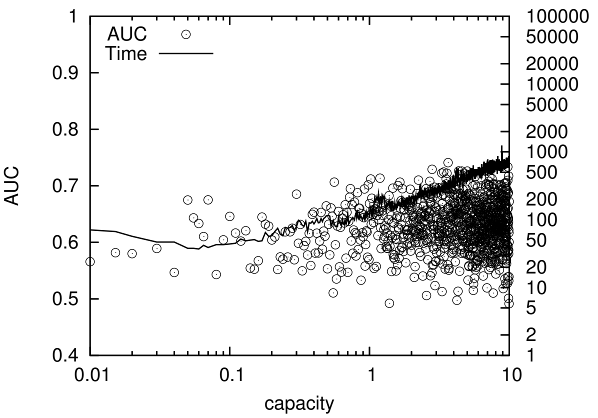

Figure 1. Erratic SVM^{light} behavior on ds1.10.

Tuning the SVM^{light} capacity and RBF gamma parameters produced significant improvements in accuracy. The final values used are available in [8]. TR-IRLS scored as well or better than other classifiers in all experiments except ds1.100 and ds1.10. These two datasets have relatively few dimensions, that likely require nonlinear boundaries for good classifications, and RBF SVMs easily beat all the linear classifiers here. However, TR-IRLS requires less time to compute a better classifier on the original ds1, than RBF SVM required for the PCA-compressed versions ds1.100 and ds1.10. A fast version of KNN for skewed-class problems [12] was more competitive on these small datasets [8]. The nearest competitor to TR-IRLS in accuracy on the larger datasets is CG-MLE. However, TR-IRLS is consistently faster and easier to implement and understand in depth. All classifiers did well on modapte.sub, with LR and SVM always above 0.977. Table 3 reports times on the first ten attributes. Again, TR-IRLS is fastest.

SVM^{light} was generally more sensitive to its parameters than TR-IRLS, but its linear kernel was surprisingly erratic

Table 2. Life sciences and link datasets. Times are in seconds, and do not include SVM tuning.

	`citeseer`		`imdb`		`ds2`		`ds1`		`ds1.100`		`ds1.10`	
Classifier	Time	AUC	Time	AUC	Time	AUC	Time	AUC	Time	AUC	Time	AUC
TR-IRLS	53	0.945	272	0.983	1460	0.722	45	0.948	35	0.913	8	0.842
CG-MLE	70	0.946	310	0.983	2851	0.724	120	0.946	294	0.916	43	0.844
SVM LIN BEST	82	0.821	647	0.949	3729	0.704	846	0.931	1744	0.882	373	0.741
SVM LIN FAST	79	0.810	564	0.938	2030	0.690	183	0.918	123	0.874	73	0.675
SVM RBF BEST	1150	0.864	4549	0.957	67118	0.700	3594	0.939	2577	0.934	167	0.876
SVM RBF FAST	408	0.798	1929	0.947	14681	0.680	1593	0.902	932	0.864	248	0.848

on `ds1.10`. The extreme changes in AUC for small changes in capacity are shown in Figure 1. The left axis represents AUC , the horizontal axis is capacity, and circles are AUC data points. The right axis and solid line show the time for each experiment. LIBSVM [1] showed similar behavior [8].

We compared TR-IRLS to SAS' `proc logistic` [5] (special thanks to Lujie Chen, Auton Lab). Because `proc logistic` lacks an option for sparse data, we made large dense subsets of `ds1`. TR-IRLS consistently ran three times faster with dense computations. SAS' memory use grew rapidly with the number of columns, and limited our tests.

5 Related work

The main debate in current LR literature is how to calculate LR parameters, see [8] and references therein. Proposed methods include variants of Newton's method, CG, iterative scaling and Gauss-Seidel/cyclic coordinate descent. These methods face the numeric and overfitting problems common to nonlinear optimization in machine learning. See surveys in [15, 20], and analysis in [8].

6 Conclusions

We presented the TR-IRLS fitting procedure and demonstrated its use with logistic regression. This combination appears faster and at least as accurate as SVMs and other logistic regression fitting procedures. TR-IRLS can be used with any generalized linear model. TR-IRLS is very simple and can be implemented from details in this paper. Our software, sources, and data are available at `http://www.autonlab.org` and `http://komarix.org`.

References

[1] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/˜cjlin/libsvm`.

[2] J. E. Gentle. *Elements of Computational Statistics*. Statistics and Computing. Springer Verlag, 2002.

[3] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Verlag, 2001.

[4] D. W. Hosmer and S. Lemeshow. *Applied Logistic Regression*. Wiley, 2nd edition, 2000.

[5] http://www.sas.com/. SAS. `http://www.sas.com/`.

[6] T. Joachims. SVM^{light}, 2002. `svmlight.joachims.org`.

[7] P. Komarek. Logistic Regression for Data Mining and High-Dimensional Classification. Technical Report TR-O4-34, Robotics Inst., Carnegie Mellon Univ., Pgh, PA, May 2004.

[8] P. Komarek. Making Logistic Regression A Core Data Mining Tool: A Practical Investigation of Accuracy, Speed, and Simplicity. Technical Report TR-O5-27, Robotics Inst., Carnegie Mellon Univ., Pgh, PA, May 2004.

[9] P. Komarek. Datasets, 2005. `http://komarix.org/ac/ds`.

[10] P. Komarek and A. Moore. Fast Robust Logistic Regression for Large Sparse Datasets with Binary Outputs. In *Artificial Intelligence and Statistics*, 2003.

[11] J. Kubica, A. Goldenberg, P. Komarek, A. Moore, and J. Schneider. A Comparison of Statistical and Machine Learning Algorithms on the Task of Link Completion. In *KDD Workshop on Link Analysis for Detecting Complex Behavior*, page 8, August 2003.

[12] T. Liu, A. Moore, and A. Gray. Efficient Exact k-NN and Nonparametric Classification in High Dimensions. In *Proc. of Neural Information Processing Systems*, 2003.

[13] P. McCullagh and J. A. Nelder. *Generalized Linear Models*, volume 37 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, 2 edition, 1989.

[14] A. McIntosh. *Fitting Linear Models: An Application of Conjugate Gradient Algorithms*, volume 10 of *Lecture Notes in Statistics*. Springer-Verlag, New York, 1982.

[15] T. P. Minka. Algorithms for maximum-likelihood logistic regression. Technical Report Stats 758, Carnegie Mellon University, October 2001.

[16] A. Moore, P. Komarek, and J. Ostlund. Activity Prediction From Links, 2004. `http://www.autonlab.org`.

[17] S. G. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, 1996.

[18] M. Orr. Introduction to Radial Basis Function Networks, 1996. `http://www.anc.ed.ac.uk/˜mjo/rbf.html`.

[19] J. R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical Report CS-94-125, Carnegie Mellon University, Pittsburgh, 1994.

[20] T. Zhang and F. J. Oles. *Text Categorization Based on Regularized Linear Classification Methods*. Kluwer, 2001.

[21] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, March 2005.