Fast Logistic Regression for Data Mining, Text Classification and Link Detection

Paul Komarek
Department of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213
komarek@cmu.edu

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

Abstract

Previous work by the authors [1] demonstrated that logistic regression can be a fast and accurate data mining tool for life sciences datasets, competitive with modern tools like support vector machines and ball-tree based K-NN. This paper has two objectives. The first objective is a serious empirical comparison of logistic regression to several classical and modern learners on a variety of learning tasks. The second is to describe our use of conjugate gradient inside an iteratively re-weighted least squares fitting procedure.

1 Introduction

This paper begins by describing and discussing logistic regression and conjugate gradient, contrasting the traditional combination with our implementation. We then describe the datasets and scoring metrics of our experiments, followed by our results in three application areas using seven datasets.

2 Introduction to logistic regression and conjugate gradient

2.1 Logistic regression

Logistic regression (LR) gets its name from the logistic curve, or sigmoid, shown in Equation 1 and Figure 1. Because this curve approaches zero and one with a controllably quick transition, it is well-suited to binary classification. Suppose our data $\mathbf{X}$ has R rows and M attributes, and R binary observations $\mathbf{y}_i$. Let $\mathbf{x_i}$ denote row i, β denote the $M+1$ model parameters, and let μ_i be the model's expected value for $\mathbf{y}_i$. The sigmoid is written as

$$\mu_\mathbf{i} = \frac{\exp(\beta_0 + \beta_1 \mathbf{x_{i1}} + \cdots \beta_M \mathbf{x_{i}}_M)}{1 + \exp(\beta_0 + \beta_1 \mathbf{x_{i1}} + \cdots \beta_M \mathbf{x_{i}}_M)} \tag{1}$$

The log likelihood for this model can be written as

$$\sum_{i=1}^{R} \mathbf{y}_i \ln(\mathbf{u}_i) + (1 - \mathbf{y}_i) \ln(1 - \mathbf{u}_i)$$

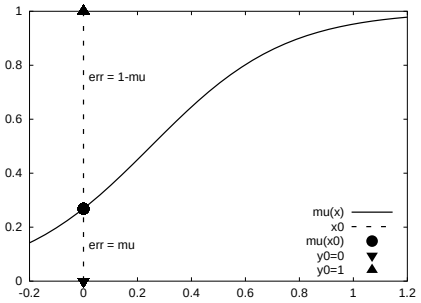

Figure 1: LR residuals vary with x

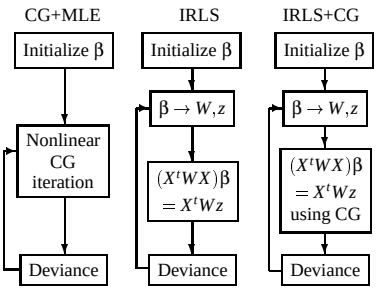

Figure 2: Schematic of LR methods.

From this, it is simple to compute the system of maximum likelihood equations

$$\sum_{i=1}^{R} (\mathbf{y}_i - \mu_i) = 0, \quad \sum_{i=1}^{R} \mathbf{x_i}_j (\mathbf{y}_i - \mu_i) = 0, \; j = 1 \ldots M$$

It is common to solve the MLE system either directly via a generic gradient descent algorithm, or indirectly using the iterative re-weighted least squares (IRLS) method [2, 3, 4, 5]. These methods are described below, and illustrated in Figure 2.

Direct: In the direct MLE approach, one initializes the LR parameter vector β and invokes a nonlinear solver on the MLE system of equations. When $\mu(\beta)$ provides an adequate model of $\mathbf{y}$, perhaps according to a likelihood metric, the nonlinear solver is terminated. It is common to use conjugate gradient (CG) as the nonlinear solver in this approach [2], and we denote this combination CG+MLE. This nonlinear application of CG requires careful attention to numerics and a well-tuned line search procedure. This method and other "direct" methods are discussed in [2].

Indirect: The indirect IRLS method starts with an estimate of β, and hence μ, and computes error weights $\mathbf{w}_i = \mu_i(1-\mu_i)$. The LR problem is linearized around μ to produce adjusted dependent covariates $\mathbf{z}$. These covariates are used with the $R \times M$ input matrix $\mathbf{X}$ and diagonal weight matrix $\mathbf{W} = \mathrm{diag}(\mathbf{w})$ in the linear regression

$$\left(\mathbf{X^T W X}\right) \beta = \mathbf{X^T W z} \tag{2}$$

The solution β is scored using the *deviance*:

$$\mathrm{DEV} = 2 \sum_{i=1}^{R} \left[\mathbf{y}_i \ln \left(\frac{\mathbf{y}_i}{\mu_i} \right) + (1 - \mathbf{y}_i) \ln \left(\frac{1 - \mathbf{y}_i}{1 - \mu_i} \right) \right] \tag{3}$$

This metric is a likelihood ratio between the current model and a saturated model with R parameters. Under the null hypothesis that $\beta_1, \ldots, \beta_M = 0$, DEV has a χ^2 distribution with M degrees of freedom [3]. This IRLS procedure is repeated until the relative difference of the deviance between iterations stabilizes. For LR and other generalized linear models using canonical link functions, IRLS is equivalent to Newton Raphson [4]. We will use the IRLS nomenclature to emphasize applicability of our method to all generalized linear models.

The residual errors in the LR model are binomially distributed with the changing variance, as seen in Figure 1. The probability that $\mathbf{x}_i$ is in the positive class is $\mu(\mathbf{x}_i)$. The observed class $\mathbf{y}_i$ will be positive or negative, yielding a residual error of $(1-\mu)$ or μ respectively. Heteroscedasticity and the $(0,1)$ range of the model distinguish LR from linear regression.

2.2 Conjugate gradient

A thorough explanation of conjugate gradient (CG) and related methods can be found in [6] or [7]. An overview of CG in the context of logistic regression can be found in [5] or [1]. CG is an iterative nonlinear minimization algorithm which approximates the objective function with a quadratic form. If the objective function is itself the quadratic form

$$\frac{1}{2}\mathbf{p}^{\mathbf{T}}\left(\mathbf{X}^{\mathbf{T}}\mathbf{X}\right)\mathbf{p} - \mathbf{b}^{\mathbf{T}}\mathbf{p}, \tag{4}$$

then CG will find the unique minimizer $\mathbf{p}$ in M iterations when using exact arithmetic. Minimization of expression (4) is equivalent to solving $\mathbf{X}^{\mathbf{T}}\mathbf{X}\mathbf{p} = \mathbf{b}$ for $\mathbf{p}$, and hence CG can be used as a linear solver. CG has three important features for our application. It is not necessary to form the Hessian of the objective function, the input matrix $\mathbf{X}$ can be rank-deficient, and CG provides good estimates of $\mathbf{p}$ in far fewer than M iterations. Alternatives to CG, such as biconjugate gradient, may have better theoretical properties when the Hessian $\mathbf{X}^{\mathbf{T}}\mathbf{X}$ is rank-deficient. We have not yet found a compelling reason to explore such algorithms for our applications.

3 Implementing fast, robust logistic regression

We prefer the IRLS approach for LR because it avoids general-purpose nonlinear optimization algorithms while maintaining applicability to generalized linear models, as described in [4]. However, the time required to solve Equation 2 exactly is unacceptable for large datasets. Furthermore, finding exact solutions for some datasets can lead to numerical difficulties and over-fitting [1]. For these reasons an approximate linear solver for Equation 2 may be preferred. We are exploring the use of CG, described in Section 2.2, in this role. A schematic of the direct CG+MLE approach, IRLS, and IRLS+CG can be found in Figure 2. To the best of our knowledge, the IRLS+CG combination has not been previously reported.

Successfully implementing and using IRLS+CG has not been straightforward. We have overcome many numerical problems and reduced the number of parameters which need manual tuning. We describe these issues below.

3.1 Terminating LR iterations

The purpose of LR is to find a maximum likelihood estimate for the parameters β, which corresponds to minimizing the deviance (3). We terminate IRLS iterations when the relative difference $\left|\text{DEV}_{(i-1)} - \text{DEV}_i\right| / \text{DEV}_i$ of iterations $(i-1)$ and i is sufficiently small. Unfortunately the deviance can behave erratically and non-monotonically, causing the relative difference to remain large at every iteration even though the deviance itself is small.

We use three separate criteria to decide when to terminate IRLS iterations:

1. Problems occur while solving Equation 2.
2. Relative difference of deviance is small enough.
3. We have performed too many IRLS iterations.

While performing IRLS iterations we keep a history of the LR parameters and deviance for every iteration, choosing the parameters associated with the smallest deviance after termination. When scoring performance using a testing dataset having a different distribution than the training set, it is useful to terminate IRLS after one iteration. Although this reduces logistic regression to something like linear regression, it works well enough that we have an `lrmax` parameter to control condition 3.

3.2 Terminating CG iterations

The most important component of our IRLS+CG implementation is the linear CG solver. When tuned correctly, it quickly converges to a solution that keeps IRLS on-course through the linearize-and-re-estimate procedure. Unlike a nonlinear CG solver, which requires a line search, the only parameter of a linear CG solver is when to terminate.

Each linear CG iteration returns not only an estimate of the minimizer for Expression 4, but also a residual error vector $\mathbf{r}$. A simple CG termination strategy is to stop when $\mathbf{r}$ is less than some tolerance `cgeps`. Unfortunately, the optimal value for `cgeps` changes not only with the dataset used, but with each cross-validation fold, and even after each IRLS iteration within a fold. Robustness to over-conservative `cgeps` settings can be achieved by keeping a history of solution estimates and the size of the associated residual vector, coupled with some or all of the following termination strategies:

1. $||\mathbf{r}_i|| < \texttt{cgeps}$.
2. $||\mathbf{r}_i|| \leq ||\mathbf{r}_{i+1}|| \leq \ldots \leq ||\mathbf{r}_{i+k}||$ for some value k.
3. $||\mathbf{r}_i|| > K||\mathbf{r}_{\text{best}}||$ for some value of K.
4. We have performed too many CG iterations.

Robustness is further enhanced by penalizing large coefficients. We are adding a ridge regression penalty of the form $\lambda \sum_{j=1}^{M} \beta_j^2$ to the residual sum-of-squares of the linear regression (2) [8]. We refer to the ridge regression parameter λ as `rrlambda`.

We have also tried terminating CG using the relative difference of the deviance. Computing deviance for each CG iteration is expensive, but we often saw fewer total CG iterations. Our experiments with this termination criterion show a small increase in scores and somewhat decreased sensitivity to the relative difference threshold, but computation time lengthens. A reduction in the time required for deviance computation, perhaps through a fast deviance estimator, may make this method more attractive in the future.

3.3 Perturbing binary observations

When using LR with binary observations it is important to remember that LR's sigmoid model, shown in Equation 1, cannot take values zero or one. If the observed outputs $\mathbf{y}_i$ are binary, IRLS may continue adjusting the components of β well beyond reasonable limits, causing overfitting, poor numerical conditioning, and wasting time. A simple solution is to shrink the output space by a small amount, e.g. from $\{0,1\}^M$ to $\{0.001, 0.999\}^M$.

4 Datasets

In this paper we illustrate the strengths and weaknesses of LR through life sciences data mining, text categorization, and link detection. When discussing datasets in this paper, each record belongs to, or is predicted to belong to, the positive or negative class. R is the number of rows in the dataset, and M is the number of attributes. The *sparsity factor* F is the proportion of nonzero entries. Thus the total number of nonzero elements in a dataset is the product MRF. Our datasets are summarized in Table 1.

4.1 Life sciences

The life sciences data mining was performed on datasets similar to the publicly available Open Compound Database and associated biological test data, as provided by the National Cancer Institute [9]. For cross-validation experiments, datasets `ds1` and `ds2` were used. Both datasets are sparse.

Table 1: Datasets.

Name	Attributes	Rows	Sparsity	Num Nonzero	Num Pos Rows
ds1	6,348	26,733	0.02199	3,732,607	804
ds2	1,143,054	88,358	0.00029	29,861,146	423
modapte	26,299	7,769	0.00211	423,025	1-2877, avg 106
modapte-test	26,299	3,019	0.00199	158,595	1-1087, avg 42
citeseer	105,354	181,395	0.00002	512,267	299
imdb	685,569	167,773	0.00002	2,442,721	824

4.2 Text classification

To discover whether our IRLS+CG implementation was fast enough for text classification, we ran tests on the Modified Apte split of the Reuters-21578 corpus. We chose the same variation of ModApte as used in [10]. After punctuation removal, stemming, and stop word removal, our version had more features than reported by Yang (26,299 vs. 24,240). Yang and Li generously offered to provide the dataset already stemmed and cleaned, but we received it too late to use in this paper's experiments. We denote our training dataset `modapte`, and our testing set `modapte-test`. Note that `modapte` and `modapte-test` have ninety binary output categories, so a classification experiment consists of ninety training and testing sessions.

4.3 Link detection

This paper uses two publicly available datasets for link detection experiments. The first, `citeseer`, is derived from the CiteSeer web site [11] and lists the names of collaborators on published materials. We extracted the most common name, `J_Lee`, to use as the output. The goal is to predict whether `J_Lee` was a collaborator for each work based on others listed for that work. The second dataset is derived from the Internet Movie Database [12] and is denoted `imdb`. Each row represents a work of entertainment, and the attributes represent people associated with that work. Again we chose the most frequently appearing attribute, `Blanc__Mel`, for the output.

5 Scoring

Two metrics are used for scoring experiments in this paper. The first metric is Area Under Curve (AUC). Suppose μ_j is our prediction for the j^{th} dataset row, and $\mathbf{y}_j$ is the observed binary classification 0 or 1 for that row. Arrange the dataset rows in descending order of μ_j such that $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_R$. Let $C(i) = \sum_{j=1}^{i} \mathbf{y}_j$. Then

$$\text{AUC} = \frac{\sum_{i=0}^{R} C(i)(1-y_i)}{C(R)(1-C(R))}$$

A more intuitive description of AUC in terms of ROC curves can be found in [1]. A higher AUC score is better, with a maximum of 1.0. We include a 95% confidence interval based on Student's T distribution when we report AUC on a cross-validation.

The second metric consists of three scores used in text classification. These are precision p, recall r, and effectiveness $F1$. To compute these scores, we threshold μ_j to produce binary predictions $\mathbf{b}_j$. Define

$$B = \sum \mathbf{b}_j \qquad Y = \sum \mathbf{y}_j \tag{5}$$

$$C = \sum \mathbf{b}_j \mathbf{y}_j \qquad I = \sum \text{xor}(\mathbf{b}_j, \mathbf{y}_j) \tag{6}$$

Table 2: Algorithm performance on ds1 and ds2, in order of AUC for ds1.

Learner	ds1 Time(s)	ds1 AUC	ds2 Time(s)	ds2 AUC
BC	12	0.885±0.012	332	0.601±0.030
K-NN	1024	0.887±0.024		
D-Tree	156	0.892±0.014	13594	0.644±0.039
LR (CG+MLE)	241	0.900±0.118	6762	0.723±0.032
Linear SVM	268	0.918±0.012		
RBF SVM	14821	0.955±0.011		
LR (IRLS+CG)	147	0.948±0.010	5736	0.717±0.036

B is the number of positive predictions, Y is the number of positive observations, C is the number of correct positive predictions, and I is the number of incorrect predictions. Let

$$p = \begin{cases} C/B & \text{if } B \neq 0 \\ 1.0 & \text{otherwise,} \end{cases} \qquad r = \begin{cases} C/Y & \text{if } Y \neq 0 \\ 1.0 & \text{otherwise,} \end{cases} \qquad F1 = 1.0 - \frac{I}{B+Y}$$

[13]. For our datasets, Y is never zero.

These scores are computed for each category in a text classification. The per-category scores are either *macro-averaged* or *micro-averaged*, defined below, to produce a composite score.

macro-average: average the precision, recall, and effectiveness across all outputs

micro-average: compute the composite precision, recall, and effectiveness using the extensions of Definitions 5 and 6 to encompass all output categories simultaneously.

6 Results

We compare several learning algorithms in the experiments below. The SVM experiments use SVM^{light} version 4.0 with its default parameters. However, in radial basis function (RBF) SVM experiments we set the RBF parameter *gamma* to 0.01 to allow them to finish within a reasonable time. For K-NN we used k=9, and for our C4.5 decision trees (D-Tree) our chi-threshold is 0.05. All LR IRLS+CG experiments use cgeps=0.0001 and rrlambda=5.0. The parameters for LR CG+MLE were tuned for each experiment to produce the best results.

6.1 Data mining

Our analysis of IRLS+CG as a data mining tool, contrasted with other popular algorithms like K-NN and SVM, was reported in [1]. Since that paper was written, many improvements have been made to our software. New and updated results may be be found in Table 2. For this paper, we have switched from Gaussian confidence intervals to Student's T confidence intervals. We believe the Student's T distribution is more appropriate for ten-fold cross-validation. Some of the algorithms are fast enough to run on the larger and noisier dataset ds2. These are reported in Table 1.

6.2 Text classification

A variety of text classification results with LR have been reported, both unfavorable, e.g. [14], and favorable, e.g. [15]. Zhang points out that the naive application of LR to text classification problems produces suboptimal scores, but careful use of regularization leads to

Table 3: Text classification results for training set `modapte` and testing set `modapte-test`. Times shown are cross-validation (to compute thresholds) + testset.

Learner	Time(s)	p	r	Micro F1	Macro F1
RBF SVM	8236+1229	0.956	0.680	0.795	0.205
Linear SVM	2382+ 286	0.930	0.733	0.820	0.278
LR (IRLS+CG)	605+ 206	0.911	0.760	0.829	0.339

Table 4: Link detection results for `citeseer` and `imdb`.

Learner	`citeseer` Time(s)	AUC	`imdb` Time(s)	AUC
LR (CG+MLE)	134	0.696±0.240	692	0.983±0.009
Linear SVM	222	0.822±0.046	1637	0.939±0.018
RBF SVM	2743	0.860±0.045	14821	0.955±0.011
LR (IRLS+CG)	116	0.948±0.021	796	0.982±0.009

classifications that compete with SVM and other popular text classifiers. Our first concern is whether IRLS+CG is sufficiently fast and accurate to be useful for text classification. As suggested by Joachims in [16], we use no feature selection and limit our preprocessing to punctuation removal, stemming and stop word removal.

The SVM results in Table 3 are a bit lower than those in [16], [10], and [15]. Differences in the processing of Reuters-21578 may explain this discrepancy. For example, Joachims has 9,962 terms before feature selection [16], while Yang and Liu report 24,240 terms before feature selection [10]. Unfortunately, none of those papers present detailed timings for their experiments. Joachims describes SVM times as comparable to a C4.5 decision tree [17], though our previous experiments show SVM^{light} performing somewhat slower than our C4.5 implementation. Zhang and Oles describe SVM and LR as slower than other methods in [15].

To determine thresholds for the components of μ in the IRLS+CG experiments, we performed a ten-fold cross-validation with `lrmax`=1. Thresholds were chosen which maximized a micro-averaged F1 score for the cross-validation predictions. These thresholds were applied to our predictions on `modapte-test`. Although SVM^{light} reports thresholds for its outputs, we were able to compute higher-scoring thresholds via a cross-validation.

Our conclusion from the experiments in Table 3 is that further exploration of IRLS+CG for fast text categorization will be worthwhile. As suggested in [16] and [15], fast and robust algorithms may reduce or remove the need for feature selection. Given the performance of SVM and IRLS+CG in this experiment, perhaps meta-features such as intra-word distance for rare words should be added to text datasets.

6.3 Link detection

The input and output configuration of these datasets is described in Section 4. The results of these binary classification experiments are summarized in Table 4.

7 Conclusions

We have demonstrated that IRLS+CG is sufficiently fast and accurate for large data mining, text categorization and link detection tasks. With appropriate regularization and over-fitting compensation, little or no tuning of LR is necessary.

8 Future work

- Investigate a divide-and-conquer approach to IRLS+CG.
- Apply our work to other generalized linear models.
- Explore meta-features such as rare word separation distance in text classification.

9 Acknowledgments

The authors would like to thank Susana Eyheramendy and David Cohn for their assistance with the text classification experiments. David Cohn also prepared the link detection datasets `citeseer` and `imdb`.

References

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

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

[3] David W. Hosmer and Stanley Lemeshow. *Applied Logistic Regression*. Wiley-Interscience, 2 edition, 2000.

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

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

[6] Anne Greenbaum. *Iterative Methods for Solving Linear Systems*, volume 17 of *Frontiers in Applied Mathematics*. SIAM, 1997.

[7] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, 1996.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Verlag, 2001.

[9] National Cancer Institute Open Compound Database, 2000. `http://cactus.nci.nih.gov/ncidb2`.

[10] Y. Yang and X. Liu. A re-examination of text categorization methods. In *22nd Annual International SIGIR*, pages 42–49, Berkley, August 1999.

[11] CiteSeer Scientific Digital Library, 2002. `http://www.citeseer.com`.

[12] Internet Movie Database, 2002. `http://www.imdb.com`.

[13] C. J. Van Rijsbergen. *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow, 1979.

[14] Hinrich Schutze, David A. Hull, and Jan O. Pedersen. A Comparison of Classifiers and Document Representations for the Routing Problem. In *Research and Development in Information Retrieval*, pages 229–237, 1995.

[15] Tong Zhang and Frank J. Oles. *Text Categorization Based on Regularized Linear Classification Methods*. Kluwer Academic Publishers, 2001.

[16] Thorsten Joachims. *Learning to Classify Text Using Support Vector Machines*. PhD thesis, Cornell University, May 2002.

[17] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[18] Susana Eyheramendy, David D. Lewis, and David Madigan. On the naive bayes model for text categorization. In *Artificial Intelligence and Statistics*, 2003.