Scalable L1-regularized logistic regression:

# IRLS-LARS

Andrea Klein & Dipan Kumar Pal

# The Problem

- We wish to do binary classification on large datasets via logistic regression with an L1 norm.

- Recall that logistic regression models the probability that an element x belongs to a class y as a sigmoid function:

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

# The Problem

- Under a Laplacian prior, the MAP estimate for the model parameters (theta) is given by:

$$\boxed{\min_{\theta} \sum_{i=1}^{M} - \log p(y^{(i)} | \mathbf{x}^{(i)}; \theta) + \beta \|\theta\|_1}$$

- Technically we solve an equivalent alternative parametrization, subject to:

$$\|\theta\|_1 \leq C$$

- Regardless of how you formulate the problem, it's ordinarily inefficient to find theta for large datasets.

# THE ALGORITHM

# IRLS-LARS

ITERATIVELY RE-WEIGHTED LEAST SQUARES

LEAST ANGLE REGRESSION

# THE ALGORITHM

## IRLS-LARS

ITERATIVELY RE-WEIGHTED LEAST SQUARES

LEAST ANGLE REGRESSION

- ON EACH STEP OF THE ALGORITHM, WE CHARACTERIZE THE UPDATE DIRECTION (GAMMA) OF THETA AS THE SOLUTION TO A LEAST-SQUARES PROBLEM:

$$\gamma^{(k)} = \arg\min_{\gamma} ||(\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{X}^{\top})\gamma - \mathbf{\Lambda}^{\frac{1}{2}}\mathbf{z}||_2^2$$

# THE ALGORITHM

IRLS-LARS

$$\mathbf{X} = [\mathbf{x}^{(1)}\mathbf{x}^{(2)} \ldots \mathbf{x}^{(M)}]$$

UPDATED ("RE-WEIGHTED") ON EACH ITERATION:

$$\begin{aligned} \Lambda_{ii} &= \sigma(\theta^{(k)\top}\mathbf{x}^{(i)})[1 - \sigma(\theta^{(k)\top}\mathbf{x}^{(i)})], \\ z_i &= \mathbf{x}^{(i)\top}\theta^{(k)} + \frac{[1 - \sigma(y^{(i)}\theta^{(k)\top}\mathbf{x}^{(i)})]y^{(i)}}{\mathbf{\Lambda}_{ii}}. \end{aligned}$$

ITERATIVELY LEAST SQUARES

ANGLE REGRESSION

- ON EACH STEP OF THE ALGORITHM, WE CHARACTERIZE THE UPDATE DIRECTION (GAMMA) OF THETA AS THE SOLUTION TO A LEAST-SQUARES PROBLEM:

$$\gamma^{(k)} = \arg\min_{\gamma} \|(\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{X}^{\top})\gamma - \mathbf{\Lambda}^{\frac{1}{2}}\mathbf{z}\|_2^2$$

# The Algorithm

**IRLS-LARS**

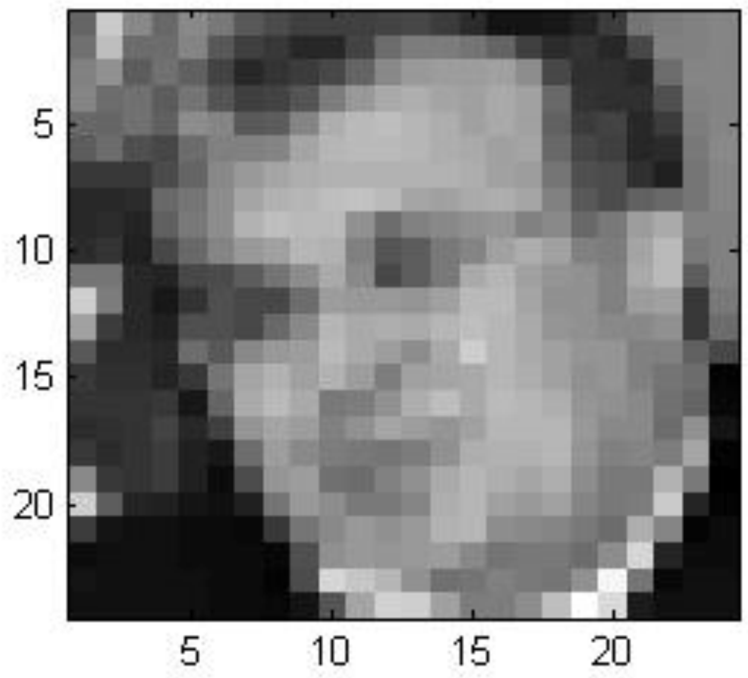
Iteratively Re-Weighted Least Squares

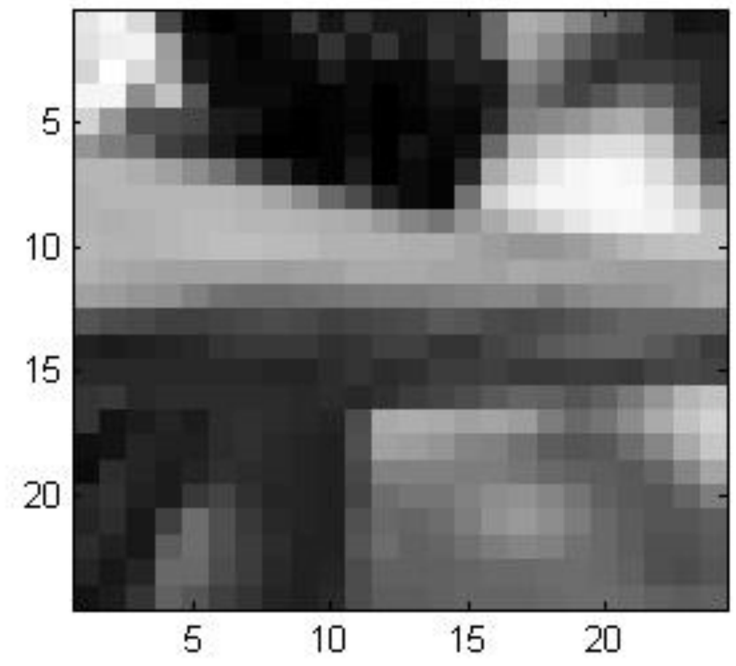
Least Angle Regression

- In practice, we use an algorithm called LARS to efficiently solve for the step direction within the IRLS formulation. Finally, once we know the direction of the update, we use a backtracking line search to determine how big a step to take.

# Now for an Experiment

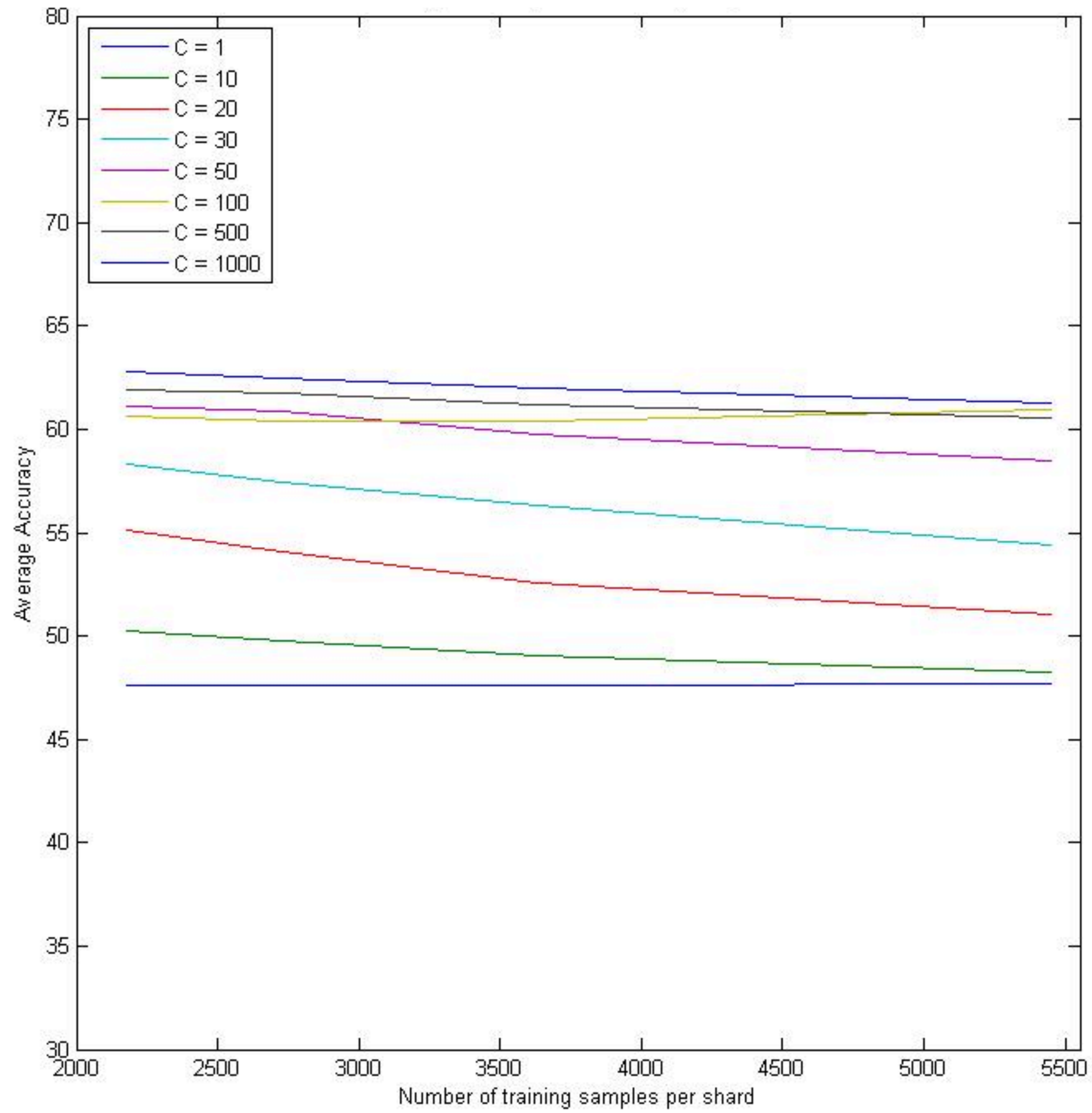
We'll demonstrate IRLS-LARS on a binary image classification task: identifying which images have faces (and which don't).



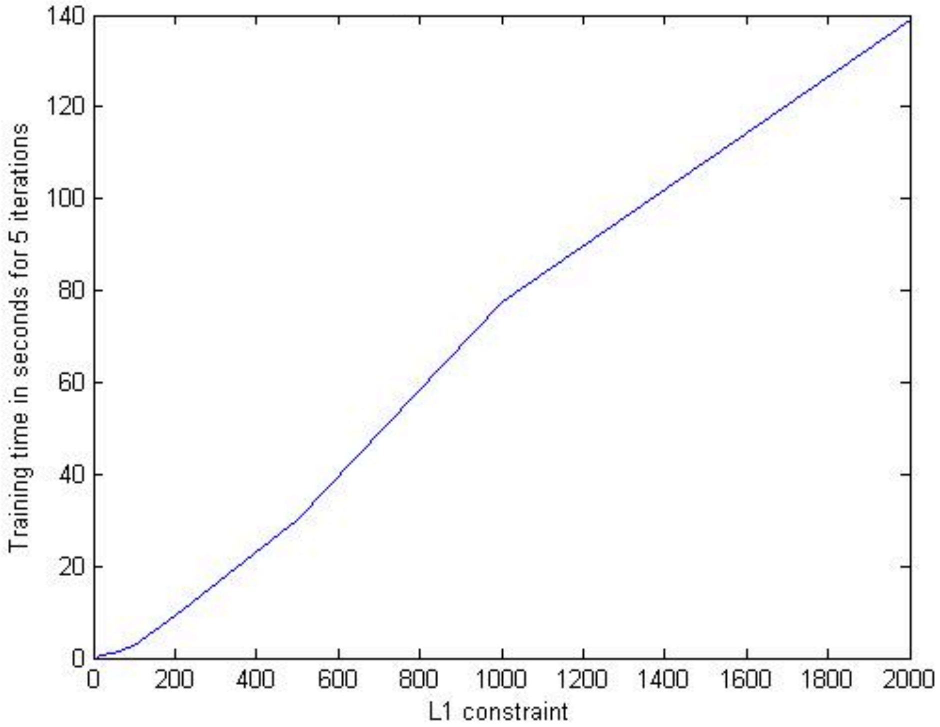
# PERFORMANCE

## Average accuracy vs. # of training samples

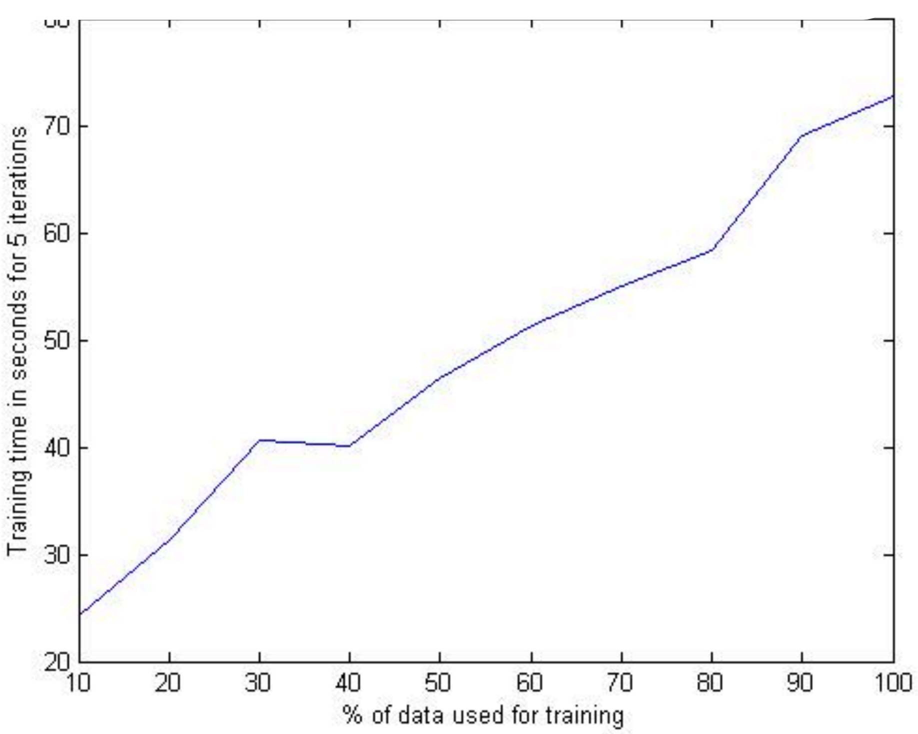


# Performance

## Training Time vs. Constraint Size




## Training Time vs. Training Size

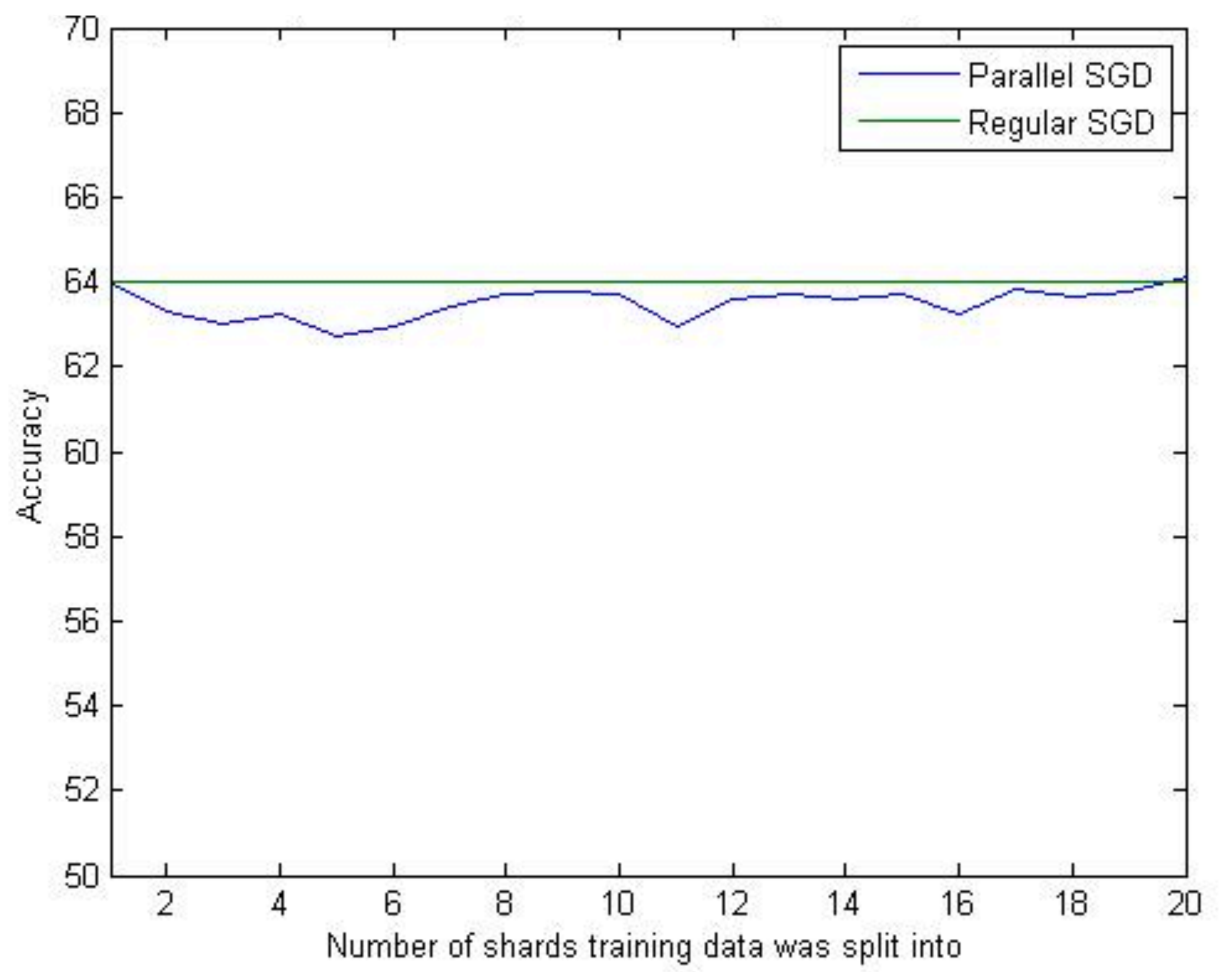


# SCALABILITY

**Algorithm 2** ParallelSGD($\{c^1, \dots c^m\}, T, \eta, w_0, k$)

**for all** $i \in \{1, \dots k\}$ **parallel do**
  $v_i = \mathrm{SGD}(\{c^1, \dots c^m\}, T, \eta, w_0)$ on client
**end for**
Aggregate from all computers $v = \frac{1}{k}\sum_{i=1}^{k} v_i$ and **return** $v$

## Relative Performance of Parallel vs. Regular SGD



# Still to come

- More strenuous (parallel) tests on larger datasets
- If time, we'd like to explore the algorithm's performance on more diverse datasets
- We may also incorporate parallel SCD (sequential coordinate descent), parallelizing over features rather than samples:

**Algorithm 2** Shotgun: Parallel SCD

Choose number of parallel updates $P \geq 1$.
Set $\mathbf{x} = \mathbf{0} \in \mathbb{R}_{+}^{2d}$
**while** not converged **do**
  Choose random subset of P weights in $\{1, \ldots, 2d\}$.
  **In parallel** on P processors
    Get assigned weight $j$.
    Set $\delta x_j \longleftarrow \max\{-x_j, -(\nabla F(\mathbf{x}))_j / \beta\}$.
    Update $x_j \longleftarrow x_j + \delta x_j$.
**end while**

# THE END!

## References:


- Efficient L1 Regularized Logistic Regression (Lee et al)
- Parallelized Stochastic Gradient Descent (Zinkevich et al)
- Parallel Coordinate Descent for L1 Regularized Loss Minimization (Bradley et al)
- Least Angle Regression (Efron et al)

# MATH APPENDIX

$$\mathbf{X} = [\mathbf{x}^{(1)}\mathbf{x}^{(2)} \ldots \mathbf{x}^{(M)}].$$

Let the diagonal matrix $\boldsymbol{\Lambda}$ and the vector $\mathbf{z}$ be defined as follows: for all $i = 1, 2, \ldots, M$:

$$\Lambda_{ii} = \sigma(\theta^{(k)\top}\mathbf{x}^{(i)})[1 - \sigma(\theta^{(k)\top}\mathbf{x}^{(i)})], \tag{7}$$

$$z_i = \mathbf{x}^{(i)\top}\theta^{(k)} + \frac{[1 - \sigma(y^{(i)}\theta^{(k)\top}\mathbf{x}^{(i)})]y^{(i)}}{\boldsymbol{\Lambda}_{ii}}. \tag{8}$$

Then, we have that $\mathbf{H}(\theta^{(k)}) = -\mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^\top$ and $\mathbf{g}(\theta^{(k)}) = \mathbf{X}\boldsymbol{\Lambda}(\mathbf{z} - \mathbf{X}^\top\theta^{(k)})$, and thus Equation (5) can be rewritten as:

$$\gamma^{(k)} = (\mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^\top)^{-1}\mathbf{X}\boldsymbol{\Lambda}\mathbf{z}. \tag{9}$$

Thus $\gamma^{(k)}$ is the solution to the following weighted least squares problem:

$$\gamma^{(k)} = \arg\min_{\gamma} \|(\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{X}^\top)\gamma - \boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{z}\|_2^2. \tag{10}$$