STAT 788 - Final Project

Daniel Hartig

May 9, 2018

How to run

The submitted code consists of six different files. The file `main.c` is one of the main executables which calls to the other files. This main has customizable command line arguments. Another main is contained in `longtest.c` which generates test data discussed later in this paper. Reading data from the file, standardizing a dataset, and generating mislabellings are done in `datagen.c` with its header file `datagen.h`. Logistic regression and associated mathematical functions are done in `logistic.c` and associated header file `logistic.h`

The customizable executable is compiled by

```
gcc -o final main.c datagen.c logistic.c -lm
```

and run by

```
./final pima 0.
```

The arguments to this script are

Variable	Description
Data Source	Source of the response data: 'pima': unedited Pima dataset response values 'b0': authors' generated response variables from Hung et al., Section 4.2
Mislabel Type	Strategy for generating mislabels from Hung et al., Section 4.1: 0: No mislabeling 1: Strategy S1 (y-dependent mislabeling) 2: Strategy S2 (x-dependent mislabeling) 3: Strategy S3 (random x, y-dependent mislabeling) 4: Strategy S4 (clustered x, y-dependent mislabeling)

The generation of test data compiled by

```
gcc -o testgen midterm.c datagen.c logistic.c -lm.
```

and run by

```
./testgen.
```

Methodology

Generating samples from the Pima set

The Pima data set is read from its input data file (`pima.dat`) and divided into two parts, the feature and response variable. The features have a column of ones added as the first column to give an intercept. This produces a different ordering of feature numbers compared to the tables in Hung. For example, in Table 1 of Hung, the nine features start with β_{01} = x1 as labeled in the data file; and end with β_{09} = the intercept. In the computational parts of this project, the intercept is pre-pended. For any length 9 vector of dimension of a matrix, the 0th item is the intercept, while the 1st item corresponds to x1 from the data file.

Once the data is read, a sample from the data set is taken. There are 768 data points in the data set, but all samples taken are hardcoded to size 500 following Hung. A random sample isolumn taken using a simplified Reservoir algorithm. Since the total dataset is small, it is used in its entirety as the reservoir.

Algorithm 1 Generate a size SAMPLESIZE sample from DATA

```
output ← empty length(SAMPLESIZE) array
selections ← empty length(DATA) array
numleft ← length(DATA)
for i ∈ (0, 1, ... SAMPLESIZE) do
  n ← RANDOMUNIFORM()
  output[i] ← DATA[selections[n]]
  numleft ← numleft − 1
  selections[n] ← selections[numleft]
end for
```

After the data set is generated, it is standardized. Standardization is done on a columnwise basis to generate a standardized matrix A^* from feature matrix A:

$$\text{Mean for column}j: \quad \mu_j = \frac{1}{n}\sum_{i=1}^{n} A_{i,j}$$

$$\text{Std Dev for column}j: \quad \sigma_j = \sqrt{\frac{\sum_{i=1}^{n}(A_{i,j}-\mu_j)}{n-1}}$$

$$\text{Standarized Matrix } A^*: \quad A^*_{i,j} = \frac{A_{i,j}-\mu_j}{\sigma_j}$$

Generating the Author's response variables

There are two options for response variables. The first is to use the original Pima dataset as is, which represents the incidence of diabetes in females members of the Pima indian tribe, based on eight features describing the individuals. This options is selected by passing `'pima'` as the first command line argument to the `'main.c'` version of the executable. Since the response variables are already read from the data file and selected along with their matching features into a sample of 500, nothing more is done in this case.

The second is to use a 'true' regression parameter set as done by Hung. This option is selected by passing `b0` as the first command line argument to the `main.c` executable. The 'true' value of the parameters are given as $B_0 = \langle 0, 1, -1, 1, 0, 0, 0, 0, 0\rangle$. The response generated from these parameters is obtained by plugging the parameters and feature matrix (X) into the logit equation:

$$Y_0 = \frac{1}{1+e^{-B_0\mathbf{X}}}.$$

After creating the author generated features, solutions to a simple logistic regression model on dataset samples did not converge. They did not converge using either the gradient descent or Newton-Raphson algorithms described below, nor when using Python's `sklearn.linear_model.LogisticRegression` or R's `glm` packages. The cause of this non-convergence is

unknown. Since this version of the dataset does not converge, the analysis in the Simulation studies section is done using the original Pima dataset, and not the author generated samples.

Generating mislabeled samples

There are five types of mislabeling schemes. The first scheme is to not mislabel anything, simply passing the dataset through as is. For all the remaining, numbered schemes, constants $u_0 = 0.05$ and $u_1 = 0.10$ are used.

Scheme S1 is y-dependent mislabeling, where the type 0 mislabel probability $\eta_0 = u_0$ and the type 1 mislabel probability $\eta_1 = u_1$. The mislabel chance is higher for successes (since u_1 is greater than u_0 than failures).

Scheme S2 is such that

$$\eta_0 = \eta_1 = u_0 + (u_1 - u_0)\,\frac{\exp(B_0^\top \mathbf{X})}{1 + \exp(B_0^\top \mathbf{X})}.$$

Here mislabeling for success and failures are identical depending on only on $\mathbf{X}$ though the logit function. Scheme S3 is very similar but exchanges the B_0 vector of author generated parameters with $b_j \in \mathbb{R}^9$, with $b_j \in \{0, 1\}$ corresponding to the success or failure of that datapoint. The individual values $b_{j,i} \sim \mathcal{N}(0, 4)$.

Finally, Scheme S4 uses distance from a certain value, based on $a \sim \mathcal{N}(2, 0.09)$, of the features $X_1 and X_3$ to weight the mislabel probability so that

$$\begin{aligned}\eta_0 &= u_0 + (u_1 - u_0)\, I\left(|X_1 - a| < 3, |X_3 + a| < 3\right)\\ eta_1 &= u_0 + (u_1 - u_0)\, I\left(|X_1 + a| < 3, |X_2 + a| < 3\right)\end{aligned}$$

Logistic Regression

Logistic regression assumes a logit link between the explanatory variables and the response variable, and a binomial distribution of errors. The logit link function is

$$\begin{aligned}\log\left(\frac{\pi(x)}{1 - \pi(x)}\right) &= \beta_0 + \beta_1 x_1 + ...\\ &= \beta x\end{aligned}$$

For each observation y_i let x_i be the explanatory variables associated with that observation and $\boldsymbol{\beta}$ be the parameters of the regression function we wish to estimate. Then

$$\pi_i(x|\beta) = \frac{1}{1 + e^{-\beta x_i}}.$$

A joint binomial probability mass function is

$$Pr(Y|X;\beta) = \prod_i \pi_i^{y_i}\,(1 - \pi_i)^{y_i}$$

with negative log likelihood function

$$\mathcal{L}\,(\beta|x, y) = -\sum_i y_i \log(\pi_i) + (1 - y_i)\log\,(1 - \pi_i)\,.$$

The derivative of $\pi = 1/(1 + \exp(-\beta x))$ can be expressed as

$$\begin{aligned}\frac{d\pi}{d\beta} &= \frac{d}{d\beta}\left(\frac{1}{1 + e^{-\beta x}}\right)\\ &= \frac{x e^{-\beta x}}{\left(1 + e^{-\beta x}\right)^2}\\ &= x\pi(1 - \pi)\end{aligned}$$

Thus the derivative of the log likelihood function is

$$\begin{aligned}\frac{d\mathcal{L}\left(\beta|x,y\right)}{d\beta} &= -\sum_i \left[y_i \frac{1}{\pi_i} \frac{d\pi_i}{d\beta} + \left(1-y_i\right) \frac{1}{1-\pi_i} \frac{-d\pi_i}{d\beta} \right] \\ &= -\sum_i y_i x_i \left(1-\pi_i\right) - \sum_i \left(y_i - 1\right) x_i \pi_i \\ &= -\sum_i x_i \left(y_i - \pi_i\right).\end{aligned}$$

The second derivative is

$$\frac{d^2\mathcal{L}\left(\beta|x,y\right)}{d\beta^2} = \sum_i x_i^2 \pi_i \left(1-\pi_i\right)$$

which is no-negative. Thus, the objective function is convex, and a global minimum for the objective function can be obtained by an iterative optimization methodology, such as gradient descent Newton's method.

Solution using Gradient Descent

Gradient descent is a first order, iterative, optimization algorithm. Applied to parameter optimization for logistic regression, it follows the most negative vector direction in the k dimensional parameter space to find the global minimum of the objective function F. Each iterative step $m \in \{0, 1, ...\}$ uses the gradient of the parameters to correct the parameters of the last step according to

$$\mathbf{b}_{m+1} = \mathbf{b}_m - \gamma \nabla F\left(\mathbf{b}_m\right).$$

Gamma is the step size. This algorithm is run until either a maximum number of iterations is run or the difference between successive step is sufficiently small ($||\mathbf{b}_{m+1}|| - ||\mathbf{b}_m|| < \delta$). The implementation is hardcoded with γ = 1e-4 and δ = 1e-6.

Solution using Newton-Raphson method

The Newton-Raphson method is a second-order, iterative, optimization algorithm. Instead of using a step size constant (γ) to determine the distance moved in the direction of the gradient, Newton-Raphson corrects the gradient with the second derivative. Newton-Raphson is usually expressed as a root finding algorithm

$$b_{m+1} = b_m - \frac{f(x_m)}{f'(x_m)}.$$

Since the objective function for logistic regression is convex, we can use this method to find the roots of the gradient to minimize the objective function (F). Since the second derivative of the objective function is a the hessian matrix ($\mathbf{H}(\mathbf{b})$, we express this as

$$\mathbf{b}_{m+1} = \mathbf{b}_m - \mathbf{H}^{-1}(\mathbf{b}_m) \nabla F(\mathbf{b}_m)$$

This algorithm runs until a maximum number of iterations is run, or the difference between successive steps is below a threshold, as with gradient descent. Newton-Raphson has the advantage of solving given problems in far fewers steps. For example, the logistic regression solution to the original Pima problem is solved in 1641 steps using gradient descent and the γ and δ parameters outlined above. Using the same tolerance (δ), Newton-Raphson generates the same solution in six steps.

However, calculating the inverse of the Hessian matrix is costly, and its cost relative to the gradient descent method increases as the feature matrix increases in size. The 1641 step solution using gradient descent runs roughly five times faster than the six-step Newton Raphson solution. There are alternative methods for estimating the Hessian Inverse that are referred to as pseudo-Newton methods, but these are not implemented here.

Simulation Studies

Logistic Regression of Mislabel Possibilities

I did not have time to complete the different types of γ- and ϵ logistic regression for this assignment. The only analysis that I implemented correctly was simple logistic regression. However, we can still use this to perform some useful analysis on the the author's modeling setup. We generate 500 data point samples from the Pima data set using both no mislabeling, and all four mislabeling schemes. Each method we generate 500 estimated parameters, and then calculate the mean and 95% confidence interval for each combination of parameter and method. The results are shown in Table 1.

As we see from the table, for the used values of u_0 and u_1, following the values given in the paper, the 95% confidence intervals of all nine features, including the intercept, have considerable overlap between the 'correctly' labeled data, and all four mis-labeling schemes. Keep in mind, this is for the original Pima data set, and not the author generated response variables, which I was unable to get a solution for using regular logistic regression.

Since the 95% confidence intervals overlap for regular logistic regression for the 'correctly' labeled and mis-labeled samples, logistic regression with this sample size does not have sufficient power to discriminate between the different labeling schemes at that confidence level. If this is also true for the author generated response variables, then it raises doubts that the classification accuracy gains from γ-logistic shown in Figure 3 of the paper are statistically significant.

This implies that the authors should have chosen a larger sample set; however, the Pima set itself is limited to 768 total entries. Perhaps a different, larger data set should have been used to demonstrate the capabilities of the author's new method.

			95% Confidence	
Parameter	Mislabel Type	Mean	Minimum	Maximum
Intercept	No Mislabel	-0.877	-0.990	-0.764
	S1	-0.803	-0.939	-0.667
	S2	-0.652	-0.798	-0.506
	S3	-0.659	-0.825	-0.493
	S4	-0.619	-0.769	-0.470
x1	No Mislabel	0.417	0.273	0.562
	S1	0.323	0.165	0.482
	S2	0.300	0.137	0.463
	S3	0.310	0.138	0.482
	S4	0.293	0.133	0.454
x2	No Mislabel	1.128	0..974	1.282
	S1	0.870	0.686	1.053
	S2	0.862	0.670	1.053
	S3	0.859	0.662	1.055
	S4	0.809	0.615	1.003
x3	No Mislabel	-0.257	-0.374	-0.139
	S1	-0.193	-0.342	-0.043
	S2	-0.195	-0.336	-0.054
	S3	-0.193	-0.347	-0.038
	S4	-0.183	-0.336	-0.030
x4	No Mislabel	0.005	-0.124	0.135
	S1	0.006	-0.145	0.158
	S2	0.006	-0.154	0.166
	S3	-0.003	-0.170	0.164
	S4	-0.001	-0.155	0.153
x5	No Mislabel	-0.140	-0.280	-0.001
	S1	-0.101	-0.263	0.060
	S2	-0.112	-0.266	0.041
	S3	-0.102	-0.262	0.057
	S4	-0.090	-0.248	0.068

x6	No Mislabel	0.719	0.563	0.875
	S1	0.538	0.345	0.731
	S2	0.521	0.344	0.698
	S3	0.514	0.324	0.703
	S4	0.491	0.319	0.662
x7	No Mislabel	0.312	0.178	0.447
	S1	0.225	0.077	0.373
	S2	0.226	0.072	0.380
	S3	0.237	0.079	0.394
	S4	0.226	0.073	0.379
x8	No Mislabel	0.176	0.033	0.317
	S1	0.136	-0.026	0.298
	S2	0.140	-0.020	0.301
	S3	0.125	-0.055	0.306
	S4	0.125	-0.043	0.293

Table 1: Distributions of Standardized Logistic Regression Parameters for 500 samples with sample size $n = 500$