Project 4: Logistic Regression with ADMM

One of the most popular methods for predicting categorical or binary data is by using logistic regression models. These models predict the log odds of observing a binary response which can then transformed into the probability of that response. As opposed to using normal equations in traditional linear regression, logistic regression coefficients are found by maximizing the log likelihood function for binomial distribution. To avoid overfitting, parameter penalizations can be added to this likelihood function to lower the variance when using other datasets. The most common penalization methods are Ridge (L2) and Lasso (L1).

When working with big data, Alternating Direction Method of Multipliers (ADMM) can be used to paralyze and speed up the fitting of these models. ADMM is an iterative method where each iteration performs a Map-Reduce job using augmented Lagrangian's. ADMM can be applied to consensus optimization problems, in particular fitting logistic regression models. First, the entire data set is mapped and split into n-partitions. Each partition is then sent to a reducer that uses an optimization method to find the parameter that minimizes equation (1):

$$\beta_i^{k+1} = \min_{\beta_i}(l_i(y_i, X_i^T\beta_i) + \frac{\rho}{2}\left\|\beta_i - \beta^k + u_i^k\right\|_2^2 \qquad (1)$$

$$u_i^{k+1} = u_i^k + (\beta_i^{k+1} - \overline{\beta^{k+1}}) \qquad (2)$$

The function $l_i(y_i, X_i^T\beta_i)$ is the negative log likelihood of binomial distribution, β_i^{k+1} is the parameter at iteration k for partition i. In addition to these, u_i^{k+1} is the updated consensus and $\overline{\beta^{k+1}}$ is the parameter average among all partitions at iteration k+1. This process ends once it converges to to a final parameter value.

In this project we were tasked with finding the coefficients to a logistic regression model given 10 partitions of a large dataset using ADMM. For the first iteration, each reducer solved for the minimum:

$$\beta_i^1 = \min_{\beta_i}(l_i(y_i, X_i^T\beta_i) + \frac{\rho}{2}\|\beta_i\|_2^2 \qquad (3)$$

To solve equation (3), an optimization method had to be selected. The traditional Newton Raphson was first considered but was found to be infeasible due to the exceedingly large diagonal matrix required for the Hessian. Instead, the BFGS algorithm, like stochastic gradient descent, was used using the '*optim'* function in R. For the first iteration in BFGS, the initial coefficients used were the ones found by fitting a logistic regression model on each respective partition. The value for the regularization hyperparameter (ρ) was found by using the *glmnet* package to perform cross validation of an L2 penalized logistic regression model on the first partition.

After all parameters for each partition was solved for, a consensus computation was performed by averaging the coefficients. The consensus parameter was sent back to each reducer where its respective consensus would be updated for the next iteration in equation (2). For every iteration going forward, equation (1) would be solved using BFGS with the same initial coefficients and regularization hyperparameter. After several ADMM Map-Reduce iterations, these parameters

closely converged to a final parameter value. Although they did not converge to zero from the previous iteration, it close enough to stop the iteration. In the table below are the parameter values:

Final Logistic Regression Coefficient Values	
Parameter	Value
X1	1.664
X2	0.642
X3	2.183
X4	-0.056
X5	-0.102
X6	-0.056
X7	-0.056
X8	-0.025
X9	-0.076
X10	0.189
X11	-0.014
X12	-0.056
X13	-0.028
X14	-0.034
X15	-0.069
X16	-0.039
X17	-0.021
X18	-0.039
X19	-0.047
X20	-0.060
X21	-0.120
X22	-0.001
X23	1.738
X24	-0.099
X25	2.913

In the folder of this document, you will find the three R scripts that were iterated to find these coefficients. The first one, ‘p4_consensus.R’ is the consensus used to average the partitioned parameters, ‘p4_initial_reducer.R’ is the reducer used in the first iteration, and ‘p4_reducer.R’ is the reducer used in the following iterations.