CS6240 Project
Distributed Computation of Linear Regression through Parallelized Matrix Multiplication and Matrix Inversion

Supplementary Text

Team Members: Komal Pardeshi and Sean Yu
Team Github: https://github.com/CS6240/project-cs6240project-sean-komal

1 Linear Regression and Ridge Regression

Linear regression is as a foundational statistical algorithm for predicting values based on unseen data. In linear regression, the goal is to find the estimators (β) that minimize the errors (ϵ) between a linear predictor and the data. Therefore, given a single set of unseen feature data (x) a prediction can be given by eq 1.

$$y = \beta_0 + \beta x + \epsilon \tag{1}$$

To find the best estimators a possible solution is to minimize the residual sum of squares (RSS). This is given by eq 2. Note y represents the values of a real dataset and $\hat{y}$ are values predicted using a set of estimators.

$$RSS = \sum_{i=1}^{n} (y - \hat{y})^2 \tag{2}$$

One common approach to linear regression is ordinary least squares (OLS), which analytically calculates estimators based on a dataset of features (X) and their corresponding values (y). This solution can be found using matrix operations and is shown in eq 3.

$$\beta = (X^T X)^{-1} X^T y \tag{3}$$

However, in the context of big data the number of features may be large and not all of them maybe useful in predicting the value. Ridge regression is an extension to the OLS model in which the estimators aim to minimize the RSS plus a penalty term consisting of the estimators. This is shown in eq 4. Note λ is a hyperparameter.

$$Error_{Ridge} = \sum_{i=1}^{n} (y - \hat{y})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{4}$$

Since the estimator values are also considered in the error to be minimized, the model is encouraged to decrease the magnitude of the estimators as long as it does not lead to an increase in RSS. Similar to the original OLS, this can also be solved through matrix operations. The solution is shown in eq 5.

$$\beta_{ridge} = (X^T X + \lambda I)^{-1} X^T y \tag{5}$$

2 Calculating the inverse matrix through Cholesky factorization

To help calculate the inverse first the matrix is decomposed using the Cholesky factorization This makes calculating the inverse easier as shown in equation 7.

$$A = LL^T \tag{6}$$

$$A^{-1} = (LL^T)^{-1} = (L^{-1})^T L^{-1} \tag{7}$$

Therefore, to solve OLS and ridge regression only the inverse of a lower triangle matrix is required. The rest of the operations are matrix multiplications. This can be done in parallel by columns of the inverted matrix. To briefly explain why this is possible, recall for a given matrix (A), the inverse (B) is defined by eq 8.

$$AB = I \tag{8}$$

Using similar intuition to horizontal matrix multiplication the first column of B can be found by solving eq 9.

$$A \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{9}$$

Similarly, the second column of B is given by the following:

$$A \begin{bmatrix} b_{12} \\ b_{22} \\ \vdots \\ b_{n2} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \tag{10}$$

Therefore, each column of the inverse matrix can be solved in parallel. In addition, solving this for a lower triangle matrix is much easier than a normal matrix. Eq 11 computes the first column of the inverse as an example.

$$\begin{bmatrix} L_{11} & 0 & 0 & \dots & 0 \\ L_{21} & L_{22} & 0 & \dots & 0 \\ L_{31} & L_{32} & L_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ L_{n1} & L_{n2} & L_{n3} & \dots & L_{1n} \end{bmatrix} \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ \\ b_{n1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ \\ 0 \end{bmatrix} \tag{11}$$

This reduces to several linear equations each with one increasing unknown variable (b).

$$\begin{aligned} L_{11}b_{11} &= 1 \\ L_{21}b_{11} + L_{22}b_{21} &= 0 \\ L_{31}b_{11} + L_{32}b_{31} + L_{33}b_{31} &= 0 \\ &\vdots \\ L_{n1}b_{11} + L_{n2}b_{31} + \cdots + L_{nn}b_{n1} &= 0 \end{aligned}$$

Since each equation only has one additional unknown the equations can be solved relatively fast. Although unfortunately this has to be done sequentially. Although solving for each column of the inverse matrix parallelly not seem like a lot of parallelism, in the context of solving regression problems it is arguable that many of the datasets typically have much more examples (rows) than features (columns) and the inverse matrix is only applied to a matrix of size of features by features.