Optimization
Project 3 – Non-Linear Programming

Deliverables

One Python code file (.py or .ipynb) and one PDF file, submitted to Canvas. Your report should go into some detail about how you solved the problem, include some graphs that explain your results, and include relevant code chunks in the final output. 66% of your grade will be based on whether you get the problem right or not, the remaining 34% will be based on the quality of the presentation of your analysis. We will re-run your code with a new data set. If you don't get the right answer or your Python file doesn't run, we will go through your code and give partial credit accordingly. The easier it is to read your code the easier it is for us to understand what you're doing, so use a lot of comments in your code!

Problem Overview

One of the most common problems in predictive analytics is variable selection for regression. Direct variable selection using optimization has long been dismissed by the statistics/analytics community because of computational difficulties. This computational issue was part of the motivation for the development of LASSO and ridge regression. However, in the recent past there have been tremendous advancements in optimization software, specifically the ability to solve mixed integer quadratic programs (MIQP). This project will pose the variable selection problem for regression as an MIQP which you will solve using gurobi. You will compare the results you find to LASSO to see if the additional 'shrinkage' component of LASSO really is more beneficial than finding the 'best' set of variables to include in your regression.

Direct Variable Selection – MIQP Problem

Given a dataset of m independent variables, X, and a dependent variable, y, the standard ordinary least squares problem is formulated as

$$\min_{\beta} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_m x_{im} - y_i)^2.$$

In order to incorporate variable selection into this problem we can include some binary variables, z_j, that force the corresponding values of β_j to be zero if z_j is zero, using the big-M method that we discussed in class, and used in the previous project. If we only want to include at most k variables from X, then we can pose this as

$$\min_{\beta,z} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_m x_{im} - y_i)^2$$
$$s.t. -Mz_j \le \beta_j \le Mz_j \quad for\ j = 1, 2, 3, \dots, m$$
$$\sum_{j=1}^{m} z_j \le k$$
$$z_j \ \ are\ binary.$$

Note that we don't ever forbid the model from having an intercept term, β_0, and that m and M are different things here. Here, k can be viewed as a hyperparameter to be chosen using cross validation.

In order to pose this in the standard framework of a quadratic programming objective let's see how we can rewrite this objective using linear algebra. Let β be an (m+1) x 1 column vector that contains $\beta_0, \dots, \beta_m$, let X be the ***n x (m+1)*** matrix that has its first column made up entirely of 1s, and columns 2 to (m+1) have the data from the m independent variables, and let y be the n x 1 column vector that has the dependent variable data. You can use np.array to convert the pandas dataframe to a matrix and then add a column of all 1s to the matrix. Then we can create an n x 1 vector whose entries are the n values inside the parentheses from the problem statement by doing the following matrix calculation: $(X\,\beta - y)$. Then if we want to take the sum of squared entries of this vector, we can multiply $(X\,\beta - y)^T * (X\,\beta - y)$. Using a few tricks from linear algebra we can pose the optimization problem's objective function as

$$\min_{\beta,z} \beta^T (X^T X)\,\beta + (-2\,y^T X)\,\beta.$$

The only issue left to be resolved is that the vector of decision variables needs to be of size (2m+1) x 1; made up of the (m+1) values of β and the m values of z, but the objective written above only includes the m+1 values of β. To fix this we can assign the Q matrix to be a (2m+1) x (2m+1) matrix where the upper left corner of the matrix is equal to X^TX, and all other values are zero. We also need the linear term of the objective to be a (2m+1) x 1 vector where the first (m+1) components are $-2y^TX$, and the rest are zeros. Now if you create the constraint matrix and right-hand-side vector from the constraints given above, you have all you need to solve the problem using gurobi.

It would be WAY easier to put this into Gurobi without these matrices and vectors, instead using tools like gp.quicksum(). But if you want to do it that way, you need to figure it out for yourself!

Indirect Variable Selection – LASSO

The LASSO version of regression is posed as

$$\min_{\beta} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_m x_{im} - y_i)^2 + \lambda \sum_{j=1}^{m} |\beta_j|,$$

where λ is a hyperparameter to be chosen using cross-validation. It turns out that if λ is large enough, several values of β will be forced to be equal to zero. This model also has the benefit of 'shrinking' the βs closer to zero, which achieves variance reduction (prevents overfitting). Note again that β_0 is not included in the λ sum. You should never penalize a model for having an intercept term. The standard package in Python to solve the LASSO problem is scikit learn. In this project you will need to use scikit learn to solve the LASSO problem.

Specifics

1) On canvas there are 2 data sets that include x and y data. One data set is a training data set, and one is a test data set. You will follow the data science pipeline carefully here. You will first do 10-fold cross validation on the training set to pick k or λ. Then using the optimal values of k or λ you will fit your βs using the entire training set. Then with those βs you will make a prediction of the y values on the test set, and compare your prediction of y, to the true value of y in the test set.

2) In order to do cross validation on the MIQP model you will have to write your own cross validation code. Randomly shuffle your data and split it into 10 folds. You can use the 'np.random.choice()' function in Python to shuffle your data. There are 50 X variables, and you will need to try k = 5, 10, 15, …, 50 in your cross validation. This means to do 10-fold cross validation with all possible values of k, you will have to solve an MIQP model 100 times! Pick the value of k that corresponds to the smallest cross validation error: for a given value of k, sum each validation set's sum of squared errors using the βs found using the other 9 folds' data to solve the MIQP. When k is 5 or 50, gurobi should solve the problem pretty quickly, but when k is 25 it will probably take a long time. Therefore, you should set a time limit for gurobi to solve each problem. Don't let the entire process run for any longer than 12 hours. Again, be smart about setting this up, so that you can run it once, save the results, and then import those results when you create the final version of your python code and pdf file. Also, set a time limit variable at the very beginning of your python code so that we can shorten the runtime of your code when we grade it.
    a. It is very important to remember that gurobi assumes all decision variables are non-negative. In order to allow your βs to be negative or positive you must set the lb value of your model to be -M for the appropriate decision variables.
    b. It is also very important that you choose M to be large enough so that no value of β is equal to M or -M. If you solve the problem and one of your βs is M or -M then you should double M and resolve the problem. Repeat this process until no β is equal to M or -M.
3) Once you find the k with the smallest cross validation error, fit the MIQP model on the entire training set using that value of k. Use the βs you find in this MIQP to make a prediction of the y values in the test set.
4) Use scikit learn to do 10-fold cross validation on the training set to pick λ. Once you find the best value of λ, fit a LASSO model to the entire training set using that value of λ. With the βs you find in that LASSO model make a prediction of the y values in the test set.
5) Pretend that you are a junior consultant at an analytics consulting firm. You frequently use LASSO in your job, but your boss has heard that the computational time of direct variable selection has decreased with the advent of better solvers. Your boss wants you to figure out if the firm should shift away from using LASSO to incorporate more direct variable selection. Write this project as if this is what you're going to deliver to your boss. Describe the advantages and disadvantages of both techniques. Your boss is pretty technical and understands optimization, so don't be afraid to include quantitative material. Your boss is also busy, so be sure to include some visualizations to get the important points across. For the purpose of your report, you can assume that your boss is interested in the data posted with the project.