Group #9
Anmol Agrawal, Jahnavi Angati, Luis Rafael Villazon Sanchez, Nikhil Suresh Nair
RM 294

VARIABLE SELECTION

Executive Summary:

Our firm's current standard practice for regression-based predictive modelling is the LASSO technique, appreciated for its simplicity and efficacy in variable selection. However, advancements in computational solvers have made direct variable selection methods, particularly Mixed Integer Quadratic Programming (MIQP), more accessible and potentially advantageous. This report evaluates whether MIQP can serve as a superior alternative to LASSO for our firm's analytics projects.

The findings from this study are crucial for businesses looking to leverage data analytics for strategic decision-making. The choice between LASSO and MIQP can significantly impact the accuracy of predictive models, operational efficiency, and ultimately, the effectiveness of data-driven business strategies.

Introduction

In the dynamic world of data-driven business strategies, selecting the optimal set of variables for regression models is not just a technical task, but a cornerstone for achieving actionable insights and maintaining competitive advantage. This report delves into a comparative analysis of two prominent variable selection methodologies: LASSO (Least Absolute Shrinkage and Selection Operator) and MIQP (Mixed Integer Quadratic Programming), evaluating their impact on predictive accuracy and their suitability for business applications.

Our approach employs an MIQP model, utilizing Gurobi's advanced optimization capabilities, to meticulously select the most relevant variables. This method shows promise in its precision and its adept handling of multicollinearity, a common hurdle in predictive modelling. On the other side of the spectrum, LASSO, renowned for its simplicity and speed, brings an attractive proposition for businesses seeking quick model deployment. Its inherent feature of inducing sparsity effectively prevents overfitting, a critical aspect in maintaining the generalizability of models.

Both MIQP and LASSO are geared towards refining the variable selection process, with an emphasis on mitigating overfitting and enhancing model clarity. MIQP shines in its ability to identify an optimal set of variables, thereby elevating the interpretability of the model and ensuring that the selected predictors are genuinely informative. LASSO, with its swift computational approach and intuitive model-building process, excels in swiftly distilling complex data into manageable, impactful insights.

In this report, we aim to balance the technical merits of each method with their practical implications for business. We seek to uncover which approach not only enhances model performance but also aligns seamlessly with business objectives, considering factors such as time-to-deployment, model complexity, and the scalability of the solution. By dissecting the strengths and limitations of both MIQP and LASSO, we provide a guided framework for businesses to make informed decisions in the realm of variable selection, ultimately aiming to leverage data analytics for strategic advantage.

MIQP Methodology

The objective of our MIQP formulation was to minimize the sum of squared errors in a regression model while selecting a subset of variables

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

MIQP was formulated to include variable selection directly in the regression model. The approach used binary variables (*zj*) and the big-M method to enable or disable individual predictors (force the corresponding values of βj to be zero if *zj* is zero). If we only want to include at most k variables from X, then we can pose this as:

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

It's important to acknowledge that our model always includes the intercept term, denoted as *β*0; this element is never excluded from the analysis. Additionally, it should be noted that in our notation, *m* and *M* represent distinct entities. In this context, *k* serves as a hyperparameter, which we determine through the process of cross-validation.

Implementation

In our approach, we employed a 10-fold cross-validation on the training dataset, a method crucial for ensuring the robustness and generalizability of our model. This process involved a thorough shuffling of the data, followed by its division into 10 equal segments. For each segment, we utilized 90% of the data for training and the remaining 10% for validation. In executing the Mixed Integer Quadratic Programming (MIQP) model with Gurobi, our objective was to minimize the sum of squared errors in the regression model. This included integrating both decision variables (*β*) for regression coefficients and binary variables (*z*) for the selection of variables. A crucial aspect of our model was the enforcement of constraints

that allowed the selection of up to *k* variables, adhering to the big-M method for variable inclusion.

The code below shows how we implemented the MIQP methodology and defined the relevant variables with bounds, set up the objective function and constraints.

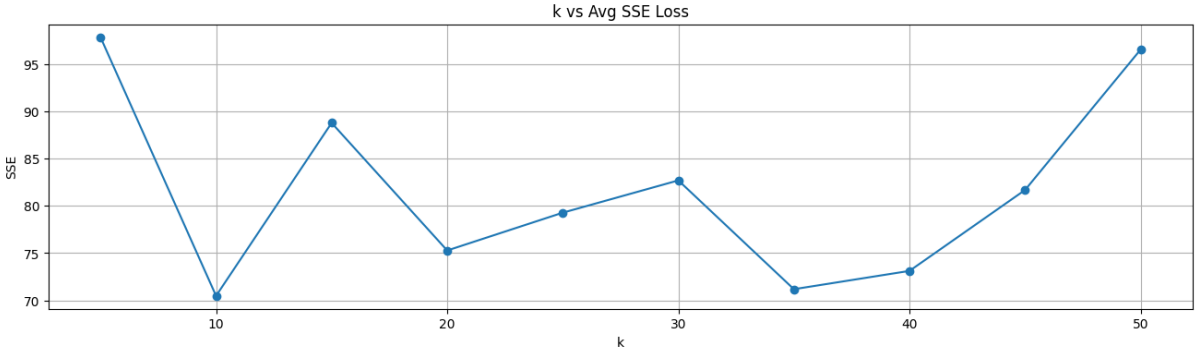
```
qpModel = gp.Model()

# Define decision and selection variables
betas = qpModel.addMVar(m + 1, lb = -np.inf)
zvars = qpModel.addMVar(m, vtype = 'B')

# Define Objective
qpModel.setObjective(
    gp.quicksum(
        (betas[0] + gp.quicksum(betas[j+1]*X_train_fold.iloc[i, j] for j in range(m)) - y_train_fold.iloc[i])
        * (betas[0] + gp.quicksum(betas[j+1]*X_train_fold.iloc[i, j] for j in range(m)) - y_train_fold.iloc[i])
        for i in range(len(X_train_fold))
    )
)

# Constraint for selecting <= k variables
con1 = qpModel.addConstr(gp.quicksum(zvars[j] for j in range(m)) <= k)
# Big-M consraint
con2 = qpModel.addConstrs(betas[j+1] <= M * zvars[j] for j in range(m))
con3 = qpModel.addConstrs(betas[j+1] >= -M * zvars[j] for j in range(m))
```

The code snippet shows that the model defines decision variables betas for regression coefficients and binary variables ‘zvars’ for selecting which features (variables) to include in the model. An objective function is specified to minimize the sum of squared errors between the predictions of the model and the actual target values in the training data. There are also constraints to ensure that exactly k features are selected. The first constraint ‘con1’ ensures that no more than k variables are selected, while the second ‘con2’ and third ‘con3’ constraints link the betas and ‘zvars’, enforcing that the coefficient betas can only be non-zero if their corresponding ‘zvars’ are set to 1.

In our implementation, the number of variables included in the model, represented by *k*, was a key hyperparameter. We tested a range of *k* values from 5 to 50, in increments of 5, to determine the optimal number through cross-validation. Special attention was given to the selection of the big-M value. We adopted an iterative approach, doubling the value of *M* and re-solving the model whenever a *βj* value reached the ±M limit. This ensured that our constraints were appropriately stringent, neither too lenient nor excessively restrictive. To manage computational resources efficiently, we set a time limit for solving each MIQP problem. This time constraint was especially vital for preventing overly prolonged runtimes, particularly for intermediate values of *k*. The extensive cross-validation process, entailing the solution of 100 MIQP models (10 for each value of *k*), culminated in the results being systematically compiled into a DataFrame and stored in a CSV file, enabling a thorough analysis of the model's performance across various *k* values.

Results

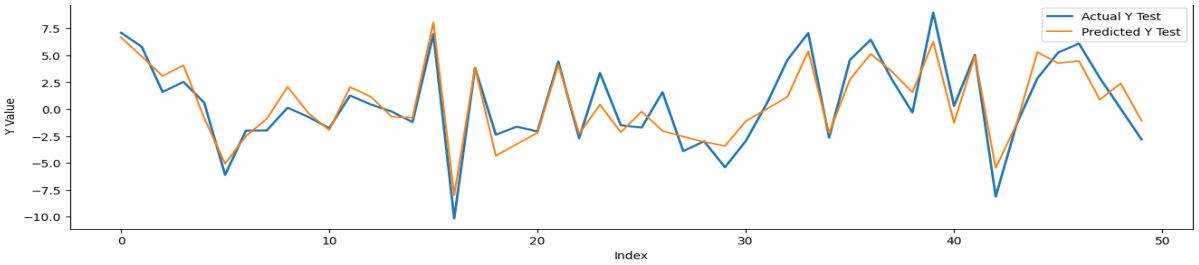
The following image illustrates a line chart that displays the relationship between various k values and their corresponding Average Sum of Squared Errors (SSE). This graphical representation is designed to aid in identifying the most effective k value for the model, which is the one that minimizes the SSE (k=10), thus optimizing the model's performance.



The graph clearly indicates that the minimum Average SSE is achieved when k equals 10. Consequently, we identify 10 as the ideal k-value for optimizing our MIQP model. Moving forward, we will adopt this k-value, apply the MIQP model to the entire training dataset, and then perform predictions on the test set to determine the model's MSE performance.

Range of K Values	5 to 50 in intervals of 5
Best K Value	10
Number of Best Non-Zero Features	9
Training Data SSE	615.017
Test Data SSE	124.741

Upon optimizing the MIQP model with the best k value of 10, we applied it to the test dataset to assess its predictive performance. This evaluation showed an SSE of 124.741, indicating the model's accuracy on unseen data. Also to visually represent the model's effectiveness, we generated a plot comparing the actual and predicted Y values obtained from the MIQP model.



The chart presents a comparison between actual values and predictions made by a MIQP with the data points indexed from 0 to just below 50. The actual values, represented by the orange line, and the predicted values, represented by the blue line, exhibit similar trends and fluctuations throughout the index range, indicating that the MIQP model has captured the underlying pattern of the dataset to a reasonable extent. However, there are noticeable divergences between the actual and predicted values at certain points, for instance, around the index of 20 and after index 40, where the predicted values either overestimate or underestimate the actual observations. The overall closeness of the two lines suggests that the MIQP method is providing a good fit for this particular dataset.

Lasso Approach

The LASSO regression can be described by the following equation:

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

where λ is determined through cross-validation. When λ is sufficiently large, it compels several β coefficients to become zero. Additionally, this model has the advantage of compressing the β coefficients towards zero, thereby reducing variance and helping to prevent the model from overfitting. It's important to note that the β0 term, or the intercept, is not subject to the λ penalty, as penalizing the model for its intercept is not standard practice.

Implementation

We applied a 10-fold cross-validation technique to tune the hyperparameters of the Lasso regression model, ensuring that our selection of the lambda parameter minimizes overfitting while maintaining predictive accuracy. The cross-validation approach allows for a comprehensive assessment of the model's performance on different subsets of the data, providing a robust estimation of its generalization capabilities.

After determining the optimal lambda value, we employed it to constrain the Lasso model, effectively shrinking less informative coefficients to zero, hence enhancing the model's interpretability. This was done by fitting the model to the entire training dataset. The model was then used to predict values on the test dataset, where we quantified the model's prediction error using the Sum of Squared Errors (SSE).

The code for fitting lasso with 10 fold CV is shown below:

```
cv_folds = 10

lasso = LassoCV(cv = cv_folds, random_state = random_state)
lasso.fit(X_train_scaled, Y_train)

print(f'Lasso Alpha: {lasso.alpha_:.3f}')
```

```
def Lasso_SSE(X_scaled, Y):
    model = Lasso(alpha = lasso.alpha_)
    model.fit(X_train_scaled, Y_train)
    Y_pred = model.predict(X_scaled)

    sse = np.sum((Y - Y_pred) ** 2)

    return model, Y_pred, sse
```

Lasso regression model is being applied to the training data using cross-validation with 10 folds, ensuring that the model is robust and generalizes well to unseen data. The LassoCV function from scikit-learn automatically selects the optimal value of the regularization parameter α through cross-validation, balancing the trade-off between fitting the training data well and keeping the model simple to avoid overfitting. The random_state parameter ensures that the results are reproducible. After fitting the model to the scaled training features X_train_scaled and target Y_train, the chosen α is printed, rounded to three decimal places, providing transparency about the regularization strength used in the model.

Results

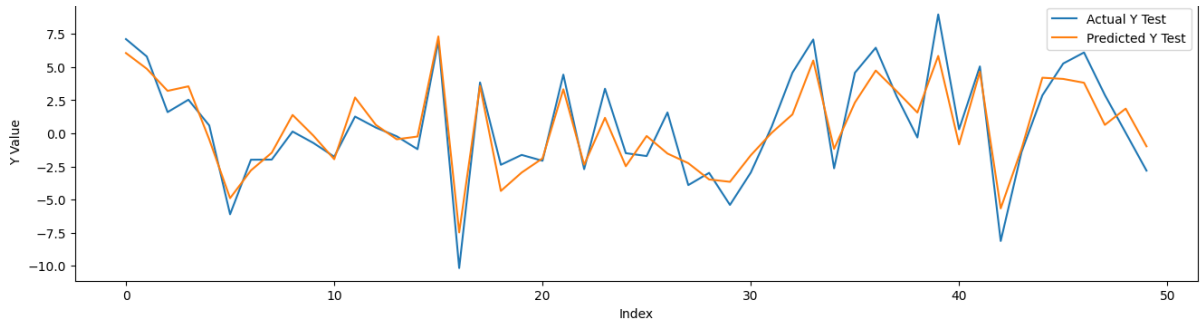
The following are the model's metrics and evaluation figures:

Number of CV folds	10
Best Lambda Value	0.085
Number of Best Non-Zero Features	18
Training Data SSE	596.611
Test Data SSE	117.833

As mentioned before and shown in the results table, the Lasso regression model was rigorously evaluated using a 10-fold cross-validation approach to ensure an unbiased estimation of its performance. The optimal lambda was determined to be 0.085, striking a balance between model complexity and predictive accuracy. Through the feature selection inherent in Lasso, 18 features were deemed significant, as they retained non-zero coefficients in the final model. The model exhibited a Sum of Squared Errors (SSE) of 596.611 on the training data, indicating the total squared deviation of the model's predictions

from the actual values during the training phase. When applied to the test data, the model achieved an SSE of 117.833, showcasing its ability to predict new, unseen data with a comparable level of accuracy. These metrics collectively demonstrate the model's effectiveness in both fitting the training data and generalizing to new data.

The next graph showcases the predictive performance of the Lasso regression model with the best-chosen lambda value.

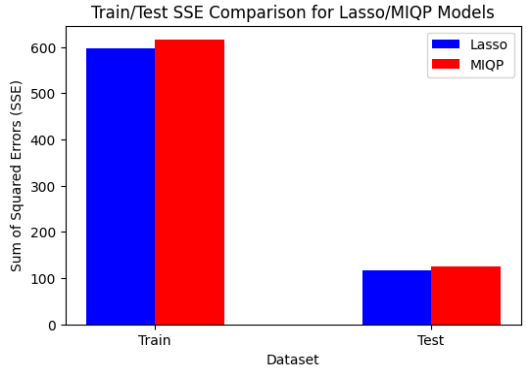


It plots the actual versus predicted values for the test set, displaying a good degree of congruence between the two, particularly evident in the range of indices from 0 to 20 and again from 40 to 50. This alignment suggests that the model, which has been optimized for both the number of features used and the degree of regularization, is proficient in capturing the trend of the dataset. However, between the indices of 20 to 40, there are visible discrepancies where the model predictions deviate from the actual observations. Despite these variances, the model's overall ability to closely follow the actual data validates its effectiveness in performing predictive analyses and reinforces its applicability to the dataset at hand.

Comparing MIQP vs Lasso

Test vs. Train SSE Plot

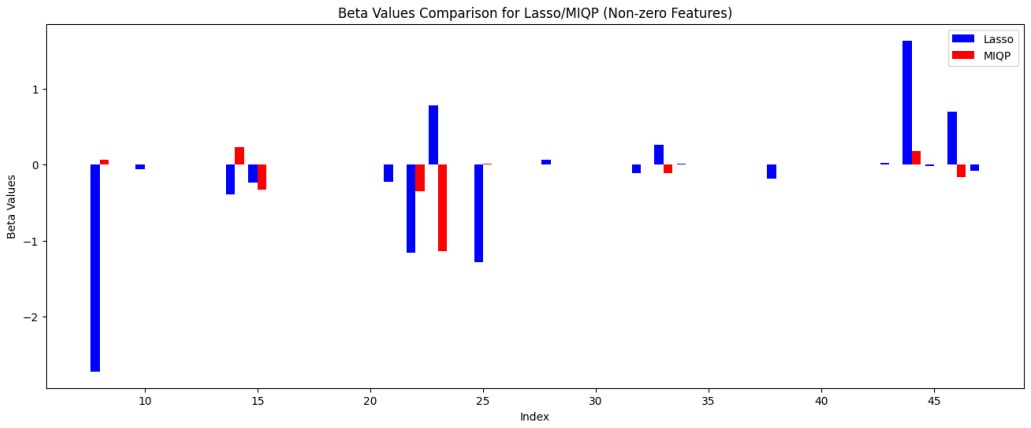
Comparing SSE across Lasso and MIQP shows us that for both Train and Test MIQP seems to have slightly higher SSEs as compared to Lasso.



The observed higher Sum of Squared Errors (SSE) in MIQP compared to LASSO for both training and testing datasets can be attributed to key methodological differences. LASSO's regularization approach not only selects variables but also shrinks their coefficients, enhancing the model's ability to generalize to new data and reduce overfitting, thus leading to lower SSE. MIQP, while offering precise variable selection, does not shrink coefficients, potentially resulting in models less adept at generalizing beyond training data. Additionally, LASSO's robustness against multicollinearity and its more straightforward handling of correlated predictors further contribute to its lower SSE as compared to the more complex and parameter-sensitive MIQP approach.

Beta Comparison Plot

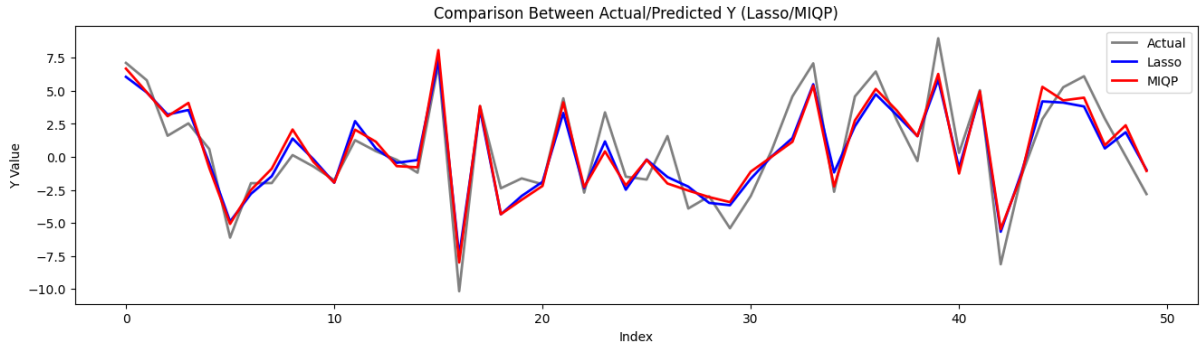
For the Lasso model the best number of features was *18* whereas for MIQP it was *9*. We also see that for the additional features that Lasso typically has, the value of those coefficients is fairly higher than for the same features in the MIQP model.



The graph showing a higher number of non-zero features with greater coefficients in the LASSO model compared to MIQP can be explained by the differences in their selection processes. LASSO tends to shrink coefficients toward zero but does not force them to be exactly zero unless the regularization is very strong, resulting in more features with larger coefficients. In contrast, MIQP performs a hard selection, often leading to fewer non-zero features with typically smaller coefficients, as it enforces a more binary inclusion without the shrinkage effect seen in LASSO. This distinction highlights LASSO's balancing act between fit and regularization, whereas MIQP offers a sparser model with a focus on feature distinction.

Actual vs. Predicted Output Comparison Plot

We see that both models seem to track relatively similar to the actual outputs.



This is likely because each method, despite different regularization strategies, can isolate key variables that significantly influence the outcome. The similarity in their performance suggests that the critical features for prediction overlap significantly between the two models. This could be a result of the dataset characteristics, such as a strong signal-to-noise ratio or low multicollinearity, allowing both LASSO's sparsity-inducing penalty and MIQP's direct selection process to identify an optimal subset of predictive features, resulting in comparably accurate models.

Conclusion and Recommendations

Our analysis reveals that Lasso regression, with its inherent regularization, tends to select more features while penalizing their coefficients, which might lead to a better generalization on unseen data by minimizing overfitting. The MIQP approach, noted for its precise selection, does not impose shrinkage on coefficients, potentially leading to models that are less generalized but may capture complexities in the data better.

Both Lasso and MIQP present a viable approach for feature selection. The decision to use either of them depends on specific requirements and domain specific context of the problem at hand.

- In the fast-paced environment of business operations, where time is often a critical factor, we recommend adopting the Lasso regression model for its efficiency and speed.
- When the business decisions are of high stakes, MIQP may be preferred despite its slower speed. Its ability to not shrink coefficients may capture the intricate dynamics of the data, which is crucial when precision outweighs the need for speed.
- Additionally, businesses may also consider a hybrid approach where both models are tested and evaluated against key performance indicators before finalizing the model choice if resources are not a constraint.