**Introduction:**

The first step is to examine the data and interpret it as much as possible. We applied the dataset to several models to find out which approach worked best. Having reviewed a variety of models, including linear regression and complex trees, and comparing the performance of each model, we have selected three that perform best. As a result, three top models will be evaluated and tuned further to optimize their performance: Forward stepwise selection, LDA and bagging.

For effective model development and selection, we have started by handling the raw data correctly, focusing on the missing values. The training data was first thoroughly checked and preprocessed to ensure a solid foundation for model building. In the first step, we inspected the dataset's basic statistics, dimensions, and column data types. The numeric columns were imputed using median for every parameter to deal with missing values. To ensure the data was suitable for modeling, we transformed all columns into integers. A new data frame 'traindata\_omitna' was created after imputation, containing the processed data, and we examined the types of columns and statistics for essential variables. To make future analyses easier, we recorded the dimensions of the final dataset and the number of rows and predictor columns to check if we are going ahead in the right direction or not.

**Model – 1: Forward Stepwise Selection**

To identify the best subset of predictors for our target variable, we choose forward stepwise selection. With 154 predictors (nvmax = 154), the forward selection algorithm is implemented using the 'traindata\_omitna' dataset. The model fits are then summarized, and diagnostic plots are created for evaluation metrics. So, the diagnostic plots below are displaying the relationship between number of variables and the Residual Sum of Squares (RSS), Adjusted R-squared (Adj-R2), Mallows' Cp (Cp), and Bayesian Information Criterion (BIC) respectively.

Line chart

Description automatically generated with medium confidence

Figure : Diagnostic plots of RSS, Adj-R2, Cp, BIC with variables 154,47,24 and 7 respectively.

Text, letter

Description automatically generated

Figure : The coefficients of the optimal model, containing 7 predictors, identified by the lowest BIC.

Looking at the above plots, we find the models which are giving us lowest RSS, Cp and BIC values and highest Adj-R2 (Indicated in red in Figure 1). From the analysis, it has become evident that the model with lowest BIC criterion value is most appropriate to move ahead. The BIC metric helps us identify an optimal model, which we will further analyze and optimize to improve our model's prediction accuracy, by using forward stepwise selection.

Next, we perform a k-fold cross validation analysis for our approach. We intend to proceed with 10-fold approach by giving the ‘k’ value as 10. To identify the optimal model, we average the cross-validation errors across all folds and plot them against the number of predictors. Considering the lowest cross-validation error, we can determine that the forward stepwise selection model with seven predictors is the best fit. In addition, this analysis demonstrates the effectiveness of cross-validation in evaluating model performance and selecting the most suitable model based on the data.

Chart, scatter chart

Description automatically generated

Figure : The 10-fold cross-validation error plotted against the number of predictors

According to Figure 3, as the number of predictors increases, the cross-validation error initially decreases, reaching a minimum value, and then increases as more predictors are added. This plot helps us to determine what the appropriate number of predictors should be in the forward stepwise selection model (indicated in red in Figure 3). In this case, we observe that the model with 7 predictors has the lowest cross-validation error, which indicates that it is the most generalizable.

A picture containing text

Description automatically generated

Figure : Optimal forward step-wise selection model with 7 predictors, showing the lowest mean cross-validation error

The cross-validation error value of 0.6378 represents the minimal mean error across all models tested using the k-fold cross-validation process. A model's predicted values are calculated by averaging the squared differences between the actual target values (traindata) for each predictor. When compared to other models with different numbers of predictors, the model with 7 predictors provides the best data with the lowest mean cross-validation error.

**Model – 2: Linear Discriminant Analysis**

We have chosen LDA as one of the models because it is classification modelling and we compared it to QDA (Quadratic Discriminant Analysis) we have found LDA has a lower error rate. LDA is a linear model that tries to find the optimal linear combination of features that maximizes the separation between different classes. The goals are to find a hyperplane that separates the classes with greatest margin. LDA uses Bayes’ theorem.

**Analysis I: Analysis with all variables:**

There are 154 variables and 479 observations (data points). In the beginning we took all the variables into account and fitted the LDA model for the observations. After train-test split of 75% training data we have found results such as:

We can see that there is a misclassification rate of 64% if we take the all the variables and an accuracy of 36%. We have converted the values into the integer and calculated the test MSE for comparison. We have got an MSE of 1.558.

A picture containing text

Description automatically generated

Figure : LDA model

A picture containing text

Description automatically generated

Figure : LDA confusion matrix

A picture containing text

Description automatically generated

Figure : Test mean error

**Analysis II - Improved Approach:**

Due to the high misclassification rate and MSE, one of the approaches to reduce the error rate is using only the most important variables and that in turn could reduce the misclassification. To find the most important variables, we have taken a few different approaches to solve this issue. One of the best methods to find those variables is to use Shrinkage Methods. We have tried both ridge and lasso to find the most important variables. Lasso Regression due to its L1 norm as the penalty term and it can completely reduce the coefficients to give a more defined set of variables compared to ridge which has L2 norm in its penalty term. and we have tried this approach with λ=0.05126 (found using cross validation). We found 21 important variables which drastically reduced the misclassification rate and gave us above 49% misclassification rate and a test MSE of 0.9583.

We tried to find other algorithms that would reduce the number variables by using selection algorithms as well to find the most important variables. After analysis we have found that using Forward Selection and Cross-validation gave us 7 variables. We have used these variables for the LDA model which further reduced our misclassification rate.

Text

Description automatically generated

Figure : Improved approach error

**Confusion Matrix:**

The LDA output indicates ˆπ1 = 0.0445 , ˆπ2 = 0.09749, ˆπ3 = 0.28133, ˆπ4 = 0.3481 and ˆπ5 = 0.22841; in other words 4.45% belong to class 1, 9.74% belong to class 2, 28.133% belong to class 3, 34.81% belong to class 4 and 22.841% belong to class 5 calculated among the training data set. It gives us a group means. This is used by LDA as estimate of

Table

Description automatically generated

Figure : Summary of the LDA improved fit

**Key insights from summary:**

* The variable Q100 has the most influence on LD1 when we are separating the classes because the absolute coefficient value is -0.92822
* The variable Q42 has the most influence on LD2 when we are separating the classes because the absolute the coefficient value is -1.269953
* The variable Q38 has the most influence on LD3 when we are separating the classes because the absolute the coefficient value is -0.96470
* The variable Q25\_6 has the most influence on LD4 when we are separating the classes because the absolute the coefficient value is -0.9286544

**The proportion of traces shows that:**

* LD1 explains 95.6% of the total variance between the classes.
* LD2 explains 3.17% of the total variance between the classes.
* LD3 explains 0.78% of the total variance between the classes.
* LD4 explains 0.45% of the total variance between the classes.

**Model – 3: Bagging**:

A study was conducted to compare the performance of bagging and random forest regression methods using the dataset. Initially, 75% of the data was randomly selected as the training set, and the remaining 25% was used for testing. Both random forest models were trained using the training set and the mean squared error (MSE) was evaluated on the test set.

To determine the optimal value of the m parameter in random forest, which specifies the number of variables randomly sampled as candidates at each split, the models were trained with different values of m ranging from 1 to 154. The test MSE was evaluated for each value of m, and a plot was created to visualize the relationship between m and the test MSE.

The plot shows that the test MSE decreased as m increased. The lowest test MSE was obtained when m was set to 154, indicating that the bagging method outperformed the random forest method in this dataset.

Chart

Description automatically generated

Figure : m values vs. Test MSE

In the same way, another study was conducted to evaluate the impact of the number of trees on the test mean squared error (MSE) in the model. The experiment involved incrementally increasing the value of ntree, representing the number of trees to be grown, from 5 to 500 while maintaining m, representing the number of variables randomly sampled at each split, at a constant value of 154. The results indicated that the test MSE reached a saturation point when the number of trees reached 100, as shown in the plot below. Based on these findings, the m value was set at 154 and the maximum number of trees was fixed at 100 for further analysis.

Chart

Description automatically generated with low confidence

Figure : ntree values vs. Test Errors

Once the m and ntree values are found out, they are used in the building of bagging trees iteratively by bootstrap sampling the training data. The output variable Q46\_2 is predicted using this model and the test MSE is calculated, which turned out to be 0.588 which is considerably less than Forward selection and LDA models’ test error.

Because the bagging model reduces the variance by estimating the mean of many trees, it is less interpretable now. So, Variable importance plot is presented below.

Chart

Description automatically generated

Figure : Variable Importance Plots

The above plot is the plot between each predictor variable and their %IncMSE, IncNodePurity numbers. The first measure, based on the mean decrease of accuracy in predictions on the out of bag samples when a given variable is excluded from the model, provides an estimate of the impact of each variable on the model's predictive accuracy. This measure is useful when the goal is to maximize the accuracy of the model's predictions. The second measure, based on the total decrease in node impurity that results from splits over that variable, provides an estimate of the contribution of each variable to the overall structure of the tree. This measure is useful when the goal is to understand the underlying relationships between the variables and the target variable. As our goal is to improve the model accuracy, the first measure explains which predictors are the most important. Of all the predictors, Q100 is the most significant one followed by Q16\_4, Q5\_1, Q29\_4, Q45\_2 and so on.

**Comparison of Models:**

We have used three models: Forward Selection, Bagging, and LDA. Forward Selection and Bagging are regression models we cannot directly compare to a classification model such as LDA. In Regression models, we mainly use the Test Mean Squared Errors to compare the models between them and for classification we calculate the misclassification rate (We have still provided test MSE for LDA). So, to compare we have used a heatmap to assess the predicted values.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model Name** | **Test MSE** | **Misclassification Error** | **Adj. R2** |
| Forward | 0.637 | - | 0.5017 |
| LDA | 0.63 | 0.43 | 0.4345 |
| Bagging | 0.588 | - |  |

A picture containing text, window blind

Description automatically generated

LDA Forward Selection Bagging y\_test

When comparing models using Test MSE as a metric, Bagging has the lowest score of 0.588, while LDA and Forward Selection follow closely with Test MSE scores of 0.63 and 0.637 respectively. It's worth noting that LDA is a classification model, so we can measure its performance in terms of misclassification rate, which in this case is 43%. Forward Selection, on the other hand, provides an Adj. R2 score.

Figure : Heatmap- Assessing Predicted Variables

The Heatmap shows the predicted values for different models, which we can compare to the true test data response variable (y\_test).

However, the Heatmap ranges from 0 to 5, which means that none of the true response values in y\_test can be below 1 since y\_test ranges from 1 to 5. This constraint does not apply to the regression models, which can predict values ranging from 0 to 5 and can also produce float values. However, this does not apply to LDA, as it only predicts the class.

In conclusion, we have chosen Bagging model as the final model.

**Phase-2: Test Error**

The best model presented in phase-1 is the bagging model, which achieved a test mean squared error (MSE) of 0.59 with the training data. However, when this same model was tested with the phase-2 data, the new test MSE obtained was 0.80.

Graphical user interface, text

Description automatically generated with medium confidence

Figure : Test MSE Phase 2

Below are the models that are fit for the complete training models after tuning (where required).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Regression Models** | **Test MSE** | **Train MSE** | **10-Fold CV Error** | **Train Model** | |
| **Adj-R2** | **R2** |
| Forward Stepwise Selection (7 Best Predictors) | 0.9394 | 0.5722 | 0.6378 | 0.5017 | 0.509 |
| Backward Stepwise Selection (6 Best Predictors) | 0.9680 | 0.5818 | 0.6219 | 0.4943 | 0.5006 |
| Ridge Regression | 0.9398 | 0.5125 | 0.7082 | 0.3510\* | 0.5601 |
| Lasso Regression (21 Best Predictors) | 0.85833 | 0.5740 | 0.6312 | 0.4847 | 0.5073 |
| Principle Component Regression (7 Forward Selection Variables) | 0.8267 | 0.6124 | 0.612 | 0.4665 | 0.4743 |
| Partial Least Squares | 0.8776 | 0.581 | 0.581 | 0.4987 | 0.5815 |
| Random Forest (m=128, ntree=100, maxnodes=50) | 0.7909 | 0.2405 | - | 0.6954 | 0.7936 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classification Models** | **Test MSE** | **Train MSE** | **Train Model** | | **Mis-classification Error** |
| **Adj-R2** | **R2** |
| LDA (7 Forward Selection Variables) | 1.08 | 0.6492 | 0.4345 | 0.4428 | 0.55 |
| QDA(7 Forward Selection Variables) | 1.39 | 0.7494 | 0.3472 | 0.3568 | 0.59 |
| Random Forest  (m=136, ntree=100, maxnodes=50) | 1.20 | 0.2359 | 0.75 | 0.79 | 0.48 |
| Support Vector Machines (linear) | 1.1 | 0.6179 | 0.4618 | 0.4696 | 0.57 |
| Support Vector Machines (radial) | 1.03 | 0.5866 | 0.4890 | 0.4965 | 0.56 |

\*- Used all predictors to calculate Adj-R2

After using Bagging, we got test error of 0.59 on the initial provided training data in Phase I, but the final model when tested on the phase-2 test data, the test error observed is 0.8. Bagging uses all the variables for its trees whereas Random uses only a subset of random subset of variables each time it produces its trees. A sensitivity study is performed on the random forest model by changing m values and ntree values sequentially. The optimal m is 128 and ntree value is 100 after which errors are saturating. **This ensured bias in the training of the model which improved our prediction error to 0.79.**

Graphical user interface, text, application

Description automatically generated

Figure : Improved Model

Please find the updated model in the R code attached.