CAP6673 Justin Johnson
Dr Taghi Khoshgoftaar Z23136514

Assignment 5
Logistic Regression vs Multilayer Perceptron

I) Introduction

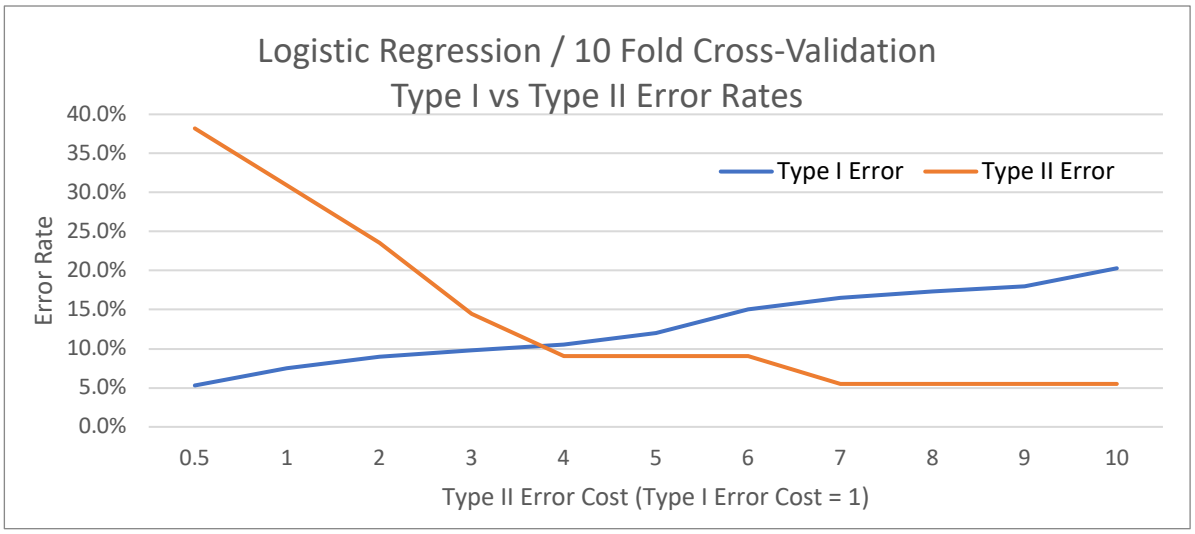
Logistic Regression and Multilayer Perceptron (MLP) classifiers are trained and evaluated with WEKA using fit and test data sets describing software modules. The data set instances contain 8 independent attributes which describe software module metrics, and a 9th dependent attribute which labels the software module as fault prone or non-fault prone. Both Logistic Regression and MLP classifiers are first evaluated using 10 fold cross-validation against a fit data set of 188 instances, and then evaluated using a separate test data set of 94 instances. For Logistic Regression, WEKA's cost sensitive classifier is used to vary the cost ratio and identify an ideal model during the cross-validation phase. The selected Logistic Regression model's test data classification results are then compared to the results produced by the Multilayer Perceptron.

II) Logistic Regression

The Logistic Regression classifier is combined with WEKA's cost sensitive classifier, and various cost ratios are applied to the classifier in order to identify an ideal model during the cross-validation stage of development. To identify an ideal model, Type I error cost is fixed at 1.0 and Type II error cost is varied between 0.5 and 10. For this assignment, model selection is determined by selecting the model that produces Type I and Type II error rates that are approximately the same, with a Type II error rate as low as possible. We are primarily concerned with reducing the Type II error rate, classification of fault prone modules as non-fault prone.

Logistic Regression 10 Fold Cross-Validation			
Type I Cost	Typie II Cost	Type I Error	Type II Error
1	0.5	5.3%	38.2%
1	1	7.5%	30.9%
1	2	9.0%	23.6%
1	3	9.8%	14.5%
1	4	10.5%	9.1%
1	5	12.0%	9.1%
1	6	15.0%	9.1%
1	7	16.5%	5.5%
1	8	17.3%	5.5%
1	9	18.0%	5.5%
1	10	20.3%	5.5%

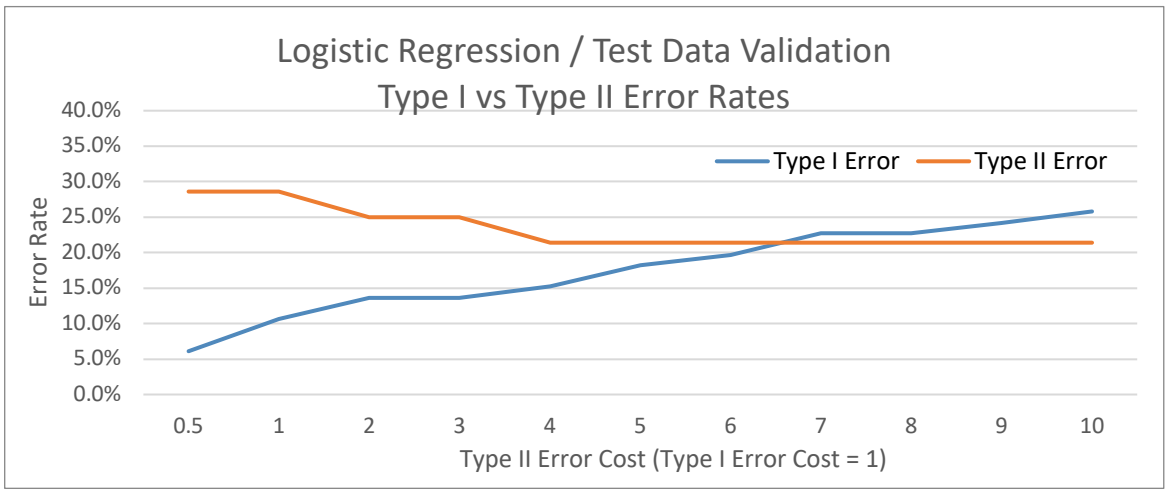
The above table displays the cross-validation results for the Logistic Regression classifier. The highlighted row, with a Type II cost of 4.0, was selected as the ideal model because Type I and Type II error rates are approximately the same and Type II error rate is low. The default cost ratio of 1 performed very poorly, with a high Type II error rate of 30.9%. Increasing the Type II error cost to 4.0 significantly reduced the Type II error rate down to 9.1%, with only a small increase in Type II error rate from 7.5% to 10.5%.



The chart of Type I vs Type II error rates as Type II error cost varies displays the tradeoff between Type I and Type II error rates. As Type II misclassifications decrease, there is an increase in Type I misclassifications. We have selected a Type II error cost of 4.0, as the error rates are balanced, and Type II error rate is low.

Next, all models are evaluated using the unseen test data set.

Logistic Regression Test Data Validation			
Type I Cost	Typie II Cost	Type I Error	Type II Error
1	0.5	6.1%	28.6%
1	1	10.6%	28.6%
1	2	13.6%	25.0%
1	3	13.6%	25.0%
1	4	15.2%	21.4%
1	5	18.2%	21.4%
1	6	19.7%	21.4%
1	7	22.7%	21.4%
1	8	22.7%	21.4%
1	9	24.2%	21.4%
1	10	25.8%	21.4%



The highlighted row in the above table denotes the model that was selected during cross-validation. The model selection strategy applied was successful in producing results with a low Type II error rate, with a Type I error rate of 15.2% and Type II error rate of 21.4%.

Compared to cross-validation, the test data results saw an increase in Type II error rate from 9.1% to 21.4% and an increase in Type I error rate from 10.5% to 15.2%. An increase in error rate is common when using a separate test data set to evaluate a model, as the model has not yet seen this data set.

Against the test data set, a default cost ratio of 1 resulted in Type I error of 10.6% and Type II error rate of 28.6%. One primary factor in these poor results is an imbalanced data set, where approximately 70% of the instances are non-fault prone, causing the model to favor non-fault prone classification. By applying cost-sensitive classification, the total number of fault prone modules that were misclassified as non-fault prone has been reduced from 28.6% to 21.4%.

III) Multilayer Perceptron

Next the Multilayer Perceptron, a class of feed forward neural networks, is trained and evaluated using the same fit and test data sets from Part II. A default cost ratio of 1 is used with the Multilayer Perceptron. WEKA MLP parameters are adjusted, setting the neural network to 1 hidden layer with 3 nodes and setting the validation set size to 10%. All other MLP parameters are left as WEKA default. Before working with the MLP classifier, both data sets are normalized to the range (0, 1).

First the MLP is trained using the normalized fit data set and evaluated using 10 fold cross-validation. The below table displays the Type I and Type II error rates:

Multilayer Perceptron 10 Fold Cross-Validation			
Type I Cost	Typie II Cost	Type I Error	Type II Error
1	1	9.8%	32.7%

The MLP cross-validation models performed poorly in comparison to the Logistic Regression cross-validation models. The Logistic Regression classifier with a cost ratio of 1 produced a Type I error rate of 7.5% and a Type II error rate of 30.9%. The MLP cross-validation saw in increase in both Type I error rate and Type II error rate.

Next, the MLP model is evaluated using the test data set. The below table displays the Type I and Type II error rates:

Multilayer Perceptron Test Validation			
Type I Cost	Typie II Cost	Type I Error	Type II Error
1	1	13.6%	25.0%

The MLP model performed slightly better against the test data set than it had during cross-validation, reducing Type II error rate from 32.7% down to 25%. There was an increase in Type I error rate from 9.8% up to 13.6%, but it can be argued that these results are better than those of the cross-validation phase because we are primarily concerned with reducing the Type II error rate.

Compared to the selected Logistic Regression model evaluated with the test data set, the MLP still performed worse, with an increase in Type II error rate from 21.4% up to 25%. The MLP produced slightly less Type I misclassifications, reducing Type I error rate from 15.2% down to 13.6%.

IV) Conclusions

Logistic Regression and Multilayer Perceptron classifiers were compared using the fault prone software module fit and test data sets. For Logistic Regression, cost sensitive classification was applied to vary the Type II error cost and select an ideal cost ratio. Once selected, the model was then evaluated using the test data set. For the Multilayer Perceptron, a default cost ratio of 1 was used. In addition, the MLP was set up to use 1 hidden layer with 3 nodes and a validation set size of 10%. The below table shows the results of each model when evaluated with the test data set.

Logistic Regression vs Multilayer Perceptron				
Comparing Test Data Evaluation Results				
Classifier	Type I Cost	Type II Cost	Type I Error	Type II Error
Logistic Regression	1	4	15.2%	21.4%
Multilayer Perceptron	1	1	13.6%	25.0%

The Logistic Regression model performed the best considering our goal of reducing Type II error rate, classifying fewer fault prone software modules as non-fault prone (Type II error). The Logistic Regression model had slightly more Type I misclassifications, but we are mostly interested in reducing Type II error rates. The MLP model actually had fewer Type I misclassifications, reducing from 15.2% down to 13.6%, but there was an increase in Type II misclassifications from 21.4% to 25%. This Type I vs Type II error rate tradeoff is expected.

The difference in classification results is small, just a 2 - 4 % difference in Type I and Type II error rates between the two classifiers. The fact that the MLP was given a default cost ratio of 1 and the performance was very similar to the Logistic Regression model which required cost sensitive classification is a significant one. The MLP was able to handle and learn the imbalanced data set (roughly 30% positive instances) without any additional work.

It should also be noted that mostly default MLP parameters were used for this assignment. I predict that tuning the MLP classifier can produce better results and potentially outperform the Logistic Regression model. Cost ratio can be adjusted to give Type II misclassifications a higher weight, the learning rate can be adjusted to improve the gradient descent optimization process and avoid local minima, and the number of hidden layers and nodes can be adjusted to identify a more appropriate neural network architecture for the problem at hand. I also predict that an increase in the training data set size will produce a more accurate model. Multilayer Perceptron learners are powerful, but can suffer from overfitting if the network is larger than what is required for the underlying data structure.