Machine Learning

Assignment 3, Group Assignment

Group 1: Siow Meng Low, Louise Fallon, Nikhita Venkatesan, George Pastakas, Cecilia Nok Sze Cheung, Steven Locorotondo

Q1: Import Pre-Processed Data and Splitting

The loans data has been preprocessed using Python code provided. There is a slight amendment to the Python file "cleanup.py": the first row of the CSV file is not required, hence we set *skiprows = 1* in *read_csv()* function. The modified Python code is submitted as well.

We first read the `.csv` file which contains the pre-processed data with 8 columns in total, including the outcomes of the loans, which can be either *"Charged off"* or *"Fully Paid"*.

Next, we split the data into:

- Training Set (20,000 records)
- Validation Set (8,000 records)
- Test Set (Remaining 10,697 records)

Q2: Classification Tree Using C50

The total number of loans in the data that fall into each category is

Table 1: Number of instances in each class

Charged Off	Fully Paid
5435	33262

The proportion of repaid loans in the dataset is 0.8595. We will now try to achieve an accuracy greater than this using classification trees. We first train a decision tree based on the training set.

The confusion matrix of the training set for the classifier we trained is

Table 2: Confusion Matrix for Training Set

actual / predicted	Charged Off	Fully Paid
Charged Off	0	2797
Fully Paid	0	17203

As observed, the classification tree classifies all training samples as *"Fully Paid"* and its accuracy is simply the proportion of *"Fully Paid"* loans in the training set, which in this case is 0.8601, close to the target accuracy of the total proportion of repaid loans.

We will now use the classifier to predict the outcome of each lean in the validation set.

The resulting confusion matrix of the validation set for the classifier is

Table 3: Confusion Matrix for Validation Set

actual / predicted	Charged Off	Fully Paid
Charged Off	0	1132
Fully Paid	0	6868

Again, all the validation records are predicted to be *"Fully Paid"*. Hence, its accuracy is the proportion of *"Fully Paid"* loans in the validation set, which in this case is 0.8585, close to the target accuracy of the total proportion of repaid loans.

To understand this behaviour, we will need to refer to the default **pruning behaviour of C5.0**. A subtree will be pruned if it has an error estimate higher than the "CF Option", with a default value 0.25 (this value can be set using *C5.0Control()*). A higher value of "CF Option" might increase the tendency of overfitting to training data.

As we know, the number of *"Fully Paid"* loans greatly outnumbers the number of *"Charged Off"* loans. Consequently, during the training phase, the algorithm discovers that further splitting the tree (using any of the seven features) does not reduce the error estimate of the subtree to below 0.25: even though the algorithm could potentially perform splitting such that *"Charged Off"* are majority categories in certain leaf nodes, this improvement is insignificant when compared to the error estimate of the subtree (due to the fact that *"Fully Repaid"* loans are the predominant class in most of the nodes).

Consequently, the algorithm prunes away all the subtrees and we are only left with one root node. Thus, the trained classification tree simply predicts all loans as *"Fully Paid"*.

Q3: C50 Classification Tree with Costs Adjustments

We will need to train the classification tree with a cost matrix to correct the default behaviour of `C5.0` in maximising the accuracy.

Granting a loan to a customer who is likely to default is much more costlier (i.e. False Negative) than denying loan to a customer who is likely able to pay back (i.e. False Positive). In the cost matrices, we set the cost of False Positive to 1 and test out different values of the cost of False Negative (ranging from 2.8 to 5.2, in increment of 0.1). The cost matrix is represented in the table below (with 'X' indicating the varying cost of False Negative which we would like to calibrate):

Table 4: Cost Matrix Representation

actual / predicted	Charged Off	Fully Paid
Charged Off	0	X
Fully Paid	1	0

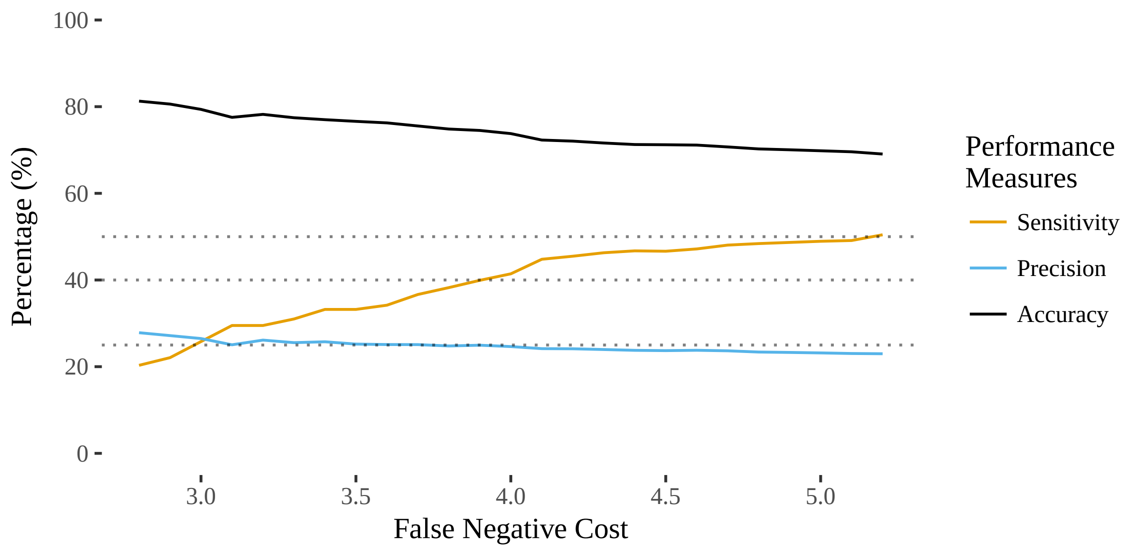
Note that in the above representation, columns are the predicted classes and rows are the actual classes. While passing in to C5.0 function in the R code, we use the transpose of this matrix since R function requires the cost matrix to be represented in the other way (i.e. columns correspond to actual classes and rows correspond to predicted classes).

Table 5: Validation Performance Using Different Cost Matrices

False Negative Cost	Sensitivity	Precision	Accuracy
2.8	0.2032	0.2785	0.8128
2.9	0.2208	0.2717	0.8060

False Negative Cost	Sensitivity	Precision	Accuracy
3.0	0.2580	0.2650	0.7938
3.1	0.2951	0.2506	0.7754
3.2	0.2951	0.2613	0.7822
3.3	0.3101	0.2555	0.7745
3.4	0.3322	0.2575	0.7700
3.5	0.3322	0.2522	0.7661
3.6	0.3419	0.2511	0.7626
3.7	0.3666	0.2509	0.7555
3.8	0.3825	0.2479	0.7484
3.9	0.3993	0.2496	0.7451
4.0	0.4143	0.2465	0.7379
4.1	0.4479	0.2417	0.7230
4.2	0.4549	0.2413	0.7205
4.3	0.4629	0.2396	0.7161
4.4	0.4673	0.2378	0.7126
4.5	0.4664	0.2371	0.7121
4.6	0.4717	0.2379	0.7114
4.7	0.4806	0.2366	0.7071
4.8	0.4841	0.2338	0.7025
4.9	0.4867	0.2329	0.7005
5.0	0.4894	0.2318	0.6982
5.1	0.4912	0.2304	0.6959
5.2	0.5044	0.2299	0.6908

From the previous table we observe that as the cost of False Negative increases, the overall accuracy and the precision of the classifier decreases while its sensitivity increases.



The sensitivity levels we are interested in are 25%, 40% and 50%.

25% Sensitivity

The costs matrix which achieves sensitivity as close as possible to 25% is

Table 6: Cost Matrix for 25% Sensitivity

actual / predicted	Charged Off	Fully Paid
Charged Off	0	3
Fully Paid	1	0

From the previous table, we can see that this specific cost matrix achieves sensitivity of 0.258 and precision of 0.265.

40% Sensitivity

The costs matrix which achieves sensitivity as close as possible to 40% is

Table 7: Cost Matrix for 40% Sensitivity

actual / predicted	Charged Off	Fully Paid
Charged Off	0	3.9
Fully Paid	1	0

From the previous table, we can see that this specific cost matrix achieves sensitivity of 0.3993 and precision of 0.2496.

50% Sensitivity

The costs matrix which achieves sensitivity as close as possible to 50% is

Table 8: Cost Matrix for 50% Sensitivity

actual / predicted	Charged Off	Fully Paid
Charged Off	0	5.2
Fully Paid	1	0

From the previous table, we can see that this specific cost matrix achieves sensitivity of 0.5044 and precision of 0.2299.

Q4: Cost Parameter Matrix for Identifying Dubious Loan Applications

The bank could use the classification tree to predict the loans that are potentially risky. The loan officers can then follow up by double-checking which of those are truly likely to be *"Charged Off"* in future.

To lower the credit risk, we will need to identify as many truly dubious loan applicants as possible. However, we can see in the earlier section, in order to reach 50% sensitivity level, the precision performance is very low. This means that out of all those applications that are highlighted as risky, only around 23% are indeed *"Charged Off"* in the validation set. The drawback in reaching a sensitivity level of close to 50% is that the loan officers will need to manually cross-check large number of applications before 50% of the truly risky loans are identified.

Nevertheless, the cost of granting a loan to risky applicants still far outweighs this labour cost. Hence we pick the cost matrix which achieves sensitivity close to 50%.

Q5: Test Set Performance

Using the cost matrix for 50% sensitivity, we retrain the classification tree using both training and validation data. By retraining the model using both training and validation set, the future performance can be more accurately estimated (compared to the scenario where only the training set is used), because we will eventually train the model on the full sample of data to maximise its performance.

The retrained model is then tested against the test set. Below tabulates the test set performance and confusion matrix.

The final performance of the test set, using the cost matrix that previously resulted in 50% sensitivity, is

Table 9: Test Set Performance

Sensitivity	Precision	Accuracy
0.5166	0.2111	0.6602

The resulting confusion matrix for the test set is

Table 10: Confusion Matrix for Test Set

actual / predicted	Charged Off	Fully Paid	*Total*
Charged Off	778	728	*1506*
Fully Paid	2907	6284	*9191*
Total	*3685*	*7012*	*10697*