BUILDING CLASSIFICATION MODELS FROM IMBALANCED FRAUD DETECTION DATA

Terence Yong Koon Beh[1], Swee Chuan Tan[2], Hwee Theng Yeo[3]
[1]*School of Business, SIM University*
[1]yky2k@yahoo.com, [2]jamestansc@unisim.edu.sg, [3]yeoht01@gmail.com

Abstract

Many real-world data sets exhibit imbalanced class distributions in which almost all instances are assigned to one class and far fewer instances to a smaller, yet usually interesting class. Building classification models from such imbalanced data sets is a relatively new challenge in the machine learning and data mining community because many traditional classification algorithms assume similar proportions of majority and minority classes. When the data is imbalanced, these algorithms generate models that achieve good classification accuracy for the majority class, but poor accuracy for the minority class. This paper reports our experience in applying data balancing techniques to develop a classifier for an imbalanced real-world fraud detection data set. We evaluated the models generated from seven classification algorithms with two simple data balancing techniques. Despite many ideas floating in the literature to tackle the imbalanced issue, our study shows the simplest data balancing technique is all that is required to significantly improve the accuracy in identifying the primary class of interest (i.e., the minority class) in all the seven algorithms tested. Our results also show that precision and recall are useful and effective measures for evaluating models created from artificially balanced data. Hence, we advise data mining practitioners to try simple data balancing first before exploring more sophisticated techniques to tackle the class imbalance problem.

Keywords: Imbalanced data, Machine Learning, Model Evaluation, Performances Measures

1. Introduction

Class distribution, i.e. the proportion of instances belonging to each class in a data set, plays a crucial role in classification. In a typical two-class domain of positive and negative instances, a data set is said to be imbalanced when one class (the majority class) is represented by a large number of negative instances and the other class (the minority class) constitutes only a very small minority of positive instances (Yen & Lee, 2009).

Classification of data with imbalanced class distribution is of significant concern in the data mining and machine learning community as imbalanced data sets are common in many real-world application domains. For example, in detection of fraudulent cases in telephone calls (Fawcett & Provost, 1997) and credit card transactions (Chan, Fan, Prodromidis & Stolfo, 1999), the number of legitimate transactions heavily outnumbers the number of fraudulent transactions. Likewise, in direct marketing (Ling & Li, 1998), most marketing campaigns commonly yield a small percentage of customer response rates of about 1%. Other examples of application domains with intrinsic imbalance include rare medical diagnosis (Witten & Frank, 2000), fault diagnosis (Yang, Tang, Shintemirov & Wu, 2009) and detection of oil spills (Kubat, Holte & Matwin, 1998).

Learning from imbalanced data sets is an important issue in supervised learning. In order to explain the implications of imbalanced learning problem in the real world, we illustrate an example from fraud detection applications. Consider a credit card transaction data set containing cases that correspond to either fraudulent or legitimate transactions. Only 3% of the data set correspond to the fraudulent (minority class) cases and the remaining cases belong to the legitimate (majority class) category. Learning from such intrinsic imbalanced data sets create issues to classification systems, issues that are not revealed when the classifiers work on relatively balanced data sets.

One issue arises since most inductive machine learning algorithms target to maximize the overall accuracy and therefore these systems commonly achieve good classification accuracy for the majority class cases only. However, the class of interest usually tilts towards correct classification of the minority class cases. For example, in detection of fraud application domains, it is more critical to detect transactions that are suspicious and potentially fraudulent more accurately as compared to the legitimate transactions. In the medical industry, wrong classification of a healthy patient as a cancerous patient or vice versa can cause serious and sometimes fatal consequences. In reality, classifiers dealing with imbalanced data sets tend to provide a severely imbalanced degree of accuracy as they usually attain high predictive accuracy over the majority class but poor performance for cases associated with the minority class. As such, it is evident that for application domains with class imbalance problem, we require a classifier that is not only sensitive enough to detect minority class instances, but also specific enough in differentiating the minority from the majority class instances.

Another important issue in learning from imbalanced data sets is evaluating the learning results appropriately. Traditionally, the performance of machine learning algorithms are evaluated using the standard performance metrics such as overall predictive accuracy and error rate. Unfortunately, since the prior probabilities of the positive and negative classes in imbalanced data sets are unequally distributed, predictive accuracy and error rate are therefore inappropriate to evaluate the learning results in such situation (Bharatheesh & Iyengar, 2004). Consider the credit card transaction data set example again. A bank wants to construct and train a classifier using the data set to predict whether a future credit card transaction is fraudulent or legitimate. The number of fraudulent transactions is only 3% of all transactions. A simple default strategy of predicting a transaction as belonging to the legitimate category yields a high accuracy of 97%. Despite the high accuracy, the classifier would not be able to correctly identify any transaction belonging to the fraudulent category within all transactions.

Recent years have seen increased interest in proposing a variety of strategies to address the issues brought by learning from imbalanced data sets. Strategies such as use of appropriate evaluation metrics (Guo, Yin, Dong, Yang & Zhou, 2008), ensemble learning methods (Galar, Fernandez, Barrenechea, Bustince & Herrera, 2011), sampling techniques (Krishnaveni & Rani, 2011) and cost-sensitive learning (Ganganwar, 2012) have been intensively reviewed as well as applied in many of today's real-world application domains with much success.

In this paper, we share our experience in applying training data balancing techniques to create some fraud detection models from an extremely imbalanced data set. We also evaluated several evaluation metrics and identify the one most suitable for our purpose.

The report is structured as follows: Section 2 presents reviews of similar work done in the fields of evaluation metrics and sampling techniques. Section 3 describes the initial assessment of data quality and pre-processing methods. The various modelling methodologies in approaching the data mining project are introduced in Section 4. In Section 5, we present our experiments and experimental results. Finally, a conclusion is provided in Section 6.

2. Related Work

This section reviews two strategies to tackle the class imbalance problem, namely using the correct model evaluation metrics, and training data sampling techniques. Previous research (Weiss & Provost, 2003) has shown it is important to use the right metric(s) to evaluate models when the data is imbalanced. Hence the first part of this section will review the different options available for model evaluation. The second part of the section will review two simple data balancing techniques, namely under-sampling of majority class training instances and over-sampling of minority class training instances. These techniques are considered because they are most appropriate for data mining users and practitioners. These techniques are conceptually simple, easy to implement and require no tweaking of the machine learning algorithms.

A. Evaluation Metrics based on Confusion Matrix

In Monard & Batista's (2003) paper, the authors explained that a confusion matrix summarizes information about actual and predicted classifications performed by a classifier. Table 1 shows a confusion matrix for a typical two-class problem with positive and negative classes.

Table 1. Confusion matrix for a two-class classification task

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Positive represents the minority class and negative represents the majority class. Generally, the minority class is the actual class of interest. True Positive indicates the number of correctly classified positive instances. True Negative indicates the number of correctly classified negative instances. Likewise for False Positive and False Negative, they indicate the number of misclassified positive instances and negative instances respectively. Standard performance metrics such as predictive accuracy and error rate can be derived from the confusion matrix in Table 1.

- Predictive Accuracy = (True Positive + True Negative) / (True Positive + False Positive + True Negative + False Negative)

- Error rate = (False Positive + False Negative) / (True Positive + False Positive + True Negative + False Negative)

In Weiss & Provost's (2003) paper, the authors used predictive accuracy and error rate to evaluate the classification learning results of twenty six data sets and concluded that usage of these metrics lead to poor performance for the minority class. For that reason, a variety of common evaluation metrics based on confusion matrix are developed to assess the performance of classifiers for imbalanced data sets. From the confusion matrix in Table 1, Galar et al. (2011) presented four evaluation metrics, i.e. False Negative Rate, False Positive Rate, True Negative Rate and True Positive Rate.

- False Negative Rate, FNR = False Negative / (True Positive + False Negative)

FNR is the percentage of positive instances misclassified as belonging to the negative class.

- False Positive Rate, FPR = False Positive / (True Negative + False Positive)

FPR is the percentage of negative instances misclassified as belonging to the positive class.

- True Negative Rate, TNR = True Negative / (False Positive + True Negative)

TNR is the percentage of negative instances correctly classified within the negative class.

- True Positive Rate, TPR = True Positive / (False Negative + True Positive)

TPR is the percentage of positive instances correctly classified within the positive class.

In another paper, Nguyen, Bouzerdoum & Phung (2010) introduced three evaluation metrics namely Precision, Recall and F-measure. These metrics are developed from the fields of information retrieval. They are used in situations when performance for the positive class (the minority class) is preferred, since both precision and recall are defined with respect to the positive class.

- Precision = True Positive / (True Positive + False Positive)

Precision is the percentage of positive predictions made by the classifier that are correct.

- Recall = True Positive / (False Negative + True Positive)

Recall is the percentage of true positive instances that are correctly detected by the classifier.

- F-measure = (2 x Recall x Precision) / (Recall + Precision)

F-measure is the harmonic mean of precision and recall (Fawcett, 2006). A high F-measure implies a high value for both precision and recall.

Additionally, Nguyen, Bouzerdoum & Phung (2010) also introduced Sensitivity, Specificity and Geometric mean (G-mean). These evaluation metrics are best utilized in situations when performance for both majority and minority classes are equally important and expected to be high simultaneously. G-mean signifies the balance between the classification performances on the two classes. This metric takes into account the sensitivity (the accuracy on the positive instances) and the specificity (the accuracy on the negative instances).

- Sensitivity = True Positive / (False Negative + True Positive) = Recall or True Positive Rate

- Specificity = True Negative / (False Positive + True Negative) = True Negative Rate

- G-mean = $\sqrt{\ }$ (Sensitivity x Specificity)

These various evaluation metrics share a common feature in that they all exhibit a high degree of independency in the cost for class and prior probabilities. In other words, these metrics are all class-independent measures and therefore they are more appropriate to evaluate the learning results compared to predictive accuracy and error rate.

B. Evaluation Metric based on Receiver Operating Characteristic (ROC)

Alternatively, the Receiver Operating Characteristic (ROC) and the area under the ROC (AUC) can be employed to evaluate the overall classification performance (Nguyen, Bouzerdoum & Phung, 2010). The ROC is a graphical representation that plots the relationship between the benefits (True Positive Rate) and costs (False Positive Rate) as the decision threshold varies. The ROC curve provides evidence that the true positive rate is directly proportional to the false positive rate. To put it simply, as true positive rate increases in the classifier, false positive rate also increases. In addition, the ROC curve facilitates clear visualization comparisons between two or more classifiers over a large span of operating points.

The AUC measure summarizes the performance of the classifier into a single quantitative measure, usually for determining which classifier is more superior. Generally, a better performing classifier has a larger AUC than that of an inferior one.

C. Sampling Techniques

In Liu's (2004) paper, he discussed the use of training data balancing techniques to tackle the class imbalance problem. Sampling can be used to change the number of training records in the majority and minority class, causing a change in the prior probabilities on each of the two classes. The main aim of sampling is to balance the class distribution of the data set. The sampling techniques can be divided into two types of categories, under-sampling of majority class training instances and over-sampling of minority class training instances.

a. Under-sampling

Under-sampling is an efficient technique that seeks to eliminate the majority class instances in the training data. Liu (2004) stated that large reduction in the overall number of records in the training data has brought significant savings in terms of training time and memory. However, as under-sampling eliminates potentially useful majority class instances, there is a possibility that much valuable information is lost during the classification process. Hence, under-sampling should be ideally applied on very large data sets in which there are adequate redundant data to be discarded (Wang, 2008).

In Ganganwar's (2012) paper, the author mentioned random under-sampling as one of the simplest and most frequently used technique. In random under-sampling, instances of the majority class are randomly eliminated until the minority to majority class ratio reaches the desired level. The main drawback is that the type of information in the majority class to be discarded cannot be controlled, particularly those potentially useful information that lies between the decision boundaries of the majority and minority class. Despite its simplicity, empirical studies have shown that random under-sampling outperforms most of the more sophisticated under-sampling techniques (Liu, 2004). As such, random under-sampling is regarded as one of the most effective sampling techniques.

b. Over-sampling

Over-sampling is another sampling technique that seeks to increase the minority class instances in the training data. As explained by Krishnaveni and Rani (2011), the benefit of over-sampling is that valuable information still remains intact during the classification process unlike under-sampling. However, Liu (2004) stressed the drawbacks include longer training time and larger amount of memory needed since the overall size of training data increases tremendously. Wang (2008) further added that over-sampling might create over-fitting problem during the classification process since it replicates existing minority class instances.

Liu (2004) commended the use of simple yet effective random over-sampling technique. Random over-sampling works similarly as random under-sampling to balance the class distribution, except that the minority class instances are now randomly replicated to the new training data. Liu (2004) stressed the importance of randomly selecting the minority class instances to be replicated from the original training data and not from the new training data because failing to do so would cause a bias in the randomness of selection.

3. Data Preprocessing

The imbalanced data set used in this study was previously obtained from EZ-R Stats, LLC, a statistical and audit software provider based in North Carolina, United States (URL: http://ezrstats.com/contact.htm). This data set is patterned closely upon two real transportation fraud schemes where employees in Wake County School submitted fraudulent invoices for school bus and automotive parts. All of the data, including numbers and amounts are strictly fictitious and have been manipulated for academic learning purpose.

The data set consists of 245,901 transaction records which 5,584 records are fraudulent and the remaining 240,317 records are legitimate. It has an imbalance ratio of 1/43 (fraudulent/legitimate) or 2.27% fraudulent samples are contained in the data set. Table 2 displays the data elements and description of the data set.

Table 2: Data elements and description of the data set

Data Type	Variable	Description
Identity Data	Vendor Number	Unique identity of vendor (If first character is a letter, then it is a contractor, otherwise a regular vendor, except that there are a series of valid vendors whose codes start with E1~E3 and also G2).
	Voucher Number	Unique identity of voucher.
	Check Number	Unique identity of check.
	Invoice Number	Unique identity of invoice.
	PO Number	Unique identity of purchase order (Zero indicates no purchase order).
Timestamp Data	Invoice Date	Date of issue for invoice.
	Payment Date	Date of payment made.
	Due Date	Date of payment due.
Transactional Data	Invoice Amount	Invoice transaction amount.
Categorical Data	Fraud Ind	Fraud indicator: Yes or No.

From Table 2, it can be observed that there are five variables belonging to the Identity Data category that uniquely identifies entities. Three variables in the Timestamp Data category contain attributes that are related to date. As for the remaining two variables, one is a continuous variable that denotes the invoice transaction amount and the other is a categorical variable with a 'Yes' or 'No' response. However, we realise that all eight variables in the Identity Data and Timestamp Data categories are inappropriate inputs for data mining because they would produce results that make no sense. For example, *Vendor Number*, *Voucher Number*, *Check Number*, *Invoice Number* and *PO Number* in the Identity Data category represent unique identifications in procurement processes and as such they are unlikely to contain useful data patterns. The same also applies to *Invoice Date*, *Payment Date* and *Due Date*. Hence, it is necessary to perform re-categorization to transform some of these variables into more meaningful variables to produce results that make sense. We will elaborate this in the following section on *Data Transformation*.

A. Data Transformation

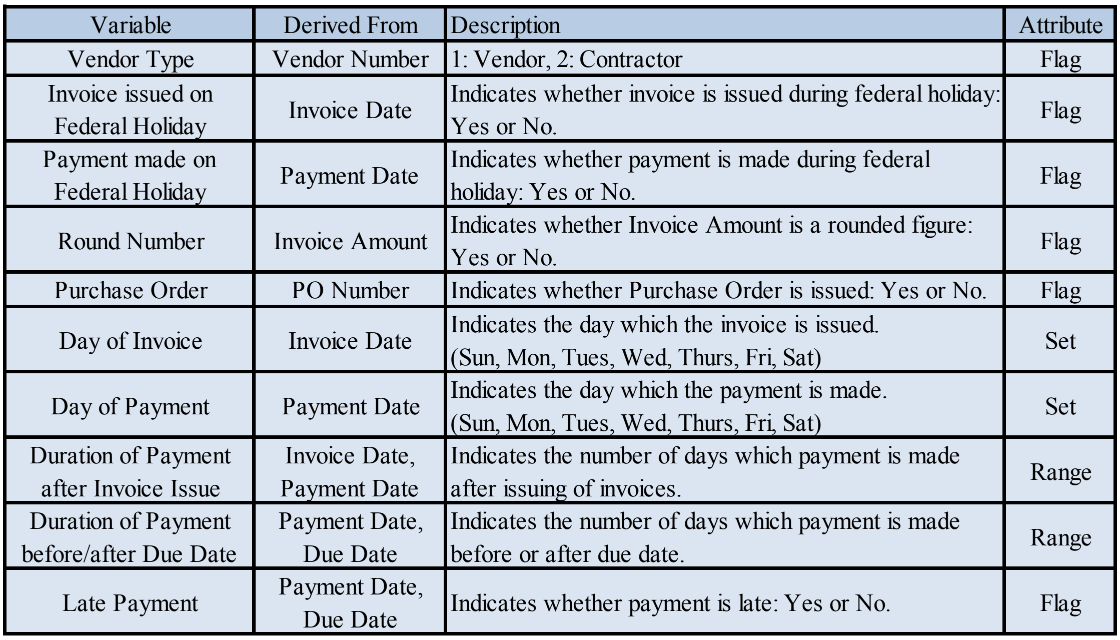
In data mining context, data transformation means the transformation of data into more appropriate forms that can be used for further analysis. For example, a timestamp data expressed as 03-08-2012, it is sometimes more appropriate to work with the data being split into three parts - one variable each for the day, month and year.

In the case of fraud, today's fraudsters continually become more innovative and resourceful in developing new and sophisticated schemes to evade detection. If one is familiar or at least understand how perpetrators go about committing these frauds, new variables can be derived to better improve the accuracy and stability of the fraud classification model. Fraud domain experts have highlighted several potential "red flags" indicators. Red flags are not evidences of fraud but rather signals known about fraudulent situations in which questions should be raised. Some examples of "red flags" indicators are as follows:

- Invoices that are issued on wee hours or non-working days or holidays.
- Payments that are approved and made on wee hours or non-working days or holidays.
- Quick settlement of payments after issuing of invoices.
- Long duration of outstanding payments.
- Amount transacted are rounded figures i.e. no decimal places.

After consideration of the above-mentioned "red flags", a total of ten new variables are derived. Table 3 displays the data elements and description of the newly derived variables.

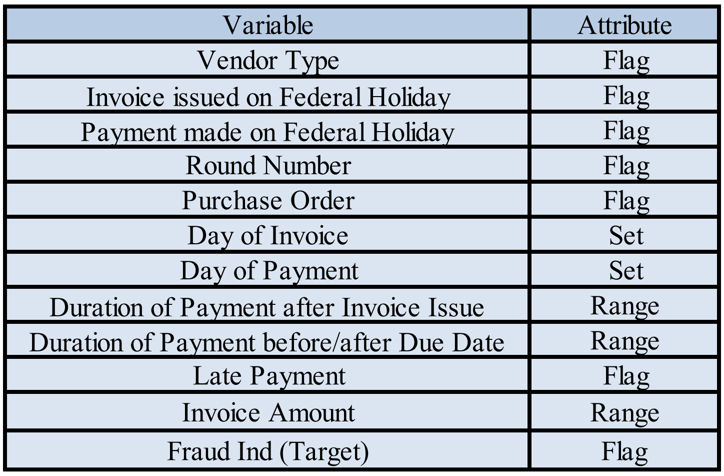
Table 3: Data elements and description of the newly derived variables



Variable	Derived From	Description	Attribute
Vendor Type	Vendor Number	1: Vendor, 2: Contractor	Flag
Invoice issued on Federal Holiday	Invoice Date	Indicates whether invoice is issued during federal holiday: Yes or No.	Flag
Payment made on Federal Holiday	Payment Date	Indicates whether payment is made during federal holiday: Yes or No.	Flag
Round Number	Invoice Amount	Indicates whether Invoice Amount is a rounded figure: Yes or No.	Flag
Purchase Order	PO Number	Indicates whether Purchase Order is issued: Yes or No.	Flag
Day of Invoice	Invoice Date	Indicates the day which the invoice is issued. (Sun, Mon, Tues, Wed, Thurs, Fri, Sat)	Set
Day of Payment	Payment Date	Indicates the day which the payment is made. (Sun, Mon, Tues, Wed, Thurs, Fri, Sat)	Set
Duration of Payment after Invoice Issue	Invoice Date, Payment Date	Indicates the number of days which payment is made after issuing of invoices.	Range
Duration of Payment before/after Due Date	Payment Date, Due Date	Indicates the number of days which payment is made before or after due date.	Range
Late Payment	Payment Date, Due Date	Indicates whether payment is late: Yes or No.	Flag

The derivation of formulas for *Day of Invoice*, *Day of Payment*, *Duration of Payment after Invoice Issue*, *Duration of Payment before/after Due Date* and *Late Payment* are shown in Appendix A. Table 4 shows the final twelve variables that are used as inputs for our predictive modelling.

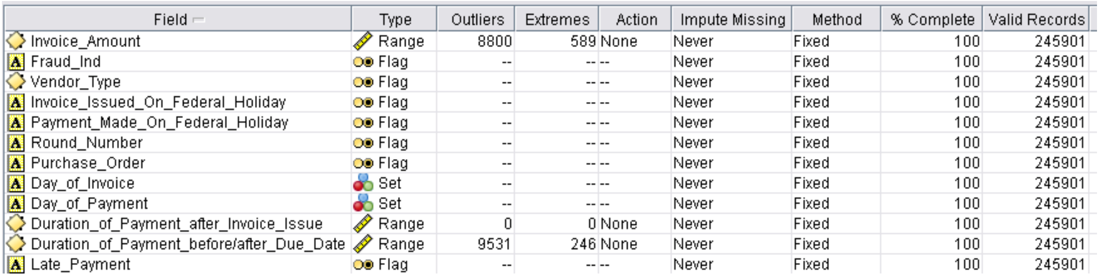
Table 4: The final twelve variables for predictive modelling



Variable	Attribute
Vendor Type	Flag
Invoice issued on Federal Holiday	Flag
Payment made on Federal Holiday	Flag
Round Number	Flag
Purchase Order	Flag
Day of Invoice	Set
Day of Payment	Set
Duration of Payment after Invoice Issue	Range
Duration of Payment before/after Due Date	Range
Late Payment	Flag
Invoice Amount	Range
Fraud Ind (Target)	Flag

B. Data Assessment

The data characteristics such as the data type, outliers, extreme values and missing values of all twelve variables are examined using the Data Audit module in the PASW Modeler 13 (SPSS Inc., (2009)) data mining software. Figure 1 displays the result of the data quality.



Field	Type	Outliers	Extremes	Action	Impute Missing	Method	% Complete	Valid Records
Invoice_Amount	Range	8800	589	None	Never	Fixed	100	245901
Fraud_Ind	Flag	--	--	--	Never	Fixed	100	245901
Vendor_Type	Flag	--	--	--	Never	Fixed	100	245901
Invoice_Issued_On_Federal_Holiday	Flag	--	--	--	Never	Fixed	100	245901
Payment_Made_On_Federal_Holiday	Flag	--	--	--	Never	Fixed	100	245901
Round_Number	Flag	--	--	--	Never	Fixed	100	245901
Purchase_Order	Flag	--	--	--	Never	Fixed	100	245901
Day_of_Invoice	Set	--	--	--	Never	Fixed	100	245901
Day_of_Payment	Set	--	--	--	Never	Fixed	100	245901
Duration_of_Payment_after_Invoice_Issue	Range	0	0	None	Never	Fixed	100	245901
Duration_of_Payment_before/after_Due_Date	Range	9531	246	None	Never	Fixed	100	245901
Late_Payment	Flag	--	--	--	Never	Fixed	100	245901

Figure 1. Data quality of the data set

From Figure 1, it can be seen that the data set is of good quality. All of the fields are 100% complete with no missing values, though *Invoice Amount* and *Duration of Payment before/after Due Date* contain some outliers and extreme values. *Invoice Amount* contains 8,800 outliers (about 3.58%) and 589 extreme values (about 0.24%). *Duration of Payment before/after Due Date* contains 9,531 outliers (about 3.88%) and 246 extreme values (about 0.1%).

As we observed none of the twelve variables contain any missing value, we shifted our attention to the outliers in *Invoice Amount* and *Duration of Payment before/after Due Date*. Outliers and missing values are inevitable in data mining. The countermeasures for dealing with outliers usually require us to either transform or remove them during the data preparation stage. However, outliers in detection of fraud application domains might represent abnormal transaction records that are fraudulent and therefore we shall leave these outliers in *Invoice Amount* and *Duration of Payment before/after Due Date* untouched for further analysis.

4. Methods

This section presents an overview of our modelling approach in investigating the effects of adapting random under-sampling and random over-sampling techniques to a variety of machine learning algorithms for class imbalance learning.

A. Modelling Framework

Figure 2 shows the overall concept of the model evaluation framework for the project execution. The imbalanced data set is first partitioned into 70% training and 30% testing data. Next, it involves training the various classification models with the training data and subsequently applies the trained models to classify the remaining and unseen testing data.

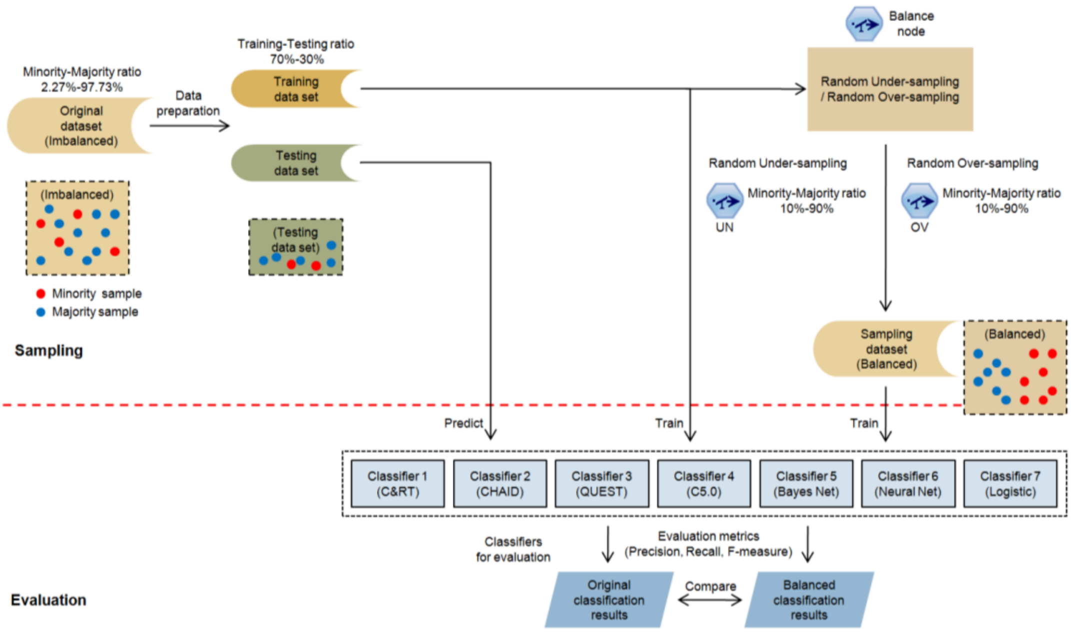

Balance node
Training-Testing ratio 70%-30%
Minority-Majority ratio 2.27%-97.73%
Original dataset (Imbalanced)
Data preparation
Training data set
Testing data set
Random Under-sampling / Random Over-sampling
Random Under-sampling
Random Over-sampling
Minority-Majority ratio 10%-90%
UN
Minority-Majority ratio 10%-90%
OV
(Imbalanced)
(Testing data set)
● Minority sample
● Majority sample
Sampling
Sampling dataset (Balanced)
(Balanced)
Predict
Train
Train
Classifier 1 (C&RT)
Classifier 2 (CHAID)
Classifier 3 (QUEST)
Classifier 4 (C5.0)
Classifier 5 (Bayes Net)
Classifier 6 (Neural Net)
Classifier 7 (Logistic)
Classifiers for evaluation
Evaluation metrics (Precision, Recall, F-measure)
Evaluation
Original classification results
Compare
Balanced classification results

Figure 2. Classification model evaluation framework overview

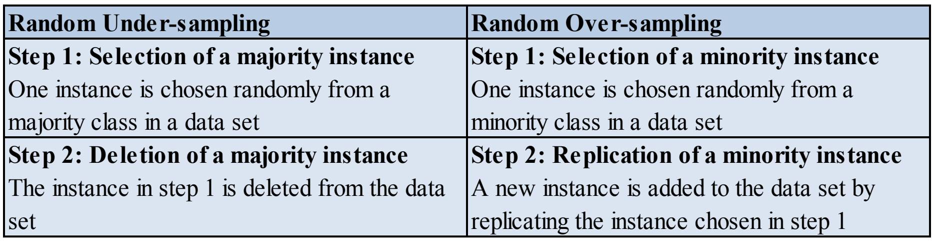
In this paper, we make use of the Balance Node module available in the PASW Modeler 13 data mining software to vary the uneven class distribution in the training data. The Balance Node module is an easy approach for performing random under-sampling and random over-sampling by eliminating legitimate transactions and replicating fraudulent transactions respectively based on specified balancing directives.

Each directive comprises of a factor and condition that instructs the balancing algorithm to increase or decrease the proportion of transactions by the factor specified when the condition holds true. Random under-sampling uses a factor lower than 1.0 to decrease the number of legitimate transactions whereas random over-sampling increases the number of fraudulent transactions with a factor higher than 1.0. Consider the training data with a minority-majority ratio of 2.28%-97.72%. In order to achieve i.e. a minority-majority ratio of 10%-90% with random under-sampling, we impose a balancing directive with a factor of 0.20987 (correct to 5 decimal places) and a condition *Fraud Ind* = "No". This means the number of legitimate transactions in the training data is reduced to 20.987% for all downstream operations.

B. Random Sampling Techniques

To recap, random under-sampling reduces the number of majority instances by eliminating the majority instances randomly whereas random over-sampling increases the number of minority instances by replicating the minority instances randomly. Table 5 displays the steps involved in the two random sampling techniques.

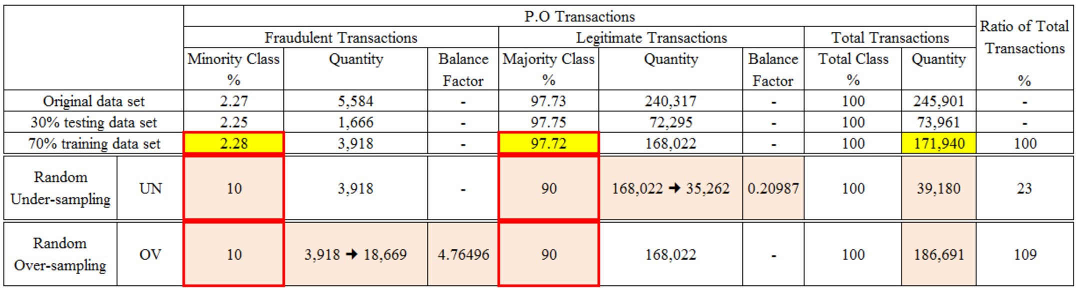
Table 5: Two-step process in random under-sampling and random over-sampling



Random Under-sampling	Random Over-sampling
Step 1: Selection of a majority instance One instance is chosen randomly from a majority class in a data set	**Step 1: Selection of a minority instance** One instance is chosen randomly from a minority class in a data set
Step 2: Deletion of a majority instance The instance in step 1 is deleted from the data set	**Step 2: Replication of a minority instance** A new instance is added to the data set by replicating the instance chosen in step 1

As illustrated in Table 5, both random sampling techniques repeat the two-step process until a predefined minority-majority ratio is achieved, i.e. 20%-80%. Since both techniques have the abilities to increase and decrease the number of instances to the desired minority-majority ratio, the predictive performance based on different minority-majority ratio can be evaluated. As such, we experimentally determine another combination of class distribution (minority-majority ratio of 10%-90%) for each random sampling technique, as shown in Table 6. We then compare these results to that of the original training data with minority-majority ratio of 2.28%-97.72%. The purpose here is to find out whether predictive performance on the minority class improves as we vary the uneven class distribution in the training data to a more balanced one. In Kamei, Monden, Matsumoto, Kakimoto & Matsumoto's (2007) paper, the authors mentioned that correction of class imbalance distribution in the data set would result in an improvement in the predictive performance on the minority class.

Table 6: One combination of class distribution for each random sampling technique



		P.O Transactions								Ratio of Total Transactions %
		Fraudulent Transactions			Legitimate Transactions			Total Transactions		
		Minority Class %	Quantity	Balance Factor	Majority Class %	Quantity	Balance Factor	Total Class %	Quantity	
Original data set		2.27	5,584	-	97.73	240,317	-	100	245,901	-
30% testing data set		2.25	1,666	-	97.75	72,295	-	100	73,961	-
70% training data set		2.28	3,918	-	97.72	168,022	-	100	171,940	100
Random Under-sampling	UN	10	3,918	-	90	168,022 ➔ 35,262	0.20987	100	39,180	23
Random Over-sampling	OV	10	3,918 ➔ 18,669	4.76496	90	168,022	-	100	186,691	109

C. *Predictive Modelling*

Predictive modelling is the prediction of future values or variables based on past historical data as inputs. The target variable *Fraud Ind* is considered as non-metric since it contains two discrete categories "Yes" and "No". Once a predictive model deals with a non-metric target, it is also known as a classifier or classification model.

In this aspect, four decision tree algorithms namely Classification and Regression Tree (C&RT) (Breiman, Fridman, Olshen & Stone, 1984), Chi-squared Automated Interaction Detector (CHAID) (Kass, 1980), Quick Unbiased Efficient Statistical Tree (QUEST) (Loh & Shih, 1997) and C5.0 (Quilan, 1996) are chosen because of their abilities to handle metric and non-metric inputs. Other state of the art machine learning algorithms such as Bayesian Network (Friedman, Geiger & Goldszmidt, 1997), Neural

Networks (McCulloch & Pitts, 1943) and Logistic Regression (Maranzato, Pereira, Neubert & Dolago, 2010) are also performed on the same training data.

D. Evaluation Metrics

In this paper, we evaluate the predictive performances of all seven machine learning algorithms by predictive accuracy, precision and recall which are based on True Negative (TN), False Negative (FN), True Positive (TP) and False Positive (FP). For a binary classification problem like this study, the elements of the confusion matrix are shown in Table 7. The confusion matrix provides the full picture in a model's ability to correctly predict or separate the legitimate and fraudulent transactions.

Table 7: Elements of confusion matrix in the project

Transactions		Predicted	
		Legitimate (-)	Fraudulent (+)
Actual	Legitimate (-)	True Negative	False Positive
	Fraudulent (+)	False Negative	True Positive

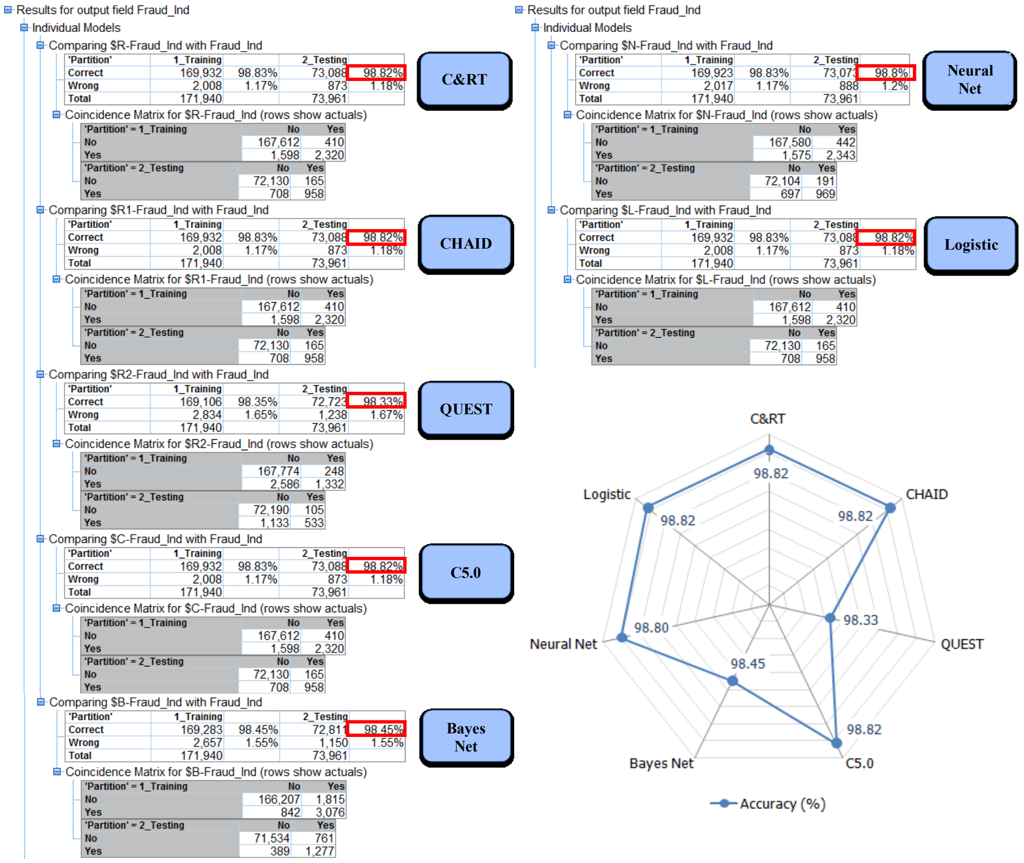
Precision, recall and predictive accuracy are very common measures in binary classifications. Precision is defined as the percentage of positive predictions made by the classifier that are correct and calculated by $\frac{TP}{(TP+FP)}$. On the other hand, recall is defined as the percentage of true positive instances that are correctly detected by the classifier and calculated by $\frac{TP}{(TP+FN)}$. Lastly, the predictive accuracy of the classifier is calculated by $\frac{TP+TN}{(TP+FP+TN+FN)}$. Since the positive class (fraudulent transactions) is the major concern in detection of fraud application domains, precision and recall are appropriate measures of performance as both metrics are defined with respect to the positive class and well-understood in such situation.

5. Modelling and Results

As mentioned in Section 4, seven machine learning algorithms were tested on their classification abilities. The data set was partitioned into 70% training and 30% testing data and modelling were carried out to test for their out-sample predictive accuracy.

A. Modelling with Original Data

Figure 3 shows the out-sample predictive evaluation of all seven algorithms on the original data (minority-majority ratio 2.28%-97.72%).



Results for output field Fraud_Ind

Individual Models

Comparing $R-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,932	98.83%	73,088	98.82%
Wrong	2,008	1.17%	873	1.18%
Total	171,940		73,961	

Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,612	410
Yes	1,598	2,320
'Partition' = 2_Testing	**No**	**Yes**
No	72,130	165
Yes	708	958

Comparing $R1-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,932	98.83%	73,088	98.82%
Wrong	2,008	1.17%	873	1.18%
Total	171,940		73,961	

Coincidence Matrix for $R1-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,612	410
Yes	1,598	2,320
'Partition' = 2_Testing	**No**	**Yes**
No	72,130	165
Yes	708	958

Comparing $R2-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,106	98.35%	72,723	98.33%
Wrong	2,834	1.65%	1,238	1.67%
Total	171,940		73,961	

Coincidence Matrix for $R2-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,774	248
Yes	2,586	1,332
'Partition' = 2_Testing	**No**	**Yes**
No	72,190	105
Yes	1,133	533

Comparing $C-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,932	98.83%	73,088	98.82%
Wrong	2,008	1.17%	873	1.18%
Total	171,940		73,961	

Coincidence Matrix for $C-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,612	410
Yes	1,598	2,320
'Partition' = 2_Testing	**No**	**Yes**
No	72,130	165
Yes	708	958

Comparing $B-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,283	98.45%	72,811	98.45%
Wrong	2,657	1.55%	1,150	1.55%
Total	171,940		73,961	

Coincidence Matrix for $B-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	166,207	1,815
Yes	842	3,076
'Partition' = 2_Testing	**No**	**Yes**
No	71,534	761
Yes	389	1,277

Results for output field Fraud_Ind

Individual Models

Comparing $N-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,923	98.83%	73,073	98.8%
Wrong	2,017	1.17%	888	1.2%
Total	171,940		73,961	

Coincidence Matrix for $N-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,580	442
Yes	1,575	2,343
'Partition' = 2_Testing	**No**	**Yes**
No	72,104	191
Yes	697	969

Comparing $L-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,932	98.83%	73,088	98.82%
Wrong	2,008	1.17%	873	1.18%
Total	171,940		73,961	

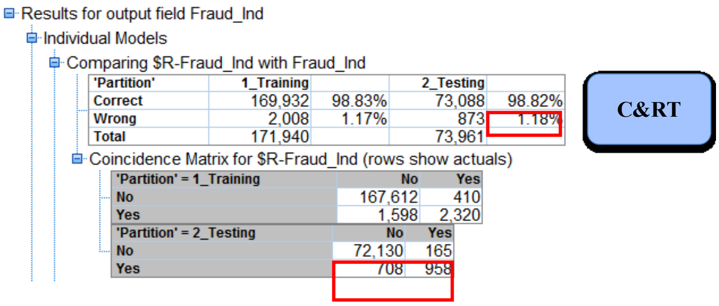
Coincidence Matrix for $L-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,612	410
Yes	1,598	2,320
'Partition' = 2_Testing	**No**	**Yes**
No	72,130	165
Yes	708	958

Figure 3. Out-sample predictive evaluation

From Figure 3, it is observed that all seven algorithms achieve excellent results in terms of predictive accuracy (98.35% ~ 98.83%) in classifying the training data. Similarly for the out-sample predictive accuracy, they performed equally well in classifying the "unseen" data (98.33% ~ 98.82%), as seen from the radar chart in Figure 3. However, we have learnt that predictive accuracy is not a proper measure of performance for detection of fraud application domain despite the good results. This is because the machine learning algorithms typically achieve good results in detecting legitimate transactions but not fraudulent transactions.

As detailed in the section on *Evaluation Metrics based on Confusion Matrix*, predictive accuracy takes into account the total number of correctly classified positive (True Positive) instances and correctly classified negative (True Negative) instances. We use the classification result from the C&RT algorithm in Figure 4 to illustrate our point.



Results for output field Fraud_Ind
Individual Models
Comparing $R-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	169,932	98.83%	73,088	98.82%
Wrong	2,008	1.17%	873	1.18%
Total	171,940		73,961	

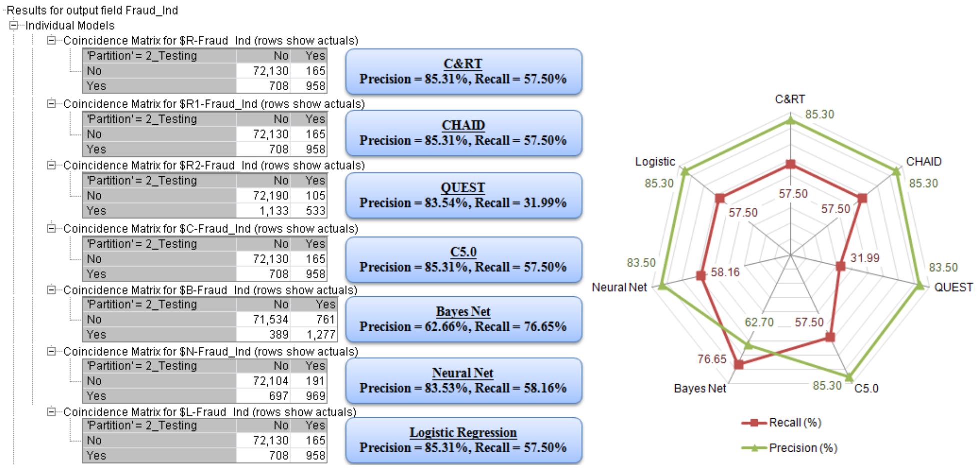
Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

'Partition' = 1_Training	No	Yes
No	167,612	410
Yes	1,598	2,320
'Partition' = 2_Testing	No	Yes
No	72,130	165
Yes	708	958

Figure 4. Confusion matrix result of C&RT

The "Partition=2_Testing" confusion matrix result shows that the C&RT algorithm has classified 72,130 True Negative and 958 True Positive instances correctly and thus achieves an accuracy of 98.82%. Taken at face value, 98.82% accuracy across the entire data set indeed appears outstanding. Yet, this description fails to reveal the fact that the C&RT algorithm is inept at identifying fraudulent transactions within all transactions, as only 958 out of 1,666 fraudulent transactions are classified correctly. Similar phenomena are observed for CHAID, QUEST, C5.0, Bayes Net, Neural Net and Logistic Regression algorithms as well.

From the confusion matrix, we next investigate the predictive performances of all seven algorithms in terms of precision and recall. The results measured in percentages are shown in Figure 5.



Results for output field Fraud_Ind
Individual Models

Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,130	165
Yes	708	958

C&RT Precision = 85.31%, Recall = 57.50%

Coincidence Matrix for $R1-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,130	165
Yes	708	958

CHAID Precision = 85.31%, Recall = 57.50%

Coincidence Matrix for $R2-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,190	105
Yes	1,133	533

QUEST Precision = 83.54%, Recall = 31.99%

Coincidence Matrix for $C-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,130	165
Yes	708	958

C5.0 Precision = 85.31%, Recall = 57.50%

Coincidence Matrix for $B-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,534	761
Yes	389	1,277

Bayes Net Precision = 62.66%, Recall = 76.65%

Coincidence Matrix for $N-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,104	191
Yes	697	969

Neural Net Precision = 83.53%, Recall = 58.16%

Coincidence Matrix for $L-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	72,130	165
Yes	708	958

Logistic Regression Precision = 85.31%, Recall = 57.50%

Figure 5. Classification results - precision and recall (original data)

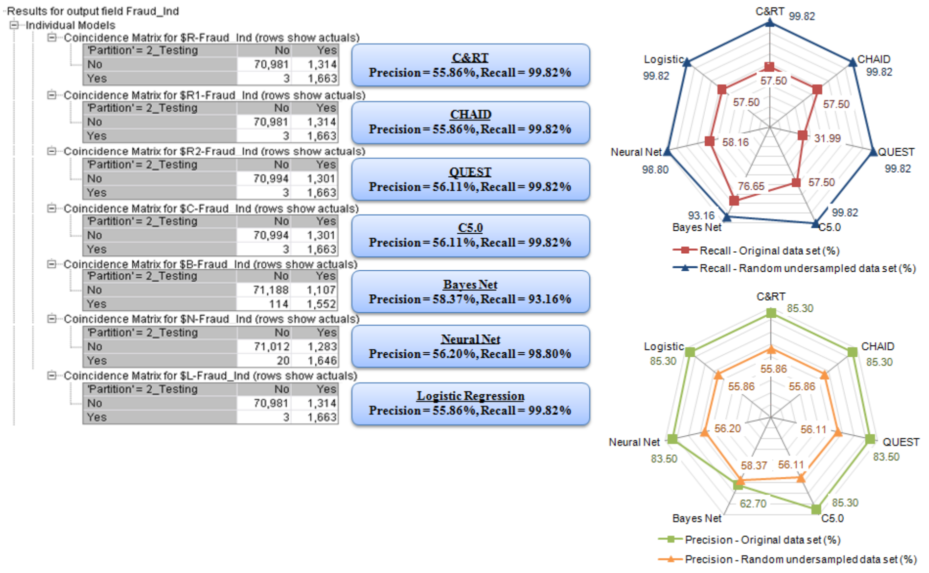
As per priori expectations, all seven algorithms show mixed performances with respect to recall. The average recall is about 57% which means that the algorithms managed to correctly classify slightly more than half of the actual fraudulent transactions as indeed fraudulent. It appears that Bayes Net yield the highest recall (76.65%) among the seven algorithms, followed by Neural Net (58.16%), C&RT, CHAID and C5.0 (57.50%). QUEST is the worst performing algorithm with the lowest recall (31.99%).

In terms of precision, most of the algorithms except Bayes Net achieve good results. The best performing algorithms are C&RT, CHAID, C5.0 and Logistic Regression. Each of them is capable of making 85.31% of positive (fraudulent) predictions that are correct. QUEST and Neural Net lose out slightly with 83.54% and 83.53% precision respectively. Bayes Net has the lowest precision (62.66%) despite having the highest recall.

All these results clearly reflect the importance of using precision and recall other than predictive accuracy to evaluate the learning results of machine learning algorithms. Interpreting the results with wrong measures certainly distort the actual performance of the classifiers and might cause serious consequences from poor decision making.

B. *Modelling with Random Under-sampled Data*

As detailed in the section on *Random Sampling Techniques*, correction of class imbalance distribution in the data set may improve the predictive performance on the minority class. In view of this, we performed another two modelling experiments with the intention of finding out whether predictive performance on detection of fraudulent transactions improves. The first model uses the random under-sampled data and the other model utilizes the random over-sampled data (to be discussed in the next section on *Modelling with Random Over-sampled Data*). Figure 6 shows the out-sample predictive performances of all seven algorithms performed on the random under-sampled data (minority-majority ratio 10%-90%) in terms of precision and recall, measured in percentages.



Results for output field Fraud_Ind

Individual Models

Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,981	1,314
Yes	3	1,663

C&RT
Precision = 55.86%, Recall = 99.82%

Coincidence Matrix for $R1-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,981	1,314
Yes	3	1,663

CHAID
Precision = 55.86%, Recall = 99.82%

Coincidence Matrix for $R2-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,994	1,301
Yes	3	1,663

QUEST
Precision = 56.11%, Recall = 99.82%

Coincidence Matrix for $C-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,994	1,301
Yes	3	1,663

C5.0
Precision = 56.11%, Recall = 99.82%

Coincidence Matrix for $B-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,188	1,107
Yes	114	1,552

Bayes Net
Precision = 58.37%, Recall = 93.16%

Coincidence Matrix for $N-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,012	1,283
Yes	20	1,646

Neural Net
Precision = 56.20%, Recall = 98.80%

Coincidence Matrix for $L-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,981	1,314
Yes	3	1,663

Logistic Regression
Precision = 55.86%, Recall = 99.82%

Figure 6. Classification results - precision and recall (random under-sampled data)

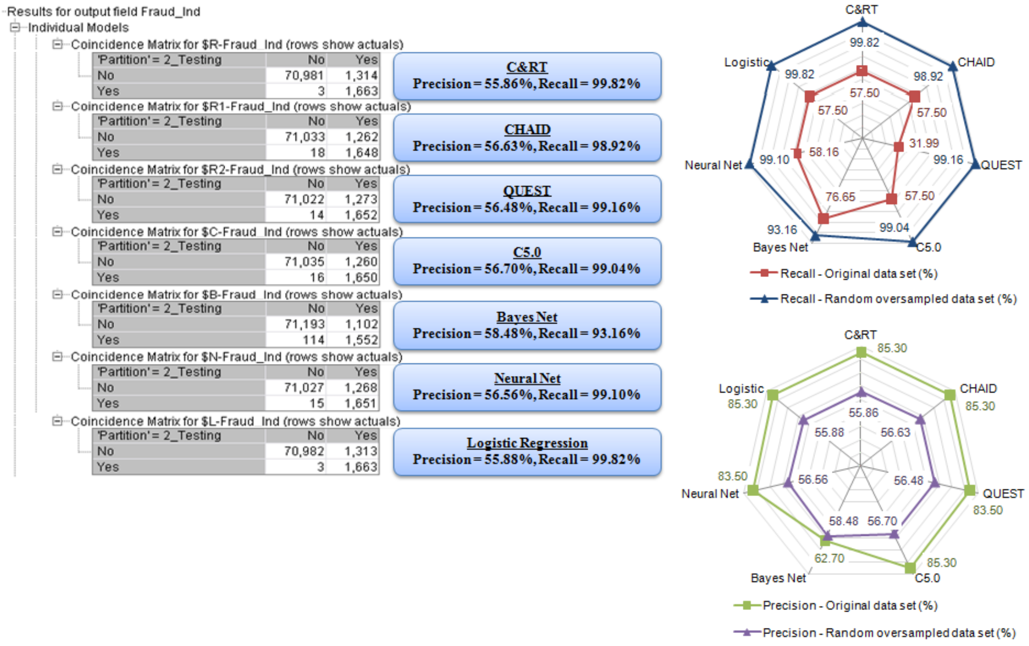
From the top radar chart in Figure 6, it is observed that there are significant improvements over recall for all seven algorithms as we increase the minority class percentage from 2.28% to 10% with the random under-sampling technique. The best performers are the four decision trees algorithms, C&RT, CHAID, QUEST, C5.0 and together with Logistic Regression. Impressively, these five algorithms are able to classify almost all fraudulent transactions as fraudulent correctly. Bayes Net yields the lowest improvement in performance since the algorithm is only able to correctly classify 93.16%

of the actual fraudulent transactions as fraudulent despite having the highest recall (76.65%) when building the predictive model with the original imbalanced data.

In terms of precision, we observed that C&RT, CHAID, QUEST, C5.0, Neural Net and Logistic Regression (bottom radar chart) saw declines around 32% ~ 35% in precision (average of 84.7% to 56%). Surprisingly, Bayes Net only saw a decline around 7% in precision, from 62.7% to 58.37%. These results clearly indicate that there is a trade off between precision and recall. The trade off between precision and recall is straightforward; an increase in precision can lower recall while an increase in recall lowers precision. Here, as we attempt to build predictive models that utilize the random under-sampling technique, recall improves at the cost of precision.

C. Modelling with Random Over-sampled Data

Figure 7 shows the out-sample predictive performances of all seven algorithms performed on the random over-sampled data (minority-majority ratio 10%-90%) in terms of precision and recall, measured in percentages.



Results for output field Fraud_Ind

Individual Models

Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,981	1,314
Yes	3	1,663

C&RT
Precision = 55.86%, Recall = 99.82%

Coincidence Matrix for $R1-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,033	1,262
Yes	18	1,648

CHAID
Precision = 56.63%, Recall = 98.92%

Coincidence Matrix for $R2-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,022	1,273
Yes	14	1,652

QUEST
Precision = 56.48%, Recall = 99.16%

Coincidence Matrix for $C-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,035	1,260
Yes	16	1,650

C5.0
Precision = 56.70%, Recall = 99.04%

Coincidence Matrix for $B-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,193	1,102
Yes	114	1,552

Bayes Net
Precision = 58.48%, Recall = 93.16%

Coincidence Matrix for $N-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	71,027	1,268
Yes	15	1,651

Neural Net
Precision = 56.56%, Recall = 99.10%

Coincidence Matrix for $L-Fraud_Ind (rows show actuals)

'Partition' = 2_Testing	No	Yes
No	70,982	1,313
Yes	3	1,663

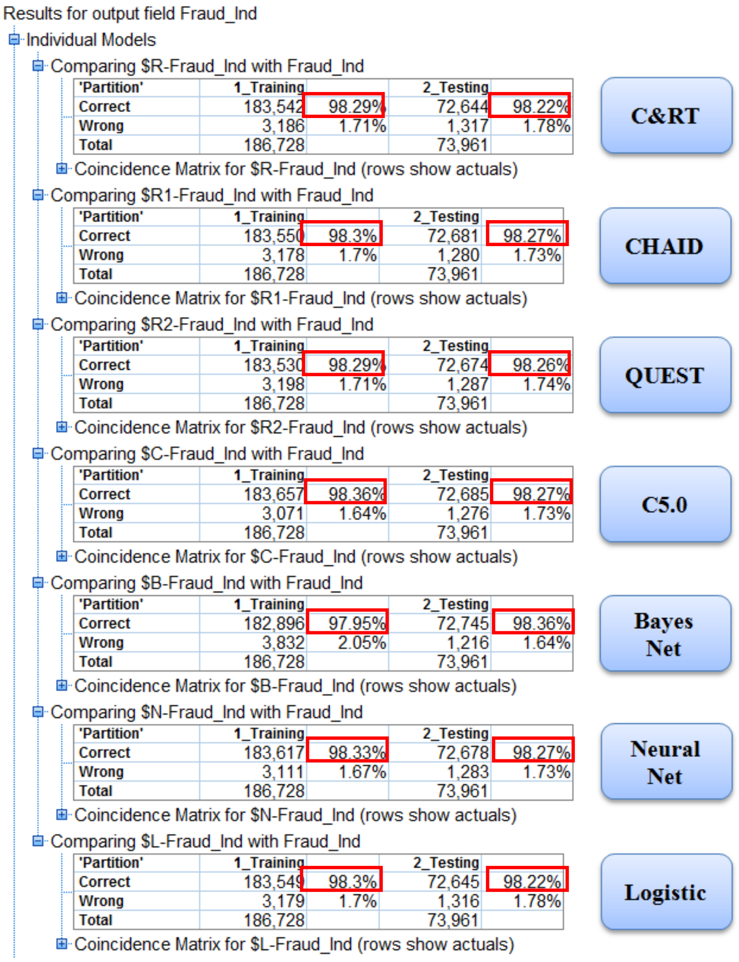
Logistic Regression
Precision = 55.88%, Recall = 99.82%

Figure 7. Classification results - precision and recall (random over-sampled data)

The top radar chart in Figure 7 shows similar results as compared to that of predictive modelling with the random under-sampled data. All seven algorithms also yield significant increase in recall as we increase the minority class percentage from 2.28% to 10% with the random over-sampling technique. The best performers are C&RT and Logistic Regression algorithms with both recall value of 99.82%, followed by QUEST (99.16%), Neural Net (99.10%), C5.0 (99.04%) and CHAID (98.92%). Bayes Net has the lowest performance with a recall value 93.16%.

Likewise for precision, predictive modelling with the random over-sampled data produces similar results as compared to its counterpart. All algorithms except Bayes Net saw declines around 32% ~ 35% in precision (average of 84.7% to 56.4%). Bayes Net saw a decline around 7% in precision, from 62.70% to 58.48%. The trade-off between precision and recall also indicates that recall improves at the cost of precision when building predictive models with the random over-sampling technique.

As mentioned earlier, the drawback of over-sampling technique is that it might cause over-fitting problem during the classification process as it replicates many existing minority class instances. One useful rule of thumb to tell a model is over-fitting is when the predictive performance on its own training set is much better than on its testing set. From Figure 8, all seven algorithms have identical predictive performances on both training and testing set, and as such we could not find concrete evidences that suggest the presence of over-fitting problem in our predictive modelling with the random over-sampling technique.



Results for output field Fraud_Ind

Individual Models

Comparing $R-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,542	98.29%	72,644	98.22%
Wrong	3,186	1.71%	1,317	1.78%
Total	186,728		73,961	

Coincidence Matrix for $R-Fraud_Ind (rows show actuals)

Comparing $R1-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,550	98.3%	72,681	98.27%
Wrong	3,178	1.7%	1,280	1.73%
Total	186,728		73,961	

Coincidence Matrix for $R1-Fraud_Ind (rows show actuals)

Comparing $R2-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,530	98.29%	72,674	98.26%
Wrong	3,198	1.71%	1,287	1.74%
Total	186,728		73,961	

Coincidence Matrix for $R2-Fraud_Ind (rows show actuals)

Comparing $C-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,657	98.36%	72,685	98.27%
Wrong	3,071	1.64%	1,276	1.73%
Total	186,728		73,961	

Coincidence Matrix for $C-Fraud_Ind (rows show actuals)

Comparing $B-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	182,896	97.95%	72,745	98.36%
Wrong	3,832	2.05%	1,216	1.64%
Total	186,728		73,961	

Coincidence Matrix for $B-Fraud_Ind (rows show actuals)

Comparing $N-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,617	98.33%	72,678	98.27%
Wrong	3,111	1.67%	1,283	1.73%
Total	186,728		73,961	

Coincidence Matrix for $N-Fraud_Ind (rows show actuals)

Comparing $L-Fraud_Ind with Fraud_Ind

'Partition'	1_Training		2_Testing	
Correct	183,549	98.3%	72,645	98.22%
Wrong	3,179	1.7%	1,316	1.78%
Total	186,728		73,961	

Coincidence Matrix for $L-Fraud_Ind (rows show actuals)

Figure 8. Out-sample predictive evaluation (random over-sampled data)

6. Conclusion

In this paper, we have presented our experience in exploring the application of evaluation metrics to an extremely imbalanced data set. With the intention of addressing the inappropriateness of predictive accuracy as measure of performance, we exploited a total of seven machine learning algorithms for our predictive modelling experiments. Using the experimental results that are obtained from the various predictive models, we have demonstrated the inappropriateness of predictive accuracy in evaluating the learning results. In learning from imbalanced data, predictive accuracy can be misleading because it causes us to favour high prediction accuracy on the legitimate transactions (usually uninteresting class) but not the fraudulent transactions (usually interesting class). In order to address this issue, we have used precision and recall to examine how data balancing alter the predictive performance of minority class instances and how it affects the classifier ability in differentiating the two classes of data.

Since our paper is an example in detection of fraud application domains, it is critical that we detect the fraudulent transactions more accurately than the legitimate transactions. The results of predictive modelling with the original imbalanced data have yielded a low to moderate recall and high precision on the minority class (fraudulent transactions). In order to improve the prediction performance on the minority class, we have adapted random under-sampling and random over-sampling techniques into all seven algorithms. Although these are very simple methods, all the models surprising shown significant improvements in the predictive performance for detection of fraudulent transactions, in which, we attained very good recall without much compromise on the precision.

It is hard to justify why one should not use these simple yet effective training data balancing techniques, unless they result in models that do not surpass that of learners using more sophisticated strategies. We further conclude that such techniques are appealing to use as the only change required is to the training data itself and not to the machining learning algorithms.

In practice, the kind of detection results generated by our models can be used to generate a preliminary first-cut list of suspicious transactions to be investigated. Further scrutinisation would be required to assess whether a transaction is worthwhile to investigate further. This usually involves assessing the investigation costs involved, the consequence of not investigating the case, the amount of money involved, and likelihood of fraud.

References

Bharatheesh, T.L. & Iyengar, S.S. (2004). Predictive Data Mining for Delinquency Modeling. Retrieved 18 July, 2012 from http://www.csc.lsu.edu/~iyengar/final-papers/Predictive%20data%20mining%20for%20delinquency%20mo.pdf

Breiman, L., Fridman, J., Olshen, R. & Stone, C.J. (1984). *Classification and Regression Tree*. Pacific California: Wadsworth & Brooks / Cole Advanced Books & Software

Chan, P.H., Fan, W., Prodromidis, A. & Stolfo, S. (1999). Distributed Data Mining in Credit Card Fraud Detection. *IEEE Intelligent Systems on Data Mining*

Das, B., Krishnan, N. C., & Cook, D. J. (2014). Handling imbalanced and overlapping classes in smart environments prompting dataset. *In Data Mining for Service*, 199-219

Fawcett, T. (2006). An Introduction to ROC Analysis. *Pattern Recognition Letters*, *27*, 861-874

Fawcett, T. & Provost, F. (1997). Adaptive Fraud Detection. *Data Mining and Knowledge Discovery*, *1* (3), 291-316

Friedman, N., Geiger, D. & Goldszmidt, M. (1997). Bayesian Network Classifiers. *Machine Learning*, *29*, 131-163

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H. & Herrera, F. (2011). A Review On Ensembles For The Class Imbalance Problem: Bagging, Boosting and Hybrid-based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, 1-22. doi: 10.1109/TSMCC.2011.2161285

Ganganwar, V. (2012). An Overview of Classification Algorithms for Imbalanced Datasets. Retrieved 19 July, 2012 from http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf

Guo, X.J., Yin, Y.L., Dong, C.L., Yang, G.P. & Zhou, G.T. (2008). On The Class Imbalance Problem. *2008 Fourth International Conference on Natural Computation*, *4*, 192-201. doi: 10.1109/ICNC.2008.871

Kamei, Y., Monden, A., Matsumoto, S., Kakimoto, T. & Matsumoto, K. (2007). The Effects of Over and Under Sampling on Fault-prone Module Detection. Retrieved 08 August, 2012 from http://se.naist.jp/achieve/pdf/259.pdf

Kass, G.V. (1980). An Exploratory Technique for Investigating Large Quantities of Categorical Data. *Applied Statistics*, *29* (2), 119-127

Krishnaveni, C.V. & Rani, T.S. (2011). On the Classification of Imbalanced Datasets. *IJCST, 2* (SP1), 145-148

Kubat, M., Holte, R. & Matwin, S. (1998). Machine Learning for the Detection of Oil Spills in Radar Images. *Machine Learning*, *30*, 195-215

Ling, C. & Li, C. (1998). Data Mining for Direct Marketing Problems and Solutions. *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, New York, U.S.A.

Liu, Y.C. (2004). The Effect of Oversampling and Undersampling on Classifying Imbalanced Text Datasets. Retrieved 28 July, 2012 from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.5878&rep=rep1&type=pdf

Loh, W.Y. & Shih, Y.S. (1997). Split Selection Methods for Classification Trees. *Statistical Sinica*, *7*, 815-840

López, V., Triguero, I., Carmona, C. J., García, S., & Herrera, F. (2013). Addressing imbalanced classification with instance generation techniques: IPADE-ID. *Neurocomputing*, 126, 15-28.

Maranzato, R., Pereira, A., Neubert, M. & Dolago, A.P. (2010). Fraud Detection in Reputation Systems in e-Markets using Logistic Regression. *Proceedings of 2010 ACM Symposium on Applied Computing*, Sierre, Switzerland

McCulloch, W.S. & Pitts, W. (1943). A Logical Calculus of Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, *5*, 115-133

Monard, M.C. & Batista, G.E.A.P.A. (2003). Learning with Skewed Class Distributions. Retrieved 18 July, 2012 from http://www.icmc.usp.br/~gbatista/files/laptec2002.pdf

Nguyen, G.H., Bouzerdoum, A. & Phung, S.L. (2010). Learning Pattern Classification Tasks with Imbalanced Data Sets. Retrieved 26 July, 2012 from http://cdn.intechweb.org/pdfs/9154.pdf

PASW Modeler 13 [Software]. (2009). Chicago: SPSS Inc

Quilan, J.R. (1996). Improved Use of Continuous Attributes in C4.5. *Journal of Artificial Intelligence Research*, 477-490

Wang, S. (2008). Class Imbalance Learning. Retrieved 28 July, 2012 from http://www.cs.bham.ac.uk/~syw/documents/progress_reports/Thesis%20Proposal%20(ShuoWang).pdf

Weiss, G.M. & Provost, F. (2003). Learning When Training Data Are Costly: The Effect of Class Distribution on Tree Induction. *Journal of Artificial Intelligence Research*, *19*, 315-354

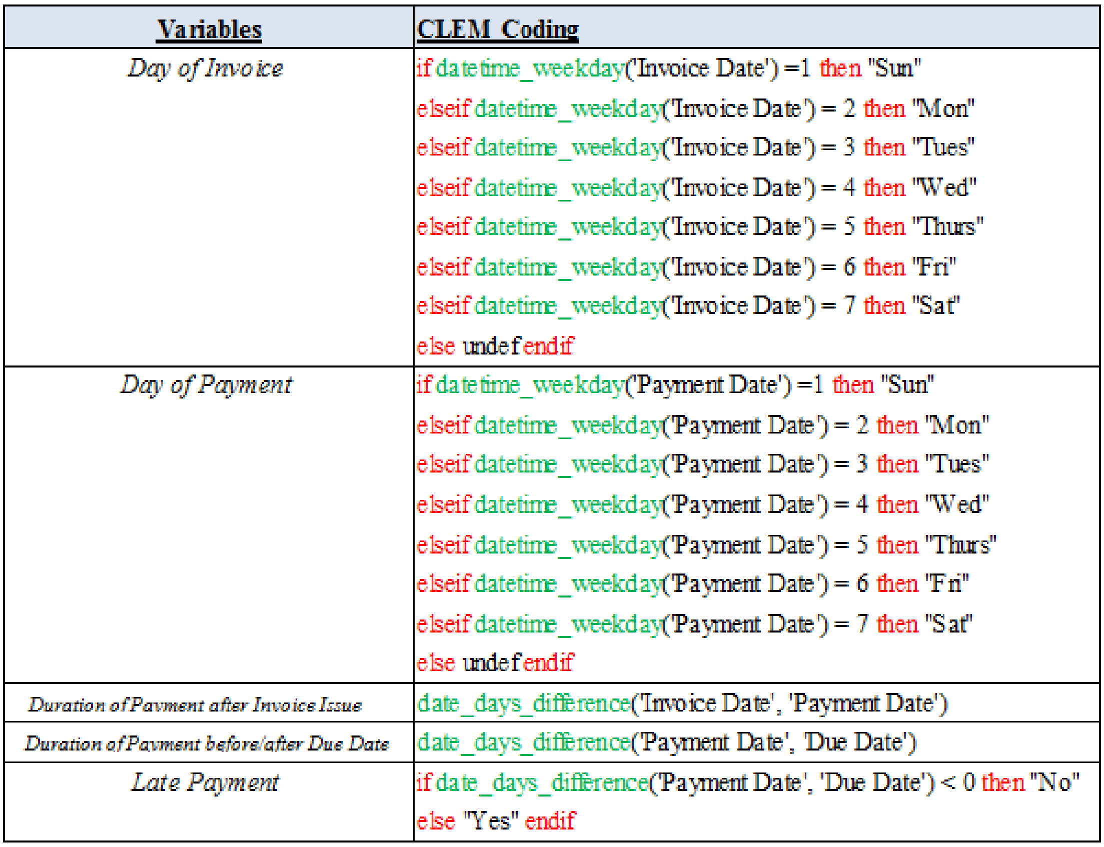
Witten, I. & Frank, E. (2000). Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. California, U.S.A.

Yang, Z., Tang, W., Shintemirov, A. & Wu, Q. (2009). Association Rule Mining Based Dissolved Gas Analysis for Fault Diagnosis of Power Transformers. *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, *39* (6), 597-610

Yen, S.J. & Lee, Y.S. (2009). Cluster-based Under-sampling Approaches for Imbalanced Data Distributions. *Expert Systems with Applications*, *36*, 5718-5727. doi: 10.1016/j.eswa.2008.06.108

Appendices

A. CLEM Coding for Data Preparation – Derivation of formula for new variables



Variables	CLEM Coding
Day of Invoice	if datetime_weekday('Invoice Date') =1 then "Sun" elseif datetime_weekday('Invoice Date') = 2 then "Mon" elseif datetime_weekday('Invoice Date') = 3 then "Tues" elseif datetime_weekday('Invoice Date') = 4 then "Wed" elseif datetime_weekday('Invoice Date') = 5 then "Thurs" elseif datetime_weekday('Invoice Date') = 6 then "Fri" elseif datetime_weekday('Invoice Date') = 7 then "Sat" else undef endif
Day of Payment	if datetime_weekday('Payment Date') =1 then "Sun" elseif datetime_weekday('Payment Date') = 2 then "Mon" elseif datetime_weekday('Payment Date') = 3 then "Tues" elseif datetime_weekday('Payment Date') = 4 then "Wed" elseif datetime_weekday('Payment Date') = 5 then "Thurs" elseif datetime_weekday('Payment Date') = 6 then "Fri" elseif datetime_weekday('Payment Date') = 7 then "Sat" else undef endif
Duration of Payment after Invoice Issue	date_days_difference('Invoice Date', 'Payment Date')
Duration of Payment before/after Due Date	date_days_difference('Payment Date', 'Due Date')

Late Payment	if date_days_difference('Payment Date', 'Due Date') < 0 then "No"
	else "Yes" endif