Rare Class Discovery Techniques for Highly Imbalanced Data

Kittipong Chomboon*, Kittisak Kerdprasop, and Nittaya Kerdprasop

Abstract—**This research aims at proposing a method to induce a classification model of minority data cases that are always predominant by a much larger majority cases. Our proposed method applies feature selection technique to choose 25% of attributes that show highly correlation between classes. Then use over-sampling technique on minority cases before making a classification. We have found from the experimental results that data of rare cases, which is normally disappeared, can be detected through our proposed method. In this research, we use R language for implementing our proposed method and other four discovery techniques.**

Index Terms—**Data Mining, Rare Class, Imbalanced Data, Data Classification, R Language.**

I. Introduction

Present computer technology has progressed at a very rapid speed and it has tremendous effect to the storage of electronic data. With enormously increasing data, conventional method to analyze data is inappropriate. Data mining is thus an essential technology and has been proven useful for automatic analysis of large data [5], [7]. Data mining is the discovery of interesting patterns from large data. Discovered patterns can be in various forms such as a tree model for classification, a set of rules for association, or a group of representative centroids for data clustering. In this paper, we focus on the discovery of a tree model. This model can be used to predict events in the future.

Decision tree induction is normally a powerful technique to discover a tree model for future event prediction. But for a highly imbalanced data in which the number of data instances in one class is extremely smaller than the number of instances in other classes. Dataset of a smaller group is called a rare class [2], [6] or a minority class. A dataset with high imbalance between majority and minority classes can cause much trouble to the tree induction algorithm. During the tree building process data instances from a minority class are normally pruned and disappear from the final tree model. This makes the tree model predict the majority class correctly, but minority class tends to be incorrectly predicted. This is a challenging problem known as a rare class mining.

In this research, we comparatively study five data preparation techniques to help decision tree induction algorithm creating a model that can predict rare class data. The various data preparation techniques include clustering, over-sampling, correlation analysis, outlier detection, and a mixture of correlation and outlier analyses. The experimentation has been performed on the RStudio environment and the program is implemented with the R language. Source code of our implementation is available in the Appendix.

Manuscript received December 5, 2012; revised January 10, 2013. This work was supported in part by grant from Suranaree University of Technology through the funding of Data Engineering Research Unit.

K. Chomboon is a master student with the School of Computer Engineering, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand (e-mail: chomboon.k@gmail.com).

K. Kerdprasop is an associate professor with the School of Computer Engineering, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand.

N. Kerdprasop is an associate professor with the School of Computer Engineering, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand.

II. Rare class classification

Rare class classification is the data mining task aiming at building a model that can correctly classify both the majority and minority classes. Classifying minority or rare class is difficult because size of the rare class is too small. Many researchers have tried to solve this problem. Alhammady and Ramanohanarao [1] proposed an algorithm called EPDT (Emerging Pattern Decision Tree) to train a decision tree that can classify rare class.

He and Ghodsi [3] proposed a classification algorithm based on the SVM (Support Vector Machine) for classifying rare objects. Wu et al. [8] proposed a technique called COG (Classification using lOcal clusterinG) that also based on the SVM classifier. McCarthy et al. [4] proposed the algorithm called Cost-Sensitive and compared the algorithm with the under-sampling and over-sampling techniques.

Over-sampling and clustering techniques as applied in several work seem to give a good result. We then perform a comparative study by applying these techniques together with the correlation and outlier detection methods. These techniques are applied in the data preparation step, whereas the tree induction is our classification method. The research methodology and the experimental results are explained in the following sections.

III. Methodology

In this section we present the discovery process of rare class on imbalanced dataset. We use SECOM dataset from the UCI (http://archive.ics.uci.edu/ml/datasets/SECOM) SECOM is data from a semi-conductor manufacturing process. It is highly-skewed. The dataset contains 1567 data instances taken from a wafer fabrication production line. Each data instance contains 590 sensor measurements. The data are labeled as -1 (means pass the test), or 1 (means fail the test). There are only 104 fail cases, whereas the pass case are 1463. This is a 1:14 proportion. The goal of SECOM dataset is a good or bad semi-conductor from manufacturing process. The overview our techniques show as Fig. 1.

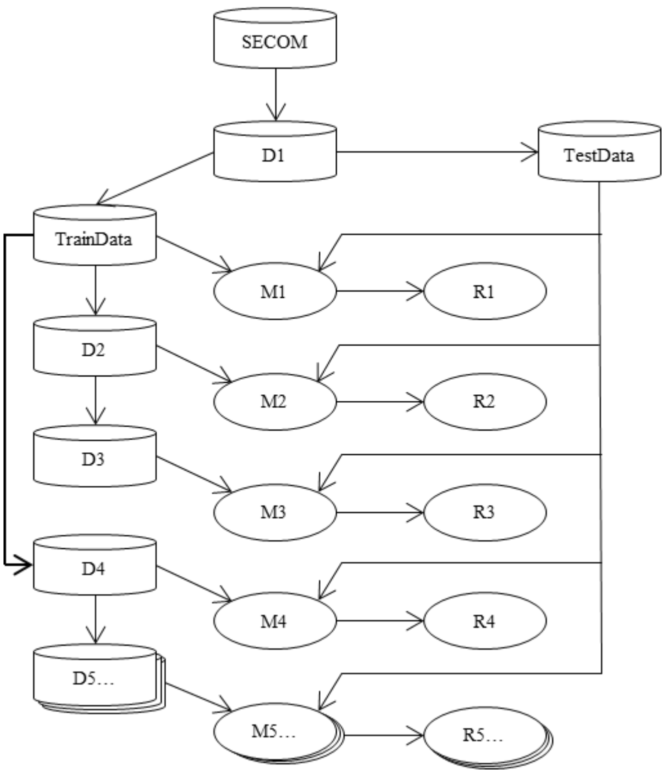


Fig. 1 The overview of five techniques to discover rare class

Then we will describe these techniques step by step. Step one: we manage missing values in SECOM dataset by replacing with median value and create a new dataset called D1, graphically show in Fig. 2.

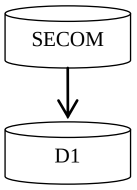


Fig. 2 Step 1: manage missing value

Step 2: we split D1 into train and test data. TrainData and TestData has ratio 70:30. Then create model from train data and evaluate accuracy with TestData, shown in Fig. 3.

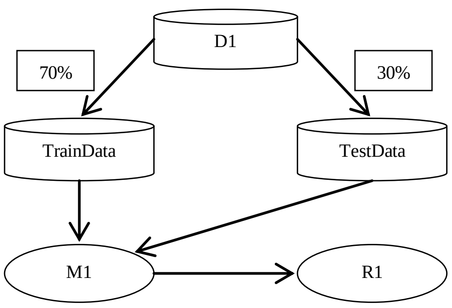


Fig. 3 Step 2: evaluate accuracy of Traindata.

Step 3: find top 25% of correlation between the class attribute and other attributes of TrainData then union with outliers. Outliers are found from each attribute and include all outliers into the TrainData. Create new dataset called D2 with correlated feature selection and outliers inclusion. Then create model form D2 and examine accuracy with TestData, show as Fig. 4.

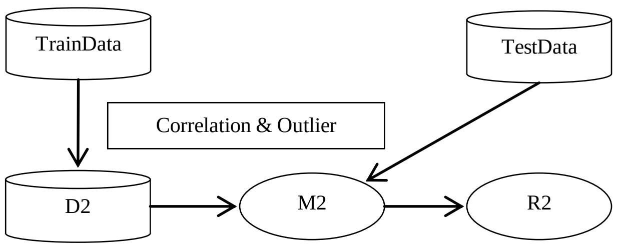


Fig. 4 Step 3: feature selection by correlation analysis and inclusion of outliers

Step 4: apply over-sampling technique on the minority class (that is, the fail cases) D2 dataset. The new dataset is called D3. Then create model from D3 and evaluate accuracy with TestData, shown in Fig. 5.

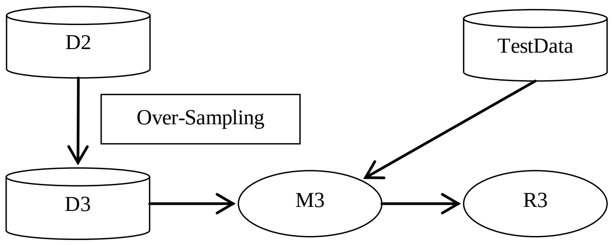


Fig. 5 Step 4: oversampling the fail cases

Step 5: find top 25% of correlation between the class attribute and other attributes of TrainData, then create new dataset with selected features and call this dataset D4. After that create a model from D4 and evaluate its accuracy with TestData, as shown in Fig. 6.

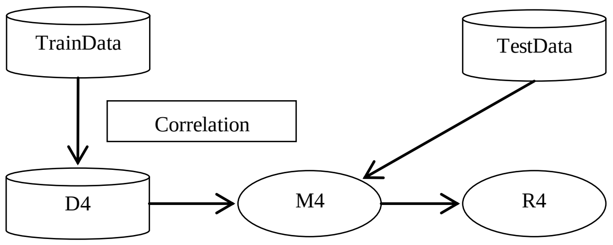


Fig. 6 Step 5: feature selection with correlation analysis

Step 6: clustering dataset D4 with the pamk function available in the RStudio program. Function pamk returns a group of two clusters. Then create model of each cluster and evaluate accuracy of each model with TestData, shown in Fig. 7.

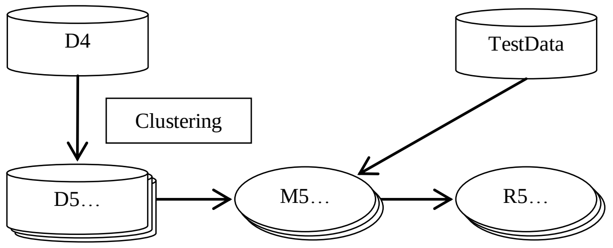


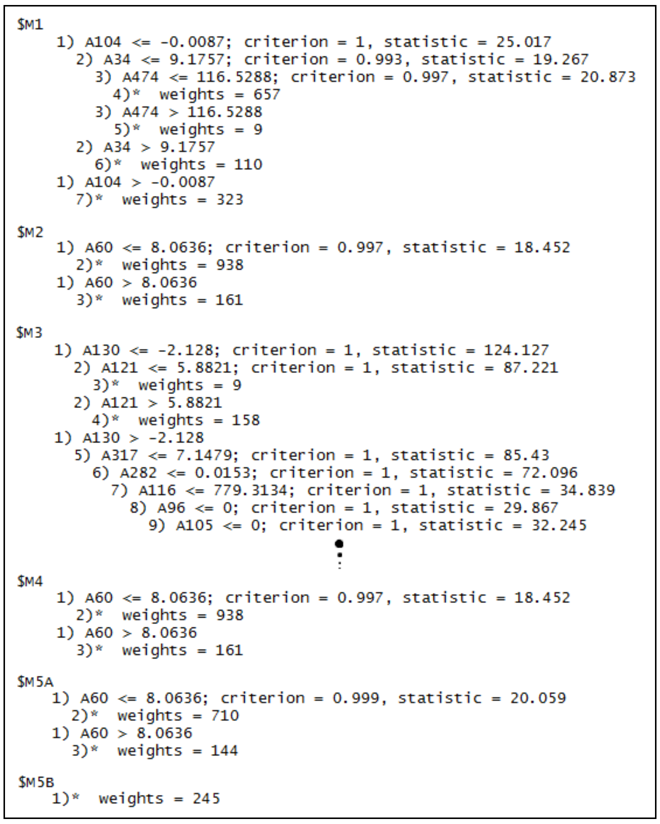
Fig. 7 Step 6: cluster data then create model from each cluster

After design techniques, we implemented by using decision tree algorithm for classification with R language by RStudio. RStudio is an open source program and easy to use as there are several statistical and data mining functions available in the library.

IV. Experimental Results

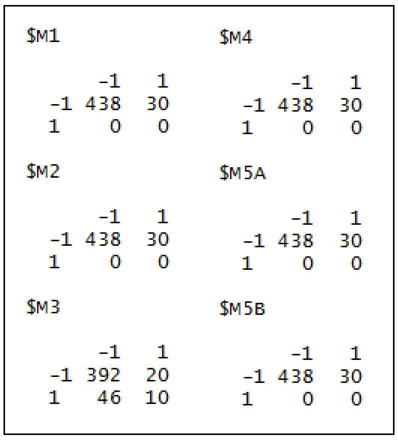
This research experimentation used SECOM (semi-conductor manufacturing process) dataset from UCI Machine Learning Repository. SECOM dataset has 591 attributes and 1567 data instances.

For rare class discovery technique comparison, we used decision tree algorithm for classification on imbalanced dataset and used R language coding on RStudio program. The classification models of the designed techniques are shown in Fig. 8. Evaluation results of the 5 models (M1, M2, M3, M4, M5) can be displayed as a confusion matrix as shown in Fig. 9.



```
$M1
    1) A104 <= -0.0087; criterion = 1, statistic = 25.017
      2) A34 <= 9.1757; criterion = 0.993, statistic = 19.267
        3) A474 <= 116.5288; criterion = 0.997, statistic = 20.873
          4)*  weights = 657
        3) A474 > 116.5288
          5)*  weights = 9
      2) A34 > 9.1757
        6)*  weights = 110
    1) A104 > -0.0087
      7)*  weights = 323

$M2
    1) A60 <= 8.0636; criterion = 0.997, statistic = 18.452
      2)*  weights = 938
    1) A60 > 8.0636
      3)*  weights = 161

$M3
    1) A130 <= -2.128; criterion = 1, statistic = 124.127
      2) A121 <= 5.8821; criterion = 1, statistic = 87.221
        3)*  weights = 9
      2) A121 > 5.8821
        4)*  weights = 158
    1) A130 > -2.128
      5) A317 <= 7.1479; criterion = 1, statistic = 85.43
        6) A282 <= 0.0153; criterion = 1, statistic = 72.096
          7) A116 <= 779.3134; criterion = 1, statistic = 34.839
            8) A96 <= 0; criterion = 1, statistic = 29.867
              9) A105 <= 0; criterion = 1, statistic = 32.245
                                   ⋮

$M4
    1) A60 <= 8.0636; criterion = 0.997, statistic = 18.452
      2)*  weights = 938
    1) A60 > 8.0636
      3)*  weights = 161

$M5A
    1) A60 <= 8.0636; criterion = 0.999, statistic = 20.059
      2)*  weights = 710
    1) A60 > 8.0636
      3)*  weights = 144

$M5B
    1)*  weights = 245
```

Fig. 8 Classification models of the designed techniques



```
$M1                    $M4

      -1   1                 -1   1
  -1 438  30             -1 438  30
  1    0   0             1    0   0

$M2                    $M5A

      -1   1                 -1   1
  -1 438  30             -1 438  30
  1    0   0             1    0   0

$M3                    $M5B

      -1   1                 -1   1
  -1 392  20             -1 438  30
  1   46  10             1    0   0
```

Fig. 9 Results of classification model evaluation

A comparative performance of each model can be summarized and shown in Table 1. It can be seen from the result that model M3 that has been built from the top 25% correlated attributes and data in rare class have been over-sampling shows the best rare class detection rate of 10 from the total 30 rare data instances. Other techniques have zero rare class detection rate.

TABLE 1
Comparative Results of Designed Techniques

Model	Accuracy	Rare class, detection rate	Recall	Precision
M1	0.93	0	0.93	1
M2	0.93	0	0.93	1
M3	0.85	0.33	0.95	0.89
M4	0.93	0	0.93	1
M5	0.93	0	0.93	1
M6	0.93	0	0.93	1

V. Conclusion

This research aims to study rare class discovery techniques for highly imbalance data using decision tree algorithm as a classification method. The results show that the best model is model 3 that used correlation-based feature selection, outlier and over-sampling techniques to prepare data for classification. Model 3 has rare class detection rate at 33%, whereas other models cannot detect rare class.

References

[1] H. Alhammady, and K. Ramamohanarao. (2004). "Using Emerging Patterns and Decision Trees in Rareclass Classification," *In Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM '04)*, pp.315-318.

[2] N.V. Chawla. (2005). "Data Mining for Imbalanced Datasets: An Overview," *The Data Mining and Knowledge Discovery Handbook*, pp.853-867.

[3] H. He, and A. Ghodsi. (2010). "Rare Class Classification by Support Vector Machine," *In Proceedings of the 20th International Conference on Pattern Recognition*, pp.548-551.

[4] K. McCarthy, B. Zabar, and G. Weiss (2005). "Does Cost-sensitive Learning Beat Sampling for Classifying Rare Classes?" *In Proceedings of the 1st International Workshop on Utilitybased Data Mining*, pp.69–77.

[5] K. Thearling (2012). "An Introduction to Data Mining," Retrieved November 3, 2012, from http://www.thearling.com/text/dmwhite/dmwhite.htm

[6] G. Weiss. (2004). “Mining With Rarity: A Unifying Framework,” *ACM-SIGKDD Explorations Vol. 6 Issue 1*, pp.7-19.
[7] Wikipedia, the free encyclopedia (2012). “Data Mining [Online],” Available URL: http://en.wikipedia.org/wiki/Data_mining
[8] J. Wu, H. Xiong, P. Wu, and J. Chen (2007). “Local Decomposition for Rare Class Analysis,” *In Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD)*, pp.814–823.

Appendix

Source code of R language for comparing rare class discovery techniques is given below. Start the program by calling “my.fn” function.

```
my.fn <- function(data.f, seed=0){
  library("party")
  library("DMwR")
  library("fpc")
  ## initial variable
  stats <- list()
  model <- list()
  result <- list()
  outlier <- vector()
  cluster <- list()
  n.col <- ncol(data.f)
  decition.name <- names(secom)[n.col]
  my.formula <- formula(paste(decition.name, "~."))

  ## initial data
  data.f[,n.col] <- factor(data.f[,n.col])
  for(i in 1:(n.col-1)){
    data.f[is.na(data.f[,i]),i] <- median(data.f[,i], na.rm=T)
  }

  ## split data
  if(seed == 0){
    seed <- sample(1000:9999,1)
  }
  set.seed(seed)
  ind <- sample(2, nrow(data.f), replace=T, prob=(c(0.7,0.3)))
  data.train <- data.f[ind==1,]
  data.test <- data.f[ind==2,]
  print(paste("$$$$ SetSeed ==",seed,"$$$$"))
  print("Split Data Complete")

  ## test for M1 model
  model$M1 <- ctree(my.formula, data=data.train)
  result$M1 <- table(predict(model$M1, newdata=data.test),
data.test[,n.col])
  print("Model M1 Complete")

  ## find outlier by boxplot.stats()
  for(i in 1:(n.col-1)){
    outlier <- union(outlier,which(data.train[,i] %in%
boxplot.stats(data.train[,i])$out))
  }
  outlier <- sort(outlier)

  ## find corelation by cor and select 25%
  cor.sample <- sample(2, ncol(data.train), replace=T,
prob=(c(0.25,0.75)))
  cor.n <- length(cor.sample[cor.sample==1])
  cor.tmp <- cor(data.train[,-n.col],
as.numeric(levels(data.train[,n.col]))[as.numeric(data.train[,n.col]
)])
  cor.name <- dimnames(cor.tmp)[1][[1]][sort(cor.tmp,
decreasing=T, index.return=T)$ix[1:cor.n]]

  ## create data1 from outlier and corelation then create
model&test to M2
  data1 <- data.train[outlier,
c(cor.name,names(data.train)[n.col])]
  model$M2 <- ctree(my.formula, data=data1)
  result$M2 <- table(predict(model$M2, newdata=data.test),
data.test[,n.col])
  print("Model M2 Complete")

  ## create data2 form data1 by decition(A591) -1 == 1 then
create model&test to M3
  data2 <- data1
  data2.n <- nrow(data2)
  data2.bad.rownames <-
rownames(data2[data2[,decition.name]==1,])
  data2.bad.length <- length(data2.bad.rownames)
  data2.inc <- data2.n-(data2.bad.length*2)
  data2.sample <- sample(data2.bad.rownames, data2.inc,
replace=T)
  data2[(data2.n+1):(data2.n+data2.inc),] <- data2[data2.sample,]
  model$M3 <- ctree(my.formula, data=data2)
  result$M3 <- table(predict(model$M3, newdata=data.test),
data.test[,n.col])
  print("Model M3 Complete")

  ## create data3 form data1 by corelation then create
model&test to M4
  data3 <- data.train[,c(cor.name,names(data.train)[n.col])]
  model$M4 <- ctree(my.formula, data=data3)
  result$M4 <- table(predict(model$M4, newdata=data.test),
data.test[,n.col])
  print("Model M4 Complete")

  ## find k form data3
  train.pam <- pamk(data3[,-ncol(data3)])
  for(i in 1:train.pam$nc){
    cluster.name <- paste("c", i, sep="")
    model.name <- paste("M5", rawToChar(as.raw(i+64)),
sep="")
    cluster[[cluster.name]] <-
data3[train.pam$pamobject$clustering==i,]
    model[[model.name]] <- ctree(my.formula,
data=cluster[[cluster.name]])
    result[[model.name]] <- table(predict(model[[model.name]],
newdata=data.test), data.test[,n.col])
    print(paste("Model",paste("M5", rawToChar(as.raw(i+64)),
sep=""),"Complete"))
  }
  print("#### Complete ####")

  stats$seed <- seed
  stats$data <- list(data=data.f, data.train=data.train,
data.test=data.test, data1=data1, data2=data2, data3=data3)
  stats$cluster <- cluster
  stats$model <- model
  stats$result <- result
  stats

}
```