A BEST WAY OF MULTI-LEVEL IMBALANCE RESEARCH DATA HANDLING USING R PROGRAMMING

Yagyanath Rimal
Faculty of Science and Technology
Pokhara University, Pokhara, Nepal

ABSTRACT: **This article discusses clearly reviewing the best way to imbalance data management analysis using R. programming, although there is a big difference between machine learning algorithms, algorithms tend to shake in front of data sets that the unbalanced classification misleading predictions and misleading precisions of the research data in the research project. Its main purpose is to explain the research data sampling method handling unbalance commissioner using the R software, whose results have been sufficiently explained with different intermediate results and graphical interpretation to accurately predict the categorical data prediction. Therefore, this paper presents the simplest way to summarize, when data sets with a large unbalanced force categorical and for the analysis of programming data in R.**

KEYWORDS: Multilevel imbalance, upscaling, down sampling, Random Forest, Cost-sensitive

INTRODUCTION

A data set is not balanced if the classification categories are not represented in approximately the same way. Unbalanced search data is a big problem when data in some categories goes beyond the different categories. Therefore, the researcher could not make precise forecasts (Manivannan, 2017). The unbalanced data refer to classification problems in which the class data substantially exceed the other categories with a substantial percentage. Unbalanced classification is a supervised learning problem in which one class far exceeds other class categories. This problem is more frequent in binary classification problems than in the analysis of classification data at several problematic levels (EVA, 2015). The unbalanced classification occurs more frequently in the binary classification than in the multilevel classification. With an unbalanced data set, the information needed to accurately forecast the minority class cannot be obtained from an algorithm. Therefore, it is recommended to use a balanced classification data set (Pozzolo, 2015). Recent years have increased the interest in applying machine learning techniques to difficult problems, many of which are characterized by unbalanced data. Most real-world classification problems show a certain level of class imbalance, that is, when each class is not an equal part of our research dataset.

The classic problem of data imbalance is recognized as most problems in the field of data extraction and machine learning, since most machine learning algorithms assume that data is equally distributed in all areas of the database. In the case of unbalanced data, the majority classes dominate the minority classes, which makes the machine learning classifications more inclined towards the majority classes. For example, suppose you have two categories A and B. Class A is 90% of your data set and class B is the remaining 10%, but you're more interested in identifying class B instances. You can get accuracy of 90% simply provides a class each time, but this provides a classifier not necessary for the intended use case. Conversely, a properly regulated method can achieve less accuracy, but would have a substantially higher positive rate, which has been optimized so that these scenarios often occur in the context of detection, such as online offensive content or bookmarks. diseases in the data doctors, 2016). What causes a bad classification of minority classes in the database. The classifiers can even predict all the test data as main classes. Some of the real examples include the detection of oil spills, the detection of network outages, the detection of fraud and rare diseases. The accuracy of the classification is not sufficient due to the performance measures that can be used (Brownlee, 2014).

When posterior diseases that analyze a model of machine learning can become a paradox of precision, it is difficult to control false positives and false negatives. This means that the patient may suffer from a rare disease, but the machine learning model does not anticipate, since most of the data comes from patients without disease (Max Schubach, 2018). Likewise, if a fraud is detected, the goal is to identify whether the transaction is fraudulent or not, could be further investigated, since the transactions are not fraudulent, this causes the model to provide fraudulent transaction monitoring for good customers valid.

To overcome these challenges, different approaches have been developed that can be implemented during the pre-processing data analysis process. A commonly used strategy is called resampling, which includes sampling

and overfishing techniques. This method is the best to use when the data set is huge and reducing the amount of training examples helps improve runtime and storage problems. The random sampling method randomly selects the majority class observations that are eliminated until the data set is balanced (Choi, 2010).

If the dataset is balanced by eliminating the instance of the class represented in excess, it is called sampling which method works only with minority classes. Replicate the observations of the minority class to balance the data. It is also known as up sampling. Excess sampling can be obtained by adding similar examples of classes with little representation to balance the ratio of partial classes. Resampling can be done with or without modifications. (Karagod, 2018).

This problem has to do with the compromise between recall (percentage of really positive cases classified as such) and accuracy (percentage of positive assessments that are really positive). In situations where we want to detect instances of a minority class, we usually worry more about recovery than accuracy, because in the context of detection, it is generally more expensive to lose a positive instance than to falsely label a negative instance. For example, if we are trying to detect offensive content, it is trivial for a manual reviewer to find that the content is not really offensive, but it is much more difficult to identify offensive content that has never been marked as such. Therefore, when comparing approaches with unbalanced classification problems, consider the use of metrics beyond precision, such as recall, accuracy. It is possible that the modification of the optimized metric during the selection of the parameters or the selection of the model is sufficient to provide a desirable performance that detects the minority class (Attenberg, 2018).

In regular learning, we treat all the incorrect classifications in the same way, which causes problems with unbalanced classification problems, since there is no additional bonus to identify the minority class compared to the majority class. Cost-sensitive learning modifies it and uses a C (p, t) function (usually represented as a matrix) that specifies the cost of erroneously classifying a class t instance as a class p. This allows us to penalize the wrong classifications of the minority class more strongly than the wrong classifications of the majority class, with the hope that this increases the true positive rate. A common scheme for this is to have the cost equal to the inverse of the proportion of the data set that makes up the class. This increases the penalty when the class size decreases.

In the most extreme cases, it might be better to think of classification in the context of anomaly detection. When we detect anomalies, we assume that there is a normal distribution (s) of data points and anything that deviates sufficiently from that distribution (s) is an anomaly. When we reorganize our classification problem into an anomaly detection problem, we consider the majority class as the "normal" distribution of points and the minority as anomalies. There are many algorithms for the detection of anomalies, such as grouping methods, the SVM of a class and isolation forests (Soni, 2018).

USE OF PROGRAMMING R

Here we are using the binary Internet database .csv with four 400 registry variables. Admission is the first dependent variable whose value is 1 when admission is granted in another way or for non-admission granted. Gpa and gra are independent variables with their field values similar to another variable in the independent variable. Category 1 is the best classification and 4 the worst classification.

```
> mydata <- read.
csv("http://www.karlin.mff.cuni.cz/~pesta/prednasky/NMFM404/Data/binary.csv")
> head (mydata)
admit the gre gpa range
1 0 380 3.61 3
2 1 660 3.67 3
3 1 800 4.00 1
……
6 1 760 3.00 2
> str (mydata)
'data.frame': 400 obs. of  4 variables:
$ admit: int  0 1 1 1 0 1 1 0 1 0 ...
$ gre  : int  380 660 800 640 520 760 560 400 540
$ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39
$ rank : int  3 3 1 4 4 2 1 2 3 2 ...
> data=mydata
> data$admit=as.factor(data$admit)
> data$rank=as.factor(data$rank)
> str(data)
'data.frame': 400 obs. of  4 variables:
$ admit: Factor w/ 2 levels "0","1": 1 2 2 2 1 2 2 1
$ gre  : int  380 660 800 640 520 760 560 400 540
$ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08
$ rank : Factor w/ 4 levels "1","2","3","4": 3 3 1 4
> summary(data)
admit       gre           gpa       rank
 0:273   Min.   :220.0   Min.   :2.260   1: 61
 1:127   1st Qu.:520.0   1st Qu.:3.130   2:151
 Median :580.0   Median :3.395   3:121
 Mean   :587.7   Mean   :3.390   4: 67
 3rd Qu.:660.0   3rd Qu.:3.670
 Max.   :800.0   Max.   :4.000
```

There are 273 applications were not accepted and 127 were accepted, the gre score with ranged min 220 to 800 max, similarly the gpa range from 2.26 to 4.00 score and rank also ranged from 1 61 applicants is best and 4 categories with 67 applicants with worst were displayed using summary command.

```
> prop.table(table(data$admit))
0      1
0.6825 0.3175
```

Converting data into proportion implies that there were 68 percent student were not admitted and 31 percent of total applicants were only admitted demonstrate imbalanced data enrollment.

```
> barplot(prop.table(table(data$admit)),col=
rainbow(2),ylim=c(0,0.7),main="Class Distribution")
```

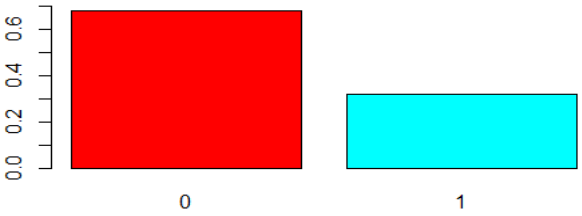

Fig:1 Bar plot of sample data

This bar plot demonstrates proportion of admit variable where 2/3 of data with 0 category were not admitted and 1/3 of data belongs to 1 category were only admitted, implied the same class imbalance problem to our data structure. If we use this data for prediction model will dominated only by not admitted category, similarly when accuracy will be calculated with considering such data will be highly influence with large sample proportion data. But our model should be best for prediction with admitted category.

```
> set.seed(123)
> ind=sample(2,nrow(data),replace=TRUE,prob=
c(.7,.3))
> train=data[ind==1,]
> test=data[ind==2,]
> table(train$admit)
0    1
188  97
```

The data patriation using set. seed always produce the same output and sample splitting with 70 and 30 splits is carried out using prob command 285 for training and 115 from test from 400 observation and stored with train and test variables.

```
> prop.table(table(train$admit))
0           1
0.6596491 0.3403509
```

Out of 285 records there were 65 percentage student were not admitted and 34 percent student were admitted.

```
> summary(train)
admit       gre           gpa       rank
 0:188   Min.   :220.0   Min.   :2.260   1: 40
 1: 97   1st Qu.:500.0   1st Qu.:3.120   2:112
 Median :580.0   Median :3.400   3: 83
 Mean   :582.4   Mean   :3.383   4: 50
 3rd Qu.:660.0   3rd Qu.:3.640
 Max.   :800.0   Max.   :4.000
```

The summary command applies in training data sets 28 5 records of random sample of 400 observations there were 188 observations in training data sets were not ad mitted and 97 were admitted displays similar activities of database.

```
> library(randomForest)
```

Random forest is basically suitable for multiclass problems, while SVM is intrinsically of two classes, so the multi-class problem of interval data must reduce it to more binary classification problems. Because the random forest works well with a combination of numeric and categorical database features when there were several scales that abound with an active encoding for categorical features with min. / Max or another scale, is strongly recommended in the pre-processing phase (Suganthan, 2018).

```
> rftrain=randomForest(admit~.,data=train)
```

Here admit is dependent with all independent variables of train data set.

```
> library(caret)
```

The caret package (Classification and Regression Training) contains functions to streamline the model training process for complex regression and classification problems. The package utilizes a number of R packages but tries not to load them all at package start-up removing formal package dependencies, the package startup time can be greatly decreased (Kuhn, 2008).

```
> confusionMatrix(predict(rftrain,test),test$admit,
positive="1")
```

A confusion matrix is a technique to summarize the performance of a classification algorithm. The accuracy of the classification alone can be misleading if you have a different number of observations in each class or if you have more than two classes in the data set. Calculating a confusion matrix can give you a better idea of what your classification model is doing and the types of mistakes you are making (Jason, 2016).

```
Confusion Matrix and Statistics
Reference
Prediction  0  1
         0 69 20
         1 16 10
Accuracy : 0.687
95% CI : (0.5938, 0.7702)
No Information Rate : 0.7391
P-Value [Acc > NIR] : 0.9142
Kappa : 0.1516
Mcnemar's Test P-Value : 0.6171
Sensitivity : 0.33333
Specificity : 0.81176
Pos Pred Value : 0.384622
Neg Pred Value : 0.77528
Prevalence : 0.26087
Detection Rate : 0.08696
Detection Prevalence : 0.22609
Balanced Accuracy : 0.57255
'Positive' Class : 1
```

The confusion matrix creates with prediction on training with test weights of admit data variable only having 1 for admission binary. After applying confusion matrix, the reference is the actual value and the prediction is the predicted from using model values, 69 applicants were

not admitted actually the model also actually predicts they were not admitted. There were 69 applicants were not admitted similarly there were 10 applicants actually admitted who were model predicts they were admitted were two correct predictions whereas 20 and 16 of diagonal values were mis-classification between model and prediction. Therefore 69+10=79 out of 115 test data is accuracy is 69 percent. This model only 69 percent accurate with 95 confidence interval the values lies in-between (0.5938, 0.7702). the information rate is largest proportion of the observed class is 73 percent (69+16)/115 which implies that if we do not apply any model there were 73 percent of total data predict actual values therefore this model will not any significant contribution. The accuracy should always grater then no information rate always. The sensitivity is 30 percent for the admitted values which is (10/30) whereas the specificity is how often 0 is predicted (69/85), those wide range describes that there is large imbalance which is dominated by class 0 whose prediction is not reliable while predicting 1 category. The best way to overcome this problem is randomly over sampling example

```
> library(ROSE)
> over=ovun.sample(admit~.,data=train,method
="over", N=376)$data
```

Here we are using ovun sampling model to the train data sets of 188*2 sample data for both

```
> table(over$admit)
0   1
188 188
> summary(over)
admit       gre          gpa       rank
 0:188   Min.   :220.0   Min.   :2.26   1: 57
 1:188   1st Qu.:520.0   1st Qu.:3.15   2:159
Median :580.0   Median :3.45   3:102
Mean   :589.8   Mean   :3.41   4: 58
3rd Qu.:660.0   3rd Qu.:3.65
Max.   :800.0   Max.   :4.00
```

This output do not differs much

```
> rfover=randomForest(admit~.,data=over)
> confusionMatrix(predict(rfover,test),test$admit,
positive="1")
Confusion Matrix and Statistics
Reference
Prediction  0  1
         0 58 14
         1 27 16
Accuracy : 0.6435
95% CI : (0.5488, 0.7306)
No Information Rate : 0.7391
P-Value [Acc > NIR] : 0.99119
Kappa : 0.1892
Mcnemar's Test P-Value : 0.06092
Sensitivity : 0.5333
Specificity : 0.6824
Pos Pred Value : 0.3721
Neg Pred Value : 0.8056
Prevalence : 0.2609
Detection Rate : 0.1391
Detection Prevalence : 0.3739
Balanced Accuracy : 0.6078
'Positive' Class : 1
```

From the above two confusion matrix output there is highly improvement for prediction of 1 correctly 16 predictions from previously 10 although there is overall accuracy decreased 64 percent from 68 percent Similarly the sensitivity is increased 53 percent form 30 percent before and specificity is decreased to 68 from 81 percent is improvement. Similarly, the under sampling from the sample 97 from the test data sets.

```
> under=ovun.sample(admit~.,data=train,method
="under", N=194)$data #97*2
> table(under$admit)
0  1
97 97
> rfunder=randomForest(admit~.,data=under)
> confusionMatrix(predict(rfunder,test),test$
admit,positive="1")
Confusion Matrix and Statistics
Reference
Prediction  0   1
         0 46  9
         1 39 21
Accuracy : 0.5826
95% CI : (0.487, 0.6739)
No Information Rate : 0.7391
P-Value [Acc > NIR] : 0.9999
Kappa : 0.1822
Mcnemar's Test P-Value : 2.842e-05
Sensitivity : 0.7000
Specificity : 0.5412
Pos Pred Value : 0.3500
Neg Pred Value : 0.8364
Prevalence : 0.2609
Detection Rate : 0.1826
Detection Prevalence : 0.5217
Balanced Accuracy : 0.6206
'Positive' Class : 1
```

From the under sampling model, the model improves a lots with predicting 1 category reached up to 21 items and sensitivity has been increased 70 percent. The test of both will be tested further.

```
> both=ovun.sample(admit~.,data=train,method
="both",p=0.5,seed=222,N=285)$data
> table(both$admit)
  0   1
134 151
> rfboth=randomForest(admit~.,data=both)
> confusionMatrix(predict(rfboth,test),test$admit,
positive="1")
Confusion Matrix and Statistics
Reference
Prediction  0  1
         0 47 14
```

1 38 16
Accuracy : 0.5478
95% CI : (0.4523, 0.6408)
No Information Rate : 0.7391
P-Value [Acc > NIR] : 0.999997
Kappa : 0.0685
Mcnemar's Test P-Value : 0.001425
Sensitivity : 0.5333
Specificity : 0.5529
Pos Pred Value : 0.2963
Neg Pred Value : 0.7705
Prevalence : 0.2609
Detection Rate : 0.1391
Detection Prevalence : 0.4696
Balanced Accuracy : 0.5431
'Positive' Class : 1

The previous output of both samples with p = 0.5 of 185 sampled data sets. Chance increases a lot. Similarly, the confusion matrix was reduced to 16 correct predictions of 1 category. Synthetic data, which are artificially created instead of being generated by real events. It is often created with the help of algorithms and used for a wide range of activities, including test data for new products and tools, for model validation and for the AI to take 500 samples (Rouse, 2018) .

```
> rose=ROSE(admit~.,data=train,N=500,seed=
111)$data
> table(rose$admit)
  0   1
234 266
> summary(rose)
admit       gre             gpa        rank
 0:234   Min.   :137.1   Min.   :2.235   1: 87
 1:266   1st Qu.:506.5   1st Qu.:3.169   2:192
 Median :585.8   Median :3.458   3:151
 Mean   :589.6   Mean   :3.437   4: 70
 3rd Qu.:680.6   3rd Qu.:3.754
 Max.   :913.2   Max.   :4.392
```

This summary report categorized 234 and 226 groups into two categories where gre goes up to 913 from 800 and gpa 4.39 out of 4 gpa when enlarging sample size.

```
> rfrose=randomForest(admit~.,data=rose)
> confusionMatrix(predict(rfrose,test),test$admit,
positive="1")
Confusion Matrix and Statistics
Reference
Prediction  0  1
          0 46 13
          1 39 17
Accuracy : 0.5478
95% CI : (0.4523, 0.6408)
No Information Rate : 0.7391
P-Value [Acc > NIR] : 0.9999969
Kappa : 0.0842
Mcnemar's Test P-Value : 0.0005265
Sensitivity : 0.5667
Specificity : 0.5412
Pos Pred Value : 0.3036
Neg Pred Value : 0.7797
Prevalence : 0.2609
Detection Rate : 0.1478
Detection Prevalence : 0.4870
Balanced Accuracy : 0.5539
'Positive' Class : 1
```

From the above rose method is quite improvement than previous both method, which could predict 1 category 17 data points accurately with 56 percent sensitivity.

CONCLUSION

Unbalanced data refer to classification problems in which there are unequal instances for different classes, whose unbalanced data is very common in the analysis of research data, but is implicit in particular for unequal Perdition in the case of categorical variables. The even more extreme imbalance is seen with fraud detection. it can be difficult to deal with a result Where one class heavily outperforms the other could be solved with the help of using static programming tools r. The following list is not exhaustive. For the sake of brevity, a quick overview is provided. weights of classes: impose a heavier cost When errors are made in the minority class, up sampling: randomly replicate instances in the minority class, synthetic minority sampling technique (beatings) establishes champions the majority classics and summarizes new minority situations through interpolation If they are themselves, these are considered as the "optimal" position on the ROC curves. The threshold invariant metrics can be improved using these methods, but the effect will not be pronounced.

REFREENCES

[1] Attenberg, J. (2018). Class Imbalance and Active Learning.

[2] Brownlee, J. (2014). Classification Accuracy is Not Enough: More Performance Measures You Can Use.

[3] Brownlee, J. (2014). Classification Accuracy is Not Enough: More Performance Measures You Can Use.

[4] Choi, J. M. (2010). A Selective Sampling Method for Imbalanced Data.

[5] EVA, G. (2015). A Tutorial on Multi-Label Learning.

[6] Henke, N. (2016). THE AGE OF ANALYTICS: compitation in data driven world.

[7] Jason, B. (2016). *Code Machine Learning Algorithms From Scratch.*

[8] Karagod, V. (2018). *Imbalanced machine learning using r.*

[9] Kuhn, M. (2008). Building Predictive Models in R Using the Caret Package.

[10] Manivannan, R. (2017). *Handling Imbalanced Data With R Data science.*

[11] Max Schubach, M. R. (2018). Imbalance-Aware Machine Learning for Predicting Rare and Common Disease-Associated Non-Coding Variants.

[12] Pozzolo, A. D. (2015). Calibrating Probability with Undersampling for Unbalanced Classification.

[13] Rouse, M. (2018). synthetic data.

[14] Soni, D. (2018). *Imbalanced data analysis .*

[15] Suganthan, R. K. (2018). Enhancing Multi-Class Classification of Random Forest using Random Vector Functional Neural.