A Minimal Book Example

Yihui Xie

2021-07-31

Contents

Chapter 1

Introduction

1.1 Abstract

In data-minig, a frequent and major issue for handling with two-class classification is to make reliable predictions with strongly imbalanced distribution of the target variable. Most of the machine learning algorithms assumes by default that all data are balanced. Empirically, a lot of dataset faces this distorsion (fraud detection, anticipation of catastrophes, donators in case of funding campain, unusual returns on stock markets, …). The majority class represents "normal cases", while the minority class represents "abnormal" cases. Because The least common values of the target variable are linked with events which are very relevant for users, we considered the minority class as positive and the majority class as negative.

The commun issue with classifiers is that they are unable to learn from the positive class. It results that the predictions are almost only negative class. Algorithms failed to predict the positive class which is properly what the users need.

Nowadays, the imbalanced data sets problem plays a key role in machine learning. During the last decades, literature was very prolific on this subject. Many tools were developped to solve this problem. This paper has neither ambition to give an exhaustive review of the existing solutions nor exploring new solutions. Moreover, we won't go to far in the explanation of the mathematical principle handling the algorithms. Our purpose is to propose some elements of solution to counteract the effect of imbalanced datatset which can be used an uderstanded by people who like data-minig without having a large scale of knowledge in this domain.

1.2 Document format

On chapter 1, we introduce some tools which allows to measure the effiency of a model.

Chapter 2 briefly presents the differents models we used. We will touch upon the math behind the classifier and the way the algorithm works.

Chapter 3 introduce some remedies to make our classifiers better predictors.

Chapter 4 is an application on three dataset with wich we can evaluate the submitted remedies.

Chapter 5 stands as a conclusion.

Chapter 2

Metrics for classification tasks

Understanding if a model is a good classifier or not require a good comprehension of the predictions he made. We want to know the global effiency but some specific element too. Is he better in predicting one or the other class? What are the strenght and weaknees of the model? Can we be confident towards the results, haw can we know they are not due to chance?

In order to compare the performance's models, we use differents "metrics". The reliability of our tools is higlhy dependant on the structure of our datas. In our case, the imbalanced sample of datas classes has to be take into account in order to find the metrics which allows to give a good evaluation of our models.

2.1 The foundation of the metrics: The Confusion Matrix

The confusion matrix presents the results obtained by a given classifier. This table provides the instances that were correctly classified (True Positive and True negative), and the instances that were wrongly classified (False Positive and False negative). From this table, we can calculate all the metrics described below

	Predicted	
	Positive	Negative
True		
Positive	TP	FN
Negative	FP	TN

2.2 Accuracy and error rate

$$error = \frac{FP + FN}{TN + TP + FP + FN}$$

$$accuracy = 1 - error$$

The first metrics is obviously the global accuracy and its complement the error rate. It is the most frequently used to estimate the performance of a model. If accuracy is too low, we deduce that our learning algolritm is globally inneficient. However In the context of imbalanced dataset, accuracy is not suitable. Indeed, because of the massive representation of the negative class, and as the classifiers failed to identify the positive class, we reach a high value of accuracy. For instance, if only 10% of the cases belong to the positive class and the classifiers predicts all cases as negative, accuracy will be at 90%. This is worthless when users objectives is to predict the rare cases.

To reflect more closely the users needs and priorities, several performance measure exist.

2.3 True Positive rate :

$$TP_{rate} = \frac{TP}{TP + FN} = \frac{TP}{P_{real}}$$

Also called sensitivity, recall or detection power. I personnaly prefer the term detection power because it is more explicit. It is the ratio of the value predicted as positive and which are actually positive among all the real positive. This is the ability of our classifier to detect the positive cases.

2.4 True Negative and Flase positive rate

Also called specificity. It is the ratio of the value predicted as negative and which are actually negative among all the real negative case.

$$TN_{rate} = \frac{TN}{TN + FP}$$

I prefer its complement, the False positive rate, also called False alarm. Indeed, the terme “False alarm” is more relevant than specificity.

$$FP_{rate} = \frac{FP}{TN + FP} = 1 - TN_{rate}$$

TPrate (detection power) and FP rate (False alarm) are often quote in the litterature as benefits and costs, respectively. These terms refers to a central

point of our problematic. Indeed, a key point to find good remedies is to make a trade-off between what it cost in terms of False alarm and the benefits gained in terms of detection power.

2.5 Positive prediction value : Precision

$$PP_{value} = \frac{TP}{TP + FP} = \frac{TP}{P_{pred}}$$

The precision measures the rate of True positive among all cases predicted as positive.

2.6 F-measure

The F-measure is a combiination of both precision and recall. This metric value is high when both recall (a measure of completness) and precision (a measure of exactness) are high (citation). Hence, this metrics is particuilary suitable on predicting the case that matter to the user.

$$F_\beta = \frac{(1 + \beta^2) \times recall \times precision}{\beta^2 \times recall + precision}$$

Beta is a coefficient to adjust the weight of recall against precision. In this paper, we choose a value of 1 which give the same weights to recall and precision.

2.7 Kappa

$$K = \frac{P_{agree} - P_{chance}}{1 - P_{chance}}$$

$$P_{agree} = \frac{TP + FN}{number\ of\ cases}$$

$$P_{chance} = \frac{P_{pre} \times P_{act}}{number\ of\ cases^2} + \frac{N_{pre} \times N_{act}}{number\ of\ cases^2}$$

Kappa is a very interesting metrics in context of imbalanced datas. The calculation is based on the difference between how much agreement(positive) is actually present ("observed") compared to how much positive would be expected to be present by chance alone ("expected"). We want to know how different the observed positive are from the expected. Kappa is a measure of this difference (citation).

score	app
< 0	Less than chance agreement
0.01-0.20	slight agreement
0.21– 0.40	Fair agreement
0.41–0.60	Moderate agreement
0.61–0.80	Substantial agreement
0.81–0.99	Almost perfect agreement

```
score <- c("< 0", "0.01-0.20","0.21- 0.40", "0.41-0.60", "0.61-0.80","0.81-0.99")
app <- c("Less than chance agreement", "slight agreement","Fair agreement","Moderate agreeme
gg <- cbind(score, app)
kable(gg) %>% kable_styling()
```

2.8 ROC Curve

A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

Chapter 3

Classifiers

3.1 LDA

3.2 Presentation

The Linear Discriminant analysis classifier is used to predict the probability of belonging to a given class based on one or multiple predictor variables. It works with continuous and/or categorical predictor variables.

Linear discriminant analysis is a generalization of Fisher's linear discriminant, a method used in statistics to find a linear combination of features that characterizes or separates two or more classes of objects.

In this approach, Fishers sought to find the linear combination of the predictors such that the between-group variance was maximized relative to the within-group variance. In other words, he wanted to find the combination of the predictors that gave maximum separation between the centers of the data while at the same time minimizing the variation within each group of data.

3.2.1 Learning LDA model

μ_k and σ_k the mean and variance for the k-class. LDA makes predictions by estimating the probability that a new set of inputs belongs to each class. LDA makes some simplifiying assumptions about your data: - that your data are gaussian - that each distribute the same variance, so $\sigma_0 = \sigma_1 = \sigma$ The model uses Bayes Theorem to estimate the probabilities

$$P(Y = c|X = x) = \frac{\pi_c f_c(x)}{\sum_{j=1}^{k} \pi_j f_j(x)}$$

Where π_c refers to the base probability of each class (c) observed in your training data (e.g. 0.5 for a 50-50 split in a two class problem). In Bayes' Theorem this

is called the prior probability.

$$\pi_c = \frac{n_c}{n}$$

Estimation of f_c is more delicate, it is the estimated probability of x belonging to the class. A Gaussian distribution function is used for f_c. Plugging the Gaussian into the above equation and simplifying we end up with the equation below. This is called a discriminate function and the class is calculated as having the largest value will be the output classification :

x will be affected to the class c for which:

$$\hat{\delta}_c(x) = \ln \hat{\pi}_c - \frac{\hat{\mu}_c^2}{2\hat{\sigma}^2} + x\frac{\hat{\mu}_c}{\hat{\sigma}^2}$$

is maximum.

Dk(x) = x * (muk/siga^2) – (muk^2/(2*sigma^2)) + ln(PIk)

$\hat{\delta}_c(x)$ is the discriminate function for class c given input x, the mu_k, $sigma^2$ and π_c are all estimated from your data.

3.3 logistic regression

In statistics, the logistic model (or logit model) is used to model the probability of a certain class. Each object being detected in the image would be assigned a probability between 0 and 1, with a sum of one. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression)

Binary Logistic Regression Major Assumptions The dependent variable should be dichotomous in nature (e.g., presence vs. absent). There should be no outliers in the data, which can be assessed by converting the continuous predictors to standardized scores, and removing values below -3.29 or greater than 3.29. There should be no high correlations (multicollinearity) among the predictors. This can be assessed by a correlation matrix among the predictors. Tabachnick and Fidell (2013) suggest that as long correlation coefficients among independent variables are less than 0.90 the assumption is met. At the center of the logistic regression analysis is the task estimating the log odds of an event. Mathematically, logistic regression estimates a multiple linear regression function defined as:

$$logit(p) = \log \frac{p(y=1)}{1-p(y=1)} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_i x_i$$

for i = 1…n .

Overfitting. When selecting the model for the logistic regression analysis, another important consideration is the model fit. Adding independent variables to a logistic regression model will always increase the amount of variance explained in the log odds (typically expressed as R^2). However, adding more and more variables to the model can result in overfitting, which reduces the generalizability of the model beyond the data on which the model is fit.

3.4 SVM

Support vector machine try to make a reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum-margin classifier; or equivalently, the perceptron of optimal stability. More formally, a support-vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin, the lower the generalization error of the classifier.

The support vector machines create an optimum hyperplane that separates the training data by the maximum margin. However, sometimes we would like to allow some misclassifications while separating categories. The SVM model has a cost function, which controls training errors and margins. For example, a small cost creates a large margin (a soft margin) and allows more misclassifications. On the other hand, a large cost creates a narrow margin (a hard margin) and permits fewer misclassifications. In this recipe, we will illustrate how the large and small cost will affect the SVM classifier.

image

3.5 classification tree

Tree-based models consist of one or more nested if-then statements for the predictors that partition the data. Within these partitions, a model is used to predict the outcome.

Choosing the trees split points Technically, for regression modeling, the split cutoff is defined so that the residual sum of squared error (RSS) is minimized across the training samples that fall within the subpartition.

Recall that, the RSS is the sum of the squared difference between the observed outcome values and the predicted ones, RSS = sum((Observeds - Predicteds)^2). See Chapter **?**(linear-regression)

In classification settings, the split point is defined so that the population in subpartitions are pure as much as possible. Two measures of purity are generally used, including the Gini index and the entropy (or information gain).

For a given subpartition, Gini = sum(p(1-p)) and entropy = -1*sum(p*log(p)), where p is the proportion of misclassified observations within the subpartition.

The sum is computed across the different categories or classes in the outcome variable. The Gini index and the entropy varie from 0 (greatest purity) to 1 (maximum degree of impurity)

image

3.6 random forest

RF classifier is an ensemble method that trains several decision trees in parallel with bootstrapping followed by aggregation, jointly referred as bagging (Fig. 9.17). Bootstrapping indicates that several individual decision trees are trained in parallel on various subsets of the training dataset using different subsets of available features. Bootstrapping ensures that each individual decision tree in the random forest is unique, which reduces the overall variance of the RF classifier. For the final decision, RF classifier aggregates the decisions of individual trees; consequently, RF classifier exhibits good generalization.

image

parameters :

mtry : number of variables randomy sampled at each split ntree : number of tree to grow nodesize : minimum number of observation in a terminal node. setting it lower heads to trees with a larger depth which means that more splits are performed until the terminal nodes. (default value is 1 for classification and 5 for regression -diaz urirarte and de andres 2006).

3.7 Naives bayes

basics of formula :

$$P(A \cap B) = P(A)P(B|A) \iff P(B|A) = \frac{P(A \cap B)}{P(A)}$$

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features

Naives : The joint probability calculation is simpler for independent events. so we consider events are independent. (it will be too complexe for more than two events)

Laplace correction will allow a small chance for these types of unforeseen circumstances (if joint event probabilty equals to 0.)

numeric data use bins to regroup them (hour cans be group by morning/afternoon/evening, temperature can be group by hot/warm/cold)

image

Chapter 4

Remedies

A lot of research have been made concerning this problem. Our goal is not to make an exhaustive review of all the technics to remedies this issue. In this study, we will focus on methods than we can repoduce with our level of competence. It appears to us that it is interesting to separate the choosen methods in three levels :

- First, some remedies we can use before launching the machine learning algorithm (Preprocessing).
- Secundly, some remedies we can use during the computation of a fitted models by the machine (learning method tuning).
- At last, som remedies that can be used after the machine learning algorithm (postprocessing).

4.1 different strategies :

- Special-purpose learning method : Modifications of the learning algorithm
  - advantages :
    - users goals are incorporated directly into the models
    - model obtained more comprehensive for the users
  - disadvantages :
    - user is restricted in his algorithmm choices or have to developp new algorithm
    - if the target of the loss function changes, model must be relearned
    - requires deep knowlegde of the learning algorithm implementation
- Data Pre-processing : changes on the data before the learning process takes place
  - advantages:
    - can be applied to any existing tools

        * choosen models are biased to the goals of the users
    - inconevnient:
        * difficult to relate modification of the data with the loss function
- Prediction Post-processing : transformations applied to the predictions of the learned model
    - advantages :
        * not necessary to know user preference biases at learning time
        * any standard learning tool can be used
    - drawbavks :
        * models do not reflect user preferences
        * models interpretability is meaningless because loss function was not optimized following user preference bias

4.2 Pre processing resampling

A lot of method exists in order to rebalanced a sample. Since the begin of the 21th century, a lot of studies appears in order to manage and discuss about resampling. Some methods are more sophisticated than other and might give better results, but they have no consensus on a technic whive have better results. Globally we can say it is the most commun and popular remedies to counteract imbalanced data for two key reasons : we can us the resample datas with all classisfiers and it gives generally pretty goods results.

The first "basic" try of resampling is obviously try to make random over sampling or down sampling. Over sampling consists in duplicate randomly positive cases in order to reach two class with approximatively the qame number of case. Downsampling consists in randomly enlever some negative cases in order to diminush the negative classes. Then we can easily imagine other way to select the cases otted or duplicate. Moreover, we can mix try to mix the two methods. Some try resampling dataset method:

Japkowicz (2000)discussed the effect of imbalance in a dataset. Two resampling methods were considered.Random resampling consisted of resampling the smaller class at random until it consisted of as many samples as the majority class and "focused resampling" consisted of resampling only those minority examples that occurred on the boundary between the minority and majority classes. Random under-sampling was considered, which involved under-sampling the majority class samples at random until their numbers matched the number of minority class samples; focused under-sampling involved under-sampling the majority class samples lying further away. She noted that both the sampling approaches were effective, and she also observed that using the sophisticated sampling techniques did not give any clear advantage in the domain considered (Japkowicz, 2000)

One approach that is particularly relevant to our work is that of Ling and Li (1998).They combined over-sampling of the minority class with under-sampling

of the majorityclass

in order to try and discuss those resamplig procedees, the part "application" shows examples of random up and down sampling in order to make comparison with more sophisticated method as SMOTE and ROSE.

4.2.1 CARET up and down sample :

"downSample will randomly sample a data set so that all classes have the same frequency as the minority class. upSample samples with replacement to make the class distributions equal"

4.2.2 ROSE

Creates a sample of synthetic data by enlarging the features space of minority and majority class examples. Operationally, the new examples are down from a conditionnal kernel density estimate of the two classes as describe in Menardi and torelli (2013)

The ROSE strategy to deal with class imbalance ROSE (Menardi and Torelli, 2014) provides a unified framework to deal simultaneously with the two above-mentioned problems of model estimation and accuracy evaluation in imbalanced learning. It builds on the generation of new artificial examples from the classes, according to a smoothed bootstrap approach (see, e.g., Efron and Tibshirani, 1993). Consider a training set Tn, of size n, whose generic row is the pair (xi, yi), i = 1, . . . , n. The class labels yi belong to the set fY0, Y1g, and xi are some related attributes supposed to be realizations of a random vector x defined on Rd, with an unknown probability density function f (x). Let the number of units in class Yj, j = 0, 1, be denoted by nj < n. The ROSE procedure for generating one new artificial example consists of the following steps: 1. Select y = Yj with probability pj. 2. Select (xi, yi) 2 Tn, such that yi = y, with probability 1 nj . 3. Sample x from KHj (, xi), with KHj a probability distribution centered at xi and covariance matrix Hj. Essentially, we draw from the training set an observation belonging to one of the two classes, and generate a new example (x, y) in its neighborhood, where the shape of the neighborhood is determined by the shape of the contour sets of K and its width is governed by Hj. It can be easily shown that, given selection of the class label Yj, the generation of new examples from Yj, according to ROSE, corresponds to the generation of data from the kernel density estimate of f (xjYj), with kernel K and smoothing matrix Hj (Menardi and Torelli, 2014). The choices of K and Hj may be then addressed by the large specialized literature on kernel density estimation (see, e.g. Bowman and Azzalini, 1997). It is worthwhile to note that, for Hj ! 0, ROSE collapses to a standard combination of over- and under-sampling. Repeating steps 1 to 3 m times produces a new synthetic training set T m, of size m, where the imbalance level is defined by the probabilities pj (if pj = 1/2, then approximately the same number of examples belong to the two classes). The size m may be set to the original training set size n or chosen in any way

4.2.3 Smote family (smoteNC, BLSMOTE, ADASYN)

smote chawla quote :

We propose an over-sampling approach in which the minority class is over-sampled by cre-ating "synthetic" examples rather than by over-sampling with replacement. This approachis inspired by a technique that proved successful in handwritten character recognition (Ha& Bunke, 1997). They created extra training data by performing certain operations onreal data. In their case, operations like rotation and skew were natural ways to perturbthe training data. We generate synthetic examples in a less application-specific manner, byoperating in "feature space" rather than "data space". The minority class is over-sampledby taking each minority class sample and introducing synthetic examples along the linesegments joining any/all of thekminority class nearest neighbors. Depending upon theamount of over-sampling required, neighbors from theknearest neighbors are randomlychosen. Our implementation currently uses five nearest neighbors. For instance, if theamount of over-sampling needed is 200%, only two neighbors from the five nearest neigh-bors are chosen and one sample is generated in the direction of each. Synthetic samplesare generated in the following way: Take the difference between the feature vector (sample)under consideration and its nearest neighbor. Multiply this difference by a random numberbetween 0 and 1, and add it to the feature vector under consideration. This causes theselection of a random point along the line segment between two specific features. Thisapproach effectively forces the decision region of the minority class to become more general

SMOTE (Chawla et. al. 2002) is a well-known algorithm to fight this problem. The general idea of this method is to artificially generate new examples of the minority class using the nearest neighbors of these cases. Furthermore, the majority class examples are also under-sampled, leading to a more balanced dataset.

The SMOTE function oversamples your rare event by using bootstrapping and k-nearest neighbor to synthetically create additional observations of that event. The definition of rare event is usually attributed to any outcome/dependent/target/response variable that happens less than 15% of the time. For more details about this algorithm, read the original white paper, SMOTE: Synthetic Minority Over-sampling Technique, from its creators.

- references
  - Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16:321-357.
  - Torgo, L. (2010) Data Mining using R: learning with case studies, CRC Press (ISBN: 9781439810187).

!!!! a lot of new techniques directly derived from this one as ADASYN

image

4.3 Learning method tuning

4.3.1 Metaparameters tuning

A first try can be to handle the parameters of the classifiers used. Empirycally we'll see part 5 application that it don't give some significants results. We coul wait for this results because parameters of classifiers are not ax on it.

4.3.2 Weighted class

We try to attribute some weights to our class but the results was not convinced. We try different ways and look for serious or significant results with this methods but it seems that it don't give results. Maybe the algorithm incorporing these results must be reshaped.

The best try trying to handle with classifiers during the learning phase was to incorporate a cost matrix. We choose to use it with decision tree because the algorithm CART shows good results in the literrature.

4.3.3 direct sensitive learning with cart classifer

The incorporation of benefits and/or costs (negative benefits) in existing algorithms, as a way to express the utility of different predictions, is one of the known approaches to cope with imbalanced domains.

Instead of optimizing the typical performance measure, such as accuracy or impurity, some models can alternatively optimize a cost or loss function that differentially weights specific types of errors. (no results with weighted data but cart works)

Theory We summarize the theory of cost-sensitive learning, published mostly in (Elkan, 2001; Zadrozny and Elkan, 2001). The theory describes how the misclassification cost plays its essential role in various cost-sensitive learning algorithms.

Cost Matrix: A matrix that assigns a cost to each cell in the confusion matrix. The example below is a cost matrix where we use the notation C() to indicate the cost, the first value represented as the predicted class and the second value represents the actual class. The names of each cell from the confusion matrix are also listed as acronyms, e.g. False Positive is FP.

table

We can see that the cost of a False Positive is C(1,0) and the cost of a False Negative is C(0,1).

This formulation and notation of the cost matrix comes from Charles Elkan's seminal 2001 paper on the topic titled "The Foundations of Cost-Sensitive Learning."

An intuition from this matrix is that the cost of misclassification is always higher than correct classification, otherwise, cost can be minimized by predicting one class.

Conceptually, the cost of labeling an example incorrectly should always be greater than the cost of labeling it correctly.

— The Foundations Of Cost-sensitive Learning, 2001.

Total Cost = C(0,1) * False Negatives + C(1,0) * False Positives This is the value that we seek to minimize in cost-sensitive learning, at least conceptually.

This cost matrix method can be used in different ways (…), in our applications part 5, we'll focus on cart.

4.4 post processinf threesholding

Alternate Cutoffs

Not efficient in my spotify works

> When there are two possible outcome categories, another method for increasing the prediction accuracy of the minority class samples is to determine alternative cutoffs for the predicted probabilities

Not a real solution for our issue because it changes our definition classes

> which effectively changes the definition of a predicted event.

Unless we do it with care / i don't think it is a good solution

> There may be situations where the sensitivity/specificity trade-off can be accomplished without severely compromising the accuracy of the majority class (which, of course, depends on the context of the problem)

Interesting if :

- focus on a compromise beetween sensityvioty and specificity. > particular target that must be met for the sensitivity or specificity,
- we want to maximise accuracy. > Find the point on the ROC curve that is closest (i.e., the shortest distance) to the perfect model (with 100 % sensitivity and 100 % specificity)
- use of Youden's J index ??? > (see Sect. 11.2), which measures the proportion of correctly predicted samples

Chapter 5

Applications

5.1 Introduction

In order (Xie, 2015) to illustrate and discuss the different remedies proposed in the previous chapter, we are handling each on different dataset. Hence we can make comparisons and try to measure their efficiency.

Our first choice as classifiers was to use LDA, LR, RF and SVM. having ascertained that LDA et LR give very similar results, we decide to substitute LR by naives bayes'classifier in order to proposed a richer experience (show plots). Notice that we first try to use glmnet instead of glm but it doesn't deliver better results (see spot.rmd). It is not unexpected that LR and LDA give nearly predictions, indeed they both are linear models and litteracy confirms they both give similar results (quote). !! maybe put it in classifiers part !!!!

Few words About the code : We don't introduce here all the manipulations done on the datasets, either the preparation of the dataset. You can find them in this github repositery, wich contains the .rmd for each dataset. In this repositery, you can also find the .R file which contains also the functions we code in order to avoid to many repetition in the code. At last, the alldat.Rdata stocked all objects built in the .rmd, it is used here to call the object we need.

We choose four dataset with different level of imabalanced.

Let's briefly presents those datasets:

- Spotify ...
- Recidivism ...
- Creditcard ...
- Hacked ...

Table of priors ratio between positive and negative class

Table 5.1: Confusion matrix spotify

	0	1	Sum	0	1	Sum
	rf			bayes		
0	4431	975	5406	4373	983	5356
1	147	115	262	205	107	312
Sum	4578	1090	5668	4578	1090	5668
	lda			svm		
0	4559	1078	5637	4566	1075	5641
1	19	12	31	12	15	27
Sum	4578	1090	5668	4578	1090	5668

Table 5.2: Spotify metrics

	accuracy	FNrate	TPrate	kappa	PrecisionPPV	Fscore
rf	0.8020466	0.8944954	0.1055046	0.1032829	0.4389313	0.1701183
Bayes	0.7904023	0.9018349	0.09816514	0.07332293	0.3429487	0.1526391
lda	0.8064573	0.9889908	0.01100917	0.01088917	0.3870968	0.02140946
svm	0.8082216	0.9862385	0.01376147	0.01772557	0.5555556	0.02685765

5.2 First Models

The function models compute our four models. We show the function in order to show the basic parameters. This parameters will be change in a following section. For now, we just want to observe results with basic parameters. This first computation can be used as a start reference to measure the remedies tested later.

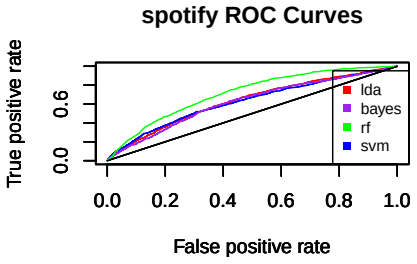
The table … shows the confusion matrix resulting to the four classifiers used on the spotofy dataset. We observe the unabilty to properly predict the unpopular songs. A look on the metrics sharps this observation.

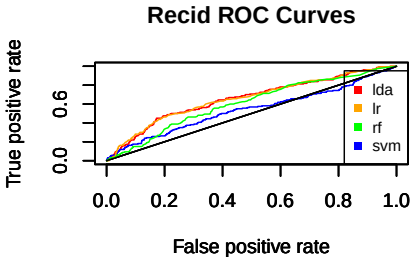
First we note that accuracy is very good, wigch confirms accuracy is not a relaible metrics concerning imbalance dataset. a simple view on Detection power(TPr) shows that we don't achieve to predict what songs are very un popular. FN rate is obviously good because of the imblalanced ratio. Here FN won't be a good metrics.

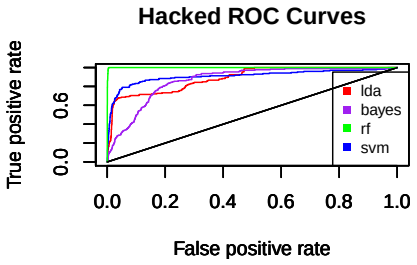
Let see the plot curve for all datasets.

Table 5.3: Hacked metrics

	accuracy	FNrate	TPrate	kappa	PrecisionPPV	Fscore
rf	0.9949815	0.05813953	0.9418605	0.9419781	0.9473684	0.9446064
Bayes	0.9477021	0.872093	0.127907	0.1597342	0.3142857	0.1818182
lda	0.9537771	0.7732558	0.2267442	0.287576	0.4814815	0.3083004
svm	0.9635499	0.7034884	0.2965116	0.4098061	0.75	0.425







!!! discuss the better results for creditcard and hacked !!! - from kaggle (datasets directly from professional use, more relevant, data more reliable) ? preprocessing (pca, onlynumeircal), extreme imbalanced ?

Even if creditcard and hacked seems not to need remedies to counteract imbalanced data, let see the detection power. We can argue that 3/4 of detection power is not enough to reassure users. In a professional use, we can wish at leat 90% of TPr.

At last, it is sure that all explicative variable are dependent to the variable to predict and and non colinear between them. for spotify we can'b sure that these variable can explain popularity, mayb there's no link between them. As the variable for recid, are they really a good choice.

On this one, we can see that only rf has very good results. Detection power and Fscore allows to judge the performance classifiers, contrarly to accuracy or FNr which are not usefull here. Rf has the better results, In second position

Table 5.4: test

	accuracy	FNrate	TPrate	kappa	PrecisionPPV	Fscore
rf	0.8278724	0.2075142	0.7924858	0.6557449	0.8528444	0.821558
Bayes	0.6241809	0.2647444	0.7352556	0.2483617	0.6016086	0.6617517
lda	0.6021188	0.3934032	0.6065968	0.2042377	0.6012124	0.6038926
svm	0.7035824	0.2505461	0.7494539	0.4071647	0.6864746	0.7165831

comes svm, even if he detects only a third among True positive, its quite a good results for a first try with an imbalanced data set, and Fscore ank kappa are not so bad.

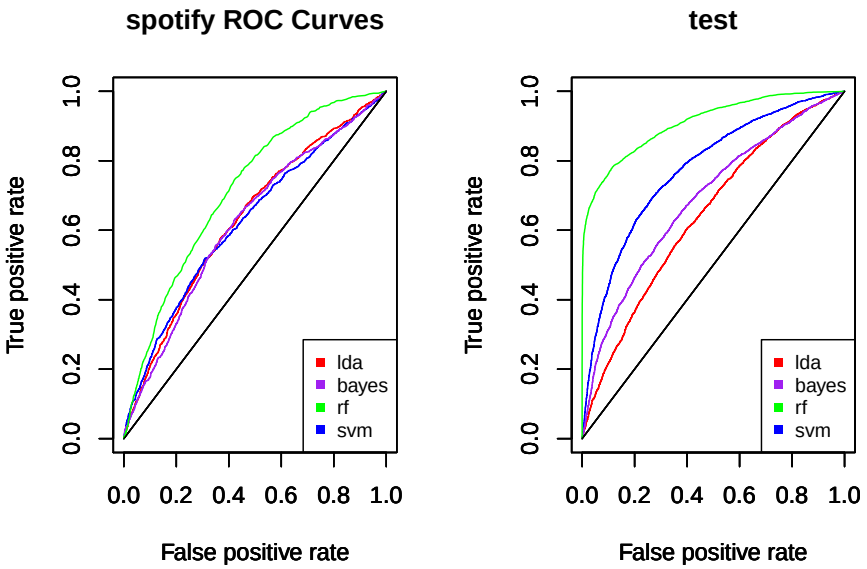
5.3 Preprocessing : Resampling methods

As we can see with the function 'resamp'in the funclin.R and the .rmd , we try many resampling with different method. As the smote methods seems to us pertinent with this approach of oversampling with some logic, we try some develepment as adasyn or BLSMOTE. Note that the original smote algorithm is only for numerical variable. Some databases have few or no numerical variable so it is not an issue but a dataset as recid (with many caterogicla variable) asks to act on this variable. We could try to transform them as numerical but we choose to use SMoteNC, which is developped to deal with bot numerical and caterogical variable.

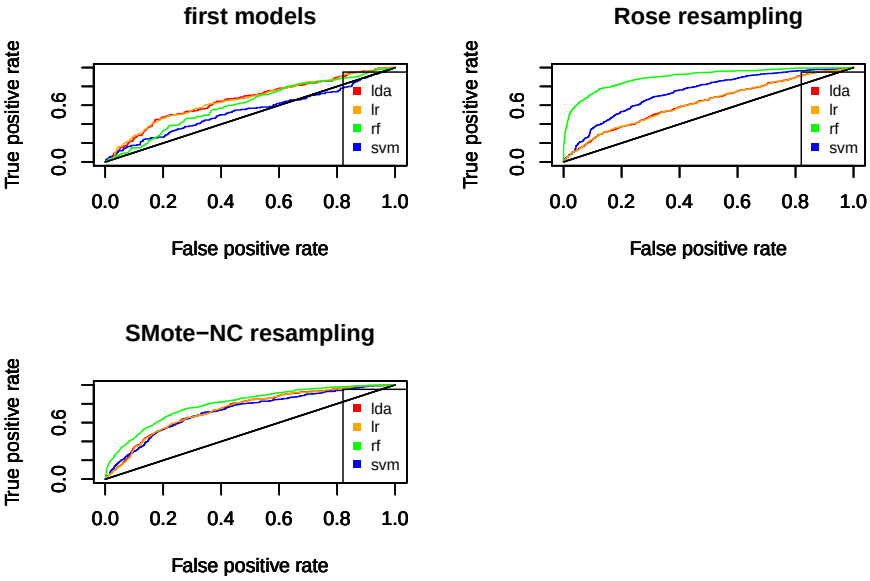
an example of downsampling with spotify

Here another example with smoteNC spotify

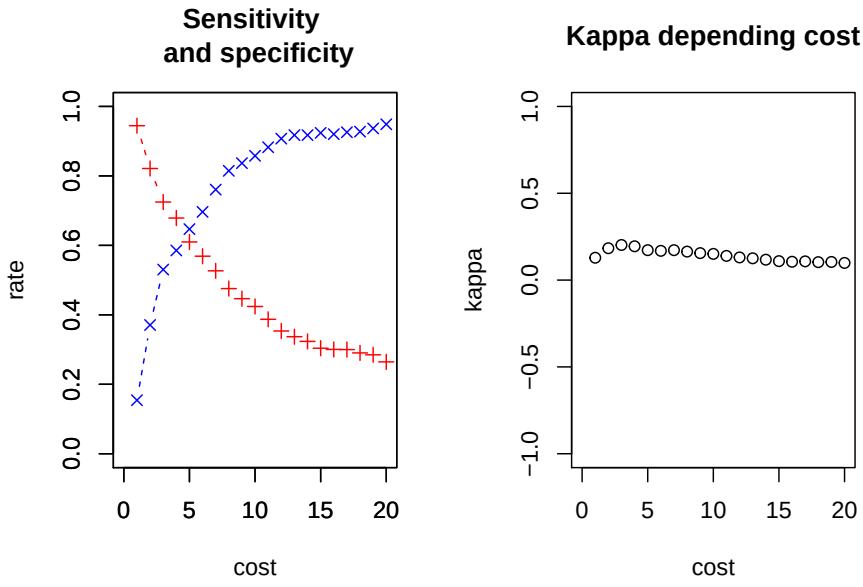
Lets compare RocCurve before and after smote resampling



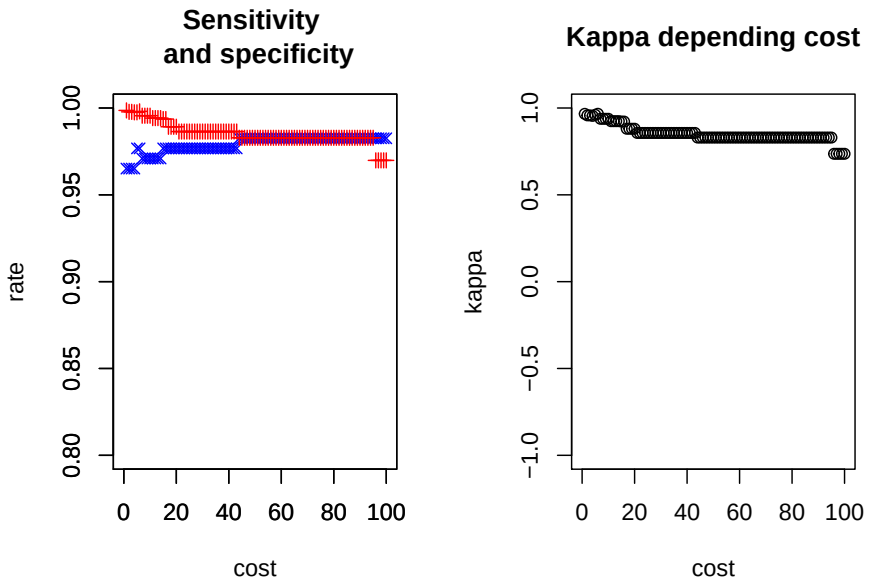

```
par(mfrow=c(2,2))
AllRoc0(predic = PredRec1, dataCl = datRecid$test$is_violent_recid,
        caption = "first models")
AllRoc0(predRoseRecid, dataCl = datasRoseRecid$test$is_violent_recid, caption = "Rose resamp
AllRoc0(predic = predRecSmc, dataCl = datrecSmc$test$is_violent_recid, caption = "SMote-NC r
```



5.4 Direct cost sensisitive learnig



```
C5graph(Hacked, "MULTIPLE_OFFENSE", captest = "Kappa depending cost", captest2 = "Sensitivit
```



5.5 Post processing Threesholding

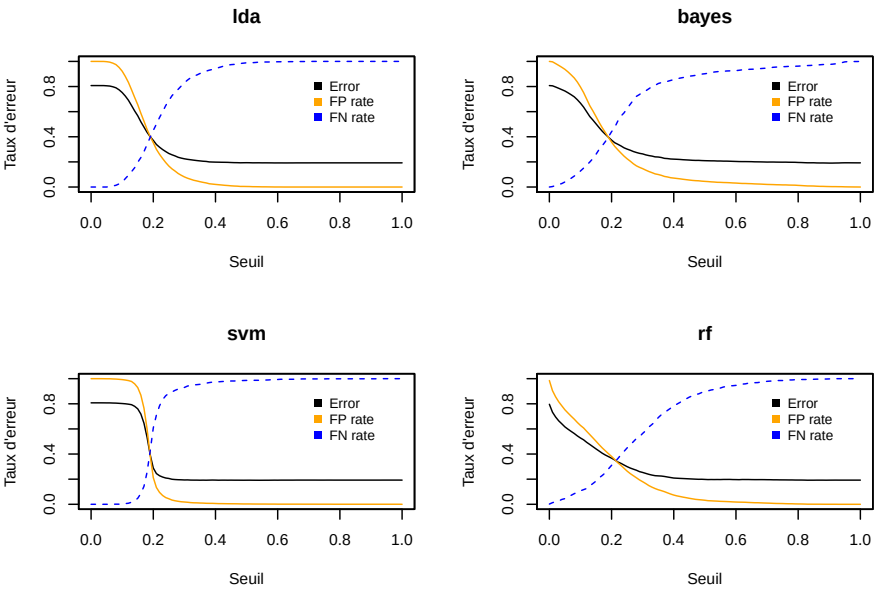


Table 5.5: perfplop

	accuracy	FNrate	TPrate	kappa	PrecisionPPV	Fscore
rf	0.6328511	0.3100917	0.6899083	0.2073542	0.3014028	0.4195258
Bayes	0.6264996	0.440367	0.559633	0.1438748	0.2714731	0.3655978
lda	0.6284404	0.4504587	0.5495413	0.1412709	0.270551	0.3625908
svm	0.7115385	0.6082569	0.3917431	0.1619352	0.305218	0.3431097

```
## Warning in if (listPred == "bayes") {: la condition a une longueur > 1 et seul
## le premier élément est utilisé
```

```
## Warning in if (listPred == "lr") {: la condition a une longueur > 1 et seul le
## premier élément est utilisé
```

Chapter 6

Summary

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.