YÖNETİM BİLİŞİM SİSTEMLERİ DERGİSİ

http://dergipark.ulakbim.gov.tr/ybs/

Yayın Geliş Tarihi: 30.05.2016 Cilt:1, Sayı:3, Yıl:2016, Sayfa 181-191

Yayına Kabul Tarihi: 15.08.2016 ISSN: 2148-3752

Online Yayın Tarihi: 05.10.2016

COMPARISON OF SAMPLING TECHNIQUES FOR IMBALANCED LEARNING

Ahmet Onur DURAHİM

Abstract

In recent years, huge increase in the number of people using Internet accompanied massive amounts of human and machine generated data recently called Big Data, where handling it efficiently is a challenging task. Along with that, valuable information that can be extracted from this data to perform data-driven decision making has attracted increased attention both from industry and academia. One of the important tasks in knowledge extraction is the classification task. However, in some of the real-world applications, dataset is either inherently skewed or collected dataset has imbalanced class distribution. Imbalance class distribution degrades the performance of several classification algorithms which generally expect balanced class distributions and assume that the cost of misclassifying an instance from both of the classes is equivalent. To tackle with this so called imbalanced learning problem, several sampling algorithms has been proposed in the literature. In this study, we compare sampling algorithms with respect to their running times and classification accuracies obtained from running classifiers trained with the sampled datasets. We find out that classification accuracies of the over-sampling methods are superior to the under-sampling methods. Sampling times are found to be similar whereas classification can be done more efficiently with under-sampling methods. Among the proposed sampling algorithms, the ADASYN method should be the preferred choice considering both execution times, increase in the data size and classification performance.

Keywords: Imbalanced Learning, Sampling Methods, Data Mining, Big Data

DENGESİZ ÖĞRENME İÇİN ÖRNEKLEME TEKNİKLERİNİN KARŞILAŞTIRILMASI

Özet

Günümüzde artan İnternet kullanıcıları sayısıyla birlikte insanlar ve makinalar tarafından büyük miktarda Büyük Veri denilen veri üretilmektedir. Bu veriyi verimli bir şekilde işlemek zor bir iştir. Bununla birlikte, bu veriden veri güdümlü karar alabilmede kullanmak üzere çok değerli olan bilgiyi çıkarabilme hem endüstrinin, hem de akademinin ilgisini çekmektedir. Bilgi çıkarmanın önemli görevlerinden birisi de sınıflandırmadır. Gerçek hayatta gördüğümüz uygulamlarda elde edilen veri seti ya doğal olarak dengesizdir yada toplanan very dengesiz sınıf dağılımına sahiptir. Dengesiz sınıf dağılımı ise, genel olarak verinin dengeli olduğunu ve yanlış sınıflandırmalarda maliyetin sınıflar arasında farklılık göstermediğini varsayan birçok sınıflandırma metodunun performansını kötüleştirmektedir. Dengesiz öğrenme problemi denilen bu sorunla başa çıkmak için literatürde birçok örnekleme metodları önerilmiştir. Bu çalışmada, bu metodların çalışma süreleri ve örnekleme yapılmış veri setleri üzerinden eğitilmiş sınıflandırıcıların sınıflandırma doğrulukları bakımından karşılaştırmaları yapılmıştır. Bu bağlamda yukari örnekleneme metodları, aşağı örnekleme metodlarına kıyasla sınıflandırma doğrulukları daha iyi çıkmıştır. Örnekleme zamanları birbirine yakın çıkmakla birlikte sınıflandırma, very setinin küçülmesi sebebiyle daha hızlı yapılabilmiştir. Sonuç olarak, sınıflandırma doğrulukları, işlem süreleri ve very setinin büyüklüğü göz önünde bulundurulduğunda, ADASYN algoritmasının tercih edilebilir olduğu tavsiye edilmektedir.

Anahtar Kelimeler: Dengesiz Öğrenme, Örnekleme Metodları, Veri Madenciliği, Büyük Veri

INTRODUCTION

Today, with the increase in the data being generated by human beings and machines, extracting useful insights from this data becomes much more attractive and valuable for the businesses doing data-driven decision making. At the same, performing this task is difficult due to the huge volume and high velocity of the data being generated. This huge volumes of data being generated fast in variety of forms is today considered as Big Data. So, it is now of utmost important being able to get those insights from big data fast and in a reliable manner.

One of the major data mining tasks for knowledge extraction from data is classification. Through classification, one tries to predict the class label of a given data using supervised learning techniques. In supervised learning, where we have actual class labels of the dataset at hand, and a classifier is generated using this dataset in order to label previously unseen data. Reliability of these classification models are tested by computing several classification performance evaluation metrics among which accuracy, precision, recall and ROC curves are the most widely used ones (Fatourechi et al., 2008)

One of the major problems in classification encountered in many real-world applications is the occurrence of imbalance class distribution in the available dataset. Here, dataset exhibits skewed distribution between its classes, where there are abundant number of instances that belong to one of the classes, and therefore this class is heavily overrepresented compared to the other class. Approaches proposed for dealing with data exhibiting imbalanced class distribution are aggregated under the name of imbalanced learning (He et al., 2009). Imbalance class distribution leads to a drastic worsening of the performance of standard classification algorithms. These algorithms generally expect balanced class distributions and assume that the cost of misclassifying a data for both classes is equivalent. Resultantly, they fail to provide good classification accuracies, especially in favor of the majority class.

To better explain this problem, consider a real-world problem classification problem, detection of cancer. Assume that available data consists of attributes of patients where 95% of them are diagnosed as cancer-free, and the remaining 5% are known to have cancer. So, if you create a classifier that always outputs "cancer-free" for any given patient, then the accuracy, defined here as hit rate, of this classifier will be 95%. Although, the high accuracy obtained from this simple classifier seems to be a great result, it misclassifies all the patients who should have been diagnosed as cancer. So, the evaluation metrics that are used in imbalanced learning becomes significantly important.

For the case of cancer detection, the cost of misclassifying a person as cancer-free is much greater than misclassifying a person as cancer and corresponding results will be devastating for those misclassified patient who has cancer. Misclassified patients may lose their lives which cannot be compared with any monetary value.

Accordingly, there are two basic problems that must be addressed for imbalanced learning. First one is the sensitivity of the classification algorithms to imbalanced training datasets which can be addressed by sampling techniques (He et al., 2009, Weiss, 2004]. Second one is the evaluation metrics that are being used to compare the success of these classifiers. (Fatourechi et al., 2008, Dal Pozzolo et al., 2013)

In this study, we have analyzed sampling techniques which attempt to overcome the imbalance class distribution problem by bringing balance to input dataset either by eliminating instances from the majority class or generating additional data for the minority class. We have chosen proposed approaches given in Table 1, which fall under two broad categories, namely under-sampling and over-sampling techniques. Under-sampling methods use different approaches to eliminate instances from the majority class to bring balance to the number of instances in each class. In contrast, over sampling methods generate data points, synthetic or replicated, that belong to the minority class.

Table 1. Under- and Over-Sampling Methods for Imbalanced Learning

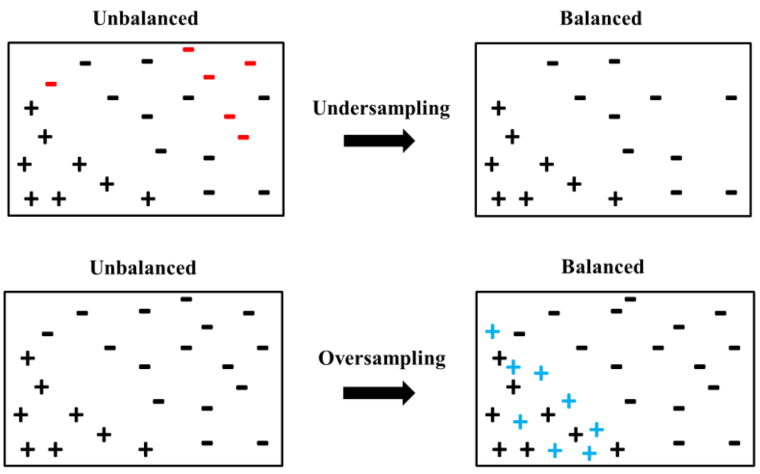
	Under-sampling	**Over-sampling**
Sampling Methods	Random majority under sampling	Random minority over sampling
	Nearmiss methods: NM1, NM2, NM3	SMOTE (Synthetic Minority Oversampling)
	Condensed Nearest Neighbor (CNN)	Borderline SMOTE B-SMOTE1, B-SMOTE2
	Edited Nearest Neighbor (ENN)	Borderline-SVM-SMOTE
	Neigboorhood Cleaning Rule	ADA-SYN (Adaptive Synthetic Sampling)
	Tomek links removal	
	One Sided Selection	

In the following section, related work will be discussed where we will elaborate on each one of the sampling methods proposed in the literate. Then, in the results section we will present comparative analysis of the chosen sampling methods. To do so, timings of sampling and classification procedures together with the results of the evaluation metrics represented by accuracy, precision, recall values and true positive rate over false positive rate will be given. Lastly, we will conclude and discuss the limitations and future work.

RELATED WORK

In this section, under-sampling and over-sampling methods will be discussed from the perspective of their differences. First of all, we will discuss under-sampling methods which target to balance class distribution through the elimination of majority class examples. Then, we will review over-sampling methods where the aim is generating instances to be added into minority class either by replicating examples from this class or by creating synthetic instances.

Figure 1. Undersampling and Oversampling



The central problem with under-sampling approaches is that, eliminating the members of the majority class may cause losing potentially valuable information from dataset at hand. Different under-sampling proposals analyzed in this study try to alleviate this problem.

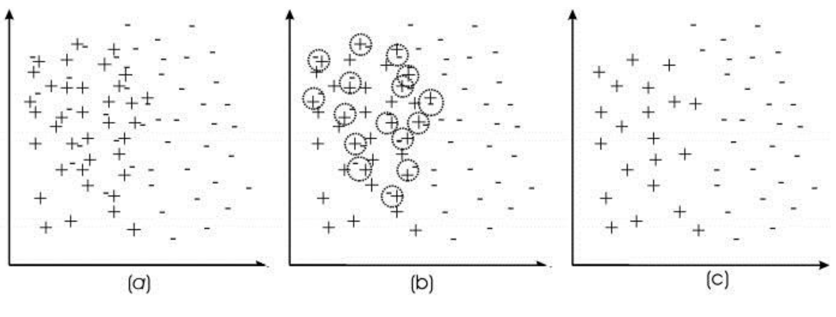
First and simplest of the under-sampling methods to be considered is the *Random Majority under-sampling* method. In this approach, instances are randomly eliminated from the majority class until the number of data points in the majority class is equal to the one in the minority class. Main concern in using this method is that one cannot know and control what information about the majority class is thrown away. As a result, one may eliminate instances that carry information about the decision boundary between the majority and minority classes which separates the two classes, and lose that valuable information.

Another proposal which consists of three different methods based on K-nearest neighbor classifier called *Nearmiss methods*, Nearmiss 1, 2 and 3, is introduced in (Mani, Inderjeet, and I. Zhang, 2003). Instead of randomly eliminating instances, these methods eliminate majority samples in an intelligible way. *Nearmiss 1* method, majority class examples that are closer to the minority class examples are chosen to be removed. Near Miss 1 selected majority examples whose average distances to three nearest minority examples are the smallest. In the second type of Nearmiss, *Nearmiss 2,* examples from the majority class with the smallest average distances to three farthest minority examples are chosen to be eliminated. Last Nearmiss method, *Nearmiss 3*, chooses a given number of majority class instances to be eliminated that are closer to each minority example. NearMiss3 ensures that each minority examples is surrounded by some majority examples (He et al., 2009). Experimental results provided in (Mani, Inderjeet, and I. Zhang, 2003) claim that Nearmiss2 method outperforms other two Nearmiss methods. Besides, it also outperforms both random under-sampling method and so-called "most distance" method where data points to be eliminated are the ones having largest distances to the three closest minority class data points.

Condensed Nearest Neighbor rule tries to find a consistent subset of the training examples (P. E. Hart, 1968). A consistent subset is defined as the subset of the original data set where by using 1-Nearest Neighbor method, this subset correctly identifies all the examples in the original dataset. This method eliminates examples that are distant from the borderline which don't possess information that is valuable to the underlying learning algorithm.

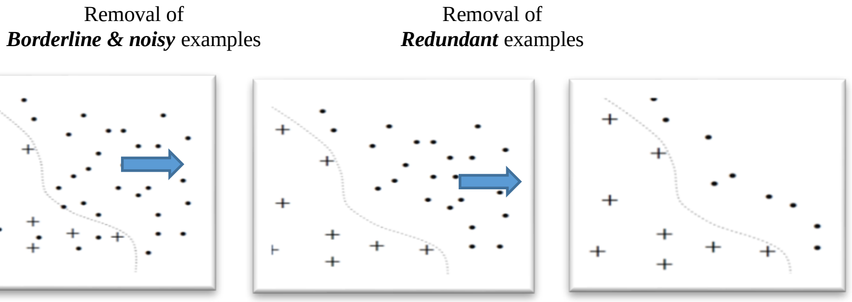
Tomek link removal is another technique proposed for imbalanced learning (I. Tomek, 1976). In this approach, examples that belong to Tomek links are removed from the dataset. A pair of examples x and y belong to a Tomek link, if there is no other example z where the distance between this example and any one of x, and y is less than the distance between x and y. Examples that form Tomek links are either considered as borderline or one of the instances that form a Tomek link is considered as a noisy example (see Figure 2 adopted from (Batista et al., 2004))

Figure 2. (a) Original data set, (b) Tomek Link identification, (c) Dataset after borderline and noisy examples removed.



Another method proposed in (Kubat, Miroslav, and Stan Matwin, 1997) is called *One Sided Selection*, OSS. OSS adapted Tomek link removal technique (I. Tomek, 1976) but it only removes examples from majority class dataset and keeps all the minority examples untouched. Here, first redundant examples are eliminated, adapting a variant of condensed nearest neighbor rule (P. E. Hart, 1968), followed by the removal of the borderline and noisy examples, adapting (I. Tomek, 1976) (see Figure 3.)

Figure 3. One Sided Selection method



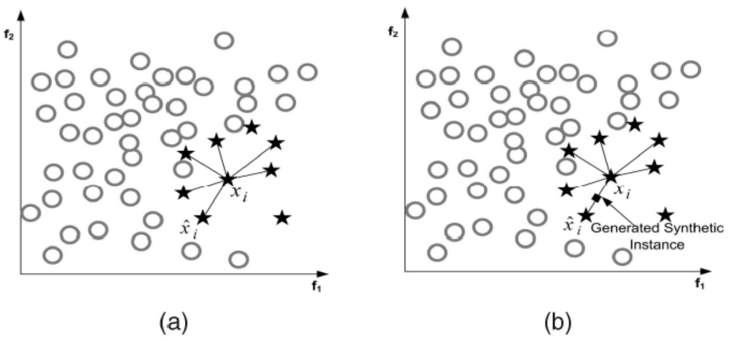
Edited Nearest Neighbor rule, ENN, (Wilson, 1972) removes from both majority and minority classes if an example has at least two of its three nearest neighbors with the opposite class label. An extension to ENN method is called *Neighborhood Cleaning Rule* is proposed in (Laurikkala, 2001) where instances are removed only from the majority class. For each example in the training set, if that example belongs to the majority class and classified as minority class by its three nearest neighbors, then it is removed from the dataset. Besides, if example belongs to the minority class and it is classified as majority class by its three nearest neighbors, then those neighbors which are members of the majority class are eliminated from the dataset.

In contrast to under-sampling methods, proposals that attempt to balance the imbalanced class distribution by generating instances for the minority class falls under the second broad category of over-sampling methods for imbalanced learning. The advantage of this category over the under-sampling method is that in these approaches you don’t lose any valuable information since you keep all of the original majority and minority class instances. However, you greatly increase the size of the training set which leads to increased sampling and model generation times.

First method to be considered under this category is the simplest approach of creating new instances at random, called *random over-sampling method*. Random over-sampling method simply duplicates the examples in the minority class until the class distributions come to a balance. This simple over-sampling method leads to overfitting problem.

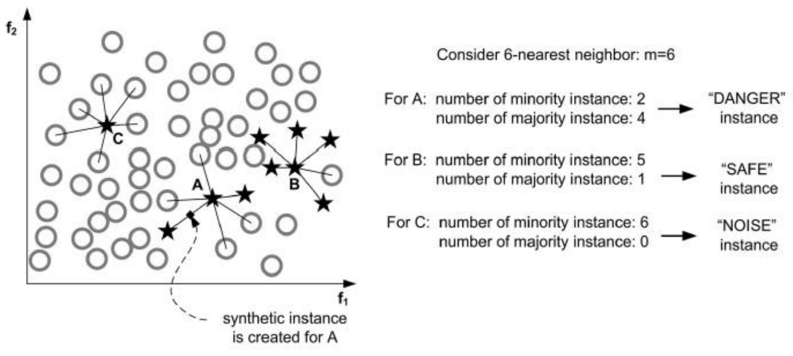
Second and most popular over-sampling method is the *Synthetic Minority Over-sampling Technique, SMOTE* (Chawla et al., 2002) which tries to avoid overfitting problem. Instead of replicating minority samples, SMOTE creates new synthetic minority class examples by interpolating between existing data points of the minority class that are closer to each other. Figure 4 taken from (He et al., 2009) illustrates this approach where number of nearest neighbors is chosen as 6 and new instance is generated by interpolating between the example in consideration and a randomly chosen data point in the nearest neighbor set.

Figure 4. Illustration of Synthetic Minority Over-sampling Technique



Basic SMOTE has problems like over generalization due to the occurrence of overlapping between classes and variance (B.X. Wang and N. Japkowicz, 2004). To overcome these problems, extensions have been proposed to the basic proposal, including Borderline-SMOTE (Han et al., 2005), Borderline-SVM-SMOTE (Nguyen et al., 2011), Adaptive Synthetic Sampling, ADASYN (He et al., 2008), and Majority weighted minority oversampling, MWMOTE (Barua et al., 2014). Borderline SMOTE is illustrated in Figure 5 adapted from (He et al., 2009), seeks to avoid creating overlapping examples by determining those instances in the "DANGER" set which represent the borderline minority class examples for which at least half of the nearest neighbors are from the majority class but not all of them. So, as opposed to SMOTE procedure where a synthetic instance is generated for each one of the examples in the minority class, in Borderline-SMOTE, synthetic examples are generated only for the minority instances in the "DANGER" set.

Figure 5. Borderline-SMOTE. Danger, Safe and Noise instances


f₂
f₁
A
B
C
synthetic instance is created for A
Consider 6-nearest neighbor: m=6
For A: number of minority instance: 2
number of majority instance: 4
"DANGER" instance
For B: number of minority instance: 5
number of majority instance: 1
"SAFE" instance
For C: number of minority instance: 6
number of majority instance: 0
"NOISE" instance

Two different procedures are proposed in (Han et al., 2005), namely Borderline-SMOTE1 and Borderline-SMOTE2. In both of these two methods, an instance is created for each example in the "DANGER" set by interpolating between these instances and their nearest neighbors of the same class. Only difference between the two is that, in the second procedure, additional instances are created again for each instance in the "DANGER" set, but now for the nearest neighbors that belong to the opposite class.

Borderline-SVM-SMOTE uses SVM method to find the borderline (Nguyen et al., 2011). ADASYN on the other hand, creates synthetic instances adaptively by considering the distribution of the dataset (He et al., 2008). MWMOTE is the last over-sampling technique that we consider in this study (Barua et al., 2014). MWMOTE generates synthetic instances using cluster-based approach instead of using k-NN method used in SMOTE variants. This way it avoids creating new instances that may reside in the majority class region.

In this study we extend previous studies, (Fatourechi et al., 2008, Japkowicz, 2000, Olivier, 2012, Dal Pozzolo et al., 2013, Dittman et al., 2014), by including recent sampling proposals. In addition to providing comparisons of the methods in terms of accuracy scores, we also provide sampling and classification timings.

METHODOLOGY

In this study, we have attempted to compare methods proposed to attack on the imbalance encountered in the datasets used in data mining tasks, both qualitatively and quantitatively. In order to obtain quantitative results, we decided to compare the sampling methods proposed for imbalanced learning discussed so far, based on their sampling times and the accuracy scores obtained from running different classification algorithms using the datasets that are preprocessed by these methods. For that purpose, first of all we have utilized python libraries to implement these methods as well as running the classification algorithms.

Then, there is a need for imbalanced datasets to be obtained on which preprocessing will be accomplished by the sampling methods considered in the study. In that respect, we have decided on using data mining datasets provided in the UCI Repository (A. Asuncion and D. J. Newman, 2007) which is the common approach also used in the previous studies similar to ours (Olivier, 2012, He et al., 2009). Since emphasis is given to two-class classification problems, we have selected two imbalanced two-class datasets, Car Evaluation and SPECT Heart, where corresponding sample sizes, number of attributes, and class distributions are given in Table 2. The reason behind this selection is the huge difference in the imbalance ratios of chosen datasets. Besides, sample size and the number of features in these datasets allow us to run the simulations in reasonable amounts of time.

Table 2. Information about size, number of attributes and imbalance ratio of the datasets used

	Car Evaluation Dataset	**SPECT Heart Dataset**
Sample Size	1728	267
Number of Attributes	6	22
Imbalance Ratio	65/1663 (≈1/25)	55/212 (≈1/4)

To get accuracy performances for the binary classifications applied on the sampled datasets, we utilize scikit-learn python library (Pedregosa et al., 2011) which is a widely used machine learning library. To evaluate sampling methods, three different classification algorithms implemented in scikit-learn are selected. These methods are Support Vector Machine (SVM) where linear kernel is chosen, Random Forest and k-Nearest Neighbor method with k = 5. All the other parameters not mentioned here are left at their default values. In order to get accuracy performances, first, sampling is performed with one of the proposed sampling methods, and then a classifier is trained with under/over sampled datasets. Classification of the original dataset is accomplished by the model generated with the sampled dataset and corresponding accuracies are computed accordingly. Results given in the following section are the averages computed over the corresponding values obtained by performing 10 runs.

In addition to classifier accuracy performances, we also provide timings for both sampling and classification procedures. Since, in the age of big data, stream processing is one of the important challenges, sampling and classification timings of the proposed approaches become exceedingly important besides the accuracies obtained. So, enterprises may be forced to make a tradeoff between running times and classification accuracies.

RESULTS

In this section we will summarize the results of the classification runs accomplished on a 64-bit Windows running computer, with 8GB's of RAM and Intel i7-4600U CPU running at @2.7 GHz. Results are shown in Figure 6 and Figure 7, for each one of the two datasets, Car Evaluation and SPECT Heart, respectively. All the execution times are given in seconds.

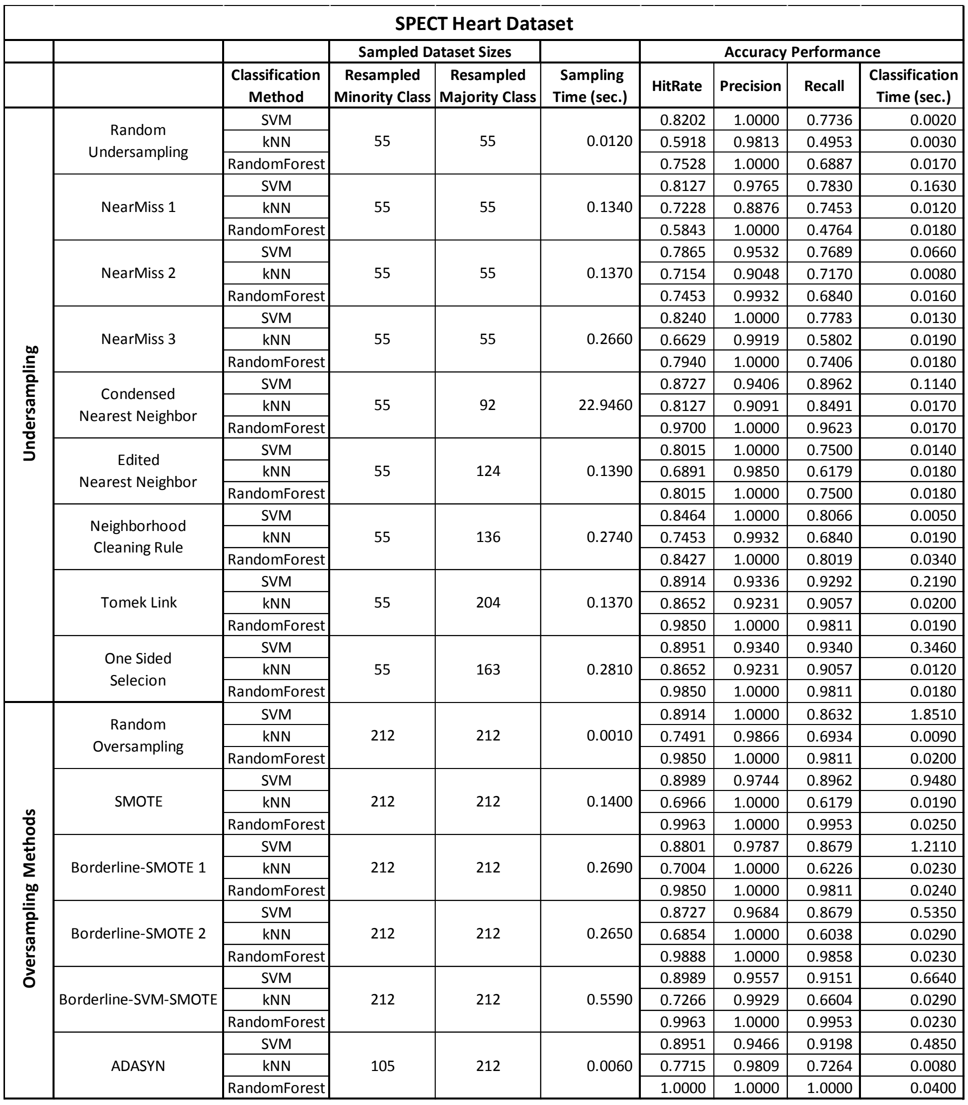
Figure 6. Results for the Car Evaluation dataset

Car Evaluation Dataset									
			Sampled Dataset Sizes			Accuracy Performance			
		Classification Method	Resampled Minority Class	Resampled Majority Class	Sampling Time (sec.)	HitRate	Precision	Recall	Classification Time (sec.)
Undersampling	Random Undersampling	SVM	65	65	0.0080	0.9167	0.3110	1.0000	0.0030
		kNN				0.7963	0.1559	1.0000	0.0090
		RandomForest				0.9797	0.6500	1.0000	0.0280
	NearMiss 1	SVM	65	65	0.1470	0.9005	0.2622	0.9077	0.0120
		kNN				0.8605	0.2124	1.0000	0.0240
		RandomForest				0.8235	0.1757	1.0000	0.0540
	NearMiss 2	SVM	65	65	0.1540	0.9618	0.4960	0.9538	0.0070
		kNN				0.8756	0.2302	0.9846	0.0200
		RandomForest				0.8079	0.1637	1.0000	0.0850
	NearMiss 3	SVM	65	65	0.2900	0.3987	0.0589	1.0000	0.0060
		kNN				0.4323	0.0621	1.0000	0.0200
		RandomForest				0.4543	0.0645	1.0000	0.0650
	Condensed Nearest Neighbor	SVM	65	135	35.3380	0.9716	0.5889	0.8154	0.0040
		kNN				0.9734	0.5856	1.0000	0.0100
		RandomForest				0.9844	0.7065	1.0000	0.0250
	Edited Nearest Neighbor	SVM	65	1608	0.1550	0.9803	0.7385	0.7385	0.0140
		kNN				0.9965	0.9836	0.9231	0.0580
		RandomForest				0.9988	0.9701	1.0000	0.0480
	Neighborhood Cleaning Rule	SVM	65	1619	0.2790	0.9803	0.7385	0.7385	0.0230
		kNN				0.9971	1.0000	0.9231	0.0680
		RandomForest				1.0000	1.0000	1.0000	0.0650
	Tomek Link	SVM	65	1661	0.1550	0.9821	0.8400	0.6462	0.0160
		kNN				0.9954	1.0000	0.8769	0.0580
		RandomForest				1.0000	1.0000	1.0000	0.0370
	One Sided Selecion	SVM	65	1628	0.2740	0.9809	0.7857	0.6769	0.0150
		kNN				0.9977	1.0000	0.9385	0.0650
		RandomForest				1.0000	1.0000	1.0000	0.0420
Oversampling Methods	Random Oversampling	SVM	1663	1663	0.0020	0.9479	0.4183	0.9846	0.0730
		kNN				0.9387	0.3801	1.0000	0.0430
		RandomForest				1.0000	1.0000	1.0000	0.0330
	SMOTE	SVM	1663	1663	0.2130	0.9525	0.4414	0.9846	0.0920
		kNN				0.9965	0.9155	1.0000	0.0550
		RandomForest				1.0000	1.0000	1.0000	0.0380
	Borderline-SMOTE 1	SVM	1663	1663	0.3210	0.9427	0.3963	1.0000	0.0680
		kNN				0.9965	0.9155	1.0000	0.0380
		RandomForest				1.0000	1.0000	1.0000	0.0690
	Borderline-SMOTE 2	SVM	1662	1663	0.3810	0.7975	0.1566	1.0000	0.2810
		kNN				0.9016	0.2766	1.0000	0.0870
		RandomForest				1.0000	1.0000	1.0000	0.0450
	Borderline-SVM-SMOTE	SVM	1663	1663	0.6540	0.9491	0.4248	1.0000	0.1410
		kNN				0.9809	0.6633	1.0000	0.0720
		RandomForest				1.0000	1.0000	1.0000	0.0770
	ADASYN	SVM	791	1663	0.0190	0.9618	0.4961	0.9846	0.0390
		kNN				1.0000	1.0000	1.0000	0.0320
		RandomForest				1.0000	1.0000	1.0000	0.0340

Considering the sampling methods based on accuracy results, in general, oversampling methods obtained better performances as compared to the undersampling methods, as it is expected. This higher accuracies obtained from the oversampling methods support the fact that by utilizing undersampling methods one loses important discriminating information available in the original dataset.

When we consider undersampling methods, first of all, we obtained similar results for the NearMiss methods provided in (Mani, Inderjeet, and I. Zhang, 2003). From the perspective of the accuracy performances, NearMiss 2 outperforms other NearMiss methods. Besides, although CNN method eliminates nearly all the imbalanced data from the majority class instances from both datasets, it exhibits similar accuracy performances to Tomek Link method which achieves the best classification accuracy scores. However, Tomek Link method is able to eliminate only 2 majority class instances from Car Evaluation, and 8 majority class instances from the SPECT Heart disease datasets. Therefore, accuracy results obtained for the Tomek Link method is somewhat misleading. So, if one must reduce the sample size, but do not want to compromise classification accuracy that much, then CNN should be the preferred sampling method.

Figure 7. Results for the SPECT Heart dataset



SPECT Heart Dataset									
			Sampled Dataset Sizes			Accuracy Performance			
		Classification Method	Resampled Minority Class	Resampled Majority Class	Sampling Time (sec.)	HitRate	Precision	Recall	Classification Time (sec.)
Undersampling	Random Undersampling	SVM	55	55	0.0120	0.8202	1.0000	0.7736	0.0020
		kNN				0.5918	0.9813	0.4953	0.0030
		RandomForest				0.7528	1.0000	0.6887	0.0170
	NearMiss 1	SVM	55	55	0.1340	0.8127	0.9765	0.7830	0.1630
		kNN				0.7228	0.8876	0.7453	0.0120
		RandomForest				0.5843	1.0000	0.4764	0.0180
	NearMiss 2	SVM	55	55	0.1370	0.7865	0.9532	0.7689	0.0660
		kNN				0.7154	0.9048	0.7170	0.0080
		RandomForest				0.7453	0.9932	0.6840	0.0160
	NearMiss 3	SVM	55	55	0.2660	0.8240	1.0000	0.7783	0.0130
		kNN				0.6629	0.9919	0.5802	0.0190
		RandomForest				0.7940	1.0000	0.7406	0.0180
	Condensed Nearest Neighbor	SVM	55	92	22.9460	0.8727	0.9406	0.8962	0.1140
		kNN				0.8127	0.9091	0.8491	0.0170
		RandomForest				0.9700	1.0000	0.9623	0.0170
	Edited Nearest Neighbor	SVM	55	124	0.1390	0.8015	1.0000	0.7500	0.0140
		kNN				0.6891	0.9850	0.6179	0.0180
		RandomForest				0.8015	1.0000	0.7500	0.0180
	Neighborhood Cleaning Rule	SVM	55	136	0.2740	0.8464	1.0000	0.8066	0.0050
		kNN				0.7453	0.9932	0.6840	0.0190
		RandomForest				0.8427	1.0000	0.8019	0.0340
	Tomek Link	SVM	55	204	0.1370	0.8914	0.9336	0.9292	0.2190
		kNN				0.8652	0.9231	0.9057	0.0200
		RandomForest				0.9850	1.0000	0.9811	0.0190
	One Sided Selecion	SVM	55	163	0.2810	0.8951	0.9340	0.9340	0.3460
		kNN				0.8652	0.9231	0.9057	0.0120
		RandomForest				0.9850	1.0000	0.9811	0.0180
Oversampling Methods	Random Oversampling	SVM	212	212	0.0010	0.8914	1.0000	0.8632	1.8510
		kNN				0.7491	0.9866	0.6934	0.0090
		RandomForest				0.9850	1.0000	0.9811	0.0200
	SMOTE	SVM	212	212	0.1400	0.8989	0.9744	0.8962	0.9480
		kNN				0.6966	1.0000	0.6179	0.0190
		RandomForest				0.9963	1.0000	0.9953	0.0250
	Borderline-SMOTE 1	SVM	212	212	0.2690	0.8801	0.9787	0.8679	1.2110
		kNN				0.7004	1.0000	0.6226	0.0230
		RandomForest				0.9850	1.0000	0.9811	0.0240
	Borderline-SMOTE 2	SVM	212	212	0.2650	0.8727	0.9684	0.8679	0.5350
		kNN				0.6854	1.0000	0.6038	0.0290
		RandomForest				0.9888	1.0000	0.9858	0.0230
	Borderline-SVM-SMOTE	SVM	212	212	0.5590	0.8989	0.9557	0.9151	0.6640
		kNN				0.7266	0.9929	0.6604	0.0290
		RandomForest				0.9963	1.0000	0.9953	0.0230
	ADASYN	SVM	105	212	0.0060	0.8951	0.9466	0.9198	0.4850
		kNN				0.7715	0.9809	0.7264	0.0080
		RandomForest				1.0000	1.0000	1.0000	0.0400

As mentioned before, oversampling methods reach higher accuracy scores when compared to undersampling ones. All the oversampling methods considered in this study except ADASYN make the size of the minority class equal to the size of the majority class. As one can see from results, although ADASYN generates less number of additional instances for the minority class data, it has achieved best accuracy scores as compared to other oversampling methods.

When we consider sampling times, we immediately notice high running times of the CNN algorithm for both datasets. Although the accuracy results obtained are promising, running times of the algorithm hinders the use of this algorithm in practical applications. So, there is a need for more efficient implementation of the CNN sampling method, maybe with some modifications to the original proposal, since it takes considerably large amount of time as compared the other methods.

In general, running times of the classification tasks performed with the sampled datasets do not exhibit notable differences. As it is expected from the resampled dataset sizes, it takes more time to perform a classification task when oversampling methods are benefitted since they increase the dataset sizes. Another important observation is the fact that number of attributes has considerably high impact, even more compared to the sample size, on the running times of the classification methods, especially for SVM.

To conclude, if the dataset size is not an issue, then one should employ ADASYN oversampling method in order to achieve high accuracy scores. On the other hand, if the size of the dataset is bigger than that can be handled efficiently, then one must employ an undersampling method. Here, if one can make CNN method more efficient then it should be the recommended method. Otherwise, ENN or NearMiss-2 methods should be the methods of choice. In addition, from the perspective of accuracy performances, SVM and Random Forest methods can be used for imbalanced learning for the types of datasets similar to ours.

CONCLUSION AND FUTURE WORK

In this study, we compare sampling algorithms with respect to their running times and classification accuracies obtained from running classifiers trained with the resampled datasets. We find out that classification accuracies of the over-sampling methods are superior to the under-sampling methods, as it is expected. Sampling times are found to be similar whereas classification can be done more efficiently with under-sampling methods due to the elimination of the instances from the majority class examples. But. this accompanied with reduced accuracy scores.

As a result, if one don’t want to compromise neither the accuracy of the classification task nor running time, then among the proposed sampling algorithms considered in this study, the ADASYN method should be the preferred choice.

As a future work, we will consider additional sampling methods proposed in the literature in addition to approaches that combine oversampling with undersampling methods. Besides, we will also compare these methods utilizing more real-world datasets with bigger sample sizes.

REFERENCES

A. Asuncion and D. J. Newman. UCI Machine Learning Repository. University of California at Irvine, School of Information and Computer Science, 2007.

Barua, Simul, Md Minarul Islam, Xin Yao, and Kazuyuki Murase. "MWMOTE--majority weighted minority oversampling technique for imbalanced data set learning." Knowledge and Data Engineering, IEEE Transactions on 26, no. 2 (2014): 405-425.

Batista, Gustavo EAPA, Ronaldo C. Prati, and Maria Carolina Monard. "A study of the behavior of several methods for balancing machine learning training data." ACM Sigkdd Explorations Newsletter 6, no. 1 (2004): 20-29.

B.X. Wang and N. Japkowicz, “Imbalanced Data Set Learning with Synthetic Samples,” Proc. IRIS Machine Learning Workshop, 2004.

Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique." Journal of artificial intelligence research (2002): 321-357.

Dal Pozzolo, Andrea, Olivier Caelen, Serge Waterschoot, and Gianluca Bontempi. "Racing for unbalanced methods selection." In Intelligent Data Engineering and Automated Learning–IDEAL 2013, pp. 24-31. Springer Berlin Heidelberg, 2013.

Dittman, David J., Taghi M. Khoshgoftaar, Randall Wald, and Amri Napolitano. "Comparison of data sampling approaches for imbalanced bioinformatics data." In The Twenty-Seventh International Flairs Conference. 2014

Fatourechi, Mehrdad, Rabab K. Ward, Steven G. Mason, Jane Huggins, A. Schlogl, and Gary E. Birch. "Comparison of evaluation metrics in classification applications with imbalanced datasets." In Machine Learning and Applications, 2008. ICMLA'08. Seventh International Conference on, pp. 777-782. IEEE, 2008.

Han, Hui, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning." In Advances in intelligent computing, pp. 878-887. Springer Berlin Heidelberg, 2005.

He, Haibo, and Edwardo A. Garcia. "Learning from imbalanced data." Knowledge and Data Engineering, IEEE Transactions on 21, no. 9 (2009): 1263-1284.

He, Haibo, Yang Bai, Edwardo A. Garcia, and Shutao Li. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning." In Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, pp. 1322-1328. IEEE, 2008.

Nguyen, Hien M., Eric W. Cooper, and Katsuari Kamei. "Borderline over-sampling for imbalanced data classification." International Journal of Knowledge Engineering and Soft Data Paradigms 3, no. 1 (2011): 4-21.

I. Tomek, “Two modifications of CNN,” IEEE Tram. Cyst., Man, Cybern., vol. SMG6, pp. 769-772, Nov. 1976.

Japkowicz, Nathalie. "Learning from imbalanced data sets: a comparison of various strategies." In AAAI workshop on learning from imbalanced data sets, vol. 68, pp. 10-15. 2000.

Kubat, Miroslav, and Stan Matwin. "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection." In In Proceedings of the Fourteenth International Conference on Machine Learning. 1997.

Laurikkala, Jorma. Improving identification of difficult small classes by balancing class distribution. Springer Berlin Heidelberg, 2001.

Mani, Inderjeet, and I. Zhang. "kNN approach to unbalanced data distributions: a case study involving information extraction." In Proceedings of workshop on learning from imbalanced datasets. 2003.

Olivier Caelen, Andrea Dal Pozzolo and Gianluca Bontempi. Comparison of balancing techniques for unbalanced datasets. Technical report, Machine Learning Group University of Bruxelles, Belgium, 2012

P. E. Hart, "The condensed nearest neighbor," IEEE Trans. Inform. Theory, vol. IT-14, pp. 515-516, May 1968.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al. "Scikit-learn: Machine learning in Python." The Journal of Machine Learning Research 12 (2011): 2825-2830.

Weiss, Gary M. "Mining with rarity: a unifying framework." ACM SIGKDD Explorations Newsletter 6, no. 1 (2004): 7-19.

Wilson, Dennis L. "Asymptotic properties of nearest neighbor rules using edited data." Systems, Man and Cybernetics, IEEE Transactions on 3 (1972): 408-421.