Generating Balanced Classifier-Independent Training Samples from Unlabeled Data

Youngja Park[1], Zijie Qi[2], Suresh N. Chari[1], and Ian M. Molloy[1]

[1] IBM T.J. Watson Research Center, P.O. Box 704,
Yorktown Heights, NY 10598, USA
{young_park,schari,molloyim}@us.ibm.com
[2] University of California Davis, Davis, CA 95616
zqi@ucdavis.edu

Abstract. We consider the problem of generating balanced training samples from an unlabeled data set with an unknown class distribution. While random sampling works well when the data is balanced, it is very ineffective for unbalanced data. Other approaches, such as active learning and cost-sensitive learning, are also suboptimal as they are classifier-dependent, and require misclassification costs and labeled samples. We propose a new strategy for generating training samples which is independent of the underlying class distribution of the data and the classifier that will be trained using the labeled data.

Our methods are iterative and can be seen as variants of active learning, where we use semi-supervised clustering at each iteration to perform biased sampling from the clusters. Several strategies are provided to estimate the underlying class distributions in the clusters and increase the balancedness in the training samples. Experiments with both highly skewed and balanced data from the UCI repository and a private data show that our algorithm produces much more balanced samples than random sampling or uncertainty sampling. Further, our sampling strategy is substantially more efficient than active learning methods. The experiments also validate that, with more balanced training data, classifiers trained with our samples outperform classifiers trained with random sampling or active learning.

1 Introduction

Supervised learning algorithms can provide promising solutions to many real-world problems such as text classification, anomaly detection and information security. A major limitation of supervised learning is the difficulty in obtaining labeled data to train predictive models. Ideally, one would like to train classifiers on diverse labeled data representative of all classes. In many domains, such as text classification or security, there is an abundant amount of unlabeled data, but obtaining a representative subset is challenging: data is typically highly skewed and sparse.

There are two widely used approaches for selecting data to label—random sampling and active learning. Random sampling, a low-cost approach, produces

P.-N. Tan et al. (Eds.): PAKDD 2012, Part I, LNAI 7301, pp. 266–281, 2012.
© Springer-Verlag Berlin Heidelberg 2012

a subset of the data with a similar distribution to the original data set, producing skewed training data for unbalanced data. Training with unbalanced labeled data yields poor results as reported in recent work on the effect of class distribution on learning and performance degradation [1–3]. Active learning produces training data incrementally by identifying the most informative data to label at each phase [4–6]. However, active learning requires knowing the classifier in advance, which is not feasible in many real applications, and requires costly re-training at each step.

In this paper, we present new strategies to generate training samples from unlabeled data to overcome limitations in random and existing active sampling methods. Our core algorithm is an iterative method, in which we generate a small fraction (e.g., 10%) of the desired training set each iteration, independently of *both* the original data distribution as well as the target classifier. More specifically, we first label a small number of randomly selected samples and subsequently apply semi-supervised clustering to embed prior knowledge (i.e., labeled samples) to produce clusters approximating the true classes [7–9]. We then estimate the class distribution of the clusters, and increase the balancedness of the training sample via biased sampling.

A simplistic strategy for biased sampling would be to assume that the class distribution of a cluster is the same as the distribution of labeled samples in the cluster, and to draw samples proportionally to the estimated class distributions. However, this assumption does not hold in early iterations when the number of labeled samples is small, and there is high uncertainty about the class distributions. We present two hybrid approaches to address this issue that perform well in practice. The first approach is to combine the estimated class distribution-based sampling and random sampling. As the number of labeled samples increases, we decrease the influence of random sampling favoring the estimation based on previously labeled samples. The second approach is for cases where additional domain knowledge is available. We use the domain knowledge to estimate class distributions. Domain knowledge may come in many forms, such as conditional probabilities and correlation, e.g., there is a heavy skew in the geographical location of servers hosting malware[10]. We perform a similar transition between the domain knowledge-based density estimation and previously labeled sample-based estimation.

We have validated these strategies on 14 data sets from the UCI data repository [11] as well as a private data set authorizing users to systems (i.e., labeled *grant* and *deny*). These data sets reflect a range of parameters: some are balanced and others highly skewed; and some have binary classes while others have multiple classes. We compare our strategies to random sampling as well as uncertainty based active sampling based on three classifiers: Naive Bayes, Logistic Regression, and SVM. The experiments show that, for highly skewed data sets, our sampling algorithm produces substantially more balanced samples than random sampling. For mildly skewed data sets, our method results in about 25% more minority samples. Similarly, our algorithm performs better than uncertainty sampling based methods for highly skewed samples, producing more than

20% more minority samples on average. For mildly skewed data sets, our algorithm's results are not statistically different from uncertainty sampling based on logistic regression. Given that uncertainty sampling requires one to fix the classifier to be trained and is much slower, we conclude that our algorithm is *always* preferable to both random and uncertainty based sampling. We test the domain knowledge based strategy on the access control permission datasets. Our result show that, in most cases, the addition of domain knowledge significantly improves the convergence of the sampling so we can produce balanced sample sets more quickly.

The quality of training data can best be evaluated by the performance of classifiers trained on this data. We have compared various sampling strategies by training and testing a range of classifiers. Our tests show that the classifiers built with our training data outperform other classifiers in most of the experimental scenarios and produce more consistent performance. Further, our classifiers often outperform uncertainty sampling on AUC and F1 measures, even when sampling and classification used the same classifier. The experimental results confirm that our sampling methods are very generic and can produce highly balanced training data irrespective of the underlying data distribution and the target classifier.

2 Related Work

There is an extensive body of work on generating "good" training data sets. A common approach is active learning, which iteratively selects informative samples, e.g., near the classification border, for human labeling [6, 12–14]. The sampling schemes most widely used in active learning are uncertainty sampling and Query-By-Committee sampling [13, 15, 16]. Uncertainty sampling selects the most informative sample determined by one classification model, while QBC sampling determines informative samples by a majority vote. A major problem with active learning is that the update process is very expensive as it requires classification of all data samples and retraining of the model at each iteration. This cost is prohibitive for large scale problems. Techniques such as batch mode active learning [17, 18] have been proposed to improve the efficiency of uncertainty learning. However, as the batch size grows, the effectiveness of active learning decreases [18–20].

Another approach is re-sampling, i.e., over- and under-sampling classes [21, 22], however this requires labeled data. Recent work combines active learning and re-sampling to address class imbalance in unlabeled data. Tomanek and Hahn [23] propose incorporating a class-specific cost in the framework of QBC-based active learning for named entity recognition. By setting a higher cost for the minority class, this method boosts the committee's disagreement value on the minority class resulting in more minority samples in the training set. Zhu and Hovy [24] incorporate over- and under-sampling in active learning for word sense disambiguation. Their algorithm uses active learning to select samples for human experts to label, and then re-samples this subset. In their experiments, under-sampling caused negative effects but over-sampling helps increase balancedness. However, both [24] and [23] are primarily designed and applied to

binary classification problems for text and are hard to generalize to multi-class problems and non-text domains.

Our approach is iterative like active learning, but it differs crucially in that it relies on semi-supervised clustering instead of classification and selects target samples based on estimated class distribution in each cluster. This makes it more general where the best classifier is not known in advance. Ours is the first attempt at using active learning with semi-supervised clustering instead of classification and thus does not suffer from over-fitting. Furthermore, since most classification methods require the presence of at least two different classes in the training set, there is a challenge in providing the initial labeling sample for active learning; an arbitrary insertion of instances from at least two classes is required. Our method does not have this limitation, and, although not shown in the experiments, performs as well with a random initial sample. Our work provides a general framework which is domain independent and can be easily customized to specific domains.

3 Generating Balanced Training Data

Our strategy for generating balanced training sets are described in this section. Section 3.1 presents a high level overview of the algorithm, and later sections provide a more formal description and specific instantiations of the key steps and discuss various tradeoffs.

3.1 Overview

Given an unlabeled dataset with unknown class distribution, potentially skewed, our goal is to produce balanced labeled samples for training predictive models. Formally, we can define the balanced training set problem as follows:

Definition 1. Let $\mathcal{D}$ be an unlabeled data set containing ℓ classes from which we wish to select $\mathcal{T}$, a subset of $\mathcal{D}$ of size N. Let $\mathcal{L}(T)$ be the labels of the training data set T, then the balancedness of $\mathcal{T}$ is defined as the distance between the label distribution of $\mathcal{L}(T)$ and the discrete uniform distribution with ℓ classes, i.e., $D(\mathsf{Uniform}(\ell) \parallel \mathsf{Multi}(\mathcal{L}(T)))$. The balanced training set problem is the problem of finding a training data set that minimizes this distance.

If we know the class labels in a given set, then we can use over- and under-sampling to draw balanced sample set [21, 22, 25]. However, the class labels are not known, so instead we must use a series of approximations to approach the results of this ideal solution. We apply an iterative semi-supervised clustering algorithm to estimate the class distribution in the unlabeled data set and guide the sample selection to produce a balanced set. In each iteration, the algorithm draws a batch of samples ($\mathcal{B}$), and domain experts provide the labels of the selected samples. The labeled samples are used in subsequent iterations.

Algorithm 1 is a high level description of our strategy. It takes three inputs: $\mathcal{D}$, an unlabeled data set; ℓ, the number of target classes in $\mathcal{D}$; and N, the number of training samples to generate. We note that the value of the input parameters

TrainingSetGeneration$(\mathcal{D}, \ell, N, [\mathcal{B}])$
if $\mathcal{B}$ *is undefined* **then**
  $\mathcal{B} \leftarrow \min\left(\left\lfloor \frac{|\mathcal{D}|}{10} \right\rfloor, \frac{N}{10}\right)$;
end
$\max_{cluster} \leftarrow \left\lfloor \frac{|\mathcal{D}|}{10} \right\rfloor$;
$\mathcal{T} \leftarrow \mathcal{L}(\mathcal{B}$ randomly selected samples);
while $|\mathcal{T}| < N$ **do**
  $\{C_1, \ldots, C_k\} \leftarrow$ *SemiSupervisedClustering*$(\mathcal{D}, \mathcal{T}, \max_{cluster})$;
  $\mathcal{T}' \leftarrow \emptyset$;
  foreach $j = 1$ **to** k **do**
    $num_j \leftarrow$ *DetermineOptimalNumberToSample*(C_j);
    $\mathcal{T}_j \leftarrow$ *MaximumEntropySampling*(C_j, num_j);
    $\mathcal{T}' \leftarrow \mathcal{T}' \cup \mathcal{T}_j$;
  end
  $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{L}(\mathcal{T}')$;
end

Algorithm 1. High level steps of the proposed algorithm

are previously determined in most applications. The number of samples to select in each iteration, $\mathcal{B}$, can be determined automatically based on $\mathcal{D}$ and N, or users can optionally set the batch size as an input parameter.

To start, we select $\mathcal{B}$ samples randomly and obtain the labels. Then, a semi-supervised clustering algorithm is applied to embed the labels obtained from the prior steps into the clustering process (Section 3.2), which can be used to approximate the class distributions in the clusters. The key intuition behind the process is that we want to extract more samples from clusters which are likely to increase the balancedness of the overall training set. Our algorithm tries to infer the class distribution of each cluster and use this to over- or under-sample. Section 3.3 describes key details of the class density estimation process, the various tradeoffs and their influence on the ultimate results. Once we determine how much to sample from each cluster, we obtain diverse samples using maximum entropy sampling (Section 3.5). We note that there is an implicit secondary optimization of maximizing the entropy of the sampled points, $\mathcal{H}(\mathcal{T})$, which is the byproduct of the real objective, maximizing the performance of a classifier trained on $\mathcal{T}$.

3.2 Semi-supervised Clustering

Semi-supervised clustering is a technique which incorporates existing information into clustering. A number of approaches have been proposed to embed constraints into existing clustering algorithms [9, 26]. We explore two different strategies: a distance metric technique for multi-variate numeric data and a heuristic to add class labels in the feature set for categorical data. We use Relevant Component Analysis (RCA) [7] for distance metric-based semi-supervised clustering, This is a Mahalanobis metric technique which finds a new space with

the most relevant features in the side information. It learns a global distance metric parameterized by a transformation matrix $\hat{C}$ to capture relevant features in the labeled sample set. It maximizes the similarity between the original data set X and the new representation Y constrained by the mutual information $I(X, Y)$. By projecting X into the new space through transformation $Y = \hat{C}^{-\frac{1}{2}}X$, two projected data objects, Y_i, Y_j, in the same connected component have a smaller distance.

Here, we sketch the steps to compute the "within-chunklet" covariance matrix (transformation matrix), $\hat{C}$. Given a data set $X = \{x_i\}_{i=1}^{N}$ and a labeled sample set $L \subset X$, suppose u connected components (i.e., chunklets) $M = \{M_j\}_{j=1}^{u}$ are obtained based on L, which satisfies $X = \bigcup_{i=1}^{u} M_j$. Let the data points in a component M_j be denoted as $\{x_{ji}\}_{i=1}^{|M_j|}$ for $1 \leq j \leq u$. Then, the covariance matrix $\hat{C}$ is defined by Equation 1, where m_j is the centroid of M_j.

$$\hat{C} = \frac{1}{N} \sum_{j=1}^{u} \sum_{i=1}^{|M_j|} (x_{ji} - m_j)(x_{ji} - m_j)^T \tag{1}$$

After projecting the data set into a new space using RCA, the data set is recursively partitioned until all the clusters are smaller than a predetermined threshold, $\max_{cluster}$. Algorithm 2 summarizes our semi-supervised clustering algorithm using RCA.

SemiSupervisedClustering$(X, \mathcal{L}(T), \max_{cluster})$
$\hat{C} = RCA(X, \mathcal{L}(T))$;
$Y = \hat{C}^{-\frac{1}{2}}X$;
$C \leftarrow \{C_1, C_2\} = BinaryClustering(Y)$;
while $\exists C_i \in C, |C_i| > \max_{cluster}$ **do**
  $\{C_{i1}, C_{i2}\} \leftarrow BinaryClustering(C_i)$;
  $C \leftarrow \big(C \setminus C_i\big) \cup \{C_{i1}, C_{i2}\}$;
end

Algorithm 2. Semi-supervised clustering algorithm to divide a data set into balanced clusters

The RCA algorithm makes several assumptions regarding the distribution of the data. Primarily, it assumes the data is multivariate normally distributed, and, if so, produces the optimal result. It has also been shown to perform well on datasets when the normally distributed assumption fails [7], including many of the UCI datasets used in this work. However, it is not known to work well for Bernoulli or categorical distributed data, such as the access control datasets, where it produces a marginal improvement, at best. Instead, we choose a simple method by augmenting the feature set with labels of known samples, i.e., $\mathcal{F} \parallel \mathcal{L}$, and assigning a default feature value, or holding out feature values, for unlabeled samples. For example, if we have ℓ class labels, we will add ℓ new binary features. If the sample has class j, we will assign feature j a value of 1, and all other label features a zero. Unlabeled samples are assigned a feature corresponding to the

prior, the fraction of labeled samples with that class label. As before, we use the recursive binary clustering technique described previously to cluster the data. We find that this simple heuristic produces good clusters and yields balanced samples more quickly for categorical data.

3.3 Determine the Optimal Number to Samples from Each Cluster

Once we have clustered the data, the key step is to estimate the class density in the clusters and use this information to perform biased sampling to increase the overall balancedness in the sample set. We assume that the semi-supervised clustering step has produced biased clusters allowing us to approximate a solution of drawing samples with known classes.

The first approach is to assume that the class distribution of the cluster is exactly the same as the class distribution of the labeled samples in this cluster. This is based on the optimistic assumption that the semi-supervised clustering works perfectly and groups together elements similar to the labeled sample. First, we determine how many samples we wish to draw from each class in this iteration from the total $\mathcal{B}$ samples to draw. Let ℓ_i^j be the number of instances of class j sampled after iteration i, and ρ_i^j be the normalized proportion of samples with class label j, i.e., $\rho_i^j = \frac{\ell_i^j}{\sum_r \ell_i^r}$. To increase balancedness, we want to sample inverse proportionally to their current distribution [21, 22, 25], i.e., $n_j = \frac{1-\rho_i^j}{\ell-1} * \mathcal{B}$, where ℓ is the number of classes. Next, we use the estimated class distribution in each cluster to determine the appropriate number of samples to draw from each class. Let θ_i^j be the probability of drawing a sample with class label j from the *previously labeled subset* of cluster i. By our assumption, this is exactly the probability of drawing a sample with class label j from the *entire* cluster. To sample n_j samples with label j, we draw $n_j \frac{\theta_i^j}{\sum_{l=1}^{k} \theta_l^j}$ samples from cluster i, where k is the number of clusters. Another strategy is to draw all n_j samples from the cluster with the maximum probability of drawing class j, however our method selects a more representative subset, and we can obtain good results even if our estimation of cluster densities is incorrect and reduces later classifier over-fitting.

In early stages of the iteration process, where there are few labeled samples, this approach does not work well. We use a hybrid approach where we select a certain percentage of $\mathcal{B}$ samples based on the estimates of class distribution and select remaining samples randomly from all clusters. We increase the influence of labeled samples over time as we obtain more labeled samples and thus better estimates on class distribution. Let $\mathcal{B}_\mathcal{L}$ be the number of samples to select based on labeled samples, and $\mathcal{B}_r$ be the number of samples to select randomly. Then, we compute $\mathcal{B}_\mathcal{L} = \omega \cdot \mathcal{B}$ and $\mathcal{B}_r = (1-\omega) \cdot \mathcal{B}$ using a sigmoid function $\omega = \frac{1}{1+e^{-\lambda t}}$, where t is the iteration number and λ a parameter that controls the rate of mixing. As t increases (i.e., number of labeled samples increases), we decay the influence of random sampling favoring empirical estimates.

3.4 Leveraging Domain Knowledge

In Section 3.3, we presented a hybrid sampling method that combines sampling based on the class distribution of each cluster and random sampling. In many settings, domain experts may have additional knowledge regarding the distribution of class labels and correlations with given features or feature values. For instance, in the problem of detecting malicious web sites, there is a heavy skew in geographical location of the web servers [10]. In access control datasets, one can expect correlations between the department of the employee and granted permissions. This section outlines a method where we can incorporate such domain knowledge to estimate the class distribution within a cluster. We use domain knowledge in the form of a correlation value between a feature and a class label. For example, $\mathsf{corr}(\mathit{Department} = 20, \mathit{class} = \mathit{grant}) = +0.1$. These correlations may be noisy and incomplete, pertaining to only a small number of features or feature values. Without loss of generality, we will only consider binary labels; the technique can readily be extended to non-binary labels.

Given a small number of feature-class and feature-value-class correlations and the feature distribution within a cluster, we can estimate the class density. We leverage some of the ideas from the MYCIN model of inexact reasoning [27]. They note that domain knowledge is often logically inconsistent and non-Bayesian. For example, given expert knowledge that $p\,(\mathit{class} = \mathit{grant} \mid \mathit{Department} = 20) = 0.6$, we *cannot* conclude that $p\,(\mathit{class} \neq \mathit{grant} \mid \mathit{Department} = 20) = 0.4$. Further, a naïve Bayesian approach requires an estimation of the global class distribution, which we assume is not known a priori. Instead, our approach is based on independently aggregative suggestive evidence and leverages properties from fuzzy logic. The correlations correspond to inference rules (e.g., $\mathit{Department} = 20 \rightarrow \mathit{class} \neq \mathit{grant}$), where the correlation coefficients are the confidence weights of the inference rules, and the feature density within each class is the degree that the given inference rule is fired. We evaluate each inference rule in support (positive correlation) and refuting (negative correlation) the class assignments, and aggregate the results using the Product T-Conorm, $\mathsf{norm}(x, y) = x + y - x * y$. We combine evidence supporting and refuting a class assignment using the rule "class 1 and not class 2," and T-Norm for conjunction, $f(x, y) = x * (1 - y)$. Finally, we use domain knowledge-based estimates to supplement the empirical estimates $\mathcal{B}_{\mathcal{L}}$. Let $\mathcal{B}_d$ be the number of samples to select based on domain knowledge, then we select $\mathcal{B} = \mathcal{B}_{\mathcal{L}} + \mathcal{B}_d$ samples in each iteration. As domain knowledge is inexact and noisy, we decay the influence of its estimates over time, favoring the empirical estimates using the sigmoid ω described in Section 3.3, i.e., $\mathcal{B}_d = (1 - \omega) \cdot \mathcal{B}$.

3.5 Maximum Entropy Sampling

Finally, given the set of clusters $\{C_i\}_{i=1}^{k}$, and the number of samples to select from each cluster, we sample to maximize the entropy of the sample $\mathcal{L}(T)$. We assume that the data in each cluster follows a Gaussian distribution. For a continuous variable $x \in C_i$, let the mean be μ, and the standard deviation be

σ, then the normal distribution $\mathcal{N}(\mu, \sigma^2)$ has maximum entropy among all real-valued distributions. The entropy for a multivariate Gaussian distribution [28] is defined as:

$$\mathcal{H}(X) = \frac{1}{2} d\,(1 + \log\,(2\pi)) + \frac{1}{2} \log\,(|\Sigma|) \tag{2}$$

where d is the dimension, Σ the covariance matrix, and $|\Sigma|$ the determinant of Σ. Thus, more variation the covariance matrix has along the principal directions, the more information it embeds.

Note that the number of possible subsets of r elements from a cluster C can grow very large (i.e., $\binom{|C|}{r}$), so finding a subset with the global maximum entropy can be computationally very intensive. We use a greedy method that selects the next sample which adds the most entropy to the existing labeled set. Our algorithm performs the covariance calculation $O(rn)$ times, while the exhaustive search approach requires $O(n^r)$. If there are no previously labeled samples, we start the selection with the two samples that have the longest distance in the cluster. The maximum entropy-based sampling method is presented in Algorithm 3.

MaximumEntropySampling$(\mathcal{T}, C, num)$
$C_U \leftarrow$ unlabeled samples in C;
$\mathcal{T}_C \leftarrow \emptyset$;
while $|\mathcal{T}_C| < num$ **do**
  $u \leftarrow \arg\max_{u_i \in C_U} \mathcal{H}(\mathcal{T} \cup \{u_i\})$;
  $\mathcal{T}_C \leftarrow \mathcal{T}_C \cup \{u\}$;
  $C_U \leftarrow C_U \setminus \{u\}$;
end
Return $\mathcal{T} \cup \mathcal{T}_C$

Algorithm 3. Maximum entropy sampling strategy

4 Experiments and Evaluation

This section presents a performance comparison of our sampling strategies with random sampling and uncertainty based sampling on a diverse collection of data sets. Our results show that our algorithm produces significantly more balanced sets than random sampling in almost all datasets. Our technique also performs much better than uncertainty based sampling for highly skewed sets, and our training samples can be used to train any classifier. We also describe results which confirm the benefits of domain knowledge.

4.1 Evaluation Setup

To evaluate the sampling strategies, we selected 14 data sets from the UCI repository [11] and a private data set containing the assignment of access control permissions to users. The data sets span a wide range of parameters and are

Table 1. Description, size, and distribution of experimental data sets

Data	#Samples	#Classes	Class Distribution
Breast Cancer	699	2	65.52% vs. 34.48%
Ionosphere	351	2	35.90% vs. 64.10%
KDD'99	49,180	2	97.35 vs. 2.65%
Page Blocks	5,028	2	97.71% vs. 2.29%
Pima Indians	768	2	65.10% vs. 34.90%
Breast Cancer (BC) Wisconsin	198	2	76.26% vs. 23.74%
Spambase	4,601	2	60.60% vs. 39.40%
SPECT	267	2	79.40% vs. 20.60%
Iris	150	3	balanced
Wine	178	3	39.89% 26.97% 33.15%
Statlog (Landsat Satellite)	6,435	6	23.82% to 9.73%
Pen Digits	10,992	10	balanced
Poker Hand	25,010	10	Two major (42.4%–50%) & eight minor (4.8%–0.02%)
Letter Recognition	20,000	26	balanced
Access Permission	3,068	2	91.72% vs. 8.28%

summarized in Table 1: some are highly skewed while others are balanced, some are multi-class while others are binary.

All UCI data sets are used unmodified except the *KDD Cup'99* set which contains a "normal" class and 20 different classes of network attacks. In this experiment, we selected only "normal" class and "guess_password" class to create a highly skewed data set. When a data set is provided with a training set and a test set separately (e.g., 'Statlog'), we combined the two sets. The features in the access control data set are typically organization attributes of a user: department name, job roles, whether the employee is a manager, etc. These categorical features are converted to binary features. Since, such access control permissions are assigned based on a combination of attributes, these data sets are also useful to assess the benefits of domain knowledge.

For each data set, we randomly select 80% of the data to be used as unlabeled data, from which training samples are generated. The remaining 20% of the samples is used to test classifiers trained with the training samples. For uncertainty-based active learning, we use three widely used classification algorithms, Naive Bayes, Logistic Regression, and SVM, and these variants are labeled *Un_Naive*, *Un_LR*, and *Un_SVM* respectively. We used the C-support vector classification (C-SVC) SVM with a radial basis function (RBF) kernel, and Logisitc Regression with RBF kernel. All classification experiments were conducted using *RapidMiner*, an open source machine learning tool kit [29]. Logistic Regression in *RapidMiner* only supports binary classification, and thus it was extended to a multi-class classifier using "one-against-all" strategy for multi-class data sets [30]. All experimental results reported here are the average of 10 runs of the experiments.

Table 2. Distance of the sampled class distributions to the uniform distribution. The best performing algorithm for each data set is highlighted in bold.

Distribution	Data	Sample Size	*Random*	*Un_Naive*	*Un_LR*	*Un_SVM*	*Our*
Highly Skewed	Poker Hand	2,000	0.574	0.570	**0.540**	0.557	**0.540**
	Page Blocks	2,000	0.676	0.657	0.646	0.656	**0.642**
	KDD'99	2,000	0.670	0.680	0.291	0.479	**0.282**
	Access Permission	2,000	0.594	0.592	0.535	0.492	**0.472**
Mildly Skewed	Statlog	2,000	0.150	0.247	0.067	**0.049**	0.118
	SPECT	110	0.427	0.419	0.278	**0.212**	0.320
	BC Wisconsin	80	0.361	**0.318**	0.359	0.377	0.324
	Wine	75	0.114	0.184	0.046	0.077	**0.039**
	Breast Cancer	280	0.229	0.190	0.170	0.201	**0.028**
	Pima Indians	310	0.215	0.083	**0.007**	0.056	0.086
	Ionosphere	140	0.208	0.140	**0.061**	0.089	0.076
	Spambase	1,845	0.151	0.199	**0.018**	0.048	0.093
Uniform	Iris	60	0.055	0.335	0.078	0.401	**0.051**
	Letter Recognition	2,000	**0.020**	0.129	0.094	0.137	0.069
	Pendigits	2,000	**0.020**	0.238	0.060	0.064	0.084

4.2 Comparison of Class Distribution in Training Samples

We first evaluate the five sampling methods by comparing the balancedness of the generated training sets. For each run, we continue sampling till the selected training sample contains 50% of the unlabeled samples or we have 2,000 samples, whichever is smaller. The evaluation metrics we use are the balancedness of the training data and the recall of the minority class. As noted above, each run is done with a random 80% of the underlying data set and the results are averaged over 10 runs. We measure the balancedness of a data set as the distance of the sampled class distribution from the uniform class distribution as defined in Definition 2.

Definition 2. *Let X be a data set with k different classes. Then the uniform distribution over X is the probability density function (PDF), $U(X)$, where $U_i = \frac{1}{k}$, for all $i \in k$. Let $P(X)$ be a PDF over the classes produced by a sampling method. Then the balancedness of the sample is defined as the Euclidean distance between the distributions $U(X)$ and $P(X)$* i.e. $d = \sqrt{\sum_{i=1}^{k}(U_i - P_i)^2}$.

Table 2 summarizes the results of balancedness comparison, and Table 3 shows the recall of minority class for all the data sets respectively. Our method produces significantly better results compared to pure random sampling. On *KDD'99*, our sampling algorithm yields 10x more minority samples on average than random. Similarly for *Page Blocks* and the access permission data set, our method produces about 2x more balanced samples. For mildly skewed data sets, our method also produces about 25% more minority samples on the average. For the data sets which are almost balanced, random is the best strategy as expected. Even in this case, our method produces results which are statistically very close to random. Thus, our method is *always* preferable to random sampling.

Table 3. The recall rates for the binary class data sets. *Min. Ratio* refers to the ratio of the minority class in the unlabeled data set. For the access permission data, the average and the standard deviation over multiple data sets are reported.

Data	*Min. Ratio*	*Random*	*Un_Naive*	*Un_LR*	*Un_SVM*	*Our*
Poker Hand	0.02	15.00	0.00	0.00	0.00	**62.50**
Page Blocks	2.29	47.61	77.07	93.91	77.83	**100.00**
KDD'99	2.65	5.06	3.63	56.53	30.89	**57.70**
Access Permission	8.04 (5.98)	24.82	25.39 (5.60)	46.32 (22.34)	**60.79 (23.87)**	58.83 (26.31)
Statlog	9.73	39.98	**59.18**	49.84	49.80	35.85
SPECT	20.56	48.18	50.91	75.91	**87.50**	66.59
BC Wisconsin	23.90	51.58	52.89	51.84	49.21	**57.11**
Wine	27.03	46.15	42.05	**60.77**	52.82	57.95
Breast Cancer	34.46	49.07	53.06	41.30	51.97	**75.44**
Pima Indians	34.96	48.56	63.63	**71.77**	64.84	62.28
Ionosphere	35.94	48.88	55.54	**64.95**	60.59	62.28
Spambase	39.41	49.91	45.66	62.00	**67.85**	55.12

Since uncertainty based sampling methods are targeted to cases where the classifier to be trained is known, the right comparison with these methods should include the performance of the resulting classifiers. Further, these algorithms are not very efficient due to re-training at each step. With these caveats, we can directly compare the balancedness of the results. For highly skewed data sets, our method performs better especially when compared to *Un_SVM* and *Un_Naive* methods. On *KDD'99*, we produce 20x and 2x more minority samples compared to *Un_Naive* and *Un_SVM* respectively, while *Un_LR* performs almost as well as our algorithm. Similarly for *Page Blocks*, we perform about 20% better than these methods. We note that our method found all minority samples for all 10 split sets for the *Page Blocks* set. For other data sets, our algorithm shows no significant statistical difference compared to these methods on almost all cases and sometimes we do better. Based on these results, we also conclude that our method is preferable to the uncertainty-based methods based on broader applicability and efficiency.

Figure 1 pictorially depicts the performance of our sampling algorithm as well as the uncertainty based sampling for a few data sets to highlight cases where our method performs better. These figures show the distance from uniform against the percentage of sampled data over iterations. The results show that our sampling technique consistently converges towards balancedness while there is some variation with uncertainty techniques, which remains true for other data sets as well. Note that the distance increases in *Page Blocks* and *Access Permission* data sets after 20% point is because our method exhausted all minority samples.

4.3 Comparison of Classification Performance

In this section, we evaluate the quality of the training samples by comparing the performance of classifiers trained on them. We apply the training samples from the five strategies to train the same type of classifiers (Naive, LR, and SVM), resulting in 15 different "training-evaluation" scenarios. Due to space limitations, we present in Table 4 the AUC and F1-measure for binary class datasets.

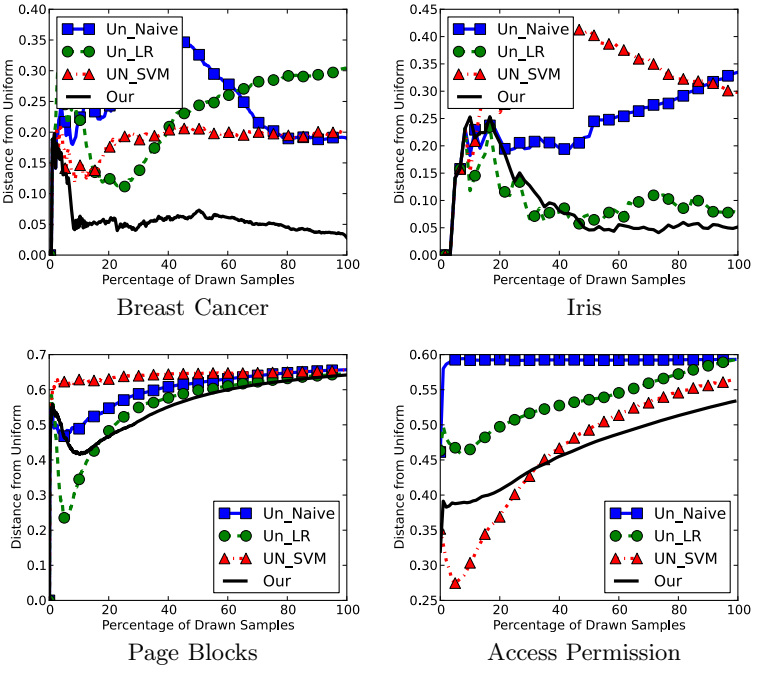


Fig. 1. Progress of balancedness increase over iterations

We expect the performance of the uncertainty sampling methods paired with their respective classifier, e.g., *Un_SVM* with SVM and *Un_LR* with Logistic Regression, to perform best. We observe this behavior on KDD and PIMA, but the off-diagonal entries for uncertainty based sampling show poor results. However, on other datasets such as Breast Cancer and SPECT data sets, our approach outperforms the competing uncertainty sampling. Furthermore, our method performs well consistently across all classifiers without being biased to a single classifier and at reduced computation cost.

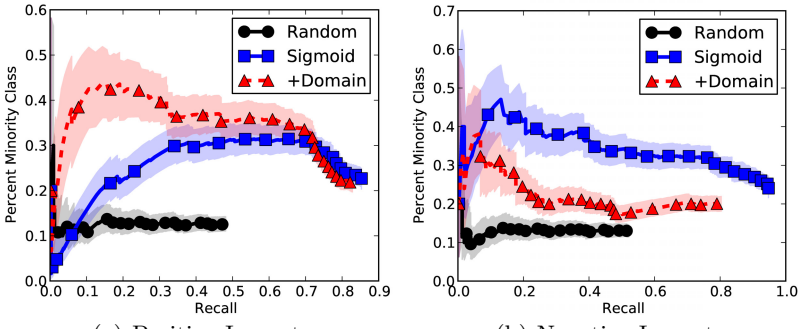
4.4 Impact of Domain Knowledge

The access control permission data sets are used to evaluate the benefit of additional domain knowledge given as a correlation of a user's attributes (e.g., department number, whether she is a manager, etc.) and the granted permission. Our evaluation of sampling with domain knowledge shows that domain knowledge (almost) always helps. There are a few cases where adding domain knowledge negatively impacts performance as shown in Fig 2(b). However, in most cases, domain knowledge substantially improves the convergence of the algorithm. The example depicted in Figure 2(a) is typical of the access control datasets. Since such domain knowledge is mostly used in the early iterations, it significantly helps speed up the convergence.

Table 4. Performance comparison of the three classifiers trained with five different sampling techniques. The figures in bold denote the best performing sampling technique for each classifier. The figures in italics denote the best performing classifier excluding the uncertainty sampling methods paired with their respective classifier.

Data Set	Breast Cancer						SPECT					
Classifier	Naive		LR		SVM		Naive		LR		SVM	
Sampling	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Random	0.959	93.52	0.982	91.16	0.549	4.41	***0.796***	50.38	0.713	34.96	0.823	49.12
Un_Naive	0.970	94.04	0.979	76.78	0.541	4.74	0.777	47.97	0.734	43.15	0.812	48.12
Un_LR	0.973	94.06	0.971	61.31	0.524	2.40	0.760	51.29	0.670	45.65	0.818	59.00
Un_SVM	0.961	94.27	0.979	91.17	0.542	2.84	0.787	53.32	0.634	45.65	0.833	53.54
Our	***0.987***	***94.41***	***0.985***	***91.67***	***0.552***	***52.92***	0.766	***54.29***	***0.736***	***45.67***	***0.839***	***56.47***

Data Set	Pima Indians						KDD					
Classifier	Naive		LR		SVM		Naive		LR		SVM	
Sampling	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Random	0.785	*60.81*	0.771	11.17	0.658	22.54	0.972	61.94	***0.999***	46.74	0.978	79.91
Un_Naive	**0.808**	**61.14**	0.745	26.53	0.593	28.93	0.964	***63.30***	0.997	21.56	0.954	69.07
Un_LR	0.800	60.10	0.768	**55.93**	0.611	***41.19***	0.905	28.56	**0.999**	**96.81**	***0.998***	*97.37*
Un_SVM	0.785	60.15	0.761	44.02	0.610	31.18	***0.979***	59.24	0.998	*92.74*	0.988	**98.10**
Our	*0.805*	59.5	***0.808***	*51.62*	***0.649***	32.78	0.910	30.54	***0.999***	90.90	0.990	90.68



(a) Positive Impact (b) Negative Impact

Fig. 2. Comparison of our algorithm ('Sigmoid'), our algorithm with domain knowledge ('+Domain'), and random sampling ('Random'). The y-axis shows the minority class density in the training data, and x-axis shows the recall of the minority class.

5 Conclusion

In this paper, we considered the problem of generating a training set that can optimize the classification accuracy and also is robust to classifier change. We confirmed through experiments that our method produces very balanced training data for highly skewed data sets and outperforms other methods in correctly classifying the minority class. For a balanced multi-class problem, our algorithm outperforms active learning by a large margin and works slightly better than random sampling. Furthermore, our algorithm is much faster compared to ac-

tive sampling. Therefore, the proposed method can be successfully applied to many real-world applications with highly unbalanced class distribution such as malware detection or fraud detection. In future work, we plan to apply kernel methods for semi-supervised clustering which can discover clusters with non-linear boundaries in the original space to better fit nonlinearly separable data.

Acknowledgement. This research is continuing through participation in the Anomaly Detection at Multiple Scales (ADAMS) program sponsored by the U.S. Defense Advanced Research Projects Agency (DARPA) under Agreement Number W911NF-11-C-0200.

References

1. Jo, T., Japkowicz, N.: Class imbalances versus small disjuncts. SIGKDD Explorations 6(1) (2004)
2. Weiss, G., Provost, F.: The effect of class distribution on classifier learning: An empirical study. Dept. of Comp. Science, Rutgers University, Tech. Rep. ML-TR-43 (2001)
3. Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: ICML (2004)
4. Dasgupta, S., Hsu, D.: Hierarchical sampling for active learning. In: ICML (2008)
5. Ertekin, S., Huang, J., Bottou, L., Giles, C.L.: Learning on the border: active learning in imbalanced data classification. In: CIKM (2007)
6. Settles, B.: Active learning literature survey. University of Wisconsin-Madison, Tech. Rep. (2009)
7. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning a mahalanobis metric from equivalence constraints. Journal of Machine Learning Research 6 (2005)
8. Wagstaff, K., Cardie, C.: Clustering with instance-level constraints. In: ICML (2000)
9. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning, with application to clustering with side-information. In: Advances in Neural Info. Proc. Systems, vol. 15. MIT Press (2003)
10. Provos, N., Mavrommatis, P., Rajab, M., Monrose, F.: All your iFRAMEs point to us. Google, Tech. Rep. (2008)
11. Frank, A., Asuncion, A.: UCI machine learning repository
12. Campbell, C., Cristianini, N., Smola, A.J.: Query learning with large margin classifiers. In: ICML (2000)
13. Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Machine Learning 28(2-3) (1997)
14. Tong, S., Koller, D.: Support vector machine active learning with application sto text classification. In: ICML (2000)
15. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: SIGIR (1994)
16. Seung, H.S., Opper, M., Sompolinsky, H.: Query by committee. In: Computational Learning Theory (1992)
17. Hoi, S.C.H., Jin, R., Zhu, J., Lyu, M.R.: Batch mode active learning and its application to medical image classification. In: ICML (2006)

18. Guo, Y., Schuurmans, D.: Discriminative batch mode active learning. In: NIPS (2007)
19. Schohn, G., Cohn, D.: Less is more: Active learning with support vector machines. In: ICML (2000)
20. Xu, Z., Hogan, C., Bauer, R.: Greedy is not enough: An efficient batch mode active learning algorithm. In: ICDM Workshops (2009)
21. Liu, X.Y., Wu, J., Zhou, Z.H.: Exploratory undersampling for class imbalance learning. In: IEEE Trans. on Sys. Man. and Cybernetics (2009)
22. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority over-sampling technique. JAIR 16 (2002)
23. Tomanek, K., Hahn, U.: Reducing class imbalance during active learning for named entity recognition. In: K-CAP (2009)
24. Zhu, J., Hovy, E.: Active learning for word sense disambiguation with methods for dddressing the class imbalance problem. In: EMNLP-CoNLL (2007)
25. wU, Y., Zhang, R., Rudnicky, E.: Data selection for speech recognition. In: ASRU (2007)
26. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.: Constrained k-means clustering with background knowledge. In: ICML (2001)
27. Shortliffe, E.H., Buchanan, B.G.: A model of inexact reasoning in medicine. Mathematical Biosciences 23(3-4) (1975)
28. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley Interscience (1991)
29. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Proc. KDD (2006)
30. Rifkin, R.M., Klautau, A.: In defense of one-vs-all classification. J. Machine Learning (1998)