# 6

# CLASS IMBALANCE AND ACTIVE LEARNING

JOSH ATTENBERG AND ŞEYDA ERTEKIN

**Abstract:** The performance of a predictive model is tightly coupled with the data used during training. While using more examples in the training will often result in a better informed, more accurate model; limits on computer memory and real-world costs associated with gathering labeled examples often constrain the amount of data that can be used for training. In settings where the number of training examples is limited, it often becomes meaningful to carefully see just which examples are selected. In *active learning* (AL), the model itself places a hands-on role in the selection of examples for labeling from a large pool of unlabeled examples. These examples are used for model training. Numerous studies have demonstrated, both empirically and theoretically, the benefits of AL: Given a fixed budget, a training system that interactively involves the current model in selecting the training examples can often result in a far greater accuracy than a system that simply selects random training examples. Imbalanced settings provide special opportunities and challenges for AL. For example, while AL can be used to build models that counteract the harmful effects of learning under class imbalance, extreme class imbalance can cause an AL strategy to "fail," preventing the selection scheme from choosing any useful examples for labeling. This chapter focuses on the interaction between AL and class imbalance, discussing (i) AL techniques designed specifically for dealing with imbalanced settings, (ii) strategies that leverage AL to overcome the deleterious effects of class imbalance, (iii) how extreme class imbalance can prevent AL systems from selecting useful examples, and alternatives to AL in these cases.

*Imbalanced Learning: Foundations, Algorithms, and Applications*, First Edition.
Edited by Haibo He and Yunqian Ma.

## 6.1 INTRODUCTION

The rich history of predictive modeling has been culminated in a diverse set of techniques capable of making accurate predictions on many real-world problems. Many of these techniques demand *training*, whereby a set of instances with ground-truth *labels* (values of a dependent variable) are observed by a model-building process that attempts to capture, at least in part, the relationship between the features of the instances and their labels. The resultant model can be applied to instances for which the label is not known, to estimate or predict the labels. These predictions depend not only on the functional structure of the model itself, but also on the particular data with which the model was trained. The accuracy of the predicted labels depends highly on the model's ability to capture an unbiased and sufficient understanding of the characteristics of different classes; in problems where the prevalence of classes is imbalanced, it is necessary to prevent the resultant model from being skewed toward the majority class and to ensure that the model is capable of reflecting the true nature of the minority class.

Another consequence of class imbalance is observed in domains where the ground-truth labels in the dataset are not available beforehand and need to be gathered on- demand at some cost. The costs associated with collecting labels may be due to human labor or is the result of costly incentives, interventions, or experiments. In these settings, simply labeling all available instances may not be practicable because of the budgetary constraints or simply a strong desire to be cost efficient. As in predictive modeling with imbalanced classes, the goal here is to ensure that the budget is not predominantly expended on getting the labels of the majority class instances, and to make sure that the set of instances to be labeled have comparable number of minority class instances as well.

In the context of learning from imbalanced datasets, the role of active learning (AL) can be viewed from two different perspectives. The first perspective considers the case where the labels for all the examples in a reasonably large, imbalanced dataset are readily available. The role of AL in this case is to reduce, and potentially eliminate, any adverse effects that the class imbalance can have on the model's generalization performance. The other perspective addresses the setting where we have prior knowledge that the dataset is imbalanced, and we would like to employ AL to select informative examples both from the majority and minority classes for labeling, subject to the constraints of a given budget. The first perspective focuses on AL's ability to address class imbalance, whereas the second perspective is concerned with the impact of class imbalance on the sampling performance of the active learner. The intent of this chapter is to present a comprehensive analysis of this interplay of AL and class imbalance. In particular, we first present techniques for dealing *with* the class imbalance problem using AL and discuss how AL can alleviate the issues that stem from class imbalance. We show that AL, even without any adjustments to target class imbalance, is an effective strategy to have a balanced view of the dataset in most cases. It is also possible to further improve the effectiveness of AL by tuning its sampling strategy in a class-specific way. Additionally, we will focus on dealing with highly

skewed datasets and their impact on the selections performed by an AL strategy. Here, we discuss the impact significant class imbalance has on AL and illustrate alternatives to traditional AL that may be considered when dealing with the most difficult, highly skewed problems.

## 6.2 ACTIVE LEARNING FOR IMBALANCED PROBLEMS

The intent of this section is to provide the reader with some background on the AL problem in the context of building cost-effective classification models. We then discuss challenges encountered by AL heuristics in settings with significant class imbalance. We then discuss the strategies specialized in overcoming the difficulties imposed by this setting.

### 6.2.1 Background on Active Learning

AL is a specialized set of machine learning techniques developed for reducing the annotation costs associated with gathering the training data required for building predictive statistical models. In many applications, *unlabeled* data comes relatively cheaply when compared to the costs associated with the acquisition of a ground-truth value of the target variable of that data. For instance, the textual content of a particular web page may be crawled readily, or the actions of a user in a social network may be collected trivially by mining the web logs in that network. However, knowing with some degree of certainty the topical categorization of a particular web page, or identifying any malicious activity of a user in a social network is likely to require costly editorial review. These costs restrict the number of examples that may be labeled, typically to a small fraction of the overall population. Because of these practical constraints typically placed on the overall number of ground-truth labels available and the tight dependence of the performance of a predictive model on the examples in its training set, the benefits of careful selection of the examples are apparent. This importance is further evidenced by the vast research literature on the topic.

While an in-depth literature review is beyond the scope of this chapter, for context we provide a brief overview of some of the more broadly cited approaches in AL. For a more thorough treatment on the history and details of AL, we direct the reader to the excellent survey by Settles [1]. AL tends to focus on two scenarios—(i) stream-based selection, where unlabeled examples are presented one at a time to a predictive model, which feeds predicted target values to a consuming process and subsequently applies an AL heuristic to decide whether some budget should be expended gathering this example's class label for subsequent re-training. (ii) pool-based AL, on the other hand, is typically an offline, iterative process. Here, a large set of unlabeled examples are presented to an AL system. During each epoch of this process, the AL system chooses one or more unlabeled examples for labeling and subsequent model training. This proceeds until the budget is exhausted or some stopping criterion is met. At this time, if the

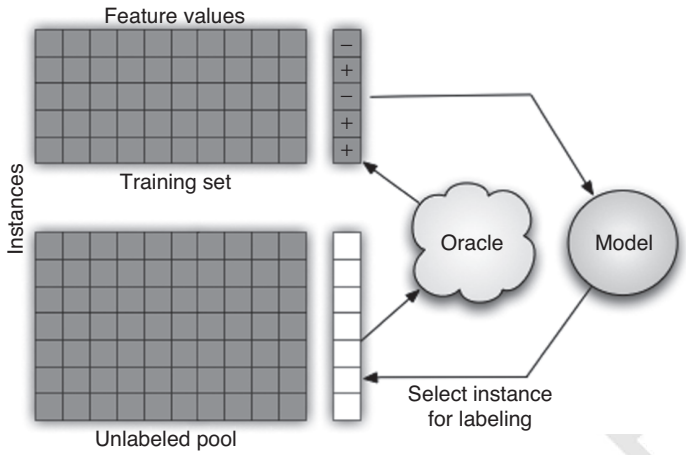


**Figure 6.1** Pool-based active learning.

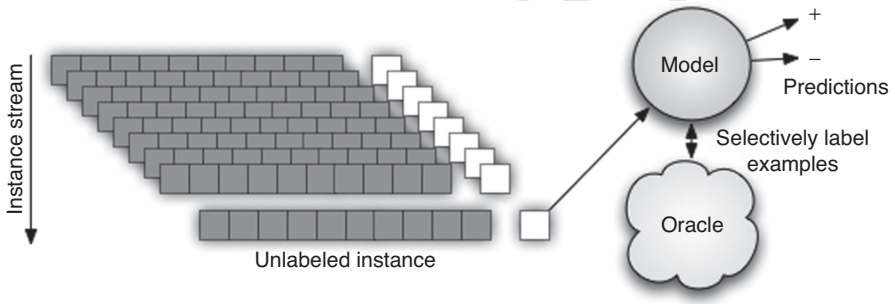


**Figure 6.2** Stream-based active learning.

predictive performance is sufficient, the model may be incorporated into an end system that feeds the model with unlabeled examples and consumes the predicted results. A diagram illustrating these two types of AL scenarios is presented in Figures 6.1 and 6.2, respectively. Owing to the greater attention given to pool-based AL in recent scientific literature and the additional power available to an AL system capable of processing a large representation of the problem space at once, the remainder of this chapter focuses on the latter of these two scenarios, the pool-based setting.

The most common techniques in AL have focused on selecting examples from a so-called region of uncertainty, the area nearest to the current model's predictive decision boundary.[1] Incorporating uncertainty into active data acquisition dates back to research focused on optimal experimental design [2], and has been

[1]For instance, the simplest case when performing binary classification would involve choosing $x' = \arg\max_x \min_y P(y|x)$, $y \in 0, 1$.

among the earliest successful examples of active machine learning techniques [3, 4]. The intuition behind uncertainty-based selection is that this region surrounding a model's decision boundary is where that model is most likely to make mistakes. Incorporating labeled examples from this region may improve the model's performance along this boundary, leading to gains in overall accuracy.

Many popular subsequent techniques are specializations of uncertainty selection, including query-by-committee-based approaches [5–7], where, given an ensemble of (valid) predictive models, examples are selected based on the level of disagreement elicited among the ensemble, and the popular "simple margin" technique proposed by Tong and Koller [8], where, given a current parameterization of a support vector machine (SVM), $w_j$, the example $x_i$ is chosen that comes closest to the decision boundary, $x_i = \arg\min_{x'} |w_j \Phi(x)|$, where $\Phi(\cdot)$ is a function mapping an example to an alternate space utilized by the kernel function in the SVM: $k(u, v) = \Phi(u)\Phi(v)$.

Expected-utility-based approaches, where examples are chosen based on the estimated expected improvement in a certain objective, are achieved by incorporating a given example into the training set.[2] Such techniques often involve costly nested cross-validation where each available example is assigned all possible label states [9–11].

### 6.2.2 Dealing with Class Imbalance Problem in Active Learning

Selecting examples from an unlabeled pool with substantial class imbalance may pose several difficulties for traditional AL. The greater proportion of examples in the majority class may lead to a model that prefers one class over another. If the labels of examples selected by an AL scheme are thought of as a random variable, the innate class imbalance in the example pool would almost certainly lead to a preference for majority examples in the training set. Unless properly dealt with,[3] this over-representation may simply lead to a predictive preference for the majority class when labeling. Typically, when making predictive models in an imbalanced setting, it is the minority class that is of interest. For instance, it is important to discover patients who have a rare but dangerous ailment based on the results of a blood test, or infrequent but costly fraud in a credit card company's transaction history. This difference in class preferences between an end system's needs and a model's tendencies causes a serious problem for AL (and predictive systems in general) in imbalanced settings. Even if the problem of highly imbalanced (although correct in terms of base rate) training set problem can be dealt with, the tendency for a selection algorithm to gather majority examples creates other problems. The nuances of the minority set may be poorly represented in the training data, leading to a "predictive misunderstanding" in

[2]Note that this selection objective may not necessarily be the same objective used during the base model's use time. For instance, examples may be selected according to their contribution to the reduction in problem uncertainty.

[3]For instance, by imbalanced learning techniques described throughout this book.

certain regions of the problem space; while a model may be able to accurately identify large regions of the minority space, portions of this space may get mislabeled, or labeled with poor quality, because of underrepresentation in the training set. At the extreme, disjunctive subregions may get missed entirely. Both of these problems are particularly acute as the class imbalance increases, and are discussed in greater detail in Section 6.6. Finally, in the initial stages of AL, when the base model is somewhat naïve, the minority class may get missed entirely as an AL heuristic probes the problem space for elusive but critical minority examples.

### 6.2.3 Addressing the Class Imbalance Problem with Active Learning

As we will demonstrate in Section 6.3, AL presents itself as an effective strategy for dealing with moderate class imbalance even without any special considerations for the skewed class distribution. However, the previously discussed difficulties imposed by more substantial class imbalance on the selective abilities of AL heuristics have led to the development of several techniques that have been specially adapted to imbalanced problem settings. These skew-specialized AL techniques incorporate an innate preference for the minority class, leading to more balanced training sets and better predictive performance in imbalanced settings. Additionally, there exists a category of density-sensitive AL techniques, techniques that explicitly incorporate the geometry of the problem space. By incorporating the knowledge of independent dimensions of the unlabeled example pool, there exists a potential for better exploration, resulting in improved resolution of rare subregions of the minority class. We detail these two broad classes of AL techniques as follows.

***6.2.3.1 Density-Sensitive Active Learning*** Utility-based selection strategies for AL attribute some score, $\mathcal{U}(\cdot)$, to each instance $x$ encapsulating how much improvement can be expected from training on that instance. Typically, the examples offering a maximum $\mathcal{U}(x)$ are selected for labeling. However, the focus on individual examples may expose the selection heuristic to outliers, individual examples that achieve a high utility score, but do not represent a sizable portion of the problem space. Density-sensitive AL heuristics seek to alleviate this problem by leveraging the entire unlabeled pool of examples available to the active learner. By explicitly incorporating the geometry of the input space when attributing some selection score to a given example, outliers, noisy examples, and sparse areas of the problem space may be avoided. The following are some exemplary AL heuristics that leverage density sensitivity.

*Information Density.* This is a general density-sensitive paradigm compatible with arbitrary utility-based active selection strategies and a variety of metrics used to compute similarity [12]. In this case, a meta-utility score is computed for each example based not only on a traditional utility score, $\mathcal{U}(x)$, but also on a measurement of that example's similarity to other instances in the problem

space. Given a similarity metric between two points, $\text{sim}(x, x')$, information density selects examples according to:

$$\mathcal{U}_m(x) = \mathcal{U}(x) \left( \frac{1}{|X|} \sum_{x' \in X} \text{sim}(x, x') \right)^{\beta}$$

Here, $\beta$ is a hyper-parameter controlling the trade-off between the raw instance-specific utility, $\mathcal{U}(x)$ and the similarity component in the overall selection criterion.

Zhu et al. [13] developed a technique similar to the information density technique of Settles and Craven, selecting the instances according a uncertainty-based criterion modified by a density factor: $\mathcal{U}_n(x) = \mathcal{U}(x)\text{KNN}(x)$, where $\text{KNN}(x)$ is the average cosine similarity of the $K$ nearest neighbors to $x$. The same authors also propose the *sampling by clustering*, a density-only AL heuristic where the problem space is clustered, and the points closest to the cluster centeroids are selected for labeling.

*Pre-Clustering.* Here it is assumed that the problem is expressed as a mixture model comprising $K$ distributions, each component model completely encoding information related to the labels of member examples—the label $y$ is conditionally independent of the covariates $x$ given knowledge of its cluster, $k$ [14]. This assumption yields a joint distribution describing the problem: $p(x, y, k) = p(x|k)p(y|k)p(k)$, yielding a poster probability on $y$:

$$p_k(y|x) = \sum_{k=1}^{K} p(y|k) \frac{p(x|k)p(k)}{p(x)}$$

In essence, this a density-weighted mixture model used for classification. The $K$ clusters are created by a application of typical clustering techniques of the data, with a cluster size used to estimate $p(k)$, and $p(y|k)$ is estimated via a logistic regression considering a cluster's representative example. A probability density is inferred for each cluster; in the example case presented in the earlier-mentioned work, an isotropic normal distribution is used, from which $p(x|k)$ can be estimated. Examples are then selected from an uncertainty score computed via the above-mentioned posterior model weighted by the probability of observing a given $x$:

$$\mathcal{U}_k(x) = (1 - |p_k(y|x)|)\, p(x)$$

Of course, there exists a variety of other techniques in the research literature designed to explicitly incorporate information related to the problem's density into an active selection criterion. McCallum and Nigam [15] modify a query-by-committee to use an exponentiated Kullback–Leibler (KL) divergence-based uncertainty metric and combine this with semi-supervised learning in the form

of an expectation maximization (EM) procedure. This combined semi-supervised AL has the benefit of ignoring regions that can be reliably "filled in" by a semi-supervised procedure, while also selecting those examples that may benefit this EM process.

Donmez et al. [16] propose a modification of the density-weighted technique of Nguyen and Smeulders. This modification simply selects examples according to the convex combination of the density-weighted technique and traditional uncertainty sampling. This hybrid approach is again incorporated into a so-called dual-active learner, where only uncertainty sampling is incorporated once the benefits of pure density-sensitive sampling seem to be diminishing.

*Alternate Density-Sensitive Heuristics.* Donmez and Carbonell [17] incorporate density into active label selection by performing a change of coordinates into a space whose metric expresses not only Euclidian similarity but also density. Examples are then chosen based on a density-weighted uncertainty metric designed to select examples in pairs—one member of the pair from each side of the current decision boundary. The motivation is that sampling from both sides of the decision boundary may yield better results than selecting from one side in isolation.

Through selection based on an "unsupervised" heuristic estimating the utility of label acquisition on the pool of unlabeled instances, Roy and McCallum [9] incorporate the geometry of the problem space into active selection implicitly. This approach attempts to quantify the improvement in model performance attributable to each unlabeled example, taken in expectation over all label assignments:

$$\mathcal{U}_E = \sum_{y' \in Y} \hat{p}(y'|x) \sum_{x' \neq X} \mathcal{U}_e(x'; x, y = y')$$

Here, the probability of class membership in the earlier-mentioned expectation comes from the base model's current posterior estimates. The utility value on the right side of the previous equation, $\mathcal{U}_e(x'; x, y = y')$, comes from assuming a label of $y'$, for example, $x$, and incorporating this pseudo-labeled example into the training set temporarily. The improvement in model performance with the inclusion of this new example is then measured. Since a selective label acquisition procedure may result in a small or arbitrarily biased set of examples, accurate evaluation through nested cross-validation is difficult. To accommodate this, Roy and McCallum propose two uncertainty measures taken over the pool of unlabeled examples, $x' \neq x$. Specifically, they look at the entropy of the posterior probabilities of examples in the pool, and the magnitude of the maximum posterior as utility measures, both estimated after the inclusion of the "new" example. Both metrics favor "sharp" posteriors, an optimization minimizing uncertainty rather than model performance; instances are selected by their reduction in uncertainty taken in expectation over the entire example pool.

***6.2.3.2 Skew-Specialized Active Learning*** Additionally, there exists a body of research literature on AL specifically to deal with class imbalance problem. Tomanek and Hahn [18] investigates query-by-committee-based approaches to sampling labeled sentences for the task of named entity recognition. The goal of their selection strategy is to encourage class-balanced selections by incorporating class-specific costs. Unlabeled instances are ordered by a class-weighted, entropy-based disagreement measure, $-\sum_{j\in\{0,1\}} b_j V(k_j)/|C| \log V(k_j)/|C|$, where $V(k_j)$ is the number of votes from a committee of size $|C|$ that an instance belongs to a class $k_j$. $b_j$ is a weight corresponding to the importance of including a certain class; a larger value of $b_j$ corresponds to a increased tendency to include examples that are thought to belong to this class. From a window $W$ of examples with highest disagreement, instances are selected greedily based on the model's estimated class membership probabilities so that the batch selected from the window has the highest probability of having a balanced class membership.

SVM-based AL has been shown [19] to be a highly effective strategy for addressing class imbalance without any skew-specific modifications to the algorithm. Bloodgood and Shanker [20] extend the benefits of SVM-based AL by proposing an approach that incorporates class-specific costs. That is, the typical $C$ factor describing an SVM's misclassification penalty is broken up into $C_+$ and $C_-$, describing the costs associated with misclassification of positive and negative examples, respectively, a common approach for improving the performance of SVMs in cost-sensitive settings. Additionally, cost-sensitive SVMs is known to yield predictive advantages in imbalanced settings by offering some preference to an otherwise overlooked class, often using the heuristic for setting class-specific costs: $C_+/C_- = |\{x|x \in -\}|/|\{x|x \in +\}|$, a ratio in inverse proportion to the number of examples in each class. However, in the AL setting, the true class ratio is unknown, and the quantity $C_+/C_-$ must be estimated by the AL system. Bloodgood and Shanker show that it is advantageous to use a preliminary stage of random selection in order to establish some estimate of the class ratio, and then proceed with example selection according to the uncertainty-based "simple margin" criterion using the appropriately tuned cost-sensitive SVM.

AL has also been studied as a way to improve the generalization performance of resampling strategies that address class imbalance. In these settings, AL is used to choose a set of instances for labeling, with sampling strategies used to improve the class distribution. Ertekin [21] presented virtual instance resampling technique using active learning (VIRTUAL), a hybrid method of oversampling and AL that forms an adaptive technique for resampling of the minority class instances. The learner selects the most informative example $x_i$ for oversampling, and the algorithm creates a synthetic instance along the direction of $x_i$'s one of $k$ neighbors. The algorithm works in an online manner and builds the classifier incrementally without the need to retrain on the entire labeled dataset after creating a new synthetic example. This approach, which we present in detail in Section 6.4, yields an efficient and scalable learning framework.

Zhu and Hovy [22] describe a bootstrap-based oversampling strategy (BootOS) that, given an example to be resampled, generates a bootstrap example based on all the $k$ neighbors of that example. At each epoch, the examples with the greatest uncertainty are selected for labeling and incorporated into a labeled set, $L$. From $L$, the proposed oversampling strategy is applied, yielding a more balanced dataset, $L'$, a dataset that is used to retrain the base model. The selection of the examples with the highest uncertainty for labeling at each iteration involves resampling the labeled examples and training a new classifier with the resampled dataset; therefore, scalability of this approach may be a concern for large-scale datasets.

In the next section, we demonstrate that the principles of AL are naturally suited to address the class imbalance problem and that AL can in fact be an effective strategy to have a balanced view of an otherwise imbalanced dataset, without the need to resort to resampling techniques. It is worth noting that the goal of the next section is not to cast AL as a replacement for resampling strategies. Rather, our main goal is to demonstrate how AL can alleviate the issues that stem from class imbalance and present AL as an alternate technique that should be considered in case a resampling approach is impractical, inefficient, or ineffective. In problems where resampling *is* the preferred solution, we show in Section 6.4 that the benefits of AL can still be leveraged to address class imbalance. In particular, we present an adaptive oversampling technique that uses AL to determine which examples to resample in an online setting. These two different approaches show the versatility of AL and the importance of selective sampling to address the class imbalance problem.

## 6.3 ACTIVE LEARNING FOR IMBALANCED DATA CLASSIFICATION

As outlined in Section 6.2.1, AL is primarily considered as a technique to reduce the number of training samples that need to be labeled for a classification task. From a traditional perspective, the active learner has access to a vast pool of unlabeled examples, and it aims to make a clever choice to select the most informative example to obtain its label. However, even in the cases where the labels of training data are already available, AL can still be leveraged to obtain the informative examples through training sets [23–25]. For example, in large-margin classifiers such as SVM, the *informativeness* of an example is synonymous with its distance to the hyperplane. The farther an example is to the hyperplane, the more the learner is confident about its true class label; hence there is little, if any, benefit that the learner can gain by asking for the label of that example. On the other hand, the examples close to the hyperplane are the ones that yield the most information to the learner. Therefore, the most commonly used AL strategy in SVMs is to check the distance of each unlabeled example to the hyperplane and focus on the examples that lie closest to the hyperplane, as they are considered to be the most informative examples to the learner [8].

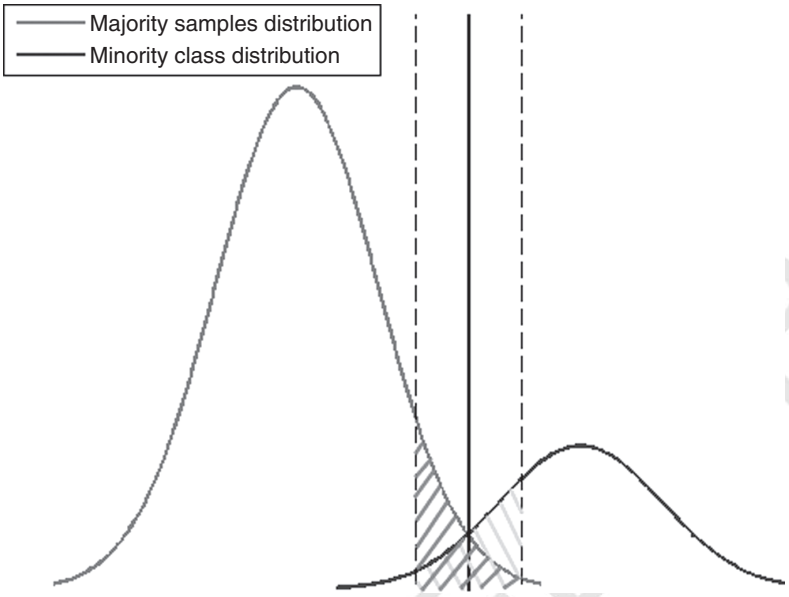


**Figure 6.3** Data within the margin is less imbalanced than the entire data.

The strategy of selecting examples within the margin also strongly addresses the problems that arise from imbalanced classes. Consider the class distributions of an imbalanced dataset presented in Figure 6.3. The shaded region corresponds to the class distribution of the data within the margin. As shown in the figure, the imbalance ratio of the classes within the margin is much smaller than the class imbalance ratio of the entire dataset. Therefore, any selection strategy that focuses on the examples in the margin most likely ends up with a more balanced class distribution than that of the entire dataset.

Throughout this section, the discussion is constrained to standard two-class classification problems using SVMs. The next section presents a brief overview of SVMs, followed by the working principles of an efficient AL algorithm in Section 6.3.2. We explain the advantage of using online SVMs with the active sample selection in Section 6.3.3.

### 6.3.1 Support Vector Machines

SVMs [26] are well known for their strong theoretical foundations, generalization performance, and ability to handle high dimensional data. In the binary classification setting, let $((x_1, y_1) \cdots (x_n, y_n))$ be the training dataset, where $x_i$ are the feature vectors representing the instances and $y_i \in (-1, +1)$ be the labels of the instances. Using the training set, SVM builds an optimum hyperplane—a linear discriminant in a higher dimensional feature space—that separates the two

classes by the largest margin. This hyperplane is obtained by minimizing the following objective function:

$$\min_{\mathbf{w},b,\xi_i} \frac{1}{2}\mathbf{w}\cdot \mathrm{w}^{\mathrm{T}} + C\sum_{i=1}^{N}\xi_i \tag{6.1}$$

$$\text{subject to } \begin{cases} \forall i \;\; y_i(\mathbf{w}^{\mathrm{T}}\Phi(x_i) - b) \geq 1 - \xi_i \\ \forall i \;\; \xi_i \geq 0 \end{cases} \tag{6.2}$$

where $\mathbf{w}$ is the norm of the hyperplane, $b$ is the offset, $y_i$ are the labels, $\Phi(\cdot)$ is the mapping from input space to feature space, and $\xi_i$ are the slack variables that permit the non-separable case by allowing misclassification of training instances. In practice, the convex quadratic programming (QP) problem in Equation 6.1 is solved by optimizing the dual cost function. The dual representation of Equation 6.1 is given as

$$\max W(\alpha) \equiv \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(\mathbf{x_i}, \mathbf{x_j}) \tag{6.3}$$

$$\text{subject to } \begin{cases} \forall i \;\; 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{N}\alpha_i y_i = 0 \end{cases} \tag{6.4}$$

where $y_i$ are the labels, $\Phi(\cdot)$ is the mapping from the input space to the feature space, $K(\mathbf{x_i}, \mathbf{x_j}) = \langle \Phi(\mathbf{x_i}), \Phi(\mathbf{x_j}) \rangle$ is the kernel matrix, and the $\alpha_i$'s are the *Lagrange multipliers*, which are nonzero only for the training instances that fall in the margin. Those training instances are called *support vectors* and they define the position of the hyperplane. After solving the QP problem, the norm of the hyperplane $\mathbf{w}$ can be represented as

$$\mathbf{w} = \sum_{i=1}^{n}\alpha_i\Phi(x_i) \tag{6.5}$$

### 6.3.2 Margin-Based Active Learning with SVMs

Note that in Equation 6.5, only the support vectors affect the SVM solution. This means that if SVM is retrained with a new set of data that consist of only those support vectors, the learner will end up finding the same hyperplane. This emphasizes the fact that not all examples are equally important in training sets. Then the question becomes how to select the most informative examples for labeling from the set of unlabeled training examples. This section focuses on a form of selection strategy called *margin-based AL*. As was highlighted earlier, in SVMs, the most informative example is believed to be the closest one to the hyperplane as it divides the *version space* into two equal parts. The aim

is to reduce the version space as fast as possible to reach the solution faster in order to avoid certain *costs* associated with the problem. For the possibility of a nonsymmetric version space, there are more complex selection methods suggested by Tong and Koller [8], but the advantage of those methods are not significant, considering their high computational costs.

***6.3.2.1 Active Learning with Small Pools*** The basic working principle of margin-based AL with SVMs is: (i) train an SVM on the existing training data, (ii) select the closest example to the hyperplane, and (iii) add the new selected example to the training set and train again. In classical AL [8], the search for the most informative example is performed over the entire dataset. Note that, each iteration of AL involves the recomputation of each training example's distance to the new hyperplane. Therefore, for large datasets, searching the entire training set is a very time-consuming and computationally expensive task.

One possible remedy for this performance bottleneck is to use the "59 trick" [27], which alleviates a full search through the entire dataset, approximating the most informative examples by examining a small constant number of randomly chosen samples. The method picks $L$ ($L \ll$ # training examples) random training samples in each iteration and selects the best (closest to the hyperplane) among them. Suppose, instead of picking the closest example among all the training samples $X_N = (x_1, x_2, \ldots, x_N)$ at each iteration, we first pick a random subset $X_L$, $L \ll N$ and select the closest sample $x_i$ from $X_L$ based on the condition that $x_i$ is among the top $p\%$ closest instances in $X_N$ with probability $(1 - \eta)$. Any numerical modification to these constraints can be met by varying the size of $L$, and is independent of $N$. To demonstrate this, the probability that at least one of the $L$ instances is among the closest $p$ is $1 - (1 - p)^L$. Owing to the requirement of $(1 - \eta)$ probability, we have

$$1 - (1 - p)^L = 1 - \eta \tag{6.6}$$

which follows the solution of $L$ in terms of $\eta$ and $p$

$$L = \frac{\log \eta}{\log(1 - p)} \tag{6.7}$$

For example, the active learner will pick one example, with 95% probability, that is among the top 5% closest instances to the hyperplane, by randomly sampling only $\lceil \log(0.05)/\log(0.95) \rceil = 59$ examples regardless of the training set size. This approach scales well since the size of the subset $L$ is independent of the training set size $N$, requires significantly less training time, and does not have an adverse effect on the classification performance of the learner.

### 6.3.3 Active Learning with Online Learning

Online learning algorithms are usually associated with problems where the complete training set is not available. However, in cases where the complete training

set *is* available, the computational properties of these algorithms can be leveraged for faster classification and incremental learning. Online learning techniques can process new data presented one at a time, as a result of either AL or random selection, and can integrate the information of the new data to the system without training on all previously seen data, thereby allowing models to be constructed incrementally. This working principle of online learning algorithms leads to speed improvements and a reduced memory footprint, making the algorithm applicable to very large datasets. More importantly, this incremental learning principle suits the nature of AL much more naturally than the batch algorithms. Empirical evidence indicates that a single presentation of each training example to the algorithm is sufficient to achieve training errors comparable to those achieved by the best minimization of the SVM objective [24].

### 6.3.4 Performance Metrics

Classification accuracy is not a good metric to evaluate classifiers in applications facing class imbalance problems. SVMs have to achieve a trade-off between maximizing the margin and minimizing the empirical error. In the non-separable case, if the misclassification penalty $C$ is very small, the SVM learner simply tends to classify every example as negative. This extreme approach maximizes the *margin* while making no classification errors on the negative instances. The only error is the cumulative error of the positive instances that are already few in numbers. Considering an imbalance ratio of 99 to 1, a classifier that classifies everything as negative, will be 99% accurate. Obviously, such a scheme would not have any practical use, as it would be unable to identify positive instances.

For the evaluation of these results, it is useful to consider several other prediction performance metrics such as $g$-means, area under the curve (AUC), and precision–recall break-even point (PRBEP), which are commonly used in imbalanced data classification. $g$-Means [28] is denoted as $g = \sqrt{\text{sensitivity} \cdot \text{specificity}}$, where sensitivity is the accuracy on the positive instances given as TruePos./(TruePos. + FalseNeg.), and specificity is the accuracy on the negative instances given as TrueNeg./(TrueNeg. + FalsePos.).

The receiver operating curve (ROC) displays the relationship between sensitivity and specificity at all possible thresholds for a binary classification scoring model, when applied to independent test data. In other words, ROC curve is a plot of the true positive rate against the false positive rate as the decision threshold is changed. The *area under the ROC* (AUROC) is a numerical measure of a model's discrimination performance and shows how successfully and correctly the model ranks and thereby separates the positive and negative observations. Since the AUC metric evaluates the classifier across the entire range of decision thresholds, it gives a good overview of the performance when the operating condition for the classifier is unknown or the classifier is expected to be used in situations with significantly different class distributions.

PRBEP is another commonly used performance metric for imbalanced data classification. PRBEP is the accuracy of the positive class at the threshold

where precision equals to recall. Precision is defined as TruePos./(TruePos. + FalsePos.), and recall is defined as TruePos./(TruePos. + FalseNeg.)

### 6.3.5 Experiments and Empirical Evaluation

We study the performance of the algorithm on various benchmark real-world datasets, including MNIST, USPS, several categories of Reuters-21578 collection, five topics from CiteSeer, and three datasets from the University of California, Irvine (UCI) repository. The characteristics of the datasets are outlined in Reference 19. In the experiments, an early stopping heuristic for AL is employed, as it has been shown that AL converges to the solution faster than the random sample selection method [19]. A theoretically sound method to stop training is when the examples in the margin are exhausted. To check whether there are still unseen training examples in the margin, the distance of the newly selected example is compared against the support vectors of the current model. If the newly selected example by AL (closest to the hyperplane) is not closer than any of the support vectors, it is concluded that the margin is exhausted. A practical implementation of this idea is to count the number of support vectors during the AL training process. If the number of the support vectors stabilizes, it implies that all possible support vectors have been selected by the AL method.

As the first experiment, examples are randomly removed from the minority class in *Adult* dataset to achieve different data imbalance ratios, comparing SVM-based AL and random sampling (RS).[4] For brevity, AL with small pools is referred to as AL as the small pools heuristic is utilized for all AL methods considered later. Comparisons of PRBEP in Figure 6.4 show an interesting behavior. As the class imbalance ratio is increased, AL curves display peaks in the early steps of the learning. This implies that by using an early stopping criteria, AL can give higher prediction performance than RS can possibly achieve even after using all the training data. The learning curves presented in Figure 6.4 demonstrate that the addition of instances to a model's training after finding those most informative instances can be detrimental to the prediction performance of the classifier, as this may cause the model to suffer from overfitting. Figure 6.4 curves show that generalization can peak to a level above that can be achieved by using all available training data. In other words, it is possible to achieve better classification performance from a small informative subset of the training data than what can be achieved using all available training data. This finding agrees with that of Schohn and Cohn [23] and strengthens the idea of applying an early stopping to AL algorithms.

For further comparison of the performance of a model built on all available data (batch) and AL subject to early halting criteria, refer to Table 6.1, comparing the $g$-means and the AUC values for these two methods. The data efficiency column for AL indicates that by processing only a portion of the examples from the

[4]Here, the random process is assumed to be uniform; examples are selected with equal probability from the available pool.

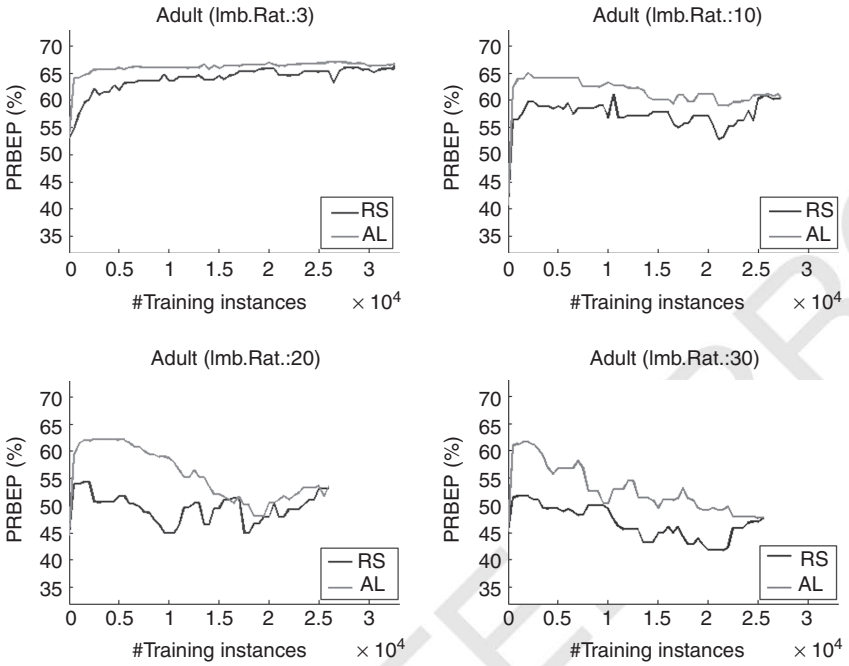


**Figure 6.4** Comparison of PRBEP of AL and RS on the adult datasets with different imbalance ratios (Imb.R.=3, 10, 20, 30).

training set, AL can achieve similar or even higher generalization performance than that of batch, which sees all the training examples. Another important observation from Table 6.1 is that support vector imbalance ratios in the final models are much less than the class imbalance ratios of the datasets. This confirms the discussion of Figure 6.3. The class imbalance ratio within the margins is much less than that of the entire data, and AL can be used to reach those informative examples that most likely become support vectors without seeing all the training examples.

Figure 6.5 investigates how the number of support vectors changes when presented with examples selected according to AL and RS. Because the base rate of the dataset gathered by RS approaches that of the example pool, the support vector imbalance ratio quickly approaches the data imbalance ratio. As learning continues, the learner should gradually see all the instances within the final margin and the support vector imbalance ratio decreases. At the end of training with RS, the support vector imbalance ratio is the data imbalance ratio within the margin. The support vector imbalance ratio curve of AL is drastically different than RS. AL intelligently picks the instances closest to the margin in each step. Since the data imbalance ratio within the margin is lower than data imbalance ratio, the support vectors in AL are more balanced than RS during learning. Using AL, the model saturates by seeing only 2000 (among 7770) training instances and

**Table 6.1 Comparison of *g*-Means and AUC for AL and RS with Entire Training Data (Batch)**

| | | g-Means (%) | | AUC (%) | | Imb. | SV- / | Data |
|---|---|---|---|---|---|---|---|---|
| | Dataset | Batch | AL | *Batch* | AL | Rat. | SV+ | Efficiency (%) |
| Reuters | Corn | 85.55 | 86.59 | 99.95 | 99.95 | 41.9 | 3.13 | 11.6 |
| | Crude | 88.34 | 89.51 | 99.74 | 99.74 | 19.0 | 2.64 | 22.6 |
| | Grain | 91.56 | 91.56 | 99.91 | 99.91 | 16.9 | 3.08 | 29.6 |
| | Interest | 78.45 | 78.46 | 99.01 | 99.04 | 21.4 | 2.19 | 30.9 |
| | Money-fx | 81.43 | 82.79 | 98.69 | 98.71 | 13.4 | 2.19 | 18.7 |
| | Ship | 75.66 | 74.92 | 99.79 | 99.80 | 38.4 | 4.28 | 20.6 |
| | Trade | 82.52 | 82.52 | 99.23 | 99.26 | 20.1 | 2.22 | 15.4 |
| | Wheat | 89.54 | 89.55 | 99.64 | 99.69 | 35.7 | 3.38 | 11.6 |
| CiteSeer | AI | 87.83 | 88.58 | 94.82 | 94.69 | 4.3 | 1.85 | 33.4 |
| | COMM | 93.02 | 93.65 | 98.13 | 98.18 | 4.2 | 2.47 | 21.3 |
| | CRYPT | 98.75 | 98.87 | 99.95 | 99.95 | 11.0 | 2.58 | 15.2 |
| | DB | 92.39 | 92.39 | 98.28 | 98.46 | 7.1 | 2.50 | 18.2 |
| | OS | 91.95 | 92.03 | 98.27 | 98.20 | 24.2 | 3.52 | 36.1 |
| UCI | Abalone-7 | 100.0 | 100.0 | 100.0 | 100.0 | 9.7 | 1.38 | 24.0 |
| | Letter-A | 99.28 | 99.54 | 99.99 | 99.99 | 24.4 | 1.46 | 27.8 |
| | Satimage | 82.41 | 83.30 | 95.13 | 95.75 | 9.7 | 2.62 | 41.7 |
| | USPS | 99.22 | 99.25 | 99.98 | 99.98 | 4.9 | 1.50 | 6.8 |
| | MNIST-8 | 98.47 | 98.37 | 99.97 | 99.97 | 9.3 | 1.59 | 11.7 |

SV ratios are given at the saturation point. Data efficiency corresponds to the percentage of training instances that AL processes to reach saturation.

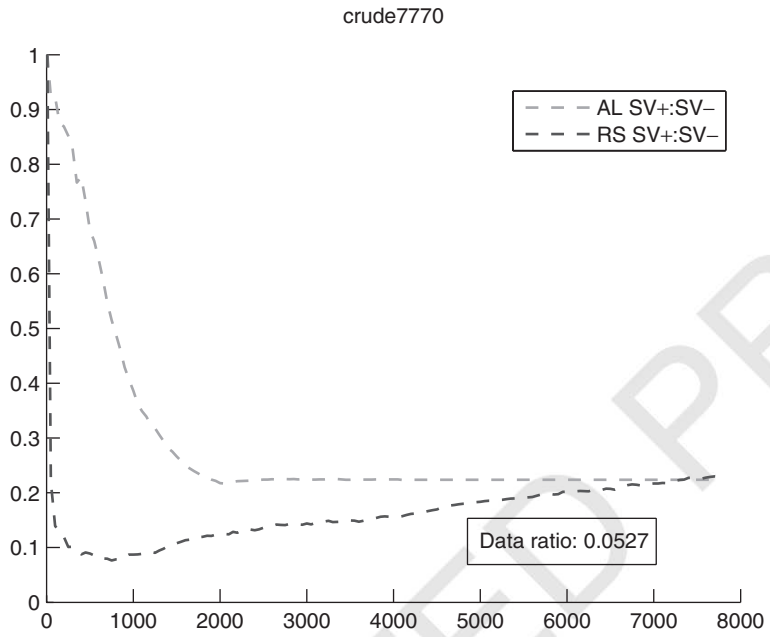


**Figure 6.5** Support vector ratios in AL and RS.

reaches the final support vector imbalance ratio. Note that both methods achieve similar support vector imbalance ratios when learning finishes, but AL achieves this in the early steps of the learning.

It is also interesting to consider the performance of AL as a selection heuristic in light of more conventional sampling strategies. Here, AL is compared to traditional under-sampling of the majority class (US) and an oversampling method (SMOTE, synthetic minority oversampling technique), both being examples of resampling techniques that require preprocessing. It has been shown that oversampling at random does not help to improve prediction performance [29]; therefore, a more complex oversampling method is required. SMOTE oversamples the minority class by creating synthetic examples rather than with replacement. The $k$ nearest positive neighbors of all positive instances are identified, and synthetic positive examples are created and placed randomly along the line segments joining the $k$-minority class nearest neighbors.

For additional comparison, the method of assigning different costs (DCs) to the positive and negative classes as the misclassification penalty parameter is examined. For instance, if the imbalance ratio of the data is $19:1$ in favor of the negative class, the cost of misclassifying a positive instance is set to be 19 times greater than that of misclassifying a negative one. We use the online SVM package LASVM[5] in all experiments. Other than the results of the methods

[5]Available at `http://leon.bottou.org/projects/lasvm`

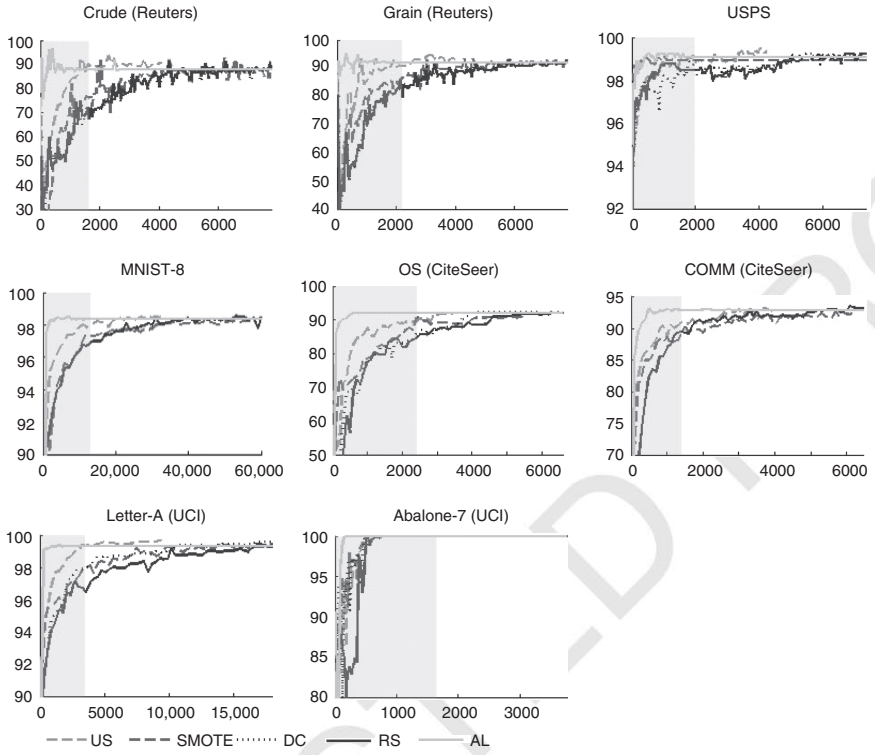


**Figure 6.6** Comparisons of $g$-means. The right border of the shaded area corresponds to the early stopping point.

addressing the class imbalance problem, the results of batch algorithm with the original training set are provided to form a baseline. LASVM is run in RS mode for US, SMOTE, and DC.

We present the comparisons of the methods for $g$-means performance metric for several datasets in Figure 6.6. The right border of the shaded light gray area is the place where the aforementioned early stopping strategy is applied. The curves in the graphs are averages of 10 runs. For completeness, all AL experiments were allowed to continue to select examples until exhaustion, bypassing any early stopping. Table 6.2 presents the PRBEP of the methods and the total running times of the SMOTE and AL on 18 benchmark and real-world datasets. The results for AL in Table 6.2 depict the results in the early stopping points. The results for the other methods in Table 6.2 depict the values at the end of the curves—when trained with the entire dataset—as those methods do not employ any early stopping criteria. We did not apply early stopping criteria to the other methods because, as observed from Figure 6.6, no early stopping criteria would achieve a comparable training time to that of AL's training time without a significant loss in their prediction performance based on convergence time. The other

**Table 6.2 Comparison of PRBEP and Training Time**

| | Metric | PRBEP | | | | | Training Time (s) | |
|---|---|---|---|---|---|---|---|---|
| | Dataset | Batch | US | SMOTE | DC | AL | SMOTE | AL |
| Reuters | Corn | 91.07 | 78.57 | 91.07 | 89.28 | 89.29 | 87 | 16 |
| | Crude | 87.83 | 85.70 | 87.83 | 87.83 | 87.83 | 129 | 41 |
| | Grain | 92.62 | 89.93 | 91.44 | 91.94 | 91.94 | 205 | 50 |
| | Interest | 76.33 | 74.04 | 77.86 | 75.57 | 75.57 | 116 | 42 |
| | Money-fx | 73.74 | 74.30 | 75.42 | 75.42 | 76.54 | 331 | 35 |
| | Ship | 86.52 | 86.50 | 88.76 | 89.89 | 89.89 | 49 | 32 |
| | Trade | 77.77 | 76.92 | 77.77 | 77.78 | 78.63 | 215 | 38 |
| | Wheat | 84.51 | 81.61 | 84.51 | 84.51 | 85.92 | 54 | 25 |
| CiteSeer | AI | 78.80 | 80.68 | 78.99 | 78.79 | 79.17 | 1402 | 125 |
| | COMM | 86.59 | 86.76 | 86.59 | 86.59 | 86.77 | 1707 | 75 |
| | CRYPT | 97.89 | 97.47 | 97.89 | 97.89 | 97.89 | 310 | 19 |
| | DB | 86.36 | 86.61 | 86.98 | 86.36 | 86.36 | 526 | 41 |
| | OS | 84.07 | 83.19 | 84.07 | 84.07 | 84.07 | 93 | 23 |
| UCI | Abalone-7 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 16 | 4 |
| | Letter-A | 99.48 | 96.45 | 99.24 | 99.35 | 99.35 | 86 | 3 |
| | Satimage | 73.46 | 68.72 | 73.46 | 73.93 | 73.93 | 63 | 21 |
| | USPS | 98.44 | 98.44 | 98.13 | 98.44 | 98.75 | 4328 | 13 |
| | MNIST-8 | 97.63 | 97.02 | 97.74 | 97.63 | 97.74 | 83, 339 | 1048 |

**Table 6.3 Support Vectors with SMOTE (SMT), AL, and VIRTUAL**

| | | Imb. | #SV(-)/#SV(+) | | | $\#SV_V(+)$/#V.I. | |
|---|---|---|---|---|---|---|---|
| | Dataset | Rt. | SMT | AL | VIRTUAL | SMT | VIRTUAL |
| Reuters | acq | 3.7 | 1.24 | 1.28 | 1.18 | 2.4% | **20.3%** |
| | corn | 41.9 | 2.29 | 3.08 | 1.95 | 17.1% | **36.6%** |
| | crude | 19.0 | 2.30 | 2.68 | 2.00 | 10.8% | **50.4%** |
| | earn | 1.7 | 1.68 | 1.89 | 1.67 | 6.0% | **24.2%** |
| | grain | 16.9 | 2.62 | 3.06 | 2.32 | 7.2% | **42.3%** |
| | interest | 21.4 | 1.84 | 2.16 | 1.66 | 13.3% | **72.2%** |
| | money-fx | 13.4 | 1.86 | 2.17 | 1.34 | 8.2% | **31.1%** |
| | ship | 38.4 | 3.45 | 4.48 | 2.80 | 20.0% | **66.5%** |
| | trade | 20.1 | 1.89 | 2.26 | 1.72 | 15.4% | **26.6%** |
| | wheat | 35.7 | 2.55 | 3.43 | 2.22 | 12.3% | **63.9%** |
| UCI | Abalone | 9.7 | 0.99 | 1.24 | 0.99 | 30.4% | **69.2%** |
| | Breast | 1.9 | 1.23 | 0.60 | 0.64 | 2.9% | **39.5%** |
| | Letter | 24.4 | 1.21 | 1.48 | 0.97 | 0.98% | **74.4%** |
| | Satimage | 9.7 | 1.31 | 1.93 | 0.92 | 37.3% | **53.8%** |

Imb.Rt. is the data imbalance ratio, and #SV(−)/#SV(+) represents the support vector imbalance ratio. The rightmost two columns compare the portion of the virtual instances selected as support vectors in SMOTE and VIRTUAL.

methods converge to similar levels of $g$-means when nearly all training instances are used, and applying an early stopping criteria would have little, if any, effect on their training times.

Since AL involves discarding some instances from the training set, it can be perceived as a type of under-sampling method. Unlike traditional US, which discards majority samples randomly, AL performs an intelligent search for the most informative ones adaptively in each iteration according to the current hyperplane. Datasets where class imbalance ratio is high such as *corn*, *wheat*, *letter*, and *satimage* observe significant decrease in PRBEP of US (Table 6.3). Note that US's under-sampling rate for the majority class in each category is set to the same value as the final support vector ratio where AL reaches in the early stopping point and RS reaches when it sees the entire training data. Although the class imbalance ratios provided to the learner in AL and US are the same, AL achieves significantly better PRBEP performance metric than US. The Wilcoxon-signed-rank test (two-tailed) reveals that the zero median hypothesis can be rejected at the significance level 1% ($p = 0.0015$), implying that AL performs statistically better than US in these 18 datasets. These results reveal the importance of using the informative instances for learning.

Table 6.2 gives the comparison of the computation times of the AL and SMOTE. Note that SMOTE requires significantly long preprocessing time that dominates the training time in large datasets, for example, MNIST-8 dataset. The low computation cost, scalability, and high prediction performance of AL suggest that AL can efficiently handle the class imbalance problem.

## 6.4 ADAPTIVE RESAMPLING WITH ACTIVE LEARNING

The analysis in Section 6.3.5 shows the effectiveness of AL on imbalanced datasets without employing any resampling techniques. This section extends the discussion on the effectiveness of AL for imbalanced data classification and demonstrates that even in cases where resampling is the preferred approach, AL can still be used to significantly improve the classification performance.

In supervised learning, a common strategy to overcome the rarity problem is to resample the original dataset to decrease the overall level of class imbalance. Resampling is done either by oversampling the minority (positive) class and/or under-sampling the majority (negative) class until the classes are approximately equally represented [28, 30–32]. Oversampling, in its simplest form, achieves a more balanced class distribution either by duplicating minority class instances or introducing new synthetic instances that belong to the minority class [30]. No information is lost in oversampling as all original instances of the minority and the majority classes are retained in the oversampled dataset. The other strategy to reduce the class imbalance is under-sampling, which eliminates some majority class instances mostly by RS.

Even though both approaches address the class imbalance problem, they also suffer some drawbacks. The under-sampling strategy can potentially sacrifice the prediction performance of the model, as it is possible to discard informative instances that the learner might benefit. Oversampling strategy, on the other hand, can be computationally overwhelming in cases with large training sets—if a complex oversampling method is used; a large computational effort must be expended during preprocessing of the data. Worse, oversampling causes longer training time during the learning process because of the increased number of training instances. In addition to suffering from increased runtime due to added computational complexity, it also necessitates an increased memory footprint due to the extra storage requirements of artificial instances. Other costs associated with the learning process (i.e., extended kernel matrix in kernel classification algorithms) further increase the burden of oversampling.

### 6.4.1 VIRTUAL: Virtual Instance Resampling Technique Using Active Learning

In this section, the focus is on the oversampling strategy for imbalanced data classification and investigate how it can benefit from the principles of AL. Our goal is to remedy the efficiency drawbacks of oversampling in imbalanced data classification and use an AL strategy to generate minority class instances only if they can be useful to the learner. VIRTUAL (virtual instance resampling technique using active learning) [22] is a hybrid method of oversampling and AL that forms an adaptive technique for resampling of the minority class instances. In contrast to traditional oversampling techniques that act as an *offline* step that generates virtual instances of the minority class before the training process, VIRTUAL leverages the power of AL to intelligently and adaptively oversample the data *during* training,

removing the need for an offline and separate preprocessing stage. Similar to the discussions in the previous section, VIRTUAL also employs an online SVM-based AL strategy. In this setting, the informativeness of instances is measured by their distance to their hyperplane, and the most informative instances are selected as the support vectors. VIRTUAL targets the set of support vectors during training, and resamples new instances based on this set. Since most support vectors are found during early stages of training, corresponding virtual examples are also created in the early stages. This prevents the algorithm from creating excessive and redundant virtual instances, and integrating the resampling process into the training stage improves the efficiency and generalization performance of the learner compared to other competitive oversampling techniques.

***6.4.1.1 Active Selection of Instances*** Let $S$ denote the pool of real and virtual training examples unseen by the learner at each AL step. Instead of searching for the most informative instance among all the samples in $S$, VIRTUAL employs the small-pool AL strategy that is discussed in section 6.3.2. From the small pool, VIRTUAL selects an instance that is closest to the hyperplane according to the current model. If the selected instance is a real positive instance (from the original training data) and becomes a support vector, VIRTUAL advances to the oversampling step, explained in the following section. Otherwise, the algorithm proceeds to the next iteration to select another instance.

***6.4.1.2 Virtual Instance Generation*** VIRTUAL oversamples the real minority instances (instances selected from the minority class of the original training data) that become support vectors in the current iteration. It selects the $k$ nearest minority class neighbors $(x_{i\to 1} \cdots x_{i\to k})$ of $x_i$ based on their similarities in the kernel-transformed higher dimensional feature space. We limit the neighboring instances of $x_i$ to the minority class so that the new virtual instances lie within the minority class distribution. Depending on the amount of oversampling required, the algorithm creates $v$ virtual instances. Each virtual instance lies on any of the line segments joining $x_i$ and its neighbor $x_{i\to j}$ $(j = 1, \ldots, k)$. In other words, a neighbor $x_{i\to j}$ is randomly picked and the virtual instance is created as $\overline{x}_v = \lambda \cdot x_i + (1 - \lambda)x_{i\to j}$, where $\lambda \in (0, 1)$ determines the placement of $\overline{x}_v$ between $x_i$ and $x_{i\to j}$. All $v$ virtual instances are added to $S$ and are eligible to be picked by the active learner in the subsequent iterations.

The pseudocode of VIRTUAL given in Algorithm 6.1 depicts the two processes described previously. In the beginning, the pool $S$ contains all real instances in the training set. At the end of each iteration, the instance selected is removed from $S$, and any virtual instances generated are included in the pool $S$. In this pseudocode, VIRTUAL terminates when there are no instances in $S$.

### 6.4.2 Remarks on VIRTUAL

We compare VIRTUAL with a popular oversampling technique SMOTE. Figure 6.7a shows the different behaviors of how SMOTE and VIRTUAL create virtual

**Algorithm 6.1** VIRTUAL

**Define:**
$X = \{x_1, x_2, \cdots, x_n\}$ : training instances
$X_R^+$ : positive real training instances
$S$ : pool of training instances for SVM
$v$ : # virtual instances to create in each iteration
$L$ : size of the small set of randomly picked samples for active sample selection

1. Initialize $S \leftarrow X$
2. **while** $S \neq \emptyset$
3. // *Active sample selection step*
4. $d_{min} \leftarrow \infty$
5. **for** $i \leftarrow 1$ to $L$
6. $x_j \leftarrow RandomSelect(S)$
7. **If** $d(x_j, hyperplane) < d_{min}$
8. $d_{min} \leftarrow d(x_j, hyperplane)$
9. $candidate \leftarrow x_j$
10. **end**
11. **end**
12. $x_s \leftarrow candidate$
13. // *Virtual Instance Generation*
14. **If** $x_s$ becomes SV **and** $x_s \in X_R^+$
15. $K \leftarrow k$ nearest neighbors of $x_s$
16. **for** $i \leftarrow 1$ to $v$
17. $x_m \leftarrow RandomSelect(K)$
18. // Create a virtual positive instance $x_{s,m}^v$ between $x_s$ and $x_m$
19. $\lambda$=random number between 0 and 1
20. $x_{s,m}^v = \lambda \cdot x_s + (1 - \lambda)x_m$
21. $S \leftarrow S \cup x_{s,m}^v$
22. **end**
23. **end**
24. $S \leftarrow S - x_s$
25. **end**

instances for the minority class. SMOTE creates virtual instance(s) for each positive example (Figure 6.7b), whereas VIRTUAL creates the majority of virtual instances around the positive canonical hyperplane (shown with a dashed line in Figure 6.7). Note that a large portion of virtual instances created by SMOTE is far away from the hyperplane and thus are not likely to be selected as support vectors. VIRTUAL, on the other hand, generates virtual instances near the real positive support vectors adaptively in the learning process. Hence, the virtual instances are near the hyperplane and thus are more informative.

We further analyze the computation complexity of SMOTE and VIRTUAL. The computation complexity of VIRTUAL is $O(|SV(+)| \cdot v \cdot \mathcal{C})$, where $v$ is the

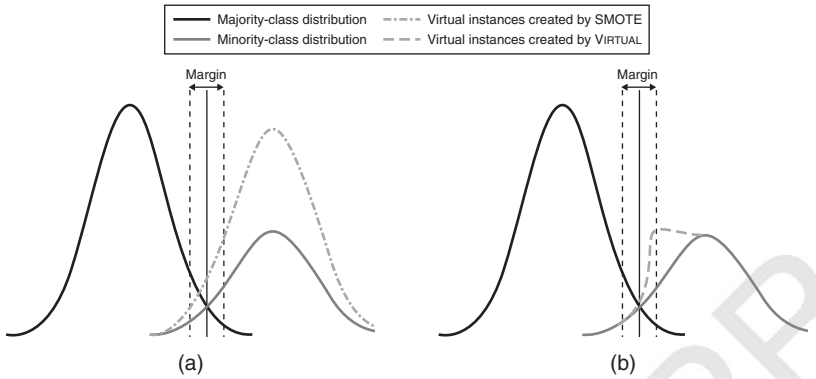


**Figure 6.7** Comparison of oversampling the minority class with SMOTE and VIRTUAL. (a) Oversampling with SMOTE and (b) oversampling with VIRTUAL.

number of virtual instances created for a real positive support vector in each iteration, $|SV(+)|$ is the number of positive support vectors, and $\mathcal{C}$ is the cost of finding k nearest neighbors. The computation complexity of SMOTE is $O(\left|X_R^+\right| \cdot v \cdot \mathcal{C})$, where $\left|X_R^+\right|$ is the number of positive training instances. $\mathcal{C}$ depends on the approach for finding $k$ nearest neighbors. The naive implementation searches all $N$ training instances for the nearest neighbors and thus $\mathcal{C} = kN$. Using advanced data structure such as kd-tree, $\mathcal{C} = k \log N$. Since $|SV(+)|$ is typically much less than $\left|X_R^+\right|$, VIRTUAL incurs lower computation overhead than SMOTE. Also, with fewer virtual instances created, the learner is less burdened with VIRTUAL. We demonstrate with empirical results that the virtual instances created with VIRTUAL are more informative and the prediction performance is also improved.

### 6.4.3 Experiments

We conduct a series of experiments on Reuters-21578 and four UCI datasets to demonstrate the efficacy of VIRTUAL. The characteristics of the datasets are detailed in Reference 22. We compare VIRTUAL with two systems, AL and SMOTE. AL adopts the traditional AL strategy without preprocessing or creating any virtual instances during learning. SMOTE, on the other hand, preprocesses the data by creating virtual instances before training and uses RS in learning. Experiments elicit the advantages of adaptive virtual sample creation in VIRTUAL.

Figures 6.8 and 6.9 provide details on the behavior of the three algorithms, SMOTE, AL, and VIRTUAL. For the Reuters datasets (Fig. 6.9), note that in all the 10 categories, VIRTUAL outperforms AL in g-means metric after saturation. The difference in performance is most pronounced in the more imbalanced categories, for example, *corn*, *interest*, and *ship*. In the less imbalanced datasets such as *acq* and *earn*, the difference in $g$-means of both methods is less noticeable. The $g$-means of SMOTE converges much slower than both AL and VIRTUAL.

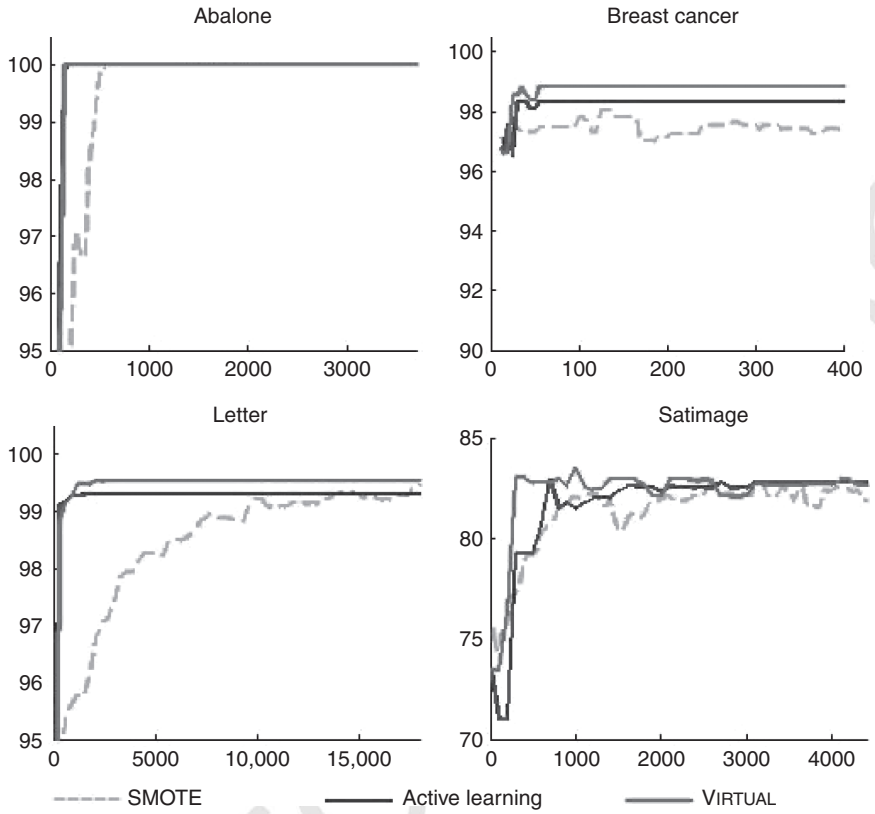


**Figure 6.8** Comparison of SMOTE, AL, and VIRTUAL on *UCI* datasets. We present the $g$-means (%) ($y$-axis) of the current model for the test set versus the number of training samples ($x$-axis) seen.

However, SMOTE converges to higher $g$-means than AL in some of the categories, indicating that the virtual positive examples provide additional information that can be used to improve the model. VIRTUAL converges to the same or even higher $g$-means than SMOTE while generating fewer virtual instances. For the UCI datasets (Fig. 6.8), VIRTUAL performs as well as AL in *abalone* in $g$-means and consistently outperforms AL and SMOTE in the other three datasets.

In Table 6.4, the support vector imbalance ratios of all the three methods are lower than the data imbalance ratio, and VIRTUAL achieves the most balanced ratios of positive and negative support vectors in the Reuters datasets. Despite that the datasets used have different data distributions, the portion of virtual instances become support vectors in VIRTUAL consistently and significantly higher than that in SMOTE. These results confirm the previous discussion that VIRTUAL is more effective in generating informative virtual instances.

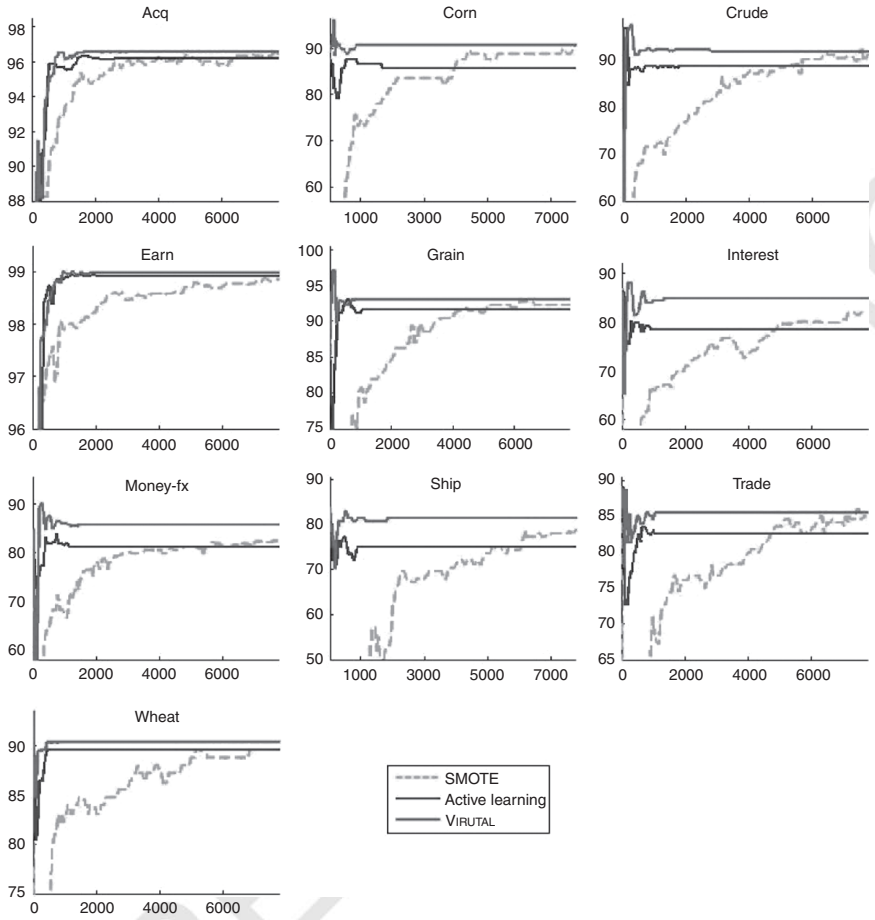


**Figure 6.9** Comparison of SMOTE, AL, and VIRTUAL on 10 largest categories of *Reuters-21578*. We show the $g$-means (%) ($y$-axis) of the current model for the test set versus the number of training samples ($x$-axis) seen.

Table 6.4 presents the $g$-means and the total learning time for SMOTE, AL, and VIRTUAL. Classical batch SVM's $g$-means values are also provided as a reference point. In Reuters datasets, VIRTUAL yields the highest $g$-means in all categories. Table 6.4 shows the effectiveness of adaptive virtual instance generation. In categories *corn*, *interest*, and *ship* with high class imbalance ratio, VIRTUAL gains substantial improvement in $g$-means. Compared to AL, VIRTUAL requires additional time for the creation of virtual instances and selection of those that may become support vectors. Despite this overhead, VIRTUAL's training times are comparable with those of AL. In the cases where minority examples are abundant, SMOTE demands substantially longer time to create virtual instances than VIRTUAL. But as the rightmost columns in Table 6.3 show, only a small fraction

**Table 6.4 g-Means and Total Learning Time Using SMOTE, AL, and VIRTUAL**

| | | g-Means (%) | | | | Total Learning time (s) | | |
|---|---|---|---|---|---|---|---|---|
| | Dataset | Batch | SMOTE | AL | VIRTUAL | SMOTE | AL | VIRTUAL |
| Reuters | acq | 96.19 (3) | 96.21 (2) | 96.19 (3) | **96.54 (1)** | 2271 | 146 | 203 |
| | corn | 85.55 (4) | 89.62 (2) | 86.59 (3) | **90.60 (1)** | 74 | 43 | 66 |
| | crude | 88.34 (4) | 91.21 (2) | 88.35 (3) | **91.74 (1)** | 238 | 113 | 129 |
| | earn | 98.92 (3) | **98.97 (1)** | 98.92 (3) | **98.97 (1)** | 4082 | 121 | 163 |
| | grain | 91.56 (4) | 92.29 (2) | 91.56 (4) | **93.00 (1)** | 296 | 134 | 143 |
| | interest | 78.45 (4) | 83.96 (2) | 78.45 (4) | **84.75 (1)** | 192 | 153 | 178 |
| | money-fx | 81.43 (3) | 83.70 (2) | 81.08 (4) | **85.61 (1)** | 363 | 93 | 116 |
| | ship | 75.66 (3) | 78.55 (2) | 74.92 (4) | **81.34 (1)** | 88 | 75 | 76 |
| | trade | 82.52 (3) | 84.52 (2) | 82.52 (3) | **85.48 (1)** | 292 | 72 | 131 |
| | wheat | 89.54 (3) | 89.50 (4) | 89.55 (2) | **90.27 (1)** | 64 | 29 | 48 |
| UCI | Abalone | **100 (1)** | **100 (1)** | **100 (1)** | **100 (1)** | 18 | 4 | 6 |
| | Breast | 98.33 (2) | 97.52 (4) | 98.33 (2) | **98.84 (1)** | 4 | 1 | 1 |
| | letter | 99.28 (3) | 99.42 (2) | 99.28 (3) | **99.54 (1)** | 83 | 5 | 6 |
| | Satimage | **83.57 (1)** | 82.61 (4) | 82.76 (3) | 82.92 (2) | 219 | 18 | 17 |

"Batch" corresponds to the classical SVM learning in batch setting without resampling. The numbers in parentheses denote the rank of the corresponding method in the dataset.

of the virtual instances created by SMOTE becomes support vectors. Therefore, SMOTE spends much time to create virtual instances that will not be used in the model. On the other hand, VIRTUAL has already a short training time and uses this time to create more informative virtual instances. In Table 6.4, the numbers in parentheses give the ranks of the $g$-means prediction performance of the four approaches. The values in bold correspond to a win and VIRTUAL wins in nearly all datasets. The Wilcoxon-signed-rank test (two-tailed) between VIRTUAL and its nearest competitor SMOTE reveals that the zero median hypothesis can be rejected at the significance level 1% ($p = 4.82 \times 10^{-4}$), implying that VIRTUAL performs statistically better than SMOTE in these 14 datasets. These results demonstrate the importance of creating synthetic samples from the informative examples rather than all the examples.

## 6.5 DIFFICULTIES WITH EXTREME CLASS IMBALANCE

Practical applications rarely provide us with data that have equal numbers of training instances of all the classes. However, in many applications, the imbalance in the distribution of naturally occurring instances is extreme. For example, when labeling web pages to identify specific content of interest, uninteresting pages may outnumber interesting ones by a million to one or worse (consider identifying web pages containing hate speech, in order to keep advertisers off them, cf. Reference 33).

The previous sections have detailed the techniques that have been developed to cope with moderate class imbalance. However, as class imbalances tends toward the extreme, AL strategies can fail completely—and this failure is not simply due to the challenges of learning models with skewed class distributions, which has received a good bit of study and has been addressed throughout this book. The lack of labeled data compounds the problem because techniques cannot concentrate on the minority instances, as the techniques are unaware which instances to focus on.

Figure 6.10 compares the AUC of logistic regression text classifiers induced by labeled instances selected with uncertainty sampling and with RS. The learning

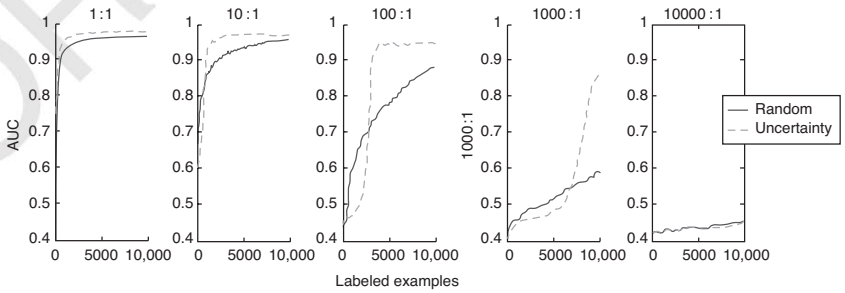


**Figure 6.10** Comparison of random sampling and uncertainty sampling on the same dataset with induced skews ranging from 1 : 1 to 10,000 : 1.

task is to differentiate sports web pages from nonsports pages. Depending on the source of the data (e.g., different impression streams from different online advertisers), one could see very different degrees of class skew in the population of relevant web pages. The panels in Figure 6.10, left-to-right, depict increasing amounts of induced class skew. On the far left, we see that for a balanced class distribution, uncertainty sampling is indeed better than RS. For a 10 : 1 distribution, uncertainty sampling has some problems very early on, but soon does better than RS—even more so than in the balanced case. However, as the skew begins to get large, not only does RS start to fail (it finds fewer and fewer minority instances, and its learning suffers), uncertainty sampling does substantially worse than random for a considerable amount labeling expenditure. In the most extreme case shown,[6] both RS and uncertainty sampling simply fail completely. RS effectively does not select any positive examples, and neither does uncertainty sampling.[7]

A practitioner well versed in the AL literature may decide he/she should use a method other than uncertainty sampling in such a highly skewed domain. A variety of techniques have been discussed in Sections 6.2–6.4 for performing AL specifically under class imbalance, including [18–21, 35], as well as for performing density-sensitive AL, where the geometry of the problem space is specifically included when making selections, including [13–15, 17, 36]. While initially appealing, as problems become increasingly difficult, these techniques may not provide results better than more traditional AL techniques—indeed class skews may be sufficiently high to thwart these techniques completely [33].

As discussed later in Section 6.8.1, Attenberg and Provost [33] proposed an alternative way of using human resources to produce labeled training set, specifically tasking people with finding class-specific instances ("guided learning") as opposed to labeling specific instances. In some domains, finding such instances may even be cheaper than labeling (per instance). Guided learning can be much more effective per instance acquired; in one of the Attenberg and Provost's experiments, it outperformed AL as long as searching for class-specific instances was less than eight times more expensive (per instance) than labeling selected instances. The generalization performance of guided learning is shown in Figure 6.12, discussed in Section 6.8.1 for the same setting as Figure 6.10.

## 6.6 DEALING WITH DISJUNCTIVE CLASSES

Even more subtly still, certain problem spaces may not have such an extreme class skew, but may still be particularly difficult because they possess important but very small disjunctive subconcepts, rather than simple continuously dense

[6] 10,000 : 1—still orders of magnitude less skewed than some categories.

[7] The curious behavior of AUC $< 0.5$ here is due to overfitting. Regularizing the logistic regression "fixes" the problem, and the curve hovers about 0.5. See another article in this issue for more insight on models exhibiting AUC $< 0.5$ [34].

regions of minority and majority instances. Prior research has shown that such "small disjuncts" can comprise a large portion of a target class in some domains [37]. For AL, these small subconcepts act as same as rare classes: if a learner has seen no instances of the subconcept, how can it "know" which instances to label? Note that this is not simply a problem of using the wrong loss function: in an AL setting, the learner does not even know that the instances of the subconcept are misclassified if no instances of a subconcept have yet been labeled. Nonetheless, in a research setting (where we know all the labels), using an undiscriminative loss function, such as classification accuracy or even the AUROC, may result in the researcher not even realizing that an important subconcept has been missed.

To demonstrate how small disjuncts influence (active) model learning, consider the following text classification problem: separating the *Science* articles from the *non-Science* articles within a subset of the 20 newsgroups benchmark set (with an induced class skew of 80−1). Figure 6.11 examines graphically the relative positions of the minority instances through the AL. The black curve shows the AUC (right vertical axis) of the models learned by a logistic regression classifier using uncertainty sampling, rescaled as follows. At each epoch, we sort all instances by their predicted probability of membership in the majority class, $\hat{P}(y = 0|x)$. The black dots in Figure 6.11 represent the minority class instances, with the value on the left vertical axis showing their relative position in this sorted list. The $x$-axis shows the AL epoch (here each epoch requests 30 new instances from the pool). The black trajectories mostly show instances' relative

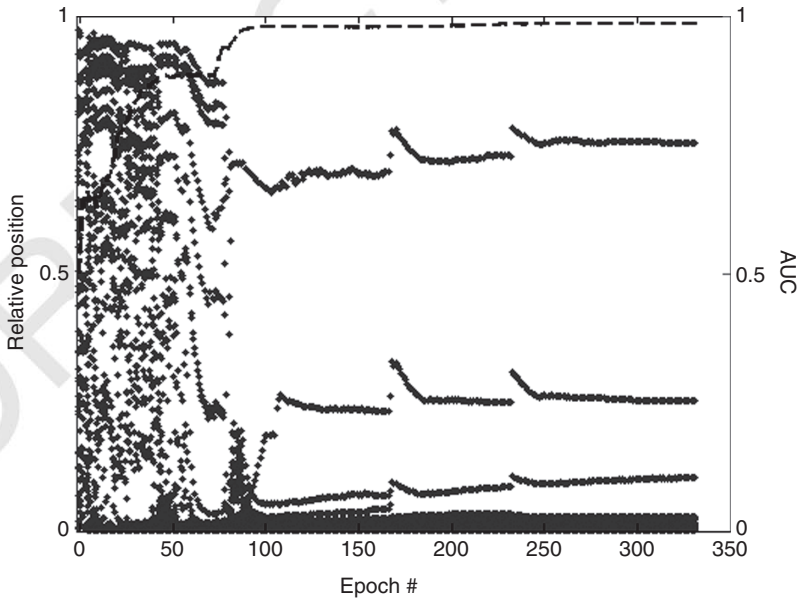


**Figure 6.11** A comparison of the learned model's ordering of the instance pool along with the quality of the cross-validated AUC.

positions changing. Minority instances drop down to the very bottom (certain minority) either because they get chosen for labeling, or because labeling some other instance caused the model to "realize" that they are minority instances.

We see that, early on, the minority instances are mixed all throughout the range of estimated probabilities, even as the generalization accuracy increases. Then the model becomes good enough that, abruptly, few minority class instances are misclassified (above $\hat{P} = 0.5$). This is the point where the learning curve levels off for the first time. However, notice that there still are some residual misclassified minority instances, and in particular that there is a cluster of them for which the model is relatively certain they are *majority* instances. It takes many epochs for the AL to select one of these, at which point the generalization performance increases markedly—apparently, this was a subconcept that was strongly misclassified by the model, and so it was not a high priority for exploration by the AL.

On the 20 newsgroups dataset, we can examine the minority instances for which $\hat{P}$ decreases the most in that late rise in the AUC curve (roughly, they *switch* from being misclassified on the lower plateau to being correctly classified afterward). Recall that the minority (positive) class here is "Science" newsgroups. It turns out that these late-switching instances are members of the cryptography (sci.crpyt) subcategory. These pages were classified as non-Science presumably because before having seen any positive instances of the subcategory, they looked much more the same as the many computer-oriented subcategories in the (much more prevalent) non-Science category. As soon as a few were labeled as Science, the model generalized its notion of Science to include this subcategory (apparently pretty well).

Density-sensitive AL techniques did not improve on uncertainty sampling for this particular domain. This was surprising, given the support we have just provided for our intuition that the concepts are disjunctive. One would expect a density-oriented technique to be appropriate for this domain. Unfortunately, in this domain—and we conjecture that this is typical of many domains with extreme class imbalance—the *majority* class is *even more disjunctive* than the minority class. For example, in 20 newsgroups, Science indeed has four very different subclasses. However, non-Science has 16 (with much more variety). Techniques that, for example, try to find as-of-yet unexplored clusters in the instance space are likely to select from the vast and varied majority class. We need more research on dealing with highly disjunctive classes, especially when the less interesting[8] class is more varied than the main class of interest.

## 6.7 STARTING COLD

The *cold start problem* has long been known to be a key difficulty in building effective classifiers quickly and cheaply via AL [13, 16]. Since the quality of

[8]How interesting a class is if it could be measured by its relative misclassification cost, for example.

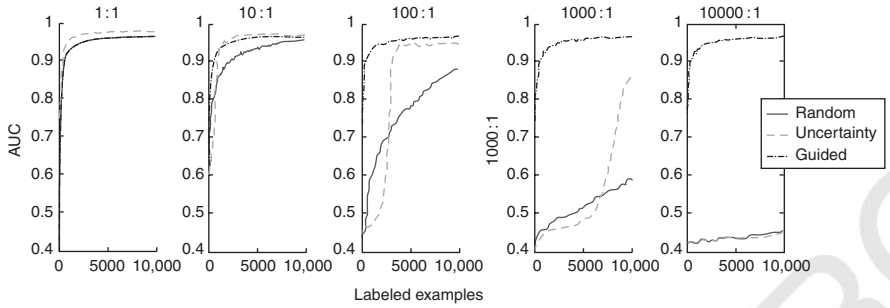


**Figure 6.12** Comparison of random sampling and uncertainty sampling and guided learning on the problem shown in Figure 6.10.

data selection directly depends on the understanding of the space provided by the "current" model, early stages of acquisitions can result in a vicious cycle of uninformative selections, leading to poor quality models and therefore to additional poor selections.

The difficulties posed by the cold start problem can be particularly acute in highly skewed or disjunctive problem spaces; informative instances may be difficult for AL to find because of their variety or rarity, potentially leading to substantial waste in data selection. Difficulties early in the AL process can, at least in part, be attributed to the base classifier's poor understanding of the problem space. This cold start problem is particularly acute in otherwise difficult domains. Since the value of subsequent label selections depends on base learner's understanding of the problem space, poor selections in the early phases of AL propagate their harm across the learning curve.

In many research papers, AL experiments are "primed" with a preselected, often class-balanced training set. As pointed out by Attenberg and Provost [33], if the possibility and procedure exist to procure a class-balanced training set to start the process, maybe the most cost-effective model-development alternative is not to do AL at all, but to just continue using this procedure. This is exemplified in Figure 6.12 [33], where the dot-and-hatched lines show the effect of investing resources to continue to procure a class-balanced, but otherwise random, training set (as compared with the active acquisition shown in Figure 6.10).

## 6.8 ALTERNATIVES TO ACTIVE LEARNING FOR IMBALANCED PROBLEMS

In addition to traditional label acquisition for unlabeled examples, there are other sorts of data that may be acquired at a cost for the purpose of building or improving statistical models. The intent of this section is to provide the reader with a brief overview of some alternative techniques for active data acquisition for predictive model construction in a cost-restrictive setting. We begin this setting with a discussion of class-conditional example acquisition, a paradigm related to AL

where examples are drawn from some available unlabeled pool in accordance to some predefined class proportion. We then go on into Section 6.8.2 to touch on active feature labeling (AFL) and active dual supervision (ADS). These two paradigms attempt to replace or supplement traditional supervised learning with class-specific associations on certain feature values. While this set of techniques requires specialized models, significant generalization performance can often be achieved at a reasonable cost by leveraging explicit feature/class relationships. This is often appealing in the active setting, where it is occasionally less challenging to identify class-indicative feature values than it is to find quality training data for labeling, particularly in the imbalanced setting.

### 6.8.1 Class-Conditional Example Acquisition

Imagine as an alternative to the traditional AL problem setting, where an oracle is queried in order to assign examples to specially selected unlabeled examples, a setting where an oracle is charged with selecting exemplars from the underlying problem space in accordance to some predefined class ratio. Consider as a motivational example, the problem of building predictive models based on data collected through an "artificial nose" with the intent of "sniffing out" explosive or hazardous chemical compounds [38–40]. In this setting, the reactivity of a large number of chemicals is already known, representing label-conditioned pools of available instances. However, producing these chemicals in a laboratory setting and running the resultant compound through the artificial nose may be an expensive, time-consuming process. While this problem may seem quite unique, many data acquisition tasks may be cast into a similar framework.

A much more general issue in selective data acquisition is the amount of control ceded to the "oracle" doing the acquisition. The work discussed so far assumes that an oracle will be queried for some specific value, and the oracle simply returns that value. However, if the oracle is actually a person, he or she may be able to apply considerable intelligence and other resources to "guide" the selection. Such guidance is especially helpful in situations where some aspect of the data is rare—where purely data-driven strategies are particularly challenged.

As discussed throughout this work, in many practical settings, one class is quite rare. As an example motivating the application of class-conditional example acquisition in practice, consider building a predictive model from scratch designed to classify web pages containing a particular topic of interest. While large absolute numbers of such web pages may be present on the web, they may be outnumbered by uninteresting pages by a million to one or worse (take, for instance, the task of detecting and removing hate speech from the web [33]). As discussed in Section 6.5, such extremely imbalanced problem settings present a particularly insidious difficulty for traditional AL techniques. In a setting with a 10,000 : 1 class ratio, a reasonably large labeling budget could be expended without observing a single minority example.[9]

[9]Note that in practice, such extremely imbalanced problem settings may actually be quite common.

***Previously, we have discussed a plethora of AL techniques specifically tuned for the high skew setting [18–21] as well as techniques where the geometry and feature density of the problem space are explicitly included when making instance selections [13–15, 17, 35, 36]. These techniques, as initially appealing as they may seem, may fail just as badly as traditional AL techniques. Class skew and subconcept rarity discussed in Section 6.6 may be sufficient to thwart them completely [33, 41].

However, in many of these extremely difficult settings, we can task humans to search the problem space for rare cases, using tools (such as search engines) and possibly interacting with the base learner. Consider the motivating example of hate speech classification on the web (from above). While an active learner may experience difficulty in exploring the details of this rare class, a human oracle armed with a search interface is likely to expose examples of hate speech quite easily. In fact, given the coverage of modern web search engines, a human can produce interesting examples from a much larger sample of the problem space far beyond that which is likely to be contained in a sample pool for AL. This is critical due to hardware-imposed constraints on the size of the pool that an active learner is able to choose from—for example, a random draw of several hundred thousand examples from the problem space may not even contain any members of the minority class or of rare disjuncts!

Guided learning is the general process of utilizing oracles to search the problem space, using their domain expertise to *seek* instances representing the interesting regions of the problem space. Figure 6.13 presents the general guided learning setting. Here, given some interface enabling the search over the domain in question, an oracle searches for interesting examples, which are either supplemented with an implicit label by the oracle, or sent for explicit labeling as a second step. These examples are then added to the training set and a model is retrained. Oracles can leverage their background knowledge of the problem being faced. In addition to simply being charged with the acquisition of class-specific examples,

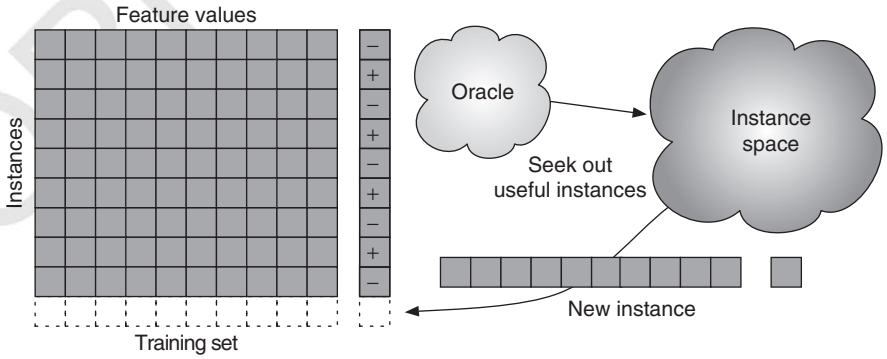


**Figure 6.13** Guided learning: an oracle selecting useful examples from the instance space.

by allowing the oracle to interact with the base learner, confusing instances, those that "fool" the model can be sought out from the problem space and used for subsequent training in the form of human-guided uncertainty sampling. This interaction with the base learner can be extended a step further—by allowing the humans to challenge the predictive accuracy of the problem space may potentially reveal "problem areas," portions of the example space where the base model performs poorly that might not be revealed through traditional techniques such as cross-validation studies [42].

Guided learning, along with alternative problem settings such as that faced by the artificial nose discussed earlier deals with situations where an oracle is able to provide "random" examples in arbitrary class proportions. It now becomes interesting to consider just what this class proportion should be? This problem appears to face the inverse of the difficulties faced by AL—labels essentially come for free, while the independent feature values are *completely* unknown and must be gathered at a cost. In this setting, it becomes important to consider the question: "In what proportion should classes be represented in a training set of a certain size?" [43].

Let us call the problem of proportioning class labels in a selection of $n$ additional training instances, "active class selection" (ACS) [38–40, 43]. This process is exemplified in Figure 6.14. In this setting, large, class-conditioned (virtual) pools of available instances with completely hidden feature values are assumed. At each epoch, $t$, of the ACS process, the task is to leverage the current model when selecting examples from these pools in a proportion believed to have the greatest effectiveness for improving the generalization performance

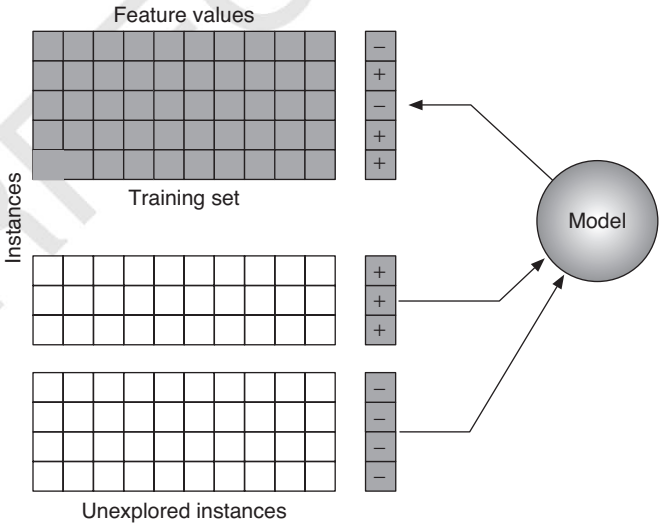


**Figure 6.14** Active class selection: gathering instances from random class-conditioned fonts in a proportion believed to offer greatest improvement in generalization performance.

of this model. The feature values for each instance are then collected and the complete instances are added to the training set. The model is reconstructed and the processes is repeated until $n$ examples are obtained (because the budget is exhausted or some other stopping criterion is met, such as a computational limit). Note that this situation can be considered to be a special case of the instance completion setting of active feature-value acquisition (cf. Reference 44). It is a degenerate special case because, before the selection, there is no information at all about the instances other than their classes.

For the specific problem at the heart of ACS, the extreme lack of information to guide selection leads to the development of unique uncertainty and utility estimators, which, in the absence of predictive covariates, require unique approximations.[10] While alternative approaches to ACS have emerged, for thematic clarity, uncertainty-based and expected-utility-based approaches will be presented first. Note that because effective classification requires that both sides of a prediction boundary be represented, unlike typical AL techniques, ACS typically *samples* classes from their respective score distributions [45, 46].

***6.8.1.1 Uncertainty-Based approaches*** This family of techniques for performing ACS is based on the volatility in the predictions made about certain classes—those classes whose cross-validated predictions are subject to the most change between successive epochs of instance selection are likely to be based on an uncertain predictor and amenable to refinement by the incorporation of additional training data [38, 40]. Analogous to the case of more traditional uncertainty-based data acquisition, several heuristics have been devised to capture the notion of variability.

One measure of the uncertainty of a learned model is how volatile its predictive performance is in the face of new training data. Take a typical learning curve, for instance, those presented in Figure 6.6. Notice that the modeling is much more volatile at the left side of the figure, showing large changes in generalization performance for the same amount of new training data. We can think that as the predictor gains knowledge of the problem space, it tends to solidify in the face of data, exhibiting less change and greater certainty. For ACS, we might wonder if the learning curves will be equally steep regardless of the class of the training data [38–40]. With this in mind, we can select instances at epoch $t$ from the classes in proportion to their improvements in accuracy at $t-1$ and $t-2$. For example, we could use cross-validation to estimate the generalization performance of the classifier with respect to each class, $\mathcal{A}(c)$; class $c$ can then be sampled according to:

$$p^t_{\mathcal{A}}(c) \propto \frac{\max\left\{0, \mathcal{A}^{t-1}(c) - \mathcal{A}^{t-2}(c)\right\}}{\sum_{c'} \max\left\{0, \mathcal{A}^{t-1}(c') - \mathcal{A}^{t-2}(c')\right\}},$$

[10]In realistic settings, for instance, such as potential application for ACS, guided learning, this lack of information assumption may be softened.

Alternatively, we could consider general volatility in class members' predicted labels, beyond improvement in the model's ability to predict the class. Again, using cross-validated predictions at successive epochs, it is possible to isolate members of each class, and observe changes in the predicted class for each instance. For example, when the predicted label of a given instance changes between successive epochs, we can deem the instance to have been *redistricted* [38–40]. Again considering the level of volatility in a model's predictions to be a measurement of uncertainty, we can sample classes at epoch $t$ according to each classes' proportional measure of redistricting:

$$p^t_{\mathcal{R}}(c) \propto \frac{\frac{1}{|c|}\sum_{x\in c}\mathbb{I}(f^{t-1}(x) \neq f^{t-2}(x))}{\sum_{c'}\frac{1}{|c'|}\sum_{x\in c'}\mathbb{I}(f^{t-1}(x) \neq f^{t-2}(x))},$$

where $\mathbb{I}(\cdot)$ is an indicator function taking the value of 1 if its argument is true and 0 otherwise. $f^{t-1}(x)$ and $f^{t-2}(x)$ are the predicted labels, for instance, $x$ from the models trained at epoch $t-1$ and $t-2$, respectively [38–40].

***6.8.1.2 Expected Class Utility*** The previously described ACS heuristics are reliant on the assumption that adding examples belonging to a particular class will improve the predictive accuracy with respect to that class. This does not directly estimate the utility of adding members of a particular class to a model's overall performance. Instead, it may be preferable to select classes whose instances' presence in the training set will reduce a model's misclassification cost by the greatest amount in expectation.

Let $\text{cost}(c_i|c_j)$ be the cost of predicting $c_i$ on an instance $x$ whose true label is $c_j$. Then the expected empirical misclassification cost over a sample dataset, $\mathbb{D}$, is:

$$\hat{R} = \frac{1}{|\mathbb{D}|}\sum_{x\in\mathbb{D}}\sum_i \hat{P}(c_i|x)\text{cost}(c_i|y),$$

where $y$ is the correct class for a given $x$. Typically in the ACS setting, this expectation would be taken over the training set (e.g., $\mathbb{D} = T$), preferably using cross-validation. In order to reduce this risk, we would like to select examples from class $c$, leading to the greatest reduction in this expected risk [39].

Consider a predictive model $\hat{P}^{T\cup c}(\cdot|x)$, a model built on the training set, $T$, supplemented with an arbitrary example belonging to class $c$. Given the opportunity to choose an additional class-representative example to the training pool, we would like to select the class that reduces the expected risk by the greatest amount:

$$\overline{c} = \arg\max_c U(c),$$

where

$$U(c) = \frac{1}{|\mathbb{D}|}\sum_{x\in\mathbb{D}}\sum_i \hat{P}^T(c_i|x)\text{cost}(c_i|y) - \frac{1}{|\mathbb{D}|}\sum_{x\in\mathbb{D}}\sum_i \hat{P}^{T\cup c}(c_i|x)\text{cost}(c_i|y).$$

Of course the benefit of adding additional examples on a test dataset is unknown. Furthermore, the impact of a particular class's examples may vary depending on the feature values of particular instances. In order to cope with these issues, we can estimate via cross-validation on the training set. Using sampling, we can try various class-conditional additions and compute the expected benefit of a class across that class's representatives in $T$, assessed on the testing folds. The earlier-mentioned utility then becomes:

$$\hat{U}(c) = E_{x \in c}\left[\frac{1}{|\mathbb{D}|}\sum_{x \in \mathbb{D}}\sum_{i} \hat{P}^{T}(c_i|x)\text{cost}(c_i|y) - \frac{1}{|\mathbb{D}|}\sum_{x \in \mathbb{D}}\sum_{i} \hat{P}^{T \cup c}(c_i|x)\text{cost}(c_i|y)\right].$$

Note that it is often preferred to add examples in batch. In this case, we may wish to sample from the classes in proportion to their respective utilities:

$$p^{t}_{\hat{U}}(c) \propto \frac{\hat{U}(c)}{\sum'_{c} \hat{U}(c)'}.$$

Further, diverse class-conditional acquisition costs can be incorporated, utilizing $\hat{U}(c)/\omega_c$ in place of $\hat{U}(c)$, where $\omega_c$ is the (expected) cost of acquiring the feature vector of an example in class $c$.

***6.8.1.3 Alternative Approaches to ACS*** In addition to uncertainty-based and utility-based techniques, there are several alternative techniques for performing ACS. Motivated by empirical results showing that barring any domain-specific information, when collecting examples for a training set of size $n$, a balanced class distribution tends to offer reasonable AUC on test data [43, 47], a reasonable baseline approach to ACS is simply to select classes in balanced proportion.

Search strategies may alternately be employed in order to reveal the most effective class ratio at each epoch. Utilizing a nested cross-validation on the training set, the space of class ratios can be explored, with the most favorable ratio being utilized at each epoch. Note that it is not possible to explore all possible class ratios in all epochs, without eventually spending too much on one class or another. Thus, as we approach $n$, we can narrow the range of class ratios, assuming that there is a problem-optimal class ratio that will become more apparent as we obtain more data [43].

It should be noted that many techniques employed for building classification models assume an identical or similar training and test distribution. Violating this assumption may lead to biased predictions on test data where classes preferentially represented in the training data are predicted more frequently. In particular, increasing the prior probability of a class increases the posterior probability of the class, moving the classification boundary for that class so that more cases are classified into that class" [48, 49]. Thus in settings where instances are selected specifically in proportions different from those seen in the wild, posterior

probability estimates should be properly calibrated to be aligned with the test data, if possible [43, 49, 50].

Prior work has repeatedly demonstrated the benefits of performing ACS beyond simply selecting random examples from an example pool for acquisition or simply using uniformly balanced selection. However, in many cases, simply casting what would typically be an AL problem into an ACS problem, and selecting examples uniformly among the classes can provide results far better than what would be possible with AL alone. For instance, the learning curves presented in Figure 6.12 compare such uniform guided learning with AL and simple random selection. Providing the model with an essentially random but class-balanced training set far exceeds the generalization performance possible for an AL strategy or by random selection once the class skew becomes substantial. More intelligent ACS strategies may make this difference even more pronounced, and should be considered if the development effort associated with incorporating such strategies would be outweighed by the savings coming from reduced data acquisition costs.

### 6.8.2 Feature-Based Learning and Active Dual Supervision

While traditional supervised learning is by far the most prevalent classification paradigm encountered in the research literature, it is not the only approach for incorporating human knowledge into a predictive system. By leveraging, for instance, class associations with certain feature values, predictive systems can be trained that offer potentially excellent generalization performance without requiring the assignment of class labels to individual instances. Consider the example domain of predicting the sentiment of movie reviews. In this context, it is clear that the presence of words such as "amazing" and "thrilling" carries an association with the positive class, while terms such as "boring" and "disappointing" evoke negative sentiment [51]. Gathering this kind of annotation leverages an oracle's prior experience with the class polarity of certain feature values—in this case, the emotion that certain terms tend to evoke. The systematic selection of feature values for labeling by a machine learning system is referred to as *active feature-value labeling*,[11]. The general setting where class associations are actively sought for both feature values and particular examples is known as *ADSs*. The process of selection for AFL and ADS is shown in Figures 6.15 and 6.16, respectively.

Of course, incorporating the class polarities associated with certain feature values typically requires specialized models whose functional form has been designed to leverage feature-based background knowledge. While a survey of models for incorporating such feature- value/class polarities is beyond the scope of this chapter, an interested reader is advised to seek any number of related papers (cf. References 52–58). However, while sophisticated models of this type

[11]For brevity, this is often shortened as AFL, a moniker that is best suited for domains with binary features

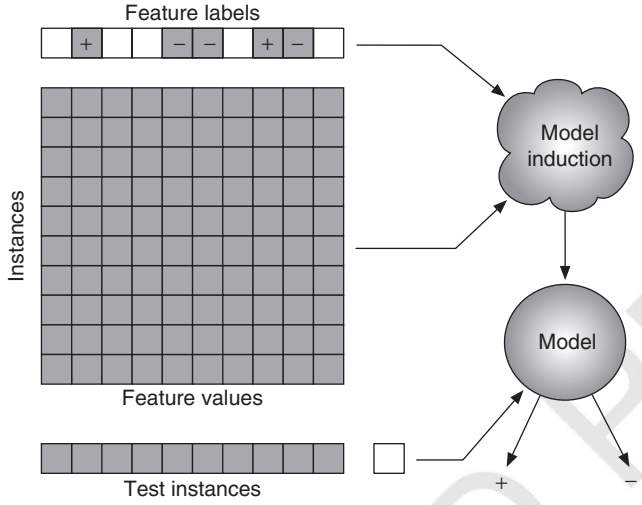


**Figure 6.15** Active feature-value labeling: selecting feature values for association with a certain class polarity.

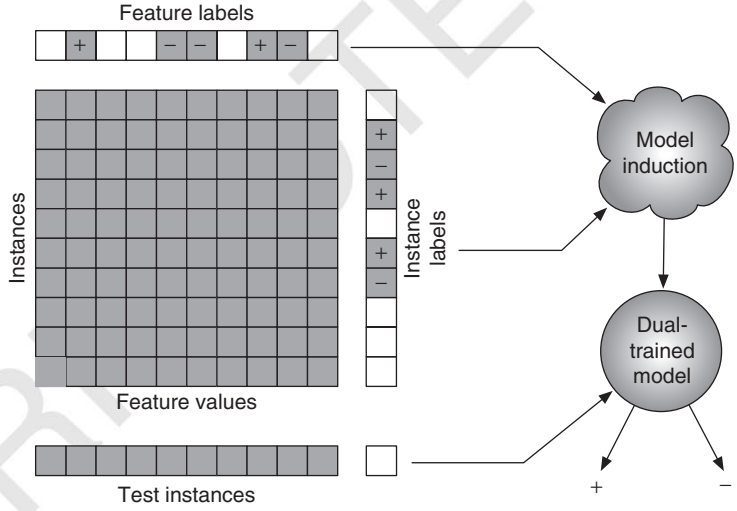


**Figure 6.16** Active dual supervision: simultaneous acquisition of label information for both feature values and instances.

have been devised, practical, yet very simple solutions are easy to imagine. Consider at the most basic extreme class assignment rules, where the presence of certain feature values (or combinations of feature values) connects an example with a particular class. A rudimentary "dual" model can also be constructed by simply averaging the class predictions of a traditional supervised machine learning model with a model based on feature-value/class relationships.

Given the option of incorporating prior knowledge associated with certain feature values into the predictive behavior of a model, the question then becomes: *which feature examples should be selected for labeling?* Initially, this may seem to be a deflection, replacing the search for useful information of one kind with another. However, there are several reasons why seeking feature values to "label" may be preferred to labeling examples. The most obvious reason for selecting feature values for labeling is that traditional is often "slow." It may take many labeled movie reviews to teach the model that "amazing" has a positive association, and not the other uninformative terms that just happen to occur in positive reviews by coincidence. In this way, the cold start problem faced in AL[12] may be less acute in AFL and ADS—while complex feature/class relationships may be difficult to achieve using AFL; reasonable generalization performance is often achievable with few requests to an oracle. Second, the labor costs associated with assigning a class polarity to a certain feature value is often quite low—it is easy for a human to associate the term *terrible* with negative movie reviews, while labeling one particular movie review as positive or negative requires reading the entire document. Note that of course not every term is polar; in the ongoing example of movie reviews, it is easy to imagine that a few terms have a natural association with the positive or negative class, while most terms on their own do not have such a polarity. However, this imbalance between polar and non-polar feature values is often far less acute than the imbalance between classes in many machine learning problems. A problem domain with a class imbalance of 1,000,000 : 1 may still have 1 in 10 features exhibiting some meaningful class association. Perhaps even more importantly, the ratio between positively and negatively linked feature values (for instance) may be far more balanced than the ratio between those classes in the wild. In fact, there is not necessarily a relationship between the base rate and the ratio of strongly identifiable positively and negatively associated feature values. While selecting useful feature values in practice is still often a challenging problem, experience has shown that it is often more informative to select random features for labeling than random examples. More intelligent selection heuristics can make this preference for AFL and ADS even stronger.

Again, giving a thorough survey of selection heuristics for performing AFL is beyond the scope of this chapter. However, we will provide the reader with a brief overview of the techniques typically employed for such tasks. As in many selective data acquisition tasks for machine learning, we see two common themes: uncertainty-based selection and expected-utility-based approaches. In the following, we will briefly present some more popular techniques for AFL delineated accordingly. We will then briefly discuss the techniques for ADS, selecting features and examples for labeling simultaneously.

***6.8.2.1 Uncertainty-Based AFL*** By far the most prevalent class of AL heuristics, uncertainty-based approaches do not have a direct analogue to the selection

[12]Recall Section 6.7

problem faced in ADS; feature-value/class associations factor quite differently into the formation and training of machine learning models than more traditional example labels. Nonetheless, approaches thematically similar to the ADS problem have been developed, where features are selected to labeling according to some notion of stability within the current model. As in traditional uncertainty selection in AL, the primary difference among these uncertainty-based ADS techniques is the manner by which feature uncertainty is estimated. In the simplest case, using a linear model, the coefficients on particular feature values may be interpreted as a measure of uncertainty, with lower coefficient magnitude corresponding to a greater degree of uncertainty [59]. The Naíve Bayes-like model used presented in Reference 52 presents a more appealing option—feature-value label uncertainty can be measured by looking at the magnitude of the log-odds of the feature-value likelihoods: $|\log p(f|+)/p(f|-)|$ for feature value $f$ and classes $+$ and $-$. Again, a smaller value corresponds to increased uncertainty.

A range of other techniques for uncertainty-based AFL exist. By the creation of one-term pseudo-documents, Godbole et al [60] coerce the notion of feature label uncertainty into a more traditional instance uncertainty framework for text classification tasks. By incorporating the label information on each feature value with unlabeled examples, Druck et al. [58] create a corresponding *generalized expectation* term that rates the model's predicted class distribution conditioned on the presence of the particular feature. This rating penalizes these predicted class distributions according to their KL-divergence from reference distributions constructed using labeled features. Similarly, Liang et al. [61] learn from labeled examples and actively selected constraints in the form of expectations with some associated noise from particular examples. Druck et al. [62] analyze several uncertainty-based selection techniques for gathering feature labels when training conditional random fields, finding that the total uncertainty (measured as the sum of the marginal entropies) tends to favor more frequent features. As a remedy, they propose an uncertainty scheme where the mean uncertainty is weighted by the log of the counts of the associated feature values.

Interestingly, even for reasonable uncertainty estimators, feature/class uncertainty may not be a desirable criterion for selection. Consider the discussion made previously regarding the preponderance of uninformative features. Clearly, in the case of document classification, terms such as "of" and "the" will seldom have any class polarity. At the same time, these terms are likely to have a high degree of uncertainty, leaving uncertainty-based approaches to perform poorly in practice. Preferring to select features based on *certainty*, that is, selecting those features with the least uncertainty, seems to work much better in practice [59, 63].

***6.8.2.2 Expected Utility-Based AFL*** As with traditional uncertainty (and certainty) sampling for AL, the query corresponding to the greatest level of uncertainty may not necessarily be the query offering the greatest level of information to the model. This is particularly true in noisy or complex environments. Instead of using a heuristic to estimate the information value of a particular feature label,

it is possible to estimate this quantity directly. Let $q$ enumerate over all possible feature values that may be queried for labels. We can estimate the expected utility of such a query by: $EU(q) = \sum_{k=1}^{K} P(q = c_k)\mathcal{U}(q = c_k)/\omega_q$, where $P(q = c_k)$ is the probability of the instance or feature queried being associated with class $c_k$, $\omega_q$ is the cost of query $q$, and $\mathcal{U}$ is some measure of the utility of $q$.[13] This results in the decision-theoretic optimal policy, which is to ask for feature labels which, once incorporated into the data, will result in the highest increase in classification performance in *expectation* [51, 63].

***6.8.2.3 Active Dual Supervision*** ADS is concerned with situations where it is possible to query an oracle for labels associated with both feature values and examples. Even though such a paradigm is concerned with the simultaneous acquisition of feature and example labels, the simplest approach is treating each acquisition problem separately and then mixing the selections somehow. Active interleaving performs a separate (un) certainty-based ordering on features and on examples, and chooses selections from the top of each ordering according to some predefined proportion. The different nature of feature value and example uncertainty values lead to incompatible quantities existing on different scales, preventing a single, unified ordering. However, expected utility can be used to compute a single unified metric, encapsulating the value of both types of data acquisition. As mentioned earlier, we are estimating the utility of a certain feature of example query $q$ as: $EU(q) = \sum_{k=1}^{K} P(q = c_k)\mathcal{U}(q = c_k)/\omega_q$. Using a single utility function for both features and examples and incorporating label acquisition costs, costs and benefits of the different types of acquisition can be optimized directly [51].

## 6.9 CONCLUSION

This chapter presents a broad perspective on the relationship between AL—the selective acquisition of labeled examples for training statistical models—and imbalanced data classification tasks where at least one of the classes in the training set is represented with much fewer instances than the other classes. Our comprehensive analysis of this relationship leads to the identification of two common associations, namely $(i)$ is the ability of AL to deal with the data imbalance problem that, when manifested in a training set, typically degrades the generalization performance of an induced model, and $(ii)$ is the impact class imbalance may have on the abilities of an otherwise reasonable AL scheme to select informative examples, a phenomenon that is particularly acute as the imbalance tends toward the extreme.

Mitigating the impact of class imbalance on the generalization performance of a predictive model, in Sections 6.3 and 6.4, we present AL as an alternative to more conventional resampling strategies. An AL strategy may select a dataset

[13]For instance, cross-validated accuracy or log-gain may be used.

that is both balanced and extremely informative in terms of model training. Early stopping criteria for halting the selection process of AL can further improve the generalization of induced models. That is, a model trained on a small but informative subsample often offers performance far exceeding what can be achieved by training on a large dataset drawn from the natural, skewed base rate. The abilities of AL to provide small, balanced training from large, but imbalanced problems are enhanced further by VIRTUAL, introduced in Section 6.4. Here, artificially generated instances supplement the pool of examples available to the active selection mechanism.

Additionally, it is noted throughout that abilities of an AL system tend to degrade as the imbalance of the underlying distribution increases. While at more moderate imbalances, the quality of the resulting training set may still be sufficient to provide usable statistical models, but at more substantial class imbalances, it may be difficult for a system based on AL to produce accurate models. Throughout Sections 6,2–6.4, we illustrate a variety of AL techniques specially adapted for imbalanced settings, techniques that may be considered by practitioners facing difficult problems. In Sections 6.5 and 6.6, we note that as a problem's class imbalance tends toward the extreme, the selective abilities of an AL heuristic may fail completely. We present several alternative approaches for data acquisition in Section 6.8, mechanisms that may alleviate the difficulties AL faces in problematic domains. Among these alternatives are guided learning and ACS in Section 6.8.1, and using associations between specific feature values and certain classes in Section 6.8.2.

Class imbalance presents a challenge to statistical models and machine learning systems in general. Because the abilities of these models are so tightly coupled with the data used for training, it is crucial to consider the selection process that generates this data. This chapter discusses specifically this problem. It is clear that when building models for challenging imbalanced domains, AL is an aspect of the approach that should not be ignored.

## REFERENCES

1. B. Settles, "Active learning literature survey," Computer Sciences Technical Report 1648, University of Wisconsin-Madison, 2009.
2. V. V. Federov, *Theory of Optimal Experiments*. New York: Academic Press, 1972.
3. D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15, pp. 201–221, 1994.
4. D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA), pp. 3–12, Springer-Verlag Inc., 1994.
5. Y. Freund, "Sifting informative examples from a random source," in *Working Notes of the Workshop on Relevance, AAAI Fall Symposium Series*, pp. 85–89, 1994.

6. I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proceedings of the Twelfth International Conference on Machine Learning* (Tahoe City, CA, USA), pp. 150–157, Morgan Kaufmann, 1995.
7. Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Information, prediction, and query by committee," in *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, pp. 483–490, Morgan Kaufmann, 1993.
8. S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2002.
9. N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *ICML*, Morgan Kaufmann, 2001.
10. R. Moskovitch, N. Nissim, D. Stopel, C. Feher, R. Englert, and Y. Elovici, "Improving the detection of unknown computer worms activity using active learning," in *Proceedings of the 30th Annual German Conference on Advances in Artificial Intelligence*, KI '07, pp. 489–493, Springer-Verlag, 2007.
11. Y. Guo and R. Greiner, "Optimistic active learning using mutual information," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI'07* (Hyderabad, India), pp. 823–829, 2007.
12. B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08 (Jeju Island, Korea), pp. 1070–1079, Association for Computational Linguistics, 2008.
13. J. Zhu, H. Wang, T. Yao, and B. K. Tsou, "Active learning with sampling by uncertainty and density for word sense disambiguation and text classification," in *COLING '08*, Association for Computational Linguistics, 2008.
14. H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *ICML*, ACM, 2004.
15. A. K. Mccallum and K. Nigam, "Employing EM in pool-based active learning for text classification," in *ICML*, 1998.
16. P. Donmez, J. G. Carbonell, and P. N. Bennett, "Dual strategy active learning," in *ECML '07*, Springer, 2007.
17. P. Donmez and J. Carbonell, "Paired sampling in density-sensitive active learning," in *Proceedings of 10th International Symposium on Artificial Intelligence and Mathematics* (Ft. Lauderdale, FL, USA), 2008.
18. K. Tomanek and U. Hahn, "Reducing class imbalance during active learning for named entity annotation," in *K-CAP '09*, 105–112, ACM, 2009.
19. S. Ertekin, J. Huang, L. Bottou, and C. L. Giles, "Learning on the border: Active learning in imbalanced data classification," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM)* (Lisbon, Portugal), pp. 127–136, ACM, 2007.
20. M. Bloodgood and K. V. Shanker, "Taking into account the differences between actively and passively acquired data: The case of active learning with support vector machines for imbalanced datasets," in *NAACL '09*, Association for Computational Linguistics, 2009.
21. S. Ertekin, *Learning in Extreme Conditions: Online and Active Learning with Massive, Imbalanced and Noisy Data*. PhD thesis, The Pennsylvania State University, 2009.

22. J. Zhu and E. Hovy, "Active learning for word sense disambiguation with methods for addressing the class imbalance problem," in *EMNLP-CoNLL 2007* (Prague, Czech Republic), 2007.
23. G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," in *Proceedings of the 17th International Conference on Machine Learning (ICML)* (Stanford, CA, USA), pp. 839–846, Morgan Kaufmann, 2000.
24. A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *Journal of Machine Learning Research*, vol. 6, pp. 1579–1619, 2005.
25. J. Huang, S. Ertekin, and C. L. Giles, "Efficient name disambiguation for large scale datasets," in *Proceedings of European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)* (Berlin, Germany), vol. 4213/2006, pp. 536–544, 2006.
26. V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.
27. A. J. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," in *Proceedings of 17th International Conference on Machine Learning (ICML)* (Stanford, CA), pp. 911–918, Morgan Kaufmann, 2000.
28. M. Kubat and S. Matwin, "Addressing the curse of imbalanced training datasets: One sided selection," in *Proceedings of 14th International Conference on Machine Learning (ICML)*, vol. 30, no. 2–3, pp. 195–215, 1997.
29. N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429–449, 2002.
30. N. V. Chawla, K. W. Bowyer., L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
31. N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proceedings of 2000 International Conference on Artificial Intelligence (IC-AI'2000)*, vol. 1, pp. 111–117, 2000.
32. C. X. Ling and C. Li, "Data mining for direct marketing: Problems and solutions," in *Knowledge Discovery and Data Mining*, pp. 73–79, 1998.
33. J. Attenberg and F. Provost, "Why label when you can search? Strategies for applying human resources to build classification models under extreme class imbalance," in *KDD*, pp. 423–432, ACM, 2010.
34. C. Perlich and G. Swirszcz, "On cross-validation and stacking: Building seemingly predictive models on random data," *SIGKDD Explorations*, vol. 12, no. 2, p. 11–15, 2010.
35. J. He and J. G. Carbonell, "Nearest-neighbor-based active learning for rare category detection," in *NIPS* (Vancouver, Canada), pp. 633–640, MIT Press, 2007.
36. Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang, "Representative sampling for text classification using support vector machines," in *ECIR*, pp. 393–407, Springer-Verlag, 2003.
37. Weiss, G. M., "The impact of small disjuncts on classifier learning," *Annals of Information Systems*, vol. 8, pp. 193–226, 2010.
38. R. Lomasky, C. Brodley, M. Aernecke, D. Walt, and M. Friedl, "Active class selection," *Machine Learning: ECML*, vol. 4701, pp. 640–647, 2007.

39. R. Lomasky, *Active Acquisition of Informative Training Data*. PhD thesis, Tufts University, 2010.
40. R. Lomasky, C. E. Brodley, S. Bencic, M. Aernecke, and D. Walt, "Guiding class selection for an artificial nose," in *NIPS Workshop on Testing of Deployable Learning and Decision Systems*, 2006.
41. J. Attenberg and F. Provost, "Inactive learning? Difficulties employing active learning in practice," *SIGKDD Explorations*, vol. 12, no. 2, pp. 36–41, 2010.
42. J. Attenberg, P. Ipeirotis, and F. Provost, "Beat the machine: Challenging workers to find the unknown unknowns," in *Proceedings of the 3rd Human Computation Workshop (HCOMP 2011)*, 2011.
43. G. M. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *Journal of Artificial Intelligence Research*, vol. 19, pp. 315–354, 2003.
44. P. Melville, F. J. Provost, and R. J. Mooney, "An expected utility approach to active feature-value acquisition," in *International Conference on Data Mining* (Houston, TX, USA), pp. 745–748, 2005.
45. M. Saar-tsechansky and F. Provost, "Active learning for class probability estimation and ranking," in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pp. 911–920, Morgan Kaufmann, 2001.
46. M. Saar-Tsechansky and F. Provost, "Active sampling for class probability estimation and ranking," *Machine Learning*, vol. 54, no. 2, pp. 153–178, 2004.
47. G. Weiss and F. Provost, "The effect of class distribution on classifier learning," 2001.
48. SAS Institute Inc., *Getting Started with SAS Enterprise Miner*. Cary, NC: SAS Institute Inc., 2001.
49. F. Provost, "Machine learning from imbalanced data sets 101," in *Proceedings of the AAAI Workshop on Imbalanced Data Sets*, 2000.
50. C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pp. 973–978, Morgan Kaufmann, 2001.
51. J. Attenberg, P. Melville, and F. J. Provost, "A unified approach to active dual supervision for labeling features and examples," in *European Conference on Machine Learning*, pp. 40–55, Springer-Verlag, 2010.
52. P. Melville, W. Gryc, and R. Lawrence, "Sentiment analysis of blogs by combining lexical knowledge with text classification," in *KDD*, pp. 1275–1284 ACM, 2009.
53. R. E. Schapire, M. Rochery, M. G. Rahim, and N. Gupta, "Incorporating prior knowledge into boosting," in *International Conference on Machine Learning (ICML)*, Morgan Kaufmann, 2002.
54. X. Wu and R. Srihari, "Incorporating prior knowledge with weighted margin support vector machines," in *Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 326–333, ACM, 2004.
55. B. Liu, X. Li, W. S. Lee, and P. Yu, "Text classification by labeling words," in *AAAI*, pp. 425–430, AAAI Press 2004.
56. A. Dayanik, D. Lewis, D. Madigan, V. Menkov, and A. Genkin, "Constructing informative prior distributions from domain knowledge in text classification," in *SIGIR*, pp. 493–500, ACM, 2006.

57. G. Kunapuli, K. P. Bennett, A. Shabbeer, R. Maclin, and J. Shavlik, "Online knowledge-based support vector machines," in *ECML/PKDD (2)* (J. L. Balczar, F. Bonchi, A. Gionis, and M. Sebag, eds.), vol. 6322, 2010.

Q11

58. G. Druck, G. Mann, and A. McCallum, "Learning from labeled features using generalized expectation criteria," in *Special Interest Group in Information Retrieval (SIGIR)*, 595–602, ACM, 2008.
59. V. Sindhwani, P. Melville, and R. Lawrence, "Uncertainty sampling and transductive experimental design for active dual supervision," in *Proceedings of the 26th International Conference on Machine Learning (ICML-09)*, pp. 953–960, ACM, 2009.
60. S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti, "Document classification through interactive supervision of document and term labels," in *Practice of Knowledge Discovery in Databases (PKDD)*, pp. 185–196, Springer-Verlag, 2004.
61. P. Liang, M. I. Jordan, and D. Klein, "Learning from measurements in exponential families," in *ICML*, pp. 641–648, ACM, 2009.
62. G. Druck, B. Settles, and A. McCallum, "Active learning by labeling features," in *Conference on Empirical Methods in Natural Language Processing (EMNLP '09)* (Singapore), pp. 81–90, Association for Computational Linguistics, 2009.
63. P. Melville and V. Sindhwani, "Active dual supervision: Reducing the cost of annotating examples and features," in *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, Association of Computational Linguistics, 2009.

**Queries in Chapter 6**

Q1. The sentence 'In settings...selected' has been rephrased for clarity. Please confirm if this is fine.

Q2. AUC has been changed to AUROC in the sentence "The AUC is a numerical ... ". Please confirm if this is fine.

Q3. Please provide the signifcance for the values set in bold in Table 6.3.

Q4. Please provide the conference venue for References 7, 9, 10, 18, 33, 36, 45, 50, 52, 54, 55, 56, 58, 60, 61.

Q5. Please provide the publisher name for References 11, 25 and 44.

Q6. Please provide the conference venue and page range for References 13, 14, 16, 51, 53, 63.

Q7. Please provide the conference venue, publisher name, and page range for References 15, 32, 42, 40, 49.

Q8. Please provide the publisher name and page range for Reference 17.

Q9. Please provide conference venue and publisher name for References 28 and 31.

Q10. Please provide the report number, place of publication and publisher's name for reference 47.

Q11. Please provide the place of publication, page range and publisher's name for Reference 57.