Rare category exploitation

Harel Lustiger

A thesis submitted towards the
degree of Master of Science in Statistics

in

The Raymond and Beverly Sackler Faculty of Exact Sciences
The Department of Statistics and Operations Research
Tel-Aviv University

Supervisor: Prof. David Steinberg

November 2016

Abstract

There are situations in which we use predictive modeling for sales purposes in a sequential fashion. Here, we get a customer data set with a finite number of instances, and only a small portion of them will respond to a given offer. The number of "future" responders is unknown and the goal is to find them while avoiding approaching the non-responders. By making sequential decisions of which customers to approach at each epoch, we face a dilemma between (i) exploitation, targeting those with the highest probability of responding or (ii) exploration, targeting those for which we are the least certain about their choice (under the assumption that they can adequately explain those in other parts of the input space of the unlabeled data set). In this paper we propose policies addressing this dilemma while exploiting prior knowledge about the settings constraints. Our main contribution is a novel algorithmic modeling approach for detecting the responders, i.e. the minority class. We find that our approach for arbitrating between exploration and exploitation often yields decent results and is robust across different data sets for finding more responders for a given number of epochs.

Contents

Chapter 1

Introduction

Predictive modeling for sales purposes (hereafter, predictive modeling) and the offered product or service are bound together by the product life cycle. By projecting the notion of product life cycle on predictive modeling, it becomes clear that the life cycle of the prediction model is dependent on the offered product's life cycle properties and the stage it is in.

The notion of product life cycle dates back to 1965 when the late T. Levitt [1] described a sales pattern of **successful** products. The pattern is described by a curve of sales volume as a function of time. In essence there are four phases in a successful product life: (i) market development (ii) market growth (iii) market maturity and (iv) market decline. To understand the practical application of this paper while staying focused on our contribution, we give a concrete real-world case of product life cycle in the banking industry. Obviously, most households own and use bank accounts, evidence that this financial service is in the market maturity stage. This situation is expected to persist with a steady growth per capita, but more important, the stored data about the individuals can be used to extend the maturity stage of the bank in terms of increased growth and increased stability. Generally speaking, there are four different tactics to expand sales:

1. Promoting more frequent usage of the product among current users.
2. Developing more varied usage of the product among current users.
3. Creating new users for the product by expanding the market.
4. Finding new uses for the basic material.

In the banking example, applying tactics on current users (first and second tactics) leads to having a portfolio of products (such as holiday loans and long-term deposits), where each one of them has its own life cycle. Some of the products are launched seasonally while others experience the decline stage and are abandoned. For the latter, the prediction model becomes irrelevant and gets pushed out of service. This property of finite time horizon of the prediction model should be taken into account in the research design for maximizing the objective function – the total number of responsive customers.

Nowadays, predictive modeling is widely used to enhance the efficiency of marketing campaigns for finding the responsive customers for the offered product or service. The need for predictive modeling to promote products among customers has primarily arisen from the acute problem of low response rate [2]. Therefore, data associated with marketing campaigns tends to be skewed towards the non-responsive

customers. In a managerial perspective, one might be concerned with the negative impact of marketing promotion on the majority of customers (i.e. the non-responsive customers) [3]. It is, therefore, important to target the responsive customers and avoid the others for both managerial and finite time horizon reasons.

Current methodologies for fitting response prediction models ignore two stages in the model's life cycle, its inception and its death. The acquisition of the initial labeled set in an unbalanced data setting can further diverge into two research fields, rare category detection [4] and rare category exploration [5]. The former aims at identifying at least one instance from the rare category in an unlabeled set. The latter deals with the discovery of the remaining instances of the desired category using the initial seed. When introduced to the finite time horizon of the model's life cycle, an intuitive idea is how to find the rest of the minority cases in the unlabeled set as quickly as possible within a time frame. In this paper we refer to this issue as rare category exploitation.

Rare category exploitation has various practical applications, especially when it can be carried out in a sequential fashion. One common method to conduct a campaign for promoting a product is by direct marketing. First, marketers try to sell the product to customers or inform them about the new product via the phone. Then, after a predefined epoch the clients' attributes with their corresponding results, successful or unsuccessful[1] are accumulated. By applying predictive modeling to the augmented data, the model can be refitted and used to assign probabilities for responding to the offer to the rest of the customers in the database. These probabilities enable the model to prioritize and select the next batch of customers to contact according to a predefined policy. Although this setting has the nature of a sequential experiment, it's common to refer to it as one-time decision making [6],[7]. One might think that introducing strategy to the decision making procedure can lead to better performance. However, the research is rather scarce in terms of sequential customer targeting, especially with finite time horizon. This study of rare category exploitation is motivated by the lack of information in this field.

Among the existing research, reinforcement learning [8],[9] and active learning [10] (both fields are discussed below) can also be used to perform sequential experiments, albeit, these two tasks are different from rare category exploitation from the following aspects. (1) First, under finite time horizon the model is discarded at the end of the experiment. As a result, we are interested in the training performance during the training phase rather than optimizing some test set measures. That is in contrast to active learning as it tries to query instances which it thinks would decrease the generalization error for an unseen data set. (2) Unlabeled instances can be queried (sampled) once, while reinforcement learning is based on the premise of sampling with replacement.

In this paper, we propose an approach known as QTMC (**Q**uickly **T**arget **M**inority **C**ases) methodology to rare category exploitation. By our approach, prior knowledge about the campaign, mainly its constraints such as duration and capacity of the system (e.g. the number of weekly calls in a call-center), are translated into a sequential experimental design with finite horizon. In the experiment continuum, between its inception and end, we query unlabeled instances in order to maximize the total number of minority cases within the time frame.

Our contributions can be summarized as follows. (1) To the best of our knowledge, it is the first research to identify and solve the problem of *rare category exploitation*. (2) We propose an efficient approach known as QTMC for rare class exploitation which outperforms the related approaches (which are not specifically designed for rare category exploitation) in terms of robustness at finding the minority

[1] We discard other outcomes, such as customer did not answer, or customer number led to fax instead phone.

cases in an unlabeled set across different data sets and classification models.

We limit our discussion to an unknown fixed number of responders in a database, whereas a known fixed number can facilitate solution of the problem but it is ill-suited for practical cases. The remaining chapters are organized as follows. We review the related work in chapter 2 and present our problem statement, evaluation metrics, proposed methodology and data sets in chapter 3. In chapter 4 we present a few strategies developed by our methodology which are then empirically evaluated and analyzed in chapter 5.

Chapter 2

Related Work

The related work to rare category exploitation in direct marketing can be classified into four categories, namely (1) data mining techniques for direct marketing (2) imbalanced learning (3) active learning for unbalanced problems, and (4) reinforcement learning. We focus on the problem of classifying instances into one of two-classes and exclude work on unsupervised learning and/or work which is based on similarity or distance performance measures as their guiding rule.

2.1 Data mining techniques for direct marketing

A useful synopsis for direct marketing is given by [11]

> ... an interactive system of marketing that uses one or more advertising media to effect a measurable response and/or transaction at any location, with this activity stored on database.

An early attempt to apply data mining techniques for direct marketing can be found in [2]. The proposed methodology defines it as a cyclic process, where several iterations are allowed until satisfactory results are obtained. Their methodology includes five steps:

1. **Getting the data**; the database is separated into two sets: customer with their response to a product offer (known as the *labeled set*[1], and prospective customers whom have not been offered the product yet (known as the *unlabeled set*).

2. **Data mining on the data sets**:

   a) Perform data preprocessing (enriching the data, cleaning the data, etc.).

   b) Splitting the labeled set into *training set* and *test set*.

   c) Apply learning algorithms (or classifiers) to the training set.

3. **Evaluation**; evaluate the patterns found on the test set. Iterate to previous steps until the results are satisfactory.

4. **Prediction**; use the patterns found to predict likely buyers.

[1] Notice that the labeled set can be empty or have insufficient information (such as having no responsive customers) for performing further operations. This issue is discussed below under subsection 2.2)

5. **Deployment**; promote to the likely buyers.

With slight modifications and two additional preliminary steps[2] we recognize this methodology as the "Cross Industry Standard Process for Data Mining" (CRISP-DM) [12], the leading standard these days for data mining. Nevertheless, different from our proposed QTMC methodology, these approaches are not specifically designed for rare category exploitation. Considering data mining as a one-time decision making process and ignoring the finite time horizon of the product offer, have two profound flaws. First, it forces practitioners to have a test set at hand which represents the unlabeled set distribution. This allocation of instances to an isolated set in the face of a rare class is at the expense of the information stored within the training set. One can continue to query customers at random until the sample size is sufficient for that allocation, however such strategy defeats the purpose of direct marketing in the first place. Second, instead of maximizing the number of buyers directly, the practitioner uses a surrogate criterion for model evaluation purposes. The experiments in this paper show that data-driven policies developed with QTMC methodology are equal or worse to random selection of unlabeled instances when evaluated on a test set using common surrogate criteria. Yet, when changing the outlook on the evaluation process into the total number of responsive customers, the former policies significantly outperform random selection of unlabeled instances.

Focusing on the application we have in mind - bank telemarketing – leads to previous works [2],[7],[13] all of which describe in detail the properties and challenges of dealing with data sets for this application. A recent article describes the content of a direct marketing database [14]; geographic and demographic information about the customers with additional information about the response from customers for a particular campaign program (such as personal loans, mortgage loans, deposit plans, etc.) through a particular campaign channel (such as "call", "email", "webpage" etc.). According to the same study, this type of data presents three challenges:

1. Unbalanced data; as the result of a low response rate.

2. Sparsity in the relevant data; typically, a customer is involved in only a subset of the programs through a subset of the available marketing channels. With dozens or hundreds of campaign channel-program combinations, we can expect most of the interactions between them and the customers to be blank.

3. High dimensionality; appears due to the rich information about the customers coupled with the variety of campaign channels and programs.

The problems posed by sparsity and high dimensionality are heavily researched in the statistical literature and go far beyond the scope of this paper. The remaining challenge of unbalanced data is our main concern, in particular under sequential settings.

2.2 Unbalanced data

As we mentioned, our data has a rare class which we are eager to find. One aspect of rarity is whether it is a relative or an absolute property [15]. The former is equivalent to the prevalence of a population where the class of primary interest is sharply outnumbered in that population, i.e. unbalanced data. The latter

[2]Business Understanding and Data Understanding.

Table 2.1: Rareness type/level scenarios.

Rareness Type / Level	**Skewed**	**Highly skewed**
Relative	Rare Category Exploration [5] / Rare Category Exploitation	Guided learning [14]
Absolute	Rare Category Detection [4]	Rare Category Detection [4] / Guided learning [14]

describes the low quantity of information in the labeled data for performing specific operations, which is fundamentally a small sample problem. A second aspect of rarity is the prevalence severity: skewed or highly skewed. Although there is no exact definition in the literature for what the majority to minority instances ratio (hereafter the imbalance ratio) is considered to be highly skewed, the same data set with an imbalance ratio of 10 evokes different problems from that with an imbalance ratio of 10^3. A different way to think about this issue is at the application level; for example, classifying web pages with unacceptable content (such as hate speech, malware, etc.) across the web can have an imbalance ratio of 10^4 to 10^7 or higher [14]. We limit our discussion to imbalance ratios ranging from 5 to 50, in which is typical of the applications we have in mind. With these two aspects, we identified and summarized in Table 2.1 the scholarly literature that is suitable for each rareness type-level region.

Importantly, rare category exploitation (top left cell) rests on the assumption that the initial training set is going to produce a useful initial probabilistic model for discriminating between classes. However, there is no assumption about the randomness or the representativeness of the training set with comparison to the real population. This is important since QTMC is a sequential methodology for querying unlabeled instances based on information from both the labeled set and the unlabeled set. Even if at inception the two are representative, as we progress along the sequence, the labeled set becomes less representative of the unlabeled set for two reasons: First, if successful, we expect the imbalance ratio of the labeled set to decrease (we detect more minorities) while the imbalance ratio of the unlabeled set increases (we deplete minority cases from the unlabeled population). Second, the querying method is not based on random sampling, thus we introduce bias to the sample by design.

In a situation when the initial training set is empty, or doesn't have instances from the rare class, one can apply *rare category detection* [4] methods to detect the first instance and proceed with *rare category exploration* [5] methods (which are specifically designed to take advantage of the rare class characteristics). However, these methods are mainly based on similarity and therefore out of the scope for this paper. In situations when one uses a human oracle for labeling instances, Attenberg and Provost [14] suggested an interesting technique called "guided learning" for utilizing human resources for model development. If possible, changing the oracle's freedom from labeling the queried instances to searching explicitly for rare cases could have profound improvement over techniques that don't do that. Moreover, incorporating this technique in the model development may be used in a sequential setting without any assumption on how the instances are sampled. In the bank direct marketing example, the desk clerks can provide that kind of information to begin with. Then, we can fit a model with this data and query the unlabeled data. This strategy shifts workloads (and costs) from the bank's clerks to call-center agents.

2.3 Active learning for unbalanced problems

In their basic form, most classifiers do not behave well on unbalanced data sets. Instead, most classifiers have predictive preference for the class with the greater proportion of examples. Consider a realistic case when 8% of the customers within a database will respond to a given offer. A model that predicts all samples are non-responsive, would reach 92% accuracy in overall prediction. Ostensibly this is a high value but such a model would be useless for identifying instances from the rare class.

This difference in class preferences between practical needs and a model's tendencies causes a serious problem. The proposed solutions in the literature include (1) oversampling the minority class or undersampling the majority class [16], [17], and (2) cost-sensitive techniques, such as assigning non-uniform misclassification costs during training in order to give additional consideration to the class of interest [18].

In their inspiring paper [19], Ertekin et al. present another method to deal with unbalanced data, specifically in situations where the learning algorithm is allowed to choose the data from which it learns. This key hypothesis is dealt in a subfield of machine learning which is known as "active learning" (see [10] for a comprehensive survey of this subject). To compare active learning to traditional (non-sequential) techniques[3] in dealing with unbalanced data sets, the databases were divided into three parts: *test set, labeled set and unlabeled set*. Then, instead of using all the available data to form the final prediction model, the models were continuously modified as unlabeled instances were appended one by one into the labeled set. Finally, all models were evaluated on the (same) test set utilizing the *precision-recall break-even point metric* (discussed in Appendix A). The major difference between active learning and sequential traditional techniques then, is by how the sequential property is utilized. The former query unlabeled instances which appear to have potential to reduce the generalization error of the model and then fit a support vector machine (SVM) model to the augmented labeled set. The latter query unlabeled instances at random, preprocess the data (with accordance to the underlying technique) and fit an SVM model to the augmented labeled set. Moreover, to show active learning can reduce the number of labeled examples required to build a decent model, the researchers used a stopping criterion to halt the active learning process at some point in the experiment (see [20] for a thorough discussion of that field). The results show that processing only a portion of the available data achieved similar or even higher generalization performance compared to different techniques that were built on all available data.

In the context of rare category exploitation, this behavior is desired and indicates that smaller sample sizes of the labeled set may cause no model performance degradation. Since active learning makes no use of a test set to query the instances for the next batch, we can encapsulate this method in our proposed QTMC methodology. Moreover, our experiments show that QTMC can add robustness to active learning when its underlying assumptions are not valid (such as following the instruction to sample in regions of high uncertain, while the prediction model is highly certain about its prediction). By contrast, the framework discussed in the paper above evaluates the different models on a test set, and therefore is not suitable for our needs. While we explicitly try to maximize the total of minority cases within the labeled set, they try to maximize an external criterion evaluated on a test set.

[3] These preprocessing techniques included undersampling (US), SMOTE (syntactic oversampling technique) [17], different costs (DC) and a benchmark of a simple model (no preprocessing).

2.4 Reinforcement learning

Reinforcement learning aims at learning input-output relationships behind the data when no explicit supervision (output data) is provided. The common analogy for this type of problem is the *multi-armed bandit* (bandit for short) [21]. In this scenario a gambler has access to a variety of slot machines that yield different rewards. Then, in a sequential fashion and with accordance to a predefined policy, the gambler chooses and pulls one slot machine handle. A non-negative reward is drawn from the reward distribution for that machine and the gambler becomes aware of the ramification of her choice. At that time the gambler faces again the dilemma of which handle to pull next – should it be the handle with highest expected payoff (namely *exploitation*) or should she try a different one (namely *exploration*)? The ultimate goal of the policy is to guide the gambler's choices for maximizing the cumulative reward constrained to the number of rounds that remain to be played.

Thus, this process can be described using action-reward tuples, where each known action has its own unknown reward distribution. Also, within this context it can be formulated as a Markov decision process (MDP[4]), such that the gambler is a descriptor of an evolving system with a finite number of states, and the gambler transition to the next state depends on her current action and reward.

Our approach to rare category exploitation is inspired by the bandit problem since detecting the minority cases (i.e. the non-negative rewards) in finite time horizon (i.e. number of rounds) is obtained without the use of a test set (i.e. no explicit supervision). Thus, reinforcement learning and its notions provides us with a framework for our QTMC methodology.

In contrast to the set-up of the bandit problem where the same bandit arm can be played repeatedly, under the basic version of direct marketing, a customer cannot be contacted more than once. Thus, learning the reward distribution is not helpful and solutions based on a MDP are not relevant to us.

[4]The general study of MDPs goes beyond the scope of this paper.

Chapter 3

Problem Formulation

3.1 Problem Statement

In this section, we present the problem statement of rare category exploitation. Table 3.1 lists the notation that will be used henceforth.

Consider a large data set $\mathcal{D} \in \mathbb{R}^{n\times(p+1)}$ where $p << n$. The $p+1$ column is the vector $y \in \{-1, +1\}$ which represents the response of each customer to a given offer. That is, we have a finite number n_{+1} of cases where $y = +1$ and correspondingly $n_{-1} = n - n_{+1}$ is the number of cases where $y = -1$. The data *imbalance ratio* is defined as $r = n_{-1}/n_{+1}$ with a typical value of $r > 7$ for the applications we have in mind. For such an unbalanced data set, the natural classes that arise are "Positive" (minority class) or "Negative" (majority class) for a customer to respond or to reject a specific product offer, respectively.

Given: By some means our data set $\mathcal{D}$ is split into two subsets:

1. a small set of labeled samples $\mathcal{L} = \{x_i, y_i\}_{i=1}^{l}$ and
2. a large pool of unlabeled samples $\mathcal{U} = \{x_j\}_{j=l+1}^{n}$.

Find: The minority cases in $\mathcal{U}$ constrained to φ and B where:

1. φ is the number of phases (epochs) in the experiment.
2. B is the batch size, that is, the number of available customers we can contact an each epoch.

Intuitively, we think as if somebody hid the true response for the bulk of the data set, and our objective is to discover as many positive cases as possible while avoiding targeting the majority class. For that end, we assume each unlabeled instance follows a probability $I_j \sim Ber(\rho_j)$ to be a minority case, where $\rho_j = Pr(Y_j = +1|x_j)$.

Note that:

1. φ, B can be derived from real-world constraints based on economic or other external factors such as the call-center capacity. Even without constraints, as in the case of mailing, one might want to minimize the miss cases in which a customer turns down an offer.
2. The information about the future is received with delay. Therefore, to benefit from the sequential processes design, where the prediction models are retrained with the accumulated data, one is obliged to work in batches.

Table 3.1: Symbol and description

Notation	Description
$\mathcal{D}$	The customer database
x_i	The i^{th} data example in $\mathcal{D}$
n	The number of all data examples in $\mathcal{D}$
p	The dimensionality of the feature space
n_{+1}, n_{-1}	The number of minority cases and majority cases in $\mathcal{D}$
r	The data imbalance ratio in $\mathcal{D}$
$\mathcal{L}$	The set of labeled samples
$\mathcal{U}$	The pool of unlabeled samples
φ	The number of phases (epochs) in the experiments
B	The batch size
ρ_j	The probability the j^{th} subject is a minority case
I_j	The indicator for whether the j^{th} subject is a minority case

3. Without loss of generalization, we set B to be constant at each epoch so the maximum number of minorities we can attain has an (optimistic) upper boundary of φB.

3.2 Evaluation in Unbalanced Domains and Finite Horizon

In this section we present the performance measure that concerns the property of the policy we wish to measure. Considering the context of the evaluation, we argue that none of the traditional performance evaluation metrics which are based on the confusion matrix (such as accuracy, precision, recall and their combinations) or scoring metrics encompasses the property of the model we wish to measure (see Appendix A for further discussion).

3.2.1 Temporal-Minority space

In the case of rare category exploitation, an adequate performance measure should be accountable for four properties of the scenario: (1) binary classification task (2) unbalanced data set (3) sequential experiment and (4) finite horizon.

We introduce the temporal-minority space (TM) illustrated in Figure 1 both with absolute values (left) and scaled values (right), where one plots the number of observations in the labeled set on the x-axis and the number of minorities within the labeled set on the y-axis. Then by applying a policy in a sequential fashion, at each epoch we choose and label B instances from the unlabeled set and append them to the labeled set. In this manner we obtain a monotone **non decreasing curve** that describes the number of minority cases as a function of the number of labeled instances. Obviously, the range of the y-axis is between 0 and the total number of minorities within the data set (as a whole). The left dashed line represents the maximum feasible number of minority cases under the problem constraints. For example, if at each epoch we choose and label B instances, then at epoch φ the maximum number of minorities we could have is $\varphi \cdot B$ (given we haven't discovered all of the minority cases within the data set). Typically, at inception we have some labeled instances, some of which are minorities. This fact is expressed by an offset on the x and y axes from the origin, respectively.

Figure 3.1 demonstrates the broad scope of a sequential experiment that starts with an initial labeled set and ends when all cases are labeled. In the continuum of the experiment, rare category exploitation

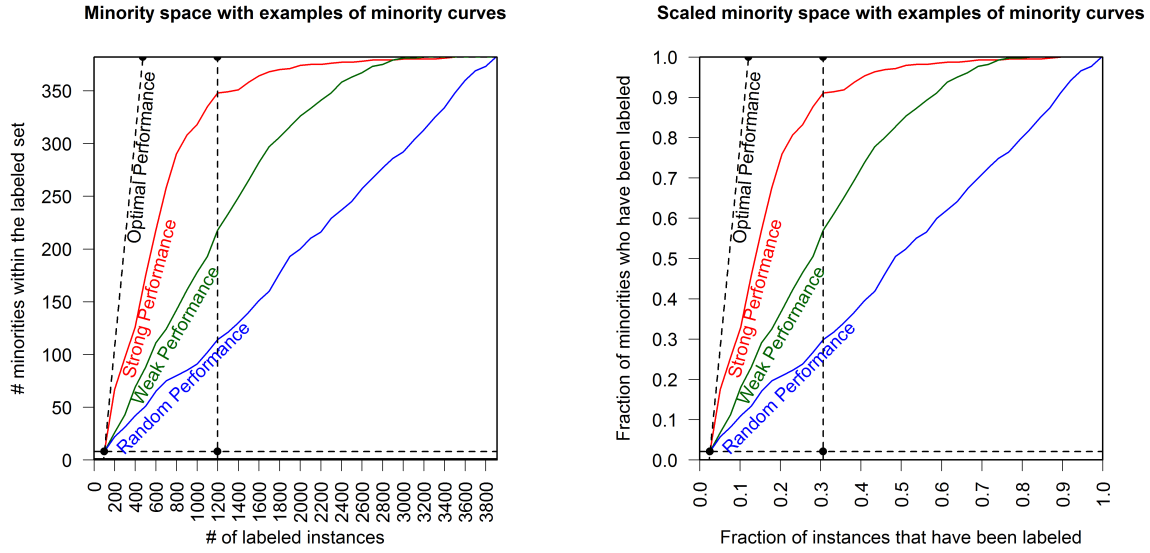

Figure 3.1: Temporal-minority space with examples of minority curves both in absolute (left) and scaled (right) values. Each curve represents the performance of a different policy on a data set.

is expressed in a confined subspace. Its right boundary, here depicted by a dashed line at $x = 1200$ or $x \approx 0.3$, is driven by the problem constraints and expresses the finite horizon of the sequential experiment. The confined right trapezoid, shown in Figure 1 with its vertices, is then the subspace where we evaluate our policies.

A closer look at the intersections between the right boundary and the different curves in Figure 3.1 illustrates a setting in which "random performance" detects 30% of all minority cases, whereas its counterpart "strong performance" reaches 90%, an evidence to its superiority. However, as we expand the subspace by moving the right boundary towards the right, these intersections become closer to each other, until they inevitably meet when there are no more minority cases in their unlabeled sets. This raises an important question, **given two (or more) policies, how can we use their curves in TM space to compare them?**

3.2.2 Comparing different policies

Results expressed in graphical measures can be more difficult to interpret than those reported in a single measure, specifically in the context of statistical significance tests. One scalar measure for evaluating different policies is to directly measure the quantity we are after – the total number of minority cases found at the end of the experiment. However, because of its conciseness, summarizing only the endpoint into a scalar metric, it lacks informativeness. Thus, it does not reflect the quickness property of the objective behind rare category exploitation.

Instead, we propose to integrate the area under the curve in temporal-minority space. This operation yields a scalar, named area under the temporal-minority curve abbreviated to **AUC-TM**[1]. Furthermore, since the TM space is confined by a closed shape, we know the maximum achievable AUC-TM and therefore we can scale it such that AUC-TM$\in [0, 1]$. If a policy has near optimal performance, it quickly

[1]The area under the TM curve can be calculated by using the trapezoidal areas created between each TM point.

detects most if not all the minority cases in the unlabeled set and the AUC-TM will be close to 1. By that AUC-TM does two things (i) assigns a value to finding minority cases sooner rather than later (2) allows us to conclude that a policy is superior to a second policy if it dominates the other for most or all of the points along their TM curves.

Note that AUC-TM calculations involve only the integration of the curve within the TM subspace of operation. Thus, the offset created by the initial training set is subtracted in the process and therefore, neither the number of initial minority cases nor the initial sample size (explicitly) affect this metric.

3.3 QTMC Methodology

This section shows that rare category exploitation can be handled using a finite sequential experiment with the objective of maximizing the AUC-TM metric. Given the problem constraints, φ and B, in what follows we present our QTMC methodology for rare category exploitation.

3.3.1 The scenario

QTMC deals with situations where data is abundant and cheap but label acquisition is bounded by some resource limitations (usually budget, bottlenecks of the labeling system or both). In this scenario, one or more learning algorithms have the freedom to sequentially query a pool of unlabeled instances which they think would help to maximize the total number of minority cases detected **over time**. Then, these queries are sent to an oracle and after some time, labeled instances are returned. The newly acquired samples are added to the labeled set, and in turn are used to retrain the classifier to choose new queries. The process is halted after a predefined timeframe or when some resource is exhausted or when further data acquisition is no longer beneficial (i.e. low quantity of minority cases left). Figure 3.2 illustrates the above process in the case of bank telemarketing.

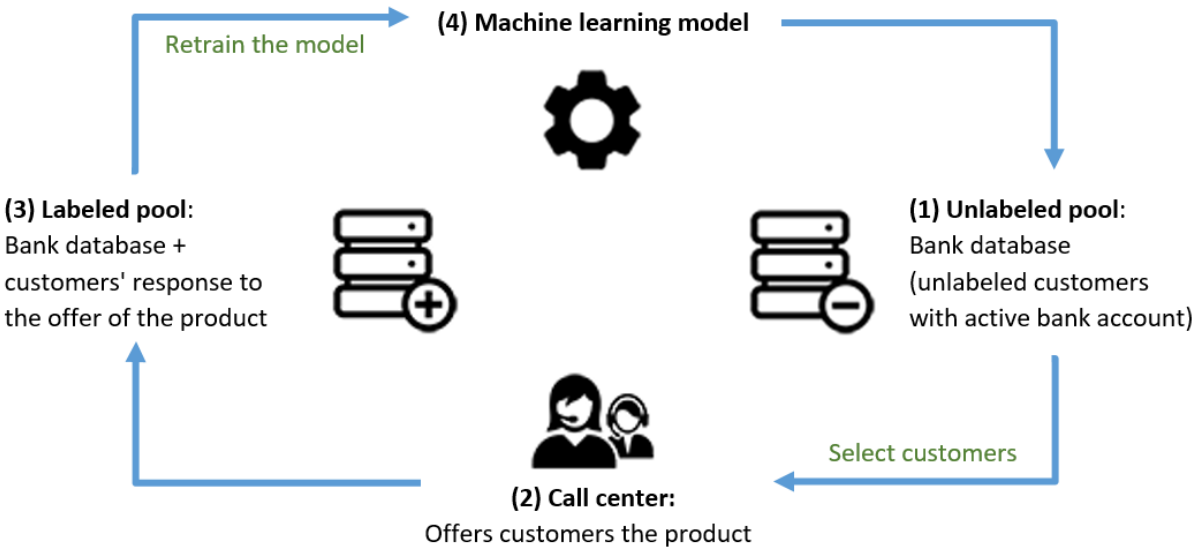

Figure 3.2: The pool-based QTMC cycle for bank telemarketing campaign.

3.3.2 Architecture

We enable the decision policy of selecting customers to utilize information from the labeled set and unlabeled set in order to make short-term operational decisions, medium-term tactical decisions and long-term strategic decisions. To do so, we fit different query learning concepts where each one represents a different assumption about what's the value of each unlabeled instance to achieve the goal – maximizing AUC-TM.

Figure 3.3 shows the 4th module presented in the pool-based QTMC cycle schematic (Figure 3.2). This module named "Policy Module" contains two main components: (1) Query Unit and (2) Policy Strategy.

A **query unit** is an implementation of a sampling approach and it usually consists of one or more classifiers (a counter example is the implementation of random sampling) with two inputs: a labeled pool and an unlabeled pool. The former is used for training the query unit while the latter is the set that can be queried. In addition, a query unit outputs ranks for the unlabeled pool according to the underlying assumption it was built upon (see "Algorithm Overview"). Note that the policy module is not limited to the number of query units it can hold, a property which allows us to design complex policies.

A **policy strategy** serves as an arbitrator, such that in addition to the query size , it gets the outputs of the query units, noted as $S_j,\ j = 1, ..., M$ and forwards a selected query Q to the human labeler (call-center in the case of bank telemarketing) with accordance to the deployed policy.

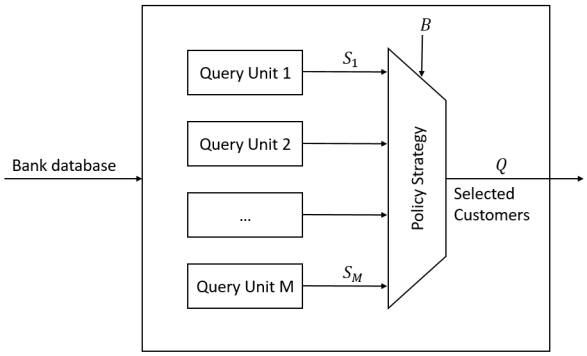

Figure 3.3: *Policy Module*, offers a unified framework to incorporate several query units' outputs into a single query batch with accordance to a predefined policy strategy. Each query unit outputs the ranks for the unlabeled instances, namely S_j. The policy strategy then arbitrates the different prospective customers into a single query batch of size B named Q.

In this paper we implement a policy module with three data-driven query units to address the explore-exploit dilemma that arises in reinforcement learning (see chapter 2). The first query unit ranks the instances within the unlabeled set in accordance to their expected influence on later rewards ("exploration"). The other two query units rank the unlabeled instances in accordance to their expected immediate rewards ("exploitation"), where the difference between them lies in their underlying assumptions about the data relation with the response variable (e.g. linear or nonlinear). Having those query units functioning enables us to develop policy strategies and compare them in terms of AUC-TM. Details about these policies and their implementations are given in the chapter 4.

3.4 Data sets

The main data set of this paper is the Portuguese bank database (denoted as *Bank*), which describes telemarketing phone call attempts to sell long-term deposits. The data set records include client information (e.g. age), telemarketing attributes (e.g. call direction), and external data (e.g. unemployment variation rate). Overall, there are 45,211 instances, of which 5289 belong to the minority class, setting the imbalance ratio at 7.5. The original intention behind this data set was to demonstrate the utility banks can have from leveraging such data. Different classification models were fitted and evaluated through holdout schematics [7]. In a more recent study by [13] this work was extended to predict the most recent 1293 contacts, from July 2012 to June 2013, with schematics of adopted moving window on the data prior to July 2012. As far as we know, this is the first time a finite sequential experiment is conducted with this data set.

Although utilizing temporal features such as the weekday and month of contact is interesting in its own right, it will not be pursued further here. Features which in practice we cannot control (e.g. unemployment variation rate) or features we don't have in advance (e.g. call duration) are discarded as well. These choices leave us with 8 out of 20 explanatory variables. More details about the *Bank* data set can be found in [13].

To demonstrate robustness across data sets, we study the performance of policies developed by QTMC on various benchmark real-world data sets, all of which are from the UCI Machine Learning Repository. *Letter* and *Satimage* are image data sets. *Letter* contains numerical attributes (statistical moments and edge counts) for the 26 capital letters in the English alphabet based on twenty different fonts with randomly added distortion. *Satimage* contains pixels information from satellite images describing different types of soil. *Abalone* contains physical measurements of sea snails where the purpose is to predict their age class. To transform this multi-class data set into a 2-class problem, the letter 'A' is used here as the minority class in *Letter* and 'class 4' is used as the minority class in *Satimage* and in *Abalone* instances labeled as 'class 7' are used to form the minority class. The prerequisites for the data set choices are (i) similar class imbalance ratio as in the bank telemarketing case ($5 \leq r \leq 50$) and (ii) fairly large number of observations ($n > 4000$).

To avoid overfitting and biasing the policy development for the *Bank* data set, we first conduct experiments with the rest of the data sets, draw conclusions and only then apply the policies to the *Bank* data set.

Table 3.2: Overview of the data sets.

Data set	**# Features**	**# Positive**	**# Negative**	**Imbalance Ratio**
Bank	8	5289	39922	7.5
Abalone-7	8	391	3786	9.7
Letter-A	16	789	19211	24.3
Setimage-4	36	606	5262	8.7

Chapter 4

Algorithms Overview

In this chapter we develop policies using the QTMC methodology which aim to offer robust AUC-TM performance under rare category exploitation. We start by constructing a non-data-driven benchmark policy that queries unlabeled instances at random, henceforth referred to as RANDOMINSTANCES. We then construct primitive building blocks: the exploration query unit and exploitation query unit and use them explicitly as policy strategies. Specifically, we consider a policy that always query the unlabeled instances with the highest probability of being minority cases, henceforth referred to as GREEDY, and a second policy which always gathers more information, henceforth referred to as INFORMATIVENESS. Importantly, the purpose of the following investigation is to identify the areas of strengths of such policies and address their shortcoming by incorporating other algorithms to create even more robust policies. For that, we present several simple strategies for the arbitration between the exploration and exploitation query units' outputs. Finally, in addition to the different sampling schemes, we use two different families of classifiers, Support Vector Machine (SVM) and Generalized Linear Models (GLM), to drive the exploitation query unit for gauging the robustness across classifiers.

4.1 RANDOMINSTANCES policy

The RANDOMINSTANCES policy is straightforward. Given the problem's constraints, B and φ, at each epoch the query unit samples without replacement B instances from the unlabeled set. Note that this policy does not aim to use the information from the input space, the sequential property of the process or prior knowledge about the finite horizon. For example, on the same data set, a setting of 10 epochs and batch size of 100 is identical to a setting of 2 epochs and a batch size of 500. Thus, at the end of the experiment the same properties would hold for the number of minority cases detected, regardless of the data set structure. For that reason RANDOMINSTANCES is robust across all settings and its discovery of minority cases at each epoch is dictated only by the imbalance ratio within the unlabeled set. However, the performance measure (specifically AUC-TM) might favor one setting over another, though, when used in simulations as a benchmark to compare with other policies, the setting across policies would be kept identical.

4.2 GREEDY policy

The GREEDY policy consists of choosing the batch of unlabeled instances with the highest probability estimations. A pseudo code for the generalized GREEDY policy is outlined in Algorithm 1. With this, we employ two versions of the exploitation query unit, one with SVM as a classifier and another with logistic regression (noted as GLM) as a classifier. Given the problem's constraints and the chosen probabilistic model, at each epoch of the experiment a classifier is exploiting what is already known in order to obtain minority cases. By this means, the newly acquired B instances are augmented to the labeled set and used to retrain the classifier for further epochs of exploitation.

input : Number of trials: φ
Probabilistic model: A
Number of query candidates: B
Labeled data set: $\mathcal{L}_0$
Unlabeled data set: $\mathcal{U}_0$

output : Updated labeled data set and unlabeled data set: $\tilde{\mathcal{L}}, \tilde{\mathcal{U}}$

for $t = 1, ..., \varphi$ **do**

1. Use the probabilistic model A to obtain a classifier h_t by fitting it on $\mathcal{L}_{t-1}$.
2. Apply the current classifier h_t to $\mathcal{U}_{t-1}$ and obtain the corresponding probabilities to be in the minority class.
3. Order the instances in descending order according to their probabilities, such that: $\hat{p}_{(1)} \geq \hat{p}_{(2)} \geq ... \geq \hat{p}_{(|\mathcal{U}_{t-1}|)}$.
4. Assign the B instances which correspond to $\hat{p}_{(j)},\ j = 1, ..., B$ to a set of query instances Q_t and obtain their labels.
5. Update the data sets as follows: $\mathcal{L}_t = \mathcal{L}_{t-1} \cup Q_t$ and then $\mathcal{U}_t = \mathcal{U}_{t-1} \setminus Q_t$.

end

Algorithm 1: A GREEDY Sampling Loop

4.3 INFORMATIVENESS policy

Instead of exploiting what is already known in order to obtain minority cases, another strategy can be to gather enough information to make the best overall decisions. Gathering more information can benefit us in two forms: *refinement* of the current classification model and *exploration* of poorly sampled regions. Thus, it helps for better understanding the unlabeled instances space which can then be exploited.

4.3.1 What is active learning?

One method to carry out that strategy is by performing active learning. As outlined in the chapter 2, active learning is primarily considered as a technique to reduce the number of training samples that need to be labeled for a classification task. Traditionally, it does so by quantifying the improvement of the classifier if an unlabeled instance is labeled and added to the training set (what we call the labeled set); this scalar is known as the **training utility value** (TUV). With these values assigned, the unlabeled instances with the highest TUVs are sampled at each epoch.

4.3.2 What active learning technique should I use?

Considering which active learning technique to implement, we notice that there is no clear winner in the competition for the best active learning technique or even a consensus about when to use each sampling technique [22],[10]. As demonstrated in Appendix B, a poor choice of technique which relies on false assumptions about the data behavior might result in performance inferior to random sampling. Typically, one uses K-fold cross-validation to compare different learning schemes when data is scarce. Sadly, there is no data to perform this procedure (and select a sampling technique) prior to the choice of a sampling technique. Even if we had an initial labeled set then (i) if we split unbalanced data into K parts, we are likely to get parts without minority instances or (ii) if the sample is not representative (say the imbalance ratio is 1) then the evaluation is unreliable.

So far the leading way to "play it safe" is to pick an active learning method that performs well on a wide range of data sets. Uncertainty sampling (discussed below) is a good candidate for that property [23],[10],[14]. This method won its credibility by being the most studied technique in the active learning literature. Furthermore, it serves as a baseline for comparison with other methods, albeit it's a two-edged sword because in these comparisons, when model class and feature set happened to be known in advance, uncertainty sampling is shown by researchers to be second-rate (Settles provides a comprehensive survey about the subject [10]).

4.3.3 What base learner should I use?

Uncertainty sampling methods iteratively request class labels for training instances whose classes are uncertain despite the previous labeled instances [23]. In other words, the less certain we are about an instance's true label, the more TUV it has. A typical uncertainty sampling method has in its heart a classifier, usually called the *base learner*. The choice of a base learner determines how to calculate the TUVs for the unlabeled instances. This brings up an important issue: what is the best base learner for the data at hand. Similarly to the problems encountered in choosing the active learning technique, there is no proper labeled set to compare between the different base learners' performances.

In this study, we demonstrate the use of a famous base learner, the SVM (specifically with radial basis kernel) which has been widely studied for this task [24]. In Appendix B we illustrate two cases, where uncertainty sampling fails and succeeds with comparison to a non-data-driven algorithm.

4.3.4 What are the components of uncertainty sampling?

To perform uncertainty sampling one needs an initial labeled set, a pool of unlabeled instances, a predefined base learner algorithm, and the ability to query instances. While these conditions are met we execute the cycle of the uncertainty sampling method described in Algorithm 2.

In the case of uncertainty sampling with SVM as a base learner, unlabeled instances that are closer to the empirical decision boundary get higher TUV scores. In our implementation, we assign each unlabeled instance $x_j \in \mathcal{U}$ a probability of being in the minority class $\hat{p}(y_j = +1|x_j)$. Regardless of which side of the decision boundary an unlabeled instance resides, the closer it gets to the decision boundary, the more uncertain we are about it. Here, an unlabeled instance with a probability of 0.5 corresponds to a datapoint that lies on the decision boundary, such that there is complete uncertainty about its true label. Furthermore,

in order to have values in the range of $[0, 1]$ we quantify the informativeness of an unlabeled instance as,

$$\text{TUV}_j = 1 - 2 \cdot |\hat{p}(y_j = +1|x_j) - 0.5|$$

Moreover, these point estimates can be transformed into ranks by simply ordering the unlabeled instances with accordance to their relative TUV within the sample. That is,

$$\text{TUV}_{(1)} \geq \text{TUV}_{(2)} \geq ... \geq \text{TUV}_{(|\mathcal{U}|)}$$

Then, the next query batch comprises of the B most informative instances which corresponds to $\text{TUV}_{(j)},\ j = 1, ..., B$.

input : Number of trials: φ
Base learner algorithm: A
Number of query candidates: B
Labeled data set: $\mathcal{L}_0$
Unlabeled data set: $\mathcal{U}_0$

output : Updated labeled data set and unlabeled data set: $\tilde{\mathcal{L}}, \tilde{\mathcal{U}}$

for $t = 1, ..., \varphi$ **do**

1. Use the base learner A_1 to obtain a classifier $h_{t,1}$ by fitting it on $\mathcal{L}_{t-1}$.
2. Apply the current classifier h_t to $\mathcal{U}_{t-1}$ and obtain the corresponding TUVs.
3. Order the instances in descending order according to their TUVs, such that: $\text{TUV}_{(1)} \geq \text{TUV}_{(2)} \geq ... \geq \text{TUV}_{(|\mathcal{U}_{t-1}|)}$.
4. Assign the B most informative instances which correspond to $\text{TUV}_{(j)},\ j = 1, ..., B$ to $\text{TUV}_{(j)},\ j = 1, ..., B$ to a set of query instances Q_t and obtain their labels.
5. Update the data sets as follows: $\mathcal{L}_t = \mathcal{L}_{t-1} \cup Q_t$ and then $\mathcal{U}_t = \mathcal{U}_{t-1} \setminus Q_t$.

end

Algorithm 2: A Generalized Uncertainty Sampling Loop

4.4 SEMIUNIFORM policy

In any specific case, whether it is better to explore or exploit depends in a complex way on the precise values of the estimates, uncertainties, and the number of remaining epochs. SEMIUNIFORM policy offers the simplest strategy for arbitrating between the exploration and exploitation query units' outputs.

This need for arbitration emerges because the group of examples which would be selected using the exploration query unit, $\text{TUV}_{(j)}, j = 1, ..., B$, and the group which would be selected using the exploitation query unit $\hat{p}_{_{(j)}}, j = 1, ..., B$, may have some intersection but may have none at all.

SEMIUNIFORM policy, employs uniform probability over the top B instances in each query unit output. First, for each query unit output, we subset the top B instances to a separate set. By reason of overlap between these sets, uniting them into a multiset would yield instances with more than a single occurrence. Second, we sequentially and randomly draw instances from the multiset while making sure that all occurrences of chosen instances are removed from it. This schema ensures that instances which have agreement among different query units, will have better chances to be selected. Algorithm 3 outlines

such an arbitration between two query units; this algorithm can be generalized to support more than two query units by simply including more query units' outputs in step 9.

input : Number of trials: φ
Base learner algorithm: A_1
Probabilistic model: A_2
Number of query candidates: B
Labeled data set: $\mathcal{L}_0$
Unlabeled data set: $\mathcal{U}_0$
output : Updated labeled data set and unlabeled data set: $\tilde{\mathcal{L}}, \tilde{\mathcal{U}}$

for $t = 1, ..., \varphi$ **do**

1. Use the base learner A to obtain a classifier h_t by fitting it on $\mathcal{L}_{t-1}$.
2. Apply the current classifier $h_{t,1}$ to $\mathcal{U}_{t-1}$ and obtain the corresponding TUVs.
3. Order the instances in descending order according to their TUVs, such that: $\text{TUV}_{(1)} \geq \text{TUV}_{(2)} \geq ... \geq \text{TUV}_{(|\mathcal{U}_{t-1}|)}$.
4. Define the B most informative instances which correspond to $\text{TUV}_{(j)},\ j = 1, ..., B$ as $S_{t,1}$.
5. Use the probabilistic model A_2 to obtain a classifier $h_{t,2}$ by fitting it on $\mathcal{L}_{t-1}$.
6. Apply the current classifier $h_{t,2}$ to $\mathcal{U}_{t-1}$ and obtain the corresponding probabilities to be in the minority class.
7. Order the instances in descending order according to their probabilities, such that: $\hat{p}_{_{(1)}} \geq \hat{p}_{_{(2)}} \geq ... \geq \hat{p}_{_{(|\mathcal{U}_{t-1}|)}}$.
8. Define the B instances which correspond to $\hat{p}_{_{(j)}},\ j = 1, ..., B$ as $S_{t,2}$.
9. **for** $b = 1, ..., B$ **do**
   a) Choose a non-empty set $S_{t,m}\ ,\ m = 1, 2$ at random.
   b) Sample x_b from $S_{t,m}$.
   c) Remove x_b occurrences from all sets.

   end
10. Assign the set of query instances $x_b,\ b = 1, ..., B$ to a set of query instances Q_t and obtain their labels.
11. Update the data sets as follows: $\mathcal{L}_t = \mathcal{L}_{t-1} \cup Q_t$ and then $\mathcal{U}_t = \mathcal{U}_{t-1} \setminus Q_t$.

end

Algorithm 3: SEMIUNIFORM Algorithm

4.5 ϵ-GREEDY policy

The ϵ-GREEDY strategy consists of selecting $100 \cdot (1 - \epsilon)$ percent of the leading instances suggested by the exploitation unit, and the rest are taken from the leading instances suggested by the exploration query unit. ϵ must be in the open interval $(0, 1)$ and its choice is left to the user. High values of ϵ would

encourage the policy to explore more, and low values result in a policy biased towards exploitation.

Inspired by reinforcement learning, there are many variations based upon heuristics involving the epoch's information. One strategy known as *probability matching* [9], reflects the idea that the number of pulls for a given handle should match its actual probability of being the optimal lever. An attempt to translate this strategy into the rare category exploitation setting would be as follows. Consider two actions $a_1, a_2 \in \mathcal{A}$ where a_1 and a_2 correspond to selecting the instances suggested by the exploration or exploitation query unit, respectively. In addition, recall $I_j \sim Ber(\rho_j)$ where $\rho_j = Pr(Y_j = +1|x_j)$ and define the *immediate reward* for the chosen B instances at epoch t as the following sum,

$$r^{(t)} = \sum_{b=1}^{B} I_b$$

We can now define the action-value as the mean reward for action a at epoch t to be,

$$Q(a) = \mathbb{E}\left[r^{(t)}|a^{(t)}\right]$$

Then, probability matching selects an action according to the probability that it is the optimal action. Formulating the last clause, the optimal value is,

$$Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

Naive estimation of $Q(a)$ can be done by summing the estimated probabilities of the top B instances according to each query strategy, that is

$$\hat{Q}(a) = \sum_{b=1}^{B} \hat{p}_b$$

The relation between the expected action-reward is then,

$$0 \leq \hat{Q}(a_1) \leq \hat{Q}(a_2) \leq B$$

Thus, with this procedure the algorithm will always exploit, albeit there are a couple reasons why this is not advisable. Consider what might happen as $\hat{Q}(a_1) \leftarrow \hat{Q}(a_2)$. Either the algorithm is more uncertain about the instances' true labels, or there are no more minority cases and the algorithm is quite certain about it. In the first scenario, it makes sense to explore more to gain information, while in the second scenario it makes sense to stop the experiment altogether (a concern outside of the scope of this paper).

Although the idea of probability matching leads us to the GREEDY strategy, it also encourages us to design an algorithm that is context aware, such that it can adaptively balance between exploration and exploitation using the epoch's information. Consequently, we propose a variant of the ϵ-GREEDY algorithm in which ϵ is dynamically determined by a statistic of the available information at each epoch. Instead of sampling the output of one query unit, we choose a portion of the top suggestions made by the exploitation query unit and the rest from the exploration query unit.

The QUANTILE-GREEDY described in Algorithm 4 , determines ϵ by using information from the current unlabeled set. At each epoch, it uses a probabilistic model to estimate the probability of each unlabeled instance to be a minority case. This information is already generated in the exploitation query unit and if accessible it is a convenient way to retrieve that information. Then, the QUANTILE-GREEDY

policy summarizes the estimated probability distribution in the form of the $B/|\mathcal{U}|$ quantile, hence the name.

By setting $\epsilon = 1 - \hat{p}_{_{(B)}}$, we select the top $K = \lceil (1-\epsilon) \cdot B \rceil \equiv \left\lceil \hat{p}_{_{(B)}} \cdot B \right\rceil$ instances suggested by the exploitation query unit, and the remaining $B - K$ instances form the exploration query unit output (after excluding those that were already chosen in the first K spots). In this way we ensure that in uncertain situations, or equivalently when the immediate reward is low, we explore more and vice versa.

Note that the quantile statistic enjoys two properties: it encapsulates information across the full distribution rather than a subset of it, and it is more appropriate in unbalance settings as the distribution is skewed towards the majority class (i.e. zero values).

input : Number of trials: φ
Base learner algorithm: A_1
Probabilistic model: A_2
Number of query candidates: B
Labeled data set: $\mathcal{L}_0$
Unlabeled data set: $\mathcal{U}_0$

output : Updated labeled data set and unlabeled data set: $\tilde{\mathcal{L}}, \tilde{\mathcal{U}}$

for $t = 1, ..., \varphi$ **do**

1. Use the base learner A to obtain a classifier h_t by fitting it on $\mathcal{L}_{t-1}$.
2. Apply the current classifier $h_{t,1}$ to $\mathcal{U}_{t-1}$ and obtain the corresponding TUVs.
3. Order the instances in descending order according to their TUVs, such that: $\text{TUV}_{(1)} \geq \text{TUV}_{(2)} \geq ... \geq \text{TUV}_{(|\mathcal{U}_{t-1}|)}$.
4. Define the B most informative instances which correspond to $\text{TUV}_{(j)},\ j = 1, ..., B$ as $S_{t,1}$.
5. Use the probabilistic model A_2 to obtain a classifier $h_{t,2}$ by fitting it on $\mathcal{L}_{t-1}$.
6. Apply the current classifier $h_{t,2}$ to $\mathcal{U}_{t-1}$ and obtain the corresponding probabilities to be in the minority class.
7. Order the instances in descending order according to their probabilities, such that: $\hat{p}_{_{(1)}} \geq \hat{p}_{_{(2)}} \geq ... \geq \hat{p}_{_{(|\mathcal{U}_{t-1}|)}}$.
8. Define the B instances which correspond to $\hat{p}_{_{(j)}},\ j = 1, ..., B$ as $S_{t,2}$.
9. Set $\epsilon = 1 - \hat{p}_{_{(B)}}$.
10. Select the first $K = \lceil (1-\epsilon) \cdot B \rceil \equiv \left\lceil \hat{p}_{_{(B)}} \cdot B \right\rceil$ from $S_{t,2}$, remove their occurrences (if any exist) in $S_{t,2}$, and select the first $B - K$ from $S_{t,1}$.
11. Assign the set of query instances $x_b,\ b = 1, ..., B$ to a set of query instances Q_t and obtain their labels.
12. Update the data sets as follows: $\mathcal{L}_t = \mathcal{L}_{t-1} \cup Q_t$ and then $\mathcal{U}_t = \mathcal{U}_{t-1} \setminus Q_t$.

end

Algorithm 4: QUANTILE-GREEDY Algorithm

Chapter 5

Experiments and Empirical Evaluation

5.1 Experimental setup

To study the performance of our approach under the rare category exploitation set-up, we start by defining the number of epochs φ, and the number of unlabeled instances B we can query at each epoch. Here, we derive those parameters from the *Bank* data set description; 1293 customers were contacted in a period of one year. In practice, if a customer hung up the phone we consider it as (immediate) rejection. In contrast, we can expect unsolicited calls won't yield immediate acceptance. That is, a non-negligible portion of the contacted customers, especially those who would accept the offer, will take time to consider the offer. This strongly suggests we should wait a period of time before retraining the model, or otherwise we may bias the model by creating even a greater imbalance ratio in the labeled set. While waiting, the results are augmented and appended to the labeled set for the next model retraining phase. Hence, we assume epochs of one month are sufficient for accumulating enough customers' responses for retraining the model. This leads us to $\varphi = 12$ and roughly $B = 12$.

5.2 Overview of simulations

Each simulation begins by isolating $\frac{1}{3}$ of the data set to serve as a test set (which is merely needed for surrogate criteria calculations such as AUC and PRBEP calculations). The remaining $\frac{2}{3}$ are the unlabeled instances. We randomly allocate 100 of these instances to the initial labeled set. To comply with the assumption that the initial labeled set is sufficient to produce a useful probabilistic model, draws where less than five minority cases are present in the labeled set, are discarded. Thus we impose $r \leq 19\ (95/5)$ in the initial labeled set. For all methods being evaluated, the same labeled and unlabeled instances are available at inception. In this way, we provide an identical starting point and identical unlabeled pool to choose from across different policies. Each run, or equivalently repetition, is carried out in a sequential fashion, where at each epoch we execute the procedures described in the chapter 4. The sequence is terminated when the unlabeled set is exhausted or after 48 epochs (4 years), whichever comes first[1]. The results we report reflect average performance over 100 repetitions, which differ from one another exclusively in the initial random assignment of the labeled set (i.e. the test set is identical across repetitions).

[1] Although it's redundant to perform more than twelve (the predefined φ) epochs, we supply the reader graphical information about the behavior of the policies in later phases.

To calculate surrogate metrics such as ROC-AUC (see Appendix A), we fit an SVM classifier to the labeled set and evaluate the performance by predicting the test set with that model. Note that our exploration query unit is based on an SVM classifier as well, while our exploitation query units are based on either an SVM or a GLM. The consistency between the query unit internal classifiers and the classifier used to predict the test set is not required, although in the active learning literature it's advised that both of them would belong to the same family of classifiers [10].

5.3 Classifiers

Unless stated otherwise we use *svm* (e1071 package) in R with default settings[2] for classification or *cv.glmnet* (glmnet package) with default settings. In both cases, before the data is fed into the classifier, we transform the factor variables into a design matrix by assigning a dummy variable for each factor level. In some cases, especially in earlier epochs, the classifier needs to handle a feature matrix (which includes the design matrix and other numeric variables) with properties of high dimensionality and sparsity. It is often the case that the data is not sufficiently rich for fitting a prediction model. Specifically, a logistic regression model requires to find a global maximum in the likelihood function; unfortunately, there are many situations in which such function has no maximum. To facilitate the problem, we start each epoch by removing dummy variables with no variance. Second, we penalize the likelihood via elastic net regularization [25] where the regularization parameters are chosen by internal procedure in the *cv.glmnet* function. Thus, no further input arguments are given to either of the classifier but for the modified data set. As mentioned before, high dimensionality and sparsity are outside the scope of this paper.

5.4 Experiments

This section describes our experiments for evaluating several policies for the rare category exploitation problem using the UCI data sets. To evaluate the efficacy of the GREEDY, INFORMATIVENESS, SEMIUNIFORM and ϵ-GREEDY policies we compare their performances to a benchmark and assess their robustness for a variety of data sets. For that purpose, we consider the RANDOMINSTANCES policy as a benchmark, which in expectation discovers minority cases at each epoch with accordance to the prevalence in the unlabeled set. The resulting comparison includes six different settings over four data-driven policies and one non-data-driven policy.

Figure 5.1 contains graphical performance measures in TM space for the sequential experiments. Different columns correspond to different data sets while different rows correspond to a different classifier in the exploitation query unit. As explained in the subsection 3.2.1 (Temporal-Minority space), the confined right trapezoid in each image defines the boundary of the experiment where each curve inside reflects an average performance over 100 repetitions. The closer a policy curve is to the left boundary, the better the performance it exhibits. Recall that our exploration query unit always uses SVM as a base learner and that the INFORMATIVENESS policy is the embodiment of a pure exploration query unit. Therefore, the different realizations of the exploitation query unit do not concern the INFORMATIVENESS performances and thus, it can serve as a visual anchor when comparing different rows. This trait also holds for the non-data-driven policy. As its name suggests, it samples instances at random and makes no

[2] radial *kernel*, *cost* of constraint violation $= 1$, and *scale* on.

use of the accumulated information. Statistics about the inter-repetition performance, shown in Figure 5.2, shed information on the variation of each policy and supply evidence for its degree of robustness across the different settings.

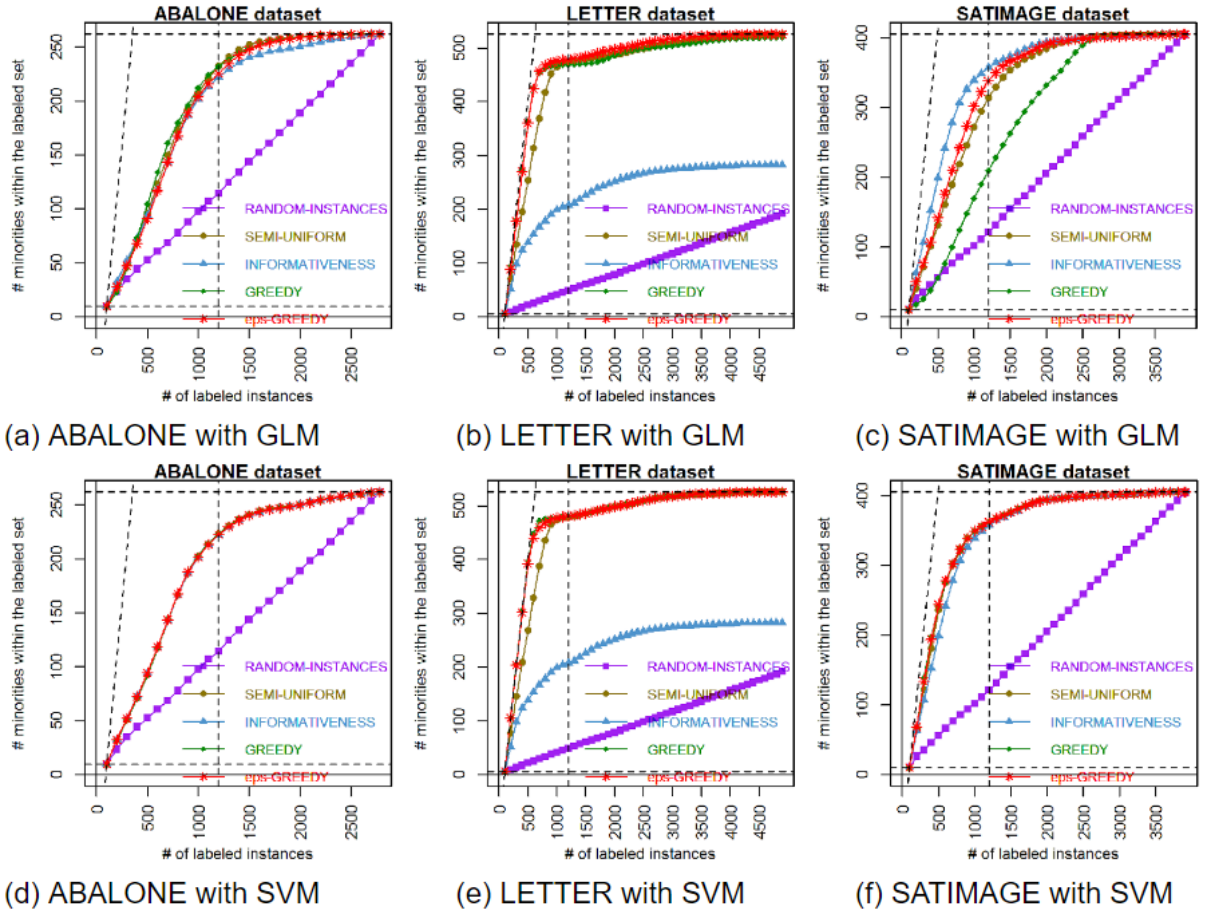

Figure 5.1: Performance of the data-driven policies relative to the non-data-driven policy under different data sets and classifiers of the exploitation query unit. Each figure (a-f) presents the temporal-minority space with TM curves averaged over 100 repetitions. In each setting, the confined right trapezoid (within the dashed lines) presents the experiment's finite horizon, where the intersection points between each curve and the right boundary are the (averaged) total number of minority cases discovered at the end of the experiment.

5.5 Analysis of the experimental results

Let us first examine the INFORMATIVENESS policy. Note that in all six settings it outperforms the benchmark in terms of AUC-TM (shown in Figure 5.2) in addition to 2-4 times increase in capturing minority cases for predefined constraints (shown in Figure 5.1). Interestingly, in the case of *Satimage* with GLM depicted in Figure 5.1(c) and Figure 5.2(c), it uniformly dominates all other policies. However, for the same data set under a different classifier, *Satimage* with SVM depicted in Figure 5.1(f) and Figure 5.2(f), there is no noticeable difference between the data-driven policies. This behavior of mixed results, suggesting that pairs are better, worse or similar to one another in terms of AUC-TM across different data sets, occurs in other data-driven policies as well. Thus, the different settings introduce extra variability for the policies' behavior, which means that we are uncertain about their behavior. However, in practice

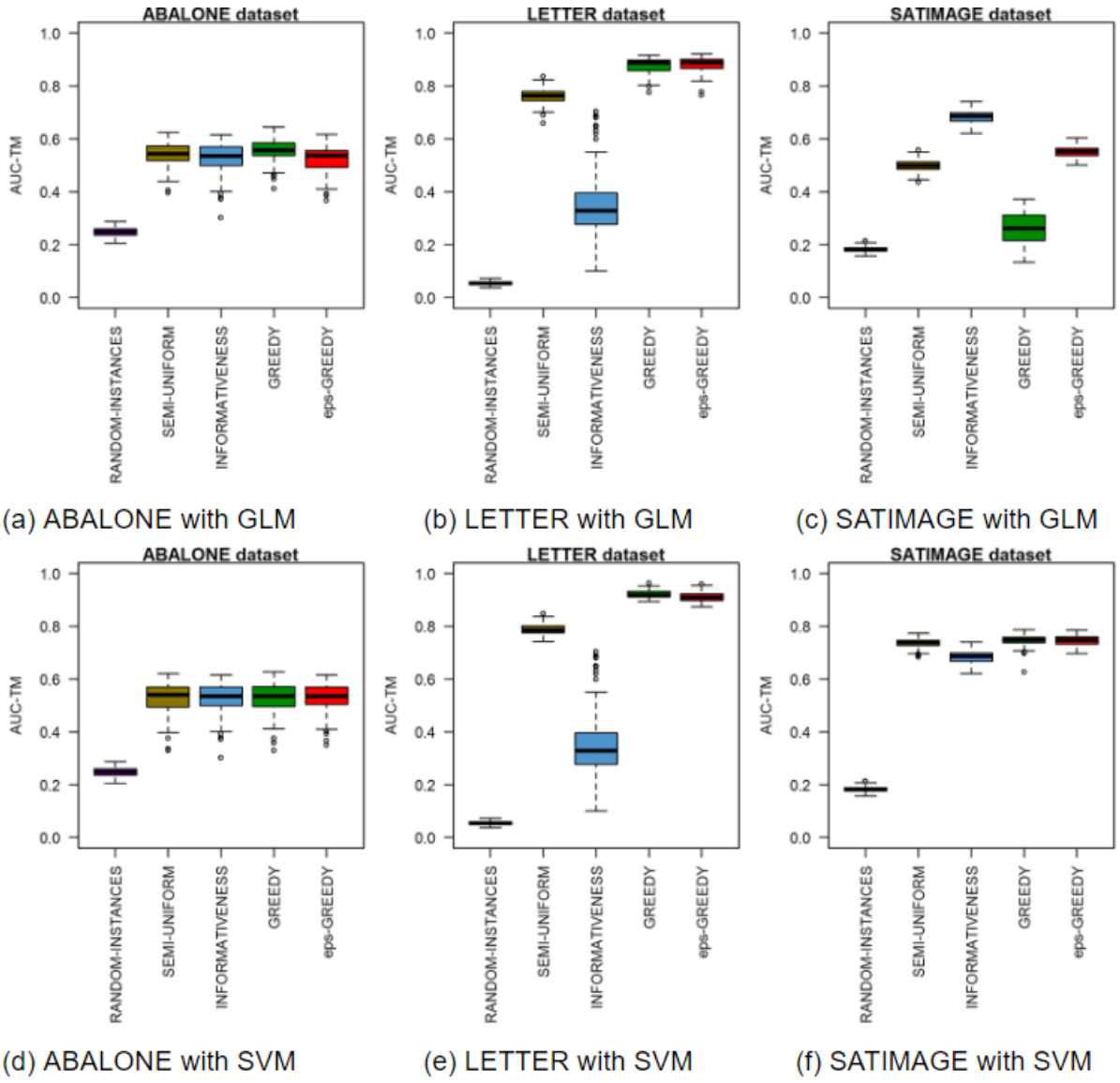

Figure 5.2: Inter-repetition performances of the data-driven policies relative to the non-data-driven policy under different data sets and classifiers of the exploitation query unit. Each figure (a-f) presents boxplots for 100 AUC-TM values accomplished by each policy under a specific setting. RANDOMINSTANCES and INFORMATIVENESS are identical across rows for the same data set by design, whereas the others alter their statistical properties such as variance and median. This demonstrates the challenge of finding robustness policies which perform well across different settings.

a policy is most useful if it exhibits robust performance across multiple settings, and does not perform poorly in any setting.

The ideal outcome of the experiment then, is a policy which satisfies two criteria: (1) **better** than the benchmark of non-data-driven policy, and (2) **not substantially worse** than other data-driven policies. To check these two assumptions, we perform Wilcoxon signed-rank test for the differences between policies' AUC-TM across 100 repetitions for one year of ongoing sequential experimentation (12 epochs). All tests are conducted at a 5% significance level with conservative correction for multiple comparisons. With 16 tests, 3 data sets and 2 classifiers families, the p-value threshold placed at 5e-4 (0.05/96).

First, we assume that the non-data-driven policy yields equal or better results than any other policy.

Table 5.1 reports the p-values for that null hypothesis against the alternative that the non-data-driven policy is worse than a specific data-driven policy (written in the column header). In all settings the AUC-TM differences between the RANDOMINSTANCES policy and the rest are statistically significant. In fact, all but one (*Satimage*-GLM-GREEDY) of the reported values are identical to the lowest achievable p-value under this test[3]. Although those finding are not surprising given the boxplots in Figure 5.2, validating that any data-driven policy satisfies the first condition is imperative.

Second, we compare all the combinations of data-driven policy pairs. In each test, the null hypothesis assumes the first policy is equal or better than the second, while the alternative states the complementary. In Appendix C, Table C.1 reports the p-values for these tests where the policy names written in the rows and columns represent the first and second policies of the pair, respectively. Cases where the null hypothesis is rejected are colored in red. Here the supremacy of a policy across all settings is not readily seen. Consequently, we count the number of times a policy had been significantly dominated by another in any of the settings (right column). However, making conclusions based only on these results disregards the information about the severity of the miss in each case. Therefore, we report the differences in percentage for each policy pair in Table 5.2 which has a similar structure to Table C.1. For example, the first red value in the top left cell indicates that INFORMATIVENESS policy achieved 97.3% of the obtained AUC-TM by SEMIUNIFORM policy under the same setting.

Table 5.1: Paired Wilcoxon for AUC-TM differences between policies across 100 repetitions. Each cell contains the p-value for the hypothesis whether the column policy is better than non-data-driven strategy (RANDOMINSTANCES policy).

Data set	GREEDY's classifier	SEMIUNIFORM	INFORMATIVENESS	GREEDY	ϵ-GREEDY
Abalone	GLM	1.98E-18	1.98E-18	1.98E-18	1.98E-18
	SVM	1.98E-18	1.98E-18	1.98E-18	1.98E-18
Letter	GLM	1.98E-18	1.98E-18	1.98E-18	1.98E-18
	SVM	1.98E-18	1.98E-18	1.98E-18	1.98E-18
Satimage	GLM	1.98E-18	1.98E-18	8.13E-17	1.98E-18
	SVM	1.98E-18	1.98E-18	1.98E-18	1.98E-18

Typically, robust policies are a bit worse than optimal where the source of their robustness is the local sub optimality. Therefore, to examine the second criteria of "not substantially worse", we summarize in Table 5.3 the number of times each policy was dominated by another side by side with the mean percentage difference when that scenario happened. In 11 out of 27 cases (the highest) the INFORMATIVENESS policy was performing worse than the other three data-driven policies, placing it as the least optimal policy. In comparison to that, the GREEDY policy was performing worse in 3 out of 27 cases (the lowest). However, when the latter happened the GREEDY policy achieved on average 46.5% AUC-TM relative to the competing policies (the worst). That result suggests that the GREEDY policy usually works well, and when it doesn't it performs substantially worse. Among the two hybrid policies, SEMIUNIFORM and ϵ-GREEDY, the latter prevailed over the former in both measured properties. The ϵ-GREEDY policy exhibited robust performance across multiple settings (with 4 cases of being dominated by other policies) and did not perform poorly in any setting with average achievement of about 8% difference from the optimal policy when the difference happens to be statistically significant.

[3]Under *wilcox.test* implementation in R

Thus far, we conducted various experiments to validate the proposed QTMC methodology to handle the rare category exploitation problem. To avoid overfitting and biasing the policy development to the *Bank* data set, we put it aside and monitored the performance and robustness of some policies across other data sets. Equipped with the aforementioned insights, we check for their validity on the *Bank* data set.

Figure 5.3 shows the behavior of different policies using the earlier experimental settings on the *Bank* data set. Because of the sheer number of minority cases (5289 with comparison to the 1200 allowed instances in the experiments), we observe that the subspace is confined by a right triangle. In both scenarios, under GLM or SVM as the classifier in the exploitation query unit, the experiment horizon depicted in Figure 5.3(a,b) have similar TM curves for all data-driven policies. Seemingly, all policies perform better than the non-data-driven policy both in terms of AUC-TM and in the total minority cases detected (by a factor of 2). The corresponding boxplots shown in Figures 6(c,d) confirm these observations. There are slight changes in the inter-repetition statistics between the two scenarios where no distinct data-driven policy looks worse than the others. Interestingly, Figures 6(a,b) give us future outlooks as if the experiments would continue longer. Specifically, in the GLM setting we observe how the TM curves branch out from one another displaying clear advantage towards the exploration-exploitation hybrid policies.

Clearly, the different classifiers and data sets' structures are not the only elements that affect the policy performance. The end point of the experiment given by the finite horizon property has a role as well. As the experiment prolongs further, new observations about the differences in the policies' performance are revealed.

Table 5.2: Paired Wilcoxon for AUC-TM differences between data-driven policies across repetitions. Each tuple checks the hypothesis of whether a policy (mentioned in the row) is **not worse** than another policy (mentioned in the column). Red colored values indicate that the null hypothesis (i.e. the row policy is not worse than the column policy) has been rejected. The values themselves are the averaged differences between the two policies (in percentage).

Data set	GREEDY's classifier	Policies	SEMIUNIFORM	INFORMATIVENESS	GREEDY	eps-GREEDY	Dominated counter
Abalone	GLM	SEMIUNIFORM	100	103.4	97.3	103.8	1
		INFORMATIVENESS	97.3	100	94.6	100.8	2
		GREEDY	103.6	107.1	100	107.4	0
		ϵ-GREEDY	96.8	99.9	94.1	100	2
	SVM	SEMIUNIFORM	100	100.1	100.2	99.7	0
		INFORMATIVENESS	100.3	100	100.3	99.7	0
		GREEDY	100.4	100.3	100	99.9	0
		ϵ-GREEDY	100.7	100.5	100.7	100	0
Letter	GLM	SEMIUNIFORM	100	237.1	86.9	86.3	2
		INFORMATIVENESS	46.3	100	40.4	40.1	3
		GREEDY	115.2	274	100	99.4	0
		ϵ-GREEDY	116	275.8	100.7	100	0
	SVM	SEMIUNIFORM	100	246.7	85.7	86.7	2
		INFORMATIVENESS	44.7	100	38.4	38.8	3
		GREEDY	116.8	288.9	100	101.1	0
		ϵ-GREEDY	115.5	286	98.9	100	1
Satimage	GLM	SEMIUNIFORM	100	73.1	199.4	90.9	2
		INFORMATIVENESS	137.2	100	273.6	124	0
		GREEDY	52.9	38.7	100	47.9	3
		ϵ-GREEDY	110.6	80.7	220.6	100	1
	SVM	SEMIUNIFORM	100	108	98.8	98.7	2
		INFORMATIVENESS	92.7	100	91.6	91.5	3
		GREEDY	101.3	109.3	100	99.9	0
		ϵ-GREEDY	101.3	109.4	100.1	100	0

Table 5.3: Summary of the null hypothesis rejections for the second set of tests. The first row counts the rejections (suggesting how many times a policy performance was significantly inferior to other policies), whereas the second row averages the AUC-TM differences (in percentage) in the cases where the null hypothesis was rejected.

	SEMIUNIFORM	INFORMATIVENESS	GREEDY	ϵ-GREEDY
# of rejections	9	11	3	4
Mean percentage difference	89.3	65.1	46.5	92.6

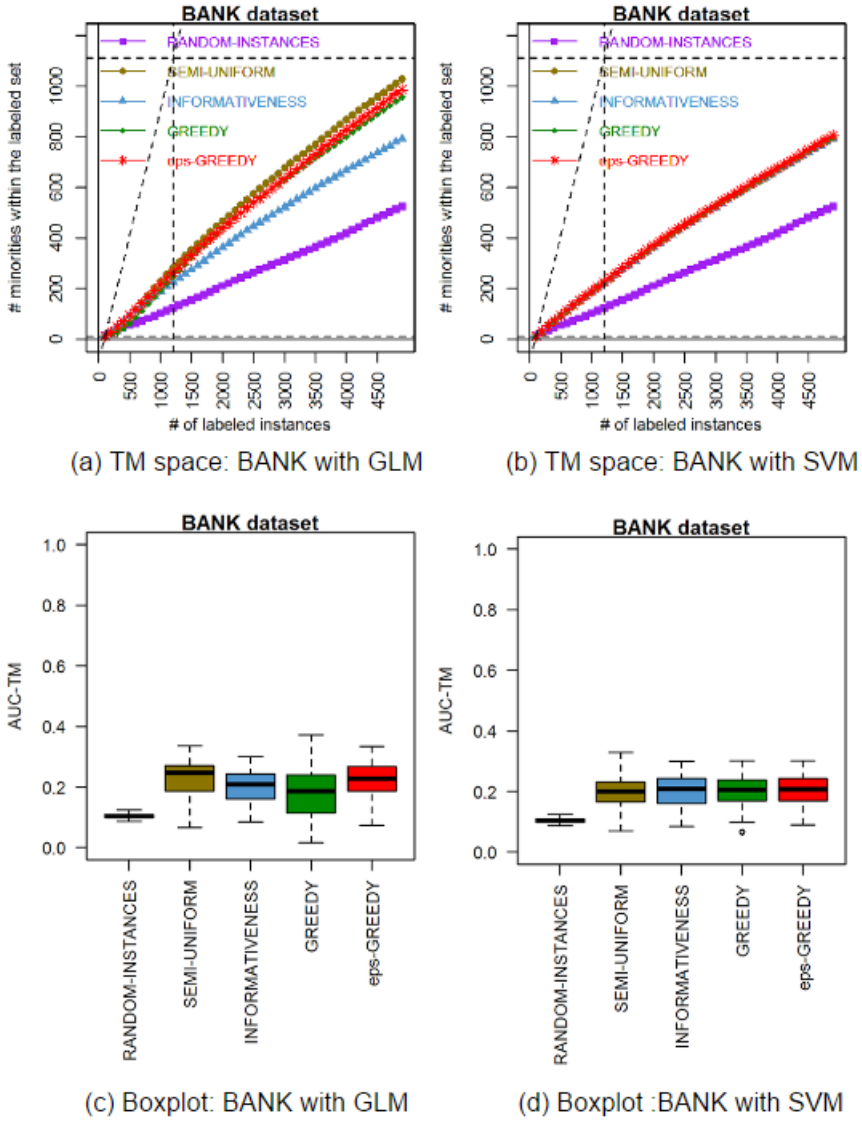

Figure 5.3: Performance of the data-driven policies in the *Bank* data set. (a) TM space with TM curves when logistic regression is employed in the exploitation query unit. (b) Same as before, but with SVM employed. (c) and (d) boxplots for the corresponding 100 AUC-TM values obtained in (a) and (b), respectively.

Chapter 6

Conclusion

In this paper, for the first time, we have identified and addressed the problem of rare category exploitation. Our key contribution is twofold: (1) We have proposed a unified framework for developing policies for rare category exploitation, and (2) we have developed data-driven policies that aim to provide robust performance across different data sets and classifiers which are not worse than non-data-driven methods.

So far the experimental results have demonstrated that policies combining exploration and exploitation strategies can serve as a "safe bet" when one faces the explore-exploit dilemma. In contrast, a common active learning approach – uncertainty sampling, is often outperformed by other data-driven policies in terms of quickly finding minority cases. In addition to that, explicitly exploiting what is already known in order to obtain minority cases works well, but when it fails the performance is substantially worse than other data-driven policies.

Since this is the first attempt to provide a common evaluation of strategies for quickly targeting the minority class, the comparison should still be viewed as preliminary. Further experiments with other data sets and settings might lead to other interesting observations.

The three main points which need to be discussed further for QTMC to be useful in real-life are:

- How to make a data-driven decision when to stop the experiment? This operating question will quickly occur to the alert practitioner. For example, if the time horizon is far and we have a rough assessment that the minority cases are almost depleted from the unlabeled pool, should we continue with the experiment?

- How the temporal features can be incorporated into the scheme? Instead of a fixed number of responders in a data set, we can imagine each customer will respond to an offer only during specific hatches in the experiment's time frame. This, for example, can be attributed to the period of the year, or to the life cycle stage of the product (innovative, mature, etc.).

- How to create robust policies that perform well across classifiers? We notice that different classifiers perform differently. Can we harness the best of each model? What is a proper way to ensemble classifiers' predictions or outputs?

Appendix A

Evaluation Metrics

In most cases, the generalization performance of a learning method relates to its prediction capability on independent test data. Under our settings, this is not the case: First, we have no test set since our objective is to discover the minority cases within the unlabeled set. Second, when the experiment is terminated, the model becomes obsolete, such that no more unseen data is predicted. In this chapter we present the common metrics for evaluation of learning algorithms and demonstrate that even though these criteria appear acceptable in other papers, they are not useful as a result of the underlying constraints present here.

A.1 Accuracy

In a two-class problem the common method for determining the performance of a classifier is through the use of a confusion matrix. Here, the classifier assigns instances to belong either to the minority (positive) class or to the majority (negative) class. Then given the true class affiliations, the decisions are categorized into four types: True positives (TP) are instances correctly assigned to the minority class. False positives (FP) refer to majority instances incorrectly assigned as minorities. True negatives (TN) are instances correctly assigned to the majority class. Finally, false negatives (FN) refer to minority instances incorrectly assigned as majorities.

The confusion matrix is a source for a wide set of evaluation metrics. One such metric, the *accuracy* rate, is the most commonly used empirical measure,

$$\text{accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

However, accuracy is inadequate when data is unbalanced. Consider a realistic case when 8% of the customers within a database will respond to a given offer. In case we randomly split the database into two sets: train and test, a model asserting the test set has no minorities has an accuracy of 92%. Allegedly, this is a high value but it is clearly ignoring the performance of the model on the class that matters

A.2 Precision and Recall

Alternatively, two other measures can be defined from the confusion matrix: *precision* and *recall*.

$$\text{precision} = \frac{TP}{TP + FP} = \text{positive predictive value}$$

$$\text{recall} = \frac{TP}{TP + FN} = \text{sensitivity}$$

Projecting these equations on the last example, we see that precision measures how often a customer that was predicted as responsive actually responded to the offer, while recall measures how often a responsive customer within the test set was predicted as such by the model. Hence, under the problem setting the goal of the model is to improve recall without impairing precision. However, increasing the recall is an error prone activity where the FP error (falsely asserting a customer is responsive) is not being taken into account while it is embodied in the precision measurement. Thus, the precision and recall of this system depends on the choice of the thresholds above which estimated probabilities are assigned to the minority class. A lower threshold means higher recall, but usually also lower precision. This behavior is shown in Figure A.1(a,b). In our example, a cutoff point of 1 (which is equivalent to the earlier rule of asserting every customer is non-responsive) achieves a recall of 0% and undefined value of precision (its denominator is zero). On the opposite side, a cutoff point of 0 (which is equivalent to a rule of asserting every customer is responsive) achieves a recall of 100% and a precision of 8%. Therefore an applicative metric has to incorporate both measurements to evaluate the model performance in the setting of an unbalanced class distribution.

A.3 PR Space and PRBEP

One option to combine recall and precision is called precision-recall break-even point (PRBEP). Using the precision-recall (PR) space shown in Figure A.1(c), one plots recall on the x-axis and precision on the y-axis. Empirical calculation of confusion matrices for a large number of cutoff points, where each cutoff point serves as a threshold above which estimated probabilities are assigned to the minority class, yields points in that space. In this manner we get the precision of a classifier as function of its recall. At some point (when $FN = FP$), precision and recall are equal; this value is known as the PRBEP. Alternatively, we can visualise this point by finding the intersection between the PR curve of a classifier and a 45 degree line from the origin. When a classifier's PR curve[1] intersects with that line, its precision is equal to its recall, namely PRBEP. Figure A.1(c) shows the PR space with two performance curves. The intersections between each curve and the diagonal line are the projections of the 2D space into a scalar representation named PRBEP.

A.4 ROC Space and AUC

A commonly used evaluation metric to represent 2-class classification models is the Receiver Operating Characteristic (ROC) curve and its scalar representation the Area Under the Curve (AUC). This curve combines *TPR* (synonym to recall and sensitivity) and *FPR*, that is

$$\text{TPR} = \frac{TP}{TP + FN} = \frac{\text{minorities classified as minorities}}{\text{total minorities}} = \text{sensitivity}$$

$$\text{FPR} = \frac{FP}{FP + TN} = \frac{\text{majorities classified as minorities}}{\text{total majorities}} = 1 - \text{specificity}$$

[1] Extending a set of points in PR space into a curve is not straightforward; as the level of recall varies, the precision changes in a nonlinear fashion due to the fact that FP replaces FN in the denominator of the precision metric. Thus, linear interpolation between the set of points is an overly-optimistic approximation (for better approximations see [26])

Then, the ROC space is represented by plotting the FPR on the x-axis and TPR on the y-axis. The ROC curve shows how the number of truly responsive customers (TP) varies with the numbers of falsely asserted non-responsive customers as responsive (FP). Finally, the AUC is used to summarize the ROC graph by integrating the area under the ROC curve. A comparison of ROC space with PR space, precision and recall is shown in Figure A.1.

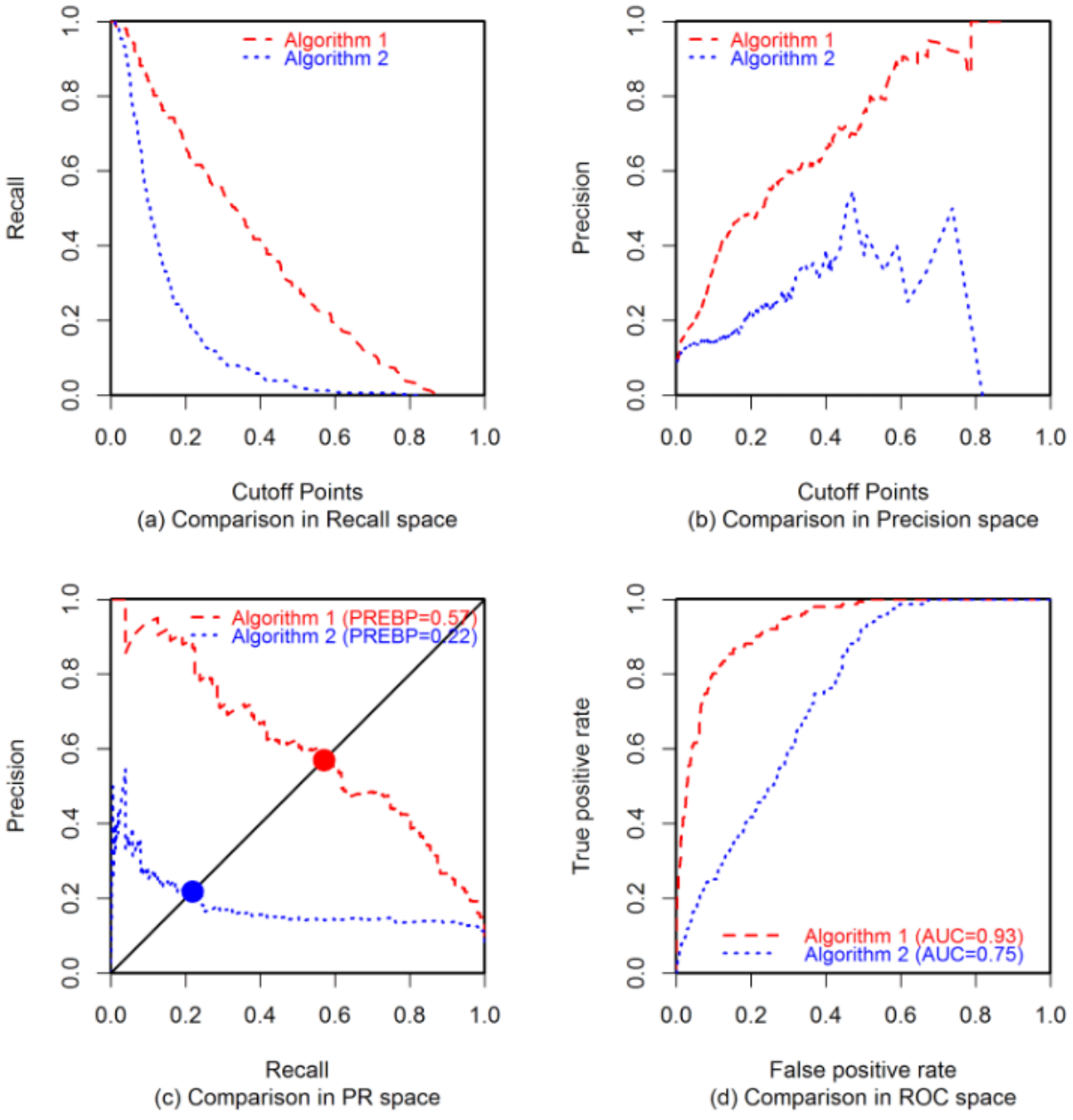

Figure A.1: Comparing the same algorithms at first epoch in four spaces (a) Recall space (b) Precision space (c) PR space and (d) ROC space. The axes are different in each space. The Recall and Precision y-axes define the PR space for the purpose of summarizing recall and precision into a scalar named PRBEP. [26] proved the relation between (c) and (d) such that a curve dominates in ROC space if and only if it dominates in PR space.

A.5 Problems with the ROC and PR spaces

ROC and PR curves describe classifiers' performance in 2D space at a specific epoch. In our application, a sequence of decisions needs to be made over time. Therefore, assuming that different epochs have similar effects on the performance is inadequate. This requires us to express the time dimension in each of these spaces. On the one hand, presentation of classifiers in a 3D space enables us to characterize the various classifiers and discover zones of optimality. On the other hand, the model evaluation process becomes cumbersome. A common remedy is to project the curves at each epoch into a scalar metric that incorporates all the details of the graph, where AUC and PRBEP are the corresponding projections of the ROC and PR curves. Inevitably, this process results in significant information loss, whereas for some criteria a compression of this kind has already occurred in the past. For example, the 2D ROC space is based on a 2x2 confusion matrix with 3 df, this suggests that additional information is lost when the ROC space is created.

Figure A.2 demonstrates how the ROC-AUC provides misleading guidance in selecting a policy to handle the rare category exploitation problem. It features two policies, which are laid out in chapter 4: RANDOMINSTANCES and INFORMATIVENESS. These policies are fitted on the *Satimage* data set (see section 3.4 for further details) and their performances are compared via AUC as a function of epoch (left side) and temporal-minority curve (right side). In fact, the latter is identical to Figure 5.1(d) where only the above-mentioned policies are displayed.

While running the simulations in chapter 5 (Experiments and Empirical Evaluation), we calculated AUC-TM and AUC-ROC concurrently by pre-allocating a data set for that purpose. In essence, the data set was split randomly into three isolated parts named the labeled set, the unlabeled set and the test set of 100 (2%), 3812 (65%) and 1956 (33%) instances, respectively. The test set is relatively large to allow better representation of the data at hand, albeit it is only crucial for calculating the AUC and has no role in constructing the temporal-minority space. At each epoch, each policy chooses 100 instances which in turn are pulled out from the unlabeled set and added to the labeled set with their corresponding labels. This process is carried out sequentially until the unlabeled set is exhausted.

Observing the AUC plot before the 1400 marker (vertical dotted line), it is clear that RANDOMINSTANCES' curve is dominant. By contrast, in the temporal-minority space INFORMATIVENESS is uniformly dominant along the process. A closer look at the marker reveals that although the AUC value of RANDOMINSTANCES is higher than that of INFORMATIVENESS (suggesting that the former is preferable), INFORMATIVENESS detects twice as many minority cases as RANDOMINSTANCES. Notice that at the marker, approximately 10% of the minority instances remain to be detected. Remarkably, as the unlabeled set drains out from minority cases and consequently **majority** instances flock into the labeled set, the AUC criterion increases. Therefore, the view taken here strongly suggests that we use criteria that can represent adequately the goal we want to achieve.

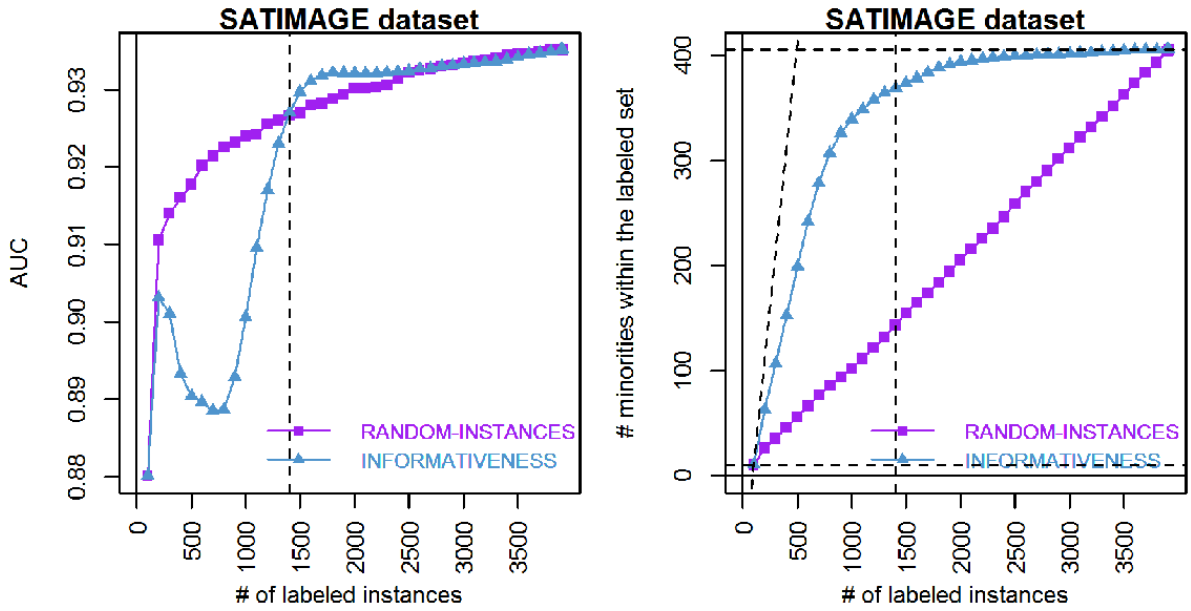

Figure A.2: Comparison of two criteria: (left) two policies measured by AUC; RANDOMINSTANCES policy dominates the evaluation until the 1400 marker. (right) the same policies viewed in the temporal-minority space; INFORMATIVENESS presents near-optimal performance, while RANDOMINSTANCES has random performance.

Appendix B

Uncertainty Sampling

In this Appendix, we illustrate the uncertainty sampling procedure in action and compare it to random sampling as a benchmark. For illustration purposes, in this chapter we employ SVM with linear kernel as the base learner in the uncertainty sampling mechanism. This SVM version finds linear boundaries in the input feature space which are easy to plot and elucidate the underlying mechanism. However, we can enlarge the feature space by using basis expansion[1] as in the case of radial kernel, usually at the expense of interpretability.

In SVM, the informativeness of an instance is synonymous with its distance to the hyperplane that forms the classification boundary. Thus, under uncertainty sampling one chooses to query the unlabeled instances that are the closest to the decision boundary. In what follows, we supply two illustrative examples of uncertainty sampling with comparison to random sampling and show where it may succeed or fail.

B.1 Scenario 1: Assumptions hold

Figure B.1 illustrates uncertainty sampling with Linear-SVM[2] as a base learner. This syntactic data set was generated from two multivariate normal distributions centered at (-2,0) and (2,0) with ones on the covariance matrix diagonal and zero otherwise. The two distributions that represent the majority and minority class are marked in Figure B.1(a) as - and +, respectively. There are 455 majority cases and 45 minority cases, which sets the imbalance ratio at . In Figure B.1(b), we hide 80% of the instances' labels randomly (which keeps the imbalance ratio the same) and fit an SVM model to the rest. Notice that for the given labeled instances an empirical separable linear decision boundary exists and it is depicted as a solid blue line with its margins and support vectors noted with circles. There are 35 minority cases left to be discovered and 365 majority cases to avoid. Figure B.1(c) describes a random draw of 30 instances which results in the discovery of 4 minority cases. Figure B.1(d) describes the choice of the closest instances to the decision boundary. In this case 19 out of 30 instances belong to the minority class. Note that among the choices, 7 minority cases dwell on the left side of the decision boundary. This helps to refine the empirical decision boundary (which was created from a random sample earlier) for future decisions and at the same time it contributes to the goal of increasing the minority cases in the present labeled set.

[1] The general study of SVMs goes beyond the scope of this paper. See [24] for further details.

[2] Here we use the implementation of svm (e1071 package) in R with linear kernel, default cost of constraint violation (= 1), and no scaling. The reader is refer to [24] for mathematical details about the parameters.

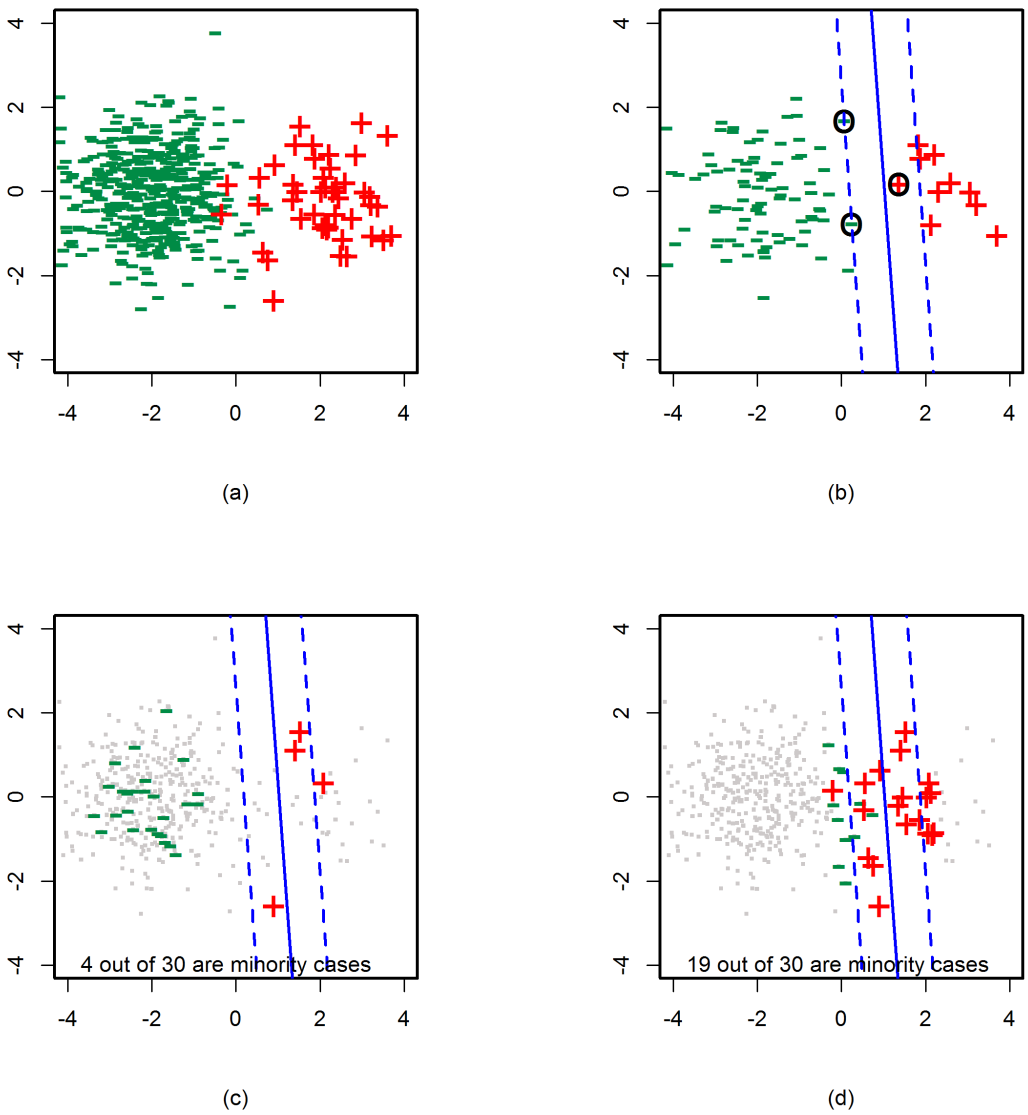

Figure B.1: An illustrative example of uncertainty sampling with Linear-SVM as base learner at the first epoch. (a) Synthetic data of 500 instances, unevenly sampled from two Gaussians with $r = 10$. (b) We randomly hide 80% of the instances' labels and fit an SVM model to the rest. The circles are the support vectors and the solid line represents the decision boundary of the classifier with its margins represented as dashed lines. (c) Choice of the next 30 instances using random sampling (shown with their real labels + or -) among the unlabeled instances (shown as dots). (d) The same as before but the next batch is chosen with uncertainty sampling.

B.2 Scenario 2: Assumptions do not hold

Each interaction between a classifier and a data set yields the ability to distinguish between classes to some degree. In cases where this degree is high, problems can occur with the uncertainty sampling schema.

Similarly to the previous scenario, we conduct an experiment with synthetic data along with one major difference: the multivariate normal means are centered at (-5,0) and (5,0), shown in Figure B.2(a). Considering the dispersion of the data, the relatively large distance between the means insures us that there

is a high degree of separation between the two classes. In Figure B.2(b) we hide 80% of the instances' labels randomly and fit an SVM model to the rest. There are 35 minority cases left to be discovered and 365 majority cases to avoid. Figure B.2(c) describes a random draw of 30 instances which results in the discovery of 4 minority cases. Figure B.2(d) describes the choice of data points with accordance to (SVM based) uncertainty sampling leading assumption that the closer a data point is to the decision boundary the more valuable it is. In this case 3 out of 30 instances belong to the minority class. Apparently, gaining more information in the case of highly distinct classes sacrifices great immediate reward for little to no benefit in understanding the unlabeled space for further exploitation.

Note that this is not a bad behavior of the classifier, on the contrary, the ability to discriminate (and make correct predictions) is a desirable attribute. Clearly, a policy which can take advantage of this behavior would gain from its benefit.

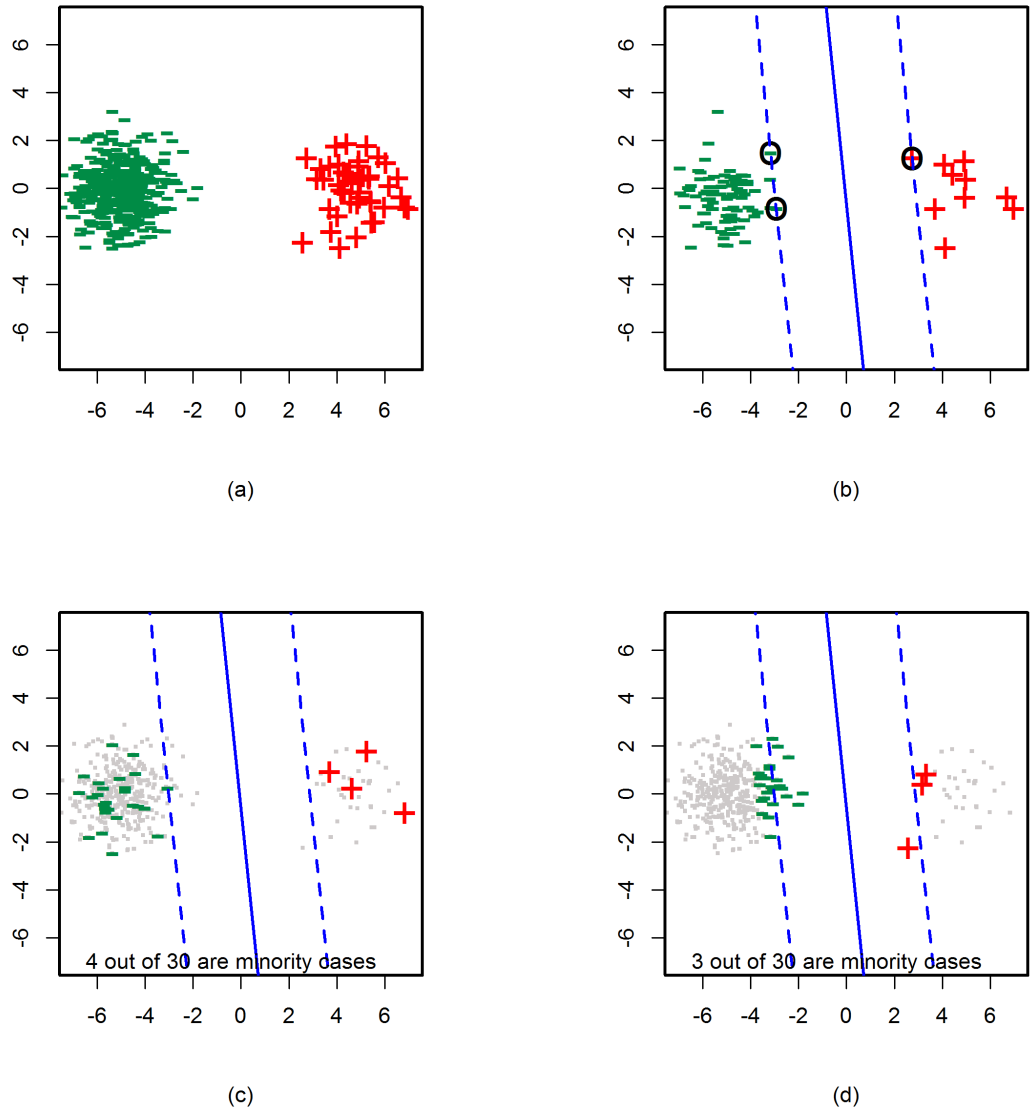

Figure B.2: An illustrative example of uncertainty sampling with Linear-SVM as base learner at the first epoch. (a) Synthetic data of 500 instances, unevenly sampled from two Gaussians with $r = 10$. (b) We randomly hide 80% of the instances' labels and fit an SVM model to the rest. The circles are the support vectors and the solid line represents the decision boundary of the classifier with its margins represented as dashed lines. (c) Choice of the next 30 instances using random sampling (shown with their real labels + or -) among the unlabeled instances (shown as dots). (d) The same as before but the next batch is chosen with uncertainty sampling.

Appendix C

Hypothesis testing p-values

Table C.1: Paired Wilcoxon for AUC-TM differences between data-driven policies across repetitions. Each tuple checks the hypothesis of whether a policy (mentioned in the row) is **not worse** than another policy (mentioned in the column) and reports the corresponding p-value. Red colored values indicate that the null hypothesis (i.e. the row policy is not worse than the column policy) has been rejected after conservative correction for multiple comparisons.

Data set	GREEDY's classifier	Policies	SEMIUNIFORM	INFORMATIVENESS	GREEDY	eps-GREEDY	Dominated counter
Abalone	GLM	SEMIUNIFORM	NA	0.999874	6.81E-05	1	1
		INFORMATIVENESS	0.000128	NA	1.57E-07	0.821149	2
		GREEDY	0.999933	1	NA	1	0
		ϵ-GREEDY	5.36E-08	0.179751	1.55E-09	NA	2
	SVM	SEMIUNIFORM	NA	0.37132	0.203169	0.164838	0
		INFORMATIVENESS	0.629979	NA	0.43691	0.096623	0
		GREEDY	0.797801	0.564465	NA	0.4856	0
		ϵ-GREEDY	0.836013	0.904076	0.51577	NA	0
Letter	GLM	SEMIUNIFORM	NA	1	1.98E-18	1.98E-18	2
		INFORMATIVENESS	1.98E-18	NA	1.98E-18	1.98E-18	3
		GREEDY	1	1	NA	0.008552	0
		ϵ-GREEDY	1	1	0.991528	NA	0
	SVM	SEMIUNIFORM	NA	1	1.98E-18	1.98E-18	2
		INFORMATIVENESS	1.98E-18	NA	1.98E-18	1.98E-18	3
		GREEDY	1	1	NA	1	0
		ϵ-GREEDY	1	1	4.69E-13	NA	1
Satimage	GLM	SEMIUNIFORM	NA	1.98E-18	1	2.89E-18	2
		INFORMATIVENESS	1	NA	1	1	0
		GREEDY	1.98E-18	1.98E-18	NA	1.98E-18	3
		ϵ-GREEDY	1	1.98E-18	1	NA	1
	SVM	SEMIUNIFORM	NA	1	1.45E-11	1.36E-14	2
		INFORMATIVENESS	1.98E-18	NA	2.52E-18	1.98E-18	3
		GREEDY	1	1	NA	0.818383	0
		ϵ-GREEDY	1	1	0.182539	NA	0

Bibliography

[1] Theodore Levitt. EXPLOIT the Product Life Cycle. *Harvard Business Review*, 43(6):81–94, 1965.

[2] Charles X. Ling and Chenghui Li. Data Mining for Direct Marketing: Problems and Solutions. *Fourth International Conference on Knowledge Discovery*, 98:1–7, 1998.

[3] Carole Page and Ye Luding. Bank managers' direct marketing dilemmas – customers' attitudes and purchase intention. *International Journal of Bank Marketing*, 21(3):147–163, 2003.

[4] Dan Pelleg and Andrew Moore. Active learning for anomaly and rare-category detection. *Advances in Neural Information Processing Systems*, 18(2):1073–1080, 2004.

[5] Hao Huang, Kevin Chiew, Yunjun Gao, Qinming He, and Qing Li. Rare category exploration. *Expert Systems with Applications*, 41(9):4197–4210, 2014.

[6] Ying Li, Pavankumar Murali, Nan Shao, and Anshul Sheopuri. Applying Data Mining Techniques to Direct Marketing: Challenges and Solutions. *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 319–327, 2015.

[7] Sérgio Moro and Raul M S Laureano. Using Data Mining for Bank Direct Marketing: An application of the CRISP-DM methodology. *European Simulation and Modelling Conference*, (Figure 1):117–121, 2011.

[8] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-Armed Bandit Allocation Indices: 2nd Edition*. John Wiley and Sons, 2011.

[9] Joannès Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3720 LNAI, pages 437–448, 2005.

[10] Burr Settles. Active Learning Literature Survey. *Machine Learning*, 15:201–221, 2010.

[11] Bob Stone and Ron Jacobs. *Successful direct marketing methods*. McGraw Hill Professional, 2008.

[12] Colin Shearer, Hugh J Watson, Daryl G Grecich, Larissa Moss, Sid Adelman, Katherine Hammer, and Stacey a Herdlein. The CRIS-DM model: The New Blueprint for Data Mining. *Journal of Data Warehousing14*, 5(4):13–22, 2000.

[13] Sergio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

[14] Josh Attenberg and Foster Provost. Why label when you can search?: alternatives to active learning for applying human resources to build classification models under extreme class imbalance. *Proceedings of the 16th ACM SIGKDD ...*, pages 423–432, 2010.

[15] Gary M Weiss. Mining with Rarity: A Unifying Framework. *SIGKDD Explorations*, 6(1):7–19, 2004.

[16] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE : Synthetic Minority Over-sampling Technique. 16:321–357, 2002.

[17] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory Undersampling for Class Imbalance Learning. *IEEE Transactions on Systems, Man and Cybernetics*, 39(2):539–550, 2009.

[18] Pedro Domingos. MetaCost: A General Method for Making Classifiers Cost-Sensitive. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 55:155–164, 1999.

[19] Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the Border : Active Learning in Imbalanced Data Classification. *Proceedings of the sixteenth ACM conference on Conference on information and knowledg*, pages 127–136, 2007.

[20] Michael Bloodgood and K. Vijay-Shanker. A Method for Stopping Active Learning Based on Stabilizing Predictions and the Need for User-Adjustable Stopping. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, (June):39–47, 2009.

[21] Richard Weber. On the Gittins Index for Multiarmed Bandits. *The Annals of Applied Probability*, 2(4):1024–1033, 1992.

[22] Josh Attenberg and Foster Provost. Inactive learning?: difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter*, 12(2):36–41, 2011.

[23] David D. Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. *Proceedings of the 11th international conference on machine learning (ICML'94)*, pages 148–156, 1994.

[24] Simon Tong and Daphne Koller. Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research*, pages 45–66, 2001.

[25] H Zou and T Hastie. Regularization and variable selection via the elastic-net. *Journal of the Royal Statistical Society*, 67:301–320, 2005.

[26] Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and {ROC} Curves. *International Conference on Machine Learning {(ICML)}*, 2006.