Data Acquisition for Improving Machine Learning Models

Yifan Li
York University
yifanli@eecs.yorku.ca

Xiaohui Yu
York University
xhyu@yorku.ca

Nick Koudas
University of Toronto
koudas@cs.toronto.edu

ABSTRACT

The vast advances in Machine Learning (ML) over the last ten years have been powered by the availability of suitably prepared data for training purposes. The future of ML-enabled enterprise hinges on data. As such, there is already a vibrant market offering data annotation services to tailor sophisticated ML models.

In this paper, inspired by the recent vision of online data markets and associated market designs, we present research on the practical problem of obtaining data in order to improve the accuracy of ML models. We consider an environment in which consumers query for data to enhance the accuracy of their models and data providers who possess data make them available for training purposes. We first formalize this interaction process laying out the suitable framework and associated parameters for data exchange. We then propose two data acquisition strategies that consider a trade-off between *exploration* during which we obtain data to learn about the distribution of a provider's data and *exploitation* during which we optimize our data inquiries utilizing the gained knowledge. In the first strategy, *Estimation and Allocation* (EA), we utilize queries to estimate the utilities of various predicates while learning about the distribution of the provider's data; then we proceed to the allocation stage in which we utilize those learned utility estimates to inform our data acquisition decisions. The second algorithmic proposal, named *Sequential Predicate Selection* (SPS), utilizes a sampling strategy to explore the distribution of the provider's data, adaptively investing more resources to parts of the data space that are statistically more promising to improve overall model accuracy.

We present a detailed experimental evaluation of our proposals utilizing a variety of ML models and associated real data sets exploring all applicable parameters of interest. Our results demonstrate the relative benefits of the proposed algorithms. Depending on the models trained and the associated learning tasks we identify trade-offs and highlight the relative benefits of each algorithm to further optimize model accuracy.

PVLDB Reference Format:
Yifan Li, Xiaohui Yu, and Nick Koudas. Data Acquisition for Improving Machine Learning Models. PVLDB, 14(10): 1832 - 1844, 2021.
doi:10.14778/3467861.3467872

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/AwesomeYifan/Data-acquisition-for-ML.

Proceedings of the VLDB Endowment, Vol. 14, No. 10 ISSN 2150-8097.
doi:10.14778/3467861.3467872

1 INTRODUCTION

Background and Motivation. Data traditionally has been an asset in deriving projections or making decisions. The prevalence of Machine Learning (ML) models across business functions necessitates access to ample and diverse data sources for training. As an answer to the vast demand for training data, numerous businesses offer data annotation services [1, 2, 4] providing annotated data in a myriad of business categories with varying degrees of specialization. It is evident that the need for specialized data to train ML models has created a corresponding market fulfilling the purpose.

At the same time, in recent years we have experienced the increasing prevalence of *online data markets* such as Dawex [3], WorldQuant [6], and Xignite [7], to name a few, which aim to make access to data a commodity, for modelling or learning purposes. In these markets the main idea is to facilitate interaction between *data providers* (e.g., individuals or organizations that possess data in diverse domains and wish to offer them to other interested parties) and *data consumers* who are interested to obtain data to accomplish certain tasks, such as training new ML models or increasing the accuracy of existing ones, or conducting statistical estimation. Since such platforms aim to adopt the characteristics of a market, data exchange carries an underlying cost (e.g., monetary value). The emergence of such markets can be viewed as an initial step to the enablement of efficient trading of data.

The design of the operating principles, market mechanisms and tradings strategies (to name a few topics) of such markets constitute open research directions and involve multiple research communities. Recently, in the database community, Fernandez et al. [21] presented their vision for a research agenda on market design in data market platforms and discussed various important research directions of broader data management interest in making the data market vision a reality.

The Problem. In this paper, we consider a domain $\Gamma = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ denotes the feature space and $\mathcal{Y}$ denotes the label space (e.g., possible class labels for classification tasks, possible values of the dependent variable for regression tasks). The purpose is to train a model for a target distribution p over Γ such that the model attains high accuracy on data drawn from the same distribution. We focus on supervised learning and assume that the data consumer (*consumer* for short) already has an ML model (e.g., CNN, SVM, a regression model) built utilizing some training data from Γ, and wishes to obtain data from a data provider (*provider*) offering $\mathcal{D}_{pool}$ drawn from the target distribution p. The aim of the consumer is to maximize the improvement in the accuracy of the model[1].

To facilitate the interaction, the provider exposes meta-data of $\mathcal{D}_{pool}$, such as the range of values in each attribute, and the set of possible labels on the records. We assume a typical query interface

[1]We use the term "accuracy" in a broad sense here, which depending on the ML model can be measured in different ways (precision for classifiers, root mean squared error for regression models, etc.), and our discussion is independent of its exact choice.

supported by both parties, akin to the prevalent application programming interfaces (API) in existence for any online service [5]. The interface supports a predicate P specifying the properties of the records requested, and an integer I denoting the number of records to obtain (e.g., 10 images with `label = 'dog'`, or 100 records with `2018≤year≤2020`). Such predicates impose multi-attribute conjunctive conditions in the more general case. After receiving the query request from the consumer, the provider randomly selects without replacement I records satisfying P from $\mathcal{D}_{pool}$ and returns these records to the consumer.

We assume that the consumer carries a *budget* B on the total number of records that can be requested[2]. The number of records requested by the consumer each time a query is issued to the provider may vary, and is decided by the consumer, as long as the total number of records requested across all queries is within the budget B. For example, if the consumer can obtain 10 images from the provider, these can be obtained by inquiring for 10 images once or for 5 images twice.

Suppose that the accuracy of the underlying model is evaluated on a testing data set $\mathcal{D}_{test} \subset \Gamma$. The task of the consumer is to identify a series of queries $\langle (P_1, I_1), (P_2, I_2), \cdots, (P_z, I_z) \rangle$ to obtain B records, where $\sum_{i=1}^{z} I_i = B$ with P_i and I_i being respectively the predicate and the number of requested records in the i-th query. Let $\mathcal{D}_{otd}$ be the records obtained from the provider using the identified queries ($|\mathcal{D}_{otd}| = B$), $\mathcal{D}_{init}$ be the data the model of the consumer is trained on initially, and $\mathcal{M}'$ be the model re-trained on all the data the consumer has after data acquisition (i.e., $\mathcal{D}_{otd} \cup \mathcal{D}_{init}$). The objective of the data acquisition process is to improve as much as possible the accuracy of $\mathcal{M}'$ on $\mathcal{D}_{test}$.

Proposed Solutions. We develop data acquisition strategies to address this problem. In particular, we consider the trade-off between *exploration* and *exploitation* in data acquisition. During exploration, requested data records from the available budget are obtained to gain more knowledge regarding the distribution the provider's data, so that better predicates can be designed for subsequent queries. During exploitation, data records are obtained based on the current information the consumer possesses. With a limited budget of records to be requested, one must strike a balance between exploration and exploitation by allocating the requested records within the budget wisely such that the accuracy of the resulting model is maximized.

We propose two methods to determine how to allocate the existing budget of records across queries (the budget allocation problem) adopting different strategies. The first solution, which we refer to as *estimation-and-allocation* (EA), consists of two stages: during the Estimation Stage, the consumer issues a number of queries obtaining a number of random records for each of them to explore from $\mathcal{D}_{pool}$; we subsequently estimate (without re-training the model) the expected improvement in model accuracy utilizing the records for each query, which we refer to as *predicate utility*. During the Allocation Stage, the consumer allocates the remaining record budget according to the estimated utilities. We investigate methods to quantify the estimation error and propose an adaptive method to balance between reducing the estimation error and controlling how much of the record budget is devoted to obtaining the estimation; as a result, budget is reserved to be allocated more effectively for predicates with high utilities. For the Allocation Stage, we propose different allocation strategies and showcase their performance under various settings in Section 6.

The second solution, which we refer to as *sequential predicate selection* (SPS), is based on the observation that for a predicate P, the associated predicate utility decreases as we obtain more records for the predicate, due to information redundancy [23]. The core idea of SPS is to iteratively pose queries requesting a small number of records while balancing between (1) obtaining more records with predicates yielding higher expected utility, and (2) closely monitoring the utility decrease of each predicate as we obtain more records. We implement this design utilizing Thompson Sampling (TS) [40, 45], an action selection method, for its simplicity and proven performance, but the design can be implemented with other action selection methods (such as the ϵ-greedy algorithm [43]) as well.

As both EA and SPS rely on the expected predicate utility, we investigate how to best estimate it without re-training the underlying ML model. The utility of a predicate P is essentially measured by the improvement in model accuracy resulting from the set of records selected by P, $\mathcal{R}_P$. We propose *novelty*, which describes how different the distribution of $\mathcal{R}_P$ is from the distribution of the records the consumer currently possesses satisfying P, as the indicator of the potential accuracy gain $\mathcal{R}_P$ brings to the model. Note that our subsequent discussion applies to other utility measures as well; we experimentally compare various measures in Section 6.11.

We evaluate the performance of EA and SPS on both traditional ML tasks and Deep Learning tasks, including spatial regression, radar data classification, and image classification, using classical ML models as well as state-of-the-art deep models. As will be shown in Section 6, the proposed methods demonstrate solid performance across a variety of settings, outperforming alternative approaches that require frequent model re-training. We also thoroughly study the effects of various parameters on EA and SPS and provide suggestions on their settings in real-world scenarios.

Contributions. Our main contributions can be summarized as follows.

- We formally define and study the problem of data acquisition for improving the performance of ML models given a budget. We consider this problem in the context of a data consumer and a data provider in a data market, and it can serve as a building block for a variety of data markets.
- We propose an estimation-and-allocation solution, EA, which first estimates the utility of each predicate with a portion of the budget, and then allocates the budget accordingly to improve the accuracy of the model.
- We devise a sequential predicate selection solution, SPS, which adaptively conducts exploration and exploitation, by iteratively requesting a small number of records in each query, aiming to improve the predicate utility estimates and utilize such estimates at the same time.
- We design methods to estimate the expected utility of a given predicate, allowing EA and SPS to proceed without necessitating the re-training of the underlying model.

[2]Such a budget can be determined based on monetary costs per record offered by the provider or by the monetary cost of each query, etc. Any mechanism to assign a value to data (e.g., price or otherwise) is completely orthogonal to our approach.

- We experimentally study the proposed solutions across a variety of settings, including but not limited to, different tasks, ML models, datasets, distributions, budget limitations, utility measures. We showcase that each solution has certain benefits and can be suitably adopted when applied to real-world settings.

The rest of this paper is organized as follows. In Section 2, we review related work. Section 3 introduces terminology and background. In Section 4, we propose the Estimation-and-Allocation method, followed by Section 5 that presents the Sequential Predicate Selection approach. The experimental evaluation is presented in Section 6. Section 7 concludes this paper.

2 RELATED WORK

Data Markets. Recently Fernandez et al. [21] presented their vision for the design of platforms to support data markets. Our work is aligned with such a vision addressing a specific problem in this setting. Data markets have been an active research topic in various communities. For example, several works adopt a game-theoretic approach of market design for such markets [8, 33, 37], exploring issues such as fairness and pricing.

Data Pricing. A lot of research has been devoted to the design of pricing mechanisms for data. A recent survey [39] presents an overview of works related to data pricing from a data science perspective. Other works [11–13, 32], present pricing-specific problems for queries and models. Pricing mechanisms for data are an orthogonal problem to our work. Any applicable pricing mechanism can be adopted to determine the number of records one can obtain based on applicable monetary budgets.

Data Acquisition. Another area related to data markets is the acquisition of data. Chen et al. [14] consider the problem of obtaining data from multiple data providers in order to improve the accuracy of linear statistical estimators. The focus is strictly on linear statistical techniques and they provide certain types of guarantees on the best strategies to adopt when providers decide to abstain from making their data available and thus data costs vary dynamically. In a related thread Kong et al. [28] study the problem of estimating Gaussian parameters and other estimators with Gaussian noise. Zhang et al. [48] study incentive mechanisms for participants in data markets to refresh their data.

Data Augmentation. Data augmentation and feature selection for machine learning have also attracted research interests recently [15, 31, 41]. Kumar et al. [31] focus on ML model training on relational data and propose methods to decide whether joining certain attributes would improve the model's accuracy. Chepurko et al. [15] design a system that automatically performs feature selections and joins so as to improve the performance of a given model. Their work, however, focuses on the selection and acquisition of (new) features rather than records, and assumes the accessibility of all features, and thus is orthogonal to the problem studied herein.

Active Learning. Active learning is concerned with interactively acquiring labels for new data points to improve the performance of machine learning models [42]. It has been applied to solve various problems in data management [36]. In a typical setting for active learning, we have access to the features of new data and have to decide the set of records for which we would like to acquire the label for a cost. In contrast, we deal with the scenario where the consumer does not have any prior knowledge of the provider's data other than the meta-data. Perhaps what is most related to ours in this area is the work by Lomasky et al. [34], which also aims to control the class of data to acquire for improving the ML models. However, they focus on the task of selecting class proportions for generating new training data. They do not explicitly optimize the use of a fixed budget, do not tackle regression problems, and require frequent re-training of the model.

Exploration vs. Exploitation. The trade-off between exploration and exploitation exists in many scenarios where a decision has to be made with incomplete knowledge. Although methods balancing this trade-off have been proposed in various contexts, such as reinforcement learning [43] and online decision making [24], they are not directly applicable to our problem setting. Nonetheless, we share a similar methodology and adopt one of the popular approaches for performing this trade-off, Thompson Sampling [40], as the framework to build the proposed SPS solution.

3 PRELIMINARIES

In this section, we formally define the terminology utilized and the problems we focus on in this paper.

Data Domain and Learning Task. We consider a supervised learning task defined on a data domain $\Gamma = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ denotes the *feature* space and $\mathcal{Y}$ denotes the *label* space (e.g., all possible class labels for classification tasks, all possible values of the dependent variable for regression tasks). Suppose there is a conditional distribution $p(y|x)$ defined over Γ, where $x \in \mathcal{X}$, $y \in \mathcal{Y}$. The learning task is to train a model $\mathcal{M}$ on a dataset that represents a distribution q that is as close as possible to the target distribution p. The accuracy of $\mathcal{M}$ is evaluated on a testing dataset $\mathcal{D}_{test} \subset \Gamma$. In accordance to any well-formed learning task, we assume that both the training data and testing data come from the same distribution p. The model $\mathcal{M}$ is evaluated using a function F based on $\mathcal{D}_{test}$, denoted as $F(\mathcal{M}; \mathcal{D}_{test})$.

Provider, Consumer, and Budget. The provider maintains a collection of data records $\mathcal{D}_{pool} \subset \Gamma$ drawn from the target distribution p, which are provided in the data market and are initially entirely invisible to the consumer. The consumer has an ML model $\mathcal{M}$ trained on data $\mathcal{D}_{init} \subset \Gamma$ drawn from p. Note that although both $\mathcal{D}_{pool}$ and $\mathcal{D}_{init}$ follow the same distribution p, they are not necessarily representative samples of p. For example, $\mathcal{D}_{pool}$ may contain a high percentage of records from one part of the data domain Γ, while $\mathcal{D}_{init}$ from another. In the degenerate case, $\mathcal{M}$ is simply a model randomly initialized without using any training data, i.e., $\mathcal{D}_{init} = \emptyset$. The consumer has a budget B, which is the maximum number of records that the consumer can obtain from the provider.

Query and Predicate. A query $Q = (P, I)$ consists of a predicate P that specifies the properties of the records the consumer would like to acquire, and an integer I that specifies the number of records requested from the provider. Let $\mathcal{R}_P \subset \mathcal{D}_{pool}$ denote the set of records that satisfy P and $\mathcal{R}_Q$ denote the set of records returned by the provider. All possible predicates admissible to the provider constitute set $\mathcal{P}$, which can be formed in various ways depending on the task at hand. In the paper we adopt a simple

yet intuitive predicate construction strategy: for a classification problem, $\mathcal{P}$ contains all predicates with a selection on the class label (e.g., `label = 'dog'`); for a regression problem, we discretize each attribute into equal-width sub-ranges, and all combinations of sub-ranges, denoted by "cells", constitute $\mathcal{P}$ (e.g., `1000≤ salary ≤ 2000 ∧ 20≤ age≤ 30`). There are many other strategies to construct and refine $\mathcal{P}$. For example, the consumer may perform cross validation on $\mathcal{D}_{init}$ and use the m labels in which $\mathcal{M}$ has the lowest accuracy to construct $\mathcal{P}$ for more targeted acquisition. One may also inject domain knowledge to the predicate construction process and only use these classes or cells related to the task as predicates. For example, a consumer training a cat/dog classifier may not consider predicate `label = 'horse'`. Refer to Section 6.1 and Section 6.7 for more details on the methodology to construct $\mathcal{P}$ adopted in this work and the associated evaluation. We note that the investigation of predicate construction strategies and their properties is an interesting direction for future work. Our emphasis is to develop methods for data acquisition that are independent of the predicate construction process.

Interaction. Each round of interaction between the consumer and the provider consists of two steps: (1) the consumer issues a query $Q = (P, I)$ to the provider, and (2) the provider returns a set of records $\mathcal{R}_Q$, where $|\mathcal{R}_Q| = I$ and each $r \in \mathcal{R}_Q$ is randomly sampled from $\mathcal{R}_P$ [3]. Without loss of generality, we assume that all records provided to the consumer are unique within the same and across different rounds of interactions. A predicate P can be reused across different rounds of interactions as long as $\mathcal{R}_P$ has not been exhausted, i.e., there are records in $\mathcal{R}_P$ that have not been acquired by the consumer yet.

Predicate utility. After each round of interaction, the consumer estimates the utility of the predicate used in the query, which is useful for planning the next round of interaction. The utility of a predicate P expresses the anticipated accuracy improvement that $\mathcal{R}_P$ brings to $\mathcal{M}$. We define a measure that we call *novelty* to quantify predicate utility. The basic idea of this measure is to quantify the difference between the data acquired in the interaction to those that the consumer currently possesses. The higher the difference, the more information this interaction brings to the consumer. Let $\mathcal{R}_{\mathcal{M}:P}$ be the records the consumer currently possesses satisfying P. The novelty of predicate P, denoted as U_P, is defined on $\mathcal{R}_{\mathcal{M}:P}$ and $\mathcal{R}_P$. More specifically, we consider a binary classification problem that treats $\mathcal{R}_{\mathcal{M}:P}$ and $\mathcal{R}_P$ as samples from class 0 and class 1 respectively, and train a classifier CLF to distinguish between the two sets of records. The utility of P is computed as follows:

$$U_P = \frac{\sum_{(x,y)\in\mathcal{R}_P} \mathbb{I}[\text{CLF}((x,y)) = 1]}{|\mathcal{R}_P|} \tag{1}$$

where $\mathbb{I}[*]$ is the indicator function that takes value 1 if statement $*$ is true and 0 otherwise, and $\text{CLF}((x,y))$ denotes the prediction for record (x,y) made by CLF. In principle, any classifier may be used as the CLF. However, in practice it is preferred to use light-weight models for faster training and inference as the computation of novelty is carried out frequently. We study the influence of different classifiers in Section 6.9.

[3]Note that if I is larger than the number of the provider's remaining records (say I_P), all of the provider's records will be returned and only I_P will be deducted from the consumer's budget.

The intuition for the design of novelty is that, if $\mathcal{R}_P$ is drawn from a distribution that is very different than the one $\mathcal{R}_{\mathcal{M}:P}$ is drawn from, then the two sets of records can be easily differentiated and U_P is high, and vice versa. Note that we only evaluate the accuracy of the classifier on $\mathcal{R}_P$, because novelty measures how different $\mathcal{R}_P$ is, given $\mathcal{R}_{\mathcal{M}:P}$, rather than the other way around. Such methods for quantifying the difference between two distributions are well adopted in the ML literature [35]. In practice, it may not be feasible for the consumer to obtain all the records in $\mathcal{R}_P$ due to budget limitations. As such, in the proposed solutions, we utilize queries based on the same predicate P returning $|\mathcal{R}_Q| \ll |\mathcal{R}_P|$ records, to estimate U_P.

Acquisition plan. The acquisition plan of the consumer consists of a sequence of interactions, $\langle (P_1, I_1), (P_2, I_2), \cdots, (P_z, I_z)\rangle$, where $\forall i \in [1, z]$, $P_i \in \mathcal{P}$ and $\sum_{i=1}^{z} I_i = B$. The consumer receives B records in total after executing the acquisition plan, denoted as $\mathcal{D}_{otd}$.

The problem of data acquisition for model improvement is defined as follows.

Definition 3.1. **Data Acquisition for Model Improvement**. Given (1) a set of records, $\mathcal{D}_{pool}$ from a provider, (2) an initial set of records, $\mathcal{D}_{init}$, possessed by a consumer, (3) the set of possible predicates, $\mathcal{P}$, (4) the initial model, $\mathcal{M}$, of the consumer, (5) a measure to evaluate model accuracy, F, and (6) the budget, B, the objective of data acquisition for model improvement is to construct an acquisition plan to maximize $F(M'; \mathcal{D}_{test})$, where M' denotes the consumer model $\mathcal{M}$ after being re-trained on $\mathcal{D}_{init} \cup \mathcal{D}_{otd}$.

4 AN ESTIMATION-AND-ALLOCATION SOLUTION

In this section, we introduce *estimation-and-allocation* (EA), a two-stage solution, to generate the acquisition plan. Essentially, stage one of EA is designed to explore, i.e., to gather more information on how useful each predicate is; while stage two is to exploit, i.e., to utilize the knowledge gained in stage one to optimize subsequent actions. Specifically, the first stage, called the *Estimation Stage*, aims to obtain accurate estimates on the utilities of predicates in $\mathcal{P}$; this is achieved via querying the provider requesting a number of records that constitute a small portion of the budget. Then in the second stage, called the *Allocation Stage*, the consumer allocates the remaining budget and issues queries to the provider based on the estimated predicate utilities. We first discuss in Section 4.1 how to ensure the quality of the utility estimates in the Estimation Stage, and present a method that could balance between quality and budget consumption (the amount of record budget spent) in Section 4.2. We then elaborate on the Allocation Stage in Section 4.3.

4.1 Estimating Predicate Utility

We now discuss how to estimate the predicate utilities. Recall that the utility U_P of a predicate P is defined as the accuracy of the classifier (denoted by CLF) in differentiating $\mathcal{R}_P$ from $\mathcal{R}_{\mathcal{M}:P}$. Since it is not possible to obtain the entire $\mathcal{R}_P$, we rely on queries using predicate P to effectively sample from it. We can estimate U_P based on the records already acquired with P, say $\hat{\mathcal{R}}_P$, and we use $\hat{U}_P$ to denote the estimated value of U_P.

The effectiveness of EA depends largely on the accuracy of the predicate utility estimates; thus we investigate how to statistically bound the estimation error, i.e., the difference between U_P and $\hat{U}_P$. To this end, we aim to find the ϵ-δ approximations of all predicate utilities, defined as follows.

Definition 4.1. ϵ-δ **Approximation**. $\hat{U}_P$ is said to be an ϵ-δ approximation of U_P if $\Pr(|U_P - \hat{U}_P| \geq \epsilon) \leq \delta$, where ϵ denotes the error bound and δ denotes the significance level.

In order to determine whether $\hat{U}_P$ is an ϵ-δ approximation of U_P, we conduct a statistical test with the following null hypothesis:

$$H_0^{(P)} : |U_P - \hat{U}_P| \geq \epsilon \tag{2}$$

where $P \in \mathcal{P}$ is an arbitrary predicate. We reject $H_0^{(P)}$ at significance level δ when the following condition is met:

$$\text{reject } H_0^{(P)} \text{ if } \Pr(|U_P - \hat{U}_P| \geq \epsilon) \leq \delta \tag{3}$$

The rejection condition bounds the probability of type I error (false rejection) of the statistical test by δ, and if $H_0^{(P)}$ can be rejected, clearly $\hat{U}_P$ is an ϵ-δ approximation of U_P.

We next discuss how to compute $\Pr(|U_P - \hat{U}_P| \geq \epsilon)$. Note that there are two sources of error in estimating U_P: (1) approximating U_P with a subset $\hat{\mathcal{R}}_P \subset \mathcal{R}_P$, and (2) the error incurred by the CLF. In our work, we focus on the error caused by insufficient records (i.e., source (1)), and we bound the model error (i.e., source (2)) following [19]. Nonetheless, both types of error can be reduced by increasing the size of $\hat{\mathcal{R}}_P$ [19]. To bound the error attributed to insufficient records, we present the following result on the distribution of $U_P - \hat{U}_P$.

Theorem 4.2. *$U_P - \hat{U}_P \sim \mathcal{N}(0, \frac{U_P(1-U_P)}{|\hat{\mathcal{R}}_P|})$, where $\mathcal{N}(0, \frac{U_P(1-U_P)}{|\hat{\mathcal{R}}_P|})$ denotes the normal distribution with mean 0 and variance $\frac{U_P(1-U_P)}{|\hat{\mathcal{R}}_P|}$.*

Proof. Since U_P is the accuracy of the binary classifier CLF in discriminating $\mathcal{R}_P$ from $\mathcal{R}_{\mathcal{M}:P}$, for each record $(x_i, y_i) \in \mathcal{R}_P$, either $\text{CLF}((x_i, y_i))= 0$ or $\text{CLF}((x_i, y_i))= 1$, corresponding to the case when (x_i, y_i) is regarded by CLF to be from $\mathcal{R}_P$ or $\mathcal{R}_{\mathcal{M}:P}$, respectively. Also, if we let $r_i = \mathbb{I}[\text{CLF}((x_i, y_i)) = 0]$, r_i can be viewed as an independent Bernoulli(p) variable, with p being the probability of $r_i = 1$. Evidently, in this case, $p = U_P$.

As $\hat{U}_P = \frac{1}{|\hat{\mathcal{R}}_P|} \sum_{(x_i,y_i)\in\hat{\mathcal{R}}_P} r_i$ and $r_i \sim \text{Bernoulli}(U_P)$, we know $\hat{U}_P * |\hat{\mathcal{R}}_P| \sim \text{Binomial}(|\hat{\mathcal{R}}_P|, U_P)$. According to the Central Limit Theorem, we have $U_P - \hat{U}_P \sim \mathcal{N}(0, \frac{U_P(1-U_P)}{|\hat{\mathcal{R}}_P|})$. □

Since $U_P(1 - U_P)$ is not known, we cannot directly estimate the difference between U_P and $\hat{U}_P$ based on Theorem 4.2. Following the standard practice [10] in estimating population mean (U_P in our case), we introduce statistic t_P as follows:

$$t_P = \frac{U_P - \hat{U}_P}{S_P/\sqrt{|\hat{\mathcal{R}}_P|}} \tag{4}$$

where $S_P = \sqrt{\hat{U}_P(1 - \hat{U}_P)}$ denotes the sample standard deviation. Clearly t_P follows t-distribution with degree of freedom $(|\hat{\mathcal{R}}_P| - 1)$.

Now we can re-write $U_P - \hat{U}_P$ as follows:

$$U_P - \hat{U}_P = t_P * S_P/\sqrt{|\hat{\mathcal{R}}_P|} \tag{5}$$

The probability of $|U_P - \hat{U}_P| \geq \epsilon$ can now be computed as follows:

$$\begin{aligned}
\Pr(|U_P - \hat{U}_P| \geq \epsilon) &= \Pr\left(|t_P * S_P/\sqrt{|\hat{\mathcal{R}}_P|}| \geq \epsilon\right) \\
&= \Pr\left(|t_P| \geq \epsilon\sqrt{|\hat{\mathcal{R}}_P|}/S_P\right) \\
&\leq \int_{-\infty}^{-\frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}} f_{n_P}(t_P)dt_P + \int_{\frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}}^{\infty} f_{n_P}(t_P)dt_P \\
&= Z_P
\end{aligned} \tag{6}$$

where $n_P = |\hat{\mathcal{R}}_P| - 1$, and f_{n_P} is the probability density function of the t-distribution with degree of freedom n_P.

Therefore, $\Pr(|U_P - \hat{U}_P| \geq \epsilon) = Z_P$, and $H_0^{(P)}$ can be rejected if $Z_P \leq \delta$. Evidently Z_P is negatively correlated to $|\hat{\mathcal{R}}_P|$, and thus if $H_0^{(P)}$ cannot be rejected, one can obtain more records utilizing predicate P and issuing additional queries to reduce the value of Z_P until $Z_P \leq \delta$. The intuition behind this process is that, the more records the consumer obtains utilizing P, the more accurate the estimate $\hat{U}_P$ is, and the smaller $|U_P - \hat{U}_P|$ would be. When the null hypotheses for all predicates in $\mathcal{P}$ can be rejected, the current predicate utility estimates are ϵ-δ approximations.

4.2 Budget-Aware Utility Estimation

The Estimation Stage has to consider two conflicting goals: providing more accurate estimation of predicate utilities and controlling the budget consumption so that there is more budget left to spend in the Allocation Stage. We thus introduce a budget-aware estimation method, which first acquires a small number of records using each predicate in $\mathcal{P}$ and computes utility estimates accordingly, and then iteratively determines for each predicate whether obtaining more records to improve the estimation accuracy is worthwhile based on a measure called *heuristic reward*. Such a measure is designed to strike a balance between estimation accuracy and budget consumption.

Since for a given significance level δ, the estimation accuracy is contingent on ϵ, we adaptively choose and adjust its value in order to yield estimates with different levels of accuracy. Intuitively, to achieve higher estimation accuracy, i.e., smaller ϵ, the consumer needs to acquire more records for estimation. Assume that the consumer has acquired records $\hat{\mathcal{R}}_P^0$ for each predicate $P \in \mathcal{P}$. We use $B' = B - \sum_{P\in\mathcal{P}} |\hat{\mathcal{R}}_P^0|$ to denote the remaining budget of the consumer. Let ϵ_0 be the minimal ϵ that causes the null hypotheses for all predicates in $\mathcal{P}$ to be rejected (note that such ϵ_0 always exists, with the extreme case being $\epsilon_0 = 1$). The heuristic reward is defined as $B' \cdot (1 - \epsilon_0)$, which is larger if (1) B' is large, meaning the consumer has more available budget, and (2) ϵ_0 is small, meaning that the estimations are accurate.

Example 4.3. Consider a consumer with budget=500 and there are 5 predicates to choose from. Assume that the consumer has acquired 5 records for each predicate, and the resulting sample standard deviations are $[0.1, 0.11, 0.12, 0.13, 0.14]$ respectively. Let the significance level δ be 0.01. Using Equation (6), we know the minimal values of ϵ causing all $H_0^{(P)}$ ($P \in \mathcal{P}$) to be rejected are

$[0.21, 0.23, 0.25, 0.27, 0.29]$, and thus the ϵ that causes all null hypotheses to be rejected, or ϵ_0, is 0.29. Since the remaining budget is 475, the heuristic reward is thus $475 \cdot (1 - 0.29) = 337.25$.

Now assume the consumer aims to determine whether reducing ϵ_0 to ϵ_b, by acquiring more records, would improve the heuristic reward. In order to compute the heuristic reward corresponding to ϵ_b, we need to estimate how many additional records need to be acquired, ΔB_b, to reach ϵ_b. We next show how to estimate the value of ΔB_b.

Recall from Equation (6) that the reject condition of $H_0^{(P)}$ is $Z_P \leq \delta$. Since $|\hat{\mathcal{R}}_P|$ is negatively correlated to Z_P, in order to get the minimal number of records to acquire to reach a given ϵ, we use the maximal Z_P, i.e., $Z_P = \delta$, and rewrite Equation (6) as follows:

$$Z_P = \delta \Leftrightarrow \int_{-\infty}^{-\frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}} f_{n_P}(t_P)dt_P + \int_{\frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}}^{\infty} f_{n_P}(t_P)dt_P = \delta$$

$$\Leftrightarrow \int_{-\infty}^{-\frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}} f_{n_P}(t_P)dt_P = \frac{\delta}{2}$$

Let $A_P = \frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}$, we can further rewrite the equation above as follows:

$$\int_{-\infty}^{-A_P} f_{n_P}(t_P)dt_P = \frac{\delta}{2} \Leftrightarrow A_P = -\text{PCT}_{n_P}(\frac{\delta}{2}) \tag{7}$$

where PCT_{n_P} denotes the percentile function of t-distribution with degree of freedom n_P.

Now having a way to determine the value of A_P using Equation (7), we rewrite $A_P = \frac{\epsilon\sqrt{|\hat{\mathcal{R}}_P|}}{S_P}$ as follows:

$$|\hat{\mathcal{R}}_P| = \left(\frac{A_P \cdot S_P}{\epsilon}\right)^2 \tag{8}$$

Equation (8) establishes the relation between the number of records currently obtained and the error bound ϵ that can be obtained using those records.

Let $\hat{\mathcal{R}}_P^b$ be the records with which we can reach error bound ϵ_b, we have:

$$|\hat{\mathcal{R}}_P^b| = \left(\frac{A_P^b \cdot S_P^b}{\epsilon_b}\right)^2 \tag{9}$$

where S_P^b denotes the sample standard deviation of $\hat{\mathcal{R}}_P^b$, and $A_P^b = -\text{PCT}_{n_P^b}(\frac{\delta}{2})$, $n_P^b = |\hat{\mathcal{R}}_P^b| - 1$. Notice that S_P asymptotically converges to the population standard deviation, and A_P^b, which is determined by PCT_{n_P}, asymptotically converges to the opposite value of the $\frac{\delta}{2}$-percentile of standard normal distribution [22]. Thus, although S_P^b and A_P^b are unknown, we choose to approximate their values by S_P^0 and A_P^0, i.e., the values computed based on $\hat{\mathcal{R}}_P^0$, as long as $|\hat{\mathcal{R}}_P^0|$ is reasonably large. We demonstrate the validity of this approximation empirically as well in Section 6.2. Therefore, we use $\left(\frac{A_P^0 \cdot S_P^0}{\epsilon_b}\right)^2$ as the least number of records required to achieve estimation error ϵ_b for predicate P.

Example 4.4. Following Example 4.3, we assume that the consumer plans to reduce the error bound to 0.15 from 0.29. The values of A_P^0 for each P (Equation (7)) are: $[4.68, 4.66, 4.64, 4.63, 4.62]$. Thus the number of records required for each predicate to reach error bound 0.15 are: $[10, 12, 14, 17, 19]$.

Since we already posses $|\hat{\mathcal{R}}_P^0|$ records satisfying P, we need to acquire $\left(\frac{A_P^0 \cdot S_P^0}{\epsilon^b}\right)^2 - |\hat{\mathcal{R}}_P^0|$ more records utilizing predicate P. The total additional records from our budget needed to reach ϵ_b from ϵ_0 is thus

$$\Delta B_b = \sum_{P \in \mathcal{P}} \left[\left(\frac{A_P^0 \cdot S_P^0}{\epsilon_b}\right)^2 - |\hat{\mathcal{R}}_P^0|\right]. \tag{10}$$

The new heuristic reward after obtaining these ΔB_b records can thus be estimated as $(B' - \Delta B_b) \cdot (1 - \epsilon_b)$. Let $\epsilon_* = \arg\max_{\epsilon_b}(B' - \Delta B_b) \cdot (1 - \epsilon_b)$, be the value ϵ_b yielding the maximal heuristic reward. The consumer then compares $B' \cdot (1 - \epsilon_0)$ with $(B' - \Delta B_{\epsilon_*}) \cdot (1 - \epsilon_*)$. If the latter value is higher, meaning that the new combination of the estimation error ϵ_* and remaining budget $(B' - \Delta B_*)$ is better, we initiate a new interaction to obtain ΔB_* more records according to Equation (10). This process continues until the above calculation indicates no more improvement in heuristic reward is possible via further interaction; we then terminate the Estimation Stage and enter the Allocation Stage.

4.3 Budget Allocation

The Allocation Stage of EA is concerned with distributing the remaining budget across all predicates in $\mathcal{P}$ utilizing their estimated utilities. We consider two budget allocation strategies, where M_P denotes the budget (number of records) allocated to a predicate P:

- **Linear Allocation**: $M_P = \frac{B * \hat{U}_P}{\sum_{P' \in \mathcal{P}} \hat{U}'_P} - B_P$, where B_P denotes the number of records obtained with P during the Estimation Stage.
- **Square-root Allocation**: $M_P = \frac{B * \sqrt{\hat{U}_P}}{\sum_{P' \in \mathcal{P}} \sqrt{\hat{U}'_P}} - B_P$.

We sort all predicates in descending order of their utilities and sequentially obtain records based on M_P starting from the predicate with the highest estimated utility. This stage continues iteratively until the budget is exhausted. We demonstrate in Section 6.4 that each allocation strategy has its own winning cases and therefore the allocation strategy can be selected based on the specific task.

4.4 The EA Algorithm

The EA solution is summarized in Algorithm 1. We first use $l\%$ (l is configurable) of the budget to acquire records using each predicate to start the estimation (Line 1). We calculate the current estimation quality ϵ_0 (Line 4) and the estimation quality that is expected to yield the highest heuristic reward ϵ_* (Line 5). The current heuristic reward is compared with the expected highest heuristic reward (Line 7), and if the former is higher, we terminate the Estimation Stage (Lines 7-8) and enter the Allocation Stage (Lines 12-13); otherwise we obtain more records based on ϵ_* (Lines 10-11) and repeat the process. Note that the Estimation Stage also terminates when the budget is exhausted. However although Line 3 provides an exit to the estimation stage, corresponding to the case when all budget is consumed during estimation, such case will never happen as exhausting all budget provides a heuristic reward of 0 and thus is prohibited by Line 7.

Since during the estimation stage more records can be acquired, it is suggested to initialize Algorithm 1 with a small value of l, as initialization with a large value of l may consume too much budget. However, if l is too small (say only 1 record for each predicate), the sample standard deviation S_P computed may accidentally be zero and as a result, Z_P is zero too (Equation (6)). Consequently, $H_o^{(P)}$ can be rejected with any ϵ (even zero) because $Z_P \leq \delta$ is always true, and the estimation stage terminates immediately and abnormally as ϵ cannot be further reduced. With these trade-offs in mind, we experimentally study the influence of the choice of l in Section 6.2.

Algorithm 1 Estimation-and-Allocation

Input: budget B, all predicates $\mathcal{P}$

Initialization: for each $P \in \mathcal{P}$, acquire l% random records in $\mathcal{R}_P$; assume remaining budget is B'.
//Estimation Stage
while $B' > 0$ **do**
  $\epsilon_0 \leftarrow$ minimal ϵ to cause the hypotheses to be rejected;
  $\epsilon_* \leftarrow \arg\max_{\epsilon_b, \epsilon_b < \epsilon_0} (B' - \Delta B_{\epsilon_b}) \cdot (1 - \epsilon_b)$;
  //ΔB_{ϵ_b} is computed with Equation (10)
  if $B' \cdot (1 - \epsilon_0) \geq (B' - \Delta B_{\epsilon_*}) \cdot (1 - \epsilon_*)$ **then**
    break;
  else
    acquire $\Delta B'_{\epsilon_b}$ more records according to Equation (10);
    $B' = B' - \Delta B_{\epsilon_b}$;
//Allocation Stage
Allocate the remaining budget using strategies in Section 4.3;

5 A SEQUENTIAL PREDICATE SELECTION SOLUTION

In this section, we adopt a Bayesian probabilistic approach and introduce an alternative solution called *Sequential Predicate Selection* (SPS). While EA employs two separate stages for exploration and exploitation, SPS utilizes many rounds of interactions, acquiring a small number of records in each interaction with varying predicates, achieving both exploration and exploitation in the same round. In particular, in each round, SPS balances between two objectives: (1) acquiring more records using predicates that are expected to provide higher accuracy improvement to the model, based on previous observations; and (2) exploring to identify other predicates that may bring even higher accuracy improvement to model accuracy. We implement the design utilizing Thompson Sampling (TS), an action selection method with proven performance [24, 40].

5.1 Framework of Thompson Sampling

Thompson Sampling [40] proceeds as follows. Given an action space $\mathcal{A}$, an agent conducts actions $a_1, a_2, \cdots$, each selected from $\mathcal{A}$, in rounds. After applying a_t in round t, the agent observes a reward r_t, which is randomly generated according to a conditional probability measure $q_\theta(\cdot|a_t)$. The agent is initially uncertain about the value of parameters θ and thus uses a prior distribution p to describe θ, which is iteratively updated based on (a_t, r_t) pairs. The target is to maximize the cumulative rewards over a given number of rounds. Given $\mathcal{A}$, p, and q, TS repeats the following steps for action selection:

(1) Sample $\hat{\theta} \sim p$, i.e., sample the parameters controlling the reward according to p (p is called the prior distribution of θ in this round).
(2) Let $a_t = \arg\max_{a \in \mathcal{A}} \mathbb{E}_{q_{\hat{\theta}}}[r_t | a_t = a]$, i.e., a_t is the action with the maximum expected reward under parameter values $\hat{\theta}$. Perform action a_t and observe r_t.
(3) Update $p = \mathbb{P}_{p,q}(\theta \in \cdot | a_t, r_t)$, i.e., p becomes the posterior distribution of θ given (a_t, r_t).

In the following, we apply TS to the problem of predicate selection, discuss the choices of p and q for the problem, and develop the update method of p from prior distribution to posterior distribution. In addition, as will be shown in Section 5.3, the rewards of predicates evolve over time in our problem, and thus we also design methods to handle changes in θ.

5.2 Sequential Predicate Selection Using Thompson Sampling

In SPS, we repeatedly issue queries to the provider in multiple rounds of interactions, until the budget B is exhausted. In each round, a query $Q = (P, I)$ is issued and we calculate a value called *query reward* for the records received, deciding on the next query to generate based on the knowledge acquired so far. The objective of the consumer is to maximize the cumulative reward for the queries issued in all rounds of interactions. In what follows, we detail the set of queries that the consumer can issue, how query reward is evaluated, and how knowledge regarding the problem space (queries and the resulting rewards) is updated.

The number of possible queries the consumer may ask is $|\mathcal{P}| \cdot B$, as each pair of $P \in \mathcal{P}$ and $I \in [1, B]$ can form a query. We assume that a fixed I, denoted by I_Δ, is used for each query. Thus, choosing a query boils down to selecting a predicate in $\mathcal{P}$, and in the sequel we use the term *predicate reward* of P and the query reward of $Q = (P, I_\Delta)$ interchangeably. We empirically study the influence of I_Δ in Section 6.5.

Let $\mathcal{R}_P^{I_\Delta}$ be the records returned by query $Q = (P, I_\Delta)$. The computation of query reward follows the spirit of predicate utility, i.e., novelty introduced in Section 3, except for that instead of using the CLF to differentiate $\mathcal{R}_P$ from $\mathcal{R}_{\mathcal{M}:P}$, the query reward differentiates $\mathcal{R}_P^{I_\Delta}$ from $\mathcal{R}_{\mathcal{M}:P}$, directly measuring the anticipated accuracy improvement $\mathcal{R}_P^{I_\Delta}$ brings to model $\mathcal{M}$. More specifically, the reward of Q is the number of records in $\mathcal{R}_P^{I_\Delta}$ that can be correctly labelled by CLF, following the developments in Section 3. Since records in $\mathcal{R}_P^{I_\Delta}$ are randomly selected from $\mathcal{R}_P$, the reward r of P can be viewed as a random variable, which is assumed to follow a distribution $\Pr(r|\theta_P, P)$, where θ_P refers to the set of parameters of this distribution. In each interaction, the consumer randomly draws a value r_P from $\Pr(r|\theta_P, P)$ for each $P \in \mathcal{P}$, selects the predicate $P^* = \arg\max_{P \in \mathcal{P}} r_P$ and issues query (P^*, I_Δ).

After obtaining the records for query (P, I_Δ), we update the distribution $\Pr(r|\theta_P, P)$, by updating the values of θ_P based on the records received. The distribution $\Pr(r|\theta_P, P)$ before and after the update are the *prior distribution* and *posterior distribution* respectively. We choose to use the Beta distribution [25] to model the

prior distribution, which has been shown to be effective in a variety of settings [40]. The Beta distribution is characterized by two parameters α and β, and it is a particularly good fit for our problem as α and β represent the pseudo counts of the number of correct and incorrect classifications we believe the CLF can make, providing our initial perspective of the reward function of P. Moreover, as shown in Section 4.1, the reward of (P, I_Δ), i.e., the number of correctly classified records in $\mathcal{R}_P^{I_\Delta}$, follows a Binomial distribution. It is known that the conjugate of Binomial distribution is the Beta distribution [17], and the posterior distribution will still be a Beta distribution, making parameter update highly tractable. We next show how to compute a Beta posterior from a Beta prior and $\mathcal{R}_P^{I_\Delta}$.

Let Beta(α, β) be the Beta distribution with two parameters α and β. In our case, we initialize both α and β to 1 for all predicates, essentially making the Beta distribution a uniform distribution, in line with the fact that the consumer has no knowledge regarding the rewards of predicates at the beginning. Suppose that the reward distribution for P is Beta(α_P, β_P) before a round of interaction, and $\mathcal{R}_P^{I_\Delta}$ is received in this round. Let $N_P^{I_\Delta}$ be the number of records correctly labelled by CLF, i.e.,

$$N_P^{I_\Delta} = \sum_{(x_i, y_i) \in \mathcal{R}_P^{I_\Delta}} \mathbb{I}[\text{CLF}((x_i, y_i)) = 0] \tag{11}$$

We can show that α_P and β_P can be updated as follows to obtain the posterior distribution conditional on $N_P^{I_\Delta}$ and I_Δ:

$$(\alpha_P, \beta_P) \leftarrow (\alpha_P, \beta_P) + (N_P^{I_\Delta}, I_\Delta - N_P^{I_\Delta}) \tag{12}$$

It follows immediately from Equation (12) that (1) the expectation of distribution Beta(α_P, β_P), computed as $\frac{\alpha_P}{\alpha_P+\beta_P}$, is proportional to the reward of P (notice that $\frac{\alpha_P}{\alpha_P+\beta_P}$ is essentially the percentage of records satisfying P that can be correctly labelled by CLF), so that the probabilities of selecting predicates with high observed rewards in future interactions are higher; and (2) after the update, $(\alpha_P + \beta_P + 2)$ is equal to the number of records obtained using P so far (as both α_P and β_P are initialized to 1). With more records acquired using P, $(\alpha_P + \beta_P)$ becomes larger and the distribution of Beta(α_P, β_P) becomes more concentrated, meaning that we are more confident regarding the expectation of P's reward. Note that this is also the reason why we use the number of correctly labelled records as the reward rather than percentage thereof: even both queries $(P, I_\Delta = 100)$ and $(P, I_\Delta = 10)$ return records of which 80% can be correctly labelled, the former should give us more confidence regarding the distribution of P's reward and thus (α_P, β_P) should be greater in this case.

5.3 Non-stationary Reward Distributions

For our discussion in Section 5.2, we have assumed that the reward distribution is stationary regardless of the number of records acquired in previous rounds. However, one can observe that as $\hat{\mathcal{R}}_P$ (the records the consumer has that satisfy predicate P) grows, the new information brought by each additional record from $\mathcal{D}_{pool}$ satisfying P decreases, and consequently the reward of P decreases. As such, not all past rewards observed should be treated equally. We should focus on the rewards observed from recent rounds, which better reflect the current reward distributions. Therefore, we modify the posterior computation in Equation (12) in a way that remembers only the rewards observed from the most recent τ rounds of interactions, as inspired by previous research on dealing with non-stationary reward distributions (e.g., [24, 40]).

More specifically, assume that the consumer has interacted with the provider using P for t rounds (including the current round), with rewards $N_P^{I_\Delta}[1], N_P^{I_\Delta}[2], \cdots, N_P^{I_\Delta}[t]$, then α_P and β_P are updated as follows in two steps:

$$\begin{aligned} &(\alpha_P, \beta_P) \leftarrow (\alpha_P, \beta_P) + (N_P^{I_\Delta}[t], I_\Delta - N_P^{I_\Delta}[t]); \\ &(\alpha_P, \beta_P) \leftarrow (\alpha_P, \beta_P) - (N_P^{I_\Delta}[t-\tau], I_\Delta - N_P^{I_\Delta}[t-\tau]), \text{ only if } t > \tau \end{aligned} \tag{13}$$

By remembering only the most recent rewards, the expectation of Beta(α_P, β_P) is closer to the current reward of predicate P. Besides, "forgetting" the previous rewards prevents Beta(α_P, β_P) from becoming too concentrated (recall that $(\alpha + \beta)$ influences how concentrated the distribution is), and thus always allows a chance for more exploration, suitable for the setting with changing rewards.

5.4 The SPS Algorithm

The operation of SPS is summarized in Algorithm 2. We initialize all reward distributions to Beta(1, 1) (Line 1). At each interaction, we randomly sample a value r_P from distribution Beta(α_P, β_P) for each P (Lines 3-4), and select the predicate P^* with the highest r_P and issue query (P^*, I_Δ) (Lines 5-6). After receiving a set of records, $\mathcal{R}_{P^*}^{I_\Delta}$, we update the values of α_P and β_P accordingly (Lines 7-8), merge $\mathcal{R}_{P^*}^{I_\Delta}$ into acquired records and deduct I_Δ from the remaining budget (Line 9). The process terminates when the budget is exhausted.

Algorithm 2 Sequential Predicate Selection

Input: budget B, all predicates $\mathcal{P}$
Output: A set of records $\mathcal{R}$
1: **Initialization:** $\forall P \in \mathcal{P}$, $\alpha_P = 1$, $\beta_P = 1$; $\mathcal{R} = \emptyset$
2: **while** $B > 0$ **do**
3: **for** P in $\mathcal{P}$ **do**
4: sample r_P from distribution Beta(α_P, β_P);
5: $P^* = \arg\max_{P \in \mathcal{P}} r_P$;
6: ask query (P^*, I_Δ) and receive records $\mathcal{R}_{P^*}^{I_\Delta}$;
7: compute $N_{P^*}^{I_\Delta} = \sum_{(x_i, y_i) \in \mathcal{R}_{P^*}^{I_\Delta}} \mathbb{I}[\text{CLF}((x_i, y_i) = 0)]$;
8: update $(\alpha_{P^*}, \beta_{P^*})$ with Equation (13);
9: $\mathcal{R} = \mathcal{R} \cup \mathcal{R}_{P^*}^{I_\Delta}$; $B = B - I_\Delta$;
10: **return** $\mathcal{R}$;

6 EXPERIMENTS

The techniques we propose are equally applicable to traditional Machine Learning [13] and Deep Learning [47] models across a variety of domains. We choose models and datasets in the experiments to reflect the wide range of applications we envision. More specifically, we choose state-of-the-art deep models as well as classical ML models to experiment with. The datasets used in the experiments include image data, spatial data, and optical-radar data, and the tasks range from classification to regression.

6.1 Settings

Datasets. We conduct experiments on four datasets. CIFAR10 and CIFAR100 [29] are image classification datasets widely used in the area of ML. The Crop mapping dataset [27] (Crop for short) contains temporal, spectral, textural, and polarimetric attributes for cropland classification. It has 175 real-valued features and one target (seven crop types). 3D Road Network [26] (RoadNet for short) is a geographical dataset consisting of tuples of longitude, latitude, and altitude. Following the instruction in [26], we use longitude and latitude as the features and altitude as the predicted value. The latter two datasets can also be found in the UCI data repository [18]. The characteristics of the four datasets are summarized in Table 1.

CIFAR10, CIFAR100, and Crop are used for classification tasks, while RoadNet is used for a regression task. To generate $\mathcal{D}_{test}$, we directly use the test sets provided by CIFAR10 and CIFAR100, and randomly select 20% records from Crop and RoadNet.

Table 1: Dataset Characteristics

dataset	*# records*	*# classes*	*# dimensions*
CIFAR10	60,000	10	1,024
CIFAR100	60,000	100	1,024
Crop	325,834	7	175
RoadNet	434,874	N/A	2

Models. For classification on CIFAR10 and CIFAR100, we adopt VGG8B with predsim loss function [38], one of the most recent and state-of-the-art deep learning structures. For classification on Crop, we use Decision Tree. For RoadNet, we use kNN Regressor [9]. We also utilized other applicable models for our evaluation (e.g., AlexNet [30], EfficientNet [44]) and observed similar trends. We use the code of VGG8B provided by the authors, and the Decision Tree and kNN Regressor implementation in scikit-learn. Default settings are adopted for all models.

Construction of $\mathcal{P}$. For CIFAR10, CIFAR100, and Crop, we build predicates based on class labels, resulting in 10 predicates for CIFAR10, 100 predicates for CIFAR100, and 7 predicates for Crop. For RoadNet, by default we discretize the data space by partitioning the range of each feature into four equal-width sub-ranges, resulting in $4^2 = 16$ cells; a predicate selects records falling into a specific cell. We study the influence of $|\mathcal{P}|$ in Section 6.7.

Construction of $\mathcal{D}_{init}$. We construct $\mathcal{D}_{init}$ using 20% records in the corresponding dataset $\mathcal{D}$ for CIFAR10 and CIFAR100, and 1% records for Crop and RoadNet. We select records from $\mathcal{R}_P$ for each $P \in \mathcal{P}$ to construct $\mathcal{D}_{init}$ following a power-law distribution. More specifically, with a random order of predicates in $\mathcal{P}$, let P_i be the i-th predicate ($i \in [1, |\mathcal{P}|]$), the number of records selected from $\mathcal{R}_{P_i}$ is proportional to i.

Selection of CLF. We use the kNN classifier with $k = 1$ as the CLF in our experiments. We study the impact of varying k values as well as utilizing diverse classifiers in Section 6.9. For Crop and RoadNet, we directly use the raw attributes as the input of CLF. In accordance to previous work on image-based kNN search [46], for CIFAR10 and CIFAR100, we first use HOG [16] (Histogram of Oriented Gradients), a widely-adopted image feature extractor, to transform an image into a feature vector, and use the transformed feature vectors as the input of CLF.

Evaluation. Let $\mathcal{D}_{otd}$ be the records acquired during the acquisition process. We train the model on $\mathcal{D}_{init} \cup \mathcal{D}_{otd}$ and evaluate on $\mathcal{D}_{test}$. For the classification tasks on CIFAR10, CIFAR100, and Crop, accuracy is computed as follows:

$$accuracy = \frac{\sum_{(x,y)\in\mathcal{D}_{test}} \mathbb{I}[\mathcal{M}(x) = y]}{|\mathcal{D}_{test}|} \tag{14}$$

where $\mathcal{M}(x)$ denotes the model output on x.

For the regression task on RoadNet, we use R^2 score to evaluate the performance, computed as follows:

$$R^2 = 1 - \frac{\sum_{(x,y)\in\mathcal{D}_{test}} (y - \mathcal{M}(x))^2}{\sum_{(x,y)\in\mathcal{D}_{test}} (y - \bar{y})^2} \tag{15}$$

where $\bar{y} = \frac{\sum_{(x,y)\in\mathcal{D}_{test}} y}{|\mathcal{D}_{test}|}$.

The acquisition process and model training are repeated ten times and the average accuracy/R^2 score is reported.

6.2 The Effect of l on EA

As described in Section 4, EA requires acquiring l% random records for each predicate from the provider for initialization. Here, we experimentally evaluate the effect of l on the performance of EA. To have a common basis for the evaluation of the trends we fix the significance level (δ) at 0.001 throughout the experiment. The results are provided in Figure 1.

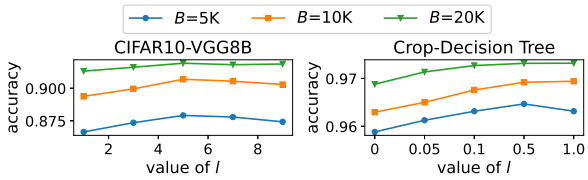

Figure 1: Effect of l on EA

As demonstrated in Figure 1, the performance of EA with a small budget is more sensitive to the choice of l, while l has a less significant impact on EA's performance when a large budget is used, except for cases where l is very small. The trade-off involved in selecting l can be summarized as follows. Using an overly-large l would result in too much budget consumption for the initialization, and consequently reduce the budget available for the Allocation Stage of EA, especially when the total budget B is small. On the other hand, using too small an l may cause the Estimation Stage to perform badly, leading to low-quality predicate utility estimates, as discussed in Section 4.4. In the following experiments, we set $l = 5$ for CIFAR10 and CIFAR100, and $l = 0.5$ for RoadNet and Crop. Since the budgets we use are fairly large for the respective dataset, the performance of EA is less dependent on the choice of l.

Takeaways. (1) An overly small or large l may harm the performance of EA. The value $l = 5$ for image data, and $l = 0.5$ for spatial data worked best during our experiments; (2) The performance of EA becomes less dependent on the choice of l as budget increases.

6.3 Estimation Accuracy of EA

In the Estimation Stage of EA, we assess the utility of predicate P, U_P, based on the records acquired with P, $\hat{\mathcal{R}}_P$. Let $\hat{U}_P$ be the estimated value of U_P. In this section, we examine the estimation accuracy. More specifically, we vary the number of records used for utility estimation, normalized by $|\mathcal{R}_P|$, i.e., $|\hat{\mathcal{R}}_P|/|\mathcal{R}_P|$, to study its effect on the absolute error of the estimation, i.e., $|U_P - \hat{U}_P|$. The results are presented in Figure 2.

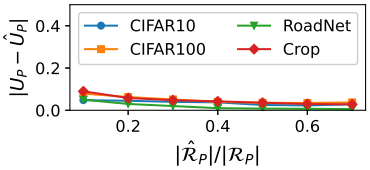

Figure 2: Estimation Accuracy

As can be observed from Figure 2, the absolute error of the estimation is consistently low (below 0.1), and increasing the number of records used for estimation further reduces the absolute error. The reason is that, according to Theorem 4.2, $U_P - \hat{U}_P$ follows a normal distribution, and increasing $|\hat{\mathcal{R}}_P|$ reduces the standard deviation of the distribution; consequently, the value of $\hat{U}_P$ converges to the true utility, U_P.

6.4 Comparison of Allocation Strategies in EA

Two strategies have been proposed in Section 4.3, namely, Linear Allocation and Square-root Allocation, which allocate the budget proportional to the estimated utility of each predicate or its square-root respectively. In this section we evaluate the impact of the allocation strategy on the performance of EA, and present the results in Figure 3. We also conduct one-tailed t-tests to determine whether the average accuracy improvement of one allocation strategy is statistically higher than the other. The cases for $B = 5,000$ and $B = 20,000$ are provided in Table 2, where μ_L (μ_S) denotes the average accuracy improvement of Linear (Square-root) Allocation.

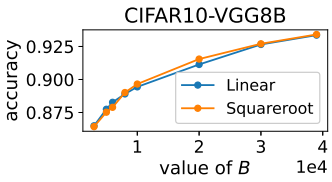

Figure 3: Allocation Strategies

Table 2: Significance Test

B	H_A	p-value
5,000	$\mu_L > \mu_S$	4e-3
20,000	$\mu_S > \mu_L$	1e-7

As is clear from Figure 3 and Table 2, different allocation strategies achieve similar performance overall, and with significance level $\alpha < 0.01$ we say Linear Allocation yields higher accuracy when the budget is relatively small (e.g., $B = 5,000$) and Square-root Allocation takes the lead when the budget is large (e.g., $B = 20,000$). This observation is the result of two competing underlying factors: on one hand, we should exploit the knowledge on the predicate utility gained from the Estimation Stage and therefore we should bias the allocation towards predicates with higher estimated utilities as much as possible (hence the superiority of Linear Allocation over Square-root Allocation for small budgets); on the other hand, as discussed in Section 5, the utility of a predicate P decreases as more records are acquired with P, i.e., the marginal benefit of obtaining records using predicates with higher estimated utilities becomes lower as the budget grows (hence Square-root Allocation outperforms Linear Allocation for large budgets). In other experiments we adopt Linear Allocation.

Takeaways. (1) Linear Allocation is better suited for data acquisition with budget $B \leq 0.2|\mathcal{D}|$ in our experiments; (2) Square-root Allocation is better suited for data acquisition with budget $B > 0.2|\mathcal{D}|$ during our evaluation.

6.5 The Effect of Batch Size on SPS

With SPS, records are acquired in small batches of size I_Δ. We evaluate the effect of I_Δ on the performance of SPS presenting our results in Figure 4.

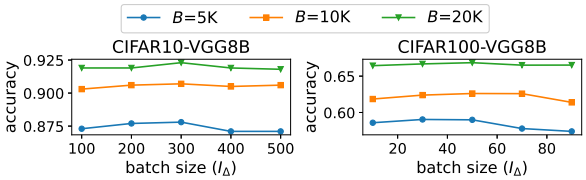

Figure 4: Effect of Batch Size on SPS

As can be observed from Figure 4, data acquisition with a small budget is more sensitive to the value of I_Δ. The trade-off in selecting I_Δ is as follows. Let (P, I_Δ) be the query. With a small I_Δ, the records returned by the provider may not be representative of $\mathcal{R}_P$, and the query reward thus computed is inaccurate. Consequently SPS cannot identify those predicates with high rewards. On the other hand, if I_Δ is too large, then each interaction consumes too much budget, limiting the exploitation of predicates with higher rewards. However, as the budget increases, SPS becomes progressively less sensitive to the batch size, because (1) the variations in the records obtained in each interaction have minimal impact on the overall estimation accuracy given a large number of interactions; (2) although exploration with large I_Δ consumes more budget, it also provides more information regarding the reward distribution (see Equation (12)), benefiting future predicate selection. In the following experiments, we select $I_\Delta = 300$ for CIFAR10, and $I_\Delta = 30$ for CIFAR100, Crop and RoadNet. Since the budgets we use are fairly large for the respective datasets, we expect less dependency on the choices of I_Δ.

Takeaways. (1) The accuracy of SPS is relatively stable over a wide range of batch size values, only slightly decreasing for an overly small or large batch size, depending on the ML/DL model and data; (2) The accuracy of SPS becomes less dependent on the batch size as budget increases.

6.6 The Effect of τ on SPS

As discussed in Section 5.3, to deal with the non-stationary reward, we only use the records acquired by the most recent τ queries with P to update the posterior distribution of P's reward. We evaluate the effect of τ on the performance of SPS, and report the results on CIFAR10 in Figure 5; similar trends can be observed on other datasets.

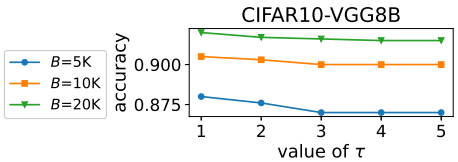

Figure 5: Effect of τ on SPS

As can be observed from Figure 5, the performance of SPS is relatively stable across different values of τ, with small τ yielding slightly higher accuracy. The reason is that, as indicated by Equation (13), with larger τ, α_P and β_P contain more dated reward observations and thus diverge from the current reward distribution;

this in turn may cause the acquisition of less useful records (in terms of accuracy improvement). Having less useful records clearly has a stronger influence on model accuracy when the total number of training records is smaller (corresponding to a smaller budget). However, with the SPS strategy, as long as the predicate with the actual maximal expected reward has a higher probability to be selected than the other predicates, the cumulative reward is likely to be maximized. Therefore, SPS is tolerant to inaccurate reward distribution estimations and robust to the value of τ. We set τ to 1 in the following experiments.

Takeaways. (1) Setting $\tau = 1$ always leads to higher accuracy during our evaluation; (2) SPS is relatively robust with respect to τ.

6.7 The Effect of $|\mathcal{P}|$

In this section we study the influence of the number of predicates, i.e., $|\mathcal{P}|$, on the accuracy of the methods. More specifically, we change $|\mathcal{P}|$ by changing either: (1) the number of labels to construct $\mathcal{P}$ for a classification dataset, or (2) the discretization granularity for a regression dataset. For case (1), we select CIFAR100 and use the m labels with which the model has the lowest accuracy (cross validated on $\mathcal{D}_{init}$) to construct $\mathcal{P}$. For case (2), we use RoadNet and partition the range of each of the two features into n equal-width sub-ranges, resulting in n^2 cells, which are used as $\mathcal{P}$. The results are presented in Figure 6.

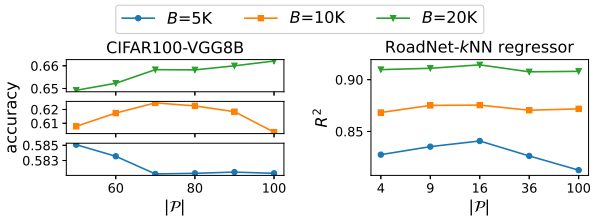

Figure 6: Effect of Space Discretization

As is clear from the results on CIFAR100, for case (1), the trend of accuracy with respect to $|\mathcal{P}|$ depends on the budget. The $|\mathcal{P}|$ yielding the maximal accuracy is 50 for B=5K, 70 for B=10K, and 100 for B=20K. The trend confirms our intuition that with a small budget, limiting the data acquisition to the predicates where the model is more error-prone reduces extra exploration cost and consequently increases the accuracy; however, this may result in over-exploitation of these predicates when the budget is large due to decreasing utility (as discussed in Section 5.3), leading to a slower increase in accuracy compared to larger $\mathcal{P}$. As can be observed from the results on RoadNet, for case (2), the R^2 score is relatively stable with respect to $|\mathcal{P}|$, with a slight increase when $|\mathcal{P}|$ is between 9 and 36. The reason is that, an overly coarse partitioning granularity (small $|\mathcal{P}|$) prevents the effective identification of the area in the data space where the model has a low R^2 score, while an overly fine partitioning granularity (large $|\mathcal{P}|$) increases the exploration cost as there are more predicates whose utilities need to be estimated.

Takeaways. During our experiments, (1) Descritizing the data space such that each cell occupies $3\% - 10\%$ of the size of the entire data space gives higher R^2 score; (2) For classification tasks with budget $B \leq 0.2|\mathcal{D}|$, using no more than 70% of all labels to construct $\mathcal{P}$ gives higher accuracy; with $B > 0.2|\mathcal{D}|$, using at least 70% of all labels to construct $\mathcal{P}$ leads to higher accuracy; (3) The performance of the proposed methods is generally stable in terms of the number of predicates.

6.8 EA vs. SPS

We now experimentally compare the two methods proposed in the paper, EA and SPS, and showcase the relative trends. We report the results in Figure 7. We also conduct one-sided t-tests to determine whether the average accuracy improvement of one method is statistically higher than the other. The cases for $B = 3,000$ and $B = 20,000$ are provided in Table 3, where μ_P (μ_E) denotes the average accuracy improvement of SPS (EA).

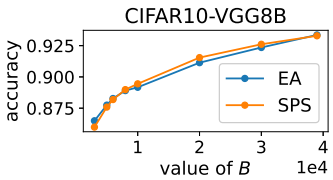

Figure 7: EA vs. SPS

Table 3: Significance Test

B	H_A	p-value
3,000	$\mu_E > \mu_P$	1e-4
20,000	$\mu_P > \mu_E$	1e-9

As can be observed from Figure 7 and Table 3, EA and SPS provide similar performance, and with significance level $\alpha < 0.001$ we say EA provides a higher accuracy gain with a small budget (e.g., $B = 3,000$) and SPS provides a higher accuracy improvement with a large budget (e.g., $B = 20,000$). The reason is that EA is a budget-aware method: with a small budget it tends to allocate less budget for utility estimation, aided by the heuristic reward, so that more budget can be allocated to predicates with high estimated utilities. The mechanism of SPS, on the other hand, is budget-agnostic, and the exploration of all predicate rewards at the start consumes budget and limits the chances to exploit predicates with high utilities, especially when the budget is small. As the budget increases, SPS starts to outperform EA since it acquires records in small batches and can flexibly adjust the acquisition strategy in face of utility changes; EA conducts one-time allocation without considering future utility changes and the records thus obtained may not be impactful to improve model accuracy.

Takeaways. In our evaluation, (1) EA is better suited for data acquisition with budget $B \leq 0.2|\mathcal{D}|$; (2) SPS is better suited for data acquisition with budget $B > 0.2|\mathcal{D}|$.

6.9 The Effect of CLF

As discussed in Section 3, the utility of a predicate is essentially the accuracy of a classifier (CLF) in differentiating $\mathcal{R}_P$ and $\mathcal{R}_{\mathcal{M}:P}$. Here we experimentally study the influence of CLF on the accuracy improvement. More specifically, since the utility computation is carried out frequently, we consider lightweight models including the kNN classifier ($k \in \{1, 3, 5\}$), the decision tree classifier, and the perceptron classifier. We report results on CIFAR10 in Figure 8; similar trends can be observed on other datasets.

As can be observed from Figure 8, the performance of the methods is not sensitive to the particular model used as the CLF and its hyper-parameters. In other experiments, we use the kNN classifier with $k = 1$.

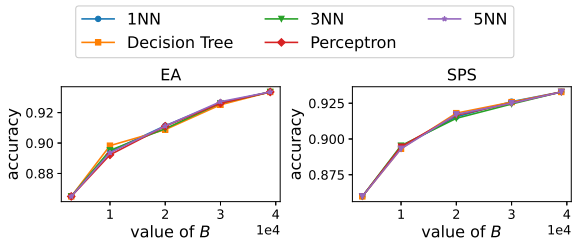

Figure 8: Effect of CLF

6.10 Comparison with Baseline Methods for Data Acquisition

The closest piece of work that can be adapted to our setting for comparison purposes is the work on active class selection (ACS) [34]. Although the problem solved therein is different, we can adapt the methods used for our setting. Therefore, we use ACS (adapted to our setting) as the baseline and present experimental results comparing it with our proposals. We note, however, that this is not a fair comparison, because ACS requires re-training the model after each interaction which can be computationally prohibitive for complex models involving large datasets, whereas ours does not.

Specifically, ACS acquires b new data records in each round (the same batch size as used in Section 6.5), which is allocated to each class uniformly (ACS-Uniform), or in proportion with the accuracy improvement for each class (ACS-AI), or the number of records in each class whose label has changed (ACS-RD) during the last round. Note that ACS-AI and ACS-RD require model re-training after new records are obtained and thus are too expensive to be applied to CIFAR10 and CIFAR100. As such, we apply ACS-Uniform to CIFAR10, and ACS-AI and ACS-RD to Crop, with results provided in Figure 9. The observations on other datasets are similar and are thus omitted.

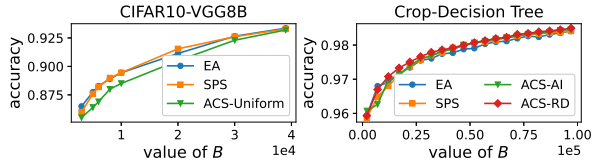

Figure 9: Comparison with Baselines

The results in Figure 9 (a) indicate that EA and SPS consistently outperform ACS-Uniform (except for the case of $B = 4,000$ when all samples are acquired) and achieve similar accuracy to baselines *that require the model to be re-trained after each interaction*, by effectively acquiring records with higher novelty that are more likely to boost accuracy.

6.11 Comparison with Utility Measures Based on Re-training/refinement

One of the main advantages of the utility measure we propose, novelty, is that it does not require the computationally expensive step of model re-training. In this section, we compare novelty with re-training-based utility measures in terms of model accuracy improvement and data acquisition cost. More specifically, we compare with a re-training-based measure where $\mathcal{M}$ is re-trained on newly-acquired records, say $\mathcal{R}_P^I$, and the improvement in accuracy after re-training is used as the utility of P. While all lightweight models are re-trained from scratch, we adopt the state-of-the-art incremental learning method, UCB [20], to refine deep models instead of conducting complete re-training to keep training overhead manageable. We report the results of using SPS together with both utility measures on CIFAR10 and Crop in Figure 10; observations are similar on other datasets and consistent in the case of using EA.

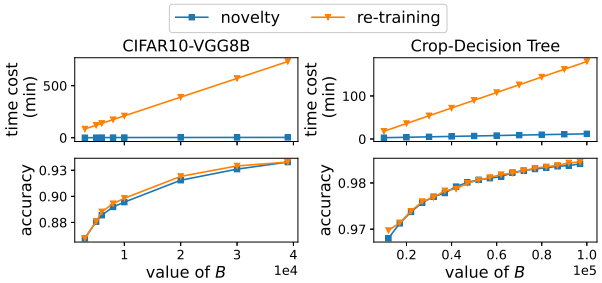

Figure 10: Comparison with Re-training-based Measure

The results in Figure 10 indicate that novelty achieves similar accuracy to the re-training-based utility measure by effectively acquiring records with higher novelty that are more likely to boost accuracy, while requiring orders of magnitude less time. Although the model can be refined incrementally with UCB, repetitive model refinement, especially when the budget is large, still results in high execution overhead. Although lightweight models such as decision trees can be constructed rapidly, repetitive construction imposes large computational overheads during the acquisition process. Novelty, on the contrary, conducts data acquisition by only looking at the data, and thus is highly efficient for practical deployments.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we have considered the problem of acquiring data in order to improve the accuracy of ML models, and laid out the framework of interaction between a provider and a consumer in the context of data markets. We have proposed two algorithmic solutions that the consumer with a limited budget could use to obtain data from the provider, striking a balance between exploration (gaining more knowledge on the data the provider possesses) and exploitation (utilizing that knowledge for allocating the limited budget for data acquisition). The first solution, EA, has two distinctive stages, Estimation Stage and Allocation Stage, focusing on exploration (obtaining accurate estimates on the predicate utilities) and exploitation (allocating the budget according to the estimates) respectively. The second solution, SPS, blends exploration and exploitation in each round of interaction, and adaptively allocates budget for the next round, investing resources into more promising areas of the data space to improve model accuracy. Results from our experimental studies have confirmed the effectiveness of our proposals, and illustrated the trade-offs and relative strengths of each proposed solution.

Our work represents the first step in dealing with data acquisition in a market setting, and research opportunities in this new area abound. For example, one could consider the problem of acquiring data from multiple providers with varying data coverage and data quality, or investigate new mechanisms for data acquisition where there is a third party (e.g., data broker) involved.

REFERENCES

[1] Amazon Mechanical Turk. https://www.mturk.com/
[2] Appen. https://appen.com/
[3] Dawex. https://www.dawex.com/en/
[4] Hive. https://thehive.ai/
[5] Twitter's enterprise API platform. https://developer.twitter.com/en/products/twitter-api/enterprise
[6] WorldQuant. https://data.worldquant.com
[7] Xignite. https://aws.amazon.com/solutionspace/financial-services/solutions/xignite-market-data-cloud-platform/
[8] Anish Agarwal, Munther A. Dahleh, and Tuhin Sarkar. 2019. A Marketplace for Data: An Algorithmic Solution. In *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019*. ACM, 701–726.
[9] Naomi S Altman. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46, 3 (1992), 175–185.
[10] Robert B Ash, B Robert, Catherine A Doleans-Dade, and A Catherine. 2000. *Probability and measure theory*. Academic Press.
[11] Magdalena Balazinska, Bill Howe, and Dan Suciu. 2011. Data markets in the cloud: An opportunity for the database community. *Proceedings of the VLDB Endowment* 4, 12 (2011), 1482–1485.
[12] Shuchi Chawla, Shaleen Deep, Paraschos Koutris, and Yifeng Teng. 2019. Revenue Maximization for Query Pricing. *Proc. VLDB Endow.* 13, 1 (2019), 1–14.
[13] Lingjiao Chen, Paraschos Koutris, and Arun Kumar. 2019. Towards Model-based Pricing for Machine Learning in a Data Marketplace. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. ACM, 1535–1552.
[14] Yiling Chen, Nicole Immorlica, Brendan Lucier, Vasilis Syrgkanis, and Juba Ziani. 2018. Optimal Data Acquisition for Statistical Estimation. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*. ACM, 27–44.
[15] Nadiia Chepurko, Ryan Marcus, Emanuel Zgraggen, Raul Castro Fernandez, Tim Kraska, and David Karger. 2020. ARDA: Automatic Relational Data Augmentation for Machine Learning. *Proc. VLDB Endow.* 13, 9 (2020), 1373–1387.
[16] Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Vol. 1. IEEE, 886–893.
[17] Persi Diaconis and Donald Ylvisaker. 1979. Conjugate priors for exponential families. *The Annals of statistics* (1979), 269–281.
[18] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
[19] Anshuman Dutt, Chi Wang, Vivek Narasayya, and Surajit Chaudhuri. 2020. Efficiently approximating selectivity functions using low overhead regression models. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2215–2228.
[20] Sayna Ebrahimi, Mohamed Elhoseiny, Trevor Darrell, and Marcus Rohrbach. 2020. Uncertainty-guided Continual Learning with Bayesian Neural Networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.
[21] Raul Castro Fernandez, Pranav Subramaniam, and Michael J Franklin. 2020. Data market platforms: Trading data assets to solve data problems. *Proceedings of the VLDB Endowment* 13, 12 (2020), 1933–1947.
[22] Ronald Aylmer Fisher. 1992. Statistical methods for research workers. In *Breakthroughs in statistics*. Springer, 66–70.
[23] Yuhong Guo and Dale Schuurmans. 2007. Discriminative batch mode active learning. *Advances in neural information processing systems* 20 (2007), 593–600.
[24] Elena Ikonomovska, Sina Jafarpour, and Ali Dasdan. 2015. Real-time bid prediction using thompson sampling-based expert selection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1869–1878.
[25] Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. 1995. *Continuous univariate distributions*. John Wiley & Sons, Ltd.
[26] Manohar Kaul, Bin Yang, and Christian S Jensen. 2013. Building accurate 3d spatial networks to enable next generation intelligent transportation systems. In *2013 IEEE 14th International Conference on Mobile Data Management*, Vol. 1. IEEE, 137–146.
[27] Iman Khosravi and Seyed Kazem Alavipanah. 2019. A random forest-based framework for crop mapping using temporal, spectral, textural and polarimetric observations. *International Journal of Remote Sensing* 40, 18 (2019), 7221–7251.
[28] Yuqing Kong, Grant Schoenebeck, Biaoshuai Tao, and Fang-Yi Yu. 2020. Information Elicitation Mechanisms for Statistical Estimation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2095–2102.
[29] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).
[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. 1106–1114.
[31] Arun Kumar, Jeffrey Naughton, Jignesh M Patel, and Xiaojin Zhu. 2016. To join or not to join? thinking twice about joins before feature selection. In *Proceedings of the 2016 International Conference on Management of Data*. 19–34.
[32] Bing-Rong Lin and Daniel Kifer. 2014. On arbitrage-free pricing for general data queries. *Proceedings of the VLDB Endowment* 7, 9 (2014), 757–768.
[33] Jinfei Liu. 2020. Dealer: End-to-End Data Marketplace with Model-based Pricing. arXiv:2003.13103 [cs.DB]
[34] R. Lomasky, C. E. Brodley, M. Aernecke, D. Walt, and M. Friedl. 2007. Active Class Selection. In *Machine Learning: ECML 2007*.
[35] David Lopez-Paz and Maxime Oquab. 2017. Revisiting Classifier Two-Sample Tests. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
[36] Lin Ma, Bailu Ding, Sudipto Das, and Adith Swaminathan. 2020. Active Learning for ML Enhanced Database Systems. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 175–191.
[37] Sameer Mehta, Milind Dawande, Ganesh Janakiraman, and Vijay S. Mookerjee. 2019. How to Sell a Dataset?: Pricing Policies for Data Monetization. In *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019*. ACM, 679.
[38] Arild Nøkland and Lars Hiller Eidnes. 2019. Training Neural Networks with Local Error Signals. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research)*, Vol. 97. PMLR, 4839–4850.
[39] Jian Pei. 2020. Data Pricing - From Economics to Data Science. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 3553–3554.
[40] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. 2017. A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038* (2017).
[41] Vraj Shah, Arun Kumar, and Xiaojin Zhu. 2017. Are Key-Foreign Key Joins Safe to Avoid when Learning High-Capacity Classifiers? *Proc. VLDB Endow.* 11, 3 (2017), 366–379.
[42] Changjian Shui, Fan Zhou, Christian Gagné, and Boyu Wang. 2020. Deep Active Learning: Unified and Principled Method for Query and Training. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. 1308–1318.
[43] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[44] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research)*, Vol. 97. PMLR, 6105–6114.
[45] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.
[46] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. 2014. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1386–1393.
[47] Xi Yan, David Acuna, and Sanja Fidler. 2020. Neural Data Server: A Large-Scale Search Engine for Transfer Learning Data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 3892–3901.
[48] Meng Zhang, Ahmed Arafa, Ermin Wei, and Randall A. Berry. 2020. Optimal and Quantized Mechanism Design for Fresh Data Acquisition. arXiv:2006.15751