AIDE: An Active Learning-Based Approach for Interactive Data Exploration

Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao

Abstract—In this paper, we argue that database systems be augmented with an automated data exploration service that methodically steers users through the data in a meaningful way. Such an automated system is crucial for deriving insights from complex datasets found in many big data applications such as scientific and healthcare applications as well as for reducing the human effort of data exploration. Towards this end, we present AIDE, an Automatic Interactive Data Exploration framework that assists users in discovering new interesting data patterns and eliminate expensive ad-hoc exploratory queries. AIDE relies on a seamless integration of classification algorithms and data management optimization techniques that collectively strive to accurately learn the user interests based on his relevance feedback on strategically collected samples. We present a number of exploration techniques as well as optimizations that minimize the number of samples presented to the user while offering interactive performance. AIDE can deliver highly accurate query predictions for very common conjunctive queries with small user effort while, given a reasonable number of samples, it can predict with high accuracy complex disjunctive queries. It provides interactive performance as it limits the user wait time per iteration of exploration to less than a few seconds.

Index Terms—Data exploration, data sampling

1 Introduction

Traditional data management systems assume that when users pose a query they a) have good knowledge of the schema, meaning and contents of the database and b) they are certain that this particular query is the one they wanted to pose. In short, traditional DBMSs are designed for applications in which the users know what they are looking for. However, as data are being collected and stored at an unprecedented rate, we are building more dynamic data-driven applications where this assumption is not always true.

Interactive data exploration (IDE) is one such example. In these applications, users are trying to make sense of the underlying data space by experimenting with queries, backtracking on the basis of query results and rewriting their queries aiming to discover interesting data objects. IDE often incorporates "human-in-the-loop" and it is fundamentally a long-running, multi-step process with the user's interests specified in imprecise terms.

One application of IDE can be found in the domain of evidence-based medicine (EBM). Such applications often involve the generation of systematic reviews, a comprehensive assessment of the totality of evidence that addresses a well-defined question, such as the effect on mortality of giving versus not giving drug A within three hours of a symptom B. While a content expert can judge whether a given clinical trial is of interest or not (e.g., by reviewing parameter values such as disease, patient age, etc.), he often does not have a priori knowledge of the exact attributes that should be used to formulate a query to collect all relevant clinical trials. Therefore the user relies on an ad hoc process that includes three steps: 1) processing numerous selection queries with iteratively varying selection predicates, 2) reviewing returned objects (i.e., trials) and classifying them to relevant and irrelevant, and 3) adjusting accordingly the selection query for the next iteration. The goal here is to discover the selection predicates that balances the trade-off between collecting all relevant objects and reducing the size of returned results. These "manual" explorations are typically labor-intensive: they may take days to weeks to complete since users need to examine thousands of objects.

Scientific applications, such as ones analysing astrophysical surveys (e.g., [1], [2]), also suffer from similar situations. Consider an astronomer looking for interesting patterns over a scientific database: they do not know what they are looking for, they only wish to find interesting patterns; they will know that something is interesting only after they find it. In this setting, there are no clear indications about how the astronomers should formulate their queries. Instead, they may want to navigate through a subspace of the data set (e.g., a region of the sky) to find objects of interest, or may want to see a few samples, provide yes/no feedback, and expect the system to find more similar objects.

To address the needs of IDE applications, we propose an *Automatic Interactive Data Exploration (AIDE)* framework that automatically discovers data relevant to her interest. Our approach unifies the three IDE steps—query formulation, query processing and result reviewing—into a single automatic process, significantly reducing the user's exploration effort and the overall exploration time. In particular, an AIDE user engages in a "conversation" with the system indicating her interests, while in the background the system builds a user model that predicts data matching these interests.

AIDE offers an iterative exploration model: in each iteration the user is prompted to provide her feedback on a set of sample objects as relevant or irrelevant to her exploration

- *K. Dimitriadou and O. Papaemmanouil are with Brandeis University, Waltham, MA 02453. E-mail: {kiki, olga}@cs.brandeis.edu.*
- *Y. Diao is with the University of Massachusetts, Amherst, MA 01003. E-mail: yanlei@cs.umass.edu.*

Manuscript received 23 Sept. 2015; revised 7 June 2016; accepted 27 July 2016. Date of publication 10 Aug. 2016; date of current version 3 Oct. 2016. Recommended for acceptance by K. Chang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2016.2599168

task. Based on her feedback, AIDE generates the user's exploration profile, i.e., a user model that classifies database objects as relevant or irrelevant. AIDE leverages this model to explore further the data space, identify strategic sampling areas and collect new samples for the next iteration. These samples are presented to the user and her new feedback is incorporated into the user model. This iterative process aims to generate a user model that identifies all relevant objects while eliminating the misclassification of irrelevant ones.

AIDE's model raises new challenges. First, AIDE operates on the unlabeled space of the whole data space that the user aims to explore. To offer effective exploration results (i.e., accurately predict the user's interests) it has to decide and retrieve in an *online* fashion the example objects to be extracted and labeled by the user. Second, to achieve desirable interactive experience for the user, AIDE needs not only to provide accurate results, but also to minimize the number of samples presented to the user (which determines the amount of user effort) as well as to reduce the sampling and space exploration overhead (which determines the user's wait time in each iteration).

These challenges cannot be addressed by existing machine learning techniques. Classification algorithms (e.g., [3]) can build the user model and the information retrieval community offers solutions on incrementally incorporating relevance feedback in these models (e.g., [4]). However, these approaches operate under the assumption that the sample set shown to the user is either known a priori or, in the case of online classification, it is provided incrementally by a different party. In other words, classification algorithms do not deal with *which* data samples to show to the user, which is one of the main research challenges for AIDE.

Active learning systems [5] also extract unlabeled samples to be labeled by a user and the goal is to achieve high accuracy using as few labeled samples as possible. In particular, pool-based sampling techniques selectively draw samples from a large pool of unlabeled data. However, these solutions exhaustively examine *all* unlabeled objects in the pool in order to identify the best samples to show to the user based on some informativeness measure [6]. Therefore, they implicitly assume negligible sample acquisition costs and hence cannot offer interactive performance on big data sets as expected by IDE applications. In either case, model learning and sample acquisition are decoupled, with the active learning algorithms not addressing the challenge of *how* to minimize the cost of sample acquisition.

To address the above challenges, AIDE closely *integrates* the active learning paradigm and sample acquisition through a set of exploration heuristics. These heuristics leverage the classification properties of decision tree learning to identify promising data exploration areas from which new samples are extracted, as well as to minimize the number of samples shown to the user. Our techniques are designed to predict linear patterns of user interests, i.e., we assume relevant objects are clustered in multi-dimensional hyper-rectangles. These interests can be expressed as range queries with disjunctive and/or conjunctive predicates.

This paper extends our previous our previous work on automatic data exploration [7], [8]. Specifically, we extended AIDE with a number of performance optimizations that are designed to reduce the total exploration overhead. Specifically, we introduce: (a) a skew-aware exploration technique that deals with both uniform and skewed data spaces as well as user interests that lie on either the sparse or dense parts of the distribution, (b) a probabilistic sampling strategy for selecting the most informative sample to present to the user; the strategy is designed to reduce the user's exploration effort and (c) an extended relevance feedback model that allows users to annotate "similar" (rather than only relevant/irrelevant) samples, allowing us to further reduce the total exploration time. We also include a new set of experimental results that demonstrate the effectiveness and efficiency of our new exploration techniques.

The specific contributions of this work are the following:

1) We introduce AIDE, a novel, automatic data exploration framework, that navigates the user throughout the data space he wishes to explore. AIDE relies on the user's feedback on example objects to generate a user model that predicts data relevant to the user. It employs a unique combination of machine learning, data exploration, and sample acquisition techniques to deliver highly accurate predictions of linear patterns of user interests with interactive performance. Our data exploration techniques leverage the properties of classification models to identify *single* objects of interest, expand them to more accurate *areas of interests*, and progressively refine the prediction of these areas. Our techniques address the trade-off between quality of results (i.e., accuracy) and efficiency (i.e., the total *exploration time* which includes the total sample reviewing time and wait time by the user).
2) We introduce new optimizations that address the presence of skew in the underlying exploration space as well as a novel probabilistic approach for identifying the most informative sample set to show to the user. We also include an extended feedback model based on which the user can also indicate similar but not necessarily relevant objects. This new model allows AIDE to focus its exploration on on certain promising domain ranges reducing significantly the user's labeling effort.
3) We evaluated AIDE using the SDSS database [2] and a user study. When compared with traditional active learning *and* passive learning (i.e., random sampling), AIDE and its novel optimizations are strictly more effective *and* efficient. AIDE can predict common conjunctive queries with a small number of samples, while given an acceptable number of labeled samples it predicts highly complex disjunctive queries with high accuracy. It offers interactive performance as the user wait time per iteration is less than a few seconds in average. Our user study revealed that AIDE can reduce the user's labeling effort by up 87 percent, with an average of 66 percent reduction. When including the sample reviewing time, it reduced the total exploration time by 47 percent in average.

The rest of the paper is organized as follows. Section 2 outlines the AIDE framework and Section 3 describes the phases of our data exploration approach. Section 4 discusses the new performance optimizations we introduce in AIDE. Section 5 presents our experimental results. Section 6 discusses the related work and we conclude in Section 7.

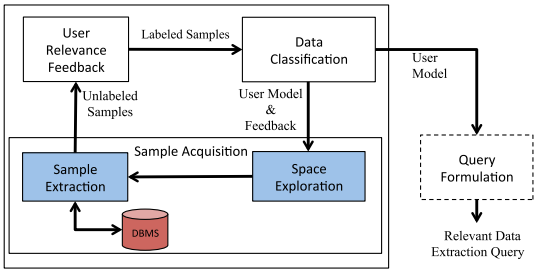


Fig. 1. Automated interactive data exploration framework.

2 AIDE Framework Overview

In this section we introduce our system model, the classification algorithms we use and we define our exploration problem.

2.1 System Model

The workflow of our exploration framework is depicted in Fig. 1. AIDE presents to the user sample database objects and requests her feedback on their relevance to her exploration task, i.e., characterize them as relevant or not. For example, in the domain of evidence-based medicine, users are shown sample clinical trials and they are asked to review their abstract and their attributes (e.g., year, outcome, patience age, medication dosage, etc) and label each sample trial as interesting or not. AIDE allows also the user to annotate samples that are similar (in some attribute) but not match exactly her interest, by marking them as "similar" samples. Finally, the user can modify her feedback on previously seen samples, however this could prolong the exploration process.

The iterative steering process starts when the user provides her feedback by labeling samples are relevant or not. The relevant and irrelevant samples are used to train a binary classification model that characterizes the user's interest, e.g., it predicts which clinical trials are relevant to the user based on the feedback collected so far (*Data Classification*)[1]. This model may use any subset of the object's attributes to characterize user interests. However, domain experts could leverage their domain knowledge to restrict the attribute set on which the exploration is performed. For instance, one could request an exploration only on the attributes *dosage* and *age*. In this case, relevant trials will be characterized on a subset of these attributes (e.g., relevant trials have dosage >45 mg).

In each iteration, more samples (e.g., records of clinical trials) are extracted and presented to the user for feedback. AIDE leverages the current user model as well as the user's feedback so far to identify promising sampling areas (*Space Exploration*) and retrieve the next sample set from the database (*Sample Extraction*). New labeled objects are incorporated with the already labeled sample set and a new classification model is built. The steering process is completed when the user terminates the process explicitly, e.g., when reaching a satisfactory set of relevant objects or when she does not wish to label more samples. Optionally, AIDE "translates" the classification model into a query expression (*Query Formulation*). This query will retrieve objects characterized as relevant by the user model (*Data Extraction Query*).

1. "Similar" samples are not included in the training of the user model. In Section 4.3 we discuss in detail how we leverage these samples.

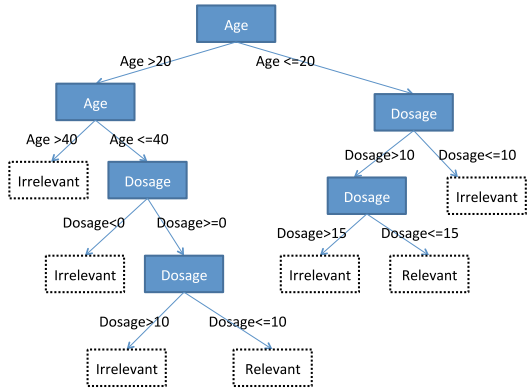


Fig. 2. An example decision tree.

AIDE strives to converge to a model that captures the user interest, i.e., eliminating irrelevant objects while identifying a large fraction of relevant ones. Each round refines the user model by exploring further the data space. The user decides on the effort he is willing to invest (i.e., number of samples he labels) while AIDE leverages his feedback to strategically sample the exploration space, i.e., collect samples that improve the accuracy of the classification model. The more effort invested in this iterative process, the more effective the user model will be.

2.2 Data Classification and Query Formulation

AIDE relies on decision tree classifiers to identify linear patterns of user interests, i.e., relevant objects clustered in multi-dimensional hyper-rectangles. Decision tree learning [3] produces classification models that predict the class of an unclassified object based on labeled training data. The major advantage of decision trees is that they provide easy to interpret models that clearly describe the features characterizing each data class. Furthermore, they perform well with large data and the decision conditions of the model can be easily translated to simple boolean expressions. This feature is important since it allows us to map decision trees to queries that retrieve the relevant data objects.

Finally, decision trees can handle both numerical and categorical data. This allows AIDE to operate on both data types assuming a distance function is provided to calculate the similarity between two data objects. Measuring the similarity between two objects is a requirement of the space exploration step. AIDE treats the similarity computation as an orthogonal step and can make use of any distance measure. For continuous data sets (e.g., numerical), the euclidean distance can be used. Computing similarity between categorical data is more challenging because there is no specific ordering between categorical values. However, various similarity measures have been proposed for categorical data, and AIDE can be extended in a straightforward way to incorporate them.

Query Formulation. Let us assume a decision tree classifier that predicts relevant and irrelevant clinical trials objects based on the attributes `age` and `dosage` (Fig. 2). This tree provides predicates that characterize the relevant class and predicates that describe the irrelevant class. In Fig. 2, the relevant class is described by the predicates $(age \leq 20 \wedge 10 < dosage \leq 15)$ and $(20 < age \leq 40 \wedge 0 \leq dosage \leq 10)$, while the irrelevant class is characterized by the predicates $(age \leq 20 \wedge dosage \leq 10)$ and $(20 < age \leq 40 \wedge dosage > 10)$ (here we ignore the predicates that refer to values outside

attribute domains, such as $age > 40$, $age < 0$, $dosage < 0$ and $dosage > 15$). Given the decision tree in Fig. 2 it is straightforward to formulate the extraction query for the relevant objects (*select * from table where (age ≤ 20 and dosage >10 and dosage ≤ 15) or (age > 20 and age ≤ 40 and dosage ≥ 0 and dosage ≥ 10)*).

2.3 Problem Definition

Given a database schema $\mathcal{D}$, let us assume the user has decided to focus his exploration on d attributes, where these d attributes may include both attributes relevant and those irrelevant to the final query that represents the true user interest. Each exploration task is then performed in a d-dimensional space of T tuples where each tuple represents an object characterized by d attributes. For a given user, our exploration space is divided to the relevant object set T^r and irrelevant set T^{nr}. Since the user's interests are unknown to AIDE, the sets T^r and T^{nr} are also unknown in advance.

AIDE aims to generate a model that predicts these two sets, i.e., classifies a tuple in T as relevant or irrelevant. To achieve that, it iteratively trains a decision tree classifier. Specifically, in each iteration i, a sample tuple set $S_i \subseteq T$ is shown to the user and his relevance feedback assigns these samples to two data classes, the relevant object class $D^r \subseteq T^r$, and the irrelevant one, $D^{nr} \subseteq T^{nr}$. Based on the samples assigned to these classes up to the i-th iteration, a new decision tree classifier C_i is generated. This classifier corresponds to a predicate set $P_i^r \bigcup P_i^{nr}$, where the predicates P_i^r characterize the relevant class and predicates P_i^{nr} describe the irrelevant one.

We measure AIDE's effectiveness (aka accuracy of a classification model) by evaluating the F-measure, the harmonic mean between precision and recall.[2] Our goal is to maximize the F-measure of the final decision tree C on the total data space T, defined as: $F(T) = \frac{2 \times precision(T) \times recall(T)}{precision(T) + recall(T)}$. The perfect precision value of 1.0 means that every object characterized as relevant by the decision tree is indeed relevant, while a good recall ensures that our final query can retrieve a good percentage of the relevant to the user objects.

3 Space Exploration Techniques

Our main research focus is on optimizing the effectiveness of the exploration (i.e., the accuracy of the final user model) while offering interactive experience to the user. To address that AIDE strives to improve on a number of efficiency factors, including the number of samples presented to the user and the number of sample extraction queries processed in the backend. In this section, we introduce our main exploration heuristics that tackle these goals.

AIDE assumes that user interests are captured by *range queries*, i.e., relevant objects are clustered in one or more areas in the data space. Therefore, our goal is to generate a user model that predicts *relevant areas*. The user model can then be translated to a range query that selects either a single multi-dimensional relevant area (conjunctive query) or multiple ones (disjunctive query).

AIDE incorporates three exploration phases. First, we focus on collecting samples from yet unexplored areas and identifying single relevant objects (*Relevant Object Discovery*). Next, we strive to leverage single relevant objects to generate a user model that identifies relevant *areas* (*Misclassified Exploitation*). Finally, given a set of discovered relevant areas, we gradually refine their boundaries (*Boundary Exploitation*). In each iteration i, these three phases define the new sample set we will present to the user. Specifically, if T_d^i, T_m^i and T_b^i samples will be selected by the object discovery, the misclassified and the boundary exploitation phase, then the user is presented with $S_i = T_d^i + T_m^i + T_b^i$ samples.

Our three exploration phases are designed to collectively increase the accuracy of the exploration results. Given a set of relevant objects from the object discovery step, the misclassified exploitation increases the number of relevant samples in our training set while reducing the misclassified objects (specifically false negatives). Hence, this step improves both the recall and the precision parameters of the F-measure metric. The boundary exploitation further refines the characterization of the already discovered relevant areas. Therefore, it discovers more relevant objects and eliminates misclassified ones, leading also to higher recall and precision. Finally, similarly to general active learning algorithms, AIDE does not provide theoretical guarantees on the number of required labeled samples since these depend on the distribution of the data spaces and the hypothesis of the user models (linear separator, homogeneous separators, etc). Next, we discuss in detail each phase. More details about these exploration techniques can be found in [7].

3.1 Relevant Object Discovery

Our first exploration phase aims to discover relevant objects by showing to the user samples from diverse data areas. To maximize the coverage of the exploration space we follow a well-structured approach that allows us to (1) ensure that the exploration space is explored widely, (2) keep track of the already explored sub-areas, and (3) explore different data areas in different granularity.

Our approach operates on a set of *hierarchical exploration grids*. Given an exploration task on d attributes, we define the *exploration space* to be the d-dimensional data area defined by the *domain* of these attributes. AIDE creates offline a set of grids and each grid divides the exploration space into d-dimensional equi-width cells. We refer to each grid as an *exploration level* and each level has a different granularity, i.e., cells of different width. The lower the exploration level the more fine-grained the grid cells (i.e., smaller cells) it includes and therefore moving between levels allows us to "zoom in/out" into specific areas as needed.

Exploration Level Construction. To generate an exploration level on a d-dimensional exploration space we divide each normalized attribute domain[3] into width ranges that cover δ percentage of the normalized domain, effectively creating $(100/\delta)^d$ grid cells. The δ parameter defines the granularity

2. Here, if tp are the true positives results of the classifier (i.e., correct classifications as relevant), fp are the false positives (i.e., irrelevant data classified as relevant) and fn are the false negatives (i.e., relevant data classified as irrelevant), we define the precision of our classifier as $precision = \frac{tp}{tp+fp}$ and the recall as $recall = \frac{tp}{tp+fn}$.

3. We normalize each domain to be between [0,100]. This allow us to reason about the distance between values uniformly across domains. Operating on actual domains will not affect the design of our framework or our results.

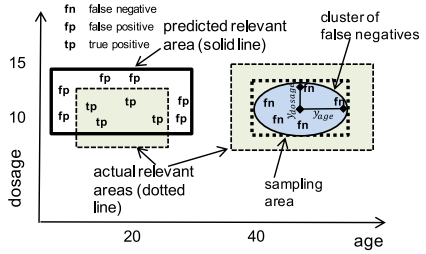


Fig. 3. Misclassified objects and cluster-based sampling.

of the specific exploration level. A lower number leads to more grid cells of smaller width per dimension. Each cell in our grid covers a certain range of attribute values for each of the d exploration attributes. Therefore, each cell includes a set of unique attribute value combinations and it includes the data objects that match these attribute values.

Discovery Phase. Our exploration starts at the highest exploration level i where $\delta = 100$ and includes a single grid cell. At each level it retrieves one random object from each non-empty cell. In the next iteration it samples the lower level $i = i + 1$ where $\delta/2^i$ until the user terminates the exploration. If no relevant object is retrieved from one cell, we can safely infer that the *whole* grid cell is not included in any relevant area. However, sub-areas of the grid could partially overlap with some relevant areas. By moving to the a lower exploration level, we "zoom-in" into this grid cell and increase the probability of discovering a relevant object.

3.2 Misclassified Samples Exploitation

While the object discovery phase bootstraps the discovery of relevant objects, it extracts at *most one* object of interest in each sampling area explored. In order to offer acceptable accuracy, decision tree classifiers require a higher number of samples from the relevant class. AIDE employs the *misclassified samples exploitation* phase which improves the accuracy our predictions by increasing the number of relevant objects in our training set.

Misclassified objects can be categorized to: (i) *false positives*, i.e., objects that are categorized as relevant by the classifier but labeled as irrelevant by the user and (ii) *false negatives*, i.e., objects labeled as relevant but categorized as irrelevant by the classifier. False positives are less common because the classifications rules of decision trees aim to maximize the homogeneity of their predicted relevant and irrelevant areas [3]. Practically, this implies that the classifier defines the relevant areas such as the relevant samples they include are maximized while minimizing the irrelevant ones. In fact, most false positives are due to wrongly predicted boundaries of these areas. Fig. 3 shows examples of false positives around a predicted relevant area. Elimination of these misclassified samples will be addressed by the boundary exploitation phase (Section 3.3).

False negatives on the other hand are objects of interest that belong in an *undiscovered* relevant area. Examples of false negative are also shown in Fig. 3. Relevant areas are undiscovered by the decision tree due to insufficient samples from within that area. Hence, AIDE increases the set of relevant samples by collecting more objects around false negatives.

Clustering-Based Exploitation. Our misclassified exploitation phase operates under the assumption that relevant tuples will be clustered close to each other, i.e., they typically form relevant areas. This implies that sampling around false negatives will increase the number of relevant samples. Furthermore, false negatives that belong in the same relevant area will be located close to each other. Hence, AIDE generates *clusters of misclassified objects* and defines a new sampling area around each cluster. Specifically, it creates clusters using the *k-means* algorithm [3] and defines one sampling area per cluster. An example of a cluster of false negatives is shown in Fig. 3.

The main challenge in this approach is identifying the number of clusters we need to create. Ideally, we would like this number to match the number of relevant areas we have "hit" so far, i.e., the number of relevant areas from within which we have collected at least one object. We argue that the number of relevant objects created by the object discovery phase is a strong indicator of the number of relevant areas we have already "hit". The object discovery phase identifies objects of interest that belong to different areas or the same relevant area. In the former case, our indicator offers correct information. In the latter case, our indicator will lead us to create more clusters than the already "hit" relevant areas. However, since these clusters belong in the same relevant area they are typically close to each other and therefore the decision tree classifier eventually "merges" them and converges to an accurate number of relevant areas.

In each iteration, the algorithm sets k to be the overall number of relevant objects discovered in the object discovery phase. Since our goal is to reduce the number of sampling areas (and therefore the number of sample extraction queries), we run the clustering-based exploitation only if k is less than the number of false negatives. Specifically, we collect samples within a distance y_i from the center of each cluster at each dimension. If no cluster is created, we sample within distance y_i at each dimension from each false negative. In either case we retrieve f random samples within the sampling area. The f value should be picked to ensure the relative proportion of relevant and irrelevant samples allows the classifier to identify the relevant areas with as few exploration rounds as possible. We observed that a relative proportion of 1:10 of relevant versus irrelevant samples is sufficient. Since the number of irrelevant samples depend on the size of the relevant area, one can use an adaptive approach and set the f value in each iteration to be the $1/10$ of the number of irrelevant samples already collected.

The parameter y_i defines the sampling area and affects the number of relevant samples we collect around each misclassified object. Setting y_i to a value that maximizes the overlap of the sampling area with the actual relevant area will allow AIDE to collect more relevant samples and identify the relevant area with less exploration rounds. If a cluster of misclassified is formed, we set y_i to be the distance of the farthest cluster member from the center of the cluster in the dimension i. This guarantees that we will collect mostly relevant samples. Otherwise, we initially set y_i to a small value (the same for all dimensions) and we automatically adjust it. Specifically, if no new relevant samples are discovered after the first sampling around a misclassified, we conclude that the sampling area exceeds the relevant area in at least one dimension. We then decrease y_i in all dimensions and we sample closer to the relevant sample. We repeat the process until the relevant samples can form a cluster of misclassified samples.

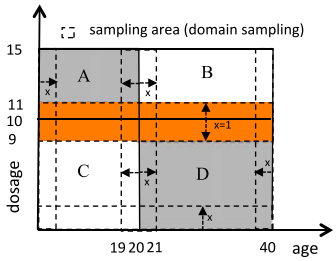


Fig. 4. Boundary exploration for the relevant areas A and D.

3.3 Boundary Exploitation

Given a set of relevant areas identified by the decision tree classifier, our next phase aims to refine these areas by incrementally adjusting their boundaries. This leads to better characterization of the user's interests, i.e., higher accuracy of our final results. In this section, we describe our general approach.

AIDE represents the decision tree classifier C_i generated at the ith iteration as a set of hyper-rectangles in a d-dimensional space defined by the predicates in $P_i^r \bigcup P_i^{nr}$, where the predicates P_i^r characterize the relevant areas and predicates P_i^{nr} describe the irrelevant areas. We iteratively refine these predicates by *shrinking* and/or *expanding* the boundaries of the hyper-rectangles. Fig. 4 shows the rectangles for the classifier in Fig. 2. If our classification is based on d attributes ($d = 2$ in our example) then a d-dimensional area defined by $p \in P_i^r$ will include objects classified as relevant (e.g., areas A and D in Fig. 4). Similarly, objects in an area defined by $p \in P_i^{nr}$ are classified as irrelevant (e.g., areas B and C in Fig. 4).

AIDE eliminates irrelevant attributes from the decision tree classifier by *domain sampling* around the boundaries. Specifically, while we shrink/expand one dimension of a relevant area we collect random samples over the *whole* domain of the remaining dimensions. Fig. 4 demonstrates our technique: while the samples we collect are within the range $11 \leq dosage \leq 9$ they are randomly distributed on the domain of the age dimension.

Our evaluation showed that this phase has the smallest impact on the effectiveness of our model: not discovering a relevant area can reduce our accuracy more than a partially discovered relevant area with imprecise boundaries. Hence, we constrain the number of samples used during this phase to α. This allows us to better utilize the user effort as he will provide feedback mostly on samples generated from the previous two, more effective phases.

Let us assume the decision tree has revealed k d-dimensional relevant areas. Each area has 2^d boundaries. Hence we collect $\alpha/(k \times 2^d)$ random samples within a normalized distance $\pm x$ from each boundary. This approach is applied across all the boundaries of the relevant hyper-rectangles, allowing us to shrink/expand each dimension of the relevant areas. The new collected samples, once labeled by the user, will increase the recall metric: they will discover more relevant tuples (if they exist) and eventually refine the boundaries of the relevant areas.

The x parameter can affect the number of samples needed to converge to the real relevant boundary. If the difference between the predicted and real boundaries is less than x, this phase will retrieve both relevant and irrelevant samples around the boundary and allow the decision tree to more accurately predict the real boundary of the relevant area. Otherwise, we will mostly collect relevant samples since the sampling area will not overlap with the relevant area. This will swill increase the number of samples we will need to converge to an accurate boundary.

This phase includes a number of further optimizations, such as detecting and avoiding sampling overlapping areas as well as adjusting the number of samples to the convergence rate of the user model. These optimizations improve AIDE's effectiveness and efficiency and they are described in detail in [7].

4 Performance Optimizations

In this section we describe a set of novel optimizations we introduced in AIDE. These include techniques that: (a) handle exploration on skewed data distributions, (b) leverage the informativeness of samples to improve AIDE's effectiveness, (c) extend the expressiveness of the user feedback model to accelerate the convergence to an accurate model and (d) reduce the size of our exploration space to offer highly interactive times. We note that the first three techniques are new optimizations that we added to the original version of AIDE introduced in [7], [8].

4.1 Skew-Aware Exploration

Skewed data distributions are prevalent in virtually every scientific domain of science. In our framework, skewed data distributions could hinder the discovery of relevant objects due to the fact that our initial exploration step (Section 3.1) distributes the number of collected samples evenly across the data space. In the presence of skew, this approach slows the convergence to an accurate user model, since dense areas will be under-sampled compared with the sparse ones. To address this challenge we introduce a new sampling technique designed to operate effectively on both uniform and skewed data distributions.

Our *hybrid* approach combines the grid-based exploration with a clustering-based sampling that identifies dense areas and increases the sampling effort within them. This clustering-based approach operates on multiple exploration levels. For a given exploration level with k clusters, we cluster all data in the dataspace using the k-means algorithm [3]. By default our highest level creates a single cluster and each level doubles the number of clusters of its previous one. The clustering is performed offline and these exploration levels can be used by all users.

AIDE maintains also its grid-based exploration levels as described in Section 3.1. For uniform distributions, the cluster-based and the grid-based sampling areas overlap. In this case, sampling within each grid cell as described in Section 3.1 is sufficient to discover relevant areas. However, in the presence of skewed exploration domains most of the clusters will be concentrated to dense areas leaving sparse areas under-sampled. Maintaining our grid-based sampling areas allows us to sample also sparse sub-areas and discover relevant areas of low density.

Our hybrid approach starts by sampling at the highest exploration level (i.e., with one cluster and one grid cell) and moves on at each iteration to the next lower level until the user terminates the exploration. At each level it samples dense areas by collecting one random sample within each cluster of that level. Next, it samples sparse sub-areas by

retrieving one random sample within specific grid cells of the same exploration level. These are the non-empty grid cells from within which no sample has been retrieved yet.

The user is presented with the samples collected by both the grid-based and the cluster-based sampling areas. This hybrid approach allows us to adjust our sample size to the skewness of our exploration space (i.e., we collect more samples from dense sub-areas) while it ensures that any sparse relevant areas will not be missed (i.e., sparse sub-areas are sufficiently explored).

4.2 Probabilistic Sampling

AIDE relies on a pool-based active learning paradigm for discovering user interests, i.e., samples are picked from a pool of unlabeled data objects and presented to the user for labeling. Existing pool-based sampling strategies [5] exhaustively examine *all* unlabeled objects available, searching for the best sample to show to the user. Clearly, such an approach cannot scale on big datasets. AIDE eliminates this exhaustive approach by *randomly* sampling a small number of strategically selected sub-areas in the exploration space.

Random sampling is highly effective. Especially in the boundary exploitation step, random sampling distributes the samples across the whole domain of our exploration attributes which eliminates irrelevant attributes from the classifier (see Section 3.3). However, it suffers from certain limitations. In particular, in the misclassified exploitation phase, random sampling treats each sample uniformly and it does not leverage the informativeness of the samples, which could potentially lead faster to an accurate user model. In other words, random sampling does not answer the question "which candidate samples to show to the user in order to reduce the total number of labeled samples needed for learning". To address this question, AIDE includes a new *probabilistic sampling* strategy for the misclassified exploitation phase.

Active learning has proposed a number of sample selection approaches that evaluate the informativeness of unlabeled samples [5]. In all these strategies the informativeness of a sample (e.g., the probability of being relevant or not) is either generated from scratch or sampled from a known distribution. In AIDE, we do not assume any distribution of relevant/irrelevant object. Instead we leverage the user's relevance feedback to calculate for each unlabeled object its informativeness, i.e., its probability of being labeled as relevant or irrelevant (aka posterior probability). Given this probability, we use the *uncertainty sampling* strategy to identify the next set of samples to show to the user.

We now discuss how to evaluate the posterior probability of unlabeled samples, given a set of relevant samples S^+ and irrelevant samples S^-. AIDE considers each labeled sample as basis for a nearest neighbour classifier with only one training sample and considers each unlabeled object to be a test example that has to be classified into the relevant or non-relevant class. We then combine these classifiers in order to "blend" information from all the user's collected feedback [9], [10].

Formally, given a sample labeled as relevant by the user s_+, the probability that an unlabeled sample x is relevant (r) is

$$p_x(r|s_+) \propto \exp(-similarity(x, s_+)),$$

where $similarity(x, s_+)$ returns the similarity value between x to s_+. Intuitively, this formula indicates that the probability of a sample x being relevant increases exponentially with its similarity to the relevant sample s_+. This is in accordance to our former argument that relevant samples will be clustered together in the exploration space and will be forming relevant areas.

Analogously, we assume that the probability of a sample x being non-relevant (n) increases exponentially when the sample is similar to a sample s_- labeled as non-relevant by the user

$$p_x(n|s_-) \propto \exp(-similarity(x, s_-)).$$

To calculate the posterior probability of a sample x being relevant, we combine the individual classifiers from the set of relevant samples S^+ and the set of irrelevant samples S_- by using the sum rule [10]. Specifically, given that $p_x(r \mid s_+) = 1 - p_x(n \mid s_-)$,

$$p_x(r|(S^+, S^-)) = \frac{\alpha}{|S^+|} \sum_{s_+ \in S^+} p_x(r|s_+) + \frac{1-\alpha}{|S^-|} \sum_{s_- \in S^-} 1 - p_x(n|s_-),$$

where α is a weighting factor we added to allow us to change the impact of the relevant and non-relevant samples. In the above formula if $\alpha = 1$ we only take into account its distance from the set of relevance samples to calculate its posterior probability. In the opposite case if $\alpha = 0$ we only consider its distance from the set of samples that are labeled as non-relevant.

Given the posterior probability of a sample, we use the *uncertainty sampling* strategy to select which samples to show to the user [5]. In uncertainty sampling the user is presented with samples for which the classifier is the most uncertain about. When using a binary classification model, like in our case, uncertainty sampling selects the sample whose posterior probability of being positive is nearest to 0.5 [5]. These are the samples that we are the least certain about their relevance.

We apply this approach in our misclassified exploitation phase as follows. Our sampling areas are defined around the clusters of misclassified we have identified. Specifically, given a cluster of size c we retrieve all samples within a distance y_i from the farthest cluster member in each dimension i. Next, we calculate the posterior probability for each of these samples and we present to the user $f \times c$ samples whose probability is closest to 0.5, where f (see Section 3.2 on how f can be set). Employing this technique allows us to discover the user's relevant area with less labeled samples proving the hypothesis that some samples are more informative than others.

4.3 Similarity Feedback Model

In our previous paragraphs we introduced exploration techniques that rely on binary relevance feedback, i.e., the user indicates whether the sample is relevant or not to her exploration task. However, there exist numerous scenarios where although the user cannot decidedly classify the relevance of an object, she can indicate whether this object is "close" to her interests. This label can be used when the user finds relevant some characteristics of the object but not necessarily all of them or if she is still uncertain about the relevance of

the object, which is often the case when the user is unfamiliar with the underlying data set.

Let us consider the case of a scientist exploring an astronomical dataset searching for clusters of sky objects with unusually high brightness. Initially, the user will be able to label star objects with high brightness values as potentially interesting. However, her understanding of which brightness values are in fact unusual crystallizes only after she has examined numerous sky objects of various brightness values. After that point she can identify unusually bright sky objects and label them as relevant. In another example, medical professionals searching for clinical trials for diabetes type A on two year old children can indicate that studies on diabetes type B on three year old children are also of possible interest to her (e.g., since the symptoms, medication and side effects for two and three year old children can be quite similar). However, she will label as relevant only trials on two year olds.

In the technical level, using a binary feedback model imposes a number of limitations to AIDE. In the previous example let's assume the user labels trials on three year old children as relevant (since it is close to the age of the actual patient). This will lead to a less accurate classification model (e.g., AIDE will steer the exploration to studies on three year olds). While the user can modify this label in subsequent iterations, this will slow the convergence of the exploration to an effective classification model. On the other hand, labeling these trials as irrelevant does not capture the similarity of these trials to the actual relevant objects (e.g., studies on three years are closer in the exploration space to the relevant trials than studies on 10 year olds). This similarity information, if expressed, could lead AIDE to focus its exploration on small ages and converge to an accurate model with less user effort.

Furthermore, the similarity feedback could help improve AIDE's efficiency when predicting small areas of interest. In our current approach, the smaller the relevant area we aim to predict, the higher user effort (i.e., number of labeled samples) is required. This is a practical challenge especially when the relevant objects are clustered within very small areas in the exploration space. The smaller the relevant area the more zoom-in operations AIDE will execute in order to discover a relevant sample from within that area (Section 3.1). These operations result in sampling more areas (i.e., grid cells), which increases the user effort as well as the number of sampling queries processed. A more expressive feedback model that allows users to indicate that a sample is "close" to a relevant object could help us direct our zoom-in operations to only promising sub-areas of the exploration space. This will lead to an accurate user model with less user effort and exploration overhead.

To address the above challenges, we extended our user feedback model as follows. Users can indicate that an object is "close" to her interests by annotating it as a "similar" sample. This label should be used for samples with at least one attribute value that appears interesting or similar ("close") to a relevant value. The user has the option to indicate these attributes, i.e., the dimension on which she found the sample to be interesting (e.g., age range in the above medical example, brightness in the scientific example). The system can then utilize this extra information to expedite the exploration process. We note that our "similarity" annotations do not constitute a new label for our classification

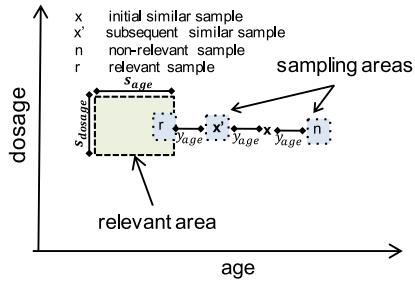


Fig. 5. Similarity feedback exploration.

model, i.e., our decision tree classifier will continue to generate classification rules that predict only the relevant and irrelevant classes. Next, we describe our technique.

Extended Feedback Exploration. We introduce one more exploration phase that defines sampling areas around each "similar" sample. Based on the definition of this label, each such sample x is potentially "close" to a relevant object in at least one of the exploration dimensions. AIDE by default assumes that this similarity may be present in all dimensions unless the user explicitly indicates for which dimensions she discovered similar values. We refer to these as the *interesting dimensions*.

Let us assume a sample x annotated as "similar' across a set of interesting dimensions d (which are a subset of the set of exploration dimensions). AIDE explores all possible interesting dimensions around x on the d dimensional space aiming to identify relevant samples. Specifically, there are 2^d possible exploration directions around the sample, i.e., for each dimension we explore both higher and lower values of the x's value on this dimension. Hence, we define 2^d sampling areas and we select one random sample close to the center of each area to present to the user. In Fig. 5 we show a scenario of a 2-dimensional exploration space (age and dosage from our medical example), where the user has indicated as interesting only a single dimension (age). Hence, we have created two sampling areas around the sample x and we have selected one sample within each of these areas.

We define the sampling areas to be located in a distance $\pm\gamma_i$ from the "similar" sample x in each interesting dimension i. If one of the new samples we present to the user is now closer to the relevant area we can expect that the user will annotate it as a "similar" sample too. In the opposite case, we assume that the user will naturally be dissatisfied with these samples and will label them as non-relevant. Eventually one of the sampling areas will overlap with the relevant area and the user will label the sample we extract from that area as relevant. Hence, sampling in a distance γ_i on each dimension i from x bring us closer or inside the relevant area. In Fig. 5, let's assume that x is a study on five years olds. If the patient's age is 3 then samples with lower age groups (e.g., sample x') will be also annotated as "similar" while samples with higher age groups (e.g., sample n) will be irrelevant.

The effectiveness of our γ_i value correlates with the range (percentage of the normalized domain) the relevant areas s_i cover in each dimension i (see Fig. 5). Let us assume $\gamma_i \leq s_i$ for some dimension i. Then in the next iteration we will sample either: a) within the relevant range in dimension i or b) closer to that relevant range compared with the previous iteration. The first case leads directly to the relevant area. In the second case we guarantee that we will "hit" the relevant

range in that dimension in d_i/γ_i iterations (i.e, after $d_i/\gamma_i - 1$ "similar" sample annotations), where d_i is the distance of the sample x from the relevant range in dimension i. In the opposite case where $\gamma_i > s_i$ we might move towards the relevant area but miss the area altogether; intuitively, our "step" is so large that we "jump" over the relevant range and never sample within it. In this case we expect the user to label the new samples we will present to her as non-relevant samples since our sampling areas are fending away from the relevant area instead of approaching it. AIDE detects this scenario and restarts this exploration phase from the original x sample but a lower γ_i value for that dimension i. Using this pattern, we keep adapting our γ_i value until we "hit" a relevant sample.

4.4 Exploration Space Reduction

Our exploration techniques rely on sending a sampling query to the back end database system for each defined sampling area. Such queries can be particularly expensive. This is especially true for the sampling queries generated by the boundary exploitation phase since they need to fully scan the whole domain of all attributes. Even when covering indexes are used to prevent access to disk, the whole index needs to be read for every query, increasing the sampling extraction overhead.

An interesting artifact of our exploration techniques is that their effectiveness does not depend on the frequency of each attribute value, or on the presence of all available tuples of our database. This is because each phase executes *random* selections within data hyper-rectangles and hence these selections do not need to be deterministic. Hence, as long as the domain value distribution within these hyper-rectangles is roughly preserved, our techniques are still equally effective. This observation allows to apply our exploration on a sampled exploration space. Specifically, we generate our sampled data sets using a simple random sampling approach that picks each tuple with the same probability [11]. We then execute our exploration on this smaller sampled space. Since this data space maintains the same value distribution of the underlying attribute domains, our approach offers a similar level of accuracy but with significantly less time overhead.

5 Experimental Evaluation

Next, we present experimental results from a micro-benchmark on the SDSS dataset [2] and from a user study.

5.1 Experimental Setup: SDSS Dataset

We implemented our framework on JVM 1.7. In our experiments we used various Sloan Digital Sky Survey datasets (SDSS) [2] with a size of 10 GB-100 GB ($3 \times 10^6 - 30 \times 10^6$ tuples). Our exploration was performed on combinations of 16 numerical attributes of the `PhotoObjAll` table with different value distributions. This allowed us to experiment with both skewed and roughly uniform exploration spaces. A covering index on these attributes was always used. We used by default a 10 GB dataset and a uniform exploration space on `rowc` and `colc`, unless otherwise noted. All experiments were run on an Intel PowerEdge R320 server with 32 GB RAM using MySQL. We used Weka [12] for executing the CART [3] decision tree and the k-means algorithms. All experiments report averages of ten runs.

Target Queries. AIDE "predicts" the selection predicates that retrieve the user's relevant objects. We focus on predicting the results of range queries (we call them *target queries*) and we vary their complexity based on: a) the number of disjunctive predicates they include (*number of relevant areas*) and b) the data space coverage of the relevant areas, i.e., the width of the range for each attribute (*relevant area size*). Specifically, we categorize relevant areas to *small*, *medium* and *large*. Small areas have attribute ranges with average width of 1-3 percent of their normalized domain, while medium areas have width 4-6 percent and large ones have 7-9 percent. We also experimented with queries with a single relevant area (conjunctive queries) as well as complex disjunctive queries that select 3, 5 and 7 relevant areas. The higher the number of relevant areas and the smaller these areas, the more challenging is to predict them.

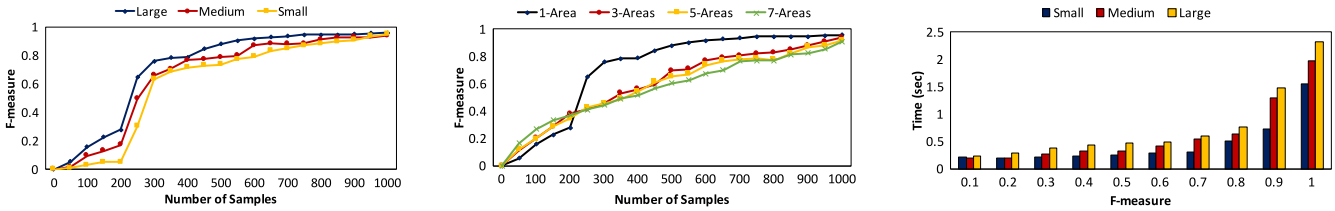
The diversity of our target query set is driven by the query characteristics we observed in the SDSS sample query set [13]. Specifically, 90 percent of their queries select a single area, while 10 percent select only 4 areas. Our experiments cover even more complex cases of 5 and 7 areas. Furthermore, 20 percent of the predicates used in SDSS queries cover 1-3.5 percent of their domain, 3 percent of them have coverage around 13, and 50 percent of the predicates have coverage 50 percent or higher while the median coverage is 3.4 percent. Our target queries have domain coverage (i.e., the relevant area size) between 1-9 percent and our results demonstrate that we perform better as the size of the areas increases. Hence, we believe that our query set has a good coverage of queries used in real-world applications while they also cover significantly more complex cases.

User Simulation. Given a target query, we simulate the user by executing the query to collect the exact *target set*of relevant tuples. We rely on this set to label the new sample set we extract in each iteration as relevant or irrelevant depending on whether they are included in the target set. We also use this set to evaluate the accuracy (F-measure) of our final predicted extraction queries.

Evaluation Metrics. We measure the accuracy of our approach using the F-measure (Section 2.3) of our final data extraction predictions and report the number of labeled samples required to reach a given accuracy level. For all our experiments we aim for accuracy 90 percent or higher unless stated otherwise. Our efficiency metric is the *system execution time* (equivalent to *user wait time*), which includes the time for the space exploration, data classification, and sample extraction. We may also report the total *exploration time*, which includes both the system execution time and the sample reviewing time by the user.

System Parameters. To understand the impact of the parameters used by our heuristics we conducted a sensitivity study. Next we discuss these parameters and the default values we used.

The α parameter is the total number of samples collected in the boundary phase. Our study showed that collecting at least two samples for each boundary is sufficient, hence for a d-dimensional exploration space we allocate $\alpha = 2 \times d$ samples for this phase. Even if the relevant dimensions are less than d the extra sample size will be small relatively to the number of samples collected through the rest of the exploration phases.



(a) Accuracy for increasing area size (1 area). (b) Accuracy for increasing number of areas (large areas). (c) Time per iteration for increasing area size (1 area).

Fig. 6. (a) and (b) AIDE's effectiveness, i.e., prediction accuracy, is shown. (c) Efficiency results, i.e., time overhead, are shown.

The x parameter is the sample distance around each boundary and ideally it should be set such that the sampling area overlaps with the relevant area (see Section 3.3). We observed that our decision trees approximate the correct boundary very well within a few iterations. Hence, we set x to 0.06 percent which lead us to sample within the relevant areas and improved our convergence rate. Note that our study revealed that setting this value between 1 to 10 percent increased the total number of samples by no more than 50 samples when running AIDE to predict areas of small, medium or large size with 90 percent accuracy. Hence, AIDE is quite robust to this parameter.

The f parameter is the number of samples we collect around each misclassified or clusters of misclassfied object and y_i is the sampling distance on dimension i around each misclassified object. Both of them can be adjusted dynamically as their ideal value depends on the size of the relevant area (see Section 3.2). Based on our sensitivity study on the relevant areas we are using, we set f to 15 samples and y_i to 2 percent of the normalized domain. These allows AIDE to collect the necessary relevant samples to bootstrap our user model within fewer iterations.

5.2 Effectiveness and Efficiency of AIDE

Fig. 6a shows AIDE's effectiveness when we increase the query complexity by varying the size of relevant areas from *Small* to *Medium* and *Large*. Our queries have one relevant area which is the most common range query in SSDS. Naturally, labeling more samples improves in all cases the accuracy. As query complexity increases the user needs to label more samples to reach a given accuracy level. By requesting feedback on only 247 out of 3×10^6 objects AIDE predicts large relevant areas with accuracy higher than 60 percent (with 386 samples we have an accuracy higher than 80 percent). In this case, the user needs to label only 0.4 percent of the total relevant objects and 0.01 percent of the irrelevant objects in the database. Furthermore, AIDE needs only 300 labeled samples to predict medium areas with 66 percent accuracy and small areas with 63 percent accuracy. *Hence, AIDE decreases the user effort (i.e., reviewing objects) to a few 100's samples compared with the state-of-the-art "manual" exploration which involves examining 1,000's of objects (e.g., target queries return 26,817-99,671 relevant objects depending on the size of the relevant areas).*

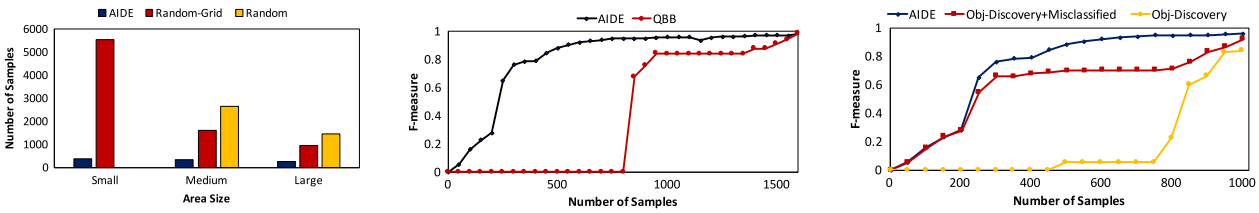
We also increased the query complexity by varying the number of areas from one (1) to seven (7). Fig. 6b shows our results for the case of large relevant areas. While AIDE can perform very well for common conjunctive queries (i.e., with one (1) relevant area), to accurately predict highly complex disjunctive queries more samples are needed. However, even for highly complex queries of seven (7) areas we get an accuracy of 60percent or higher with a reasonable number of samples (500 labeled samples).

Fig. 6c shows the user's wait time (seconds in average per iteration). In all cases, high accuracy requires the extraction of more samples which increases the exploration time. The complexity of the query (size of relevant areas) also affects the time overhead. Searching for larger relevant areas leads to more sample extraction queries around the boundaries of these relevant areas. However, our time overhead is acceptable: to get an accuracy of 60 percent the user wait time per iteration is less than 0.55 seconds for all area sizes, while to get highly accurate predictions (90 percent-100 percent) the user experiences 1.7 seconds wait time in average.

Comparison with Random Sampling. Next, we compared AIDE with two exploration techniques that rely on random sampling. *Random* selects randomly 20 samples per iteration, presents them to the user for feedback and then builds a classification model. *Random-Grid* uses our exploration exploration grid (Section 3.1) and selects one random sample within each grid cell, i.e., it collect samples that are evenly distributed across the exploration space. This approach also collects 20 samples per iteration. For comparison reasons, AIDE also limits the number of new samples it extracts per iteration: we sum the number of samples needed for the boundary and the misclassified exploitation and we use the remaining out of 20 samples to sample grid cells.

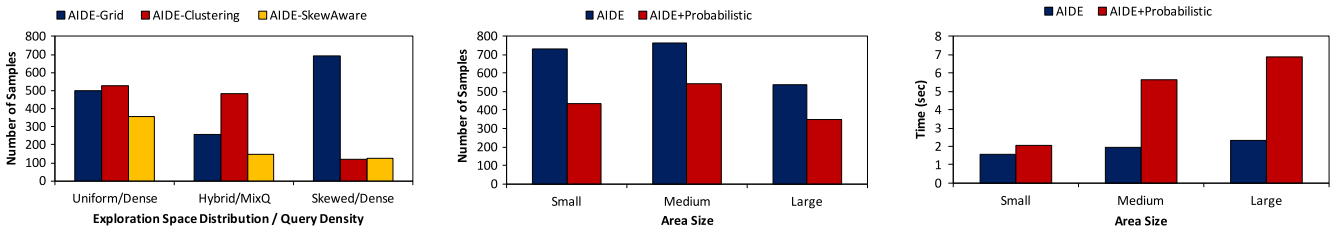
Fig. 7a shows the number of samples needed to achieve an accuracy of at least 70percent when our target queries have one relevant area of varying size. For this experiment, we set F-measure to 70percent because *Random-Grid* and *Random* cannot converge to accuracy higher than 70percent for most area types given a reasonable number of samples (less than 6,000). AIDE is consistently highly effective: it requires no more than 373 samples for any area size always outperforming the baselines. *Random* fails to discover small areas of interest even when we increase the labeled set to 6,000 samples, while *Random-Grid* needs 5,520 samples in average. For medium and large areas *Random* and *Random-Grid* are still highly ineffective compared with AIDE.

Comparison with Active Learning. Fig. 7b compares the prediction accuracy of AIDE with Query By Bagging (*QBB*) [14], an active learning technique for decision tree classifiers. The results are for a single large area. Given a new set of labeled objects at each round, QBB creates an ensemble of decision trees on different training sets generated through sample replacement. It then examines all database objects to select for labeling the one on which these classifiers disagree most. QBB assumes a training set is provided a-priori with sufficient relevant/irrelevant samples to bootstrap the generation of the decision trees. Aiming to support real-life exploration scenarios, AIDE does not have this assumption. Hence, we modified



(a) Comparison to random exploration (1 area). (b) Comparison to Query by Bagging (1 large area). (c) Impact of exploration phases (1 large area).

Fig. 7. (a) and (b) compare AIDE with other exploration techniques while (c) demonstrates the effectiveness of the exploration phases.



(a) Accuracy for skewed data sets (1 large area). (b) Accuracy of probabilistic sampling (1 area). (c) Time for probabilistic sampling (1 area).

Fig. 8. Impact of performance optimizations: (a) evaluation of the skew-aware exploration. (b) and (c) Evaluation of the probabilistic sampling.

QBB to use AIDE's object discovery phase to collect its initial labeled sample set. Fig. 7b reveals that AIDE performs better than QBB, i.e., converges to a high accuracy with less samples than QBB. This is because AIDE generates the initial training set more strategically, e.g., after hitting the first few relevant samples, the misclassified exploitation phase discovers enough relevant objects to boostrap the generation of its decision tree and in the following steps the boundary exploitation improves the accuracy of that tree with very few samples. Furthermore, QBB requires a much higher exploration time than AIDE, as it needs to examine all database objects in each iteration in order to decide which one to present to the user.

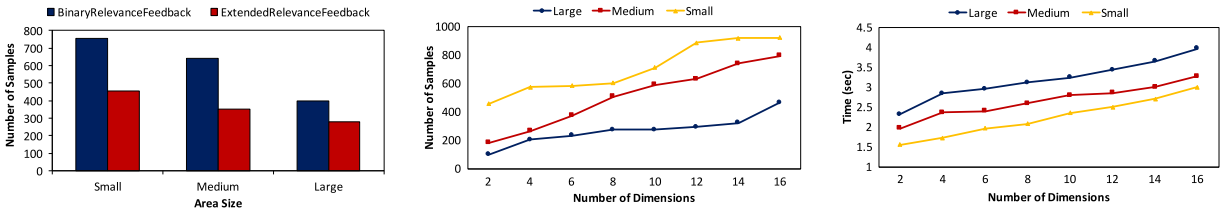
Impact of Exploration Phases. We also studied the impact of each exploration phase independently. Fig. 7c presents the number of samples we need to reach different accuracy levels for queries with one large relevant area. We compare AIDE with two variants: one that uses only the object discovery phase (*Obj-Discovery*) and one that adds only the misclassified exploitation phase (*Obj-Discovery+Misclassified*). The results show that combining all three phases of AIDE gives the best results, i.e, better accuracy with fewer labeled samples. Specifically, using only the object discovery phase requires more than 800 labeled samples to reach an accuracy higher than 20 percent. Adding the misclassified exploitation to the object discovery phase increases the accuracy by an average of 54 percent. Finally, adding the boundary exploitation phase further improves the accuracy by an average of 15 percent. *Hence, combining all three phases is highly effective in predicting relevant areas while reducing the amount of user effort.*

5.3 Skewed Exploration Spaces

We also studied AIDE in the presence of skewed exploration spaces. We experimented with three types of 2-dimensional exploration spaces: (a) *Uniform* where we use two roughly uniform domains (`rowc`, `colc`), (b) *Hybrid* that includes one skewed (`dec`) and one uniform domain (`rowc`) and (c) *Skewed* that uses two skewed domains (`dec`, `ra`). We also experimented with the density of the target queries: (a) *Dense* queries involve dense relevant areas and (b) *MixQ* queries cover both sparse and dense ranges of the relevant domains. Fig. 8a shows the number of samples needed to achieve accuracy greater than 90 percent for queries with one large relevant area. We compare three variants of our system: (a) AIDE-Grid that uses the grid-based technique for the relevant object discovery phase, (b) *AIDE-Clustering* that uses only clustering-based sampled for skewed distributions but not sampling within grid cells and (c) *AIDE-SkewAware* that is a hybrid of the two previous techniques as described in Section 4.1.

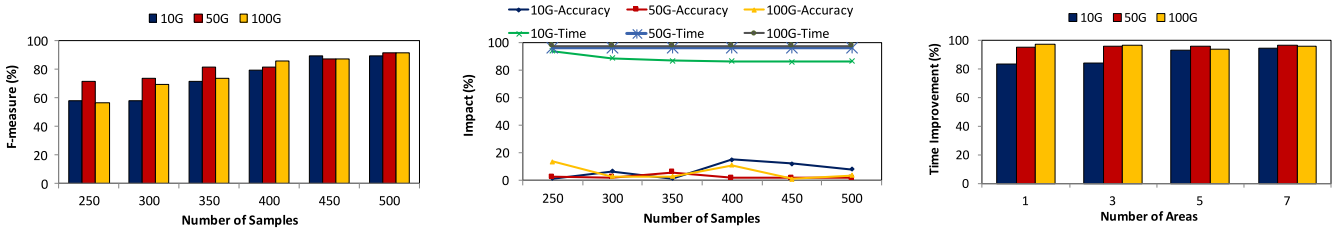
The results show that AIDE-SkewAware works best under any combination of query density and exploration space distribution. When the distribution is uniform (Uniform) clusters and grid cells are highly aligned providing roughly the same results for all three techniques. Note that in this case all our relevant areas will be dense. In the highly skewed data space (Skewed) we also used only dense relevant areas as the sparse areas were practically non populated. Here, both the clustering-based technique and the skew-aware technique outperform the grid-based approach requiring 82 percent less samples. This is because clusters are formulated in the dense sub-space while grid cells are created uniformly across the data space covering non populated exploration areas. This allows AIDE-Clustering and AIDE-SkewAware to sample smaller, finer-grained areas than the grid-based approach, eliminating the need to zoom into the next exploration level.

Finally, for the case of hybrid distributions (Hybrid) we picked our relevant area to cover both dense ranges (for the uniform domain) and sparse ranges in the skewed domain, resulting to our mixed query case (MixQ). Here, the clustering technique creates most of its clusters on the dense areas and hence fails to discover relevant objects in the sparse ones. It therefore has to zoom into finer exploration levels and it requires 47percent more samples to converge to the same accuracy as the grid-based technique. However, AIDE-SkewAware samples both the dense areas where the clusters are located and the sparse areas which are covered by the grid cell and it discovers the relevant area. *We conclude that combining sampling within clusters and grid cells is the best strategy for exploring both skewed and non skewed domains.*



(a) Accuracy for extended feedback model. (b) Accuracy for multi-dimensional exploration spaces. (c) Time for multi-dimensional exploration spaces.

Fig. 9. (a) Impact of similarity feedback model (1 area). (b) and (c) Evaluation for multi-dimensional exploration spaces (1 area).



(a) Accuracy for increasing data set size (1 area). (b) Impact of space reduction on accuracy and time. (c) Time improvement of sampled datasets.

Fig. 10. (a) AIDE's effectiveness on big data sets is shown and (b) and (c) show the impact of our exploration space reduction.

5.4 Probabilistic Sampling

Next, we examine the effectiveness and efficiency of the probabilistic sampling technique (Section 4.2). In Fig. 8b we measure the number of samples needed to reach an F-measure greater than 90 percent when using the probabilistic sampling technique in the misclassified exploitation phase (*AIDE+Probabilistic*). The figure presents the results for small, medium and large areas. *The results show that AIDE requires less labeled samples to reach an accuracy when using the probabilistic sampling strategy*. In average this new approach can reduce the user effort by 35 percent. This confirms our hypothesis that some samples in the misclassified sampling areas are more informative than others and they can be leveraged to improve the user's experience.

We also studied the overhead of this approach. Fig. 8c shows that the uncertainty sampling technique increases our user wait time per iteration in all cases. This is because in each iteration we have to extract all samples within the sampling area and for each sample calculate its posterior probability and decide whether to present it to the user or not. As a result, the user wait time per iteration was increased by 50 percent in average. However, the time overhead was less than 2.9 seconds in average which should not affect the user's interactive experience.

5.5 Similarity Feedback Model

We also studied the effect of extending our relevance feedback model to include labels for similar but not necessarily relevant samples. Here, we label as "similar" the samples that are within a distance less than 10 percent from an actual relevant object (this distance is measured in any of the exploration dimensions). Otherwise, we label the sample as irrelevant. Fig. 9a presents AIDE's effectiveness when using the binary feedback approach and the extended feedback model. Here, we vary the size of the target relevant area from small to medium and large and we measure the number of samples AIDE needs to reach an F-measure higher than 90 percent. *The results indicate that annotating the similarity of objects can significantly reduce the labeling effort of the user*. This improvement is 38 percent in average across all area sizes. This feedback is particularly useful in the case of the small relevant areas where the user effort can be significant. Here, the user's "similar" labels steer the exploration towards the direction of the relevant area and the labeling effort is significantly reduced. We also measured the impact of this model on the user wait time and in all cases it was under 0.1 seconds which should be unnoticeable by the user. We omit the graph due to space limitations.

5.6 Scalability

Exploration Space Dimensionality. Fig. 9b shows the number of labeled samples required to reach more than 90 percent accuracy as we increase the dimensions of a skewed exploration space from 2 up to 16 and vary the size of the relevant areas. Our target queries have conjunctions on two attributes. The graph reveals that the labeling overhead increases linearly to the number of dimensions. This is because more dimensions increase the number of sampling areas for the object discovery phase and boundary phase (more attributes/boundaries appear in the decision tree split rules). However, even with 16 dimensions AIDE needs only 923 samples to reach more than 90 percent accuracy. We anticipate that in real scenarios the dimensionality of the data space will be significantly less. As an example, 1.8 million SDSS queries collected in April 2016 revealed that 54 percent of user queries include less than 4 dimensions. Fig. 9c shows that even with 16 dimensions the user wait-time per iteration is always less than 4.1 seconds for all area sizes. The results reveal a small increase in the user's wait time as we add more dimensions.

Database Size. Fig. 10a shows AIDE's accuracy with a given number of labeled samples for dataset sizes of 10 GB, 50 GB and 100 GB. Our target queries have one large relevant area and the average number of relevant objects increases as we increase the size of the dataset (our target query returns in average 26,817 relevant objects in the 10 GB, 120,136 objects in the 50 GB and 238,898 objects in the 100 GB database). AIDE predicts these objects in all datasets with high accuracy without increasing the user's effort. *We conclude that the size of the database does not affect our effectiveness*. AIDE consistently achieves high accuracy of more than 80 percent on big data sets with only a few hundred samples

(e.g., 400 samples). These results were consistent even for more complex queries with multiple relevant areas.

Exploration Space Reduction. Applying our techniques to larger datasets increases the time overhead since our sampling queries have higher response times. One optimization is to execute our exploration on a sampled database (Section 4.4). In this experiment, we sampled datasets of 10 GB, 50 GB, 100 GB and generated the 10 percent sampled datasets of 1 GB, 5 GB and 10 GB, respectively. Fig. 10b shows the absolute difference of the final accuracy (*10 GB-Accuracy*, *50 GB-Accuracy*, *100 GB-Accuracy*) when AIDE is applied on the sampled and on the total datasets. The average difference is no more than 7.15 percent for the 10GB, 2.72 percent for the 50 GB and 5.85 percent for the 100 GB data set. In the same figure we also show the improvement of the system execution time (*10 GB-Time*, *50 GB-Time*, *100 GB-Time*). For 10 GB (and a sampled dataset of 1 GB) this time is reduced by 88 percent in average, while for the larger datasets of 50 GB and 100 GB it is reduced by 96-97percent.

Fig. 10c shows the improvement of the system execution time when AIDE runs over the sampled data sets. Here, we measure the system execution time to reach an accuracy higher than 90 percent and for varying number of large areas. The average time per iteration is 2.8 seconds for the 10 GB, 37.7 for the 50 GB and 111 for the 100 GB database. By operating on the sampled datasets we improved our time by more than 84 percent while our average improvement for each query type was more than 91 percent. Our improved iteration time is 0.37 second for the 10 GB, 2.14 seconds for the 50 GB and 5.3 seconds for the 100 GB dataset, in average. *Hence, AIDE can scale to big datasets by applying its techniques on sampled datasets. This incurs low impact on the accuracy while it significantly improves the system execution time.*

5.7 User Study Evaluation

Our user study used the AuctionMark dataset [15] that includes information on auction items and their bids. We chose this "intuitive" dataset, as opposed to the SDSS dataset, because the user study requires identifying users with sufficient understanding of the domain. We identified a group of graduate students with SQL experience and designed their exploration task to be "identifying auction items that are good deals". Note that users should not have an upfront understanding of the exact selection predicates that would collect all relevant objects.

The exploration data set had a size of 1.77 GB and it was derived from the ITEM table of AuctionMark benchmark. It included seven attributes: initial price, current price, number of bids, number of comments, number of days an item is in an auction, the difference between the initial and current item price, and the days until the auction is closed for that item. Each user explored the data set "manually", i.e., iteratively formulating exploratory queries and reviewing their results until he obtained a query, Q, that satisfied his interests. In each iteration, i, we recorded (a) the number of objects, o_i, returned by the user query and (b) the time the user spent reviewing those objects, t_i. Thus, for each user we were able to calculate the average review time per object, $t = \frac{\sum^N t_i}{\sum^N o_i}$, where N is the total number of queries the user executed during his exploration. This time varied significantly by user depending on the time each one took to decide whether he was interested or not in the returned results of his manual queries.

TABLE 1
User Study Results

User	Manual: returned objects	Manual: reviewed objects	AIDE: reviewed objects	Reviewing savings (%)	Manual: time (min)	AIDE: time (min)
1	253,461	312	204.9	**34.3%**	60	39.7
2	656,880	160	82.4	**48.5%**	70	36.3
3	933,500	1,240	157	**87.3%**	60	7.9
4	180,907	600	319	**46.8%**	50	28.2
5	2,446,180	650	288.5	**55.6%**	60	27.5
6	1,467,708	750	334.5	**55.3%**	75	33.8
7	567,894	1,064	288.4	**72.8%**	90	24.8

Our user did not directly interact with AIDE. Instead we took the user's final query Q as the true interest of the user and used it to simulate the user. In other words, we labeled each new sample as relevant if the sample was included in the results of Q and as irrelevant otherwise. This guaranteed that objects received the same label in both the manual and AIDE's exploration.

We then measured how well AIDE can predict Q by simulating each user 10 times. For each run, we measured the number of labeled samples that AIDE needed in order to discover the results of Q with 100 percent accuracy. We report the average number of labeled samples during the 10 runs in Table 1. We also compared the total exploration time of the two techniques, where the exploration time consists of the system execution time (equivalent to the user wait time) and the object review time. For AIDE, since the actual review time was not available in simulation, we measured the review time as the average review time per object, collected from the manual exploration, multiplied by the number of samples that AIDE needed to reach 100 percent accuracy.

The results demonstrated that AIDE would be able to reduce the user's reviewing effort by 66 percent in average (*Reviewing savings* column in Table 1). Furthermore, with the manual exploration users were shown 100s of thousands objects in total (*Manual returned objects*) while AIDE shows them only a few hundred strategically selected samples. Furthermore, with the manual exploration our users needed about an hour to complete their task (*Manual time*). AIDE was able to reduce the exploration time 47 percent in average (*AIDE time*). We believe these time savings will be even more pronounced for more complex exploration tasks (e.g., in astronomical or medical domains) where examining the relevance of an object requires significant time.

Our user study revealed that five out of the seven users used only two attributes to characterize their interests. Similarly to our SDSS workload, the most common type of query was conjunctive queries that selected a single relevant area. Our exploration domain was highly skewed and all our relevant areas were on dense regions. These characteristics indicate that our micro-benchmark on the SDSS dataset was representative of common exploration tasks while it also covered highly more complex cases, i.e., small relevant areas and disjunctive queries selecting multiple areas.

6 Related Work

Query by Example. Related work on "Query-By-Example" (QBE) we originally proposed in [16]. Most recent

work includes querying knowledge graphs by example tuples [17], formulating join queries based on example output tuples [18] and inferring user queries by asking for feedback on database tuples [19], [20]. Finally, in [21] they learn user queries based on given value assignments used in the intended query. These systems provide alternative front-end query interfaces that assist the user formulate her query and do not attempt to understand user interests nor retrieve "similar" data objects which is AIDE's focus.

Data Exploration. Numerous recent research efforts focus on data exploration. The vision for automatic, interactive navigation in databases was first discussed in [22] and later on in [23]. YMALDB [24] supports data exploration by recommending to the user data similar to her query results. DICE [25] supports exploration of data cubes using faceted search and in [26] they propose a new "drill-down" operator for exploring and summarizing groups of tuples. SciBORQ [27] relies on hierarchical database samples to support scientific exploration queries within strict query execution times. Idreos et al. [28] envision a system for interactive data processing tasks aiming to reduce the time spent on data analysis. In [29] interactively explores the space based on statistical properties of the data and provides query suggestions for further exploration while in [30] they propose a technique for providing feedback during the query specification and eventually guiding the user towards her intended query. In [31] users rely on prefetching and incremental online processing to offer interactive exploration times for window-based queries. SearchLight [32] offers fast searching, mining and exploration of multidimensional data based on constraint programming. All the above systems are different than AIDE: we rely on the user's feedback on data samples to predict the user's data interests and we focus on identifying strategic sampling areas that allow for accurate predictions. In [33] the authors propose a system for faceted navigation of query results. This work uses a different feedback model than AIDE; the user provides feedback on facet conditions and not database samples and our optimization goal to reduce the number of samples does not apply on faceted search.

Query Relaxation. Query relaxation techniques have also been proposed for supporting exploration in databases [34]. In [35], [36] they refine SQL queries to satisfy cardinality constraints on the query result. In [37] they rely on multi-dimensional histograms and distance metrics for range queries for accurate query size estimation. These solutions are orthogonal to our problem; they focus on adjusting the query parameters to reach a cardinality goal and therefore cannot characterize user interests. In [38] they relax query conditions in order to obtain non-empty query results. This work employs a different feedback model than AIDE: the user rejects/accepts query modifications and not database samples. It is nontrivial, in fact technically challenging, to equate these two feedback models or transform one type of feedback to the other.

Active Learning. The active learning community has proposed solutions that maximize the learning outcome while minimizing the number of samples labeled by the user [6], [39]. However, these techniques assume either small datasets or negligible sample extraction costs which is not a valid assumption when datasets span 100s of GBs and interactive performance is expected. Relevance feedback have been studied for image retrieval [40], document ranking [41], information extraction and segmentation [42] and word disambiguation [43]. All these solutions are designed for specific data types (images or text) and do not optimize for efficient sample acquisition and data space exploration.

Collaborative and Interactive Systems. In [44] a collaborative system is proposed to facilitate formulation of SQL queries based on past queries and in [45] they use collaborative filtering to provide query recommendations. However, both these systems do not predict "similar" data object. In [46] they cluster related queries as a means of understanding the intents of a given user query. The focus is on web searches and not structured databases.

7 Conclusion

Interactive Data Exploration is a key ingredient of a diverse set of discovery-oriented application. In these applications, data discovery is a highly ad hoc interactive process where users execute numerous exploration queries using varying predicates aiming to balance the trade-off between collecting all relevant information and reducing the size of returned data. Therefore, there is a strong need to support these human-in-the-loop applications by assisting their navigation in the data space.

In this paper, we introduce AIDE, an *Automatic Interactive Data Exploration* system, that iteratively steers the user towards interesting data areas and "predicts" her objects of interest. Our approach leverages relevance feedback on database samples to model user interests and strategically collects more samples to refine the model while minimizing the user effort. AIDE integrates machine learning and data management techniques to provide effective data exploration results (matching the user's interests with high accuracy) as well as interactive performance (limiting the user wait time per iteration to less than a few seconds). Our experiments indicate that AIDE is a practical exploration framework as it significantly reduces the user effort and the total exploration time compared with the current state-of-the-art approach of manual exploration as well as traditional active learning techniques.

References

[1] Large Synoptic Survey Telescope. (2015). [Online]. Available: http://http:/www.lsst.org/
[2] Sloan Digital Sky Survey. (2015). [Online]. Available: http://www.sdss.org/
[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. London, U.K.: Chapman Hall/CRC, 1984.
[4] X. S. Zhou and T. Huang, "Relevance feedback in image retrieval: A comprehensive review," *Multimedia Syst.*, vol. 8, no. 2, pp. 95–145, 2003.
[5] B. Settles, "Active learning literature survey," *Synthesis Lectures Artificial Intell. Mach. Learn.*, vol. 6, no. 1, pp. 1–114, 2012.
[6] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 441–448.
[7] K. Dimitriadou, O. Papaemmanoui, and Y. Diao, "Explore-by-example: An automatic query steering framework for interactive data exploration," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 517–528.
[8] Y. Diao, et al., "AIDE: An automatic user navigation service for interactive data exploration (demo)," *Proc. VLDB Endowment*, vol. 8, no. 12, pp. 1964–1967 2015.
[9] T. Deselaers, R. Paredes, E. Vidal, and H. Ney, "Learning weighted distances for relevance feedback in image retrieval," in *Proc. 19th Int. Conf. Pattern Recognition*, 2008, pp. 1–4.
[10] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[11] F. Olken and D. Rotem, "Random sampling from databases - a survey," *Statistics Comput.*, vol. 5, no. 1, pp. 25–42, 1994.
[12] Weka: Data mining software in java. (2015). [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/
[13] *SDSS Sample Queries.* (2015). [Online]. Available: http://cas.sdss.org/dr4/en/help/docs/realquery.asp
[14] K. Dwyer and R. Holte, "Decision tree instability and active learning," in *Proc. 18th Eur. Conf. Mach. Learn.*, 2007, pp. 128–139.
[15] AuctionMark Benchmark. (2015). [Online]. Available: http://hstore.cs.brown.edu/projects/auctionmark/
[16] M. M. Zloof, "Query-by-example: the invocation and definition of tables and forms," in *Proc. 1st Int. Conf. Very Large Data Bases*, 1975, pp. 1–24.
[17] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas, "Exemplar queries: Give me an example of what you need," *Proc. VLDB Endowment*, vol. 7, pp. 365–376, 2014.
[18] Y. Shen, K. Chakrabarti, S. Chaudhuri, B. Ding, and L. Novik, "Discovering queries based on example tuples," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 493–504.
[19] H. Li, C. Chan, and D. Maier, "Query from examples: An iterative, data-driven approach to query construction," *Proc. VLDB Endowment*, vol. 8, pp. 2158–2169, 2015.
[20] A. Bonifati, R. Ciucanu, and S. Staworko, "Interactive inference of join queries," in *Proc. 17th Int. Conf. Extending Database Technol.*, 2014, pp. 451–462.
[21] A. Abouzied, D. Angluin, C. Papadimitriou, J. M. Hellerstein, and A. Silberschatz, "Learning and verifying quantified boolean queries by example," in *Proc. 32nd ACM SIGMOD-SIGACT-SIGAI Symp. Principles Database Syst.*, 2013, pp. 49–60.
[22] U. Çetintemel, et al., "Query steering for interactive data exploration," presented at the *6th Biennial Conf. Innovative Data Syst. Res.*, Asilomar, CA, USA, 2013.
[23] A. Wasay, M. Athanassoulis, and S. Idreos, "Queriosity: Automated data exploration," in *Proc. IEEE Int. Congr. Big Data*, 2015, pp. 716–719.
[24] M. Drosou and E. Pitoura, "YMALDB: Exploring relational databases via result-driven recommendations," *VLDB J.*, vol. 22, pp. 849–874, 2013.
[25] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi, "Distributed and interactive cube exploration," in *Proc. IEEE 30th Int. Conf. Data Eng.*, 2014, pp. 472–483.
[26] M. Joglekar, H. Garcia-Molina, and A. G. Parameswaran, "Smart drill-down: A new data exploration operator," *Proc. VLDB Endowment*, vol. 8, pp. 1928–1931, 2015.
[27] L. Sidirourgos, M. Kersten, and P. Boncz, "SciBORQ: Scientific data management with bounds on runtime and quality," in *Proc. Int. Conf. Innovative Data Syst. Res.*, 2011, pp. 296–301.
[28] M. L. Kersten, S. Idreos, S. Manegold, and E. Liarou, "The researcher's guide to the data deluge: Querying a scientific database in just a few seconds," *Proc. VLDB Endowment*, vol. 4, pp. 1474–1477, 2011.
[29] Sellam, et al., "Meet Charles, big data query advisor," presented at the *6th Biennial Conf. Innovative Data Syst. Res.*, Asilomar, CA, USA, 2013.
[30] L. Jiang and A. Nandi, "SnapToQuery: Providing interactive feedback during exploratory query specification," *Proc. VLDB Endowment*, vol. 8, pp. 1250–1261, 2015.
[31] A. Kalinin, U. Cetintemel, and S. Zdonic, "Interactive data exploration using semantic windows," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 505–516.
[32] A. Kalinin, U. Çetintemel, and S. B. Zdonik, "Searchlight: Enabling integrated search and exploration over large multidimensional data," *Proc. VLDB Endowment*, vol. 8, pp. 1094–1105, 2015.
[33] A. Kashyap, V. Hristidis, and M. Petropoulos, "Facetor: Cost-driven exploration of faceted query results," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manage.*, 2010, pp. 719–728.
[34] S. Chaudhuri, "Generalization and a framework for query modification," in *Proc. 6th Int. Conf. Data Eng.*, 1990, pp. 138–145.
[35] C. Mishra and N. Koudas, "Interactive query refinement," in *Proc. 12th Int. Conf. Extending Database Technol. Advances Database Technol.*, 2009, pp. 862–863.
[36] N. Koudas, C. Li, A. K. H. Tung, and R. Vernica, "Relaxing join and selection queries," *Proc. 32nd Int. Conf. Very Large Data Bases*, 2006, pp. 199–210.
[37] A. Kadlag, A. V. Wanjari, J. Freire, and J. R. Haritsa, "Supporting exploratory queries in databases," in *Proc. 9th Int. Conf. Database Syst. Advanced Appl.*, 2004, pp. 594–605.
[38] D. Mottin, S. B. Roy, G. Das, T. Palpanas, and Y. Velegrakis, "A probabilistic optimization framework for the empty-answer problem," *Proc. VLDB Endowment*, vol. 6, pp. 1762–1773, 2013.
[39] S. Sarawagi and A. Bhamidipaty, "Interactive deduplication using active learning," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 269–278.
[40] N. Panda, K.-S. Goh, and E. Y. Chang, "Active learning in very large databases," *Multimedia Tools Appl.*, vol. 31, no. 3, pp. 249–267, 2006.
[41] I. Ruthven and M. Lalmas, "A survey on the use of relevance feedback for information access systems," *Knowl. Eng. Rev.*, vol. 18, no. 2, pp. 95–145, 2003.
[42] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2008, pp. 1070–1079.
[43] J. Zhu, "Active learning for word sense disambiguation with methods for addressing the class imbalance problem," in *Proc. ACL*, 2007, pp. 783–790.
[44] Khoussainova et al., "A case for a collaborative query management system," presented at the *4th Biennial Conf. Innovative Data Syst. Res.*, Asilomar, CA, USA, 2009.
[45] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis, "Query recommendations for interactive database exploration," in *Proc. 21st Int. Conf. Scientific Statistical Database Manage.*, 2009, pp. 3–18.
[46] E. Sadikov, J. Madhavan, L. Wang, and A. Halevy, "Clustering query refinements by user intent," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 841–850.

Kyriaki Dimitriadou received the BA degree in applied informatics from the University of Macedonia, Greece, and the MA degree in computer science from Brandeis. She is currently working toward the PhD degree in computer science at Brandeis University. Her research interests are in database systems with a focus on data exploration.

Olga Papaemmanouil received the undergraduate degree in computer engineering and informatics from the University of Patras, Greece, the MSc degree in information systems from the Athens University of Economics and Business and the PhD degree from Brown University, in 2008. She has been an assistant professor of computer science with Brandeis University since 2009. Her research interests include data management and distributed systems with focus on data streams, query performance, cloud databases, and data exploration. She received a US NSF CAREER Award (2013) and a Paris Kanellakis Fellow (2002).

Yanlei Diao received the bachelor's degree from Fudan University, in 1998, the MPhil degree from the Hong Kong University of Science and Technology, in 2000, and the PhD degree in computer science from the University of California, Berkeley, in 2005. She is an associate professor of computer science with the University of Massachusetts. Her research interests include information architectures and data management systems, with a focus on big data analysis, data streams, interactive data exploration, and uncertain data management. She received the 2013 CRA-W Borg Early Career Award, the US NSF CAREER Award, the IBM Scalable Innovation Faculty Award, and the finalist for the Microsoft Research New Faculty Fellowship. Her PhD dissertation won the 2006 ACM-SIGMOD Dissertation Award Honorable Mention.