AIDE: An Automatic User Navigation System for Interactive Data Exploration

Yanlei Diao±, Kyriaki Dimitriadou*, Zhan Li*, Wenzhao Liu±,
Olga Papaemmanouil*, Kemi Peng*, Liping Peng±

±{yanlei, wenzhao, lppeng}@cs.umass.edu, *{kiki, zhanli, kermit, olga}@cs.brandeis.edu

±UMass Amherst, *Brandeis University

ABSTRACT

Data analysts often engage in data exploration tasks to discover interesting data patterns, without knowing exactly what they are looking for. Such exploration tasks can be very labor-intensive because they often require the user to review many results of ad-hoc queries and adjust the predicates of subsequent queries to balance the trade-off between collecting all interesting information and reducing the size of returned data. In this demonstration we introduce *AIDE* , a system that automates these exploration tasks. AIDE steers the user towards interesting data areas based on her relevance feedback on database samples, aiming to achieve the goal of identifying all database objects that match the user interest with high efficiency. In our demonstration, conference attendees will see AIDE in action for a variety of exploration tasks on real-world datasets.

1. INTRODUCTION

Traditional DBMSs are suited for applications in which the questions to be asked are already well understood. There is, however, a class of Interactive Data Exploration (IDE) applications in which this is not the case. Examples of such interactive applications include, but are not limited to, scientific computing, financial analysis, evidence-based medicine, and genomics.

IDE is fundamentally a long-running, multi-step process with end-goals not stated explicitly. Users try to make sense of the underlying data space by navigating through it. The process includes a great deal of experimentation with queries, backtracking on the basis of query results, and revision of results at various points in the process. To make the most of the increasingly complex big data sets, users need an automated service to effectively and efficiently guide them through the data space. One example can be found in scientific applications (e.g., LSST [3], SDSS [4]) which collect enormous data sets periodically. Here, as data volumes and the user community continue to grow, there is a strong need for interactive data exploration: that is, when a scientist connects to an extremely large database, she may not be able to express her data interests precisely. Instead, she may want to navigate through a subspace of the data set (e.g., a region of the sky) to find the objects of interest, or may want to see a few samples, provide yes/no feedback, and expect the data management system to find more similar objects.

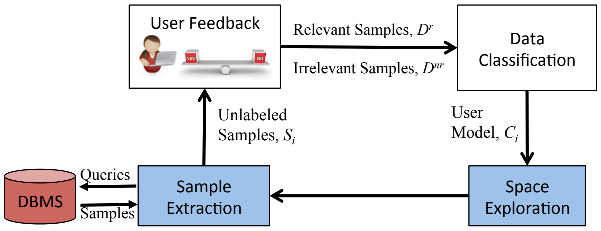


Figure 1: The AIDE framework

This demonstration will introduce *AIDE* (*Automatic Interactive Data Exploration)*, a data navigation system that addresses the above challenges. AIDE is an automated data exploration system that steers the user towards interesting data areas based on her relevance feedback on database samples, aiming to achieve the goal of identifying all database objects that match the user interest with high efficiency. It relies on a combination of machine learning techniques and sample selection algorithms to provide effective data exploration results (matching the user interest well) as well as high interactive performance over databases of large sizes.

2. SYSTEM OVERVIEW

AIDE employs an automatic user steering approach: it *automatically* learns the user's interests and drives the exploration process towards data relevant to these interests. To achieve this, it relies on an active learning model that iteratively requests user feedback on strategically collected data samples. In a nutshell, the user engages in a "conversation" with the system by characterizing a set of data samples as relevant or irrelevant to her interest. The user feedback is incrementally incorporated into the system and used to gradually improve its effectiveness, that is, to identify interesting data spaces for further exploration and eventually generate a user model that precisely predicts the set of data matching the user interest.

Initially the user is given a database schema $\mathcal{D}$ and elects d attributes for data exploration, where these d attributes may include attributes both relevant and irrelevant to the final expression that captures the true user interest. The exploration is performed in a d-dimensional space of T tuples where each tuple represents a d-dimensional object. For a given user, our exploration space is divided to the relevant object set T^r and irrelevant set T^{nr}.

The steering process starts when the user provides feedback on the relevance of the first set of retrieved samples. We assume a binary relevance system where the user indicates whether a data object is relevant or not to her. The labeled samples are used to train a classification model of the user interest, e.g., it predicts which objects are relevant to the user based on the feedback collected so far

Proceedings of the VLDB Endowment, Vol. 8, No. 12
Copyright 2015 VLDB Endowment 2150-8097/15/08.

(*Data Classification*). The *User Model* may use any subset of the d attributes of the exploration space to characterize user interests.

Each iteration i refines the characterization of the user interest by exploring further the data space. Specifically, we leverage the previous user model C_{i-1} to identify promising data areas to be sampled further (*Space Exploration*) and to retrieve the next sample set $S_i \subseteq T$ to show to the user (*Sample Extraction*). Based on the collected labeled samples up to the i-th iteration, a new user model C_i is generated.

The above steps are executed iteratively towards convergence to a user model that captures the user interest, i.e., eliminating irrelevant objects while identifying all or most relevant to the user objects. 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. Hence, the user decides on the effort she is willing to invest (i.e., number of samples she labels) while the system leverages her feedback to maximize the accuracy of her user interest model.

Our exploration approach, first introduced in [8], fundamentally differs from active learning theory (e.g., [6]) which often requires searching the entire database to find the "best" sample to show to the user next. Such exhaustive search is infeasible for ever-growing databases sizes and precludes any interactive performance. Instead, AIDE seeks to identify promising subareas of the data space and samples in those areas that will quickly increase the accuracy of our model, leading to both effective and highly efficient exploration.

2.1 Discovering Linear Patterns

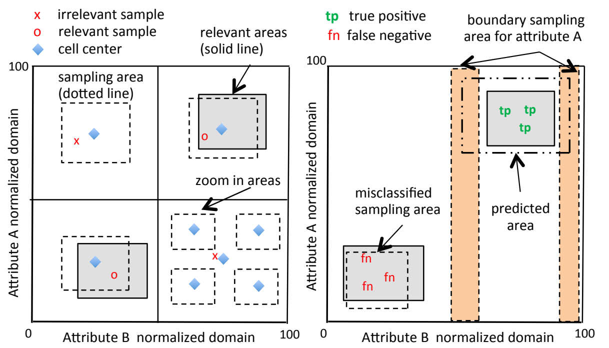
In [8] we introduced a number of data exploration techniques for discovering **linear patterns**, i.e., interests captured by conjunction and/or disjunction of linear (range) predicates. We refer to such interests as *relevant areas* in the d-dimensional exploration space.

To identify linear patterns, our system relies on decision tree classifiers and includes three exploration phases which are designed to improve the F-measure of the final decision tree C on the total data space T, defined as $F(T) = \frac{2\times precision(T)\times recall(T)}{precision(T)+recall(T)}$, while offering interactive performance to the user. This effectiveness measure captures the trade-off between collecting all relevant information and reducing the size of returned data. Next we highlight our exploration techniques. More details can be found in [8].

Relevant Object Discovery. Our first exploration phase aims to increase the probability of "hitting" data areas of interest to the user. For uniform data distributions, AIDE uses a d-dimensional grid, defined by the normalized domains of the d exploration attributes the user picked, and it randomly samples the center of each grid cell (see Figure 2(a)). If no relevant object is retrieved from one cell, we further explore this grid cell by"zooming-in", i.e., splitting the specific cell to finer sampling areas. An example of this operation is shown in the low right grid cell in Figure 2(a).

To handle skewed distributions AIDE uses the k-means algorithm [7] to partition the data space into k clusters. Each cluster includes similar objects (where similarity is defined by a distance function) and AIDE collects samples around the centroid of each cluster. This approach allows us to apply more dense sampling in the dense subspaces. AIDE also creates multiple exploration levels, where higher levels include fewer clusters than lower ones, allowing for to zoom-in into the dense parts of the data space.

Misclassified Exploitation. The grid-based sampling strives to identify single points of interest, one per each grid cell. In order to expand them to relevant *areas*, a significant number of relevant objects from within each area needs to be fed to the classification algorithm. To address this, AIDE defines a sampling area around objects labeled as relevant but classified as non-relevant and collects random samples within this area. To further reduce the sampling areas, clustering techniques are used to identify clusters of false negatives (which most likely belong to the same relevant area) and create a sample area around each of the generated clusters (see Figure 2(b)). This technique increases the F-measure since it increases the relevant samples while reducing the misclassified ones.


x irrelevant sample
o relevant sample
◆ cell center
relevant areas (solid line)
sampling area (dotted line)
zoom in areas
100
0
100
Attribute A normalized domain
Attribute B normalized domain
x
o
o
x
tp true positive
fn false negative
boundary sampling area for attribute A
tp
tp
tp
predicted area
misclassified sampling area
fn
fn
fn
100
0
100
Attribute A normalized domain
Attribute B normalized domain

(a) Relevant Object Discovery (b) Misclassified and Boundary Exploitation

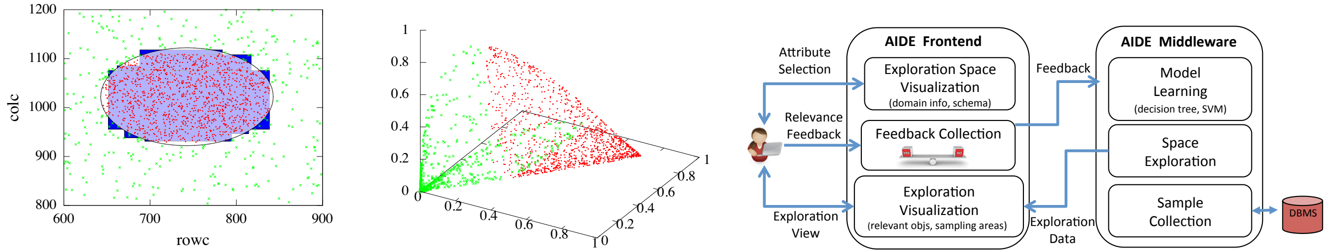
Figure 2: Exploration phases for the linear patterns

Boundary Exploitation. Given a set of relevant areas identified by the decision tree classifier, our next step is to refine them by incrementally adjusting their boundaries. This leads to better characterization of the user's interests, i.e., higher accuracy of our final results. For a relevant area characterized by a predicate on a set of attributes $< t_1, ..., t_d >$ we collect random samples across the domain of each attribute t_i such as the distance of each value of t_i is less than a value k from the boundary of the relevant area. Figure 2(b) shows an example of a predicted area and the sampling around the boundaries of the attribute A. The approach is applied in parallel to all the boundaries of the hyper-rectangles for the relevant areas, allowing us to shrink/expand each area as we get more feedback from the user. By refining the boundaries of the relevant areas, AIDE discovers more relevant tuples and increases its accuracy.

2.2 Discovering Non-Linear Patterns

Non-linear patterns, i.e., patterns that cannot be captured by range predicates, are prevalent in applications ranging from location-based searches to scientific exploration tasks using complex predicates. While our decision tree based approach can approximate non-linear patterns, it suffers from poor performance. For example, to predict a circle-shaped relevant area "$(rowc - 742.76)^2 + (colc - 1022.18)^2 < 100^2$" on two location attributes $rowc$ and $colc$ in the SDSS dataset [4], the decision tree model required over 2000 training samples and approximated the circle region using 58 range predicates combined through conjunction/disjunction, as illustrated in Figure 3(a). This motivated us to seek a more efficient approach to supporting non-linear patterns, reducing both the user labeling effort and the querying and sampling cost in the database.

Our new approach uses Support Vector Machines (SVMs) as the classification algorithm. Here, the training set (i.e., labeled samples) in the *data space* is mapped, via a *kernel function*, to a higher-dimensional *feature space* where examples of different classes are linearly separable. Figure 3(b) shows a 3-dimensional feature space (manually selected by us) where the training points of the circle area in Figure 3(a) are linearly separated; in practice an SVM may need many more dimensions to see such linear separation. Then among the many hyperplanes that might classify the data in the feature space, SVM selects the one, called the *decision boundary* $\mathcal{L}$, with the largest distance to the nearest mapped samples of any class; this boundary $\mathcal{L}$ is used as the model of user



(a) A circle area (green area) approximated by decision trees (blue area), with training points in 2-D data space (red: Y, green: N).

(b) Linear separation of training points in a 3-D feature space using SVM (red: Y, green: N).

(c) AIDE architecture.

Figure 3: **(a-b) SVM for learning a non-linear (circle) relevant area and (c) the AIDE architecture.**

interest as it separates relevant from irrelevant objects. The main challenge here is to identify at each iteration of the exploration process, the next to-be-labeled sample that can quickly improve the accuracy of the current user model $\mathcal{L}$.

Recent active learning theory [6] proposed to choose the example closest to the current decision boundary. However, they suggest a search through the entire dataset in *each* iteration, which is prohibitively expensive. AIDE puts active learning theory into practice: we find the unlabeled example closest to the current decision boundary $\mathcal{L}$ without retrieving all the tuples and evaluating their distances to $\mathcal{L}$. AIDE includes the following two techniques for identifying samples to show to the user in each iteration.

Bounding the search area using decision trees. We define a δ-region around the current SVM decision boundary $\mathcal{L}$ and form a two-class training dataset such that points inside the δ-region are the relevant class and points outside the δ-region are not. Then a decision tree can be trained to approximate the δ-region and can be easily translated to an exploration query, Q, to send to the database $\mathcal{D}$. Finally given the query result $Q(\mathcal{D}) \subseteq \mathcal{D}$, we iterate over this set and find the example closest to $\mathcal{L}$. Note that δ can be set to balance two trends: a too small δ can lead to too few training points in the relevant class while a too large δ may result in $Q(\mathcal{D}) = \mathcal{D}$.

Branch and bound search. AIDE also builds indexes such as R-trees [5] and CF trees [9] over the database, and perform fast branch-and-bound search over these indexes. Take R-tree for example. Each R-tree node offers a hyper-rectangle, $[a_j, b_j]$, $j = 1, \ldots, d$, as a minimum bounding box of all the data points reachable from this node. Given the current SVM decision boundary $\mathcal{L}$, we search the R-tree top-down in a depth-first fashion and always maintain the current closest tuple, $\boldsymbol{x}^*$, and its distance to $\mathcal{L}$, $f(\boldsymbol{x}^*, \mathcal{L}) \overset{\text{def}}{=} f^*$. Note that $f^* = +\infty$ before any leaf node is visited. For each intermediate node visited, we dynamically compute a lower bound of the distance from any point in its hyper-rectangle to $\mathcal{L}$ by calling a constrained optimization solver: $\min_{\boldsymbol{x}} f(\boldsymbol{x}, \mathcal{L})$ s.t. $a_j \leq \boldsymbol{x}^{(j)} \leq b_j$, $j = 1, \ldots, d$. If the lower bound is already higher than f^*, we can prune the entire subtree rooted at this node. Once we reach a leaf node, we can update $\boldsymbol{x}^*$ and f^* accordingly. Then the final $\boldsymbol{x}^*$ is the closest tuple to $\mathcal{L}$ in the entire database.

3. USER INTERFACE

Our AIDE prototype is designed on top of a traditional relational database system and its architecture (shown in Figure 3(c)) includes three software layers: (1) an interactive visualization front-end, (2) the AIDE middleware that implements all our exploration techniques as detailed in §2.1 to §2.2, and (3) a database backend supporting our data exploration techniques.

Our visualization front-end provides several functionalities. The user is initially presented with the database schema and she can select an initial subset of attributes of interest, which will be refined later by the data exploration. Our front-end can also visualize domain distributions of each attribute to further allow the user to filter attributes based on the domain characteristics and restrict the value ranges of the relevant attributes for consideration (e.g., focus on a dense region or a region close to a landmark). Users can select between different types of plots of the underlying data distributions, such as histograms and heat maps. Figure 4(a) shows a histogram example on an attribute in the SDSS [4] dataset.

In the next step the system starts a series of iterations of sample labeling, model learning and space exploration. The visualization front-end supports this process by visualizing various subspaces of the exploration attributes, presenting data samples to the user, collecting yes/no labels from the user regarding the relevance of the shown samples and showing the locations of labeled samples in the exploration space. Figure 4(b) shows an example of this interface.

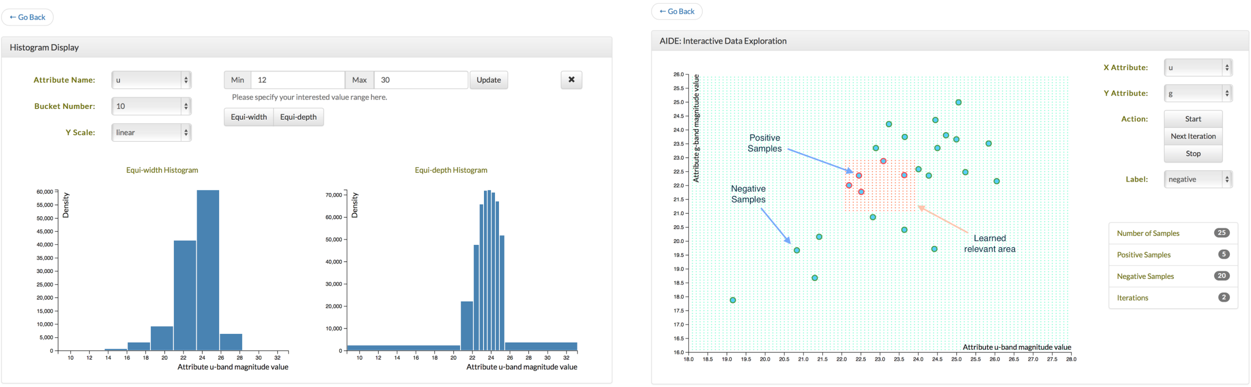
Sitting below the visualization front-end is the "automatic user steering" layer (AIDE middleware in Figure 3(c)), which is the heart of our system. This component is implemented in Java, with a few machine learning libraries integrated in the system. At each iteration it incorporates the newly collected labeled samples and generates a new classification model. At any point the user can request a visualization of the current user model (i.e., decision tree or SVM decision boundary) which entails highlighting the objects classified as relevant to the user. The user can then decide to stop the exploration (if she is satisfied with the current set of identified objects) or to proceed to the next round of exploration.

The database backend uses PostgreSQL. The database engine includes various sampling techniques implemented as stored procedures. These techniques are designed to support the exploration approaches we discussed in §2.1 to §2.2. For example one procedure supports the decision tree approach to learning linear patterns (§2.1) by selecting a predefined number of random samples within a given distance from the center of a d-dimensional area, while other procedures support random and weighted sampling.

4. DEMONSTRATION

In our demonstration attendees will be able to explore the following real world datasets.

1. *AuctionMark* [1]: It includes information on action items and their bids (e.g., the initial/current price, number of bids, and number of days an item in an auction). We expect that attendees will have a sufficient understanding of this domain and will be able to easily formulate ad-hoc, intuitive, exploration tasks (e.g., "identify auction items that are good deals"). This dataset has size of 1.77GB.



Histogram Display

Attribute Name: u

Bucket Number: 10

Y Scale: linear

Min 12 Max 30

Please specify your interested value range here.

AIDE: Interactive Data Exploration

X Attribute: u

Y Attribute: g

Label: negative

Number of Samples	25
Positive Samples	5
Negative Samples	20
Iterations	2

(a) Histogram visualization for exploration attributes.

(b) Exploration visualization (learned areas, labeled samples).

Figure 4: AIDE Frontend Interface

2. *Sloan Digital Sky Survey* [4]: This is a scientific data set generated by digital surveys of stars and galaxies and it is often used by astronomers to find "observations" of interesting sky objects. It has a complex schema and a large data size. We will use it to demonstrate the efficiency of our optimizations. Furthermore, this dataset includes attributes with different value distributions, allowing us to experiment with both skewed and uniform exploration spaces. We will use various datasets with size of 1GB-100GB.

We are also investigating alternative datasets that include both numerical and categorical domains (e.g., US housing and used cars datasets available through the DAIDEM Lab [2]).

During the demonstration, we will run AIDE on two laptop machines and our backend will use the PostgreSQL database engine. Smaller datasets will be stored on the local disk and an external hard drive will be used for our largest datasets.

Our audience will observe the following demonstration scenarios for predicting both linear and non linear patterns of user interests. Specifically, we will demonstrate the effectiveness of the decision tree based techniques described in §2.1 for predicting linear patterns, while we will demonstrate the SVM-based techniques described in §2.2 by predicting non linear patterns.

Scenario 1: System utility In this scenario, attendees will explore our system's utility by comparing it to the traditional manual exploration approach. Specifically, they can pick a dataset and a set of exploration attributes, and then start an ad-hoc search for "interesting" areas (e.g., search for "good deals" in the Auctions [1]") by writing their own series of SQL queries that potential captures that interest. After reviewing their results they can adjust their query predicates aiming to collect only relevant objects. This iterative process will continue until the users are satisfied with their query output. The results of final query will be then treated as the user's interest, i.e., her relevant objects, and we will use AIDE to automatically identify them. Using AIDE attendees will review and provide relevance feedback only on a few selected samples. Attendees will observe that through AIDE their relevant objects can be identified with significant less reviewing effort and user wait time, similar to the results of our user study [8].

Scenario 2: System step-through In this scenario we will demonstrate to attendees the exploration techniques used by our system. Attendees can start searching for interesting areas using AIDE and at each iteration, they will observe through our front-end: (a) the collected relevance feedback on data samples, (b) the prediction of the current classification model, and (c) the sampling areas for collecting the next set of samples. Using our visualization interface we will guided them step-be-step through each of the techniques described in §2.1 to §2.2, demonstrating how they lead to the selection of the sampling areas for the next iteration. This scenario will work with either pre-defined relevant areas (i.e., for which the relevance feedback is known a-priori) or ad-hoc relevant areas that the attendees have highlighted for us through the visualization interface. In both cases, the target set of relevant objects are known in advance which will allow attendees to confirm that AIDE improves its accuracy in each iteration.

Scenario 3: Optimization effectiveness This scenario will use pre-defined relevant areas and attendees will be able to observe various real-time experiments with a range of queries and exploration spaces by adjusting various "knobs" such as the data size, the number of exploration attributes, and data distributions (i.e., degree of skewness). They will also be able to change the configuration of relevant areas (i.e., "large" vs "small" areas, the number of relevant areas) and observe the effects on our system effectiveness and efficiency. In addition, attendees will be able to select the exploration techniques and optimizations that AIDE applies and observe the impact on the user wait time and accuracy.

5. CONCLUSION

Through our demonstration of AIDE we will illustrate that (a) it is feasible to automate the labor-intensive task of data exploration, and (b) combining machine learning algorithms and data management optimizations can lead to interactive exploration performance and reduction of the user exploration and data reviewing effort.

6. ACKNOWLEDGMENTS

This work was funded in part by NSF under grants IIS-1253196 and IIS-1218524.

7. REFERENCES

[1] AuctionMark Benchmark, http://hstore.cs.brown.edu/projects/auctionmark/.

[2] DIADEM Lab, University of Oxford, https://www.cs.ox.ac.uk/projects/DIADEM/.

[3] Large Synoptic Survey Telescope, http://http:/www.lsst.org/.

[4] Sloan Digital Sky Survey, http://www.sdss.org/.

[5] N. Beckmann, H.-P. Kriegel, et al. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *SIGMOD*, 1990.

[6] A. Bordes, S. Ertekin, et al. Fast kernel classifiers with online and active learning. *J. Mach. Learn. Res.*, 6:1579–1619, Dec. 2005.

[7] L. Breiman, J. H. Friedman, et al. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[8] K. Dimitriadou, O. Papaemmanouil, and Y. Diao. Explore-by-Example: An Automatic Query Steering Framework for Interactive Data Exploration. In *SIGMOD*, 2014.

[9] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In *SIGMOD*, 1996.