Visual Exploration of Machine Learning Results using Data Cube Analysis

Minsuk Kahng
Georgia Tech
Atlanta, GA, USA
kahng@gatech.edu

Dezhi Fang
Georgia Tech
Atlanta, GA, USA
dezhifang@gatech.edu

Duen Horng (Polo) Chau
Georgia Tech
Atlanta, GA, USA
polo@gatech.edu

ABSTRACT

As complex machine learning systems become more widely adopted, it becomes increasingly challenging for users to understand models or interpret the results generated from the models. We present our ongoing work on developing interactive and visual approaches for exploring and understanding machine learning results using data cube analysis. We propose *MLCube*, a data cube inspired framework that enables users to define instance subsets using feature conditions and computes aggregate statistics and evaluation metrics over the subsets. We also design *MLCube Explorer*, an interactive visualization tool for comparing models' performances over the subsets. Users can interactively specify operations, such as drilling down to specific instance subsets, to perform more in-depth exploration. Through a usage scenario, we demonstrate how *MLCube Explorer* works with a public advertisement click log data set, to help a user build new advertisement click prediction models that advance over an existing model.

CCS Concepts

•**Information systems** → *Data management systems;* Online analytical processing; •**Human-centered computing** → *Visualization;* •**Computing methodologies** → *Machine learning;*

Keywords

Interactive data analysis; machine learning; data cube; data visualization

1. INTRODUCTION

As machine learning systems become more widely adopted, they are also becoming increasingly complex. Applying machine learning techniques on large-scale, real-world problems often entails many steps, including feature extraction, feature transformations, model selection, and model evaluation [4, 21]. Each component itself may introduce its own complexity. For example, it is non-trivial to extract meaningful feature sets from a large number of attributes [2]. In practice, as machine learning systems increase in size and complexity, they are often viewed as "black box", as there are no effective ways for understanding the internal mechanisms of these complex systems or interpreting their model results [11, 19].

The importance of helping users interpret machine learning models has received increasing attention. Recent works [12, 1, 11, 19] highlighted that while overall model accuracy can be used to select models, users often want to understand why and when a model would perform better than others, so that they can trust the model and know how to improve the model. Current interpretation approaches often focus on explaining single models (e.g., computing feature importance from a boosted tree), but they cannot be directly applied on other models (e.g., neural networks) since the internal working mechanisms of different models can vary widely [11, 19]. Current visualization approaches primarily support instance-based explanation (e.g., how individual instances contribute to a model's accuracy) [1, 18]; more work is needed to find out if they may scale up to larger data sets or more complex systems.

Introducing *MLCube Explorer*. This paper presents our ongoing work on developing an interactive visualization tool for comparing machine learning models' performances and exploring model results using data cube analysis. Our goal is to help users interactively explore and determine the right abstraction level of analysis — as comparing two models by their overall accuracies is often too coarse and not conducive to discovering contributing causes; and inspecting individual instances within a large data set is too fine-grained and may not scale — our work helps user reach the "happy medium."

Specifically, our proposed *MLCube* framework enables users to define **instance subsets** using *relational selections* over features, and compute aggregate statistics and evaluation metrics over the subsets. Through our *MLCube Explorer* (Figure 1), users can visually explore these subsets and interactively specify operators to further analyze the results. For example, they can drill down into subsets to explore relationships among features and examine how they affect model results. Users can freely define subsets with both raw data attributes and transformed features.

Drilling down model results. *MLCube* introduces a new way for users to select instance subsets using both data *attributes* (e.g., the titles of text documents) and *features*, which are often derived from attributes (e.g., number of

HILDA'16, June 26 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939502.2939503

Figure 1: A screenshot of *MLCube Explorer*, our interactive visualization tool for analyzing and comparing machine learning results. Each row represents a subset. The *subset summary view* (middle) visually shows several statistics for each subset (e.g., count, proportion of positive instances, distribution of prediction scores, and model's accuracy). The *correlation matrix view* (right) visualizes accuracy differences between two models for a subset combination (e.g., `user_age_group=1 AND position=3`). A cell with a larger circle means there are more instances in that subset combination. Yellow means Model A outperforming Model B; green means Model B outperforming Model A. The darker the color, the greater the performance difference. Users can interact with the interface in several ways, including drill-down, sorting subsets, and adding new subsets.

terms in the titles). Prior research has shown that leveraging *features* in explanations is a key to interpreting machine learning results [12, 5, 11]. The feature-based analysis can generalize to any models that share the same feature sets, unlike model-specific explanations [24]. Our approach advances over prior work [6], by allowing users to interactively select subsets based on their knowledge of any feature transformations that have been carried out, and also keep track of the intermediate stages in the workflow. This functionality is important because raw data attributes are often transformed into features through feature engineering (e.g., as in calculating the number of terms from the titles) [2]. Slicing results by features may impede user understanding, since revealing relationships between the data attributes and the behavior of machine learning algorithms could accelerate understanding of model behavior [16, 18].

Interactive visualization. To help users quickly get an overview of the data and model results and spot interesting patterns and anomalies, *MLCube Explorer* allows users to visually explore aggregate statistics over subsets of data instances and interactively drill down into models. This enables users to find interesting patterns between features and model results, leading to discovering insights that help them understand the mechanisms of the models and further improve their performance.

Our contributions are:

- *MLCube*, a data cube inspired framework that enables users to explore aggregate statistics and evaluation metrics over the user-defined subsets. (Section 3)
- *MLCube Explorer*, a visualization tool for interactively exploring *MLCube* for analyzing and comparing models' performances. (Section 4)

We demonstrate how *MLCube Explorer* works with a public advertisement click log data set through a usage scenario of building advertisement click prediction models. (Section 4.3)

2. BACKGROUND: A TYPICAL MACHINE LEARNING PIPELINE

In this section, we describe a typical workflow of building machine learning models for data sets (see Figure 2), to motivate and provide the context for *MLCube*'s contributions. For example, as *MLCube* is defined over data, features, and model results, we will first explain the terminology and symbols for describing them. Building machine learning models in practice often involves several steps, including data pre-processing, feature extraction, feature transformations, model building, and model evaluations [4, 21]. To illustrate with a concrete example, we use the task from the 2012 KDD Cup competition (Track 2),[1] whose goal is to build advertisement click prediction models.

A **raw data table R** is a relation having a set of attributes A and consisting of a set of instances. Each instance $\mathbf{r}_i \in \mathbf{R}$ consists of a set of attribute values $\mathbf{r}_i[A_j]$. An attribute value could be either single-valued or multi-valued, where its data type could be integer, float, or text. For the case of advertisement prediction, each instance represents an event in which an advertisement is shown to a user under a certain setting. Its attributes include user ID, age, gender, ad ID, the title of an ad, query ID, query text, position of ad on a webpage, binary value of whether the user clicked the ad (1 if clicked; 0 otherwise), etc. Using the database terminology, the raw data table **R** can be thought of as a joined relation of a log table (i.e., fact table) with several entity tables (e.g., Users, Ads) [14].

The next step is the *feature extraction* or *feature transformation* procedure that constructs a set of features from a raw data table **R**. This step consists of a set of **feature functions** $\mathcal{F}$, taking a raw data table **R** as an input and producing a **feature vector table X** (and **labels y**) that will be used as an input for a learning model [3]. Each feature function $F_j \in \mathcal{F}$ produces a j-th feature value x_{ij} for a given instance r_i. In other words, each instance r_i will be transformed into a feature vector $\mathbf{x}_i = (x_{i1}, ..., x_{ij}, ...)$ and a label $y_i \in \{0, 1\}$. Some feature functions may simply select an attribute (e.g., user's age), while others may perform computation. For example, an *average function* may compute the average click-through rate for each ad ID; a *tf-idf function* may calculate tf-idf text similarity between a query and the title of an advertisement [25].

Given a feature vector table and labels, engineers would then run different machine learning algorithms on them. Once a **prediction model** $h(\mathbf{X}, \mathbf{y})$ is constructed (e.g., logistic regression), it will be used to classify test instances $\mathbf{x}_i$. For each instance, the model produces a **prediction score** s_i and determines its corresponding predicted label $\hat{y}_i \in \{0, 1\}$. The performance of the models is evaluated using a **evaluation measure** l (e.g., accuracy, AUC score), which takes as input a list of (label y, score s or predicted label $\hat{y}$) pairs, and outputs a single value (i.e., the measure).

3. MLCUBE: DATA CUBES FOR MACHINE LEARNING

We present *MLCube*, a data cube inspired framework for analyzing machine learning model results. Our approach enables users to flexibly analyze and understand model results at the *subset* level, through interactively exploring and

[1] http://www.kddcup2012.org/c/kddcup2012-track2

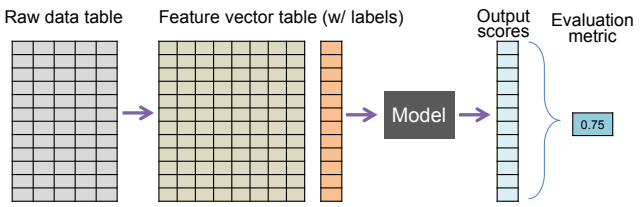

Figure 2: Typical machine learning pipeline

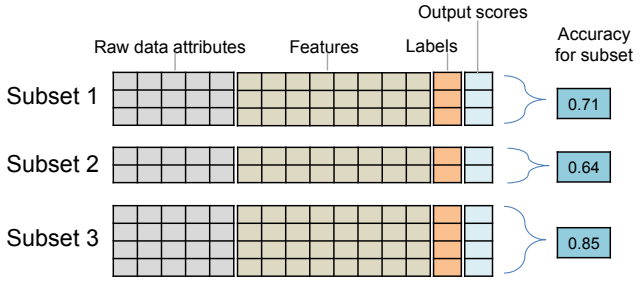

Figure 3: MLCube enables subset-level analysis of machine learning results by computing aggregate statistics (e.g., accuracy) for a subset of instances.

generating a wide range of instance subsets (see Figure 3). A subset is defined as a relational selection over a feature vector table **X** or the raw data table **R** (e.g.,`user_gender = 'female'`). *MLCube* computes aggregate statistics (e.g., accuracy) for all user-defined subsets.

While OLAP is traditionally defined over a fact table consisting of a set of dimension attributes and a measure attribute, *MLCube* is defined over $\mathbf{R} \bowtie \mathbf{X} \bowtie \mathbf{y} \bowtie \mathbf{s} \bowtie \hat{\mathbf{y}}$. The primary keys (PKs) of all five relations are instances' unique ID, and all join conditions apply only on the PKs. As all intermediate data are included, a subset can be defined not only over features, but over data attributes (e.g., `ad_title contains 'car'`), or over a combination of multiple components (e.g., `ad_title contains 'car' AND tfidf_sim_query_title >= 0.7 AND label = 1`).

For the *measure attributes* of the cube (i.e., values to be aggregated), we find the following measures particularly useful for analyzing model results:

- **Accuracy (or any evaluation measure, such as AUC) for a model:** Computing accuracy by subsets allows engineers to understand which data regions work better or worse for a selected model. If accuracy for a certain subset is relatively low, engineers may want to inspect the corresponding instances.
- **Accuracy difference between two models:** Comparing a new model to a control model in a subset-level is particularly useful for model selection, as it could describe which parts of data helps improve or degrade the model performance.
- **Number of instances** *(model-independent)*: Instance counts help engineers understand the data distributions and find important subsets (e.g., they may ignore subsets with very small number of instances).
- **Proportion of positive instances** *(model-independent)*: Helps with feature engineering by showing which features are more discriminative.

In addition to the above measures, there could be many useful measures we can define. For example, computing score distributions of positive (or negative) instances helps people understand the result of a model.

While subsets can be defined as any relational selection with SQL-like expression, a set of dimension attributes (i.e., categorical) is often selected in practice because of scalability issues, since infinite number of subsets could be generated. By default, *MLCube* selects all categorical attributes (i.e., cardinality less than certain threshold) and create discrete bins for selected numerical (continuous) attributes and features. To speed up statistics computation over subsets, the *MLCube* is then partially materialized for these subsets. In our implementation, we use an algorithm described in [17] (Algorithm 1) with Apache Spark[2] and constrain the maximum number of dimensions to 4.

4. VISUAL EXPLORATION OF MLCUBE

This section presents *MLCube Explorer*, an interactive visualization tool for exploring machine learning results using *MLCube*. Interactive visualizations has been proven to be very effective for finding interesting patterns and spotting anomalies from large, multidimensional data by effectively representing data and allowing users to interact with them [22, 23, 1, 15]. We first introduce the interface of our visualization tool and describe operations for users to interact with the interface. Then we present a usage scenario of how the tool can help a machine learning engineer understand models.

4.1 User Interface

Figure 1 shows a screenshot of *MLCube Explorer*, visualizing the performances of (one or) two user-selected models by subsets. Each row represents a subset. By default, we show all subsets consisting of one selection predicate chosen from feature vectors. The column of each subset row is divided into two areas: (1) the **subset summary view** which shows summary statistics for each subset and (2) the **correlation matrix view** which visualizes its pairwise correlations to other subsets. As for the subset summary view, we visualize the number of instances, the proportion of positive instances, the prediction score distributions of positive/negative instances for each model, and the accuracy value for each model. As for the correlation matrix view, each cell visualizes the accuracy difference between two models for a subset combination (e.g., `user_age_group=1 AND position=3`). A cell with a larger circle means there are more instances in that subset combination. Yellow means Model A outperforming Model B; green means Model B outperforming Model A. The darker the color, the greater the performance difference.

4.2 Interactive Operations

Users can interact with the interface to further explore *MLCube* using the following operations.

1. **Drill-down/Roll-up into a subset:** By clicking a subset (e.g., `user_age_group = 0`), its predicate will be applied to all other subsets, updating all values and visual elements in all rows and cells.

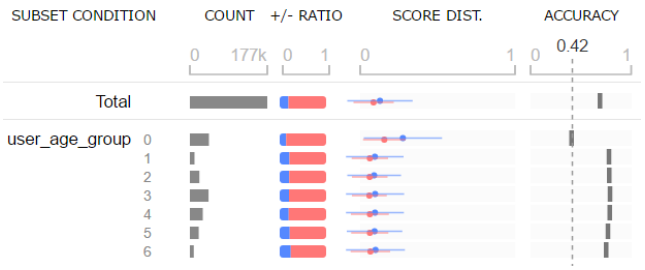

Figure 4: Our user Jane finds that a subset of instances "`user_age_group = 0`" performs distinctly worse than the other age groups, indicated by the left-most solid bar in the accuracy column.

2. **Adding user-defined subsets:** Define a new subset based on a user-defined relational selection predicate. The new subset will be added as a new row.

3. **Using different measures:** Different measures may be used in the correlation matrix view (e.g., AUC score difference, proportion of positive instances).

4. **Sorting subsets:** Subset rows can be sorted using any measure attributes (e.g., count) to help reveal interesting patterns and spot anomalies.

4.3 Usage Scenario

We present a usage scenario for *MLCube Explorer* to demonstrate how it may help our user Jane, a machine learning engineer working at a search engine company, to build new advertisement click prediction models that advance over an existing model.

Jane uses a public data set from the 2012 KDD Cup competition.[3] It is an advertisement click log from the Tencent search engine, soso.com. Each data instance describes information about a user, an ad, a query, and whether the user clicked the ad. Jane implements some of the learning features presented in the winning team's report [25] and created a few models, including logistic regression, decision tree, and boosted tree.

Recognizing data encoding issue. Jane begins her exploration by visualizing the existing model to understand its performance. She quickly finds that a subset of instances "`user_age_group = 0`" performs distinctly worse than the other age groups, indicated by the left-most solid bar in the accuracy column in Figure 4. To understand why, she examines how the age groups were defined. She realizes that the feature function that generates this feature has encoded null data as 0 and considered this feature as numerical variables. She thinks that this might cause the degraded performance. To fix this issue, she redefines this feature as a categorical variable, instead of a numerical variable, which could improve the performance of the model by separating the null instances from others.

Analyzing the performance improvement. After getting the hints for improving the model performance, she now would like to try different learning algorithms and compare their performance with that of the baseline model. To create a visualization, she sets the baseline boosted tree model as model A (shown in dark yellow in Figure 5) and

[2]http://spark.apache.org/

[3]http://www.kddcup2012.org/c/kddcup2012-track2

Figure 5: Example of analyzing performance improvement. Jane sees model B has significantly improved over model A for the subset "`user_age_group = 0`". She drills down into that subset by clicking it and observes interesting patterns between accuracy and the `tfidf_sim_query_title` feature.

picks a logistic regression model with additional features as model B (shown in green). The visualization shows that, overall, model B outperforms model A. In particular, Jane sees model B has significantly improved over model A for the subset "`user_age_group = 0`" (Figure 5a). To further analyze this subset, she drills down into it by clicking it, and she observes a few important patterns for the `tfidf_sim_query_title` feature[4] (see Figure 5b): (1) the majority of model B's improvement over model A comes from subsets with lower similarity scores (the wide gaps between yellow and green bars) — this means model B is quite accurate even when the advertisement titles are not that similar to the user's search query; (2) the accuracy difference between model A and B decreases as the similarity increases. In addition to the "`user_age_group = 0`" subset, she finds out that model B outperforms model A for several other subsets. Thanks to these discoveries, she decides to look into the model further by creating more variations with different parameters and features.

4.4 System Implementation

We implemented (1) a simple declarative machine learning framework following the pipeline in Section 2, (2) *MLCube* which works on top of the framework (Section 3), and (3) *MLCube Explorer* (Section 4). The framework is implemented based on the pipeline introduced in Section 2 using Python, scikit-learn, and PostgreSQL. Within the framework, we implemented several learning features presented in the report by the winner of the KDD Cup [25] and implemented several models, including logistic regression, decision tree, and boosted tree, also based on the report. As we mentioned earlier, *MLCube* is partially materialized with a algorithm described in [17]. *MLCube Explorer* is written in HTML, JavaScript, and D3.js. It can run on any modern web browser. When a user specifies two created models to compare, the server returns the corresponding *MLCube* in JSON format and the client code generates the visualization.

[4]This feature measures similarity between a query and the title of an ad by representing each text field as a TF-IDF term vector and computing cosine similarity between two vectors [25].

5. RELATED WORK

Importance of model interpretation. As the complexity of machine learning algorithms increases, many researchers have recognized the importance of interpreting the models [12, 11]. Traditional approaches focused on methods that explain specific models, such as computing the importance of each feature and visualizing the internal structure of models [24]. However, these approaches are difficult for explaining complex models and comparing different algorithms [19]. Kulesza et al. [12] presented a list of questions that users may ask about models and used interactive visualizations that explain the reasons for the models' predictions.

Visualization tools. Many researchers have developed interactive visualization tools that help users understand machine learning models [1, 11, 12, 5]. Amershi et al. [1] developed a tool that shows the distribution of instance results and let users examine them by each instance, but this type of visualization often suffers from scalability issues, as it is impossible to visualize all instances in screen. Researchers have utilized features to explain how they affect models [12, 11, 10, 5]: Kulesza et al. [12] used the importance weight of each feature in the Naive Bayes algorithm, and Krause et al. used *partial dependence* that shows the relationships between features and results. Furthermore, to enable users to analyze results not only by predefined features, McMahan et al. [16] presented their internal tool that enables users to define subsets for comparing models. One limitation of current approaches is that they do not consider the entire pipeline, which limits the expressiveness of the subset definition and the understanding of the holistic picture [18]. Our tool aims to enable users to slice the model results in various ways for interactively analyzing and understanding them.

Leveraging data pipeline. Patel et al. [18] presented a development environment for developers to implement classification models. They argued that interactive tools that support the entire machine learning process can accelerate the understanding of models. Sculley et al. [21] also argued that the bottleneck of practical machine learning systems is often caused by the lack of data dependency over the machine learning pipelines. The database community acknowledges the importance of managing data flow. With this expertise, many researchers have studied on helping machine learning engineers perform *feature engineering* [2, 26], but only a few researchers have studied on model selection or interpretation [13, 6]. Chen et al. [6] developed a *prediction cube* framework to examine the effect of features using data cube analysis. Our approach advances over prior work by allowing users to interactively define subsets over attributes or any other intermediate data and developing visualizations for the cubes.

6. ONGOING WORK

This work opens up many interesting scalability challenges. We plan to develop efficient materialization techniques (e.g., by using monotonicity property, parallel computation) [17]. As the cube is accessed by interactive tools, it would also be possible to interactively materialize cubes while users navigate cubes by predicting the next possible user steps as in [9, 15]. In addition, we plan to develop efficient techniques to rank interesting subsets (e.g., subsets with the largest accuracy differences between models) [7, 20, 8].

We will also continue to improve the design of visualization interfaces and interactions to improve the usability and utility of the tool [1, 5]. We plan to conduct a series of user studies to evaluate how our tool can help machine learning engineers ease their workflow of developing effective machine learning models with a deeper understanding of the relationships between data and models.

7. ACKNOWLEDGMENTS

We thank Thomas Dudziak, Hussein Mehanna, Sofus Macskássy, Liang Xiong, and Oliver Zeldin for their advice and feedback. This material is based upon work supported by the NSF Graduate Research Fellowship Program under Grant No. DGE-1148903. This work is also supported by NSF grants TWC-1526254, IIS-1563816, and by gifts from Google, Symantec, Yahoo, eBay, Intel, Amazon, and LogicBlox.

8. REFERENCES

[1] S. Amershi, M. Chickering, S. M. Drucker, B. Lee, P. Simard, and J. Suh. Modeltracker: Redesigning performance analysis tools for machine learning. In *CHI*, pages 337–346. ACM, 2015.

[2] M. R. Anderson, D. Antenucci, V. Bittorf, M. Burgess, M. J. Cafarella, A. Kumar, F. Niu, Y. Park, C. Ré, and C. Zhang. Brainwash: A data system for feature engineering. In *CIDR*, 2013.

[3] M. R. Anderson and M. Cafarella. Input selection for fast feature engineering. In *ICDE*, 2016.

[4] R. Barga, V. Fontama, and W. H. Tok. *Predictive analytics with Microsoft Azure machine learning (2nd Edition)*. Apress, 2015.

[5] M. Brooks, S. Amershi, B. Lee, S. M. Drucker, A. Kapoor, and P. Simard. Featureinsight: Visual support for error-driven feature ideation in text classification. In *IEEE Conference on Visual Analytics Science and Technology*, pages 105–112. IEEE, 2015.

[6] B.-C. Chen, L. Chen, Y. Lin, and R. Ramakrishnan. Prediction cubes. In *VLDB*, pages 982–993, 2005.

[7] M. Das, S. Amer-Yahia, G. Das, and C. Yu. Mri: Meaningful interpretations of collaborative ratings. *Proceedings of the VLDB Endowment*, 4(11), 2011.

[8] M. Joglekar, H. Garcia-Molina, and A. Parameswaran. Interactive data exploration with smart drill-down. In *ICDE*, 2016.

[9] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi. Distributed and interactive cube exploration. In *ICDE*, pages 472–483. IEEE, 2014.

[10] J. Krause, A. Perer, and E. Bertini. Infuse: interactive feature selection for predictive modeling of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1614–1623, 2014.

[11] J. Krause, A. Perer, and K. Ng. Interacting with predictions: Visual inspection of black-box machine learning models. In *CHI*, pages 5686–5697. ACM, 2016.

[12] T. Kulesza, M. Burnett, W.-K. Wong, and S. Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *IUI*, pages 126–137. ACM, 2015.

[13] A. Kumar, R. McCann, J. Naughton, and J. M. Patel. Model selection management systems: The next frontier of advanced analytics. *ACM SIGMOD Record*, 2015.

[14] A. Kumar, J. Naughton, and J. M. Patel. Learning generalized linear models over normalized data. In *SIGMOD*, pages 1969–1984. ACM, 2015.

[15] Z. Liu, B. Jiang, and J. Heer. immens: Real-time visual querying of big data. *Computer Graphics Forum (Proceedings of EuroVis)*, 32(3pt4):421–430, 2013.

[16] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, et al. Ad click prediction: a view from the trenches. In *KDD*, pages 1222–1230. ACM, 2013.

[17] A. Nandi, C. Yu, P. Bohannon, and R. Ramakrishnan. Data cube materialization and mining over mapreduce. *IEEE Transactions on Knowledge and Data Engineering*, 24(10):1747–1759, 2012.

[18] K. Patel, N. Bancroft, S. M. Drucker, J. Fogarty, A. J. Ko, and J. Landay. Gestalt: integrated support for implementation and analysis in machine learning. In *UIST*, pages 37–46. ACM, 2010.

[19] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*. ACM, 2016.

[20] S. Sarawagi and G. Sathe. i3: intelligent, interactive investigation of olap data cubes. *ACM SIGMOD Record*, 29(2):589, 2000.

[21] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison. Hidden technical debt in machine learning systems. In *NIPS*, pages 2494–2502, 2015.

[22] C. Stolte, D. Tang, and P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, 2002.

[23] C. Stolte, D. Tang, and P. Hanrahan. Multiscale visualization using data cubes. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):176–187, 2003.

[24] S. Van Den Elzen and J. J. Van Wijk. Baobabview: Interactive construction and analysis of decision trees. In *IEEE Conference on Visual Analytics Science and Technology*, pages 151–160. IEEE, 2011.

[25] K.-W. Wu, C.-S. Ferng, C.-H. Ho, A.-C. Liang, C.-H. Huang, W.-Y. Shen, J.-Y. Jiang, M.-H. Yang, T.-W. Lin, C.-P. Lee, et al. A two-stage ensemble of diverse models for advertisement ranking in kdd cup 2012. In *ACM KDD Cup Workshop*, 2012.

[26] C. Zhang, A. Kumar, and C. Ré. Materialization optimizations for feature selection workloads. In *SIGMOD*, pages 265–276. ACM, 2014.