Provenance for Data Mining

Boris Glavic
IIT

Javed Siddique
University of Toronto

Periklis Andritsos
University of Toronto

Renée J. Miller
University of Toronto

Abstract

Data mining aims at extracting useful information from large datasets. Most data mining approaches reduce the input data to produce a smaller output summarizing the mining result. While the purpose of data mining (extracting information) necessitates this reduction in size, the loss of information it entails can be problematic. Specifically, the results of data mining may be more confusing than insightful, if the user is not able to understand on which input data they are based and how they were created. In this paper, we argue that the user needs access to the provenance of mining results. Provenance, while extensively studied by the database, workflow, and distributed systems communities, has not yet been considered for data mining. We analyze the differences between database, workflow, and data mining provenance, suggest new types of provenance, and identify new use-cases for provenance in data mining. To illustrate our ideas, we present a more detailed discussion of these concepts for two typical data mining algorithms: frequent itemset mining and multi-dimensional scaling.

1 Provenance for Data Mining

While some related work from the data mining community has considered techniques for visualizing mining results [8], evaluating their interestingness [13], or detecting causal relationships [20], there are no tools that compute mining provenance, that is, the reasons for why and how a certain result was produced. Similar to provenance for databases or workflows, data mining provenance could be defined in different ways with different use-cases in mind. Before discussing the requirements, challenges, and use-cases, note that this paper focuses on provenance for data mining, which is unrelated to previous approaches that apply data mining techniques to compute or analyze provenance [7].

Many provenance models define provenance as a subset of the input data that caused an output of interest to appear in the result of a transformation. For example, a standard database provenance model named *Why*-provenance [6] considers a set of input tuples of a query to be in the provenance of an output tuple if they are *sufficient* to derive the output through the query. Other provenance models use necessity, preservation of equivalence [12], or causality [6] to model these data dependencies between inputs and outputs. For simplicity, and lack of a better term, we will refer to all these models as forms of *why-provenance*.

Why-provenance. The concepts underlying relational why-provenance models (sufficiency, necessity, preservation of equivalence, and causality) are also meaningful for data mining. Retrieving the inputs that influenced a result is especially useful for data mining, because most data mining algorithms generate a small and condensed result from a large input data set. While this reduction is in line with the purpose of data mining (finding useful information in data), it can be problematic, because data reduction is lossy. Provenance can help to selectively recover this information for an output of interest, thus, helping us to better understand the result. Efficiently generating why-provenance for data mining techniques may not be trivial, because of the large number of inputs that influence a result. Furthermore, unless we can generalize the processing of data mining algorithms in terms of their provenance behaviour, efficient approaches for provenance generation would have to be developed from scratch for each such algorithm. One idea to explore in this context is to model data mining operations as workflows or database queries and use standard fine-grained provenance models from these application domains. However, this approach may not be applicable to all data mining algorithms and could be less efficient than specialized provenance tracking algorithms for data mining. While traditional why-provenance can be adapted for data mining, its usefulness is limited by the fact that

(1) the inputs of a data mining algorithm may have been heavily preprocessed and (2) that they do not exploit contextual information.

Tracing provenance through preprocessing. Mining is often preceded by preprocessing such as feature extraction, de-duplication or cleaning. Thus, in addition to tracing the provenance of a mining result back to the relevant inputs, we may want to further track the provenance of the input data through preprocessing steps. Since, preprocessing is often expressed as a workflow (e.g., using an ETL tool) it may be possible to adapt existing workflow provenance approaches such as VisTrails [5], Kepler [4], Taverna [10], or the approach from Amsterdamer et al. [3] for this purpose.

Enriching provenance with "contextual" data. Traditional mining algorithms do not use "contextual" data. For example, assume we are applying a classification algorithm to a data set of Facebook pictures to determine whether photos depict flowers. The algorithm is applied to the raw image data. If additional information about a photo, such as the location or timestamp are available, we could associate it with the provenance (the input photos). For instance, we may realize that photos of flowers taken with snow in the background (cold locations) are often not recognized as flowers. Contextual data is often more useful and manageable than the actual "raw" provenance and, thus, can aid in dealing with the large amounts of provenance related to data mining results (for an example see Section 2). Which part of the context is needed may vary from use-case to use-case. The provenance system should thus enable us to choose the context.

Responsibility for data mining provenance. Certain data mining algorithms, such as clustering, generate a small number of outputs from large input data sets. For instance, the *k-means* [16] clustering technique generates exactly k outputs (the clusters) from an arbitrary number of inputs. Naturally, we can assume that some points inside a cluster have higher influence in its creation than others. Pure why-provenance is not very useful for such algorithms, because each output will depend on a large number of inputs and the influence of an input on the result is not modelled. Attributing an "amount of influence" to an input has been modeled as *responsibility* for boolean expressions - an approach that was recently adopted to database provenance [17] . We argue that it can equally be adopted in data mining provenance. For instance, the responsibility model mentioned above defines responsibility of an input tuple t with respect to an output tuple o and query q as the inverse of the minimum number of inputs that have to be removed (called the minimal *contingency* [17]) before the input t becomes a *counterfactual cause*, *i.e.*, the removal of t causes the output o to be removed from the result of q. At first sight, this concept seems to be directly translatable to data mining algorithms. In clustering, the responsibility of a data point p can be defined as the number of points that we need to remove before p, such that the cluster (containing p) is removed from the output. However, p will always be in some cluster if the clustering represents all data. Hence, to develop a notion of responsibility for clustering, we need to consider the change in the cluster structure after removing one or more points from the input data. For instance, we may consider each point p in a cluster as a cause and calculate its responsibility as the amount of change to the clustering that is caused by removing p. For k-means the change could be defined as the distance between an original cluster mean and a new mean after removal of point p. In contrast to the responsibility model mentioned above, this definition would not take interdependencies between inputs into account (which requires a notion of contingency). Extending this idea, we could consider every set of points as a contingency and calculate the responsibility of a point according to a given contingency in the same manner as for causes, but computed over the clustering that is generated from the input minus the contingency. The global responsibility of a point can then be defined as the sum of responsibilities for all potential contingencies (weighted by the size of the contingencies and normalized by the total number of possible contingencies).

In summary, existing notions of responsibility would have to be recast and adapted for data mining. Responsibility provides a natural way to extract interesting parts from the provenance (we might only return the top-k inputs in the provenance according to responsibility) and, thus, may be used to address the problem of large amounts of provenance attached to a single result. The latter is particularly interesting in clustering since by ranking the influence of input tuples we are able to better understand the cluster quality. If all responsibility scores are similar for the points in a cluster, this means that its quality is high. In other words, it is not easy to find a point to remove from it, without taking away valuable information responsible for its creation.

Parameter vs. Data Responsibility. Most mining algorithms are sensitive to changes in input parameters such as the number of clusters k for k-means clustering [16], the minimum support for frequent itemset mining [2], the distance measure for density based clustering algorithms, or the number of dimensions and fitness-measure for multidimensional scaling [14]. We could define notions of responsibility that exclusively capture the responsibility of data (as discussed above), or alternatively, new notions that model the relative dependence of a result on the data vs. parameter settings (see Section 3 for an example). Conceptually, this type of responsibility is related to approaches in clustering that measure how stable a clustering is as parameters change [15].

How- and process-provenance. So far we have limited the discussion to data provenance, *i.e.*, which parts of the input data influence an output. Workflow provenance and certain types of database provenance (provenance polynomials [12] and transformation provenance [9]) model in which way the relevant input data is used by the transformation or which parts of a transformation (*e.g.*, a workflow) influence the result. This type of provenance, which has been called *process provenance* and sometimes has been termed *how-provenance* (when recording the disjunctive and conjunctive use of data in provenance polynomials), may also be useful in a data mining context. Another line of provenance research that could be applied for this purpose are adaptations of program analysis techniques for provenance (e.g., [18]). Consider the iterative *k-means* clustering algorithm. Starting from randomly chosen cluster means, the algorithm repeats the following two steps until the solution converges. In the first step, each point is assigned to the cluster with the nearest mean according to some given distance function. Then, a new mean for the cluster is computed and this becomes the new centroid of the points in it. One prevalent problem in clustering is handling data sets that contain outliers, that is, points far from their cluster centroid. For example, k-means clustering is sensitive to outliers, because outliers have a disproportionately large effect on cluster means. Assume we define the how-provenance of a point in a cluster produced by k-means as the list of means to which it has been assigned by the algorithm. If we combine how-provenance with a notion of responsibility, the effect of outliers can be detected easily. For example, assume we expected two points to be in different clusters, but the algorithm assigned them to the same cluster. We could search for points that have high responsibility for a mean that is in the how-provenance of both points. Such a point may be an outlier that caused the two clusters to be merged.

Provenance for missing results. Extensions of (mostly database) provenance techniques have been used to determine why a particular answer is not in the result of a transformation (query). Conceptually, these approaches can be classified into two categories: 1) approaches that compute a change to the input data that would cause the missing answers to appear in the result and 2) approaches that determine how to change the query to cause the missing result to show up. The first type can be more or less directly applied to clustering algorithms. The second approach could be adapted to compute how to tweak the parameters of the mining algorithm to generated the missing result. However, both approaches would be prohibitively expensive if we have to rerun a mining algorithm for each modification to the input or parameters. These approaches require incremental adaptation of mining results based on changes to the inputs and parameters. This idea has been used by Ikeda et al. [11] to provide incremental maintenance for workflows based on provenance. Recent work in duplicate detection compactly represents multiple de-duplicated data sets produced from different parameter settings for a duplicate detection algorithm. Arguably, similar ideas could be applied for data mining provenance.

In summary, we analyzed how existing notions of provenance may transfer to data mining, outlined new types of provenance that are useful in a data mining context, and sketched challenges for realizing provenance management for such algorithms. In the remainder of the paper, we discuss more concrete versions of these concepts for two specific data mining algorithms.

2 Frequent Itemset Mining

One of the most prevalent data mining tasks is *Frequent Itemset Mining (FIM)*. Given a set of transactions that are sets of items from a fixed domain $\mathbb{D}$ of items, FIM computes subsets of $\mathbb{D}$, called *frequent itemsets*, which appear in a fraction of transactions above a certain *minimum support* threshold σ.

Why-provenance. The why-provenance of a frequent itemset should model which input transactions are used to construct an output frequent itemset. Arguably, the transactions in the input that contain an itemset I caused I to be frequent. Thus, we could simply define this set of transactions to be the why-provenance of I.

Definition 1 (Why-Provenance) *The why-provenance $\mathscr{W}(I)$ of an itemset I in a database D is the set of transactions containing I: $\mathscr{W}(I) = \{t | I \subseteq t \wedge t \in D\}$.*

FIM is an example of a mining algorithm that operates on a preprocessed input. Usually we can expect the database containing transaction data to store additional information about each transaction (e.g., information about the customer or the store). A meaningful why-provenance model should link this contextual information with the provenance. We envision a provenance management system for FIM that allows a user to select what part of the available data should be linked to the transactions in the why-provenance of a frequent itemset and to ask queries over this information.

Example 1 *A popular example in the frequent itemset mining literature is a survey [1] studying the behavior of young supermarket shoppers. This survey noted that $\{$Diaper, Beer$\}$ is a frequent itemset, i.e., these items are often bought together by shoppers. Though $\{$Diaper, Beer$\}$ may be interesting as it is an unexpected result, to interpret this result we may want to understand* why *it is frequent. Using our definition of why-provenance, we* *would say it is frequent because it appears in a specific set of perhaps one million transactions. However, this set alone does not give us much insight.*

The actual source of the {Diaper, Beer} story [1] cites that this frequent itemset comes from analyzing data from only 25 OSCO drug stores, between 5:00PM and 7:00PM. This information is part of the provenance information for the {Diaper, Beer} itemset as it describes (more intuitively to a human) the set of transactions which contributed to making {Diaper, Beer} frequent. Being able to link the why-provenance with such contextual information (such as the stores in which the transactions took place), can add tremendous value to the mining results. Given a system as envisioned above, the user can retrieve the why-provenance of an itemset and also specify what attributes should be used to describe the provenance to get a more concise (and understandable) description of the why-provenance.

How-provenance. For queries, how-provenance explains how tuples are used and combined in a query. In FIM, transactions are not combined in complex ways, rather it is the way items are combined within transactions that determines what itemsets will be frequent. Hence, for itemsets, we want to explain how items and sets of items co-occur and to give insight into how the transactions supporting an itemset (the why-provenance) are actually giving evidence for the itemset [19].

3 Multi Dimensional Scaling

Multi-dimensional Scaling [14] (MDS) maps a set of observations with pair-wise similarities into an *m*-dimensional space so that the distance of the points representing the observations in that space reflects the pairwise similarities. MDS algorithms usually use a fitness measure to model how well the input similarities are preserved by a mapping. Thus, an MDS problem can be represented as an optimization problem where the goal is to find a solution (layout) with maximal fitness. An example application for MDS is marketing. Customers are asked to rate similarities of products and concepts (e.g., how "classy" is a car) and MDS is used to create a two-dimensional layout that depicts these similarities.

MDS is an example of a data mining algorithm that extracts relevant information from the input by compressing it while trying to preserve the characteristics of the input data. This concept has also been applied by other data mining approaches including dimensionality reduction techniques such as principal component analysis. The loss of information entailed by the compression may cause misinterpretations. For example, if two cars are close in the layout this either means that they are considered similar (according to the similarity matrix generated from customer perceptions) or that they were put close together because this maximized the fitness (or any combination of these two causes).

Even simple why-provenance, e.g., defining the why-provenance of a set of points in the layout as their original pair-wise similarities, would help to distinguish between actual similarities in the data and similarities that are artifacts of the fitness-measure. More complex responsibility-style provenance could be used to further explain which other similarities or parts of the fitness-measure or algorithm caused the similarity in the layout to differ from the "real" similarity.

4 Conclusions

In this paper, we discuss how traditional notions of provenance translate to data mining. We identify new use-cases, the need for novel types of provenance that can be used to better interpret data mining results, and the need to analyze to what extent a result is based on the data vs. based on the parameter choices for the algorithm. We consider how the concept of responsibility may be adapted for data mining algorithms by considering gradual changes to a specific result instead of the existence of that result in the output. We argue that provenance should be enriched with contextual information to improve its utility and to make the large amounts of provenance generated for mining results more manageable. By means of two use cases - frequent itemset mining and multidimensional scaling - we illustrate these generic concepts on concrete mining algorithms.

References

[1] What is the "true story" about data mining, beer and diapers? http://www.dssresources.com/newsletters/66.php.

[2] Agrawal, R., and Srikant, R. Fast algorithms for mining association rules. In *VLDB* (1994), pp. 487–499.

[3] Amsterdamer, Y., Davidson, S., Deutch, D., Milo, T., Stoyanovich, J., and Tannen, V. Putting Lipstick on Pig: Enabling Database-style Workflow Provenance. *PVLDB 5*, 4 (2011), 346–357.

[4] Anand, M. K., Bowers, S., McPhillips, T., and Ludäscher, B. Efficient Provenance Storage over Nested Data Collections. In *EDBT* (2009), pp. 958–969.

[5] Callahan, S., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., and Vo, H. VisTrails: Visualization meets Data Management. In *SIGMOD (demonstration)* (2006), pp. 745–747.

[6] Cheney, J., Chiticariu, L., and Tan, W.-C. Provenance in Databases: Why, How, and Where. *Foundations and Trends in Databases 1*, 4 (2009), 379–474.

[7] Deolalikar, V., and Laffitte, H. Provenance as data mining: Combining file system metadata with content analysis. In *TaPP* (2009), pp. 1–10.

[8] Geng, L., and Hamilton, H. Interestingness measures for data mining: A survey. *CSUR 38*, 3 (2006), 9.

[9] Glavic, B., Alonso, G., Miller, R. J., and Haas, L. M. TRAMP: Understanding the Behavior of Schema Mappings through Provenance. *PVLDB 3*, 1 (2010), 1314–1325.

[10] Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., and Oinn, T. Taverna: A Tool for Building and Running Workflows of Services. *Nucleic acids research 34*, Web Server issue (2006), W729.

[11] Ikeda, R., Salihoglu, S., and Widom, J. Provenance-based refresh in data-oriented workflows. In *CIKM* (2011), pp. 1659–1668.

[12] Karvounarakis, G., and Green, T. Semiring-annotated data: Queries and provenance. *SIGMOD Record 41*, 3 (2012), 5–14.

[13] Keim, D., and Kriegel, H. Visualization techniques for mining large databases: A comparison. *TKDE 8*, 6 (1996), 923–938.

[14] Kruskal, J. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika 29*, 1 (1964), 1–27.

[15] Kuncheva, L., and Vetrov, D. Evaluation of stability of k-means cluster ensembles with respect to random initialization. *TPAMI 28*, 11 (2006), 1798–1808.

[16] MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Fifth Berkeley Symposium on Mathematical statistics and probability* (1967), pp. 281–297.

[17] Meliou, A., Gatterbauer, W., Halpern, J., Koch, C., Moore, K., and Suciu, D. Causality in databases. *IEEE Data Engineering Bulletin 33*, 3 (2010), 59–67.

[18] Perera, R., Acar, U., Cheney, J., and Levy, P. Functional programs that explain their work. In *SIGPLAN* (2012), pp. 365–376.

[19] Siddique, J., Glavic, B., and Miller, R. J. Provenance management for frequent itemsets. Tech. rep., http://dblab.cs.toronto.edu/project/provenance4mining.

[20] Silverstein, C., Brin, S., Motwani, R., and Ullman, J. Scalable techniques for mining causal structures. *Data Mining and Knowledge Discovery 4*, 2 (2000), 163–192.