Introduction

Whenever we approach the so-called "data mining" problem, we realize it means different things to different people. Scientists and analysts — the consumers of algorithms and data products — relate to the various tasks: pattern recognition, structural organization, regression, anomaly finding, and so on. On top of that, we as computer scientists — producers of algorithms and tools — break it down to its building blocks: statistics, computational complexity, and knowledge management.

On first glance, it would seem this disparity has the potential for many false expectations and impossible requirements. But the truth is that this very tension is what advances research in the field. Here is how it typically happens. A scientist has had access to some source of data, say experiments performed in his lab. Over time he had accumulated a set of tools and techniques to analyze it. But recently, the amount of data has become much larger. Possibly, new internet-based collaboration points give him easy access to the results of other researchers' work. Or perhaps new machinery and methods are producing data orders of magnitude better — and faster — than before. The Sloan Digital Sky Survey is a prime example of this. The goal is to map, in detail, one-quarter of the entire sky. The estimated size of the catalog, due to be completed in 2007, is 200 million objects, including images and spectroscopic data. The database will then encompass 5 terabytes of catalog data, and 25 terabytes of data overall.

The unforeseen outcome of such endeavors is that suddenly, the old tools become useless. It might be because their theoretic complexity is poor and they blow up on large inputs. Or because study of a single experiment is no longer interesting, when one can potentially draw conclusions based on thousands of similar observations. Or because the rate at which new results come exceeds the ability of an expert to internalize it all, as the old summarization and visualization methods are inadequate.

This is the light under which the issues addressed in this work are best viewed. Fundamentally, it deals with the difficulties of computer-literate and resourceful scientists in a new world of abundant data. More concretely, we break this down into several distinct components, each attempting to solve an admittedly small aspect of the problem. The unifying element is the task - *clustering*. Historically, this is task that does not have a good definition that is both general and statistically rigorous. We offer the intuitive definition of partitioning a given unlabeled data set into groups such that elements in each group are somewhat more similar to each other (in some undefined measure of similarity) than they are to elements in other groups.[1]

Clustering can help in understanding the nature of a given data set in several ways. First, the membership function by itself is meaningful, as it allows further research involving just the part of the data that is of interest. For example, large-scale cosmology simulations as well as recent astronomical surveys enable computation of the correlation between the number of galaxies in a galaxy cluster, and the amount of dark matter in it ("halo occupation distribution"). But before doing this, the mapping from each galaxy to its owning cluster needs to be established.

A second potentially useful output is the number of clusters, if it is estimated by the algorithm. This can serve as a characteristic of the data. Again we relate to the universe example above for an example. By looking at the distribution of galaxy

[1]Later we weaken this definition even further by considering the extension where each element does not have to fully belong to just one class.

cluster sizes, one can "profile" a given universe. This is potentially useful when judging if a universe simulated from specific parameters is similar to the observed universe (and also when analyzing the effect of changes to the simulation parameters). Here, the number of clusters — typically in the order of thousands — clearly has to be estimated from the data.

Third, the very description of the clusters defines subregions of the data space. These descriptions can be used to achieve insights into the data. For example, if the regions are convex, one can come up with unseen examples that would be included in a given cluster. This ability can be useful for computer program verification tools which aim to increase test-suite coverage.

Fourth, if the statistical model fitted to the data is a probability density estimator, it can be used in a variety of related tasks. All the models described in this work meet this criterion. Below we show how to use such models in an anomaly-hunting task.

Note that we assume here that clusters form a flat hierarchy. This is somewhat arbitrary, as a huge body of existing work deals with hierarchical clustering and fitting of taxonomies. Much of that work is focused on information retrieval. Therefore, it is sufficiently different from the kinds of data analyzed here to be outside the scope of this work.

Clustering is used in a multitude of application areas. Some of them are:

- Large-scale cosmological simulations.
- Astronomical data analysis.
- Bioinformatics.
- Computer architecture.

- Musical information retrieval.
- Verification of computer programs.
- Natural language processing.
- Epidemiology.
- Highway traffic analysis.

Diverse as they are, in all of them we encounter similar phenomena. First, labels for individual samples are rare or nonexistent (and too costly to obtain in the general case). Second, the data is too voluminous to be entirely eyeballed by a human expert. In fact, often it is too voluminous to even process mechanically quickly enough. To illustrate the last point, consider an anomaly-hunting application which asks a human expert for labels for a very small number of examples. Given those, it refines the statistical model using the given examples and the full data set, and the cycle repeats. Regardless of data set size, the computer run needs to finish quickly, or else the expert would lose concentration.

Returning to the historical angle, most of the data analysts are already familiar with some clustering method or another. The problem is that it is too slow on big inputs. Generally, there are three approaches to address the speed issue:

1. Develop new algorithms and data-organization methods, such that the statistical qualities of the data can be approximated quickly. Use the approximated measures to generate output in the same form as the existing algorithms.
2. Develop exact algorithms that output the exact same answer as the original method. Enhance the data organization to support this kind of operation.

3. Develop near-exact algorithms using advanced data organization. Allow a user-defined degree of error, or a probabilistic chance of making a mistake. Typically those parameters will be very small.

The first example of the first approach is BIRCH (Zhang et al., 1995). It is an approximate clusterer optimized for on-disk storage of large data sets. The clusters are grown in a heuristic way. Very little can be said on the quality of the output clusters, or about their difference from those obtained by some other method.

Another example of the first approach is sub-sampling. The idea is simple: randomly select a small population from the input set and run the algorithm of choice on it. A variant of this uses the results together with the original data set as if they were created directly from the original set. For example, one might create clusters based on a small sample, and then use the cluster centroids (or any other meaningful property) to assign class membership to points in the original data.

Often, this approach is taken without much consideration of the statistical consequences. Not surprisingly, they can be severe. For example, the 2-point correlation function is used in cosmology to characterize sets of astronomical objects. It is well-known that for the rich structure observed in our universe, straightforward sub-sampling does not preserve the 2-point correlation function. And the same most likely holds for other measures.

My conclusion is that there is merit in expending the effort to develop schemes that can handle large data without affecting output quality. When this is too hard, we would still like to bound the error in some way. This work aims to show that this goal is achievable.

Below I describe how to accelerate several known algorithms, such that their output can be used in exactly the same way as the output from the respective original published versions. Empirical evaluation shows great speed-ups for many of them —

often two orders of magnitude faster than a straightforward implementation, measured on actual data sets used by scientists. In one case the run time is even sub-linear in the input size, and only depends on intrinsic properties of the data.

Chapter **??** looks at the familiar K-means algorithm and shows how it can be accelerated by re-structuring the data. The output is exact (meaning the same as it would be for a non-optimized algorithm). Chapter **??** uses the same fast data structure to build a framework supporting estimation of the number of clusters K. The framework exploits the data structure to accelerate the statistical test used for model selection. It is also general in the sense that it allows a variety of statistical measures for scoring and decision between different models. Chapter **??** takes a detour to look at human comprehensibility. It proposes a new statistical model which lends itself to succinct descriptions of clusters. This description can be read by a domain expert with no knowledge of machine learning, and its predicates can be interpreted directly in the application domain. In Chapter **??** we return to dealing with a well-known statistical algorithm. This time we focus on dependency trees as grown by popular the Chow-Liu algorithm and propose a "probably approximately correct" algorithm to fit them. It can decide to consider just a subset of the data for certain computations, if this can be justified by data data already scanned. In practice, this typically happens very quickly, resulting in large speed-ups. In Chapter **??** we consider the task of anomaly-hunting in large noisy sets, where the classes containing the anomalies are extremely rare. For help, we consult an oracle for labels for a very small number of examples, which naturally touches on active learning. This work uses the fast dependency tree learner, however it is not dependent on it and can use other models as components. Finally, in Chapter **??** I describe a visual tool based on these ideas, which enables an expert to interact with the data and find anomalies quickly.

Bibliography

Zhang, T., Ramakrishnan, R., & Livny, M. (1995). BIRCH: An efficient data clustering method for very large databases,. *Proceedings of ACM SIGMOD* (pp. 103–114).