Data Exploration

Heli Helskyaho

21.4.2016

Seminar on Big Data Management
References


Introduction to Data Exploration

• Data exploration is to efficiently extract knowledge from data, even though you do not know what you are looking for.

• *Exploratory computing*: a conversation between a user and a computer.
  – The system provides active support; “step-by-step “conversation” of a user and a system that “help each other” to refine the data exploration process, ultimately gathering new knowledge that concretely fulfills the user needs”
  – exploration process is investigation, exploration-seeking, comparison-making, and learning
  – Wide range of different techniques is needed (the use of statistics and data analysis, query suggestion, advanced visualization tools, etc.)
Not a new thing...

• J. W. Tukey. Exploratory data analysis. Addison-Wesley, Reading, MA, 1977: “with exploratory data analysis the researcher explores the data in many possible ways, including the use of graphical tools like boxplots or histograms, gaining knowledge from the way data are displayed”
Problems and new problems

• We do not know what we are looking for
• Must support also non-technical users (journalists, investors, politicians,...)
• More and more data
• Different data models and formats
• Loading in progress while data exploration going on
• All must be done efficiently and fast
The Conversation

• To start the conversation the system suggests some initial relevant features.
• A feature is the set of values taken by one or more attributes in the tuple-sets of the view lattice, while relevance is defined starting from the statistical properties of these values.
The AcmeBand example

• AcmeBand, a fitness tracker
  – a wrist-worn smartband that continuously records user steps, and can be used to track sleep at night
Example conversation

• (S1) “It might be interesting to explore the types of activities. In fact: running is the most frequent activity (over 50%), cycling the least frequent one (less than 20%)”

• (S2) “It might be interesting to explore the sex of users with running activities. In fact: more than 65% of the runners are male”

• (S3) “It might be interesting to explore differences in the distribution of the length of the running activities between male and female. In fact: male users generally have longer running activities”.
(S1) “It might be interesting to explore the types of activities. In fact: running is the most frequent activity (over 50%), cycling the least frequent one (less than 20%)”

The user likes it and picks Running
The Computer suggests S1.1

• (S1.1) “It might be interesting to explore the region of users with running activities. In fact: while users are evenly distributed across regions, 65% of users with running activities are in the west and only 15% in the south”.

• □Type=running (Location □ AcmeUser □ Activity)

• The user likes it and picks South

• □Type=running, region=south (Location □ AcmeUser □ Activity)
The Computer suggests S1.1.1

• (S1.1.1) “It might be interesting to explore the sex of users with running activities in the south region. In fact: while sex values are usually evenly distributed among users, only 10% of users with running activities in the south are women”.

• The user likes it

• □ Type=running, region=south, sex=female (Location □ AcmeUser □ Activity)
How to proceed?

• Now the user is exploring the set of tuples corresponding to female runners in the south
• He/she may
  – ask the system for further advices
  – browse the actual database records
  – use an externally computed distribution of values to look for relevant features
Adding an external source

• Maybe the user is interested in studying the quality of sleep and downloads from a Web site an Excel sheet stating that over 60% of the users complain about the quality of their sleep
• That data will be imported to the system
• the system suggests that, unexpectedly, 85% of sleep periods of southern women that run are of good quality
• The user has learned something new from the data
Challenges in short

- Usability
- Performance
The Process

• the system populates the lattice with a set of initial views based on tables and foreign keys
• Each view node is a concept of the conversation
  – The table Activity is a concept ”activity”
  – the join of AcmeUser and Location is a concept of “user location”
• The system builds histograms for the attributes of these views, and looks for those that might have some interest for the user
  – It may be relevant if it is different from, or maybe close to, the user’s expectations
  – the users’ expectations may derive from their previous background, common knowledge, previous exploration of other portions of the database, etc.
• Different tests are used to define the interest (test(d,d’))
• Depending the position in the lattice the test is different
• The notion of relevance is based on the frequency distribution of attributes in the view lattice.
These tests could be for instance:

- **the two-sided t-test (two-tailed t-test)**, assessing variations in the mean value of two Gaussian-distributed subsets.
- **the two-sided Wilcoxon rank sum test**, assessing variations in the median value of two distribution-free subsets.
- **the one/two-sample Chi-square test**, for categorial data comparison, assessing the distribution of a subset with discrete values w.r.t. a reference distribution or assessing variations in proportions between two subsets with discrete values.
- **the one/two-sample Kolmogorov-Smirnov test**, to assess whether a subset comes from a reference continuous probability density function or whether two subsets have been generated by two continuous different probability density functions.
Faceted search

• One of the techniques we just used is called a *faceted search* (faceted navigation, faceted browsing)

• Wikipedia: ”a technique for accessing information organized according to a faceted classification system, allowing users to explore a collection of information by applying multiple filters”

• In our example Activity is a facet and Running is a constraint or a facet value

• The user can browse the set making use of the facets and of their values and will inspect the set of items that satisfy the selection criteria.
We need Fast Statistical Operators

• We also need the availability of fast operators to compute statistical properties of the data and ability to compare them
• For example subgroup discovery to discover the maximal subgroups of a set of objects that are statistically “most interesting” with respect to a property of interest
• critical step is a statistical algorithm to measure the difference between two tuple-sets with a common target feature, in order to compute the relative relevance
The difference between two tuple-sets

• Four steps that are conducted iteratively:
  – Sampling
  – Comparison
  – Iteration
  – Query ranking
Step 1 – Sampling

• sample tuple sets $T^{Q_1}$ and $T^{Q_2}$ to extract subsets $q_1$ and $q_2$ of cardinality much lower than the one of $T^{Q_1}$, $T^{Q_2}$, i.e., $|q_i| << T^{Q_i}$.

• Different sampling strategies:
  – sequential
  – random
  – hybrid
Step 2 – Comparison

- Let $X_1$ and $X_2$ be the projections of $q_1$ and $q_2$ over a specific attribute (feature).
- The data in $X_1$ and $X_2$ can be either numerical or categorical.
- The comparison step aims at assessing the discrepancy between $X_1$ and $X_2$ through theoretically-grounded statistical hypothesis tests, of the form $test_i(X_1, X_2)$. 
Step 3 – Iteration

• Repeat the extraction and comparison steps $M$ times.
• At the $j$-th iteration, a new pair of subsets $X_1$ and $X_2$ are extracted and $\text{test}_i(X_1, X_2)$ is computed.
• If the test *rejects the null hypothesis*, we stop the incremental procedure since we have enough statistical confidence that there is a *difference* in the data distributions of $T^{Q_1}$ and $T^{Q_2}$.
• Otherwise, the procedure proceeds to the next iteration.
Step 4 – Query ranking

- The difference between empirical distributions of the pairs chosen is computed using the Hellinger distance*. Based on this, we can rank the tuple sets to find out those exhibiting the largest differences.

*Hellinger Distance: For probability distributions $P = \{p_i\}_{i \in [n]}$, $Q = \{q_i\}_{i \in [n]}$ supported in $[n]$, the Hellinger distance is $h(P,Q) = (1/\sqrt{2}) \| \sqrt{P} - \sqrt{Q} \|_2$
Challenges for the User Interaction

• The user interface should help the user to find information she/he was not aware of and to find even more information based on the information already found.

• The user interface should suggest points of views to look at on the data and based on users choices suggest more points of views.

• The data is in different formats, it is a large amount of different kinds of data and analyzing it takes time.

• The conversation must be fluently responsive.

• Not all users are IT professionals

• The query result visualization to understand the data and to communicate with the exploration software would be valuable
User Interaction

1. Systems that assist SQL query formulation
2. Systems that automate the data exploration process by identifying and presenting relevant data items
3. Novel query interfaces such as keyword search queries over databases and gestural queries
4. Visualization
User Interaction, Exploration Interfaces

- Discover data objects that are relevant to the user
  - User modeling approaches: online classification based on relevance-feedback, models built by variations of facet search techniques

- Assist users formulate their exploratory queries
  - Systems designed for users who are not aware of the exact query predicates but who are often aware of data items relevant to their exploration task or tuples that should be present in their query output

- Teach users to build complex queries based on example tuples

- Techniques for recommending SQL queries

- For example dbTouch or GestureDB are visual tools for specifying relational queries and for processing data directly in an exploratory way
The role of visualization for data analytics is huge (a picture tells more than 1000 words)

- visualization tools that assist users in navigating the underlying data structures
- tools that incorporate new types of interactions such as collaborative annotations and searches
- Query result reduction (aggregation, sampling and/or filtering operations)
- Query result visualization
- a novel declarative visualization language to bridge the gap between traditional database optimizations and visualization
User Interaction, User Involvement

• the relevance of an answer for a specific user
  – predict users’ interests and anticipations in order to issue the most relevant (possibly serendipitous) answer.
  – classify different notions of relevance
  – devise strategies to customize the system response to the user behavior
  – User groups (subgroups)

• raises the important issue of developing intuitive visualization techniques
User Interaction, Responsiveness

• The conversation must be fluent
  – users will not tolerate long computing times
  – Fast, incremental histogram computation is a good example of a technique that can be effectively employed for speeding up the assessment of relevance in a conversation step.
The Performance
Data Size

• Summarizing: fully pre-computing the lattice is not possible and not smart.
• The size of data summarized must be large enough to estimate statistical parameters and distributions, but manageable from the computation viewpoint.
• Data mining techniques.
• Histograms, presenting a histogram algebra for efficiently executing queries on the histograms instead of on the data, and estimating the quality of the approximate answers.
• Effective pruning techniques for the view lattice: not all node-paths are interesting from the user viewpoint, and the ones that fail to satisfy this requirement should be discarded as soon as possible.
Different Layers

- the user interaction/user interface
- the middleware
- the database layer/engine

- Each of these PLUS their combinations
Middleware

• Building various optimizations on top of the database layer/engine
• Can be used to improve the efficiency of the data exploration without changing the underlying architecture
• The exploration tasks are computationally heavy and can be speedup in the middleware layer with Data Prefetching (and background execution) and Query Approximation
• Both of these have their challenges due to large and heterogeneous data sets.
Middleware, Data Prefetching

- *caching* data sets which are likely to be used
- the *trade-off* between introducing new results and re-using cached ones
- the main *challenge* is identifying the data set with the highest utility
Middleware, Query Approximation

- The system offers approximate answers
- Allows users to get a quick sense of whether a particular query reveals anything interesting about the data
- Need solutions that process queries on sampled data sets to provide fast query response times
- the *trade-off* between results accuracy and query performance
Problems with Database Engine/Layer

• Problems with both saving and retrieving the data
• The way the data is stored defines the best possible ways of accessing it
• We do not know what we are looking for and therefore are unable to define the ”best” way of storing
• how can the architecture of database systems be redesigned to aid data exploration tasks?
Database Engine/Layer, Solutions

1. Adaptive Indexing
2. Adaptive Loading
3. Adaptive Storage
4. Flexible Architecture
5. Architectures tailored for approximation processing
Adaptive Indexing

- creating indexes incrementally and adaptively during query processing based on the columns, tables and value ranges that queries request
- Indexes are built gradually; as more queries arrive indexes are continuously fine-tuned
Adaptive Loading

• not all data is needed
  – users might start querying a database system before all data is loaded
  – users might want to leave some parts of the data unloaded
Adaptive Storage

• There is no one perfect solution for storage layout for all the data.
• Adaptive Storage can be used for different needs of storage layouts which will be useful since in data exploration we do not know while loading the data what storage layout is the best for that data set.
• Examples
  – OctopusDB (a central log and Storage Views)
  – H₂O (supports multiple storage layouts and data access patterns, decides during query processing, which design is best for classes of queries)
Flexible Architecture

• can tune the architecture to the task at hand
  – by having a declarative interface for the physical data layouts
  – for the whole engine
  – using organic databases to continuously match incoming data and queries
    • Start with a schema that is ”good-enough”, schema evolution while loading and querying
CPUs vs GPUs

- Fast implementations of statistical computations over the GPU (graphics processing unit)
- Dividing the workload to CPU and GPU have been given promising results

“CPU VERSUS GPU: A simple way to understand the difference between a CPU and GPU is to compare how they process tasks. A CPU consists of a few cores optimized for sequential serial processing while a GPU has a massively parallel architecture consisting of thousands of smaller, more efficient cores designed for handling multiple tasks simultaneously.” - http://www.nvidia.com/object/what-is-gpu-computing.html
Architectures tailored for approximation processing

• Approximate processing is an important tool to support data exploration

• An architecture where storage and access patterns are tailored to support sampling-based query processing efficiently
  – efficient access and updates of sampled data sets
  – In the DB core

• DICE combines sampling with speculative execution in the same engine

• Exploiting this knowledge to design new engines is one of the open challenges.
Conclusion

• Data exploration is about finding new information and knowledge from the data without knowing what you are looking for.
Conclusion

There are a lot of problems related to data exploration:

1. Non-technical people must be able to use it and that gives high demand on the user interface.
2. We do not know what we are looking for from the data so the exploration techniques must be good and improving on every search.
3. The data is large and in various formats and data models so searching that kind of heterogeneous data fast and efficiently is not easy.

• That gives challenges to all the levels: user interface, middleware and the database layer.
Conclusion

• The two main problems:
  1. Usability
  2. Performance