Automated Feature Extraction from Databases for Data Analysis and Modelling

Diego Arenas
da60@st-andrews.ac.uk
School of Computer Science
University of St Andrews
St Andrews, Scotland, UK

Simón C. Smith
artificialsimon@ed.ac.uk
Institute of Perception, Action and Behaviour
The University of Edinburgh
Scotland, UK

CCS CONCEPTS

• **Theory of computation → Data integration**; • **Information systems →** *Join algorithms*; Mediators and data integration.

KEYWORDS

databases, metadata, probability, feature extraction, dimensionality reduction

1 EXPLORATORY DATA ANALYSIS

Database modelling and integration are challenging tasks when there is little or no prior knowledge about the structure of the data in the repositories. To help data scientists with these tasks, we focus on automatic feature extraction. These features are useful as prior knowledge to make better decisions on how to integrate different data sources and for choosing the most suitable variables for data modelling.

We propose a, to the best of our knowledge, new feature to quantify the relationship between data sets assuming same domains. This feature is defined as a two-way relation of overlapping data weighted by its local frequency.

For large databases, the design and implementation of tools to extract the relevant relations can be a time-consuming task. We present an automated solution for relational databases that traverses the data and extract features with minimum human intervention. This tool analyses the database, with a set of traditional measures taken from classical statistics and information theory.

Some of the advantages of the use of metadata include less storage space, less network traffic, and has the potential of increment the performance for big databases with sparse columns as will only use the count of unique values.

Exploratory data analysis is a field with several applications in different domains. Several methods have been proposed in the field of statistics related with exploratory data analysis [4]. Reverse engineering approaches [3] on relational databases [1] may help to extract relevant information from unknown data sources.

Discovering relationships between tables without prior knowledge of their content can be a challenging task. Computing measures of proximity and/or dissimilarity between nominal attributes are good approximations [2].

2 AUTOMATED FEATURES EXTRACTION

Exploring new data sources can be encapsulated in an repeatable algorithm. The analysis we can apply depends on the data types of the columns. For numerical columns we can compute some stats such as average, median, standard deviation, variance, range, quartiles, and percentiles. We can compute the frequency of the data values for discrete variables. For time series data, we can organise it in bins to explore trends.

Rather than querying the whole set of data every time, the proposed library creates a metadata based on the features extracted from the database. To build the metadata it is necessary a full scan of each column at least once, after that the exploration and analysis can be fully performed using the metadata database.

We assume a client-server software architecture where most of the processing occurs in the source server side. First, the program will generate SQL queries using data from the information schema of the source database. Second, the library will send these queries to the source server to be processed and wait for the results to insert them into the metadata database. Only when sampling data from the source server, the program will process data. This processing may happen when the table is too big and a sampling of the data can give an approximate distribution of the data and statistics.

The extracted metadata is:

- For each table: server name, catalogue name, schema name, and table name, number of rows, and number of columns.
- For each column: column name, data type, ordinal position, number of unique values, and number of NULL values.
- For each discrete column: number of values.
- For each numeric column: average, standard deviation, variance, maximum, minimum, range, percentiles 1,

2.5, 5, 10, 25, 50, 75, 90, 95, 97.5, and 99, kurthosis, skewness.

- For each time-series column: frequency per month.
- For each column, the entropy and mutual information regarding the other columns of the table.

The Auto-EDA library

The automated exploratory data analysis library (auto-EDA) simplifies the process described in this pages. The library only requires two database connections, one to the source database and one to the metadata database and will generate the SQL queries to extract the metadata.

JOIN operation by addition

We propose an attribute overlapping meassure. For each column we will have 1) its unique data values, 2) the number of appearances of that unique value in the table or its frequency, and 3) the frequency percentage, calculated as the individual frequency divided by the total number or rows or the sum of the individual frequencies of the data values of the column.

We use the frequency to compute joins between columns. **We join two tables using its metadata, we compare the data values from both tables and then we add up the frequency values**. This operation will always be a 1-to-1 operation as the metadata stores unique values.

Query 1. `SELECT SUM(T1.FREQUENCY_PERCENTAGE) AS A_IN_B FROM DATA_VALUES AS T1 LEFT JOIN DATA_VALUES AS T2 ON T1.COLUMN = ‘c’ AND T2.COLUMN = ‘c’;`

Query 2. `SELECT SUM(T2.FREQUENCY_PERCENTAGE) AS B_IN_A FROM DATA_VALUES AS T1 RIGHT JOIN DATA_VALUES AS T2 ON T1.COLUMN = ‘c’ AND T2.COLUMN = ‘c’;`

From *Query 1* we know the percentage of overlapping, similar values, between the columns c in A and B, which can be translated as the join operation with less data in the process. We have similar information, but in the opposite direction using *Query 2.*

We can derive some use cases from the frequency:

Use case 1: To check if two columns from different tables can join or not. This operation is a cartesian product search on 1-to-1 relationships. This dimensionality reduction is useful for processing large data sets.

Use case 2: To check for similar attributes among different tables comparing the stats of the numerical variables and, in the same way, comparing the data values and adding the frequency up to a certain threshold to recommend them to the analyst as a candidate relation.

Use case 3: Join between tables with many-to-many relationship. Using unique data values with the frequency data the joins are evaluated with summarised data creating 1-to-1 relationships making it possible to evaluate a join between tables with a many-to-many relationship. Assuming that exists a third table with unique values that joins both tables.

Use case 4. Fast discovery of data integration among multiple databases and different DBMS. As the metadata database keep information about multiple sources, this can be used to check for relationships between the data from different databases. Also, a sub-case would be when we have a set of source database and we add a new database to the project. With this tool, the check for compatibility or relationships in the new data source would require less steps as half of the metadata has already been collected.

3 RESULTS AND CONTRIBUTION

The main results and contributions are:

- The introduction of a classification meassure to discriminate the attribute overlapping among variables of different source.
- An open source automated Python library to extract information from databases.[1]
- A reduced dimensionality summary of the original database. Allowing for fast query of relationships among databases, tables and columns.

REFERENCES

[1] Edgar F Codd. 1970. A relational model of data for large shared data banks. *Commun. ACM* 13, 6 (1970), 377–387.

[2] Jiawei Han, Jian Pei, and Micheline Kamber. 2011. *Data mining: concepts and techniques.* Elsevier.

[3] Nadira Lammari, Isabelle Comyn-Wattiau, and Jacky Akoka. 2007. Extracting generalization hierarchies from relational databases: A reverse engineering approach. *Data & Knowledge Engineering* 63, 2 (2007), 568–589.

[4] Chong Ho Yu. 1977. Exploratory data analysis. *Methods* 2 (1977), 131–160.

[1] https://github.com/darenasc/auto-eda