MEng Thesis Proposal: The Data Science Machine

Max Kanter

Friday 19th December, 2014

1 Introduction

As more data is collected in all facets of life, data science is increasing in relevance. Data collection is a crucial part of a wide range of disciplines from life science to finance to education to ecommerce and many more. Making sense of all this data is the job of a data scientist.

Currently, generating insights can be a time consuming process. It involves preprocessing data, generating hypotheses to test, engineering features, and training models. Automatically performing the job of an expert data scientist would have a large impact on most industries.

Because of their potential for impact, I seek to explore several questions. How can we reduce time to process, analyze, and derive insights from the data? How close can we get to a general purpose system to handle the job of a data scientist? Which parts of the insight generating process is human interaction most important? This proposal presents The ”Data Science Machine” (The DSM) to answer these questions.

2 Related Work

Automating data science is something that others have seen as valuable. Work on providing value though automated data science is broken up between academic and commercial work.

2.1 Academic

BayesDB is a database that allows queries to return probabilistic results [1]. It is powered by CrossCat which uses a hierarchical, nonparametric Bayesian model to estimate the full joint distribution over variables in a given table. The results of estimating the conditional probability distributions are made available through the Bayesian Query Language (BQL), which is similar to SQL, but allow the developer to infer missing values and simulate new rows.

There are a few major difference between the proposal for The DSM and BayesDB. BayesDB uses a fixed model for how the data is generated. This means that it has difficult representing certain types of data like time series. The authors mention that supporting this type of data still requires traditional machine learning techniques. Even more, BayesDB requires data to be stored in BayesDB as opposed to traditional databases. Finally, BayesDB does not take advantage of cross table relationships to improve results.

Another approach is that of the The Automatic Statistician [2]. It has similar goals as The Data Science Machine to automatically generate insights on data. It does this by using Gaussian Processes to represent an unknown regression. Currently, it produces reports on time series data. The reports step through explaining the data by adding one component at a time that best explains the data.

2.2 Commerical

There are several commerical solution that make claims to automatically generating insights. It's hard to determine the exact machinery behind their results because they are closed source.

Automated Insights (http://automtegrationatedinsights.com) creates natural language reports with statistics. The report it creates are impressive, but appear to only contain simple statistics rather than harder problems such as prediction.

Ayasdi (http://www.ayasdi.com) uses topological methods based on Gunnar Carlsson's research at Stanford. It appears to be use those methods to reveal to users important patterns or clusters in the data. It is important because it tries to use relationships between part of a data set to reveal insights.

Looker (http://www.looker.com) is noteworthy because it connects directly into a company's data source. It doesn't produce results automatically, but the company validates the idea that it is important to make the integration of a data analytics system easy for commercial adoption.

3 Proposal

For my thesis, I propose The Data Science Machine, which is an automated general purpose system to extract data insights from relational databases. The goal of The DSM is to automatically generate data insights from data. There are four guiding design ideas that make the system unique from past work

1. **Allow flexibility on how data gets into the system** The DSM expect data to enter the system in the form of a relational database.
2. **Use as all data available for analysis** This means not only using the values of the data, but the relationships between values.
3. **Reusable feature engineering** The DSM generates a large set of candidate features from predefined recipes. It assumes the machine learning model will handle the feature selection.
4. **Provide actionable insights to the end user** The DSM will go beyond just summarizing the data and providing basic statistics.

3.1 Design Overview

The system bundles all the necessary functionality together in a 3 stage pipeline as follows

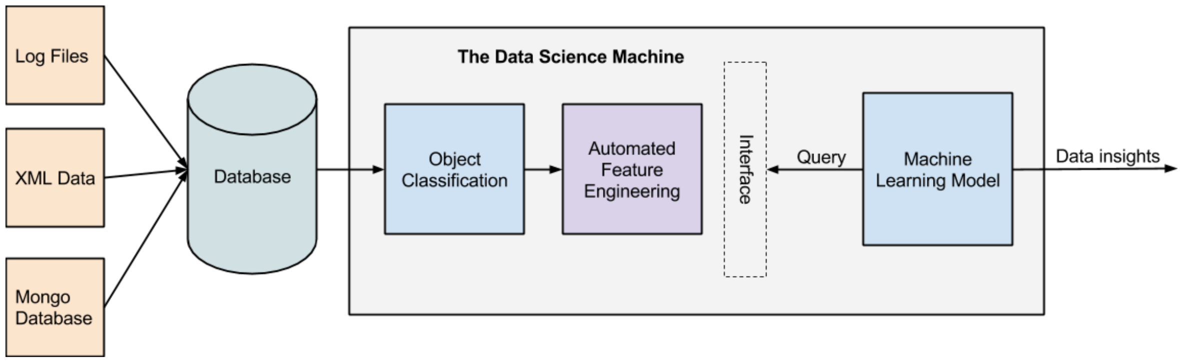


Figure 1

1. Identify entities and transactions relationships in a database
2. Construct new candidate features to describe and summarize the data
3. Provide end user data insights using machine learning methods.

4 Database

The data necessary to run analysis may come from many disparate sources. Those sources could be different types of existing databases, log files, or third party data providers. The Data Science machine assumes that this data has been processed and put into a standard schema.

I plan to focus on relational databases. This is because relational databases enforce schemas that The DSM will take advantage of in entity and transaction classification. The database schema may be determined before or after the decision to use The DSM. Therefore, it is important to be careful about the assumptions about how the database is structured. For now, the DSM will assume that the design of the data is sensible and tries to follow schema design standards

For instance, The DSM assumes the database is normalized. This means the database is divided into tables such that column values are not repeated. Often data is denormalized for performance reasons. It is likely that this case will work the proposed design, just not as well.

5 Object Classification

The only requirement to be an object is to have an unique identifier. An object is the general terms for a row in a data base. Optionally, an object can contain metadata about the object itself or fields that reference other objects. There are two types of objects in The DSM: entities and transactions.

An **entity** is an object that has a mapping between an an id and properties. There is only one instance of an entity in a database. For example, a row of a user table is an entity. It is possible that the the information for an entity is contained across multiple tables of the database.

A **transaction** is an object with a relation to one or more objects. There can be multiple instances of a transaction that have the same relations. For example, a purchase on an ecommerce site. It has relations to the user entity that made the purchase, the product entity that was purchased, as well as properties such as price paid, time of purchase, etc. The might be multiple purchases by the same user for the same product, but at different times.

An object template is a set of descriptors that defines how to construct a feature vector for an object. For a given type of entity or transaction all columns must be the same. There are two different ways to go from an object to feature vector depending on if the object is an entity or transaction. For entities, the feature vector is just the columns for a row. For transactions, the feature vector is the columns for the row concatenated with the columns of any foreign key relationships.

5.1 Classification algorithm

In relational database there are three types of relationships. One-to-one, one-to-many, many-to-many. Using existing software libraries[1][2], The DSM can extract these relationships between tables in the database. All objects in a table will be of the same type. If an object is the “one” in a relationship or has no relationship to other objects, then it is classified as an entity. If an object is the “many” in a relationship, then it is a transaction.

6 Feature Engineering

The feature engineering step assumes that it has an entity or transaction template. This template can be used to construct new features based on a predefined set of recipes. Recipes are made from expert knowledge previous successful features engineering techniques.

DSM must identify the type of feature a column contains. The assignments The DSM will initial support are boolean, categorical, float, datetime, or string. This is crucial because different type of variables require different methods. Additionally, there would be a performance overhead to consider prediction problems that aren’t necessary.

6.1 Feature recipes

The Data Science Machine defines two types of features that it can automatically engineer: instance level and collection level. Instance level means that only one instance of an entity or transaction is used. Collection level means that a new feature comes from considering several instances.

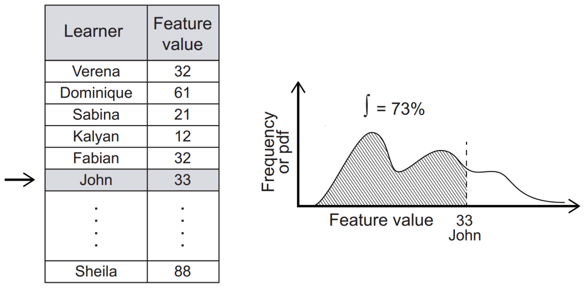
Instance level features

1. Float Columns.
    (a) Binary operations: performed on pairs of columns
2. Datetime columns:
    (a) Day of week: convert a time stamp to the day of the week (could also be month or year)
3. String columns:

[1]SQLAlchemy: www.sqlalchemy.org
[2]sqlacodegen: https://bitbucket.org/agronholm/sqlacodegen/

(a) Length : the number of characters in a string

Collection level features



Learner	Feature value
Verena	32
Dominique	61
Sabina	21
Kalyan	12
Fabian	32
John	33
⋮	⋮
Sheila	88

Figure 2: Calculating a collection level feature.

1. Float Columns.

   (a) Percentile: calculate the relative position of value to other values in a column. An example is shown for John in Figure 2.

2. Datetime columns:

   (a) Assemble a time series and use the other columns as observations. For the observations, The DSM can add approximate derivatives for float column. More advance analysis could be done to infer the hidden states of a hidden Markov model and then add those as new features.

3. String columns:

   (a) Train a topic model that models the in text in a particular add features for topic weights.

4. Aggregate statistics of transactions for a given object

   (a) Ideas currently include: count, sum, mean, median, mode (especially for categorical variables), maximum, minimum, standard deviation.

It is important to note that all new columns are tagged with the column names that went into them. It is important to keep track of them for the data insight step presented in Section 7.

6.2 How to aggregate

Changing how The DSM aggregates transactions can expose additional information in the data. For example, when it performs the aggregation of transactions for an object, it is possible to aggregate based on a third entity that is related to the transactions.

To exemplify this, consider a database for an ecommerce site that contains three tables: Users, Purchases, Purchase_Categories . The Users table can have has relations to rows in the Purchases table where each row references one row in the Purchase_Categories table. When The DSM performs the aggregation for the an instance of the 'user' entity, it can do a different aggregation of the 'purchase' transactions for each of the categories (e.g. electronics, apparel, etc).

7 Machine Learning Models and Automated Data Insights

The feature engineering component presents an interface that is natural for existing machine learning models to use. While the actual storage is a database, the interface presents essential provides an interface to a the co-variate matrix that algorithms expect. This means The DSM can take advantage of existing system within the ALFA group to do this step.

One idea for generating insight will be to look for strong correlations between features of different objects. As mentioned early, new feature columns have been tagged with the columns that went into it. This is important because these columns will be strongly correlated and The DSM should filter them out.

8 Examples

We now look at how The DSM will evaluate entities and transactions in a real schema. Figure 3 show the schema for MOOCdb. MOOCdb is a standardize schema for performing data science on Massive Open Online Courses [3].

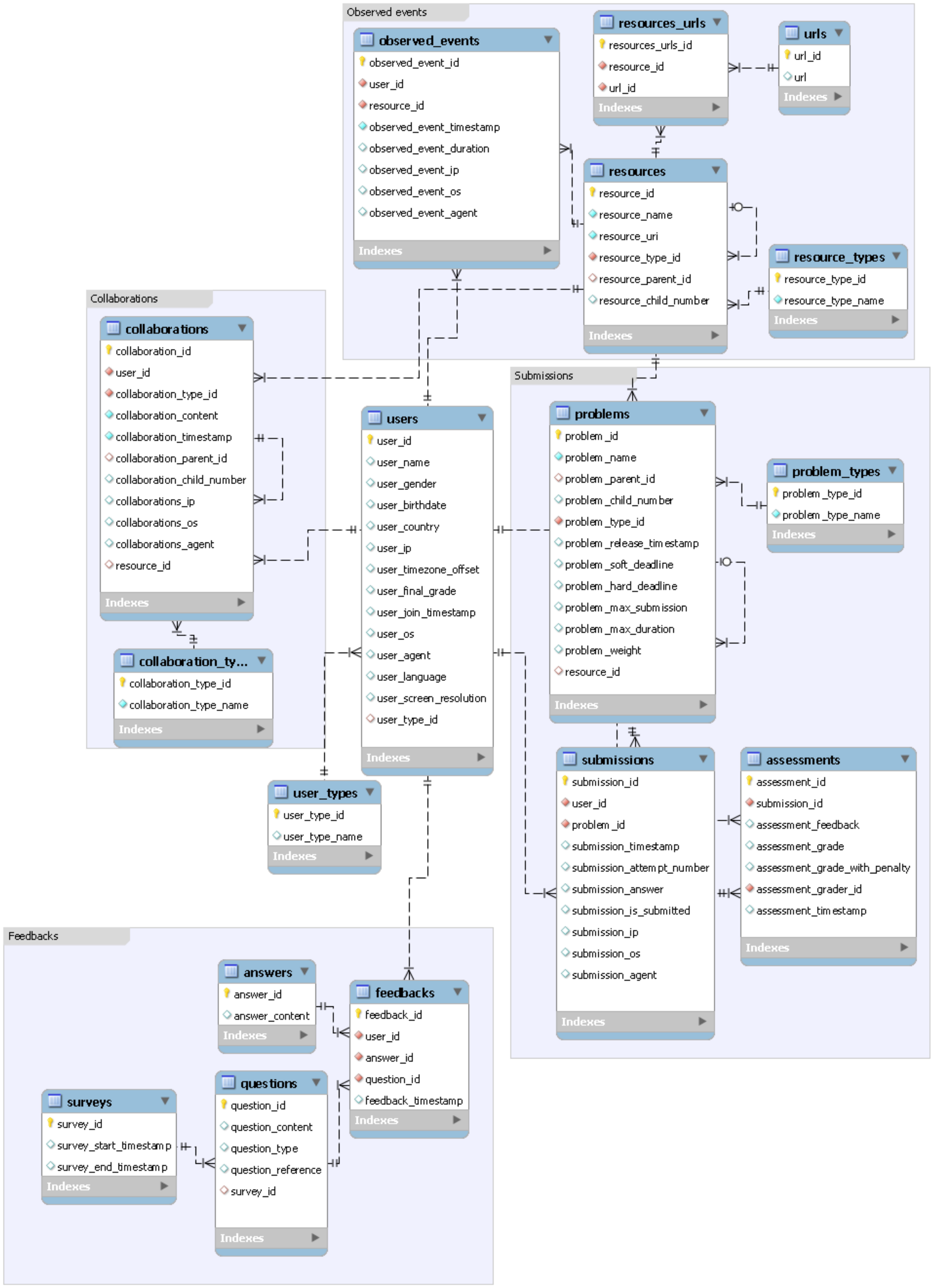


Figure 3: The Schema for MOOCdb

Object Classification

Using the rules described Section 5.1 I created list how each table is classified. The schema shows the different one-to-one, one-to-many, many-to-many relationships that I used to make the following table.

Entities	**Transactions**
urls	res_urls
resource_types	observed_events
collaborations_type	resources
user_types	collborations
problem_types	users
surveys	problems
answers	submissions
	assessments
	questions
	feedbacks

It might seem unintuitive that the user table is classified as a transaction in MOOCdb. However, this is because there is also a user_type table. Thus, in terms of The DSM the rows of the user table are transactions of the user_type table. At this point in time, this design seems ideal. However, it is interesting to not that if the information in the user_types table with the user table was concatenated, The DSM would consider the user table to be an entity. However, if it did this it would lose the ability to automatically generate insight on the instances in user_types. Thus, I conclude the current behavior is ideal.

Feature Engineering

Once the tables are classified as entities and transactions, we can perform automated feature engineering as described in Section 6.

Several noteworthy features would likely be created. Some that I would guess would work are counts of the number of submissions a given user had or the sum of the time spent using a given resource.

9 Evaluation

The performance of The DSM will be evaluated on three data sets that The ALFA Group already has access to. Those data sets are MOOCs, credit card transactions, and surveys from a technology consulting company.

References

[1] Jay Baxter. *BayesDB: querying the probable implications of tabular data.* PhD thesis, Massachusetts Institute of Technology, 2014.

[2] James Robert Lloyd, David Duvenaud, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Automatic construction and Natural-Language description of nonparametric regression models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2014.

[3] Kalyan Veeramachaneni, Franck Dernoncourt, Colin Taylor, Zachary Pardos, and Una-May O'Reilly. Moocdb: Developing data standards for mooc data science. In *AIED 2013 Workshops Proceedings Volume*, page 17. Citeseer, 2013.