Data Science with PostgreSQL

Balázs Bárány

Data Scientist

pgconf.de 2015
Contents

Introduction – What is Data Science?
   Process model

Tools and methods of Data Scientists

Data Science with PostgreSQL
   Business & data understanding
   Preprocessing
   Modeling
   Evaluation
   Deployment

Summary
Sexiest job of the 21st century

- According to Google, LinkedIn, ...
Sexiest job of the 21st century

- According to Google, LinkedIn, ...
- Who is a Data Scientist?
Data Science with PostgreSQL

Introduction – What is Data Science?

Data Science Venn Diagram

(c) Drew Conway, 2010. CC-BY-NC
Tasks of data scientists

- Get data from various sources
  - Big data?
Tasks of data scientists

- Get data from various sources
  - Big data?
- Mash up & format for analysis
Tasks of data scientists

- Get data from various sources
  - Big data?
- Mash up & format for analysis
- Analyze & visualize
Tasks of data scientists

- Get data from various sources
  - Big data?
- Mash up & format for analysis
- Analyze & visualize
- Predict & prescribe
Tasks of data scientists

- Get data from various sources
  - Big data?
- Mash up & format for analysis
- Analyze & visualize
- Predict & prescribe
- Operationalize
The Data Mining process

Cross Industry Standard Process for Data Mining (Kenneth Jensen/Wikimedia Commons)
Tools and methods
Scripting and programming

- R
- Python with extensions
- Octave/Matlab, other mathematic languages
- Hadoop and Big Data programming libraries (mostly Java)
- Cloud services
Integrated GUI tools

- (partly) Open Source: RapidMiner, KNIME, Orange
- Data Warehouse tools extended for analytics: Pentaho, Talend
- Many commercial tools, e.g. SAS, IBM SPSS
- Hadoop-related newcomers: e.g. Datameer
Data Infrastructure

- Databases and data stores
  - Relational, NoSQL
  - Hadoop clusters
  - In-memory
- Data streams
- Free-form data: text, images, video, audio, ...
- Web APIs
- Open Data
Data acquisition and preprocessing

- Data ingestion in raw format
Data acquisition and preprocessing

- Data ingestion in raw format
- Joining, aggregating, filtering, calculating, ...
Data acquisition and preprocessing

- Data ingestion in raw format
- Joining, aggregating, filtering, calculating, ...
- Data cleansing
  - Missing values
  - Abnormal values
Data acquisition and preprocessing

- Data ingestion in raw format
- Joining, aggregating, filtering, calculating, ...
- Data cleansing
  - Missing values
  - Abnormal values
- Result: data set suitable for analytics
Predictive Modeling

- Supervised and unsupervised methods
  - Target variable known or not
Predictive Modeling

- Supervised and unsupervised methods
  - Target variable known or not
- Classification (supervised): Prediction of a class or category
- Regression (supervised): Prediction of numeric value
Predictive Modeling

- Supervised and unsupervised methods
  - Target variable known or not
- Classification (supervised): Prediction of a class or category
- Regression (supervised): Prediction of numeric value
- Clustering (unsupervised): Automatic “grouping” of data
- Association analysis, outlier detection, time series prediction, ...

Data Science with PostgreSQL
- Tools and methods of Data Scientists
Deployment and operationalization

- Model application to new data $\Rightarrow$ prediction + confidence
- What to do with predictions?
Deployment and operationalization

- Model application to new data $\Rightarrow$ prediction + confidence
- What to do with predictions?
  - Store in ERP or CRM
  - Tell someone (email, popup)
  - Add a label (e.g. mark email as spam)
Deployment and operationalization

- Model application to new data => prediction + confidence
- What to do with predictions?
  - Store in ERP or CRM
  - Tell someone (email, popup)
  - Add a label (e.g. mark email as spam)
- Interrupt financial transaction => prescription
- Order supplies => prescription
- ...
Doing Data Science with PostgreSQL
Caveats

- This stuff is not easy
Caveats

- This stuff is not easy
- Must be root and postgres
  - Maintain your PostgreSQL yourself
  - Able to compile stuff
Caveats

▶ This stuff is not easy

▶ Must be root and postgres
  ▶ Maintain your PostgreSQL yourself
  ▶ Able to compile stuff

▶ You should ask ;-)
  ▶ your boss
  ▶ co-workers
  ▶ customer
Data Science with PostgreSQL

Business & data understanding
Business understanding

- What is the purpose of the business?
- What are existing processes?
- Drivers of business success
Business understanding

- What is the purpose of the business?
- What are existing processes?
- Drivers of business success
- Project goals and challenges
- Availability of data and resources
- Success criteria
Business understanding

- What is the purpose of the business?
- What are existing processes?
- Drivers of business success
- Project goals and challenges
- Availability of data and resources
- Success criteria
- Not a technical activity, PostgreSQL can’t help much
Data understanding

- Existing data
  - Entities and covered concepts
  - Complete? Correct? In suitable form?
  - Usable? (regulations, access constraints, ...)

- Connecting separate data sources
  - Simple or complex IDs

- Data size
  - Too small
  - Too big

- Suitability for predictive modeling
  - Target variable?

- Attribute types
Data understanding

- Existing data
  - Entities and covered concepts
  - Complete? Correct? In suitable form?
  - Usable? (regulations, access constraints, ...)

- Connecting separate data sources
  - Simple or complex IDs
Data understanding

- Existing data
  - Entities and covered concepts
  - Complete? Correct? In suitable form?
  - Usable? (regulations, access constraints, ...)

- Connecting separate data sources
  - Simple or complex IDs

- Data size
  - Too small
  - Too big
Data understanding

- Existing data
  - Entities and covered concepts
  - Complete? Correct? In suitable form?
  - Usable? (regulations, access constraints, ...)

- Connecting separate data sources
  - Simple or complex IDs

- Data size
  - Too small
  - Too big

- Suitability for predictive modeling
  - Target variable?
  - Attribute types
Data understanding with PostgreSQL

▶ Get data into PostgreSQL
  ▶ Classical import process
  ▶ Foreign Data Wrappers
Data understanding with PostgreSQL

- Get data into PostgreSQL
  - Classical import process
  - Foreign Data Wrappers

- Analyze data distribution
  - Group by and aggregate
    - Count, Count Distinct, Min, Max
  - Count NULLs
  - Search for missing links (incomplete foreign keys)
Data understanding with PostgreSQL

- Get data into PostgreSQL
  - Classical import process
  - Foreign Data Wrappers
- Analyze data distribution
  - Group by and aggregate
    - Count, Count Distinct, Min, Max
  - Count NULLs
  - Search for missing links (incomplete foreign keys)
- Analyze “surprises”
  - Impossible values
  - Missing values in “required” fields
Data understanding with PostgreSQL – summary

- Good SQL knowledge required
- Tedious manual process
  - repetitive
  - not suitable for large number of attributes
- No built-in visualization
Data understanding with PostgreSQL – summary

- Good SQL knowledge required
- Tedious manual process
  - repetitive
  - not suitable for large number of attributes
- No built-in visualization
- Or maybe...
### SQL bar chart output

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Count</th>
<th>Bar Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>foreign_table_options</td>
<td>5</td>
<td>###</td>
</tr>
<tr>
<td>foreign_tables</td>
<td>5</td>
<td>###</td>
</tr>
<tr>
<td>information_schema_catalog_name</td>
<td>1</td>
<td>#</td>
</tr>
<tr>
<td>key_column_usage</td>
<td>9</td>
<td>#####</td>
</tr>
<tr>
<td>parameters</td>
<td>32</td>
<td>#</td>
</tr>
<tr>
<td>_pg_foreign_data_wrappers</td>
<td>7</td>
<td>#</td>
</tr>
<tr>
<td>_pg_foreign_servers</td>
<td>9</td>
<td>####</td>
</tr>
<tr>
<td>_pg_foreign_table_columns</td>
<td>4</td>
<td>#</td>
</tr>
<tr>
<td>_pg_foreign_tables</td>
<td>7</td>
<td>####</td>
</tr>
<tr>
<td>_pg_user_mappings</td>
<td>7</td>
<td>####</td>
</tr>
<tr>
<td>referential_constraints</td>
<td>9</td>
<td>#####</td>
</tr>
<tr>
<td>role_column_grants</td>
<td>8</td>
<td>#</td>
</tr>
<tr>
<td>role_routine_grants</td>
<td>10</td>
<td>######</td>
</tr>
<tr>
<td>role_table_grants</td>
<td>8</td>
<td>#</td>
</tr>
<tr>
<td>role_udt_grants</td>
<td>7</td>
<td>####</td>
</tr>
<tr>
<td>role_usage_grants</td>
<td>8</td>
<td>####</td>
</tr>
<tr>
<td>routine_privileges</td>
<td>10</td>
<td>######</td>
</tr>
<tr>
<td>routines</td>
<td>82</td>
<td>#</td>
</tr>
<tr>
<td>schemata</td>
<td>7</td>
<td>#</td>
</tr>
<tr>
<td>sequences</td>
<td>12</td>
<td>######</td>
</tr>
<tr>
<td>sql_features</td>
<td>7</td>
<td>#</td>
</tr>
<tr>
<td>sql_implementation_info</td>
<td>5</td>
<td>#</td>
</tr>
<tr>
<td>sql_languages</td>
<td>7</td>
<td>####</td>
</tr>
<tr>
<td>sql_packages</td>
<td>5</td>
<td>#</td>
</tr>
<tr>
<td>sql_parts</td>
<td>5</td>
<td>#</td>
</tr>
<tr>
<td>sql_sizing</td>
<td>4</td>
<td>#</td>
</tr>
<tr>
<td>sql_sizing_profiles</td>
<td>5</td>
<td>#</td>
</tr>
</tbody>
</table>

---

The bar chart shows the frequency of different table names in a SQL query output, with the bar height indicating the count of occurrences.
Bar chart from GUI tool
Boxplot output
Data understanding wrap up

- DBMS not built for this
- It can support more specialized tools
Data understanding wrap up

- DBMS not built for this
- It can support more specialized tools
- Introduction: R
  - “A free software environment for statistical computing and graphics”
  - Available in PostgreSQL
PL/R: A statistical language for PostgreSQL

- R as a standalone language
  - Mathematical and statistical methods
  - Powerful visualization functions
  - Classical, modern and bleeding edge modeling
  - Arrays and data frames are central data types
  - Operates only in memory
PL/R: A statistical language for PostgreSQL

- R as a standalone language
  - Mathematical and statistical methods
  - Powerful visualization functions
  - Classical, modern and bleeding edge modeling
  - Arrays and data frames are central data types
  - Operates only in memory

- PL/R: R as a loadable procedural language for PostgreSQL
  - First released in 2003 by Joe Conway
  - License: GPL
R usage outside of PostgreSQL

- Development environments
  - RStudio (AGPL or commercial, local & web)
  - RKward, Cantor (KDE projects)
  - StatET (Eclipse)

- Frontends
  - R Commander
  - Deducer
  - Rattle

- Web framework: Shiny (AGPL or commercial)
Working with R in PostgreSQL

- **Install functions in the database**

**Example**

```sql
select install_rcmd('myfunction <-function(x)
{print(x)}
');
```

- **Install without function body**

**Example**

```sql
CREATE FUNCTION rnorm
    (n integer, mean double precision, sd double precision)
RETURNS double precision[]
AS ''
LANGUAGE 'plr';
```
Using R in PostgreSQL for data understanding

- Advanced visualization
- Data distributions
- Advanced statistics
Using R in PostgreSQL for data understanding

- Advanced visualization
- Data distributions
- Advanced statistics

- Execution in the database
  - Clumsy, but direct data access
Using R in PostgreSQL for data understanding

- Advanced visualization
- Data distributions
- Advanced statistics
- Execution in the database
  - Clumsy, but direct data access
- Execution outside
  - Simple and interactive, but data transfer
Preprocessing
Preprocessing

- What databases are built for
Data Science with PostgreSQL

Preprocessing

- What databases are built for
  - Rows: very dynamic
    - Easy to create new rows by joining
    - Easy to filter
  - Columns: not so much
    - Easy to create new columns
    - Only explicit access
Preprocessing

- What databases are built for
  - Rows: very dynamic
    - Easy to create new rows by joining
    - Easy to filter
  - Columns: not so much
    - Easy to create new columns
    - Only explicit access

- Wider interpretation of preprocessing
  - Enrichment with external data
  - New attributes from existing ones
  - Recoding, recalculation
  - Missing value handling
Preprocessing: organizing workflow

- Common Table Expressions
  - organize processing steps
  - partial and intermediate results

Example

```
WITH source AS (  
  SELECT *, ROW_NUMBER() OVER () AS rownum  
  FROM source_table  
),
no_missings AS (  
  SELECT *  
  FROM source  
  WHERE field1 IS NOT NULL  
     AND field2 IS NOT NULL  
)  
```

etc.
Preprocessing: attribute creation

- Aggregation

  ▶ Partial aggregation by window functions
  ▶ In-group measures, e.g. ratio \( \frac{\text{att}}{\text{SUM}(\text{att}) \text{ OVER (PARTITION BY ...)} } \)
  ▶ In-group numbering \( \text{ROW_NUMBER()} \text{ OVER (PARTITION BY ... ORDER BY ...)} \)
  ▶ Comparing to previous/next value \( \text{att} - \text{LAG}(\text{att}, 1) \text{ OVER (ORDER BY ...)} \)

Much easier in SQL than programming languages and data mining tools
Preprocessing: attribute creation

- Aggregation

- Partial aggregation by window functions
  - In-group measures, e.g. ratio
  - \( \frac{\text{att}}{\text{SUM(att)}} \) \text{OVER (PARTITION BY ...)}
Preprocessing: attribute creation

- Aggregation

- Partial aggregation by window functions
  - In-group measures, e.g. ratio
    - \( \text{att} / \text{SUM(\text{att}) OVER (PARTITION BY ...)} \)
  - In-group numbering
    - \( \text{ROW_NUMBER() OVER (PARTITION BY ... ORDER BY ...)} \)

Much easier in SQL than programming languages and data mining tools.
Preprocessing: attribute creation

- Aggregation

- Partial aggregation by window functions
  - In-group measures, e.g. ratio
    - \( att / \text{SUM}(att) \text{ OVER (PARTITION BY ...)} \)
  - In-group numbering
    - \( \text{ROW_NUMBER(} \text{ OVER (PARTITION BY ... ORDER BY ...} \) \)

- Comparing to previous/next value
  - \( att - \text{LAG}(att, 1) \text{ OVER (ORDER BY ...} \)
Preprocessing: attribute creation

- Aggregation
- Partial aggregation by window functions
  - In-group measures, e.g. ratio
    - \( \text{att} / \text{SUM(\text{att}) OVER (PARTITION BY \ldots)} \)
  - In-group numbering
    - \( \text{ROW_NUMBER()} OVER (PARTITION BY \ldots \text{ ORDER BY \ldots}) \)
- Comparing to previous/next value
  - \( \text{att} - \text{LAG(\text{att}, 1) OVER (ORDER BY \ldots)} \)
- Much easier in SQL than programming languages and data mining tools
Preprocessing: enrichment

- PostGIS for geodata
Preprocessing: enrichment

- PostGIS for geodata
- Foreign data wrappers (see PostgreSQL Wiki)
Preprocessing: enrichment

- PostGIS for geodata
- Foreign data wrappers (see PostgreSQL Wiki)
  - Other databases (other PostgreSQL server, MySQL, Oracle, MSSQL, JDBC, SQL Alchemy ...)
  - NoSQL databases (MongoDB, Cassandra, CouchDB, Redis, ...)

---

Data Science with PostgreSQL

- Preprocessing
Preprocessing: enrichment

- PostGIS for geodata
- Foreign data wrappers (see PostgreSQL Wiki)
  - Other databases (other PostgreSQL server, MySQL, Oracle, MSSQL, JDBC, SQL Alchemy ...)
  - NoSQL databases (MongoDB, Cassandra, CouchDB, Redis, ...)
- Big Data (Hadoop Hive, Impala)
Preprocessing: enrichment

- PostGIS for geodata
- Foreign data wrappers (see PostgreSQL Wiki)
  - Other databases (other PostgreSQL server, MySQL, Oracle, MSSQL, JDBC, SQL Alchemy ...)
  - NoSQL databases (MongoDB, Cassandra, CouchDB, Redis, ...)
- Big Data (Hadoop Hive, Impala)
- Network sources - Multicorn (RSS, IMAP, Twitter, S3, ...)
- Files (CSV, ZIP, JSON, ...)
Preprocessing: enrichment

- PostGIS for geodata
- Foreign data wrappers (see PostgreSQL Wiki)
  - Other databases (other PostgreSQL server, MySQL, Oracle, MSSQL, JDBC, SQL Alchemy ...)
  - NoSQL databases (MongoDB, Cassandra, CouchDB, Redis, ...)
- Big Data (Hadoop Hive, Impala)
- Network sources - Multicorn (RSS, IMAP, Twitter, S3, ...)
- Files (CSV, ZIP, JSON, ...)
- Write your own in C or Python or Ruby
Data Science with PostgreSQL

Modeling
Model development

- Machine learning algorithms not well suited for SQL
Model development

- Machine learning algorithms not well suited for SQL

- Some attempts to build them
  - Naive Bayes, Linear Regression
  - Difficult for more advanced algorithms
Model development

- Machine learning algorithms not well suited for SQL

- Some attempts to build them
  - Naive Bayes, Linear Regression
  - Difficult for more advanced algorithms

- Better done in specialized language or tool
  - PL/R
  - PL/Python
PL/Python

- Python procedural language available in PostgreSQL
- scikit-learn: Machine learning toolbox for Python
  - Classification, regression, clustering
  - Model selection, validation
  - Preprocessing
- matplotlib: Generic and statistical plotting library
PL/Python

- Python procedural language available in PostgreSQL
- scikit-learn: Machine learning toolbox for Python
  - Classification, regression, clustering
  - Model selection, validation
  - Preprocessing
- matplotlib: Generic and statistical plotting library
- PL/Python is an alternative to PL/R
Evaluation
Evaluation of modeling results

- Models return predictions
- Prediction can be compared to known result (target variable)
- Measures of model performance: Accuracy, precision, recall, ...
Evaluation of modeling results

- Models return predictions
- Prediction can be compared to known result (target variable)
- Measures of model performance: Accuracy, precision, recall, ...
- Results on the training set meaningless
Evaluation of modeling results

- Models return predictions
- Prediction can be compared to known result (target variable)
- Measures of model performance: Accuracy, precision, recall, ...
- Results on the training set meaningless
- Split validation
- Cross validation
Evaluation results

- “Good result” depends on the application
- If not good enough,
Evaluation results

- “Good result” depends on the application
- If not good enough,
  - get more data
Evaluation results

- "Good result" depends on the application
- If not good enough,
  - get more data
  - do more preprocessing
Evaluation results

- “Good result” depends on the application
- If not good enough,
  - get more data
  - do more preprocessing
  - select better classifier
Evaluation results

- “Good result” depends on the application
- If not good enough,
  - get more data
  - do more preprocessing
  - select better classifier
  - optimize classifier parameters
Evaluation results

- “Good result” depends on the application
- If not good enough,
  - get more data
  - do more preprocessing
  - select better classifier
  - optimize classifier parameters
- Cycle: preprocessing - modeling - evaluation
Evaluation results

- “Good result” depends on the application
- If not good enough,
  - get more data
  - do more preprocessing
  - select better classifier
  - optimize classifier parameters
- Cycle: preprocessing - modeling - evaluation
- Better done in data mining environment
Deployment
Advantages of deployment in the database:

- Less overhead
Deployment

- Advantages of deployment in the database:
  - Less overhead
  - Instant application using triggers
Deployment

- Advantages of deployment in the database:
  - Less overhead
  - Instant application using triggers
  - Well-known execution environment
Deployment

- Advantages of deployment in the database:
  - Less overhead
  - Instant application using triggers
  - Well-known execution environment
  - Functionality available over standard interface
Deployment

- Advantages of deployment in the database:
  - Less overhead
  - Instant application using triggers
  - Well-known execution environment
  - Functionality available over standard interface
- Some models easily expressed in SQL
Deployment of PL/R or PL/Python models

- Model developed in database or outside
Deployment of PL/R or PL/Python models

- Model developed in database or outside
- Put into global context
  - PL/R: `load("saved object", .GlobalEnv)`
  - PL/Python: Global dictionary “GD”
- Application function in matching language
  - Uses existing model
  - Returns target variable

Trigger function or UPDATE uses application function
Deployment of PL/R or PL/Python models

- Model developed in database or outside
- Put into global context
  - PL/R: `load("saved object", .GlobalEnv)`
  - PL/Python: Global dictionary “GD”
- Application function in matching language
  - Uses existing model
  - Returns target variable
- Trigger function or UPDATE uses application function
Summary

- PostgreSQL’s support for data science tasks
  - Best: preprocessing, deployment
Summary

- PostgreSQL’s support for data science tasks
  - Best: preprocessing, deployment
- Modern SQL for preprocessing
Summary

- PostgreSQL’s support for data science tasks
  - Best: preprocessing, deployment
- Modern SQL for preprocessing
- Foreign Data Wrappers for data integration
Summary

- PostgreSQL’s support for data science tasks
  - Best: preprocessing, deployment
- Modern SQL for preprocessing
- Foreign Data Wrappers for data integration
- Procedural languages for data mining
Questions?

- Balázs Bárány, <balazs@tud.at>
- https://datascientist.at/