Large-scale Data Quality Verification How to Unit-test Your Data with Deequ

Presenter: Philipp Schmidt

Amazon Research

research

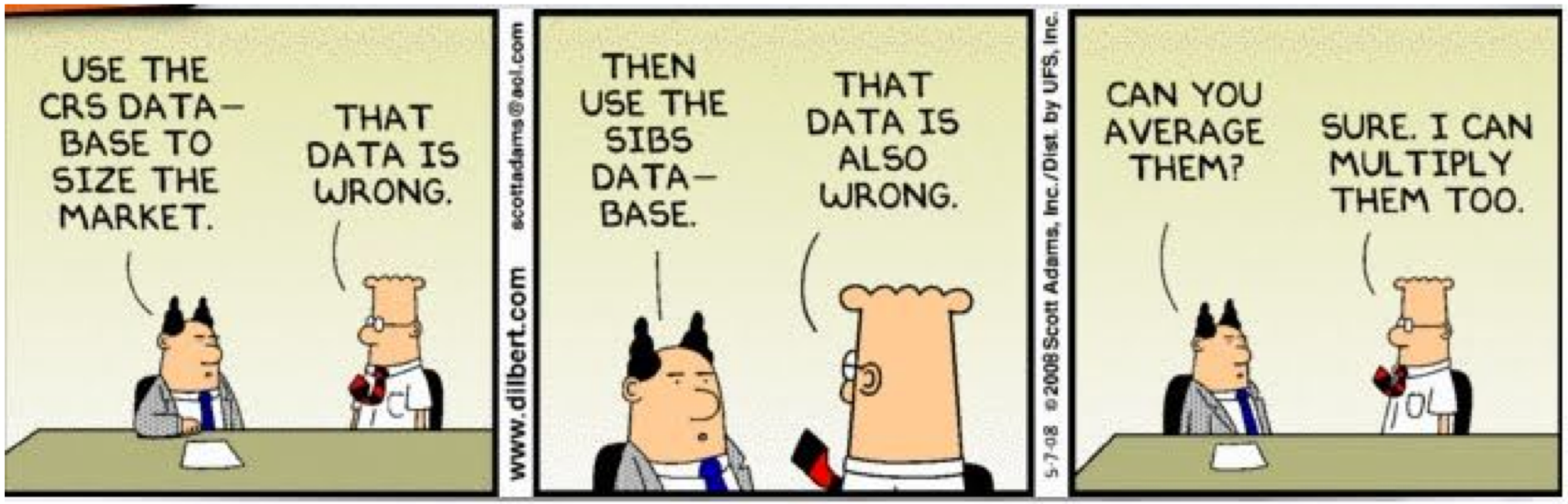
What‘s in it for me?

- Learn how to unit-test your data
  - On data of any scale
  - Every day, also as part of ETL pipelines
  - To fail fast and improve
- Low entry barrier for first usage of deequ
  - You can start verifying data quality today
- Deequ is available on GitHub

Why should we care about Data Quality?

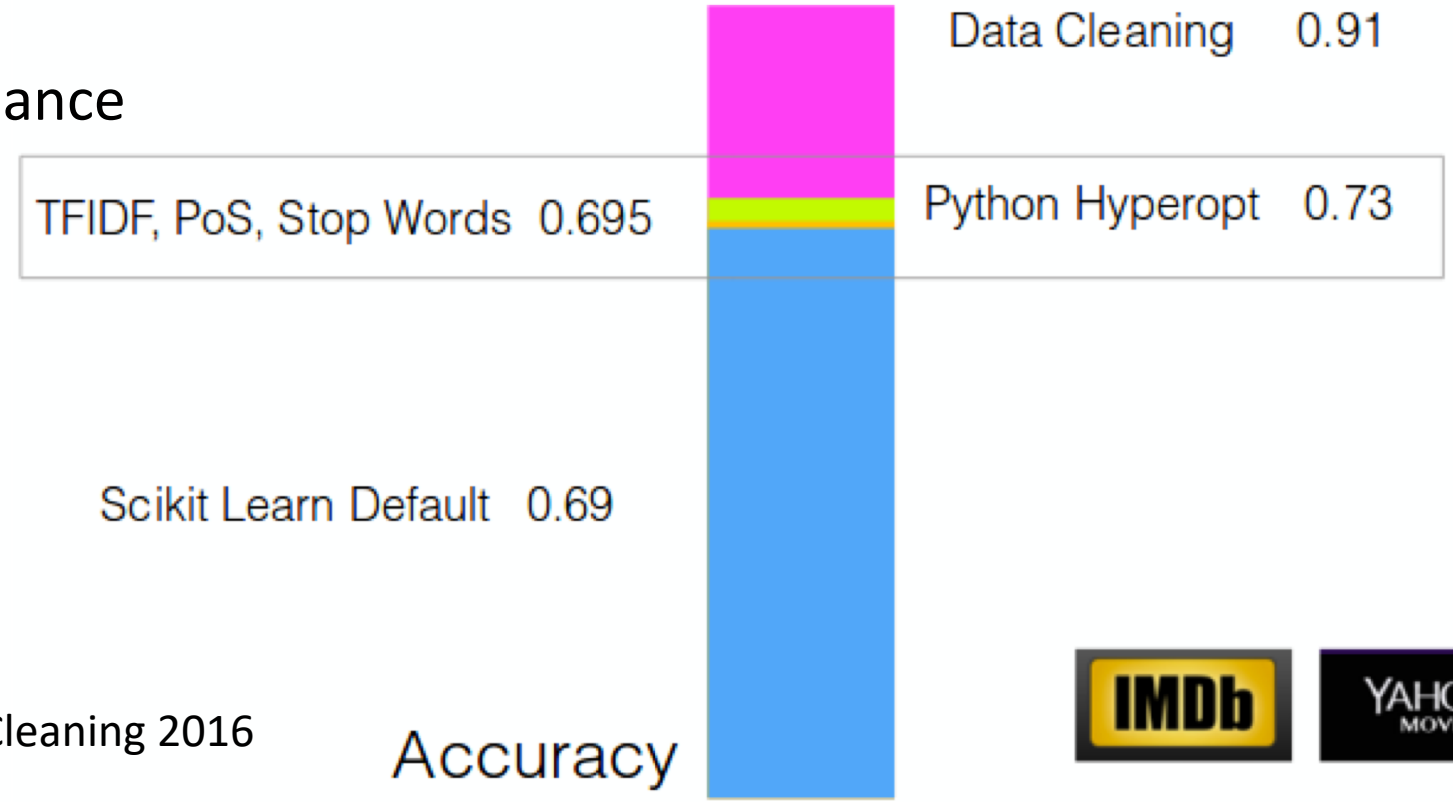
Why should we care about Data Quality?

- **impact on business decisions**
  - Missing or incorrect data results in wrong decision making



Why should we care about Data Quality?

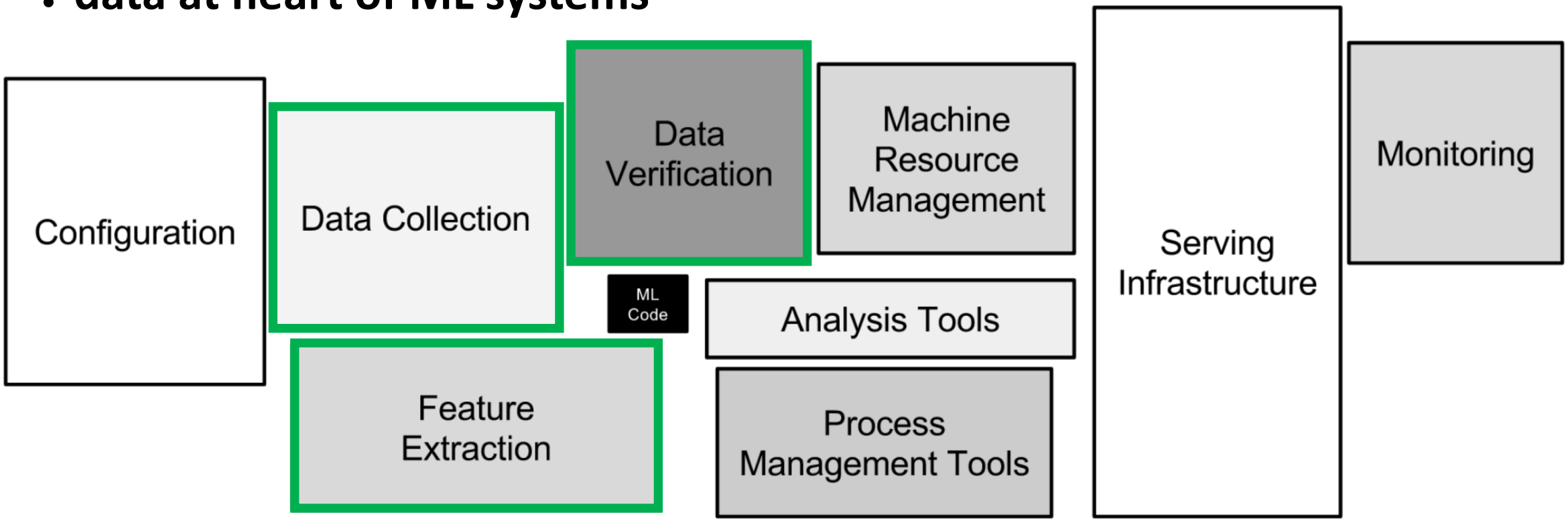
- **impact on ML models**
  - clean data can greatly improve model performance



Krishnan et al., SIGMOD Tutorial on Data Cleaning 2016

Why should we care about Data Quality?

- **data at heart of ML systems**



Sculley et al., Hidden technical debt in ML Systems, NIPS 2015

research

Why should we care about Data Quality?

- **impact on operational stability**
  - missing and inconsistent data can cause havoc in production systems

Crashes (e.g., due to NullPointerExceptions for missing attributes)

HTTP Status 500 -

type Exception report

message

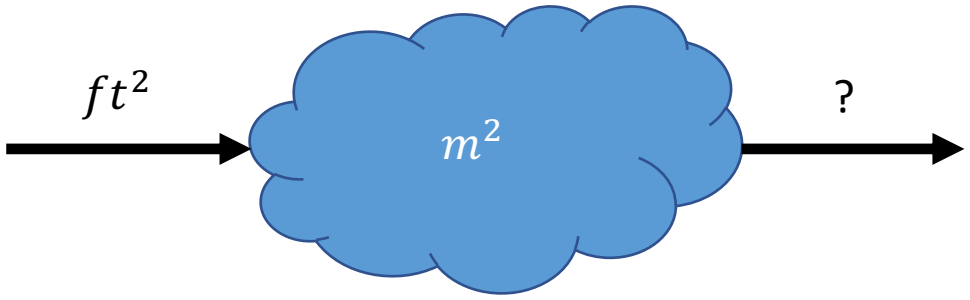
description The server encountered an internal error () that prevented it from fulfilling this request.

exception

```
java.lang.NullPointerException
        javax.servlet.GenericServlet.getServletContext(GenericServlet.java:125)
        com.swiftmobil.LocationTrackerServlet.doGet(LocationTrackerServlet.java:112)
        javax.servlet.http.HttpServlet.service(HttpServlet.java:621)
        javax.servlet.http.HttpServlet.service(HttpServlet.java:722)
```

note The full stack trace of the root cause is available in the Apache Tomcat/7.0.29 logs.

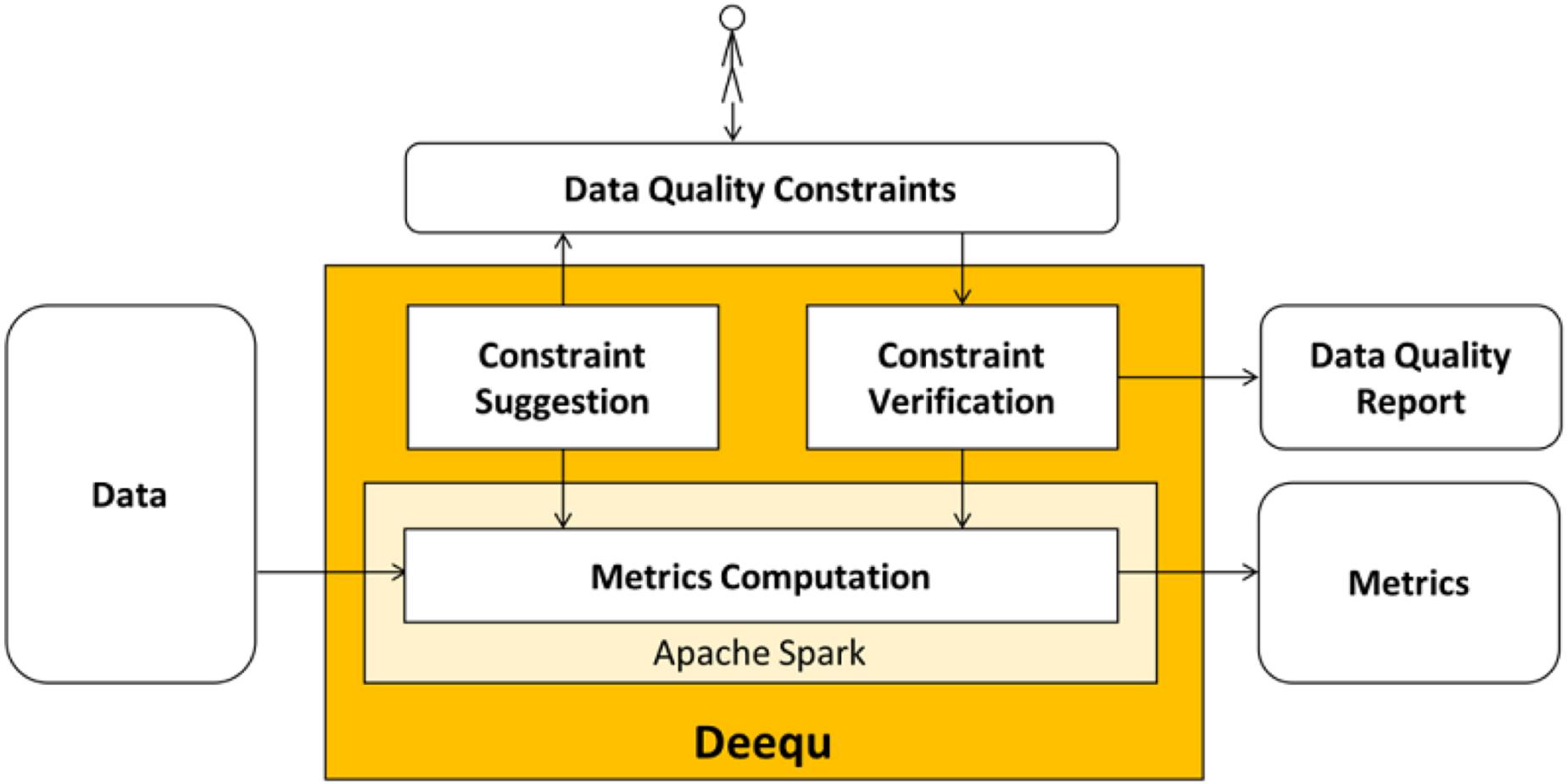
Wrong predictions (e.g., change of scale in attribute)



Quality Assurance

- **Software**
  - Established practice to have tests for software components
    - Unit-tests
    - Integration-tests
    - …
- **Data**
  - Often tedious, repetetive and done in ad-hoc fashion
  - **unit-tests for data: Deequ**

Overview of Deequ



research

Constraint verification in deequ

- A unit-test for data
  - Scales to big data sets
    - Metrics computed as SQL aggregation queries in Apache Spark
  - Computes several data quality metrics (e.g., how many NULLs are there?)
    - Completeness/Uniqueness/Compliance/…
  - Executes user-defined validation code (e.g., are there less than 2% NULLS?)

A Unit Test for Data

```
val numTitles = callRestService(...)

VerificationSuite()
  .onData(data)
  // data integrity
  .addCheck(Check(Level.Error)
    .isComplete("customerId", "title")
    .isUnique("customerId")
    .hasCountDistinct("title", _ == numTitles)
    .hasHistogramValues("deviceType", _.ratio("phone") <= 0.84))
    .isInValidRange("priority", ("hi", "lo"))
    // also check whether the current data size is similar to the
    // previously calculated ones
    .useRepository(FileSystemMetricsRepository(”s3://…”))
    .addAnomalyCheck(OnlineNormal(stdDevs=3), Size())
    .run()
```

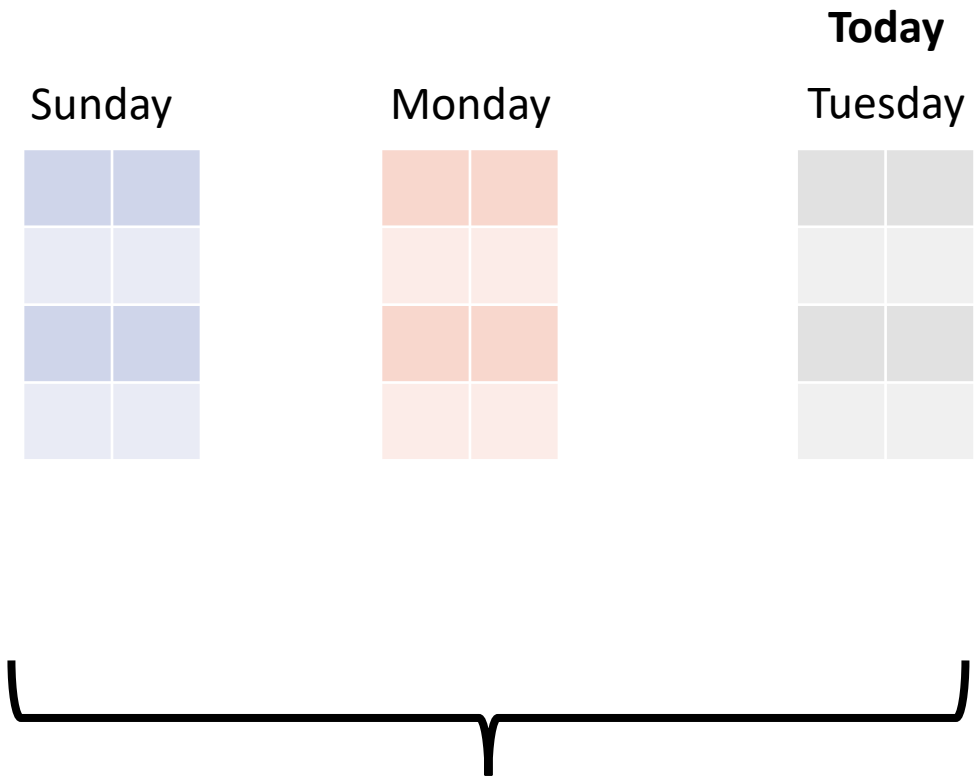
research

Data quality verification for partitioned data

- **Example**: Impression logs with daily partitions
- Verification of data quality constraints
  - **Every day**
  - **Incrementally, on all data**

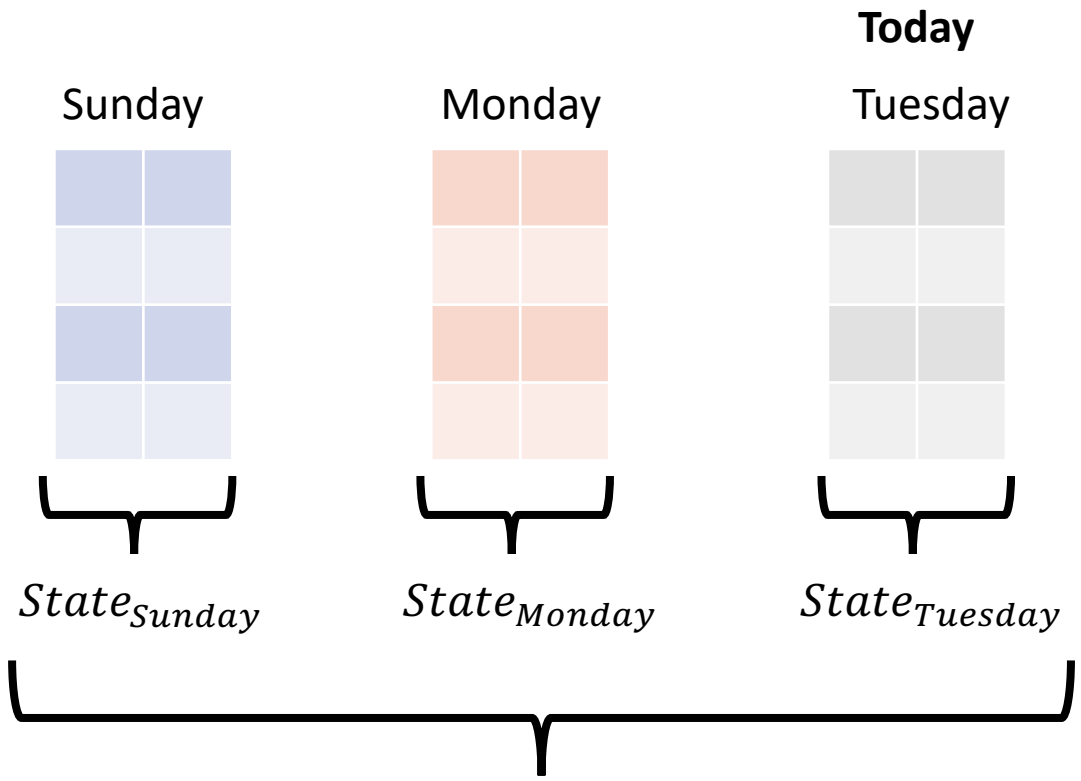
Naïve

- Global constraint evaluations scans all available data
- Computational load proportional to overall data size



Incremental

- Global constraint evaluation combines partition states
- Computational load proportional to partition data size



research

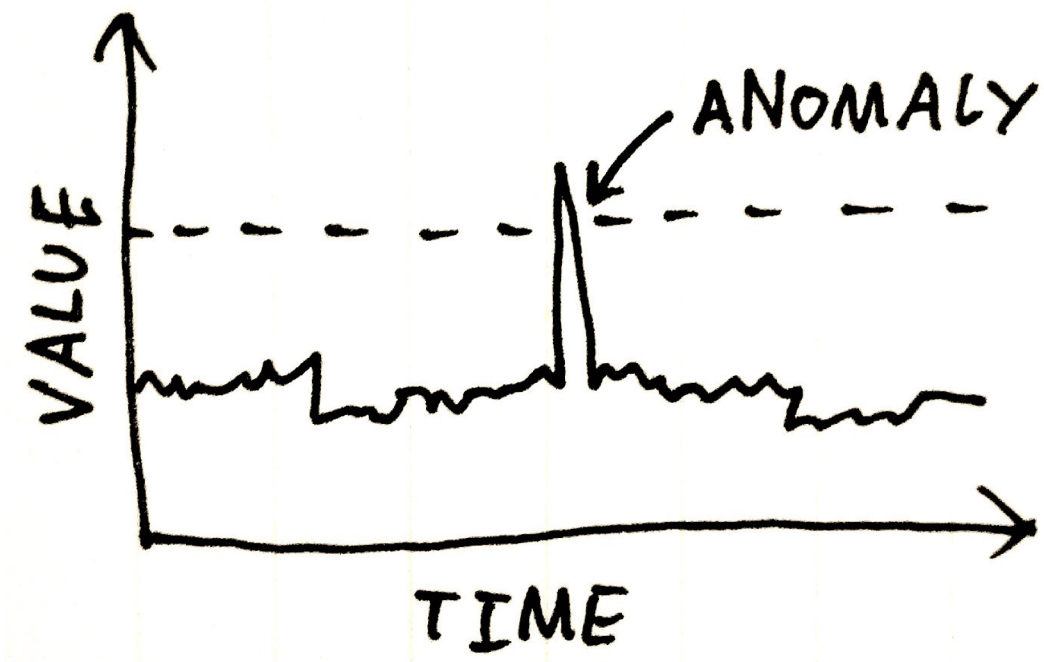
Data quality verification for partitioned data

```
val completeness = Completeness("origin")

// Compute state of the changed partition
val newStateToday =
  completeness.computeStateFrom(newPartitionToday)

// Load states of non-changed partitions
val (stateSunday, stateMonday) = loadPreviousStates("…")

// Sum of the states of the individual partitions
val newTableState = stateSunday + stateMonday + newStateToday

// Compute the completeness of 'origin' in the whole table from the new
state
val newTableCompleteness = completeness.computeMetricFrom(newTableState)
```

research

Continuous data quality verification

- Data quality metrics computed on a regular basis (e.g., every day)
- Detect sudden changes of data quality metrics without the need to configure explicit thresholds



research

A Unit Test for Data

```
val numTitles = callRestService(...)

VerificationSuite()
  .onData(data)
  // data integrity
  .addCheck(Check(Level.Error)
    // also check whether the current data size is similar to the
    // previously calculated ones
    .useRepository(FileSystemMetricsRepository("s3://…"))
    .addAnomalyCheck(OnlineNormal(stdDevs=3), Size())
    .run()
```

research

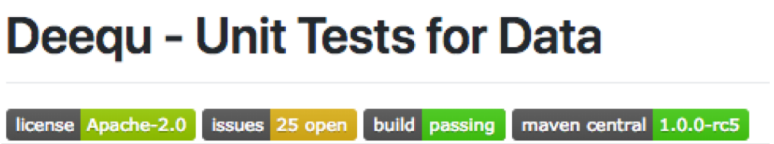
Summary

- Data central to human and algorithmic decision making
- Data quality verification usually done in ad-hoc fashion
- Deequ enables you to assert data quality at scale with a concise API
  - Efficient constraint verification for partitioned data
  - Data quality verification without explicit assertions

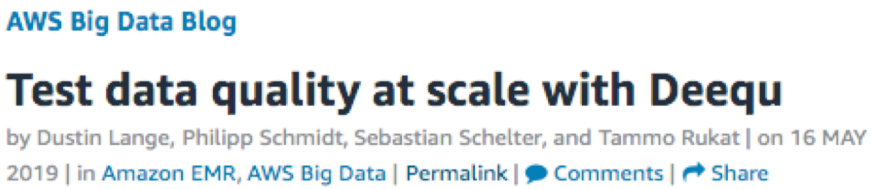
research

Further information

- **See our VLDB 2018 paper „Automating Data Quality Verification at Scale“ for more details and experiments**
  - https://dl.acm.org/citation.cfm?id=3275547



- **Deequ is open source**
  - https://github.com/awslabs/deequ



- **AWS Big Data Blog Post**
  - https://aws.amazon.com/blogs/big-data/test-data-quality-at-scale-with-deequ

research