

# DATA QUALITY AND DATA PROGRAMMING

*"Data cleaning and repairing account for about 60% of the work of data scientists."*

Eunsuk Kang

Required reading:

- Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F. and Grafberger, A., 2018. [Automating large-scale data quality verification](#). Proceedings of the VLDB Endowment, 11(12), pp.1781-1794.
- Nick Hynes, D. Sculley, Michael Terry. "[The Data Linter: Lightweight Automated Sanity Checking for ML Data Sets](#)." NIPS Workshop on ML Systems (2017)

# LEARNING GOALS

- Design and implement automated quality assurance steps that check data schema conformance and distributions
- Devise thresholds for detecting data drift and schema violations
- Describe common data cleaning steps and their purpose and risks
- Evaluate the robustness of AI components with regard to noisy or incorrect data
- Understanding the better models vs more data tradeoffs
- Programatically collect, manage, and enhance training data

# DATA-QUALITY CHALLENGES

# CASE STUDY: INVENTORY MANAGEMENT



# INVENTORY DATABASE

Product Database:

ID	Name	Weight	Description	Size	Vendor
...	...	...	...	...	...

Stock:

ProductID	Location	Quantity
...	...	...

Sales history:

UserID	ProductId	DateTime	Quantity	Price
...	...	...	...	...

# WHAT MAKES GOOD QUALITY DATA?

- Accuracy
  - The data was recorded correctly.
- Completeness
  - All relevant data was recorded.
- Uniqueness
  - The entries are recorded once.
- Consistency
  - The data agrees with itself.
- Timeliness
  - The data is kept up to date.

# DATA IS NOISY

- Unreliable sensors or data entry
  - Wrong results and computations, crashes
  - Duplicate data, near-duplicate data
  - Out of order data
  - Data format invalid
- 
- **Examples?**

# DATA CHANGES

- System objective changes over time
- Software components are upgraded or replaced
- Prediction models change
- Quality of supplied data changes
- User behavior changes
- Assumptions about the environment no longer hold
  
- **Examples?**



# USERS MAY DELIBERATELY CHANGE DATA

- Users react to model output
- Users try to game/deceive the model
- **Examples?**

# MANY DATA SOURCES



*sources of different reliability and quality*

# ACCURACY VS PRECISION

- Accuracy: Reported values (on average) represent real value
- Precision: Repeated measurements yield the same result
- Accurate, but imprecise: Average over multiple measurements
- Inaccurate, but precise: Systematic measurement problem, misleading





# DATA QUALITY AND MACHINE LEARNING

- More data -> better models (up to a point, diminishing effects)
- Noisy data (imprecise) -> less confident models, more data needed
  - some ML techniques are more or less robust to noise (more on robustness in a later lecture)
- Inaccurate data -> misleading models, biased models
- Need the "right" data
- Invest in data quality, not just quantity

# DATA CLEANING

*Data cleaning and repairing account for about 60% of the work of data scientists.*

Quote: Gil Press. “[Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says.](#)”  
Forbes Magazine, 2016.



Source: Rahm, Erhard, and Hong Hai Do. [Data cleaning: Problems and current approaches](#). IEEE Data Eng. Bull. 23.4 (2000): 3-13.

# SINGLE-SOURCE PROBLEM EXAMPLES

Further readings: Rahm, Erhard, and Hong Hai Do. [Data cleaning: Problems and current approaches](#). IEEE Data Eng. Bull. 23.4 (2000): 3-13.





# SINGLE-SOURCE PROBLEM EXAMPLES

- Schema level:
  - Illegal attribute values: `bdate=30.13.70`
  - Violated attribute dependencies: `age=22, bdate=12.02.70`
  - Uniqueness violation: `(name="John Smith", SSN="123456"), (name="Peter Miller", SSN="123456")`
  - Referential integrity violation: `emp=(name="John Smith", deptno=127)` if department 127 not defined



# SINGLE-SOURCE PROBLEM EXAMPLES

- Schema level:
  - Illegal attribute values: `bdate=30.13.70`
  - Violated attribute dependencies: `age=22, bdate=12.02.70`
  - Uniqueness violation: `(name="John Smith", SSN="123456"), (name="Peter Miller", SSN="123456")`
  - Referential integrity violation: `emp=(name="John Smith", deptno=127)` if department 127 not defined
- Instance level:
  - Missing values: `phone=9999-999999`
  - Misspellings: `city=Pittsburg`
  - Misfielded values: `city=USA`
  - Duplicate records: `name=John Smith, name=J. Smith`
  - Wrong reference: `emp=(name="John Smith", deptno=127)` if department 127 defined but wrong



# DIRTY DATA: EXAMPLE

TABLE: CUSTOMER

ID	Name	Birthday	Age	Sex	Phone	ZIP
3456	Ford, Harrison	18.2.76	43	M	9999999999	15232
3456	Mark Hamil	33.8.81	43	M	6173128718	17121
3457	Kim Kardashian	11.10.56	63	M	4159102371	94016

TABLE: ADDRESS

ZIP	City	State
15232	Pittsburgh	PA
94016	Sam Francisco	CA
73301	Austin	Texas

*Problems with the data?*

# DISCUSSION: POTENTIAL DATA QUALITY PROBLEMS?



# DATA CLEANING OVERVIEW

- Data analysis / Error detection
  - Error types: e.g. schema constraints, referential integrity, duplication
  - Single-source vs multi-source problems
  - Detection in input data vs detection in later stages (more context)
- Error repair
  - Repair data vs repair rules, one at a time or holistic
  - Data transformation or mapping
  - Automated vs human guided



# ERROR DETECTION

- Illegal values: min, max, variance, deviations, cardinality
- Misspelling: sorting + manual inspection, dictionary lookup
- Missing values: null values, default values
- Duplication: sorting, edit distance, normalization

# ERROR DETECTION: EXAMPLE

TABLE: CUSTOMER

ID	Name	Birthday	Age	Sex	Phone	ZIP
3456	Ford, Harrison	18.2.76	43	M	9999999999	15232
3456	Mark Hamil	33.8.81	43	M	6173128718	17121
3457	Kim Kardashian	11.10.56	63	M	4159102371	94016

TABLE: ADDRESS

ZIP	City	State
15232	Pittsburgh	PA
94016	Sam Francisco	CA
73301	Austin	Texas

Q. Can we (automatically) detect errors? Which errors are problem-dependent?

# COMMON STRATEGIES FOR CLEANING

- Enforce schema constraints
  - e.g., delete rows with missing data or use defaults
- Explore sources of errors
  - e.g., debugging missing values, outliers
- Remove outliers
  - e.g., Testing for normal distribution, remove  $> 2\sigma$
- Normalization
  - e.g., range  $[0, 1]$ , power transform
- Fill in missing values

# DATA CLEANING TOOLS

Google refine government IT contracts Permalink

Facet / Filter Undo / Redo 5

Refresh Reset All Remove All

**Type of Contract** change

783 choices Sort by: name count Cluster

Firm Fixed Price 836  
FFP: Firm Fixed Price 679  
T&M: Time & Materials 361  
Time and Materials 232  
Time & Materials 189 edit include  
CPFF: Cost Plus Fixed Fee 183  
CPAF: Cost Plus Award Fee 130  
Task Based Indefinite  
Delivery/Indefinite Quantity (ID/IQ) Time &  
Materials (T&M) Task Order 115  
Firm-Fixed-Price 112  
Fixed Price 105

**5200 rows**

Show as: rows records Show: 5 10 25 50 rows

		Contract ID	Contractor Name	Type of Contract	Date of A
1.	1939	ASAP SOFTWARE EXPRESS INC DELL MARKETING L.P.	Microsoft Enterprise Agreement	04/01/2009	
2.	1940	BMC SOFTWARE DISTRIBUTION INCORPORATED	Remedy Service Desk Maintenance	04/01/2009	
3.	1941	GOVCONNECTION INCORPORATED	Cisco SmartNet	05/01/2009	
4.	1942	ITS CORPORATION	Time & Materials	12/31/2008	
5.	7490	SENET INTERNATIONAL CORPORATIO	Firm Fixed Price C&A	05/04/2009	
6.	1945		firm fixed price	01/26/2009	
7.	1946	IT FEDERAL SALES LIMITED LIABILITY COMPANY	firm fixed price	10/01/2009	
8.	1947		firm fixed price	09/30/2009	

OpenRefine (formerly Google Refine), Trifacta Wrangler, Drake, etc.,

# DIFFERENT CLEANING TOOLS

- Outlier detection
- Data deduplication
- Data transformation
- Rule-based data cleaning and rule discovery
  - (conditional) functional dependencies and other constraints
- Probabilistic data cleaning

Further reading: Ilyas, Ihab F., and Xu Chu. [Data cleaning](#). Morgan & Claypool, 2019.

# DATA SCHEMA

# DATA SCHEMA

- Define expected format of data
  - expected fields and their types
  - expected ranges for values
  - constraints among values (within and across sources)
- Data can be automatically checked against schema
- Protects against change; explicit interface between components

# SCHEMA IN RELATIONAL DATABASES

```
CREATE TABLE employees (  
    emp_no      INT          NOT NULL,  
    birth_date  DATE         NOT NULL,  
    name        VARCHAR(30)  NOT NULL,  
    PRIMARY KEY (emp_no));  
CREATE TABLE departments (  
    dept_no     CHAR(4)       NOT NULL,  
    dept_name   VARCHAR(40)   NOT NULL,  
    PRIMARY KEY (dept_no), UNIQUE KEY (dept_name));  
CREATE TABLE dept_manager (  
    dept_no     CHAR(4)       NOT NULL,  
    emp_no      INT          NOT NULL,  
    FOREIGN KEY (emp_no) REFERENCES employees (emp_no),  
    FOREIGN KEY (dept_no) REFERENCES departments (dept_no),  
    PRIMARY KEY (emp_no, dept_no));
```



# SCHEMA-LESS DATA EXCHANGE

- CSV files
- Key-value stores (JSoN, XML, Nosql databases)
- Message brokers
- REST API calls
- R/Pandas Dataframes

```
1::Toy Story (1995)::Animation|Children's|Comedy  
2::Jumanji (1995)::Adventure|Children's|Fantasy  
3::Grumpier Old Men (1995)::Comedy|Romance
```

```
10|53|M|lawyer|90703  
11|39|F|other|30329  
12|28|F|other|06405  
13|47|M|educator|29206
```

# EXAMPLE: APACHE AVRO

```
{  "type": "record",
  "namespace": "com.example",
  "name": "Customer",
  "fields": [{
    "name": "first_name",
    "type": "string",
    "doc": "First Name of Customer"
  },
  {
    "name": "age",
    "type": "int",
    "doc": "Age at the time of registration"
  }
]
}
```

# EXAMPLE: APACHE AVRO

- Schema specification in JSON format
- Serialization and deserialization with automated checking
- Native support in Kafka
- Benefits
  - Serialization in space efficient format
  - APIs for most languages (ORM-like)
  - Versioning constraints on schemas
- Drawbacks
  - Reading/writing overhead
  - Binary data format, extra tools needed for reading
  - Requires external schema and maintenance
  - Learning overhead

## Speaker notes

Further readings eg <https://medium.com/@stephane.maarek/introduction-to-schemas-in-apache-kafka-with-the-confluent-schema-registry-3bf55e401321>, <https://www.confluent.io/blog/avro-kafka-data/>,  
<https://avro.apache.org/docs/current/>

# MANY SCHEMA FORMATS

## Examples

- Avro
- XML Schema
- Protobuf
- Thrift
- Parquet
- ORC

# DISCUSSION: DATA SCHEMA FOR INVENTORY SYSTEM?

Product Database:

ID	Name	Weight	Description	Size	Vendor
...	...	...	...	...	...


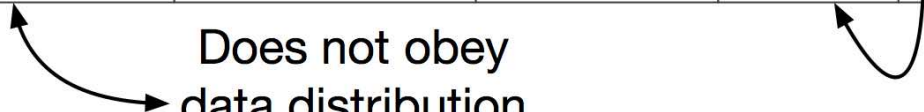

Stock:

ProductID	Location	Quantity
...	...	...

Sales history:

UserID	ProductId	DateTime	Quantity	Price
...	...	...	...	...

# DETECTING INCONSISTENCIES

	DBAName	AKAName	Address	City	State	Zip	
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	<b>Chicago</b>	IL	<b>60608</b>	 Conflicts
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	<b>60609</b>	
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	<b>60609</b>	
t4	<b>Johnnyo's</b>	Johnnyo's	3465 S Morgan ST	<b>Cicago</b>	IL	60608	
 Does not obey data distribution							 Conflict





# DATA QUALITY RULES

- Invariants on data that must hold
- Typically about relationships of multiple attributes or data sources, eg.
  - ZIP code and city name should correspond
  - User ID should refer to existing user
  - SSN should be unique
  - For two people in the same state, the person with the lower income should not have the higher tax rate
- Classic integrity constraints in databases or conditional constraints
- Rules can be used to reject data or repair it

# EXMAPLE: HOLOCLEAN



- User provides rules as integrity constraints (e.g., "two entries with the same name can't have different city")
- Detect violations of the rules in the data; also detect statistical outliers
- Automatically generate repair candidates (with probabilities)

Image source: Theo Rekatsinas, Ihab Ilyas, and Chris Ré, “[HoloClean - Weakly Supervised Data Repairing](#).” Blog, 2017.

# DISCOVERY OF DATA QUALITY RULES

- Rules directly taken from external databases
  - e.g. zip code directory
- Given clean data,
  - several algorithms that find functional relationships ( $X \Rightarrow Y$ ) among columns
  - algorithms that find conditional relationships (if  $Z$  then  $X \Rightarrow Y$ )
  - algorithms that find denial constraints ( $X$  and  $Y$  cannot cooccur in a row)
- Given mostly clean data (probabilistic view),
  - algorithms to find likely rules (e.g., association rule mining)
  - outlier and anomaly detection
- Given labeled dirty data or user feedback,
  - supervised and active learning to learn and revise rules
  - supervised learning to learn repairs (e.g., spell checking)

Further reading: Ilyas, Ihab F., and Xu Chu. [Data cleaning](#). Morgan & Claypool, 2019.

# ASSOCIATION RULE MINING

- Sale 1: Bread, Milk
- Sale 2: Bread, Diaper, Beer, Eggs
- Sale 3: Milk, Diaper, Beer, Coke
- Sale 4: Bread, Milk, Diaper, Beer
- Sale 5: Bread, Milk, Diaper, Coke

## Rules

- {Diaper, Beer} -> Milk (40% support, 66% confidence)
- Milk -> {Diaper, Beer} (40% support, 50% confidence)
- {Diaper, Beer} -> Bread (40% support, 66% confidence)

*(also useful tool for exploratory data analysis)*

Further readings: Standard algorithms and many variations, see [Wikipedia](#)

# DISCUSSION: DATA QUALITY RULES IN INVENTORY SYSTEM



# DATA LINTER

Further readings: Nick Hynes, D. Sculley, Michael Terry. "[The Data Linter: Lightweight Automated Sanity Checking for ML Data Sets](#)." NIPS Workshop on ML Systems (2017)

# EXCURSION: STATIC ANALYSIS AND CODE LINTERS

*Automate routine inspection tasks*

```
if (user.jobTitle = "manager") {  
    ...  
}
```

```
function fn() {  
    x = 1;  
    return x;  
    x = 3; // dead code  
}
```

```
PrintWriter log = null;  
if (anyLogging) log = new PrintWriter(...);  
if (detailedLogging) log.println("Log started");
```

# STATIC ANALYSIS

- Analyzes the structure/possible executions of the code, without running it
- Different levels of sophistication
  - Simple heuristic and code patterns (linters)
  - Sound reasoning about all possible program executions
- Tradeoff between false positives and false negatives
- Often supporting annotations needed (e.g., @Nullable)
- Tools widely available, open source and commercial



```
1
2
3  for (i = 0; i <= 10; i++)
4  {
5      if (i == 3)
6
7
8
9      var out = "The value is now " + i;
10     document.write(out);
11     document.write(text: "<br/>");
12 }
```

ESLint: Opening curly brace does not appear on the same line as controlling statement. (brace-style)





# A LINTER FOR DATA?



# DATA LINTER AT GOOGLE

- Miscoding
  - Number, date, time as string
  - Enum as real
  - Tokenizable string (long strings, all unique)
  - Zip code as number
- Outliers and scaling
  - Unnormalized feature (varies widely)
  - Tailed distributions
  - Uncommon sign
- Packaging
  - Duplicate rows
  - Empty/missing data

Further readings: Hynes, Nick, D. Sculley, and Michael Terry. [The data linter: Lightweight, automated sanity checking for ML data sets](#). NIPS MLSys Workshop. 2017.

# DETECTING DRIFT

# DRIFT & MODEL DECAY

*in all cases, models are less effective over time*

# DRIFT & MODEL DECAY

*in all cases, models are less effective over time*

- Concept drift
  - properties to predict change over time (e.g., what is credit card fraud)
  - over time: different expected outputs for same inputs
  - model has not learned the relevant concepts

# DRIFT & MODEL DECAY

*in all cases, models are less effective over time*

- Concept drift
  - properties to predict change over time (e.g., what is credit card fraud)
  - over time: different expected outputs for same inputs
  - model has not learned the relevant concepts
- Data drift
  - characteristics of input data changes (e.g., customers with face masks)
  - input data differs from training data
  - over time: predictions less confident, further from training data

# DRIFT & MODEL DECAY

*in all cases, models are less effective over time*

- Concept drift
  - properties to predict change over time (e.g., what is credit card fraud)
  - over time: different expected outputs for same inputs
  - model has not learned the relevant concepts
- Data drift
  - characteristics of input data changes (e.g., customers with face masks)
  - input data differs from training data
  - over time: predictions less confident, further from training data
- Upstream data changes
  - external changes in data pipeline (e.g., format changes in weather service)
  - model interprets input data incorrectly
  - over time: abrupt changes due to faulty inputs



# DRIFT & MODEL DECAY

*in all cases, models are less effective over time*

- Concept drift
  - properties to predict change over time (e.g., what is credit card fraud)
  - over time: different expected outputs for same inputs
  - model has not learned the relevant concepts
- Data drift
  - characteristics of input data changes (e.g., customers with face masks)
  - input data differs from training data
  - over time: predictions less confident, further from training data
- Upstream data changes
  - external changes in data pipeline (e.g., format changes in weather service)
  - model interprets input data incorrectly
  - over time: abrupt changes due to faulty inputs

## Speaker notes

- fix1: retrain with new training data or relabeled old training data
  - fix2: retrain with new data
  - fix3: fix pipeline, retrain entirely

# ON TERMINOLOGY

- Concept and data drift are separate concepts
- In practice and literature not always clearly distinguished
- Colloquially encompasses all forms of model degradations and environment changes
- Define term for target audience

# WATCH FOR DEGRADATION IN PREDICTION ACCURACY



Image source: Joel Thomas and Clemens Mewald. [Productionizing Machine Learning: From Deployment to Drift Detection](#). Databricks Blog, 2019

# INDICATORS OF CONCEPT DRIFT

*How to detect concept drift in production?*



# INDICATORS OF CONCEPT DRIFT

- Model degradations observed with telemetry
- Telemetry indicates different outputs over time for similar inputs
- Relabeling training data changes labels
- Interpretable ML models indicate rules that no longer fit

*(many papers on this topic, typically on statistical detection)*

# DEALING WITH DRIFT

- Regularly retrain model on recent data
  - Use evaluation in production to detect decaying model performance
- Involve humans when increasing inconsistencies detected
  - Monitoring thresholds, automation
- Monitoring, monitoring, monitoring!

# DIFFERENT FORMS OF DATA DRIFT

- Structural drift
  - Data schema changes, sometimes by infrastructure changes
  - e.g., 4124784115 -> 412 - 478 - 4115
- Semantic drift
  - Meaning of data changes, same schema
  - e.g., Netflix switches from 5-star to +/- rating, but still uses 1 and 5
- Distribution changes
  - e.g., credit card fraud differs to evade detection
  - e.g., marketing affects sales of certain items





# DETECTING DATA DRIFT

- Compare distributions over time (e.g., t-test)
- Detect both sudden jumps and gradual changes
- Distributions can be manually specified or learned (see invariant detection)



# DATA DISTRIBUTION ANALYSIS

- Plot distributions of features (histograms, density plots, kernel density estimation)
  - Identify which features drift
- Define distance function between inputs and identify distance to closest training data (eg., wasserstein and energy distance, see also kNN)
- Formal models for *data drift contribution* etc exist
- Anomaly detection and "out of distribution" detection
- Observe distribution of output labels

# DATA DISTRIBUTION EXAMPLE

<https://rpubs.com/ablythe/520912>

# MICROSOFT AZURE DATA DRIFT DASHBOARD



Image source and further readings: [Detect data drift \(preview\) on models deployed to Azure Kubernetes Service \(AKS\)](#)

# DISCUSSION: INVENTORY SYSTEM

*What kind of drift might be expected? What kind of detection/monitoring?*



# **QUALITY ASSURANCE FOR THE DATA PROCESSING PIPELINES**

# ERROR HANDLING AND TESTING IN PIPELINE

Avoid silent failures!

- Write testable data acquisition and feature extraction code
- Test this code (unit test, positive and negative tests)
- Test retry mechanism for acquisition + error reporting
- Test correct detection and handling of invalid input
- Catch and report errors in feature extraction
- Test correct detection of data drift
- Test correct triggering of monitoring system
- Detect stale data, stale models

*More in a later lecture.*

# SUMMARY

- Data and data quality are essential
- Data from many sources, often inaccurate, imprecise, inconsistent, incomplete, ... -- many different forms of data quality problems
- Understand the data with exploratory data analysis
- Many mechanisms for enforcing consistency and cleaning
  - Data schema ensures format consistency
  - Data quality rules ensure invariants across data points
  - Data linter detects common problems
- Concept and data drift are key challenges -- monitor
- Quality assurance for the data processing pipelines

