SCIENCE
PASSION
TECHNOLOGY

Architecture of ML Systems 09 Data Acquisition and Preparation

Matthias Boehm

Graz University of Technology, Austria
Computer Science and Biomedical Engineering
Institute of Interactive Systems and Data Science
BMVIT endowed chair for Data Management

Last update: June 07, 2019

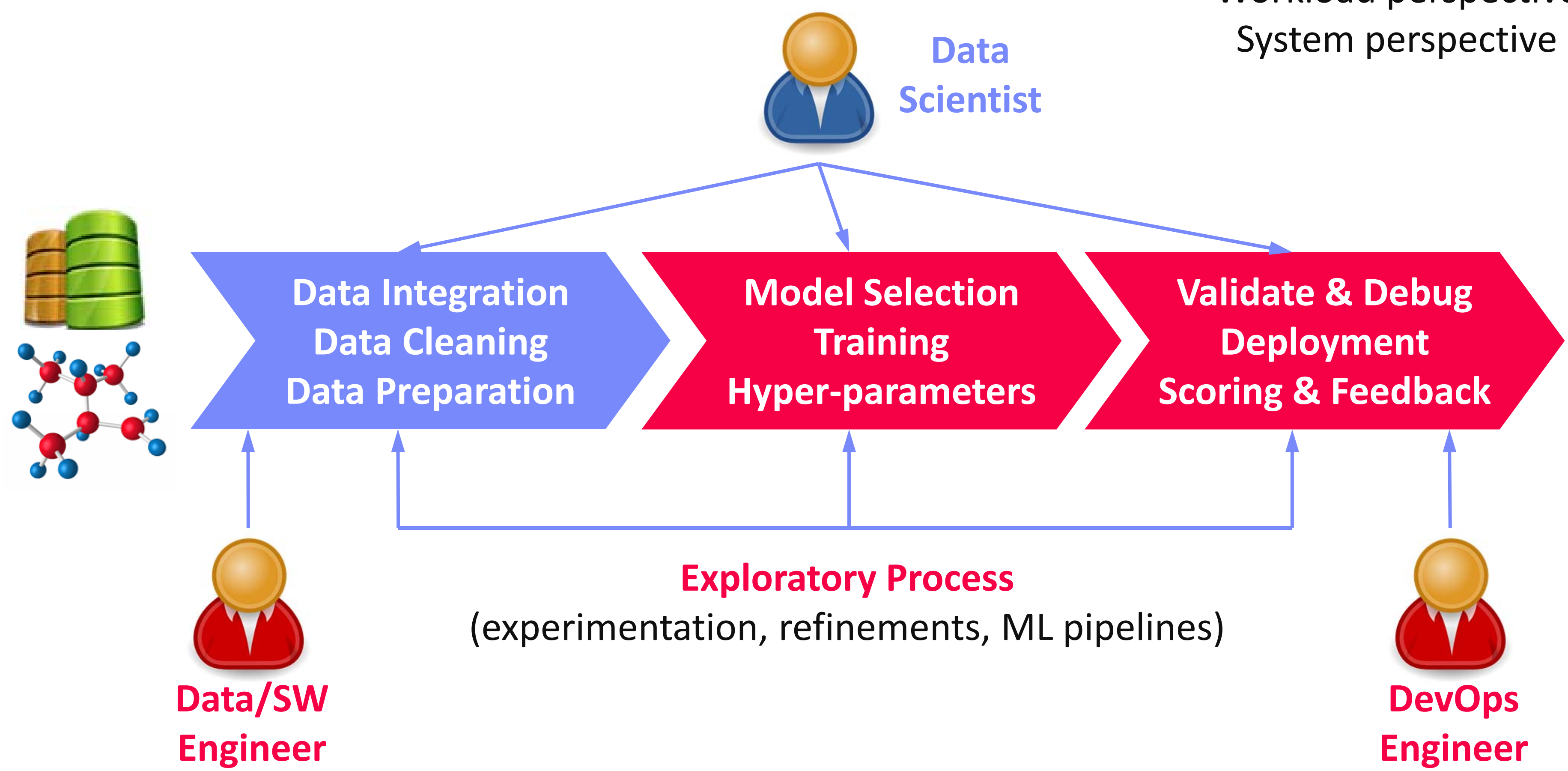
Announcements/Org

- **#1 Programming/Analysis Projects**
  - **#1 Auto Differentiation**
  - **#5 LLVM Code Generator**
  - **#12 Information Extraction from Unstructured PDF/HTML**

  ➔ Keep code PRs / status updates in mind

Recap: The Data Science Lifecycle

Data-centric View:
Application perspective
Workload perspective
System perspective

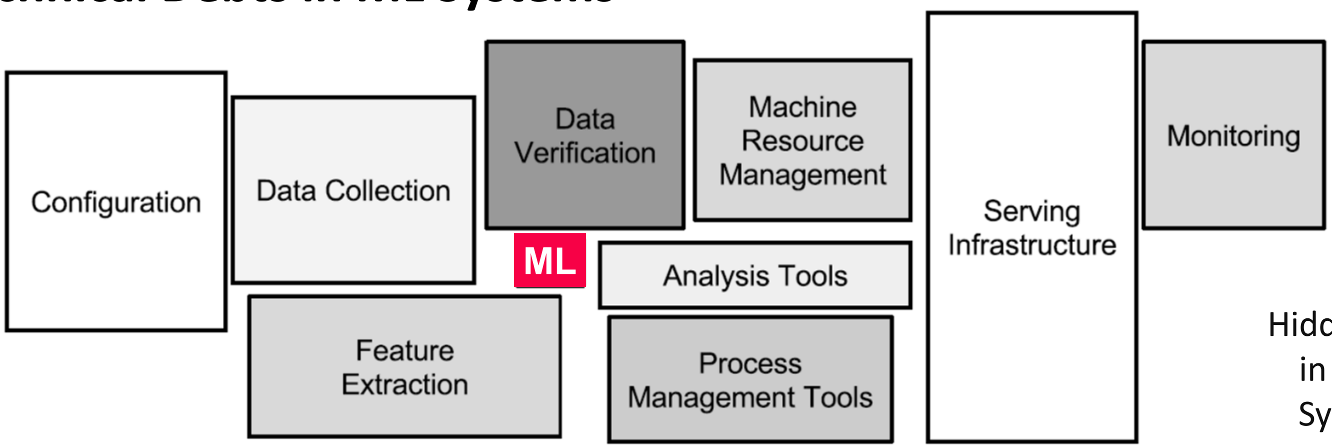


The 80% Argument

- **Data Sourcing Effort**
  - Data scientists spend **80-90% time** on finding relevant datasets and data integration/cleaning.

[Michael Stonebraker, Ihab F. Ilyas: Data Integration: The Current Status and the Way Forward. **IEEE Data Eng. Bull. 41(2) (2018)**]

- **Technical Debts in ML Systems**



[D. Sculley et al.: Hidden Technical Debt in Machine Learning Systems. **NIPS 2015**]

  - Glue code, pipeline jungles, dead code paths
  - Plain-old-data types, multiple languages, prototypes
  - Abstraction and configuration debts
  - Data testing, reproducibility, process management, and cultural debts

Agenda

- **Data Acquisition and Integration**
- **Data Preparation and Feature Engineering**
- **Data Transformation and Cleaning**
- **Data Augmentation**

"least enjoyable tasks in data science lifecycle"

Data Acquisition and Integration

Data Integration for ML and
ML for Data Integration

Data Sources and Heterogeneity

- **Terminology**
  - **Integration** (Latin integer = whole): consolidation of data objects / sources
  - **Homogeneity** (Greek homo/homoios = same): similarity
  - **Heterogeneity**: dissimilarity, different representation / meaning
- **Heterogeneous IT Infrastructure**
  - Common enterprise IT infrastructure contains >100s of heterogeneous systems and applications
  - E.g., health care data management: 20 - 120 systems

- **Multi-Modal Data (example health care)**
  - Structured patient data, patient records incl. prescribed drugs
  - Knowledge base drug APIs (active pharmaceutical ingredients) + interactions
  - Doctor notes (text), diagnostic codes, outcomes
  - Radiology images (e.g., MRI scans), patient videos
  - Time series (e.g., EEG, ECoG, heart rate, blood pressure)

➔ **Early vs late fusion**

Types of Data Formats

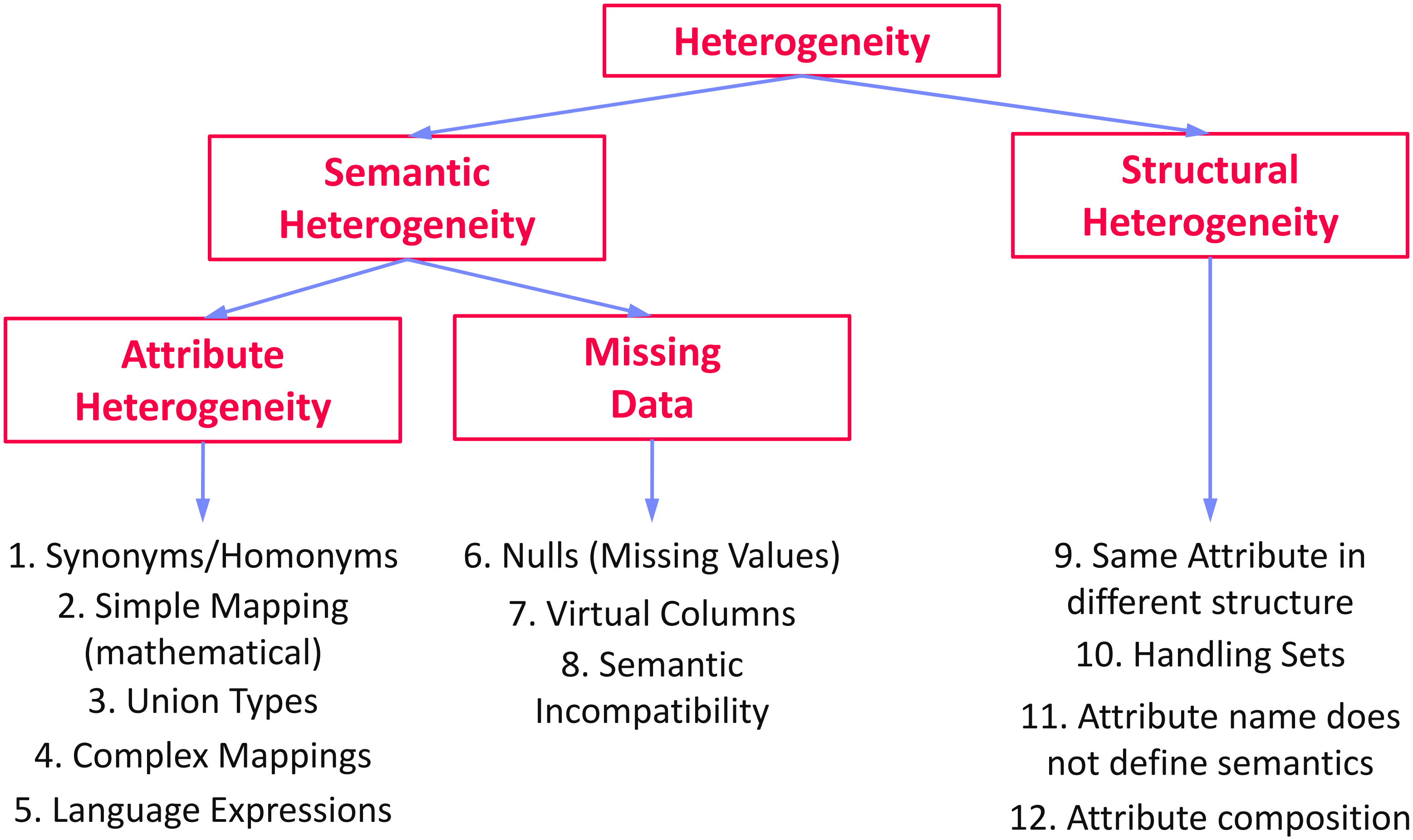
- **General-Purpose Formats**
  - CSV (comma separated values), JSON (javascript object notation), XML, Protobuf
  - CLI/API access to DBs, KV-stores, doc-stores, time series DBs, etc
- **Sparse Matrix Formats**
  - Matrix market: text IJV (row, col, value)
  - Libsvm: text compressed sparse rows
  - Scientific formats: NetCDF, HDF5

```
%%MatrixMarket matrix coordinate real general
% ------------------------------------------
% 0 or more comment lines
% ------------------------------------------
5  5  8
1 1 1.000e+00
2 2 1.050e+01
3 3 1.500e-02
1 4 6.000e+00
4 2 2.505e+02
4 4 -2.800e+02
4 5 3.332e+01
5 5 1.200e+01
```

- **Large-Scale Data Format**
  - Parquet (columnar file format)
  - Arrow (cross-platform columnar in-memory data)
- **Domain-Specific Formats**
  - Health care: DICOM images, HL7 message (health-level seven, XML)
  - Automotive: MDF (measurements), CDF (calibrations), ADF (auto-lead XML)
  - Smart production: OPC (open platform communications)

Types of Heterogeneity

[J. Hammer, M. Stonebraker, and O. Topsakal: THALIA: Test Harness for the Assessment of Legacy Information Integration Approaches. U Florida, TR05-001, **2005**]



Identification of Data Sources

- **Data Catalogs**
  - Data curation in repositories for finding relevant datasets in data lakes
  - Augment data with open and linked data sources

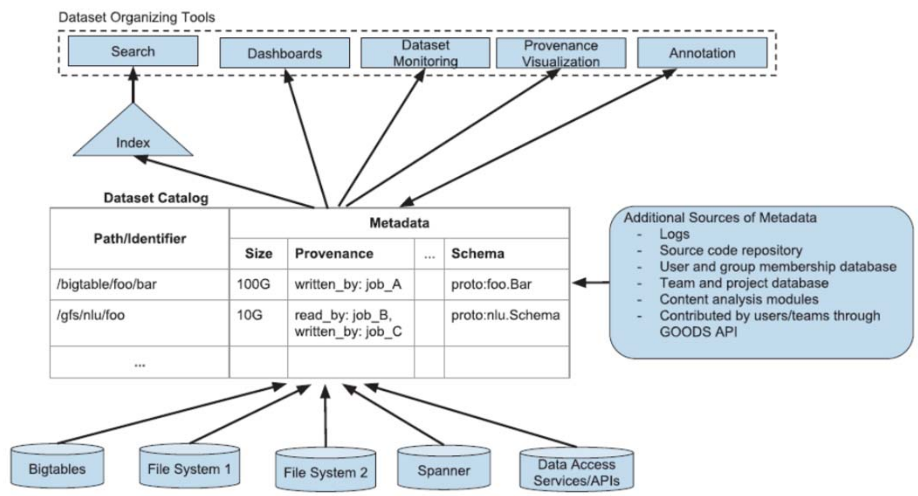
- **Examples**

[Alon Y. Halevy et al: Goods: Organizing Google's Datasets. **SIGMOD 2016**]

SAP Data Hub

[SAP Sapphire Now 2019]

Google Data Search

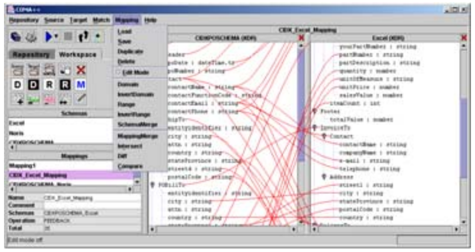


Schema Detection and Integration

- **Schema Detection**
  - Sample of the input dataset → infer the schema (e.g., data types)

- **Schema Matching**
  - Semi-automatic mapping of schema S1 to schema S2
  - Approaches: Schema- vs instance-based; element- vs structure-based; linguistic vs rules
  - Hybrid and composite matchers
  - Global schema matching (one-to-one): stable marriage problem

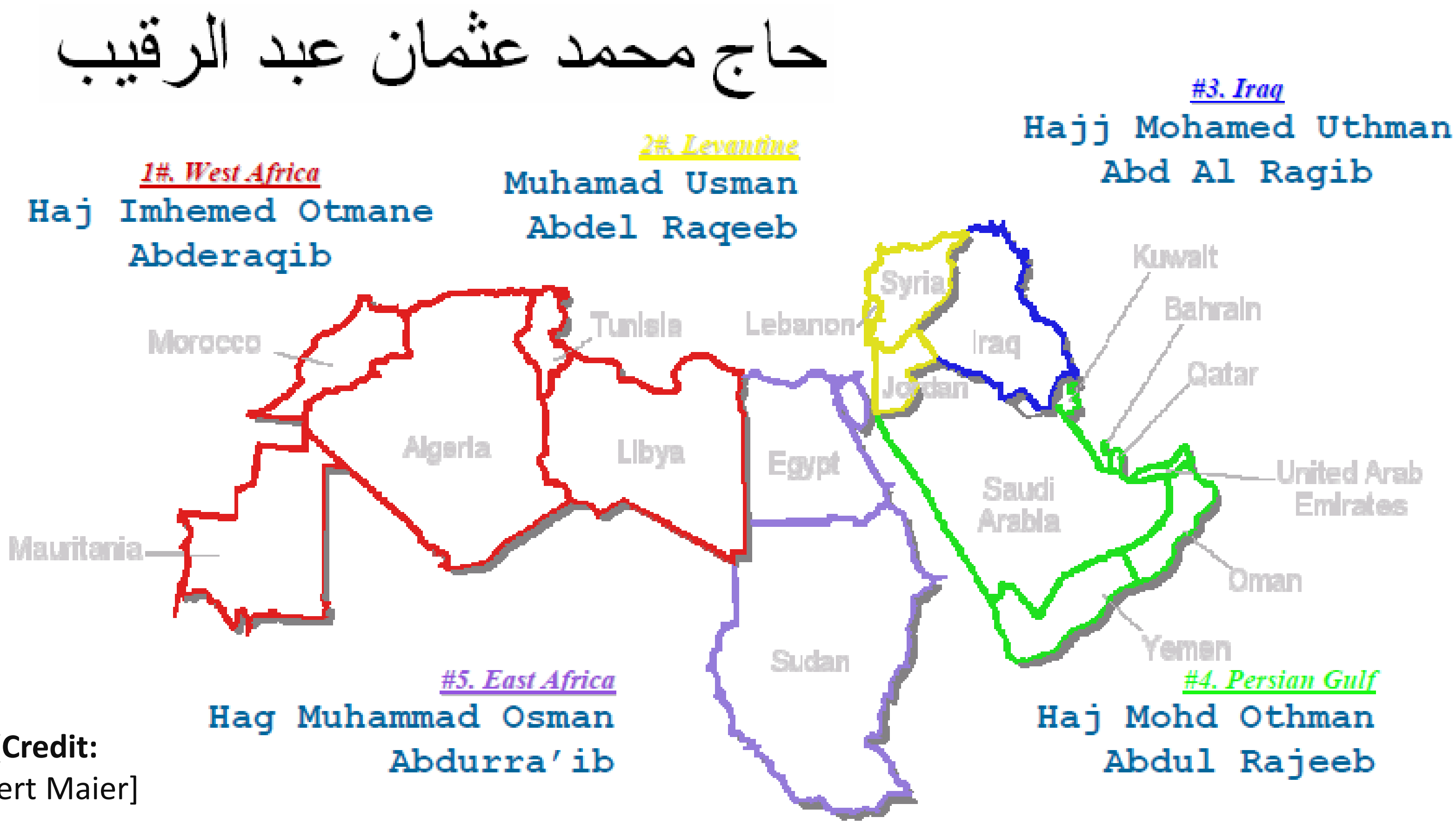
[**Credit:** Erhard Rahm]



- **Schema Mapping**
  - Given two schemas and correspondences, generate transformation program
  - **Challenges:** complex mappings (1:N cardinality), new values, PK-FK relations and nesting, creation of duplicates, different data types, sematic preserving

Excursus: Sources of Duplicates

- **A Name in Different Arabic Countries**



[**Credit:** Albert Maier]

Excursus: Sources of Duplicates, cont.

- **Misspellings are Very Common**

488941 britney spears	29 britent spears	9 brinttany spears	5 brney spears	3 britiy spears
40134 brittany spears	29 brittnany spears	9 britanay spears	5 broitney spears	3 britmeny spear
36315 brittney spears	29 britttany spears	9 britinany spears	5 brotny spears	3 britneeey spea
24342 britany spears	29 btiney spears	9 britn spears	5 bruteny spears	3 britnehy spear
7331 britny spears	26 birttney spears	9 britnew spears	5 btiyney spears	3 britnely spear
6633 briteny spears	26 breitney spears	9 britneyn spears	5 btrittney spears	3 britnesy spear
2696 britteny spears	26 brinity spears	9 britrney spears	5 gritney spears	3 britnetty spea
1807 briney spears	26 britenay spears	9 brtiny spears	5 spritney spears	3 britnex spears
1635 brittny spears	26 britneyt spears	9 brtittney spears	4 bittny spears	3 britneyxxx spe
1479 brintey spears	26 brittan spears	9 brtny spears	4 bnritney spears	3 britnity spear
1479 britanny spears	26 brittne spears	9 brytny spears	4 brandy spears	3 britntey spear
1338 britiny spears	26 btittany spears	9 rbitney spears	4 brbritney spears	3 britnyey spear
1211 britnet spears	24 beitney spears	8 birtiny spears	4 breatiny spears	3 britterny spea
1096 britiney spears	24 birteny spears	8 bithney spears	4 breetney spears	3 brittneey spea
991 britaney spears	24 brightney spears	8 brattany spears	4 bretiney spears	3 brittnney spea
991 britnay spears	24 brintiny spears	8 breitny spears	4 brfitney spears	3 brittnyey spea
811 brithney spears	24 britanty spears	8 breteny spears	4 briattany spears	3 brityen spears
811 brtiney spears	24 britenny spears	8 brightny spears	4 brieteny spears	3 briytney spear
664 birtney spears	24 britini spears	8 brintay spears	4 briety spears	3 brltney spears
664 brintney spears	24 britnwy spears	8 brinttey spears	4 briitny spears	3 broteny spears
664 briteney spears	24 brittni spears	8 briotney spears	4 briittany spears	3 brtaney spears
601 bitney spears	24 brittnie spears	8 britanys spears	4 brinie spears	3 brtiiany spear
601 brinty spears	21 biritney spears	8 britley spears	4 brinteney spears	3 brtinay spears
544 brittaney spears	21 birtany spears	8 britneyb spears	4 brintne spears	3 brtinney spear
544 brittnay spears	21 biteny spears	8 britnrey spears	4 britaby spears	3 brtitany spear
364 britey spears	21 bratney spears	8 britnty spears	4 britaey spears	3 brtiteny spear
364 brittiny spears	21 britani spears	8 brittner spears	4 britainey spears	3 brtnet spears
329 brtney spears	21 britanie spears	8 brottany spears	4 britinie spears	3 brytiny spears
269 bretney spears	21 briteany spears	7 baritney spears	4 britinney spears	3 btney spears
269 britneys spears	21 brittay spears	7 birntey spears	4 britmney spears	3 drittney spear
244 britne spears	21 brittinay spears	7 biteney spears	4 britnear spears	3 pretney spears
244 brytney spears	21 brtany spears	7 bitiny spears	4 britnel spears	3 rbritney spear
220 breatney spears	21 brtiany spears	7 breateny spears	4 britneuy spears	2 barittany spea
220 britiany spears	19 birney spears	7 brianty spears	4 britnewy spears	2 bbbritney spea
199 britnney spears	19 brirtney spears	7 brintye spears	4 britnmey spears	2 bbitney spears
163 britnry spears	19 britnaey spears	7 britianny spears	4 brittaby spears	2 bbritny spears

Corrupted Data

- **Heterogeneity of Data Sources**
  - Update anomalies on denormalized data / eventual consistency
  - Changes of app/preprocessing over time (US vs us) → inconsistencies
- **Human Error**
  - Errors in semi-manual data collection, laziness (see default values), bias
  - Errors in data labeling (especially if large-scale: crowd workers / users)
- **Measurement/Processing Errors**
  - Unreliable HW/SW and measurement equipment (e.g., batteries)
  - Harsh environments (temperature, movement) → aging

Uniqueness & duplicates — **Contradictions & wrong values** — **Missing Values** — **Ref. Integrity**

[**Credit:** Felix Naumann]

ID	Name	BDay	Age	Sex	Phone	Zip
3	Smith, Jane	05/06/1975	44	F	999-9999	98120
3	John Smith	38/12/1963	55	M	867-4511	11111
7	Jane Smith	05/06/1975	24	F	567-3211	98120

Zip	City
98120	San Jose
90001	Lost Angeles

Typos

Sanity Checks before Training First Model

- **Check a feature's min, max, and most common value**
  - Ex: Latitude values must be within the range [-90, 90] or [-π/2, π/2]

- **The histograms of continuous or categorical values are as expected**
  - Ex: There are similar numbers of positive and negative labels

- **Whether a feature is present in enough examples**
  - Ex: Country code must be in at least 70% of the examples

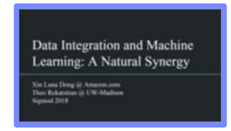
- **Whether a feature has the right number of values (i.e., cardinality)**
  - Ex: There cannot be more than one age of a person


Google
Data Management Challenges in Production Machine Learning
Neoklis Polyzotis, Sudip Roy, Steven Whang, Martin Zinkevich

[Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, Martin Zinkevich: Data Management Challenges in Production Machine Learning. **SIGMOD 2017**]

Data Integration for ML and ML for DI

[Xin Luna Dong, Theodoros Rekatsinas: Data Integration and Machine Learning: A Natural Synergy. **SIGMOD 2018**]



- **#1 Data Extraction**
  - Extracting structured data from un/semi-structured data
  - Rule- and ML-based extractors, combination w/ CNN
- **#2 Schema Alignment**
  - Schema matching for consolidating data from heterogeneous systems
  - Spatial and Temporal alignment via provenance and query processing (e.g., sensor readings for object along a production pipeline)
- **#3 Entity Linking**
  - Linking records to entities (deduplication)
  - Blocking, pairwise matching, clustering, ML, Deep ML (via entity embedding)
- **#4 Data Fusion**
  - Resolve conflicts, necessary in presence of erroneous data
  - Rule- and ML-based, probabilistic GM, Deep ML (RBMs, graph embeddings)

Data Preparation and Feature Engineering

Overview Feature Engineering

- **Terminology**
  - Matrix X of m observations (rows) and n features (columns)
  - **Continuous features:** numerical values (aka scale features)
  - **Categorical features:** non-numerical values, represent groups
  - **Ordinal features:** non-numerical values, associated ranking
  - Feature space: multi-dimensional space of features → curse of dimensionality
- **Feature Engineering**
  - Bringing multi-modal data and features into numeric representation
  - Use domain expertise to expose potentially predictive features to the ML model training algorithm
- **Excursus: Representation Learning**
  - Neural networks can be viewed as combined representation learning and model training (pros and cons: learned, repeated)
  - Mostly homogeneous inputs (e.g., image), research on multi-modal learning

➔ **Principle: If same accuracy, prefer simple model** (cheap, robust, explainable)

Recoding

- **Summary**
  - Numerical encoding of categorical features (arbitrary strings)
  - Map distinct values to integer domain (potentially combined w/ one-hot)

City	State
San Jose	CA
New York	NY
San Francisco	CA
Seattle	WA
New York	NY
Boston	MA
San Francisco	CA
Los Angeles	CA
Seattle	WA

Dictionaries

```
{San Jose : 1,
New York : 2,
San Francisco : 3,
Seattle : 4,
Boston : 5,
Los Angeles : 6}

{CA : 1,
NY : 2,
WA : 3,
MA : 4}
```

City	State
1	1
2	2
3	1
4	3
2	2
5	4
3	1
6	1
4	3

Feature Hashing

- **Summary**
  - Numerical encoding of categorical features (arbitrary strings)
  - Hash input to k buckets via hash(value) % k (often combined w/ one-hot)

for k = 5:

City	hash(value) % k	City
San Jose	1993955031 % 5 → 1	1
New York	1382994575 % 5 → 0	0
San Francisco	1540367136 % 5 → 1	1
Seattle	-661909336 % 5 → 1	1
New York	1993955031 % 5 → 1	1
Boston	1995575789 % 5 → 4	4
San Francisco	1540367136 % 5 → 1	1
Los Angeles	-425347233 % 5 → 3	3
Seattle	-661909336 % 5 → 1	1

Efficient, but **collisions**

Binning (see also Quantization, Binarization)

- **Summary**
  - Encode of numerical features to integer domain (often combined w/ one-hot)
  - **Equi-width:** split (max-min)-range into k equal-sized buckets
  - **Equi-height:** compute data-driven ranges for k balanced buckets

Sqft
928.5
451
570.3
1,273
1,239
711.3
1,114
867

```
min = 451
max = 1,273
range = 822
```

Equal-sized numerical buckets (with k=3)

```
[451, 725) → 1
[725, 999) → 2
[999, 1,273] → 3
```

Allows modelling small, medium, large apartments

Sqft-Bins
2
1
1
3
3
1
3
2

One-hot Encoding

- **Summary**
  - Encode integer feature of cardinality d into sparse 0/1 vector of length d
  - Feature vectors of input features concatenated in sequence

City	State
1	1
2	2
3	1
4	3
2	2
5	4
3	1
6	1
4	3

→

C1	C2	C3	C4	C5	C6	S1	S2	S3	S4
1	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	1	0	0
0	0	1	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	1	1	0	0	0
0	0	0	1	0	0	0	0	1	0

Derived Features

- **Intercept Computation**
  - Add a column of ones to X for computing the intercept as a weight
  - Applies to regression and classification

```
X = cbind(X,
  matrix(1, nrow(X), 1));
```

- **Non-Linear Relationships**
  - Can be explicitly materialized as feature combinations
  - Example: Assumptions of underlying physical system
  - Arbitrary complex feature interactions: e.g., X_1^2 * X_2

```
// y ~ b1*X1 + b2*X1^2
X = cbind(X, X^2);
```

NLP Features

- **Basic NLP Feature Extraction**
  - **Sentence/word tokenization:** split into sentences/words (e.g., via stop words)
  - **Part of Speech (PoS) tagging:** label words verb, noun, adjectives (syntactic)
  - **Semantic role labeling:** label entities with their roles in actions (semantic)

- **Bag of Words and N-Grams**
  - Represent sentences as **bag** (multisets)

A B C A B E.
A D E D E D.

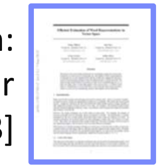
→

A	B	C	D	E
2	2	1	0	1
1	0	0	3	2

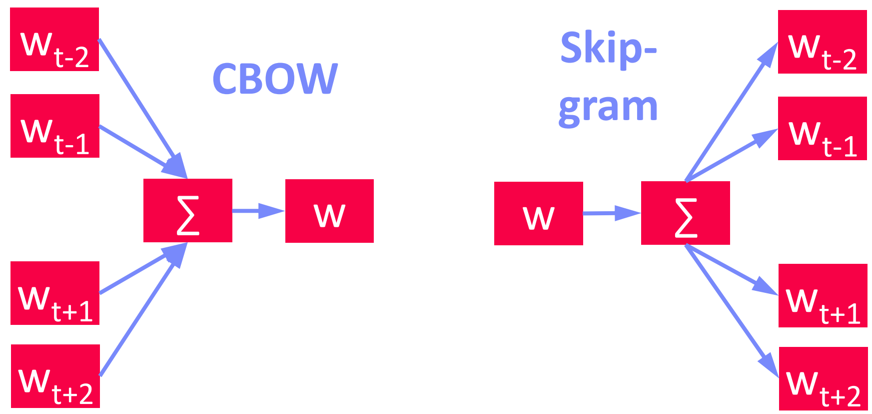
  - **Bi-grams:** bag-of-words for 2-sequences of words (order preserving)
  - **N-grams:** generalization of bi-grams to arbitrary-length sequences

NLP Features, cont.

[Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space. **ICLR (Workshop) 2013**]



- **Word Embeddings**
  - Trained (word → vector) mappings (~ 50-300 dims)
  - **Word2vec:** continuous bag-of-words (CBOW) or continuous skip-gram (github.com/dav/word2vec)
  - Subsampling frequent words
  - **Semantic preserving arithmetic operations** (+ ~ * of context distributions)




```
vec(Paris) ≈ vec(Berlin)
 – vec(Germany) + vec(France)
```

- **Use in Practice**
  - Often pre-trained word embeddings used in an **application-agnostic** way
  - If necessary, fine-tuning or training for task / domain
  - Various extensions: **Sentence2Vec**, **Doc2Vec**, **Node2Vec**

Example Spark ML

- **API Design**
  - **Transformers:** Feature transformations and learned models
  - **Estimators:** Algorithm that can be fit to produce a transformer
  - Compose ML pipelines from chains of transformers and estimators

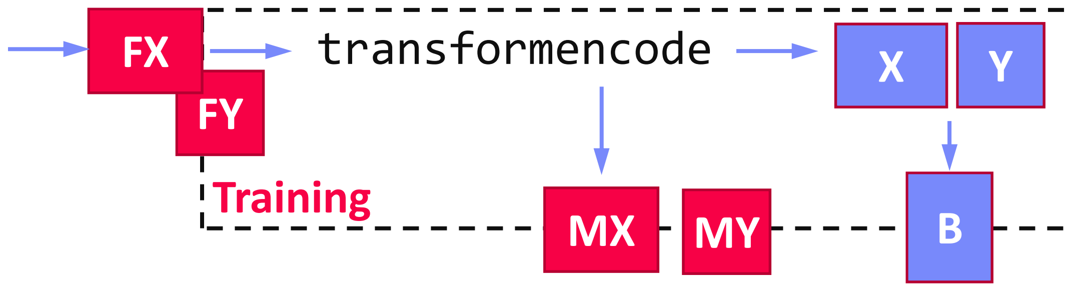
- **Example Pipeline**

```
// define pipeline stages
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(),
                      outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)

// create pipeline transformer via fit
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])
model = pipeline.fit(training)

// use of resulting ML pipeline
prediction = model.transform(test)
```

[https://spark.apache.org/docs/2.4.3/ml-pipeline.html]

Example SystemML/SystemDS

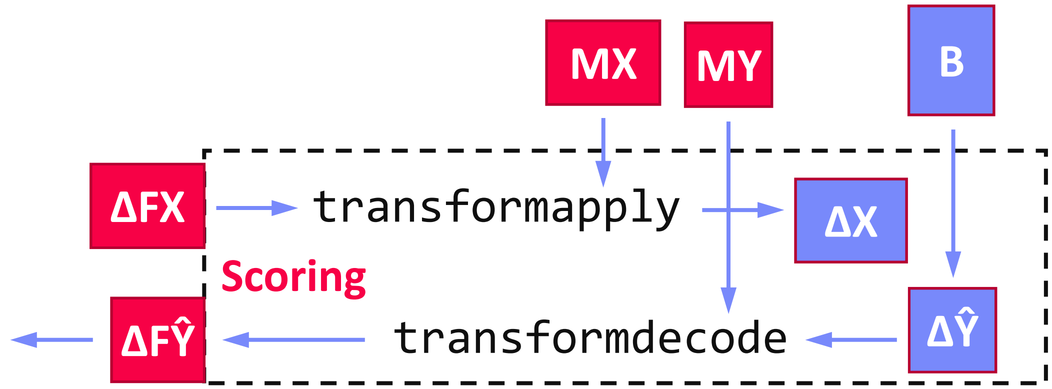
- **Feature Transformation during Training**




```
# read tokenized words
FX = read("./input/FX", data_type=FRAME); # sentence id, word, count
FY = read("./input/FY", data_type=FRAME); # sentence id, labels

# encode and one-hot encoding
[X0, MX] = transformencode(target=FX, spec="{recode:[2]}");
[Y0, MY] = transformencode(target=FY, spec="{recode:[2]}");
X = table(X0[,1], X0[,2], X0[,3]); # bag of words
Y = table(Y0[,1], Y0[,2]);         # bag of words

# model training via multi-label, multi-nominal logical regression
B = mlogreg(X, Y);
```

Example SystemML/SystemDS, cont.

- **Feature Transformation during Scoring**




```
# read tokenized words of test sentences
dFX = read("./input/dFX", data_type=FRAME); # sentence id, word, count

# encode and one-hot encoding
dX0 = transformapply(target=dFX, spec="{recode:[2]}", meta=MX);
dX = table(dX0[,1], dX0[,2], dX0[,3]); # bag of words

# model scoring and postprocessing (reshape, attach sentence ID, etc)
dYhat = (X %*% B) >= theta; ...;

# decode output labels: sentence id, label word
dFYhat = transformdecode(target=dYhat, spec="{recode:[2]}", meta=MY);
```

Data Transformation and Cleaning

Standardization and Normalization

- **#1 Standardization**
  - Centering and scaling to mean 0 and variance 1
  - **Ensures well-behaved training**
  - **Densifying operation**
  - Awareness of NaNs
  - Batch normalization in DNN: standardization of activations

```
X = X - colMeans(X);
X = X / sqrt(colVars(X));

X = replace(X, pattern=NaN,
  replacement=0); #robustness
```

- **#2 Normalization**
  - Rescale values into common range [0,1]
  - **Avoid bias to large-scale features**
  - Aka min-max normalization
  - Does not handle outliers

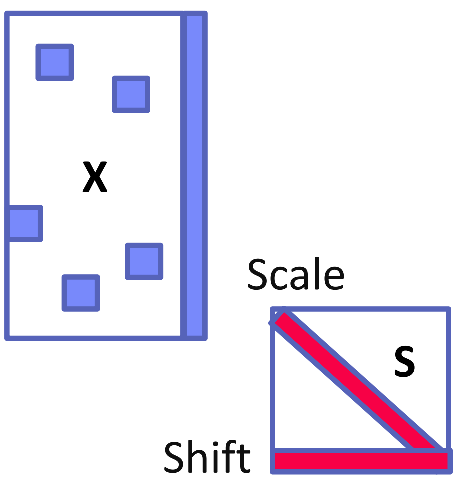
```
X = (X - colMins(X))
  / (colMaxs(X) - colMins(X));
```

Standardization and Normalization, cont.

- **#3 Deferred Standardization**
  - Avoid densifying dataset upfront by pushing standardization into inner loop iterations (use of dataset)
  - Let matrix-multiplication chain optimization + rewrites do the rest

- **Example**

Input w/ column of ones (intercept)




```
# operation w/ early standardized X
q = t(X) %*% diag(w) %*% X %*% B;
```

Substitute X with X %*% S

```
# operation w/ deferred standardization
q = t(S) %*% t(X) %*% diag(w)
   %*% X %*% S %*% B;
```

Outlier Detection and Removal

- **Winsorizing**
  - Replace tails of data distribution at user-specified threshold
  - Quantiles / std-dev
  - ➔ Reduce skew

```
# compute quantiles for lower and upper
q = quantile(X, matrix("0.05 0.95", 2, 1));

# replace values outside [ql,qu] w/ ql and qu
Y = ifelse(X < q[1,1], q[1,1], X);
Y = ifelse(Y > q[2,1], q[2,1], Y);
```

- **Truncation/Trimming**
  - See winsorizing, but remove data outside lower / upper thresholds

```
# remove values outside [ql,qu]
I = X < q[1,1] | X > q[1,1];
Y = removeEmpty(X, “rows”, select = I);
```

- **Largest Difference from Mean**

```
# determine largest diff from mean
I = (colMaxs(X)-colMeans(X))
  > (colMeans(X)-colMins(X));
Y = ifelse(I, colMaxs(X), colMins(X));
```

w/ CSE

Outlier Detection and Removal, cont.

- **Types of Outliers**
  - **Point outliers:** single data points far from the data distribution
  - **Contextual outliers:** noise or other systematic anomalies in data
  - **Sequence outliers:** sequence of values shows abnormal shape / aggregate
  - Univariate vs multivariate analysis
  - Beware of underlying assumptions (distributions)

- **Iterative Algorithms**
  - Iterative **winsoring/trimming** to X std-devs of mean
  - Various **clustering** algorithms (partitioning and density-based models)
  - **Frequent itemset mining** → rare itemset mining / sequence mining
  - **Probabilistic** and statistical modeling

Missing Value Imputation

- **Missing Value**
  - Application context defines if 0 is missing value or not
  - If differences between 0 and missing values, use NA or NaN
- **Basic Value Imputation**
  - General-purpose: replace by user-specified **constant**
  - **Continuous variables:** replace by **mean**
  - **Categorical variables:** replace by **median** or **mode** (most frequent category)
- **Iterative Algorithms** (chained-equation imputation methods)
  - Train ML model on available data to predict missing information (feature k → label, split data into training: observed, and scoring: missing)
  - Noise reduction: train models over feature subsets + averaging
- **Dynamic Imputation**
  - Data exploration w/ on-the-fly imputation
  - Optimal placement of imputation operations

[Jose Cambronero, John K. Feser, Micah Smith, Samuel Madden: Query Optimization for Dynamic Imputation. **PVLDB 2017**]

Excursus: Time Series Recovery

- **Motivating Use Case**
  - Given overlapping weekly aggregates y (daily moving average)
  - Reconstruct the original time series X
- **Problem Formulation**
  - Aggregates y
  - Original time series X (unknown)
  - Mapping O of subsets of X to y
  - ➔ **Least squares regression problem**

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}}_{\mathbf{O}} \times \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\mathbf{y}}$$

- **Advanced Method**
  - Discrete Cosine Transform (DCT) (sparsest spectral representation)
  - Non-negativity and smoothness constraints

[Faisal M. Almutairi et al: HomeRun: Scalable Sparse-Spectrum Reconstruction of Aggregated Historical Data. **PVLDB 2018**]

Selected Research Prototypes

- **ActiveClean (SampleClean)**
  - Suggest sample of data for manual cleaning (rule/ML-based detectors, **Simpson's paradox**)
  - Update dirty model with gradients of cleaned data (weighted gradients of previous clean data and newly cleaned data)

[Sanjay Krishnan et al: ActiveClean: Interactive Data Cleaning For Statistical Modeling. **PVLDB 2016**]

- **HoloClean**
  - Clean and enrich based on quality rules, value correlations, and reference data
  - Probabilistic models for capturing data generation
  - HoloDetect
    - **Learn data representations** of errors
    - **Data augmentation** w/ erroneous data from sample of clean data

[Alireza Heidari, Joshua McGrath, Ihab F. Ilyas, Theodoros Rekatsinas: HoloDetect: Few-Shot Learning for Error Detection, **SIGMOD 2019**]

- **Other Systems**
  - **AlphaClean** (generate data cleaning pipelines) [preprint]
  - **BoostClean** (generate repairs for domain value violations) [preprint]
  - Automated verification of data quality rules/constraints [PVLDB’18]

Data Augmentation

Next Week

Summary and Conclusions

- **Data Acquisition, Cleaning and Preparation**
  - Data Collection and Integration
  - Data Preparation and Feature Engineering
  - Data Transformation and Cleaning
  - Data Augmentation → **Next Week**

- **Next Lectures**
  - **10 Model Selection and Management** [Jun 14]
    - Including feature and model selection techniques
  - **11 Model Deployment and Serving** [Jun 21]
  - **12 Project Presentations, Conclusions, Q&A** [Jun 28]
    - Discussion current status