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
BMK endowed chair for Data Management

Last update: May 20, 2021

Announcements/Org

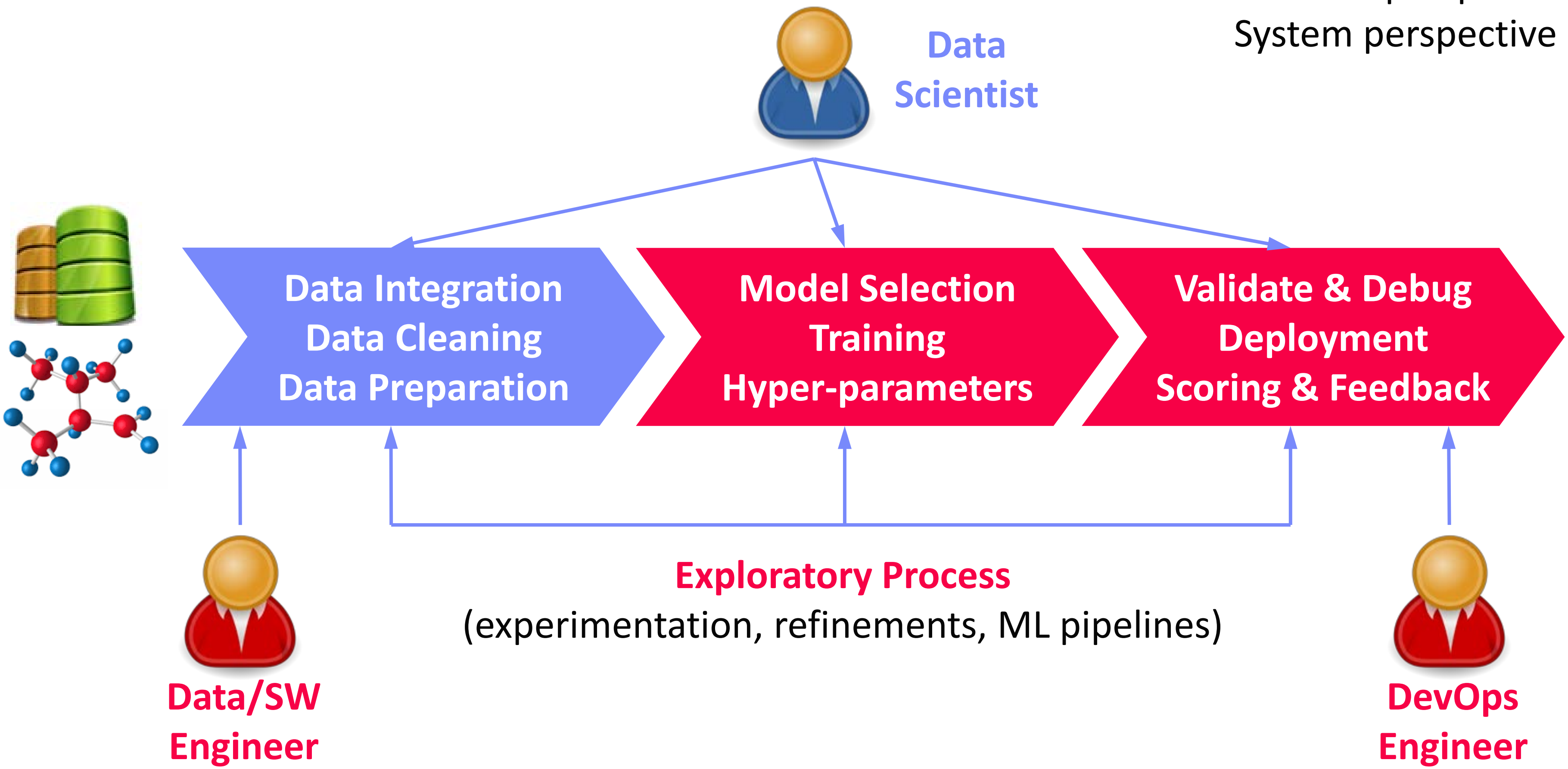
- **#1 Video Recording**
  - Link in **TeachCenter** & **TUbe** (lectures will be public)
  - https://tugraz.webex.com/meet/m.boehm
  - Corona traffic light **RED** → May 17: **ORANGE** (but **tests** required)
- **#2 Programming Projects / Exercises** (36/57)
  - **Apache SystemDS:** 24 projects / 37 students
  - **DAPHNE:** 2 projects / 2 students
  - **Exercises:** 10 projects / 18 students → TeachCenter

Deadline:
June 30 (soft)

- **#3 Exam Preferences**
  - Oral vs written exams? Requested Dates?
- **#4 Spark Summit 2021** (aka Data + AI Summit 2021)
  - May 24 – May 28, free registration
  - https://databricks.com/dataaisummit/north-america-2021

Recap: The Data Science Lifecycle

Data-centric View:
Application perspective
Workload perspective
System perspective

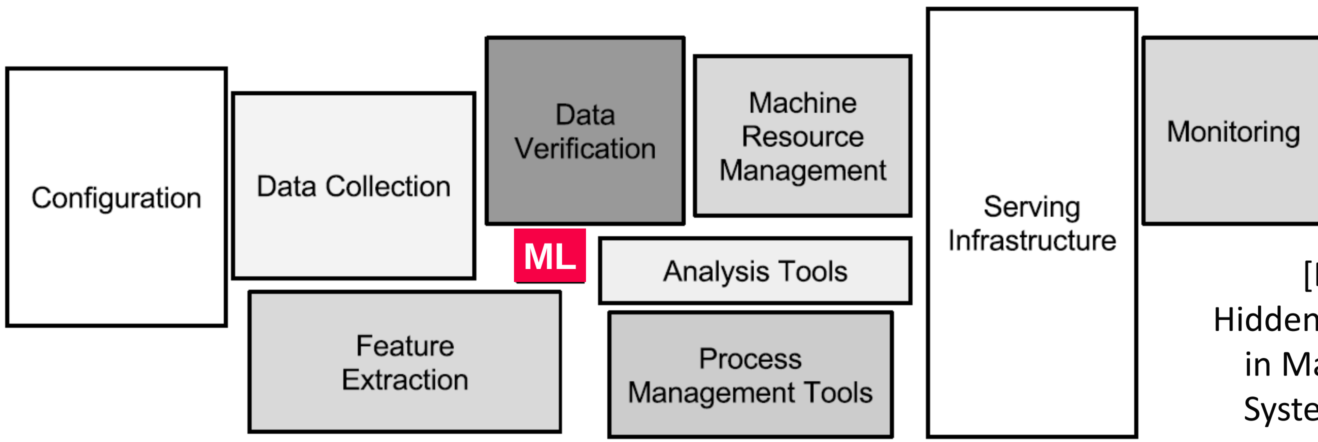


The 80% Argument

- **Data Sourcing Effort**
  - Data scientists spend **80-90% time** on finding, integrating, cleaning datasets

[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 (arrays), multiple languages, prototypes
  - Abstraction and configuration debts
  - Data testing, reproducibility, process management, and cultural debts

Agenda

- **Data Acquisition, Integration, and Validation**
- **Feature Transformations and Engineering**
- **Data Preparation and Cleaning**
- **Data Augmentation** (next week)

"least enjoyable tasks in data science lifecycle"

Data Integration and Large-Scale Analysis (DIA)
(bachelor/master)

Data Acquisition, Integration, and Data Validation

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 and distributed systems and applications**
  - E.g., health care data management: 20 - 120 systems

[**Credit:** Albert Maier]

- **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)

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 Formats**
  - **Parquet** (columnar file format)
  - **Arrow** (cross-platform columnar in-memory data)
- **Domain-Specific Formats**
  - Health care: **DICOM** images, **HL7** messages (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**]

Heterogeneity

- **Semantic Heterogeneity**
  - **Attribute Heterogeneity**
    - 1. Synonyms/homonyms
    - 2. Simple mapping (mathematical)
    - 3. Different data types
    - 4. Complex mappings
    - 5. Language expressions
  - **Missing Data**
    - 6. Nulls (Missing Values)
    - 7. Virtual columns
    - 8. Semantic incompatibility
- **Structural Heterogeneity**
  - 9. Same attribute in different structure
  - 10. Handling Sets
  - 11. Attribute name w/o semantics
  - 12. Attribute composition

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

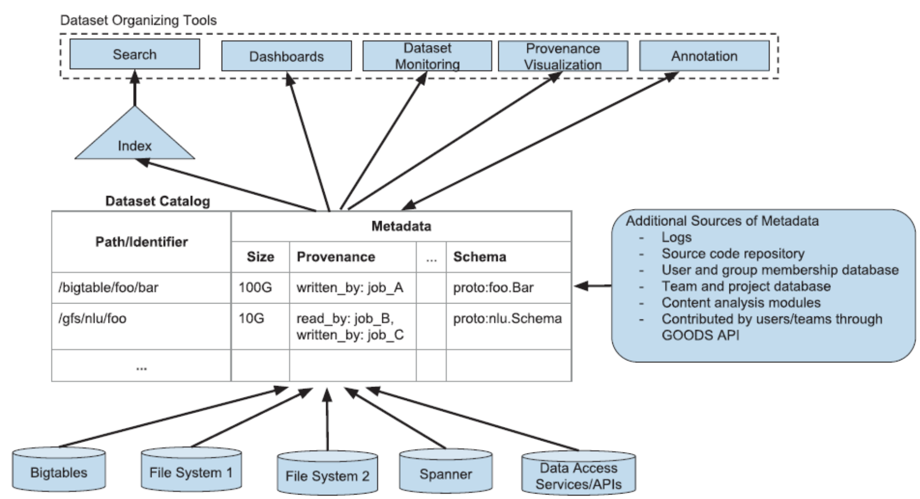
- **Examples**

[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

- **Syntactic Schema Detection**
  - Sample of the input dataset
  - Extract basic data types via rules, and regular expressions

```
./data/players.csv:
pid,name,pos,jnum,ncid,tid
5435,Miroslav Klose,FW,11,789,144
6909,Manuel Neuer,GK,1,163,308
```

```
Dataset<Row> ds = sc.read()
  .format("csv")
  .option("header", true)
  .option("inferSchema", true)
  .option("samplingRatio", 0.001)
  .load("./data/players.csv");
```

```
StructType(
  StructField(pid,IntegerType,true),
  StructField(name,StringType,true),
  StructField(pos,StringType,true),
  StructField(jnum,IntegerType,true),
  StructField(ncid,IntegerType,true),
  StructField(tid,IntegerType,true))
```

- **Feature Type Detection**
  - Numerical vs Categorical vs Ordinal
  - Rules and trained ML models

[Vraj Shah, Jonathan Lacanlale, Premanand Kumar, Kevin Yang, Arun Kumar: Towards Benchmarking Feature Type Inference for AutoML Platforms, **SIGMOD 2021**]

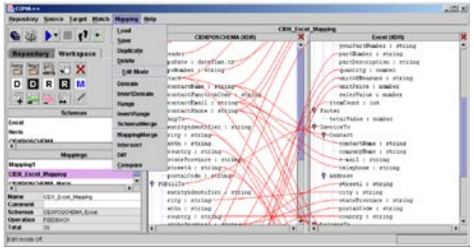
- **Semantic Type Detection**
  - Extract common feature types (e.g., location, date, rank, name)

[Madelon Hulsebos et al: Sherlock: A Deep Learning Approach to Semantic Data Type Detection. **KDD 2019**]

Schema Detection and Integration, cont.

- **Schema Matching**
  - Semi-automatic mapping of schema S1 to schema S2
    → **output:** schema correspondences
  - **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
    - Many-to-one: hospitals-residents / college-admission problems

[**Credit:** Erhard Rahm]



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

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

Examples (aka errors are everywhere)

- **DM SS'19 (Soccer World Cups)**

Commits on Apr 21, 2019

[MINOR] Fix 2002 match final scores, squad club
mboehm7 committed on Apr 21

[MINOR] Fixed mapping hansa rostock, and cons
mboehm7 committed on Apr 21

[MINOR] Fix null in match type (due to input file
mboehm7 committed on Apr 21

Commits on Apr 19, 2019

Fixed squads issues (resolved null clubs, non-unique clubs, player name)
mboehm7 committed on Apr 19

Commits on Apr 18, 2019

[MINOR] Fix squad club-country mapping, unique player names
mboehm7 committed on Apr 18

[MINOR] Fix squad club-country mapping, and spurious spaces
mboehm7 committed on Apr 18

- **DM WS'19/20 (Airports and Airlines)** ☺

Commits on Oct 7, 2019

New airports and flights datasets (cleaned) ...
OlgaOvcharenko authored and mboehm7 committed

Commits on Oct 30, 2019

Fix data issues: redundant plane types in routes
mboehm7 committed 14 days ago

```
- US,DFW,LIT,ER4;M83;M83
+ US,DFW,LIT,ER4;M83
```

Fix data issues: referential integrity country names
mboehm7 committed 14 days ago

```
- Oyo Ollombo Airport,Oyo,Congo (Brazzaville),O
- Beni Airport,Beni,Congo (Kinshasa),BNC,FZNP,0.575,2
+ Beni Airport,Beni,Democratic Republic of Congo,BNC,
```

Fix data issue: spelling united kingdom
mboehm7 committed 14 days ago

```
- RAF St Athan,4Q,STN,UNited Kingdom,N
+ RAF St Athan,4Q,STN,United Kingdom,N
```

- **DM SS'20 (DBLP Publications)**

Commits on Mar 13, 2020

Fix conf.csv header meta data (inconsistent number of
mboehm7 committed on Mar 14

Fix csv quoting (escaped quotes within fields)
mboehm7 committed on Mar 14

Fix publication titles (punctuation) and csv delimiters
mboehm7 committed on Mar 14

Updated dblp publications datasets (DB pubs only, clea
mboehm7 committed on Mar 13

Commits on Mar 14, 2020

Extract and clean city/country f
mboehm7 committed on Mar 14

Fix various columns by expecte
mboehm7 committed on Mar 14

Fix person/theses affiliation cou
mboehm7 committed on Mar 14

Fix conference title normalizati
mboehm7 committed on Mar 14

Fix normalization of conference
mboehm7 committed on Mar 14

Fix affiliation countries via robu
mboehm7 committed on Mar 14

Commits on Apr 6, 2020

Updated dblp publications rea
mboehm7 committed on Apr 6

Revert too aggressive matchin
mboehm7 committed on Apr 6

Additional cleaning of instituti
mboehm7 committed on Apr 6

Fix conference venues (consist
mboehm7 committed on Apr 6

Fix incorrect year in journal vol
mboehm7 committed on Apr 6

Fix handling of special characters beyond
mboehm7 committed on Apr 6

Commits on Apr 5, 2020

Initial deduplication of person affiliations and thesis schools
mboehm7 committed on Apr 5

Additional country cleaning (for person affiliations)
mboehm7 committed on Apr 5

Fix country name consistency (UK, Tunisia, The Netherlands, Autralia)
mboehm7 committed on Apr 5

Simplify dataset encoding (no quoting, no escaped quoates, etc)
mboehm7 committed on Apr 5

Fix head
mboe

Commits on Apr 22, 2020

Fix special character in french thesis
mboehm7 committed on Apr 22

Examples (aka errors are everywhere), cont.

- **DM SS'20, cont.**
  (DBLP Publications) → as a great, curated dataset

2013

[b1] Matei A. Zaharia:
An Architecture for and Fast and General Data Processing on Large Clusters.
University of California, Berkeley, USA, 2013

Wrong meta data from UC Berkeley

> Home > Persons

[–] **Person information**

- *affiliation (PhD 2018):* Improving clinical decisions using correspondences within and across electronic health records

2018

[b1] Jen Jian Gong:
Improving clinical decisions using correspondences within and across electronic health records. Massachusetts Institute of Technology, Cambridge, USA, 2018

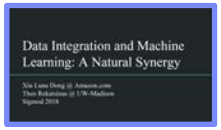
Misplaced data (wrong affiliation, MIT)

- **DM WS'20/21**
  (Movies and Actors)
- **DM SS'21**
  (Summer Olympics)

1) Best-effort automated cleaning
2) Reference impl data ingestion into relational schema + expected results of query processing
3) Decentralized validation (~600 students)

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 Validation

Sanity checks on expected shape
before training first model

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


Data Management Challenges in Production Machine Learning

(**Google Research**)

- **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

Data Validation, cont.

[Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Bießmann, Andreas Grafberger: Automating Large-Scale Data Quality Verification. **PVLDB 2018**]

(**Amazon Research**)

- **Constraints and Metrics for quality check UDFs**

constraint	arguments
dimension *completeness*	
`isComplete`	column
`hasCompleteness`	column, udf
dimension *consistency*	
`isUnique`	column
`hasUniqueness`	column, udf
`hasDistinctness`	column, udf
`isInRange`	column, value range
`hasConsistentType`	column
`isNonNegative`	column
`isLessThan`	column pair
`satisfies`	predicate
`satisfiesIf`	predicate pair
`hasPredictability`	column, column(s), udf
statistics (can be used to verify dimension *consistenc*	
`hasSize`	udf
`hasTypeConsistency`	column, udf
`hasCountDistinct`	column
`hasApproxCountDistinct`	column, udf
`hasMin`	column, udf
`hasMax`	column, udf
`hasMean`	column, udf
`hasStandardDeviation`	column, udf
`hasApproxQuantile`	column, quantile, udf
`hasEntropy`	column, udf
`hasMutualInformation`	column pair, udf
`hasHistogramValues`	column, udf
`hasCorrelation`	column pair, udf
time	
`hasNoAnomalies`	metric, detector

metric
dimension *completeness*
`Completeness`
dimension *consistency*
`Size`
`Compliance`
`Uniqueness`
`Distinctness`
`ValueRange`
`DataType`
`Predictability`
statistics (can be used to
`Minimum`
`Maximum`
`Mean`
`StandardDeviation`
`CountDistinct`
`ApproxCountDistinct`
`ApproxQuantile`
`Correlation`
`Entropy`
`Histogram`
`MutualInformation`

Organizational Lesson: benefit of shared vocabulary/procedures

Technical Lesson: fast/scalable; reduce manual and ad-hoc analysis

- **Approach**
  - **#1** Quality checks on basic metrics, computed in **Apache Spark**
  - **#2** **Incremental maintenance** of metrics and quality checks

Data Validation, cont.

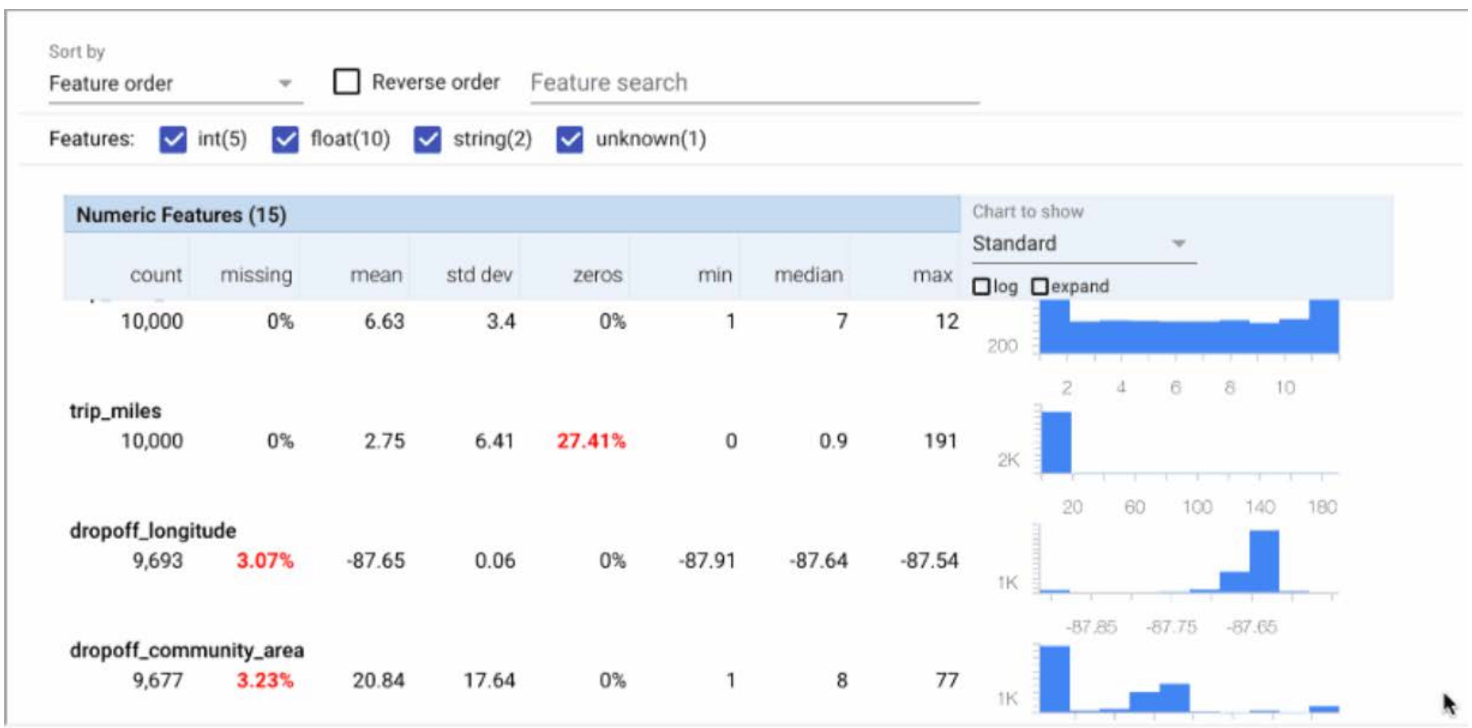
[Mike Dreves; Gene Huang; Zhuo Peng; Neoklis Polyzotis; Evan Rosen; Paul Suganthan: From Data to Models and Back. **DEEM 2020**]

[Eric Breck, Neoklis Polyzotis, Sudip Roy, Steven Whang, Martin Zinkevich: Data Validation for Machine Learning. **MLSys 2019**]

[Emily Caveness et al: TensorFlow Data Validation: Data Analysis and Validation in Continuous ML Pipelines. **SIGMOD 2020**]

- **TensorFlow Data Validation (TFDV)**
  - Library or TFX components
  - Stats computation, schema extraction, validation checks, anomaly detection

(**Google**)



Feature Transformations 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**
  - Bring multi-modal data and features into numeric representation
  - Use domain expertise to expose predictive features to ML model training
- **Excursus: Representation Learning**
  - Neural networks can be viewed as combined representation learning and model training (pros and cons: learned, repeatable)
  - 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)

City
San Jose
New York
San Francisco
Seattle
New York
Boston
San Francisco
Los Angeles
Seattle

for k = 5:

```
1993955031 % 5 → 1
1382994575 % 5 → 0
1540367136 % 5 → 1
-661909336 % 5 → 1
1993955031 % 5 → 1
1995575789 % 5 → 4
1540367136 % 5 → 1
-425347233 % 5 → 3
-661909336 % 5 → 1
```

Efficient, but **collisions**

City
1
0
1
1
1
4
1
3
1

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 (see also Dummy Coding)

- **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

Hybrid Feature Transformations

- **Combinations**
  - Different encoders for different columns
  - Binning + one-hot encoding
  - Recoding + one-hot encoding
  - Feature hashing + one-hot encoding

How to parallelize effectively?
Bachelor Thesis Lukas Erlbacher

- **Top-K Recoding/Feature Hashing**
  - Recoding top-k most frequent values (no collisions in frequent values)
  - Feature Hashing for others (collisions, but bounded #)
  - "Vocabulary encoding"

	City	Count
1	New York	8,336,817
2	San Jose	1,026,350
3	San Francisco	883,305
	Seattle	704,352
	Boston	684,379
	…	…
	Graz	291,072

Feature Hashing k=2

[Doris Xin et al: Production Machine Learning Pipelines: Empirical Analysis and Optimization Opportunities, **SIGMOD 2021**]

Derived Features

- **#1 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));
```

- **#2 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);
```

- **#3 Windowing**
  - Tumbling or sliding window over time series
  - Compute aggregates or existence of events

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)

    Who did **what** to **whom** at **where**?

- **Bag of Words (BOW) 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

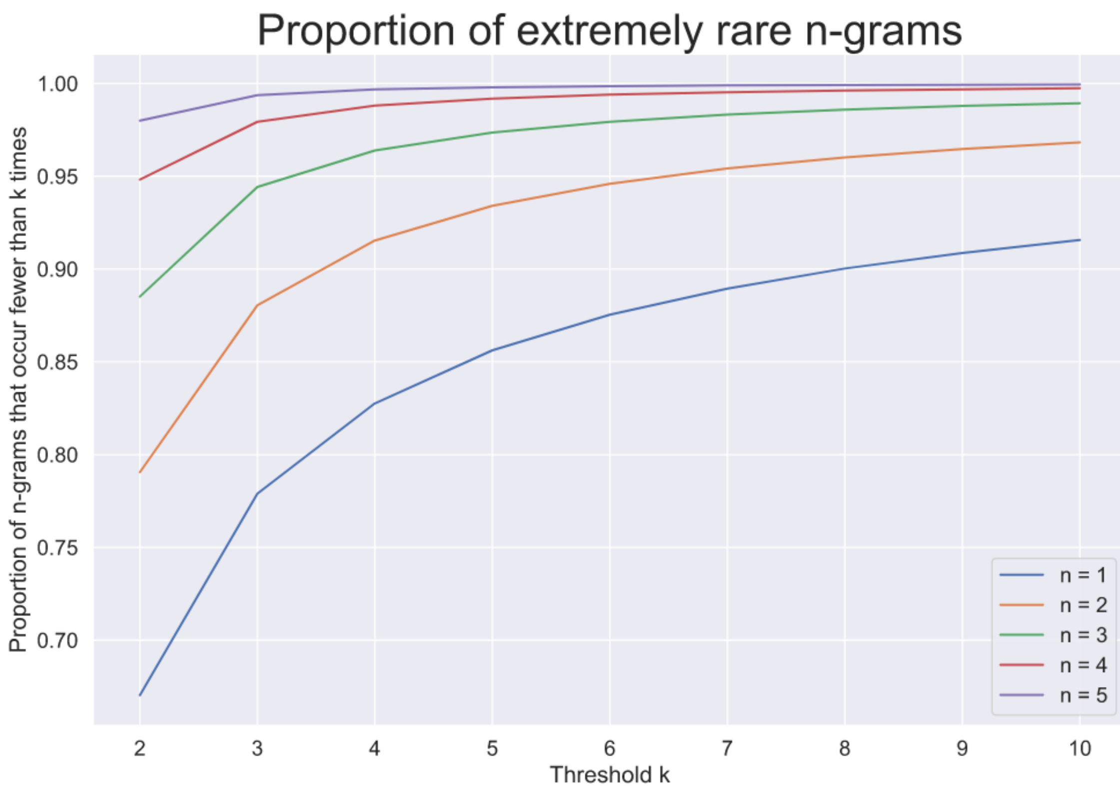
  - **Bi-grams:** bag-of-words for 2-sequences of words (order preserving)
  - **N-grams:** generalization of bi-grams to arbitrary-length sequences

NLP Features, cont.

[John Hallman: Efficient Featurization of Common N-grams via Dynamic Programming. https://sisudata.com/blog/efficient-featurization-common-ngrams-via-dynamic-programming, 2021]

- **Common N-Grams**
  - Prune n-grams that appear <5 times, → 99.3% reduction
  - **Lattice-based pruning** (Apriori monotonicity property)

- **Example**
  - Amazon Reviews Dataset
  - 67% of words appear just once

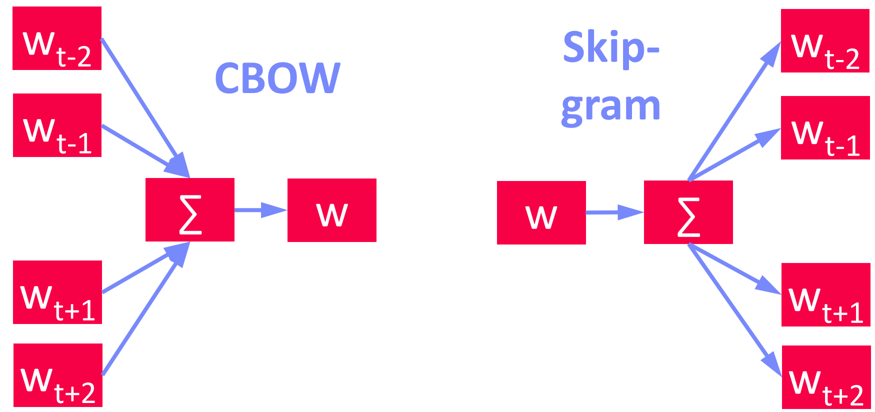


NLP Features, cont.

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

github.com/dav/word2vec

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



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

- **Follow-up Work**
  - Often pre-trained word embeddings; fine-tuning if necessary for task/domain
  - Various extensions/advancements: **Sentence2Vec**, **Doc2Vec**, **Node2Vec**
  - **BERT**, **RoBERTa**, **ALBERT**, **StructBERT**

[Jacob Devlin et al. : **BERT:** Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT (1) 2019]

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

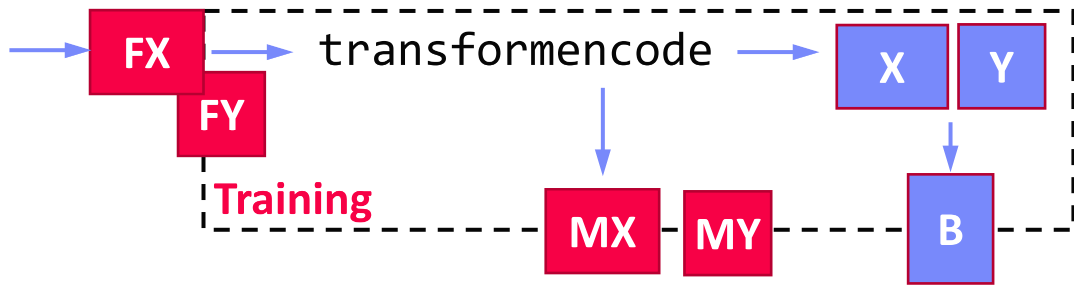
- **Example Pipeline**

```
// 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

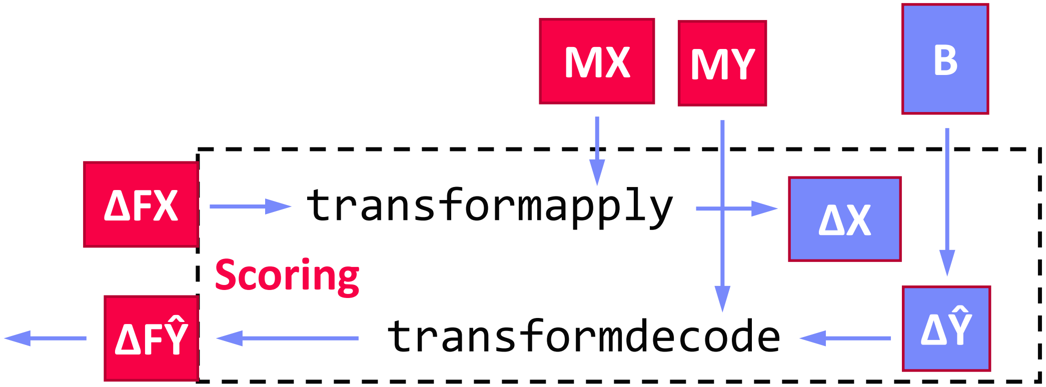
- **Feature Transformation during Training**




```
# 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 Preparation and Cleaning

Standardization/Normalization

- **#1 Standardization**
  - Centering and scaling to mean 0 and variance 1
  - **Ensures well-behaved training** (and distance computation)
  - **Densifying operation** / **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 (Min-Max) Normalization**
  - Rescale values into common range [0,1]
  - **Avoid bias to large-scale features**
  - Does not handle outliers

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

Recommended Reading

[Andreas C. Mueller: Preprocessing and Feature Transformations, **Applied ML Lecture 2020**, https://www.youtube.com/watch?v=XpOBSaktb6s]

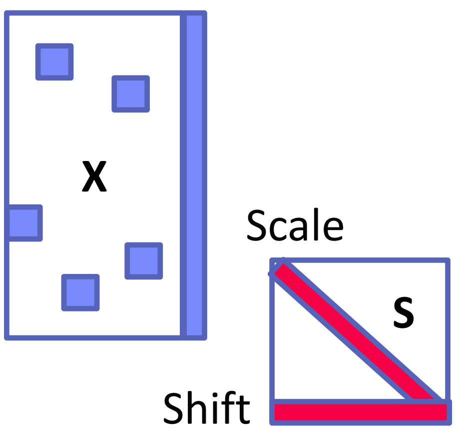
Standardization/Normalization, cont.

Apache SystemML™

[**Credit:**
Alexandre (Sasha)
V. Evfimievski]

- **#3 Deferred Standardization**
  - Avoid densifying dataset upfront by pushing standardization into inner loop iterations
  - Let **matrix-multiplication chain optimization** + rewrites do the rest
- **Example GLM/lmCG**

Input w/ column of ones (intercept)



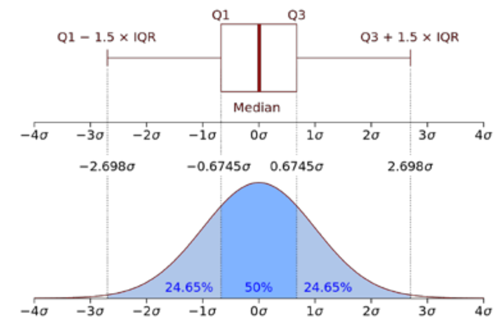

```
# operation w/ early standardized X
q = t(X) %*% diag(w) %*% X %*% B;
```

Substitute X with X %*% S

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

```
q = t(S) %*% (t(X) %*% (diag(w)
  %*% (X %*% (S %*% B))));
```

Winsorizing and Trimming



[**Credit:** https://en.wikipedia.org]

- **Recap: Quantiles**
  - Quantile Q_p w/ $p \in (0,1)$ defined as $P[X \leq x] = p$

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

```
# compute quantiles for lower and upper
ql = quantile(X, 0.05);
qu = quantile(X, 0.95);

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

SystemDS:
`winsorize()`
`outlier()`
`outlierByIQR()`
`outlierBySd()`

- **Truncation/Trimming**
  - **Remove** tails of data distribution at user-specified threshold

```
# remove values outside [ql,qu]
I = X < qu | X > ql;
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(xor(I,op), colMaxs(X), colMins(X));
```

Constraints and Outliers

Route **Planes**

(Airline, From, To)

```
- US,DFW,LIT,ER4;M83;M83
+ US,DFW,LIT,ER4;M83
```

- **(Semi-)Automatic Approach: Expectations!**
  - PK → Values must be unique and defined (not null)
  - Exact PK-FK → Inclusion dependencies
  - Noisy PK-FK → Robust inclusion dependencies |R[X]∈S[Y]| / |R[X]| > δ
  - Semantics of attributes → Value ranges / # distinct values

    Age=9999?
  - Invariant to capitalization

    Patterns → regular expressions

```
- RAF St Athan,4Q,STN,UNited Kingdom,N
+ RAF St Athan,4Q,STN,United Kingdom,N
```

2019-11-15 vs Nov 15, 2019

- **Formal Constraints**
  - Functional dependencies (FD), conditional FDs (CFD), metric dependencies
  - Inclusion dependencies, matching dependencies
  - Denial constraints

$$\forall t_\alpha t_\beta \in R: \neg(t_\alpha.Role = t_\beta.Role \wedge t_\alpha.City = 'NYC' \wedge t_\beta.City \neq 'NYC' \wedge t_\alpha.Salary < t_\beta.Salary)$$

- **Outlier Terminology**
  - **Outlier Detection:** detect and remove unwanted data points
  - **Anomaly Detection:** detect and extract rare/unusual/interesting events

Outliers and Outlier Detection

- **Types of Outliers**

  [Varun Chandola, Arindam Banerjee, Vipin Kumar: Anomaly detection: A survey. **ACM Comput. Surv. 2009**]

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

- **Types of Outlier Detection**

  [Victoria J. Hodge, Jim Austin: A Survey of Outlier Detection Methodologies. **Artif. Intell. Rev. 2004**]

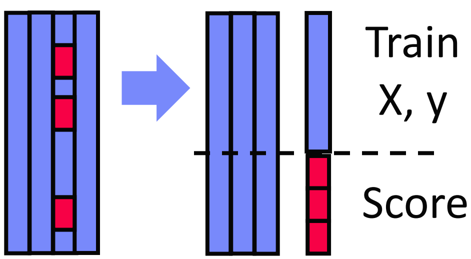
  

  - **Type 1 Unsupervised:** No prior knowledge of data, similar to unsupervised **clustering** → **expectations:** distance, # errors
  - **Type 2 Supervised:** Labeled normal and abnormal data, similar to supervised **classification**
  - **Type 3 Normal Model:** Represent normal behavior, similar to **pattern recognition** → **expectations:** rules/constraints

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**
  - MCAR **Continuous variables:** replace by **mean**
  - **Categorical variables:** replace by **median** or **mode**
- **Iterative Algorithms** (**chained-equation imputation**)
  - MAR Train ML model to predict missing information (feature k → label, split data into observed/missing)
  - Noise reduction: feature subsets + averaging
- **Dynamic Imputation**
  - Data exploration w/ on-the-fly imputation
  - Optimal placement of imputation operations

SystemDS:
`mice()`



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

Summary and Q&A

[Andreas C. Mueller: Preprocessing and Feature Transformations, **Applied ML Lecture 2020**]

- **Data Acquisition, Integration, and Validation**
- **Feature Transformations and Feature Engineering**
- **Data Preparation and Cleaning**

"Coming up with features is difficult, time-consuming, requires expert knowledge. "Applied machine learning" is basically feature engineering"
– Andrew Ng

- **Next Lectures**
  - **10 Model Selection and Management** [May 28]
    - Incl Data Augmentation
  - **11 Model Debugging Techniques** [Jun 04]
  - **12 Model Serving Systems and Techniques** [Jun 11]