Data Validation and Data Cleaning

Sebastian Schelter
Moore-Sloan Fellow, Center for Data Science, New York University

Overview

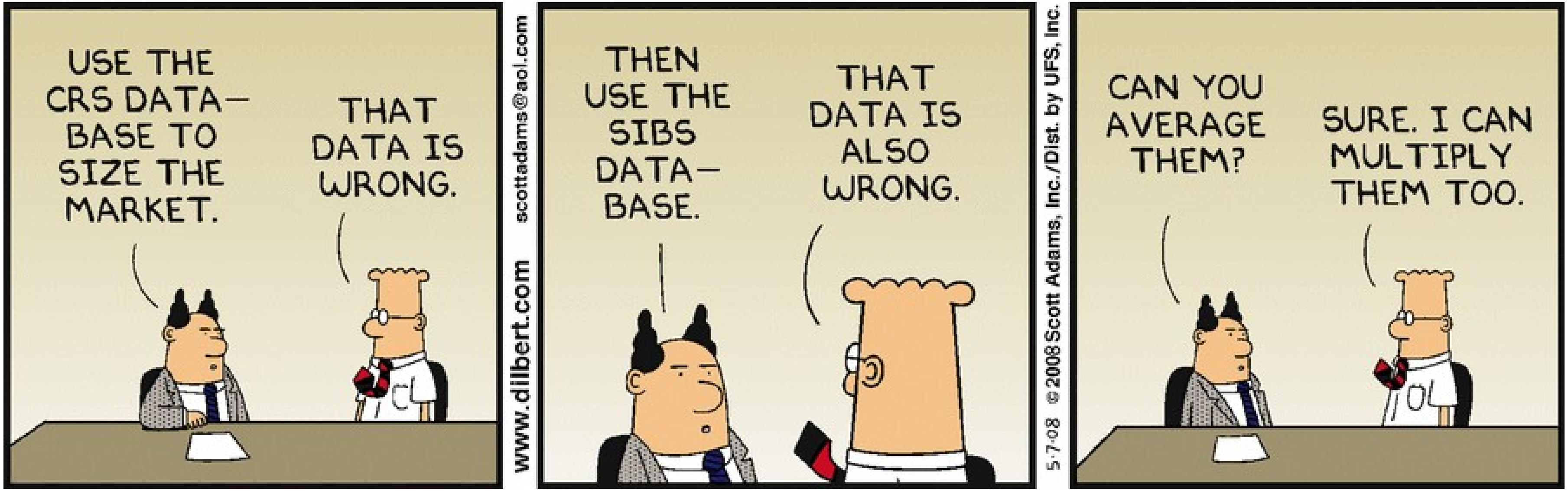
Center for Data Science
NYU

- Introduction & Overview
- Exemplary Error Detection and Data Cleaning Techniques
  - Quantitative Data: Robust Univariate Outlier Detection
  - Categorical Data: String Normalization
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - Missing Value Imputation using Supervised Learning
- Summary & References

Overview

Center for Data Science
NYU

- **Introduction & Overview**
- Exemplary Error Detection and Data Cleaning Techniques
  - Quantitative Data: Robust Univariate Outlier Detection
  - Categorical Data: String Normalization
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - Missing Value Imputation using Supervised Learning
- Summary & References



https://dilbert.com/strip/2008-05-07

Why is Data Quality Important?

Center for Data Science NYU

- **Impact on organisational decisions**
  - missing or incorrect data can result in wrong decision making
- **Legal obligations in certain business scenarios**
  - plug type information required for selling electric devices in EU

- **Impact on machine learning models**
  - Cleaner data can greatly improve model performance
- **Potential for causing biased decisions in ML-based systems**
  - Not well understood, area of active research

- **Operational stability: missing and inconsistent data can cause havoc in production systems**
  - Crashes (e.g., due to “NullPointerExceptions” for missing attributes)
  - Wrong predictions (e.g., change of scale in attributes)

Data: Academia vs the Real-World

- **Academic datasets**
  - Static
  - Often down-sampled, cleaned and aggregated before publication
  - Attributes typically well understood
  - Most of time: size convenient for processing on desktop machines
  - Example: UCI ML datasets

- **Real-world data**
  - Constantly changing
  - Often hundreds of attributes
  - Data originates from multiple sources / people / teams / systems
  - Several potentially inconsistent copies
  - Often too large to conveniently handle on a desktop machine
  - Often difficult to access (e.g., data compressed and partitioned in a distributed filesystem)

Changes in Data Collection Strategies

Center for Data Science NYU

- **Pre-Internet Era**
  - Data collected in transactional, relational databases
  - “Extract-Transform-Load” export to data warehouses for analysis (relational databases optimized for analytical workloads)
  - Modelling of the data and its schema before collection

- **Internet Era: “Collect first, analyze later”**
  - Advent of the internet gave rise to vast amount of semi-structured data
  - New data stores established (key-value stores, document databases, data lakes)
    - Scale to very large datasets
    - Relaxed consistency (e.g. no distributed transactions)
    - Enforce fewer modelling decisions at collection time
    - “Schema-on-Read”: application has to determine how to interpret data
  - Economic incentives
    - Decreasing storage costs
    - Data becomes valuable as input to ML-based applications

Sources of Error in Data

Center for Data Science NYU

- **Data entry errors**
  - Typos in forms
  - Different spellings for the same real-world entity (e.g., addresses, names)
- **Measurement errors**
  - Outside interference in measurement process
  - Placement of sensors
- **Distillation errors**
  - Editorial bias in data summaries
  - Domain-specific statistical analyses not understood by database manager
- **Data integration errors**
  - Resolution of inconsistencies w.r.t. duplicate entries
  - Unification of units, measurement periods

Hellerstein: "Quantitative data cleaning for large databases.", 2008

Dimensions of Data Quality

- **Completeness**
  - Degree to which data required to describe a real-world object is available
- **Consistency: Intra-relation constraints (range of admissible values)**
  - Specific data type, interval for a numerical column, set of values for a categorical column
- **Consistency: Inter-relation constraints**
  - Validity of references to other data entries (e.g., “foreign keys” in databases)
- **Syntactic and semantic accuracy**
  - Syntactic accuracy compares the representation of a value with a corresponding definition domain
    - E.g.: value *blue* for `color` attribute syntatically accurate for *red* product in online shop
  - Semantic accuracy compares a value with its real-world representation
    - E.g.: value *XL* for `color` attribute neither syntactically nor semantically accurate for this product

Batini, Carlo, et al. "Methodologies for data quality assessment and improvement."
ACM Computing Surveys 41.3 (2009): 16.

Approaches to Improve Data Quality

- **Data entry interface design**
  - Enforce integrity constraints (e.g., constraints on numeric values, referential integrity)
  - Can force users to “invent” dirty data
- **Organisational management**
  - Streamlining of processes for data collection and analysis
  - Capturing of lineage and metadata
- **Automated data auditing and data cleaning**
  - Application of automated techniques to identify and rectify data errors
- **Exploratory data analysis and data cleaning**
  - Human-in-the-loop approach necessary most of the time
  - Interaction between data visualisation and data cleaning
  - Iterative process

Approaches to Improve Data Quality

Center for Data Science
NYU

- **Data entry interface design**
  - Enforce integrity constraints (e.g., constraints on numeric values, referential integrity)
  - Can force users to “invent” dirty data
- **Organisational management**
  - Streamlining of processes for data collection and analysis
  - Capturing of lineage and metadata
- **Automated data auditing and data cleaning**
  - Application of automated techniques to identify and rectify data errors
- **Exploratory data analysis and data cleaning**
  - Human-in-the-loop approach necessary most of the time
  - Interaction between data visualisation and data cleaning
  - Iterative process

Data Cleaning: Types and Techniques

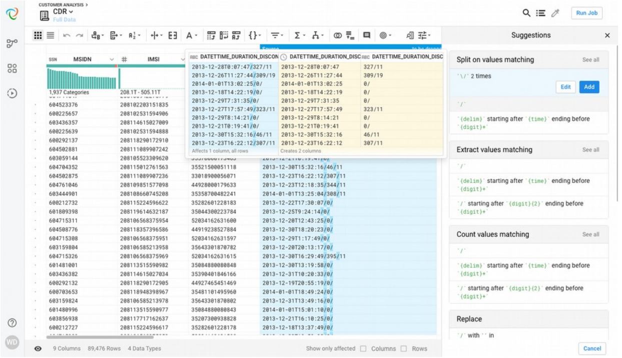
Center for Data Science
NYU

- **Quantitative data**
  - Integers or floating point numbers in different shapes (sets, tensors, time series)
  - Challenges: unit conversion (especially for volatile units like currency)
  - Foundation of cleaning techniques: **outlier detection**
- **Categorical data**
  - Names or codes to assign data into groups, no ordering or distance defined
  - Common problem: misspelling upon data entry
  - Foundation of cleaning techniques: **normalization / deduplication**
- **Postal addresses**
  - Special case of categorical data, typically entered as free text
  - Major challenge: **deduplication**
- **Identifiers / Keys**
  - Unique identifiers for data objects (e.g., product codes, phone numbers, SSNs)
  - Challenge: detect reuse of identifier across distinct objects
  - Challenge: Ensure **referential integrity**

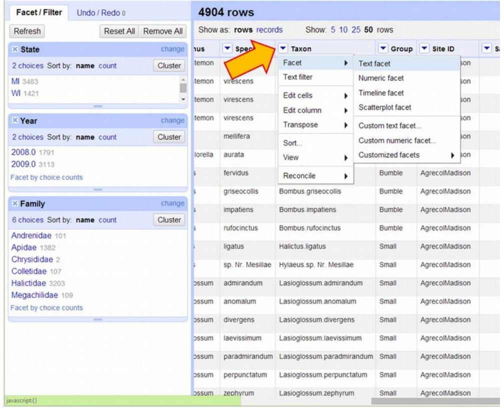
The Need for the “Human in the Loop“

- **Unrealistic assumptions about error detection in academia:**
  - Existence of error detecting rules assumed:
    Integrity Constraints, Functional Dependencies,
    Conditional Functional Dependencies, Denial Constraints
  - Often focus on most efficient and accurate way to
    apply cleaning steps according to rules

- **In practice: error detection already a very hard problem**
  - Consequence: Human-in-the-loop solutions required
  - Data exploration and visualisation crucial
  - Iterative cleaning
  - Popular implementations: Open Refine, Trifacta



https://trifacta.com



https://practicaldatamanagement.files.wordpress.com/
2014/05/glbrc-bees-openrefine2.jpg

Overview

Center for
Data Science
NYU

- Introduction & Overview
- **Exemplary Error Detection and Data Cleaning Techniques**
  - **Quantitative Data: Robust Univariate Outlier Detection**
  - Categorical Data: String Normalization
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - Missing Value Imputation using Supervised Learning
- Summary & References

Robust Univariate Outlier Detection

- **Univariate analysis**
  - Simple approach: investigate the set of values of a single attribute of our dataset
  - Statistical perspective: values considered to be a sample of some data generating process
- **Center & Dispersion**
  - Set of values has a *center* that defines what is “average”
  - Set of values has a *dispersion* that defines what is “far from average”
- **Outlier detection**
  - Assumption: erroneous values “far away” from typical values in the set
  - Approach: identify outliers using statistical techniques
  - Problem: How to reliably compute them when the data is dirty / erroneous?

Hellerstein: "Quantitative data cleaning for large databases", 2008

Example: Age Data

- **Set of age values of employees in a company:**

  12 13 14 21 22 26 33 35 36 37 39 42 45 47 54 57 61 68 450

Example: Age Data

- **Set of age values of employees in a company:**

12 13 14 21 22 26 33 35 36 37 39 42 45 47 54 57 61 68 450

minors

impossible age

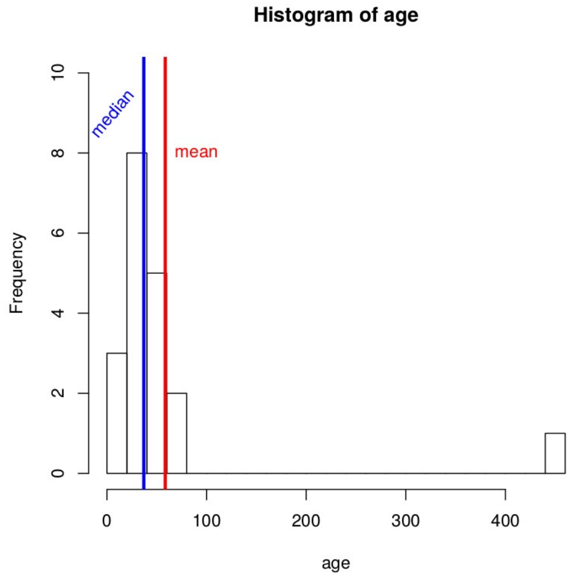
Example: Age Data

Center for Data Science
NYU

- **Set of age values of employees in a company:**

  12 13 14 21 22 26 33 35 36 37 39 42 45 47 54 57 61 68 450

- **Potential approach:**
  - Assume normal distribution of age values
  - Compute mean and standard deviation
  - Flag values more 2 standard devations away from mean
  - Interval is [96 – 2 * 59, 9 + 2 * 59] = [-22, 214]
  - Misses first three values!



- **Problem: “Masking”**
  - Magnitude of one outlier shifts center and dispersion
  - “Masks” other outliers

Robust Statistics

- **Idea: consider effect of corrupted data values on distribution**
  - Estimators should be robust to such corruptions
  - *Breakdown point*: threshold of corrupt values before estimator produces arbitrarily erroneous results

- **Robust Centers**
  - *Median*: value for which half of the dataset is smaller (affected by position not magnitude of outliers)
  - *Trimmed Mean*: remove k% of highest and lowest values, compute mean from rest

- **Robust Dispersion**
  - *Mean Absolute Deviation*: robust analogy to standard deviation
  - Measures median distance of all values from the sample median

Example: Age Data

Center for Data Science
NYU

- **Set of age values of employees in a company:**

  12 13 14 21 22 26 33 35 36 37 39 42 45 47 54 57 61 68 450

- **Cleaned set of age values:**

  21 22 22 23 24 26 33 35 36 37 39 42 45 45 47 54 57 61 68

- **Robust centers in example closer to center on clean data:**
  - Median 37 (dirty) 39 (clean)
  - Mean ~96 (dirty) ~40 (clean)
  - 10%-Trimmed mean ~39 (dirty)
- **Robust dispersion provides better interval on dirty data:**
  - 1 standard deviation [37, 155] (includes six non-outliers)
  - 1.48 MAD [16, 61] (includes one non-outlier)

Overview

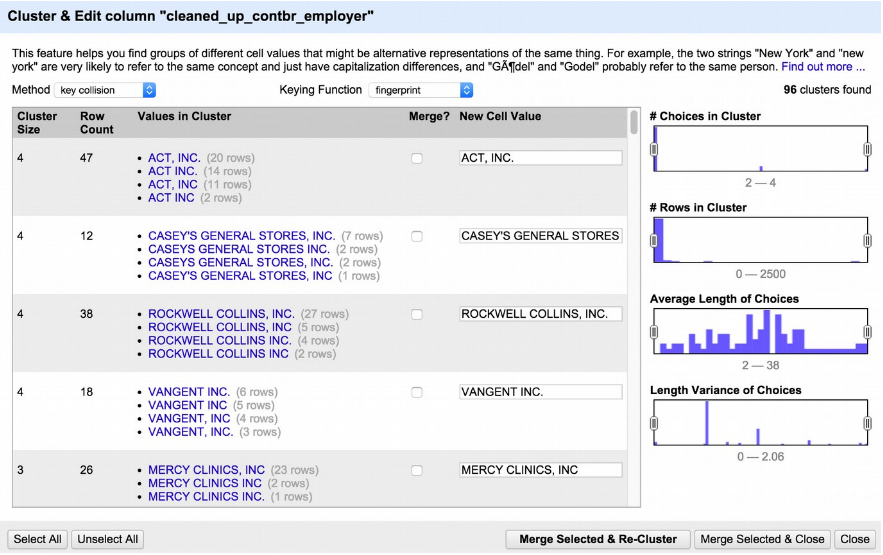
- Introduction & Overview
- **Exemplary Error Detection and Data Cleaning Techniques**
  - Quantitative Data: Robust Univariate Outlier Detection
  - **Categorical Data: String Normalization**
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - Missing Value Imputation using Supervised Learning
- Summary & References

Normalization of String Data

- **Free-text entry of categorical attributes very error-prone:**
  - Different spellings (Jérôme vs Jerome)
  - Different punctuation (ACME Inc. vs ACME, Inc)
  - Typos (Alice → Ailce)
  - Misunderstandings (Rupert → Robert)

- **Normalization with simple heuristic clustering algorithm:**
  - Keying function *k*
  - Compute key *k(s)* per string *s*
  - group pairs *(s, k(s))* by *k(s)* and count pairs
  - Automatic: Replace all strings in a group with string with highest cardinality
  - Human-in-the-Loop: shows groups and statistics to user

- Extensively used in **OpenRefine**



http://www.padjo.org/files/tutorials/open-refine/fingerprint-cluster-popup.png

String Normalization

- **“Fingerprint keying”:** remove punctuation and case sensitivity
  - remove whitespace around the string
  - lowercase the string
  - remove all punctuation and control characters
  - find ASCII equivalents of characters
  - tokenize (split by whitespace)
  - order fragments and deduplicate them

```
ACT,INC. → act inc
ACT INC  → act inc
ACT,Inc  → act inc
Act Inc  → act inc
```

String Normalization

Center for Data Science NYU

- **“SOUNDEX”:** Algorithm for **phonetic indexing of English strings**
  - Save the first letter.
  - Remove all occurrences of a, e, i, o, u, y, h, w
  - Replace all consonants (include the first letter) with digits as follows:
    b, f, p, v → 1 ; c, g, j, k, q, s, x, z → 2 ; d, t → 3, l → 4 ; m, n → 5 ; r → 6
  - Replace all adjacent same digits with one digit.
  - If the saved letter's digit is the same as the resulting first digit, remove the digit (keep the letter).
  - Append 3 zeros if result contains less than 3 digits. Remove all except first letter and 3 digits after it

```
Robert → R163
Rupert → R163
```

Knuth: “The art of computer programming: sorting and searching”, Vol. 3., 1997

Overview

Center for Data Science
NYU

- Introduction & Overview
- **Exemplary Error Detection and Data Cleaning Techniques**
  - Quantitative Data: Robust Univariate Outlier Detection
  - Categorical Data: String Normalization
  - **Candidate Key Detection at Scale with Hyperloglog Sketches**
  - Missing Value Imputation using Supervised Learning
- Summary & References

Frequency Statistics of Categorical Data

Center for Data Science NYU

- **In some cases: frequency of values more important than actual values**
  - Especially for categorical data attributes (where values have no ordering and no distance)
  - E.g. “species code” in a dataset of animal sightings

- **Application: Discovery of “Candidate Keys”**
  - Key: attribute or combination of attributes that uniquely identifies a tuple in a relation
  - In clean data:
    - Frequency of every value of the candidate key attribute should be 1
    - Number of distinct values equals number of tuples
  - Both conditions can be violated in case of dirty data

Heuristics for Discovering "Dirty Keys"

Center for Data Science NYU

- **Idea:** discover attributes intended to be used as keys in dirty data
- **"Unique Row Ratio"**
  - Ratio of distinct values of an attribute to the number of tuples
  - Attribute is potential key if heuristic close to 1.0
  - Problem: "frequency outliers": small number of values with very high frequency often caused by UIs forcing users to "invent" common "dummy values" like 00000 or 12345
- **"Unique Value Ratio"**
  - Ratio of unique values to number of distinct values
  - Attribute is potential key if heuristic close to 1.0
  - More robust against frequency outliers
- **Problem of both approaches**: high memory requirements during computation

item_id
1
3
000
4
5
6
8
10
000
000

Cardinality Estimation with HLL Sketches

- **Problem: exact counting requires memory linear in the number of distinct elements**
  - E.g., to maintain a hashtable with values and counts
  - Does not scale to large or unbounded datasets

- HyperLogLog (HLL) Sketch
  - **“sketch” data structure:** approximate counting with drastically less memory
  - Uses **randomization to approximate the cardinality of a multiset**

Flajolet "Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm.", Discrete Mathematics and Theoretical Computer Science, (2007).
Heule: "HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm.", EDBT, (2013).

HyperLogLog: Idea

- Apply **hash function *h*** to every element to be counted (*h* must produce uniformely distributed outputs)

- Keep track of the **maximum number of leading zeros** of the bit representations of all observed hash values

- Intuitively: **hash values with more leading zeros are less likely and indicate a larger cardinality**

- If bit pattern $0^{q-1}1$ is observed at the beginning of a hash value, estimate size of multiset as 2^q

```
h(“hello”) → 10011
h(“world”) → 11011
h(“hello”) → 10011
h(“alice”) → 00101
h(“world”) → 11011
```

HyperLogLog: Details

- Algorithm applies several **techniques to reduce variability of these measurements**
  - Input stream divided into m substreams S_i with $m = 2^p$
  - p number of bits of hash values to store
  - array of registers M, $M[i]$ stores max number of leading zeros + 1 from stream S_i
  - Final estimate uses bias-corrected harmonic mean of the estimations on the substreams

$$\alpha_m m^2 \sum_{i=1}^{m} 2^{-M[i]}$$

- Extremely **powerful in practice**
  - **Low memory requirements**: e.g., SparkSQL implementation uses less than 3.5 KB for the registers, works on billions of elements
  - **Easy to parallelize** as registers can be cheaply merged via max function
  - Allows to run **cardinality estimation** on multiple columns of huge tables **with a single scan**

- Basis of key detection in data validation library “deequ”

  https://github.com/awslabs/deequ

Overview

- Introduction & Overview
- **Exemplary Error Detection and Data Cleaning Techniques**
  - Quantitative Data: Robust Univariate Outlier Detection
  - Categorical Data: String Normalization
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - **Missing Value Imputation**
- Summary & References

Missing Value Imputation

- Missing data is a central data quality problem
- Missing for various reasons
  - **Missing Completely at Random** (MCAR)
  - **Missing at Random** (MAR)
  - **Not Missing at Random** (NMAR)

- Various ways to handle missing data for ML applications
  - **Complete-case analysis** (remove examples with missing attributes)
  - Add **placeholder symbol** for missing values
  - **Impute missing values**
    - Often implemented with techniques from popular ML libraries, like mean and mode imputation
    - ML: supervised learning for missing value imputation

Imputation of Categorical Data (1)

- Assume tabular data
- Want to impute missing values in a column with categorical data

- Idea: apply techniques from supervised learning

- Example: product catalog, colors missing
  p(color=yellow | other columns, imputation model)

- Treat imputation problem as multi-class classification problem

Product Type	Description	Size	Color
Shoe	Ideal for running …	12UK	Black
SDCards	Best SDCard ever …	8GB	Blue
Dress	This **yellow** dress …	M	**?**

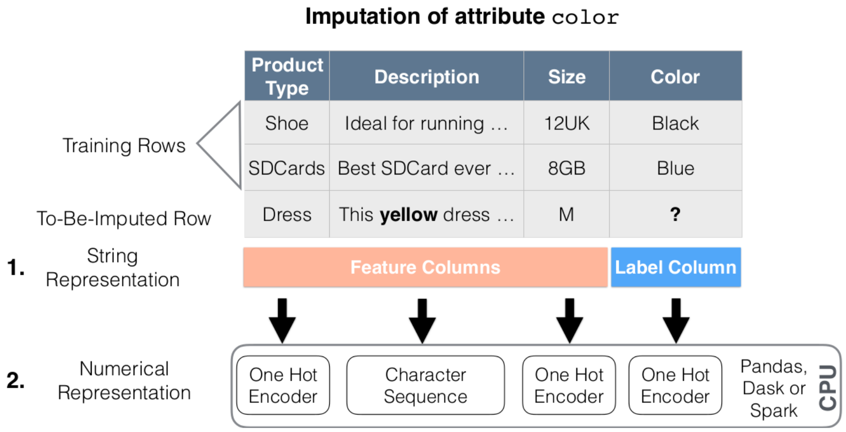
Biessmann, et al. "Deep Learning for Missing Value Imputation in Tables with Non-Numerical Data." ACM CIKM (2018).

Imputation of Categorical Data (2)

NYU Center for Data Science

- Must encode table data from feature columns to a numerical representation

- Standard encoding techniques
  - **"One-hot" encoding of categorical columns** (zero vector with as many dimensions as distinct values, 1 in corresponding dimensions)
  - **Standardisation of numerical columns** (substract mean, divide by standard deviation)
  - **Character sequences for textual columns**



Imputation of Categorical Data (3)

- **Train neural network to predict likelihood of values to impute**

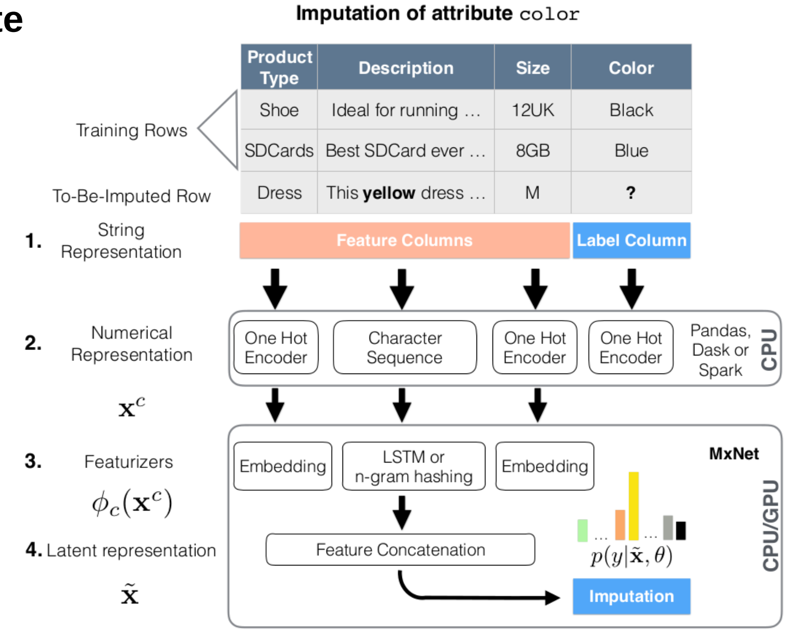
$$p(y|\tilde{\mathbf{x}}, \boldsymbol{\theta}) = \mathtt{softmax}\,[\mathbf{W}\tilde{\mathbf{x}} + \mathbf{b}]$$

- **Concatenation of featurizers into single feature vector**

$$\tilde{\mathbf{x}} = [\phi_1(\mathbf{x}^1), \phi_2(\mathbf{x}^2), \ldots, \phi_C(\mathbf{x}^C)] \in \mathbb{R}^D$$

- Standard featurization techniques
  - Embeddings for one-hot encoded categories
  - Hashed n-grams or LSTMs for character sequences

- Open source implementation “datawig” available at https://github.com/awslabs/datawig



Overview

Center for Data Science
NYU

- Introduction & Overview
- Exemplary Error Detection and Data Cleaning Techniques
  - Quantitative Data: Robust Univariate Outlier Detection
  - Categorical Data: String Normalization
  - Candidate Key Detection at Scale with Hyperloglog Sketches
  - Missing Value Imputation
- **Summary & References**

Summary

- Data quality important for: decision making, conforming to legal obligations, improving the performance of ML models, operation of data processing systems
- Real-world data is always messy and difficult to handle

- Dimensions of data quality: completeness, consistency, syntactic & semantic accuracy

- Data cleaning techniques
  - Quantitative data: outlier detection
  - Categorical data: normalisation / deduplication
  - Postal addresses: deduplication
  - Identifiers / keys: ensuring referential integrity

- Error detection is already a very hard problem: typically requires iterative cleaning, visualisation and a human-in-the-loop

References

Center for Data Science NYU

- Hellerstein, Joseph M. "Quantitative data cleaning for large databases." United Nations Economic Commission for Europe (UNECE) (2008).
- Batini, Carlo, et al. "Methodologies for data quality assessment and improvement." ACM Computing Surveys 41.3 (2009): 16.
- Geerts, Floris, et al. "The LLUNATIC data-cleaning framework." PLVDB 6.9 (2013): 625-636.
- Chu, Xu, et al. "Discovering denial constraints." PVLDB 6.13 (2013): 1498-1509.
- Flajolet, Philippe, et al. "Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm.", Discrete Mathematics and Theoretical Computer Science, (2007).
- Heule, Stefan et al. "HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm.", EDBT, (2013).
- Schelter, Sebastian, et al. "Automating large-scale data quality verification." PVLDB 11.12 (2018): 1781-1794.
- Biessmann, Felix, et al. "Deep Learning for Missing Value Imputation in Tables with Non-Numerical Data." ACM CIKM (2018).