

# DATA QUALITY

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

Christian Kaestner

Required reading:

- Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021, May). “[Everyone wants to do the model work, not the data work](#)”: Data Cascades in High-Stakes AI. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-15).

Recommended 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)

# ADMINISTRATIVA: MIDTERM

- Midterm in 1 week
- During lecture, 80 min, here
- Answer questions related to a given scenario
- All lecture content, reading, recitations in scope -- focus on topics you had opportunity to practice
- No electronics, can bring 6 pages notes on paper (handwritten or typed, both sides)
- Old midterms online, see course webpage

# ADMINISTRATIVA: HOMEWORK I3

- Open ended: Try a tool and write a blog post about it
- Any tool related to building ML-enabled systems
  - Except: No pure ML frameworks
  - ML pipelines, data engineering, operations, ...
  - Open source, academic, or commercial; local or cloud
  - Also look for competitors of tools of interest
- Claim tool in Spreadsheet, first come first serve
- 1 week assignment (despite due in 3 weeks)
- Past tools: [Algorithmia](#), [Amazon Elastic MapReduce](#), [Apache Flink](#), [Azure ML](#), [Dask](#), [Databricks](#), [DataRobot](#), [Google Cloud AutoML](#), [IBM Watson Studio](#), [LaunchDarkly](#), [Metaflow](#), [Pycaret](#), [Split.io](#), [TensorBoard](#), [Weights and Biases](#), [Amazon Sagemaker](#), [Apache Airflow](#), [Apache Flume](#), [Apache Hadoop](#), [Apache Spark](#), [Auto-Surprise](#), [BentoML](#), [CML](#), [Cortex](#), [DVC](#), [Grafana](#), [Great Expectations](#), [Holoclean](#), [Kedro](#), [Kubeflow](#), [Kubernetes](#), [Luigi](#), [MLflow](#), [ModelDB](#), [Neo4j](#), [pydqc](#), [Snorkel](#), [TensorFlow Lite](#), [TPOT](#)
- *17-745 students: Research project instead, contact us now*

# LEARNING GOALS

- Distinguish precision and accuracy; understanding the better models vs more data tradeoffs
- Use schema languages to enforce data schemas
- Design and implement automated quality assurance steps that check data schema conformance and distributions
- Devise infrastructure for detecting data drift and schema violations
- Consider data quality as part of a system; design an organization that values data quality

# **DATA-QUALITY CHALLENGES**

*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.

# CASE STUDY: INVENTORY MANAGEMENT



# DATA COMES FROM MANY SOURCES

- Manually entered
- Actions from IT systems
- Logging information, traces of user interactions
- Sensor data
- Crowdsourced

# MANY DATA SOURCES



*sources of different reliability and quality*

# INVENTORY DATABASE

Product Database:

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

Stock:

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

Sales history:

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

# **RAW DATA IS AN OXYMORON**

Recommended Reading: Gitelman, Lisa, Virginia Jackson, Daniel Rosenberg, Travis D. Williams, Kevin R. Brine, Mary Poovey, Matthew Stanley et al. "Data bite man: The work of sustaining a long-term study." In "Raw Data" Is an Oxymoron, (2013), MIT Press: 147-166.

# 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 in inventory system?

# DATA CHANGES

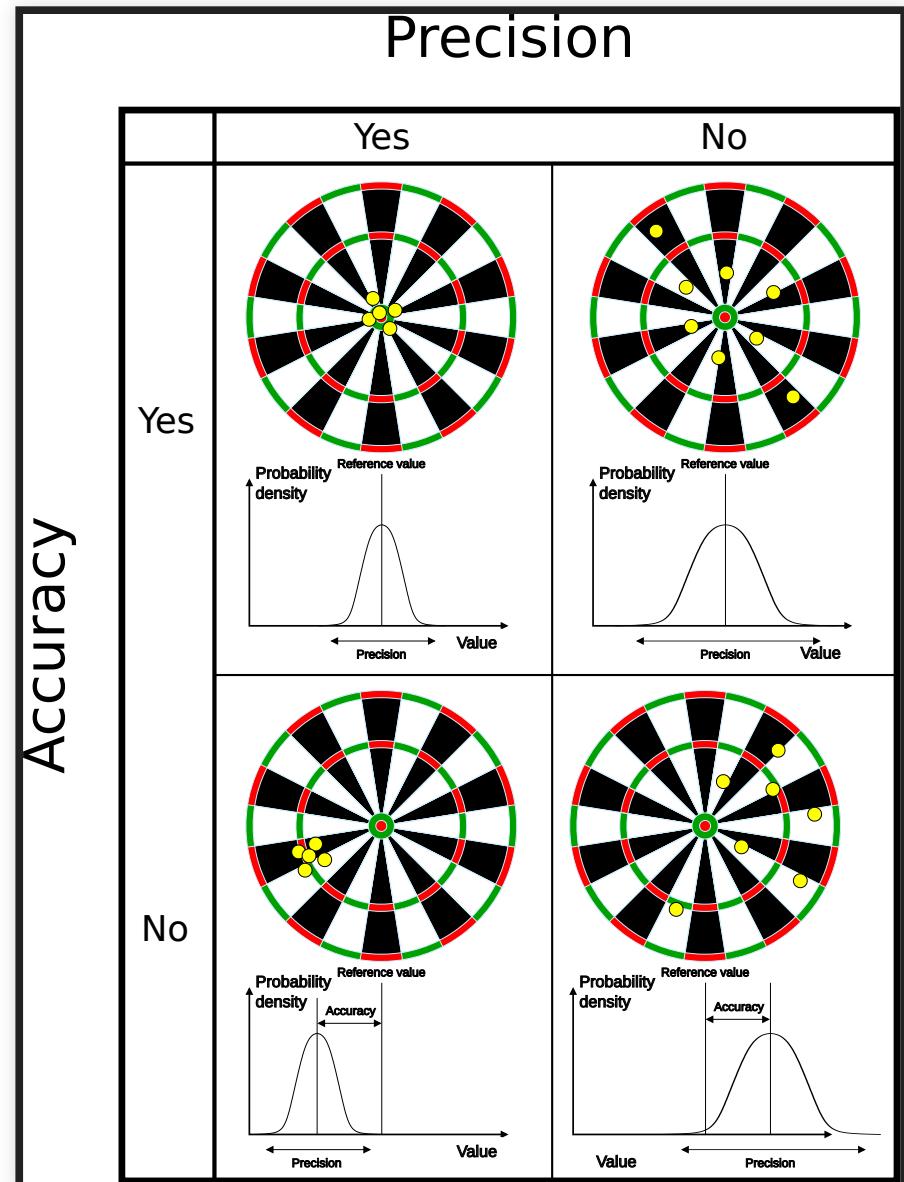
- System objective changes over time
- Software components are upgraded or replaced
- Prediction models change
- Quality of supplied data changes
- User behavior changes
- Assumptions about the environment no longer hold
- Examples in inventory system?

# USERS MAY DELIBERATELY CHANGE DATA

- Users react to model output
- Users try to game/deceive the model
- Examples in inventory system?

# 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

# POOR DATA QUALITY HAS CONSEQUENCES

(often delayed consequences)

# EXAMPLE: SYSTEMATIC BIAS IN LABELING

- Poor data quality leads to poor models
- Not detectable in offline evaluation
- Problem in production -- now difficult to correct

TECH \ AMAZON \ ARTIFICIAL INTELLIGENCE

## Amazon reportedly scraps internal AI recruiting tool that was biased against women

*The secret program penalized applications that contained the word “women’s”*

By James Vincent | Oct 10, 2018, 7:09am EDT

# DELAYED FIXES INCREASE REPAIR COST



# DATA CASCADES



Detection almost always delayed! Expensive rework.

Difficult to detect in offline evaluation.

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021, May). “[Everyone wants to do the model work, not the data work](#)”: Data Cascades in High-Stakes AI. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-15).

# DATA SCHEMA

Ensuring basic consistency about shape and types

# 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?*

## Data Quality Problems



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

# SCHEMA PROBLEMS

- 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

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

# 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));
```

# WHICH PROBLEMS ARE SCHEMA PROBLEMS?

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

# WHAT HAPPENS WHEN NEW DATA VIOLATES SCHEMA?



# 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

# SCHEMA LIBRARY: 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"
  }
]}
```

# SCHEMA LIBRARY: 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 LIBRARIES/FORMATS

## Examples

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

# DISCUSSION: DATA SCHEMA CONSTRAINTS FOR INVENTORY SYSTEM?

Product Database:

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

Stock:

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

Sales history:

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

# SUMMARY: SCHEMA

- Basic structure and type definition of data
- Well supported in databases and many tools
- Very low bar

# INSTANCE-LEVEL PROBLEMS

Inconsistencies, wrong values

# 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 beyond schema problems?*

# INSTANCE-LEVEL PROBLEMS

- 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

Can we detect these?

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

# DISCUSSION: INSTANCE-LEVEL PROBLEMS IN SCENARIO?



# DATA CLEANING OVERVIEW

- Data analysis / Error detection
  - Usually focused on specific kind of problems, e.g., duplication, typos, missing values, distribution shift
  - 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 EXAMPLES

- 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?

# EXAMPLE TOOL: GREAT EXPECTATIONS

```
expect_column_values_to_be_between(  
    column="passenger_count",  
    min_value=1,  
    max_value=6  
)
```

The screenshot shows the Great Expectations web interface for a taxi demo dataset. The top navigation bar includes the project name "great\_expectations", a "Home" link, and a timestamp "2020-08-19T02:46:09.241003Z". Below the navigation is a table of contents with links to "Overview", "Table-Level Expectations", and "passenger\_count". The main content area displays validation results for the "passenger\_count" column. A table lists expectations and their observed values:

Status	Expectation	Observed Value
Green	values must never be null.	100% not null
Green	distinct values must belong to this set: 1.0 2.0 3.0 4.0 5.0 6.0.	[1.0, 2.0, 3.0, 4.0, 5.0, 6.0]

Below the table are two histograms comparing the observed distribution of passenger counts with a target distribution. The first histogram shows the fraction of values for each passenger count from 1 to 6. The second histogram shows the KL Divergence between the observed and target distributions, with a threshold of 0.6.

<https://greatexpectations.io/>

# 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

# MACHINE LEARNING FOR 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>
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

Conflicts

Does not obey data distribution

Conflict

The diagram illustrates data inconsistencies in a database table. It highlights several issues:

- Conflicts:** Arrows point from the word "Conflicts" to the ZIP code **60608** in row t1 and the city **Cicago** in row t4.
- Conflict:** An arrow points from the word "Conflict" to the ZIP code **60608** in row t4.
- Does not obey data distribution:** A curved arrow points from the phrase "Does not obey data distribution" to the DBAName **Johnnyo's** in row t4.

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

# EXAMPLE: 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 Illyas, 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

- $\{\text{Diaper}, \text{Beer}\} \rightarrow \text{Milk}$  (40% support, 66% confidence)
- $\text{Milk} \rightarrow \{\text{Diaper}, \text{Beer}\}$  (40% support, 50% confidence)
- $\{\text{Diaper}, \text{Beer}\} \rightarrow \text{Bread}$  (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



# DETECTING DRIFT

# MONITORING FOR CHANGES

public-bq  
new\_york\_311.311\_service\_requests

Incoming 311 service requests complaints in New York

Configure Run checks

Overview Documentation Checked daily when data is fresh using `created_date`

Aug 22, 2021 Sun 22 Mon 23 Tue 24 Wed 25 Thu 26 Fri 27 Sat 28

Data Freshness 1 / 1 passed Data Volume 1 / 1 passed

Missing Data 1 / 3 failed Table Anomalies 1 / 3 failed

Key Metrics 1 / 3 failed Validation Rules 17 / 17 passed

Columns 

`bigrquery-public-data.new_york_311.311_service_requests` columns, database types, and the distribution of their most common values on `2021-08-22` per `created_date`

Column	Type	Distribution
<code>unique_key</code>	INT64	all other
<code>created_date</code>	TIMESTAMP	all other
<code>closed_date</code>	TIMESTAMP	all other NULL
<code>agency</code>	STRING	NYPD DCP RPD
<code>agency_name</code>	STRING	New York City Police Department Department...
<code>complaint_type</code>	STRING	Noise... Sewer all other
<code>descriptor</code>	STRING	Loud/Music... all other
<code>location_type</code>	STRING	Street/Sidewalk NULL
<code>incident_zip</code>	STRING	10468 all other
<code>incident_address</code>	STRING	all other
<code>street_name</code>	STRING	all other
<code>cross_street_1</code>	STRING	all other NULL
<code>cross_street_2</code>	STRING	all other NULL
<code>intersection_street_1</code>	STRING	all other NULL

<https://www.anomalo.com/>

# DRIFT & MODEL DECAY

- **Concept drift** (or concept shift)
  - 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** (or covariate shift or population 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, new worker performing manual entry)
  - model interprets input data incorrectly
  - over time: abrupt changes due to faulty inputs

How do we fix these drifts?

## 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

# BREAKOUT: DRIFT IN THE INVENTORY SYSTEM

*What kind of drift might be expected?*

As a group, in slack #lecture write plausible example of:

- Concept Drift:
- Data Drift:
- Upstream data changes:



# WATCH FOR DEGRADATION IN PREDICTION ACCURACY

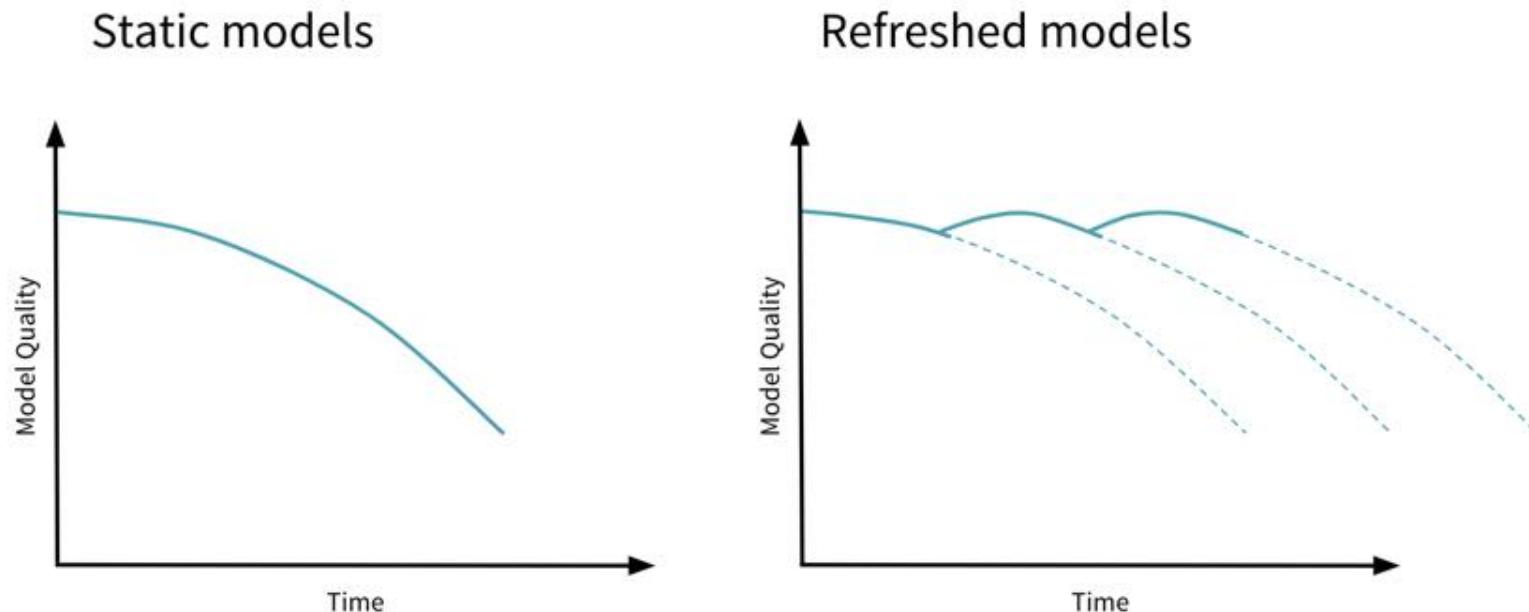


Image source: Joel Thomas and Clemens Mewald. [Productionizing Machine Learning: From Deployment to Drift Detection](#). Databricks Blog, 2019

# 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!

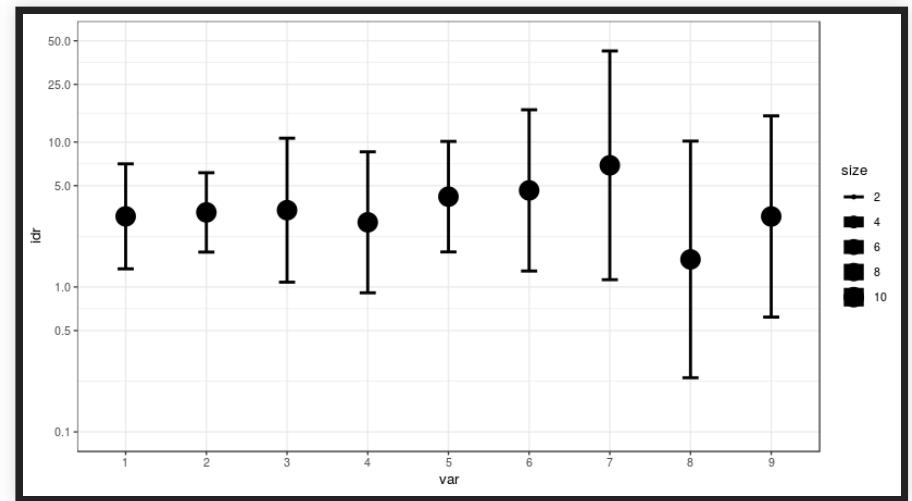
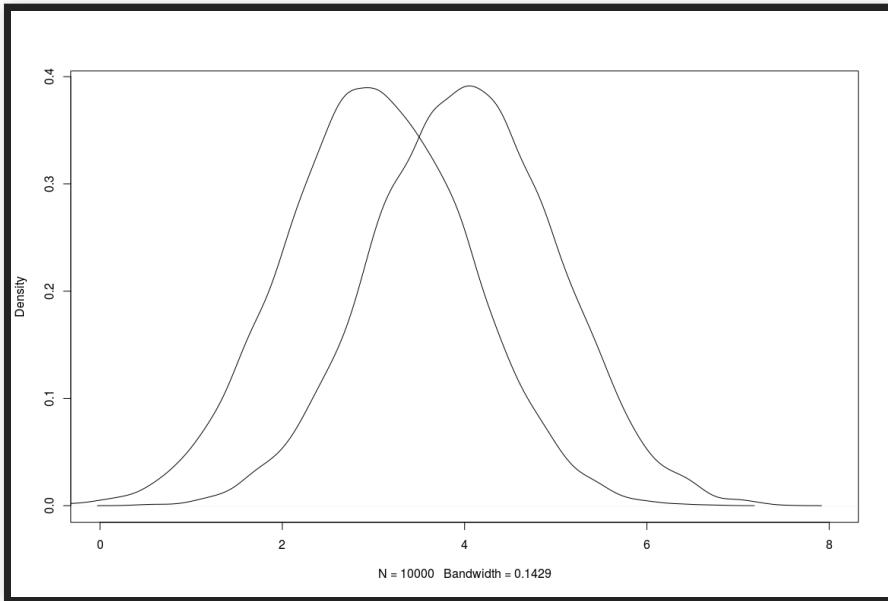
# DIFFERENT FORMS OF DATA DRIFT

- 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 \(AKS\)](#)

# BREAKOUT: DRIFT IN THE INVENTORY SYSTEM

*What kind of monitoring for drift in Inventory scenario?*



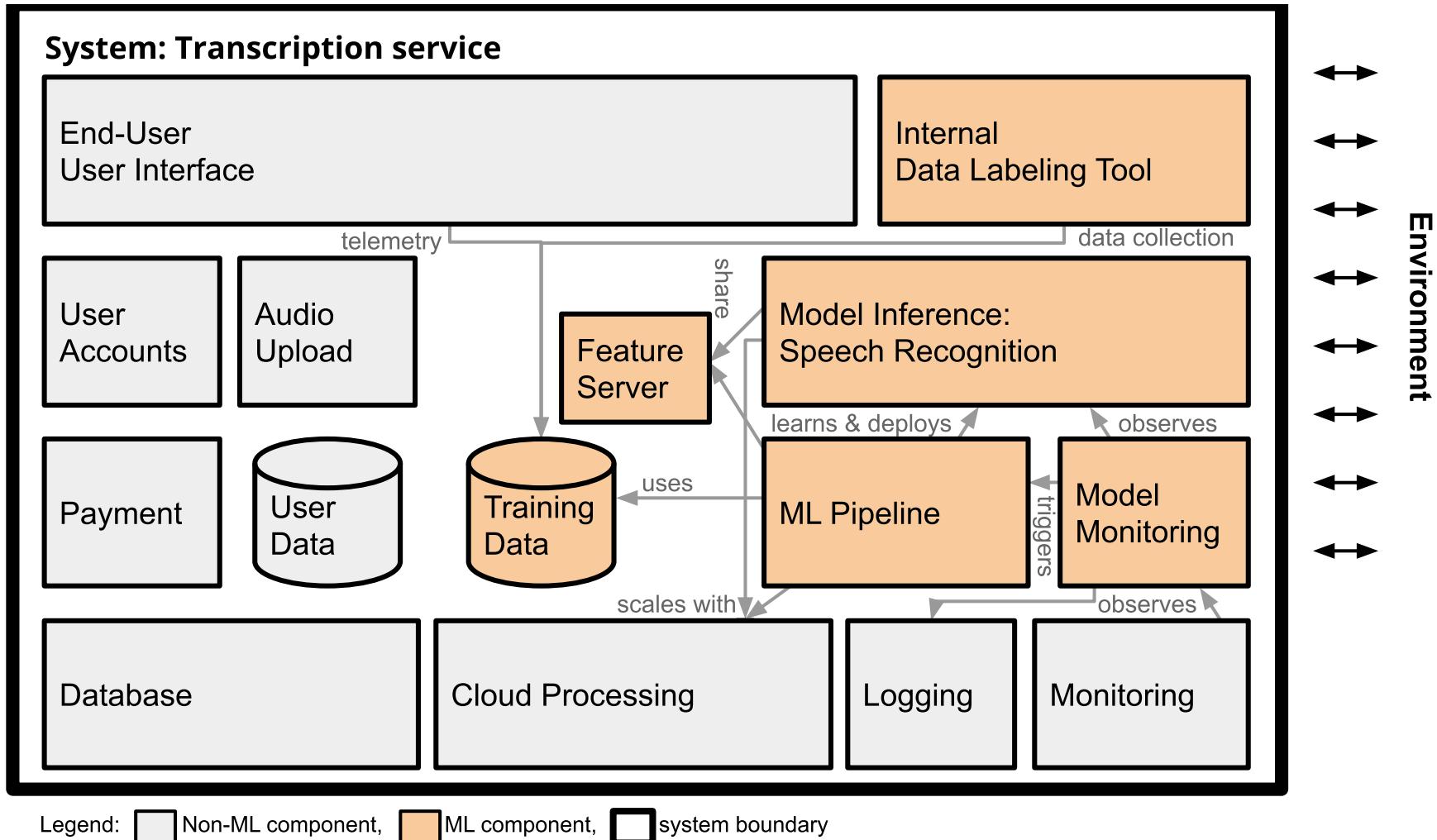
# DATA QUALITY IS A SYSTEM-WIDE CONCERN



*Everyone wants to do the model work, not the data work*

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021, May). “[Everyone wants to do the model work, not the data work](#)”: Data Cascades in High-Stakes AI. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-15).

# DATA FLOWS ACROSS COMPONENTS



# DATA QUALITY IS A SYSTEM-WIDE CONCERN

- Data flows across components
  - e.g., from user interface into database to crowd-sourced labeling team into ML pipeline
- Documentation at the interfaces is important
- Humans interacting with the system
  - Entering data
  - Labeling data
  - Observed with sensors/telemetry
  - Incentives, power structures, recognition
- Organizational practices
  - Value, attention, and resources given to data quality

# DATA QUALITY DOCUMENTATION

- Teams rarely document expectations of data quantity or quality
  - Data quality tests are rare, but some teams adopt defensive monitoring
    - Local tests about assumed structure and distribution of data
    - Identify drift early and reach out to producing teams
  - Several ideas for documenting distributions, including [Datasheets](#) and [Dataset Nutrition Label](#)
    - Mostly focused on static datasets, describing origin, consideration, labeling procedure, and distributions
    - [Example](#)
- 
- Nahar, Nadia, Shurui Zhou, Grace Lewis, and Christian Kästner. "[Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process](#)." In Proceedings of the 44th International Conference on Software Engineering (ICSE), May 2022.
  - Gebru, Timnit, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé lii, and Kate Crawford. "[Datasheets for datasets](#)." Communications of the ACM 64, no. 12 (2021): 86-92.

# COMMON DATA CASCADES

- Interacting with physical world brittleness
  - Idealized data, ignoring realities and change of real-world data
  - Static data, one time learning mindset, no planning for evolution
- Inadequate domain expertise
  - Not understanding the data and its context
  - Involving experts only late for trouble shooting
- Conflicting reward systems
  - Missing incentives for data quality
  - Not recognizing data quality importance, discard as technicality
  - Missing data literacy with partners
- Poor (cross-organizational) documentation
  - Conflicts at team/organization boundary
  - Undetected drift

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021, May). “[Everyone wants to do the model work, not the data work](#)”: Data Cascades in High-Stakes AI. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-15).

# DISCUSSION: POSSIBLE DATA CASCADES IN INVENTORY SCENARIO?

- Interacting with physical world brittleness
- Inadequate domain expertise
- Conflicting reward systems
- Poor (cross-organizational) documentation



# ETHICS AND POLITICS OF DATA

*Raw data is an oxymoron*



# INCENTIVES FOR DATA QUALITY? VALUING DATA WORK?



# **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.*

# BONUS: 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

The screenshot shows a code editor interface with two tabs: `index.js` and `.eslintrc`. The `index.js` tab is active, displaying the following code:

```
1
2
3     for (i = 0; i <= 10; i++)
4     {
5         if (i === 2)
6             ESLint: Opening curly brace does not appear on the same line as controlling
7             statement. (brace-style)
8             }
9         var out = "The value is now " + i;
10        document.write(out);
11        document.write( text: "<br/>" );
12    }
```

An ESLint error message is displayed in a tooltip over the opening brace of the `for` loop on line 4. The message reads: `ESLint: Opening curly brace does not appear on the same line as controlling statement. (brace-style)`. The code editor has a dark theme with syntax highlighting for different programming elements.



# 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.

# SUMMARY

- Data and data quality are essential
- Data from many sources, often inaccurate, imprecise, inconsistent, incomplete, ... -- many different forms of data quality problems
- Many mechanisms for enforcing consistency and cleaning
  - Data schema ensures format consistency
  - Data quality rules ensure invariants across data points
- Concept and data drift are key challenges -- monitor
- Data quality is a system-level concern
  - Data quality at the interface between components
  - Documentation and monitoring often poor
  - Involves organizational structures, incentives, ethics, ...
- Quality assurance for the data processing pipelines