

# Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

# Collecting, Labeling, and Validating Data



DeepLearning.AI

---

# Welcome

# The importance of data

*“Data is the hardest part of ML and the most important piece to get right...  
Broken data is the most common cause of problems in production ML systems”*  
- Scaling Machine Learning at Uber with Michelangelo - Uber

*“No other activity in the machine learning life cycle has a higher return on investment than improving the data a model has access to.”*  
- Feast: Bridging ML Models and Data - Gojek

# Introduction to Machine Learning Engineering for Production



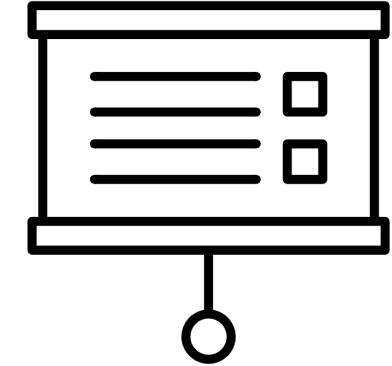
DeepLearning.AI

---

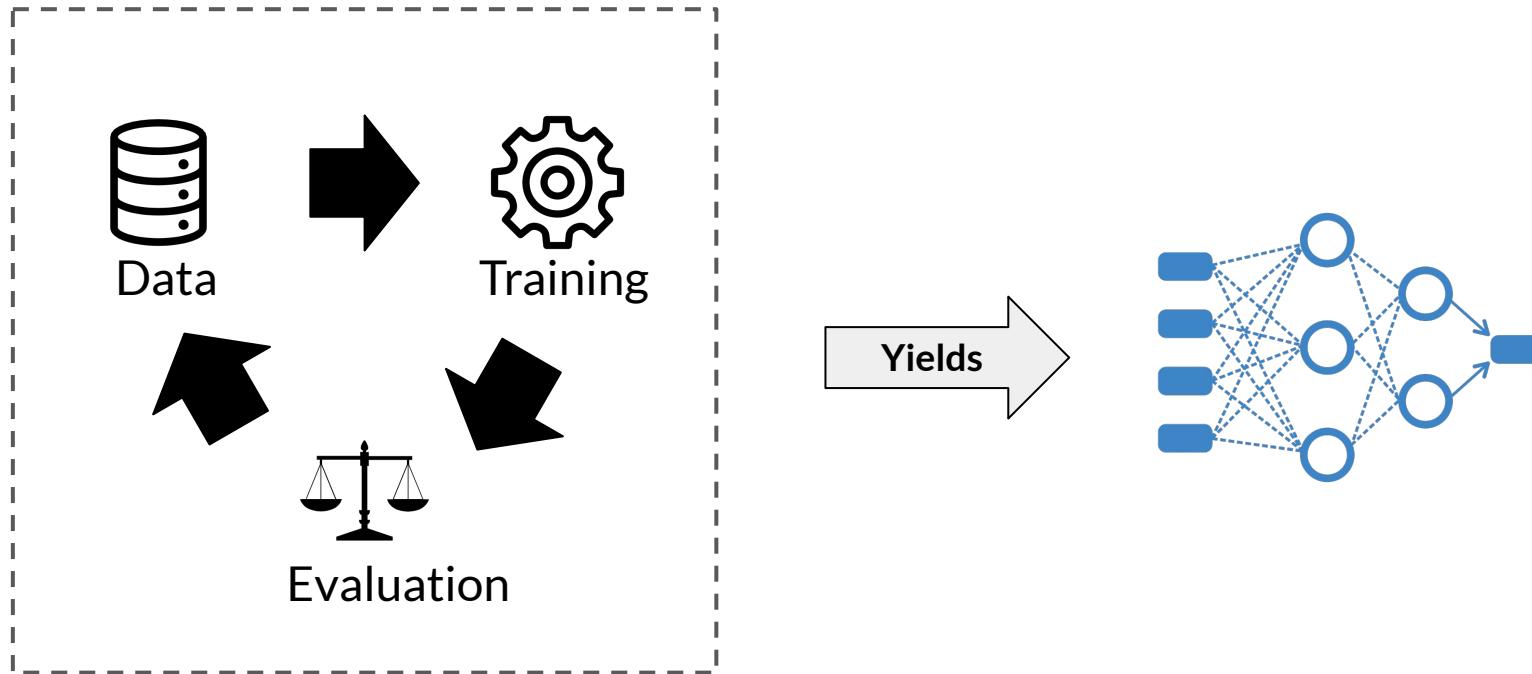
## Overview

# Outline

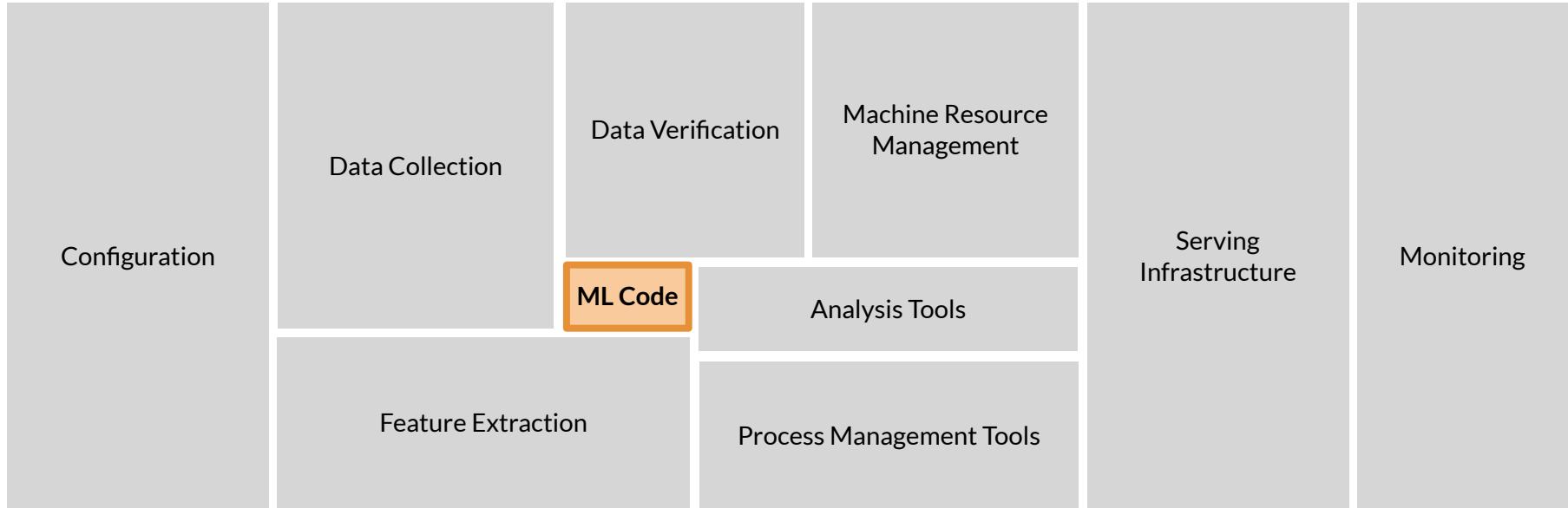
- Machine learning (ML) engineering for production: overview
- Production ML = ML development + software development
- Challenges in production ML



# Traditional ML modeling



# Production ML systems require so much more

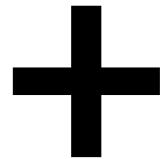


# ML modeling vs production ML

	Academic/Research ML	Production ML
Data	Static	Dynamic - Shifting
Priority for design	Highest overall accuracy	Fast inference, good interpretability
Model training	Optimal tuning and training	Continuously assess and retrain
Fairness	Very important	Crucial
Challenge	High accuracy algorithm	Entire system

# Production machine learning

Machine learning  
development



Modern software  
development

# Managing the entire life cycle of data

- Labeling
- Feature space coverage
- Minimal dimensionality
- Maximum predictive data
- Fairness
- Rare conditions

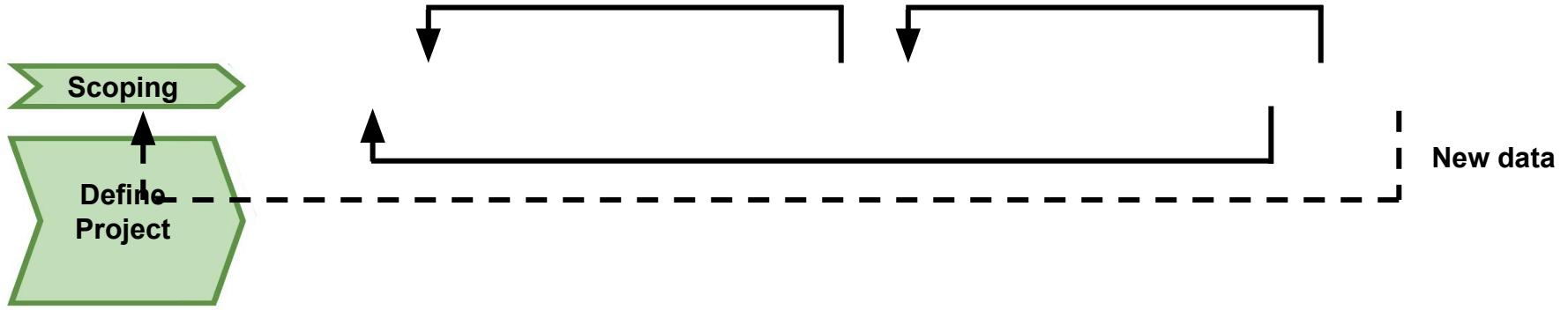
# Modern software development

Accounts for:

- Scalability
- Extensibility
- Configuration
- Consistency & reproducibility
- Safety & security
- Modularity
- Testability
- Monitoring
- Best practices



# Production machine learning system



# Challenges in production grade ML

- Build integrated ML systems
- Continuously operate it in production
- Handle continuously changing data
- Optimize compute resource costs



# Introduction to Machine Learning Engineering for Production



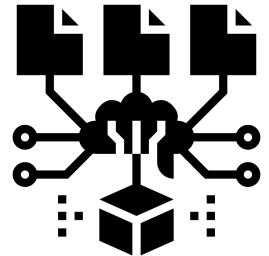
DeepLearning.AI

---

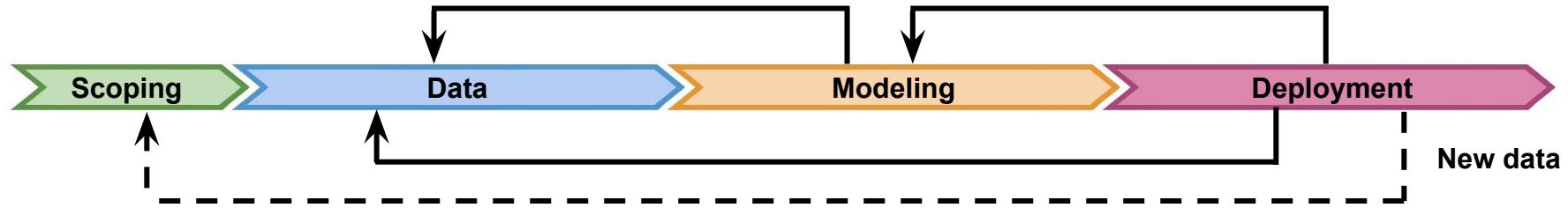
## ML Pipelines

# Outline

- ML Pipelines
- Directed Acyclic Graphs and Pipeline Orchestration Frameworks
- Intro to TensorFlow Extended (TFX)



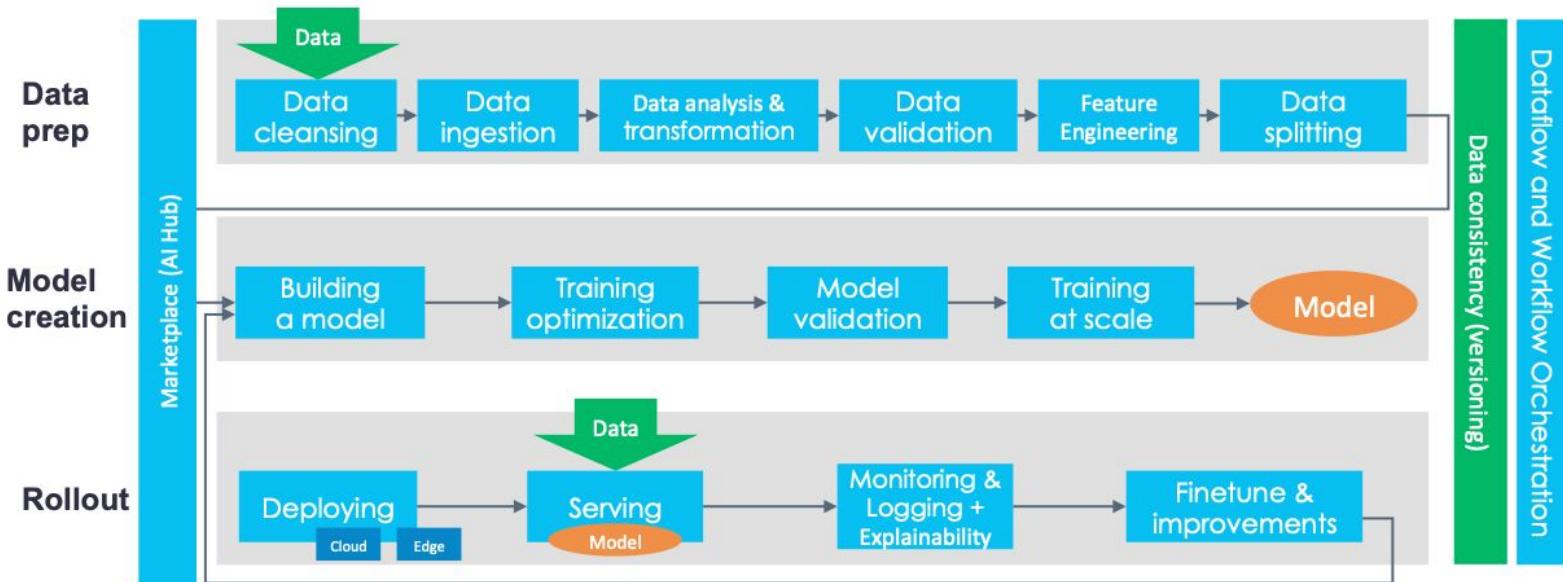
# ML pipelines



Infrastructure for  
automating, monitoring, and maintaining  
model training and deployment

# Production ML infrastructure

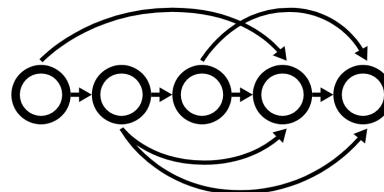
## CD Foundation MLOps reference architecture



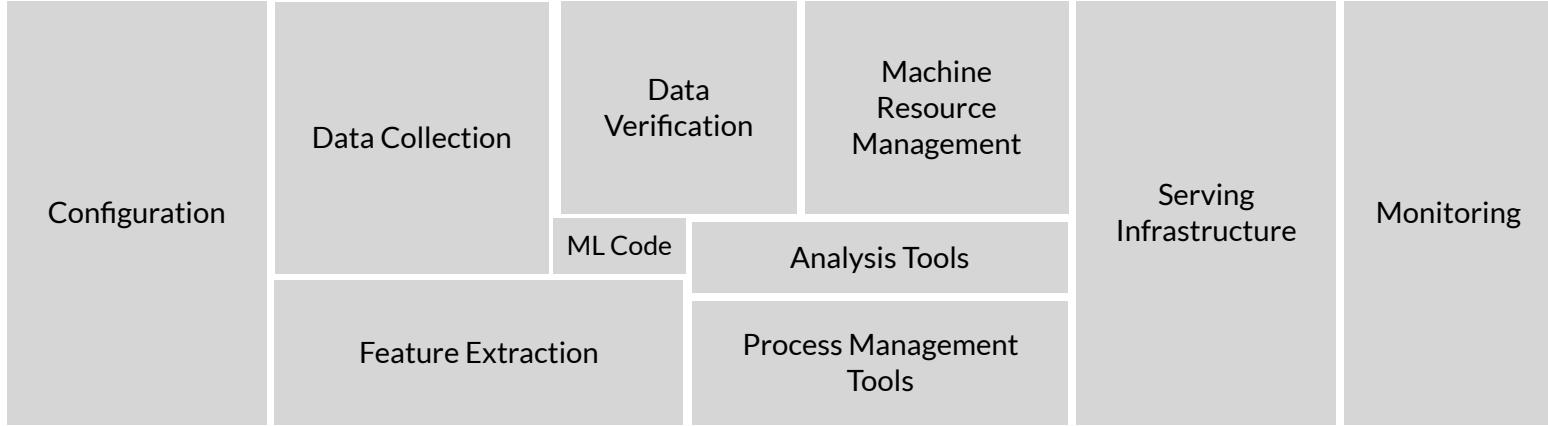
# Directed acyclic graphs



- A directed acyclic graph (DAG) is a directed graph that has no cycles
- ML pipeline workflows are usually DAGs
- DAGs define the sequencing of the tasks to be performed, based on their relationships and dependencies.



# Pipeline orchestration frameworks



- Responsible for scheduling the various components in an ML pipeline DAG dependencies
- Help with pipeline automation
- Examples: Airflow, Argo, Celery, Luigi, Kubeflow

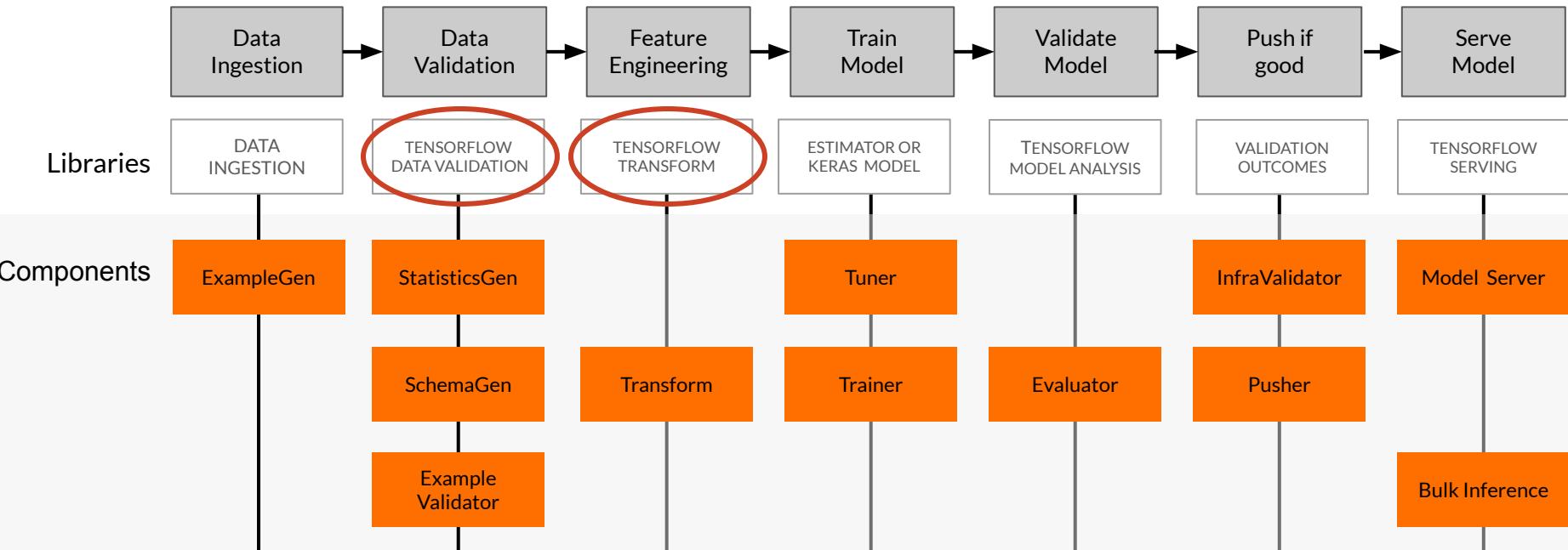
# TensorFlow Extended (TFX)

End-to-end platform for deploying production ML pipelines

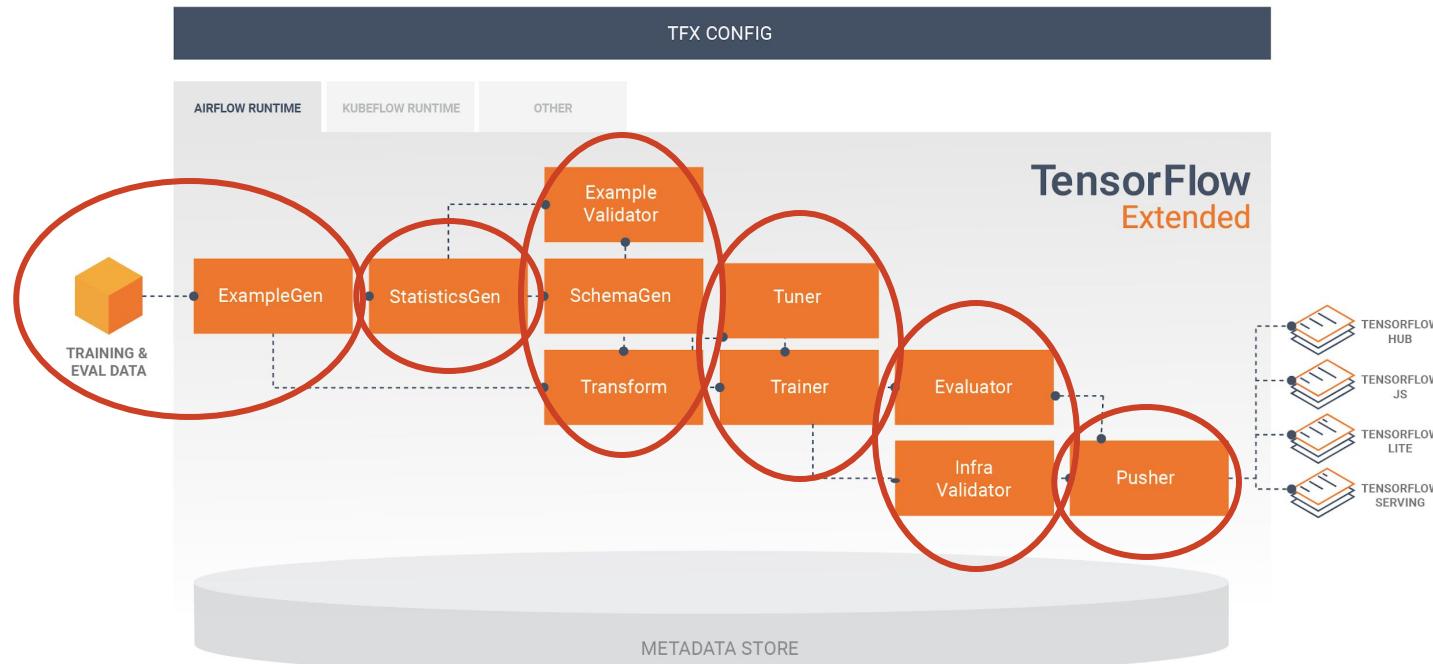


Sequence of components that are designed for scalable, high-performance machine learning tasks

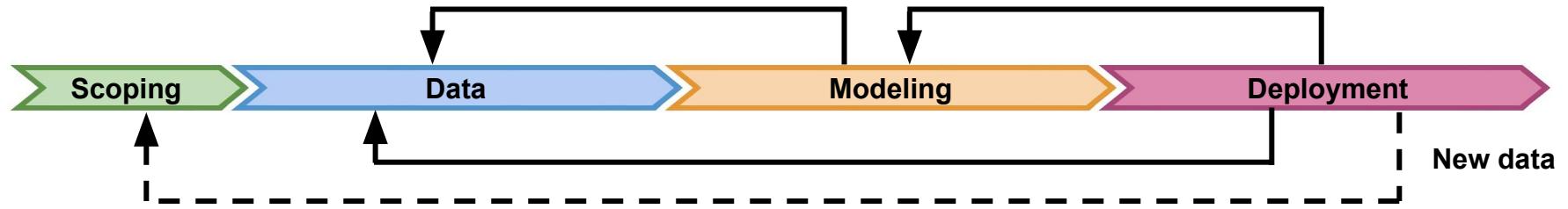
# TFX production components



# TFX Hello World



# Key points



- Production ML pipelines: automating, monitoring, and maintaining end-to-end processes
- Production ML is much more than just ML code
  - ML development + software development
- TFX is an open-source end-to-end ML platform

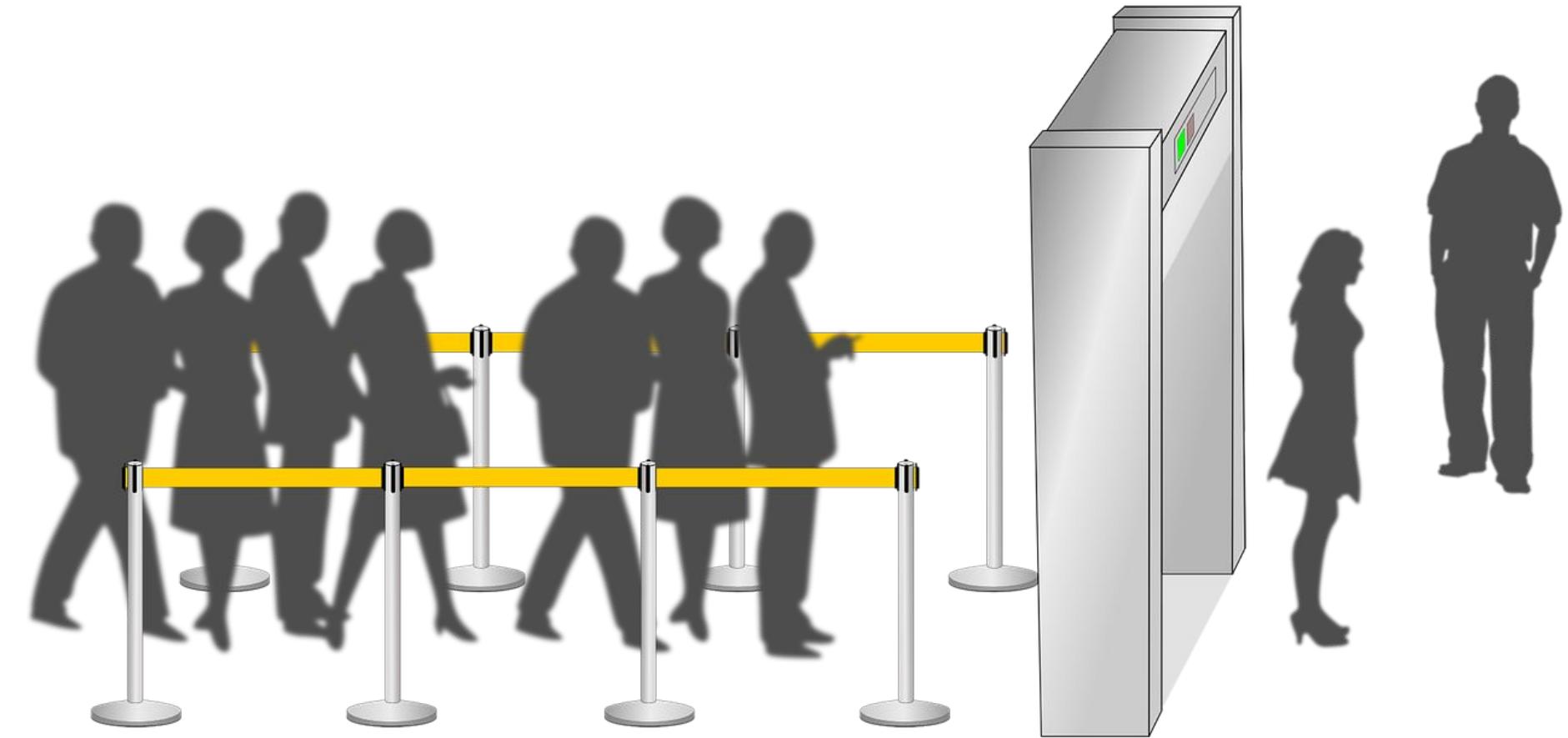


DeepLearning.AI

## Collecting Data

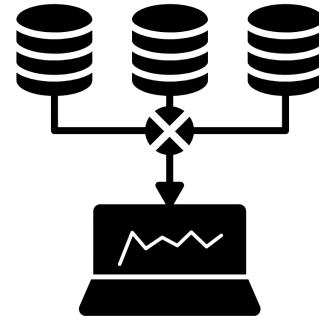
---

## Importance of Data



# Outline

- Importance of data quality
- Data pipeline: data collection, ingestion and preparation
- Data collection and monitoring



# The importance of data

*“Data is the hardest part of ML and the most important piece to get right... Broken data is the most common cause of problems in production ML systems”*

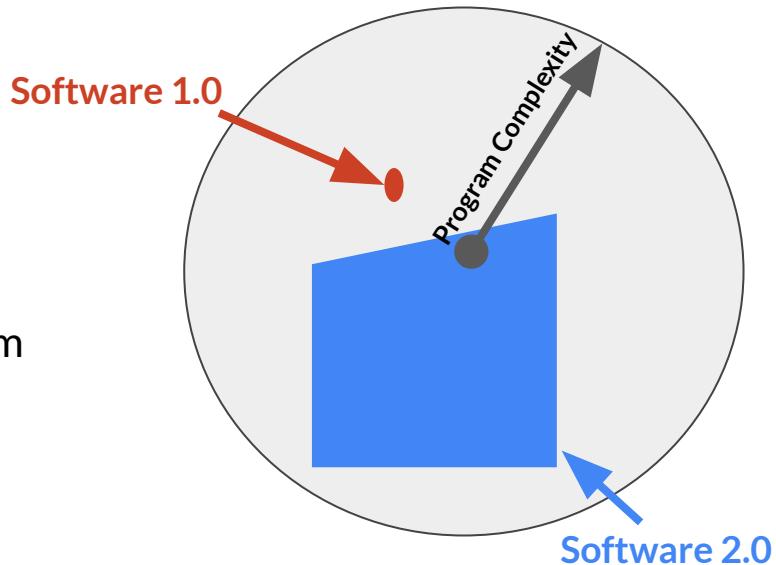
- Scaling Machine Learning at Uber with Michelangelo - Uber

*“No other activity in the machine learning life cycle has a higher return on investment than improving the data a model has access to.”*

- Feast: Bridging ML Models and Data - Gojek

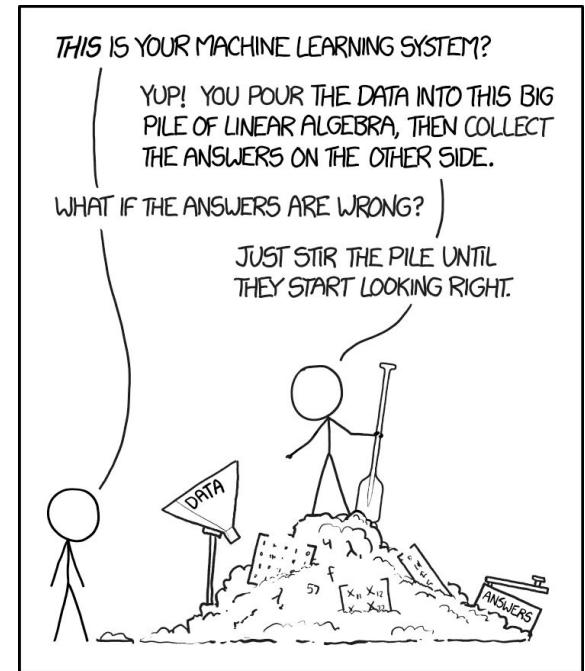
# ML: Data is a first class citizen

- Software 1.0
  - Explicit instructions to the computer
- Software 2.0
  - Specify some goal on the behavior of a program
  - Find solution using optimization techniques
  - Good data is key for success
  - Code in Software = Data in ML



# Everything starts with data

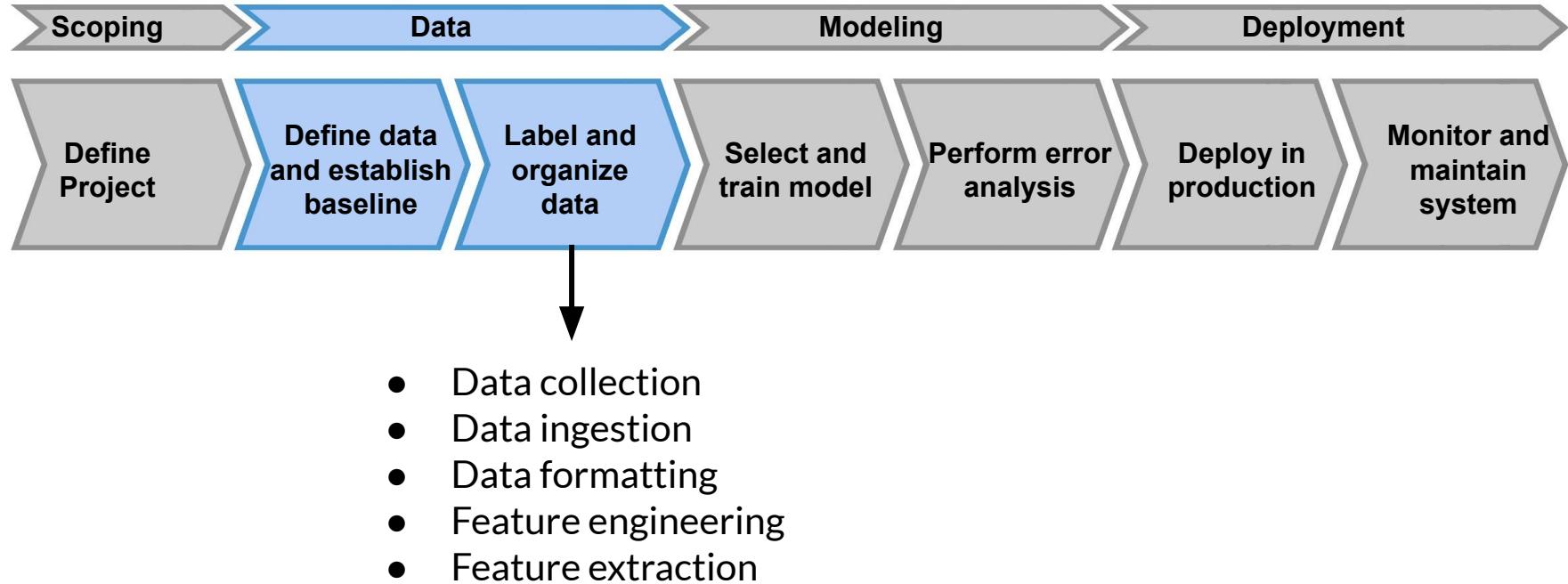
- Models aren't magic
- Meaningful data:
  - maximize predictive content
  - remove non-informative data
  - feature space coverage



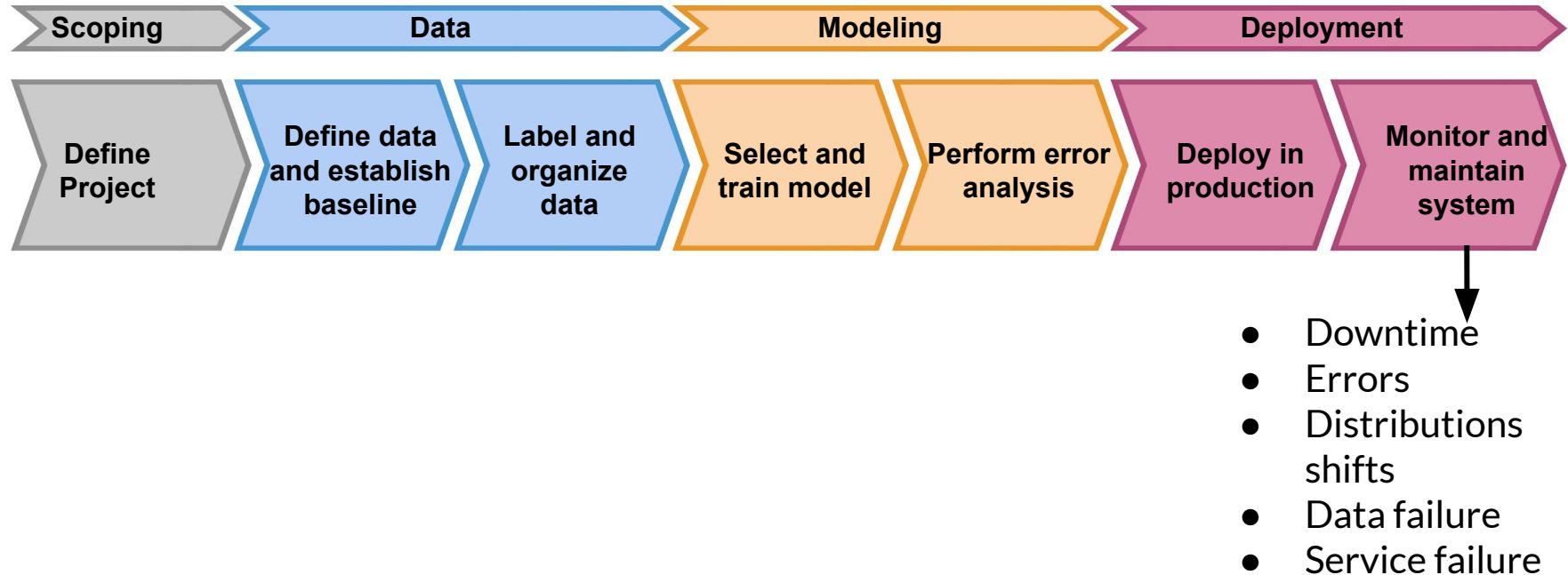
# Garbage in, garbage out

$$f(\text{garbage}) = \text{garbage}$$

# Data pipeline



# Data collection and monitoring



# Key Points

- Understand users, translate user needs into data problems
- Ensure data coverage and high predictive signal
- Source, store and monitor quality data responsibly



DeepLearning.AI

## Collecting Data

---

# Example Application: Suggesting Runs



# Example application: Suggesting runs

<b>Users</b>	Runners
<b>User Need</b>	Run more often
<b>User Actions</b>	Complete run using the app
<b>ML System Output</b>	<ul style="list-style-type: none"><li>• What routes to suggest</li><li>• When to suggest them</li></ul>
<b>ML System Learning</b>	<ul style="list-style-type: none"><li>• Patterns of behaviour around accepting run prompts</li><li>• Completing runs</li><li>• Improving consistency</li></ul>



# Key considerations

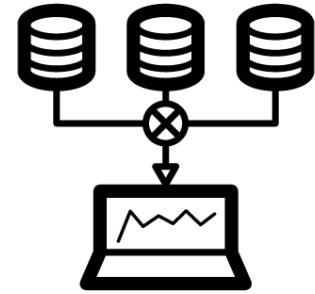
- Data availability and collection
  - What kind of/how much data is available?
  - How often does the new data come in?
  - Is it annotated?
    - If not, how hard/expensive is it to get it labeled?
- Translate user needs into data needs
  - Data needed
  - Features needed
  - Labels needed

 Example dataset

FEATURES					LABELS	
EXAMPLES	Runner ID	Run	Runner Time	Elevation	Fun	LABELS
EXAMPLES	AV3DE	Boston Marathon	03:40:32	1,300 ft	Low	LABELS
	X8KGF	Seattle Oktoberfest 5k	00:35:40	0 ft	High	
	BH9IU	Houston Half-marathon	02:01:18	200 ft	Medium	

# Get to know your data

- Identify data sources
- Check if they are refreshed
- Consistency for values, units, & data types
- Monitor outliers and errors



# Dataset issues

- Inconsistent formatting
  - Is zero “0”, “0.0”, or an indicator of a missing measurement
- Compounding errors from other ML Models
- Monitor data sources for system issues and outages

# Measure data effectiveness

- Intuition about data value can be misleading
  - Which features have predictive value and which ones do not?
- Feature engineering helps to maximize the predictive signals
- Feature selection helps to measure the predictive signals

# Translate user needs into data needs

<b>Data Needed</b>	<ul style="list-style-type: none"><li>• Running data from the app</li><li>• Demographic data</li><li>• Local geographic data</li></ul>
--------------------	--

# Translate user needs into data needs

Features Needed	
	<ul style="list-style-type: none"><li>• Runner demographics</li><li>• Time of day</li><li>• Run completion rate</li><li>• Pace</li><li>• Distance ran</li><li>• Elevation gained</li><li>• Heart rate</li></ul>

# Translate user needs into data needs

## Labels Needed

- Runner acceptance or rejection of app suggestions
- User generated feedback regarding why suggestion was rejected
- User rating of enjoyment of recommended runs

# Key points

- Understand your user, translate their needs into data problems
  - What kind of/how much data is available
  - What are the details and issues of your data
  - What are your predictive features
  - What are the labels you are tracking
  - What are your metrics





DeepLearning.AI

## Collecting Data

---

# Responsible Data: Security, Privacy & Fairness

# Outline

- Data Sourcing
- Data Security and User Privacy
- Bias and Fairness



# Avoiding problematic biases in datasets

Example: classifier trained on the Open Images dataset



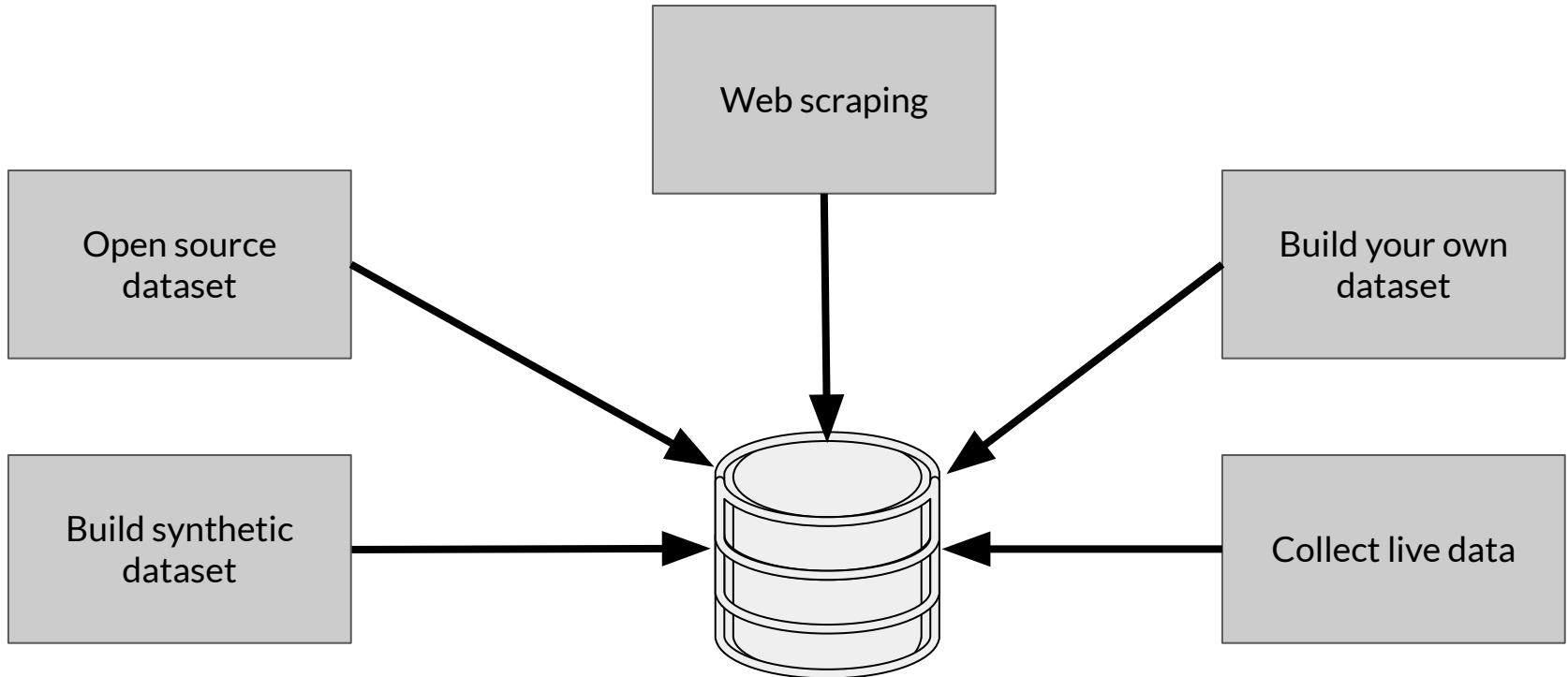
*ceremony,  
wedding, bride,  
man, groom,  
woman, dress*

*bride,  
ceremony,  
wedding, dress,  
woman*

*ceremony,  
bride, wedding,  
man, groom,  
woman, dress*

*person, people*

# Source Data Responsibly



# Data security and privacy

- Data collection and management isn't just about your model
  - Give user control of what data can be collected
  - Is there a risk of inadvertently revealing user data?
- Compliance with regulations and policies (e.g. GDPR)

# Users privacy

- Protect personally identifiable information
  - Aggregation - replace unique values with summary value
  - Redaction - remove some data to create less complete picture

# How ML systems can fail users



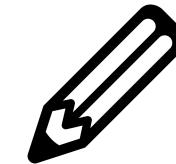
Fair



Accountable



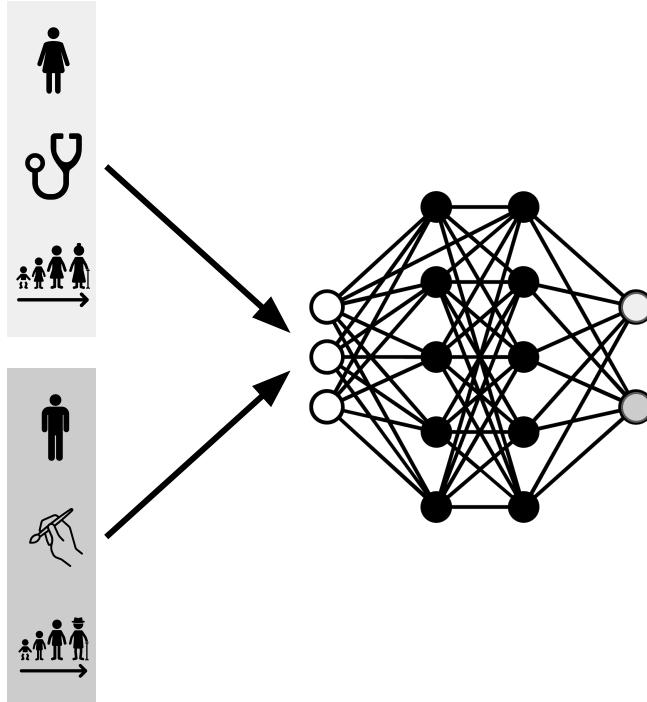
Transparent



Explainable

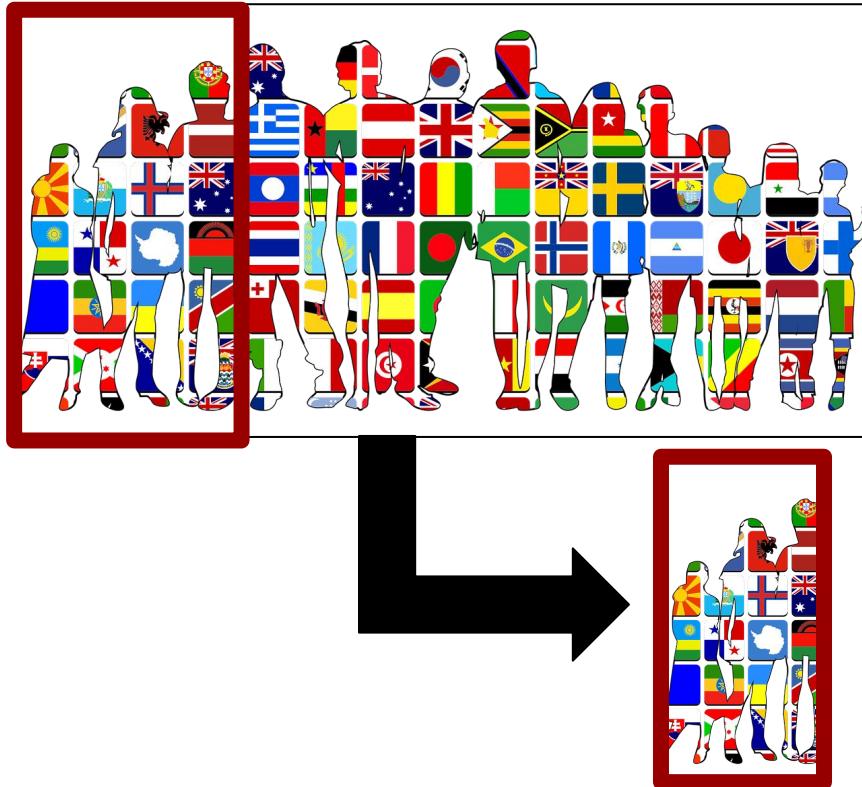
- Representational harm
- Opportunity denial
- Disproportionate product failure
- Harm by disadvantage

# Commit to fairness



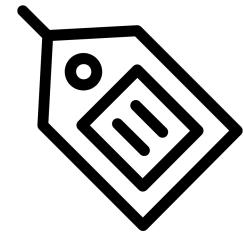
- Make sure your models are fair
  - Group fairness, equal accuracy
- Bias in human labeled and/or collected data
- ML Models can amplify biases

# Biased data representation

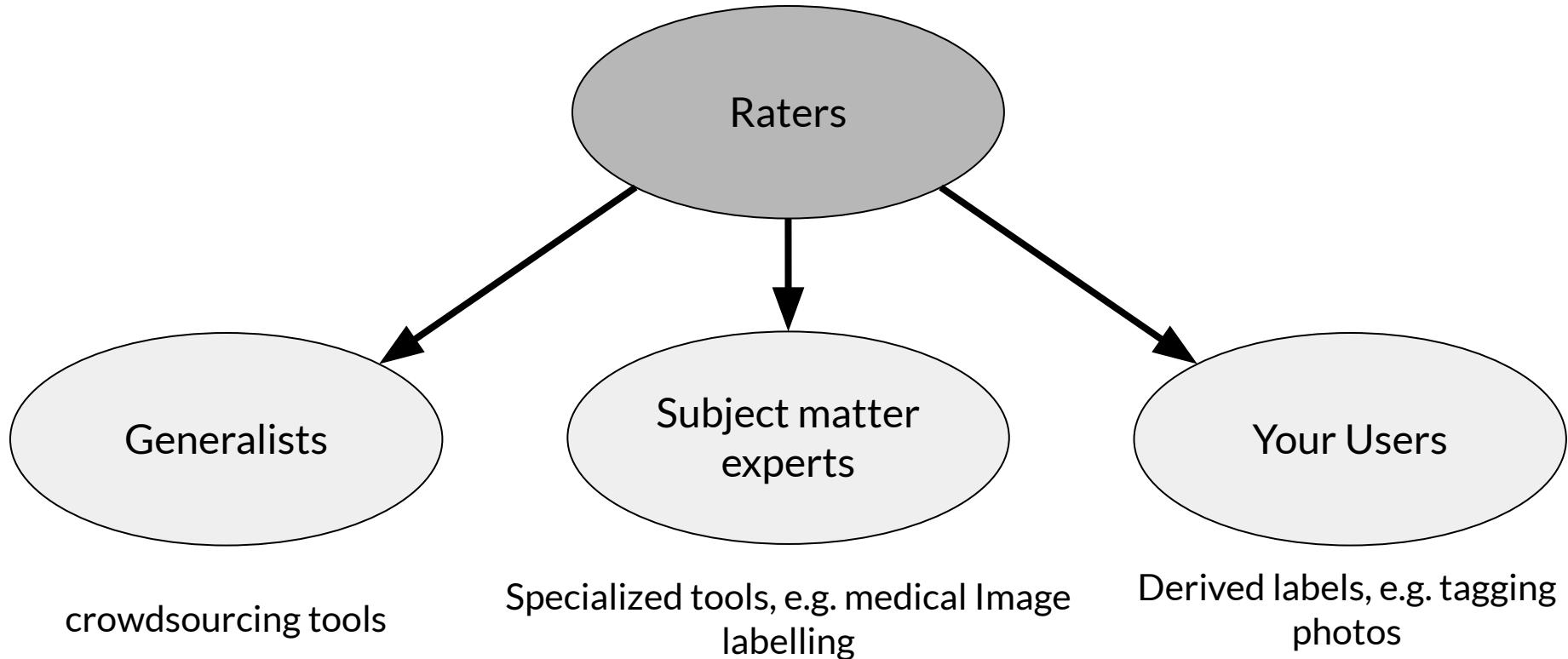


# Reducing bias: Design fair labeling systems

- Accurate labels are necessary for supervised learning
- Labeling can be done:
  - Automation (logging or weak supervision)
  - Humans (aka “Raters”, often semi-supervised)



# Types of human raters



# Key points

- Ensure rater pool diversity
- Investigate rater context and incentives
- Evaluate rater tools
- Manage cost
- Determine freshness requirements



DeepLearning.AI

## Labeling Data

---

# Case Study: Degraded Model Performance

# You're an Online Retailer Selling Shoes ...

Your model predicts  
**click-through rates**  
**(CTR)**, helping you decide  
how much inventory to  
order



# When suddenly

Your AUC and prediction accuracy  
have dropped on men's dress shoes!





# Why?



# How do we know that we have a problem?





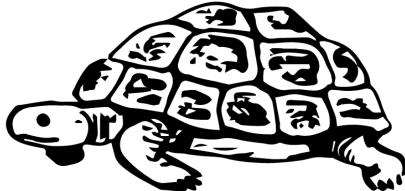
# Case study: taking action

- How to detect problems early on?
- What are the possible causes?
- What can be done to solve these?

# What causes problems?

Kinds of problems:

- Slow - example: drift
- Fast - example: bad sensor, bad software update



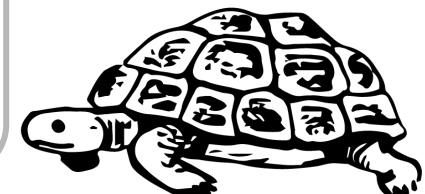
# Gradual problems

## Data changes

- Trend and seasonality
- Distribution of features changes
- Relative importance of features changes

## World changes

- Styles change
- Scope and processes change
- Competitors change
- Business expands to other geos



# Sudden problems

## Data collection problem

- Bad sensor/camera
- Bad log data
- Moved or disabled sensors/cameras

## Systems problem

- Bad software update
- Loss of network connectivity
- System down
- Bad credentials



# Why “Understand” the model?

- Mispredictions do not have uniform **cost** to your business
- The **data you have** is rarely the data you wish you had
- Model objective is nearly always a **proxy** for your business objectives
- Some percentage of your customers may have a **bad experience**

**The real world does not stand still!**



DeepLearning.AI

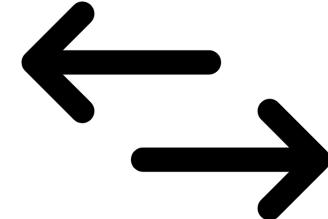
## Labeling Data

---

Data and Concept  
Change in  
Production ML

# Outline

- Detecting problems with deployed models
  - Data and concept change
- Changing ground truth
  - Easy problems
  - Harder problems
  - Really hard problems



# Detecting problems with deployed models

- Data and scope changes
- Monitor models and validate data to find problems early
- Changing ground truth: **label** new training data

# Easy problems

- Ground truth changes slowly (months, years)
- Model retraining driven by:
  - Model improvements, better data
  - Changes in software and/or systems
- Labeling
  - Curated datasets
  - Crowd-based



# Harder problems

- Ground truth changes faster (weeks)
- Model retraining driven by:
  - Declining model performance
  - Model improvements, better data
  - Changes in software and/or system
- Labeling
  - Direct feedback
  - Crowd-based



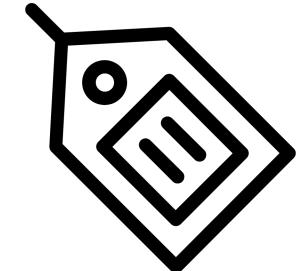
# Really hard problems

- Ground truth changes very fast (days, hours, min)
- Model retraining driven by:
  - Declining model performance
  - Model improvements, better data
  - Changes in software and/or system
- Labeling
  - Direct feedback
  - Weak supervision



# Key points

- Model performance decays over time
  - Data and Concept Drift
- Model retraining helps to improve performance
  - Data labeling for changing ground truth and scarce labels





DeepLearning.AI

## Labeling Data

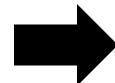
---

# Process Feedback and Human Labeling

# Data labeling

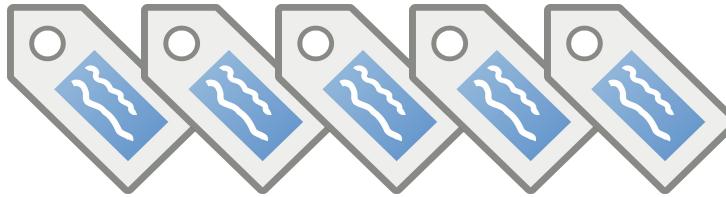
## Variety of Methods

- Process Feedback (Direct Labeling)
- Human Labeling
- ~~Semi Supervised Labeling~~
- ~~Active Learning~~
- ~~Weak Supervision~~



Practice later as advanced  
labeling methods

# Data labeling



**Process Feedback**

Example: Actual vs predicted click-through

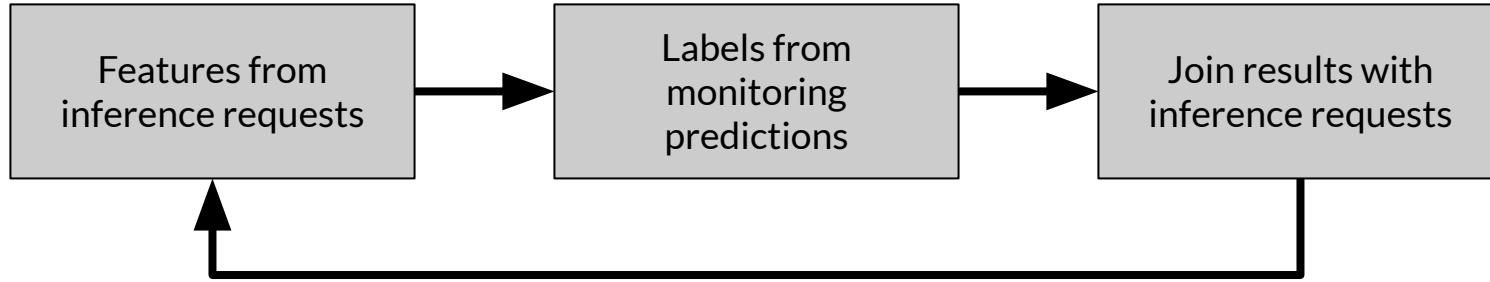
**Human Labeling**

Example: Cardiologists labeling MRI images

# Why is labeling important in production ML?

- Using business/organisation available data
- Frequent model retraining
- Labeling ongoing and critical process
- Creating a training datasets requires labels

# Direct labeling: continuous creation of training dataset



Similar to reinforcement learning  
rewards

# Process feedback - advantages

- Training dataset continuous creation
- Labels evolve quickly
- Captures strong label signals

# Process feedback - disadvantages

- Hindered by inherent nature of the problem
- Failure to capture ground truth
- Largely bespoke design

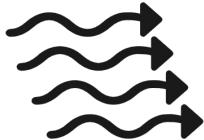
# Process feedback - Open-Source log analysis tools



## Logstash

Free and open source data processing pipeline

- Ingests data from a multitude of sources
- Transforms it
- Sends it to your favorite "stash."

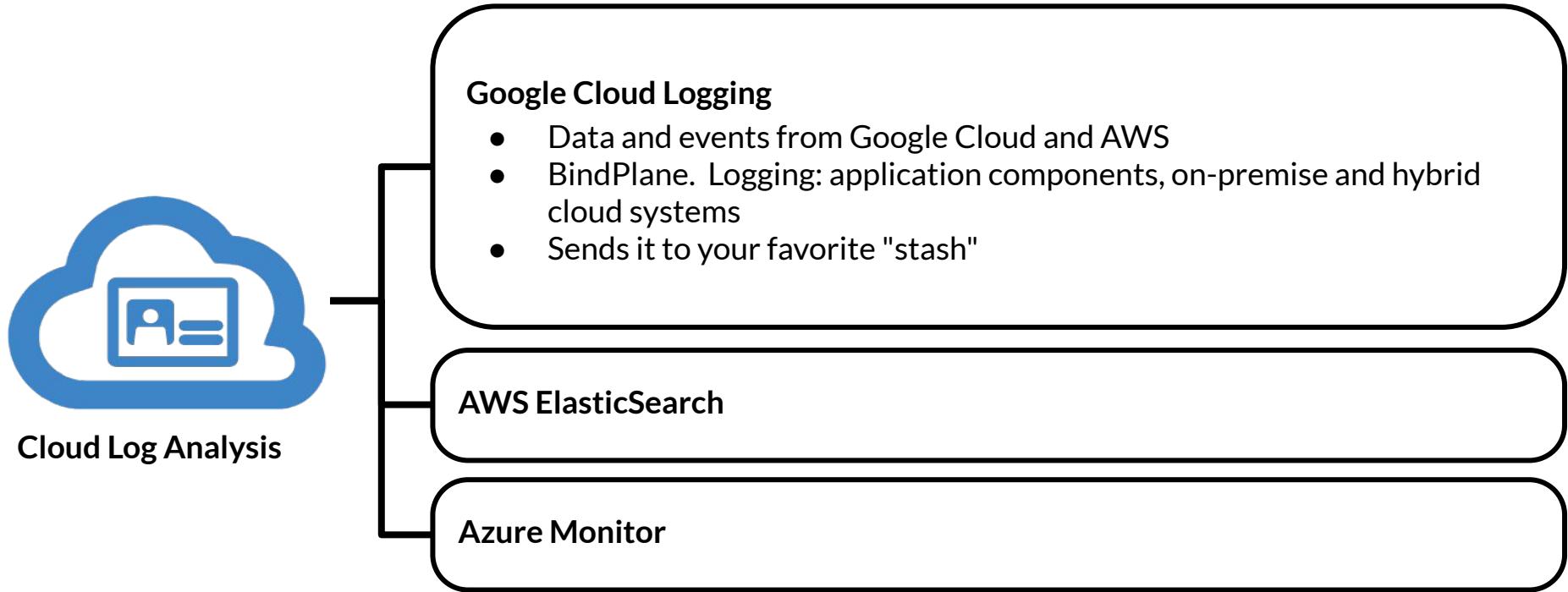


## Fluentd

Open source data collector

Unify the data collection and consumption

# Process feedback - Cloud log analytics



# Human labeling

People (“raters”) to examine data and assign labels manually



Raw data



Unlabeled and ambiguous data  
is sent to raters for annotation



A training data set is  
ready for use

# Human labeling - Methodology



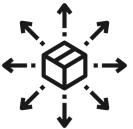
Unlabeled data is collected



Human “raters” are recruited



Instructions to guide raters are created



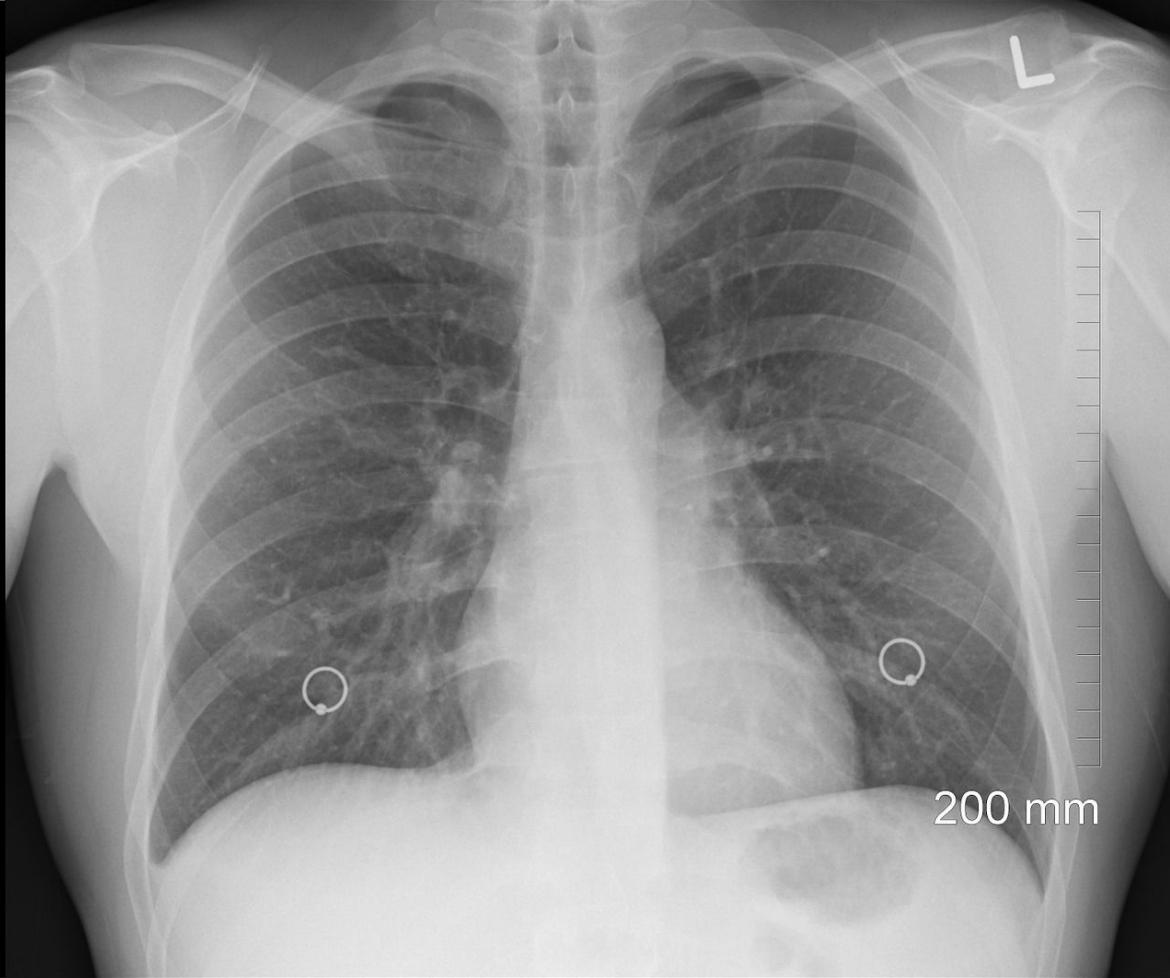
Data is divided and assigned to raters



Labels are collected and conflicts resolved

# Human labeling - advantages

- More labels
- Pure supervised learning



# Human labeling - Disadvantages



Quality consistency: Many datasets  
difficult for human labeling



Slow

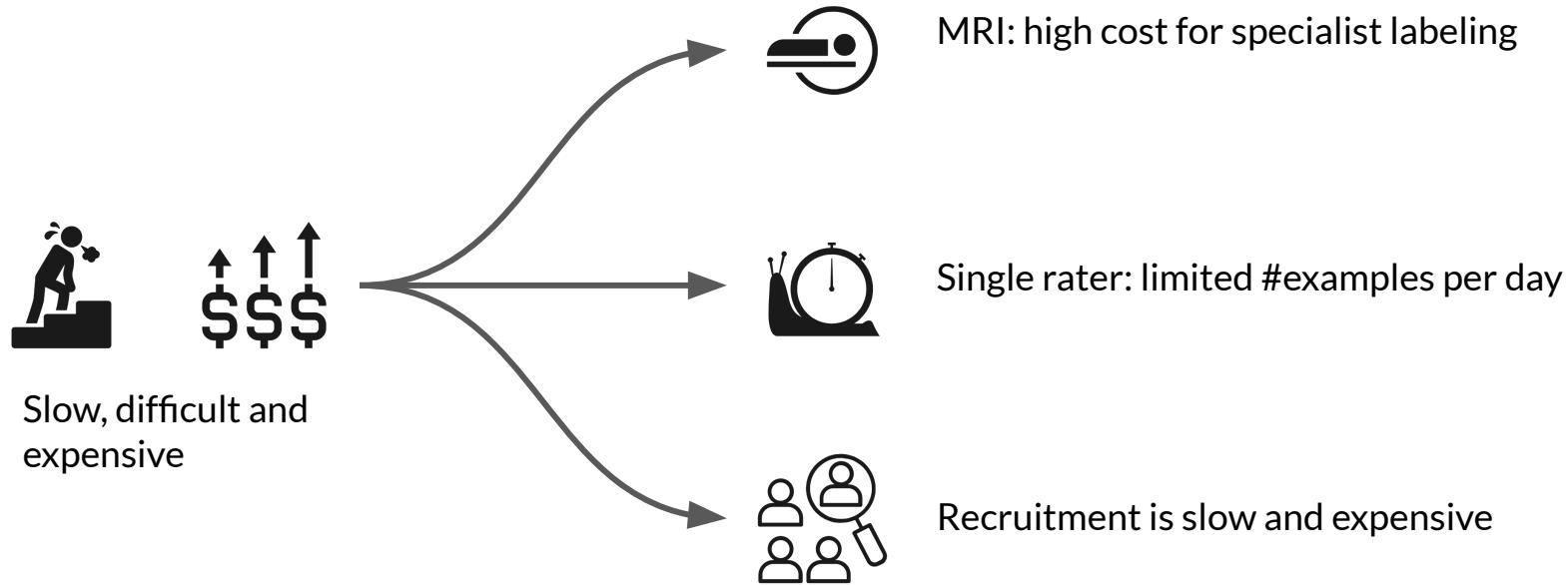


Expensive



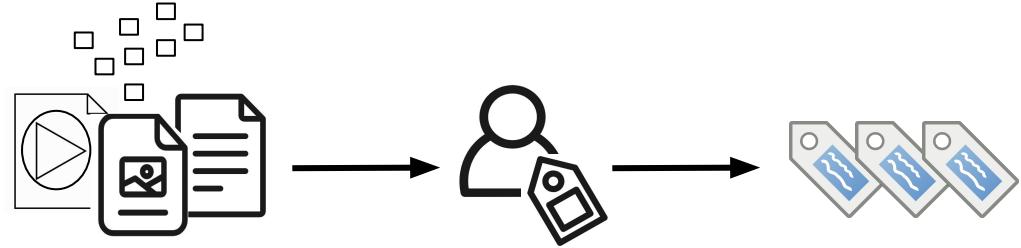
Small dataset curation

# Why is human labeling a problem?



# Key points

- Various methods of data labeling
  - Process feedback
  - Human labeling
- Advantages and disadvantages of both





DeepLearning.AI

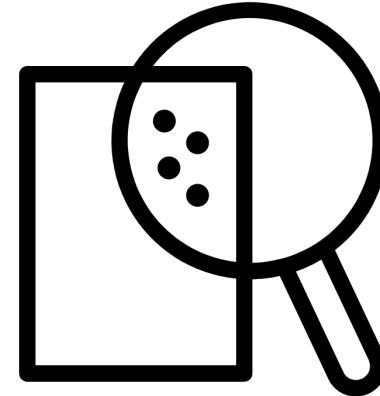
## Validating Data

---

## Detecting Data Issues

# Outline

- Data issues
  - Drift and skew
    - Data and concept Drift
    - Schema Skew
    - Distribution Skew
- Detecting data issues



# Drift and skew

## Drift

Changes in data over time, such as data collected once a day

## Skew

Difference between two static versions, or different sources, such as training set and serving set

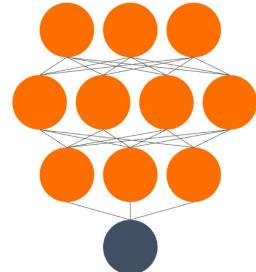
# Typical ML pipeline

During **training**

Data



Batch processing

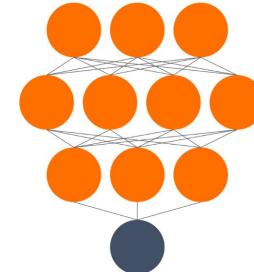


During **serving**

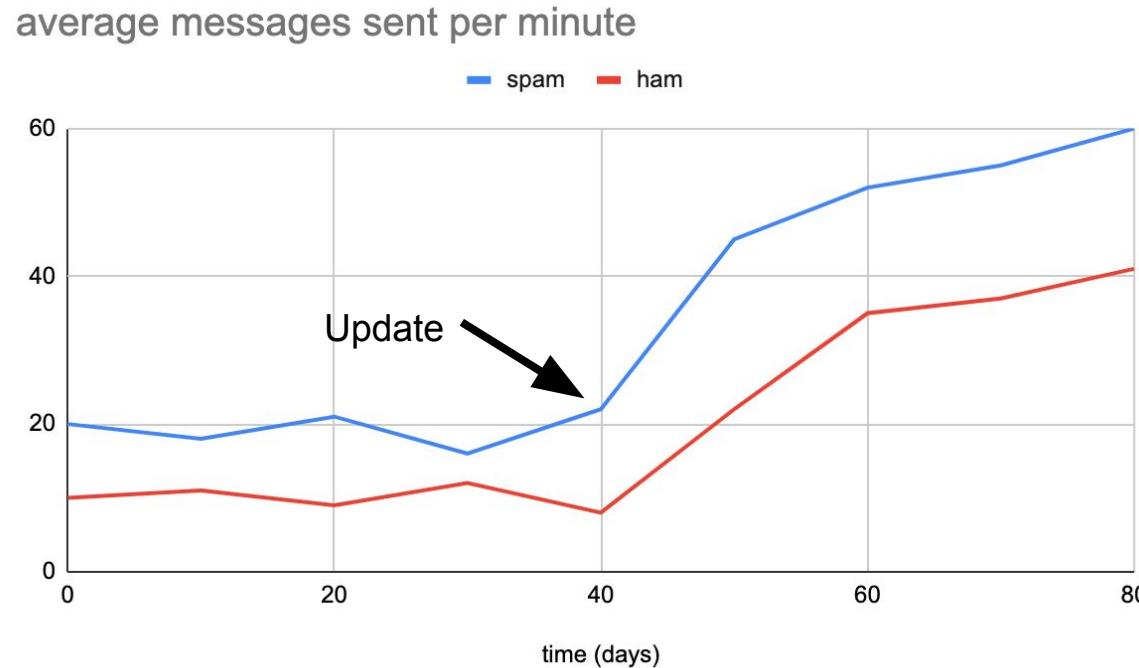
Request



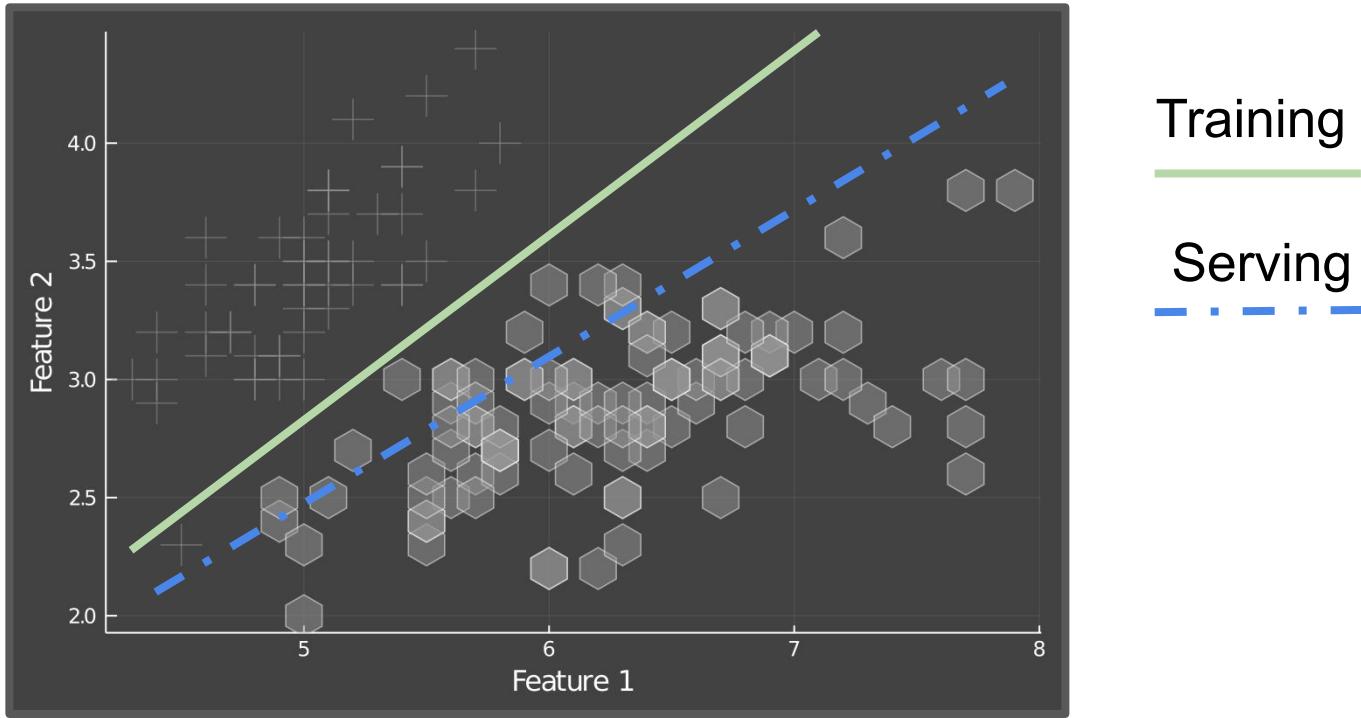
Real-time  
processing



# Model Decay : Data drift



# Performance decay : Concept drift



# Detecting data issues

- Detecting schema skew
  - Training and serving data do not conform to the same schema
- Detecting distribution skew
  - Dataset shift → covariate or concept shift
- Requires continuous evaluation

# Detecting distribution skew

	Training	Serving
Joint	$P_{\text{train}}(y, x)$	$P_{\text{serve}}(y, x)$
Conditional	$P_{\text{train}}(y x)$	$P_{\text{serve}}(y x)$
Marginal	$P_{\text{train}}(x)$	$P_{\text{serve}}(x)$

Dataset shift

$$P_{\text{train}}(y, x) \neq P_{\text{serve}}(y, x)$$

Covariate shift

$$P_{\text{train}}(y|x) = P_{\text{serve}}(y|x)$$

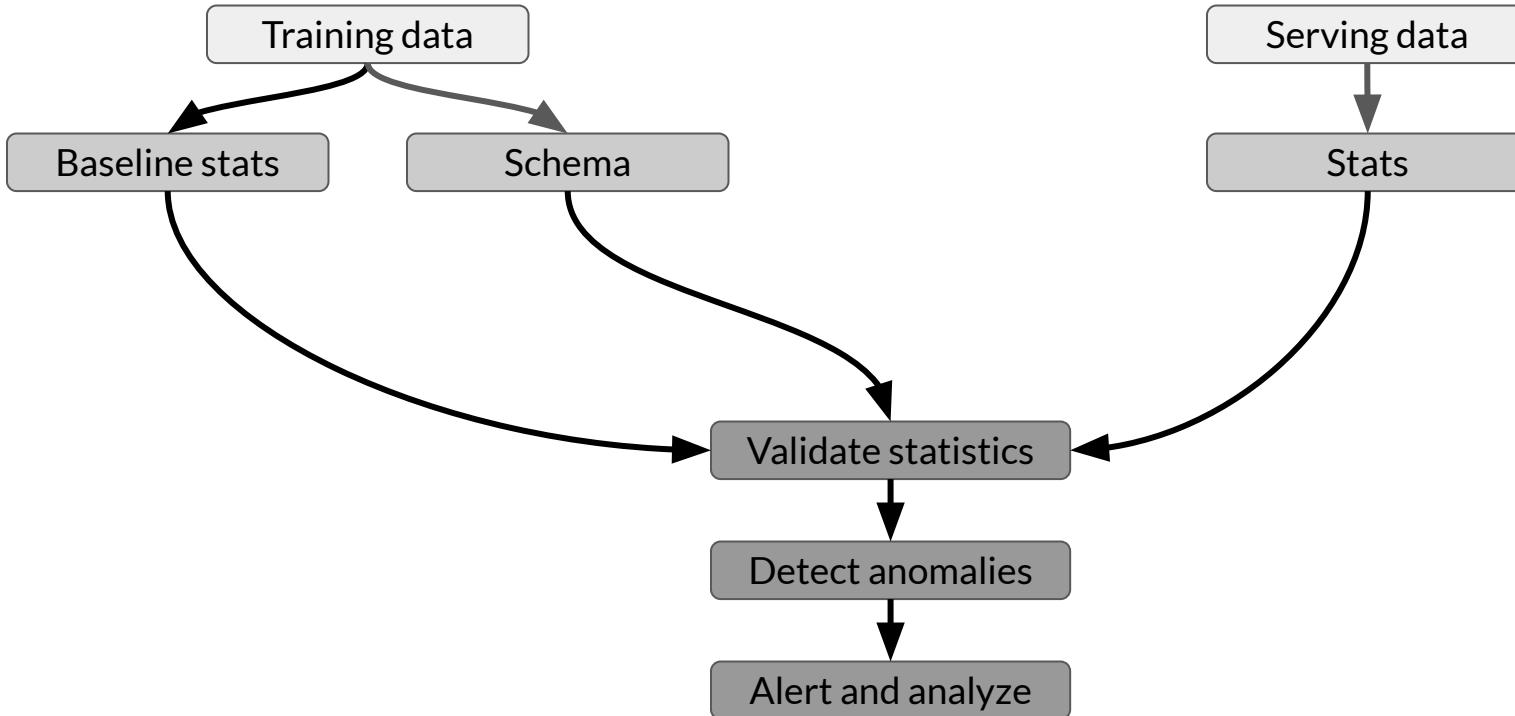
$$P_{\text{train}}(x) \neq P_{\text{serve}}(x)$$

Concept shift

$$P_{\text{train}}(y|x) \neq P_{\text{serve}}(y|x)$$

$$P_{\text{train}}(x) = P_{\text{serve}}(x)$$

# Skew detection workflow





DeepLearning.AI

## Validating Data

---

TensorFlow  
Data Validation

# TensorFlow Data Validation (TFDV)

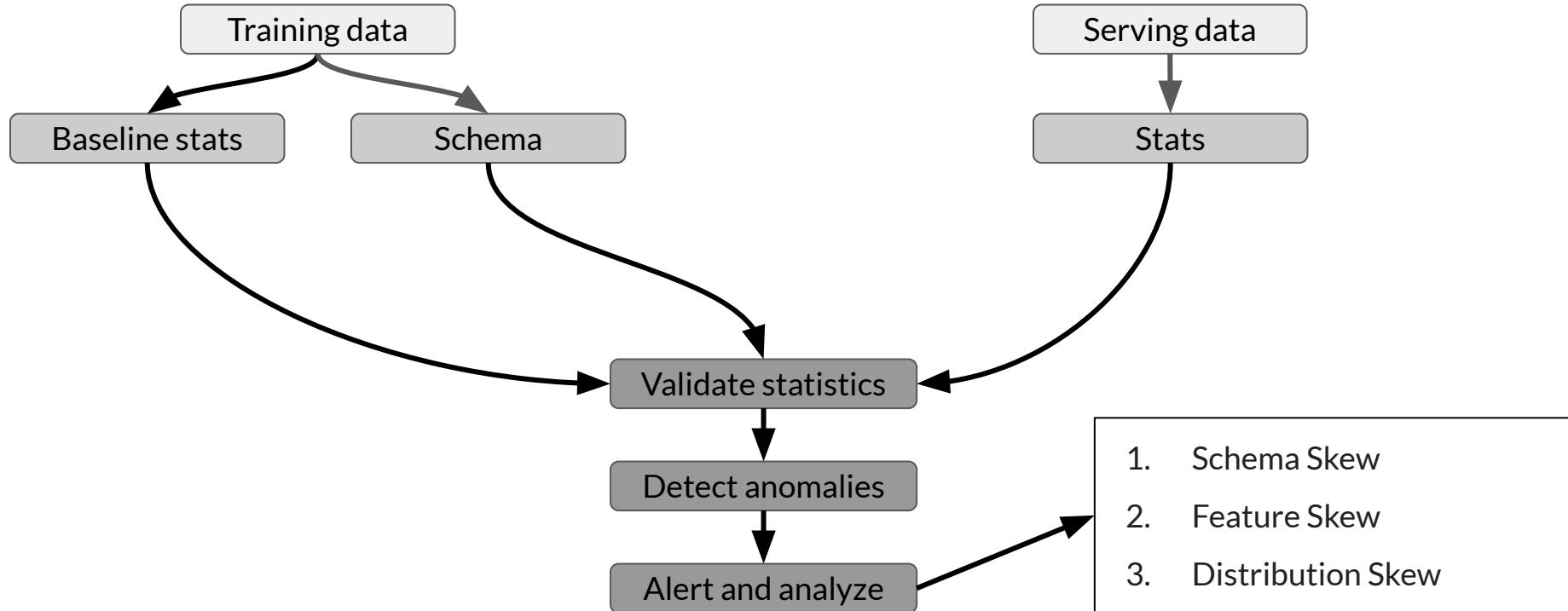


- Understand, validate, and monitor ML data at scale
- Used to analyze and validate petabytes of data at Google every day
- Proven track record in helping TFX users maintain the health of their ML pipelines

# TFDV capabilities

- Generates data statistics and browser visualizations
- Infers the data schema
- Performs validity checks against schema
- Detects training/serving skew

# Skew detection - TFDV

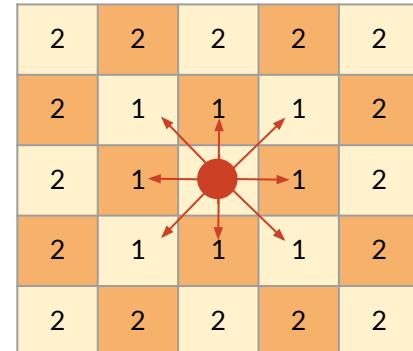


# Skew - TFDV

- Supported for categorical features
- Expressed in terms of L-infinity distance (Chebyshev Distance):

$$D_{\text{Chebyshev}}(x, y) = \max_i(|x_i - y_i|)$$

- Set a threshold to receive warnings



# Schema skew

Serving and training data don't conform to same schema:

- For example, `int != float`

# Feature skew

Training **feature values** are different than the serving **feature values**:

- Feature values are modified between training and serving time
- Transformation applied only in one of the two instances

# Distribution skew

**Distribution** of serving and training dataset is significantly different:

- Faulty sampling method during training
- Different data sources for training and serving data
- Trend, seasonality, changes in data over time

# Key points

- TFDV: Descriptive statistics at scale with the embedded facets visualizations
- It provides insight into:
  - What are the underlying statistics of your data
  - How does your training, evaluation, and serving dataset statistics compare
  - How can you detect and fix data anomalies

# Wrap up

- Differences between ML modeling and a production ML system
- Responsible data collection for building a fair production ML system
- Process feedback and human labeling
- Detecting data issues

**Practice data validation with TFDV in this week's exercise notebook**

**Test your skills with the programming assignment**