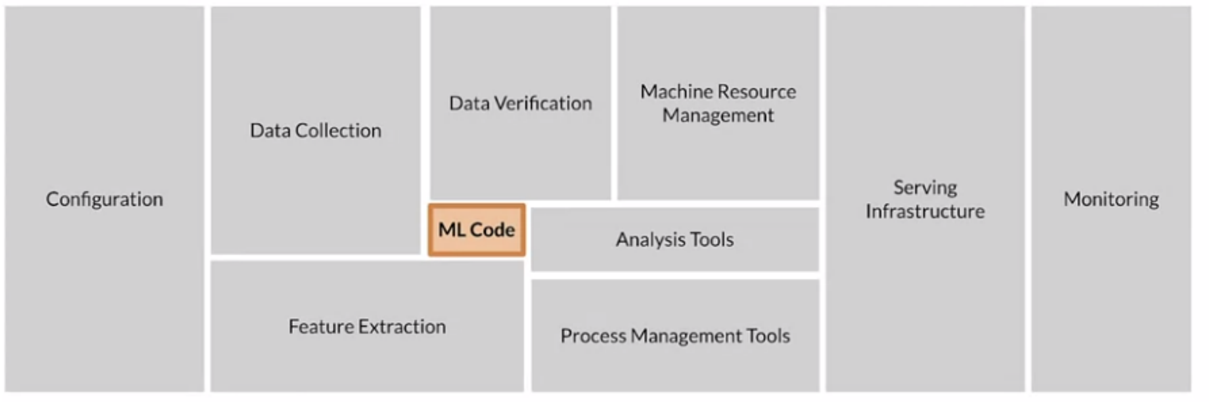
Second Course - Machine Learning Data lifecycle in Production

Overview:

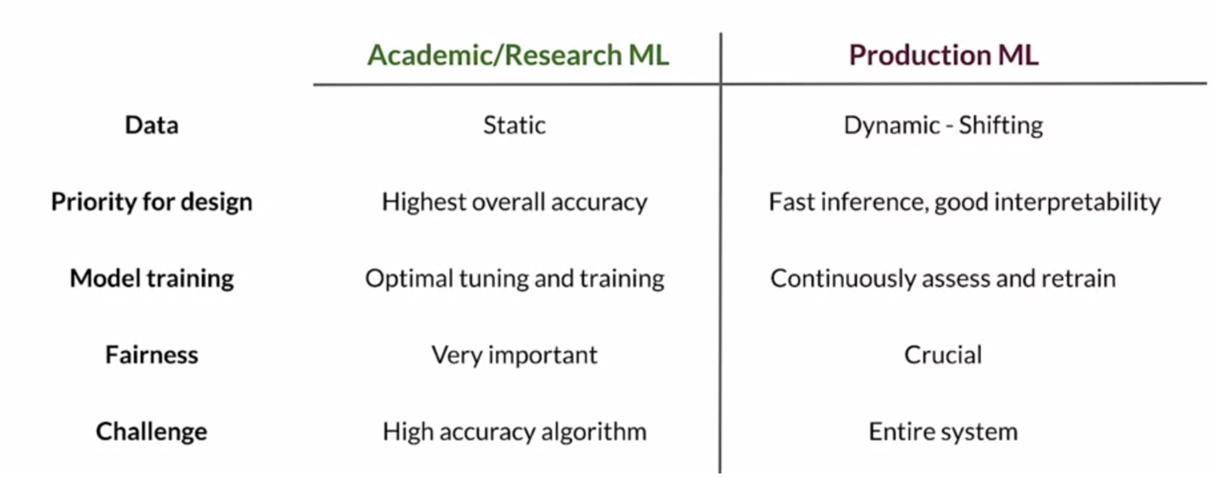
- How to build data pipelines?
- How to maintain data pipelines?
- Explore data journey in tfx frameworks (TensorFlowdatasetssystems

Week - 1 - all about data - Collection, Labeling, and Validating data.

Production ML = ML development + software development



Difference between ml in a research environment and a production environment:



	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

Managing the entire life cycle of data:

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

Accounts for:

- Scalability - Down and Up
- Extensibility

- Configuration
- Consistency & reproducibility
- Safety & Security
- Modularity
- Testability
- Monitoring
- Best practices

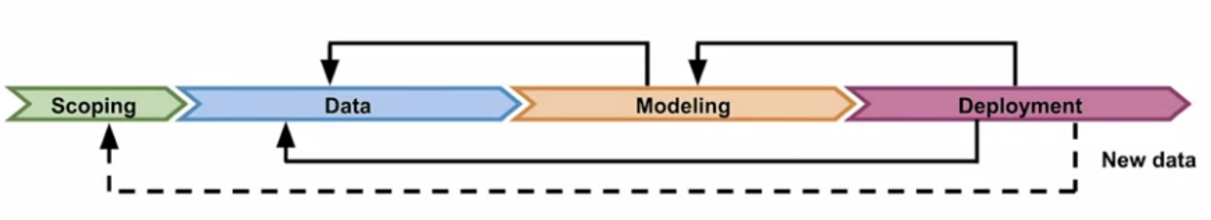
Challenges in production-grade ML

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

ML pipelines:

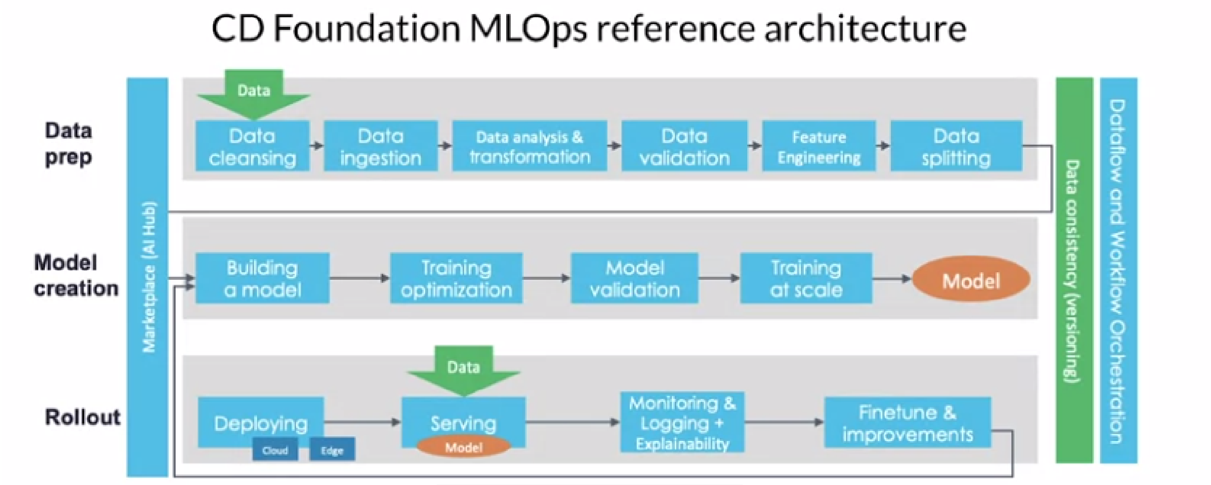
- ML Pipelines
- Directed Acyclic Graphs and Pipeline Orchestration Frameworks
- Intro to TensorFlow Extended (TFX)

ML pipeline is the heart of a machine learning system, it is a software architecture to implement the following process:



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

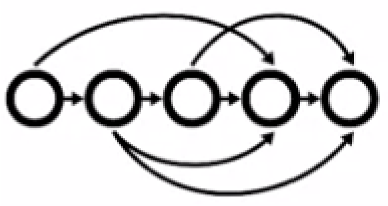
Example:



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, sometimes it can have cycles.



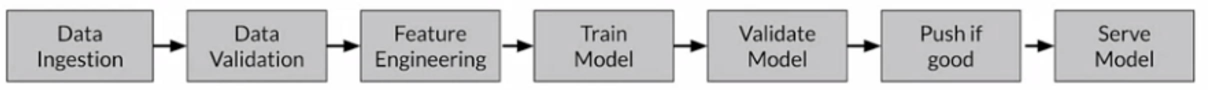
Pipeline orchestration frameworks:

Pipeline orchestrates are responsible for scheduling pipeline's workflows execution.

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

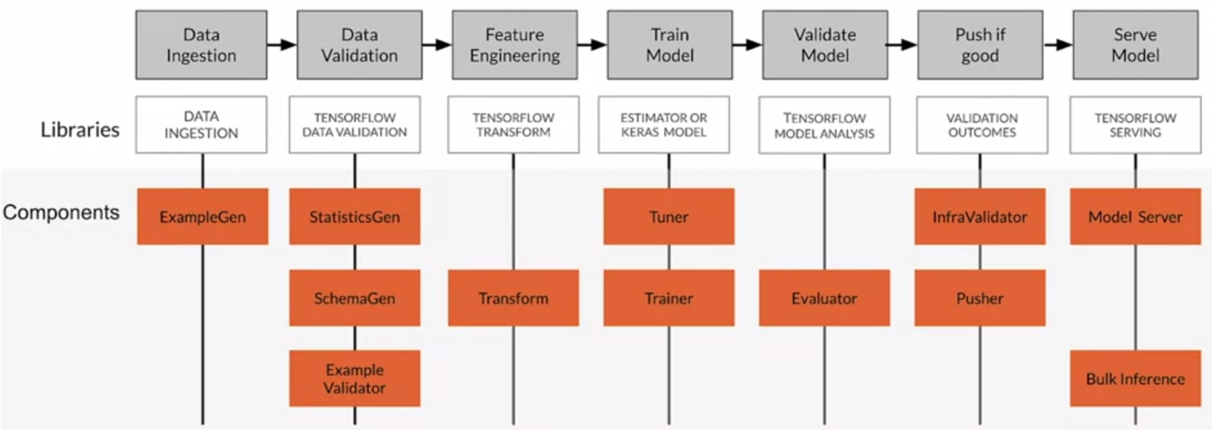
Tensorflow Extended (TFX)

An end-to-end platform for deploying production ML pipelines:



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

TFX production components:

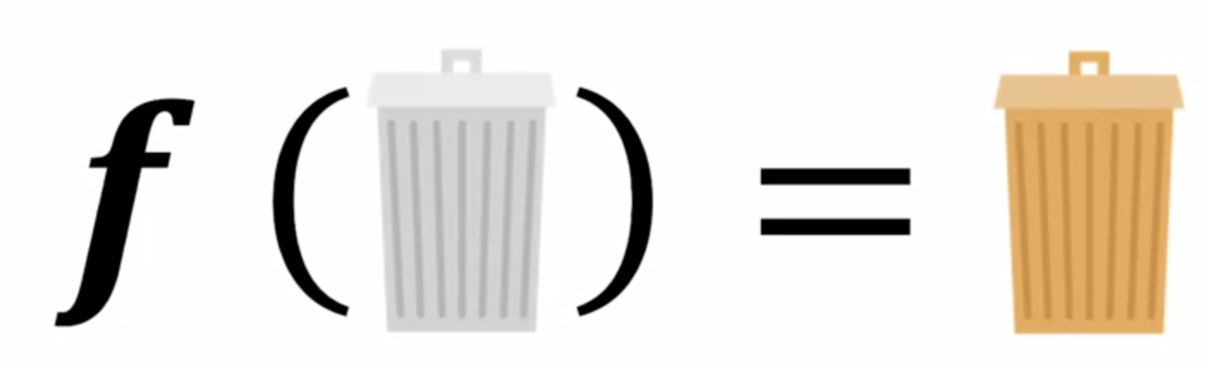


	Data Ingestion	Data Validation	Feature Engineering	Train Model	Validate Model	Push if good	Serve Model
Libraries	DATA INGESTION	TENSORFLOW DATA VALIDATION	TENSORFLOW TRANSFORM	ESTIMATOR OR KERAS MODEL	TENSORFLOW MODEL ANALYSIS	VALIDATION OUTCOMES	TENSORFLOW SERVING
Components	ExampleGen	StatisticsGen SchemaGen Example Validator	Transform	Tuner Trainer	Evaluator	InfraValidator Pusher	Model Server Bulk Inference

Collecting data:

- Importance of data quality
- Data pipeline: data collection, ingestion, and preparation of data. (Automation)
- Data collection and monitoring
- Models aren't magic
- Meaningful data:
    - maximize predictive content
    - remove non-informative data

    - feature space coverage



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

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:

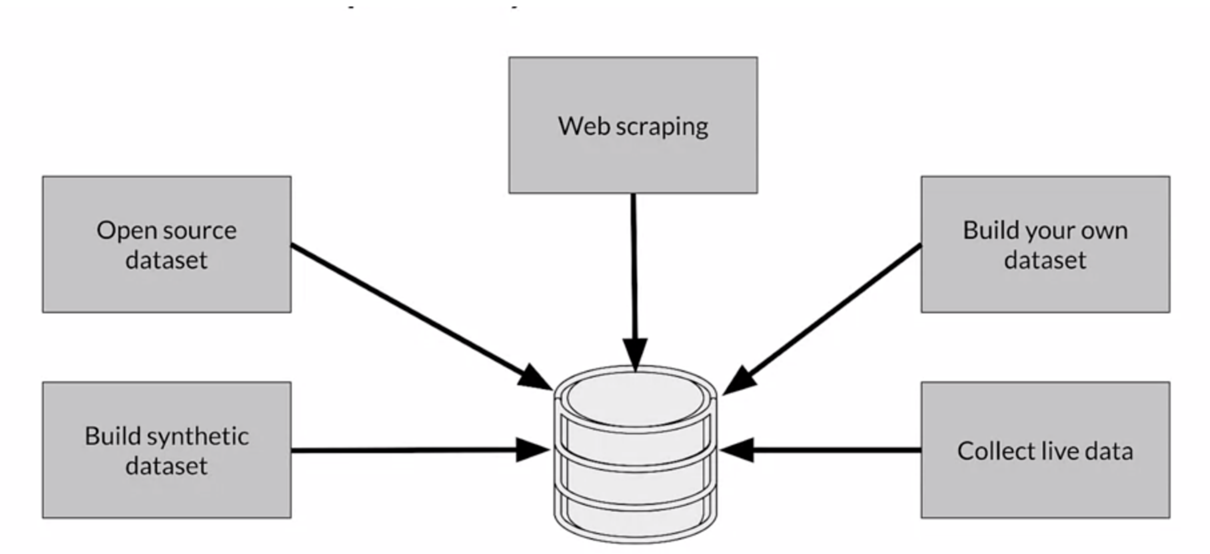
- Inconsistent formatting
    - Is zero “0”, “0.0”, or an indicator of a missing measurement.
- Compounding errors from other ML Models
- Monitor data sources for systems issues and outages

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.

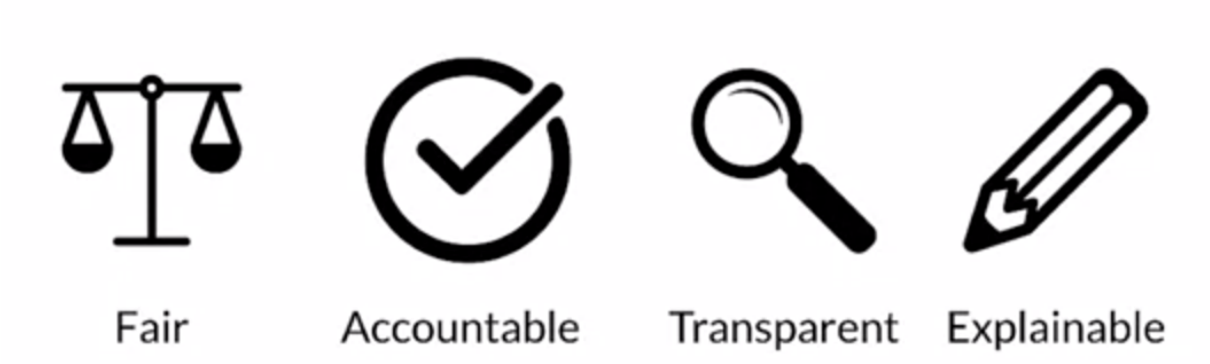
Collecting data:

- Data sourcing
- Data security and User privacy
- Bias and Fairness



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



Failing:

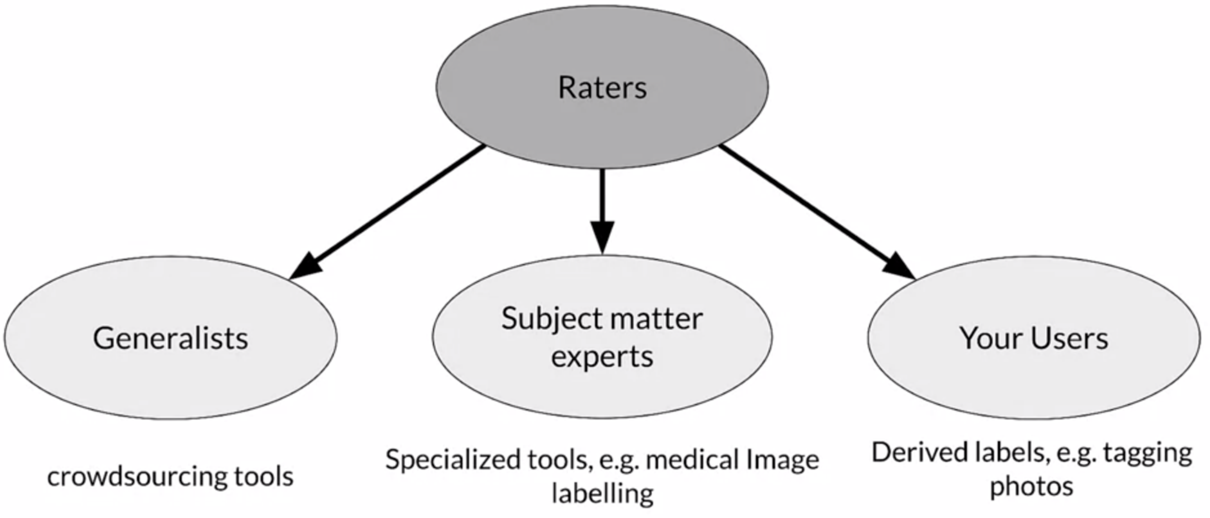
- Representational harm
- Opportunity denial
- Disproportionate product failure
- Harm by disadvantage

Commit to fairness:

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

Reducing bias:

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


Raters
Generalists
Subject matter experts
Your Users
crowdsourcing tools
Specialized tools, e.g. medical Image labelling
Derived labels, e.g. tagging photos

Labeling data:

- Slow: Data drift over time
- Fast: bad sensor, bad software update (problems with the system)
- Gradual:
  - 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
- Data collection:
  - Bad sensor/camera
  - Bad log data
  - Moved or disabled sensors/cameras
- Systems problem:
  - Bad software update
  - Loss of network connectivity
  - Systems down

To keep in mind:

- The data we have is rarely the data you wish you had, bad data quality is always an issue.

- Model objective is nearly always a **proxy** for your business objectives (a way to make some decision but not de decision itself)
- Some percentage of your customers may have a **bad experience**
- THE REAL WORLD WILL NOT STAND STILL, "**the one constant in the world is change"**

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 retrained 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 systems
- Labeling:
  - Direct feedback
  - Weak supervision

Model performance always decays over time, the only thing that changes is the decay ratio, how fast the performance decays?

- Data and concept drift.

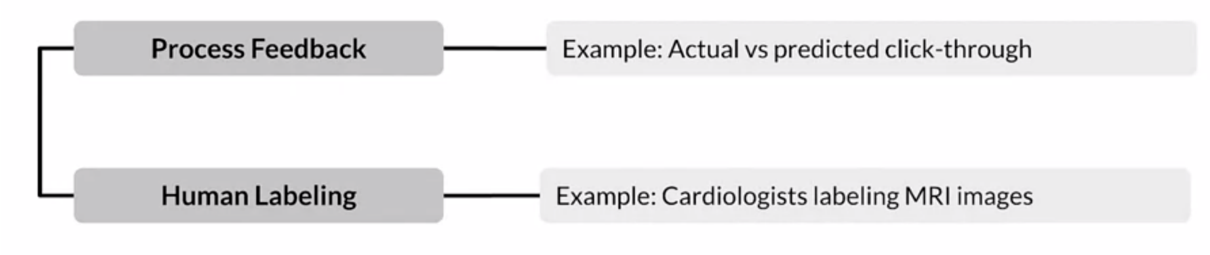
Model retraining helps to improve performance

- Data labeling for changing ground truth and scarce labels.

Data labeling methods:

- Most commons:
  - Process Feedback (Direct Labeling)
  - Human Labeling

- Advance methods:
    - Semi-Supervised Labeling
    - Active Learning
    - Weak Supervision

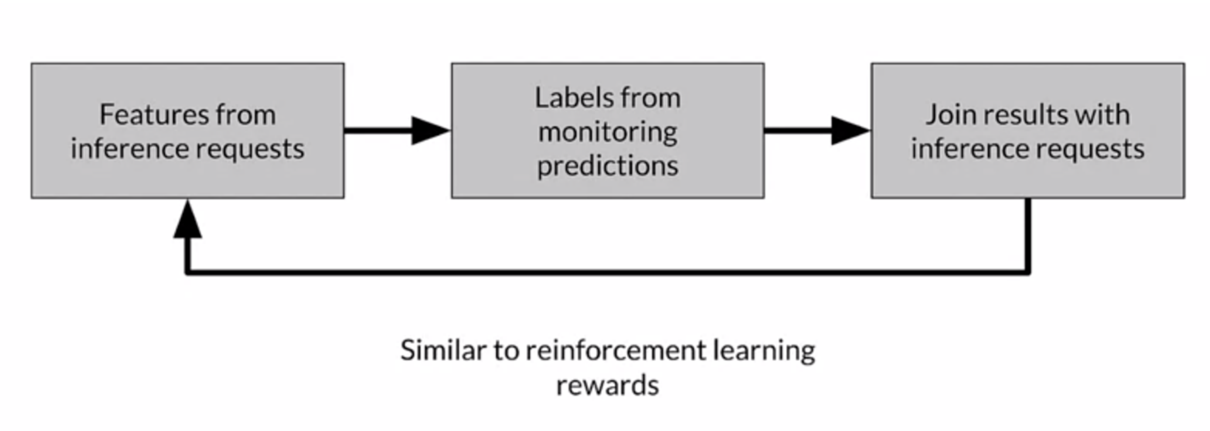

Process Feedback
Example: Actual vs predicted click-through
Human Labeling
Example: Cardiologists labeling MRI images

Why is it important?

Most businesses have a lot of data but it is unlabel, so we cannot apply supervised learning techniques. It is possible to apply unsupervised learning and have good results; but in many cases supervised learning is the best option.

- Using business/organisation available data.
- Frequent model retraining (the frequency depends on domain problem)
- Labeling ongoing and critical process
- Creating a training datasets requires labels (Thinks on how to do this)

Direct labeling: Continuous creation of training dataset


Features from inference requests
Labels from monitoring predictions
Join results with inference requests
Similar to reinforcement learning rewards

Advantages:

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

Disadvantages:

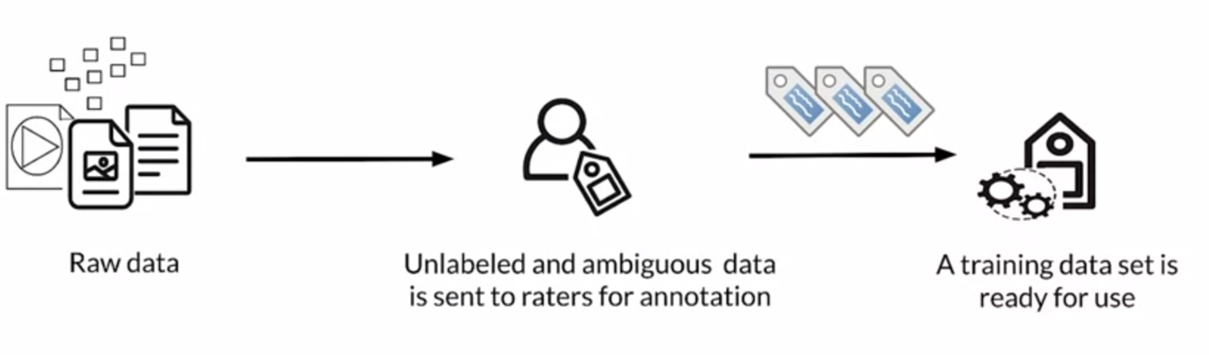
- In many domains is not possible to do this
- Hindered by inherent nature to the problem
- Failure to capture ground truth
- Largely bespoke design

Tools:

- **Logstash:** Free and open source data processing pipeline

    - Ingest 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
- **Cloud logs analytics:**
    - Google cloud logging
    - AWS elasticsearch
    - Azure monitor

Human Labeling (“raters”)


Raw data
Unlabeled and ambiguous data is sent to raters for annotation
A training data set is ready for use

Methodology:

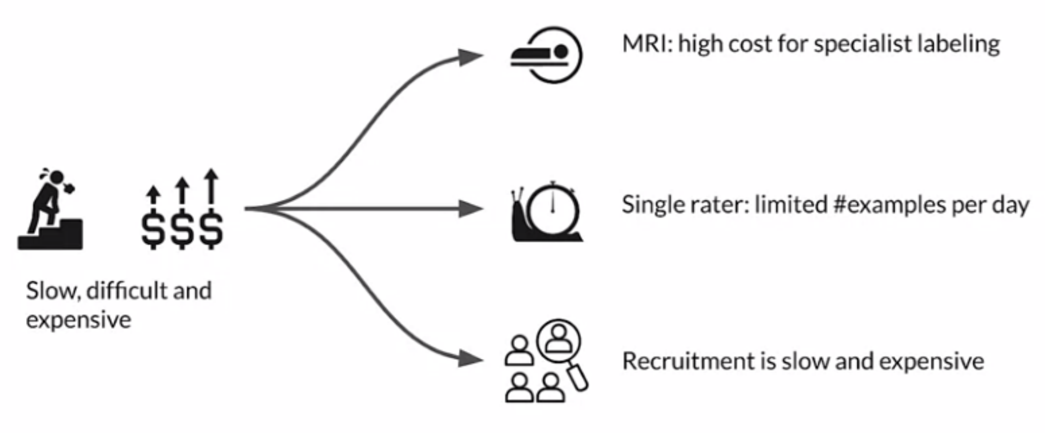
- Unlabeled data is collected
- Human “raters” are recruited
- Instructions to guide raters are created
- Data is divided and assigned to raters
- Labels are collected and conflicts resolved

Advantages:

- More labels
- Pure supervised learning

Disadvantages:

- Complex problems need expertise rates, example: radiologist
- Quality consistency: Many datasets difficult for human labeling
- Slow
- Expensive
- Small dataset curation due to time and costs.

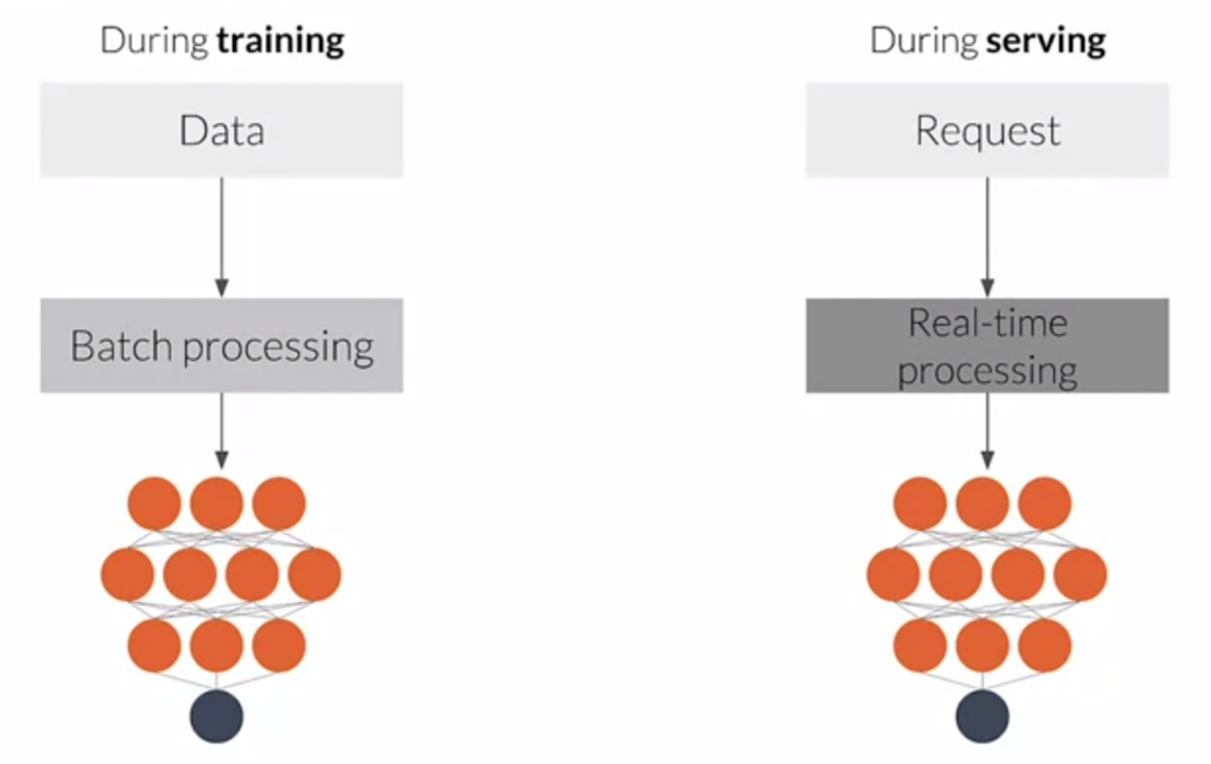

Slow, difficult and expensive
$$$
MRI: high cost for specialist labeling
Single rater: limited #examples per day
Recruitment is slow and expensive

How to know if our data is good or not?

Validating data and data issues

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

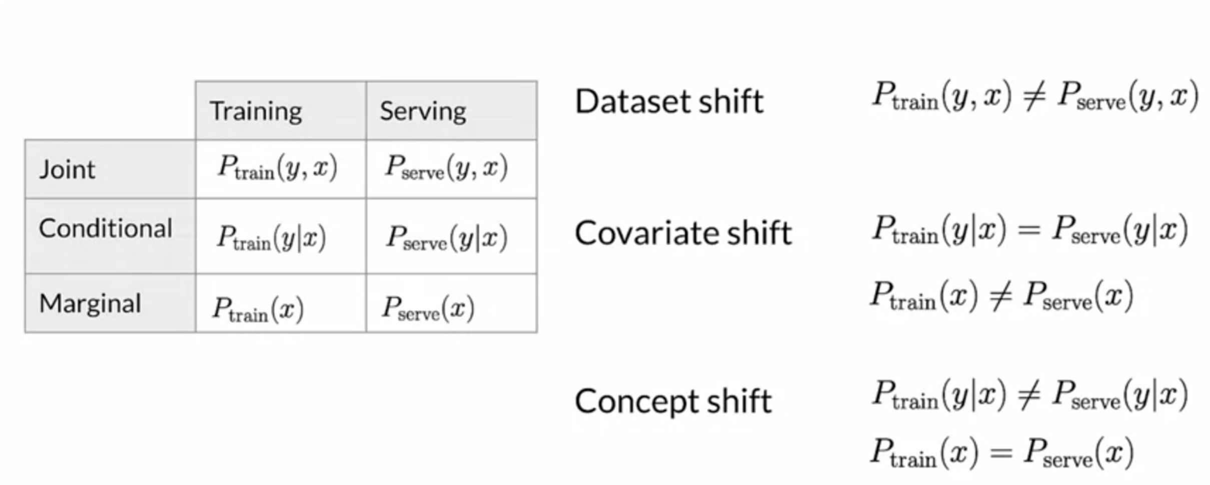
Drift: Changes in data over time, such as data collected once a day
Skew: Difference between two static versions, or different sources, such as training set and serving set



Model decay is usually due to data drift or changes in the world.
Concept drift: Changes in the statistical process of the labels over time, changing of the mapping (X -> Y) (changes of the model)

Detecting data issues:

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



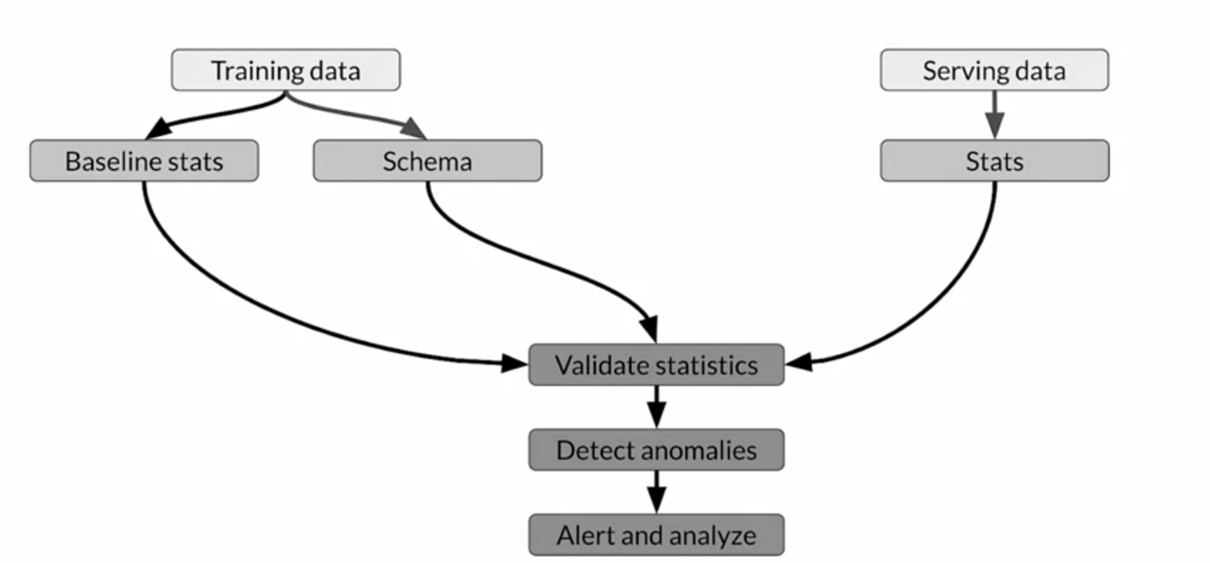
	Training	Serving
Joint	$P_{\text{train}}(y,x)$	$P_{\text{serve}}(y,x)$
Conditional	$P_{\text{train}}(y\|x)$	$P_{\text{serve}}(y\|x)$
Marginal	$P_{\text{train}}(x)$	$P_{\text{serve}}(x)$

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 (baseline)



Tensorflow data validation:

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

Week 1: Collecting, Labeling and Validating Data

This is a compilation of optional resources including URLs and papers appearing in lecture videos. If you wish to dive more deeply into the topics covered this week, feel free to check out these optional references. You won’t have to read these to complete this week’s practice quizzes.

MLops

Data 1st class citizen

Runners app

Rules of ML

Bias in datasets

Logstash

Fluentd

Google Cloud Logging

AWS ElasticSearch

Azure Monitor

TFDV

Chebyshev distance

Papers

Konstantinos, Katsiapis, Karmarkar, A., Altay, A., Zaks, A., Polyzotis, N., … Li, Z. (2020). Towards ML Engineering: A brief history of TensorFlow Extended (TFX). http://arxiv.org/abs/2010.02013

Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2020). Challenges in deploying machine learning: A survey of case studies. http://arxiv.org/abs/2011.09926

ML code fraction:

Sculley, D., Holt, G., Golovin, D., Davydov, E., & Phillips, T. (n.d.). Hidden technical debt in machine learning systems. Retrieved April 28, 2021, from Nips.cc https://papers.nips.cc/paper/2015/file/86df7dcfd896fcaf2674f757a2463eba-Paper.pdf