2024 LINC 3.0 + GSAI Joint Seminar
Industrial AI - Best Practices in Semiconductor Manufacturing

Sunghee Yun

Co-founder / CAIO - AI Technology & Product Strategy

Erudio Bio, Inc.

About Speaker

- *Co-founder / CAIO - AI Technology & Product Strategy @ Erudio Bio, Inc., CA, USA*
- Advisory Professor, Electrical Engineering and Computer Science @ DGIST, Korea
- Adjunct Professor, Electronic Engineering Department @ Sogang University, Seoul
- Technology Consultant @ Gerson Lehrman Gruop (GLG), NYC, USA
- *Co-founder / CTO & Chief Applied Scientist @ Gauss Labs Inc., Palo Alto, USA $\sim$ 2023*
- Senior Applied Scientist @ Amazon.com, Inc., Vancouver, BC, Canada $\sim$ 2020
- Principal Engineer @ Software R&D Center of Samsung DS Division, Korea $\sim$ 2017
- Principal Engineer @ Strategic Marketing Team of Memory Business Unit $\sim$ 2016
- Principal Engineer @ Memory DT Team of DRAM Development Lab. $\sim$ 2015
- Senior Engineer @ CAE Team of Samsung Semiconductor $\sim$ 2012
- M.S. & Ph.D. - Electrical Engineering (EE) @ Stanford University $\sim$ 2004
- B.S. - Electrical Engineering (EE) @ Seoul National University $\sim$ 1998

Exciting career journey

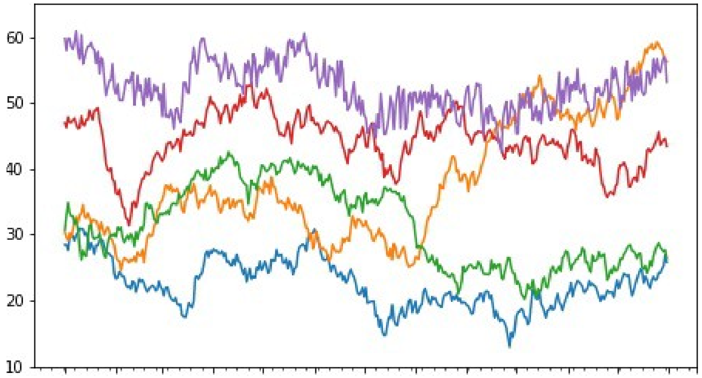
- B.S. - EE @ SNU & M.S. & Ph.D. - EE @ Stanford Univ.
  - *Convex Optimization - theory / algorithms / applications - under supervision of Prof. Stephen P. Boyd*
  - connectionists were depressed . . .
- Principal Engineer @ Memory Design Technology Team
  - develop variety of optimization tools for & and partner with *DRAM / NAND Flash / PE / Test Teams*
- Senior Applied Scientist @ Amazon
  - *S-Team Goal project (Jeff Bezos's project) - Amazon shopping app customer engagement opt using AI - increased by 200MM USD*
- Co-founder / CTO & Chief Applied Scientist @ Gauss Labs Inc.
  - *lead develop & productionize industrial AI products, team building*
  - market, product & investment strategies
- Co-founder / CAIO - AI Technology & Product Strategy @ Erudio Bio, Inc.
  - *biotech AI technology & products, team building*

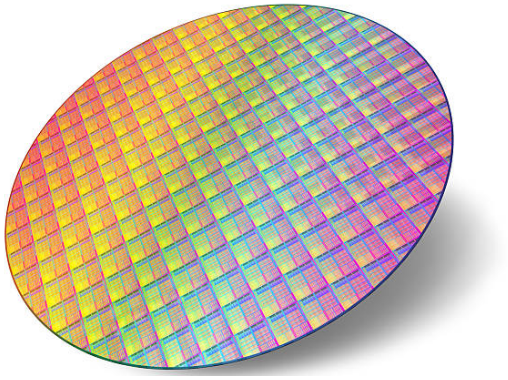
Today

- Machine learning algorithms for time-series (TS) data
  - supervised learning for time-series
  - time-series anomaly detection
  - credibility interval evaluation - prediction of uncertainty of predictions
- TS learning applications in manufacturing
  - virtual metrology
  - root cause analysis
- Manufacturing AI Software System

Why TS learning?

- all data coming from manufacturing are TS data
  - sensor data, process times, image & other measurements, . . .
- amount of TS data is huge
  - tera-scale data per day generated in semiconductor manufacturing lines





Machine Learning for TS

TS data

- definition of times-series:

$$x : T \to \mathbf{R}^n \text{ where } T = \{\ldots, t_{-2}, t_{-1}, t_0, t_1, t_2, \ldots\} \subseteq \mathbf{R}$$

- example: material measurements: when $n = 4$

$$x_t = \left[\begin{array}{c} \text{thickness}(t) \\ \text{temperature}(t) \\ \text{pressure}(t) \\ \text{feature\_size}(t) \end{array}\right]$$

- for (semi-)supervised learning, we assume two time series

$$x : T \to \mathbf{R}^n \text{ and } y : T \to \mathbf{R}^m$$

Time index

- time index does not have to be *time* index

- general definition

$$x : T \to \mathbf{R}^n \text{ where } T = \{\ldots, s_{-2}, s_{-1}, s_0, s_1, s_2, \ldots\}$$

  where $\cdots < s_{-1} < s_0 < s_1 < \cdots$ defines *an* ordering ($e.g.$, total ordering)

- for example, x_s and $y(s)$ can represent the features and target values for a processed material ($e.g.$, wafer in semiconductor manufacturing), s, where they are not measured at the same time

- (throughout this talk, though, we will use time-index)

Supervised Learning

Supervised learning for TS

- canonical problem:

$$\begin{array}{rl}\text{(stochastically) predict} & y_{t_k} \\ \text{given} & x_{t_k}, x_{t_{k-1}}, \ldots, y_{t_{k-1}}, y_{t_{k-2}}, \ldots\end{array}$$

- various methods exist - depend assumptions on data
  - $e.g.$, if assume joint probability distribution, optimal solutions exist in LSE sense

- however, will *not* make such assumptions

Problem formulation

- canonical problem formulation:

$$\begin{array}{ll}\text{minimize} & \sum_{k=1}^{K} w_{K-k}\, l(y_{t_k}, \hat{y}_{t_k}) \\ \text{subject to} & \hat{y}_{t_k} = g_k(x_{t_k}, x_{t_{k-1}}, \ldots, y_{t_{k-1}}, y_{t_{k-2}}, \ldots)\end{array}$$

  where

  - $g_1, g_2, \ldots : \mathcal{D} \to \mathbf{R}^m$ - optimization variables
  - $\mathcal{D} = \mathbf{R}^n \times \mathbf{R}^n \times \cdots \times Q \times Q \times \cdots$ - domain of g_k where $Q = \mathbf{R}^m \cup \{\text{null}\}$
  - $l : \mathbf{R}^m \times \mathbf{R}^m \to \mathbf{R}_+$ - loss function
  - w_i - (nonincreasing) weight on loss

- no label is given for some k, *i.e.*, $y(t_k) = \text{null}$

ML solution candidates

- ignore temporal dependency - $\hat{y}_{t_k} = g(x_{t_k})$
  - supervised learing such as DL ($e.g.$, MLP), decision trees
  - classiscal statistical learning such as lasso, ridge regression, partial least squares
  - boosting algorithms such at `XGBoost`

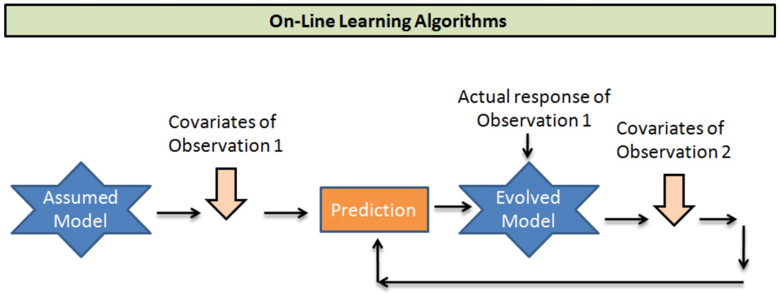
- consider temporal dependency - sequential MLs
  - RNN-base: LSTM, GRUs
  - attention mechanism, $e.g.$, classical attention-type or Transformer-type architectures

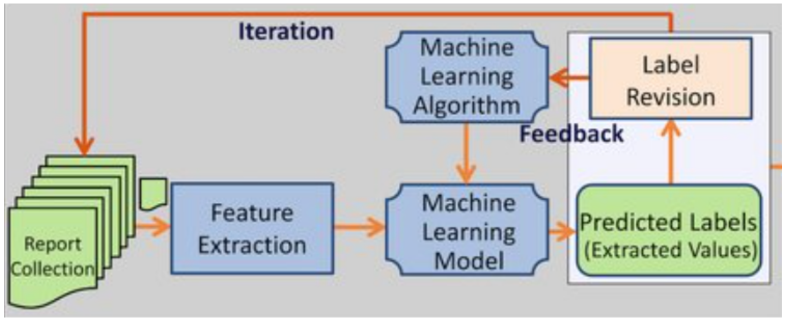
Difficulties with manufacturing applications

- for many manufacturing applications
  - exist shift & drift
    - $p(x_{t_k}, x_{t_{k-1}}, \ldots)$ changes over time
    - $p(y_{t_k} | x_{t_k}, x_{t_{k-1}}, \ldots, y_{t_{k-1}}, y_{t_{k-2}}, \ldots)$ changes over time
  - hence, traditional off-line training *seldom* works!
  - also, DL-type algorithms do not work, either
    - shift/drift → data got stale quickly, effectively less data
    - hence, data hungry DL not fit well
- have been verified by many instances and trial-and-errors

Practical approach

- learned from many trial-and-errors that online learning works!
- online learning
  - update your model g_k after observing
    - ∗ the current and past x's; $x_{t_k}, x_{t_{k-1}}, \ldots$
    - ∗ the past y's; $y(t_{k-1}), y(t_{k-2}), \ldots$


On-Line Learning Algorithms
Assumed Model
Covariates of Observation 1
Prediction
Actual response of Observation 1
Evolved Model
Covariates of Observation 2



One Solution - prediction based on experts' advice

- assume p_k experts: $f_{i,k} : \mathbf{R}^n \to \mathbf{R}^m$ $(i = 1, 2, \ldots, p_k)$ for each time step, t_k
  - $f_{i,k}$ can be DNN, (online) ridge regression, or other statistical learning algorithms
- model predictor at time step k, $g_k : \mathbf{R}^n \to \mathbf{R}^m$ as weighted sum of experts:

$$g_k = w_{1,k} f_{1,k} + w_{2,k} f_{2,k} + \cdots + w_{p_k,k} f_{p_k,k} = \sum_{i=1}^{p_k} w_{i,k} f_{i,k}$$

- online learning and inferencing procedure:
  - if $y(t_k) \neq$ null, *i.e.*, new observation available, update $f_{i,k}$ and $w_{i,k}$
  - if $y(t_k) =$ null, *i.e.*, no observation is available, predict $\hat{y}_k(t_k) = g_k(x_{t_k})$

Algorithm description

- set $k = 0$
  - given $(x_{t_k}, y(t_k))$, predict $\hat{y}_{i,k}(t_k) = f_{i,k}(x_{t_k})$
    - if $y(t_k) \neq$ null
      - predict $\hat{y}(t_k) = y(t_k)$
      - update $f_{i,k} \to f_{i,k+1}$ based on $(x_{t_k}, y(t_k))$
      - update $w_{i,k} \to w_{i,k+1}$ based on prediction error, $y(t_k) - \hat{y}_{i,k}(t_k)$
    - if $y(t_k) =$ null
      - predict $\hat{y}(t_k) = g_k(x_{t_k}) = \sum_{i=1}^{p} w_{i,k}\hat{y}_{i,k}(t_k)$
      - update $f_{i,k+1} := f_{i,k}$ (not update)
      - update $w_{i,k+1} := w_{i,k}$ (not update)
- udpate $k := k + 1$ and repeat

What about unlabelled data?

- semi-supervised learning via representation learning

- extension of model update rules for unlabelled data - for some statitsical learning methods

- *NO* right answer for this!

  – pre-cost-benefit analysis strongly recommended - actually *must-do*

Credibility intervals

- prediction of uncertainty of prediction

- every point prediction is wrong!

  – $\mathbf{Prob}(\hat{y}_t = y_t) = 0$

- reliability of prediction matters
  – *none* literature deals with this (properly)

- critical for our customers, $e.g.$, *downstream applications*
  – if used for APC, need to know when it should be used
  – sometimes, *more crucial than algorithm accuracy*

Credibility intervals

- multiple criteria
  - probability of true value falling into an interval: for fixed $a > 0$

$$\mathbf{Prob}(|Y_k - \hat{Y}_k| < a) = \mathbf{Prob}(Y_k \in (\hat{Y}_k - a, \hat{Y}_k + a))$$

  - predictive distribution size: find $a > 0$ such that

$$\mathbf{Prob}(|Y_k - \hat{Y}_k| < a) = 90\%, \ \ e.g.$$

  - distribution of Y_k: find PDF of Y_k
- our solution - Bayesian inference
  - given initial distribution or prior, $p(x)$
  - update $p(x)$ with new data using Bayesian inference

Bayesian approach for credibility intervals

- assume conditional distribution ith predictor parameterized by $\theta_{i,k} \in \Theta$

$$p_{i,k}(y(t_k)|x_{t_k}, x_{t_{k-1}}, \ldots, y(t_{k-1}), y(t_{k-2}), \ldots) = p_{i,k}(y(t_k); x_{t_k}, \theta_{i,k})$$

  - depends on prior & current input, $i.e.$, $\theta_{i,k}$ & x_{t_k}

- update $\theta_{i,k+1}$ from $\theta_{i,k}$ after observing true $y(t_k)$ using Bayesian rule

$$p(w; \theta_{i,k+1}) := p(w|y(t_k); x_{t_k}, \theta_{i,k}) = \frac{p(y(t_k)|w, x_{t_k})p(w; \theta_{i,k})}{\int p(y(t_k)|w, x_{t_k})p(w; \theta_{i,k})dw}$$

- *if* $p(\cdot; \theta)$ is conjugate prior, can update $\theta_{i,k}$ very efficiently in online manner *within fraction of milliseconds*

Credibility interval evaluation for expert-based online learning

- reminder: online learning method based on expert advice is given by

$$g_k = w_{1,k} f_{1,k} + w_{2,k} f_{2,k} + \cdots + w_{p,k} f_{p,k} = \sum_{i=1}^{p} w_{i,k} f_{i,k}$$

- assume that $f_{i,k}$ is parameterized by $\theta_{i,k}$
- *if* we can calculate $p(\theta_{i,k})$
  – can evaluate the *predictive distribution*

$$p_{i,k}(y(t_k); x_{t_k}) = \int p(y; x_{t_k}, \theta_{i,k}) p(\theta_{i,k}) d\theta_{i,k}$$

- problem to solve: evaluate distribution of g_k given $p_{i,k}$

- independent case: if $p_{1,k}, \ldots, p_{p,k}$ are (statistically) independent, then PDF of $g_k(x_{t_k})$ can be calculated by

$$\frac{p_{1,k}(y/w_{1,k}; x_{t_k})}{w_{1,k}} \star \cdots \star \frac{p_{p,k}(y/w_{p,k}; x_{t_k})}{w_{p,k}}$$

- Gaussian case: $p_{1,k}, \ldots, p_{p,k}$ are Gaussians with correlation coefficient matrixa R, *i.e.*,

$$p_{i,k} \sim \mathcal{N}(\mu_{i,k}, \sigma_{i,k}^2)$$

$$R = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} & \cdots & \rho_{1,p} \\ \rho_{1,2} & 1 & \rho_{2,3} & \cdots & \rho_{2,p} \\ \rho_{1,3} & \rho_{2,3} & 1 & \cdots & \rho_{3,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{1,p} & \rho_{2,p} & \rho_{3,p} & \cdots & 1 \end{bmatrix} \in \mathbf{R}^{p \times p}$$

- then g_k is also Gaussian

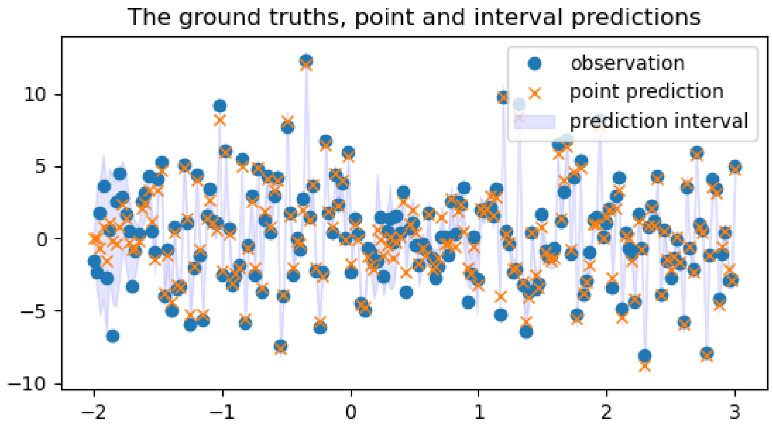
$$\mathcal{N}(w_k^T \mu_k, w_k^T \,\mathbf{diag}(\sigma_k) R \,\mathbf{diag}(\sigma_k) w_k)$$

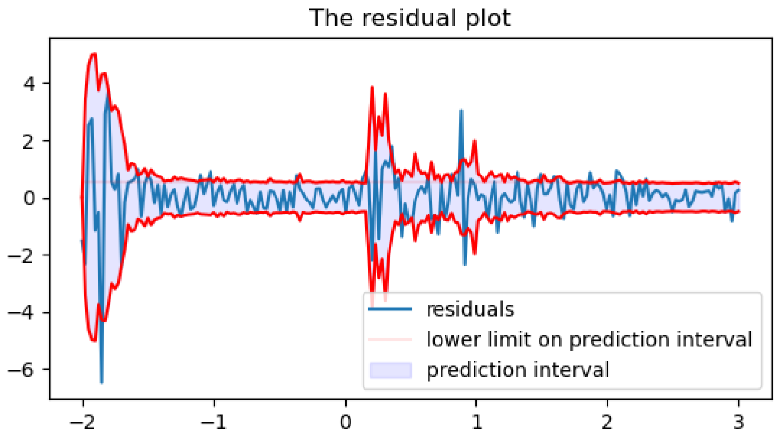
where

$$\begin{array}{rcl}
w_k & = & \left[\begin{array}{ccc} w_{1,k} & \cdots & w_{p,k} \end{array}\right]^T \in \mathbf{R}^p \\
\mu_k & = & \left[\begin{array}{ccc} \mu_{1,k}(x_{t_k}) & \cdots & \mu_{p,k}(x_{t_k}) \end{array}\right]^T \in \mathbf{R}^p \\
\sigma_k & = & \left[\begin{array}{ccc} \sigma_{1,k}(x_{t_k}) & \cdots & \sigma_{p,k}(x_{t_k}) \end{array}\right]^T \in \mathbf{R}^p
\end{array}$$

Real application

- observe
  - initially predictor *not sure* about its prediction
  - after a while, the *credibility interval (CI)* converges
  - when shift happens, CI increases (as it should be)
- this information *crucial for downstream applications*, *e.g.*, process control

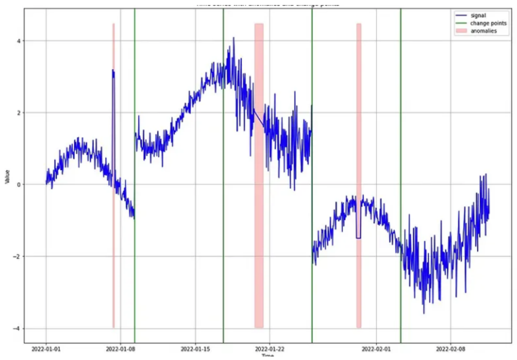


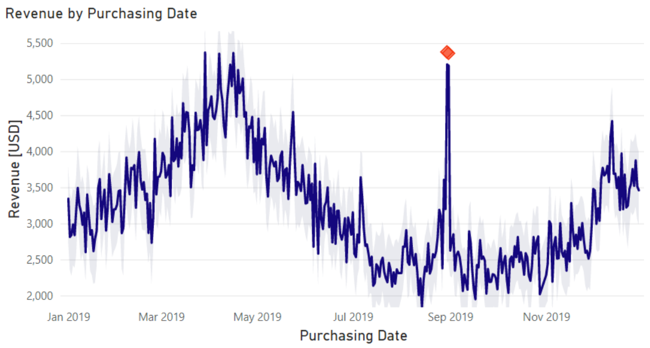


Anomaly Detection

TS anomaly detection problems

- types of anomaly detection problems - given $x : T \to \mathbf{R}^n$
  - point anomaly - find x_{t_k} considerably different from other data
  - segment anomaly - find k_1 and k_2 s.t. TS segment $x_{t_k}|_{k=k_1}^{k_2}$ is considerably different from other data
  - sequence anomaly - given $x^1, \ldots, x^n : T \to \mathbf{R}$, find x^i considerably different from other TSs





TS segment anomaly detection algorithm

- use classification - given $x_{t_j}|_{j=k-l+1}^{k}$, $i.e.$, segment of length, l
  - training:
    - one classifier, c, and, p feature extractors, f_i
    - for each k
      - extract p features using extractors - $y_{i,k} = f_i \left(x_{t_j}|_{j=k-l+1}^{k} \right)$
      - train the classifier, c, with $(y_{1,k}, 1)$, $(y_{2,k}, 2)$, . . . , $(y_{p,k}, p)$, as training data
  - inferencing:
    - given new segment $x_{t_j}|_{j=k-l+1}^{k}$, apply c to the extracted features, $y_{i,k}$
    - if substantically different from $(1, 2, \ldots, p)$, it is anomaly
      - "difference" quantified by some *anomaly score*, $e.g.$, KL divergence or entropy

Other TS anomaly detection methods

- using matrix factorizating similar to topic modeling
- classification and regression trees (CART)
- detection using forecasing
- clustering-based anomaly detection
- autoencoders

ML Applications in Manufacturing

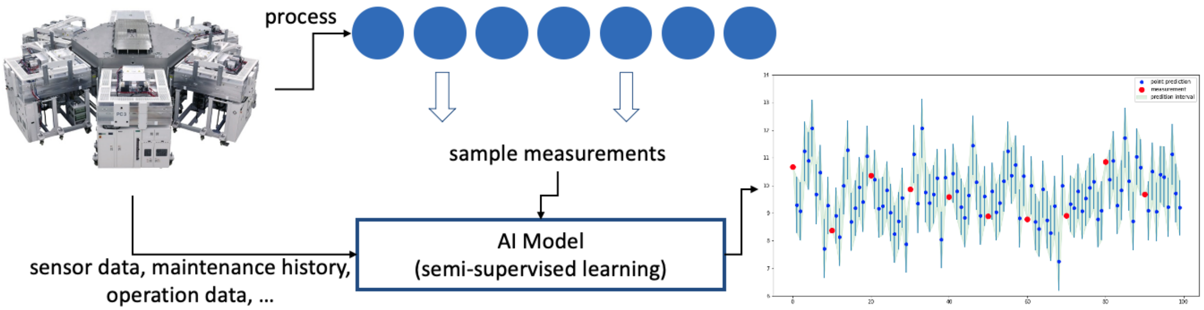
Virtual Metrology

Virtual metrology

- in many cases, we cannot measure all processed materials for fundamental reasons
  - measurement equipment is too expensive
  - no room in the factory for many measurement equipment
  - measuring every materials hinders production speed inducing low throughput
- thus, we do sampling (with very low smapling rate)
  - in semiconductor manufacturing line, avarage sampling rate is less than 1%
- problem: we want to predict the measurement of unmeasured material using indirect signals such as
  - sensor data, maintenance history, operation data, . . .

VM

- difficulties
  - covariate shift and concept drift due to, *e.g.*, preventive maintenance, chamber contamniation, *etc.*
  - hence, data becomes stale quickly
- MUE provides the uncertainty level of our prediction, *i.e.*, *credibility intervals*
  - process engineers can judge when they can trust the predictions by how much
  - we can monitor performance degradation



process

sample measurements

AI Model
(semi-supervised learning)

sensor data, maintenance history,
operation data, ...

Applications of VM

- why do we even develop VM?

- focus on the values we deliver to out customers; want VM to be used for
  - process (feedback) control → average matters
  - detecting equipment out-of-control status → anomalies matters
  - detecting root caues for yield drop
  - predicting (future) yield

Root Cause Analysis

Root cause analysis by anomaly detection

- background: statistical process control (SPC)
  - conventional old method used in manufacturing (since 1950's)
  - monitor measurement and alert when things go wrong
  - things go wrong defined by rules; examples:
    - measument out of $(\mu - 3\sigma, \mu + 3\sigma)$,
    - three consecutive measurements out of $(\mu - 2\sigma, \mu + 2\sigma)$
- our problem: when SPC alarm goes off, find the responsible (chamber in) equipment

Root cause analysis by anomaly detection

- two methods exist: (1) segment anomaly detection and (2) sequence anomaly detection

- two types of data exist: (1) sensor data and (2) processed material measurement data

- problems: given TS data $x_e(t_0), x_e(t_1), \ldots$ for each entity $e \in E$ (entity refers to equipment, chamber, station, $etc.$)
  - find entity e that shows abnormal behavior using segment anomaly detection
  - find entiry e that is different from other entities using sequence anomaly detection

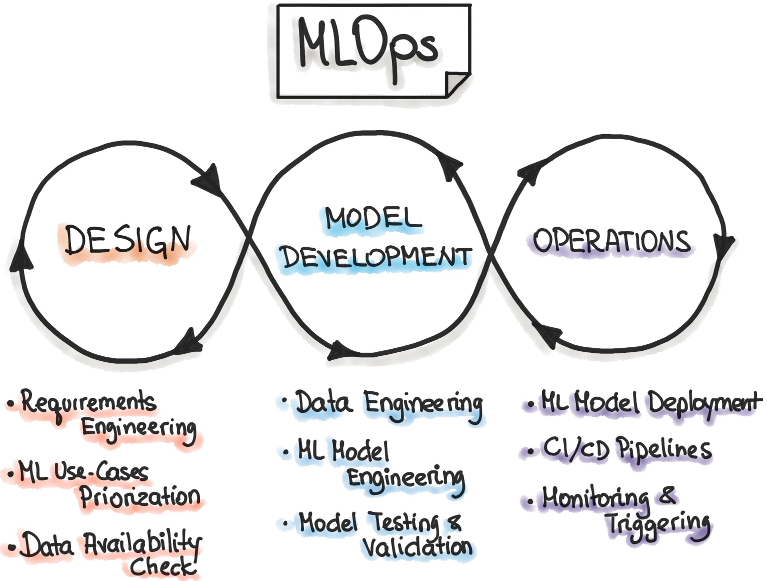
Manufacturing AI Productionization

Minimally required (*i.e.*, necessary) efforts

- MLOps - for CI/CD
- data preprocessing - missing values, inconsistent names, difference among different systems
- feature extraction & selection
- monitoring & retraining
- notification, via messengers or emails
- mainline merge approvals by humans
- data latency, data reliability, & data availability

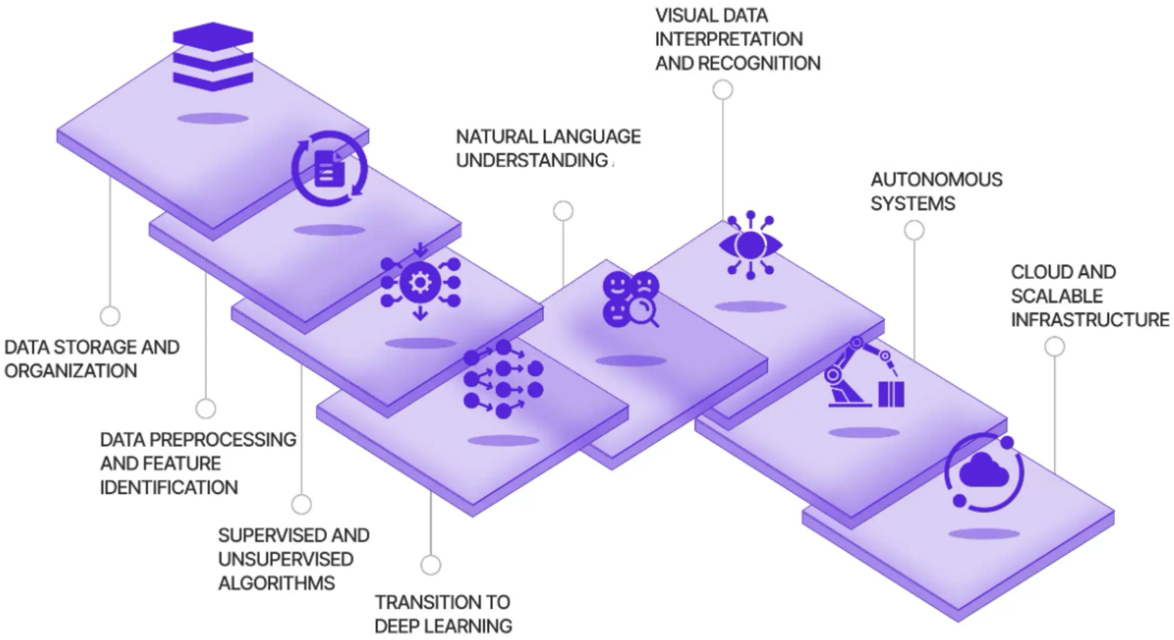
MLOps

- environment for flexible and agile exploration - exploratory data analysis (EDA)
- fast & efficient iteration of algorithm selection, experiements, & analysis
- training / validation (or dev) / test data split critical!
- seamless productionization from, *e.g.*, Jupyter notebook to production-ready code
- monitorning, good! metrics, notification, re-training


MLOps
DESIGN
MODEL DEVELOPMENT
OPERATIONS
• Requirements Engineering
• ML Use-Cases Priorization
• Data Availability Check
• Data Engineering
• ML Model Engineering
• Model Testing & Validation
• ML Model Deployment
• CI/CD Pipelines
• Monitoring & Triggering

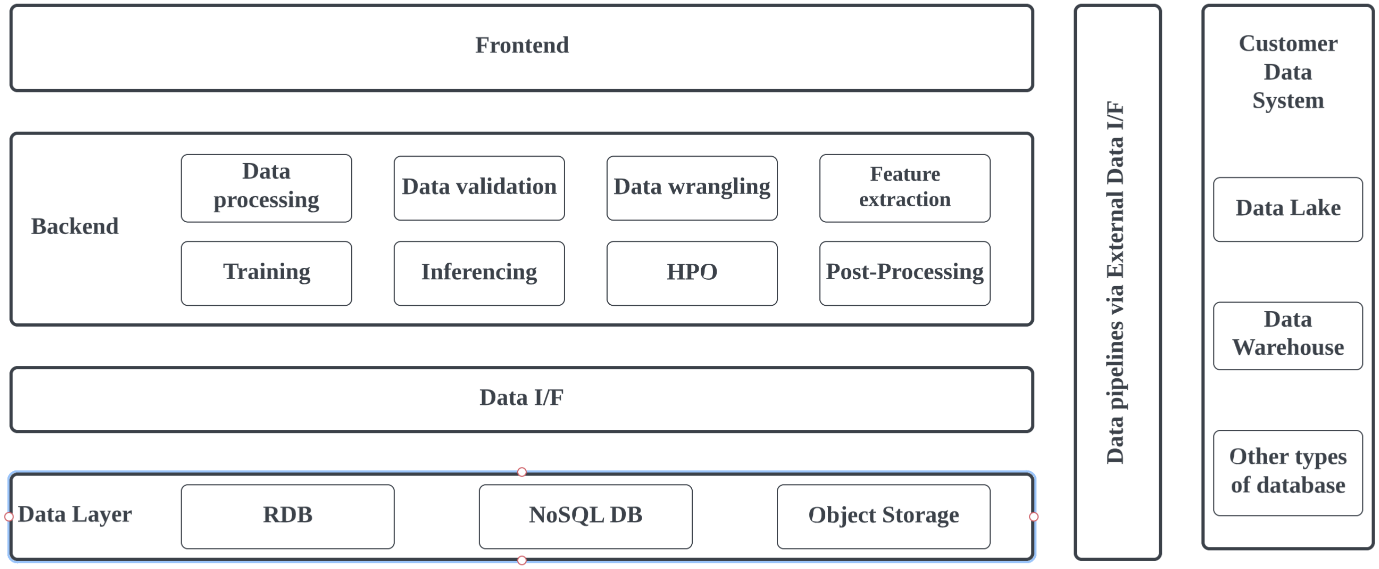
Manufacturing AI Software System

- data, data, data! – store, persist, retrieve, data quality
- seamless pipeline for development, testing, running deployed services
- development envinroment should be built separately


VISUAL DATA INTERPRETATION AND RECOGNITION
NATURAL LANGUAGE UNDERSTANDING
AUTONOMOUS SYSTEMS
CLOUD AND SCALABLE INFRASTRUCTURE
DATA STORAGE AND ORGANIZATION
DATA PREPROCESSING AND FEATURE IDENTIFICATION
SUPERVISED AND UNSUPERVISED ALGORITHMS
TRANSITION TO DEEP LEARNING

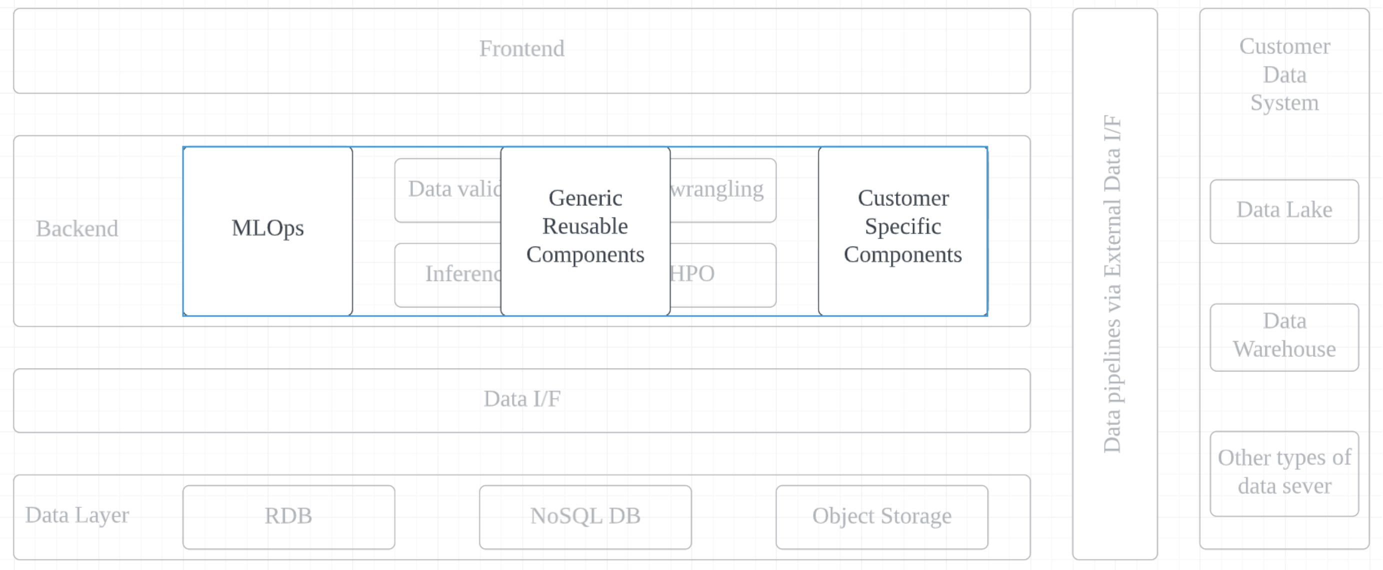
Manufacturing AI System Architecture

- frontend / backend / data I/F / data layer
- efficient and effective MLOps in backend or development environment



Reusuable components vs customer specific components

- make sure to have two separate components; generic reusable and customer specific
- generic models should be tuned for each use case
- generic model library grows as interacting with more and more customers



Conclusion

- TS ML applications found in every place in manufacturing
- concept drift and data noise make them very challenging, but have working solutions
- solutions: TS supervised learning, TS anomaly detection, model uncertainty estimation
- real bottlenecks in reality
  - data quality, prepocessing, monitoring, notification, and retraining
  - data latency, avaiability, and reliability
  - excellency in software platform design and development using cloud services

Thank You!