# DGIST 2022 ICE Invited Seminar: Time-series learning and software platform for Manufacturing AI


**Sunghee Yun**

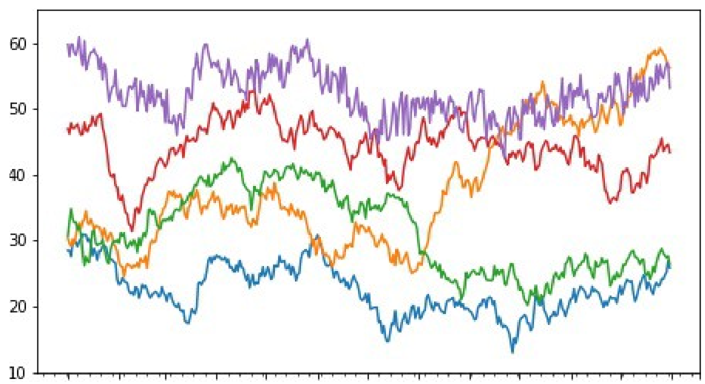
**Gauss Labs**

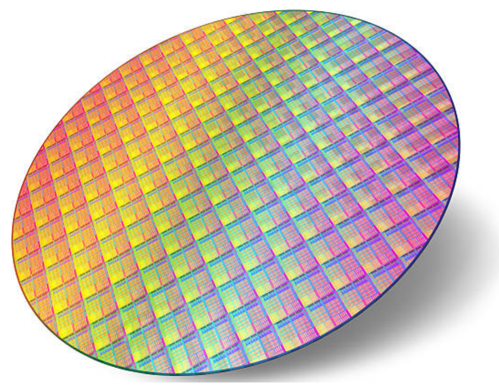
# Today

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

# Why time-series (TS) learning?

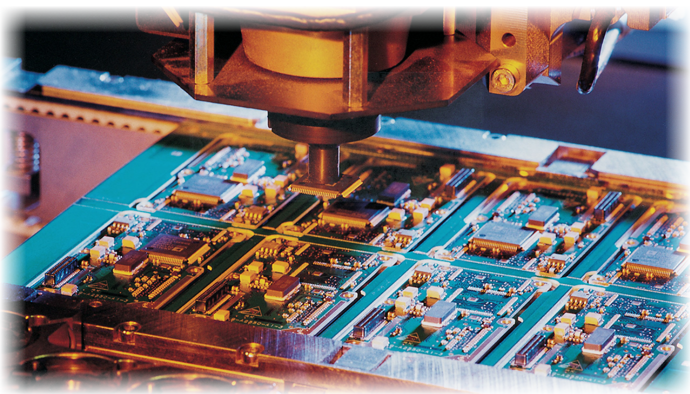
- (almost) all the data coming from manufacturing environment are TS data
  - sensor data, sound data, process times, material measurement, images, yield, *etc.*
- sheer amount of TS data is huge
  - tera-scale data per day generated in semiconductor manufacturing lines

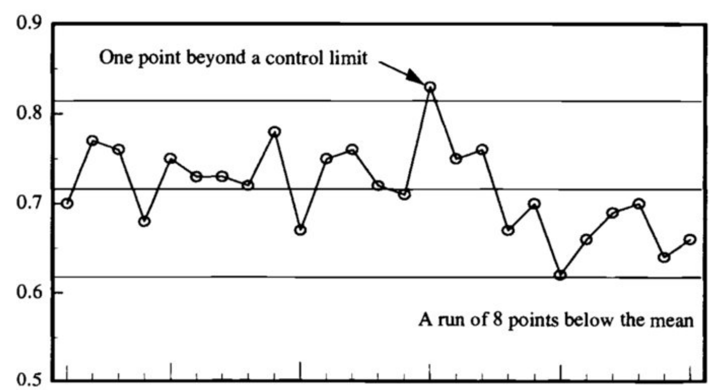


# Why TS learning?

- manufacturing application is about one of the following:
  - prediction of TS values - virtual metrology, yield prediction
  - classification of TS values - equipment anomaly alarms
  - anomaly detection on TS data - root cause analysis, yield analysis





# Machine Learning algorithms for TS data

# 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 $n = 3$

$$x(t) = \begin{bmatrix} \text{thickness}(t) \\ \text{temperature}(t) \\ \text{pressure}(t) \\ \text{feature\_size}(t) \end{bmatrix}$$

- 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

- more general defintion

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

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

- for example, $x(s)$ and $y(s)$ can represent the features and target values for a processed material, $s$, where they are not measured at the same time

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

# Supervised learning for TS

- canonical problem:

$$\begin{array}{ll} \text{predict} & y(t_k) \\ \text{given} & x(t_k), x(t_{k-1}), \ldots \text{ and } y(t_{k-1}), y(t_{k-2}), \ldots \end{array}$$

- lots of methods exist depending on assumptions of the data
  - for example, if we assume joint probability distribution of the data, we can have optimal solutions in certain criteria
- however, in this talk, we will *not* make such assumptions

# Problem formulation

- canonical problem formulation:

$$\begin{array}{ll} \text{minimize} & \sum_{k=0}^{K} l(y(t_k), \hat{y}_k(t_k)) \\ \text{subject to} & \hat{y}_k(t_k) = g_k(x(t_k), x(t_{k-1}), \ldots, y(t_{k-1}), y(t_{k-2}), \ldots) \end{array}$$

  where
  - $g_0, g_1, \ldots : \mathcal{D} \to \mathbf{R}^m$ are optimization variables,
  - $\mathcal{D} = \mathbf{R}^n \times \mathbf{R}^n \times \cdots \mathbf{R}^m \cup \{\text{null}\} \times \mathbf{R}^m \cup \{\text{null}\} \times \cdots$ is domain of $g_k$,
  - $l : \mathbf{R}^m \times \mathbf{R}^m \to \mathbf{R}_+$ is loss function
- assume that for some $k$, no label is given, *i.e.*, $y(t_k) = \text{null}$
- this is *online* learning, *i.e.*, $g_k$ is updated (or not) for every step, $k$

# ML solution candidates

- ignore temporal dependency and predict $y(t_k)$ from $x(t_k)$, $\hat{y}_k(t_k) = g(x(t_k))$
  - supervised learing such as tree algorithms ($e.g.$, random forest)
  - classiscal statistical learning ($e.g.$, partial least squares),
  - boosting algorithms ($e.g.$, gradient boosting)
  - deep neural net (DNN)
- use sequential learning methods
  - recurrent neural network (RNN), long short-term memory (LSTM)
  - Transformer-type approaches using attention mechanism

# Difficulties with manufacturing applications

- for many manufacturing applications
  - covariate shift and concept drift exist:
    * $p(x(t_k), x(t_{k-1}), \ldots)$ 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 *doesn't* work!
  - DL-type algorithms do not work, either, because
    * shift/drift $\rightarrow$ data got stale quickly
    * hence, data hungry DL do not work

# A solution: prediction based on expert advice

- assume $p$ experts: $f_{i,k} : \mathbf{R}^n \to \mathbf{R}^m$ $(i = 1, 2, \ldots, p)$ for each time step, $t_k$
  - $f_{i,k}$ can be classical statistical learning, deep neural net, *etc.*
- model predictor at time step $k$, $g_k : \mathbf{R}^n \to \mathbf{R}^m$ as weighted sum of experts:

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

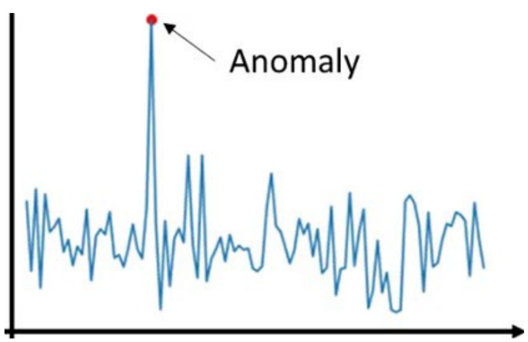
- online learning and inferencing procedure:
  - if $y(t_k) \neq$ null, *i.e.*, new observation available, update $f_{i,k}$ and $w_{i,k}$
  - if $y(t_k) =$ null, *i.e.*, no observation is available, predict $\hat{y}_k(t_k) = g_k(x(t_k))$

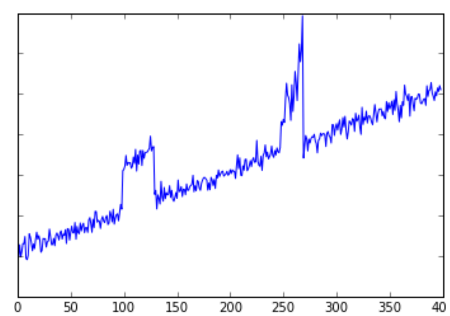
# Algorithm description

- 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

# TS anomaly detection

- three types of anomaly detection: given TS $x : T \to \mathbf{R}^n$
  - point anomaly: find $k$ such that $x(t_k)$ is considerably different from most of the other data
  - segment anomaly: find $k_1$ and $k_2$ such that TS segment $x(t_k)|_{k=k_1}^{k_2}$ is considerably different from most of the other data
  - sequence anomaly: given $x_1, \ldots, x_n : T \to \mathbf{R}$, find $x_i$ such that it is considerably different from the other TS, *i.e.*, $x_j$ $(j \neq i)$



# A TS segment anomaly detection algorithm

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

# Prediction of uncertainty of prediction

- every point prediction is wrong!
  - $\mathbf{Prob}(\hat{Y}_k = Y_k) = 0$

- more importantly, want to know how reliable our prediction is

- we call this method of *predicting of uncertainty of predictive* model uncertainty estimation (MUE)

# Model uncertainty estimation (MUE)

- multiple ways to measure this:

  (1) probability of true value falling into an interval: for fixed $a > 0$

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

  (2) predictive distribution size: find $a > 0$ such that

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

  (3) distribution of $Y_k$: find PDF of $Y_k$

- solving (3) readily solves (1) and (2)

# MUE 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 $p_{i,k}$

# MUE for expert-based online learning

- 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

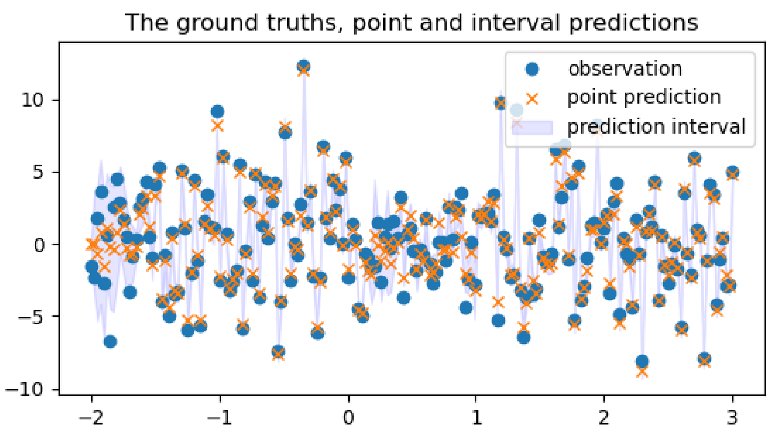
$$\mathcal{N}(w_k^T \mu_k, w_k^T \mathbf{diag}(\sigma_k) R \, \mathbf{diag}(\sigma_k) w_k)$$

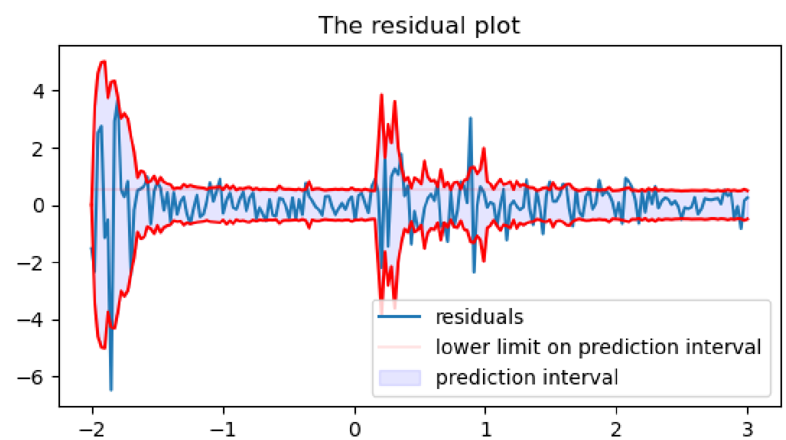
where

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

# MUE application example

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



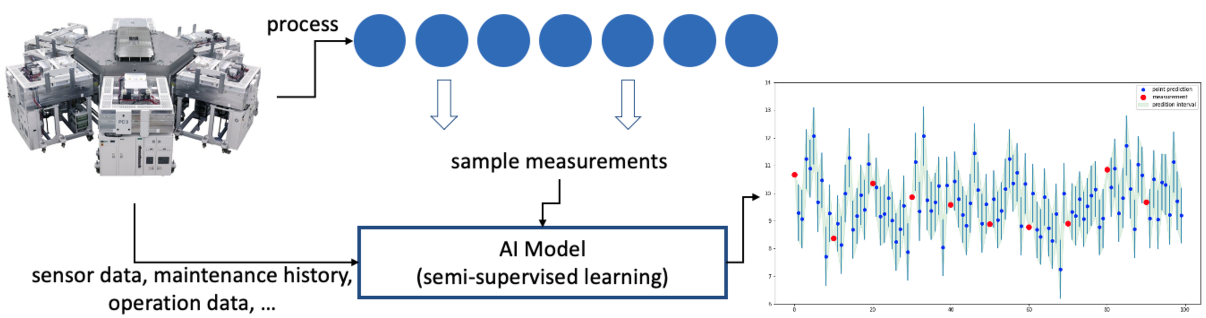
# TS Learning Applications in Manufacturing

# Virtual metrology (VM)

- 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
- **online learning method based on expert advice** can be used for the solution
- 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



# 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 $\rightarrow$ average matters
  - detecting equipment out-of-control status $\rightarrow$ anomalies matters
  - detecting root caues for yield drop
  - predicting (future) yield

# Different error measures depending on VM applications

- mean-square-error (MSE) for run-to-run control (where $\mathcal{K}$ is test index set)
  - $\sqrt{\sum_{k\in\mathcal{K}}(y(t_k) - \hat{y}(t_k))^2/|\mathcal{K}|}$
- mean-p-norm-error (MPE) for anomaly detection (for some $p > 2$)
  - $\left(\sum_{k\in\mathcal{K}} |y(t_k) - \hat{y}(t_k)|^p/|\mathcal{K}|\right)^{1/p}$
- soft-max error (SME) for anomaly detection (for some $\alpha > 0$)
  - $\log\left(\sum_{k\in\mathcal{K}} \exp(\alpha\|y(t_k) - \hat{y}(t_k)\|_1)\right)/\alpha$
- R-squared ($R^2$)
  - $1 - \frac{\sum_{k\in\mathcal{K}}(y(t_k)-\hat{y}(t_k))^2}{\sum_{k\in\mathcal{K}}(y(t_k)-\bar{y})^2}$

# 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

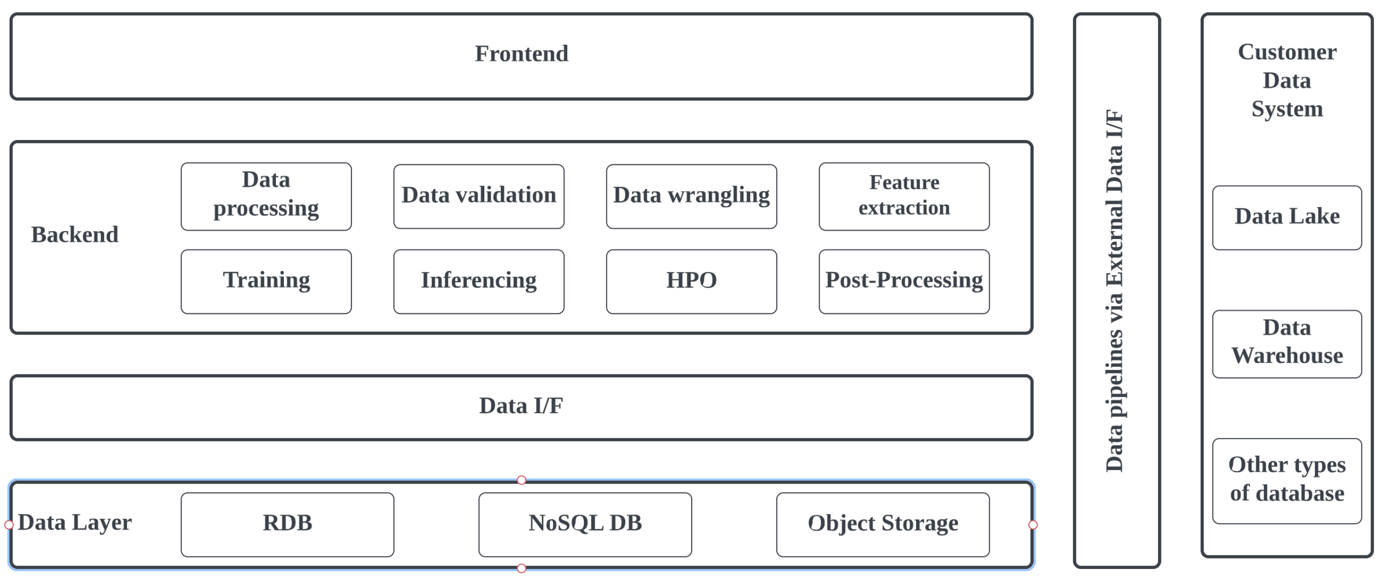
# Root cause analysis by anomaly detection

- 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 Software System Development

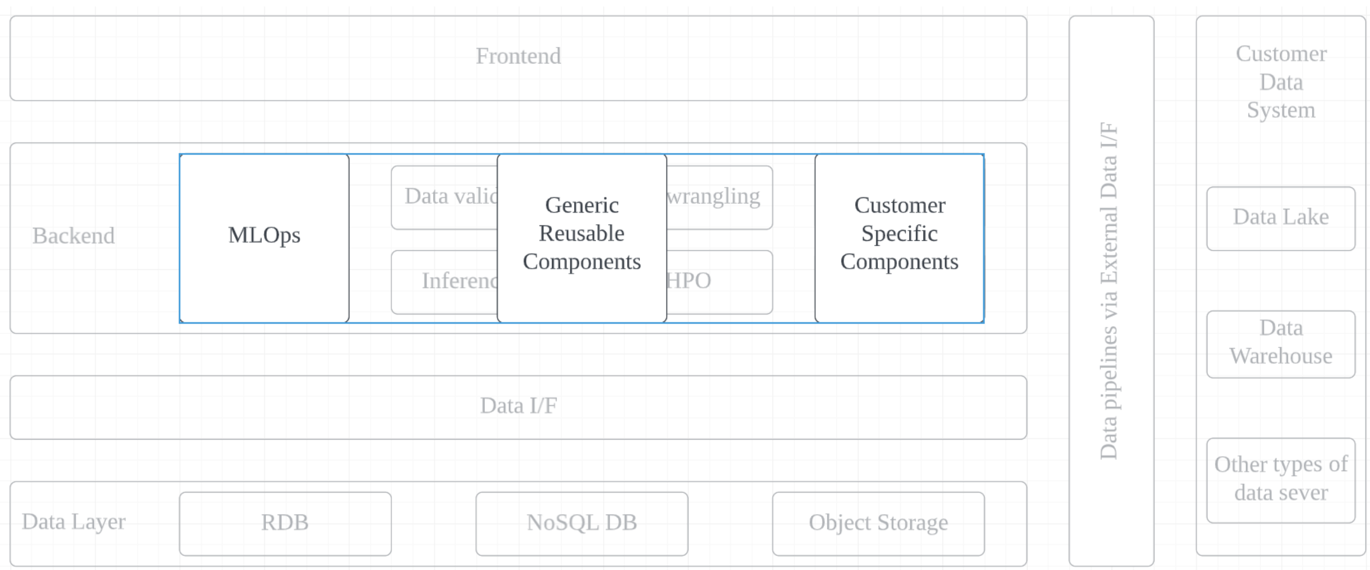
# Manufacturing AI Software System

- frontend / backend / data layers with interfaces
- external IFs for data pipelines (with security considerations)
- development envinroment should be built separately



# 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 customer (or use cases)
- generic model library grows as interacting with more and more customers



# Conclusion

- TS learning and anomaly detection occur at various places in manufacturing AI applications
- concept drift and data noise make them very challenging, but have working solutions
- solutions: TS supervised learning, TS anomaly detection, model uncertainty estimation
- lots of applications exist
  - virtual metrology, root cause analysis, yield prediction, failure pattern analysis, predictive maintenance, *etc.*
- various algorithm and software design considerations
  - software architecture, private/public cloud services
  - reusability vs domain specificity