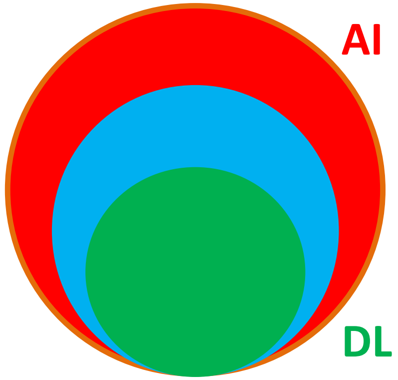
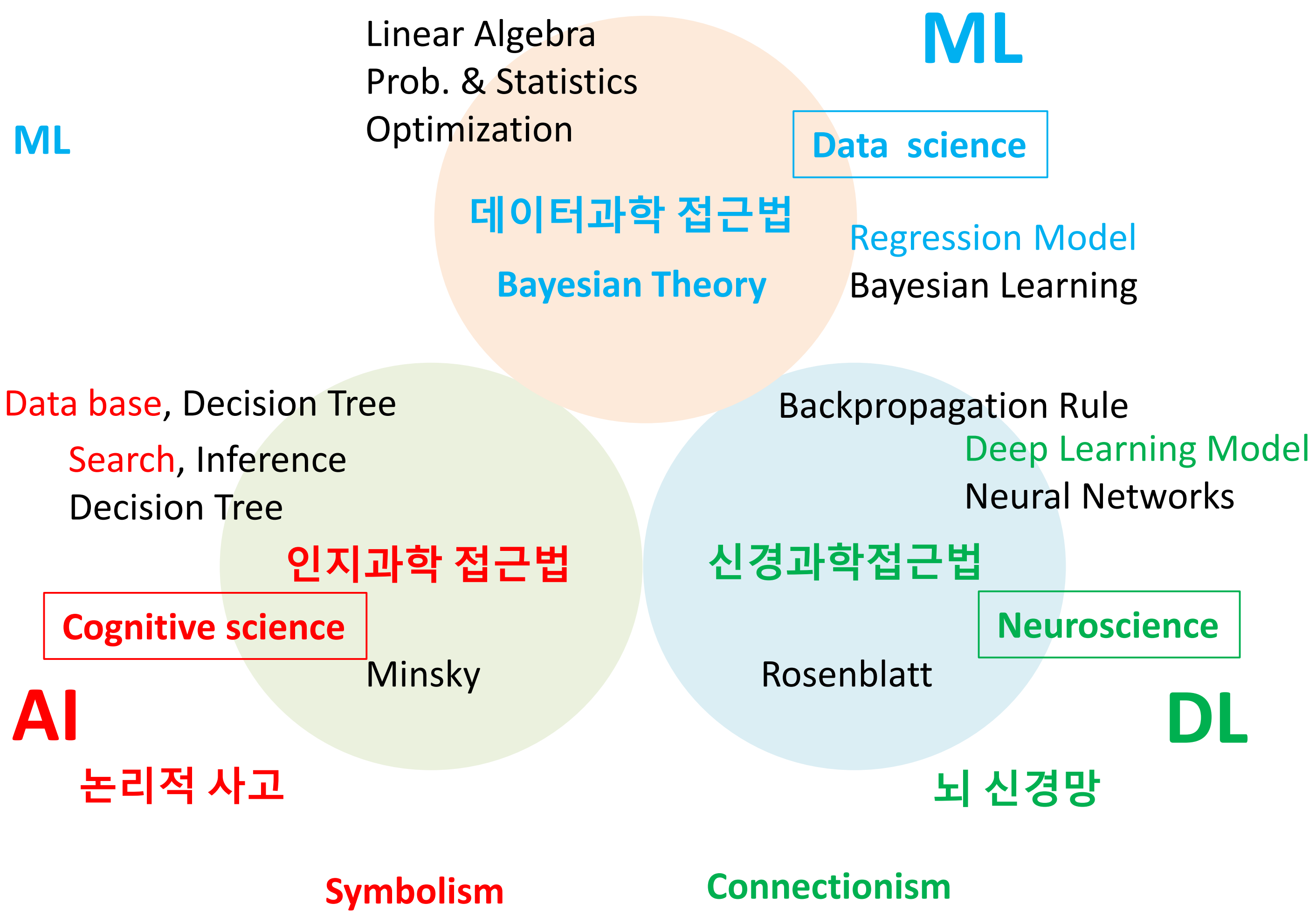
# ADVANCED MACHINE LEARNING

JIN YOUNG CHOI

ECE, SEOUL NATIONAL UNIVERSITY

# Artificial Intelligence




Model hypothesis
Linear regression
Gaussian: $\mu, \sigma$

↓

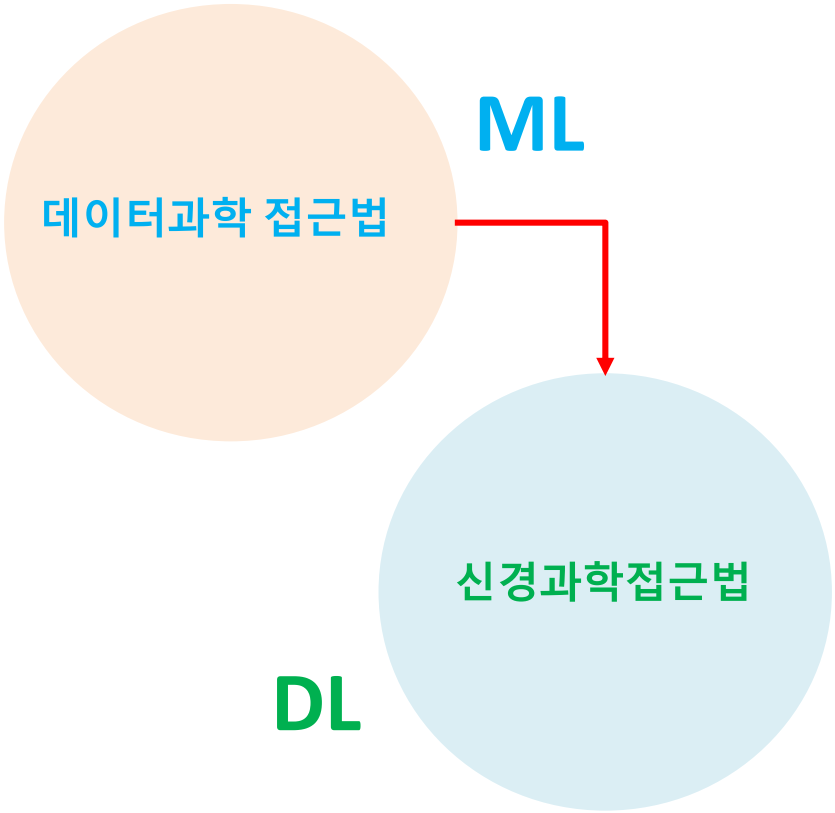
Bayesian learning
Nonlinear model
Density estimation

↓

Deep learning
Neural-net model
Universal mapping

Open access to AI
English terminology
Math. notations

# Machine Learning




Mathematics
- Linear Algebra
- Probability & Statistics
- Optimization
- **Information Theory**

Regression
- **Linear Model**
- **Piecewise Linear Model**
- **Gaussian Process Model**
- Deep Learning Model

Bayesian Framework for (Un/Semi/Self-)Supervised Learning
- **Bayesian Decision**
- **Parametric Density Estimation**
- **Non-parametric Density Estimation**
- Bayesian Networks
- Variational Auto-Encoder

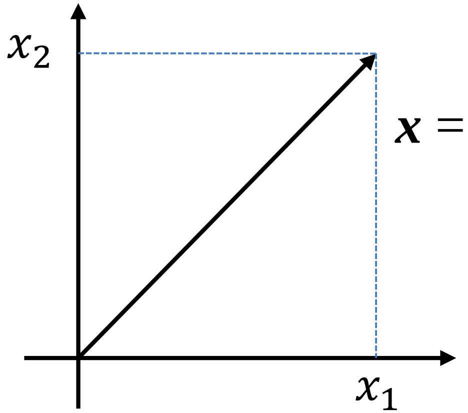
# REVIEW: VECTOR, MATRIX, TENSOR

JIN YOUNG CHOI

ECE, SEOUL NATIONAL UNIVERSITY

# $n$-tuple Vector

- $n$-Tuple Vector




$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \triangleq (x_1, x_2)$$

- Inner Product, Dot Product $\boxed{\text{similarity operator}}$

*for real field*

$$< x, y >= x \cdot y = x^T y = [x_1 \ x_2] \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = x_1 y_1 + x_2 y_2$$

*for complex field*

$$< x, y >= x \cdot y = y^* x = [\bar{y}_1 \ \bar{y}_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1 \bar{y}_1 + x_2 \bar{y}_2$$

# Matrix

- Matrix Notation

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

→ $A$ is an $m \times n$ matrix ($m$ by $n$ matrix)

$m$: number of rows; $n$: number of columns

- Matrix Addition, Scalar Multiplication

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ 4 & 5 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 7 & 9 \\ 11 & 13 \end{bmatrix}, \qquad 5\begin{bmatrix} 0 & 1 & 0 \\ 2 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 5 & 0 \\ 10 & 15 & 10 \end{bmatrix}$$

$3 \times 2 \ + \ 3 \times 2 \ = 3 \times 2$

# Matrix

- Matrix Multiplication
  - Matrix & Vector Multiplication

$$Ax = \begin{bmatrix} 1 & 1 & 6 \\ 3 & 0 & 3 \\ 1 & 1 & 4 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 0 \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} x = \begin{bmatrix} r_1^T x \\ r_2^T x \\ r_3^T x \end{bmatrix} = \begin{bmatrix} 1\cdot 2 + 1\cdot 5 + 6\cdot 0 \\ 3\cdot 2 + 0\cdot 5 + 3\cdot 0 \\ 1\cdot 2 + 1\cdot 5 + 4\cdot 0 \end{bmatrix} = \begin{bmatrix} 7 \\ 6 \\ 7 \end{bmatrix}$$

  - Matrix & Matrix Multiplication

$$AB = \begin{bmatrix} 1 & 1 & 6 \\ 3 & 0 & 3 \\ 1 & 1 & 4 \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 5 & 4 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} \begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} r_1^T x_1 & r_1^T x_2 \\ r_2^T x_1 & r_2^T x_2 \\ r_3^T x_1 & r_3^T x_2 \end{bmatrix}$$

$$= \begin{bmatrix} 1\cdot 2 + 1\cdot 5 + 6\cdot 0 & 1\cdot 3 + 1\cdot 4 + 6\cdot 1 \\ 3\cdot 2 + 0\cdot 5 + 3\cdot 0 & 3\cdot 3 + 0\cdot 4 + 3\cdot 1 \\ 1\cdot 2 + 1\cdot 5 + 4\cdot 0 & 1\cdot 3 + 1\cdot 4 + 4\cdot 1 \end{bmatrix} = \begin{bmatrix} 7 & 13 \\ 6 & 12 \\ 7 & 11 \end{bmatrix}$$

$$ABC \leftarrow n \times m \cdot m \times p \cdot p \times q = n \times q$$

# Matrix

- The transpose of a matrix $A$ is denoted by $A^T$
  - $A^T_{ij} = A_{ji}$
  - The $i$-th row of $A^T$ = the row vector from the $i$-th column of $A$
  - $A$: $m \times n$, then $A^T$: $n \times m$
- Some important results
  - $(AB)^T = (B)^T(A)^T$
  - $(A^{-1})^T = (A^T)^{-1}$
- *Def*: $A$ is called a symmetric matrix if $A^T = A$

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 2 & 3 & 4 \end{bmatrix} \quad A^T = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 1 & 1 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 3 & 2 \\ 4 & 5 & 7 \end{bmatrix} \qquad B^T = \begin{bmatrix} 1 & 4 \\ 3 & 5 \\ 2 & 7 \end{bmatrix}$$

# Exercise

- For $X^T := [x_1 \quad x_2]$, where $x_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, x_2 = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$, Calculate $x_1^T X^T$ and $X x_1$.

- Sol.

# Exercise

- For $X^T := [x_1 \quad x_2]$, where $x_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, x_2 = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$, Calculate $x_1^T X^T$ and $Xx_1$.

- Sol.

$$x_1^T X^T = x_1^T [x_1 \quad x_2] = [x_1^T x_1 \quad x_1^T x_2] = [5 \quad -10]$$

$$Xx_1 = \begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix} x_1 = \begin{bmatrix} x_1^T x_1 \\ x_2^T x_1 \end{bmatrix} = \begin{bmatrix} 5 \\ -10 \end{bmatrix}$$

# Exercise

- For $X^T := [x_1 \quad x_2]$, where $x_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, x_2 = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$, Calculate $X^T X$ and $X X^T$.

- Sol.

# Exercise

- For $X^T \coloneqq [x_1 \quad x_2]$, where $x_1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, x_2 = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$, Calculate $X^TX$ and $XX^T$.

- Sol.

$$X^TX = [x_1 \quad x_2]\begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix} = x_1x_1^T + x_2x_2^T = \begin{bmatrix} 1 \\ -2 \end{bmatrix}[1 \quad -2] + \begin{bmatrix} -4 \\ 3 \end{bmatrix}[-4 \quad 3]$$
$$= \begin{bmatrix} 1 \times 1 & 1 \times (-2) \\ (-2) \times 1 & (-2) \times (-2) \end{bmatrix} + \begin{bmatrix} 16 & -12 \\ -12 & 9 \end{bmatrix} = \begin{bmatrix} 17 & -14 \\ -14 & 13 \end{bmatrix}$$

$$XX^T = \begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix}[x_1 \quad x_2] = \begin{bmatrix} x_1^Tx_1 & x_1^Tx_2 \\ x_2^Tx_1 & x_2^Tx_2 \end{bmatrix} = \begin{bmatrix} 5 & -10 \\ -10 & 25 \end{bmatrix}$$

# Neural Computation

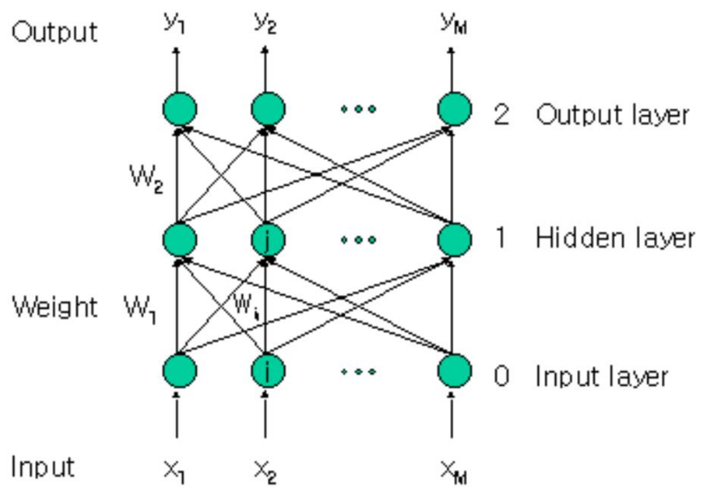
- For $w_i = \begin{bmatrix} w_{i1} \\ \vdots \\ w_{iM} \end{bmatrix}, x_p = \begin{bmatrix} x_{p1} \\ \vdots \\ x_{pM} \end{bmatrix}$,

$$h_i(x_p) = \sigma(w_i^T x_p + b_i) = \sigma(x_i^T w_i + b_i) = \sigma(\textstyle\sum_j w_{ij} x_{ij} +$$

- Matrix Form (Linear Transformation)

$$h(x_p) = \begin{bmatrix} h_1 \\ \vdots \\ h_N \end{bmatrix} = \begin{bmatrix} w_1^T \\ \vdots \\ w_N^T \end{bmatrix} x_p + \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} = W x_p + b$$

where $W = \begin{bmatrix} w_1^T \\ \vdots \\ w_N^T \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1M} \\ \vdots & \ddots & \vdots \\ w_{N1} & \cdots & w_{NM} \end{bmatrix} = [w_1 \quad \cdots \quad w_N]^T$



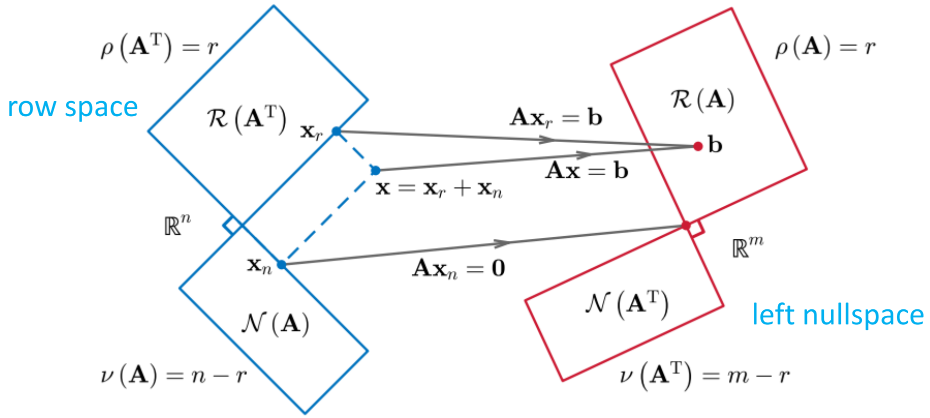
# Tensor, Concatenation

- Vector

  $n \times 1$ vector $x_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in} \end{bmatrix}$

- Matrix (linear map, 선형사상)

  $n \times m$ matrix $X_k = [x_1 \quad \cdots \quad x_m]$

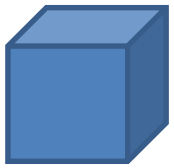


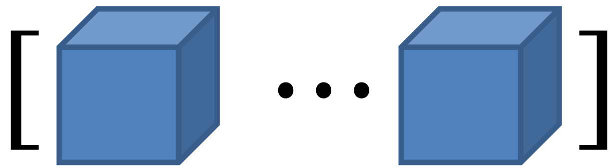

- Concatenation

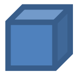
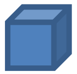
  $nm \times 1$ vector $x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$

- Tensor (multilinear map, 다중 선형사상)

  $n \times m \times p$ tensor $Y = [X_1 \quad \cdots \quad X_p]$

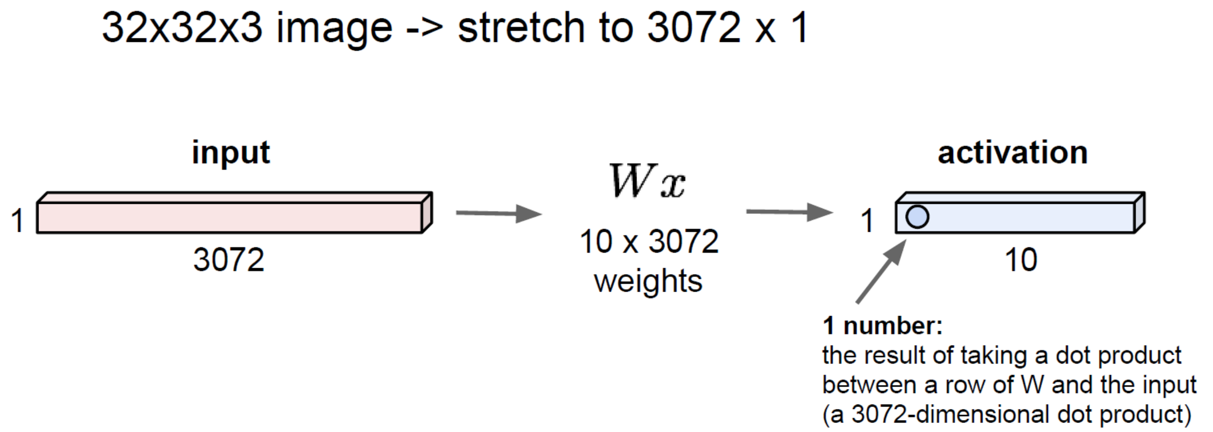




- <  ,  > =?

# Convolution in CNN

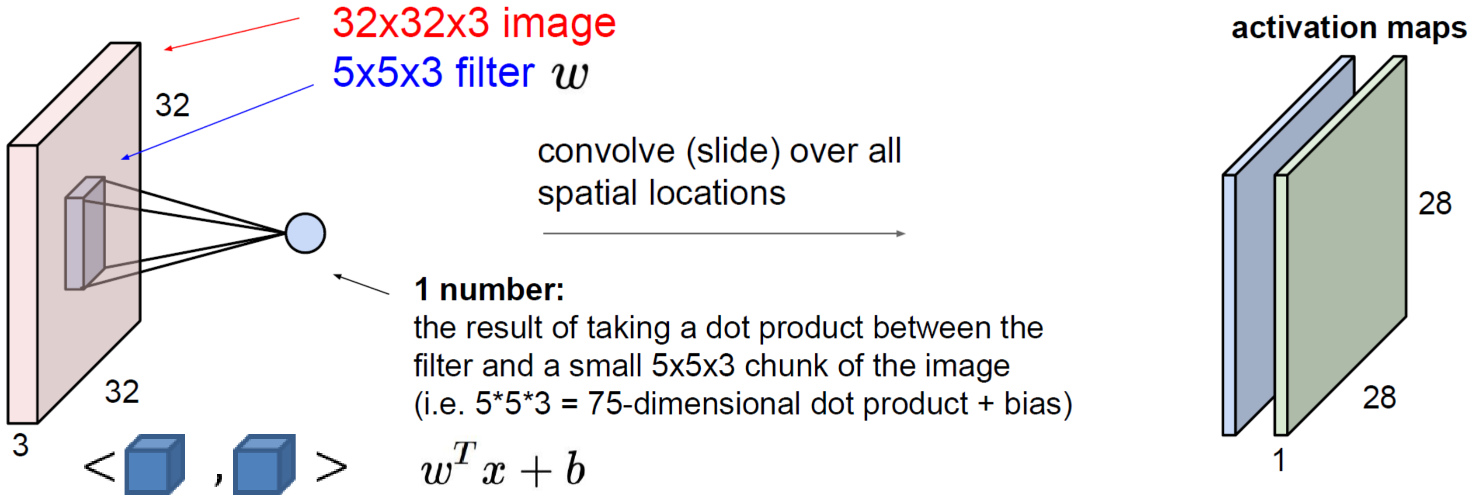
- Fully Connected Layer




$$\boxed{\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}}$$

$$\boxed{\mathbf{10 \times 1} \quad \mathbf{10 \times 3072} \quad \mathbf{3072 \times 1}}$$

- Convolution Layer



# LINEAR REGRESSION

**JIN YOUNG CHOI**

**ECE, SEOUL NATIONAL UNIVERSITY**

http://3.droppdf.com/files/pjxkI/regression-analysis-by-example-5th-edition.pdf
https://github.com/jwangjie/Gaussian-Processes-Regression-Tutorial

# Regression Analysis

- For independent random variable $X$, and dependent random variable $Y$, assume they have a functional correlation between them, i.e.

$$Y = f(X)$$

- Regression: a process to find a parametric model $\hat{f}$ that gives the best fit of $f$ for the observed samples

$$Y = \hat{f}(X) + \epsilon, \quad X\text{: predictor r.v., } Y\text{: response r.v.}$$

- Assume $E(\epsilon) = 0$, var$(\epsilon) = \sigma^2$, then $E(Y|x) = \hat{f}(x)$ for an observed non-random value $x$
- $\hat{f}$ can be estimated from the sample pairs $\{(y_i, x_i)|\ i = 1, 2, \cdots, n\}$

$$y_i = \hat{f}(x_i) + \epsilon_i,\ \ i = 1,\ \cdots,\ n,$$

where $\epsilon_i$ are i.i.d. zero mean and variance $\sigma^2$

identical independent

# Simple Linear Regression

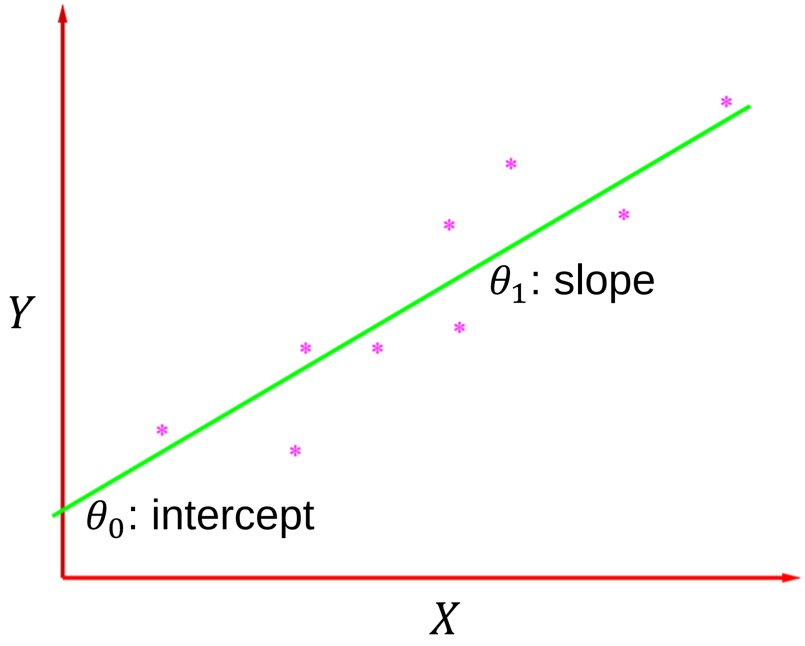
- Simple linear regression model

$$Y = \theta_0 + \theta_1 X + \epsilon$$

$$y_i = \theta_0 + \theta_1 x_i + \epsilon_i,\ i = 1,\ \cdots,\ n,$$

where $\theta_0$: intercept, $\theta_1$: slope

| Observation Number | Response Y | Predictor X |
|---|---|---|
| 1 | $y_1$ | $x_1$ |
| 2 | $y_2$ | $x_2$ |
| 3 | $y_3$ | $x_3$ |
| ⋮ | ⋮ | ⋮ |
| $n$ | $y_n$ | $x_n$ |



# Simple Linear Regression

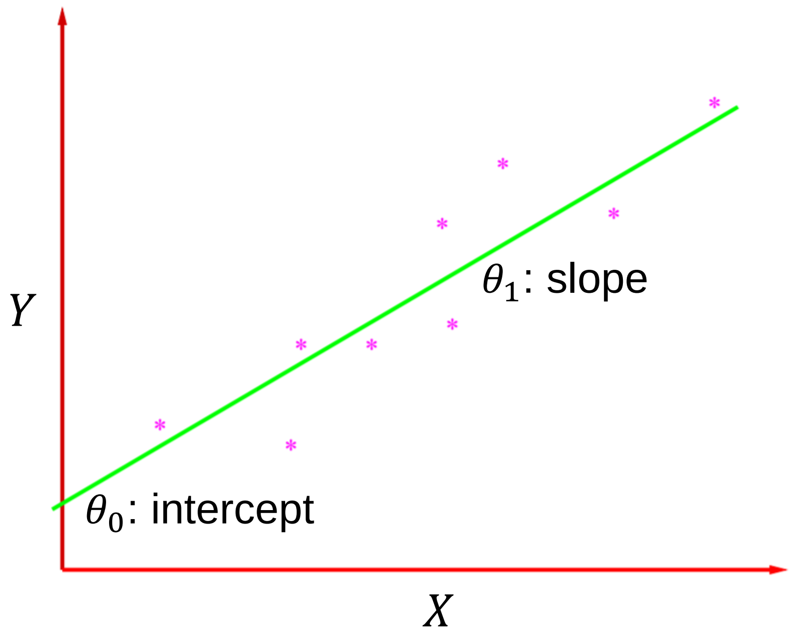
- Correlation of $Y$ & $X$

$$Y = \theta_0 + \theta_1 X + \epsilon$$

$$\text{Cov}(Y, X) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})\,(x_i - \bar{x})$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ , $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

| Observation Number | Response Y | Predictor X |
|---|---|---|
| 1 | $y_1$ | $x_1$ |
| 2 | $y_2$ | $x_2$ |
| 3 | $y_3$ | $x_3$ |
| ⋮ | ⋮ | ⋮ |
| n | $y_n$ | $x_n$ |



# Simple Linear Regression

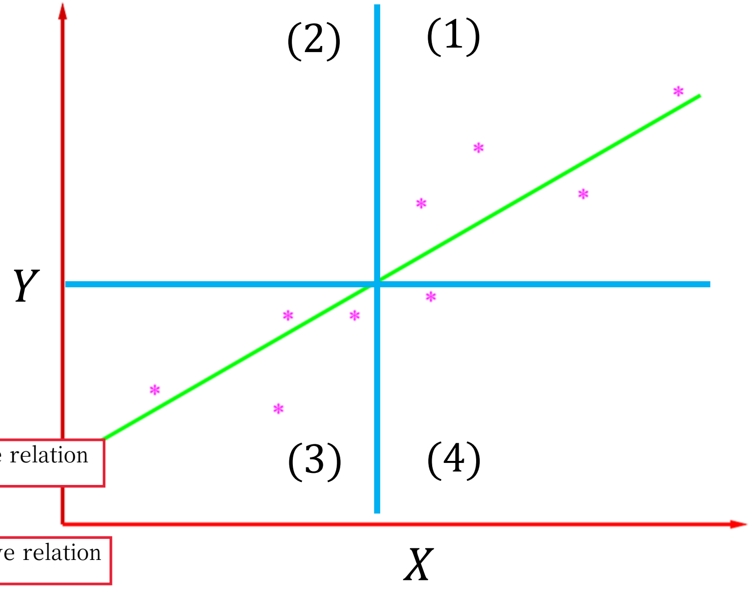
- Correlation of $Y$ & $X$

$$Y = \theta_0 + \theta_1 X + \epsilon$$

$$\mathrm{Cov}(Y, X) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})\,(x_i - \bar{x})$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i\,,\ \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

| Q | $y_i - \bar{y}$ | $x_i - \bar{x}$ | $(y_i - \bar{y})(x_i - \bar{x})$ |
|---|---|---|---|
| (1) | $+$ | $+$ | $+$ |
| (2) | $+$ | $-$ | $-$ |
| (3) | $-$ | $-$ | $+$ |
| (4) | $-$ | $+$ | $-$ |




$\theta_1 \geq 0 \longrightarrow \mathrm{Cov}(Y, X) \geq 0$ positive relation

$\theta_1 < 0 \longrightarrow \mathrm{Cov}(Y, X) < 0$ negative relation

0 => uncorelated

# Simple Linear Regression

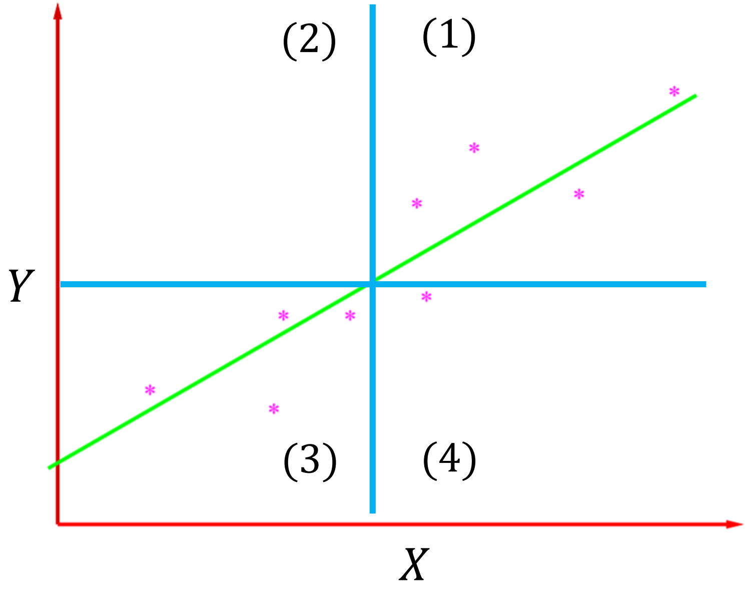
- Correlation Coefficient of $Y$ & $X$

$$Y = \theta_0 + \theta_1 X + \epsilon$$

$$-1 \le \rho(Y,X) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i-\bar{y}}{\sigma_y}\right)\left(\frac{x_i-\bar{x}}{\sigma_x}\right) \le 1$$

where $\sigma_y^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i-\bar{y})^2$ , $\sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2$

| Q | $y_i - \bar{y}$ | $x_i - \bar{x}$ | $(y_i - \bar{y})(x_i - \bar{x})$ |
|---|---|---|---|
| (1) | $+$ | $+$ | $+$ |
| (2) | $+$ | $-$ | $-$ |
| (3) | $-$ | $-$ | $+$ |
| (4) | $-$ | $+$ | $-$ |




$\theta_1 \ge 0 \longrightarrow 1 \ge \rho\,(Y,X) \ge 0$

$\theta_1 < 0 \longrightarrow -1 \le \rho\,(Y,X) < 0$

# Parameter Estimation

- ## Least Squares Estimation

Parameters are estimated by maximum likelihood estimation (MLE)

$$\epsilon_i = y_i - \theta_0 + \theta_1 x_i,\ i = 1,\ \cdots,\ n, \quad \epsilon_i \sim N(0, \sigma^2)$$

MLE:

maximum likely - hood

$$(\hat{\theta}_0,\ \hat{\theta}_1) = \underset{(\theta_0,\theta_1)}{\operatorname{argmax}} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{{\epsilon_i}^2}{2\sigma^2})$$

$$(\hat{\theta}_0,\ \hat{\theta}_1) = \underset{(\theta_0,\theta_1)}{\operatorname{argmax}} \ln \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{{\epsilon_i}^2}{2\sigma^2})$$

$$(\hat{\theta}_0,\ \hat{\theta}_1) = \underset{(\theta_0,\theta_1)}{\operatorname{argmin}} \sum_{i=1}^{n} {\epsilon_i}^2 = \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)^2$$

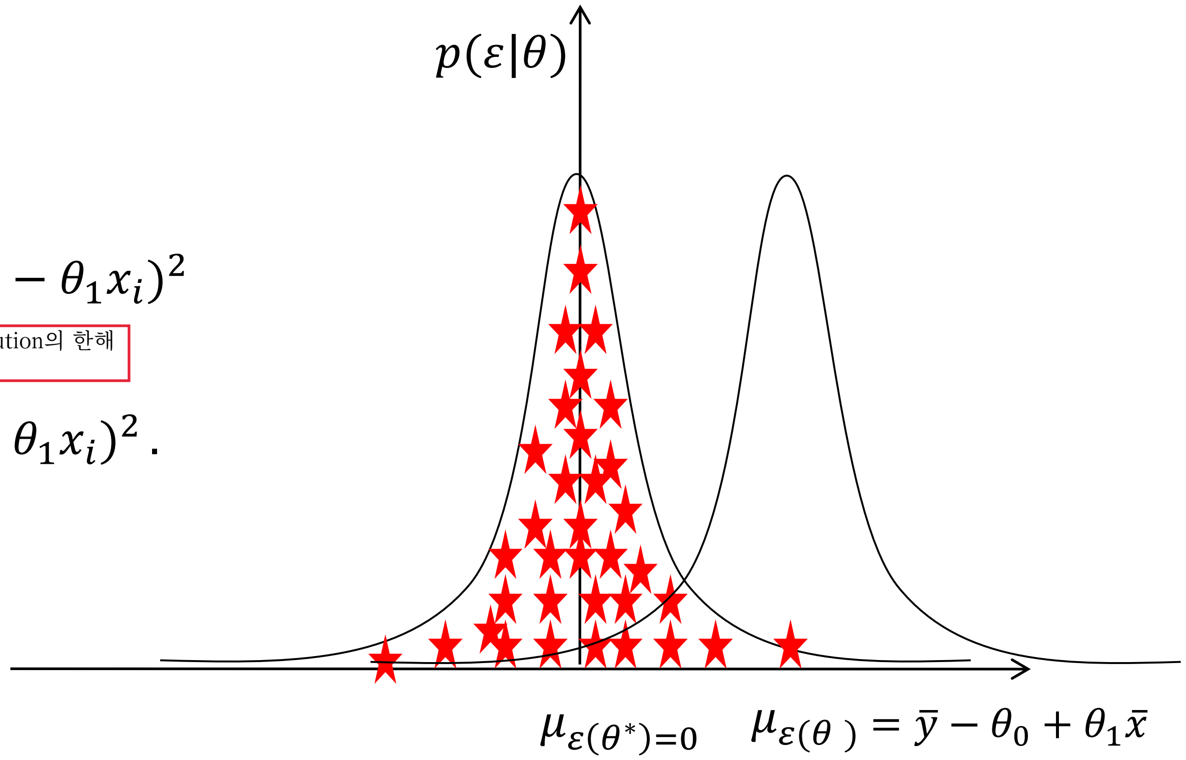
normal distribution의 한해서 LSE와 같다

LSE:

minimizing $\quad S(\theta_0,\ \theta_1) = \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)^2$.

Solution:

by $\partial S/\partial\theta_0 = 0, \partial S/\partial\theta_1 = 0$ at $\hat{\theta}_0$ & $\hat{\theta}_1$,



# Parameter Estimation

- Least Squares Estimation

$$\epsilon_i = y_i - \theta_0 + \theta_1 x_i, \qquad i = 1, \quad \cdots, \quad n.$$

LSE:

$$(\hat{\theta}_0,\ \hat{\theta}_1) = \underset{(\theta_0,\theta_1)}{\operatorname{argmin}}\ S(\theta_0,\ \theta_1) = \sum_{i=1}^{n}(y_i - \theta_0 - \theta_1 x_i)^2 .$$

Solution:

by $\partial S/\partial\theta_0 = 0, \partial S/\partial\theta_1 = 0$ at $\hat{\theta}_0$ & $\hat{\theta}_1$,

$$\sum_{i=1}^{n}\left(y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i\right) = 0, \quad \rightarrow \boxed{\hat{\theta}_0 = \bar{y} - \hat{\theta}_1 \bar{x}}$$

$$\sum_{i=1}^{n}\left(y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i\right)x_i = 0, \rightarrow \sum_{i=1}^{n}\left(y_i - \bar{y} - \hat{\theta}_1(x_i - \bar{x})\right)(x_i - \bar{x} + \bar{x}) = 0,$$

$$\rightarrow \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) - \hat{\theta}_1 \sum_{i=1}^{n}(x_i - \bar{x})^2 = 0 \rightarrow \boxed{\hat{\theta}_1 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

covariance/variance of x

# Parameter Estimation

- Least squares regression line

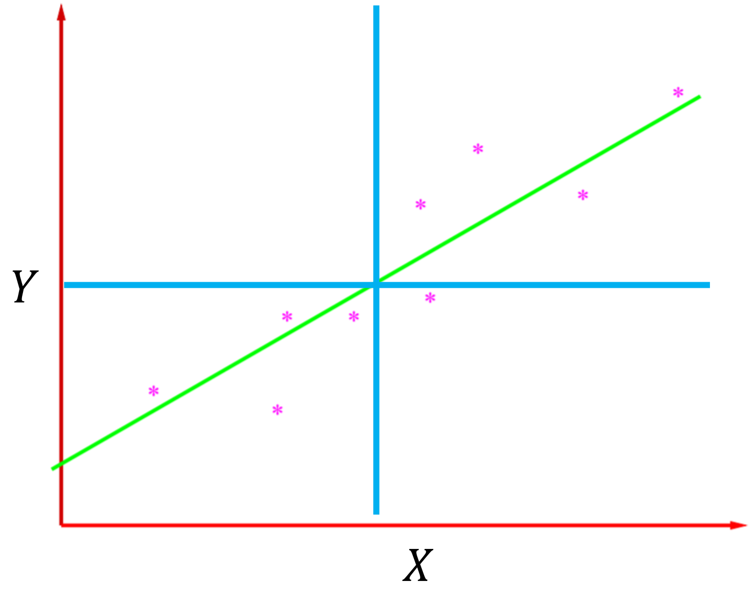
$$\hat{Y} = \hat{\theta}_0 + \hat{\theta}_1 X.$$

Fitted values:

$$\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_i, \qquad i = 1,\cdots,n.$$

Error to the $i$-th observation:

$$e_i = y_i - \hat{y}_i,\ \ i = 1,\cdots,n.$$




Alternative formula for $\hat{\theta}_1$:

$$\hat{\theta}_1 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{Cov(Y,X)}{Var(X)} = \frac{\rho(Y,X)\sigma_x\sigma_y}{\sigma_x^2} = \rho(Y,X)\frac{\sigma_y}{\sigma_x}$$

→ slope has the same sign with the correlation coefficient($\rho(Y,X)$)

# Measuring the Quality of Fit

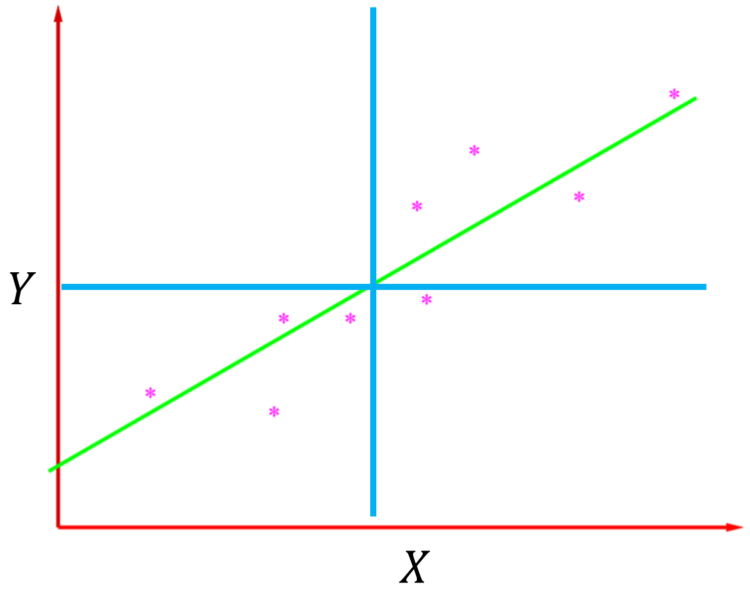
correlation btw Y & Y_hat

- Original Model:

$$Y = \theta_0 + \theta_1 X + \epsilon.$$

Least squares regression line:

$$\hat{Y} = \hat{\theta}_0 + \hat{\theta}_1 X.$$




- Correlation between $Y$ & $\hat{Y}$ :

$$\rho\left(Y, \hat{Y}\right) = \frac{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)\left(\hat{y}_i - \bar{\hat{y}}\right)}{\sqrt{\left(\sum_{i=1}^{n}(y_i - \bar{y})^2 \sum_{i=1}^{n}\left(\hat{y}_i - \bar{\hat{y}}\right)^2\right)}}$$

Note that $\rho(Y, \hat{Y})$ can not be negative. Why?

Note that $\rho\left(Y, \hat{Y}\right) = 1$ implies the perfect fit.

# Measuring the Quality of Fit

- Goodness-of-fit index:

  $SST$: $\sum_{i=1}^{n}(y_i - \bar{y})^2$, $SST$: Total sum of squares

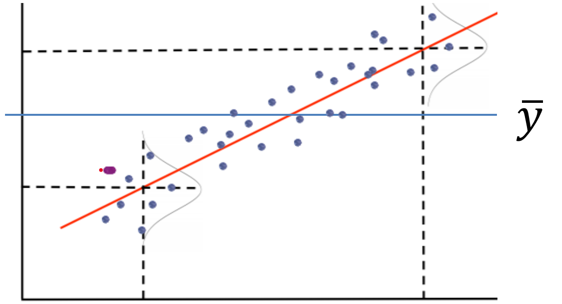
  $SSR$: $\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$, $SSR$: Regression (explained) sum of squares

  $SSE$: $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, $SSE$: Residual (error) sum of squares

- Interpretation:

| | | | | |
|---|---|---|---|---|
| $y_i$ | $=$ | $\hat{y}_i$ | $+$ | $y_i - \hat{y}_i$ |
| Observed | $=$ | Fit | $+$ | Error |
| $y_i - \bar{y}$ | $=$ | $\hat{y}_i - \bar{y}$ | $+$ | $y_i - \hat{y}_i$ |
| Deviation | | Deviation to Fit | | Residual |
| $SST$ | $=$ | $SSR$ | $+$ | $SSE$ |

$\because \sum_{i=1}^{n}(\hat{y}_i - \bar{y})(y_i - \hat{y}_i) = 0$ [1]




- $R^2$: Coefficient of determination

  $$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$ ($R = 1$ implies the perfect fit)



[1] https://math.stackexchange.com/questions/709419/prove-sst-ssessr

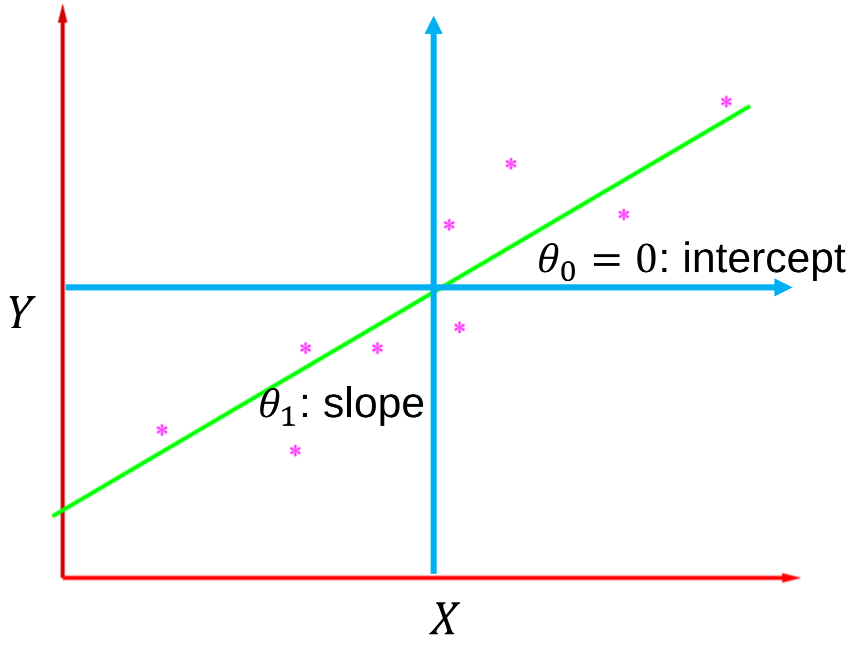
# Regression Line through Origin

- Simple linear regression model

$$Y = \theta_0 + \theta_1 X + \epsilon$$

$$Y = \theta_1 X + \epsilon,$$ *no-intercept* model, $\bar{y} = \bar{x} = 0$

| Observation Number | Response Y | Predictor X |
|---|---|---|
| 1 | $y_1 - \bar{y}$ | $x_1 - \bar{x}$ |
| 2 | $y_2 - \bar{y}$ | $x_2 - \bar{x}$ |
| 3 | $y_3 - \bar{y}$ | $x_3 - \bar{x}$ |
| ⋮ | ⋮ | ⋮ |
| n | $y_n - \bar{y}$ | $x_n - \bar{x}$ |



# Regression Line through Origin

- *no-intercept* model

$$y_i = \theta_1 x_i + \epsilon_i,$$
$$\hat{y}_i = \hat{\theta}_1 x_i, \ i = 1,\cdots,n$$
$$e_i = y_i - \hat{y}_i.$$

$$Cov(Y,X) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) \rightarrow Cov(Y,X) = \frac{1}{n}\sum_{i=1}^{n} y_i x_i$$

$$\rho(Y,X) = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i x_i}{\sigma_y \sigma_x}, \quad \sigma_y^2 = \frac{1}{n}\sum_{i=1}^{n} {y_i}^2, \sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n} {x_i}^2$$

$$\hat{\theta}_1 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \rightarrow \hat{\theta}_1 = \frac{\sum_{i=1}^{n} y_i x_i}{\sum_{i=1}^{n} {x_i}^2} = \frac{Cov(Y,X)}{\sigma_x^2} = \rho(Y,X)\frac{\sigma_y}{\sigma_x}$$

$$R^2 = \frac{\sum_{i=1}^{n} {\hat{y}_i}^2}{\sum_{i=1}^{n} {y_i}^2} = 1 - \frac{\sum_{i=1}^{n} {e_i}^2}{\sum_{i=1}^{n} {y_i}^2}$$

# Multivariate Linear Regression

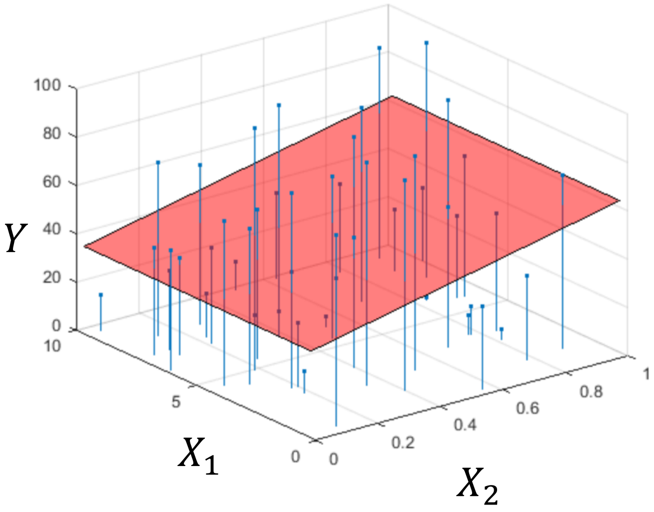
- Multivariate linear regression model: $p$ predictor (explanatory) variables

  $$Y = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_p X_p + \epsilon$$

  $$y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_p x_{ip} + \epsilon_i,\ i = 1,\ \cdots,\ n,$$

  where $\theta_0$: intercept, $(\theta_1, \theta_2, \cdots, \theta_p)$: normal vector $(ex.;\ y = w^T x + b)$

| $i$ | $Y$ | Predictor | | | |
|---|---|---|---|---|---|
| | | $X_1$ | $X_2$ | $\cdots$ | $X_p$ |
| 1 | $y_1$ | $x_{11}$ | $x_{12}$ | $\cdots$ | $x_{1p}$ |
| 2 | $y_2$ | $x_{21}$ | $x_{22}$ | $\cdots$ | $x_{2p}$ |
| 3 | $y_3$ | $x_{31}$ | $x_{32}$ | $\cdots$ | $x_{3p}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| n | $y_n$ | $x_{n1}$ | $x_{n2}$ | $\cdots$ | $x_{np}$ |



# Multivariate Linear Regression

- Multivariate linear regression model: $p$ predictor (explanatory) variables

  $Y \ = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_p X_p + \epsilon$

  $y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_p x_{ip} + \epsilon_i,\ \ i = 1,\ \cdots,\ n,$

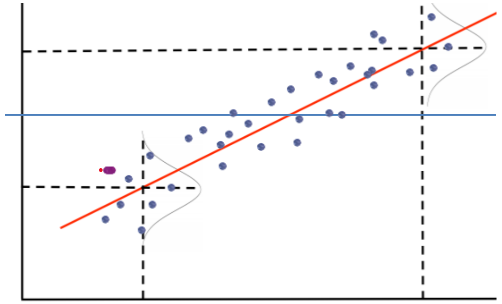
  where $\theta_0$: intercept, $(\theta_1, \theta_2, \cdots, \theta_p)$: normal vector
- Fitted model by LSE: $n - p - 1$; degree of freedom $(df)$; $p+1$; $\#\ of\ estimated\ parameters$

  $\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_{i1} + \hat{\theta}_2 x_{i2} + \cdots + \hat{\theta}_p x_{ip}, i = 1,\cdots, n,$

  $e_i = y_i - \hat{y}_i.$
- Measuring Quality of Fit:

$$\rho(Y,\hat{Y}) = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\left(\sum_{i=1}^{n}(y_i-\bar{y})^2 \sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2\right)}}$$

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{n} {e_i}^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$




- Adjusted $R^2$ : $R_a^2 = 1 - \frac{1/(n-p-1) \sum_{i=1}^{n} {e_i}^2}{1/(n-1) \sum_{i=1}^{n}(y_i-\bar{y})^2} = 1 - \frac{n-1}{n-p-1}(1 - R^2)$

DOF 로 나눠서 unbiased estimiate

# Multivariate Linear Regression

- Tests of Hypotheses for Multivariate linear model

$$Y = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_p X_p + \epsilon$$

$$y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_p x_{ip} + \epsilon_i,\ i = 1,\ \cdots,\ n,$$

where $\theta_0$: intercept, $(\theta_1, \theta_2, \cdots, \theta_p)$: normal vector

- Hypotheses: $H_0$: Reduced model (RM), $H_1$: Full model (FM)
  1. All the regression coefficients associated with the predictor variables are zero.
  2. Some of the regression coefficients are zero.
  3. Some of the regression coefficients are equal to each other.
  4. The regression parameters satisfy certain specified constraints (ex. $|\theta_i| \leq \alpha$).
- Sum of Squares: $SSE(RM) \geq SSE(FM)$

$$SSE(FM) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$SSE(RM) = \sum_{i=1}^{n} (y_i - \hat{y}_i^*)^2$$

- $F$-test: $F = \dfrac{[SSE(RM) - SSE(FM)]/(p+1-k)}{SSE(FM)/(n-p-1)}$ ($F$ is large $\rightarrow$ $RM$ is inadequate†)

† The critical values are given in Table A.4 and A.5 in "Regression Analysis by Example", S. Chatterjee et.al., Wiley.

# Parameter Estimation in Matrix form

- Least Squares Estimation

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{\epsilon}\|^2 = \|\boldsymbol{y} - \mathbf{X}\theta\|^2 \cong S(\theta)$$

Solution:

$$\nabla_\theta S(\theta) = 0 \text{ at } \hat{\theta}$$

$$\nabla_\theta (\boldsymbol{y} - \mathbf{X}\theta)^T (\boldsymbol{y} - \mathbf{X}\theta) = 0 \text{ at } \hat{\theta}$$

$$2\Phi^T \left(\boldsymbol{y} - \mathbf{X}\hat{\theta}\right) = 0$$

$$\Phi^T \boldsymbol{y} - \mathbf{X}^T\mathbf{X}\hat{\theta} = 0$$

$$\hat{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T \boldsymbol{y}$$

# Parameter Estimation in Matrix form

- Least Squares Estimation

$$\hat{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y} \leftarrow \boldsymbol{y} = \mathbf{X}\theta + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim N(0, \sigma^2\mathbf{I})$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_k^T \end{bmatrix} \theta + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_k \end{bmatrix}, \mathbf{X}_k = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k1} & x_{k2} & \cdots & x_{kp} \end{bmatrix}$$

$$y_i = \mathbf{x}_i^T\theta + \epsilon_i$$

$$y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_p x_{ip} + \epsilon_i,$$

$$i = 1, \cdots, k, \cdots, n, \cdots$$

- Observation Matrix

$$\mathbf{X}_k = [\mathbf{x}_1\ \mathbf{x}_2\ \ \cdots\ \mathbf{x}_k]^T \rightarrow \underbrace{\mathbf{X}_k^T}_{\boxed{(p+1)\times k}} \mathbf{X}_k = [\mathbf{x}_1\ \mathbf{x}_2\ \ \cdots\ \mathbf{x}_k] \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_k^T \end{bmatrix} = \sum_{i=1}^{k} \mathbf{x}_i\,\mathbf{x}_i^T$$

$$\boldsymbol{y}_k = [y_1\ \ y_2\ \ \cdots\ \ y_k]^T$$

- Recursive Least Squares

$$\hat{\theta}_k = (\mathbf{X}_k^T\mathbf{X}_k)^{-1}\mathbf{X}_k^T\boldsymbol{y}_k \rightarrow \hat{\theta}_{k+1} = (\mathbf{X}_k^T\mathbf{X}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^T)^{-1}\mathbf{X}_{k+1}^T\boldsymbol{y}_{k+1}$$

# Parameter Estimation in Matrix form

- Matrix Inversion Lemma

$$(A+BDC)^{-1}=A^{-1}-A^{-1}B(D^{-1}+CA^{-1}B)^{-1}C\,A^{-1}$$

Sherman-Morrison formula: $(A+uv^T)^{-1}=A^{-1}-\frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$

- Recursive Least Squares

$$\hat{\theta}_{k+1}=(\mathbf{X}_k^T\mathbf{X}_k+\mathbf{x}_{k+1}\mathbf{x}_{k+1}^T)^{-1}\mathbf{X}_{k+1}^T\boldsymbol{y}_{k+1}$$

define $P_k\cong(\mathbf{X}_k^T\mathbf{X}_k)^{-1}$,

$$\hat{\theta}_{k+1}=(P_k^{-1}+\mathbf{x}_{k+1}\mathbf{x}_{k+1}^T)^{-1}\mathbf{X}_{k+1}^T\boldsymbol{y}_{k+1}$$

$$=\left(P_k-\frac{P_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^TP_k}{1+\mathbf{x}_{k+1}^TP_k\mathbf{x}_{k+1}}\right)\mathbf{X}_{k+1}^T\boldsymbol{y}_{k+1},$$ (don’t need inverse)

define $G_k\cong\frac{P_k\mathbf{x}_{k+1}}{1+\mathbf{x}_{k+1}^TP_k\mathbf{x}_{k+1}}\implies P_{k+1}=P_k-\frac{P_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^TP_k}{1+\mathbf{x}_{k+1}^TP_k\mathbf{x}_{k+1}}=\boxed{P_k-G_k\mathbf{x}_{k+1}^TP_k}$

data 새로 들어왔을때 인버스 안구해도 됨

# Parameter Estimation in Matrix form

- Recursive Least Squares (cont.)

$$\hat{\theta}_{k+1} = \left(P_k - G_k \mathbf{x}_{k+1}^T P_k\right)[\mathbf{X}_k^T \quad \mathbf{x}_{k+1}]\begin{bmatrix}\mathbf{y}_k \\ y_{k+1}\end{bmatrix}$$

$$= \left(P_k - G_k \mathbf{x}_{k+1}^T P_k\right)\left(\mathbf{X}_k^T \boldsymbol{y}_k + \mathbf{x}_{k+1} y_{k+1}\right)$$

$$= \left(I - G_k \mathbf{x}_{k+1}^T\right)\left(P_k \mathbf{X}_k^T \boldsymbol{y}_k + P_k \mathbf{x}_{k+1} y_{k+1}\right)$$

$$= \left(I - G_k \mathbf{x}_{k+1}^T\right)\left(\hat{\theta}_k + P_k \mathbf{x}_{k+1} y_{k+1}\right)$$

$$= \hat{\theta}_k - G_k \mathbf{x}_{k+1}^T \hat{\theta}_k + P_k \mathbf{x}_{k+1} y_{k+1} - G_k \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1} y_{k+1}$$

$$= \hat{\theta}_k - G_k \mathbf{x}_{k+1}^T \hat{\theta}_k + G_k y_{k+1} + G_k \phi_{k+1}^T P_k \mathbf{x}_{k+1} y_{k+1} - G_k \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1} y_{k+1}$$

$$\boxed{\begin{aligned}\mathbf{X}_k &= [\mathbf{x}_1\ \mathbf{x}_2 \ \cdots \ \mathbf{x}_k]^T \\ \boldsymbol{y}_k &= [y_1\ y_2 \ \cdots \ y_k]^T\end{aligned}}$$

$$\boxed{P_k \cong (\mathbf{X}_k^T \mathbf{X}_k)^{-1}}$$

$$\boxed{G_k \cong \frac{P_k \mathbf{x}_{k+1}}{1 + \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}}}$$

$$\boxed{P_{k+1} = P_k - G_k \mathbf{x}_{k+1}^T P_k}$$

$$\boxed{\begin{aligned}\hat{\theta}_k &= (\mathbf{X}_k^T \mathbf{X}_k)^{-1} \mathbf{X}_k^T \boldsymbol{y}_k \\ &= P_k\ \mathbf{X}_k^T \boldsymbol{y}_k\end{aligned}}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + G_k\left(y_{k+1} - \mathbf{x}_{k+1}^T \hat{\theta}_k\right),\ P_0 = \alpha \mathbf{I},\ \alpha \gg 1.$$

$$\hat{\theta} = \hat{\theta}_n \qquad \boxed{G_k \cong \frac{P_k \mathbf{x}_{k+1}}{1 + \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}}} \qquad \boxed{P_{k+1} = P_k - G_k \mathbf{x}_{k+1}^T P_k}$$

# Parameter Estimation in Matrix form

- Weighted Recursive Least Squares

$$\hat{\theta}_{k+1} = (\lambda \mathbf{X}_k^T \mathbf{X}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^T)^{-1} \mathbf{X}_{k+1}^T \boldsymbol{y}_{k+1}\ ,\ 0 < \lambda < 1$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + G_k(y_{k+1} - \mathbf{x}_{k+1}^T \hat{\theta}_k),\ P_0 = \alpha \mathbf{I},\ \alpha \gg 1$$

$$\hat{\theta} = \hat{\theta}_n$$

$$G_k \cong \frac{\lambda^{-1} P_k \mathbf{x}_{k+1}}{1 + \lambda^{-1} \mathbf{x}_{k+1}^T P_k \mathbf{x}_{k+1}}$$

$$P_{k+1} = \lambda^{-1} P_k - \lambda^{-1} G_k \mathbf{x}_{k+1}^T P_k$$

$$\mathbf{X}_k = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_k]^T$$
$$\boldsymbol{y}_k = [y_1\ y_2\ \cdots\ y_k]^T$$

$$P_k \cong (\mathbf{X}_k^T \mathbf{X}_k)^{-1}$$

$$\lambda^{-1} P_k \cong (\lambda \mathbf{X}_k^T \mathbf{X}_k)^{-1}$$

# Quality of Fit in Matrix form

- Regression model in matrix form

$$\boldsymbol{y} = \mathbf{X}\theta + \boldsymbol{\epsilon}$$

- Estimated parameter

$$\hat{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\,\boldsymbol{y} = \theta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\epsilon} \text{ (unbiased estimate)}$$

- Confidence Interval

$$E\left(\hat{\theta}\right) = \theta,$$

$$E\left((\theta-\hat{\theta})^T(\theta-\hat{\theta})\right) = E\boldsymbol{\epsilon}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\epsilon} = E\;Tr(\boldsymbol{\epsilon}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\epsilon})$$

$$= E\;Tr((\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\cancel{\mathbf{X})}^{-1}\mathbf{X}^T\cancel{\mathbf{X}\boldsymbol{\epsilon}^T}\boldsymbol{\epsilon}) = Tr\left((\mathbf{X}^T\mathbf{X})^{-1}\right)\sigma^2 \rightarrow \hat{\theta} = \theta \pm \alpha\sigma$$

- Prediction

$$\hat{\boldsymbol{y}} = \mathbf{X}\hat{\theta} = \mathbf{X}\theta + \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\epsilon} = \mathbf{X}\theta + \mathbb{H}\boldsymbol{\epsilon},$$

where $\mathbb{H}$ is symmetric and idempotent ($\mathbb{H}^2 = \mathbb{H}$), $\mathbb{H}\mathbf{X} = \mathbf{X}$.

$$\mathbb{H}\hat{\boldsymbol{y}} = \mathbb{H}\mathbf{X}\theta + \mathbb{H}\boldsymbol{\epsilon} = \mathbf{X}\theta + \mathbb{H}\boldsymbol{\epsilon} = \hat{\boldsymbol{y}}$$

- Residual vector : $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}} = (\mathbf{I} - \mathbb{H})\boldsymbol{\epsilon}$

# Quality of Fit in Matrix form

- Residual vector

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}} = (\mathbf{I} - \mathbb{H})\boldsymbol{\epsilon}$$

$$E(\boldsymbol{e}^T\boldsymbol{e}) = E(\boldsymbol{\epsilon}^T(\mathbf{I} - \mathbb{H})(\mathbf{I} - \mathbb{H})\boldsymbol{\epsilon}) = E(\boldsymbol{\epsilon}^T(\mathbf{I} - \mathbb{H})\boldsymbol{\epsilon})$$
$$= Tr(\mathbf{I} - \mathbb{H})E(\boldsymbol{\epsilon}^T\boldsymbol{\epsilon}) = Tr(\mathbf{I} - \mathbb{H})n\sigma^2 = (n-p-1)n\sigma^2$$

here

$$Tr(\mathbf{I} - \mathbb{H}) = Tr(\mathbf{I}) - Tr(\mathbb{H}) = n - Tr(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T)$$
$$= n - Tr\left((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\right) = n - (p+1), p+1: \#\ of\ parameters$$

$$\boxed{(p+1) \times n \cdot \mathrm{n} \times \ (p+1)}$$

hence

$$E(\boldsymbol{e}^T\boldsymbol{e}/(n-p-1)) = n\sigma^2 \rightarrow \frac{\boldsymbol{e}^T\boldsymbol{e}}{n-p-1}$$: unbiased estimate of $n\sigma^2$ $\boxed{E(\boldsymbol{e}\boldsymbol{e}^T/(n-p-1)) = \sigma^2\mathbf{I}}$

- Coefficient of Determination

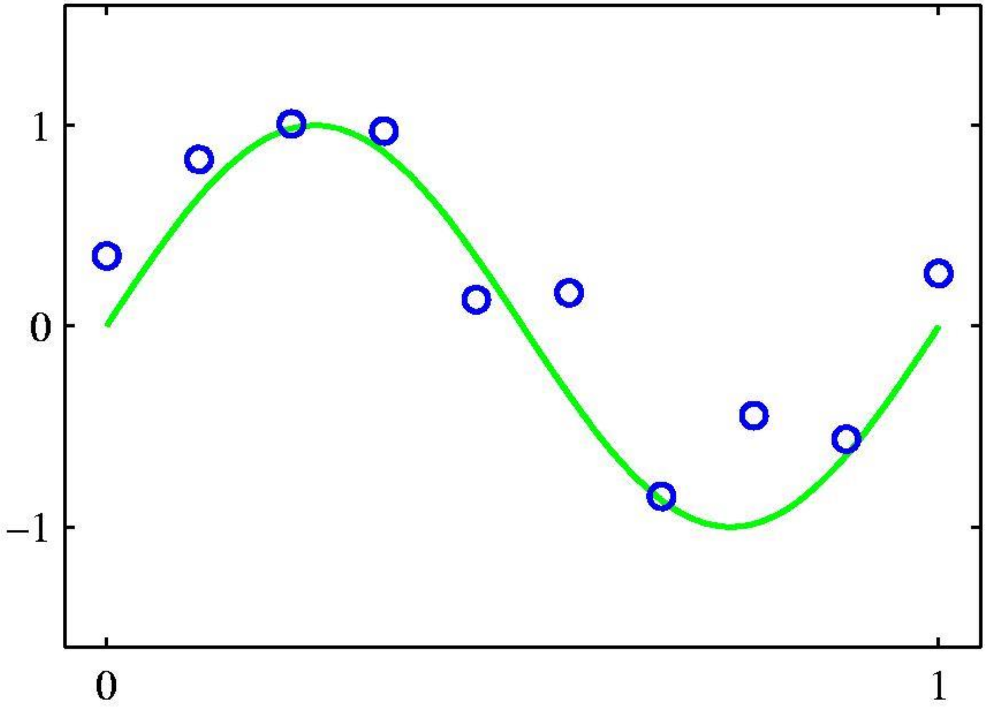
$$R^2 = 1 - \frac{\boldsymbol{e}^T\boldsymbol{e}}{(\boldsymbol{y} - \bar{y}\mathbf{1})^T(\boldsymbol{y} - \bar{y}\mathbf{1})}, R_a^2 = 1 - \frac{\boldsymbol{e}^T\boldsymbol{e}/(n-p-1)}{(\boldsymbol{y} - \bar{y}\mathbf{1})^T(\boldsymbol{y} - \bar{y}\mathbf{1})/(n-1)}$$

# PARTIAL LEAST SQUARES REGRESSION

**JIN YOUNG CHOI**

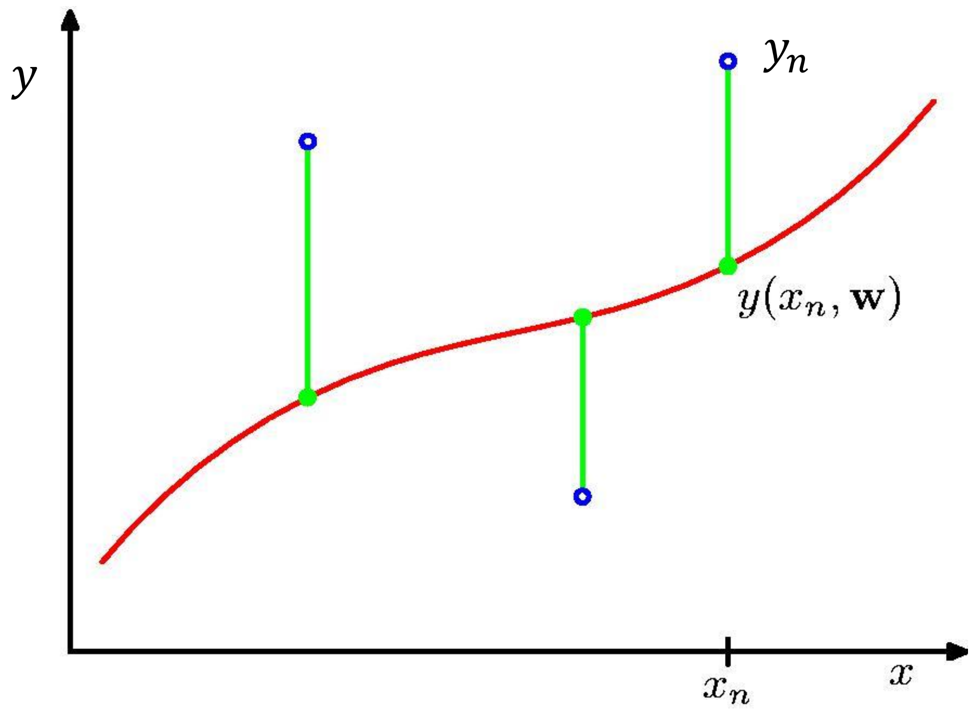
**ECE, SEOUL NATIONAL UNIVERSITY**

# Overfitting and Underfitting

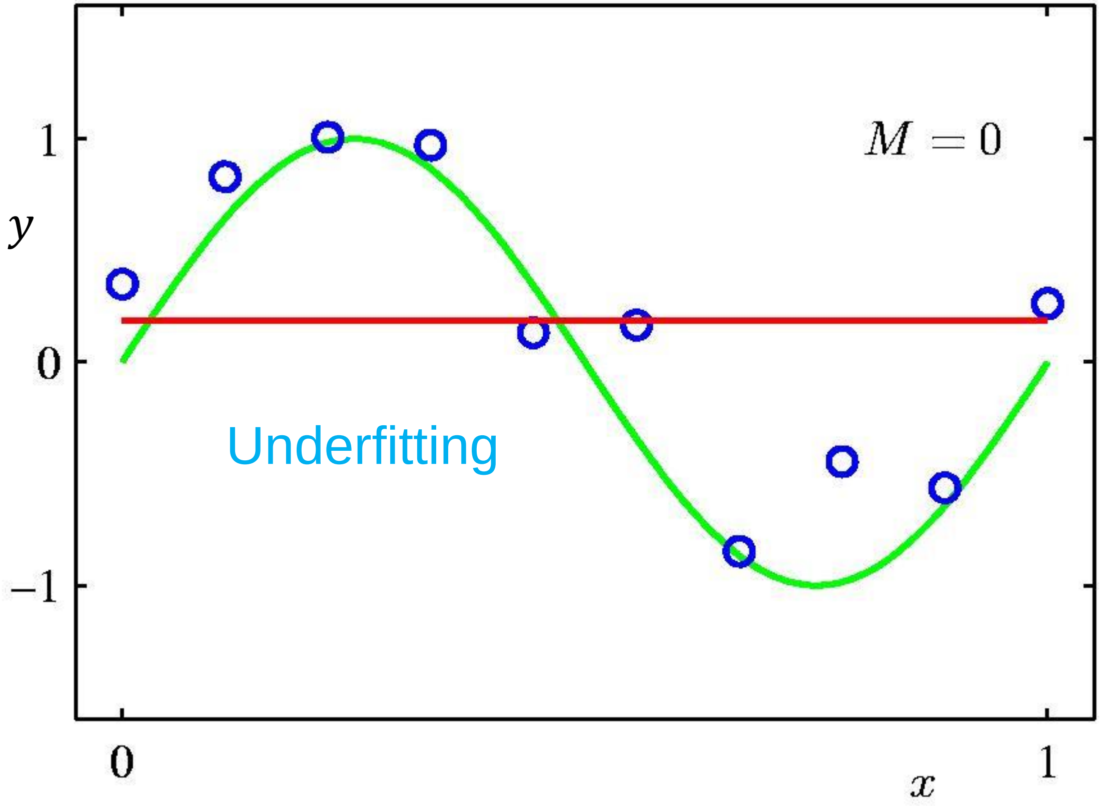



$$Y = \theta_0 + \theta_1 X + \theta_2 X^2 + \cdots + \theta_M X^M + \epsilon$$

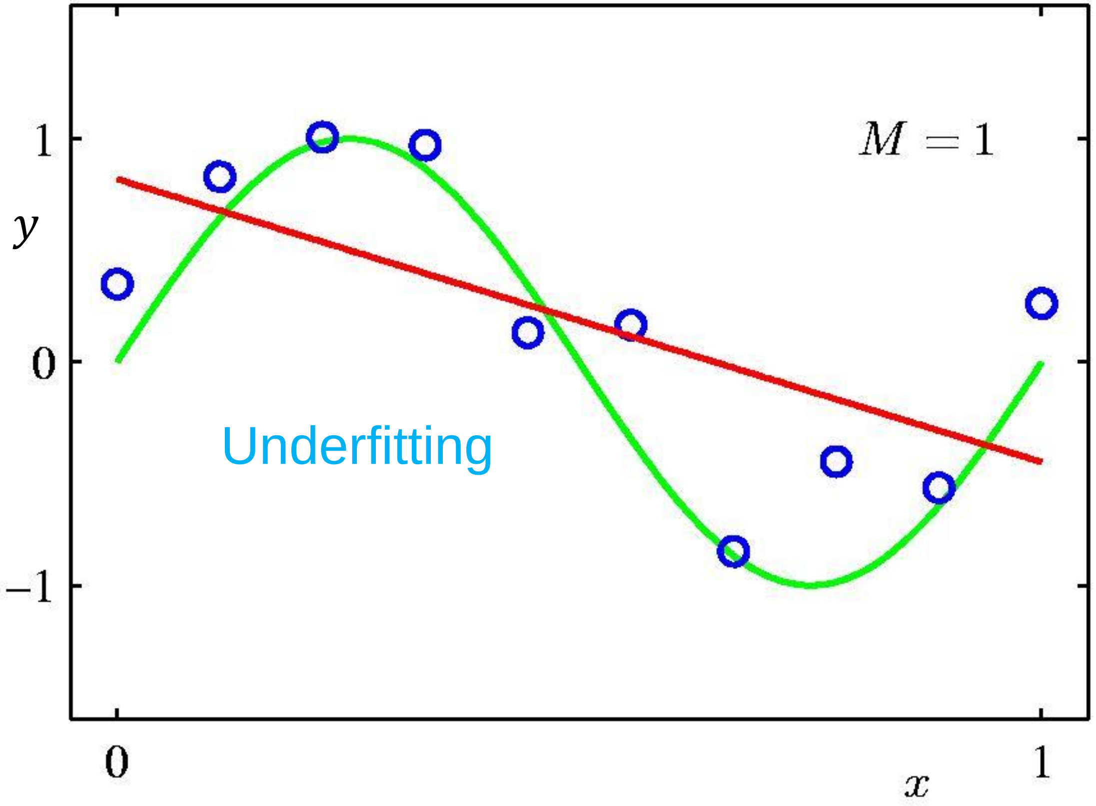
# Sum-of-Squares Error Function



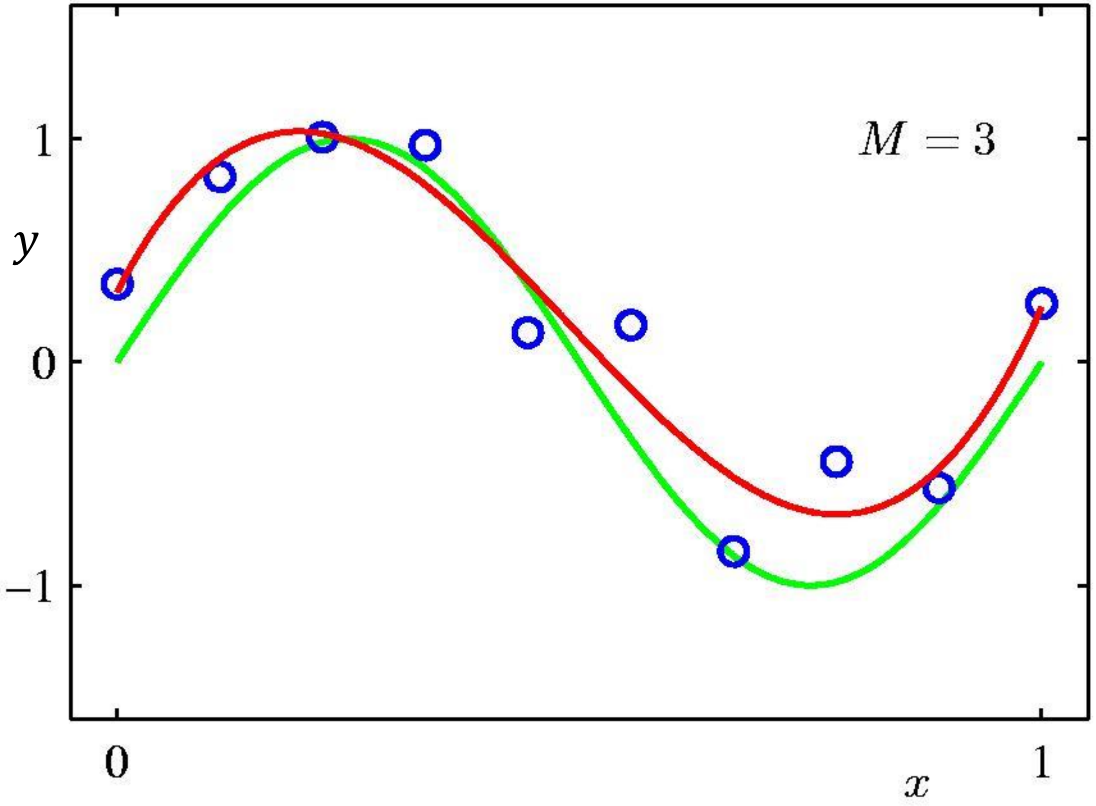
# $0^{th}$ Order Polynomial



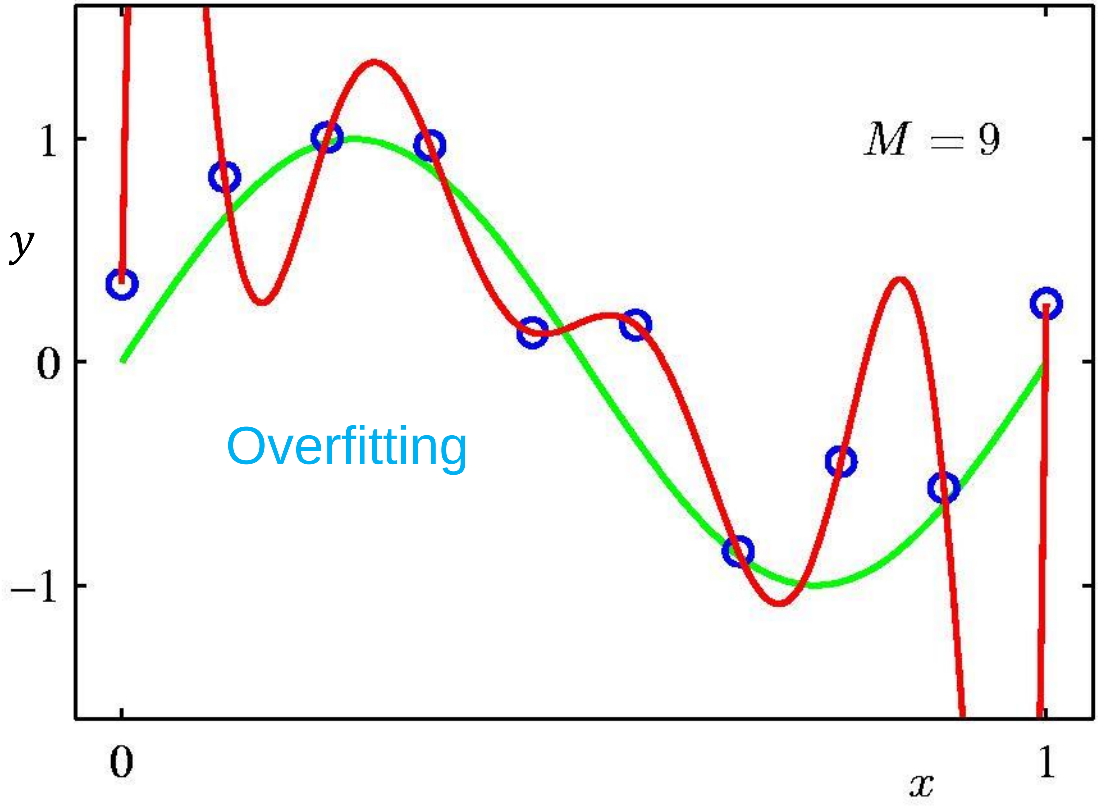
# $1^{st}$ Order Polynomial



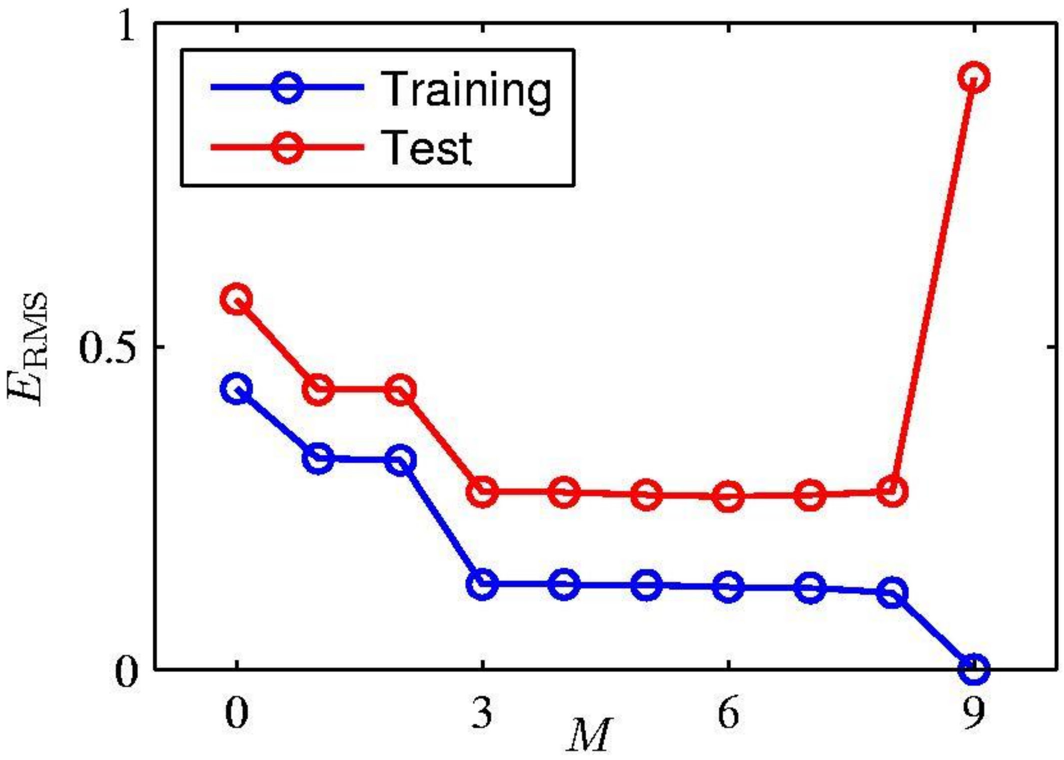
# 3rd Order Polynomial



# $9^{th}$ Order Polynomial



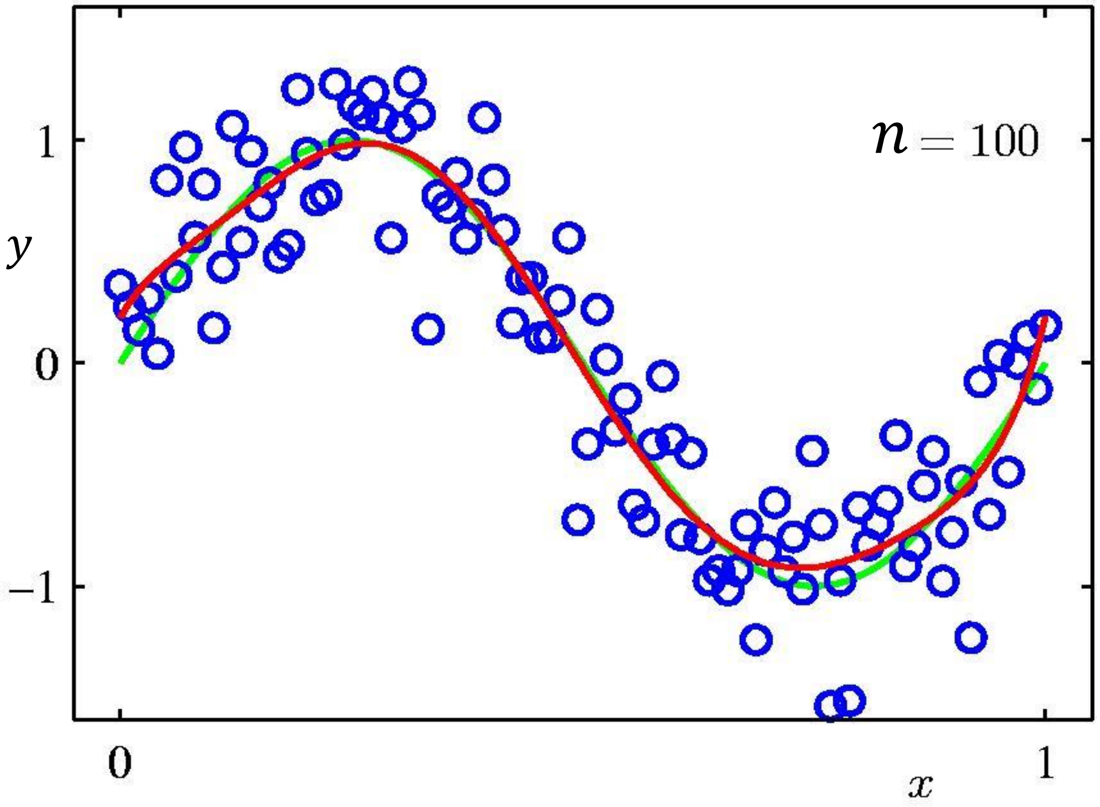
# Over-fitting




Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{E(\theta^\star)/n}$

# Data Set Size:

9th Order Polynomial



# Bias and Variance in Parameter Estimation

- Mean Squared Error(MSE) decomposition

$$MSE(\hat{\theta}) = E\left((\hat{\theta}-\theta)^2\right)$$

$$= E\left(\left(\hat{\theta}-E(\hat{\theta})+E(\hat{\theta})-\theta\right)^2\right)$$

$$= E\left(\left(\hat{\theta}-E(\hat{\theta})\right)^2 + 2\left(\hat{\theta}-E(\hat{\theta})\right)\left(E(\hat{\theta})-\theta\right) + \left(E(\hat{\theta})-\theta\right)^2\right)$$

$$= E\left(\hat{\theta}-E(\hat{\theta})\right)^2 + 2\boxed{\overbrace{E\left(\hat{\theta}-E(\hat{\theta})\right)}^{E(\hat{\theta})-E(\hat{\theta})=0}}\left(E(\hat{\theta})-\theta\right) + \left(E(\hat{\theta})-\theta\right)^2$$

$$= E\left(\hat{\theta}-E(\hat{\theta})\right)^2 + \left(E(\hat{\theta})-\theta\right)^2$$

고정 $= Var(\hat{\theta}) + Bias(\hat{\theta},\theta)^2$

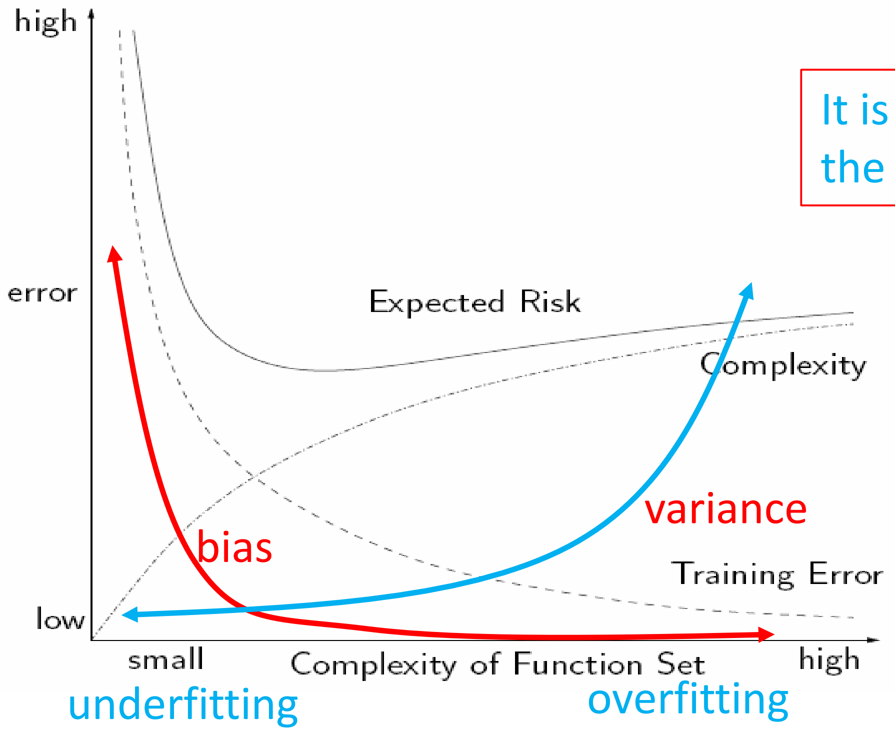
overfitting underfitting

# Bias and Variance in Parameter Estimation

- Mean Squared Error(MSE) decomposition

$$MSE(\hat{\theta}) = E\left((\hat{\theta} - \theta)^2\right)$$

$$= Var(\hat{\theta}) + Bias(\hat{\theta}, \theta)^2$$



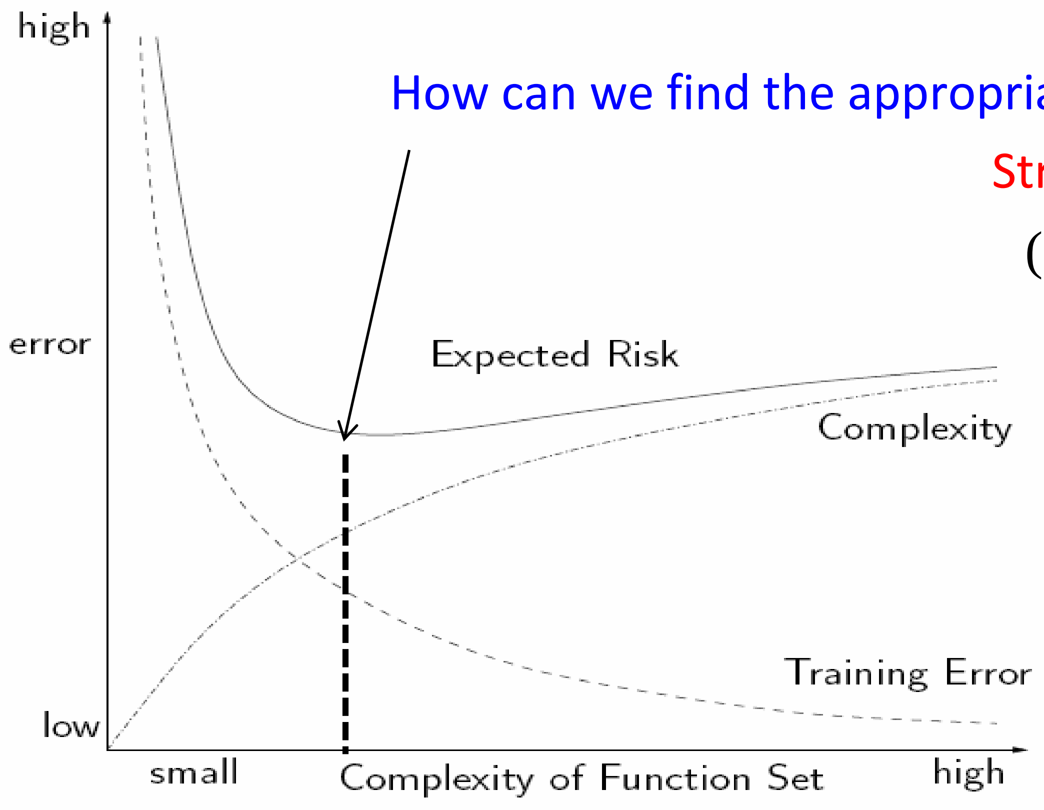

It is known that the variance becomes high and the bias low as the model complexity increases[†].



[†] https://datacadamia.com/data_mining/bias_trade-off

# Structural Risk Minimization

- For fixed training samples $n$




How can we find the appropriate complexity ?

Structural Risk Minimization

$$(\theta_{emp}, h_{emp}) = \arg\min_{\theta,h} R_{emp}(\theta, h)$$

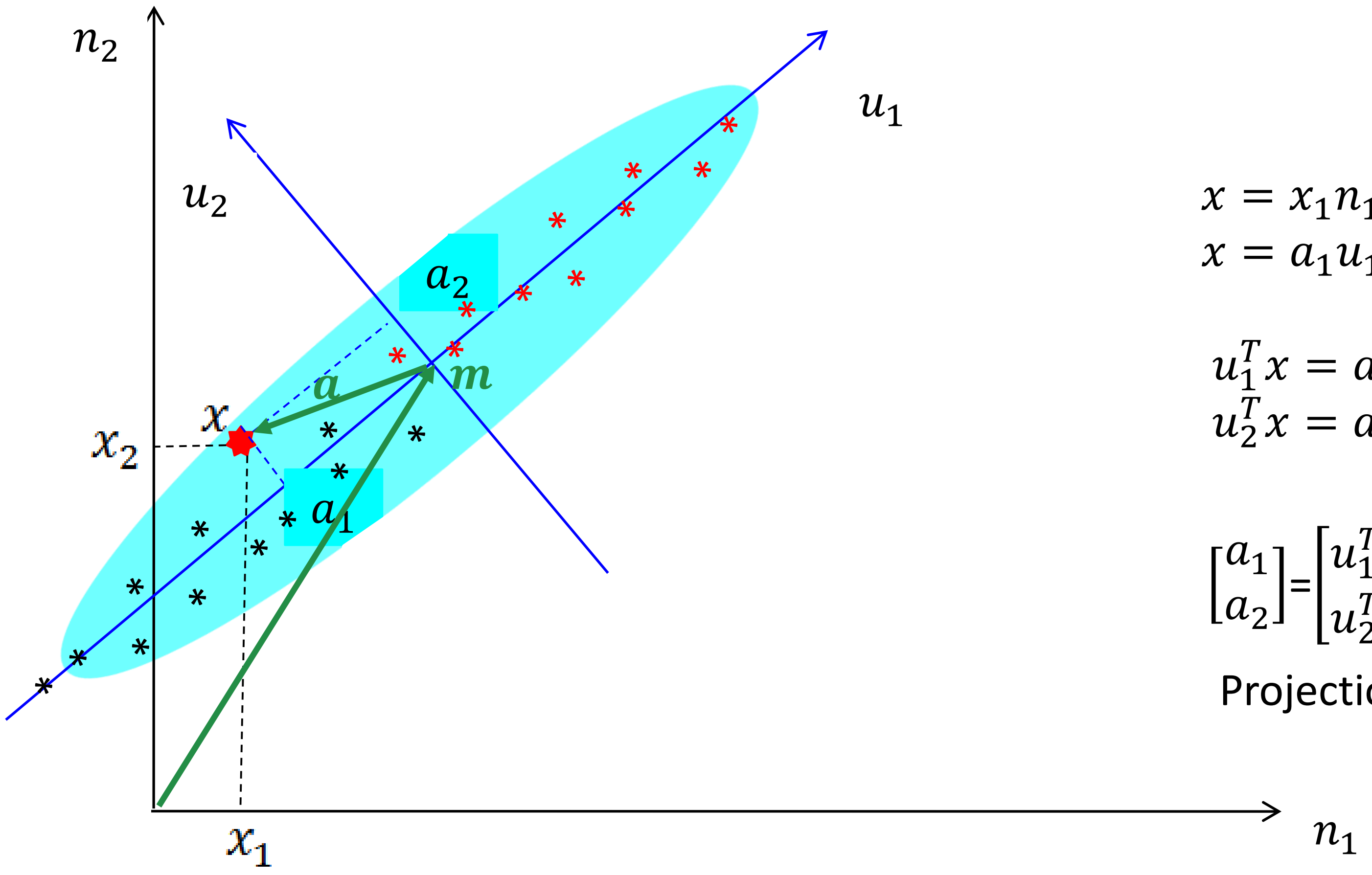
$\Downarrow$

Pruning (F-test, PCA)

Regularization
- Ridge regression
- Lasso regression
- Elastic regression

SVM

# Principal Component Analysis(Dim. Reduction)




$x = x_1 n_1 + x_2 n_2 = [n_1\ n_2]x = Nx$
$x = a_1 u_1 + a_2 u_2 + m = [u_1\ u_2]a + m = Ua + m$

$u_1^T x = a_1 u_1^T u_1 + a_2 u_1^T u_2 + u_1^T m = a_1 + u_1^T m$
$u_2^T x = a_1 u_2^T u_1 + a_2 u_2^T u_2 + u_2^T m = a_2 + u_2^T m$

$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \end{bmatrix} (x - m) \rightarrow a = U^T (x - m)$

Projection: $a_1 = u_1^T (x - m) \leftrightarrow a_1 u_1 = x - m$

$a_{ik} = \mathbf{u}_k^T (\mathbf{x}_i - \mathbf{m}) \leftrightarrow a_k^i \mathbf{u}_k = \mathbf{x}_i - \mathbf{m}$ $\quad J_1(\boldsymbol{a}, \mathbf{u}) = \sum_{i=1}^{n} ||\mathbf{m} + a_{ik}\mathbf{u}_k - \mathbf{x}_i||^2 = \sum_{i=1}^{n} ||a_{ik}\mathbf{u}_k - (\mathbf{x}_i - \mathbf{m})||^2$

# PCA ; Scatter matrix

$$\mathbf{S} = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T$$

- $J_1(\boldsymbol{a}, \mathbf{u}) = -\mathbf{u}^T \mathbf{S} \mathbf{u} + \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{m}||^2$
- Vector $\mathbf{u}$ that minimizes $J_1$ also maximizes $\mathbf{u}^T \mathbf{S} \mathbf{u}$.
- So we find $\mathbf{u}$, which maximize $\mathbf{u}^T \mathbf{S} \mathbf{u}$

  subject to $\|\mathbf{u}\|=1$

- Let $\lambda$ be Lagrange multiplier.
- Differentiating $L$ with respect to $\mathbf{u}$ : $L = \mathbf{u}^T \mathbf{S} \mathbf{u} - \lambda(\mathbf{u}^T \mathbf{u} - 1)$

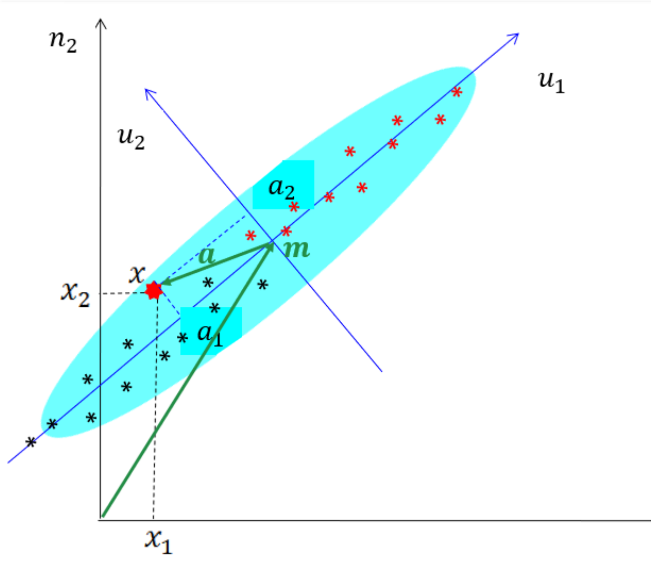
$$\nabla_{\mathbf{u}} L(\mathbf{u}) = 2\mathbf{S}\mathbf{u} - 2\lambda\mathbf{u}$$

- By setting to zero we see that $\mathbf{u}$ is an eigenvector of $\mathbf{S}$:

$$\mathbf{S}\mathbf{u} = \lambda\mathbf{u} \rightarrow \mathbf{u}^T \mathbf{S} \mathbf{u} = \lambda$$

- To maximize $\mathbf{u}^T \mathbf{S} \mathbf{u}$, takes maximal $\lambda_1$, then $\mathbf{u}_1$ is the first principal coordinate.

maximal eigenvalue problem




For symmetric $\mathbf{S}$,
$\{\lambda_i\}$ are real & distinctive,
$\{\mathbf{u}_i\}$ can be orthonormal.

$$\mathbf{S} = \boldsymbol{V}^{-1}\boldsymbol{\Lambda}\boldsymbol{V} = \boldsymbol{V}^T\boldsymbol{\Lambda}\boldsymbol{V}^{-T}$$
$$\rightarrow \boldsymbol{V}^T = \boldsymbol{V}^{-1} \rightarrow \boldsymbol{V}\boldsymbol{V}^T = \mathbf{I}$$

# PCA ; Scatter matrix

- The result is easily extended to $d'$ dimensional projection:

$$\hat{\mathbf{x}}_i = \mathbf{m} + \sum_{k=1}^{d'} a_{ik}\mathbf{u}_k \quad \text{where} \quad d' \leq d$$

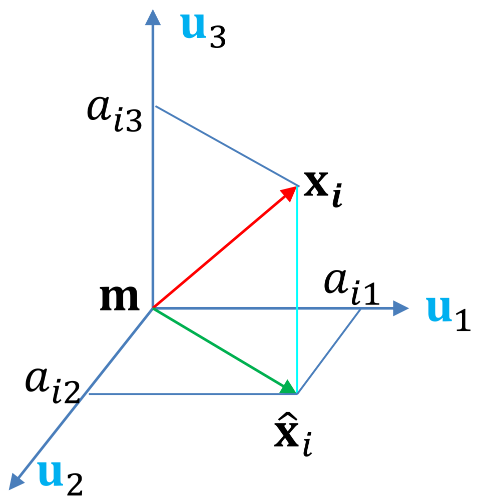
- The criterion function

$$J_{d'} = \sum_{i=1}^{n} \left\| \left( \mathbf{m} + \sum_{k=1}^{d'} a_{ik}\mathbf{u}_k \right) - \mathbf{x}_i \right\|^2, \mathbf{x}_i = \mathbf{m} + \sum_{k=1}^{d} a_{ik}\mathbf{u}_k$$

=sum_(d~d')(lamda^2)

is minimized when $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{d'}$ are the eigenvectors having the largest eigenvalues.

- The coefficients $a_{ik} = \mathbf{u}_k^T(\mathbf{x}_i - \mathbf{m}), k = 1, \ldots, d'$ are *principal components.*



# PCA

- Principal Component Analysis
  - ✓Feature Extraction
  - ✓Dimension Reduction

$$J_{d'} = \sum_{i=1}^{n} \left\| \left( \mathbf{m} + \sum_{k=1}^{d'} a_{ik} \mathbf{u}_k \right) - \mathbf{x}^i \right\|^2$$

$$\mathbf{S} = \sum_{i=1}^{n} ( \mathbf{x}_i \mathbf{x}_i^T - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T$$

$$\mathbf{S} = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i{}^T = \mathbf{X}\mathbf{X}^T$$

where $\mathbf{X} = [\mathbf{x}_1 \ \ \mathbf{x}_2 \ \ \cdots \ \ \mathbf{x}_n \,]$

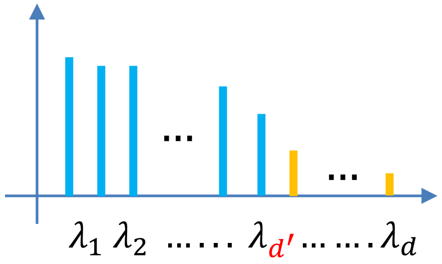
$$\mathbf{S}\mathbf{u} = \lambda \mathbf{u}$$

$$J(\mathbf{u}) = -\mathbf{u}^T \mathbf{S} \mathbf{u} = -\lambda$$

$$a_{ik} = \mathbf{u}_k{}^T (\mathbf{x}_i - \mathbf{m}), k = 1, \cdots, d'$$

$$\begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{id'} \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \vdots \\ \mathbf{u}_{d'}^T \end{bmatrix} (\mathbf{x}_i - \mathbf{m})$$

$$\mathbf{a}_i = \mathbf{U}^T (\mathbf{x}_i - \mathbf{m})$$



# Partial Least Squares

- Matrix-vector form for Multivariate Regression with no-intercept

$$y_i = \mathbf{x}_i^T \theta + \epsilon_i,\ i = 1,\ \cdots,\ n$$
$$\boldsymbol{y} = \mathbf{X}^T \theta + \boldsymbol{\epsilon},\ \mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \ \cdots\ \mathbf{x}_n]$$

$$\boxed{\begin{array}{l} \mathbf{x}_i = [x_{i1} \cdots x_{ip}]^T,\ \theta = [\theta_1\ \cdots\ \theta_p]^T \\ \mathbf{x}_i = \mathbf{x}_i^o - m,\ m = 1/n \sum_i \mathbf{x}_i^o \end{array}}$$

- Goal: reduce the input & parameter dimension: $p > q$

$$\mathbf{x}_i = [x_{i1} \cdots x_{ip}]^T,\ \theta = [\theta_1\ \cdots\ \theta_p]^T \longrightarrow \mathbf{z}_i = [z_{i1} \cdots z_{iq}]^T,\ \theta = [\theta_1\ \cdots\ \theta_q]^T$$

# Principal Component Regression

$\boxed{\mathbf{a}_i \;= \mathbf{U}^T\,(\mathbf{x}_i - \mathbf{m})}$

- Principal Component Analysis for $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_n]$

$$\mathbf{S} = \sum_{i=1}^{n} \mathbf{x}_i\,\mathbf{x}_i{}^T = \mathbf{X}\mathbf{X}^T,\ \ \mathbf{S}\mathbf{u}_k = \lambda_k \mathbf{u}_k, \lambda_1 > \lambda_2 \cdots > \lambda_p$$

$$\boxed{\mathbf{cov}(\mathbf{X},\mathbf{X}) = \frac{1}{n}\mathbf{X}\,\mathbf{X}^T}$$

- Reduced dim. vector ($q < p$ dim.)

$$\underset{\boxed{q\times 1}}{\boldsymbol{z}_i} = \bar{\mathbf{U}}^T\mathbf{x}_i,\ \ \underset{\boxed{p\times q}}{\bar{\mathbf{U}}} = [\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_q]$$

$\xleftarrow[\mathrm{U}^T = \mathrm{U}^{-1}]{\text{Orthonormal eigenvectors } \{\mathbf{u}_i\}}$ $\mathbf{S}$ is symmetric

$$\mathbf{Z} = [\mathbf{z}_1\ \mathbf{z}_2\ \cdots\ \mathbf{z}_n] = \bar{\mathbf{U}}^T\,[\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_n]$$

$$\mathbf{Z} = \bar{\mathbf{U}}^T\mathbf{X} \rightarrow \mathbf{Z}^T = \mathbf{X}^T\bar{\mathbf{U}},\quad \boldsymbol{y} = \mathbf{X}^T\theta + \boldsymbol{\epsilon} \approx \boldsymbol{y} = \mathbf{Z}^T\bar{\mathbf{U}}^T\theta + \boldsymbol{\epsilon} = \boldsymbol{y} = \mathbf{Z}^T\vartheta + \boldsymbol{\epsilon}$$

- Applying LS algorithm to $\boldsymbol{y} = \mathbf{Z}^T\vartheta + \boldsymbol{\epsilon}$

$$\hat{\vartheta} = \underset{\vartheta}{\operatorname{argmin}}\|\boldsymbol{\epsilon}\|^2 = \|\boldsymbol{y} - \mathbf{Z}^T\vartheta\|^2 \rightarrow \boxed{\hat{\vartheta} = (\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\boldsymbol{y} \rightarrow \hat{y} = \boldsymbol{z}^T\hat{\vartheta}\,,\ \ \boldsymbol{z} = \bar{\mathbf{U}}^T\mathbf{x}}$$

# Partial Least Squares

- Nonlinear Iterative Partial Least Squares (NIPALS) algorithm

$$\mathbf{X}\mathbf{X}^T\mathbf{u} = \lambda\mathbf{u}$$

Let $\mathbf{t} = \mathbf{X}^T\mathbf{u}$

$$\mathbf{u} = \frac{1}{\lambda}\mathbf{X}\mathbf{t}$$

Since $\|\mathbf{u}\| := 1 = \frac{1}{\lambda}\|\mathbf{X}\mathbf{t}\|$

$$\lambda = \|\mathbf{X}\mathbf{t}\|$$

$$\bar{\mathbf{U}} = [\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_q]\ ,\ \boldsymbol{z}_i = \bar{\mathbf{U}}^T\mathbf{x}_i$$

$$\mathbf{Z} = \bar{\mathbf{U}}^T\mathbf{X} \rightarrow \mathbf{Z}^T = \mathbf{X}^T\bar{\mathbf{U}},\quad \mathbf{Z} = [\mathbf{z}_1\ \mathbf{z}_2\ \cdots\ \mathbf{z}_{\boldsymbol{n}}]$$

$\mathbf{t} := \mathbf{x}_j$ for some $j$

Loop

$$\mathbf{u} = \mathbf{X}\mathbf{t}/\|\mathbf{X}\mathbf{t}\|$$

$$\mathbf{t} = \mathbf{X}^T\mathbf{u}$$

Until $\mathbf{t}$ stop changing

$$\mathbf{X}^T := \mathbf{X}^T - \mathbf{t}\mathbf{u}^{\boldsymbol{T}} = \mathbf{X}^T(\mathrm{I} - \mathbf{u}\mathbf{u}^{\boldsymbol{T}})$$

Repeat the Loop up to a small $\|\mathbf{X}\mathbf{t}\|$

- Applying LS algorithm to $\boldsymbol{y} = \mathbf{Z}^T\vartheta + \boldsymbol{\epsilon}$

$$\hat{\vartheta} = \underset{\vartheta}{\operatorname{argmin}}\|\boldsymbol{\epsilon}\|^2 = \|\boldsymbol{y} - \mathbf{Z}^T\vartheta\|^2 \rightarrow \hat{\vartheta} = (\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\boldsymbol{y} \rightarrow \hat{\mathrm{y}} = \boldsymbol{z}^T\hat{\vartheta},\ \boldsymbol{z} = \bar{\mathbf{U}}^T\mathbf{x}$$

# Ridge Regression for Regularization

- $l_2$ regularization term is added

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma\|\theta\|_2^2 \; (= S(\theta))$$

- solution:

$$\nabla_\theta\left((\boldsymbol{y} - \mathbf{X}\theta)^T(\boldsymbol{y} - \mathbf{X}\theta) + \gamma\theta^T\theta\right) = 0 \text{ at } \hat{\theta}$$

$$2\Phi^T\left(\boldsymbol{y} - \mathbf{X}\hat{\theta}\right) + 2\gamma\hat{\theta} = 0$$

$$\boxed{\hat{\theta} = (\mathbf{X}^T\mathbf{X} - \gamma\mathbf{I})^{-1}\,\Phi^T\,\boldsymbol{y}}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + G_k(y_{k+1} - \mathbf{x}_{k+1}^T\hat{\theta}_k),$$

$$G_k \cong \frac{\lambda^{-1}P_k\mathbf{x}_{k+1}}{1+\lambda^{-1}\mathbf{x}_{k+1}^T P_k\mathbf{x}_{k+1}}$$

$$P_{k+1} = \lambda^{-1}P_k - \lambda^{-1}G_k\mathbf{x}_{k+1}^T P_k\,, \qquad P_0 = -\frac{1}{\gamma}\mathbf{I}$$

$$P_0 = \alpha\mathbf{I},\; \alpha \gg 1$$

# Lasso Regression for Regularization

- LASSO(Least Absolute Shrinkage Selector Operator)
- $l_1$ regularization term is added

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma\|\theta\|_1$$

- solution: $l_1$ norm is not differentiable→ constrained convex form

  by adding new optimization variables,

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma\mathbf{1}^T\boldsymbol{s}$$

subject to $|\theta_i| \leq s_i, \quad i = 1, \cdots, n$

$$\boxed{\begin{aligned} &\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma\mathbf{1}^T\boldsymbol{s} \\ &\text{subject to } -s_i \leq \theta_i \leq s_i, \quad i = 1, \cdots, n \end{aligned}}$$

# Elastic Regression for Regularization

- Ridge + LASSO

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma_1 \|\theta\|_2^2 + \gamma_2 \|\theta\|_1$$

- solution: $l_1$ norm is not differentiable→ constrained convex form

  by adding new optimization variables,

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma_1 \|\theta\|_2^2 + \gamma_2 \mathbf{1}^T \boldsymbol{s}$$

$$\text{subject to} \quad |\theta_i| \le s_i, \quad i = 1, \cdots, n$$

$$\boxed{\begin{aligned} \hat{\theta} &= \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma_1 \|\theta\|_2^2 + \gamma_2 \mathbf{1}^T \boldsymbol{s} \\ &\text{subject to} \quad -s_i \le \theta_i \le s_i, \quad i = 1, \cdots, n \end{aligned}}$$

# Exercise

- Linear Time varying case with known input

- Regression Model

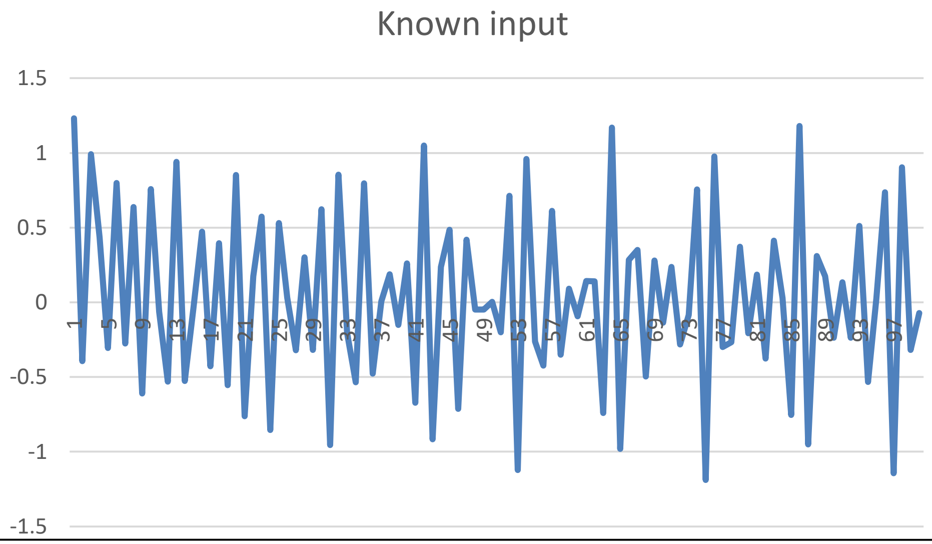
$$y_k = \boldsymbol{\theta}^T \mathbf{x}_k + \epsilon_k,$$

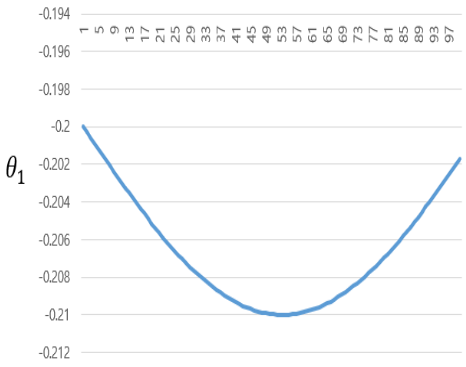
$$\theta^T = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]\ ,\ \mathbf{x}_k = [y_{k-1} \quad y_{k-2} \quad u_{k-1} \quad u_{k-2}]^T$$

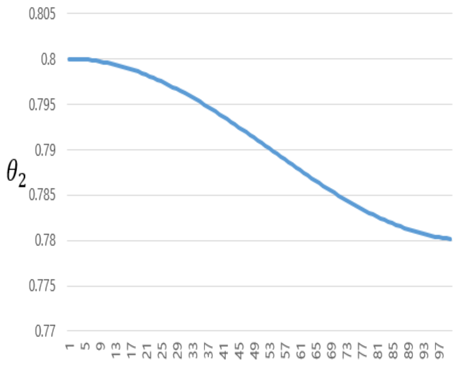
$$\theta_1 = -\big(0.2 + 0.1sin(0.1k)\big),\ \ \theta_2 = \big(0.79 + 0.1cos(0.1k)\big),$$

$$\theta_3 = 0.1,\ \theta_4 = -1.2,\quad u_k = \sin(4k)\cos(0.3k)$$

$$y_0 = 0.45$$
$$y_1 = 0.9$$



# Code (Weighted RLS with Ridge Regularization)

### 1. Load libraries

In [1]:

```
%matplotlib inline

import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn import linear_model
import matplotlib.pyplot as plt
```

### 2. Prepare Data

- Load data
- Split data to training and test data

In [2]:

```
## Load data
df = pd.read_csv('tsp_dataset/ARMA_data2.csv', header=0)
df.head()
```

$$y_k = \boldsymbol{\theta}^T \mathbf{x}_k + \epsilon_k,$$
$$\theta^T = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]\ ,\ \mathbf{x}_k = [y_{k-1} \quad y_{k-2} \quad u_{k-1} \quad u_{k-2}]^T$$
$$\theta_1 = -\big(0.2 + 0.1 sin(0.1k)\big),\ \ \theta_2 = \big(0.79 + 0.1 cos(0.1k)\big),$$
$$\theta_3 = 0.1,\ \ \theta_4 = -1.2,\quad u_k = \sin(4k)\cos(0.3k)$$

Out[2]:

| | y_k | y_k-1 | y_k-2 | u_k-1 | u_k-2 |
|---|---|---|---|---|---|
| 0 | 1.232163 | 0.900000 | 0.450000 | 0.723001 | -0.816553 |
| 1 | -0.394408 | 1.232163 | 0.900000 | 0.000000 | 0.723001 |
| 2 | 0.992526 | -0.394408 | 1.232163 | -0.723001 | 0.000000 |
| 3 | 0.434349 | 0.992526 | -0.394408 | 0.816553 | -0.723001 |
| 4 | -0.306657 | 0.434349 | 0.992526 | -0.333539 | 0.816553 |

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \boldsymbol{X}\theta = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}\theta = \begin{bmatrix} y_0 & y_{-1} & u_0 & u_{-1} \\ y_1 & y_0 & u_1 & u_0 \\ \vdots & \vdots & \vdots & \vdots \\ y_n & y_{n-1} & u_n & u_{n-1} \end{bmatrix}\theta$$

# Code (Weighted RLS with Ridge Regularization)

In [4]:

```python
# Make Training data & Test data (Split data 8:2)
train_length = int(len(df) * 0.8) # use 80% data for training.
train_data = df[:train_length]
X_train = train_data.loc[:, train_data.columns != 'y_k'].to_numpy()
Y_train = train_data.loc[:, train_data.columns == 'y_k'].to_numpy()
test_data = df[train_length:]
X_test  = test_data.loc[:, test_data.columns != 'y_k'].to_numpy()
Y_test  = test_data.loc[:, test_data.columns == 'y_k'].to_numpy()
```

$$
\begin{array}{c}
y_k \\
\boldsymbol{y}=\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}
\end{array}
= \boldsymbol{X}\theta = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}\theta =
\begin{array}{c}
\begin{array}{cccc} y_{k-1} & y_{k-2} & u_{k-1} & u_{k-2} \end{array} \\
\begin{bmatrix} y_0 & y_{-1} & u_0 & u_{-1} \\ y_1 & y_0 & u_1 & u_0 \\ \vdots & \vdots & \vdots & \vdots \\ y_n & y_{n-1} & u_n & u_{n-1} \end{bmatrix}
\end{array}\theta
$$

3 weighted RLS

In [121]:

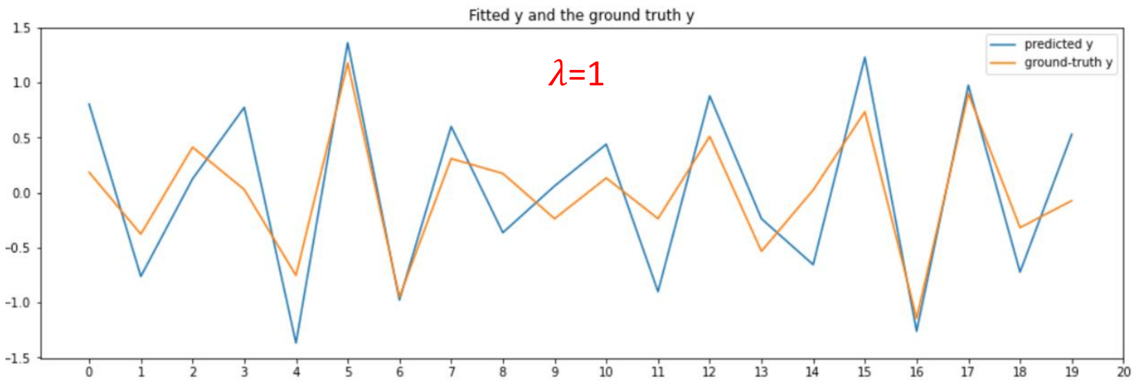
```python
# Define RLS function
# Input: X, Y, initial theta, initial Pk, lambda (We used 1/lambda as lambda here.)
# Output: Final Pk, theta_list
def RLS(X, Y, theta_0, Pk, lamb):
    theta = []
    for k in range(len(X)):
        phi_k1 = X[k].reshape(dim,1)    ;행벡터로 변환
        G_k = lamb*Pk.dot(phi_k1) / (1 + lamb*phi_k1.T.dot(Pk).dot(phi_k1))
        P_next = lamb*Pk - lamb*G_k.dot(phi_k1.T).dot(Pk)
        theta_next = theta_0 + (G_k * (Y[k]-(phi_k1.T).dot(theta_0)))
        # estimation and prediction results
        theta.append(theta_next)

        #update
        theta_0 = theta_next
        Pk = P_next
    return Pk, theta
```

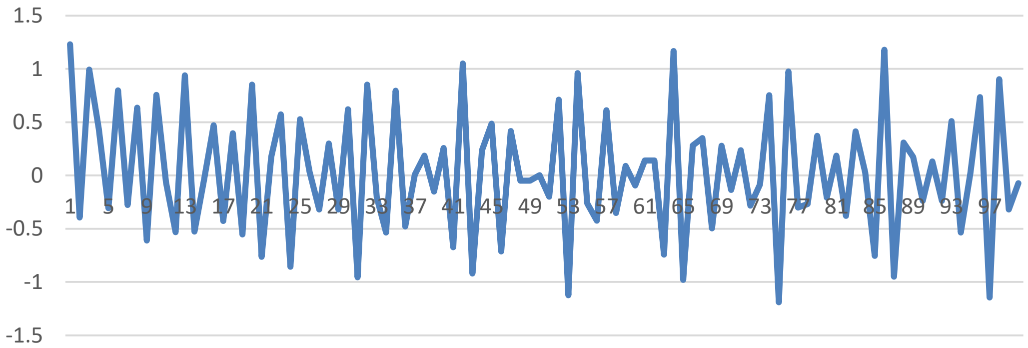
$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + G_{k-1}(y_k - \mathbf{x}_k^T\hat{\boldsymbol{\theta}}_{k-1}),$$

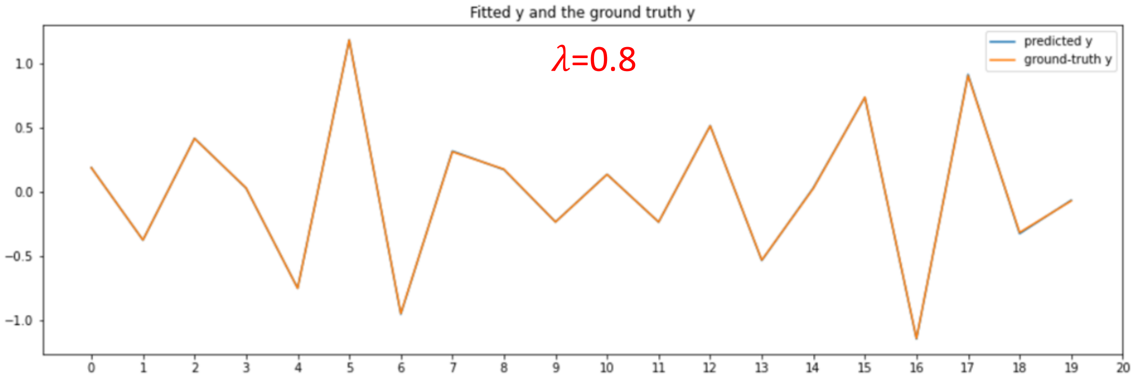
$$G_{k-1} \cong \frac{\lambda^{-1} P_{k-1}\mathbf{x}_k}{1+\lambda^{-1}\mathbf{x}_k^T P_{k-1}\mathbf{x}_k}$$

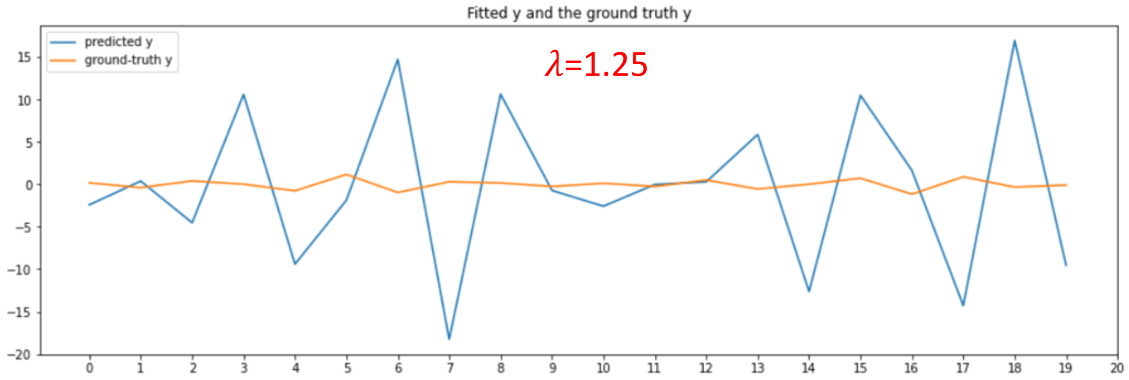
$$P_k = \lambda^{-1} P_{k-1} - \lambda^{-1} G_{k-1}\mathbf{x}_k^T P_{k-1}, \quad P_0 = -\gamma\mathbf{I}$$

# Code (Weighted RLS with Ridge Regularization)






$$\hat{\theta}_{k+1} = (\lambda \mathbf{X}_k^T \mathbf{X}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^T)^{-1}\mathbf{X}_{k+1}^T \boldsymbol{y}_{k+1} \qquad P_0 = -\gamma \mathbf{I},\ \gamma=0.3$$

# Ridge Regression for Regularization

- $l_2$ regularization term is added

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\theta\|^2 + \gamma\|\theta\|_2^2 \left(= S(\theta)\right)$$

- solution:

$$\nabla_\theta\left((\boldsymbol{y} - \mathbf{X}\theta)^T (\boldsymbol{y} - \mathbf{X}\theta) + \gamma\theta^T\theta\right) = 0 \text{ at } \hat{\theta}$$

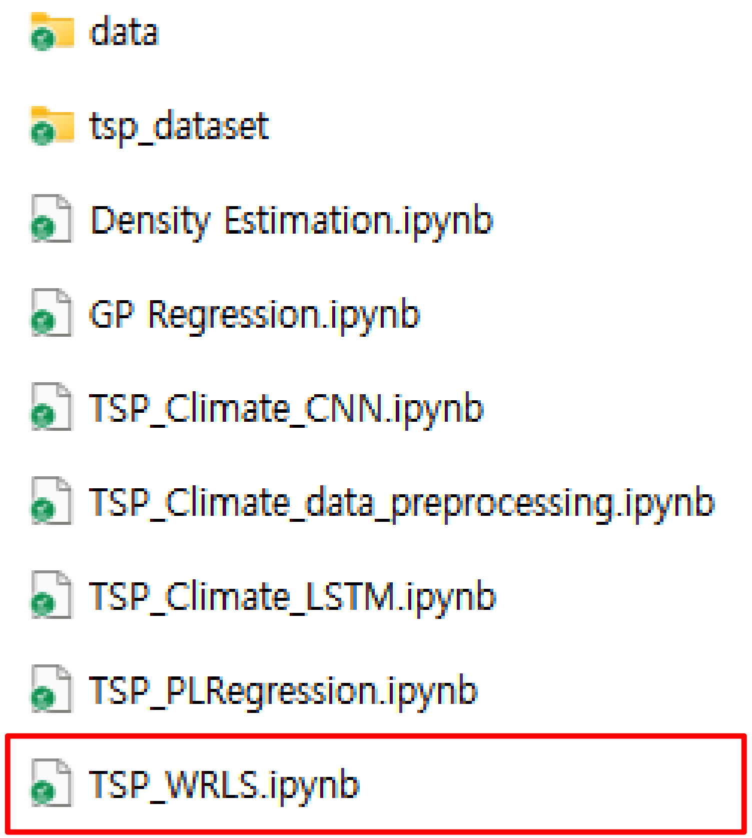
$$2\Phi^T\left(\boldsymbol{y} - \mathbf{X}\hat{\theta}\right) + 2\gamma\hat{\theta} = 0$$

$$\boxed{\hat{\theta} = (\mathbf{X}^T\mathbf{X} - \gamma\mathbf{I})^{-1}\,\Phi^T\,\boldsymbol{y}}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + G_k(y_{k+1} - \mathbf{x}_{k+1}^T\hat{\theta}_k),$$

$$G_k \cong \frac{\lambda^{-1}P_k\mathbf{x}_{k+1}}{1+\lambda^{-1}\mathbf{x}_{k+1}^T P_k\mathbf{x}_{k+1}}$$

$$P_{k+1} = \lambda^{-1}P_k - \lambda^{-1}G_k\mathbf{x}_{k+1}^T P_k\,,\; P_0 = -\gamma\mathbf{I} \quad P_0 = \alpha\mathbf{I},\; \alpha \gg 1$$



# NONLINEAR REGRESSION

JIN YOUNG CHOI

ECE, SEOUL NATIONAL UNIVERSITY

# Parameter Estimation in Matrix form

- $\hat{f}$ can be estimated from the sample pairs $\{(y_i, x_i) | \; i = 1, 2, \cdots, n\}$

$$y_k = \hat{f}(x_k) + \epsilon_k, \;\; k = 1, \;\; \cdots, \;\; n,$$

where $\epsilon_k$ are i.i.d. zero mean and variance $\sigma^2$

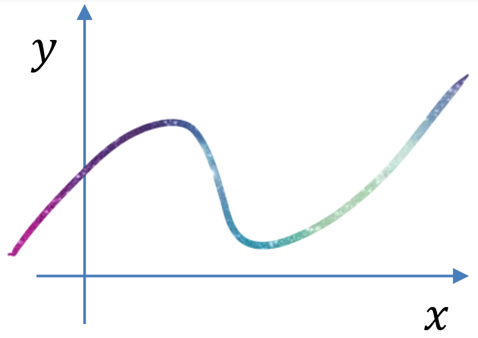
- $\hat{f}$ is an arbitrary nonlinear function
- Nonlinear regression models
  - Higher order polynomial linear model
  - Sine/cosine basis linear model
  - Radial basis linear model
  - Wavelet basis linear model
  - Piecewise linear model
  - Neural network model (Sigmoid basis linear model)
    - Convolution neural network model
    - LSTM model (time series model)

# High-order Regression

- High-order polynomial regression model

$$Y = \theta_0 + \theta_1 X + \theta_2 X^2 + \cdots + \theta_m X^m + \epsilon$$

$$y_i = \theta_0 + \theta_1 x_i + \theta_2 x_i^2 + \cdots + \theta_m x_i^m + \epsilon_i, \ i = 1, \ \cdots, \ n.$$




- High-order multivariate regression model

$$Y = \theta_0 + \theta_1 X_1 + \cdots + \theta_k X_k + \cdots + \theta_{k(\pi)} X_{\pi_1} \cdots X_{\pi_j} \cdots + \cdots + \theta_p X_{\mu(m)}^m + \epsilon$$

$$y_i = \theta_0 + \theta_1 x_{i1} + \cdots + \theta_k x_{ik} + \cdots + \theta_{k(\pi)} x_{i\pi_1} \cdots x_{i\pi_j} \cdots + \cdots + \theta_M x_{ip}^m + \epsilon_i$$

- Matrix-vector form

Let $\theta = [\theta_0 \ \ \theta_1 \ \ \cdots \ \theta_p]^T, \quad \phi_i = [1 \ \ \phi_{i1} \cdots \phi_{ip}]^T$

$y = [y_1 \ \ y_2 \ \ \cdots \ \ y_n]^T, \quad \epsilon = [\epsilon_1 \ \ \epsilon_2 \ \ \cdots \ \ \epsilon_n]^T$

Then $y_i = \phi_i^T \theta + \epsilon_i, \ i = 1, \ \cdots, \ n.$

$y = \Phi\theta + \epsilon, \quad \Phi = [\phi_1 \ \phi_2 \ \ \cdots \ \phi_n]^T$

$$\Phi = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ 1 & x_3 & x_3^2 & \cdots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^m \end{bmatrix}$$

# Basis-function Regression

- Matrix-vector form of General Regression

Let $\theta = [\theta_0 \ \ \theta_1 \ \ \cdots \ \theta_p]^T, \quad \phi_i = [1 \ \ \phi_{i1} \cdots \phi_{ip}]^T$

$\boldsymbol{y} = [y_1 \ \ y_2 \ \ \cdots \ \ y_n]^T, \quad \boldsymbol{\epsilon} = [\epsilon_1 \ \ \epsilon_2 \ \ \cdots \ \ \epsilon_n]^T$

Then $y_i = \phi_i^T \theta + \epsilon_i, \ i = 1, \ \cdots, \ n.$

$\boldsymbol{y} = \Phi\theta + \boldsymbol{\epsilon}, \quad \Phi = [\phi_1 \ \phi_2 \ \ \cdots \ \phi_n]^T$

- Basis for General Regression
  - sin, cos basis: $\phi_{im} = \sin \omega_m x_i$ or $\cos \omega_m x_i$
  - radial basis: $\phi_{im} = \exp \frac{-\|x_i - \mu_m\|^2}{\sigma_m^2}$
  - sigmoid basis: $\phi_{im} = \frac{1}{1+\exp(-w_m^T x_i - b_m)}$ or $\frac{\exp(w_m^T x_i + b_m)}{1+\exp(w_m^T x_i + b_m)}$

Logistic Regression

# Parameter Estimation in Matrix form

- Least Squares Estimation

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \|\boldsymbol{\epsilon}\|^2 = \|\boldsymbol{y} - \Phi\theta\|^2 \cong S(\theta)$$

$\boxed{\boldsymbol{y} - \mathbf{X}\theta}$

Solution:

$$\nabla_\theta S(\theta) = 0 \text{ at } \hat{\theta}$$

$$\nabla_\theta (\boldsymbol{y} - \Phi\theta)^T (\boldsymbol{y} - \Phi\theta) = 0 \text{ at } \hat{\theta}$$

$$2\Phi^T \left(\boldsymbol{y} - \Phi\hat{\theta}\right) = 0$$

$$\Phi^T \boldsymbol{y} - \Phi^T \Phi\hat{\theta} = 0$$

$$\hat{\theta} = (\Phi^T\Phi)^{-1} \Phi^T \boldsymbol{y}$$

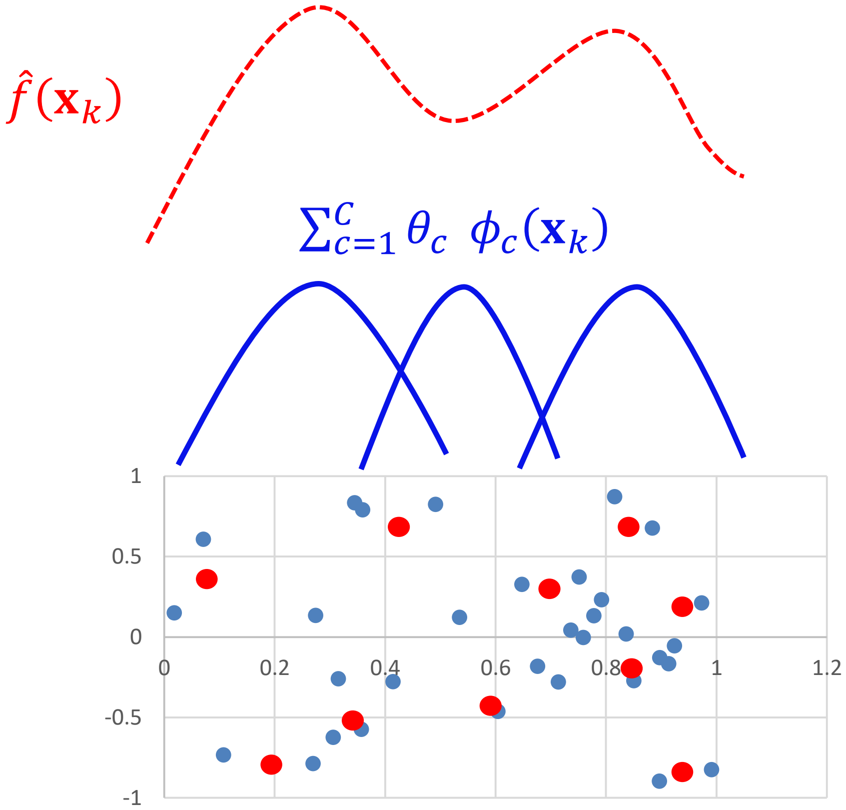
# Radial Basis Regression Model

- Radial Basis Regression Model
  $$y_k = \hat{f}(x_k) + \epsilon_k = \sum_{c=1}^{C} \theta_c \ \phi_c(\mathbf{x}_k) + \epsilon_k = \boldsymbol{\theta}^T \boldsymbol{\phi}_k + \epsilon_k,$$
  $$\phi_c(\mathbf{x}_k) = \frac{g(\mathbf{x}_k, \mu_c, \sigma_c^2)}{\sum_{c=1}^{C} g(\mathbf{x}_k, \mu_c, \sigma_c^2)}, \quad g(\mathbf{x}_k, \mu_c, \sigma_c^2) = e^{-\|\mathbf{x}_k - \mu_c\|^2/\sigma_c^2}$$
- Parameters, regression variables
  $$\mathbf{x}_k = [x_1, \cdots, x_d]^T$$
  $$\boldsymbol{\phi}_k = [\phi_1(\mathbf{x}_k), \phi_2(\mathbf{x}_k), \cdots, \phi_C(\mathbf{x}_k)]^T$$
  $$\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_C]^T$$
- Apply LSE with affine AR Model
  - $\Phi^T = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_K]$
  - $\boldsymbol{y} = [y_1, y_2, \cdots, y_K]^T$
  - $\boldsymbol{y} = \Phi\boldsymbol{\theta} + \boldsymbol{\epsilon}$
  - Apply LSE: $\widehat{\boldsymbol{\theta}} = (\Phi^T\Phi)^{-1}\Phi^T \boldsymbol{y}$
- How to determine $\mu_c, \ \sigma^2$?

  $(\mu_c, \sigma_c^2)$: mean and variance of $c$-th cluster of $\{\mathbf{x}_k\}$

  K-means clustering

  Expectation-Maximization algorithm



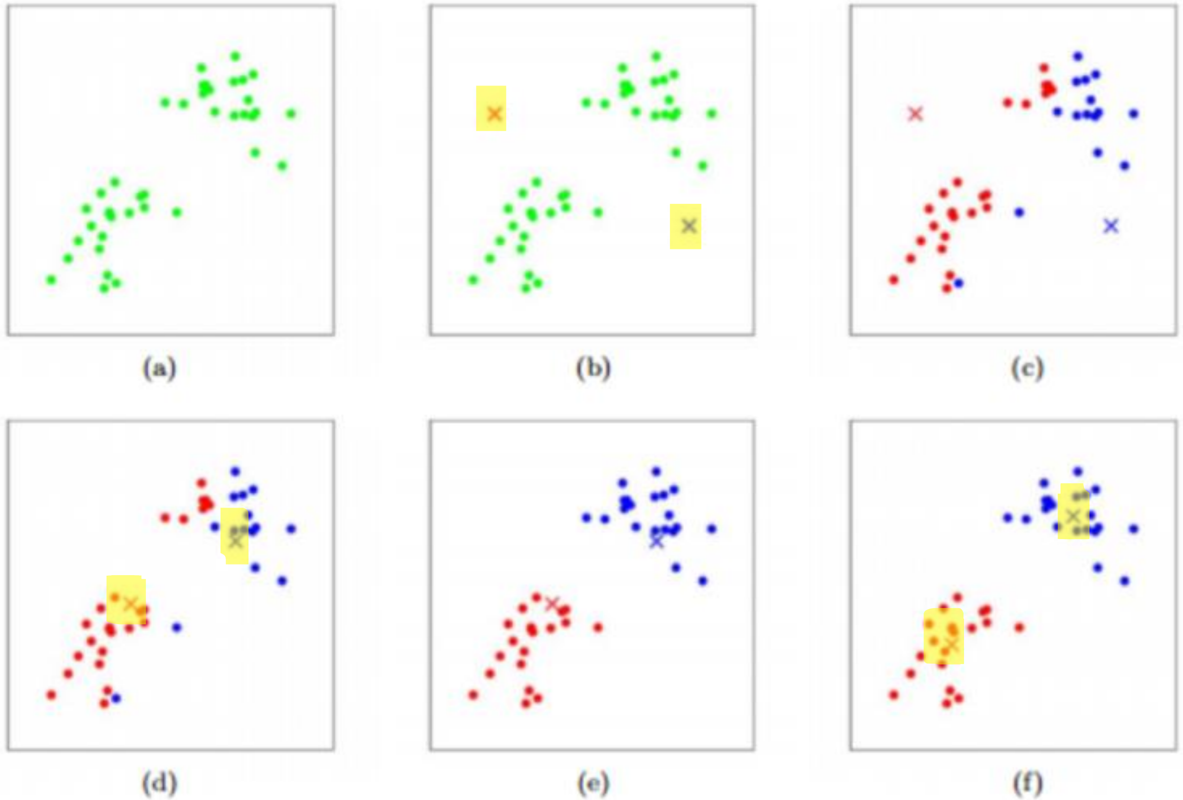
# Code

1. 일단 K개의 임의의 중심점(centroid)을 배치하고
2. 각 데이터들을 가장 가까운 중심점으로 할당한다. (일종의 군집을 형성한다.)
3. 군집으로 지정된 데이터들을 기반으로 해당 군집의 중심점을 업데이트한다.
4. 2번, 3번 단계를 그래서 수렴이 될 때까지, 즉 더이상 중심점이 업데이트 되지 않을 때까지 반복한다.

- Radial Basis Regression Model (Manifold Regularization)

$$y_k = f(y_{k-1}, u_{k-1}) = f(\mathbf{x}_k) \approx \sum_{c=1}^{C} \theta_c \; \phi_c(\mathbf{x}_k) = \boldsymbol{\theta}^T \boldsymbol{\phi}_k,$$

$$\phi_c(\mathbf{x}_k) = \frac{g(\mathbf{x}_k, \mu_c, \sigma_c^2)}{\sum_{c=1}^{C} g(\mathbf{x}_k, \mu_c, \sigma_c^2)}, \quad g(\mathbf{x}_k, \mu_c, \sigma_c^2) = e^{-\|\mathbf{x}_k - \mu_c\|^2/\sigma_c^2}$$



# Piecewise Linear Regression Model

- Piecewise Linear Regression Model

$$y_k = \sum_{c=1}^{C} \boldsymbol{\theta}_c^T \mathbf{x}_k \phi_c(\mathbf{x}_k),\ \phi_c(\mathbf{x}_k) = \frac{g(\mathbf{x}_k, \mu_c, \sigma^2)}{\sum_{c=1}^{C} g(\mathbf{x}_k, \mu_c, \sigma^2)},$$

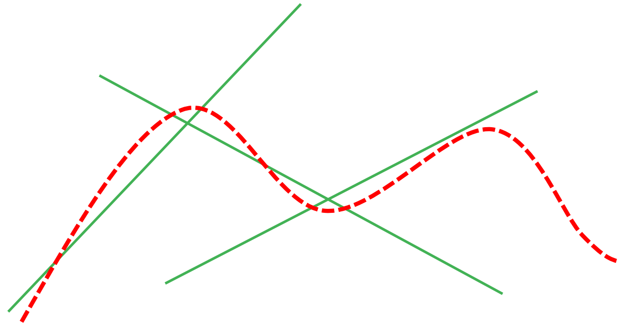
$$y_k = \underbrace{[\boldsymbol{\theta}_1^T \quad \cdots \quad \boldsymbol{\theta}_C^T]}_{\text{concatenated vector}} \underbrace{\begin{bmatrix} \mathbf{x}_k \phi_1(\mathbf{x}_k) \\ \vdots \\ \mathbf{x}_k \phi_C(\mathbf{x}_k) \end{bmatrix}}_{\text{concatenated vector}} = \underbrace{\boldsymbol{\theta}^T \boldsymbol{\psi}_k}_{\text{Scalar}}$$

$$\boxed{y_k = \sum_{c=1}^{C} \theta_c \ \phi_c(\mathbf{x}_k)}$$

$$y_{c,k} = \boldsymbol{\theta}_c^T \mathbf{x}_k$$

$$\mathbf{x}_k = [x_1, x_2, \cdots, x_p, 1]^T$$

$$\boldsymbol{\theta}_c^T = [\theta_{c,1}, \theta_{c,2}, \cdots, \theta_{c,n}, \theta_{c,0}]$$



- Apply Least Squares Estimation,

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi})^{-1} \boldsymbol{\Psi}^T \mathbf{y}$$

$$\boldsymbol{\Psi}^T = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_1, \cdots, \boldsymbol{\psi}_K]$$

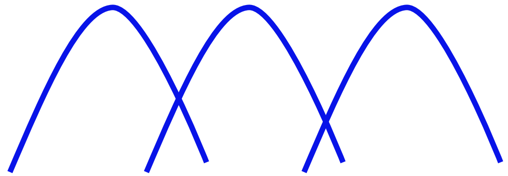
$$\mathbf{y} = [y_1, y_2, \cdots, y_K]^T = \boldsymbol{\Psi}\boldsymbol{\theta}$$

- How to determine $\mu_c,\ \sigma^2$?

  $(\mu_c, \sigma_c^2)$: mean and variance of $c$-th cluster of $\{\mathbf{x}_k\}$

  K-means clustering

  Expectation-Maximization algorithm



# RBF Regression & Piecewise Linear Regression

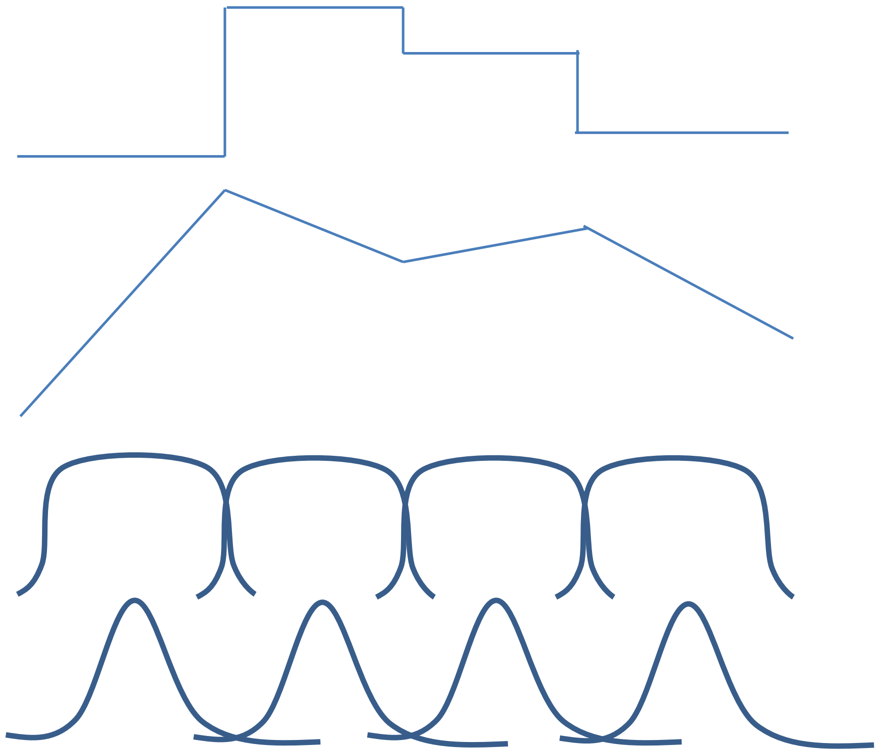
- Radial basis regression과 Piecewise Linear regression의 차이.
- local constant approximation과 local linear approximation 의 차이임. 다차원으로 갈 수록 차이가 심해 짐.

$y_k = \theta_{c,1}$ around $\mu_c$

$y_k = \theta_{c,0} + \theta_{c,1}x_{k,1} + \cdots + \theta_{c,n}x_{k,n} = \boldsymbol{\theta}_c^T\mathbf{x}_k$ around $\mu_c$

$$y_k = \sum_{c=1}^{C} \theta_c \; \phi_c(\mathbf{x}_k)$$

$$y_k = \sum_{c=1}^{C} \boldsymbol{\theta}_c^T\mathbf{x}_k\phi_c(\mathbf{x}_k),$$



# LSE: regression variable, parameters dim., Matrix, vector form

$$y_k = \theta_1 y_{k-1} + \theta_2\, y_{k-2} + \theta_3 u_{k-1} + \theta_4 u_{k-2} \rightarrow \theta^T \mathbf{x}_k$$

$$\theta^T = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4], \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ u_{k-1} \\ u_{k-2} \end{bmatrix} = \mathbf{x}_k$$

| | y_k | y_k-1 | y_k-2 | u_k-1 | u_k-2 |
|---|---|---|---|---|---|
| 0 | 1.232163 | 0.900000 | 0.450000 | 0.723001 | -0.816553 |
| 1 | -0.394408 | 1.232163 | 0.900000 | 0.000000 | 0.723001 |
| 2 | 0.992526 | -0.394408 | 1.232163 | -0.723001 | 0.000000 |
| 3 | 0.434349 | 0.992526 | -0.394408 | 0.816553 | -0.723001 |
| 4 | -0.306657 | 0.434349 | 0.992526 | -0.333539 | 0.816553 |

$$\begin{aligned} y_0 &= \theta^T \mathbf{x_0} \\ y_1 &= \theta^T \mathbf{x_1} \\ y_2 &= \theta^T \mathbf{x_2} \\ y_3 &= \theta^T \mathbf{x_3} \\ y_4 &= \theta^T \mathbf{x_4} \end{aligned} \rightarrow \begin{aligned} y_0 &= \mathbf{x}_0^T \theta \\ y_1 &= \mathbf{x}_1^T \theta \\ y_2 &= \mathbf{x}_2^T \theta \\ y_3 &= \mathbf{x}_3^T \theta \\ y_4 &= \mathbf{x}_4^T \theta \end{aligned} \rightarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0^T \theta \\ \mathbf{x}_1^T \theta \\ \mathbf{x}_2^T \theta \\ \mathbf{x}_3^T \theta \\ \mathbf{x}_4^T \theta \end{bmatrix} \rightarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0^T \\ \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \end{bmatrix} \theta$$

$$\mathbf{y} = \mathbf{X}\theta$$

$$\mathbf{x}_k^T = [y_{k-1} \quad y_{k-2} \quad u_{k-1} \quad u_{k-2}], \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \theta$$

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} y_{-1} & y_{-2} & u_{-1} & u_{-2} \\ y_0 & y_{-1} & u_0 & u_{-1} \\ y_1 & y_0 & u_1 & u_0 \\ y_2 & y_1 & u_2 & u_1 \\ y_3 & y_3 & u_3 & u_2 \end{bmatrix} \theta$$

$$\mathbf{y} = \mathbf{X}\theta \rightarrow \mathbf{X}^T\mathbf{X}\theta = \mathbf{X}^T\mathbf{y} \rightarrow \theta = \mathbf{X}^T\mathbf{X}^{-1}\mathbf{X}^T\mathbf{y}$$

# Piecewise LR model

$$y_k = \sum_{c=1}^{C} \boldsymbol{\theta}_c^T \mathbf{x}_k \phi_c(\mathbf{x}_k)$$
$$= \boldsymbol{\theta}_1^T \mathbf{x}_k \phi_1(\mathbf{x}_k) + \boldsymbol{\theta}_2^T \mathbf{x}_k \phi_2(\mathbf{x}_k) + \boldsymbol{\theta}_3^T \mathbf{x}_k \phi_c(\mathbf{x}_k)$$
$$= [\boldsymbol{\theta}_1^T \quad \boldsymbol{\theta}_2^T \quad \boldsymbol{\theta}_3^T] \begin{bmatrix} \mathbf{x}_k \phi_1(\mathbf{x}_k) \\ \mathbf{x}_k \phi_2(\mathbf{x}_k) \\ \mathbf{x}_k \phi_3(\mathbf{x}_k) \end{bmatrix}$$
$$= [\theta_{1,1} \quad \theta_{1,2} \quad \theta_{1,3} \quad \theta_{1,4} \quad \theta_{2,1} \quad \theta_{2,2} \quad \theta_{2,3} \quad \theta_{2,4} \quad \theta_{3,1} \quad \theta_{3,2} \quad \theta_{3,3} \quad \theta_{4,4}] \begin{bmatrix} \varphi_{1,1}(\mathbf{x}_k) \\ \varphi_{1,2}(\mathbf{x}_k) \\ \varphi_{1,3}(\mathbf{x}_k) \\ \varphi_{1,4}(\mathbf{x}_k) \\ \varphi_{2,1}(\mathbf{x}_k) \\ \varphi_{2,2}(\mathbf{x}_k) \\ \varphi_{2,3}(\mathbf{x}_k) \\ \varphi_{2,4}(\mathbf{x}_k) \\ \varphi_{3,1}(\mathbf{x}_k) \\ \varphi_{3,2}(\mathbf{x}_k) \\ \varphi_{3,3}(\mathbf{x}_k) \\ \varphi_{3,4}(\mathbf{x}_k) \end{bmatrix} = \theta^T \varphi_k$$

$$\theta_c^T = [\theta_{c,1} \quad \theta_{c,2} \quad \theta_{c,3} \quad \theta_{c,4}], \ \mathbf{x}_k = \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ u_{k-1} \\ u_{k-2} \end{bmatrix}$$

$$\mathbf{x}_k \phi_c(\mathbf{x}_k) = \begin{bmatrix} y_{k-1} \\ y_{k-2} \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \phi_c(\mathbf{x}_k) = \begin{bmatrix} y_{k-1}\phi_c(\mathbf{x}_k) \\ y_{k-2}\phi_c(\mathbf{x}_k) \\ u_{k-1}\phi_c(\mathbf{x}_k) \\ u_{k-2}\phi_c(\mathbf{x}_k) \end{bmatrix} := \begin{bmatrix} \varphi_{c,1}(\mathbf{x}_k) \\ \varphi_{c,2}(\mathbf{x}_k) \\ \varphi_{c,3}(\mathbf{x}_k) \\ \varphi_{c,4}(\mathbf{x}_k) \end{bmatrix}$$

# Monthly Weather Timeseries Forecasting

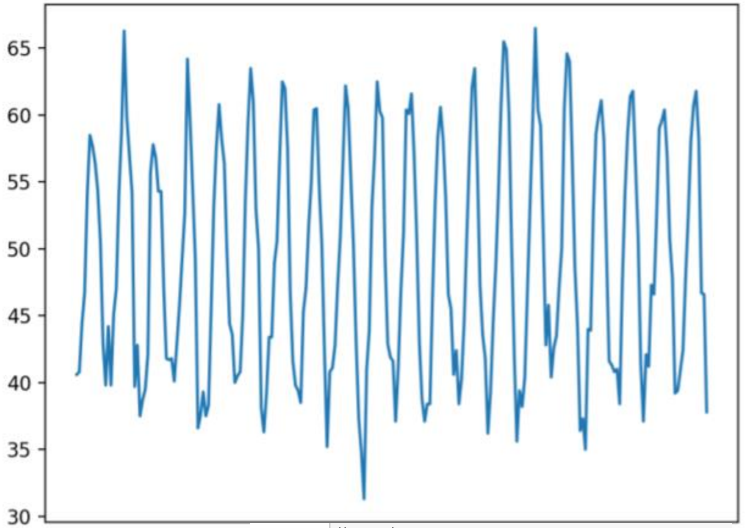
$$y_k = \theta_{c,0} + \theta_{c,1} y_{k-1} + \cdots + \theta_{c,n} y_{k-p} = \boldsymbol{\theta}_c^T \mathbf{x}_k$$

In [29]:
```python
## Make data for training
def make_batch(input_data, sl):
    train_x = []
    train_y = []
    L = len(input_data)
    for i in range(L-sl):
        train_seq = input_data[i:i+sl]
        train_label = input_data[i+sl:i+sl+1]
        train_x.append(train_seq)
        train_y.append(train_label)
    return train_x, train_y
```

In [30]:
```python
sequence_length = 3 # Hyperparameter

# Make Training data & Test data (Split data 8:2)
X, Y = make_batch(df['Temperature'].to_numpy(), sequence_length)
train_length = int(len(Y) * 0.8) # use 80% data for training
X_train = X[:train_length]
X_train = np.hstack((X_train,np.ones((len(X_train),1)))) #Add 1 for intercept
Y_train = Y[:train_length]

X_test = X[train_length:]
X_test = np.hstack((X_test,np.ones((len(X_test),1)))) #Add 1 for intercept
Y_test = Y[train_length:]
```

```
"Month","Temperature"
"1920-01",40.6
"1920-02",40.8
"1920-03",44.4
"1920-04",46.7
"1920-05",54.1
"1920-06",58.5
"1920-07",57.7
"1920-08",56.4
"1920-09",54.3
"1920-10",50.5
"1920-11",42.9
"1920-12",39.8
"1921-01",44.2
"1921-02",39.8
"1921-03",45.1
"1921-04",47.0
"1921-05",54.1
"1921-06",58.7
"1921-07",66.3
"1921-08",59.9
"1921-09",57.0
"1921-10",54.2
"1921-11",39.7
"1921-12",42.8
"1922-01",37.5
"1922-02",38.7
"1922-03",39.5
"1922-04",42.1
"1922-05",55.7
"1922-06",57.8
"1922-07",56.8
"1922-08",54.3
"1922-09",54.3
"1922-10",47.1
"1922-11",41.8
"1922-12",41.7
"1923-01",41.8
```

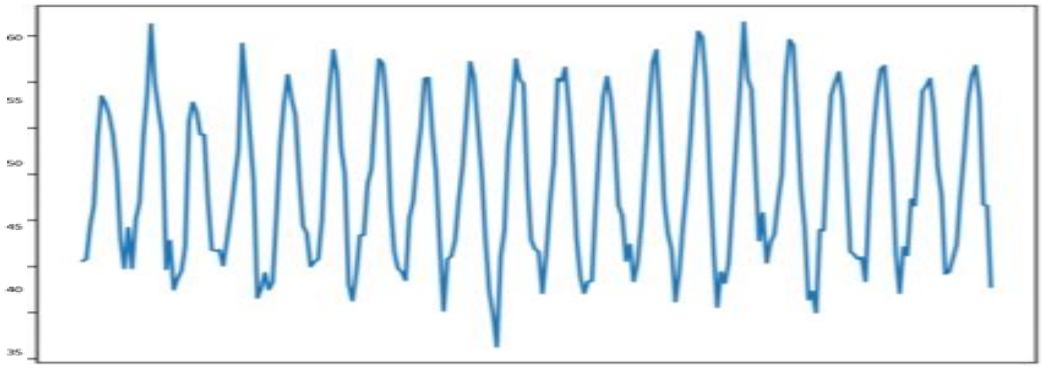
$\mathbf{x}_1$ $y_1$ $\mathbf{x}_k$ $y_k$

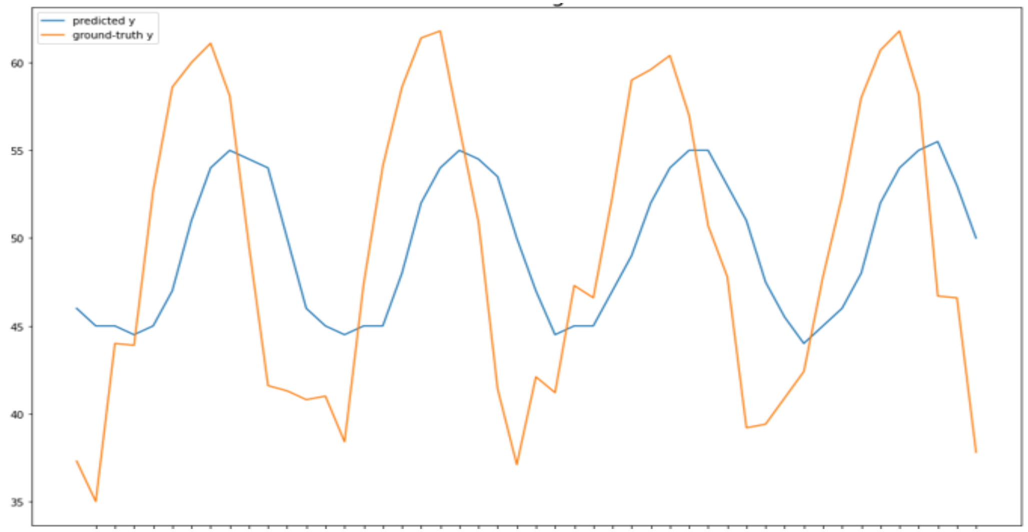


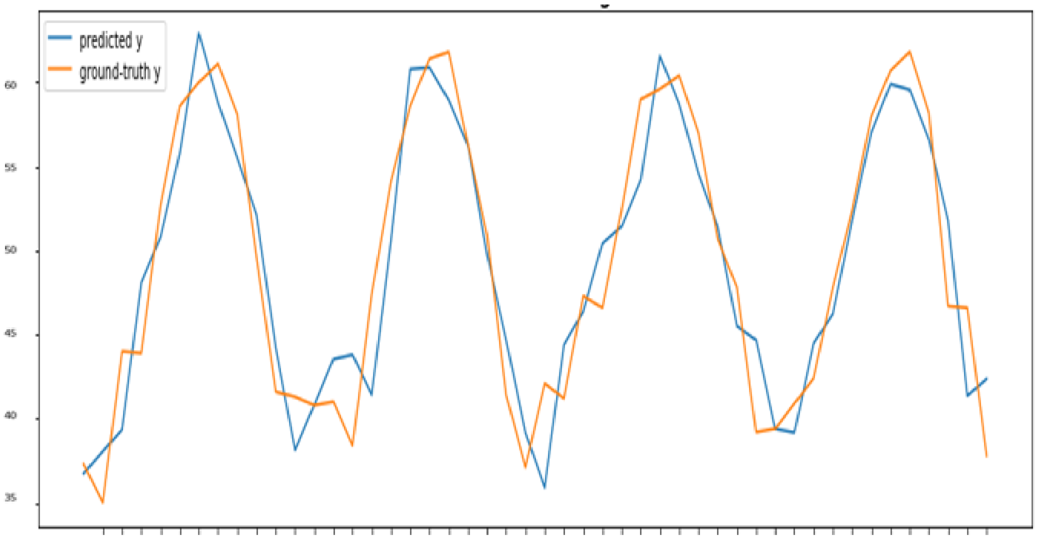

X_train

Out[6]:
```
array([[40.6, 40.8, 44.4,  1. ],
       [40.8, 44.4, 46.7,  1. ],
       [44.4, 46.7, 54.1,  1. ],
       [46.7, 54.1, 58.5,  1. ],
       [54.1, 58.5, 57.7,  1. ],
       [58.5, 57.7, 56.4,  1. ],
       [57.7, 56.4, 54.3,  1. ],
       [56.4, 54.3, 50.5,  1. ],
       [54.3, 50.5, 42.9,  1. ],
       [50.5, 42.9, 39.8,  1. ],
       [42.9, 39.8, 44.2,  1. ],
       [39.8, 44.2, 39.8,  1. ],
       [44.2, 39.8, 45.1,  1. ],
       [39.8, 45.1, 47. ,  1. ],
       [45.1, 47. , 54.1,  1. ],
       [47. , 54.1, 58.7,  1. ],
       [54.1, 58.7, 66.3,  1. ],
       [58.7, 66.3, 59.9,  1. ],
       [66.3, 59.9, 57. ,  1. ],
```

# Monthly Weather Timeseries Forecasting

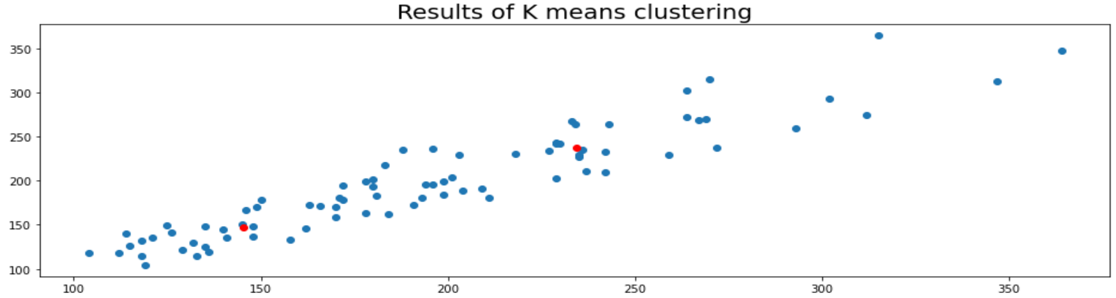



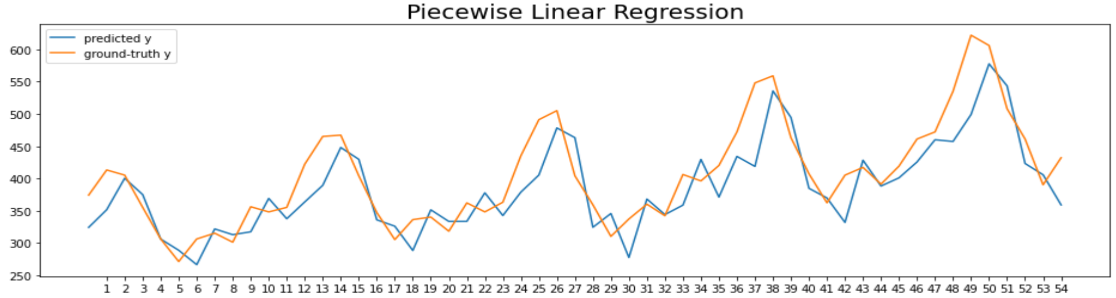


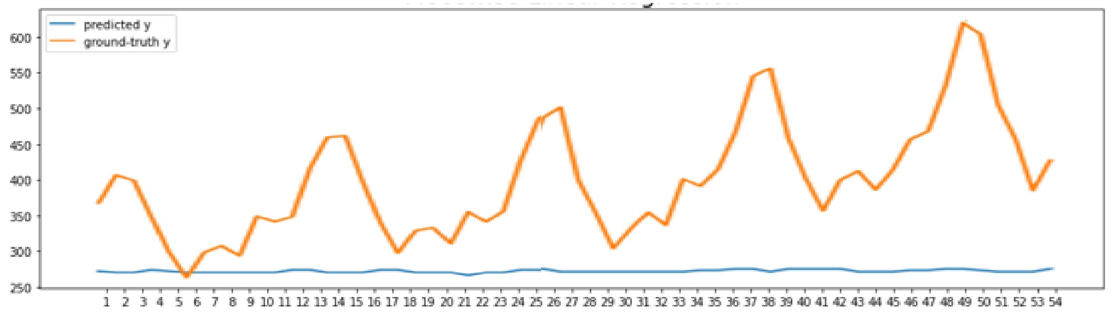



# Passenger Timeseries

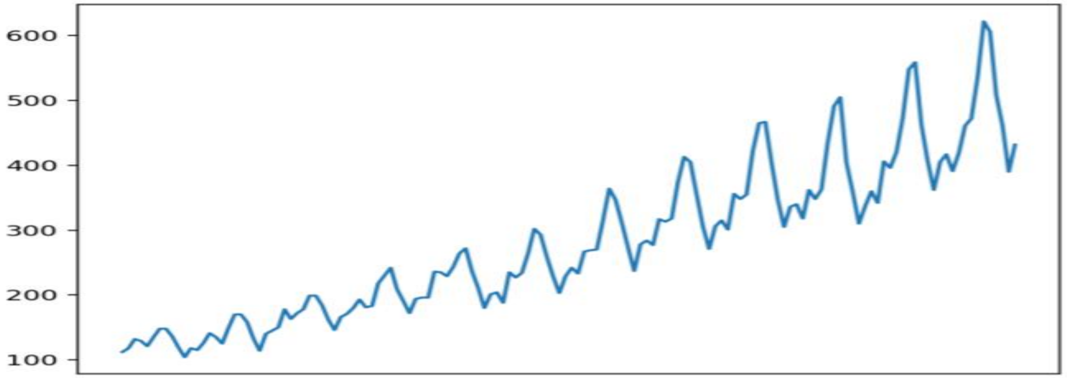
```
n_cluster = 2
PR_CLASSIFIER = PiecewiseRegression(X_train, Y_train, X_test, Y_test, k=n_cluster, plotKmeans=True, usePiewise=True)
PR_CLASSIFIER.fit()
```




**Non stationary (Trends) process**



# Piecewise Linear Regression Model

- Piecewise Linear Regression Model

$$y_k = \sum_{c=1}^{C} \boldsymbol{\theta}_c^T \mathbf{x}_k \phi_c(\mathbf{x}_k),\ \phi_c(\mathbf{x}_k) = \frac{g(\mathbf{x}_k, \mu_c, \sigma^2)}{\sum_{c=1}^{C} g(\mathbf{x}_k, \mu_c, \sigma^2)},$$

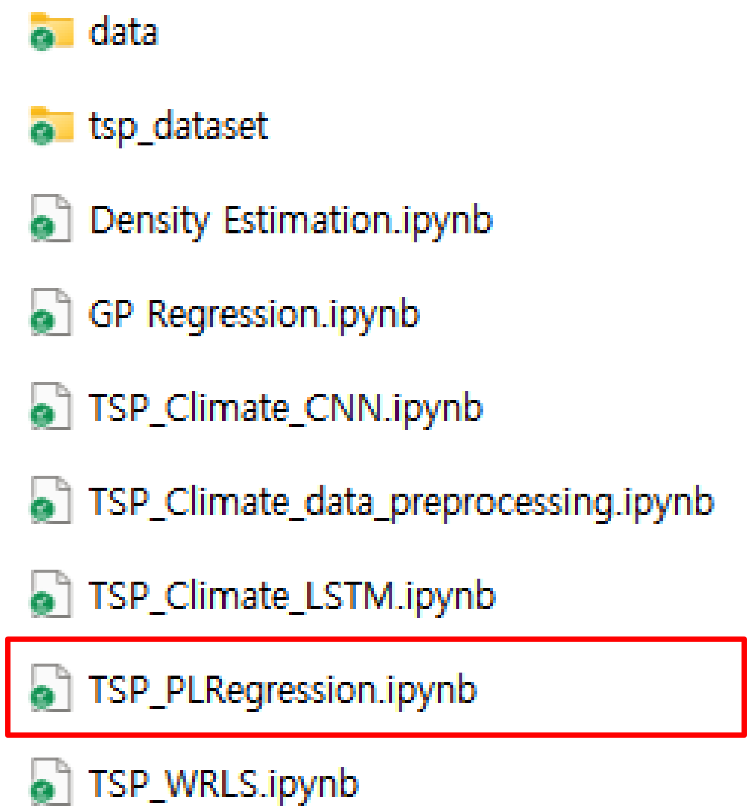
$$y_k = [\boldsymbol{\theta}_1^T \quad \cdots \quad \boldsymbol{\theta}_C^T] \begin{bmatrix} \mathbf{x}_k \phi_1(\mathbf{x}_k) \\ \vdots \\ \mathbf{x}_k \phi_C(\mathbf{x}_k) \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{\psi}_k$$

- Apply Least Squares Estimation,

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Psi}^T \boldsymbol{\Psi})^{-1} \boldsymbol{\Psi}^T \mathbf{y}$$

$$\boldsymbol{\Psi}^T = [\boldsymbol{\psi}_1, \boldsymbol{\psi}_1, \cdots, \boldsymbol{\psi}_K]$$

$$\mathbf{y} = [y_1, y_2, \cdots, y_K]^T = \boldsymbol{\Psi\theta}$$

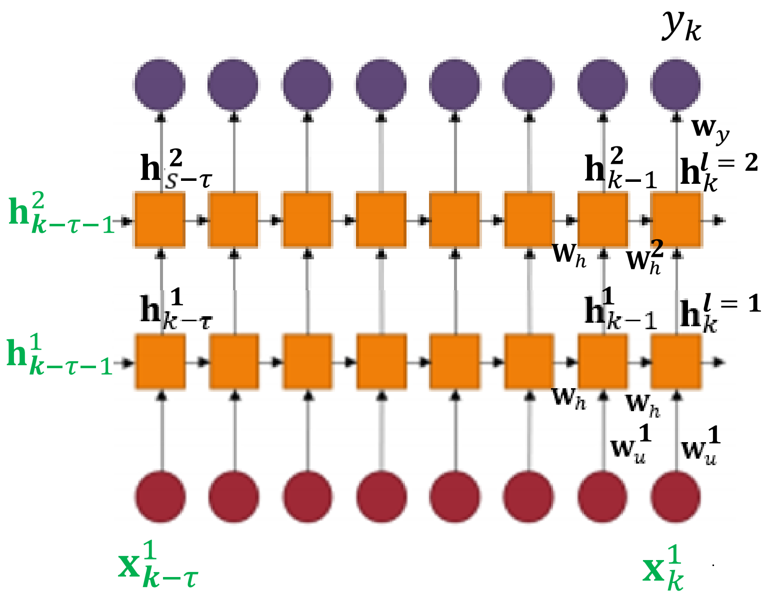


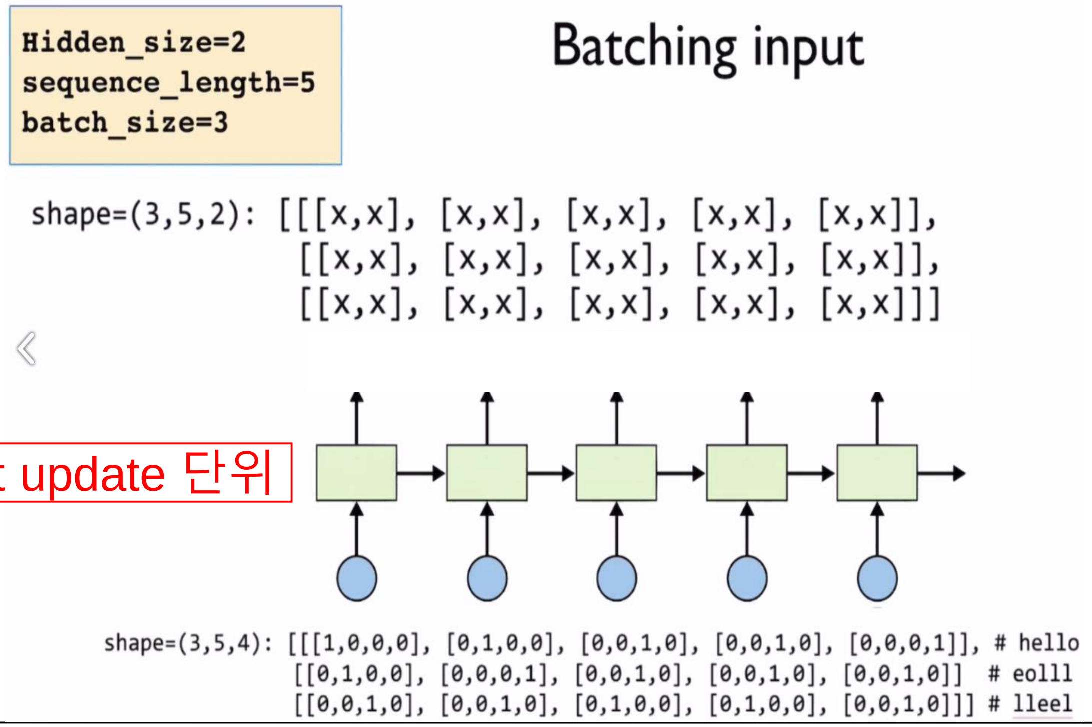
# Recurrent Neural Network

- Recurrent Neural Networks(RNN)

$$\mathbf{h}_k^l = \sigma\left(\mathbf{W}_h^l \mathbf{h}_{k-1}^l + \mathbf{W}_u^l \mathbf{h}_k^{l-1} + b_h^l\right), \quad \mathbf{h}_k^0 = \mathbf{x}_k$$

$$y_k = w_y^T \mathbf{h}_k^L$$



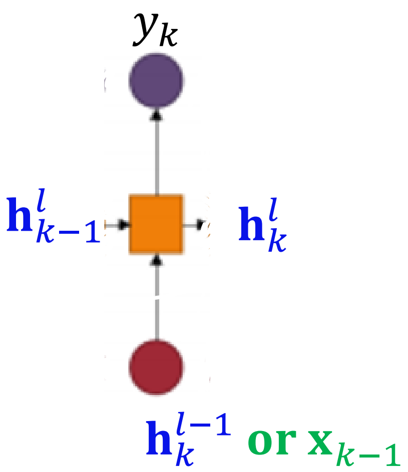

Batch: weight update 단위

# LSTM Models for Univariate Forecasting

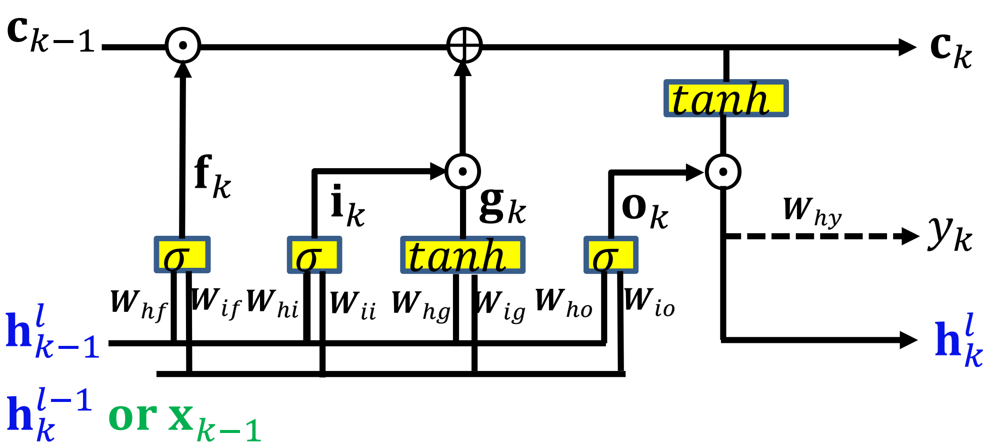
- multilayer RNN

$$\mathbf{h}_k^l = \sigma\left(\mathbf{W}_h^l \mathbf{h}_{k-1}^l + \mathbf{W}_u^l \mathbf{h}_k^{l-1} + b_h^l\right), \qquad \mathbf{h}_k^0 = \mathbf{x}_{k-1},$$

$$y_k = \mathbf{w}_y^T \mathbf{h}_k^L + b_y$$



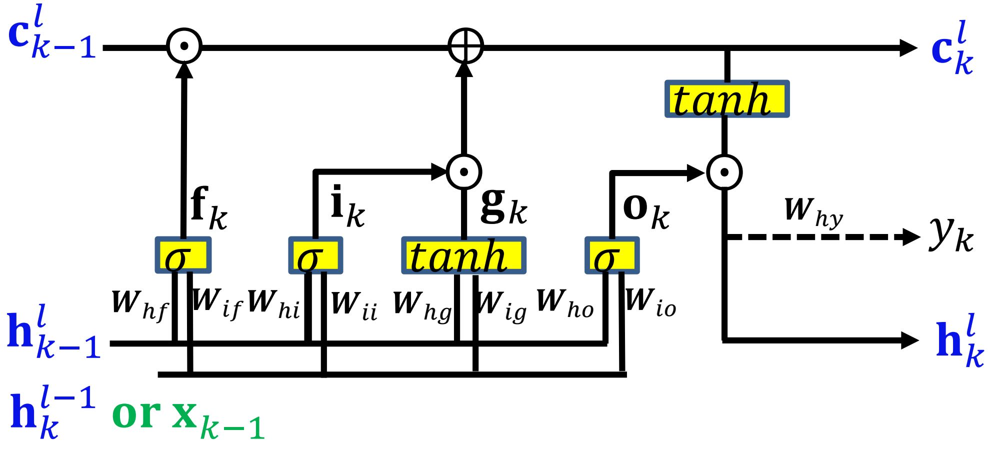

- multilayer LSTM



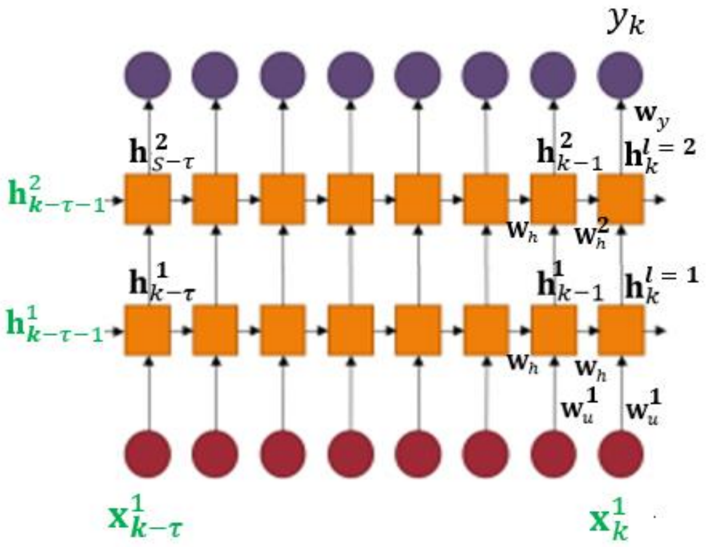
# LSTM (Pytoch)

- LSTM(Long Short-Term Memory) Structure
- Input gate: $\mathbf{i}_k = \sigma(\boldsymbol{W}_{ii}\mathbf{x}_{k-1} + \boldsymbol{W}_{hi}\mathbf{h}_{k-1} + b_i)$
- Forget gate: $\mathbf{f}_k = \sigma(\boldsymbol{W}_{if}\mathbf{x}_{k-1} + \boldsymbol{W}_{hf}\mathbf{h}_{k-1} + b_f)$
- Cell input: $\mathbf{g}_k = tanh(\boldsymbol{W}_{ig}\mathbf{x}_{k-1} + \boldsymbol{W}_{hg}\mathbf{h}_{k-1} + b_g)$
- Output gate: $\mathbf{o}_k = \sigma(\boldsymbol{W}_{io}\mathbf{x}_{k-1} + \boldsymbol{W}_{ho}\mathbf{h}_{k-1} + b_o)$
- Cell state: $\mathbf{c}_k = \mathbf{f}_k \odot \mathbf{c}_{k-1} + \mathbf{i}_k \odot \mathbf{g}_k,$
- Hidden state: $\mathbf{h}_k = \mathbf{o}_k \odot tanh(\mathbf{c}_k)$
- Output: $y_k = \boldsymbol{W}_{hy}\mathbf{h}_k + b_y$
- $\odot$ is the Hadamard product.
- multilayer LSTM: input of $l$-th layer; $\mathbf{h}_k^{l-1} \cdot \delta_k^{(l-1)}$; $\mathbf{h}_{k-1}^0 = \mathbf{x}_{k-1}$, $\mathbf{x}_{k-1} = [y_{k-1} \ \cdots \ y_{k-n}]^T$

  dropout $\delta_k^{(l-1)}$ is 0 with Bernoulli probability $p$.
- For details, see https://arxiv.org/pdf/1402.1128.pdf.



# LSTM modeling: Time series with sensor data




$$\mathbf{y}_k = f(y_{k-1}, \cdots, y_{k-n}, \qquad \mathbf{u}_{k}, \cdots, \mathbf{u}_{k-m})$$

- **LSTM structure**
  - $\mathbf{h}^l_k = \mathcal{L}^l(\mathbf{h}^l_{k-1}, \mathbf{v}^l_k), \quad \mathbf{v}^l_{k-1} = \mathbf{h}^{l-1}_{k-1}, \mathbf{v}^0_k = \mathbf{u}_k$

    $\mathbf{y}_k = \mathcal{M}_h(\mathbf{h}^L_k)$, linear activation

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $y_0$ | 0 | 17.323 | 2018-11-01 오후 10:02:14 | LC | $\mathbf{u}_0$ | I | 0.513768 | -2.85806 | 0.25331 | 1.066695 | -0.99039 | 0.109287 | 0.637969 | 0.107826 | 0.028429 | -0.09702 | -1.4788 | 0.021096 | -0.51944 |
| 2 | $y_1$ | 0 | 2.362 | 2018-11-01 오후 10:36:07 | GC | $\mathbf{u}_1$ | I | 0.519603 | -2.85734 | 0.25331 | 1.361685 | -0.95257 | 0.093751 | 0.522997 | 0.104698 | 0.028429 | -0.0906 | 0.827349 | 0.021096 | -0.53528 |
| 3 | $y_2$ | 0 | 6.299 | 2018-11-01 오후 11:10:02 | HA | $\mathbf{u}_2$ | I | 1.311815 | -2.85662 | 0.25331 | 1.038415 | -0.97356 | 0.056471 | -0.45419 | -0.01887 | 0.026592 | -0.08417 | 0.38363 | -0.11055 | -0.27943 |
| 4 | $y_3$ | 0 | 4.724 | 2018-11-01 오후 11:43:57 | DD | | I | 1.322554 | -2.85589 | 0.25331 | 1.265814 | -0.99595 | 0.009863 | 0.609262 | 0.106262 | 0.028429 | -0.09548 | 0.57707 | 0.021096 | -0.32551 |
| 5 | | 0 | 1.575 | 2018-11-01 오후 3:49:14 | HB | | I | 1.206569 | -2.866 | 0.25331 | -0.84464 | -0.95613 | 0.081335 | 0.522997 | 0.103132 | 0.028429 | -0.09214 | 0.075594 | 0.021096 | -0.76208 |
| 6 | | 0 | 8.661 | 2018-11-01 오후 4:23:10 | GD | | I | 1.265453 | -2.86528 | 0.25331 | 1.142898 | -0.94285 | 0.118615 | -1.01462 | -0.02357 | 0.025804 | -0.09342 | 0.409988 | -0.11055 | -0.48607 |
| 7 | | 0 | 5.512 | 2018-11-01 오후 4:57:06 | HB | | I | 1.23363 | -2.86456 | 0.25331 | 1.299379 | -0.96232 | 0.22995 | 0.635928 | -0.02031 | -0.36483 | -0.10187 | 0.47209 | -0.11042 | -0.44749 |
| 8 | | 0 | 4.724 | 2018-11-01 오후 5:31:01 | EA | | I | 1.278365 | -2.86384 | 0.25331 | 1.066059 | -0.95356 | 0.096461 | -0.74023 | -0.025 | -0.36483 | -0.10058 | 0.364933 | -0.11042 | -0.24681 |
| 9 | | 0 | 3.15 | 2018-11-01 오후 6:04:54 | JA | | I | 1.291916 | -2.86311 | 0.25331 | 1.008914 | -0.94024 | 0.127943 | 0.637969 | -0.02357 | 0.027379 | -0.08905 | 0.478517 | -0.11055 | -0.48728 |
| 10 | $y_k$ | 0 | 2.362 | 2018-11-01 오후 6:38:48 | HA | $\mathbf{u}_k$ | I | 0.717022 | -2.86239 | 0.25331 | 0.963446 | -0.97605 | 0.124823 | 0.522997 | 0.106262 | 0.028429 | -0.08623 | 0.549567 | 0.021096 | -0.31231 |
| 11 | | 0 | 2.362 | 2018-11-01 오후 7:12:42 | CA | | F | 0.515051 | -2.86167 | 0.25331 | 1.156932 | -0.96019 | 0.139918 | -0.02243 | -0.02031 | -0.36483 | -0.09519 | -1.53664 | -0.11042 | -0.44989 |
| 12 | | 0 | 7.874 | 2018-11-01 오후 7:46:37 | GD | | I | 0.527688 | -2.86095 | 0.25331 | 1.208045 | -0.95293 | 0.053351 | 0.609262 | 0.106262 | 0.028429 | -0.09214 | -1.53953 | 0.021096 | -0.31807 |
| 13 | | 0 | 3.15 | 2018-11-01 오후 8:20:30 | EB | | I | 0.517192 | -2.86023 | 0.25331 | 1.102341 | -1.01006 | 0.087543 | -2.02 | 0.101569 | 0.028429 | -0.11167 | 0.237176 | 0.021096 | -0.31663 |
| 14 | | 0 | 4.724 | 2018-11-01 오후 8:54:24 | EA | | I | 0.512105 | -2.8595 | 0.25331 | 1.162565 | -0.98576 | 0.093751 | 0.609262 | 0.101569 | 0.028429 | -0.08751 | 0.207839 | 0.021096 | -0.52712 |
| 15 | | 0 | 5.512 | 2018-11-01 오후 9:28:19 | HB | | I | 0.517841 | -2.85878 | 0.25331 | 1.012603 | -0.99158 | 0.084423 | -1.34518 | 0.103132 | 0.028429 | -0.08803 | 0.4093 | 0.021096 | -0.35263 |

# CNN modeling

$$\mathbf{y}_k = f(\mathbf{y}_{k-1}, \cdots, \mathbf{y}_{k-n}, \quad \mathbf{u}_{k,} \cdots, \mathbf{u}_{k-m})$$

$$\mathbf{X}_k = [\mathbf{y}_{k-1}, \cdots, \mathbf{y}_{k-n}]$$ matrix

$$\mathbf{U}_k = [\mathbf{u}_{k,} \cdots, \mathbf{u}_{k-m}]$$ matrix

- **CNN structure**
  - MLP($\mathcal{M}_h$): nonlinear function modelling
  - CNN($\mathcal{C}_x$, $\mathcal{C}_u$): MLP+feature embedding
  - $\mathbf{g}_k = \mathcal{C}_x(\mathbf{X}_k)$,
  - $\mathbf{f}_k = \mathcal{C}_u(\mathbf{U}_k)$,
  - $\mathbf{y}_k = \mathcal{M}_y(\mathbf{g}_k \| \mathbf{f}_k)$, linear activation in output layer
  - $\|$: concatenation

$\mathbf{X}_{k-1}$ $\mathbf{U}_k$ $\mathbf{y}_k$

| ID | Train_ | Target1 | Target2 | Target3 | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | B1 | B2 | B3 | B4 | B5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | -1.210937 | -1.520172 | -0.061984 | -1.245195 | -0.078650 | -0.608693 | -0.537828 | -0.835959 | -1.828848 | 0.179504 | -1.303546 | 1.066706 | 0.749929 | 0.608403 | 0.990293 | -0.335114 | 4.495308 | 0.261637 | 0.265954 | 0.265873 | 0.274630 | -1.00194 |
| 2 | 0 | 0.062207 | -0.372017 | -0.832656 | -1.252860 | -0.077278 | -0.608876 | -0.702306 | -0.485177 | -1.977951 | 0.352962 | -1.294930 | 0.952974 | 0.922161 | 0.537881 | 0.894918 | -0.336552 | 4.481047 | 0.262060 | 0.265954 | 0.266370 | 0.275049 | -0.99800 |
| 3 | 0 | -0.854031 | 0.229287 | -0.99625 | 0.420249 | -0.060809 | 2.485554 | -0.868590 | 0.784345 | -0.505027 | 0.080081 | -1.289737 | 1.564064 | 1.575548 | 0.575966 | 0.973334 | -0.333643 | -0.632738 | 0.787846 | 0.784667 | 0.787903 | 0.771161 | 0.813149 |
| 4 | 0 | 1.556712 | 0.414122 | 0.169924 | -1.254138 | -0.078650 | -0.408605 | -0.867770 | -0.499199 | -1.994576 | 0.381245 | -1.295840 | 0.832759 | 0.914001 | 0.625614 | 0.982019 | -0.333387 | 4.519075 | 0.262482 | 0.266364 | 0.266370 | 0.275553 | -0.99400 |
| 5 | 0 | 0.002451 | 0.076440 | -0.49899 | 0.422804 | -0.082767 | 1.393101 | -0.211825 | 0.795429 | -0.483017 | 0.100322 | -1.290693 | 1.554636 | 1.587094 | 0.661770 | 0.986650 | -0.331501 | -0.640343 | 0.788437 | 0.785180 | 0.788285 | 0.771161 | 0.818056 |
| 6 | 0 | 1.019554 | 0.310086 | 1.526019 | -1.255735 | -0.077278 | -0.408605 | -0.870889 | -0.506127 | -1.984008 | 0.300820 | -1.294880 | 0.832170 | 0.929908 | 0.638722 | 0.960192 | -0.333783 | 4.519075 | 0.262989 | 0.266793 | 0.266781 | 0.275553 | -0.98995 |
| 7 | 0 | -0.412162 | 0.630619 | -1.01855 | 0.425199 | -0.062182 | 1.769545 | -0.370887 | 0.780565 | -0.468179 | 0.158050 | -1.290211 | 1.564654 | 1.563889 | 0.584732 | 0.908118 | -0.334770 | -0.640818 | 0.788437 | 0.785753 | 0.788769 | 0.771531 | 0.822621 |
| 8 | 0 | -0.543489 | -0.926579 | 0.390245 | -1.238009 | -0.078650 | -0.593723 | -0.369409 | -0.548094 | -1.896980 | 0.430418 | -1.301671 | 0.773241 | 0.898616 | 0.575126 | 0.998282 | -0.333839 | 4.399763 | 0.262989 | 0.267312 | 0.267229 | 0.275984 | -0.98573 |
| 9 | 0 | 1.063058 | 1.925986 | 0.659833 | 0.420249 | -0.082767 | 0.488870 | -0.370887 | 0.793318 | -0.489523 | 0.117253 | -1.289433 | 1.564654 | 1.573348 | 0.586770 | 0.910240 | -0.336425 | -0.640818 | 0.788805 | 0.785753 | 0.789314 | 0.772029 | 0.827300 |
| 10 | 1 | | | | -1.237370 | -0.078650 | -0.061554 | -0.209527 | -0.369618 | -1.879355 | 0.253710 | -1.303273 | 0.831581 | 0.999929 | 0.616758 | 0.998582 | -0.333361 | 4.281401 | 0.263417 | 0.267312 | 0.267756 | 0.276421 | -0.98168 |
| 11 | 0 | 1.224227 | 0.809512 | 0.603603 | 0.425199 | -0.060809 | 0.129588 | -0.706902 | 0.782451 | -0.505027 | 0.141698 | -1.286403 | 1.554636 | 1.582351 | 0.621618 | 0.913355 | -0.333006 | -0.632738 | 0.789288 | 0.786115 | 0.789314 | 0.772545 | 0.831807 |
| 12 | 0 | -0.873124 | -0.789904 | -0.374621 | -1.233218 | -0.077278 | -0.242838 | -0.209363 | -0.449925 | -1.691492 | 0.091718 | -1.302302 | 0.833348 | 0.937341 | 0.642248 | 0.982019 | -0.333068 | 4.709214 | 0.263858 | 0.267740 | 0.267756 | 0.276943 | -0.97763 |
| 13 | 0 | 0.112203 | -0.169036 | 0.094579 | 0.425199 | -0.082767 | 0.660114 | -0.378602 | 0.789739 | -0.472810 | 0.057675 | -1.291320 | 1.564654 | 1.582230 | 0.632688 | 0.966664 | -0.332780 | -0.640818 | 0.789843 | 0.786622 | 0.789683 | 0.772545 | 0.836600 |
| 14 | 1 | | | | -1.227309 | -0.078650 | -0.609058 | -0.702142 | -0.475795 | -1.895601 | 0.381352 | -1.295297 | 0.772063 | 0.935745 | 0.630418 | 1.052108 | -0.331072 | 4.637436 | 0.264334 | 0.268157 | 0.268179 | 0.276943 | -0.97363 |
| 15 | 0 | -1.101305 | -0.398615 | 0.014284 | 0.422644 | -0.060809 | 0.486132 | -0.045213 | 0.782460 | -0.509031 | 0.115434 | -1.289795 | 1.564064 | 1.584299 | 0.634615 | 1.011977 | -0.332381 | -0.640343 | 0.789843 | 0.787172 | 0.790191 | 0.772934 | 0.841393 |

# CNN modeling

```python
self.time_conv = nn.Sequential(
            nn.Conv1d(input_size, 200, 4, stride=2),
            nn.Tanh(),
            nn.Conv1d(200, 150, 4),
            nn.Tanh(),
            nn.Conv1d(150, 100, 4),
            nn.Tanh(),
            nn.Conv1d(100, input_size, 3),
            nn.Tanh(),
)

self.x_conv = nn.Sequential(
            nn.Conv1d(sequence_length - 1, sequence_length - 1, 3),
            nn.Tanh(),
)
```

**CNN structure**

- MLP($\mathcal{M}_h$): nonlinear function modelling
- CNN($\mathcal{C}_y$, $\mathcal{C}_u$): MLP+feature embedding
- $\mathbf{g}_k = \mathcal{C}_x(\mathbf{X}_k)$,
- $\mathbf{f}_k = \mathcal{C}_u(\mathbf{U}_k)$,
- $\mathbf{y}_k = \mathcal{M}_y(\mathbf{g}_k \| \mathbf{f}_k)$, linear activation in output layer
- $\|$: concatenation

```python
def forward(self, u, x):
    b, ll, tt = u.shape
    u = u.reshape(b, tt, ll)
    output = self.time_conv(u)

    x = self.x_conv(x)
```

$88 \times 97 \times 20$

$88 \times 20 \times 3$

97x1
100x3 ;97
100x3
150x4 ;100
150x6
200x4 ;150
200x9
stride=2 97x4 ;200
97x20
Filter

Training data pair 88

$88 \times 20 \times 97$ : Tensor_u
$88 \times 20 \times 3$ : Tensor_x
$88 \times 1 \times 1$ : Tensor_y

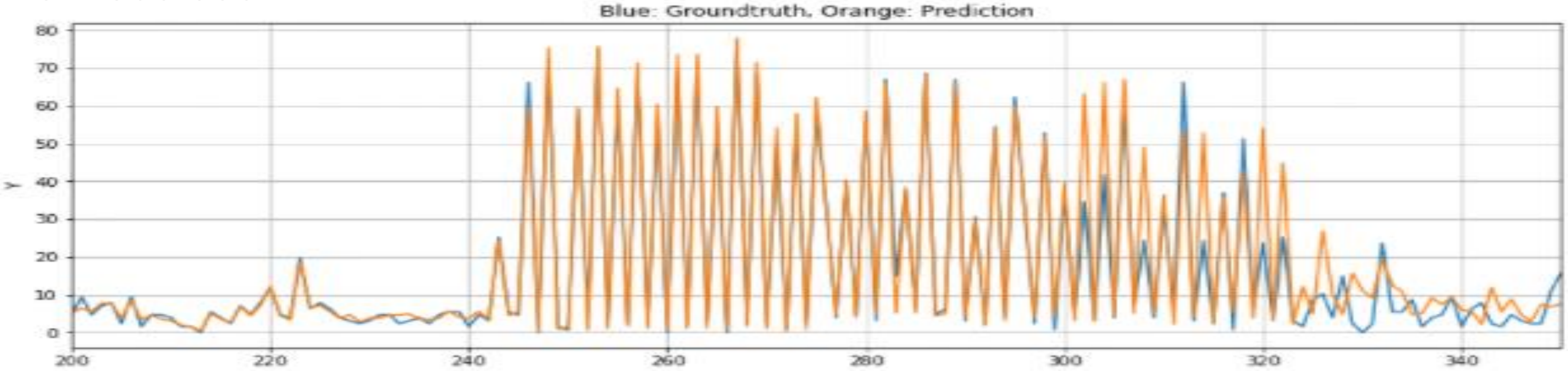
$\mathbf{X}_k$ $\mathbf{U}_k$ $\mathbf{y}_k$

| ID | Train_ | Target1 | Target2 | Target3 | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | B1 | B2 | B3 | B4 | B5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | -1.21093 | -1.52017 | -0.06198 | -1.245195 | -0.07865 | -0.60869 | -0.53782 | -0.83595 | -1.82884 | 0.179504 | -1.30354 | 1.066706 | 0.749929 | 0.608403 | 0.990293 | -0.33511 | 4.495308 | 0.261637 | 0.265954 | 0.265873 | 0.274630 | -1.00194 |
| 2 | 0 | 0.062207 | -0.37201 | -0.83265 | -1.25286 | -0.07727 | -0.60887 | -0.70230 | -0.48517 | -1.977951 | 0.352962 | -1.29493 | 0.952974 | 0.922161 | 0.537881 | 0.894918 | -0.33655 | 4.481047 | 0.262060 | 0.265954 | 0.266370 | 0.275049 | -0.99800 |
| 3 | 0 | -0.854031 | 0.229287 | -0.99625 | 0.420249 | -0.06080 | 2.485554 | -0.86859 | 0.784345 | -0.50502 | 0.080081 | -1.289737 | 1.564064 | 1.575548 | 0.575966 | 0.973334 | -0.33364 | -0.63273 | 0.787846 | 0.784667 | 0.787903 | 0.771161 | 0.813149 |
| 4 | 0 | 1.556712 | 0.414122 | 0.16992 | -1.25413 | -0.07865 | -0.40860 | -0.86777 | -0.49919 | -1.99457 | 0.381245 | -1.29584 | 0.832759 | 0.914001 | 0.625614 | 0.982019 | -0.333387 | 4.519075 | 0.262482 | 0.266364 | 0.266370 | 0.275553 | -0.99400 |
| 5 | 0 | 0.002451 | 0.076440 | -0.49899 | 0.422804 | -0.082767 | 1.393101 | -0.211825 | 0.795429 | -0.483017 | 0.100322 | -1.29069 | 1.554636 | 1.587094 | 0.661770 | 0.986650 | -0.331501 | -0.64034 | 0.788437 | 0.785180 | 0.788285 | 0.771161 | 0.818056 |
| 6 | 0 | 1.019554 | 0.310086 | 1.52601 | -1.255735 | -0.07727 | -0.40860 | -0.87088 | -0.506127 | -1.98400 | 0.300820 | -1.29488 | 0.832170 | 0.929908 | 0.638722 | 0.960192 | -0.33378 | 4.519075 | 0.262989 | 0.266793 | 0.266781 | 0.275553 | -0.98995 |
| 7 | 0 | -0.412162 | 0.030619 | -1.01835 | 0.425199 | -0.062182 | 1.769545 | -0.370887 | 0.780565 | -0.46817 | 0.158050 | -1.290211 | 1.564654 | 1.563889 | 0.584732 | 0.908118 | -0.33477 | -0.64081 | 0.788437 | 0.785753 | 0.788769 | 0.771531 | 0.822621 |
| 8 | 0 | -0.54348 | -0.92657 | 0.390245 | -1.23800 | -0.07865 | -0.59372 | -0.36940 | -0.548094 | -1.89698 | 0.430418 | -1.301671 | 0.773241 | 0.898616 | 0.575126 | 0.998282 | -0.33383 | 4.399763 | 0.262989 | 0.267312 | 0.267229 | 0.275984 | -0.98573 |
| 9 | 0 | 1.063058 | 1.925986 | 0.659833 | 0.420249 | -0.082767 | 0.488870 | -0.370887 | 0.793318 | -0.48952 | 0.117253 | -1.28943 | 1.564654 | 1.573348 | 0.586770 | 0.910240 | -0.336425 | -0.64081 | 0.788805 | 0.785753 | 0.789314 | 0.772029 | 0.82730 |
| 10 | 1 | | | | -1.23737 | -0.07865 | -0.06155 | -0.209527 | -0.36961 | -1.879355 | 0.253710 | -1.303273 | 0.831581 | 0.999929 | 0.616758 | 0.998582 | -0.333361 | 4.281401 | 0.263417 | 0.267312 | 0.267756 | 0.276421 | -0.98168 |
| 11 | 0 | 1.224227 | 0.809512 | 0.603603 | 0.425199 | -0.06080 | 0.129588 | -0.70690 | 0.782451 | -0.505027 | 0.141698 | -1.28640 | 1.554636 | 1.582351 | 0.621618 | 0.913355 | -0.33300 | -0.63273 | 0.789288 | 0.786115 | 0.789314 | 0.772545 | 0.831807 |
| 12 | 0 | -0.87312 | -0.78990 | -0.374621 | -1.23321 | -0.07727 | -0.24283 | -0.20936 | -0.44992 | -1.691492 | 0.091718 | -1.302302 | 0.833348 | 0.937341 | 0.642248 | 0.982019 | -0.33306 | 4.709214 | 0.263858 | 0.267740 | 0.267756 | 0.276943 | -0.97763 |
| 13 | 0 | 0.112203 | -0.16903 | 0.094579 | 0.425199 | -0.082767 | 0.660114 | -0.378602 | 0.789739 | -0.47281 | 0.057675 | -1.29132 | 1.564654 | 1.582230 | 0.632688 | 0.966664 | -0.33278 | -0.64081 | 0.789843 | 0.786622 | 0.789683 | 0.772545 | 0.83660 |
| 14 | 1 | | | | -1.22730 | -0.07865 | -0.60905 | -0.70214 | -0.47579 | -1.895601 | 0.381352 | -1.295297 | 0.772063 | 0.935745 | 0.630418 | 1.052108 | -0.33107 | 4.637436 | 0.264334 | 0.268157 | 0.268179 | 0.276943 | -0.97363 |
| 15 | 0 | -1.10130 | -0.39861 | 0.014284 | 0.422644 | -0.06080 | 0.486132 | -0.04521 | 0.782460 | -0.509031 | 0.115434 | -1.289795 | 1.564064 | 1.584299 | 0.634615 | 1.011977 | -0.332381 | -0.64034 | 0.789843 | 0.787172 | 0.790191 | 0.772934 | 0.84139 |

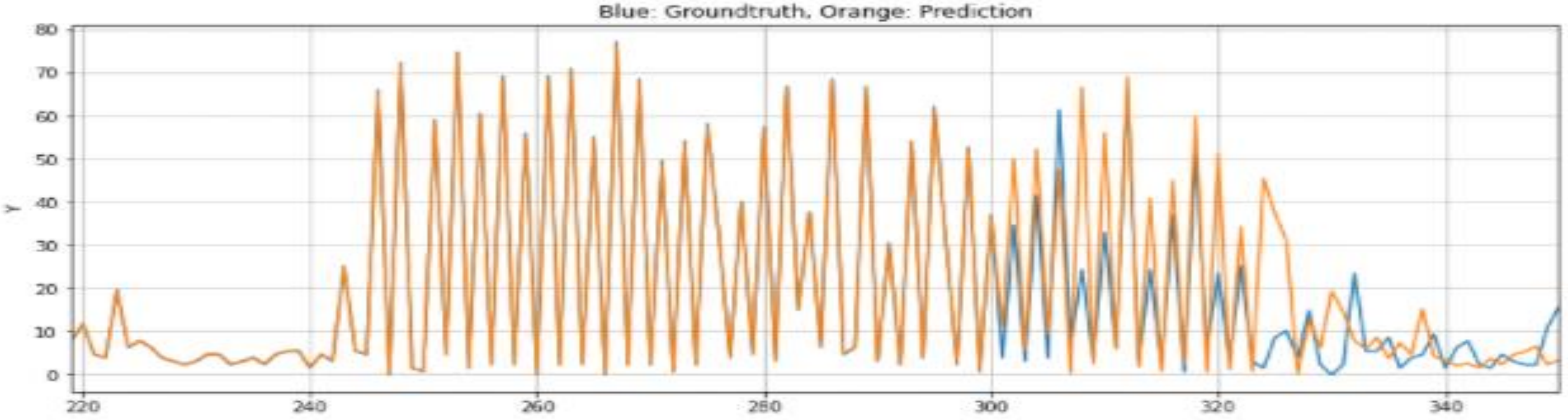
# Data1: LSTM Training vs. CNN

- Training: 0:300
- Validation: 300:350

LSTM




CNN



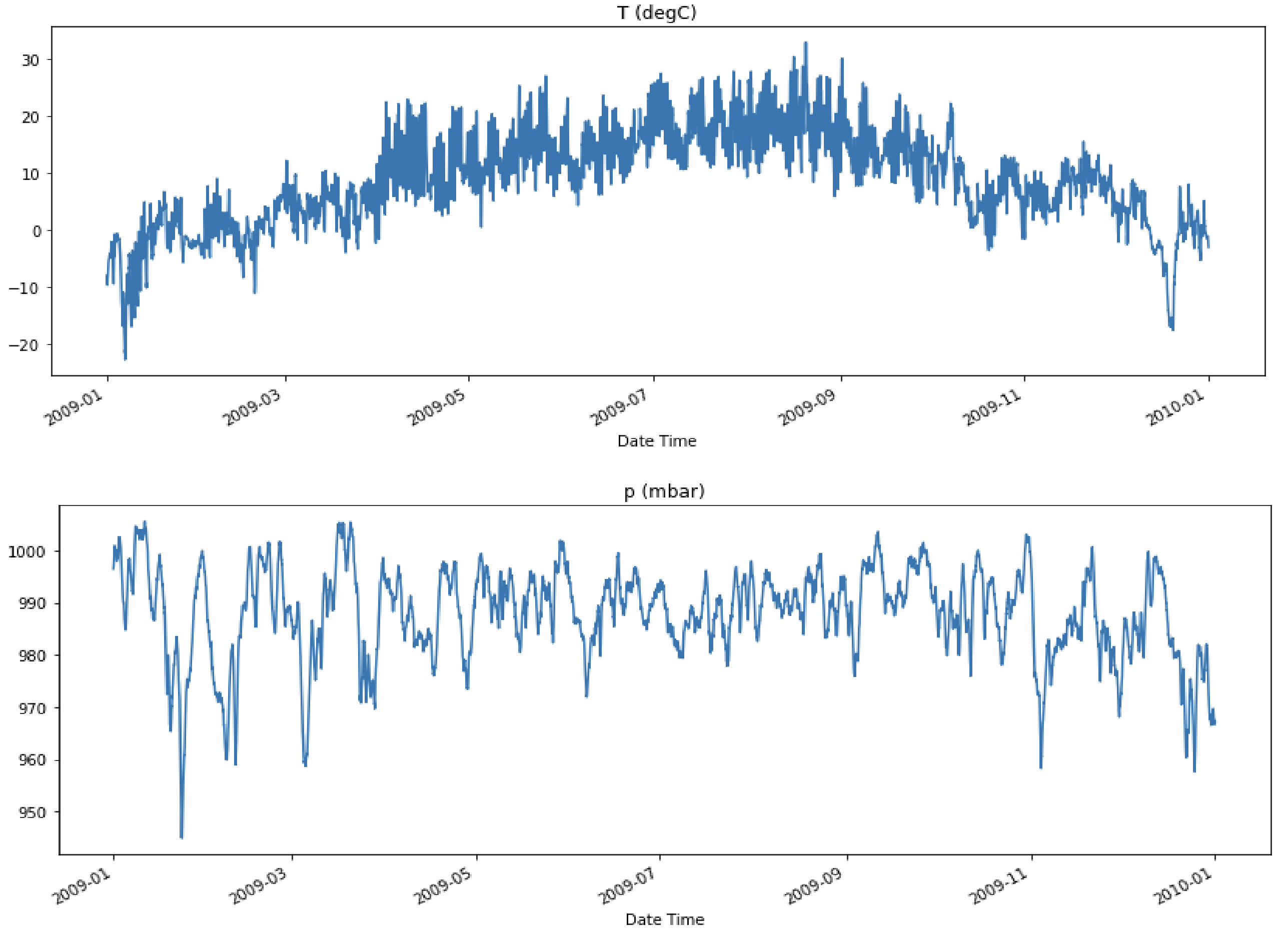
# Climate Series dataset

- p (mbar) 기압 과 T (degC) 를 예측하는 LSTM 구현
- 학습에 사용되는 입력은 6. rh (%) 부터 15. wd (deg) 까지 9개의 입력변수를 preprocess 함. 아래 링크 참고.
- https://www.tensorflow.org/tutorials/structured_data/time_series?hl=ko#특성_엔지니어링

| Index | Features | Format | Description |
|---|---|---|---|
| 1 | Date Time | 01.01.2009 00:10:00 | Date-time reference |
| 2 | **p (mbar)** | 996.52 | The pascal SI derived unit of pressure used to quantify internal pressure. Meteorological reports typically state atmospheric pressure in millibars. |
| 3 | **T (degC)** | -8.02 | Temperature in Celsius |
| 4 | ~~Tpot (K)~~ | 265.4 | Temperature in Kelvin |
| 5 | ~~Tdew (degC)~~ | -8.9 | Temperature in Celsius relative to humidity. Dew Point is a measure of the absolute amount of water in the air, the DP is the temperature at which the air cannot hold all the moisture in it and water condenses. |
| 6 | rh (%) | 93.3 | Relative Humidity is a measure of how saturated the air is with water vapor, the %RH determines the amount of water contained within collection objects. |
| 7 | VPmax (mbar) | 3.33 | Saturation vapor pressure |
| 8 | VPact (mbar) | 3.11 | Vapor pressure |
| 9 | VPdef (mbar) | 0.22 | Vapor pressure deficit |
| 10 | sh (g/kg) | 1.94 | Specific humidity |
| 11 | H2OC (mmol/mol) | 3.12 | Water vapor concentration |
| 12 | rho (g/m ** 3) | 1307.75 | Airtight |
| 13 | wv (m/s) | 1.03 | Wind speed |
| 14 | max. wv (m/s) | 1.75 | Maximum wind speed |
| 15 | wd (deg) | 152.3 | Wind direction in degrees |

# 다변량 예측 : 기압 p (mbar)와 기온 T (degC)

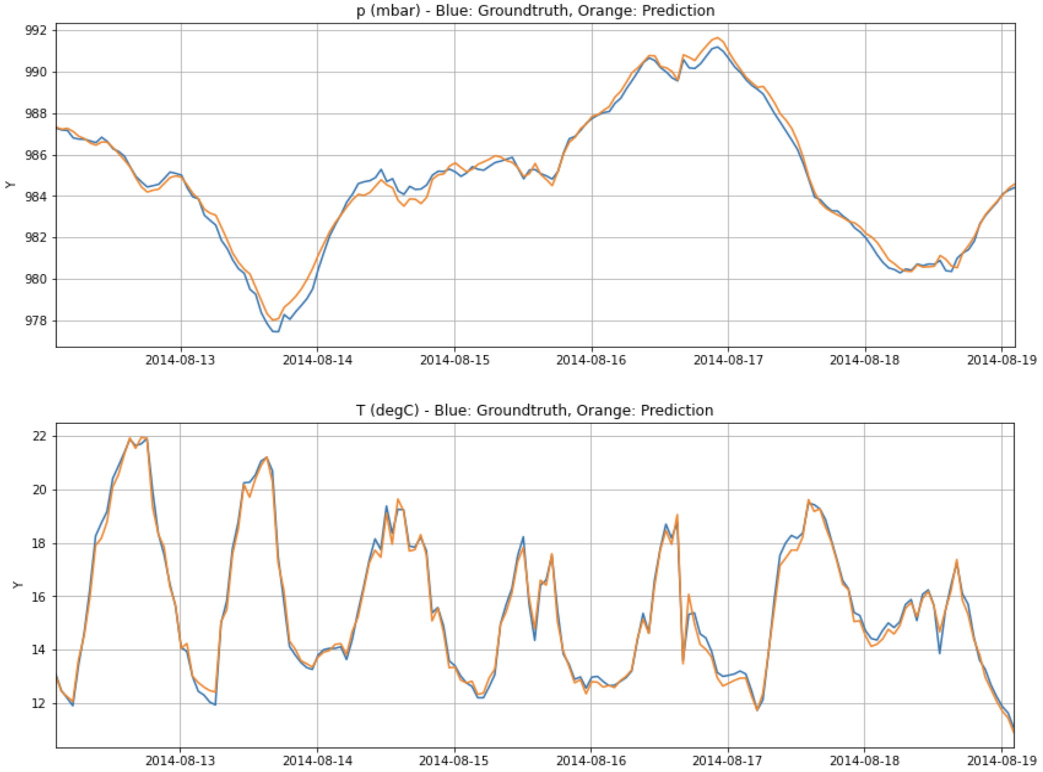
Training set : 2009년 동안의 온도와 기압.



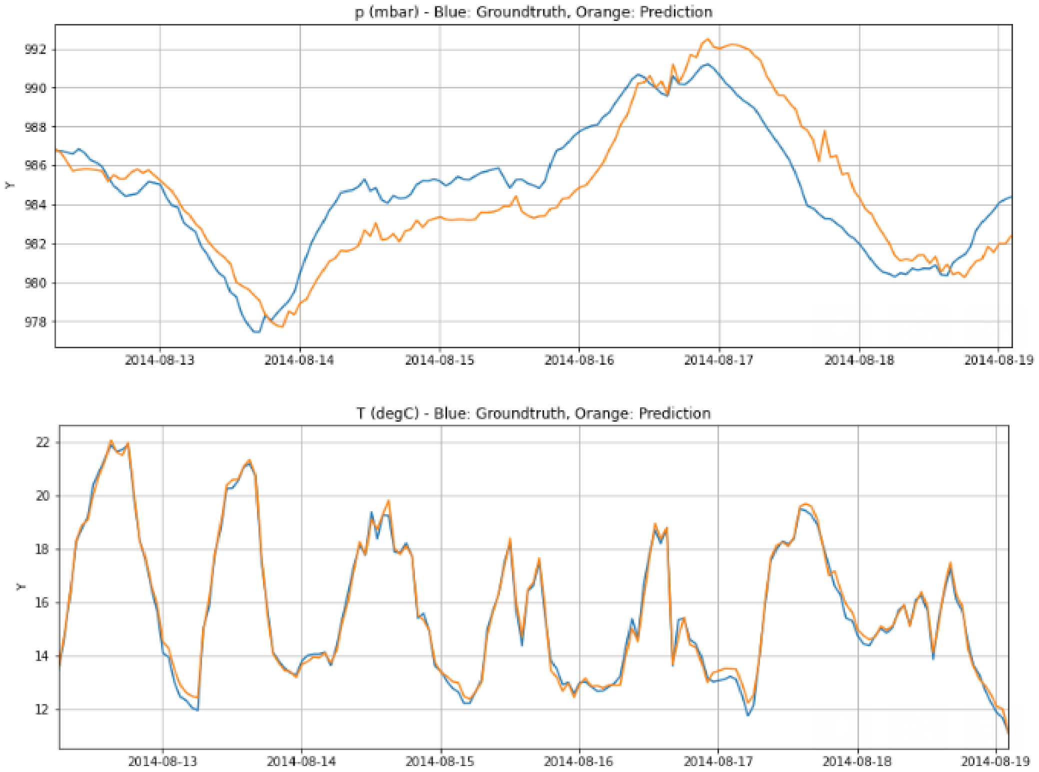
# Test set: 2014년 8월 13~19일

previous week (8월 5~12일) 구간 예측 후
hidden state, cell state로부터 (13~19일)예측
LSTM 이 CNN 보다 압력 예측이 좀 더 우수함을 보임. CNN도 튜닝을 더하면 좋을 수 있음.

LSTM



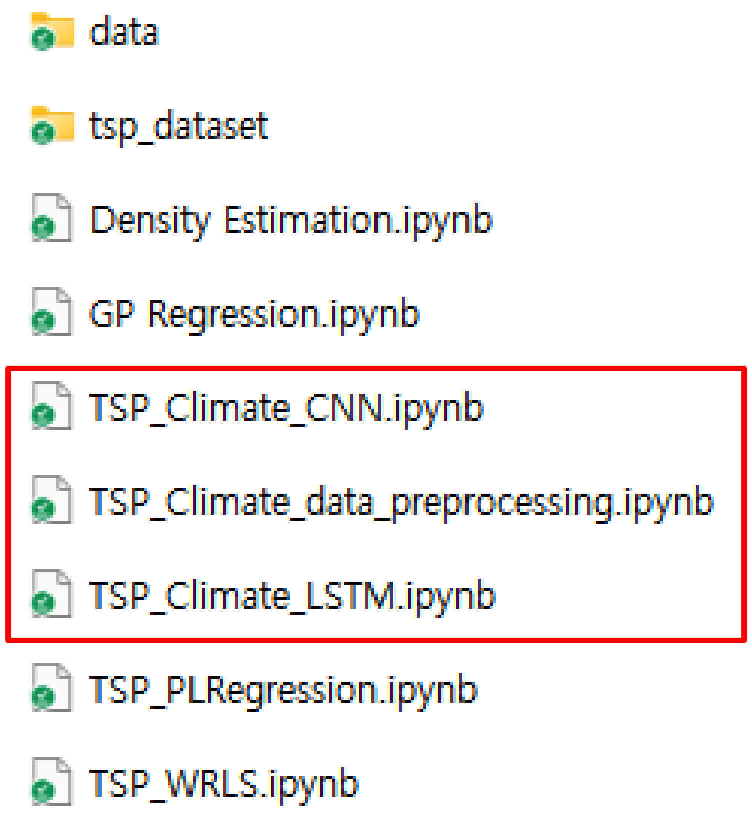

CNN



# Piecewise Linear Regression Model

- **LSTM structure**
  - $\mathbf{h}_k^l = \mathcal{L}^l\left(\mathbf{h}_{k-1}^l, \mathbf{v}_k^l\right), \quad \mathbf{v}_{k-1}^l = \mathbf{h}_{k-1}^{l-1}, \ \mathbf{v}_k^0 = \mathbf{u}_k$
    $\mathbf{y}_k = \mathcal{M}_h(\mathbf{h}_k^L)$, linear activation

- **CNN structure**
  - MLP($\mathcal{M}_h$): nonlinear function modelling
  - CNN($\mathcal{C}_x$, $\mathcal{C}_u$): MLP+feature embedding
  - $\mathbf{g}_k = \mathcal{C}_x(\mathbf{X}_k)$,
  - $\mathbf{f}_k = \mathcal{C}_u(\mathbf{U}_k)$,
  - $\mathbf{y}_k = \mathcal{M}_y(\mathbf{g}_k \| \ \mathbf{f}_k)$, linear activation in output layer
  - $\|$: concatenation



# Further Evaluation Metrics

Metrics

- Precision
- Recall
- Accuracy
- F1 score
- ROC(Receiver Operating Characteristic) curve
- AUC(Area Under Curve)

# Evaluation

Measures (Classification or Hypothesis Test)

| | | Actual Labels | |
|---|---|---|---|
| | | Positive(1) | Negative(0) |
| Prediction Results | Positive(1) | **True Positive(TP)** | **False Positive(FP)** |
| | Negative(0) | **False Negative(FN)** | True Negative(TN) |

Precision $= {}^{TP}/_{TP+FP}$ : Positive 로 예측 한 것 중에 제대로 맞춘 비율

Recall $= {}^{TP}/_{TP+FN}$ : 실제 Positive 중에서 예측을 맞춘 비율

Recall =Sensitivity,  Specificity $= {}^{TN}/_{TN+FP}$

# Evaluation

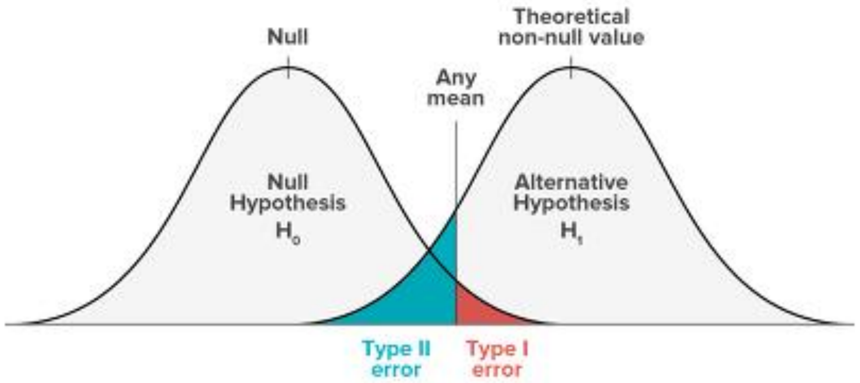
Precision-Recall Trade-off (ex, Hypothesis Test, 가설 검정)

| | | $H_0$ | |
|---|---|---|---|
| | | True | False |
| Test Results | Accept | **True Positive(TP)** | **Type 2 error(FP)** |
| | Reject | **Type 1 error(FN)** | True Negative(TN) |

Precision $= {}^{TP}/_{TP+FP}$ , Recall $= {}^{TP}/_{TP+FN}$

Type 1 error =P(reject $\boldsymbol{H_0}$| $\boldsymbol{H_0}$ is true)

Type 2 error =P(accept $\boldsymbol{H_0}$| $\boldsymbol{H_0}$ is not true)



# Evaluation

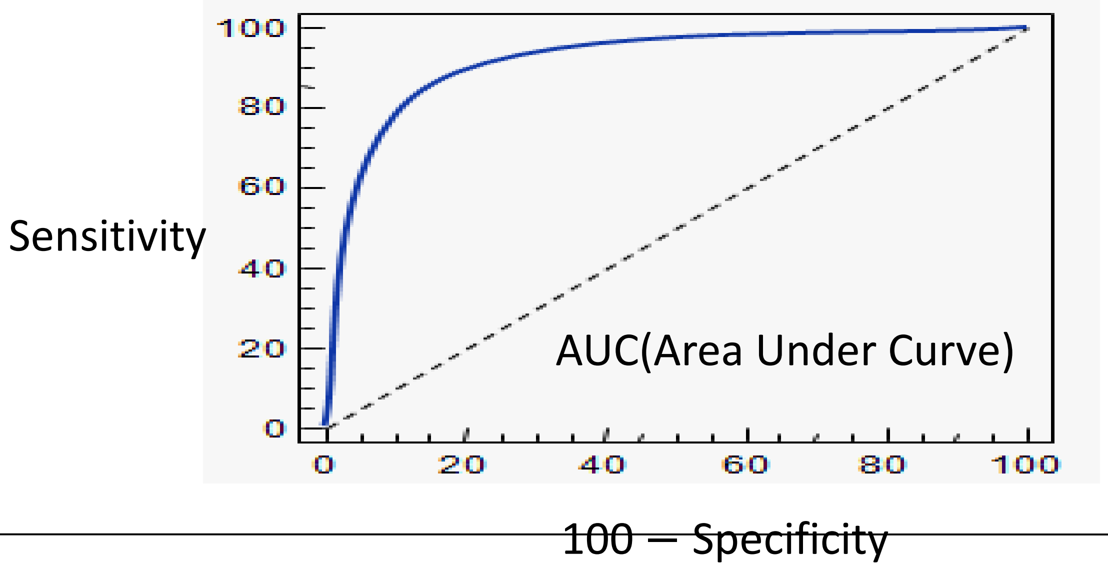
## ROC(Receiver Operating Characteristic) curve

| | | Actual Labels | |
|---|---|---|---|
| | | Positive(1) | Negative(0) |
| Prediction Results | Positive(1) | **True Positive(TP)** | **False Positive(FP)** |
| | Negative(0) | **False Negative(FN)** | True Negative(TN) |

Sensitivity= $TP/_{TP+FN}$

Specificity = $TN/_{TN+FP}$

AUC(Area Under Curve)



# Evaluation

## Accuracy

| | | Actual Labels | |
|---|---|---|---|
| | | Positive(1) | Negative(0) |
| Prediction Results | Positive(1) | **True Positive(TP)** | **False Positive(FP)** |
| | Negative(0) | **False Negative(FN)** | True Negative(TN) |

Specificity $= {TN}/{TN+FP}$

Sensitivity(Recall) $= {TP}/{TP+FN}$

Accuracy $= {TP+TN}/{TP+FN+TN+FP}$

# Evaluation

## F1 score

| | | Actual Labels | |
|---|---|---|---|
| | | Positive(1) | Negative(0) |
| Prediction Results | Positive(1) | **True Positive(TP)** | **False Positive(FP)** |
| | Negative(0) | **False Negative(FN)** | True Negative(TN) |

Precision $= {}^{TP}/_{TP+FP}$

Recall $= {}^{TP}/_{TP+FN}$

F1 score $= 2 \times \dfrac{Precision \times Recall}{Precision + Recall}$ (Precision과 Recall의 조화평균)

# GAUSSIAN PROCESS REGRESSION

**JIN YOUNG CHOI**

**ECE, SEOUL NATIONAL UNIVERSITY**

https://arxiv.org/pdf/2009.10862.pdf
https://github.com/jwangjie/Gaussian-Processes-Regression-Tutorial
http://mlg.eng.cam.ac.uk/tutorials/06/es.pdf
https://www.sciencedirect.com/science/article/abs/pii/S0022249617302158
http://www.gaussianprocess.org/gpml/chapters/RW.pdf

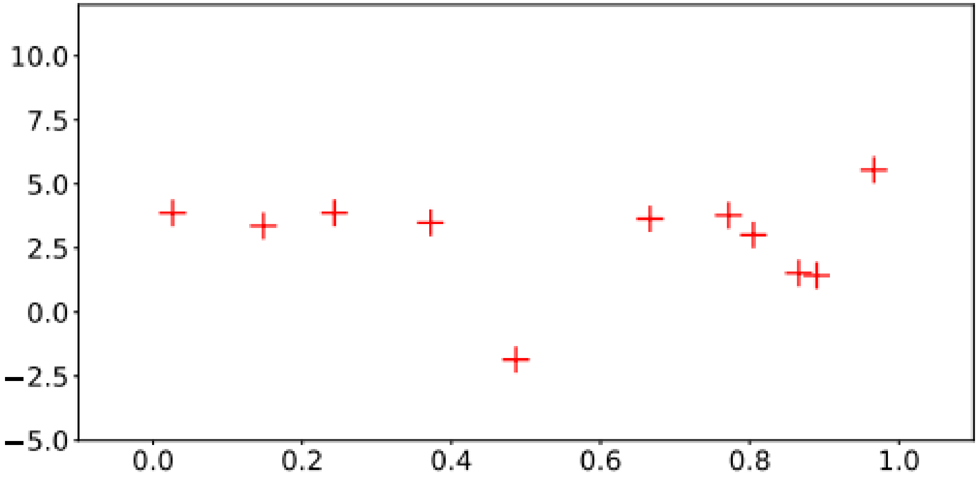
# Gaussian Process Regression

- General regression model (single variable)

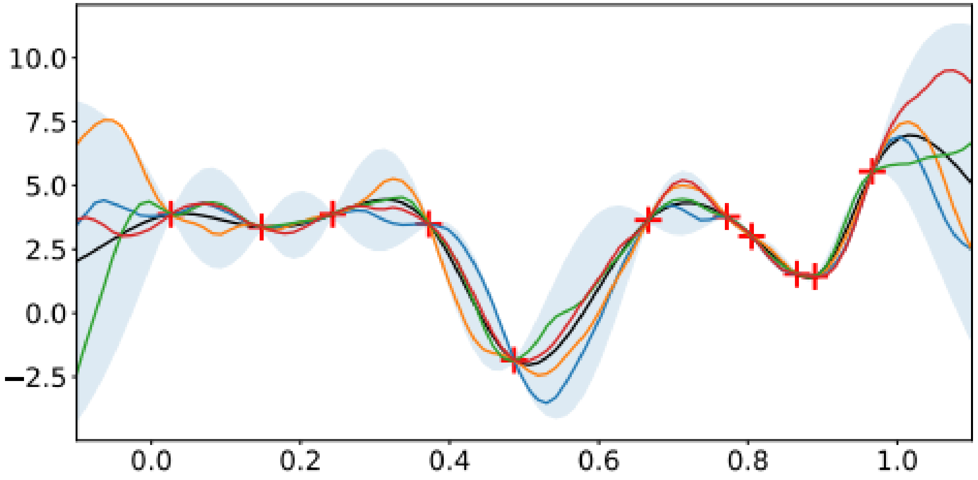
$$y = f(x) + \epsilon,$$

where $\epsilon \sim N(0, \sigma^2)$ and so $x, y$ are Gaussian random variables.

- Goal : to estimate $f(x)$ with uncertainty from observation data $D = \{(x_i, y_i) | i = 1, \cdots, n\}$
- $x_i, y_i$ are treated as Gaussian random variables.




**(a)** Data point observations




**(b)** Five possible regression functions by GPR

# Gaussian Process Regression

- General regression model (single variable)

$$y = f(x) + \epsilon,$$

where $\epsilon \sim N(0, \sigma^2)$ and so $x, y$ are Gaussian random variables.

- Define

$$\mathbf{x}^{\mathrm{T}} = [x_1 \quad \cdots \quad x_n], \qquad \mathbf{y}^{\mathrm{T}} = [y_1 \quad \cdots \quad y_n], \qquad \mathbf{f} := \mathbf{f}(\mathbf{x}) = [f(x_1) \quad \cdots \quad f(x_n)].$$

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} - \boldsymbol{\mu}\right)^T \Sigma^{-1}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} - \boldsymbol{\mu}\right)\right] := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

- Conditional probability (recall)

$$f(\mathbf{y}|\mathbf{x}) = \frac{1}{(2\pi)^{\frac{k}{2}}\sqrt{\det\Sigma_{\mathbf{y}|\mathbf{x}}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_{\mathbf{y}|\mathbf{x}})^t \Sigma_{\mathbf{y}|\mathbf{x}}^{-1}(\mathbf{y} - \mu_{\mathbf{y}|\mathbf{x}})\right),$$

where

$\mu_{\mathbf{y}|\mathbf{x}} = A(\mathbf{x} - \mu_{\mathbf{x}}) + \mu_{\mathbf{y}}$ and

$\Sigma_{\mathbf{y}|\mathbf{x}} = \Sigma_{\mathbf{y}} - A\Sigma_{\mathbf{xy}}$, where $A\Sigma_{\mathbf{x}} = \Sigma_{\mathbf{yx}}$.

# Gaussian Process Regression

- General regression model (single variable)

$$y = f(x) + \epsilon,$$

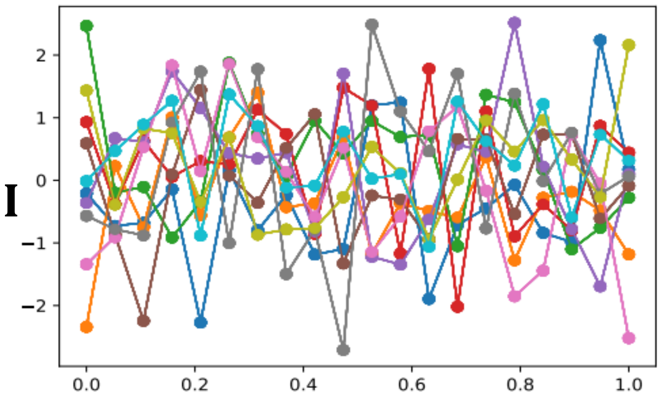
where $\epsilon \sim N(0, \sigma^2)$ and so $x, y$ are Gaussian random variables.

- Define

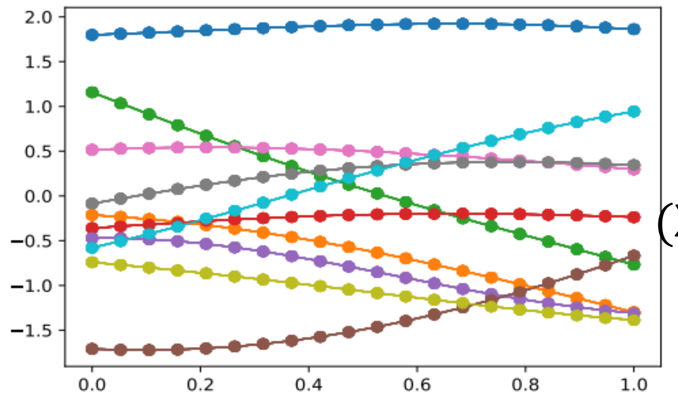
$$\mathbf{x} = [x_1 \quad \cdots \quad x_n], \qquad \mathbf{y} = [y_1 \quad \cdots \quad y_n], \qquad \mathbf{f} := \mathbf{f}(\mathbf{x}) = [f(x_1) \quad \cdots \quad f(x_n)].$$

$$f(\mathbf{y}|\mathbf{x}) = \frac{1}{(2\pi)^{\frac{k}{2}}\sqrt{\det\Sigma_{\mathbf{y}|\mathbf{x}}}} \exp\left(-\frac{1}{2}(\mathbf{y} - \mu_{\mathbf{y}|\mathbf{x}})^t \Sigma_{\mathbf{y}|\mathbf{x}}^{-1}(\mathbf{y} - \mu_{\mathbf{y}|\mathbf{x}})\right), := \mathcal{N}(\mu_{\mathbf{y}|\mathbf{x}}, \Sigma_{\mathbf{y}|\mathbf{x}})$$

$$\Sigma_{\mathbf{y}|\mathbf{x}} = \sigma^2 \mathbf{I}$$




Independent Gaussian




Jointly Gaussian

$$x = f^{-1}(y)$$

$$\Sigma_{\mathbf{y}|\mathbf{x}} \neq \sigma^2 \mathbf{I}$$

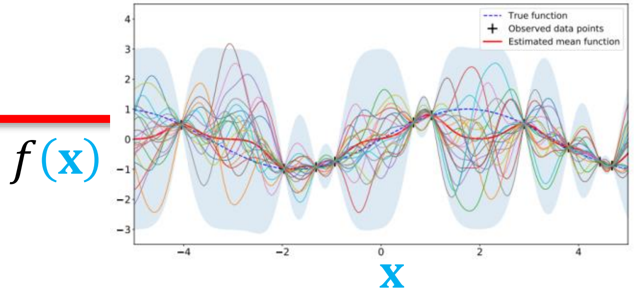
$$(\Sigma_{\mathbf{y}|\mathbf{x}})_{ij} = cov(x_i, x_j)$$

$$:= exp\left(-\frac{(x_i - x_j)^2}{2}\right)$$

a RBF kernel

x의 variance로 y의 variance 추정

# Gaussian Process Regression



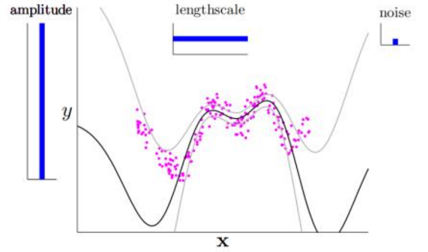

- Gaussian Processes ($\mathcal{GP}$) for multivariate regression

$$y = f(\mathbf{x}) + \epsilon.$$

- define $\mu_f(\mathbf{x}) := \mathbb{E}(f(\mathbf{x}))$, then we assume $f(\mathbf{x})$ is distributed as a Gaussian process

$$f(\mathbf{x}) \sim \mathcal{GP}\left(\mu_f(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$$

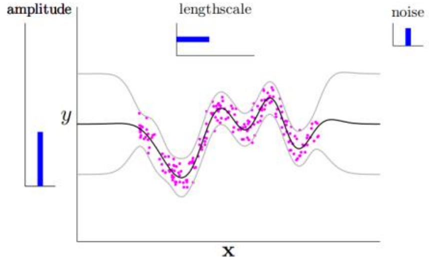
where $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\left[\left(f(\mathbf{x}) - \mu_f(\mathbf{x})\right)\left(f(\mathbf{x}') - \mu_f(\mathbf{x}')\right)\right]$ called the kernel of $\mathcal{GP}$.




- The kernel is based on assumptions such as smoothness, that is, similar $\mathbf{x}, \mathbf{x}'$ yields similar $f(\mathbf{x})$ and $f(\mathbf{x}')$. Thus a popular kernel is

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \mathbf{exp}\left(-\frac{1}{2\lambda}(\mathbf{x} - \mathbf{x}')^T (\mathbf{x} - \mathbf{x}')\right),$$

where hyperparameters $\lambda$ and $\sigma_f^2$ represents the length-scale and signal ($f$) variance to control relation between $\mathbf{x}$ and $f(\mathbf{x})$.



# Gaussian Process Regression

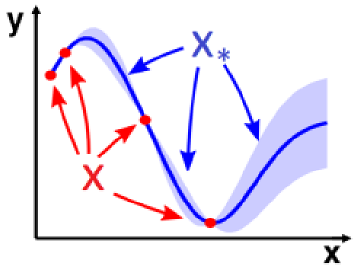
$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2\lambda}(\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}')\right)$$

## Modeling of prior sampling function of $\mathcal{GP}$

- Denote $\mathbf{X} = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]$, $\mathbf{y}^T = [y_1 \quad \cdots \quad y_n]$, $\mathbf{f}^T := [f(\mathbf{x}_1) \quad \cdots \quad f(\mathbf{x}_n)]$.

  Let $\mathbf{X}_* = [\mathbf{x}_1^* \quad \cdots \quad \mathbf{x}_n^*]$ be a matrix containing new input points. Then define the kernel matrix as

$$\mathbf{K}(\mathbf{X}_*,\mathbf{X}_*) = \begin{bmatrix} k(\mathbf{x}_1^*,\mathbf{x}_1^*) & k(\mathbf{x}_1^*,\mathbf{x}_2^*) & \cdots & k(\mathbf{x}_1^*,\mathbf{x}_n^*) \\ k(\mathbf{x}_2^*,\mathbf{x}_1^*) & k(\mathbf{x}_2^*,\mathbf{x}_2^*) & \cdots & k(\mathbf{x}_2^*,\mathbf{x}_n^*) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_n^*,\mathbf{x}_1^*) & k(\mathbf{x}_n^*,\mathbf{x}_2^*) & \cdots & k(\mathbf{x}_n^*,\mathbf{x}_n^*) \end{bmatrix}$$




- Choosing the prior mean function $\mu_f(\mathbf{x}) = 0$, we can sample values of $f$ at inputs $\mathbf{X}_*$ from $\mathcal{GP}$ as

$$\mathbf{f}_* \sim \mathcal{N}\left(0, \mathbf{K}(\mathbf{X}_*,\mathbf{X}_*)\right)$$

  which is the prior distribution model without observation data $D = \{(\mathbf{x}_i, y_i) | i = 1, \cdots, n\}$.

# Gaussian Process Regression

## Posterior predictions from a $\mathcal{GP}$

- Observations are $D = \{(\mathbf{x}_i, y_i)|i = 1, \cdots, n\} = \{\mathbf{X}, \mathbf{y}\}$ , $\mathbf{X} = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_n]$, $\mathbf{y}^T = [y_1 \quad \cdots \quad y_n]$.
- The predictions for new inputs $\mathbf{X}_* = [\mathbf{x}_1^* \quad \cdots \quad \mathbf{x}_n^*]$ by drawing $\mathbf{f}_*$ from the posterior distribution $p(f\ |D)$.

  A joint Gaussian distribution of $\mathbf{y}$ and $\mathbf{f}_*$ Let $\mathbf{X}_*$ follows

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} = \begin{bmatrix} \mathbf{f}(X) + \epsilon \\ \mathbf{f}(\mathbf{X}_*) \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I} & \mathbf{K}(\mathbf{X}, \mathbf{X}_*) \\ \mathbf{K}(\mathbf{X}_*, \mathbf{X}) & \mathbf{K}(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix}\right),$$

  where $\sigma_\epsilon^2$ is the assumed noise level of the observations.
- The conditional distribution $p(\mathbf{f}_*|\mathbf{X}, \mathbf{y}, \mathbf{X}_*)$ can be derived to a multivariate normal distribution with mean

$$\mu_{\mathbf{f}_*}(\mathbf{X}_*) = \mathbf{K}(\mathbf{X}_*, \mathbf{X})[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I}]^{-1}\mathbf{y}$$

  and variance

$$cov_{\mathbf{f}_*}(\mathbf{X}_*) = \mathbf{K}(\mathbf{X}_*, \mathbf{X}_*) - \mathbf{K}(\mathbf{X}_*, \mathbf{X})[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I}]^{-1}\mathbf{K}(\mathbf{X}, \mathbf{X}_*)$$

# Gaussian Process Regression

## Posterior predictions from a $\mathcal{GP}$

- The mean function of the $\mathcal{GP}$ can be given as

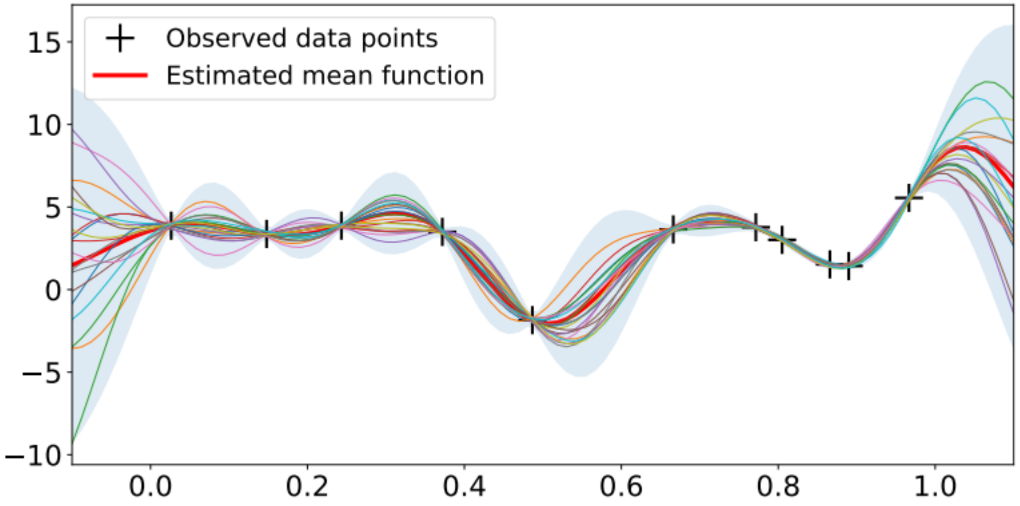
$$\mu_f(\mathbf{x}) = \mathbf{K}(\mathbf{x},\mathbf{X})[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2\mathbf{I}]^{-1}\mathbf{y}$$

and covariance function as

$$cov_f(\mathbf{x},\mathbf{x}') = k(\mathbf{x},\mathbf{x}') - \mathbf{K}(\mathbf{x},\mathbf{X})[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2\mathbf{I}]^{-1}\mathbf{K}(\mathbf{X},\mathbf{x}')$$

$$\mathbf{K}(\mathbf{x},\mathbf{X}) = [k(\mathbf{x},\mathbf{x}_1) \quad k(\mathbf{x},\mathbf{x}_2) \quad \cdots \quad k(\mathbf{x},\mathbf{x}_n)]$$

$$\mathbf{K}(\mathbf{X},\mathbf{x}') = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}') \\ k(\mathbf{x}_2,\mathbf{x}') \\ \vdots \\ k(\mathbf{x}_n,\mathbf{x}') \end{bmatrix}$$

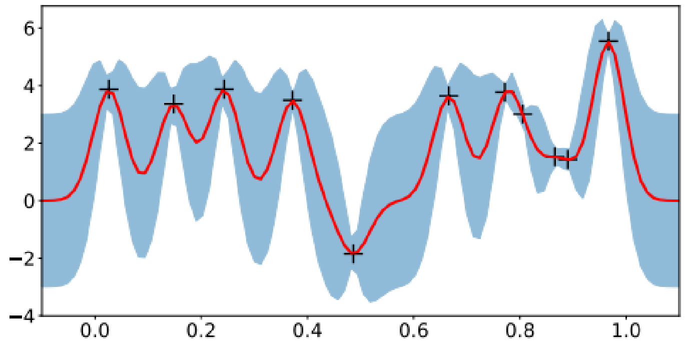


# Gaussian Process Regression

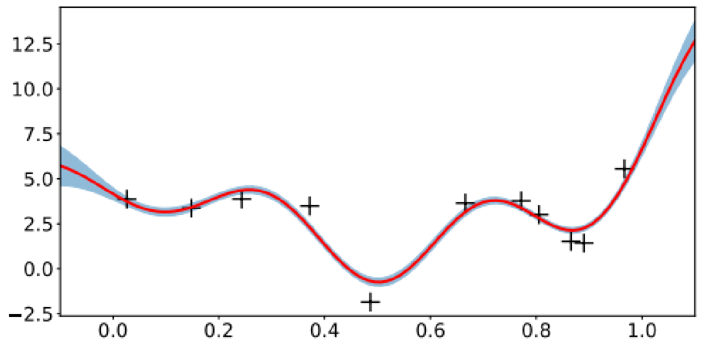
- The effect of the hyperparameters $\lambda$ and $\sigma_f^2$ of the kernel

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \mathbf{exp}\left(-\frac{1}{2\lambda}(\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}')\right) \approx \mathbb{E}\left[\left(f(\mathbf{x})-\mu_f(\mathbf{x})\right)\left(f(\mathbf{x}')-\mu_f(\mathbf{x}')\right)\right],$$

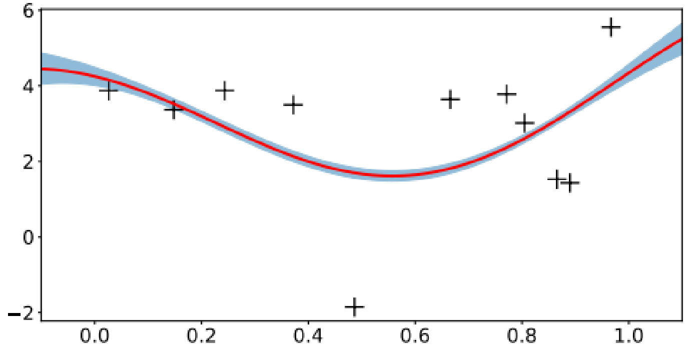
$\lambda$ : length-scale, $\sigma_f^2$ : signal ($f$) variance to control relation between $\mathbf{x}$ and $f(\mathbf{x})$.




Small $\lambda$




Medium $\lambda$




Large $\lambda$

# Gaussian Process Regression

$$p(\mathbf{y}|\mathbf{X}) = \frac{1}{(2\pi)^{d/2}\left|\Sigma_{\mathbf{y}|X}\right|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{y}-\mu_{\mathbf{y}|X})^T {\Sigma_{\mathbf{y}|\mathbf{X}}}^{-1}(\mathbf{y}-\mu_{\mathbf{y}|X})\right]$$

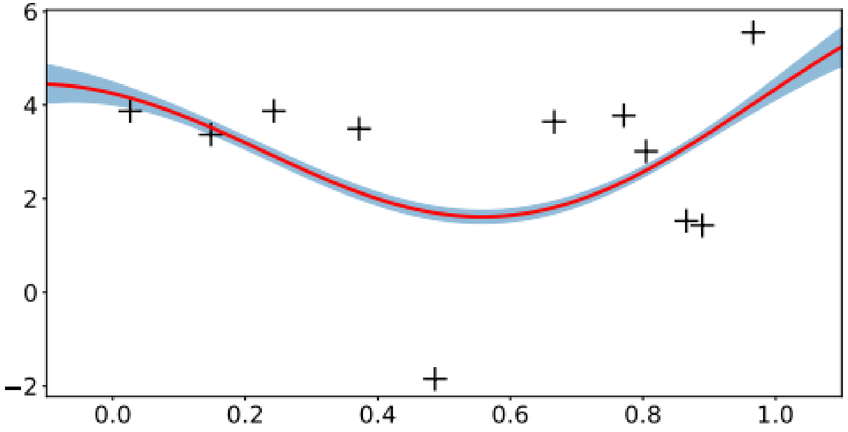
$$\boxed{\begin{bmatrix}\mathbf{y}\\ \mathbf{f}_*\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathbf{K}(\mathbf{X},\mathbf{X})+\sigma_\epsilon^2\mathbf{I} & \mathbf{K}(\mathbf{X},\mathbf{X}_*)\\ \mathbf{K}(\mathbf{X}_*,\mathbf{X}) & \mathbf{K}(\mathbf{X}_*,\mathbf{X}_*)\end{bmatrix}\right)}$$

$$\boxed{k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \mathbf{exp}\left(-\frac{1}{2\lambda}(\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}')\right)}$$

- The optimized hyperparameters $\lambda$ and $\sigma_f^2$

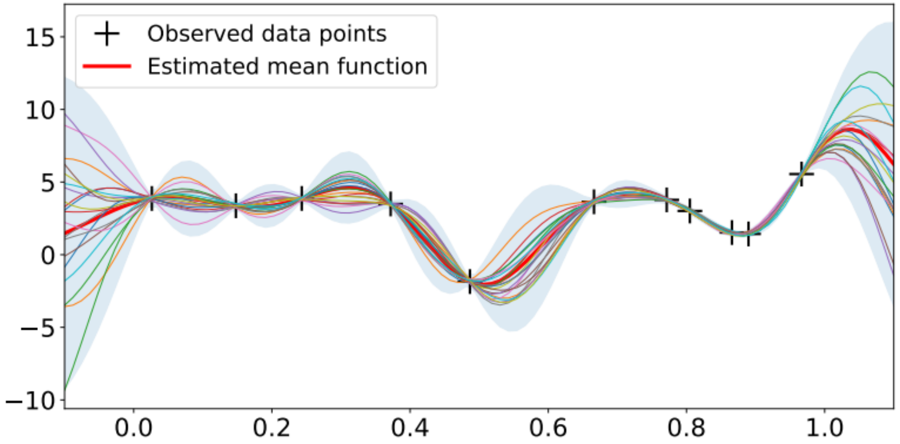
$$\lambda, \sigma_f^2 = \max_{\lambda,\, \sigma_f^2} \log p(\mathbf{y}|\mathbf{X})$$

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^T[\mathbf{K}(\mathbf{X},\mathbf{X})+\sigma_\epsilon^2\mathbf{I}]^{-1}\mathbf{y} - \frac{1}{2}\log\det[\mathbf{K}(\mathbf{X},\mathbf{X})+\sigma_\epsilon^2\mathbf{I}] - \frac{n}{2}\log 2\pi$$




$\sigma_f = 0.0067$

$\lambda = 0.0967$



# Gaussian Process Regression

## Posterior predictions from a $\mathcal{GP}$

- The mean function of the $\mathcal{GP}$ can be given as

$$\mu_f(\mathbf{x}) = \mathbf{K}(\mathbf{x},\mathbf{X})[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2\mathbf{I}]^{-1}\mathbf{y}$$

and covariance function as

$$cov_f(\mathbf{x},\mathbf{x}') = k(\mathbf{x},\mathbf{x}') - \mathbf{K}(\mathbf{x},\mathbf{X})[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2\mathbf{I}]^{-1}\mathbf{K}(\mathbf{X},\mathbf{x}')$$

$$\mathbf{K}(\mathbf{x},\mathbf{X}) = [k(\mathbf{x},\mathbf{x}_1) \quad k(\mathbf{x},\mathbf{x}_2) \quad \cdots \quad k(\mathbf{x},\mathbf{x}_n)]$$

$$\mathbf{K}(\mathbf{X},\mathbf{x}') = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}') \\ k(\mathbf{x}_2,\mathbf{x}') \\ \vdots \\ k(\mathbf{x}_n,\mathbf{x}') \end{bmatrix}$$