中国传媒大学

深度学习与类脑计算

（四）

曹立宏

脑科学与智能媒体研究院

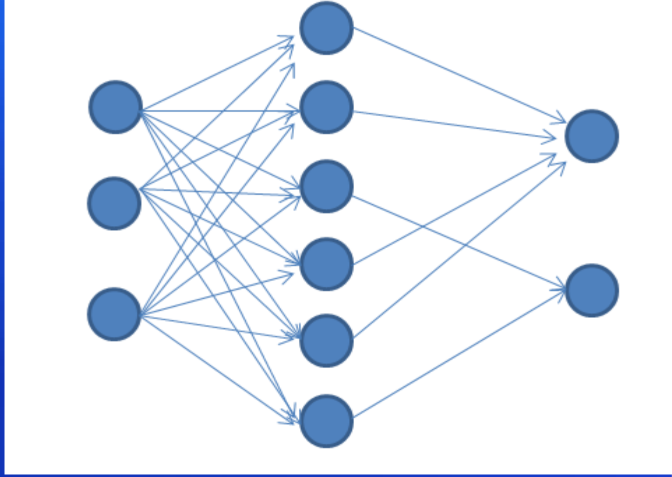
Review

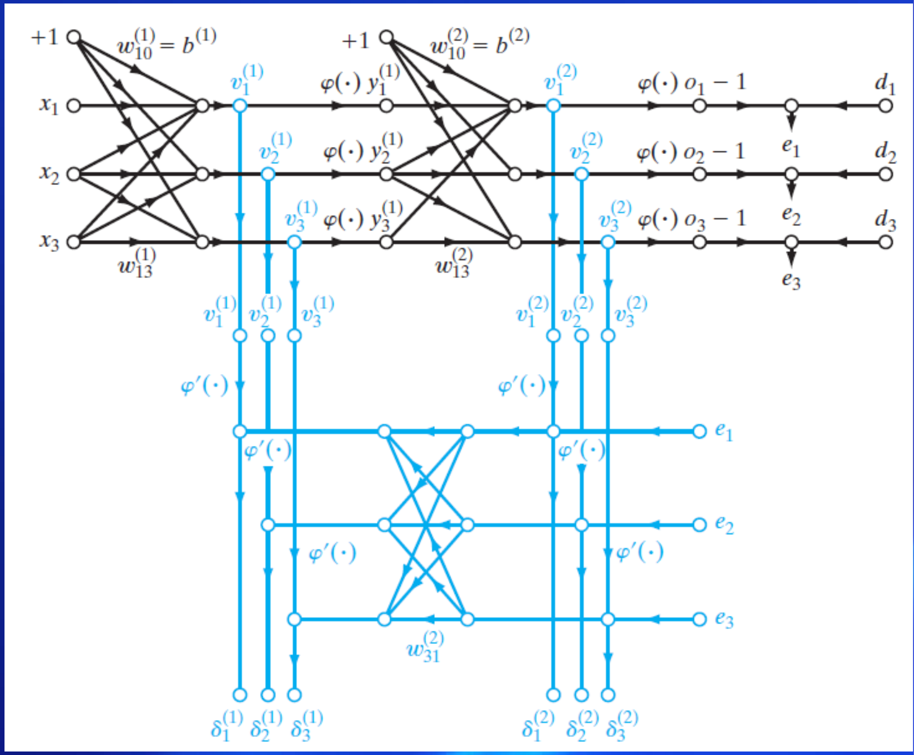
- Universal Approximation Theorem
- MLP with BP

Overfitting

Underfitting

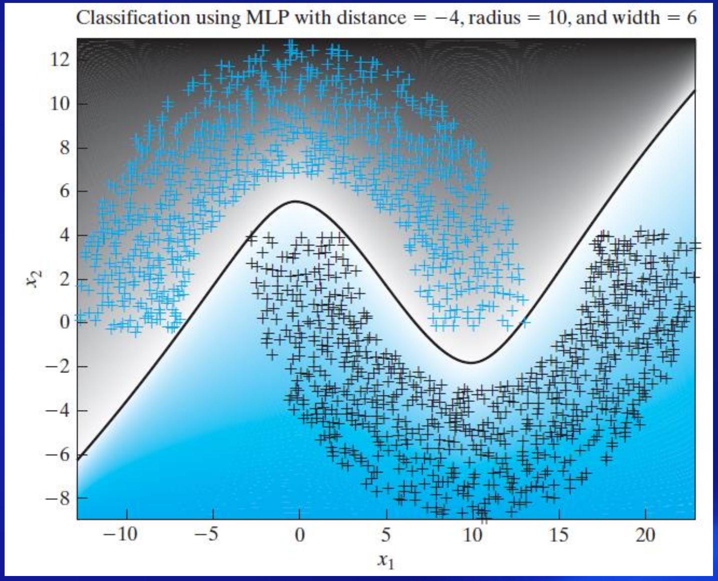
Curse of Dimensionality

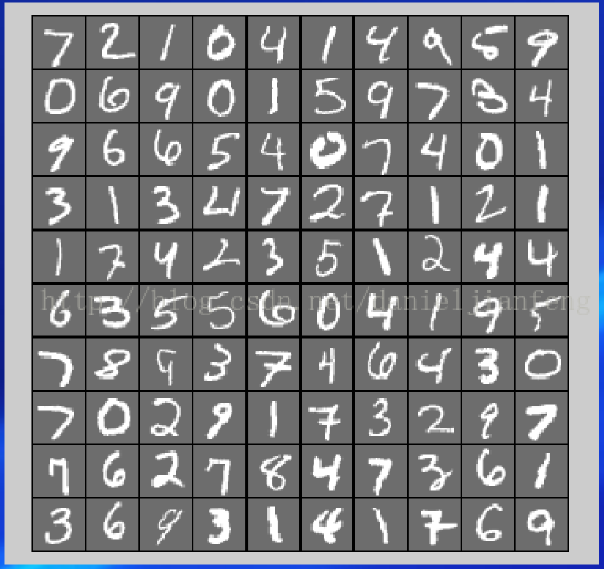




Homework

- Program BP algorithm in Python
- Pattern classification (ref. p150-153 on Haykin)
- Change the activation function of hidden layers to ReLU
- The MNIST Database （陈雯婕）



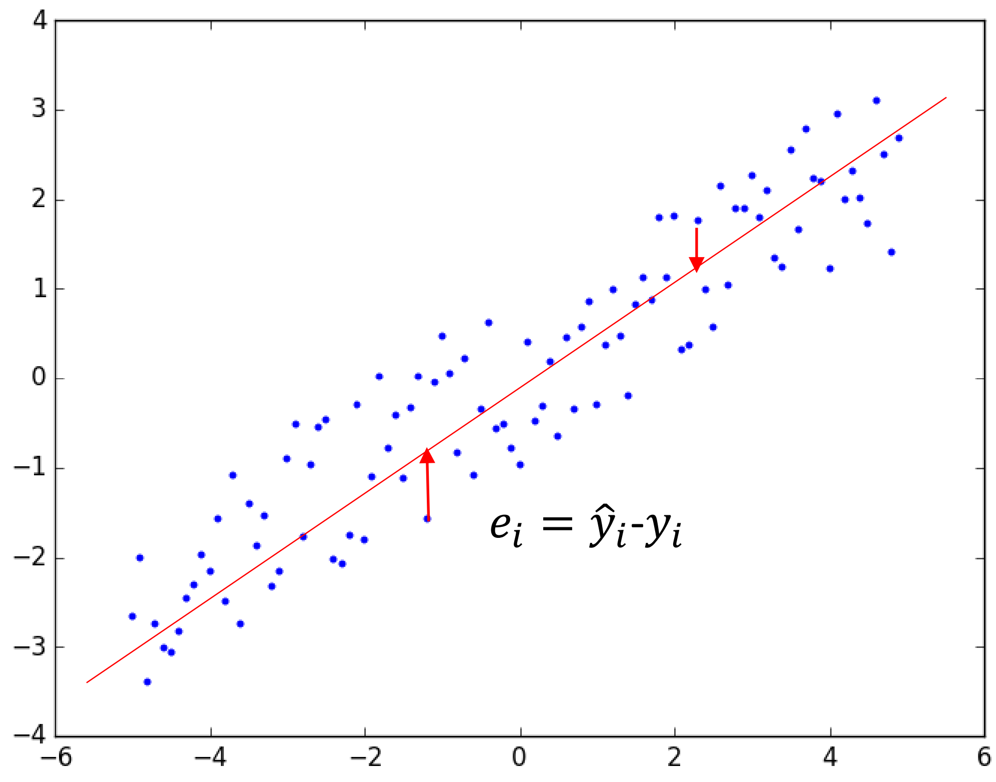


最小二乘法

- 假设

$$y = f(x)$$

$$\hat{y}_i = f(x_i)$$



想法1

$$\mathcal{E} = \sum_{i=1}^{N} {e_i}^2$$

想法2

$$\mathcal{E} = \sum_{i=1}^{N} |e_i|$$

想法n

假设1 $y = f(x) = ax + b$

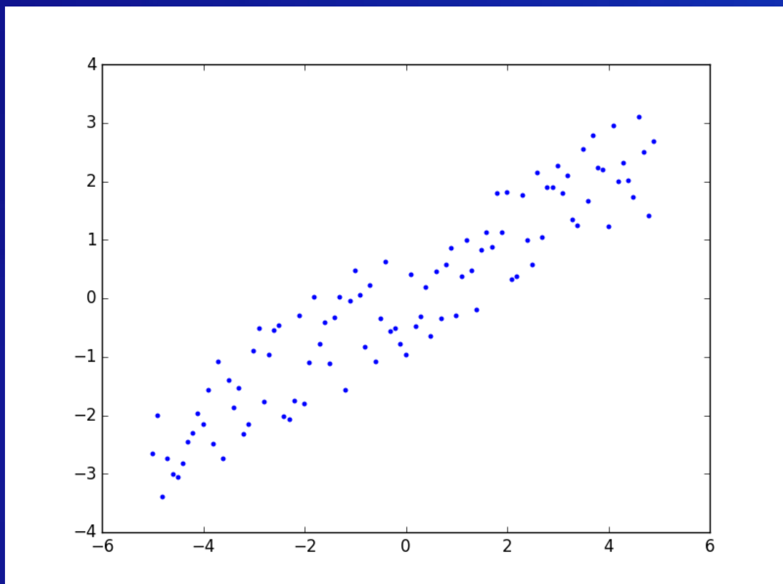
假设2 $y = f(x) = ax^2 + bx + c$

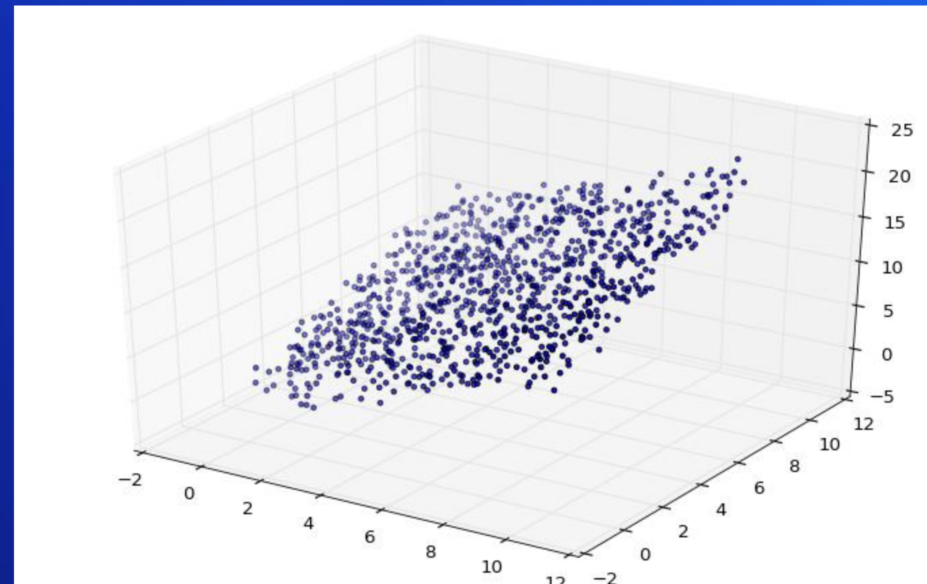
如果X的观察也不准确呢？

如果我们知道Y的观察比X的准1倍呢？

如何知道因该用几阶呢？

Regression





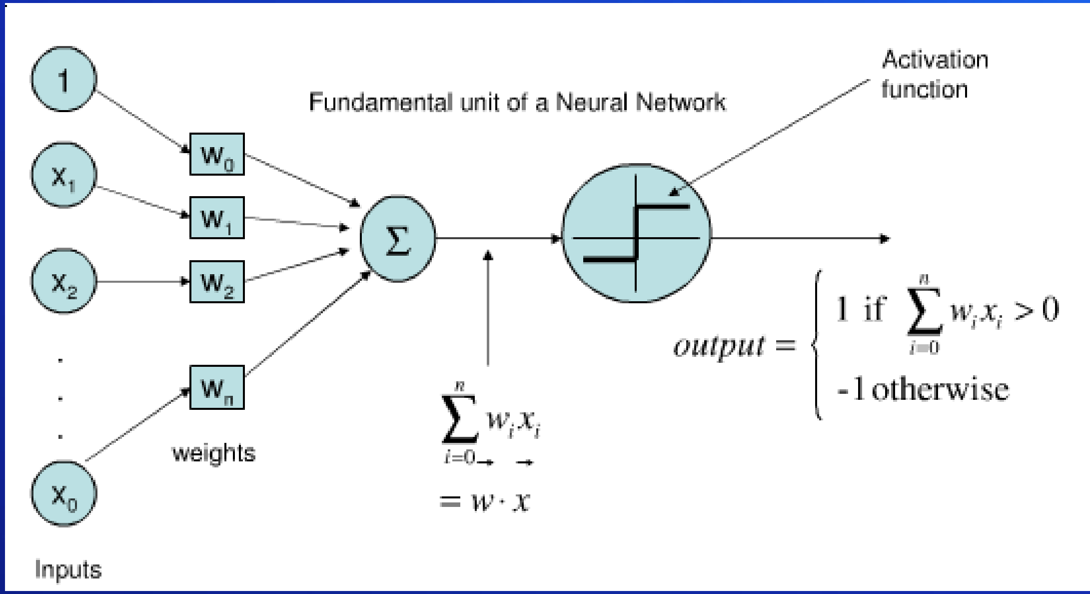
$$g:\ R^m \rightarrow R^1 \qquad \{\vec{x}_i, y_i, |\ i=1,\ \dots,\ \mathrm{N}\}$$

$$y = g(\vec{x}) = f(\vec{x}, \vec{w}) \qquad \vec{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_k \end{pmatrix}$$

Linear Regression

- Perceptron

$$f(\vec{x},\vec{w}) = \vec{w}^T\vec{x} + w_0$$



- General

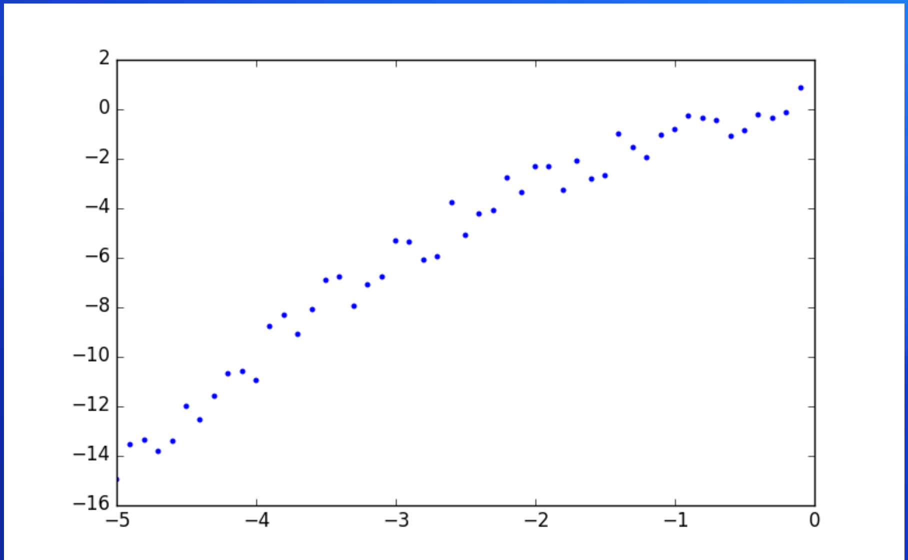
$$f(\vec{x},\vec{w}) = \vec{w}^T\vec{x}$$

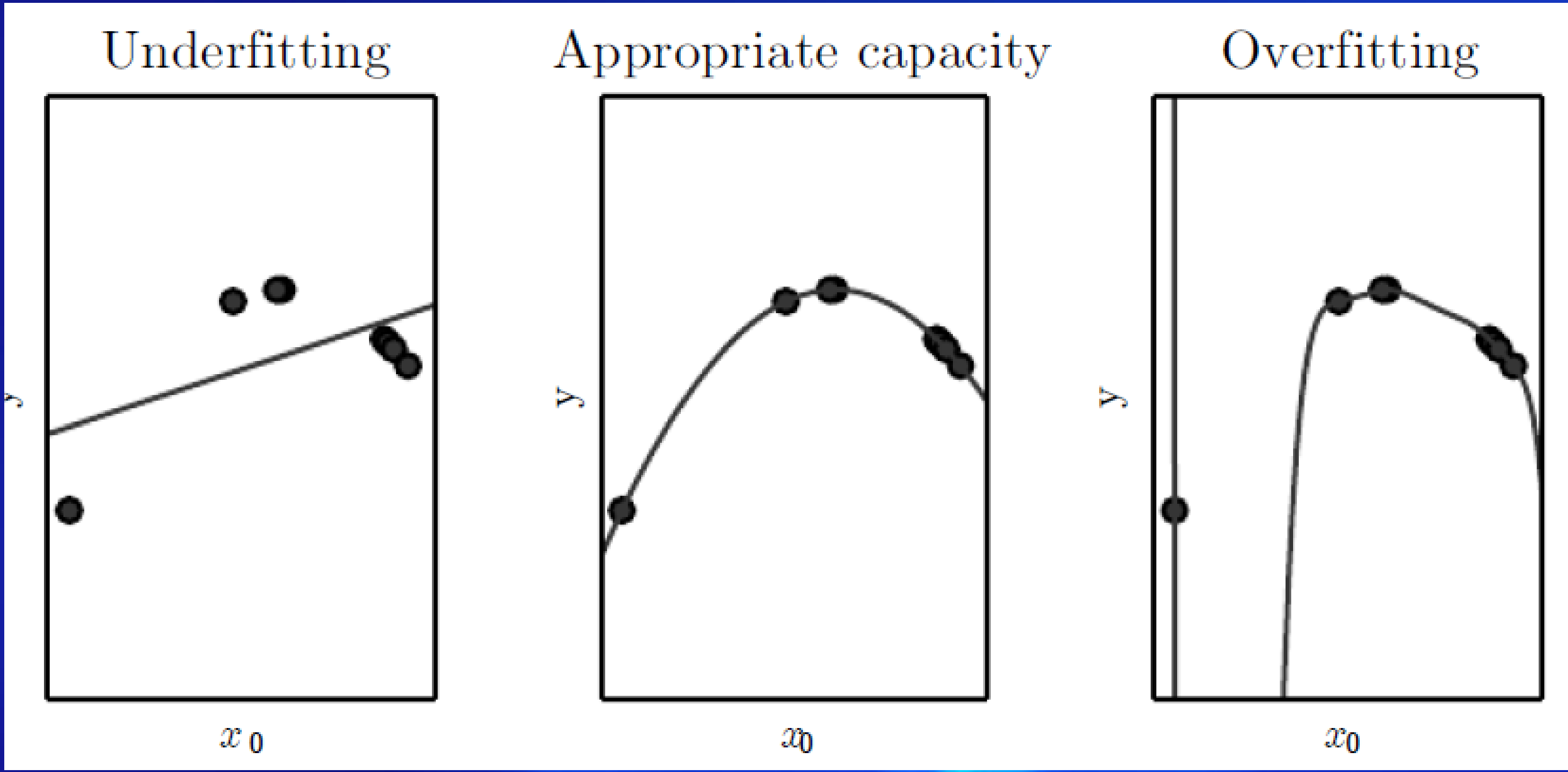
$$x_0 = 1$$

- Goal is to find $\hat{\vec{w}}$, s.t., $y_i - f(\vec{x}_i, \hat{\vec{w}})$ Small for all i=1,..N

More importantly for new $\vec{x}$

Underfitting and Overfitting





Errors-Overfitting-Underfitting

- Dataset
  - Training set
  - Test set

$$\left\{\vec{x}_i, y_i, \middle| \; i=1,2,\ldots,N,N+1,\ldots N+M\right\}$$

- Training error
  - Training set

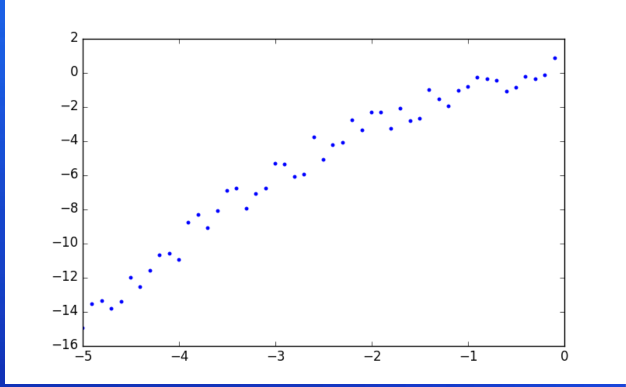
$$MSE_{training} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - f\left(\vec{x}_i, \hat{\vec{w}}\right)\right)^2$$

- Generalization or test error
  - Testing set

$$MSE_{test} = \frac{1}{M}\sum_{i=N+1}^{N+M}\left(y_i - f\left(\vec{x}_i, \hat{\vec{w}}\right)\right)^2$$

- Underfitting
  - $MSE_{training}$ is NOT small
- Overfitting
  - $MSE_{training} \ll MSE_{test}$

Capacity and Structure



- Structure implies capacity
  - Linear form can't make good fit for data generated by non-linear function
  - Can't do much for underfitting
- Expand structures/parameters to increase capacity

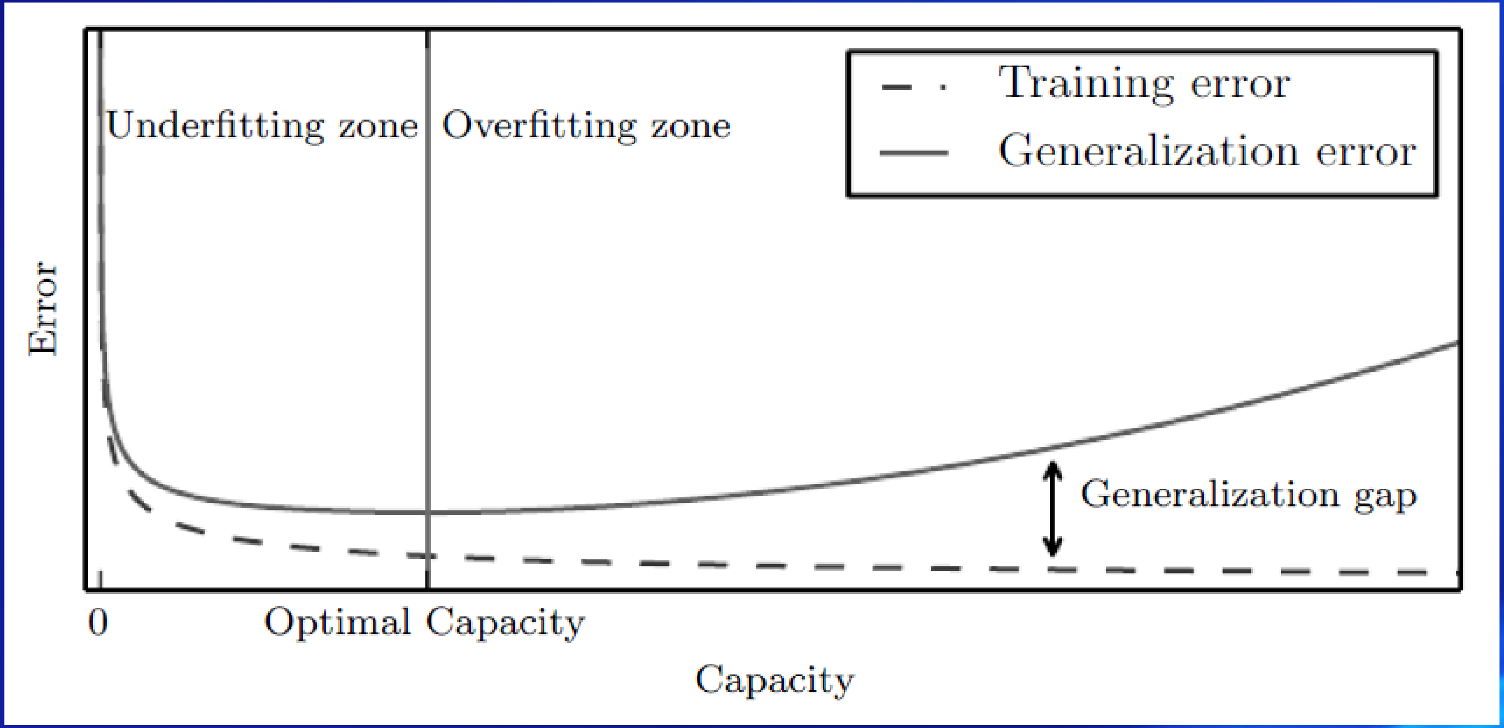
$$f(\vec{x}) = \overrightarrow{w}^T \vec{x} + \sum_i \sum_j \theta_{ij} x_i x_j$$

We can eliminate or regularize parameters for overfitting!

- Non-Parametric model has the highest capacity

Capacity-Overfitting-Underfitting

- Typical relationship between capacity and error



REGULARIZATION

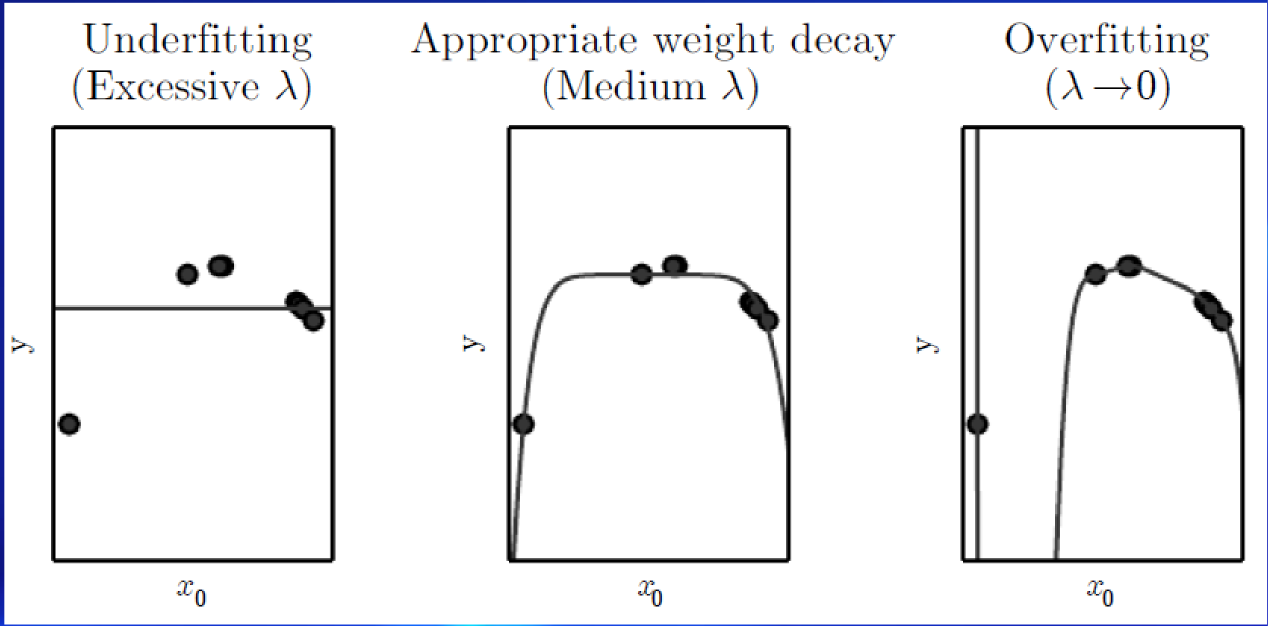
- Regression with regularization

$$MSE_{training}(\vec{w}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\vec{x}_i, \vec{w}))^2$$

$$J(\vec{w}) = MSE_{training}(\vec{w}) + \lambda \nabla f^T \nabla f$$ $\nabla f = \vec{w}$ Linear Reg

$$J(\vec{w}) = MSE_{training}(\vec{w}) + \lambda \Omega(\vec{w})$$ Regularizer

Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error



Estimation, Bias and Variance

- Point estimation $x_i \sim F(x, \theta), i = 1, \ldots m$

$$\hat{\theta}_m = g(\vec{x}_1, \vec{x}_2 \ldots, \vec{x}_m)$$

Random Variable/Vector

$$Bias(\hat{\theta}_m) = E(\hat{\theta}_m) - \theta$$

Unbiased if $Bias(\hat{\theta}_m)$ =0

asymptotically Unbiased if $\lim_{m \to \infty} Bias(\hat{\theta}_m)$ =0

$$Var(\hat{\theta}_m) = E(\hat{\theta}_m - E(\hat{\theta}_m))^2$$

$$SE(\hat{\theta}_m) = \sqrt{Var(\hat{\theta}_m)}$$

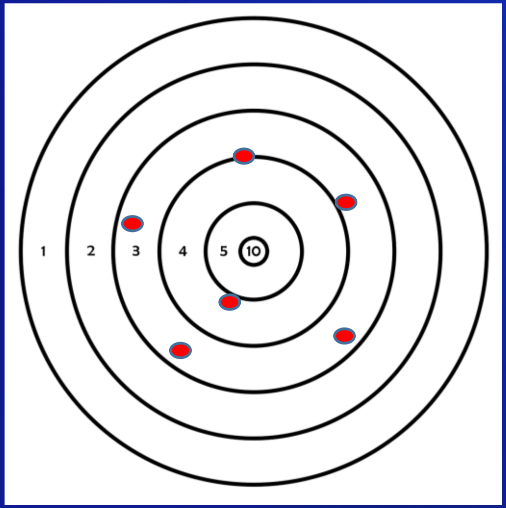
Standard error

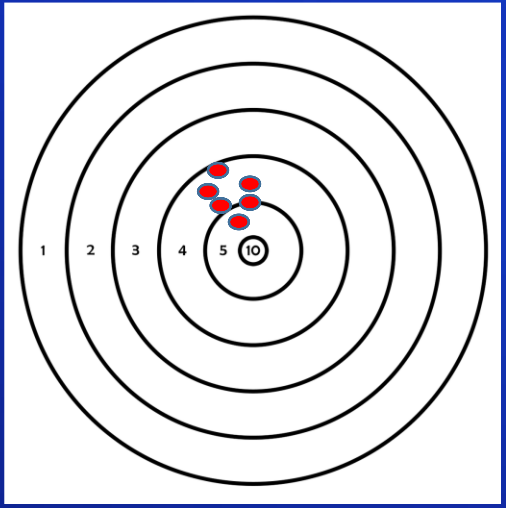
For Gaussian i.i.d. distribution, $x_i \sim N(\mu, \sigma^2), i = 1, \ldots m$
1, Sample mean is unbiased.
2, Sample variance is biased, but asymptotically unbiased
3, $SE(\hat{\mu}) = \sigma/\sqrt{m}$

$$\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

Mean Squared Error (MSE)

- Which one is better?



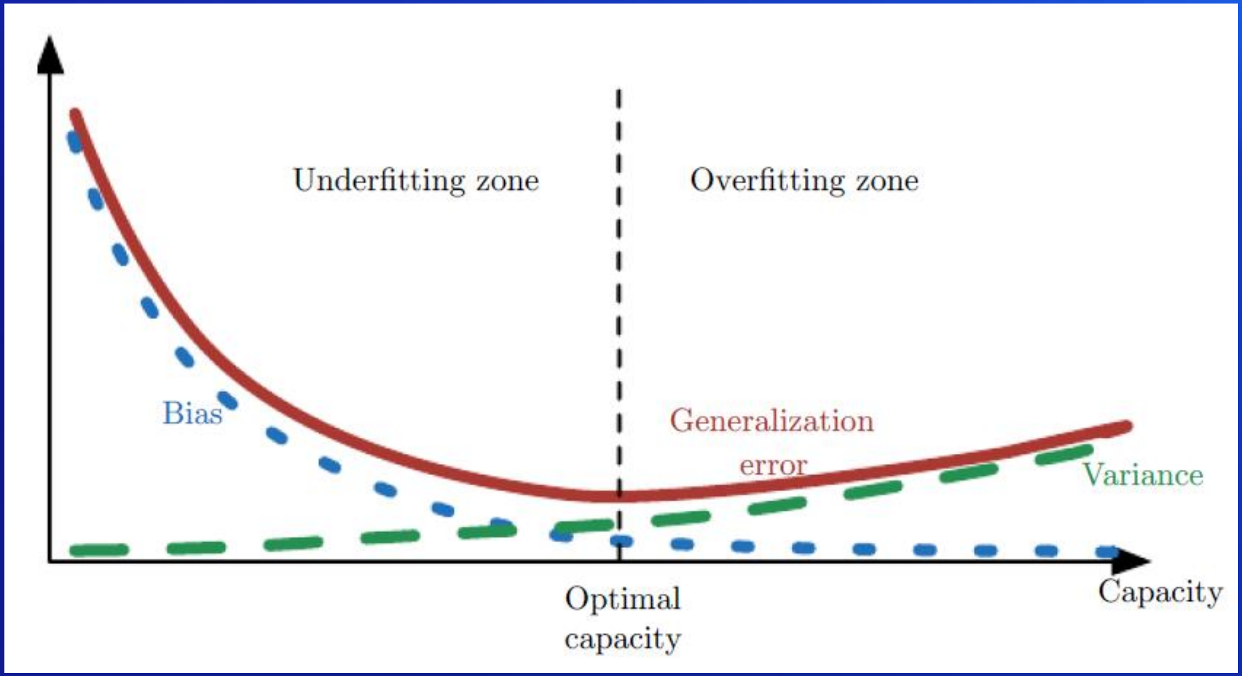


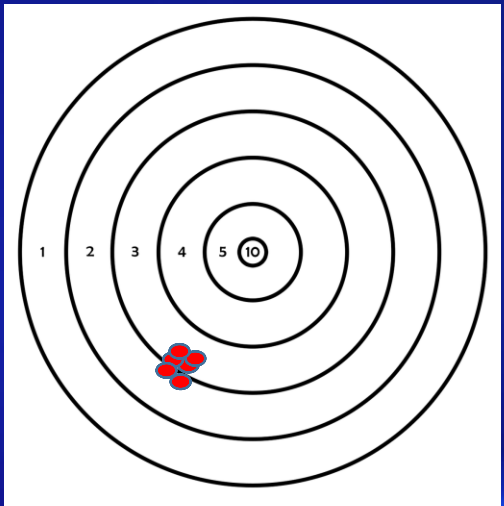
$$MSE(\hat{\theta}_m) = E\left[\left(\hat{\theta}_m - \theta\right)^2\right] = Bias^2(\hat{\theta}_m) + Var(\hat{\theta}_m)$$

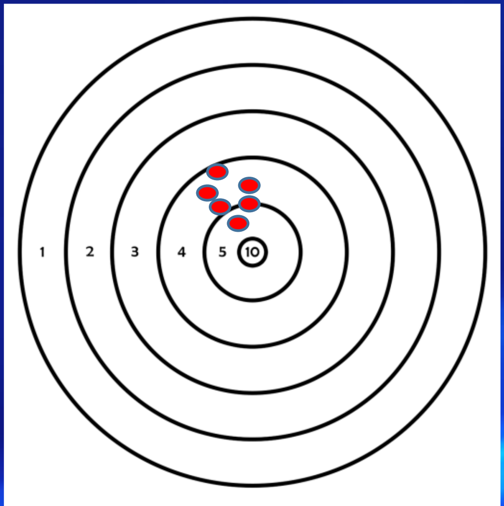
$$Bias(\hat{\theta}_m) = E(\hat{\theta}_m) - \theta$$

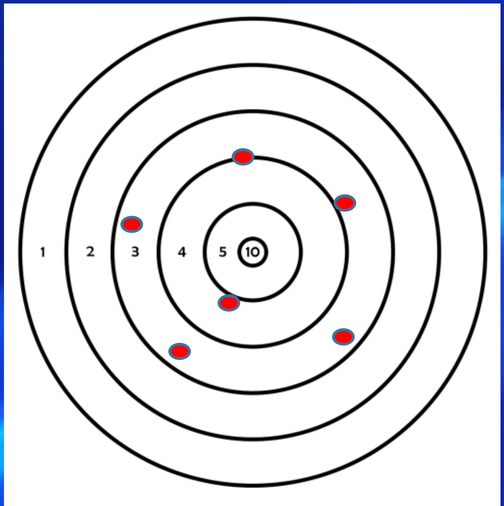
$$Var(\hat{\theta}_m) = E(\hat{\theta}_m - E(\hat{\theta}_m))^2$$

Tradeoff of Bias and Variance









Maximum Likelihood Estimation

$\vec{x}_i \sim p(\vec{x};\ \theta),\ \ i = 1, \ldots m$ i.i.d.

$$\theta_{ML} = arg \max_{\theta} \prod_{i=1}^{m} p(\vec{x}_i;\ \theta)$$

$$\theta_{ML} = arg \max_{\theta} \sum_{i=1}^{m} Log\, p(\vec{x}_i;\ \theta)$$

Maximum Likelihood Estimation for Regression

$$\hat{y}_i = f(x_i, \vec{\theta}) \sim N(y_i, \sigma) \text{ i.i.d.}$$

$$\sum_{i=1}^{m} Log\, p(\vec{x}_i;\, \theta) = -mlog\sigma - \frac{m}{2}\log(2\pi) - \sum_{i=1}^{m} \frac{(\hat{y}_i - y_i)^2}{2\sigma^2}$$

$$\sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \sum_{i=1}^{m} \left(f(x_i, \vec{\theta}) - y_i\right)^2 = m * MSE_{Train}$$

- In this case ML estimation is as same as minimizing MSE for training set
- In general, ML estimation has consistency and efficiency

Bayesian Statistics

- Bayesian View
  - Given observations, what's the probability of parameters?

The Prior

$$p(\vec{\theta}|\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m) = \frac{p(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m|\vec{\theta})p(\vec{\theta})}{p(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m)}$$

  - The estimation of parameters is the one

s.t. $\vec{\theta}_{MAP} = Arg \max_{\vec{\theta}} p(\vec{\theta}|\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m)$

Maximum A Posteriori (MAP) estimation

MAP estimation justify ML estimation by the prior

$$\begin{aligned}\vec{\theta}_{MAP} &= Arg \max_{\vec{\theta}} p(\vec{\theta}|\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m) \\ &= Arg \max_{\vec{\theta}} \left[\log\left(p(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_m|\vec{\theta})\right) + \log(p(\vec{\theta}))\right]\end{aligned}$$

When we have no idea about prior, it vanishes!

Bayesian Linear Regression

$$\hat{y}_i = f(\vec{x}_i, \vec{w}) = \vec{w}^T \vec{x}_i \sim N(y_i, \sigma) \text{ i.i.d.}$$

$$Y = \vec{w}^T X \quad p(Y|\vec{w}, X) \sim N(\vec{w}^T X, I)$$

$$p(Y|\vec{w}, X) \propto exp(-\frac{1}{2}(Y - X\vec{w})^T (Y - X\vec{w}))$$

IF $p(\vec{w}) \sim N(\vec{0}, \frac{1}{\alpha} I)$

α 越大，先验性越强！$\vec{w}$越小

Then $p(\vec{w}|Y, X) = p(Y|\vec{w}, X) p(\vec{w}) = \cdots$

$$\vec{w}_{MAP} = (X^T X + \alpha I)^{-1} (X^T Y)$$

相当于 Regularization

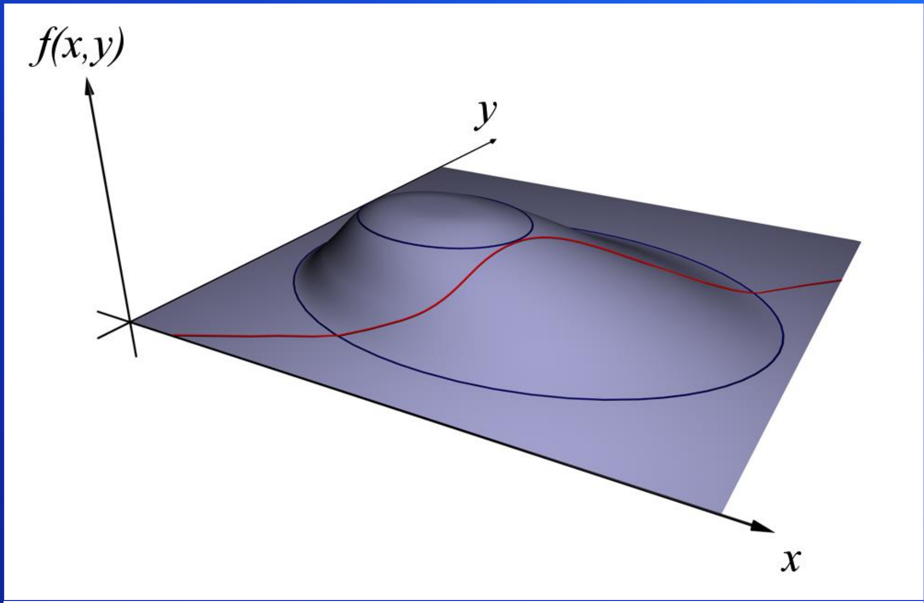
$$J(\vec{w}) = MSE_{training}(\vec{w}) + \lambda \nabla f^T \nabla f \qquad \nabla f = \vec{w}$$

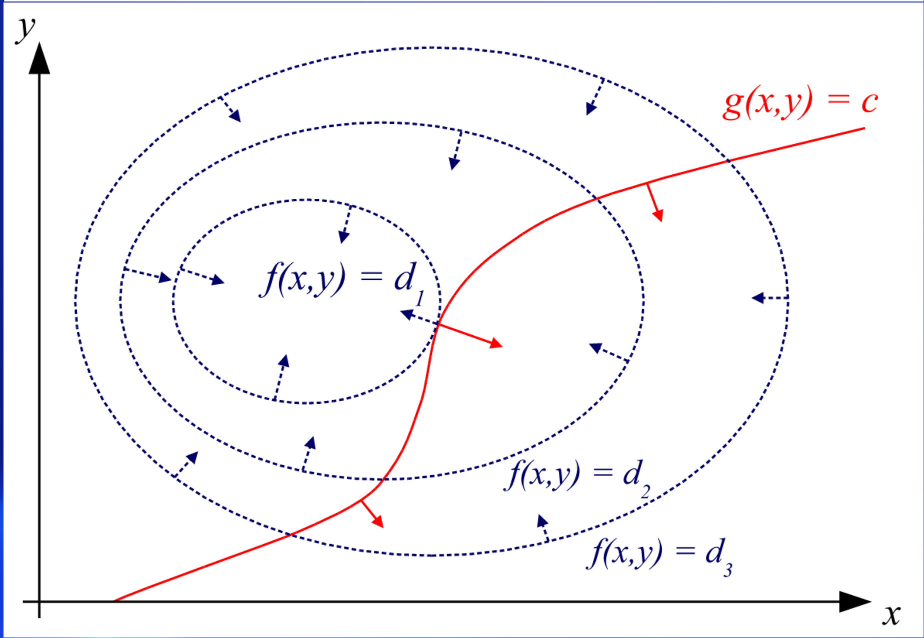
Bayesian or MAP estimation or regularization is more balanced

Lagrange Multiplier

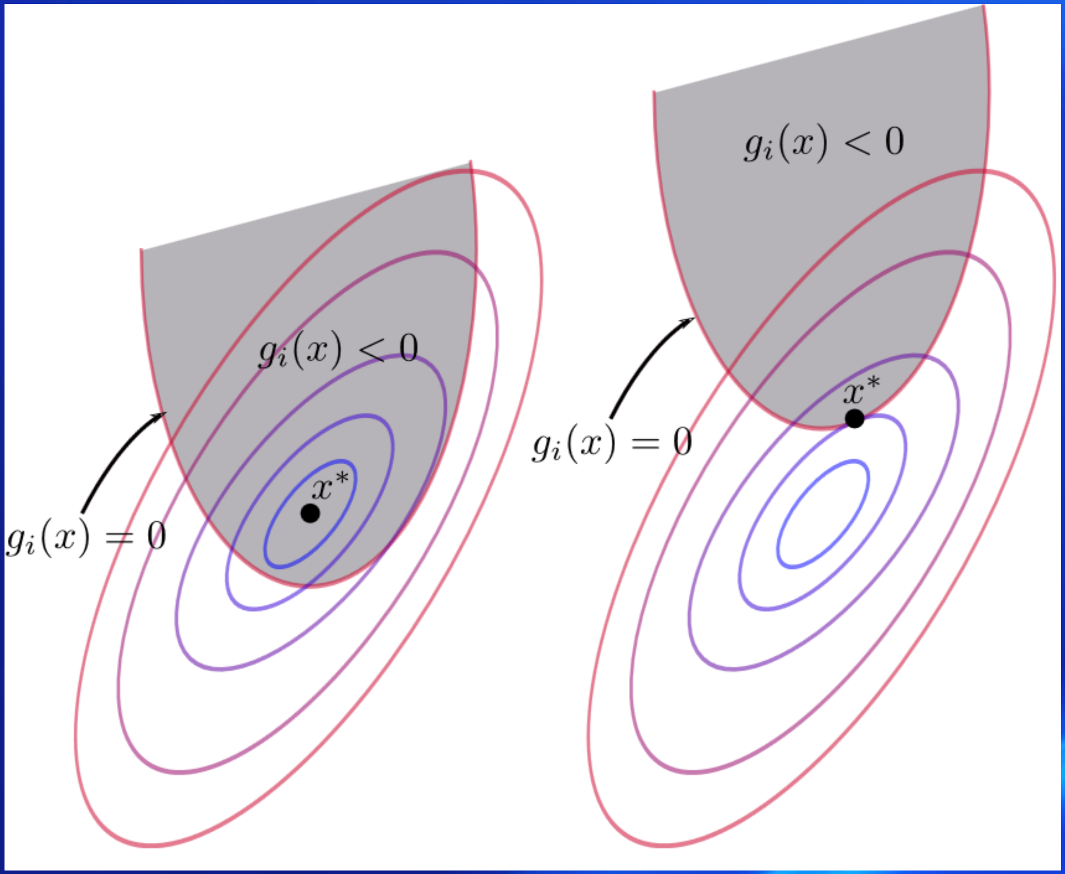
- Maximize $f(x, y)$
- Subject to: $g(x, y) = 0$

$$L(x, y, \lambda) = f(x, y) + \lambda g(x, y)$$





KKT



From Regression to Categorization

- Discrete responses
- Supervised Learning
  - Logistic Regression
  - SVM
- Unsupervised Learning
  - Dimension Reduction
  - K-mean Clustering
  - Kernel Method
  - RBF Network

Homework

- Finishing up the last homework
- How to write a technical report?
- Prove: Bayesian or MAP estimation for linear regression is equivalent to regularization of MSE
- We（陈雯婕）will give you a dataset RegData2D

$$\left\{x_i, y_i \;\middle|\; i=1,2,\ldots,N,N+1,\ldots N+M\right\}$$

use whatever you have learnt to find the best relationship between x and y. Write a report, and specifically pay attention to underfitting and overfitting and how you come up with your best solutions.

- (optional) We（陈雯婕）will give you another dataset RegData3D, $\left\{x_i, y_i, z_i \middle| i=1,2,\ldots,N,N+1,\ldots N+M\right\}$ do the same.