南方科技大学

MAT8034: Machine Learning

Regularization and Model Selection

Fang Kong

https://fangkongx.github.io/Teaching/MAT8034/Spring2025/index.html

Part of slide credit: Stanford CS229

Outline

- Regularization
- Implicit regularization effect
- Model selection via cross validation
- Bayesian statistics and regularization

Regularization

Intuition

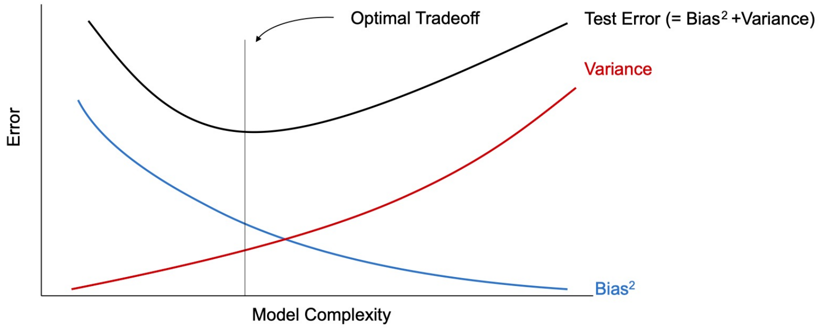


Figure 8.8: An illustration of the typical bias-variance tradeoff.

- Recall in the last lecture
  - Complex models may cause overfitting
- Objective
  - Choose a proper model complexity to achieve the optimal bias-variance tradeoff
- Model complexity measure
  - A function of the parameters, e.g., L2 norm of the parameters

Regularization

- Meaning of regularization
  - Adding an additional term to control the model complexity and prevent overfitting

$$J_\lambda(\theta) = J(\theta) + \lambda R(\theta)$$

  - $J(\theta)$: the original loss, e.g., MSE
  - $R(\theta)$: the regularizer, typically non-negative
  - $\lambda \geq 0$: regularization parameter

Regularization (cont'd)

- Meaning of regularization
  - Adding an additional term to control the model complexity and prevent overfitting

$$J_\lambda(\theta) = J(\theta) + \lambda R(\theta)$$

- Intuition
  - Find a model that both fit the data (a small loss $J(\theta)$) and have a small model complexity ($R(\theta)$)
  - λ balances the loss and model complexity

Regularization (cont'd)

- Meaning of regularization
  - Adding an additional term to control the model complexity and prevent overfitting

$$J_\lambda(\theta) = J(\theta) + \lambda R(\theta)$$

- Intuition
  - Find a model that both fit the data (a small loss $J(\theta)$) and have a small model complexity ($R(\theta)$)
  - λ balances the loss and model complexity
    - Small λ: minimizing the original loss with the regularizer as the tie-breaker
    - Large λ: tends to find a simpler model

Common regularizer

- L2 regularization
  - $R(\theta) = \frac{1}{2}||\theta||_2$
  - Prefer models with smaller L2 norm

- Also referred to as weight decay
  - Consider the stochastic gradient descent

$$\begin{aligned} \theta \leftarrow \theta - \eta \nabla J_\lambda(\theta) &= \theta - \eta\lambda\theta - \eta\nabla J(\theta) \\ &= \underbrace{(1 - \lambda\eta)\theta}_{\text{decaying weights}} \; -\eta\nabla J(\theta) \end{aligned}$$

Common regularizer (cont’d)

- Other influences: Impose structures on the model parameters
  - Suppose we already know the model is sparse
  - i.e., the non-zeros is small
  - We can then add regularization term:$||\theta||_0$
  - Such regularization narrows our search space and makes the complexity of the model family smaller

  - Regularization loss with $||\theta||_0$ is not continuous
  - Replace $||\theta||_0$ by $||\theta||_1$ in the loss can have similar effect of sparsity

Implicit regularization effect

Implicit regularization effect

- New concept/phenomenon observed in the deep learning era

- Meaning
  - The optimizers can implicitly impose structures on parameters beyond what has been imposed by the regularized loss

Intuition

- In most classic settings
  - The optimal solution is unique
  - Any reasonable optimizer should converge to this point
- In deep learning
  - There are usually more than one (approximate) global minimum
  - Different optimizers may converge to different global minima
  - Though they have similar training loss
    - The solution may have dramatically different generalization performance

Illustration

-

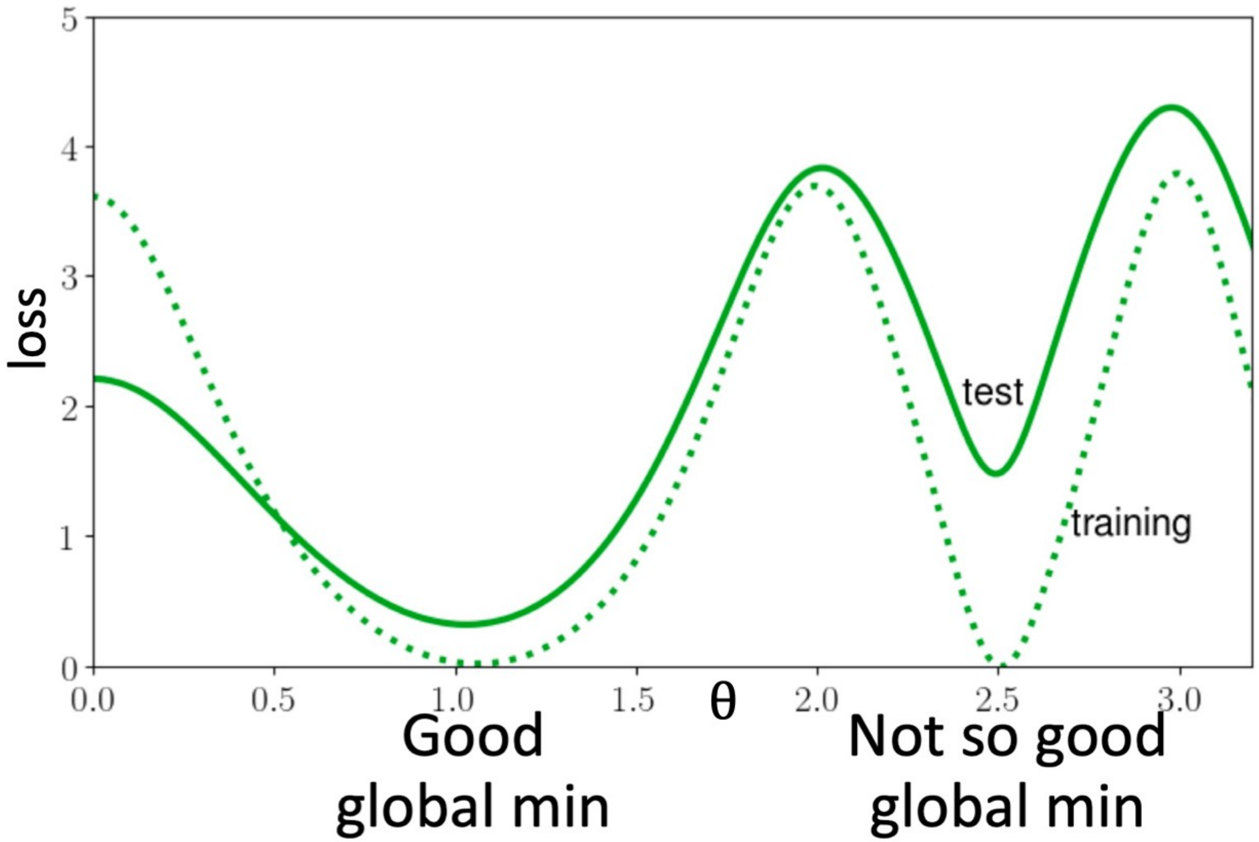


Illustration (cont’d)

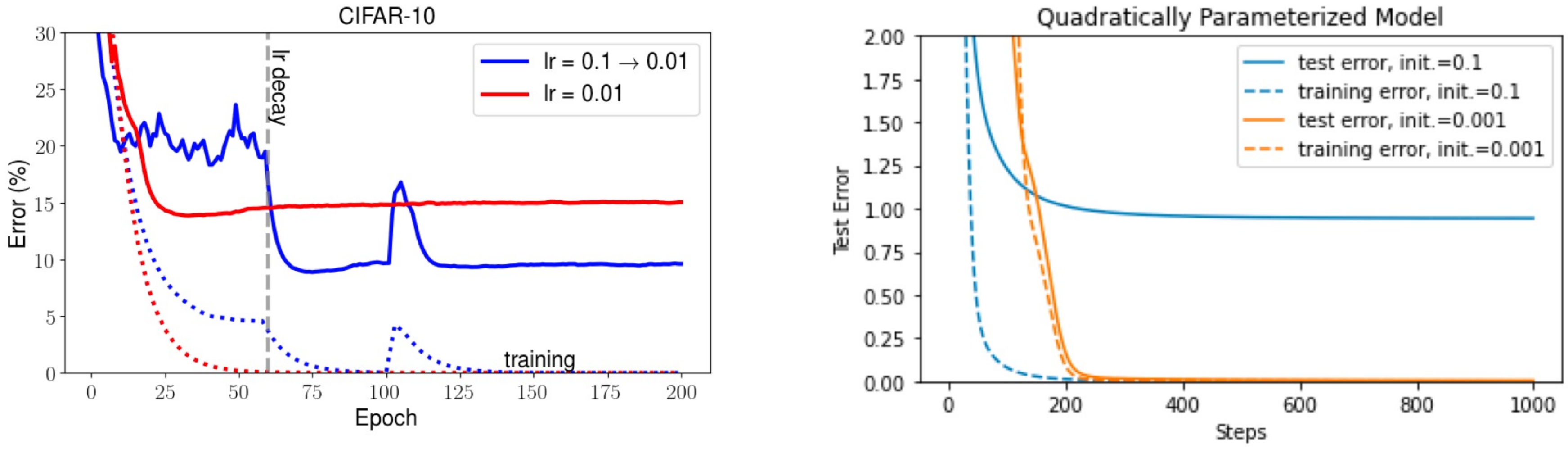


Figure 9.2: **Left:** Performance of neural networks trained by two different learning rates schedules on the CIFAR-10 dataset. Although both experiments used exactly the same regularized losses and the optimizers fit the training data perfectly, the models’ generalization performance differ much. **Right:** On a different synthetic dataset, optimizers with different initializations have the same training error but different generalization performance.[4]

Summary

- The role of optimizer
  - Not only minimizing the loss, but also imposes implicit regularization and affects the generalization of the model
  - Even though it achieves a small training error, there is still a space of improving generalization

What type of global minima may generalize better?

- Still an active research area

- Some heuristics
  - Larger initial learning rate
  - Smaller initialization
  - Smaller batch size
  - Introducing momentum

Model selection

Formulation

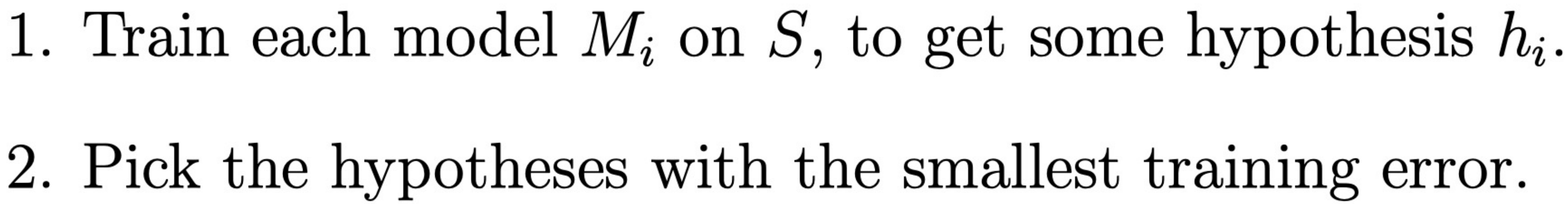
- To solve a problem, which model should we choose?
  - SVM or logistic regression?
  - For kernel methods, which order k of polynomial?

$$h_\theta(x) = g(\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_k x^k)$$

- Denote $\mathcal{M} = \{M_1, \ldots, M_d\}$ as all the models to choose

Solution 1: Select the one with the minimum training loss?

- Given the training set S

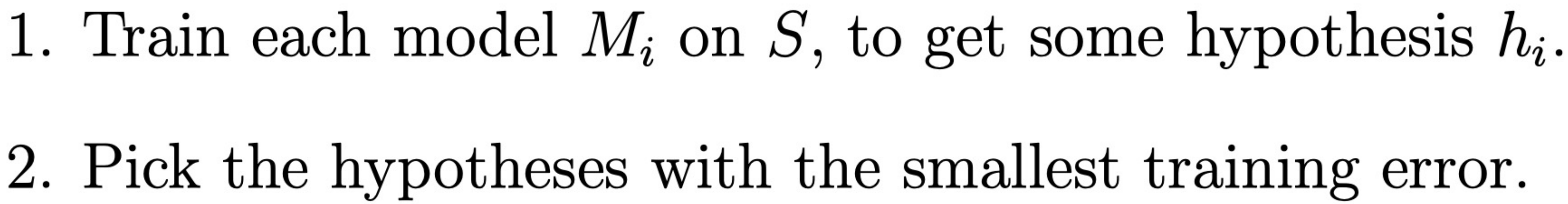


1. Train each model M_i on S, to get some hypothesis h_i.
2. Pick the hypotheses with the smallest training error.

- What’s the problem?

Solution 1: Select the one with the minimum training loss?

- Given the training set S



1. Train each model M_i on S, to get some hypothesis h_i.
2. Pick the hypotheses with the smallest training error.

- What’s the problem?
  - Lower training error prefers complex models
  - These models usually overfits

Solution 2: Hold-out cross validation

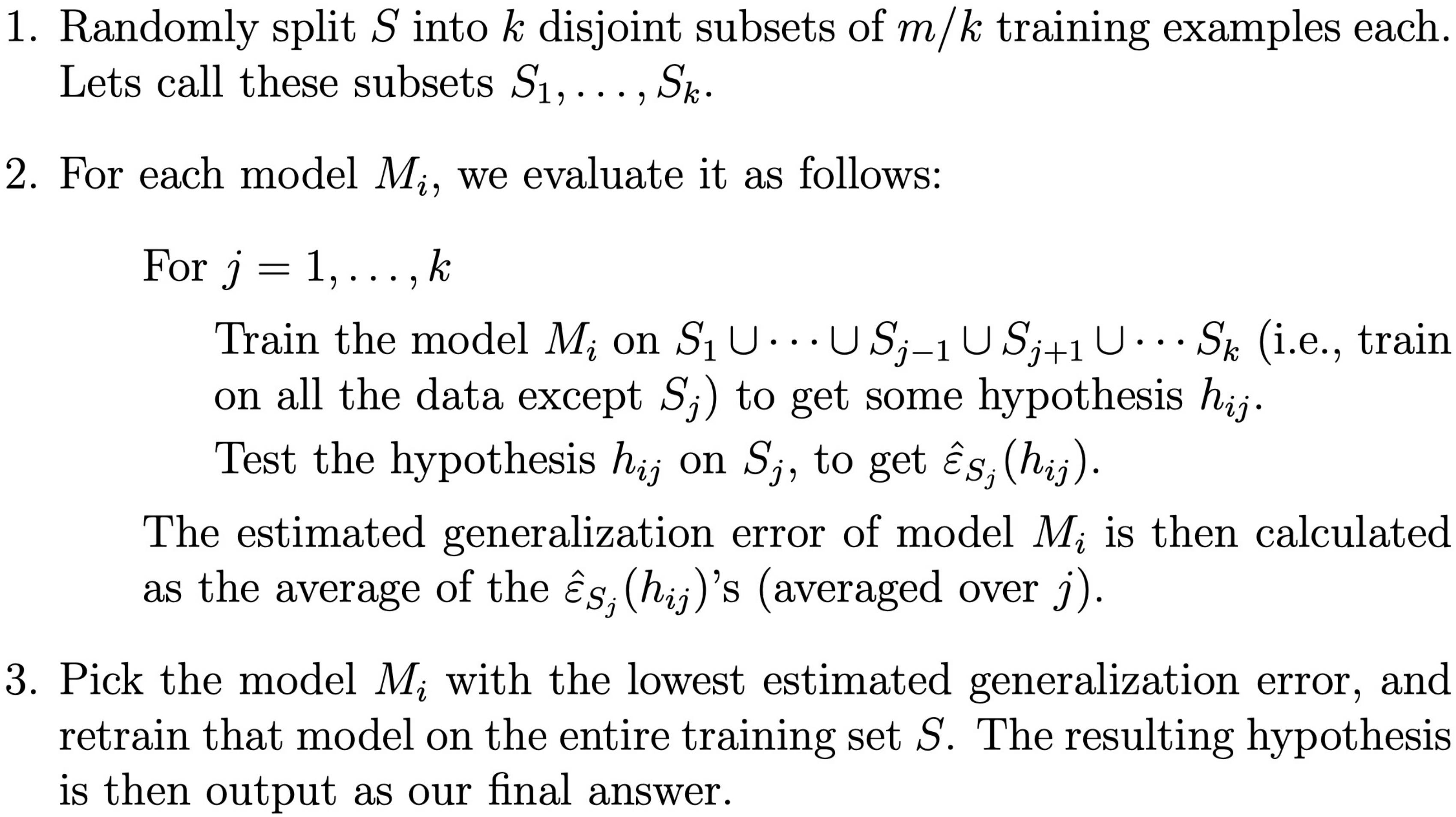
- Split the training set S
  - $S = S_{train}$(usually 70%) $+ S_{cv}$ (usually 30%)
  - Train each model M_i on S_{train} only, to get some hypothesis h_i
  - Evaluate h_i on S_{cv}, denote the error as $\hat{\varepsilon}_{S_{\mathrm{cv}}}(h_i)$ (validation error)
  - Pick the hypothesis with the smallest validation error

- The CV set plays the role of testing set
- Evaluate the model in terms of approximate generalization error
- Avoid overfitting

Problem of hold-out cross validation

- The final model is only trained on 70% of the training set
- Especially in the case with small training set
  - Waste about 30% of the data

Improvement: k-fold cross validation



1. Randomly split S into k disjoint subsets of m/k training examples each. Lets call these subsets $S_1, \ldots, S_k$.

2. For each model M_i, we evaluate it as follows:

   For $j = 1, \ldots, k$

      Train the model M_i on $S_1 \cup \cdots \cup S_{j-1} \cup S_{j+1} \cup \cdots S_k$ (i.e., train on all the data except S_j) to get some hypothesis h_{ij}.

      Test the hypothesis h_{ij} on S_j, to get $\hat{\varepsilon}_{S_j}(h_{ij})$.

   The estimated generalization error of model M_i is then calculated as the average of the $\hat{\varepsilon}_{S_j}(h_{ij})$'s (averaged over j).

3. Pick the model M_i with the lowest estimated generalization error, and retrain that model on the entire training set S. The resulting hypothesis is then output as our final answer.

- Typical choice: k=10

Bayesian statistics and regularization

Frequentist V.S. Bayesian

- Consider θ as the model parameter
- Frequentist view
  - θ is constant-valued but unknown
  - We need to estimate this parameter, such as MLE

$$\theta_{\mathrm{MLE}} = \arg\max_{\theta} \prod_{i=1}^{n} p(y^{(i)} | x^{(i)}; \theta).$$

- Bayesian review
  - θ is a random variable with unknown value
  - We can specify a prior distribution $p(\theta)$ on θ that expresses our “prior beliefs” about the parameters

Bayesian view

- Given a training set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
- Compute the posterior of θ

$$\begin{aligned} p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\ &= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)\right) p(\theta)}{\int_\theta \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)p(\theta)\right) d\theta} \end{aligned}$$

- To predict the label of a new data x

$$p(y|x, S) = \int_\theta p(y|x, \theta)p(\theta|S)d\theta \qquad \mathrm{E}[y|x, S] = \int_y yp(y|x, S)dy$$

Bayesian view (cont'd)

- Given a training set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
- Compute the posterior of θ

$$\begin{aligned} p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\ &= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)\right) p(\theta)}{\int_\theta \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)p(\theta)\right) d\theta} \end{aligned}$$

Computationally difficult!

- To predict the label of a new data x

$$p(y|x, S) = \int_\theta p(y|x, \theta)p(\theta|S)d\theta \qquad \mathrm{E}[y|x, S] = \int_y yp(y|x, S)dy$$

Maximum a posteriori (MAP)

$$
\begin{aligned}
p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\
&= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)\right) p(\theta)}{\int_\theta \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)p(\theta)\right) d\theta}
\end{aligned}
$$

- Approximate the posterior distribution for θ
- Use single point estimate

$$\theta_{\text{MAP}} = \arg\max_\theta \prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)p(\theta)$$

Maximum a posteriori (MAP)

$$\begin{aligned} p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\ &= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)\right) p(\theta)}{\int_\theta \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)p(\theta)\right) d\theta} \end{aligned}$$

- Approximate the posterior distribution for θ
- Use single point estimate

$$\theta_{\text{MAP}} = \arg\max_\theta \prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)\boxed{p(\theta)}$$

Additional term compared with MLE

- The prior $p(\theta)$ is usually assumed to be $\theta \sim \mathcal{N}(0, \tau^2 I)$
- Parameters with smaller norm are more preferred than MLE
- Less susceptible to overfitting

Summary

- Regularization
  - Intuition, common regularizers
- Implicit regularization effect
  - Optimizer in deep learning
- Model selection via cross validation
  - Hold-out CV
  - K-fold CV
- Bayesian statistics and regularization
  - MAP