Underfitting, Overfitting and Cross-Validation

Prof. Mingkui Tan

SCUT Machine Intelligence Laboratory (SMIL)

Contents

Contents

Why We Need Validation ?

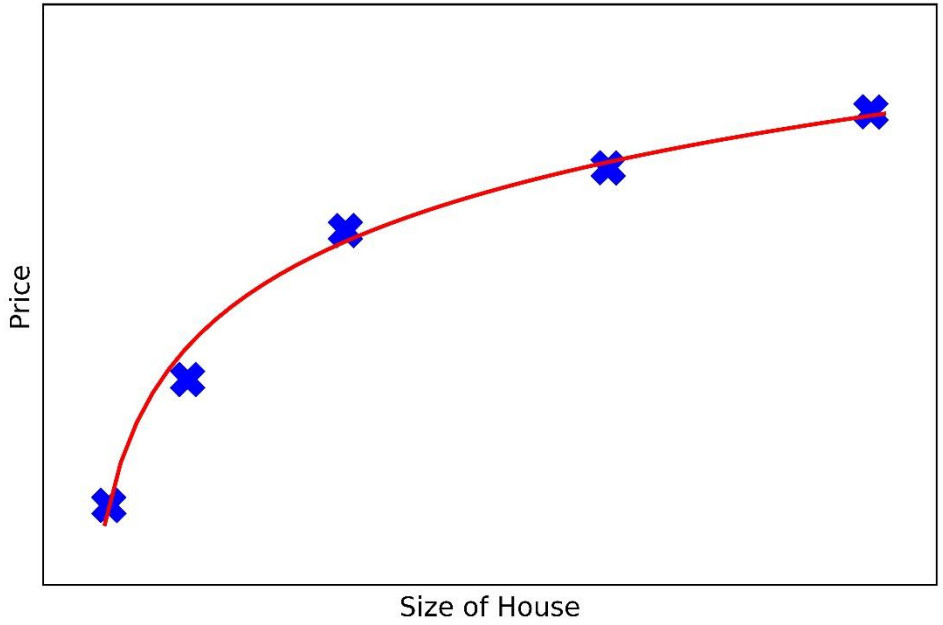
- Business Reasons
  - Need to choose the best model
  - Measure accuracy/power of the selected model
  - Good to measure ROI of the modeling project
- Statistical Reasons
  - Model building techniques are inherently designed to minimize “loss” or “bias”
  - To an extent, a model will always fit “noise” as well as “signal”
  - If you just fit a bunch of models on a given dataset and choose the “best” one, it will likely be overly “optimistic”

Notations

- Target Variable: y
  - What we are trying to predict
    - house price, ...
- Input Variables: $\mathrm{x} = [x_1, x_2, \cdots, x_m]$
  - Input features used to make predictions
    - size of house, location of house, ...
- Predictive Model: $\hat{y} = f(\mathrm{x})$
  - Estimate the unknown value $\hat{y}$ based on known values $\{x_i\}_{i=1}^{m}$

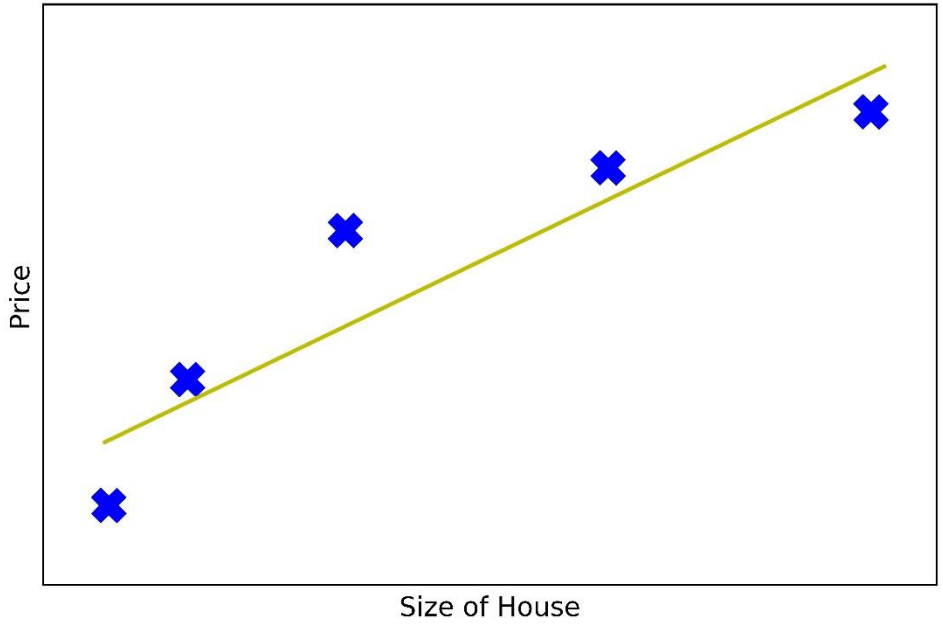
General Fitting Scheme

- Model is fitting and can capture the underlying trend of the data



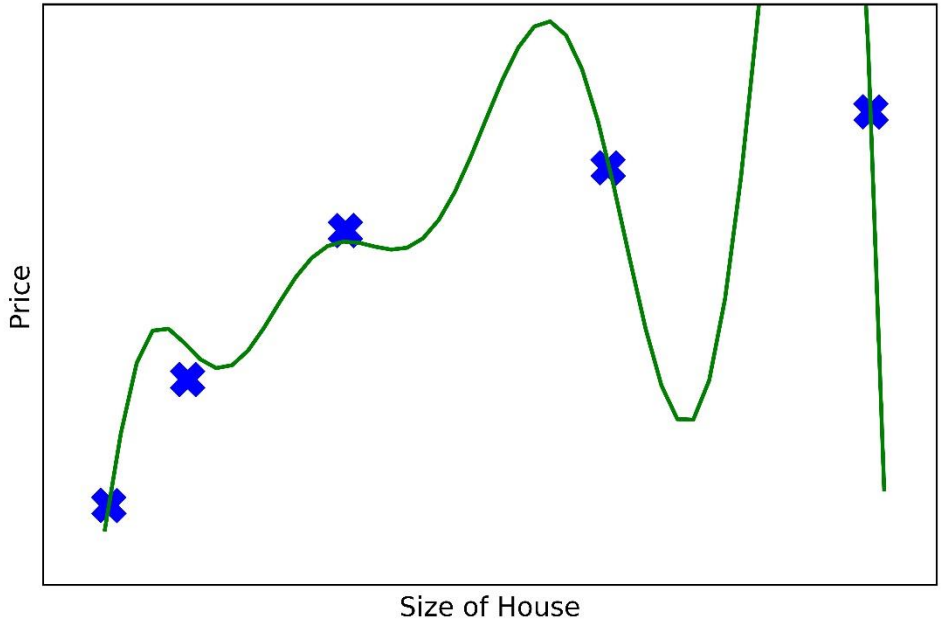
Underfitting

- Model cannot capture the underlying trend of the data

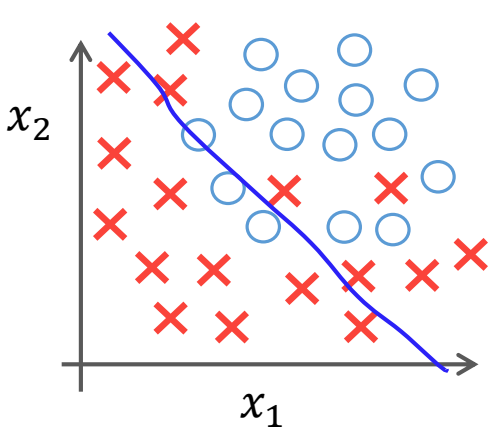


Overfitting

- Model is too complex to capture the true trend
- Model seeks to fit the noise or outlier of the data

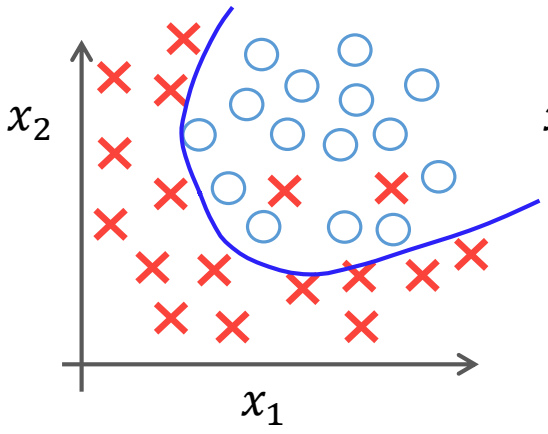


Underfitting vs Overfitting

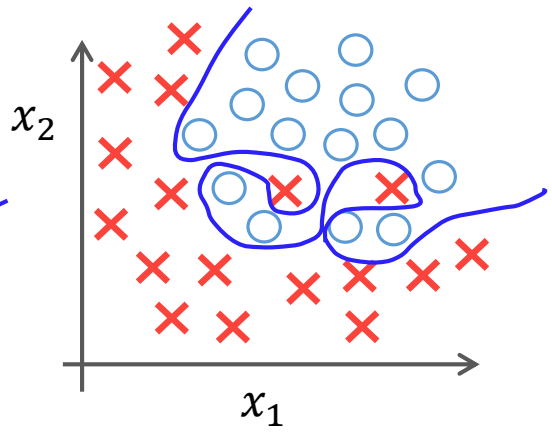


$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \cdots$

- Comprehend from Taylor expansion
  - The more terms, the more complex, the more power

$$f(x) = f(x_0) + \nabla f(x_0)^{\mathrm{T}}(x - x_0) + \frac{1}{2!}(x - x_0)^{\mathrm{T}} \nabla^2 f(x_0)(x - x_0) + \cdots$$

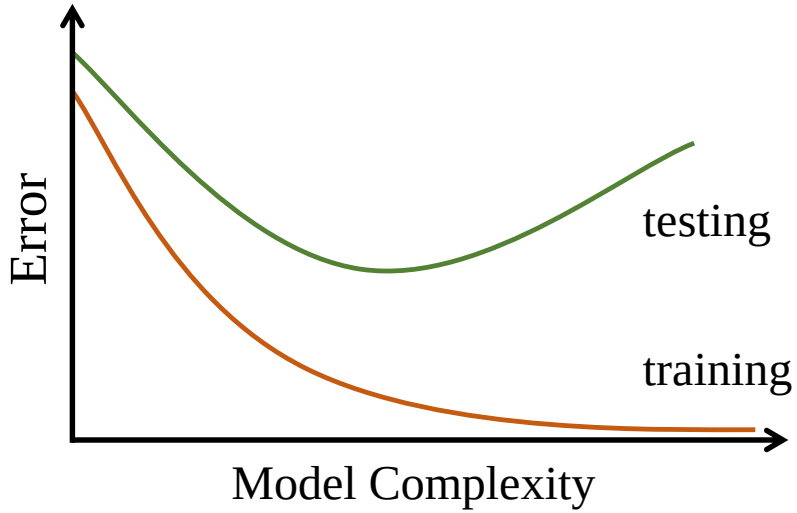
Signs of Underfitting and Overfitting

- How to judge underfitting or overfitting?
  - Underfitting: If the training set's error is relatively large and the generalization error is large
    - Need to increase capacity (complexity of models)
  - Overfitting: If the training set's error is relatively small and the generalization error is large
    - Need to decrease capacity (complexity of models)
    - Or increase training set

Contents

Bias-Variance Trade-off

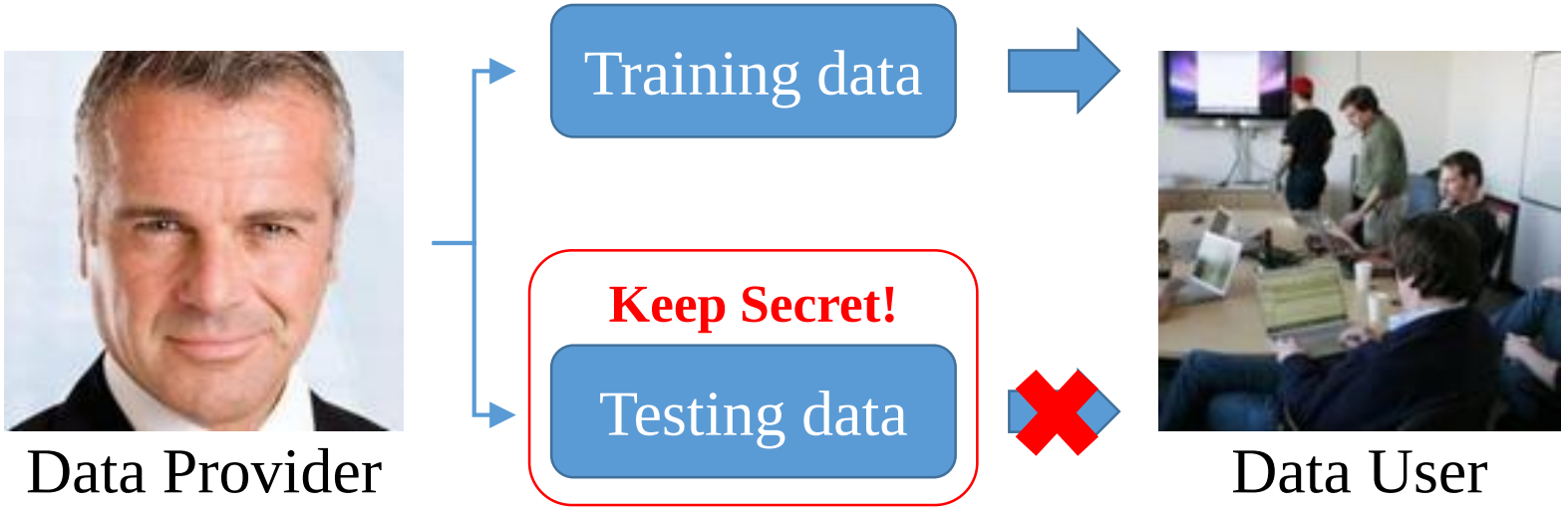
- Complex model
  - Too complex can diminish model's accuracy on future data (called the Bias-Variance Trade-off)
  - Low Bias
    - Model fits well on the training data
  - High Variance
    - Model is more likely to make a wrong prediction



Contents

Data Split

- Simplest idea: Split data into 2 sets
  - Training set: Data used to fit the model
  - Testing set: Data used to evaluate the model

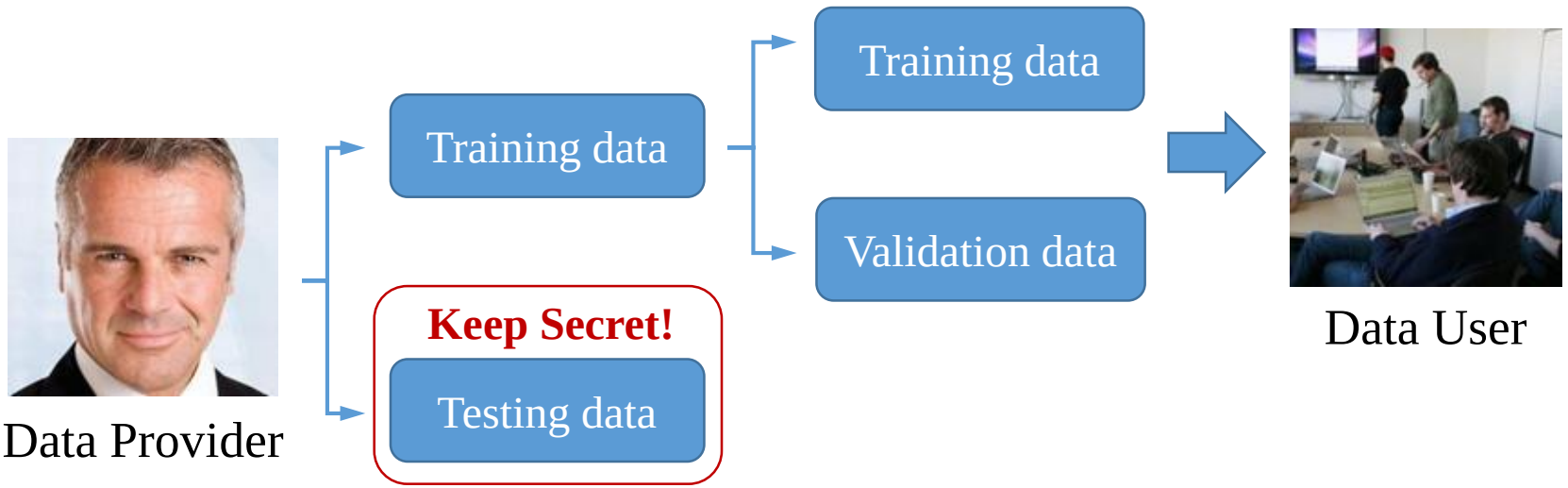


Data Split

- Training set
  - $\approx$ 70% of data, $x^{(i)}$, $y^{(i)}$, total m examples
- Testing set
  - $\approx$ 30% of data, $x_{test}^{(i)}$, $y_{test}^{(i)}$, total m_{test} examples
- It's better to choose examples for training/testing set randomly

Data Split

- Another idea: Split data into 3 sets
  - We still have training and testing sets
  - But additionally we have a validation set to test the performance of our model depending on the parameter



Data Split

- Training set
  - $\approx$ 60% of data, $x^{(i)}$, $y^{(i)}$, total m examples
- Validation set
  - $\approx$ 20% of data, $x_{cv}^{(i)}$, $y_{cv}^{(i)}$, total m_{cv} examples
- Testing set
  - $\approx$ 20% of data, $x_{test}^{(i)}$, $y_{test}^{(i)}$, total m_{test} examples

Validation for Evaluation

- Training error

$$J_{train}(\theta) = \frac{1}{2m}\sum cost\left(x^{(i)}, y^{(i)}\right)$$

- Validation error

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum cost\left(x_{cv}^{(i)}, y_{cv}^{(i)}\right)$$

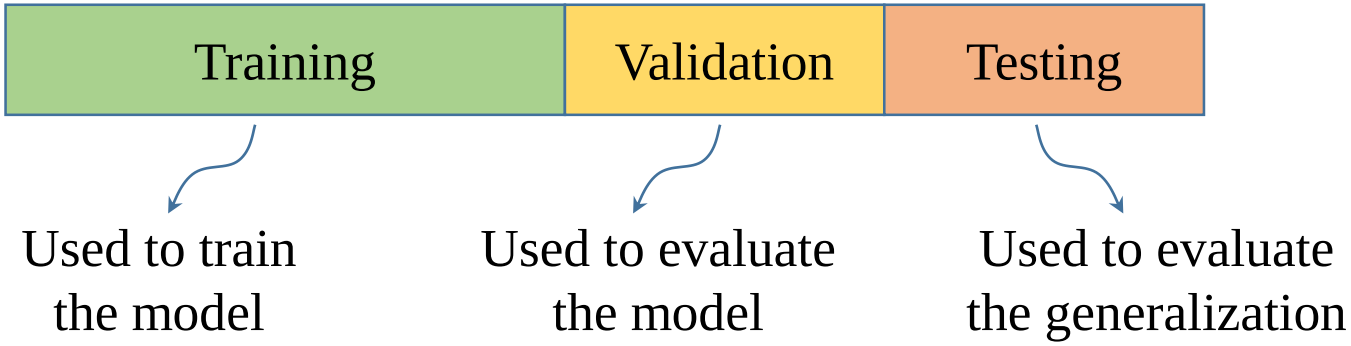
- Testing error

$$J_{test}(\theta) = \frac{1}{2m_{test}}\sum cost\left(x_{test}^{(i)}, y_{test}^{(i)}\right)$$

- For model selection
  - Obtain $\theta^{(1)}, \cdots, \theta^{(d)}$ and select the best (lowest) $J_{cv}\left(\theta^{(i)}\right)$
- For final evaluation
  - Evaluate generalization error $J_{test}\left(\theta^{(i)}\right)$ on testing set

Tuning Learning Parameter

- Use validation set for tuning hyper-parameters
- Use testing set only for final evaluation



Tuning Regularization Parameter λ

- Suppose we are fitting a model with high-order polynomial

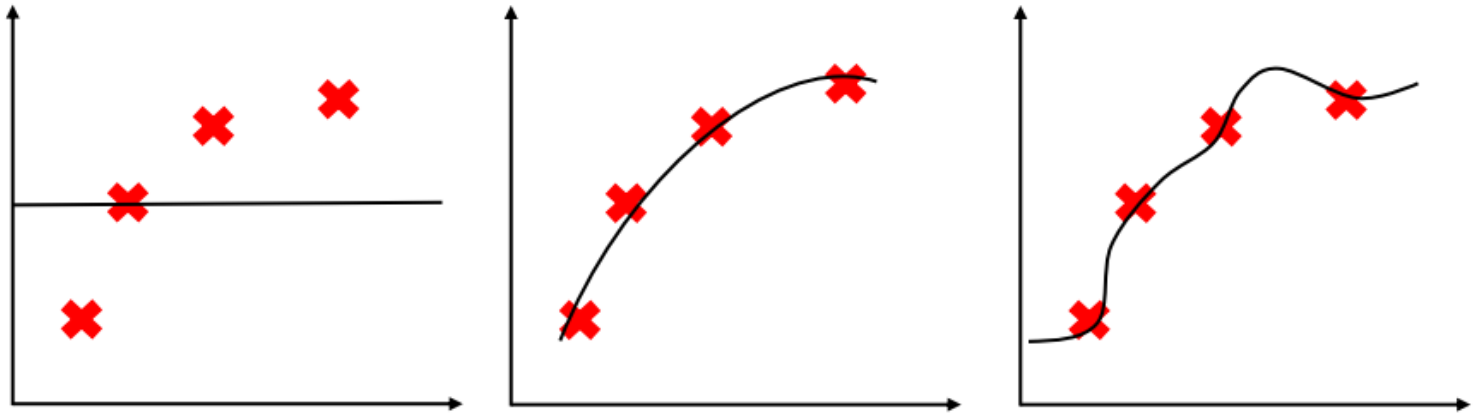
$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_n x^n$$

- To prevent overfitting, we use regularization

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} cost(h_\theta(x_i), y_i) + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

Tuning Regularization Parameter λ

- If λ is too large, all θ are penalized and $\theta_1 \approx \theta_2 \approx \cdots \approx 0$, $h_\theta(x) \approx \theta_0$
- If λ is intermediate, the model fits well
- If λ is too small, the model fits too well, i.e. overfitting



λ is large $\quad$ λ is intermediate $\quad$ λ is small

Tuning Regularization Parameter λ

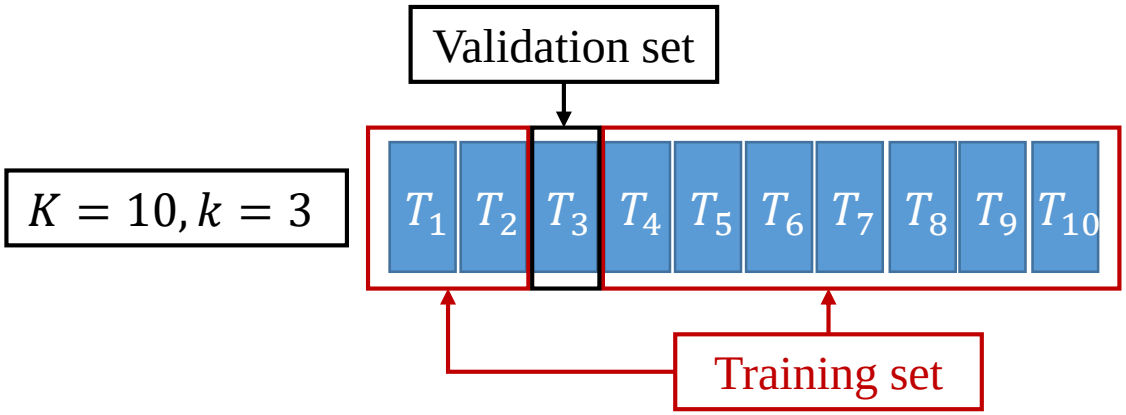
- Choose good λ on validation set
  - Choose a range of possible values for $\lambda(0.02, \cdots, 0.24)$
  - That gives us 12 models with different parameters λ to check
  - For each λ_i:
    - Calculate θ_i
    - Calculate $J_{cv}(\theta_i)$
    - Take λ_i with lowest $J_{cv}(\theta_i)$
  - Finally, we report the test error as $J_{test}(\theta_i)$

K-Fold Cross-validation

- If we want to reduce variability in the data
  - We can use multiple rounds of cross-validation using different partitions
  - And then, average the result over all rounds
- Given a data *S* sampled from the population *D*

K-Fold Cross-validation

- Split data S into K equal disjoint subsets $(T_1, \cdots, T_K)$
- Perform following steps for $k = 1, \cdots, K$
  - Use $R_k = S - T_k$ as the training set
  - Build classifier C_k using R_k
  - Use T_k as the validation set, compute error $Err_k = error(C_k, T_k)$
- Let $Err^{ave} = \frac{1}{K}\sum_{k=1}^{K} Err_k$
  - This is the averaged error rate



Tuning Learning Parameter

- Choosing λ with K-Fold Cross-validation
  - Split your data into training, validation and testing set
  - For every possible value λ, estimate the error rate
  - Select λ with least average error rage Err^{ave}
  - Final evaluation on testing set

Thank You