Model Selection and Regularization

Regularization

Marek Petrik

2/23/2017

Last Time

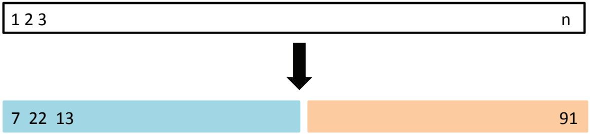
- Successfully using basic machine learning methods
- Problems:
  1. How well is the machine learning method doing
  2. Which method is best for my problem?
  3. How many features (and which ones) to use?
  4. What is the uncertainty in the learned parameters?

Last Time

- Successfully using basic machine learning methods
- Problems:
  1. How well is the machine learning method doing
  2. Which method is best for my problem?
  3. How many features (and which ones) to use?
  4. What is the uncertainty in the learned parameters?
- Methods:
  1. Validation set
  2. Leave one out cross-validation
  3. k-fold cross validation
  4. Bootstrapping

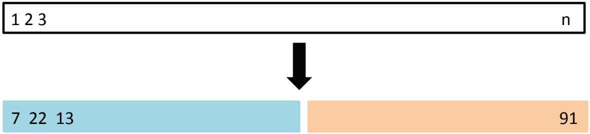
Solution 1: Validation Set

- Just evaluate how well the method works on the test set
- **Randomly** split data to:
    1. Training set: about half of all data
    2. Validation set (AKA hold-out set): remaining half



Solution 1: Validation Set

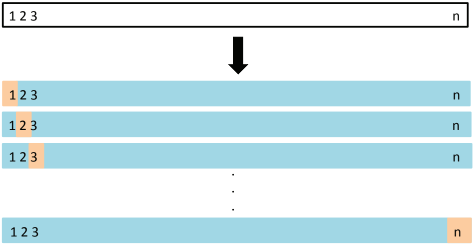
- Just evaluate how well the method works on the test set
- **Randomly** split data to:
  1. Training set: about half of all data
  2. Validation set (AKA hold-out set): remaining half



- Choose the number of features/representation based on minimizing error on **validation set**

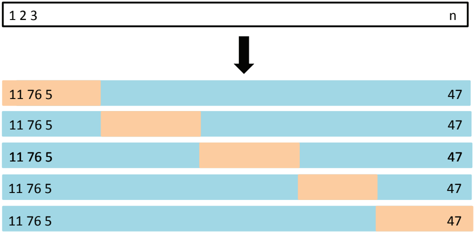
Solution 2: Leave-one-out

- Addresses problems with validation set
- Split the data set into 2 parts:
  1. Training: Size $n-1$
  2. Validation: Size 1
- Repeat n times: to get n learning problems



Solution 3: k-fold Cross-validation

- Hybrid between validation set and LOO
- Split training set into k subsets
  1. Training set: $n - {}^{n}/_{k}$
  2. Test set: ${}^{1}/_{k}n$
- k learning problems



- Cross-validation error:

$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{MSE}_i$$

Limits of Cross-validation

- Successfully using basic machine learning methods
- Cross-validation is not the end of the story

Limits of Cross-validation

- Successfully using basic machine learning methods
- Cross-validation is not the end of the story
- Feasible to test options:
  1. $\text{mpg} = \beta_0 + \beta_1 \text{ power}$
  2. $\text{mpg} = \beta_0 + \beta_1 \text{ power} + \beta_2 \text{ power}^2$
  3. $\text{mpg} = \beta_0 + \beta_1 \text{ power} + \beta_2 \text{ power}^2 + \beta_3 \text{ power}^3$
  4. $\text{mpg} = \beta_0 + \beta_1 \text{ power} + \beta_2 \text{ power}^2 + \beta_3 \text{ power}^3 + \beta_4 \text{ power}^4$
  5. $\ldots$

Limits of Cross-validation

- Successfully using basic machine learning methods
- Cross-validation is not the end of the story
- Feasible to test options:
  1. $\text{mpg} = \beta_0 + \beta_1 \, \text{power}$
  2. $\text{mpg} = \beta_0 + \beta_1 \, \text{power} + \beta_2 \, \text{power}^2$
  3. $\text{mpg} = \beta_0 + \beta_1 \, \text{power} + \beta_2 \, \text{power}^2 + \beta_3 \, \text{power}^3$
  4. $\text{mpg} = \beta_0 + \beta_1 \, \text{power} + \beta_2 \, \text{power}^2 + \beta_3 \, \text{power}^3 + \beta_4 \, \text{power}^4$
  5. ...
- This is just one feature!
- What is we add displacement, weight, topspeed, wheelsize, ...?

Limits of Cross-validation

- Successfully using basic machine learning methods
- Cross-validation is not the end of the story
- Feasible to test options:
  1. $\text{mpg} = \beta_0 + \beta_1\,\text{power}$
  2. $\text{mpg} = \beta_0 + \beta_1\,\text{power} + \beta_2\,\text{power}^2$
  3. $\text{mpg} = \beta_0 + \beta_1\,\text{power} + \beta_2\,\text{power}^2 + \beta_3\,\text{power}^3$
  4. $\text{mpg} = \beta_0 + \beta_1\,\text{power} + \beta_2\,\text{power}^2 + \beta_3\,\text{power}^3 + \beta_4\,\text{power}^4$
  5. ...
- This is just one feature!
- What is we add displacement, weight, topspeed, wheelsize, ...?
- Exponential growth!

Today

- How to choose the right features if we have (too) many options

- Methods:
  1. Subset selection
  2. Regularization (shrinkage)
  3. Dimensionality reduction (next class)

Importance of Feature Engineering

- MNIST handwritten digit recognition (see R notebook)
- Example results: see `http://yann.lecun.com/exdb/mnist/`

Why Few Features

1. **Improve prediction accuracy**: reduce overfitting

2. **Improve interpretability**: small number of coefficients are easier to understand

Best Subset Selection

- Want to find a subset of p features
- The subset should be small and predict well
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

Algorithm 1: Best Subset Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
    Fit all $\binom{p}{k}$ models that contain k features ;
    $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Achieving Scalability

- Complexity of *Best Subset Selection*?

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?
- $O(2^p)$!

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?
- $O(2^p)$!

- Heuristic approaches:
  1. **Stepwise selection**: Solve the problem approximately: greedy
  2. **Regularization**: Solve a different (easier) problem: relaxation

Forward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively add more features
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

Algorithm 2: Forward Stepwise Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features) ;
for $k = 0, 1, 2, \ldots, p-1$ **do**
  Fit all $p-k$ models that augment $\mathcal{M}_k$ by one new feature ;
  $\mathcal{M}_{k+1} \leftarrow$ best of $p-k$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Forward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively add more features
- Example: credit $\sim$ rating + income + student + limit

Algorithm 3: Forward Stepwise Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features) ;
for $k = 0, 1, 2, \ldots, p-1$ **do**
    Fit all $p-k$ models that augment $\mathcal{M}_k$ by one new feature ;
    $\mathcal{M}_{k+1} \leftarrow$ best of $p-k$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

- Complexity?

Forward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively add more features
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

---

Algorithm 4: Forward Stepwise Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features) ;
for $k = 0, 1, 2, \ldots, p-1$ **do**
    Fit all $p-k$ models that augment $\mathcal{M}_k$ by one new feature ;
    $\mathcal{M}_{k+1} \leftarrow$ best of $p-k$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

---

- Complexity? $O(p^2)$

Backward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively remove features
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

Algorithm 5: Forward Stepwise Selection

$\mathcal{M}_p \leftarrow$ *full model* (all features) ;
for $k = p, p-1, \ldots, 1$ **do**
    Fit all k models that remove one feature from $\mathcal{M}_k$;
    $\mathcal{M}_{k-1} \leftarrow$ best of k models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Backward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively remove features
- Example: credit $\sim$ rating + income + student + limit

Algorithm 6: Forward Stepwise Selection

$\mathcal{M}_p \leftarrow$ *full model* (all features) ;
for $k = p, p-1, \ldots, 1$ **do**
    Fit all k models that remove one feature from $\mathcal{M}_k$;
    $\mathcal{M}_{k-1} \leftarrow$ best of k models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

- Complexity?

Backward Stepwise Selection

- Greedy approximation of *Best Subset Selection*
- Iteratively remove features
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

Algorithm 7: Forward Stepwise Selection

$\mathcal{M}_p \leftarrow$ *full model* (all features) ;
for $k = p, p-1, \ldots, 1$ **do**
    Fit all k models that remove one feature from $\mathcal{M}_k$;
    $\mathcal{M}_{k-1} \leftarrow$ best of k models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

- Complexity? $O(p^2)$

Which Metric to Use?

Algorithm 8: Best Subset Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Which Metric to Use?

Algorithm 9: Best Subset Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
    Fit all $\binom{p}{k}$ models that contain k features ;
    $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

1. **Direct error estimate**: Cross validation, precise but computationally intensive

Which Metric to Use?

Algorithm 10: Best Subset Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

1. **Direct error estimate**: Cross validation, precise but computationally intensive
2. **Indirect error estimate**: Mellow's C_p:

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2) \text{ where } \hat{\sigma}^2 \approx \mathrm{Var}[\epsilon]$$

Akaike information criterion, BIC, and many others. Theoretical foundations

Which Metric to Use?

Algorithm 11: Best Subset Selection

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

1. **Direct error estimate**: Cross validation, precise but computationally intensive
2. **Indirect error estimate**: Mellow's C_p:

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2) \text{ where } \hat{\sigma}^2 \approx \mathrm{Var}[\epsilon]$$

   Akaike information criterion, BIC, and many others. Theoretical foundations
3. **Interpretability Penalty**: What is the cost of extra features

Regularization

1. **Stepwise selection**: Solve the problem approximately

2. **Regularization**: Solve a different (easier) problem: relaxation
   - Solve a machine learning problem, but penalize solutions that use "too much" of the features

Recall: Linear Regression

- With one feature:

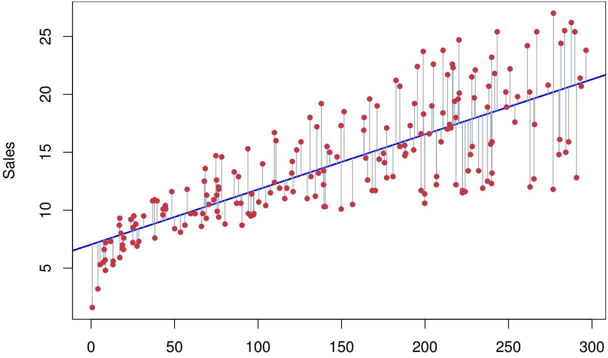
$$Y \approx \beta_0 + \beta_1 X \qquad Y = \beta_0 + \beta_1 X + \epsilon$$

- Prediction:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

- Errors (y_i are true values):

$$e_i = y_i - \hat{y}_i$$



Recall: Solving Linear Regression

- Errors (y_i are true values):

$$e_i = y_i - \hat{y}_i$$

- Residual Sum of Squares

$$\mathrm{RSS} = e_1^2 + e_2^2 + e_3^2 + \cdots + e_n^2 = \sum_{i=1}^{n} e_i^2$$

- Equivalently:

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2$$

- **Minimize RSS** (for p features, x_{ij}: ith sample, jth feature)

$$\min_{\beta} \mathrm{RSS}(\beta) = \min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

Regularization

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j \beta_j^2 =$$

$$\min_\beta \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_j \beta_j^2$$

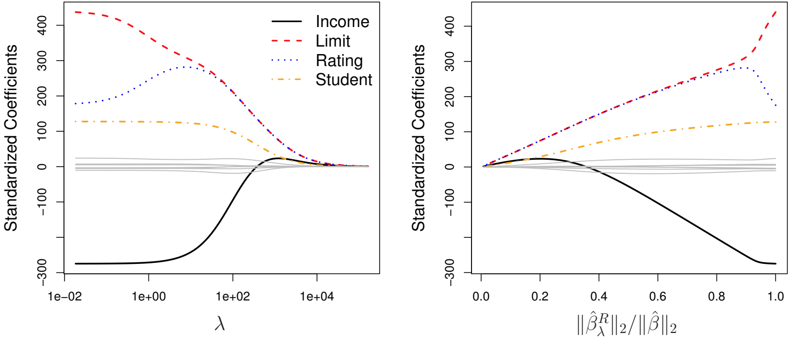
- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j |\beta_j| =$$

$$\min_\beta \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_j |\beta_j|$$

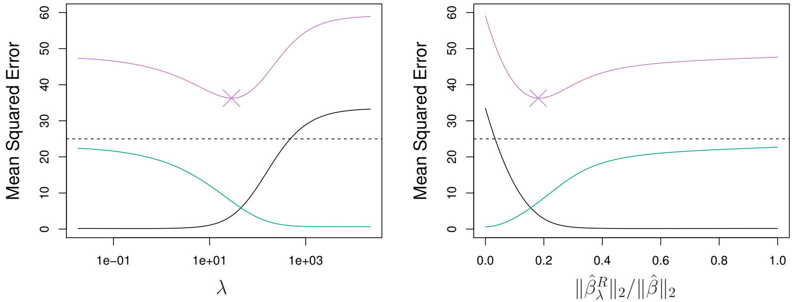
- Approximations to the ℓ_0 solution

Ridge Regression: Coefficient Values



Why Ridge Regression Works

- Bias-variance trade-off
- Increasing λ increases bias



purple: test MSE, black: bias, green: variance

Regularization

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_{\beta} \mathrm{RSS}(\beta) + \lambda \sum_{j} \beta_j^2 =$$

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j} \beta_j^2$$

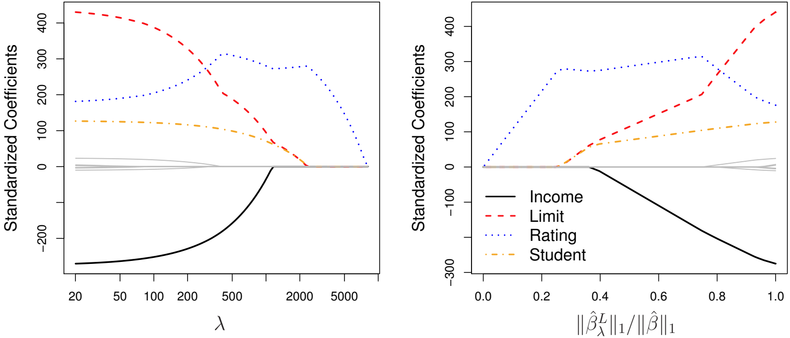
- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_{\beta} \mathrm{RSS}(\beta) + \lambda \sum_{j} |\beta_j| =$$

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j} |\beta_j|$$

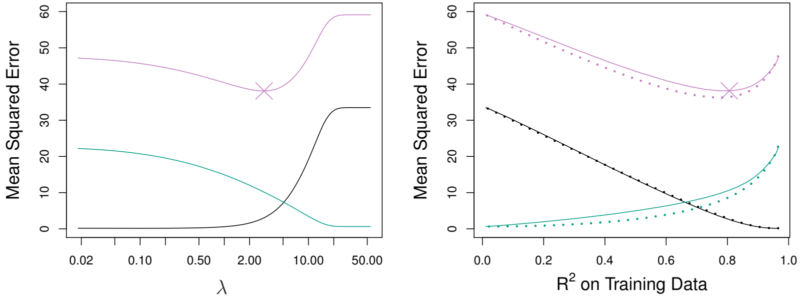
- Approximations to the ℓ_0 solution

Lasso: Coefficient Values



Why Lasso Works

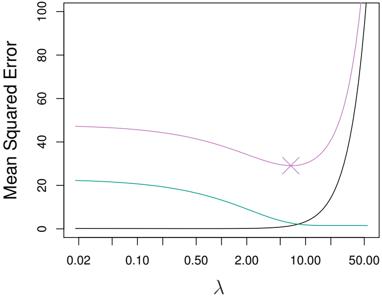
- Bias-variance trade-off
- Increasing λ increases bias
- Example: all features relevant



purple: test MSE, black: bias, green: variance
dotted (ridge)

Why Lasso Works

- Bias-variance trade-off
- Increasing λ increases bias
- Example: some features relevant



purple: test MSE, black: bias, green: variance

dotted (ridge)

Regularization

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j \beta_j^2 =$$

$$\min_\beta \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_j \beta_j^2$$

- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j |\beta_j| =$$

$$\min_\beta \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_j |\beta_j|$$

- Approximations to the ℓ_0 solution

Regularization: Constrained Formulation

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \text{ subject to } \sum_j \beta_j^2 \leq s$$

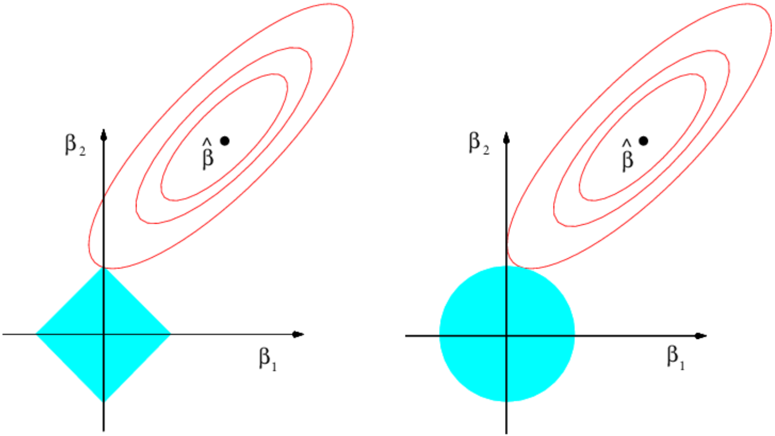
- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \text{ subject to } \sum_j |\beta_j| \leq s$$

- Approximations to the ℓ_0 solution

Lasso Solutions are Sparse

Constrained Lasso (left) vs Constrained Ridge Regression (right)



Constraints are blue, red are contours of the objective

How to Choose λ?

How to Choose λ?

- Cross-validation