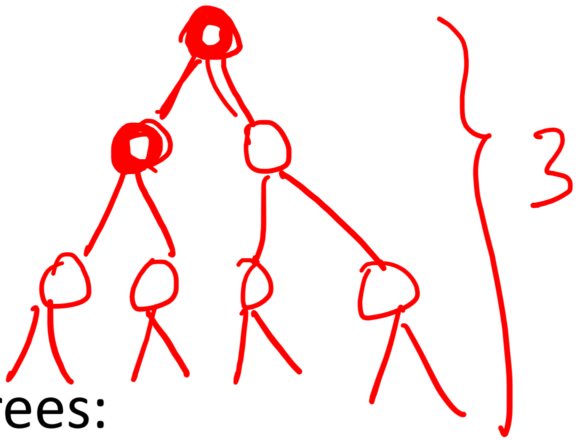
10-301/601: Introduction to Machine Learning Lecture 5 –Model Selection

Henry Chai & Matt Gormley & Hoda Heidari

9/13/23

Model Selection



- Terminology:
  - **Model** ≈ the hypothesis space in which the learning algorithm searches for a classifier to return
  - **Parameters** = numeric values or structure selected by the learning algorithm
  - **Hyperparameters** = tunable aspects of the model that need to be specified before learning can happen, set outside of the training procedure

- Example – Decision Trees:
  - Model = the set of all possible trees, potentially limited by some hyperparameter, e.g., max depth (see below)
  - Parameters = structure of a specific tree, i.e., the order in which features are split on
  - Hyperparameters = max depth, splitting criterion, etc…

Model Selection

- Terminology:
  - **Model** $\approx$ the hypothesis space in which the learning algorithm searches for a classifier to return
  - **Parameters** = numeric values or structure selected by the learning algorithm
  - **Hyperparameters** = tunable aspects of the model that need to be specified before learning can happen, set outside of the training procedure

- Example – kNN:
  - Model = the set of all possible nearest neighbor classifiers
  - Parameters = none! kNN is a non-parametric model
  - Hyperparameters = k

Parametric vs. Nonparametric Models

- Parametric models (e.g., decision trees)
  - Have a parametrized form with parameters learned from training data
  - Can discard training data after parameters have been learned.
  - Cannot exactly model every target function
- Nonparametric models (e.g., kNN)
  - Have no parameters that are learned from training data; can still have *hyperparameters*
  - Training data generally needs to be stored in order to make predictions
  - Can recover any target function given enough data

Model Selection vs Hyperparameter Optimization

- Hyperparameter optimization can be considered a special case of model selection
  - Changing the hyperparameters changes the hypothesis space or the set of potential classifiers returned by the learning algorithm
- Deciding between a decision tree and kNN (model selection) vs. selecting a value of k for kNN (hyperparameter optimization)
- Both model selection and hyperparameter optimization happen outside the regular training procedure

Setting k

- When $k = 1$:
  - many, complicated decision boundaries
  - liable to overfit
- When $k = N$:
  - no decision boundaries; always predicts the most common label in the training data (majority vote)
  - liable to underfit
- k controls the complexity of the hypothesis set $\Longrightarrow$ k affects how well the learned hypothesis will generalize

Setting k

- Theorem:
  - If k is some function of N s.t. $k(N) \to \infty$ and $\frac{k(N)}{N} \to 0$ as $N \to \infty$ …
  - … then (under certain assumptions) the true error of a kNN model $\to$ the Bayes error rate
- Practical heuristics:
  - $k = \lfloor \sqrt{N} \rfloor$
  - $k = 3$
- Perform model selection!

Model Selection with Test Sets?

- Given $\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{test}$, suppose we have multiple candidate models:

$$\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_M$$

- Learn a classifier from each model using only $\mathcal{D}_{train}$:

$$h_1 \in \mathcal{H}_1, h_2 \in \mathcal{H}_2, \dots, h_M \in \mathcal{H}_M$$

- Evaluate each one using $\mathcal{D}_{test}$ and choose the one with lowest test error:

$$\hat{m} = \underset{m \in \{1,\dots,M\}}{\operatorname{argmin}} \; err(h_m, \mathcal{D}_{test})$$

- Is $err(h_{\hat{m}}, \mathcal{D}_{test})$ a good estimate of $err(h_{\hat{m}})$?

Model Selection with Validation Sets

- Given $\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$, suppose we have multiple candidate models:

$$\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_M$$

- Learn a classifier from each model using only $\mathcal{D}_{train}$:

$$h_1 \in \mathcal{H}_1, h_2 \in \mathcal{H}_2, \ldots, h_M \in \mathcal{H}_M$$

- Evaluate each one using $\mathcal{D}_{val}$ and choose the one with lowest *validation* error:

$$\hat{m} = \underset{m \in \{1,\ldots,M\}}{\operatorname{argmin}} \; err(h_m, \mathcal{D}_{val})$$

- Now $err(h_{\hat{m}}, \mathcal{D}_{test})$ is a good estimate of $err(h_{\hat{m}})$!

Hyperparameter Optimization with Validation Sets

- Given $\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$, suppose we have multiple candidate hyperparameter settings:

$$\theta_1, \theta_2, \dots, \theta_M$$

- Learn a classifier for each setting using only $\mathcal{D}_{train}$:

$$h_1, h_2, \dots, h_M$$

- Evaluate each one using $\mathcal{D}_{val}$ and choose the one with lowest *validation* error:

$$\hat{m} = \underset{m \in \{1,\dots,M\}}{\operatorname{argmin}} \; err(h_m, \mathcal{D}_{val})$$

- Now $err(h_{\hat{m}}, \mathcal{D}_{test})$ is a good estimate of $err(h_{\hat{m}})$!

Setting k for kNN with Validation Sets

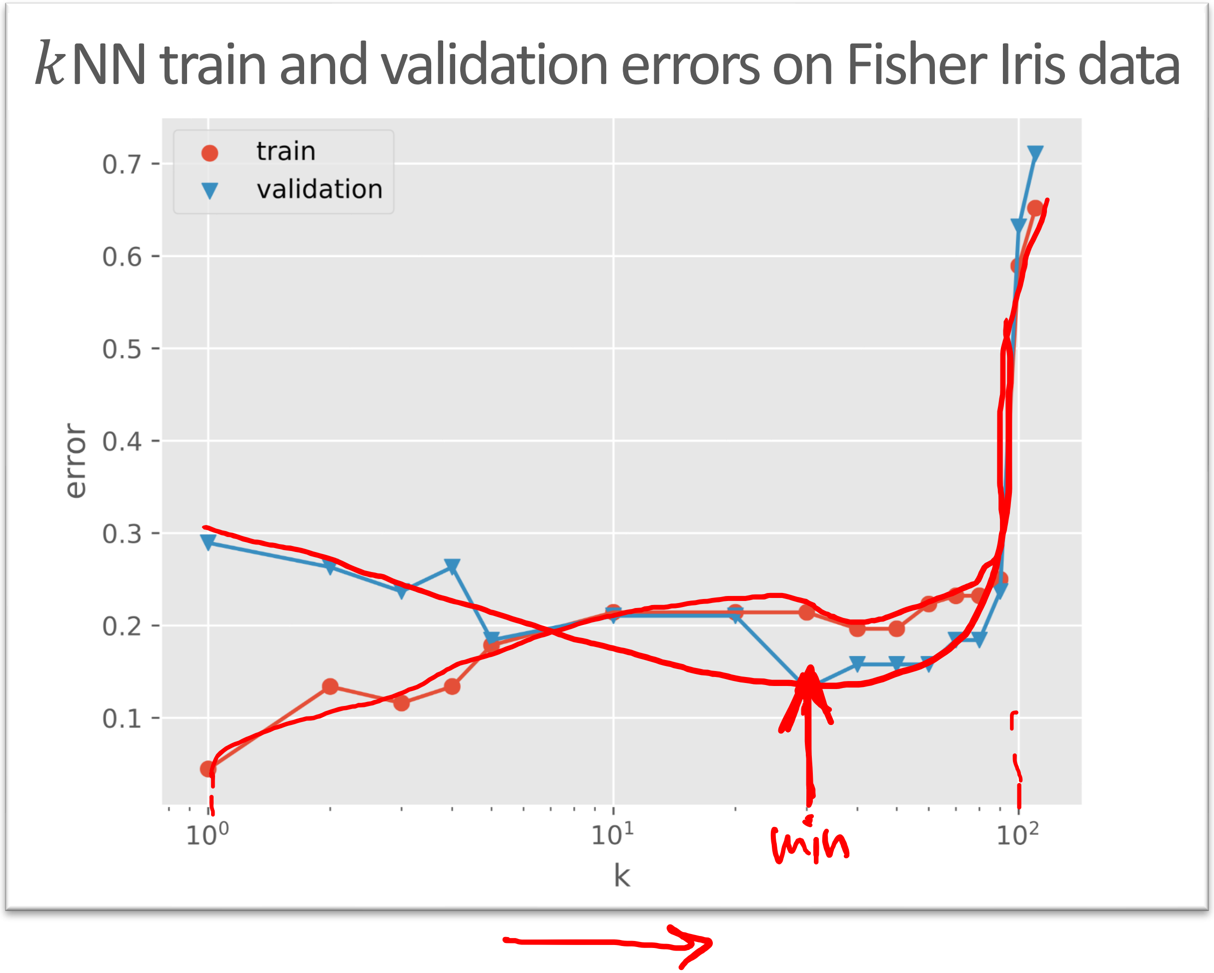


Figure courtesy of Matt Gormley

How should we partition our dataset?

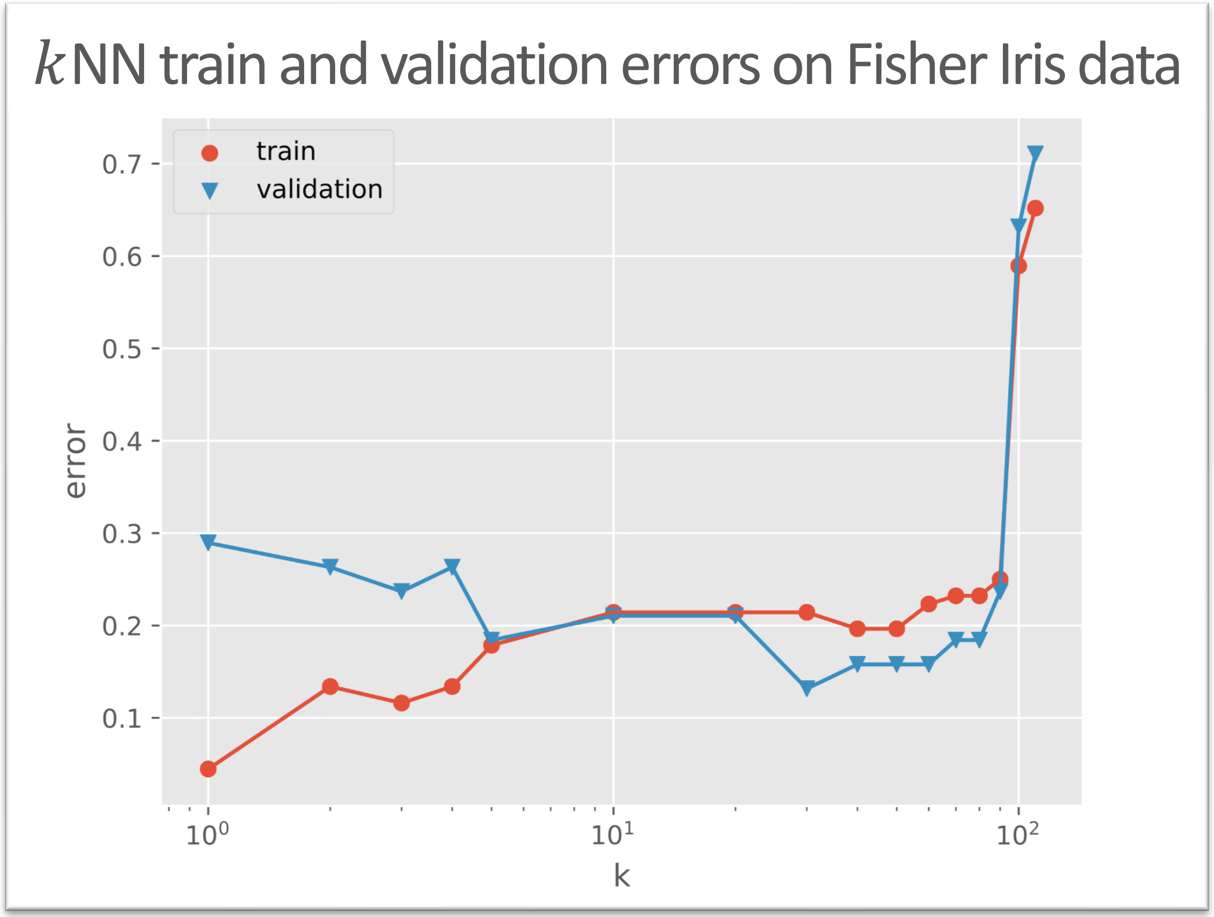
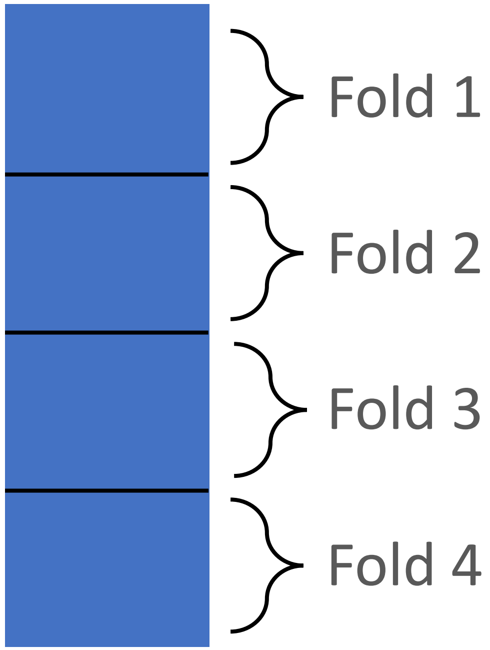


Figure courtesy of Matt Gormley

K-fold cross-validation

- Given $\mathcal{D}$, split $\mathcal{D}$ into K equally sized datasets or folds: $\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K$
- Use each one as a validation set once:

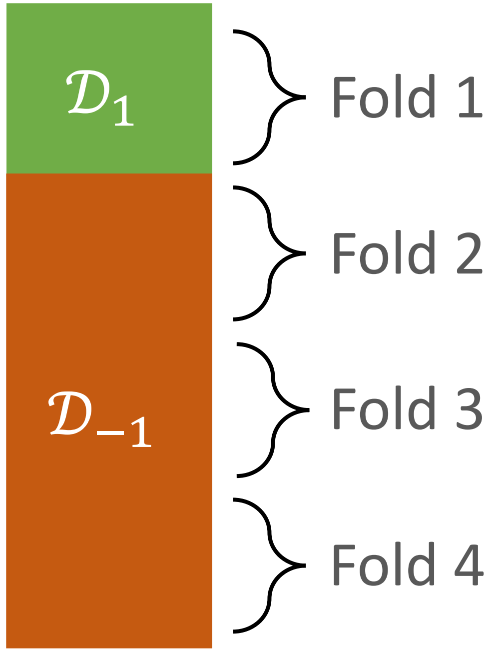


- Let h_{-i} be the classifier learned using $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$ (all folds other than $\mathcal{D}_i$) and let $e_i = err(h_{-i}, \mathcal{D}_i)$
- The K-fold cross validation error is

$$err_{cv_K} = \frac{1}{K} \sum_{i=1}^{K} e_i$$

K-fold cross-validation

- Given $\mathcal{D}$, split $\mathcal{D}$ into K equally sized datasets or folds: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$
- Use each one as a validation set once:

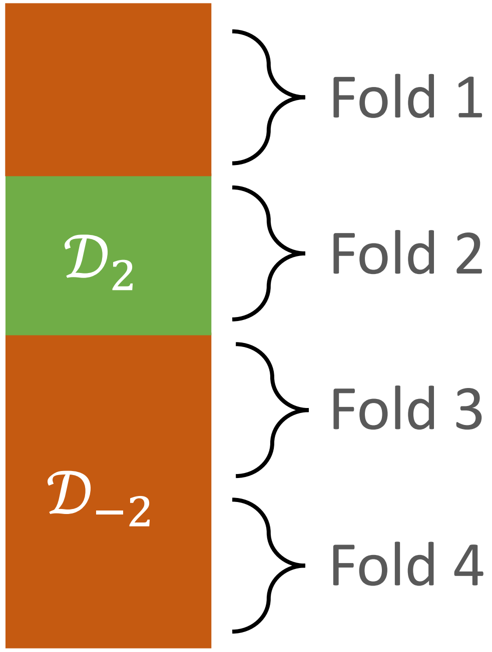


- Let h_{-i} be the classifier learned using $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$ (all folds other than $\mathcal{D}_i$) and let $e_i = err(h_{-i}, \mathcal{D}_i)$
- The K-fold cross validation error is

$$err_{cv_K} = \frac{1}{K} \sum_{i=1}^{K} e_i$$

K-fold cross-validation

- Given $\mathcal{D}$, split $\mathcal{D}$ into K equally sized datasets or folds: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$
- Use each one as a validation set once:

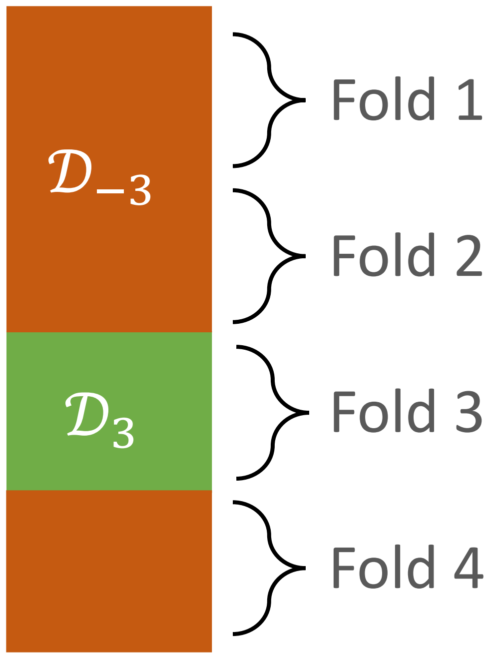


- Let h_{-i} be the classifier learned using $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$ (all folds other than $\mathcal{D}_i$) and let $e_i = err(h_{-i}, \mathcal{D}_i)$
- The K-fold cross validation error is

$$err_{cv_K} = \frac{1}{K} \sum_{i=1}^{K} e_i$$

K-fold cross-validation

- Given $\mathcal{D}$, split $\mathcal{D}$ into K equally sized datasets or folds: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$
- Use each one as a validation set once:

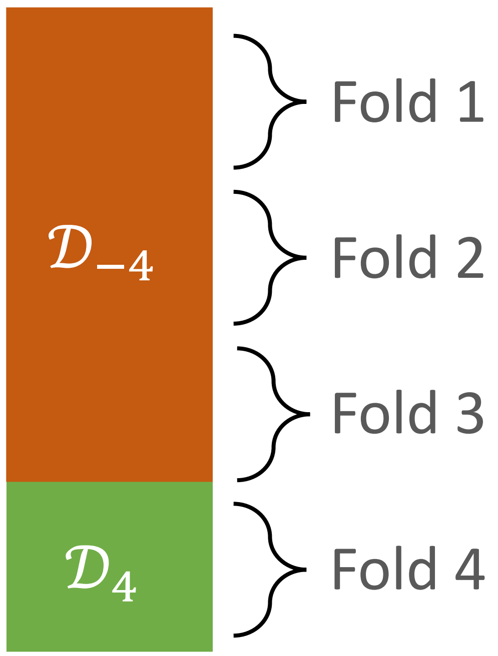


- Let h_{-i} be the classifier learned using $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$ (all folds other than $\mathcal{D}_i$) and let $e_i = err(h_{-i}, \mathcal{D}_i)$
- The K-fold cross validation error is

$$err_{cv_K} = \frac{1}{K} \sum_{i=1}^{K} e_i$$

K-fold cross-validation

- Given $\mathcal{D}$, split $\mathcal{D}$ into K equally sized datasets or folds: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$
- Use each one as a validation set once:

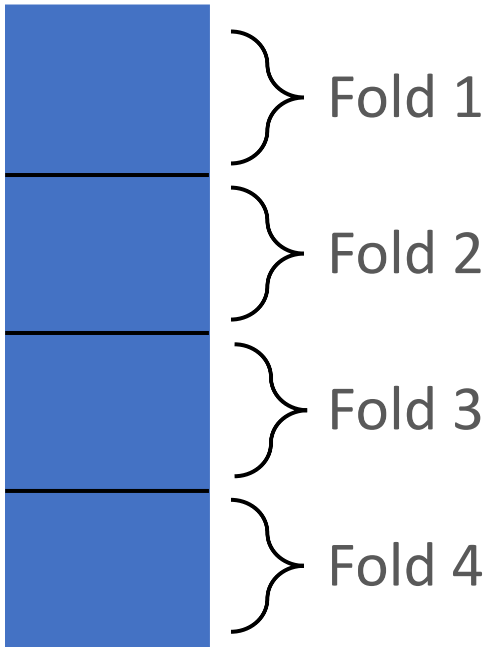


- Let h_{-i} be the classifier learned using $\mathcal{D}_{-i} = \mathcal{D} \backslash \mathcal{D}_i$ (all folds other than $\mathcal{D}_i$) and let $e_i = err(h_{-i}, \mathcal{D}_i)$
- The K-fold cross validation error is

$$err_{cv_K} = \frac{1}{K} \sum_{i=1}^{K} e_i$$

K-fold cross-validation

- Special case when $K = N$: *Leave-one-out cross-validation*
- Choosing between m candidates requires training mK times



Summary

	Input	Output
Training	• training dataset • hyperparameters	• best model parameters
Hyperparameter Optimization	• training dataset • validation dataset	• best hyperparameters
Cross-Validation	• training dataset • validation dataset	• cross-validation error
Testing	• test dataset • classifier	• test error