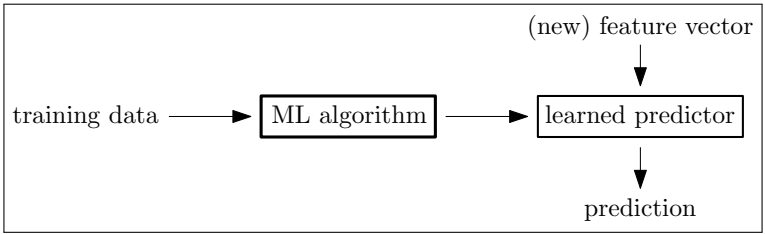
Model selection

COMS 4721 Spring 2022
Daniel Hsu

Hyperparameters and Model Selection

Supervised learning framework



- Basic supervised learning framework assumes that there is a single ML algorithm that we will use
- In practice, we may want to consider many different ML algorithms

Hyperparameters and “model selection”

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)

Hyperparameters and “model selection”

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)
  - E.g., choice of stopping criterion in greedy heuristic for decision trees

Hyperparameters and "model selection"

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)
  - E.g., choice of stopping criterion in greedy heuristic for decision trees
- Variations in feature representations & predictor templates also regarded as hyperparameters

Hyperparameters and "model selection"

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)
  - E.g., choice of stopping criterion in greedy heuristic for decision trees
- Variations in feature representations & predictor templates also regarded as hyperparameters
  - E.g., allowed forms of predicates in decision trees

Hyperparameters and "model selection"

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)
  - E.g., choice of stopping criterion in greedy heuristic for decision trees
- Variations in feature representations & predictor templates also regarded as hyperparameters
  - E.g., allowed forms of predicates in decision trees

Question: Is there a data-driven way to set hyperparameters?

Hyperparameters and "model selection"

- Many ML algorithms have **hyperparameters:**
  parameters of ML algorithm ($\neq$ parameters of predictor learned by ML algorithm)
  - E.g., choice of stopping criterion in greedy heuristic for decision trees
- Variations in feature representations & predictor templates also regarded as hyperparameters
  - E.g., allowed forms of predicates in decision trees

Question: Is there a data-driven way to set hyperparameters?

Answer: "**Model selection**" (a.k.a. **hyperparameter tuning**)

- Pretend ML algorithm with different hyperparameter settings are *different ML algorithms*

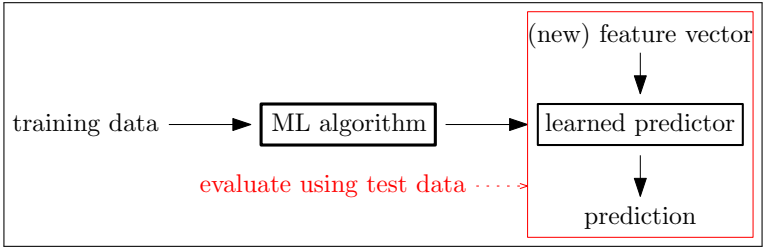
$$A_1,\ A_2,\ A_3,\ \ldots$$

- Given the same input data, they would learn some (possibly different) predictors

$$f_1,\ f_2,\ f_3,\ \ldots$$

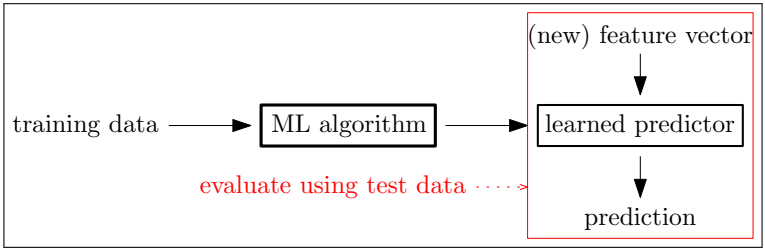
- How do we decide which ML algorithm to use?

Problematic approach to model selection



Evaluation of learned predictor is carried out on **test data**

Problematic approach to model selection

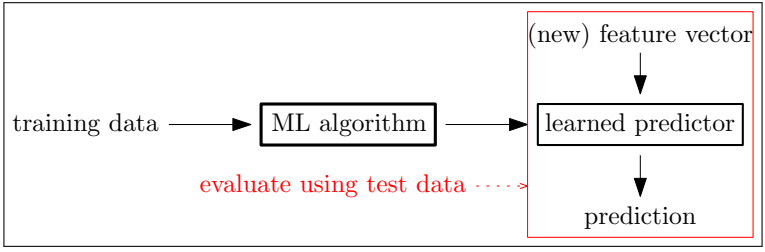


Evaluation of learned predictor is carried out on **test data**

A problematic approach:

1. Run ML algorithms $A_1, A_2, A_3, \ldots$ on training data to get predictors $f_1, f_2, f_3, \ldots$
2. Evaluate $f_1, f_2, f_3, \ldots$ on test data, and return the best predictor (e.g., lowest test error rate)

Problematic approach to model selection



Evaluation of learned predictor is carried out on **test data**

A problematic approach:

1. Run ML algorithms $A_1, A_2, A_3, \ldots$ on training data to get predictors $f_1, f_2, f_3, \ldots$
2. Evaluate $f_1, f_2, f_3, \ldots$ on test data, and return the best predictor (e.g., lowest test error rate)

Flaw: The "test data" are being used for training

Cross Validation

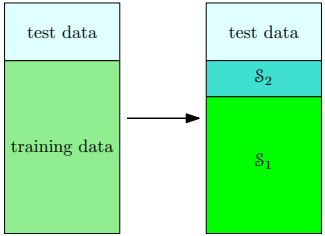
Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

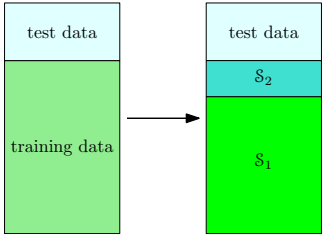
1. Given training data $\mathcal{S}$, split into two parts, $\mathcal{S}_1$ and $\mathcal{S}_2$ in some manner (e.g., randomly)



Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

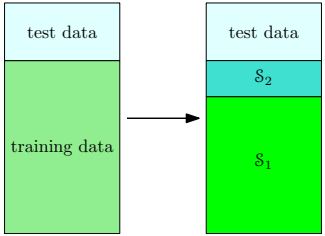
1. Given training data $\mathcal{S}$, split into two parts, $\mathcal{S}_1$ and $\mathcal{S}_2$ in some manner (e.g., randomly)
2. Run ML algorithms $A_1, A_2, A_3, \ldots$ on $\mathcal{S}_1$ to get predictors $f_1, f_2, f_3, \ldots$



Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

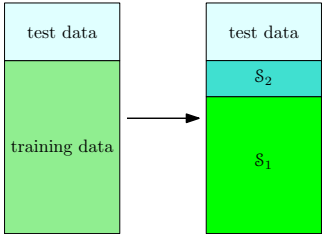
1. Given training data $\mathcal{S}$, split into two parts, $\mathcal{S}_1$ and $\mathcal{S}_2$ in some manner (e.g., randomly)
2. Run ML algorithms $A_1, A_2, A_3, \ldots$ on $\mathcal{S}_1$ to get predictors $f_1, f_2, f_3, \ldots$
3. Evaluate $f_1, f_2, f_3, \ldots$ on $\mathcal{S}_2$ (e.g., in terms of error rate on $\mathcal{S}_2$)
   - Let $i^* \in \{1, 2, \ldots\}$ be the index of the best predictor as above



Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

1. Given training data $\mathcal{S}$, split into two parts, $\mathcal{S}_1$ and $\mathcal{S}_2$ in some manner (e.g., randomly)
2. Run ML algorithms $A_1, A_2, A_3, \ldots$ on $\mathcal{S}_1$ to get predictors $f_1, f_2, f_3, \ldots$
3. Evaluate $f_1, f_2, f_3, \ldots$ on $\mathcal{S}_2$ (e.g., in terms of error rate on $\mathcal{S}_2$)
   - Let $i^* \in \{1, 2, \ldots\}$ be the index of the best predictor as above
4. Return A_{i^*} as the selected ML algorithm

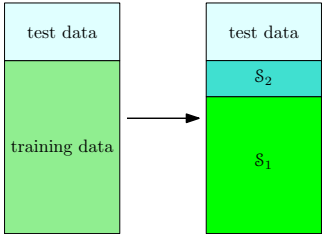


Model selection by cross validation ("hold-out method")

Cross validation: model selection method that simulates training/testing using only training data

1. Given training data $\mathcal{S}$, split into two parts, $\mathcal{S}_1$ and $\mathcal{S}_2$ in some manner (e.g., randomly)
2. Run ML algorithms $A_1, A_2, A_3, \ldots$ on $\mathcal{S}_1$ to get predictors $f_1, f_2, f_3, \ldots$
3. Evaluate $f_1, f_2, f_3, \ldots$ on $\mathcal{S}_2$ (e.g., in terms of error rate on $\mathcal{S}_2$)
   - Let $i^* \in \{1, 2, \ldots\}$ be the index of the best predictor as above
4. Return A_{i^*} as the selected ML algorithm

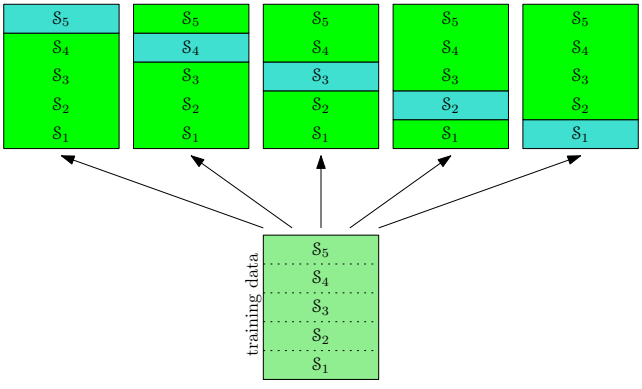
(Final step: Run A_{i^*} on the training data to get final predictor f, which can be evaluated on test data)



Variants of cross validation: K-fold cross validation

K-fold cross validation:

- Each A_i is run and evaluated K times
- Select the A_i with best average evaluation



5-fold cross validation

Variants of cross validation: Leave-one-out cross validation

Leave-one-out cross validation:

- K-fold cross validation with $K = n$
- Pro: A_i is always run on $n - 1$ of the training data
- Con: Running each algorithm n times can be expensive

Domain-specific cross validation

Consider the training/testing setup that makes sense for the application

Domain-specific cross validation

Consider the training/testing setup that makes sense for the application

Example: Time-ordered data

- Suppose there is reason to believe that examples from the "past" are not interchangeable with examples from the "future"

Domain-specific cross validation

Consider the training/testing setup that makes sense for the application

Example: Time-ordered data

- Suppose there is reason to believe that examples from the "past" are not interchangeable with examples from the "future"
  - I.e., only makes sense to train on examples from "past", and test on examples from "future"

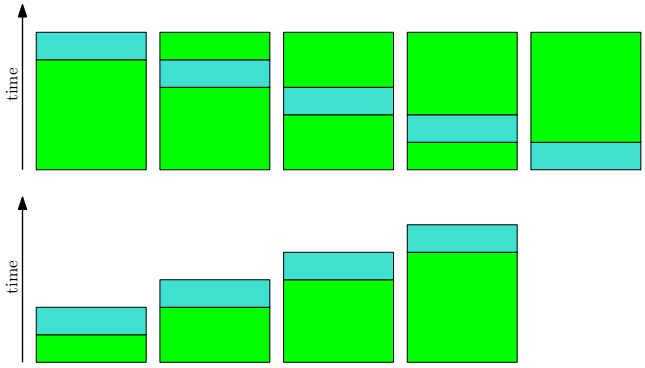
Domain-specific cross validation

Consider the training/testing setup that makes sense for the application

Example: Time-ordered data

- Suppose there is reason to believe that examples from the "past" are not interchangeable with examples from the "future"
  - I.e., only makes sense to train on examples from "past", and test on examples from "future"

Which form of cross validation makes sense?



Discussion

For model selection, cross validation simulates training/testing **using only training data**

Discussion

For model selection, cross validation simulates training/testing **using only training data**

- Useful whenever ML algorithm has hyperparameters (pretty much all of them)

Discussion

For model selection, cross validation simulates training/testing **using only training data**

- Useful whenever ML algorithm has hyperparameters (pretty much all of them)
- **Ideally:** Test data is never looked at until development is done and it is time for evaluation

Discussion

For model selection, cross validation simulates training/testing **using only training data**

- Useful whenever ML algorithm has hyperparameters (pretty much all of them)
- **Ideally:** Test data is never looked at until development is done and it is time for evaluation
- **In practice:** Development process is iterative, and there is inevitably "leakage" of information from test data into development choices

Discussion

For model selection, cross validation simulates training/testing **using only training data**

- Useful whenever ML algorithm has hyperparameters (pretty much all of them)
- **Ideally:** Test data is never looked at until development is done and it is time for evaluation
- **In practice:** Development process is iterative, and there is inevitably "leakage" of information from test data into development choices
  - **Where possible:** Periodically acquire new test data so that reliable evaluations are possible