Model selection

Fraida Fund

Contents

Model selection problems

Model selection problem: how to select the $f()$ that maps features X to target y?

We'll look at two examples of model selection problems, but there are many more.

Choosing model complexity

We need to select a model of appropriate complexity -

- what does that mean, and
- how do we select one?

Model order selection problem

- Given data $(x_i, y_i), i = 1 \cdots, N$ (one feature)
- Polynomial model: $\widehat{y} = w_0 + w_1 x + \cdots + w_d x^d$
- d is degree of polynomial, called **model order**
- **Model order selection problem**: choosing d

Using loss function for model order selection?

Suppose we would "search" over each possible d:

- Fit model of order d on training data, get $\mathbf{w}$
- Compute predictions on training data
- Compute loss function on training data: $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$
- Select d that minimizes loss
- Problem: loss function always decreasing with d (training error decreases with model complexity!)

Feature selection problem

Given high dimensional data $\mathbf{X} \in R^{n \times d}$ and target variable y,

Select a subset of $k << d$ features, $\mathbf{X}_S \in R^{n \times k}$ that is most relevant to target y.

- Linear model: $\widehat{y} = w_0 + w_1 x_1 + \cdots + w_d x_d$
- Many features, only some are relevant
- **Feature selection problem**: fit a model with a small number of features

Why use a subset of features?

- High risk of overfitting if you use all features!
- For linear regression, there's a unique OLS solution only if $n \geq d$
- For linear regression, when $N \geq p$, variance increases linearly with number of parameters, inversely with number of samples. (Not derived in class, but read extra notes posted after class at home.)

Validation

Simple train/validation/test split

- Divide data into training, validation, test sets
- For each candidate model, learn model parameters on training set
- Measure error for all models on validation set
- Select model that minimizes error on validation set
- Evaluate model on test set

Note: sometimes you'll hear "validation set" and "test set" used according to the reverse meanings.

Simple train/validation/test algorithm (1)

- Split X, y into training, validation, and test.
- Loop over models of increasing complexity: For $p = 1, \ldots, p_{max}$,
  - **Fit**: $\widehat{w}_p = \text{fit}_p(X_{tr}, y_{tr})$
  - **Predict**: $\widehat{y}_{v,p} = \text{pred}(X_v, \widehat{w}_p)$
  - **Score**: $S_p = \text{score}(y_v, \widehat{y}_{v,p})$

Simple train/validation/test algorithm (1)

- Select model order with best score (here, assuming "lower is better"):

$$p^* = \underset{p}{\text{argmin}}\, S_p$$

- Evaluate:

$$S_{p^*} = \text{score}(y_{ts}, \widehat{y}_{ts,p^*}), \quad \widehat{y}_{ts,p^*} = \text{pred}(X_{ts}, \widehat{w}_{p^*})$$

Problems with simple split

- Fitted model (and test error!) varies a lot depending on samples selected for training and validation.
- Fewer samples available for estimating parameters.
- Especially bad for problems with small number of samples.

K-fold cross validation

Alternative to simple split:

- Divide data into K equal-sized parts (typically 5, 10)
- For each of the "splits": evaluate model using $K-1$ parts for training, last part for validation
- Average the K validation scores and choose based on average

K-fold CV illustrated

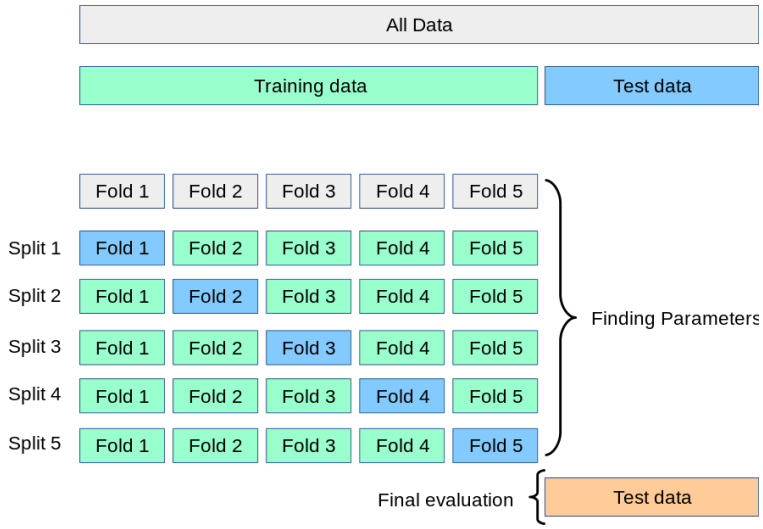

Figure 1: K-fold CV

K-fold CV - algorithm (1)

Outer loop over folds: for $i = 1$ to K

- Get training and validation sets for fold i:
- **Inner loop** over models of increasing complexity: For $p = 1$ to p_{max},
  - **Fit**: $\hat{w}_{p,i} = \text{fit}_p(X_{tr_i}, y_{tr_i})$
  - **Predict**: $\hat{y}_{v_i,p} = \text{pred}(X_{v_i}, \hat{w}_{p,i})$
  - **Score**: $S_{p,i} = score(y_{v_i}, \hat{y}_{v_i,p})$

K-fold CV - algorithm (2)

- Find average score (across K scores) for each model: $\bar{S}_p$
- Select model with best *average* score: $p^* = \text{argmin}_p \bar{S}_p$
- Re-train model on entire training set: $\hat{w}_{p^*} = \text{fit}_p(X_{tr}, y_{tr})$
- Evaluate new fitted model on test set

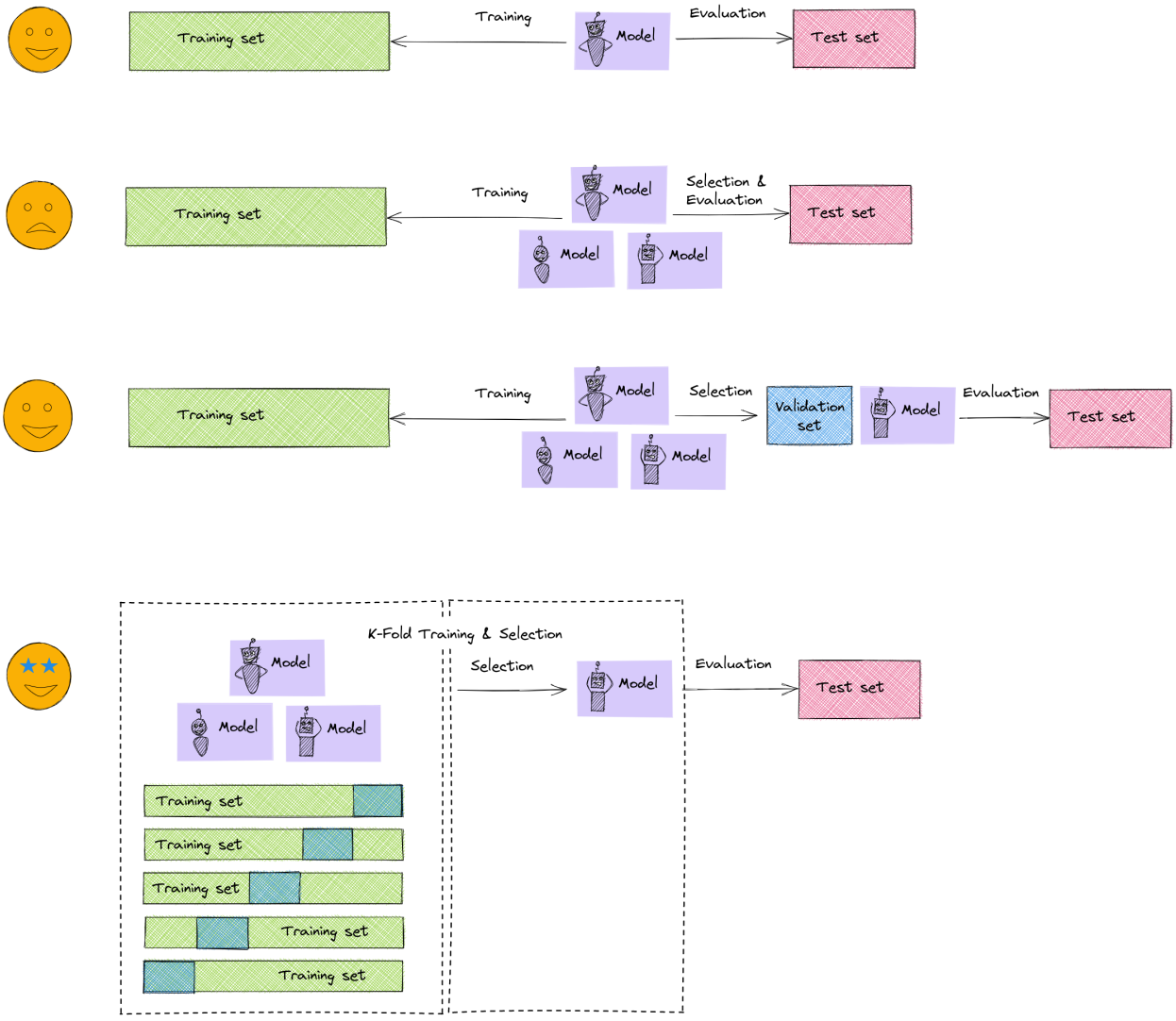

Figure 2: Summary of approaches. Source.

K-fold CV - how to split?

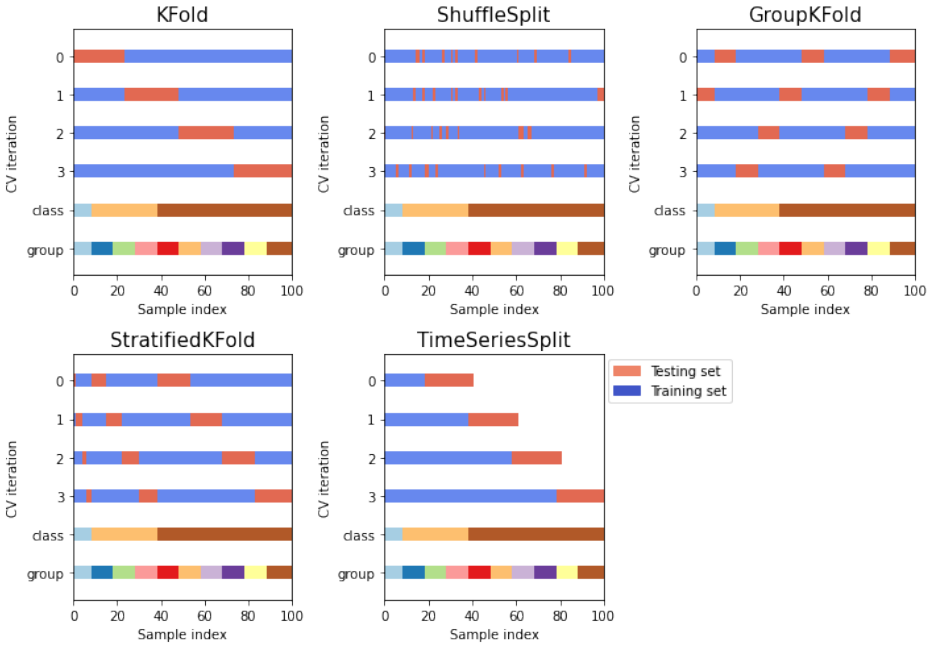

Figure 3: K-fold CV variations.

Selecting the right K-fold CV is very important for avoiding data leakage!

Refer to the function documentation for more examples.

One standard error rule

- Model selection that minimizes mean error often results in too-complex model
- One standard error rule: use simplest model where mean error is within one SE of the minimum mean error

One standard error rule - algorithm (1)

- Given data X, y
- Compute score $S_{p,i}$ for model p on fold i (of K)
- Compute average ($\bar{S}_p$), standard deviation σ_p, and standard error of scores:

$$SE_p = \frac{\sigma_p}{\sqrt{K-1}}$$

One standard error rule - algorithm (2)

"Best score" model selection: $p^* = \text{argmin}_p \bar{S}_p$

One SE rule for "lower is better" scoring metric: Compute target score: $S_t = \bar{S}_{p^*} + SE_{p^*}$

then select simplest model with score lower than target:

$$p^{*,\text{1SE}} = \min\{p | \bar{S}_p \leq S_t\}$$

Note: this assumes you are using a "smaller is better" metric such as MSE. If you are using a "larger is better" metric, like R2, how would we change the algorithm?

One standard error rule - algorithm (3)

"Best score" model selection: $p^* = \text{argmax}_p \bar{S}_p$

One SE rule for "higher is better" scoring metric: Compute target score: $S_t = \bar{S}_{p^*} - SE_{p^*}$

then select simplest model with score higher than target:

$$p^{*,\text{1SE}} = \min\{p | \bar{S}_p \geq S_t\}$$