Week 6: Cross-validation

MATH-517 Statistical Computation and Visualization

Tomas Masak

October 28th 2022

Principal Component Analysis (PCA)

(1) linear combinations with maximal variance (Pearson, 1901)

$$v_1 = \underset{\|v\|=1}{\arg\max}\ v^\top \widehat{\mathbf{\Sigma}} v$$

$$v_2 = \underset{\|v\|=1, v^\top v_1 = 0}{\arg\max}\ v^\top \widehat{\mathbf{\Sigma}} v$$

$$\vdots$$

(2) minimum-error projection into lower dimension (Hotelling, 1933)

$$\underset{V \in \mathbb{R}^{p \times r}, V^\top V = I}{\arg\min} \sum_{i=1}^{n} \|x_i - \mathbf{V}\mathbf{V}^\top x_i\|_2^2$$

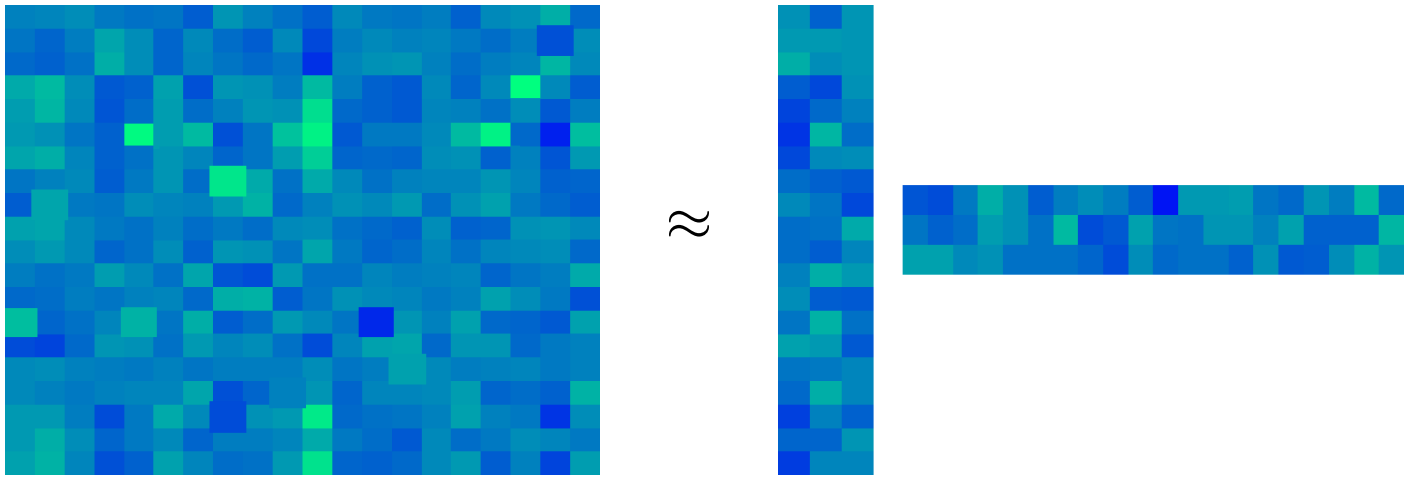
(3) low-rank matrix approximation (Eckart & Young, 1936)

$$\underset{\mathrm{rank}(\mathbf{L})=r}{\arg\min} \|\mathbf{X} - \mathbf{L}\|_2^2$$

Simply given by truncating the **SVD** decomposition: $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$

(3) Low-rank Matrix Approximation

Visualization for $r = 3$:



$$\mathbf{X} \approx \mathbf{L} = \mathbf{A}\mathbf{V}^\top = \sum_{i=1}^{r} \mathbf{a}_i \mathbf{v}_i^\top$$

Tuning Parameter Selection

Many procedures (estimators, algorithms, etc.) require choice of a certain tuning parameter

1. KDE, $h > 0$

$$\widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

2. Local Polynomial Regression (with a fixed degree p), $h > 0$

$$\arg\min_{\beta \in \mathbb{R}^{p+1}} \sum_{i=1}^{n} [Y_i - \beta_0 - \beta_1(X_i - x) - \ldots - \beta_p(X_i - x)^p]^2 K\left(\frac{X_i - x}{h}\right)$$

Tuning Parameter Selection

Many procedures (estimators, algorithms, etc.) require choice of a certain tuning parameter

3. Ridge Regression $\lambda > 0$ (similarly Lasso: replace $\|\cdot\|_2^2$ by $\|\cdot\|_1$)

$$\arg\min_\beta \sum_{n=1}^{N} \left(y_n - x_n^\top \beta\right)^2 + \lambda \|\beta\|_2^2.$$

4. PCA, $r \in \mathbb{N}$

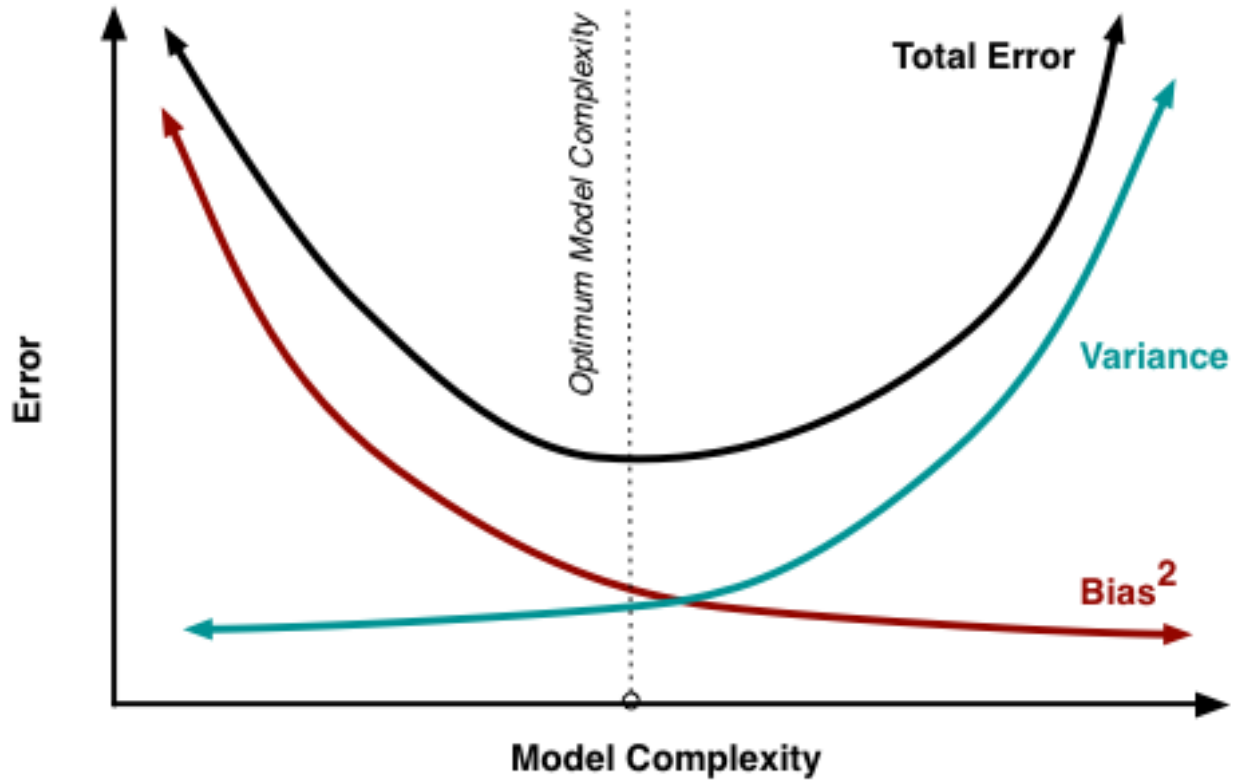
$$\underset{\operatorname{rank}(\mathbf{L})=r}{\arg\min} \|\mathbf{X} - \mathbf{L}\|_2^2$$

And many others...

In all cases, **cross-validation (CV)** can be used to select the tuning parameters.

- not always straightforward!

Bias-variance Trade-off



Local Polynomial Regression

Setup: A sample $(x_1, y_1)^\top, \ldots, (x_N, y_N)^\top \in \mathbb{R}^2$ from a population $Y = m(X) + \epsilon$ with $X \perp\!\!\!\perp \epsilon$ and for a fixed bandwidth h, we are estimating $m(x) = \mathbb{E}[Y|X = x]$ as $\widehat{m}_h(x)$ by e.g. local linear regression.

Question: How to choose h? (I.e. how to obtain a good bias-variance trade-off?)

What is the measure of how good our estimator $\widehat{m}_h(x)$ for a given bandwidth is?

$$MISE(\widehat{m}_h) = \int \mathbb{E}\big(\widehat{m}_h(x) - m(x)\big)^2 f_X(x) dx,$$

- let's choose h that minimizes MISE

Local Polynomial Regression

But we don't know m. How about using

$$\frac{1}{N}\sum_{n=1}^{N}\left(Y_n - \widehat{m}_h(X_n)\right)^2.$$

as a proxy for MISE?

That's a *bad idea*, because $\left(Y_n - \widehat{m}_h(X_n)\right)^2 \to 0$ for $h \to 0$

- this is called *overfitting*
- the problem lies in validating given h on data used to fit the model

Instead, consider this to approximate MISE:

$$CV(h) = \frac{1}{N}\sum_{n=1}^{N}\left(Y_n - \widehat{m}_h^{(-n)}(X_n)\right)^2,$$

where $\widehat{m}_h^{(-n)}(X_n)$ is the model fitted without n-th observation.

CV for Local Polynomial Regression

$$CV(h) = \frac{1}{N}\sum_{n=1}^{N}\left(Y_n - \widehat{m}_h^{(-n)}(X_n)\right)^2$$

Since $Y = m(X) + \epsilon$, we can write

$$\begin{aligned} CV(h) &= \frac{1}{N}\sum_{n=1}^{N}\left(Y_n - m(X_n) + m(X_n) - \widehat{m}_h^{(-n)}(X_n)\right)^2 \\ &= \frac{1}{N}\sum_{n=1}^{N}\epsilon_n^2 + \frac{2}{N}\sum_{n=1}^{N}\epsilon_n\left(m(X_n) - \widehat{m}_h^{(-n)}(X_n)\right) \\ &\qquad + \frac{1}{N}\sum_{n=1}^{N}\underbrace{\left(m(X_n) - \widehat{m}_h^{(-n)}(X_n)\right)^2}_{\mathbb{E}\star = MISE}, \end{aligned}$$

$$MISE(\widehat{m}_h) = \int \mathbb{E}\left(\widehat{m}_h(x) - m(x)\right)^2 f_X(x)dx$$

CV can be Easy for Prediction

More generally: $(x_1, y_1)^\top, \ldots, (x_N, y_N)^\top \in \mathbb{R}^{p+1}$

Model for prediction: $\widehat{Y} = \widehat{m}(X)$

How good is the model: measured by a loss function, e.g. $\mathbb{E}\big(Y - \widehat{m}(X)\big)^2$

- other losses possible, e.g. when undershooting better than overshooting

If another data set $(x_1^\star, y_1^\star)^\top, \ldots, (x_M^\star, y_M^\star)^\top$ available (generated by the same process as the original data set), we can approximate loss empirically

$$\frac{1}{M} \sum_{m=1}^{M} (y_k^\star - \widehat{m}(x_k^\star))^2$$

CV is the alternative when no other data set available:

$$CV(\widehat{m}) := \frac{1}{N} \sum_{n=1}^{N} \big(y_n - \widehat{m}^{(-n)}(x_n)\big)^2,$$

where $\widehat{m}^{(-n)}$ is the model fitted without the n-th observation

CV can be Easy for Prediction

It can often be shown (under assumptions!) like in the case of local polynomial regression that

$$CV(\widehat{m}) \to \mathbb{E}\big(Y - \widehat{m}(X)\big)^2$$

CV can be used to compare candidate models $\widehat{m}_1, \ldots, \widehat{m}_j$

- can be completely different models
  - typically is the same model for different tuning parameter values
  - combinations possible
- select the model for which the CV criterion is minimized
- beware: when not in the "vanilla" iid case (e.g. times series, stratified data, etc.), things are not so straightforward

But there are computational costs. The model has to be re-fitted for

- all the tuning parameter values considered
- every data point left out
  - actually, this might not be necessary...

Computational Shortcut for Linear Smoothers

If $\widehat{m}$ is a linear smoother, i.e. the predictions $\widehat{y}_n = \widehat{m}(x_n)$ are given all together as

$$\widehat{\mathbf{y}} = \mathbf{S}\mathbf{y}$$

where $\mathbf{S} \in \mathbb{R}^{N \times N}$ depends on x's, then re-fitting (leaving out data points one by one) may not be necessary!

Example: Ridge regression is a linear smoother

$$\arg\min_\beta \sum_{n=1}^{N} \left(y_n - x_n^\top \beta\right)^2 + \lambda \|\beta\|_2^2.$$

- $\widehat{\beta} = (\mathbf{X}^\top \mathbf{X} + \lambda I)^{-1} \mathbf{X}^\top \mathbf{y}$
- $\widehat{\mathbf{y}} = \underbrace{\mathbf{X}(\mathbf{X}^\top \mathbf{X} + \lambda I)^{-1} \mathbf{X}^\top}_{=:\mathbf{S}} \mathbf{y}$

$$CV(\lambda) = \frac{1}{N} \sum_{n=1}^{N} \left(y_n - \mathbf{x}_n^\top \widehat{\beta}^{(-n)}\right)^2 = \frac{1}{N} \sum_{n=1}^{N} \left(\frac{y_n - \widehat{m}(x_n)}{1 - s_{nn}}\right)^2$$

Example: Ridge Regression

Noticing $\widehat{\beta}^{(-n)} = (\mathbf{X}^\top \mathbf{X} + \lambda I - \mathbf{x}_n \mathbf{x}_n^\top)^{-1}(\mathbf{X}^\top \mathbf{y} - \mathbf{x}_n y_n)$, we can use Sherman-Morrison formula:

- denoting $\mathbf{A} := \mathbf{X}^\top \mathbf{X} + \lambda I$
- $\alpha_n := 1 - \mathbf{x}_n^\top \mathbf{A}^{-1} \mathbf{x}_n^\top$

$$\begin{aligned}\widehat{\beta}^{(-n)} &= \left(\mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{x}_n \mathbf{x}_n^\top \mathbf{A}^{-1}}{1 - \mathbf{x}_n^\top \mathbf{A}^{-1} \mathbf{x}_n}\right)(\mathbf{X}^\top \mathbf{y} - \mathbf{x}_n y_n) \\ &= \widehat{\beta} - \frac{1}{\alpha_n}(\mathbf{A}^{-1}\mathbf{x}_n \mathbf{x}_n^\top \widehat{\beta} - \mathbf{A}^{-1}\mathbf{x}_n y_n).\end{aligned}$$

Plug this back into the general CV formula and do some more simple to obtain the last formula on the previous slide.

- check out lecture notes for details, if interested

Computational Shortcut for Linear Smoothers

A similar computational shortcut possible for

- linear models
- local constant regression
  - what about other polynomial orders?
- ridge regression
- KDE (when working on a grid)

On the other hand, such shortcuts not possible for

- lasso
- many other penalized or otherwise complicated estimators

When a computational shortcut impossible, perform K-fold CV instead.

K-fold CV

Split the data set randomly into $K \in \mathbb{N}$ subsets (*folds*) of approximately equal size:

- folds $J_k \subset \{1, \ldots, N\}$ for $k = 1, \ldots, K$ such that $J_k \cap J_{k'} = \emptyset$ for $k \neq k'$ and $\bigcup_{k=1}^{K} J_k = \{1, \ldots, n\}$
- in practice, choose $K = 5$ or $K = 10$, perform random permutation of indices and split:

```
N <- 20
K <- 5
ind <- matrix(sample(1:N),ncol=K)
ind
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   19   18    4    1    8
## [2,]   14   12    7   20    2
## [3,]   11    3    6   16    9
## [4,]   10   15    5   17   13
```

K-fold CV

Instead of the (leave-one-out) CV criterion

$$CV(\widehat{m}) := \frac{1}{N}\sum_{n=1}^{N}\left(y_n - \widehat{m}^{(-n)}(x_n)\right)^2,$$

use the K-fold CV criterion:

$$CV_K(\widehat{m}) = K^{-1}\sum_{k=1}^{K}|J_k|^{-1}\sum_{n\in J_k}\left(Y_n - \widehat{m}^{(-J_k)}(X_n)\right)^2,$$

where $m^{(-J_k)}$ is the model fitted without the data in the k-th fold J_k

- requires every candidate model to be fit K-times
- it is difficult to study properties of $CV_K(\widehat{m})$ properly, one usually examines whether leave-one-out CV works and, if yes and if no computational shortcuts available, resorts to K-fold CV for computational reasons

Section 1

CV for Unsupervised Problems

Bandwidth Selection for KDE

Sample $X_1, \ldots, X_N$ from f, goal is to estimate $f(x)$ by

$$\widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

- no response here!

A good estimator (a well-chosen h) minimizes

$$\begin{aligned} MISE(\widehat{f}_h) &= \mathbb{E} \int \left(\widehat{f}_h(x) - f(x)\right)^2 dx \\ &= \mathbb{E} \underbrace{\int \left[\widehat{f}_h(x)\right]^2 dx}_{\|\widehat{f}_h(x)\|_2^2} - 2\, \underbrace{\mathbb{E} \int \widehat{f}_h(x) f(x) dx}_{\text{the CV idea?}} + \underbrace{\int \left[f(x)\right]^2 dx}_{\text{no } h \text{ here}}. \end{aligned}$$

Bandwidth Selection for KDE

The CV idea: see how your estimator behaves on a left-out datum:

$$\mathbb{E}\widehat{f}_h^{(-n)}(X_n) = \mathbb{E}\frac{1}{(n-1)h}\sum_{j\neq n} K\left(\frac{X_n - X_j}{h}\right) = \mathbb{E}\frac{1}{h}K\left(\frac{X_1 - X_2}{h}\right)$$

$$= \int \underbrace{\int \frac{1}{h}K\left(\frac{x-y}{h}\right) f(y)dy}_{\mathbb{E}\widehat{f}_h(x)} f(x)dx = \mathbb{E}\int \widehat{f}_h(x)f(x)dx.$$

$\Rightarrow$ $N^{-1}\sum_{n=1}^{N}\widehat{f}_h^{(-n)}(X_n)$ approximates $\mathbb{E}\int \widehat{f}_h(x)f(x)dx$

$$\Rightarrow \qquad MISE(\widehat{f}_h) = \mathbb{E}\int \left[\widehat{f}_h(x)\right]^2 dx - 2\mathbb{E}\int \widehat{f}_h(x)f(x)dx + \int \left[f(x)\right]^2 dx.$$

can be estimated up to a constant by

$$CV(h) = \int \left[\widehat{f}_h(x)\right]^2 dx - \frac{2}{N}\sum_{n=1}^{N}\widehat{f}_h^{(-n)}(X_n)$$

CV for PCA

$$\underset{\text{rank}(\mathbf{L})=r}{\arg\min} \|\mathbf{X} - \mathbf{L}\|_2^2$$

How to choose the rank r?

Many people try the following K-fold CV scheme:

- split data into K folds $J_1, \ldots, J_K$
- **for** $k = 1, \ldots, K$
  - solve $\widehat{\mathbf{L}} = \underset{\text{rank}(\mathbf{L})=r}{\arg\min} \|\mathbf{X}[J_k^c,] - \mathbf{L}\|_2^2$
  - calculate $Err_k(r) = \sum_{n \in J_k} \|x_n - P_{\widehat{L}} x_n\|_2^2$
- **end for**
- choose $\widehat{r} = \underset{r}{\arg\min} \sum_{k=1}^{K} |J_k|^{-1} Err_k(r)$

But this is wrong! (as $r \nearrow$ we have $\|x_j - P_{\widehat{L}} x_j\| \searrow$, so r is overestimated)

In the PCA bible (Jolliffe, 2002), there are two other ways how to do CV for PCA, but one of them is outdated and the second also wrong.

Intermezzo: Linear Prediction for Gaussian Vectors

For $X \sim \mathcal{N}(\mu, \Sigma)$ split into

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \qquad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix},$$

the conditional expectation of X_1 given X_2 is given by

$$\mathbb{E}_{\mu,\Sigma}\left[X_1 \middle| X_2 = \mathbf{x}_2\right] = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$

Intermezzo: Linear Prediction for Gaussian Vectors

For $X \sim \mathcal{N}(\mu, \Sigma)$ split into

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \qquad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix},$$

the conditional expectation of X_1 given X_2 is given by

$$\mathbb{E}_{\mu,\Sigma}\left[X_1 \middle| X_2 = \mathbf{x}_2\right] = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_2)$$

Assume we have a sample $X_1, \ldots, X_N$ from which we obtain estimators $\widehat{\mu}$ and $\widehat{\Sigma}$, and a new incomplete observation $X^\star = (X_1^\star, X_2^\star)^\top$, where only $X_2^\star$ is observed. We simply

- plug in the estimators $\widehat{\mu}$ and $\widehat{\Sigma}$ into the conditional expectation above, and
- obtain a predictor $\widehat{X}_1^\star = \widehat{\mu}_1 + \widehat{\Sigma}_{12}\widehat{\Sigma}_{22}^{-1}(\mathbf{x}_2 - \mu_2)$

Even without Gaussianity, this is the best linear unbiased predictor (BLUP).

- The quality of BLUP depends on the quality of the estimators $\widehat{\mu}$ and $\widehat{\Sigma}$.

CV for PCA Repaired

Assume that data $\mathbf{x}_n \in \mathbb{R}^p$ are i.i.d. realizations of $X \sim \mathcal{N}(\mu, \Sigma)$.

- split data into K folds $J_1, \ldots, J_K$
- **for** $k = 1, \ldots, K$
  - estimate μ and Σ empirically using all but the k-th fold J_k, but truncate Σ to be rank-r
  - **for** $n \in J_k$
    - split $\mathbf{x}_n$ a "missing" part $\mathbf{x}^{miss}$ that will be used for validation and an "observed" part $\mathbf{x}^{obs}$
    - predict $\mathbf{x}_n^{miss}$ from $\mathbf{x}_n^{obs}$ as discussed on the previous slide
  - **end for**
  - calculate $Err_k(r) = \sum_{n \in J_k} \|\mathbf{x}_n^{obs} - \widehat{\mathbf{x}}_n^{obs}\|_2^2$
- **end for**
- choose $\widehat{r} = \arg\min_r \sum_{k=1}^K |J_k|^{-1} Err_k(r)$

Is there a bias-variance trade-off now?

Assignment 4 [5 %]

Consider a subset of `mcycle` data (of the `MASS` package) for `times` ≤ 40 and use cross-validation to select

- the polynomial degree of p from candidate values $p = 1, 2, 3$, and
- the bandwidth h from candidate values $h = 3, 4, \ldots, 15$

for a local polynomial smoother as implemented by the `locpol()` function from the package of the same name.

Notes:

- Compare your results with what you would expect based on Manual 10 in order to avoid wrong conclusions.
  - even better: use your own visualizations to verify your progress
- You may run into issues for large p and small h if you use small number of folds.
- Beware of how your points are ordered.

Data for Assignment 4

```
library(MASS)
data(mcycle)
mcycle <- mcycle[mcycle$times <= 40, ]
plot(mcycle$times,mcycle$accel)
```