Lecture 6 - Resampling

Computational Statistics and Applications

Vu Quoc Hoang (vqhoang@fit.hcmus.edu.vn)
Tran Thi Thao Nhi (thaonhitt2005@gmail.com)
FIT - HCMUS

February 5, 2023

Agenda

1. Introduction
2. Permutation resampling
3. Bootstrapping
4. Cross-validation

Agenda

1. **Introduction**
2. Permutation resampling
3. Bootstrapping
4. Cross-validation

Introduction

Resampling is a simulated technique in which **data** is reused for generation process. (Monte Carlo method requires a statistical model to generate samples). While such methods will typically be less accurate than Monte Carlo methods, their advantages are the simplicity of the resulting algorithms and the wide applicability of these methods.

Three common resampling methods

- Permutation resampling,
- Bootstrapping,
- Cross-validation.

Agenda

1. Introduction
2. **Permutation resampling**
3. Bootstrapping
4. Cross-validation

Permutation resampling

Permutation resampling methods use **sampling without replacement** on samples, or equivalently, re-arranging the order or picking out permutations or "shuffling" the samples. This procedure is typically used in the area of statistical test, which is called as **permutation test**.

Example. The following table records the height (in meters) of 10 females and 10 males.

Female	1.66	1.58	1.58	1.56	1.63	1.51	1.67	1.57	1.61	1.59
Male	1.77	1.66	1.66	1.64	1.73	1.57	1.78	1.65	1.71	1.68

We find out, from the table, that the mean height for females and males is 1.60, 1.68 (meters) respectively. Question: Are females "really" shorter than males?

Permutation resampling - Example (cont.)

Using **t-test** for the 3 following hypotheses

1. 10 females and 10 males are randomly (and independently) picked out,
2. the height of females and males is normal distributed,
3. the variance of height is the same in both groups.

For test "the mean height for females is less than that for males" (one-sided test for a "less" hypothesis), we have the p-**value** is around 0.0012.
Then, females are "really" shorter than males.

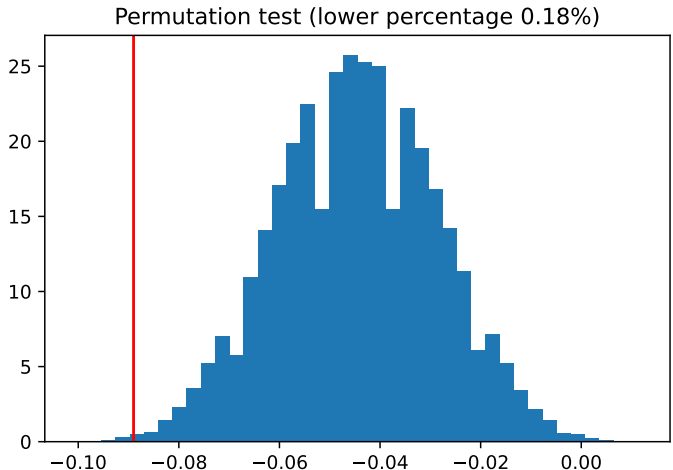
However, those hypotheses might be wrong when we apply permutation test for another hypothesis: females are not "really" shorter than males. Then, the difference in the mean height between females and males ($1.60 - 1.68 = -0.08$ meters) is likely to appear due to "random chance" and the difference becomes "common case" when the observed samples in both groups are randomly reshuffled.

Permutation resampling - Example (cont.)

```
def rand_perm(x1, x2):
    n1, n2 = len(x1), len(x2)
    inds = np.arange(n1 + n2)
    np.random.shuffle(inds)
    x = np.hstack((x1, x2))
    return np.mean(x[inds[:n1]]) - np.mean(x[n1:])

N = 100000
samples = np.fromiter((rand_perm(female_height,
                male_height) for _ in range(N)), "float")
diff = np.mean(female_height) - np.mean(male_height)
np.mean(samples <= diff)
# 0.00176
```

Permutation resampling - Example (cont.)



Agenda

1. Introduction
2. Permutation resampling
3. **Bootstrapping**
4. Cross-validation

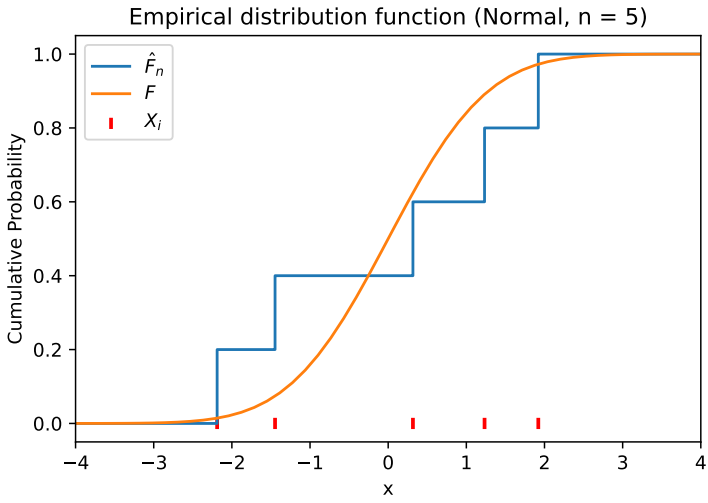
Empirical distribution function

Given a random sequence $D = \{X_i : i = 1, ..., n\}$ from distribution F, i.e. $X_1, ..., X_n$ is independently and uniformly distributed by **distribution function** (cumulative distribution function, cdf) F. **Empirical distribution function** of D is given by

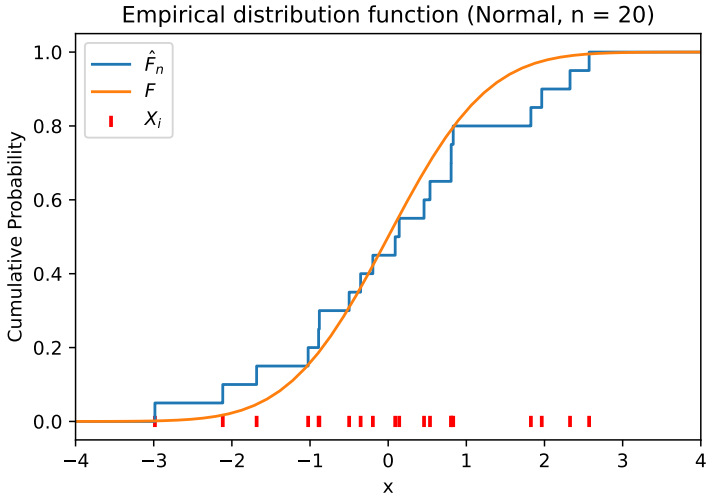
$$\hat{F}_D(x) = \hat{F}_n(x) = \hat{F}(x) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_{(-\infty,x]}(X_i) = \frac{\text{occurrences } X_i \leq x}{n}, x \in \mathbb{R}.$$

- $\hat{F}_n$ is the distribution function of discrete random variable that takes $X_1, ..., X_n$ as its value with the same probability $\frac{1}{n}$.
- "$\hat{F}_n \approx F$" for "sufficiently large" n.

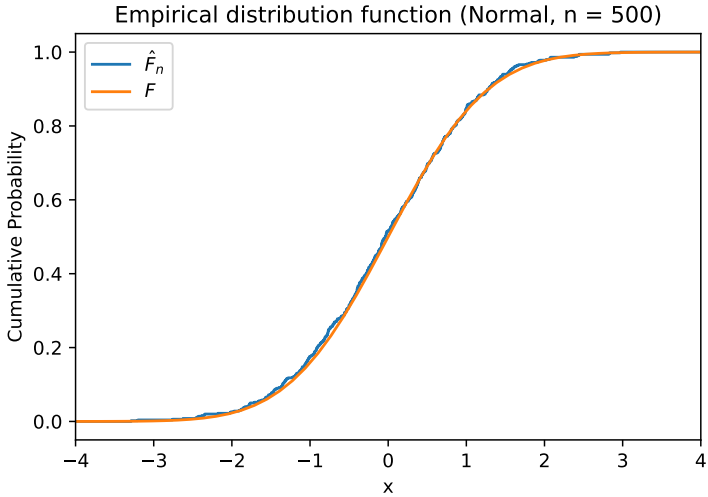
Empirical distribution function (cont.)



Empirical distribution function (cont.)



Empirical distribution function (cont.)



Bootstrapping

To generate a sample size m from an empirical distribution function $\hat{F}_D$ of $D = \{X_1, ..., X_n\}$, we can follow two steps:

1. Generate $I_1, ..., I_m \overset{\text{iid}}{\sim} \mathcal{U}\{1, ..., n\}$,
2. Return the samples $\{X_{I_1}, ..., X_{I_m}\}$ (a.k.a a **bootstrap sample** size m from D).

This method uses **sampling with replacement** from D.

Suppose that we have random samples $D = \{X_1, ..., X_n\}$ and a **statistic** $\hat{\theta} = \hat{\theta}(D)$ of samples. To estimate a **parameter** $\theta = \theta(F)$ for the distribution function F,

- If the samples can be drawn from F, then we use Monte Carlo methods (learned from the previous lectures), to generate $D_1, ..., D_N$ and compute $\hat{\theta}(D_1), ..., \hat{\theta}(D_N)$. Then, we can evaluate the obtained estimate $\hat{\theta}$.
- If we only have observed samples D, then we can use the resampling methods for $\hat{F}_D$ to generate a bootstrap sample size n $D_1^*, ..., D_N^*$ and compute $\hat{\theta}(D_1^*), ..., \hat{\theta}(D_N^*)$. Since "$\hat{F}_n \approx F$", we can still evaluate the obtained estimate $\hat{\theta}$.

Bootstrapping - Example 1

Suppose that we want to examine the average height of a population of all residents of a city. From the population, we collect a sample including the height of $n = 20$ random residents as shown in the following table.

1.56 1.47 1.59 1.65 1.62 1.78 1.69 1.49 1.92 1.55
1.65 1.52 1.65 1.60 1.71 1.48 1.69 1.65 1.59 1.74

Let μ be the mean height of the population, $\bar{x}$ be the mean height of $n = 20$ residents in the sample

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i.$$

Though the parameter μ is unknown, the statistic $\bar{x}$ can be used "appropriately" to estimate μ, which leads to an approximation as follows:

$$\mu \approx \bar{x} = 1.63.$$

Bootstrapping - Example 1 (cont.)

However, we cannot be "confident" about the obtained estimate since observations may be completely different in each collected samples. Then instead of a estimate, an estimated interval is obtained for μ, which takes into account the variation of $\bar{x}$ on the sample size $n = 20$ collected from the population.
According to the central limit theorem, if the sample size n is sufficiently large, $\bar{x}$ follows an "approximately" normal distribution, or

$$\bar{x} \sim \mathcal{N}\left(\mu, \sigma^2/n\right)$$

where σ^2 is the variance height of the population. Then, the estimated interval $1 - \alpha$ for μ is given by

$$\bar{x} \pm z_{1-\alpha/2} se$$

where $z_{1-\alpha/2}$ is the $1 - \alpha/2$-quantile of the standard normal distribution $Z \sim \mathcal{N}(0, 1)$ and se is the standard error of $\bar{x}$ given as:

$$se = \sqrt{Var(\bar{x})} = \frac{\sigma}{\sqrt{n}}$$

Bootstrapping - Example 1 (cont.)

If the value of σ cannot be evaluated, we approximate the standard error to $se \approx \hat{se} = \frac{s}{\sqrt{n}}$ where s is standard deviation of the samples

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

Replacing s by σ is actually not always be the case; instead, we can use Student's t-distribution to get a better approximation. This leads to the following expression:

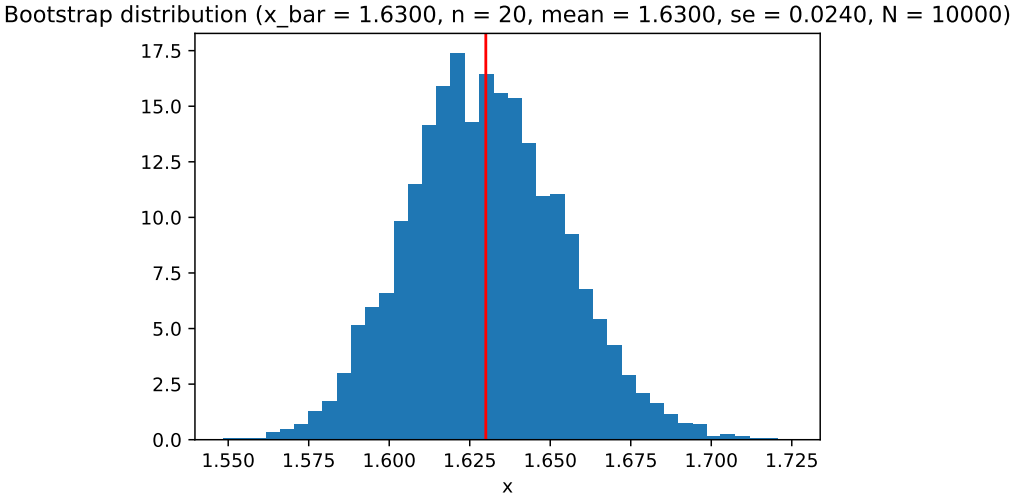
$$\frac{\bar{x} - \mu}{s/\sqrt{n}} \sim t^{n-1}$$

In this case, the estimated interval $1 - \alpha$ for μ is $\bar{x} \pm t^{n-1}_{1-\alpha/2}\frac{s}{\sqrt{n}}$ where $t^{n-1}_{1-\alpha/2}$ is the $1 - \alpha/2$-quantile of Student's t-distribution with $n - 1$ degrees of freedom.

Bootstrapping - Example 1 (cont.)

```
x = np.array([...])
n = len(x)
x_bar = np.mean(x)
N = 10000
boot_dist = [np.mean(np.random.choice(x, size=n, replace=
    True)) for _ in range(N)]
plt.hist(boot_dist, bins=40, density=True)
anpha = 1 - 95/100
print(np.quantile(boot_dist, [anpha/2, 1 - anpha/2]))
# [1.585 1.679]
se = np.std(x)/(n**0.5)
t = stats.t.ppf(1 - anpha/2, df=n-1)
print([x_bar - t*se, x_bar + t*se])
# [1.5798984074733546, 1.6801015925266456]
```

Bootstrapping - Example 1 (cont.)



Bootstrapping - Example 2

Linear regression y according to x by given data

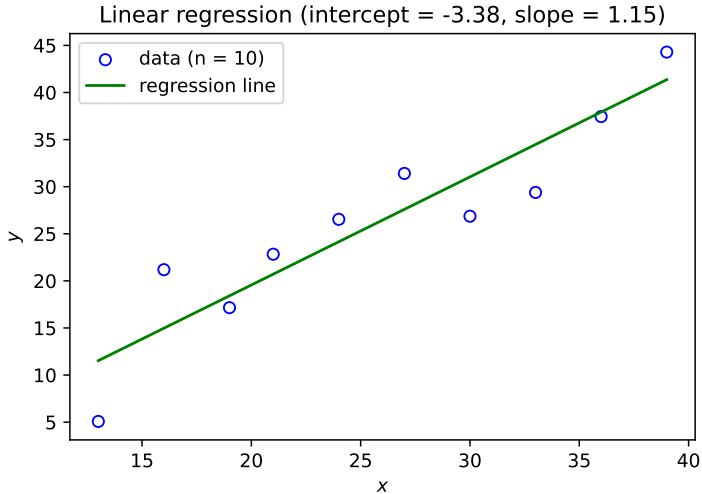
x	y	x	y
13	5.0768	27	31.4085
16	21.1897	30	26.8648
19	17.1548	33	29.3894
21	22.8325	36	37.4476
24	26.5348	39	44.2920

Model

$$y = \beta_0 + \beta_1 x + \epsilon,$$

β_0: intercept, β_1: slope, ϵ: random variable describing "noise".

Bootstrapping - Example 2 (cont.)



Bootstrapping - Example 2 (cont.)

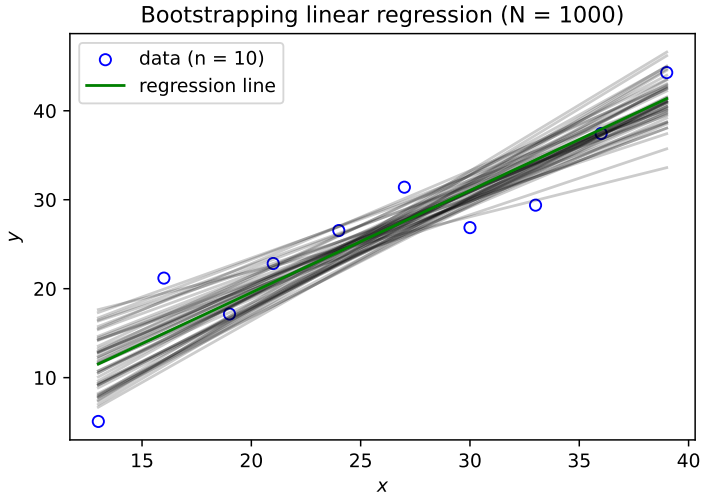
For a "deeper" model analysis (for instance, to qualify the confidence interval), we can use the bootstrapping technique in the following way:

1. Denote a sample $D = \{(x_i, y_i) : i = 1, ..., n\}$, drawn a bootstrap sample of same size as the original one $D^* = \{(x_i^*, y_i^*) : i = 1, ..., n\}$. (Note that, x_i, y_i are "corresponding".)
2. From bootstrap sample D^* calculate β_0^*, β_1^* (by the methods learned from the previous class).
3. Repeat step 1 and 2 (N times) to get the bootstrap distribution for (β_0, β_1).
4. Estimate or analyze any properties of the data based on the obtained bootstrap distribution (such as a confidence interval for β_1, or a confidence interval for y given x).

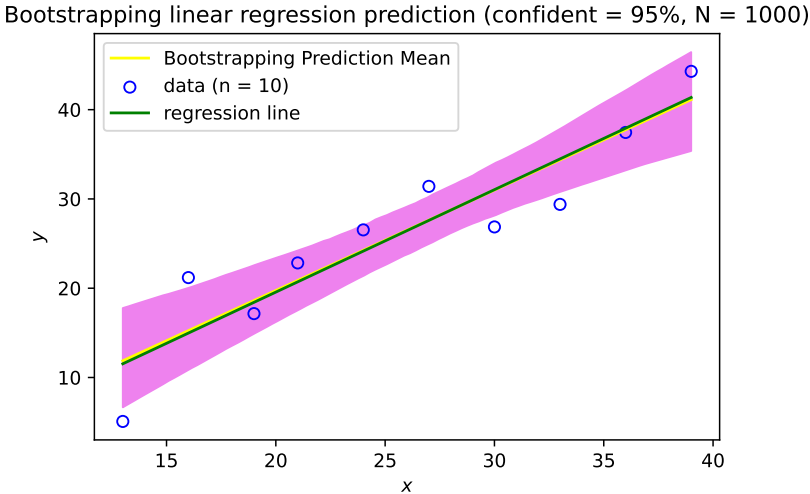
Bootstrapping - Example 2 (cont.)

```
def linregress_bootstrap(x, y, N):
    intercept, slope  = [], []
    for _ in range(N):
        index = np.random.choice(np.arange(len(x)), size=
           len(x), replace=True)
        x_boots, y_boots = x[index], y[index]
        lin_model = stats.linregress(x_boots, y_boots)
        intercept.append(lin_model.intercept)
        slope.append(lin_model.slope)
    return {"intercept": np.array(intercept),
            "slope": np.array(slope)}
boots_dist = linregress_bootstrap(x, y, 1000)
np.quantile(boots_dist["slope"], [0.025, 0.975])
# [0.64 1.43]
```

Bootstrapping - Example 2 (cont.)



Bootstrapping - Example 2 (cont.)



Agenda

1. Introduction
2. Permutation resampling
3. Bootstrapping
4. **Cross-validation**

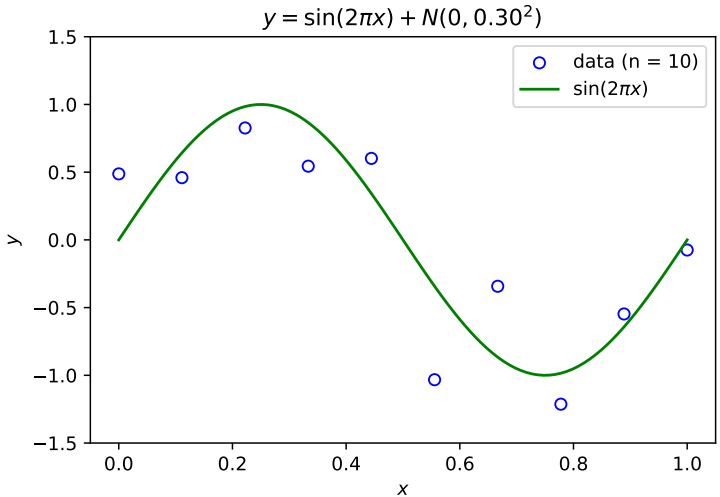
Example - Least square

Problem. Suppose that 2 variables x, y are somewhat correlated, such as

$$y = f(x) + \epsilon \text{ vi } f(x) = \sin(2\pi x) \text{ and } \epsilon \sim \mathcal{N}(0, \sigma^2).$$

Let $D = \{\mathbf{x}, \mathbf{y}\}$ be a sample where $\mathbf{x} = (x_1, x_2, ..., x_n)$ and $\mathbf{y} = (y_1, y_2, ..., y_n)$ respectively, specify the correlation between x, y; for instance, indicate f.

Example - Least square (cont.)



Example - Least square (cont.)

Solution. For "simplicity", we assume that f can be "modeled" by a polynomial of degree d given by

$$y = \sum_{j=0}^{d} w_j x^j = w_0 + w_1 x + w_2 x^2 + ... + w_d x^d, d \in \mathbb{N}, w_j \in \mathbb{R}.$$

We need to find the "best fit" d and $\mathbf{w} = (w_0, w_1, ..., w_d)$. In particular, we find suitable d and $\mathbf{w}$ where for the "overall" x $\hat{y} = \sum_{j=0}^{d} w_j x$ is as "fit" to $y = f(x)$ as possible. Since the observation D simply consists of n points x, y, we "temporarity" indicate d and $\mathbf{w}$ as **mean squared error** (MSE) on D

$$\mathcal{L}_D(d, \mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{d} w_j x_i^j \right)^2 .$$

Example - Least square (cont.)

Let d be fixed and define a matrix:

$$\mathbf{X} = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^d \end{pmatrix},$$

then the $\mathcal{L}_D(d, \mathbf{w})$ can be rewritten as follows:

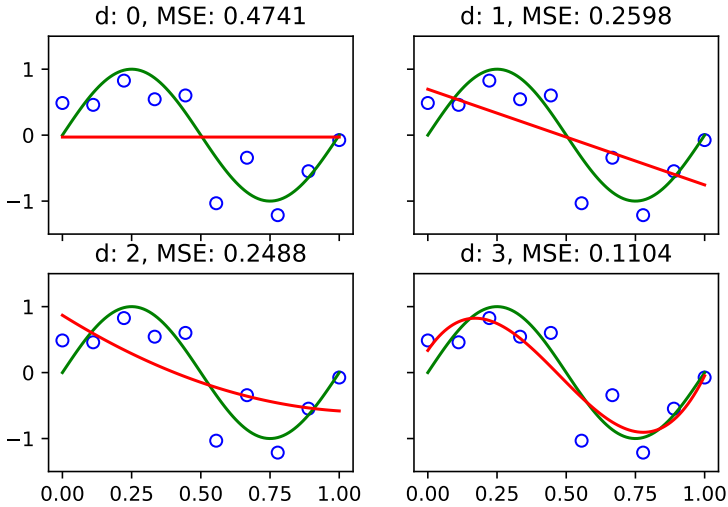
$$\mathcal{L}_D(d, \mathbf{w}) = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2.$$

The best **w** from **Least squares** method is given by

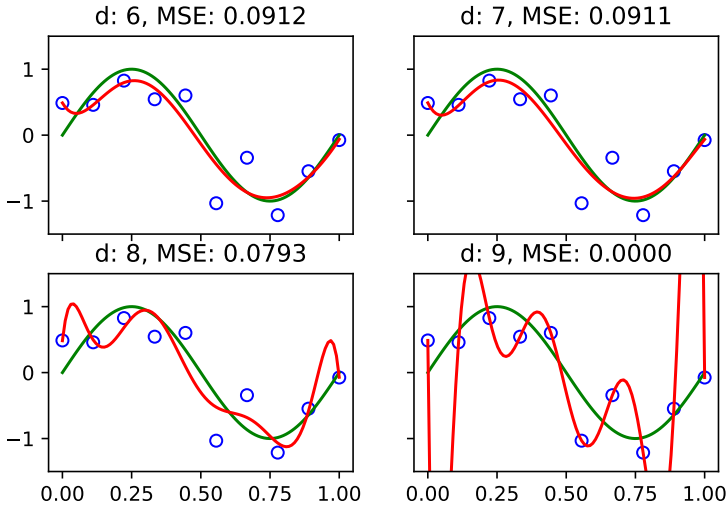
$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}}\ \mathcal{L}_D(d, \mathbf{w}) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{X}^\dagger\mathbf{y},$$

where $\mathbf{X}^\dagger$ is **pseudo-inverse** matrix of $\mathbf{X}$.

Example - Least square (cont.)



Example - Least square (cont.)



Example - Least square (cont.)

Note that

- If d is small ($d = 0, 1, 2$), the obtained polynomial is **"underfitting"** the training data. Poor performance in an underfitting model could be because the model is too simple to describe the target well.
- If d is large ($d = 8, 9$), the polynomial is **"overfitting"**. An overfitting model contains too much complexity, resulting in high **"noise"** and high error rates on test data.
- The larger d, the smaller MSE on data. However, the model should perform well on general x, y, especially on unseen/new x, y. This property is called as **generalization** of model.

Question: how to choose the best d?
Solution: cross-validatation
Principle: a set of data which is used in the evaluation (validation) must be distinct from **training** data.

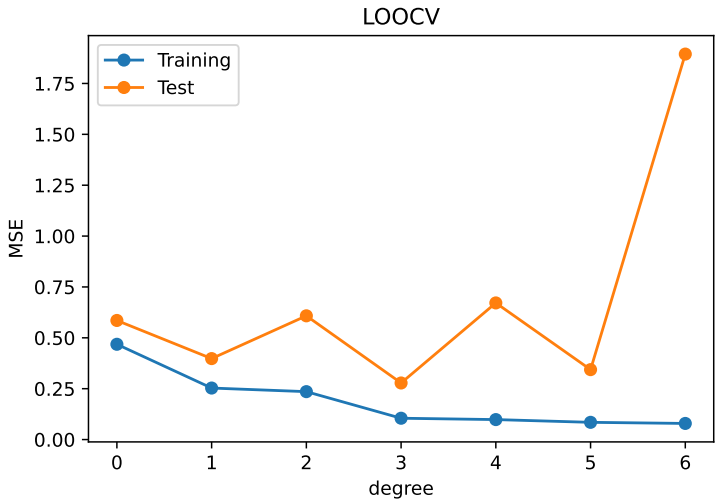
Cross-validation

- k-**fold cross-valiation** partitions the data into k subsamples of equal size, alternatively use $k-1$ subsamples to train the model and the omitted subsample to evaluate, average the predictive performance in k runs.
- **Leave-one-out cross-validation** (LOOCV) is a specific version of k-fold CV where $k = n$.
- More specifically, there is **leave-p-out cross-validation** (LpO CV)
  - Consider all ways to partition data into 2 distinct subsamples,
  - a subsample size $n - p$ to train,
  - a subsample size p to evaluate,
  - then, average the predictive performance in all runs.
- LOOCV is the particular case of LpO CV where $p = 1$.

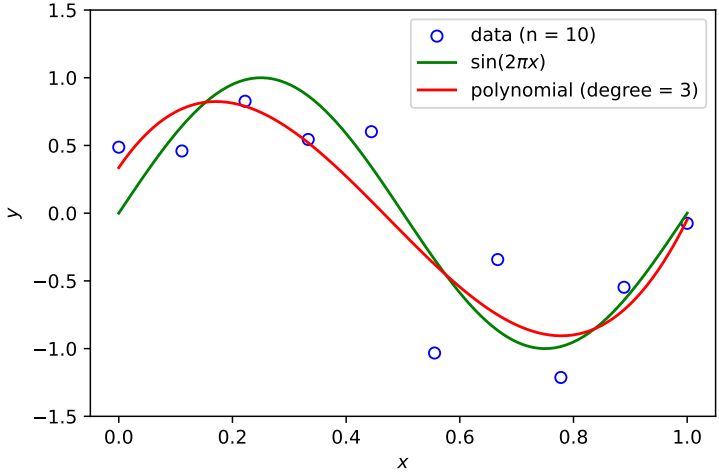
Cross-validation (cont.)

- In case of LpO CV, since all subsamples of data are used for validating (and training), this method is known as **exhaustive cross-validation**.
- In constrast, k-fold CV is a **non-exhaustive cross-validation**.
- Exhaustive cross-validations often provide a better measurement but incur higher costs. For instance, LpO CV requires training and validating the model $\binom{n}{p}$ times, instead of k times in k-fold CV.
- See more about cross-validatation on `https://en.wikipedia.org/wiki/Cross-validation_(statistics)`.

Cross-validation - Example (cont.)



Cross-validation - Example (cont.)



Refenrences

Chapter 2, 6-9, 15, 21. Thomas M. Carsey, Jeffrey J. Harden. *Monte Carlo Simulation and Resampling Methods for Social Science*. SAGE Publications, 2014.

Chapter 7. Dirk P. Kroese, Joshua C.C. Chan. *Statistical Modeling and Computation*. Springer, 2014.

Chapter 8, 9. Jochen Voss. *An Introduction to Statistical Computing - A Simulation-based Approach*. John Wiley & Sons, 2014.