Introduction to Data Science

- Statistical Inference: Estimators and Confidence Intervals -

Mauricio Molina

Keio University, Faculty of Economics

June 4, 2025

Contents

Law of Large Numbers — Intuition

Statement (weak form)

For independent, identically distributed (i.i.d.) random variables $X_1, X_2, \ldots$ with finite mean μ, the sample average

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

converges in probability to μ as $n \to \infty$.

Key implications for data science:

- Estimates get *more reliable* with a larger sample.
- Random fluctuations diminish at rate $\mathcal{O}(1/\sqrt{n})$.
- Foundation for Monte-Carlo methods and A/B testing.

LLN Example — Rolling a Die

1. **Set-up (fair die)** Possible outcomes: $1, 2, 3, 4, 5, 6$ with equal probability $1/6$.
2. **Population mean** $E[X] = \sum_{k=1}^{6} k\,P(X = k) = \frac{21}{6} = 3.5.$
3. **Experiment**
   - Roll the die n times (e.g. $n = 10, 100, 1{,}000, \dots$).
   - After each roll t, record the running average $\overline{X}_t = \frac{1}{t}\sum_{i=1}^{t} X_i$.
4. **Quantity of interest** How quickly does $\overline{X}_t$ get close to 3.5 as t grows?
5. **Prediction from the Law of Large Numbers** For any tolerance $\varepsilon > 0$,
   $P\big(|\overline{X}_t - 3.5| > \varepsilon\big) \longrightarrow 0 \quad (t \to \infty).$

Take-away: The average of many rolls behaves almost deterministically, providing a concrete, intuitive case of the LLN.

Python Demo — Running Mean of Die Rolls

Example of LLN

```
import numpy as np
import matplotlib.pyplot as plt

calc_times = 1000
sample_array = np.array([1, 2, 3, 4, 5, 6])
num_cnt = np.arange(1, calc_times+1)

for i in range(4):
p = np.random.choice(sample_array, calc_times).cumsum()
plt.plot(p/ num_cnt, label=f"#{i+1} experiment")
plt.legend()
plt.grid(True)
```

Central Limit Theorem — Idea

Statement (Lindeberg–Lévy version)

For i.i.d. variables with mean μ and variance σ^2, the standardised sample mean

$$Z_n = \frac{\sqrt{n}\,(\overline{X}_n - \mu)}{\sigma}$$

converges in distribution to the standard normal $\mathcal{N}(0, 1)$.

Consequences:

- Sampling distributions often *look normal* even when raw data do not.
- Enables confidence intervals, t-tests, and many inferential methods.

Sampling Means from a Die

1. **Underlying population** Fair six-sided die with mean $\mu = 3.5$ and variance $\sigma^2 = E[(X - 3.5)^2] = \frac{35}{12} \approx 2.92$.
2. **Choose a sample size** Fix $n = 30$.
3. **Repeat the sampling process**
   - Generate n independent rolls $\{X_1, \ldots, X_n\}$.
   - Compute the *sample mean* $\overline{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$.
   - Store this one number.
4. **Monte-Carlo loop** Do the above $N_{\text{samples}} = 10{,}000$ times to build a large collection $\{\overline{X}^{(1)}, \ldots, \overline{X}^{(N_s)}\}$.
5. **Analyse the empirical distribution**
   - Plot a histogram of the 10,000 sample means.
   - Overlay the theoretical normal curve $\mathcal{N}(\mu, \sigma/\sqrt{n})$.
6. **Central Limit Theorem prediction** As n grows, the distribution of $\sqrt{n}\,(\overline{X} - \mu)/\sigma$ approaches the standard normal, *regardless* of the discrete, non-normal nature of the die outcomes.

Sample Means & the CLT

Example of CLT

```
from scipy.stats import norm

def simulate_die_clt(n_samples: int = 10_000, sample_size: int = 30):
 """
 Demonstrate the Central Limit Theorem with a fair six-sided die.
 Parameters
 ----------
 n_samples   : int number of samples
 sample_size : int size of each sample
 Returns
 -------
 sample_means : np.ndarray, shape (n_samples,)
 Array containing the sample mean from each replication.
 """
 # Draw all rolls in one vectorised call: shape = (n_samples, sample_size)
 samples = np.random.randint(1, 7, size=(n_samples, sample_size))
 sample_means = samples.mean(axis=1)

 mu, sigma2 = 3.5, 35 / 12
 sigma = np.sqrt(sigma2)
 x = np.linspace(sample_means.min(), sample_means.max(), 300)

 plt.figure()
 plt.hist(sample_means, bins=30, density=True, alpha=0.7, label='Simulated means')
 plt.plot(x, norm.pdf(x, mu, sigma / np.sqrt(sample_size)), linewidth=2, label=r'$\mathcal{N}(\mu,\sigma/\sqrt{n})$')
 plt.xlabel(f'Sample mean (n = {sample_size})')
 plt.ylabel('Density')
 plt.title(f'CLT: {n_samples:,} means of {sample_size} die rolls')
 plt.legend()
```

CLT for Sums

Restating in terms of the sum $S_n = \sum_{i=1}^{n} X_i$:

$$S_n \approx \mathcal{N}\left(n\mu,\ \sigma\sqrt{n}\right).$$

Useful for modelling aggregate demand, total claim sizes, etc.

When Is n "Large" Enough?

- **Population normal** $\Rightarrow \overline{X}$ is normal for *any* n.
- **Population approximately symmetric**: $\overline{X}$ becomes nearly normal for relatively small n (often $n \geq 5$). Recall the dice example where $n = 3$ already looked mound–shaped.
- **Population skewed**: need larger samples; common guideline $n \geq 30$ before $\overline{X}$ is close to normal.
- Always check plots / skewness; in practice you may need even bigger n for heavy–tailed data.

Skewed Population

Exponential Distribution Example

```
# Simulate from an exponential distribution (skewed) and show CLT effect
import seaborn as sns; sns.set()

np.random.seed(1)
# rate parameter
lambda = 1
for n in [5, 30, 100]:
 means = np.random.exponential(scale=1/lambda, size=(10000, n)).mean(axis=1)
 sns.histplot(means, stat='density', bins=40, label=f'n={n}', kde=True)
 plt.axvline(1, ls='--')
plt.title('Sampling distribution of mean (Exponential population)');
plt.xlabel('sample mean'); plt.legend(); plt.show()
```

Key Take-aways for Practice

- CLT justifies using z or t intervals for sufficiently large n.
- Always consider underlying shape: skewed/heavy-tailed needs bigger n.
- Simulation is a powerful tool to check adequacy of the normal approximation in a specific case.

Estimators: What Are They?

- In statistics we rarely know a population parameter θ (e.g. mean, variance, correlation). Instead, we construct a *rule* that turns data into a number.
- **Estimator**

$$\widehat{\theta} \; = \; T\big(X_1, X_2, \ldots, X_n\big) \quad \text{(a function of the sample)}$$

  A random variable because it depends on the random sample.
- Examples
  - Sample mean $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$ estimates the population mean μ.
  - Sample variance $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$ estimates σ^2.
- Desirable properties (preview) unbiasedness, consistency, efficiency, robustness.

Point Estimation in Practice

1. **Choose an estimator** Decide on $T(\cdot)$ based on theoretical properties or convenience.
2. **Plug in the data**
$$\widehat{\theta}_{\text{obs}} = T(x_1, x_2, \ldots, x_n) \quad \text{(a single number)}$$
This is the *point estimate*.
3. **Interpret** Use $\widehat{\theta}_{\text{obs}}$ as your best guess for θ.
4. **Illustration — Die Example**
   - Parameter: $\mu = 3.5$.
   - Estimator: sample mean $\overline{X}_n$.
   - One simulation run with $n = 30$ might give $\overline{x}_n = 3.77$.
   - 3.77 is the point estimate; the estimator's distribution (via CLT) tells us its precision.
5. **Next steps** Interval estimation and hypothesis testing build on these ideas.

Bias of an Estimator

Definition

For an estimator $\widehat{\theta}$ of a parameter θ,

$$\text{Bias}(\widehat{\theta}) = \mathbf{E}[\widehat{\theta}] - \theta.$$

Interpretation

- Positive bias $\Rightarrow$ systematic over-estimation.
- Negative bias $\Rightarrow$ systematic under-estimation.
- Zero bias $\Rightarrow$ *unbiased*.

Example — Sample Variance

$$\widetilde{S}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2 \quad \Longrightarrow \quad \text{Bias}(\widetilde{S}^2) = -\frac{\sigma^2}{n}.$$

Dividing by $(n-1)$ instead of n removes this bias.

Mean-Squared Error (MSE) Decomposition

$$\mathrm{MSE}(\widehat{\theta}) = \mathrm{Var}(\widehat{\theta}) + \left[\text{Bias}(\widehat{\theta})\right]^2.$$

Shows the trade-off between variance and bias (e.g. ridge regression).

Unbiasedness

Definition

An estimator $\widehat{\theta}$ of a parameter θ is **unbiased** if

$$\mathbb{E}[\hat{\theta}] = \theta.$$

Why care? On average, you neither systematically over- nor under-estimate θ.

- **Example 1 — Sample mean**

  $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$ is unbiased for the population mean μ.

- **Example 2 — Sample variance**

  $$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$$

  is unbiased for σ^2, whereas $\frac{1}{n}\sum_{i}(X_i - \overline{X}_n)^2$ is *biased*.

- **Trade-off:** An unbiased estimator can still have large variance. Sometimes a *slight* bias is acceptable for a big variance reduction (e.g. ridge regression).

Consistency

Definition

An estimator $\widehat{\theta}_n$ is **consistent** for θ if

$$\widehat{\theta}_n \xrightarrow{p} \theta \quad (n \to \infty),$$

i.e. for every $\varepsilon > 0$, $P\big(|\widehat{\theta}_n - \theta| > \varepsilon\big) \to 0$.

Connection to LLN The sample mean $\overline{X}_n$ is consistent for μ because the LLN says exactly this.

- **Intuition**
  With more data, the estimator "homes in" on the truth.
- **Rate of convergence**
  Many estimators satisfy $\sqrt{n}\big(\widehat{\theta}_n - \theta\big) \xrightarrow{d} \mathcal{N}(0, \sigma^2)$ (by the CLT), giving a typical $1/\sqrt{n}$ precision improvement.
- **Implication**
  Consistency is a *minimum* requirement; biased but consistent estimators are common (e.g. maximum-likelihood estimates for small n).

Standard Error (SE)

Definition

The **standard error** of an estimator is its standard deviation:

$$\mathrm{SE}(\widehat{\theta}) = \sqrt{\mathrm{Var}(\widehat{\theta})}.$$

Why it matters

- Measures the typical sampling fluctuation around θ.
- Central ingredient in confidence intervals and hypothesis tests.

Example — Sample Mean of Die Rolls

$$\widehat{\mu} = \overline{X}_n, \qquad \mathrm{SE}(\overline{X}_n) = \frac{\sigma}{\sqrt{n}} = \sqrt{\frac{35}{12n}}.$$

- Doubling sample size $\Rightarrow$ SE shrinks by $\sqrt{2}$.
- LLN + CLT: SE $\to 0$ as $n \to \infty$, distribution $\approx \mathcal{N}$.

Estimated SE in practice

Replace unknown σ with its sample estimate $\widehat{\sigma}$: $\widehat{\mathrm{SE}}(\overline{X}_n) = \widehat{\sigma}/\sqrt{n}$. Used everywhere from regression output to A/B test dashboards.

Confidence Intervals — Concept

Idea

A $(1-\alpha)\,\%$ **confidence interval (CI)** for a parameter θ is a random interval $\big[L(X_{1:n}),\ U(X_{1:n})\big]$ constructed from the sample such that

$$P(\,\theta \in [L, U]\,) = 1 - \alpha.$$

Interpretation (95 % case, $\alpha = 0.05$) If we repeated the study many times, about 95 would contain the true θ. *The probability statement concerns the procedure, not the realised bounds.*

Why care?

- Quantifies the *precision* of a point estimate.
- Basis for significance tests and "margin of error" in polls.
- Width shrinks $\propto 1/\sqrt{n}$ — more data, narrower CI.

CI for a Mean (Large n or Known σ)

Setup

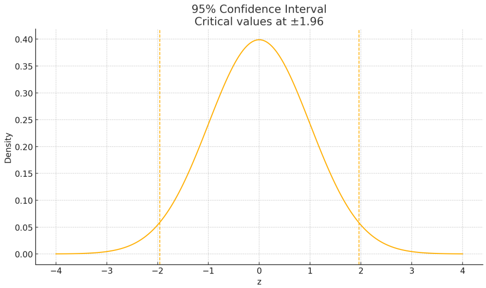
- i.i.d. sample $X_1, \ldots, X_n$, population mean μ, variance σ^2.
- For large n (CLT) or known σ, the $100(1-\alpha)\%$ CI is

$$\boxed{\overline{X}_n \pm z_{\alpha/2} \frac{\sigma}{\sqrt{n}}}$$

  where $z_{\alpha/2}$ is the $\alpha/2$ upper quantile of $\mathcal{N}(0,1)$ (e.g. 1.96 for 95 %).
- If σ unknown *and* n small, replace $z_{\alpha/2}$ with $t_{\alpha/2,\, n-1}$ and σ with S (sample sd).

Properties

- Centre $=$ point estimate $\overline{X}_n$.
- Half–width $=$ SE$\times$ critical value.
- Width $\downarrow$ as $n\uparrow$ or $\alpha\downarrow$.



Area under curve outside the CI $= \alpha$ (two tails).

Example — Die Rolls ($n = 30$)

Sample outcome (one run):
$\overline{x}_{30} = 3.33,\ S^2 = 3.35 \Rightarrow S = 1.83.$

95 % CI using $t_{0.025,\,29} = 2.045$

$$3.33 \pm 2.045 \frac{1.83}{\sqrt{30}} = 3.33 \pm 0.68 * 0.33 \implies [2.65,\ 4.02].$$

Example: if we repeated the 30-roll experiment many times, about 95 % of the resulting intervals would contain the true mean $\mu = 3.5$.

CI for mean

```
np.random.seed(0)

def mean_ci(data, alpha=0.05):
 n  = len(data)
 x_bar  = np.mean(data)
 s  = np.std(data, ddof=1)
 t_crit = t.ppf(1-alpha/2, df=n-1)
 half_width = t_crit * s / np.sqrt(n)
 return x_bar - half_width, x_bar + half_width

data = np.random.randint(1, 7, 30)
x_bar = np.mean(data)
s  = np.std(data, ddof=1)
ci = mean_ci(data)
print(f"sample mean: {x_bar}\n sample variance: {s}\n")
print(ci)
```

Assignment 8

Answer all three questions in a Jupyter Notebook. Show your Python code (when requested) and a short explanation for every result. Upload the completed `.ipynb` to K-LMS by next Tuesday at midnight.

Q1: Load Kangle's Car Price dataset and solve the following exercises.

- Randomly shuffle the rows; take the first k observations for each $k = 1, \ldots, 100$.
- Plot the running average of `price` versus k and add a horizontal line at the full-sample mean.
- Briefly explain how the plot illustrates the Law of Large Numbers.

Q2: Load Student Performance Math dataset and solve the following exercises.

- For $n \in \{10, 30, 100\}$ draw 1 000 single random samples of `G1`, store each sample mean.
- For each n: draw a histogram, and overlay $N(\mu, \sigma/\sqrt{n})$ using the *population* μ, σ of the whole file.
- Discuss how the shape changes with n and relate your findings to the Central Limit Theorem.

Q3: From the car price dataset, extract the variable `horsepower`. Treat the entire column as the population.

- Draw one simple random sample of size $n = 40$ *with replacement* and compute a 95 % t-interval for the population mean μ_{hp}.
- Now repeat the previous step 500 times, storing the interval width each time. Plot a histogram of the 500 widths and report their average.
- In 2–3 sentences, explain what you observe in the previous histogram.