Your accuracy was one lucky split.

D8A.ACADEMY/RESOURCES/CROSS-VALIDATION

WHY

one split	one lucky draw
k folds	k honest scores
report mean ± std	not the best one

THE LOOP

KFold(n_splits=5)	the splitter
cross_val_score	one-liner CV
shuffle=True	unless time data

VARIANTS

StratifiedKFold	keeps class ratio
GroupKFold	no user in both sides
TimeSeriesSplit	← for anything dated.

LEAKAGE

scale inside folds	never before split
Pipeline()	does it right free
impute inside too	same rule

NUMBERS

k = 5	the default
k = 10	small datasets
LeaveOneOut	tiny data only

GOTCHAS

CV on test set	← now it is train data.
KFold + imbalance	folds miss a class
huge std	model is unstable

WORTH MEMORIZING

01 The honest five lines

Scaling lives inside the pipeline, so every fold learns preprocessing from its own train side only.

```
from sklearn.pipeline import make_pipeline
pipe = make_pipeline(StandardScaler(),
                     LogisticRegression())
scores = cross_val_score(pipe, X, y,
          cv=StratifiedKFold(5, shuffle=True))
print(scores.mean(), scores.std())
```

→ Scale before the split instead, and the test rows have already whispered their mean to the model.

02 Trap: the wrong splitter

KFold is the default, not the answer. The data decides the splitter.

YOUR DATA	PLAIN KFOLD DOES	USE
95/5 imbalance	folds with zero positives	StratifiedKFold
many rows per user	same user on both sides	GroupKFold
timestamps	trains on the future	TimeSeriesSplit

→ The GroupKFold one is the interview favorite: leaking a user across the split inflates scores and nobody notices until production.