Machine Learning Fundamentals - LS2026
Model selection

Czech Technical University in Prague
V. Franc

Model selection

Given:

- Examples $D = ((x_1, y_1), \ldots, (x_n, y_n))$ drawn i.i.d. from an unknown distribution $p(x, y)$ over $\mathcal{X} \times \mathcal{Y}$.
- A set of learning algorithms $\mathcal{A} = \{A_1, A_2, \ldots, A_L\}$ where each

$$A_i \colon \cup_{m=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}_i$$

  learns a predictor $\hat{h} = A_i(D_{\text{trn}})$ from training examples D_{trn}.

Goal:

1. Learn a predictor $\hat{h} \colon \mathcal{X} \to \mathcal{Y}$ with small true risk

$$R(p, \hat{h}) = \mathbb{E}_{(x,y)\sim p}[\ell(y, \hat{h}(x))] \,.$$

2. Estimate the true risk $R(p, \hat{h})$ of the learned predictor.

Model selection

Examples:

- **Selection of hypothesis space.** Predictors learned by ERM with different hypothesis classes $\{\mathcal{H}_1, \ldots, \mathcal{H}_L\}$:

$$A_i(D_{\text{trn}}) = \arg\min_{h \in \mathcal{H}_i} \hat{R}(D_{\text{trn}}, h)\,, \qquad i \in \{1, \ldots, L\}$$

  where

$$\hat{R}(D, h) = \frac{1}{|D|} \sum_{i=1}^{|D|} \ell(y_i, h(x_i))\,.$$

- **SVM hyper-parameter tuning.** RBF kernel SVM classifiers learned with kernel width and regularization constant from $((\gamma_1, C_1), \ldots, (\gamma_L, C_L))$:

$$A_i(D_{\text{trn}}) = \text{SVM}(D_{\text{trn}}, \gamma_i, C_i)\,, \qquad i \in \{1, \ldots, L\}\,.$$

Structural Risk Minimization

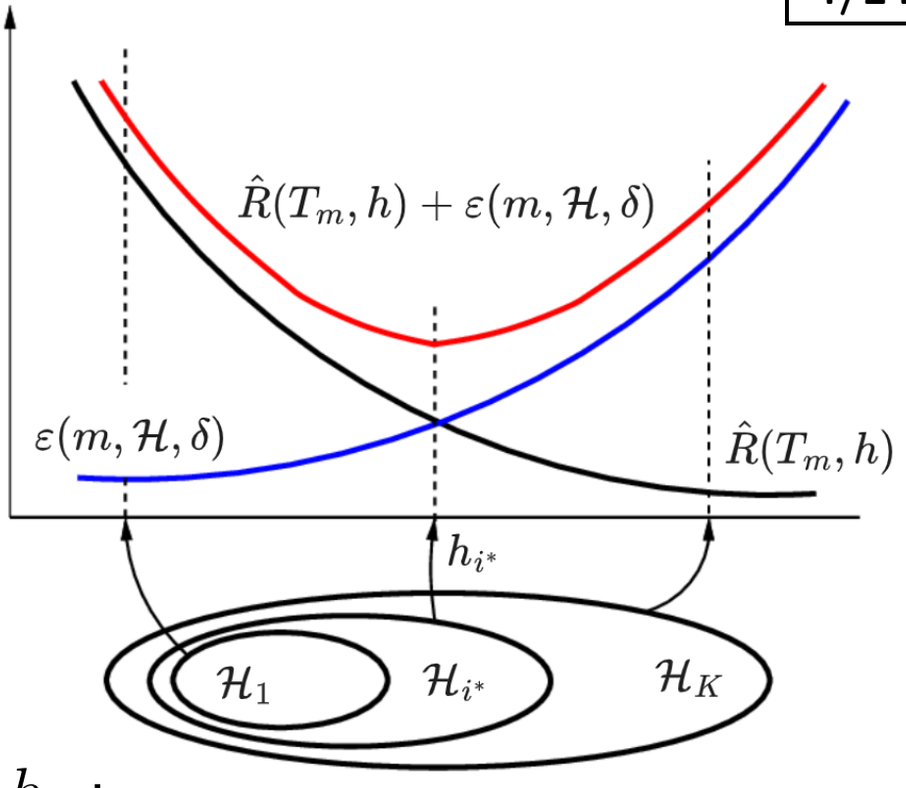
Algorithm:

1. Construct a nested sequence:

$$\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_L$$

2. Apply ERM for each $i \in \{1, \ldots, L\}$:

$$h_i = \arg\min_{h \in \mathcal{H}_i} \hat{R}(D_{\text{trn}}, h)$$



3. Use VC bound to select the best model h_{i^*}:

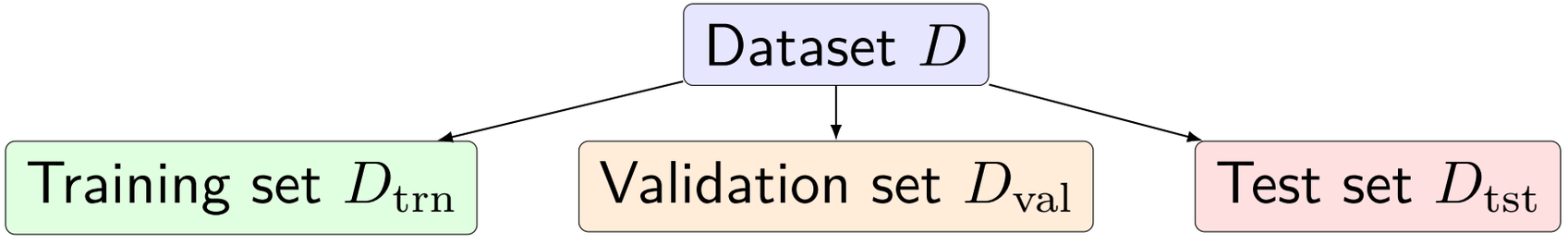
$$i^* = \arg\min_{i=1,\ldots,K} \left[\hat{R}(D_{\text{trn}}, h_i) + \varepsilon\left(\text{VCdim}(\mathcal{H}_\text{i}), \frac{1}{m}, \frac{1}{\delta} \right) \right]$$

Beautiful theory but not used in practice

- The bounds are loose – VC generalization bounds are worst-case bounds.
- The VC dimension is often unknown or hard to compute.
- Tighter bounds exist but still are not tight enough.

Hold-Out Set Approach for Model Selection

Main idea: Train on one part of the data, select on another part, test only at the end.



Train
predictor with each candidate $A \in \{A_1, \ldots, A_L\}$

Select
choose A_* with best validation score

Assess
estimate final performance only once

Typical split: 60/20/20 or 70/15/15 (depends on dataset size).

Hold-Out Set Approach for Model Selection

Algorithm:

1. Randomly partition available data D into three disjoint subsets:

$$D_{\text{trn}},\ D_{\text{val}},\ D_{\text{tst}}.$$

2. Select the best model based on validation error:

$$A_* = \underset{A \in \{A_1, \ldots, A_L\}}{\arg\min} \hat{R}(D_{\text{val}}, A(D_{\text{trn}}))$$

3. Retrain on training and validation data:

$$\hat{h} = A_*(D_{\text{trn}} \cup D_{\text{val}})$$

4. Compute the test error:

$$\hat{R}_{\text{test}} = \hat{R}(D_{\text{tst}}, \hat{h})$$

Hold-Out Set Approach for Model Selection

Advantages

- Simple and easy to explain
- Computationally cheap
- Natural separation between tuning and final testing
- Works well when the dataset is large

Limitations

- Wastes data for training when data are scarce
- Validation estimate may have high variance

Common pitfalls

- **Using the test set for tuning:** then it is no longer a valid test set.
- **Data leakage:** preprocessing must be fit on the training portion only.
- **Too many trials on validation set:** can overfit the validation process.
- **Ignoring class imbalance:** use stratified splitting in classification.

The K-fold cross-validation error

1. Randomly partition D into K disjoint, roughly equal-sized subsets (**folds**) $F_1, F_2, \ldots, F_K$.
2. For each fold $k = 1, \ldots, K$:
   (a) **Train** on $D \setminus F_k$ (all data except fold k):
   $$\hat{h}^{(-k)} = A(D \setminus F_k)$$
   (b) **Evaluate** on F_k (the held-out fold):
   $$r_k = \hat{R}(F_k, \hat{h}^{(-k)})$$
3. Average the K error estimates:
   $$\hat{R}_{\mathrm{CV}} = \frac{1}{K} \sum_{k=1}^{K} r_k$$

Every example is used for **both training and testing**, exactly once for testing.

The K-fold cross-validation error

Example: 5-fold cross-validation error

Fold 1:	Test	Train	Train	Train	Train	$\rightarrow r_1$
Fold 2:	Train	Test	Train	Train	Train	$\rightarrow r_2$
Fold 3:	Train	Train	Test	Train	Train	$\rightarrow r_3$
Fold 4:	Train	Train	Train	Test	Train	$\rightarrow r_4$
Fold 5:	Train	Train	Train	Train	Test	$\rightarrow r_5$

$$\hat{R}_{\mathrm{CV}} = \tfrac{1}{5}(r_1 + r_2 + r_3 + r_4 + r_5)$$

The CV error $\hat{R}_{\mathrm{CV}}$ estimates the expected error of a predictor trained by the algorithm A on n examples:

$$R(p, H) = \mathbb{E}_{(x,y)\sim p, D\sim p^n}[\ell(y, H(x, D))]$$

where

$$H(x, D) = \hat{h}(x) \quad \text{and} \quad \hat{h} = A(D)$$

The K-fold cross-validation procedure

Small K (e.g. $K = 2$)

- Each training set has $n/2$ points
- Model trained on less data $\Rightarrow$ worse fit
- $\hat{R}_{\mathrm{CV}}$ has **pessimistic bias** (overestimates true error of model trained on n points)
- Lower correlation between folds $\Rightarrow$ potentially lower variance
- Low compute

Large K (e.g. $K = n$, LOO)

- Each training set has $n-1$ points
- Nearly unbiased for the risk at sample size $n-1$
- Training sets overlap by $\frac{n-2}{n-1}$ $\Rightarrow$ fold scores are **highly correlated**
- High variance of $\hat{R}_{\mathrm{CV}}$ due to correlation
- High compute

Common choices: $K = 5$ or $K = 10$ as a practical compromise.

Stratified K-fold cross-validation

Problem: Random partitioning can create folds with unbalanced class distributions, especially with imbalanced classes or small datasets.

Solution: Stratified K-fold CV ensures each fold preserves the class proportions of the full dataset.

- For each class y, distribute its n_y samples across folds so each fold receives $\lfloor n_y/K \rfloor$ or $\lceil n_y/K \rceil$ samples of class y.
- Reduces variance of the CV estimator without introducing bias.
- Default for classification in `scikit-learn`'s `cross_val_score`.

Example: 100 samples, 10% positive, $K = 5$

- Unstratified: a fold might get 0 or 5 positives
- Stratified: each fold gets exactly 2 positives

Using K-fold CV for model selection

Algorithm:

1. Select the best model based on K-fold CV:

$$A_* = \underset{A \in \{A_1, \ldots, A_L\}}{\arg\min} \hat{R}_{\mathrm{CV}}(D, A)$$

2. Retrain on the entire training set: $\hat{h} = A_*(D)$

Common pitfall:

- The CV error $\hat{R}_{\mathrm{CV}}(D, A_*)$ is **not** an unbiased estimate of the true risk $R(p, \hat{h})$.
- The best model A_* was selected to minimize CV error $\Rightarrow \hat{R}_{\mathrm{CV}}(D, A_*)$ is optimistically biased; the bias grows with L and shrinks with n.

Nested cross-validation

Two loops with different jobs:

Outer loop (K_{out} folds): Estimates generalization error.

Inner loop (K_{in} folds within each outer-train): Model selection.

1. Randomly partition data D into K_{out} equal-sized folds $F_1, F_2, \ldots, F_{K_{\text{out}}}$.
2. For each outer fold k:
   (a) Find the best model: $A_k^* = \arg\min_{A \in \{A_1, \ldots, A_L\}} \hat{R}_{\text{CV}}(D \setminus F_k, A)$
   (b) Retrain: $\hat{h}^{(-k)} = A_k^*(D \setminus F_k)$
   (c) Evaluate: $r_k = \hat{R}(F_k, \hat{h}^{(-k)})$

$$\hat{R}_{\text{nested}} = \frac{1}{K_{\text{out}}} \sum_{k=1}^{K_{\text{out}}} r_k$$

Remark: Different outer folds may select **different** A_k^* which is expected and informative.

Nested cross-validation

Outer loop (fold 1 of 5)

Test	Train	Train	Train	Train

↓

Inner loop (K_{in}**-fold CV on outer train)**

Test	Train	Train	Train	Train
Train	Test	Train	Train	Train
Train	Train	Test	Train	Train
Train	Train	Train	Test	Train
Train	Train	Train	Train	Test

↓

Select best model A_1^*, retrain on full outer train

↓

Evaluate on outer test $\rightarrow r_1$

Using nested CV for model selection

Algorithm:

1. Compute the nested CV error $\hat{R}_{\text{nested}}$ for the dataset D and the algorithms $\{A_1, \ldots, A_L\}$, using K_{out} outer and K_{in} inner folds.
2. Select the best model based on K_{in}-fold CV on all data:

$$A_* = \underset{A \in \{A_1, \ldots, A_L\}}{\arg\min} \hat{R}_{\text{CV}}(D, A)$$

3. Retrain on the entire dataset: $\hat{h} = A_*(D)$

Summary:

- The algorithm outputs a predictor $\hat{h}$ trained on all data D using a reasonably good proxy of the best algorithm from $\{A_1, \ldots, A_L\}$.
- The algorithm outputs $\hat{R}_{\text{nested}}$ as an estimate of the true error $R(p, \hat{h})$; though it is pessimistically biased (each inner model trains on less data than $\hat{h}$).
- What about the confidence of this estimate?

Confidence intervals

Setting: We observe K measurements $r_1, r_2, \ldots, r_K$ drawn i.i.d. from a normal distribution with unknown mean μ and variance σ^2.

We compute the **sample mean** and **unbiased sample variance**:

$$\bar{r} = \frac{1}{K}\sum_{i=1}^{K} r_i, \qquad \hat{\sigma}^2 = \frac{1}{K-1}\sum_{i=1}^{K}(r_i - \bar{r})^2$$

Goal: Construct an interval $[\bar{r} - \delta,\ \bar{r} + \delta]$ that contains the true mean μ with a specified probability $1 - \alpha$ (e.g. 95%).

Remark: If σ^2 was known, then $\bar{r} - \mu$ has the normal distribution with zero mean and the variance σ/K. The challenge is that σ^2 must be estimated from the same data.

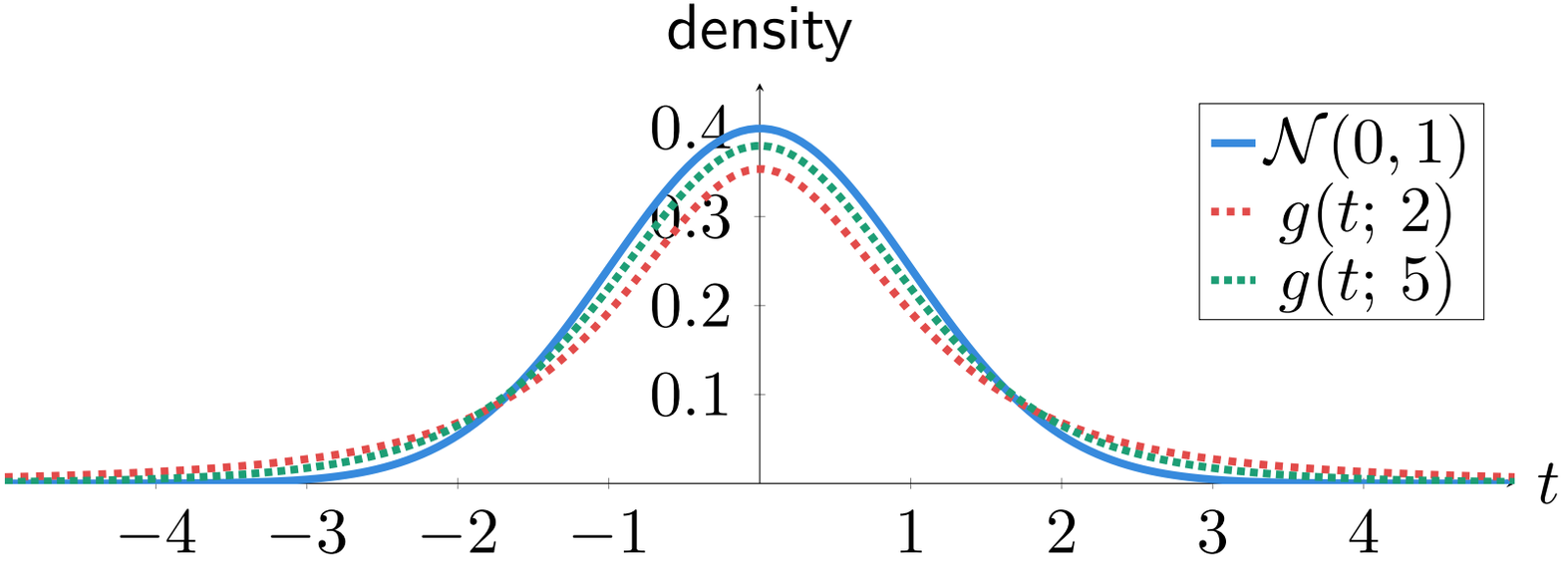
Student's t-distribution

The standardized quantity

$$t = \frac{\bar{r} - \mu}{\hat{\sigma}/\sqrt{K}}$$

follows a **Student's t-distribution** with $\nu = K - 1$ **degrees of freedom**, which has density:

$$g(t;\, \nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

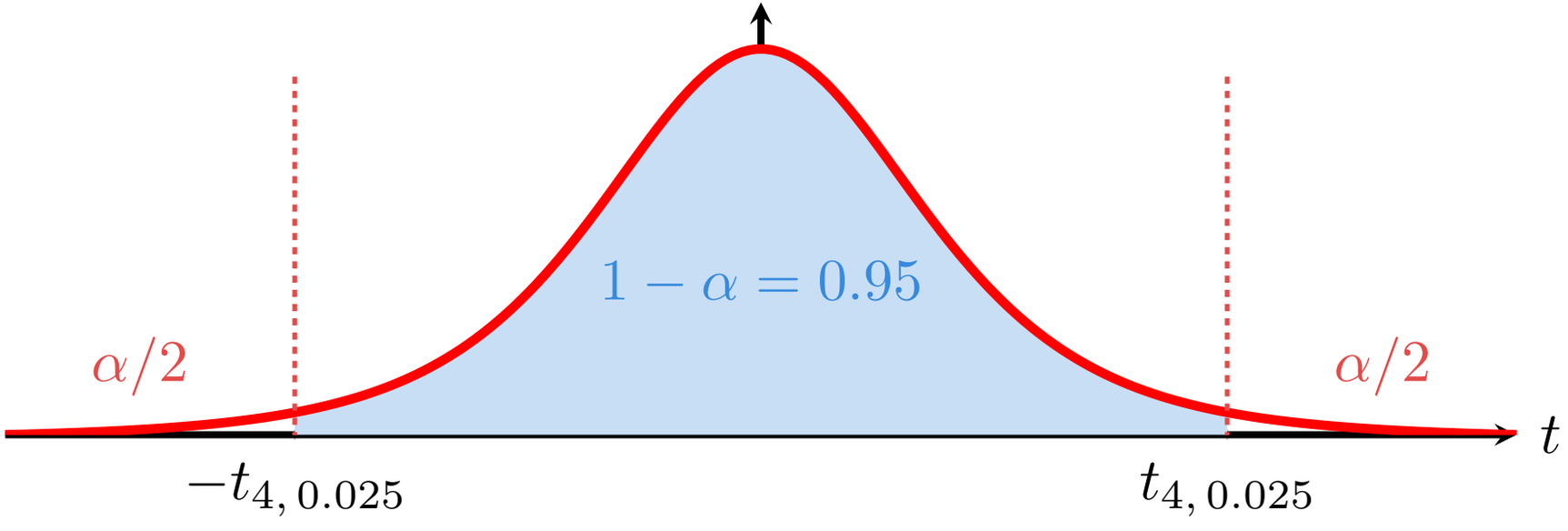


Constructing the confidence interval

Since $t = \dfrac{\bar{r} - \mu}{\hat{\sigma}/\sqrt{K}} \sim g(t;\, K{-}1)$, we find the critical value $t_{K-1,\,\alpha/2}$ such that:

$$\mathbb{P}\left(-t_{K-1,\,\alpha/2} \;\leq\; \frac{\bar{r} - \mu}{\hat{\sigma}/\sqrt{K}} \;\leq\; t_{K-1,\,\alpha/2} \right) = 1 - \alpha$$

Example: $g(t;\, \nu{=}4)$, 95% central region



The interval $\bar{r} \pm t_{4,\,0.025} \cdot \hat{\sigma}/\sqrt{K}$ captures μ with probability 0.95.

Constructing the confidence interval

By rearranging the terms in

$$\mathbb{P}\left(-t_{K-1,\,\alpha/2} \;\leq\; \frac{\bar{r}-\mu}{\hat{\sigma}/\sqrt{K}} \;\leq\; t_{K-1,\,\alpha/2} \right) = 1-\alpha$$

we get:

$$\boxed{\mu \in \left[\bar{r} \pm t_{K-1,\,\alpha/2} \cdot \frac{\hat{\sigma}}{\sqrt{K}} \right] \quad \text{with probability} \quad 1-\alpha}$$

The interval has three components:

- Estimated mean: $\bar{r} = \frac{1}{K}\sum_{k=1}^{K} r_k$ of μ
- Estimated std. deviation: $\hat{\sigma}^2 = \frac{1}{K-1}\sum_{k=1}^{K}(r_k - \bar{r})^2$
- Critical value: $t_{K-1,\,\alpha/2}$ (scipy.stats.t.ppf)

CI for hold-out test set

Given a **fixed, already-trained** predictor $\hat{h}$ and an independent i.i.d generated test set $D = ((x_i, y_i) \in \mathcal{X} \times \mathcal{Y} \mid i = 1, \ldots, n)$, compute the per-example losses:

$$r_i = \ell\big(y_i,\, \hat{h}(x_i)\big), \qquad i = 1, \ldots, n$$

and t-distribution based confidence interval:

$$R(p, \hat{h}) \in \left[\hat{R}(D, \hat{h}) \;\pm\; t_{n-1,\,\alpha/2} \cdot \frac{\hat{\sigma}_\ell}{\sqrt{n}}\right] \quad \text{with probability} \quad 1 - \alpha$$

where $\hat{R}(D, \hat{h})$ is the test error and $\hat{\sigma}_\ell^2 = \frac{1}{n-1} \sum_{i=1}^{n} (r_i - \hat{R}(D, \hat{h}))^2$.

The CI is valid provided:

1. i.i.d. assumption: since $\hat{h}$ is fixed, the r_i are **i.i.d.**
2. Normality: for regression problems or classification with moderate error rates and $n \geq 50$, the t-interval is a good approximation.

The validity of t-distribution CI for the cross-validation

The t-distribution based CI is exact when:

1. **Independence**: the estimates $r_1, \ldots, r_K$ are mutually independent
   - Violated in cross-validation (training sets overlap).
2. **Identical distribution**: all $r_1, \ldots, r_K$ have the same distribution
   - Approximately true for equal-sized folds.
3. **Normality**: the random variables $r_1, \ldots, r_K$ are normally distributed
   - Reasonable for large K due to the Central Limit Theorem.

Independence is the assumption that breaks most severely in cross-validation.

CI for K-fold cross-validation

In K-fold CV, the fold scores $r_1, \ldots, r_K$ are **positively correlated** because training sets overlap.

The true variance of the mean is:

$$\mathrm{Var}(\bar{r}) = \frac{\sigma^2}{K} \underbrace{\big(1 + \rho(K-1)\big)}_{>1 \text{ when } \rho>0}$$

The standard t-interval uses only $\hat{\sigma}^2/K$, ignoring the $\rho(K-1)$ term.

Consequence:

- The standard interval is **too narrow** $\Rightarrow$ actual coverage is **below** $1-\alpha$ (e.g. 80% instead of 95%).

Fix (Nadeau & Bengio, 2003):

- Replace $\hat{\sigma}^2/K$ with $\hat{\sigma}^2 \cdot (1/K + n_{\text{test}}/n_{\text{train}})$.
- Derived for Monte Carlo cross-validation (independent random train/test splits); applied as a heuristic for standard K-fold CV.

Corrected CI for K-fold cross-validation (Nadeau & Bengio, 2003)

1. The CV procedure gives the fold errors: $r_1, r_2, \ldots, r_K$.
2. Compute the mean (i.e. the CV error):

$$\hat{R}_{\mathrm{CV}} = \frac{1}{K} \sum_{k=1}^{K} r_k$$

3. Compute the sample variance of the fold errors:

$$\hat{\sigma}^2 = \frac{1}{K} \sum_{k=1}^{K} (r_k - \hat{R}_{\mathrm{CV}})^2$$

4. The approximate CI with coverage probability $1 - \alpha$:

$$R(p, \hat{h}) \in \hat{R}_{\mathrm{CV}} \pm t_{K-1,\,\alpha/2} \cdot \sqrt{\hat{\sigma}^2 \cdot \left(\frac{1}{K} + \frac{1}{K-1} \right)}$$

Remark: This is not an exact CI (the t-distribution is approximate due to correlated folds), but it has been found to work well empirically.

Summary

- **SRM:** theoretically elegant but impractical (loose VC bounds, unknown VC dimension).
- **Hold-out approach:** simple but wastes data; beware of test set contamination and data leakage.
- K**-fold CV:** every example used for both training and testing; choice of K involves a bias–variance tradeoff ($K{=}5$ or $K{=}10$ typical); use **stratification** for classification.
- **Nested CV:** separates model selection (inner loop) from performance estimation (outer loop); avoids optimistic bias of reporting $\hat{R}_{\mathrm{CV}}$ of the selected model.
- **Confidence intervals:**
  - Hold-out test set: standard t-interval applies (i.i.d. losses).
  - K-fold CV: fold scores are correlated $\Rightarrow$ naive CI is too narrow $\Rightarrow$ use the Nadeau & Bengio corrected variance $\hat{\sigma}^2 \cdot (1/K + 1/(K{-}1))$.