

Lecture 8a

Reproducibility

Outline

Motivations

- ▶ Repeatability vs. reproducibility
- ▶ Examples

Testing reproducibility

- ▶ Holdout procedure
- ▶ Cross-validation procedure

Improving the model

- ▶ Regularization
- ▶ Vapnik-Chervonenkis theory

Part 1

Motivations

Motivations

Examples of data science results:

- ▶ “The principal component $\mathbf{u}_1 = (0.1, 0.6)$ explains 60% of the variance in the data.”
- ▶ “Clusters $\boldsymbol{\mu}_1 = (0.2, 0.5)$, $\boldsymbol{\mu}_2 = (1.0, -0.6)$, $\boldsymbol{\mu}_3 = (-0.4, 0.8)$ explain 75% of the variance in the data.”
- ▶ “With x_1, x_2 and y being the two data modalities, the linear combination $z = 1.1 \cdot x_1 - 0.4 \cdot x_2$ correlates with variable y with a coefficient of 0.92.”
- ▶ “The hyperplane $\mathcal{H} = \{0.5 \cdot x_1 - 2.5 \cdot x_2 = 1 \mid \mathbf{x} \in \mathbb{R}^2\}$ separates the data from group 1 and group 2 with error rate 5%.”

Question: Are these results reproducible/repeatable?

Motivations

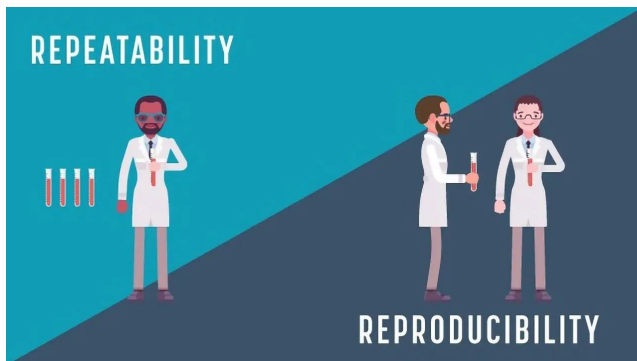


Image source: R. Mackenzie, *Repeatability vs. Reproducibility*, Technology Networks (2021)

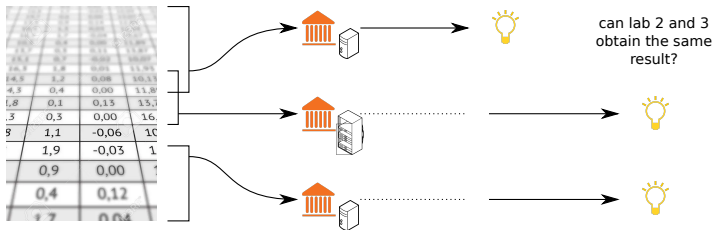
- ▶ Most computer-based experiments are *repeatable* (assuming enough compute power and availability of code/data).
- ▶ The result of an experiment is *reproducible* if it is not affected substantially by a *limited perturbation* of the experimental setup, e.g. resampling the data, changing the seed, minor variations in the experimental protocol.

Motivations

true data-generating process (e.g. whole-population)

labs access different data and run different systems

lab 1 conduct an experiment and get some result



- ▶ Before Lab 1 publishes the result of its experiment, it wishes to verify internally whether it is reproducible.
- ▶ An approach for Lab 1 is to test whether the outcome remains the same under small variations of data and experimental protocol (i.e. simulating what might occur in Labs 2 and 3).

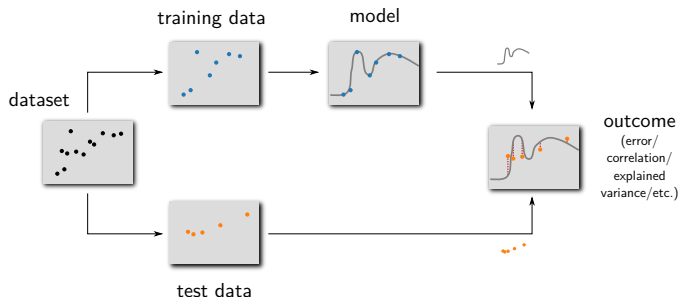
Part 2

Testing Reproducibility

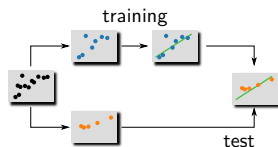
Holdout Procedure

Idea:

- ▶ Artificially simulate the scenario of another lab reproducing the experiment, by randomly partitioning the available data into *training data* and *test data*.
- ▶ Learn the model on the training data, and report the result (e.g. correlation, error, explained variance, etc.) on the test data.



Example 1: PCA



Holdout procedure:

- ▶ Learn the first principal component \mathbf{u}_1 on the *training data*.

$$\mathbf{C}_{\text{train}} = \text{Cov}(\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{train}}) \quad (\text{estimate the covariance})$$

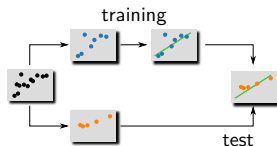
$$\mathbf{u}_1 = \arg \max_{\mathbf{u}} \mathbf{u}^{\top} \mathbf{C}_{\text{train}} \mathbf{u} \quad (\text{extract the PCA-1 direction})$$

- ▶ Measure the variance explained by \mathbf{u}_1 on the *test data*.

$$\mathbf{C}_{\text{test}} = \text{Cov}(\mathbf{X}_{\text{test}}, \mathbf{X}_{\text{test}}) \quad (\text{estimate the covariance})$$

$$\lambda_{\text{test}} = \mathbf{u}_1^{\top} \mathbf{C}_{\text{test}} \mathbf{u}_1 \quad (\text{compute the explained variance})$$

Example 2: Least Squares Regression



Holdout procedure:

- ▶ Train the regression model on the *training data*:

$$C_{xx} = \text{Cov}(X_{\text{train}}, X_{\text{train}}) \quad (\text{estimate the auto-covariance})$$

$$C_{xy} = \text{Cov}(X_{\text{train}}, Y_{\text{train}}) \quad (\text{estimate the cross-covariance})$$

$$\mathbf{w} = C_{xx}^{-1} C_{xy}$$

$$b = \mathbb{E}[Y_{\text{train}}] - \mathbf{w}^{\top} \mathbb{E}[X_{\text{train}}]$$

- ▶ Evaluate the error of the model (\mathbf{w}, b) on the *test data*:

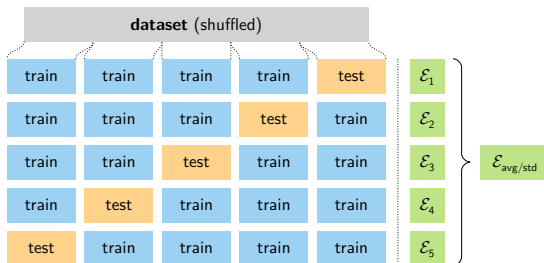
$$\hat{Y}_{\text{test}} = \mathbf{w}^{\top} X_{\text{test}} + b \quad (\text{predict the test data})$$

$$\mathcal{E}_{\text{test}} = \mathbb{E}[(\hat{Y}_{\text{test}} - Y_{\text{test}})^2] \quad (\text{compute the prediction error})$$

Cross-Validation Procedure

Observations:

- ▶ If splitting data evenly into a training and test set, there is not much data left for training and evaluation respectively → less good models and noisier error estimates.
- ▶ The problem can be addressed by *cross-validation*, i.e. repeating the analysis over multiple splits, and averaging the results.



- ▶ Because of averaging reduces the noise of error estimates, it is reasonable to allocate most data for training (e.g. 80%).

Training Error vs. Test Error

Example: Fake reviews detection ('Restaurant Dataset')

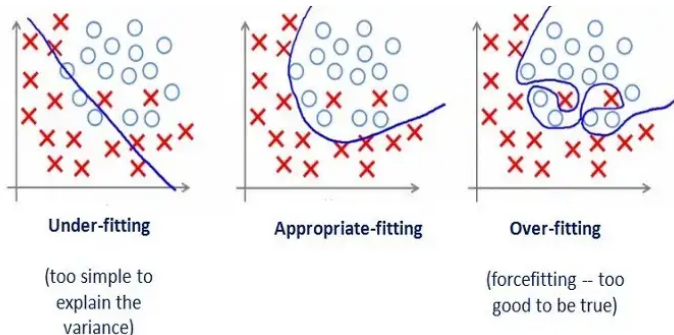
Classifier	Training Error	Test Error
Decision Tree	0.102	0.455
Random Forest	0.057	0.318
SVM	0.239	0.409
XGBT	0.182	0.364
MLP	0.0	0.318
Bagging Ensemble (SVM)	0.227	0.318
Adaboost Ensemble (SVM)	0.250	0.273
Bagging Ensemble (MLP)	0.034	0.318
Adaboost Ensemble (MLP)	0.068	0.227

Image source: <https://doi.org/10.48550/arXiv.2006.07912>

Observations:

- ▶ Typically large gap between training and test error
- ▶ The model that performs best on training data (here MLP) does not necessarily perform best on the test set.
- ▶ This is due to overfitting.

The Problem of Overfitting



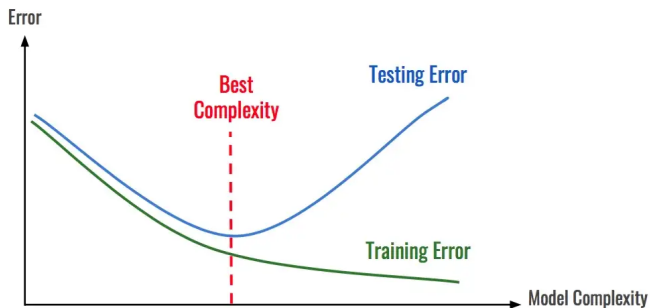
Overfitting:

- ▶ The model tries to extract more variations than can possibly be recovered from the input features.
- ▶ While it lowers the training error, the model will perform worse on out-of-sample data.

Part 3

Controlling Modeling Complexity

Controlling Modeling Complexity



Hypothesis:

- By controlling the complexity of the model, one can reach a good tradeoff between underfitting and overfitting, and perform better on the test data.

Controlling Modeling Complexity

Remark:

- ▶ We need a mechanism within the learning procedure to control model complexity, so that we can adjust the learning procedure to avoid overfitting.

Two main approaches:

- ▶ *Feature selection*: Define a limited set of features \mathcal{I} (with $|\mathcal{I}| < d$) that the machine learning model can use for its training/predictions. The lower the number of features, the less likely the model will overfit.
- ▶ *Model regularization*: Penalize models that are overly sensitive to small variations of the data, e.g. for a linear model, by imposing the constraint $\|\mathbf{w}\|^2 \leq R$ on the weights. The lower the hyperparameter R , the less likely the model will overfit.

Example: Least Squares Regression

Recall:

- ▶ The least square regression objective is given by:

$$\min_{\mathbf{w}} \mathbb{E}[\|\mathbf{w}^\top \mathbf{x} - y\|^2]$$

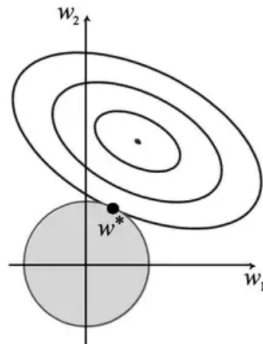
where we assume $\mathbb{E}[\mathbf{x}] = \mathbf{0}$ and $\mathbb{E}[y] = 0$ (i.e. centered data).

Idea:

- ▶ Optimize the objective above subject to the constraint that the model should have limited sensitivity to the data, i.e.

$$\|\mathbf{w}\|^2 \leq S$$

- ▶ The resulting constrained optimization problem can be addressed with the framework of Lagrange multipliers / KKT conditions.



Regularized Least Square Regression

Step 1: Build the Lagrangian:

$$\mathcal{L}(\mathbf{w}, \lambda) = \frac{1}{2} \mathbb{E}[\|\mathbf{w}^\top \mathbf{x} - y\|^2] + \frac{1}{2} \lambda (\|\mathbf{w}\|^2 - S)$$

Step 2: Verify Slater's conditions and apply the KKT conditions.

- ▶ The KKT stationarity condition gives us:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \underbrace{\mathbb{E}[\mathbf{x}\mathbf{x}^\top]}_{C_{xx}} \mathbf{w} - \underbrace{\mathbb{E}[\mathbf{x}y]}_{C_{xy}} + \lambda \mathbf{w} \stackrel{\text{def}}{=} \mathbf{0}$$

And therefore,

$$\mathbf{w} = (C_{xx} + \lambda I)^{-1} C_{xy}$$

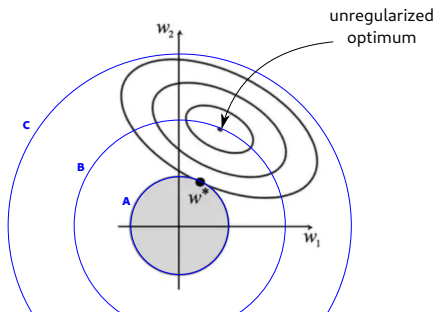
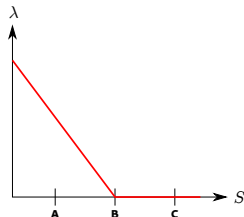
- ▶ The KKT complementary slackness condition gives us:

$$\lambda \cdot (\|\mathbf{w}\|^2 - S) = 0$$

If $\lambda > 0$, then $\|\mathbf{w}\|^2 = S$. If $\|\mathbf{w}\|^2 < S$, then $\lambda = 0$ (i.e. standard regression).

Regularized Least Square Regression

Interpretation of λ and S :



Idea:

- ▶ Don't try to resolve λ from S . Since we want to try many parameter S anyways, just try various parameters λ (e.g. between 10^{-7} and 10^{12} , logarithmically spaced).

Regularized Least Square Regression

Example: Least squares regression on mtcars dataset.

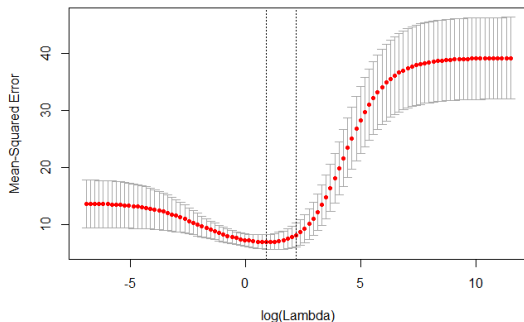


Image source:
<https://www.datacamp.com/tutorial/tutorial-ridge-lasso-elastic-net>

Observation:

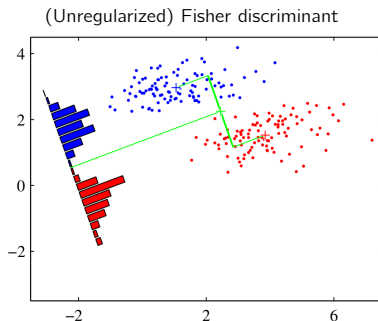
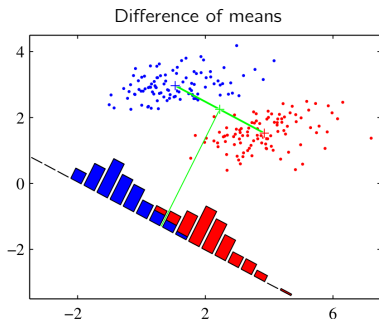
- ▶ Test error is indeed minimized for some intermediate value of λ .

Note:

- ▶ We cannot select the best parameter λ from this curve. Otherwise, we would overfit on the test set and bias our result favorably. We need a separate subset of the data for model selection (→ Lecture 8b).

Regularized Fisher Discriminant

- ▶ Like for least squares regression, the Fisher discriminant with regularization can be expressed as $\mathbf{w} = (C_{xx} + \lambda I)^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)$.
- ▶ Increasing λ makes the inverse covariance term more similar to an identity, and the direction \mathbf{w} becomes aligned with that of the difference of means ($\mathbf{w} = \boldsymbol{\mu}_2 - \boldsymbol{\mu}_1$).



Part 4

Vapnik-Chervonenkis Theory

Formalizing the Learning Problem

Observation:

- ▶ So far, we have observed *empirically* that regularization, by reducing model complexity, improves the performance of the model on the test data.

Question:

- ▶ Can we connect formally model complexity to the true (i.e. reproducible) error of a model?

Preliminaries:

- ▶ Given some ground-truth joint distribution $p(\mathbf{x}, y)$ over the input $\mathbf{x} \in \mathbb{R}^d$ and the output $y \in \mathbb{R}$, we are able to formulate the true error of some predictor $f : \mathbb{R}^d \rightarrow \mathbb{R}$ by the integral:

$$\mathcal{E}_{\text{true}}(f) = \int \ell(f(\mathbf{x}), y) dp(\mathbf{x}, y)$$

where $\ell(f(\mathbf{x}), y)$ is some *loss function*, e.g. $1_{\{\text{sign}(f(\mathbf{x})) \neq y\}}$ for classification, or $(f(\mathbf{x}) - y)^2$ for regression.

- ▶ In practice, p is unknown and we only know $\mathcal{E}_{\text{train}}(f) = \frac{1}{N} \sum_{i=1}^N \ell(f(\mathbf{x}_i), y_i)$.

VC Theory

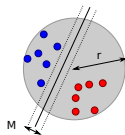
Vapnik-Chervonenkis theory provide a connection between training and true error in the context of classification. Specifically, it shows that with probability $1 - \delta$,

$$\mathcal{E}_{\text{true}}(f) - \mathcal{E}_{\text{train}}(f) \leq \sqrt{\frac{h_{\mathcal{F}} \cdot \left(\log \left(\frac{2N}{h_{\mathcal{F}}} \right) + 1 \right) - \log \left(\frac{\delta}{4} \right)}{N}}.$$

where $h_{\mathcal{F}}$ is the VC dimension measuring the complexity of the class of models \mathcal{F} which our model is selected from, specifically, the *maximum* number of data points that the function class \mathcal{F} can *always* classify in *any* possible way.

Examples:

- ▶ $f(\mathbf{x}) = \text{sign}(\sin(\mathbf{w}^T \mathbf{x} + b))$
 $h_{\mathcal{F}} = \infty$
- ▶ $f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$
 $h_{\mathcal{F}} = d + 1$
- ▶ $f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$ with $\frac{1}{\|\mathbf{w}\|} = \frac{M}{2}$
 $h_{\mathcal{F}} \leq \min \left\{ d + 1, \left\lceil \frac{4r^2}{M^2} \right\rceil + 1 \right\}$



\Rightarrow Large datasets, linear models, few dimensions, and margins, all demonstrably contribute to make the training and true error similar.


VC Dimension

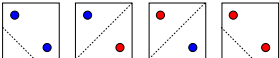
Recall:

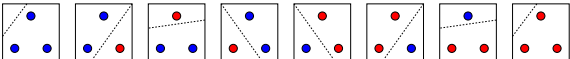
- ▶ The VC-dimension $h_{\mathcal{F}}$ is the *maximum* number of data points that the function class \mathcal{F} can *always* classify in *any* possible way.


Example:

- ▶ Calculating the VC dimension of the set of linear classifiers in \mathbb{R}^2 :

$VC_{\mathcal{F}} \geq 1?$  yes

$VC_{\mathcal{F}} \geq 2?$  yes

$VC_{\mathcal{F}} \geq 3?$  yes

$VC_{\mathcal{F}} \geq 4?$  no $\Rightarrow VC_{\mathcal{F}} = 3$

Summary

Summary

- ▶ For scientific results to be accepted, they need to be *reproducible* by other actors.
- ▶ Other actors may not have exactly the same data or environment. Hence, it is not sufficient that the outcome is reproducible under exactly the same conditions (aka. repeatability), it should also hold for *reasonable variations* of those conditions (e.g. slightly different data).
- ▶ *Holdout* and *cross-validation* are flexible tools to test how well results obtained with a model continue to hold on different data.
- ▶ Failure to reproduce results can often be attributed to *overfitting* (a model that captures too many variations in the training data).
- ▶ Overfitting can be prevented by applying *regularization* techniques.
- ▶ The Vapnik-Chervonenkis theory formally connects the complexity of the model and its capabilities to predict well the true data.