

DSC 240: Machine Learning

ML basics (continue) / linear Algebra

Jan 11, 2025

Instructor: Prof. Yu-Xiang Wang

Today

- Finishing ML Basics
- Holdout and Cross-validation
- Linear Algebra Review
- Standard Notations in Machine Learning

Recap: Elements of Machine Learning

- Input Space / Feature / Feature vector
- Label Space / Labels
- Models / Hypothesis Class / Hypothesis
- Goal for machine learning: a selection problem
 - Requires a criterion for selection.

Recap: Key terminology (for evaluation)

$$\text{False positive rate (FPR)} = \frac{FP}{N} = \alpha$$

$$\text{Accuracy} = \frac{TP+TN}{P+N} = \left(\frac{P}{P+N}\right) TPR + \left(\frac{N}{P+N}\right) TNR$$

$$\text{False negative (miss) rate (FNR)} = \frac{FN}{P} = \beta$$

$$\text{Error rate} = \frac{FP+FN}{P+N}$$

$$\text{True positive rate (TPR)} = \frac{TP}{P} = \text{Sensitivity} = \text{Recall} = 1 - \beta$$

$$\text{Precision} = \frac{TP}{\hat{P}}$$

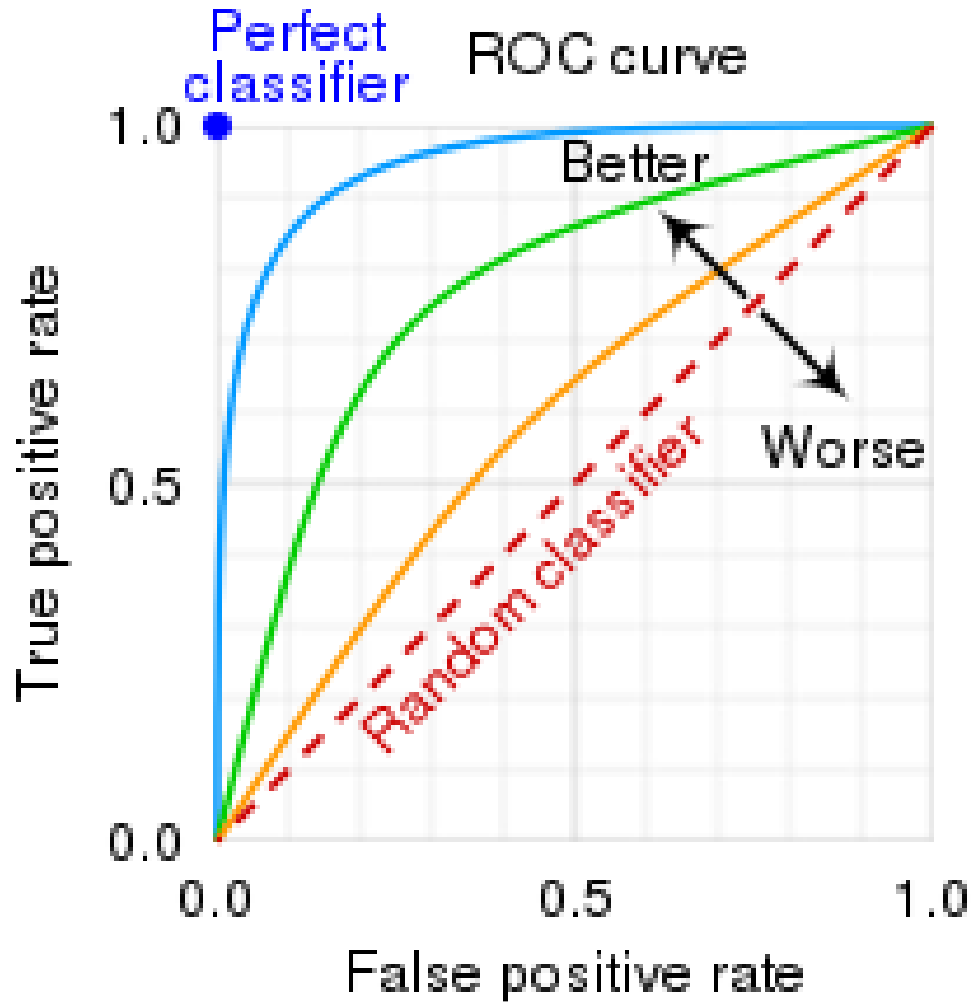
$$\text{True negative rate (TNR)} = \frac{TN}{N} = \text{Specificity} = 1 - \alpha$$

$$\text{F1 score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \cdot TP}{P + \hat{P}}$$

Predicted class
 \hat{y}

		Actual class y		
		1	0	
1	1	TP	FP	<i>Estimated positive \hat{P}</i>
	0	FN	TN	
		<i>Positives P</i>	<i>Negative N</i>	TOTAL

Response Operator Characteristic (ROC) curve



Single number summary of any
“**score function**”

AUC: **A**rea **U**nder the ROC **C**urve

What are there good ranges of AUC values?

(illustration from wikipedia)

Checkpoint: Performance metrics

(a) Accuracy

(b) Precision

(c) Recall

(d) F1 score

(e) Area under the ROC

If “error” is the metric of interest, then how to learn linear classifier in a non-linearly separable case?

- Training data:

$$(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$

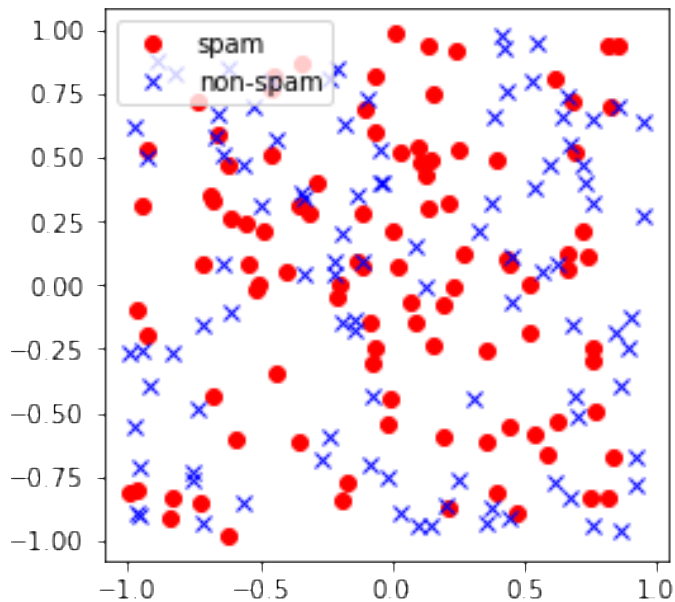
- Solving the following optimization problem:

$$\min_{w \in \mathbb{R}^d} \text{Error}(w) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}(h_w(x_i) \neq y_i)$$

- Learning: Find the linear classifier that makes **the smallest number of mistakes** on the training data.

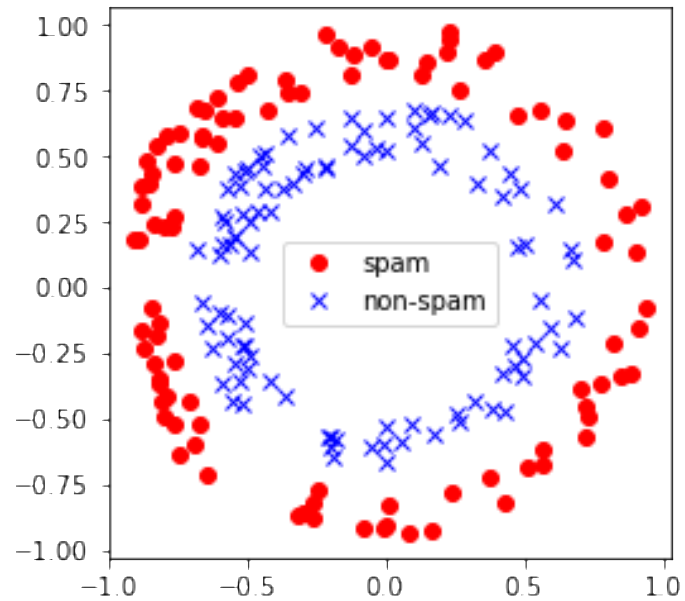
What happens if the linear classifier with the smallest number of mistakes still makes a mistake 49% of the time?

Case 1:



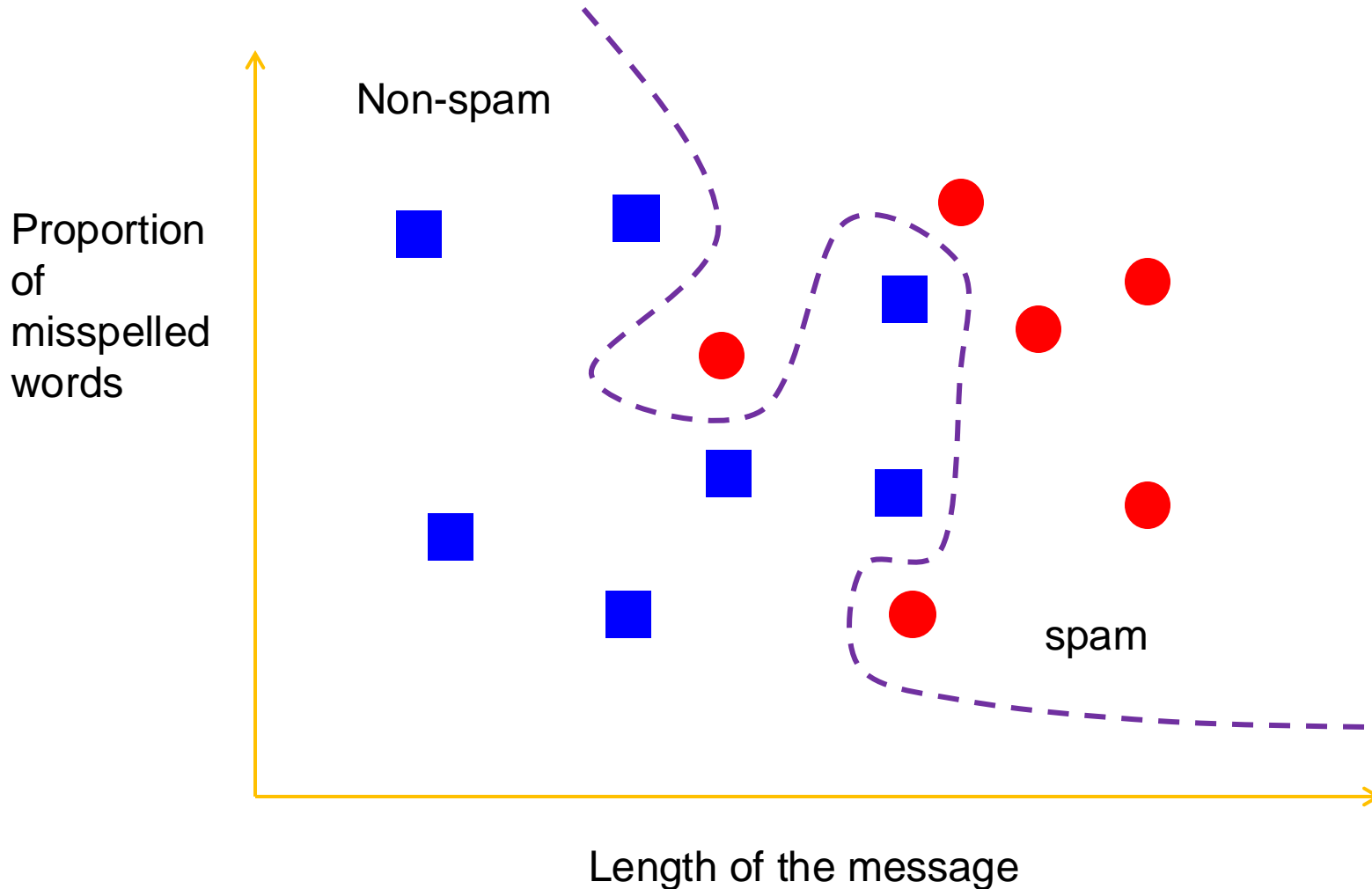
There is no information about the label in the features.
No classifiers are able to do well.

Case 2:

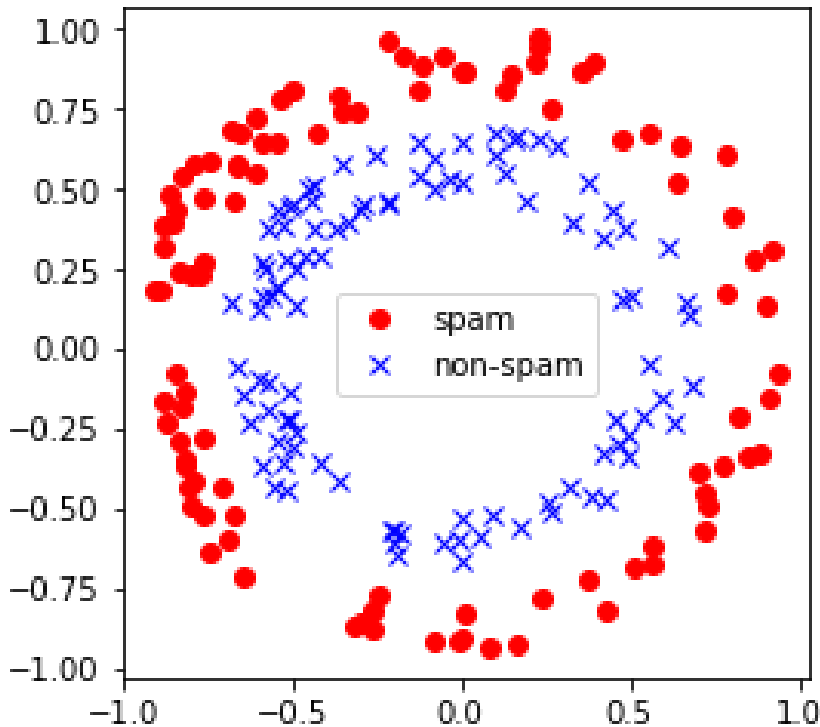


There are some nonlinear classifier that works. But no linear classifiers will do better than chance.

Going to higher dimensions? Maybe we can also allow non-linear decision boundaries?



Example: Feature transformation.



What we can do:

$$(\tilde{x}_1, \tilde{x}_2) = \left(\sqrt{x_1^2 + x_2^2}, \arctan(x_2/x_1) \right)$$

In the redefined space, the two classes are now linearly separable.

We will learn how to do this systematically in [Lecture 12 - 13](#)

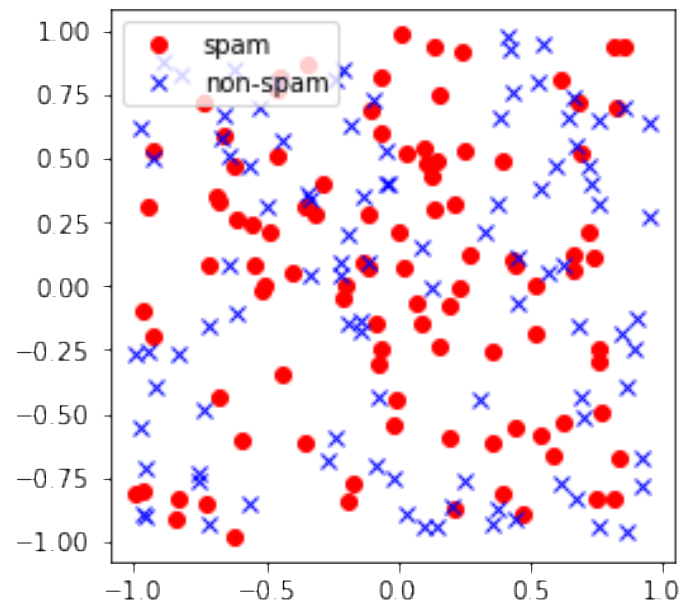
Nonparametric classifiers

- Increasing the complexity of the classifier as we get more data
- For example:
 - We can use the entire training dataset as “free parameters” of the classifier.
 - k-Nearest Neighbor
 - Kernel methods (lifting to infinite dimensional space)
 - Neural networks (design a model for a fixed data size)

Question: What is the classification error of 1-NN classifiers?

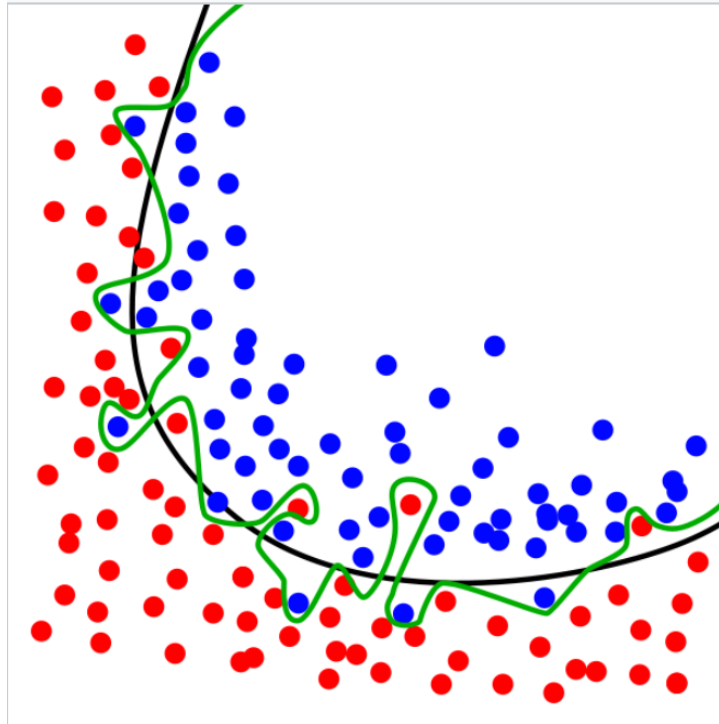
We can make the classifiers arbitrarily accurate... with 1-NN classifier; or with bigger and bigger neural networks.

- Even if the data look like:



- **What went wrong?**

The problem of Overfitting



The green line represents an overfitted model. While the green line best follows the training data, it is too dependent on that data and it is likely to have a higher error rate on new unseen data.

Another example of overfitting in the problem of “Curve Fitting” (We will see it again in Lecture 6)

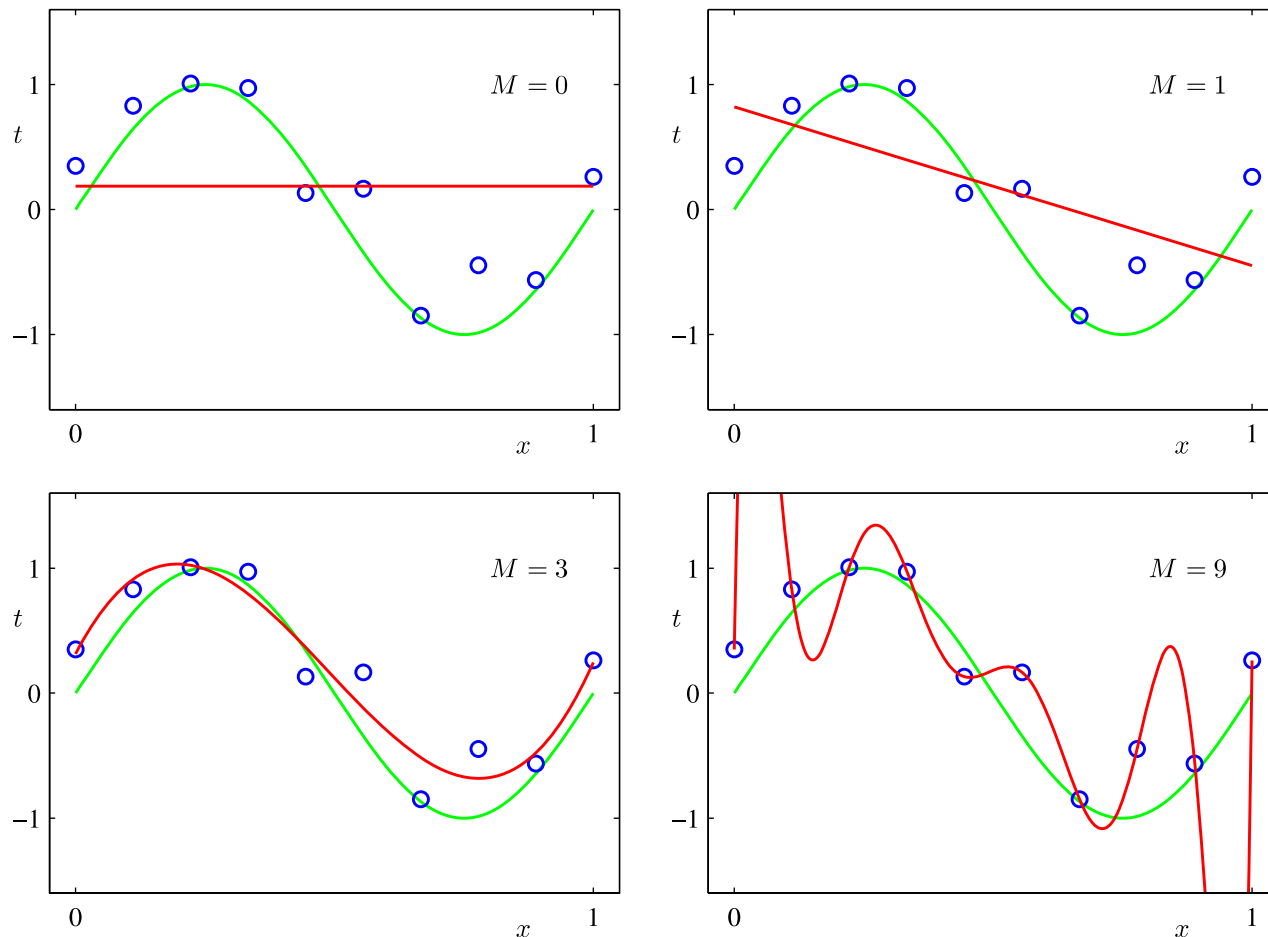


Figure 1.4 Plots of polynomials having various orders M , shown as red curves, fitted to the data set shown in Figure 1.2.

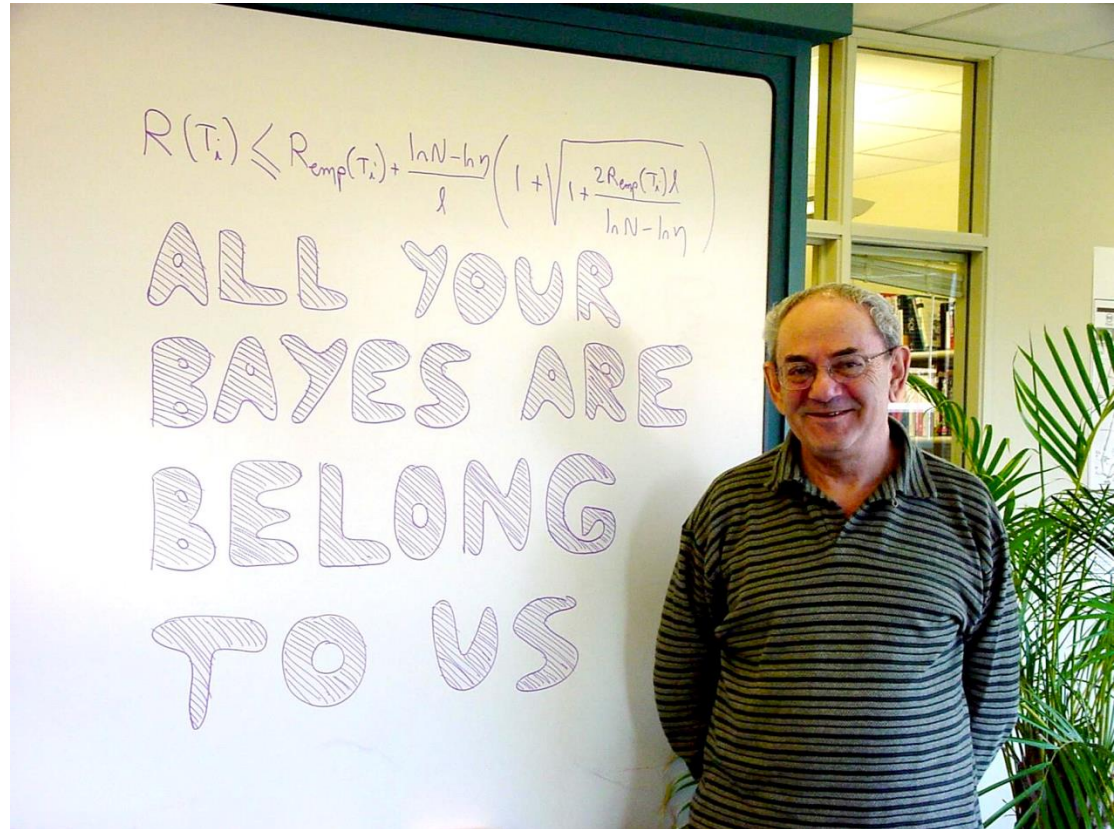
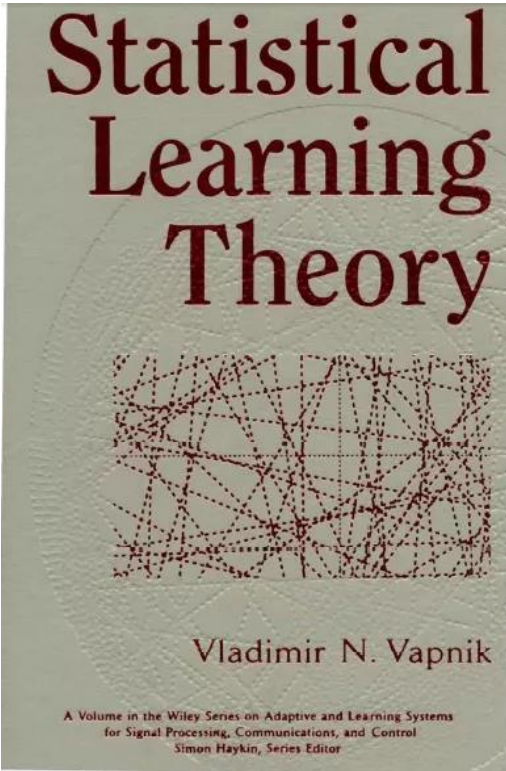
Fundamental question in ML:

The learner **only sees the “training data”** but ideally **wants to do well on “new data”!**

- The problem of generalization.
- All performance metrics we learned before should be calculated on the new data.
- The issue is that we don't have access to new data...

Statistical Learning Theory

TL;DR: Prove that the $|\text{train error} - \text{test error}| \rightarrow 0$, thus showing that ML works.

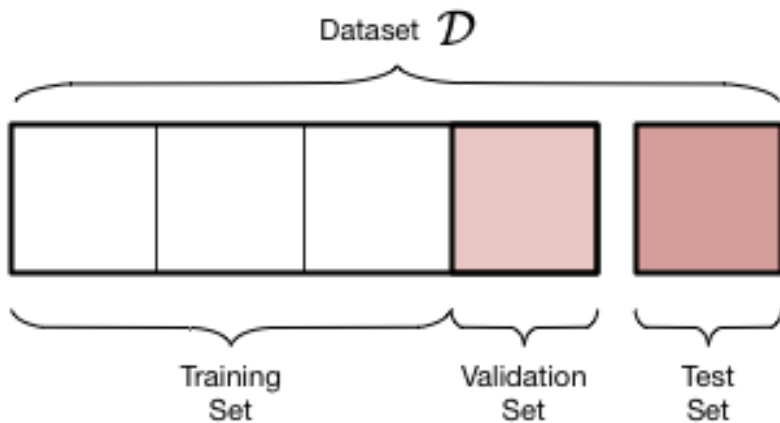


Key Assumption: The data is i.i.d. (Independent and Identically Distributed)

Closely related to Empirical Process Theory, Computational Learning Theory.

What do we do in practice?

Data Splitting methods: Holdout



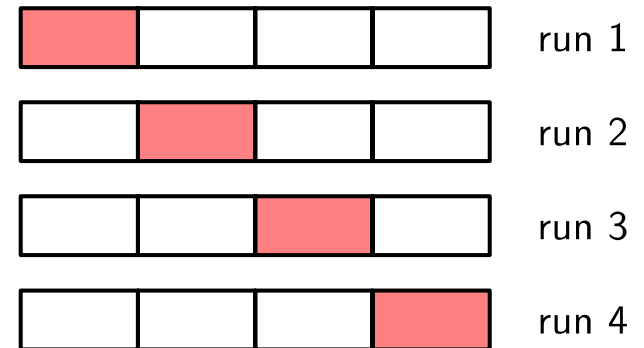
Validation set is used for **hyperparameter search** (also known as **model-selection**):

- choosing decision tree vs. linear classifier
- Select features, tune hyperparameters

Test set is used only once to report the final results.

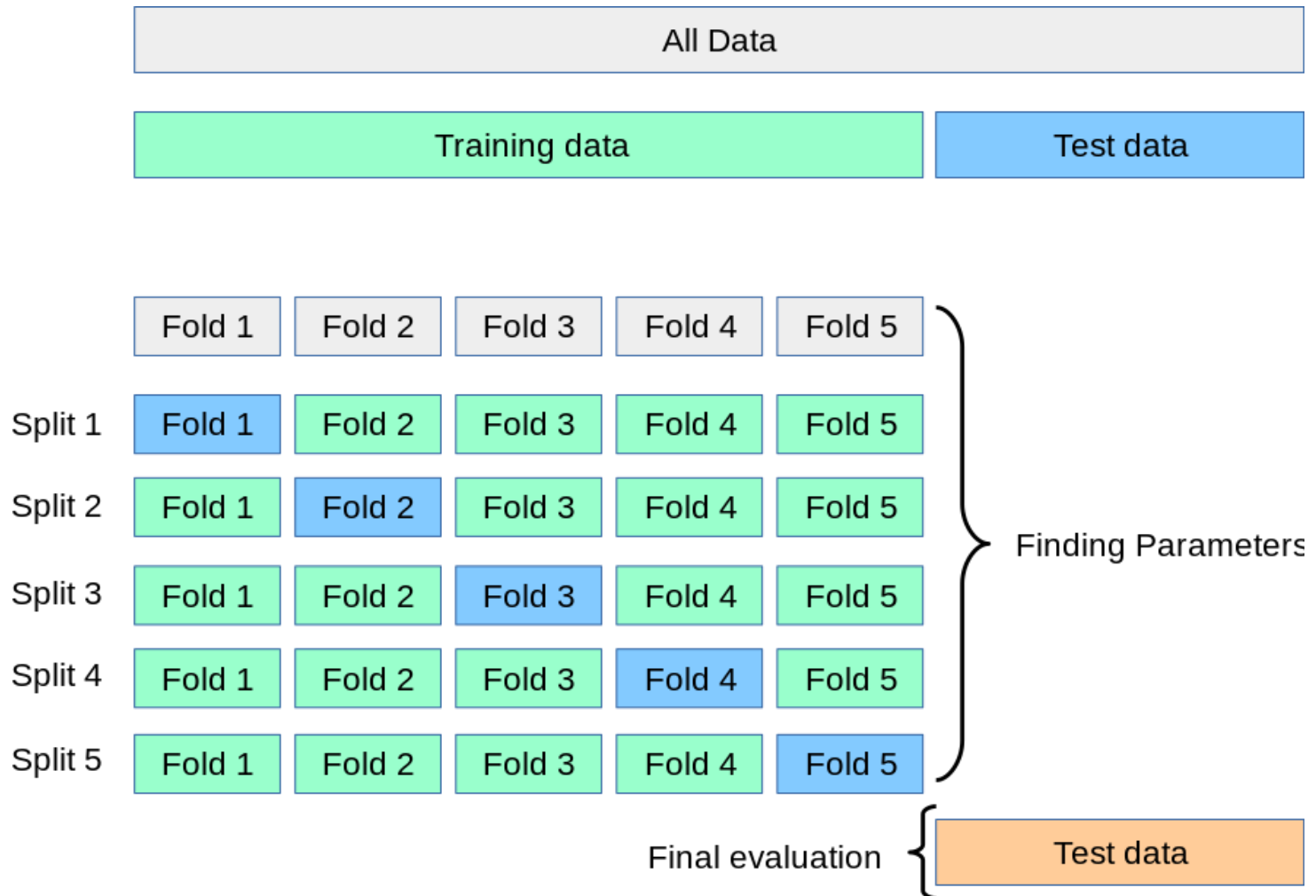
Data Splitting methods: Cross-Validation

Figure 1.18 The technique of S -fold cross-validation, illustrated here for the case of $S = 4$, involves taking the available data and partitioning it into S groups (in the simplest case these are of equal size). Then $S - 1$ of the groups are used to train a set of models that are then evaluated on the remaining group. This procedure is then repeated for all S possible choices for the held-out group, indicated here by the red blocks, and the performance scores from the S runs are then averaged.



- Pros:
 - No assumption on the data generating distributions, except iid.
 - Do not waste data, comparing to holdout.
- Cons:
 - It evaluates a model trained on $(S-1)/S$ fraction of the data
 - Computation cost = $O(S * \text{number of models to select from})$

A practical workflow for machine learning



Case study: Biotech startup

- Problem (true story, according to Alex Smola)
 - Biotech startup wants to detect a type of cancer
 - Easy to get blood samples from sick patients
 - Hard to get blood samples from healthy ones.
- Solution?
 - Get blood samples from university students
 - Use them as healthy reference.
 - Classifier gets 100% accuracy.
- What is wrong?

The problem of distribution shift

Training data



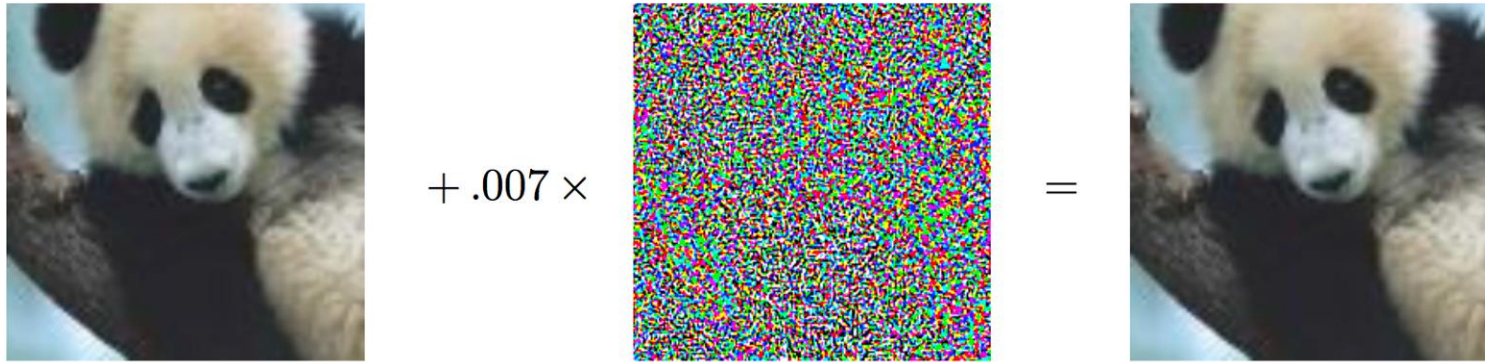
Test data received during:

Prediction / inference / Deployment



** Machine learning is only “**guaranteed to work**” when the training data are **drawn i.i.d.** from the **same distribution** as the new data that we will receive in the “inference” phase.

“Adversarial Examples” are consequences of distribution-shift



“panda”

57.7% confidence

noise

“gibbon”

99.3% confidence

(Goodfellow et al., 2015)

Very subtle distribution shift can break a trained classifier catastrophically.

Quick checkpoint

- Feature extraction
- Specifying a “hypothesis class”
 - indexed by “free parameters”
- Learning == search for the best hypothesis
- Ideally, we want to minimize “test error”
 - but all we have access to is the training data.
 - minimize “training error” (Statistical learning theory says that this is OK)
 - We have a practical way --- data-splitting --- to evaluate a classifier

Remainder of the lecture

- A selective review of linear algebra
- Focus on concepts relevant to machine learning
 - Credit to Prof. Shiyu Chang
- A summary of ML notations attached in the rest of this slide deck.

Mathematically defining the supervised learning problem

- Feature space: \mathcal{X}
- Label space: \mathcal{Y}
- A classifier (hypothesis): $h : \mathcal{X} \rightarrow \mathcal{Y}$
- A hypothesis class: \mathcal{H}
- Data: $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- Learning task: Find $h \in \mathcal{H}$ that “works well”.

Notations from linear algebra

- Matrices and vectors

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R}. \quad \mathbf{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \in \mathbb{R}^3$$

- Transpose and inverse

$$\mathbf{A}^T \in \mathbb{R}^{n \times m}$$

$$\mathbf{A}^{-1} \mathbf{A} = \mathbf{I}$$

- Inner product / dot product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y} = \sum_{i=1}^n x_i y_i.$$

- Norms

$$\|\mathbf{x}\| := \sqrt{\sum_i x_i^2}$$

$$\|\mathbf{x}\|_p := \left(\sum_i x_i^p \right)^{1/p}$$

Other useful notations

$B = (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)$	(Ordered) tuple
$\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3]$	Matrix of column vectors stacked horizontally
$\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$	Set of vectors (unordered)
\mathbb{Z}, \mathbb{N}	Integers and natural numbers, respectively
\mathbb{R}, \mathbb{C}	Real and complex numbers, respectively
\mathbb{R}^n	n -dimensional vector space of real numbers
$\forall x$	Universal quantifier: for all x
$\exists x$	Existential quantifier: there exists x

$$[n] := \{1, 2, 3, \dots, n\}$$

$|\mathcal{S}|$ — cardinality of a set \mathcal{S} e.g., $|[n]| = n$

Indicator (one-zero) function: $\mathbb{I}[\text{condition}] \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if condition is true} \\ 0 & \text{otherwise.} \end{cases}$

Conventions and typical meaning of specific variables in machine learning

- x : input
- y : output
- z : input-output pair
- d : dimensionality
- n : number of examples

The “hat” notation, e.g.: \hat{h} , \hat{f} , $\hat{\theta}$, $\hat{\mathbb{E}}$ associated with being an estimate, computed as a function of the data

The “star” notation, e.g.: h^* , f^* , θ^* , p^* , R^* associated with being “optimal”

Notations from probability

$\mathbb{E}_{\mathcal{D}}$ [Function of an r.v. X]

$\mathbb{P}_{\mathcal{D}}$ [Event]

$f_{X \sim \mathcal{D}}(x)$

$F_{X \sim \mathcal{D}}(x)$

Conditional expectation / conditional probability / density

$\mathbb{E}[\text{Func}(X, Y) | Y]$

$\mathbb{P}[\text{Event_of}(X, Y) | Y]$

$f(x|y)$

Loss, Risk, Empirical Risk: What do we mean by working well?

- Loss function

$$\ell(h, (x, y))$$

- Risk function

$$R(h, \mathcal{D}) = \mathbb{E}_{\mathcal{D}}[\ell(h, (x_i, y_i))]$$

- Empirical risk

$$\hat{R}(h, \text{Data}) = \frac{1}{n} \sum_{i=1}^n \ell(h, (x_i, y_i))$$