

Learning from Data

1. Overview

© Prakash Ishwar
Spring 2017

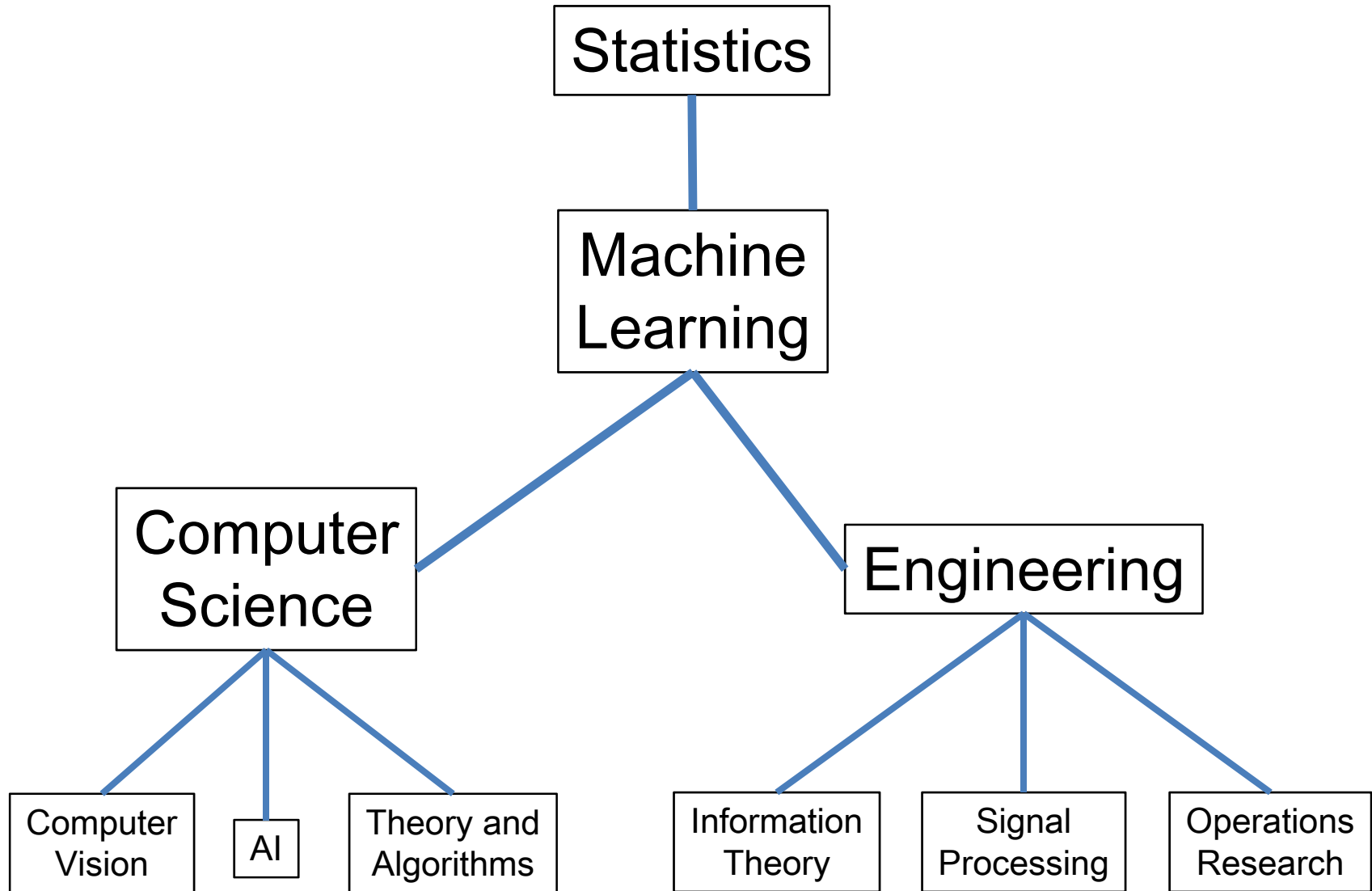
Machine Learning

- Era of “BIG” data deluge:
 - Almost 1 trillion websites
 - 1 hour of video uploaded to YouTube every second
 - Walmart handles 1 million transactions per hour
 - Genomes of 1000s people sequenced, each 3.8 billion base pairs long...
- ➔ Need automated methods of data analysis
- Machine Learning = set of methods that can:
 - automatically learn regularities in observed data and
 - use them to make accurate decisions about unobserved data

Applications

- **Detection:** spam, fraud (credit card), network intrusion, face (in images), motion (in video),...
- **Recognition:** characters, speech, face, actions,...
- **Prediction:** weather conditions, stock prices, target position, drug-response, missing values,...
- **Clustering:** image segmentation (medicine/ astronomy), gene families, communities in social networks,...

Connections



Skill Set

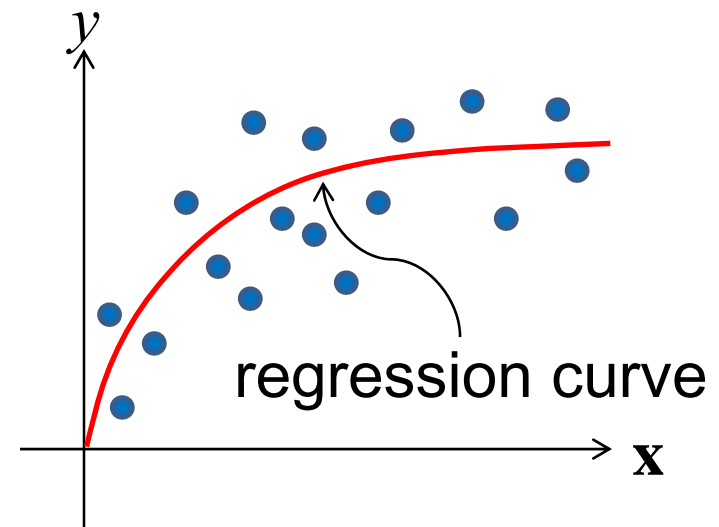
- Probability
- Optimization
 - Linear Algebra
 - Multivariate Calculus
- Algorithms
- Software

Main classes of learning problems

- Supervised (predictive) learning: given **examples** with **labels**, predict labels for all **unseen** examples
 - Classification: label = category
 - Regression: label = quantity (real value)
 - Ranking: label = order (relative position)

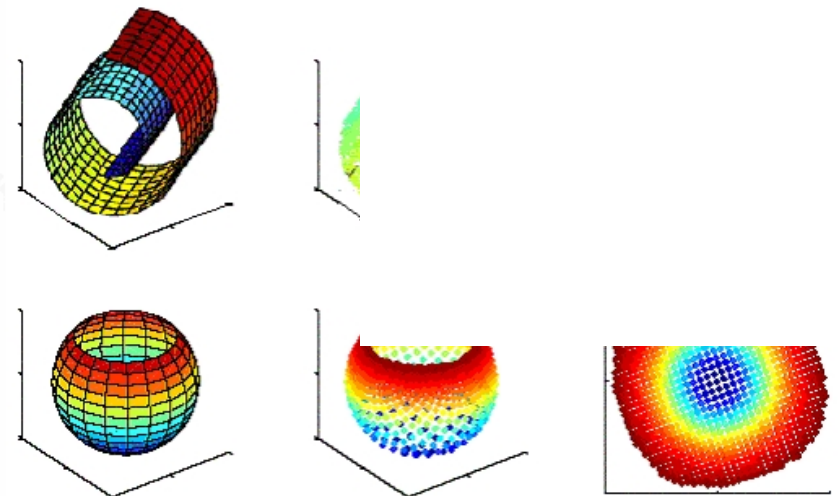
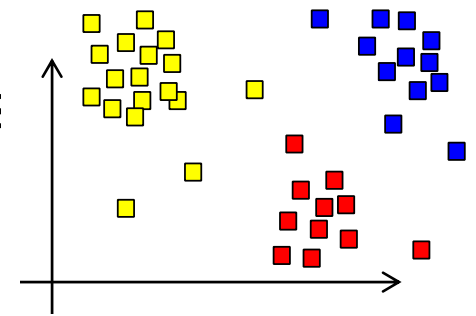


\mathbf{x} = facial geometry features
 y = gender label



Main classes of learning problems

- **Unsupervised (descriptive) learning:** given **un-labeled** examples, find hidden structure
 - (Probability) Density Estimation
 - **Clustering:** partition examples into groups that are intrinsically similar and extrinsically dissimilar
 - **Dimensionality Reduction:** decrease the number of variables used to represent examples while preserving some properties of the original representation



Other learning scenarios

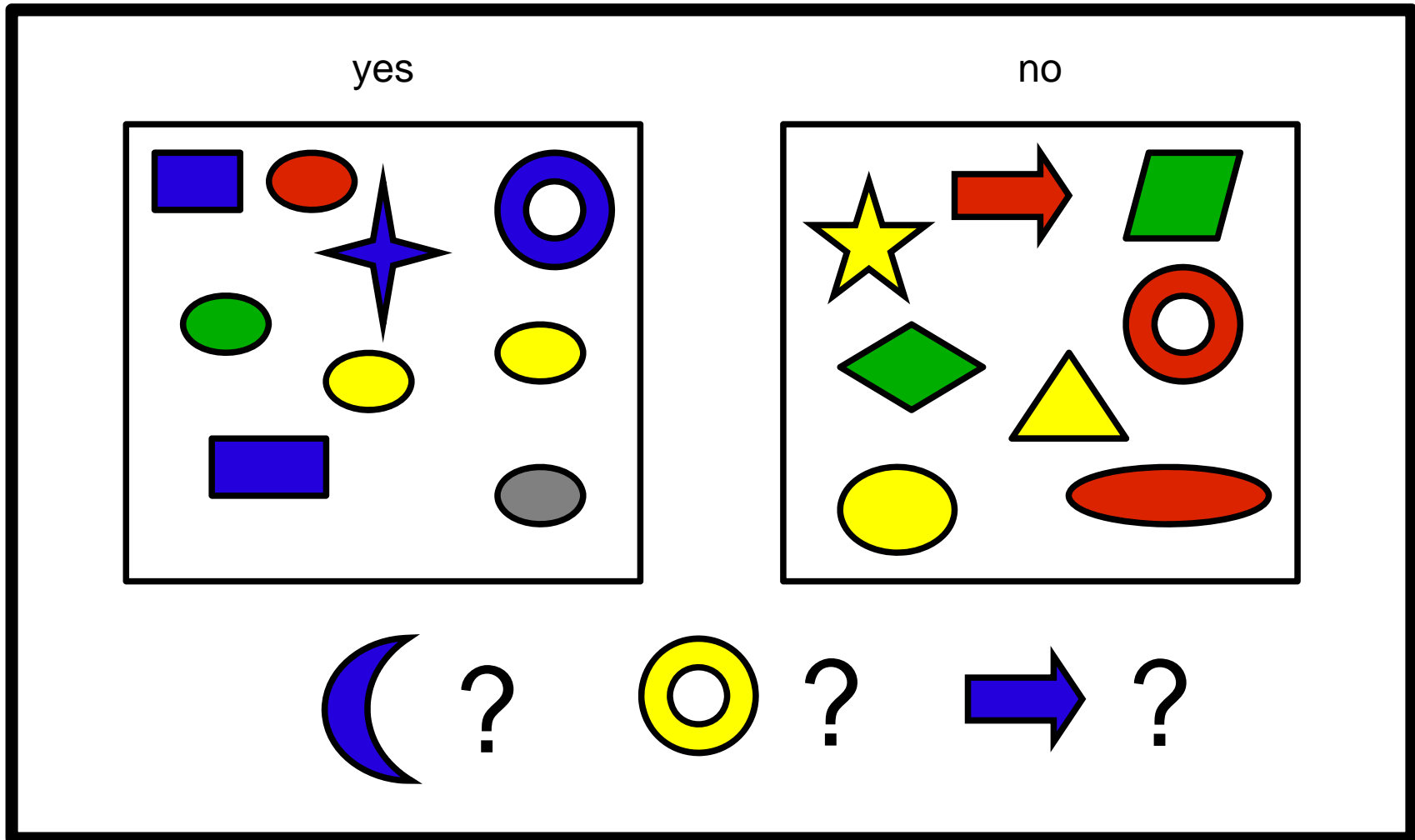
- **Semi-supervised learning:** given both labeled and unlabeled examples, predict labels for all **unseen** examples
- **Transductive inference:** similar to semi-supervised, but only need to predict labels for **seen** unlabeled examples, not all unseen examples
- **Active learning:** sequentially choose unlabeled examples for labeling to accurately and quickly predict labels for all **unseen** examples

Other learning scenarios

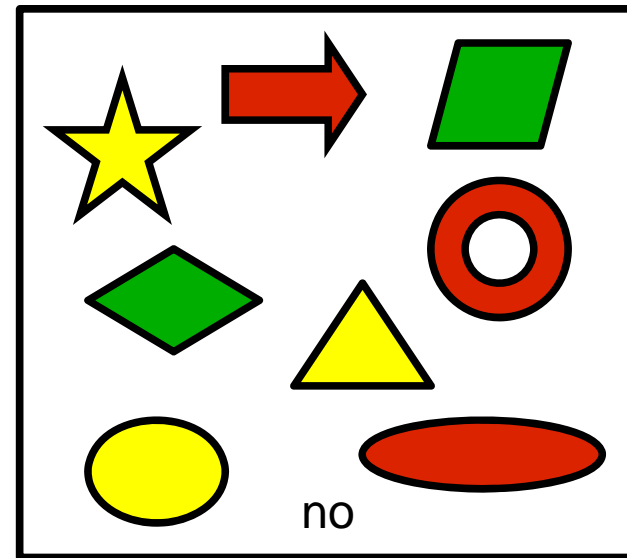
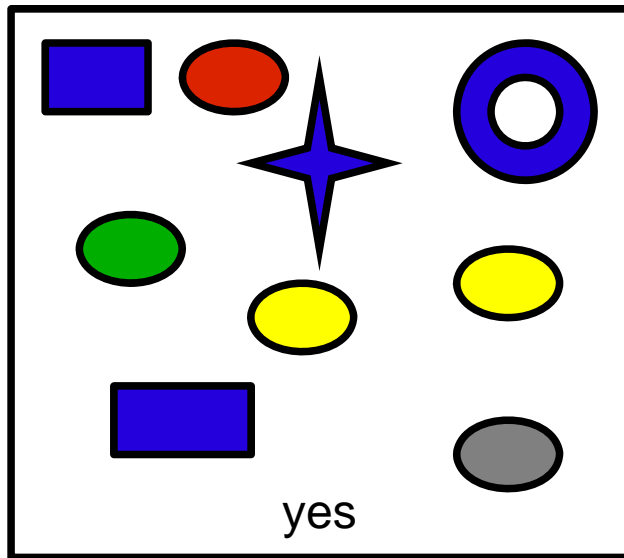
- **Online learning:** multiple rounds of learning and decision making: receive unlabeled example → predict label → receive true label → incur loss → receive unlabeled example → ...
Goal: minimize cumulative loss
- **Reinforcement learning:** multi-round learning like online, but:
 - no long-term reward (only immediate reward)
 - decisions can change environment
 - e.g., search a terrain or the internet and learn as much new information as possible: explore vs exploit tradeoff




Supervised Learning

- Toy binary classification example

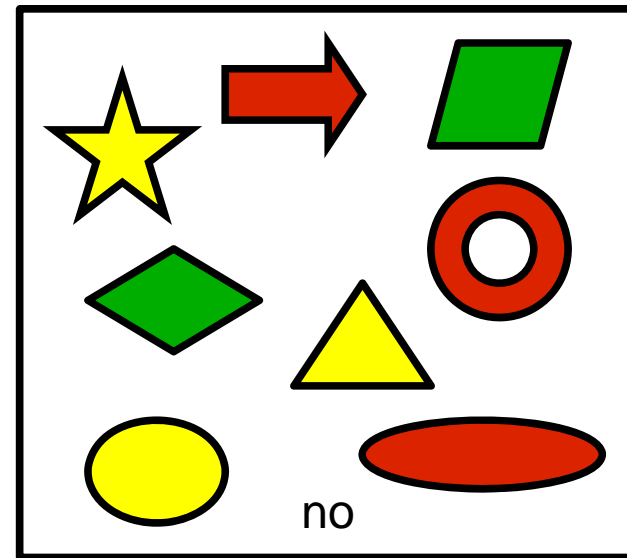
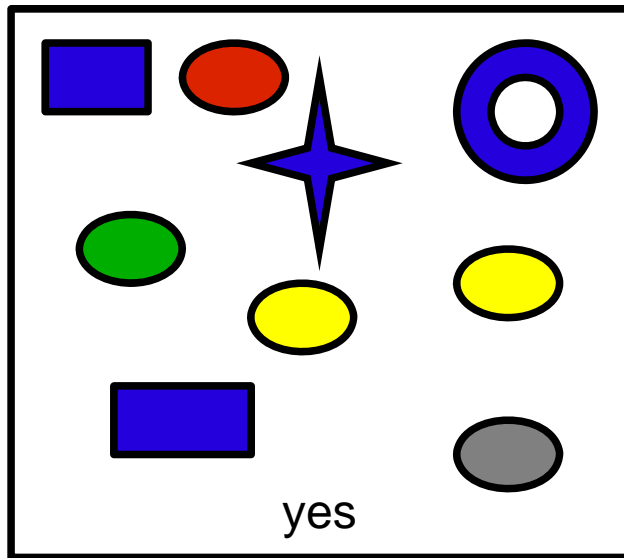


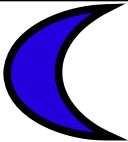
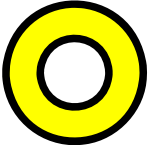
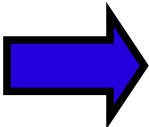
Supervised Learning



Training sample	Feature vector: $\mathbf{x}_i = (\text{color, shape, size})^T$	Label (Yes \rightarrow 1, No \rightarrow 0)
	$\mathbf{x}_1 = (\text{blue, rectangle, } 10)^T$	$y_1 = 1$
	$\mathbf{x}_2 = (\text{red, ellipse, } 2.4)^T$	$y_2 = 1$
	$\mathbf{x}_3 = (\text{red, ellipse, } 20.7)^T$	$y_3 = 0$
...

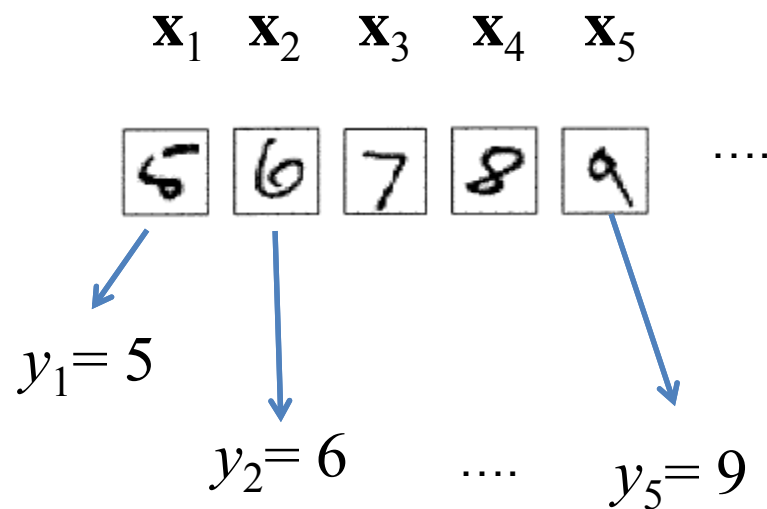
Supervised Learning



Test sample	Feature vector: $\mathbf{x} = (\text{color, shape, size})^T$	Label (Yes \rightarrow 1, No \rightarrow 0)
	$\mathbf{x} = (\text{blue, crescent, } 17)^T$	$y = 1$
	$\mathbf{x} = (\text{yellow, ring, } 8)^T$	$y = ?$
	$\mathbf{x} = (\text{blue, arrow, } 11)^T$	$y = ?$

Supervised Learning

- Real-world example: **character recognition**
 - **MNIST dataset**: document image preprocessed to isolate and center individual handwritten characters into 28 x 28 grayscale images (grayscale value = integer in 0-255)
 - 60,000 training and 10,000 test images of digits 0-9 with ground-truth labels



7	2	1	0	4	1	4	9	5	9
0	6	9	0	1	5	9	7	8	4
9	6	6	5	4	0	7	4	0	1
3	1	3	4	7	2	7	1	2	1
1	7	4	2	3	5	1	2	4	4
6	3	5	5	6	0	4	1	9	5
7	8	9	3	7	4	6	4	3	0
7	0	2	9	1	7	3	2	9	7
7	6	2	7	8	4	7	3	6	1
3	6	9	3	1	4	1	7	6	9

Supervised Learning

- More real-world examples
 - Medicine:
 - \mathbf{x} = symptoms, y = disease-type
 - \mathbf{x} = EEG signal, y = seizure / no-seizure
 - Social Networks:
 - \mathbf{x} = profile-pair, y = friend / not friend
 - Image/Video Analysis, Computer Vision:
 - \mathbf{x} = image/video snippet, y = object-category, activity, etc.

Need for Probability

- Inductive reasoning:
 - going from particular instances to broader generalizations, e.g., $1 \rightarrow 2$, $2 \rightarrow 4$, $3 \rightarrow ?$
 - inherently uncertain: allows for the possibility of the conclusion to be false even if all premises are true
- Randomness (variability/complexity/uncertainty) inherent in most real-world data, e.g.,

3 3 3 3 8

- Examples, features, and labels \rightarrow random variables

Definitions, Terminology, Notation

- Examples
- Features: $\mathbf{x} \in \mathcal{X}$ (feature/input space)
- Labels: $y \in \mathcal{Y}$ (label/output space)
- Training-samples: $\mathcal{D} = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$, $\mathbf{z}_i = (\mathbf{x}_i, y_i)$
- Validation-samples
- Test-samples
- Hypothesis set: $\mathcal{H} = \{h : \mathcal{X} \rightarrow \mathcal{Y}\}$
- Loss function: $\ell(\mathbf{x}, y, h) : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \rightarrow \mathbb{R}_+ = [0, \infty)$

Definitions, Terminology, Notation

- Examples:

- data used for training or evaluation
- e.g., email messages in Spam Detection

- Features:

- set of attributes, represented as vector $\mathbf{x} \in \mathcal{X}$ (in some feature/input space, e.g., \mathbb{R}^d), associated to example
- e.g., message length, name of sender, presence of keywords in subject/body, etc., in Spam Detection

Definitions, Terminology, Notation

- **Features:**

- 3 types:

- **Categorical/Nominal:** no intrinsic value or ordering, just a listing, e.g., gender, blood-type, shape, etc.
 - **Quantitative:** has intrinsic numerical value, e.g., physical measurements (space, time, mass, charge, etc.); can be discrete (counts) or continuous; represented by real numbers
 - **Ordinal:** has intrinsic ordering, but no numeric value, e.g., economic-bracket: $\in \{\text{poor, middle-class, rich}\}$

- **Euclidean-space embedding:** convert categorical/ ordinal variable into a real-valued vector.

- **Categorical:** e.g., for blood-type $\in \{A, B, AB, O\}$,
 $A \rightarrow (1, 0, 0, 0)^T$, $B \rightarrow (0, 1, 0, 0)^T$, $AB \rightarrow (0, 0, 1, 0)^T$, $O \rightarrow (0, 0, 0, 1)^T$
 - **Ordinal:** e.g., $\{\text{low} \rightarrow \frac{1}{3}, \text{medium} \rightarrow \frac{2}{3}, \text{high} \rightarrow 1\}$

Definitions, Terminology, Notation

- **Labels:** categories, values, or ordering in output/label space assigned to examples
 - **Classification:** label $y \in \mathcal{Y} = \{1, \dots, m\}$, m = number of classes
 - **Regression:** label $y \in \mathcal{Y} = \mathbb{R}$, a real value
- **Training-samples:** $\mathcal{D} = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$, $\mathbf{z}_j = (\mathbf{x}_j, y_j)$
 - examples (features + labels) used to train a learning algorithm
- **Validation-samples:**
 - examples used to **tune parameters** of learning algorithm, e.g., polynomial degree in regression (as opposed to the polynomial coefficients which are learned during training)

Definitions, Terminology, Notation

- **Test-samples:**
 - examples (features + labels) used to **evaluate performance** of learning algorithm
 - must be kept separate from training and validation samples and not made available in the learning stage
 - predictions on test-sample features compared against test-sample labels to measure performance
- **Hypothesis set:** $\mathcal{H} = \{h : \mathcal{X} \rightarrow \mathcal{Y}\}$
 - set of functions or **decision rules** mapping feature vectors to labels
 - e.g., if $\mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{-1, +1\}$ (binary classification), a **hyperplane classifier** is described by: $h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$
 - may have additional **tuning parameters** α , e.g. poly deg.

Definitions, Terminology, Notation

- **Loss function:** $\ell(\mathbf{x}, y, h) : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \rightarrow \mathbb{R}_+ = [0, \infty)$
 - function that quantifies the degree of disagreement between the predicted and true label
 - **0-1 loss (classification):** $\ell(\mathbf{x}, y, h) = 1(h(\mathbf{x}) \neq y)$
 - **squared loss (regression):** $\ell(\mathbf{x}, y, h) = (h(\mathbf{x}) - y)^2$
 - **Note:** the loss function is really only a function of y and $h(\mathbf{x})$: $\ell(\mathbf{x}, y, h) \equiv \ell(y, h(\mathbf{x}))$

Main stages of any learning problem

- Data-partitioning → Training, validation, testing sets
- Pre-processing and Feature Selection → Black Art
- Training
- Validation
- Testing

Main stages of any learning problem

- **Data-partitioning** → Training, validation, testing sets
 - available data randomly partitioned into training, validation, and testing sets
 - # validation samples → # parameters to be tuned
 - # training samples > # testing samples when total number of labeled samples is small
 - must ensure that all 3 have similar composition, e.g., similar proportion of all classes in a classification problem
- **Pre-processing and Feature Selection**
 - leverages **prior** domain knowledge/beliefs
 - often “messy”, heuristic
 - critical step: can dramatically effect performance, positively or negatively

Main stages of any learning problem

- **Pre-processing**

- **Missing data:** either discard or “fill-in” with most frequently seen values from similar samples
- **Imbalanced data:** replicate/re-sample/increase weight of under-represented populations or discard/down-sample/decrease weight over-represented ones
- **Outliers:** detect samples most inconsistent with bulk of samples. Either remove them or design a robust loss function that is insensitive to outliers.
- **Prefiltering:** reduce effects of noise, aliasing, quantization, misalignments
- **Normalization:** compensate for unequal dynamic ranges of variables, e.g., mean-variance equalization, known nonlinearities in sensors
- **Data augmentation:** make dataset richer by incorporating desirable invariances, e.g., color, position, orientation of objects in images.

- **Feature selection**

- Identify most relevant subsets or functions of input variables
- Unsupervised learning, specifically, clustering and dimensionality reduction play important roles

Main stages of any learning problem

- Training

- Given training set: $\mathcal{D} = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$, $\mathbf{z}_j = (\mathbf{x}_j, y_j)$
- fix tuning parameters α and select best decision rule
- Structural Risk Minimization (SRM) Principle:

$$\mathcal{L}_{\text{train}}(h, \alpha, \lambda) = \left[\underbrace{\frac{1}{n} \sum_{j=1}^n \ell(\mathbf{x}_j, y_j, h)}_{\substack{\text{Empirical Risk} \\ \text{(data term)}}} + \underbrace{\frac{\lambda}{n} \text{complexity}(h)}_{\substack{\text{Structure Penalty} \\ \text{(prior beliefs)}}} \right]$$

$$h^{\text{SRM}}(\alpha, \lambda) = \arg \min_{h \in \mathcal{H}} \mathcal{L}_{\text{train}}(h, \alpha, \lambda)$$

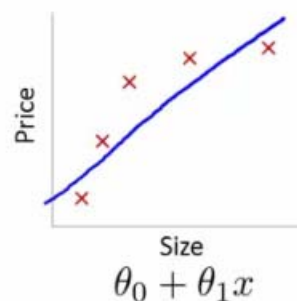
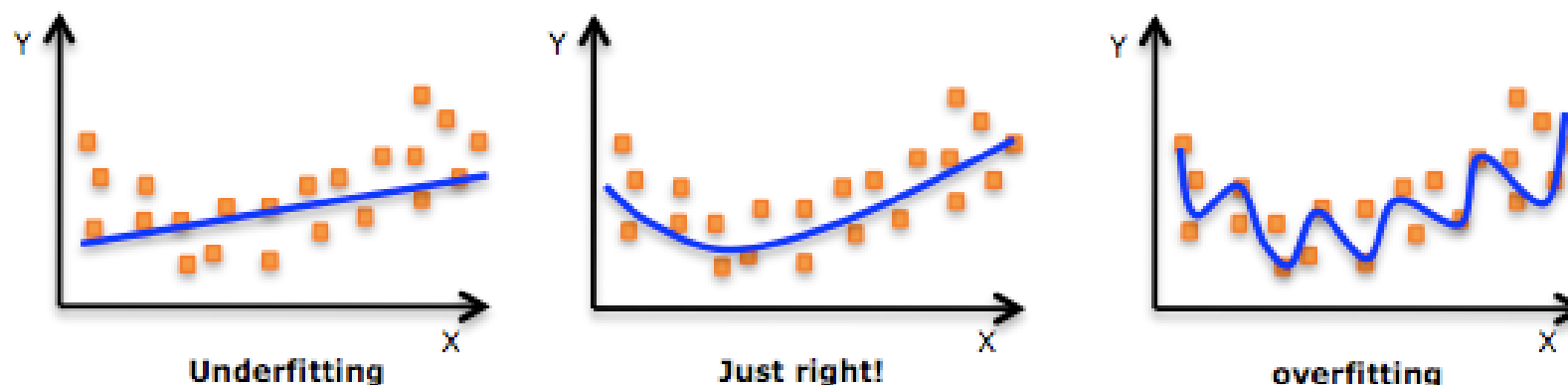
Main stages of any learning problem

- Training

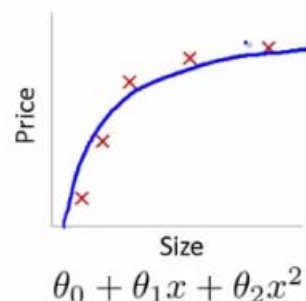
- Empirical Risk Minimization (ERM): special case of SRM with $\lambda=0$. Corresponds to minimizing only 1st term of $\mathcal{L}_{\text{train}}(h, \alpha, \lambda)$.
- Overfitting: if $n \ll \#$ free parameters in h (complex model), solution based on ERM will be too noisy (will have high variance) → model starts “memorizing” training data instead of learning to generalize from trend
- Complexity regularization (2nd term of $\mathcal{L}_{\text{train}}(h, \alpha, \lambda)$): mitigates overfitting by biasing solutions to be faithful to prior beliefs

Bias-Variance tradeoff

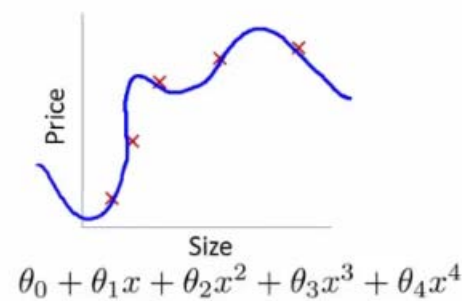
- $\lambda = 0$: Prior beliefs completely ignored. Complex solutions have low bias, but high variance
- $\lambda \gg 1$: Prior beliefs dominate. Solutions have low variance, but high bias



High bias
(underfit)



"Just right"



High variance
(overfit)

Main stages of any learning problem

- **Validation**

- Learn best values of α, λ using validation samples
- Best α_*, λ_* minimizes the validation error:

$$\mathcal{L}_{\text{valid}}(\alpha, \lambda) = \frac{1}{|\text{validation set}|} \sum_{(\mathbf{x}, y) \in \text{validation set}} \ell(\mathbf{x}, y, h^{\text{SRM}}(\alpha, \lambda))$$

$$(\alpha_*, \lambda_*) = \arg \min_{\alpha, \lambda} \mathcal{L}_{\text{valid}}(\alpha, \lambda)$$

- **Testing**

- Evaluate performance using test samples
- Test error:

$$\mathcal{L}_{\text{test}}(\alpha_*, \lambda_*) = \frac{1}{|\text{test set}|} \sum_{(\mathbf{x}, y) \in \text{test set}} \ell(\mathbf{x}, y, h^{\text{SRM}}(\alpha_*, \lambda_*))$$

Main stages of any learning problem

- Testing

- Performance metrics other than the loss function used for training and validation may be used for evaluation:

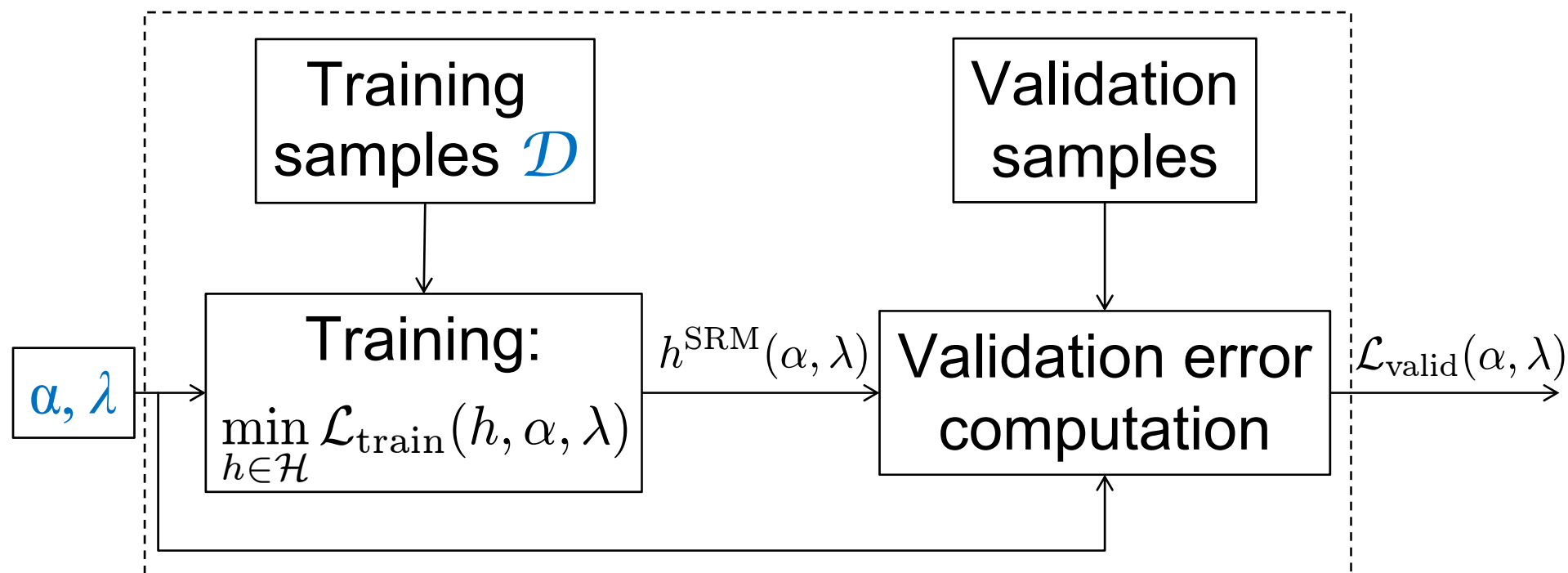
- Correct Classification Rate (CCR)
 - Confusion Matrix
 - Prob. of False Alarm, Missed Detection, Precision, Recall, Sensitivity, Specificity, F-score, Area Under the ROC Curve, etc.
 - Mean Square Error (MSE), Mean Absolute Error (MAE),
 - Predictive log-probability

- Small training error \nRightarrow Small validation or test error
 - Memorization vs Generalization (Overfitting)

Training-Validation-Testing summary

- **Training:** given $\alpha, \lambda, \mathcal{D}$ find best $h^{\text{SRM}}(\alpha, \lambda)$ in \mathcal{H} by minimizing structural risk
- **Validation:** given validation samples, find best α_*, λ_* by minimizing validation error of $h^{\text{SRM}}(\alpha, \lambda)$ over choices of α, λ
 - In practice, the search for the best α, λ is typically done via a greedy coarse-to-fine grid search
- **Testing:** use $\alpha_*, \lambda_*, h^{\text{SRM}}(\alpha_*, \lambda_*)$ to evaluate performance on test samples

Training-Validation-Testing summary



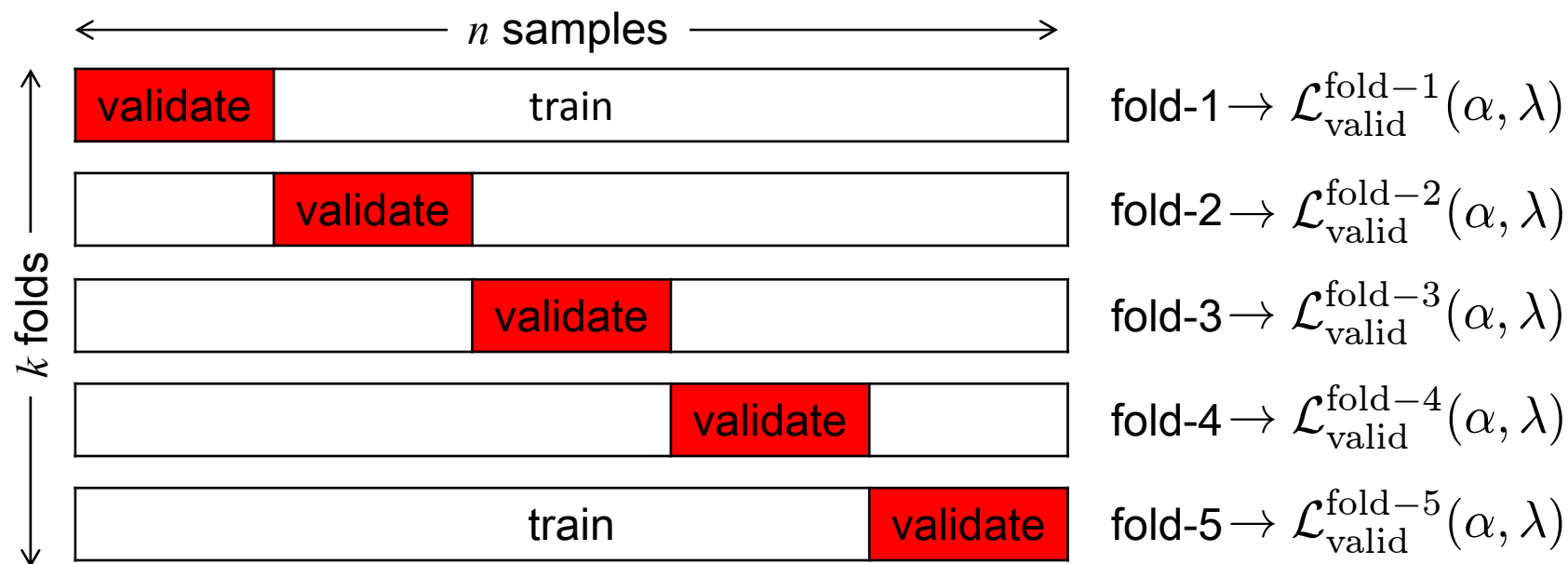
- **Validation:** given validation samples, find best α_*, λ_* by minimizing validation error of $h^{\text{SRM}}(\alpha, \lambda)$ over choices of α, λ
- **Testing:** use $\alpha_*, \lambda_*, h^{\text{SRM}}(\alpha_*, \lambda_*)$ to evaluate performance on test samples

Example: polynomial regression

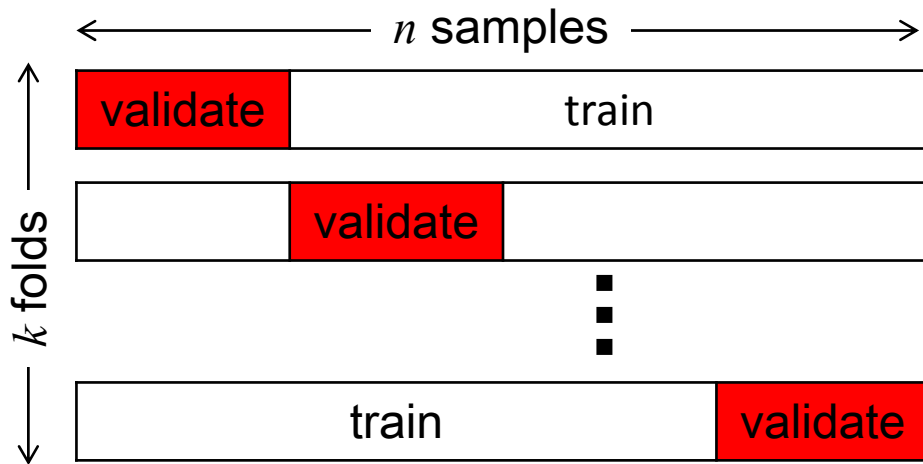
- $\mathcal{X} = \mathcal{Y} = \mathbb{R}$,
- $\mathcal{H} = \{h(x) = \sum_{k=0}^{\alpha} h_k x^k\}$,
- $\ell(x, y, h) = (y - h(x))^2$
- $\mathcal{L}_{\text{train}}(h, \alpha, \lambda) = \frac{1}{n} \sum_{j=1}^n \left(y_j - \sum_{k=0}^{\alpha} h_k x_j^k \right)^2 + \lambda \sum_{k=0}^{\alpha} |h_k|$
- $(h_0^{\text{SRM}}(\alpha, \lambda), \dots, h_{\alpha}^{\text{SRM}}(\alpha, \lambda)) = \arg \min_{h_0, \dots, h_{\alpha}} \mathcal{L}_{\text{train}}(h_0, \dots, h_{\alpha}, \alpha, \lambda)$
- $\mathcal{L}_{\text{valid}}(\alpha, \lambda) = \frac{1}{|\text{validation set}|} \sum_{(x, y) \in \text{validation set}} \left(y - \sum_{k=0}^{\alpha} h_k^{\text{SRM}}(\alpha, \lambda) x^k \right)^2$
- $(\alpha_*, \lambda_*) = \arg \min_{\alpha, \lambda} \mathcal{L}_{\text{valid}}(\alpha, \lambda)$
- $\mathcal{L}_{\text{test}}(\alpha_*, \lambda_*) = \frac{1}{|\text{test set}|} \sum_{(x, y) \in \text{test set}} \left(y - \sum_{k=0}^{\alpha_*} h_k^{\text{SRM}}(\alpha_*, \lambda_*) x^k \right)^2$

Cross-Validation

- Not enough labeled data for creating separate validation samples
- Idea: cycle samples through training and validation
- Can also do this for testing (if needed)
- How? Many approaches: bootstrap, exhaustive methods,...focus on ***k*-fold cross-validation**



Cross-Validation



$$\mathcal{L}_{\text{CV}}(\alpha, \lambda) = \frac{1}{k} \sum_{j=1}^k \mathcal{L}_{\text{valid}}^{\text{fold}-j}(\alpha, \lambda)$$

$$(\alpha_*, \lambda_*) = \arg \min_{\alpha, \lambda} \mathcal{L}_{\text{CV}}(\alpha, \lambda)$$

- Then train on all n samples with α_*, λ_* to obtain $h^{\text{SRM}}(\alpha_*, \lambda_*)$
- Then evaluate performance on test samples
- Holdout cross-validation: $k = 2$
- Leave-one-out-cross-validation (LOOCV): $k = n$
- Typical choices for number of folds: $k = 5, 10$

Next: Probabilistic framework

- Optimum decision rule: MAP, ML
- Generative vs Discriminative Learning
- Frequentist vs Bayesian
- Parametric vs Non-parametric models
- Asymptotic consistency, sample complexity, computational complexity (time, storage)