Machine Learning 2

C. Sparse Models / 4. Automatic Relevance Determination

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

			A. Advanced Supervised Learning
Fri.	24.4.	(1)	A.1 Generalized Linear Models
Fri.	1.5.	—	— *Labour Day* —
Fri.	8.5.	(2)	A.2 Gaussian Processes
Fri.	15.5.	(3)	A.3 Advanced Support Vector Machines
			B. Ensembles
Fri.	22.5.	(4)	B.1 Stacking
			& B.2 Boosting
Fri.	29.5.	(5)	B.3 Mixtures of Experts
Fri.	5.6.	—	— *Pentecoste Break* —
			C. Sparse Models
Fri.	12.6.	(6)	C.1 Homotopy and Least Angle Regression
Fri.	19.6.	(7)	C.2 Proximal Gradients
Fri.	26.6.	(8)	C.3 Laplace Priors
Fri.	3.7.	(9)	C.4 Automatic Relevance Determination
			D. Complex Predictors
Fri.	10.7.	(10)	D.1 Latent Dirichlet Allocation (LDA)
Fri.	17.7.	(11)	Q & A

Outline

1. Automatic Relevance Determination (ARD)

2. A note on Model Complexity

Outline

Linear Regression plus ARD Regularization

Linear Regression plus L2 regularization (Ridge Regression):

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \sigma_\beta^2 I)$$

Linear Regression plus ARD Regularization:

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \operatorname{diag}(\sigma_{\beta_1}^2, \ldots, \sigma_{\beta_M}^2))$$

Linear Regression plus ARD Regularization

Linear Regression plus L2 regularization (Ridge Regression):

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \sigma_\beta^2 I)$$

Linear Regression plus ARD Regularization:

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \mathrm{diag}(\sigma_{\beta_1}^2, \ldots, \sigma_{\beta_M}^2))$$

Idea:

- use a different regularization weight for each predictor x_m.

Linear Regression plus ARD Regularization

Linear Regression plus L2 regularization (Ridge Regression):

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \sigma_\beta^2 I)$$

Linear Regression plus ARD Regularization:

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \operatorname{diag}(\sigma_{\beta_1}^2, \ldots, \sigma_{\beta_M}^2))$$

Idea:

- use a different regularization weight for each predictor x_m.
- but M hyperparameters are too many to learn by grid search.

Linear Regression plus ARD Regularization

Linear Regression plus L2 regularization (Ridge Regression):

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \sigma_\beta^2 I)$$

Linear Regression plus ARD Regularization:

$$p(y_n \mid x_n, \beta, \sigma_y^2) := \mathcal{N}(y_n \mid \beta^T x_n, \sigma_y^2)$$
$$p(\beta) := \mathcal{N}(\beta \mid 0, \Sigma_\beta := \mathrm{diag}(\sigma_{\beta_1}^2, \ldots, \sigma_{\beta_M}^2))$$
$$p(\sigma_y^2) := \mathrm{InvGamma}(\sigma_y^2 \mid c, d)$$
$$p(\sigma_{\beta_m}^2) := \mathrm{InvGamma}(\sigma_{\beta_m}^2 \mid a, b), \quad m = 1, \ldots, M$$

Idea:

- use a different regularization weight for each predictor x_m.
- but M hyperparameters are too many to learn by grid search.
- hence put a hyperprior on top.

Empirical Bayes

Maximum Likelihood (ML): $$\hat{\theta} := \arg\max_{\theta} p(\mathcal{D} \mid \theta)$$

Full Bayes: $$(\hat{\theta}, \hat{\eta}) \sim p(\theta, \eta \mid \mathcal{D}) \propto p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)\, p(\eta)$$

Empirical Bayes

Maximum Likelihood (ML): $\hat{\theta} := \arg\max_{\theta} p(\mathcal{D} \mid \theta)$

Maximum Aposteriori (MAP): $\hat{\theta} := \arg\max_{\theta} p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)$

Full Bayes: $(\hat{\theta}, \hat{\eta}) \sim p(\theta, \eta \mid \mathcal{D}) \propto p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)\, p(\eta)$

Empirical Bayes

Maximum Likelihood (ML): $\hat{\theta} := \arg\max_{\theta} p(\mathcal{D} \mid \theta)$

Maximum Aposteriori (MAP): $\hat{\theta} := \arg\max_{\theta} p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)$

ML-II (Empirical Bayes):

$$\hat{\eta} := \arg\max_{\eta} p(\mathcal{D} \mid \eta) = \arg\max_{\eta} \int p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta) d\theta$$

$$\hat{\theta} \sim p(\theta \mid \mathcal{D}, \hat{\eta}) \propto p(\mathcal{D} \mid \theta)\, p(\theta \mid \hat{\eta})$$

MAP-II:

$$\hat{\eta} := \arg\max_{\eta} p(\mathcal{D} \mid \eta)\, p(\eta) = \arg\max_{\eta} \int p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)\, p(\eta) d\theta$$

$$\hat{\theta} \sim p(\theta \mid \mathcal{D}, \hat{\eta}) \propto p(\mathcal{D} \mid \theta)\, p(\theta \mid \hat{\eta})$$

Full Bayes: $(\hat{\theta}, \hat{\eta}) \sim p(\theta, \eta \mid \mathcal{D}) \propto p(\mathcal{D} \mid \theta)\, p(\theta \mid \eta)\, p(\eta)$

Marginal Likelihood

Without hyperpriors:

$$p(y \mid X, \sigma_y^2, \Sigma_\beta) = \int \mathcal{N}(y \mid X\beta, \sigma_y^2 I)\mathcal{N}(\beta \mid 0, \Sigma_\beta) d\beta$$
$$= \mathcal{N}(y \mid 0, \sigma_y^2 I + X\Sigma_\beta X^T)$$

Marginal Likelihood

Without hyperpriors:

$$p(y \mid X, \sigma_y^2, \Sigma_\beta) = \int \mathcal{N}(y \mid X\beta, \sigma_y^2 I)\,\mathcal{N}(\beta \mid 0, \Sigma_\beta)\,d\beta$$
$$= \mathcal{N}(y \mid 0, \underbrace{\sigma_y^2 I + X\Sigma_\beta X^T}_{=: C_y})$$

$$\ell(\sigma_y^2, \Sigma_\beta) := -\log p(y \mid X, \sigma_y^2, \Sigma_\beta) \propto \log |C_y| + y^T C_y^{-1} y$$

Marginal Likelihood

Without hyperpriors:

$$\begin{aligned} p(y \mid X, \sigma_y^2, \Sigma_\beta) = & \int \mathcal{N}(y \mid X\beta, \sigma_y^2 I) \mathcal{N}(\beta \mid 0, \Sigma_\beta) d\beta \\ & = \mathcal{N}(y \mid 0, \underbrace{\sigma_y^2 I + X\Sigma_\beta X^T}_{=: C_y}) \end{aligned}$$

$$\ell(\sigma_y^2, \Sigma_\beta) := -\log p(y \mid X, \sigma_y^2, \Sigma_\beta) \propto \log |C_y| + y^T C_y^{-1} y$$

With hyperpriors:

$$\begin{aligned} \ell(\sigma_y^2, \Sigma_\beta) := & -\log p(y \mid X, \sigma_y^2, \Sigma_\beta)\, p(\sigma_y^2 \mid c, d)\, p(\Sigma_\beta \mid a, b) \\ & \propto \log |C_y| + y^T C_y^{-1} y + \sum_{m=1}^{M} (-a \log \sigma_{\beta_m}^2 - b/\sigma_{\beta_m}^2) \\ & \qquad - c \log \sigma_y^2 - d/\sigma_y^2 \end{aligned}$$

Bayes Rule for Linear Gaussian Systems

For an LGS

$$p(x) := \mathcal{N}(x \mid \mu_x, \Sigma_x)$$
$$p(y \mid x) := \mathcal{N}(y \mid Ax + b, \Sigma_y)$$

Bayes' Rule reads:

$$p(x \mid y) = \mathcal{N}(x \mid \mu_{x|y}, \Sigma_{x|y})$$

$$\text{with} \quad \Sigma_{x|y} := (\Sigma_x^{-1} + A^T \Sigma_y^{-1} A)^{-1}$$
$$\mu_{x|y} := \Sigma_{x|y} \left(A^T \Sigma_y^{-1}(y - b) + \Sigma_x^{-1} \mu_x \right)$$

Inferring Parameters β

$$
\begin{aligned}
p(\beta \mid X, y, \sigma_y^2, \Sigma_\beta) &= \frac{1}{Z}\mathcal{N}(\beta \mid 0, \Sigma_\beta)\,\mathcal{N}(y \mid X\beta, \sigma_y^2\, I) \\
&= \mathcal{N}(\beta \mid \mu_\beta := \frac{1}{\sigma_y^2} C_\beta X^T y, C_\beta := (\frac{1}{\sigma_y^2} X^T X + \Sigma_\beta^{-1})^{-1})
\end{aligned}
$$

using Bayes Rule

for $\Sigma_\beta = \infty I$: unregularized estimates

$$
= \mathcal{N}(\beta \mid (X^T X)^{-1} X^T y, \sigma_y^2 (X^T X)^{-1})
$$

for $\sigma_y^2 = \infty$: overregularized estimates

$$
= \mathcal{N}(\beta \mid 0, \Sigma_\beta)
$$

Equivalent MAP Estimation Problem

$$\ell(\beta) = \frac{1}{\sigma_y^2} ||y - X\beta||_2^2 + \min_{\sigma^2_{\beta_1},\ldots,\sigma^2_{\beta_M} \geq 0} \log |\sigma_y^2 I + X \mathrm{diag}(\sigma_\beta^2) X^T| + \sum_{m=1}^{M} \frac{\beta_m^2}{\sigma^2_{\beta_m}}$$

Two Rules for Expectations

$$i) \quad \mathbb{E}(\text{tr}(g(X)) = \text{tr}(\mathbb{E}(g(X))$$

$$ii) \quad \mathbb{E}(X^T A X) = \mu^T A \mu + \text{tr}(A\Sigma), \quad \mu := \mathbb{E}(X), \Sigma := \mathbb{V}(X)$$

Two Rules for Expectations

$$i)\quad \mathbb{E}(\mathrm{tr}(g(X)) = \mathrm{tr}(\mathbb{E}(g(X))$$

$$ii)\quad \mathbb{E}(X^T AX) = \mu^T A\mu + \mathrm{tr}(A\Sigma), \quad \mu := \mathbb{E}(X), \Sigma := \mathbb{V}(X)$$

proof:

$$\begin{aligned}
\mathbb{E}(X^T AX) &= \mathbb{E}((\mu+Y)^T A(\mu+Y)), \quad Y := X-\mu, \mathbb{E}(Y)=0, \mathbb{V}(Y) = \\
&= \mu^T A\mu + 2\mu^T A\,\mathbb{E}(Y) + \mathbb{E}(Y^T AY) \\
\mathbb{E}(Y^T AY) &= \mathbb{E}(\mathrm{tr}(Y^T AY)) \quad \text{as } Y^T AT \text{ is a scalar} \\
&= \mathbb{E}(\mathrm{tr}(AYY^T)) \quad \text{as} \quad \text{as } Y^T AT \text{ is a scalar trace allows permutations of matrices} \\
&= \mathrm{tr}(\mathbb{E}(AYY^T)) \\
&= \mathrm{tr}(A\,\mathbb{E}(YY^T)) \\
&= \mathrm{tr}(A\Sigma)
\end{aligned}$$

Learning ARD I: via EM

$$\ell(\sigma_y^2, \Sigma_\beta, \mu_\beta, C_\beta) := E_{\beta\sim\mathcal{N}(\mu_\beta, C_\beta)}(\log p(y \mid X, \beta, \sigma_y^2, \Sigma_\beta))$$

$$= E_{\beta\sim\mathcal{N}(\mu_\beta, C_\beta)}(\log \mathcal{N}(y \mid X\beta, \sigma_y^2) + \log \mathcal{N}(\beta \mid 0, \Sigma_\beta))$$

$$+ \sum_{m=1}^{M} \log \mathrm{InvGamma}(\sigma_{\beta_m}^2 \mid a, b) + \log \mathrm{InvGamma}(\sigma_y^2 \mid c, d)$$

$$\propto E_{\beta\sim\mathcal{N}(\mu_\beta, C_\beta)}(-\frac{N}{2}\log\sigma_y^2 - \frac{1}{2\sigma_y^2}||y - X\beta||^2 - \frac{1}{2}\sum_m \log\sigma_{\beta_m}^2 - \frac{1}{2}\mathrm{tr}\Sigma_\beta^{-1}\beta\beta^T$$

$$+ \sum_{m=1}^{M}(-a\log\sigma_{\beta_m}^2 - b/\sigma_{\beta_m}^2) - c\log\sigma_y^2 - d/\sigma_y^2$$

$$= -\frac{N}{2}\log\sigma_y^2 - \frac{1}{2\sigma_y^2}(||y - X\mu_\beta||^2 + \mathrm{tr}(X^T X C_\beta)) - \frac{1}{2}\sum_m \log\sigma_{\beta_m}^2$$

$$-\frac{1}{2}\mathrm{tr}\Sigma_\beta^{-1}(\mu_\beta\mu_\beta^T + C_\beta) + \sum_{m=1}^{M}(-a\log\sigma_{\beta_m}^2 - b/\sigma_{\beta_m}^2) - c\log\sigma_y^2 - d/\sigma_y^2$$

Learning ARD I: via EM

$$
\begin{aligned}
&\ell(\ldots)\\
=\ & -\frac{N}{2}\log\sigma_y^2 - \frac{1}{2\sigma_y^2}(||y - X\mu_\beta||^2 + \mathrm{tr}(X^T X C_\beta)) - \frac{1}{2}\sum_m \log\sigma_{\beta_m}^2\\
& -\frac{1}{2}\mathrm{tr}\Sigma_\beta^{-1}(\mu_\beta\mu_\beta^T + C_\beta) + \sum_{m=1}^{M}(-a\log\sigma_{\beta_m}^2 - b/\sigma_{\beta_m}^2) - c\log\sigma_y^2 - d/\sigma_y^2\\
\propto\ & -(2c+N)\log\sigma_y^2 - (2d + ||y - X\mu_\beta||^2 + \mathrm{tr}(X^T X C_\beta))\frac{1}{\sigma_y^2}\\
& -\sum_m (2a+1)\log\sigma_{\beta_m}^2 + 2b/\sigma_{\beta_m}^2 - \mathrm{tr}\Sigma_\beta^{-1}(\mu_\beta\mu_\beta^T + C_\beta)
\end{aligned}
$$

Learning ARD I: via EM

$$
\begin{aligned}
\ell(\ldots) = & -(2c+N)\log\sigma_y^2 - (2d + ||y - X\mu_\beta||^2 + \mathrm{tr}(X^T X C_\beta))\frac{1}{\sigma_y^2} \\
& - \sum_m (2a+1)\log\sigma^2_{\beta_m} + 2b/\sigma^2_{\beta_m} - \mathrm{tr}\Sigma_\beta^{-1}(\mu_\beta\mu_\beta^T + C_\beta)
\end{aligned}
$$

$$
0 \overset{!}{=} \frac{\partial\ell}{\partial\sigma^2_{\beta_m}} = -(2a+1)\frac{1}{\sigma^2_{\beta_m}} + (2b + (\mu_\beta)^2_m + (C_\beta)_{m,m})/(\sigma^2_{\beta_m})^2
$$

$$
\sigma^2_{\beta_m} = \frac{2b + (\mu_\beta)^2_m + (C_\beta)_{m,m}}{2a+1}
$$

$$
0 \overset{!}{=} \frac{\partial\ell}{\partial\sigma^2_y}
$$

$$
\rightsquigarrow \sigma^2_y = \frac{2d + ||y - X\mu_\beta||^2 + \mathrm{tr}(X^T X C_\beta)}{2c+N}
$$

which can be accelerated using

$$
C_\beta X^T X = \sigma_y^{2\text{old}}(I - C_\beta\Sigma_\beta^{-1}), \quad \mathrm{tr}(\ldots) = \sigma_y^{2\text{old}}\sum 1 - \frac{(C_\beta)_{m,m}}{\sigma^2_{\beta_m}}
$$

Learning ARD I: via EM

given: data X, y, hyperprior parameters a, b, c, d.
initialize: $\sigma^2_{\beta_m} := 1, \sigma^2_y := 1$
iteratively fit:

$$C_\beta := (\frac{1}{\sigma^2_y} X^T X + \Sigma_\beta^{-1})^{-1}), \quad \Sigma_\beta := \text{diag}(\sigma^2_{\beta_1}, \ldots, \sigma^2_{\beta_M})$$

$$\mu_\beta := \frac{1}{\sigma^2_y} C_\beta X^T y$$

$$\sigma^2_{\beta_m} := \frac{2b + (\mu_\beta)^2_m + (C_\beta)_{m,m}}{2a+1}$$

$$\sigma^2_y := \frac{2d + ||y - X\mu_\beta||^2 + \text{tr}(X^T X C_\beta)}{2c+N}$$

finally yielding:

$$\beta \sim \mathcal{N}(\mu_\beta, C_\beta)$$

Learning ARD II: Fixed Point Algorithm

iteratively fit:

$$
\begin{aligned}
\sigma^2_{\beta_m} &:= \frac{2b + (\mu_\beta)^2_m}{2a + \gamma_m} \\
\sigma^2_y &:= \frac{2d + ||y - X\mu_\beta||^2}{2c + N - \sum_m \gamma_m} \\
C_\beta &:= (\frac{1}{\sigma^2_y} X^T X + \Sigma_\beta^{-1})^{-1} \\
\mu_\beta &:= \frac{1}{\sigma^2_y} C_\beta X^T y \\
\gamma_m &:= 1 - \frac{(C_\beta)_{m,m}}{\sigma^2_{\beta_m}}, \quad m := 1, \ldots, M
\end{aligned}
$$

Learning ARD III: Iteratively Reweighted L1

The ARD regularization term

$$R(\sigma_\beta^2) := \log |C_y(\sigma_\beta^2)| = \log |\sigma_y^2 I + X\Sigma_\beta X^T|, \quad \Sigma_\beta := \text{diag}(\sigma_\beta^2)$$

is concave in σ_β^2 and thus can be written as

$$\begin{aligned} R(\sigma_\beta^2) &= \min_\lambda \lambda^T \sigma_\beta^2 - R^*(\lambda) \\ R^*(\lambda) &= \min_{\tilde\sigma_\beta^2} \lambda^T \tilde\sigma_\beta^2 - \log |C_y(\tilde\sigma_\beta^2)| \end{aligned}$$

The relaxed function

$$R(\sigma_\beta^2, \lambda) := \lambda^T \sigma_\beta^2 - R^*(\lambda) = \lambda^T \sigma_\beta^2 - \min_{\tilde\sigma_\beta^2} \lambda^T \tilde\sigma_\beta^2 - \log |C_y(\tilde\sigma_\beta^2)|$$

for fixed σ_β^2 is minimized by

$$\lambda = \nabla_{\sigma_\beta^2} \log |C_y(\sigma_\beta^2)|$$

Learning ARD III: Iteratively Reweighted L1

Instead of σ^2_β Wipf/Nagarajan 2008 use

$$\sigma^2_{\beta_m} \overset{??}{\rightarrow} \lambda_m^{\frac{1}{2}} |\beta_m|$$

finally yielding the iterative procedure:

$$\beta^{(t+1)} := \arg\min_{\beta} \ell(\beta) + \sum_{m=1}^{M} \lambda_m^{(t)} |\beta_m|$$

and to find $\lambda^{(t)}$:

$$\lambda_m^{(0)} := 1$$

$$\lambda_m^{(t+1)} := (X_{.,m}(\sigma_y^2 I + X \operatorname{diag}(\frac{1}{\lambda_1^{(t)}}, \ldots, \frac{1}{\lambda_M^{(t)}}) \operatorname{diag}(|\beta_1^{(t)}|, \ldots, |\beta_M^{(t)}|))^{-1} X_{.,m})^{\frac{1}{2}}$$

ARD for Classification

- so far, everything was developed for linear regression.
- for logistic regression, for EM the E-step cannot be done analytically.
  - possibly use variational approximation
  - use Gaussian approximation (Laplace approximation)
- the iteratively reweighted learning algorithm still works.

Remarks

- ARD is a good example for a (arguably simple) hierarchical Bayesian model.
- ARD has to be diligently evaluated against simple baselines such as normalizing the data with a vanilla L1/L2 regularized model.

Outline

Model Complexity, Bias & Variance

Example (Linear models)

- $\hat{y}(x) = \beta_1 \cdot x$
- $\hat{y}(x) = (\beta_1 + \beta_2 + \ldots + \beta_K) \cdot x$

Both models have the same bias and variance! $\rightsquigarrow$ redundant parameters!

Model Complexity, Bias & Variance

Example (Linear models)

- $\hat{y}(x) = \beta_1 \cdot x$
- $\hat{y}(x) = (\beta_1 + \beta_2 + \ldots + \beta_K) \cdot x$

Both models have the same bias and variance! $\rightsquigarrow$ redundant parameters!

Example (1-parameter model)

- $\hat{y}(x) = \sin(\theta x)$

Can achieve 100% accuracy on any finite 1D binary classification dataset.
$\longrightarrow$ A single real number can store an infinite amount of information!

Model Complexity, Bias & Variance

Example (Linear models)

- $\hat{y}(x) = \beta_1 \cdot x$
- $\hat{y}(x) = (\beta_1 + \beta_2 + \ldots + \beta_K) \cdot x$

Both models have the same bias and variance! $\rightsquigarrow$ redundant parameters!

Example (1-parameter model)

- $\hat{y}(x) = \sin(\theta x)$

Can achieve 100% accuracy on any finite 1D binary classification dataset.
$\longrightarrow$ A single real number can store an infinite amount of information!

Example (Neural Network)

- Network 1: vanilla MLP
- Network 2: sparse Network with skip connections

Network 2 is more complex when both have same amount of parameters!

Measures of Model Complexity

- Parameter Counting
  - only really works when comparing models with the same architecture
  - even then not guaranteed to be useful

Measures of Model Complexity

- Parameter Counting
  - only really works when comparing models with the same architecture
  - even then not guaranteed to be useful
- Information Criteria (e.g. BIC, AIC)
  - Both very crude tools (lots of approximations used in derivation)
  - Both ignorant about the model architecture

Measures of Model Complexity

- Parameter Counting
  - only really works when comparing models with the same architecture
  - even then not guaranteed to be useful
- Information Criteria (e.g. BIC, AIC)
  - Both very crude tools (lots of approximations used in derivation)
  - Both ignorant about the model architecture
- VC-dimension
  - "What is size the the smallest binary classification problem that the model cannot solve."

Measures of Model Complexity

- Parameter Counting
  - only really works when comparing models with the same architecture
  - even then not guaranteed to be useful
- Information Criteria (e.g. BIC, AIC)
  - Both very crude tools (lots of approximations used in derivation)
  - Both ignorant about the model architecture
- VC-dimension
  - "What is size the the smallest binary classification problem that the model cannot solve."
- Rademacher Complexity
  - "How good can the model simulate noise."

Measures of Model Complexity

- Parameter Counting
  - only really works when comparing models with the same architecture
  - even then not guaranteed to be useful
- Information Criteria (e.g. BIC, AIC)
  - Both very crude tools (lots of approximations used in derivation)
  - Both ignorant about the model architecture
- VC-dimension
  - "What is size the the smallest binary classification problem that the model cannot solve."
- Rademacher Complexity
  - "How good can the model simulate noise."
- Kolmogorov Complexity & Minimum Description Length
  - "What is the minimal size of a program that implements the model."
  - **uncomputable!**

Kolmogorov Complexity - Mandelbrot Fractal

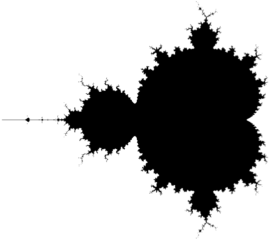
Generated by a simple formula:

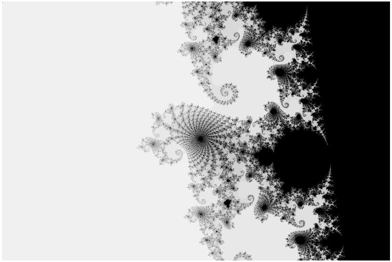
Does the iteration

$$z_{k+1} = z_k^2 + c \quad z_0 = 0$$

diverge? (with $z, c \in \mathbb{C}$)

- Yes: c belongs to class 1 (white)
- No: c belongs to class 0 (black)





images: wikipedia.org

Kolmogorov Complexity - Mandelbrot Fractal

Generated by a simple formula:

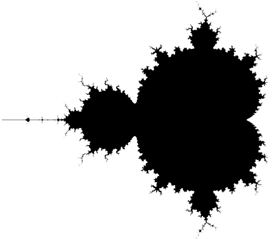
Does the iteration

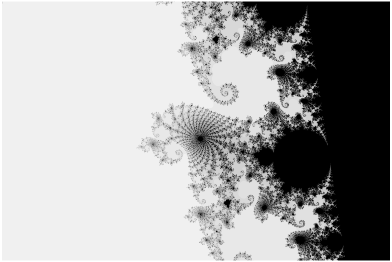
$$z_{k+1} = z_k^2 + c \quad z_0 = 0$$

diverge? (with $z, c \in \mathbb{C}$)

- Yes: c belongs to class 1 (white)
- No: c belongs to class 0 (black)

Very simple rules lead to incredible complexity.





images: wikipedia.org

Kolmogorov Complexity - Mandelbrot Fractal

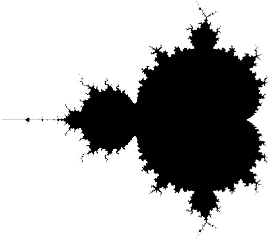
Generated by a simple formula:

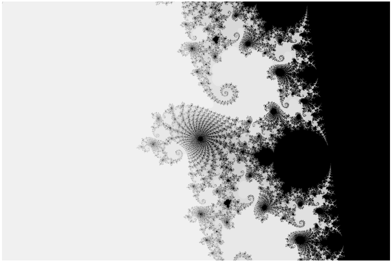
Does the iteration

$$z_{k+1} = z_k^2 + c \quad z_0 = 0$$

diverge? (with $z, c \in \mathbb{C}$)

- Yes: c belongs to class 1 (white)
- No: c belongs to class 0 (black)





images: wikipedia.org

Very simple rules lead to incredible complexity.

It would be very hard to reconstruct the rules, if we only know the image. In fact, in general it is impossible!
$\leadsto$ uncomputability

Further Readings

- L1 regularization: [**?**, chapter 13.3–5], [**?**, chapter 3.4, 3.8, 4.4.4], [**?**, chapter 3.1.4].
  - LAR, LARS: [**?**, chapter 3.4.4], [**?**, chapter 13.4.2],
- Non-convex regularizers: [**?**, chapter 13.6].
- Automatic Relevance Determination (ARD): [**?**, chapter 13.7], [**?**, chapter 11.9.1], [**?**, chapter 7.2.2].
  - see also [**?**].
- Sparse Coding: [**?**, chapter 13.8].

References

Christopher M. Bishop.
Pattern recognition and machine learning, volume 1.
springer New York, 2006.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin.
The elements of statistical learning: data mining, inference and prediction, volume 27.
Springer, 2005.

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.

David P. Wipf and Srikantan S. Nagarajan.
A New View of Automatic Relevance Determination.
In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1625–1632. Curran Associates, Inc., 2008.