Last lecture

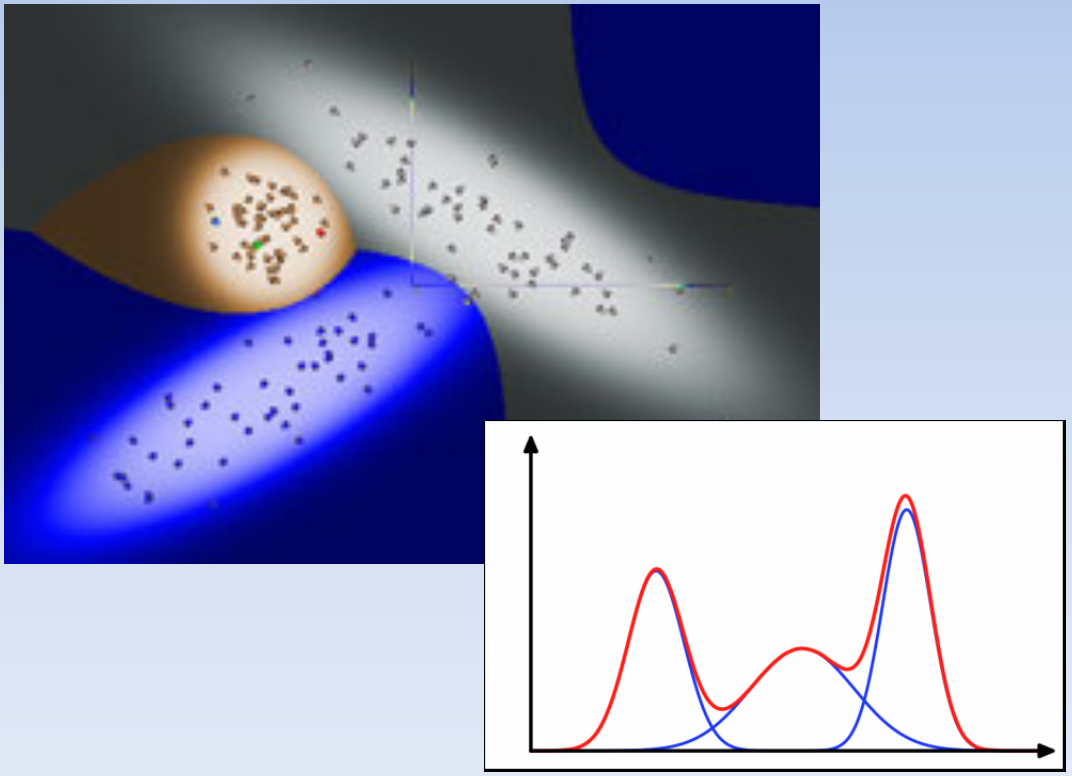
UCB - CS189
Introduction to Machine Learning
Fall 2015
Lecture 15: Gaussian Processes

Isabelle Guyon

ChaLearn

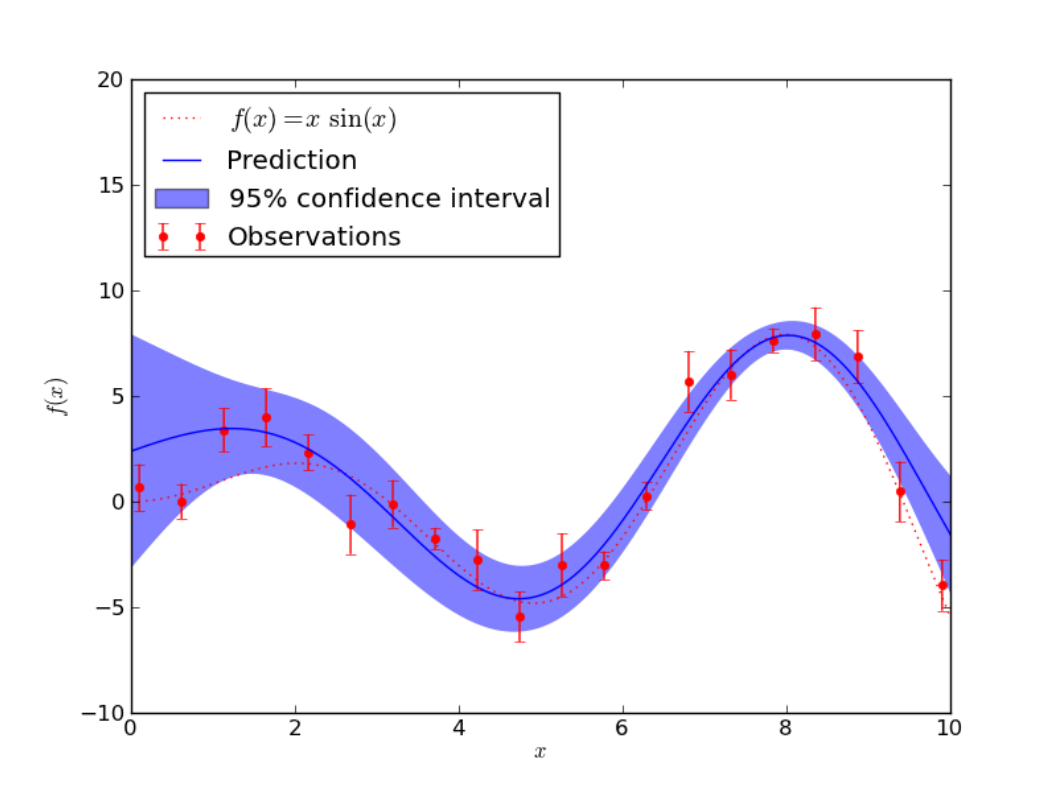
Come to my office hours…
Wed 2:30-4:30 Soda 329

Last time: Mixture models



Come to my office hours…
Wed 2:30-4:30 Soda 329

Today: Gaussian processes



Review

Data generating models

$$P(X, Y) = P(X)P(Y|X) = P(Y)P(X|Y)$$

P(X)

P(Y|X)

96 98 78 6 63 79…

5

P(Y)

P(X|Y)

Discriminative models

$$f(\mathbf{x}) = \left(\sum_k \alpha_k \left[\Phi(\mathbf{x}^k) \right) \bullet \Phi(\mathbf{x}) \right]$$

$$\mathbf{w} \qquad k(\mathbf{x}^k, \mathbf{x}; \theta)$$

Algorithm:
Hebb's rule
$\mathbf{w} = \Sigma_{k=1:N} y^k \mathbf{x}^k$
$\mathbf{w} = X^T \mathbf{y}$

Hebb's method
- $\alpha_{k=} y_k$
- $k(\mathbf{x}^k, \mathbf{x}) = \mathbf{x}^k \bullet \mathbf{x}$
- No θ
- $f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x}$

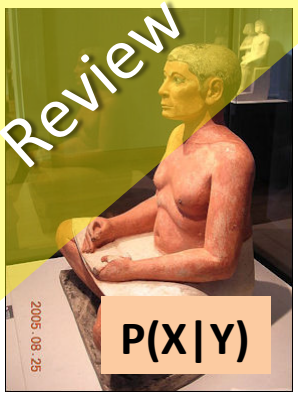
"Perceptrons"
- α_k fitted to R_{train}
- $k(\mathbf{x}^k, \mathbf{x}) = \mathbf{x}^k \bullet \mathbf{x}$
- No θ
- $f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x}$

Algorithms:
SVM
Logistic regression
Ridge regression
$\mathbf{w} = X^+ \mathbf{y} = \Sigma^{-1} X^T \mathbf{y}$

Parzen windows
- $\alpha_{k=} y_k$
- Any $k(\mathbf{x}^k, \mathbf{x}; \theta)$
- θ fitted to R_{CV}
- $f(\mathbf{x}) = \Sigma_k \alpha_k k(\mathbf{x}^k, \mathbf{x}; \theta)$

Kernel methods
- α_k fitted to R_{train}
- Any $k(\mathbf{x}^k, \mathbf{x}; \theta)$
- θ fitted to R_{CV}
- $f(\mathbf{x}) = \Sigma_k \alpha_k k(\mathbf{x}^k, \mathbf{x}; \theta)$

Generative models



Today

Algorithm:
Centroid
$\mathbf{w} = \Sigma_k y^k \mathbf{x}^k = X^T \mathbf{y}$
$y^k \in \{+1/N_1, -1/N_0\}$
$\mathbf{w} = \mu^{[1]} - \mu^{[0]}$

LDA

Mixture models

Naïve Bayes

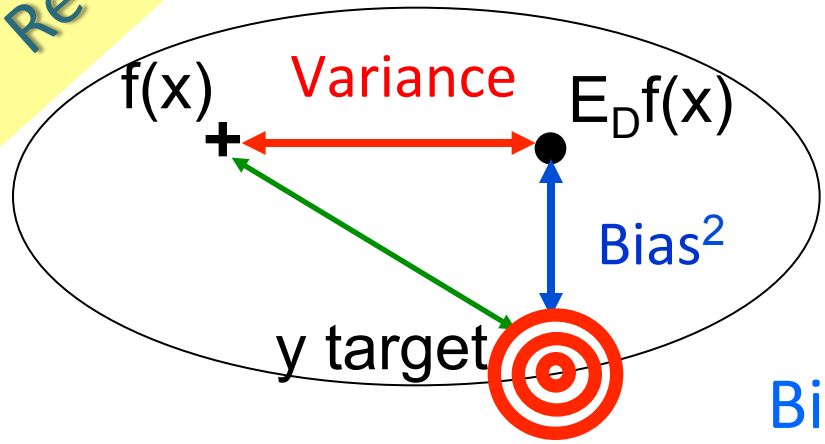
Algorithm:
$\mathbf{w} = \Sigma^{-1}(\mu^{[1]} - \mu^{[0]})$
$\Sigma = X^T X + \lambda I, \quad \lambda > 0$
$\mathbf{w} = \Sigma^{-1} X^T \mathbf{y}$
$y^k \in \{+1/N_1, -1/N_0\}$

Gaussian processes

Last time

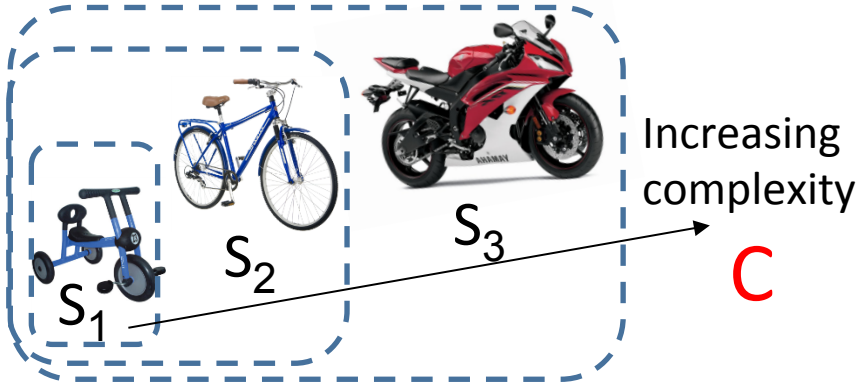
Review

Wrong assumptions don't really matter!



Bias *vs.* variance tradeoff

$$E_D[f(\mathbf{x})-y]^2 = [E_D f(\mathbf{x})-y]^2 + E_D[f(\mathbf{x})-E_D f(\mathbf{x})]^2$$



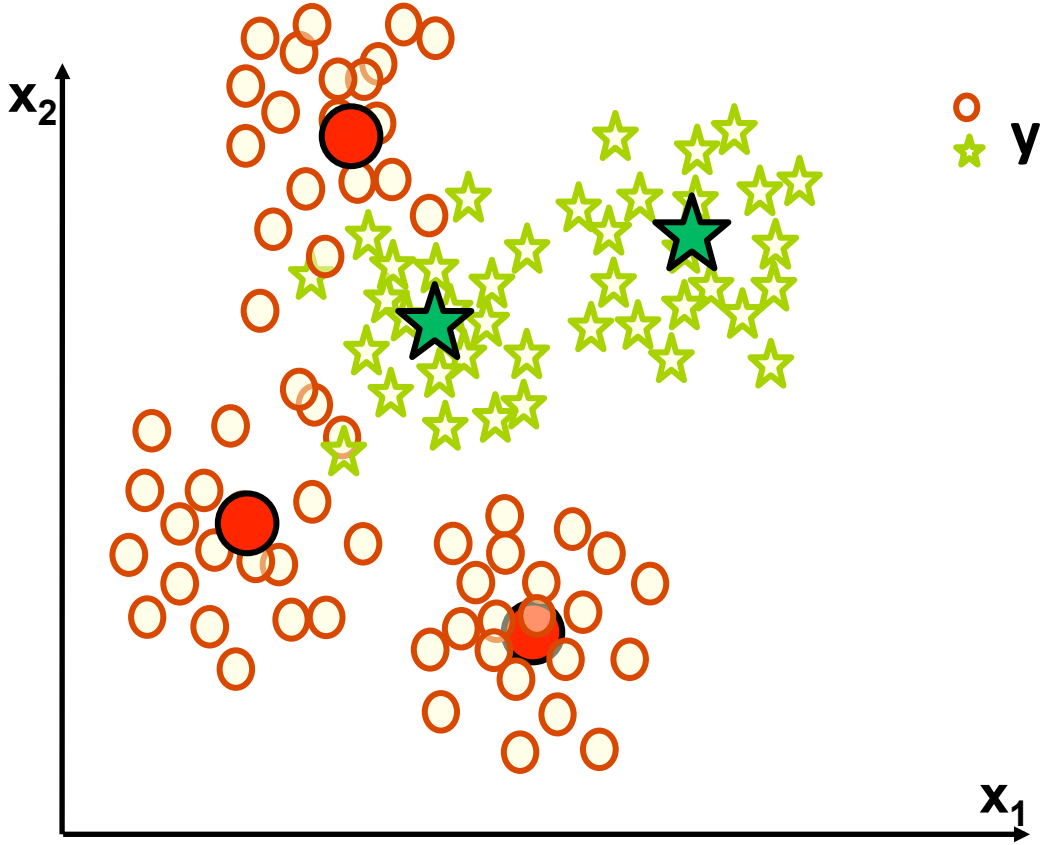
Structural risk minimization

$$R[f] \le R_{emp}[f] + \varepsilon(\delta, C/N)$$

Bias is "good",

if lowering C helps you get better generalization error

Mixture models

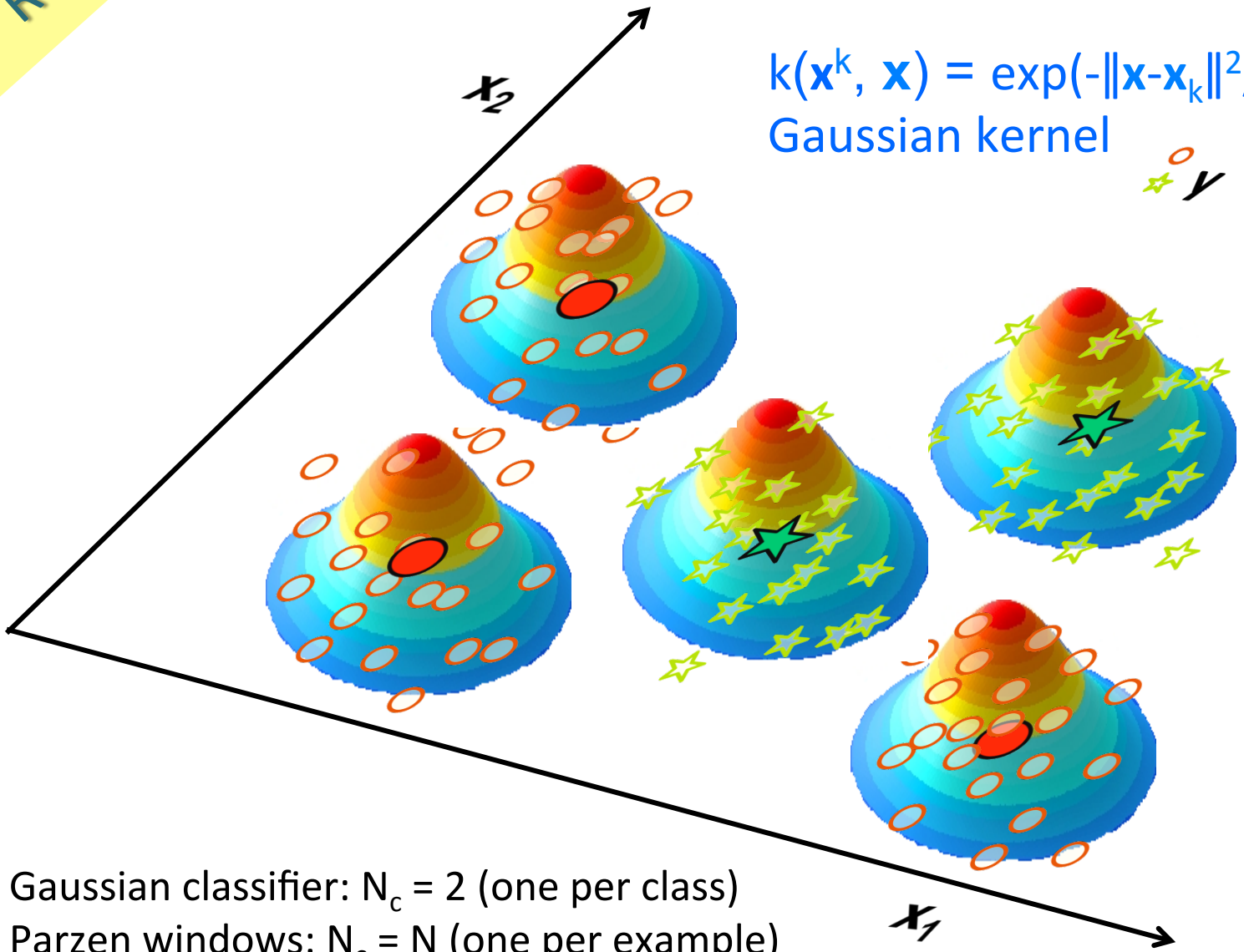


$$f(\mathbf{x}) = \Sigma_{k=1:Nc}\, \alpha_k\, k(\boldsymbol{\mu}^k, \mathbf{x}; \boldsymbol{\sigma}) + b$$

Review

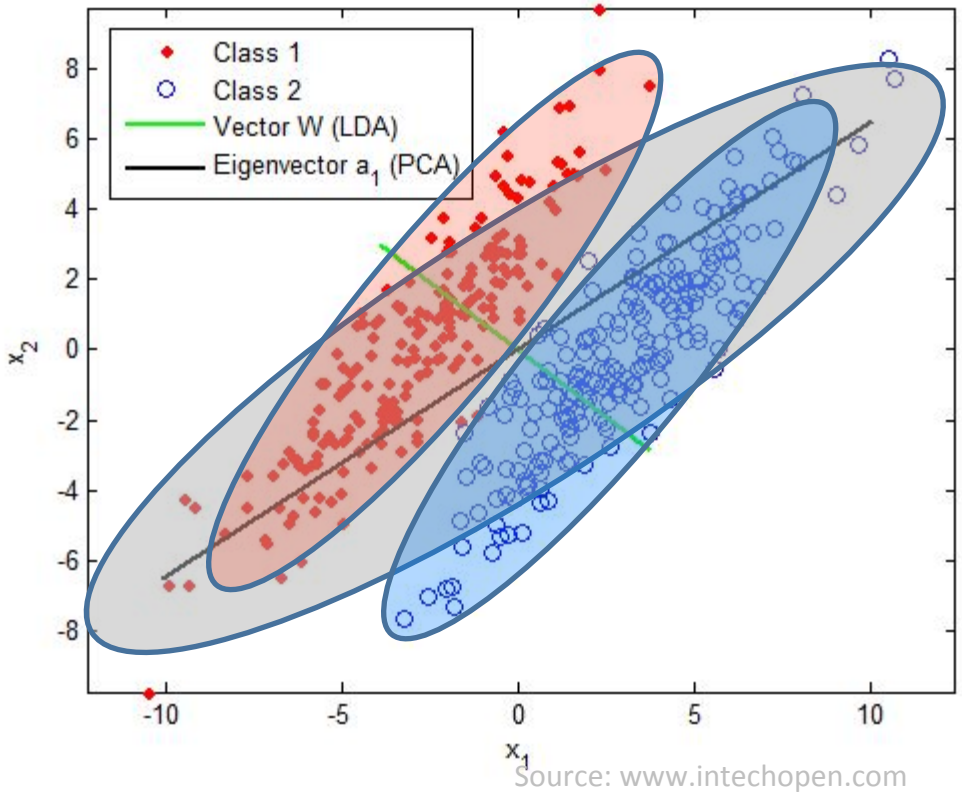
Gaussian mixtures

$k(\mathbf{x}^k, \mathbf{x}) = \exp(-\|\mathbf{x}-\mathbf{x}_k\|^2/2\sigma^2)$
Gaussian kernel



Gaussian classifier: $N_c = 2$ (one per class)
Parzen windows: $N_c = N$ (one per example)

How to “kernelize” PCA and LDA?



Source: www.intechopen.com

Apply whitening in both case to get $\Xi = X \Sigma^{-1/2}$

PCA: Σ is the TOTAL covariance matrix.

LDA: Σ is the POOLED WITHIN CLASS covariance matrix.

Σ^{-1} for BIG data

X^TX dim(d,d)
XX^T dim(N,N)
which one do I rather invert?

Singular value decomposition:

$X=VSU^T$, with $U^TU=1$ and $V^TV=1$

S diagonal dim(r, r): singular values, r = rank(X)≤min(d,N)

$X^TX=US^2U^T$ and $XX^T=VS^2V^T$ dim(U)=(d, r) dim(V)=(N, r)

cheap $\Xi = XU$, $\Sigma^{1/2} = USU^T$, $\Sigma^{-1} = US^{-2}U^T$, etc.

r is small + keep only largest singular values.

Kernel trick:

$d>>N \rightarrow XX^T = VS^2V^T$ $XX^T \rightarrow K$ (kernel trick)

Instead of $\Xi = XU$ use $\Xi = VS$, much cheaper!

$\Xi_{new} = X_{new} U$? $X=VSU^T$ so $U = X^TVS^{-1}$

$\Xi_{new} = X_{new} X^TVS^{-1}$ $X_{new} X^T \rightarrow [k(x^h, x^k)]$ (kernel trick)

Parameter estimation

- Maximum likelihood (ML):

Param = *argmax*(P(data|model))

likelihood

Examples:

$\boldsymbol{\mu}^{[y]} = \sum_{\mathbf{x}^k \in y} \mathbf{x}^k$

$\sigma_{ij} = (1/N) \sum_{k=1:N} (x_i^k - \mu_i)(x_j^k - \mu_j)$

- Maximum A Posteriori (MAP):

Param = *argmax*(P(model|data))

posterior

= *argmax*(P(data|model)P(model))

likelihood prior

Bayesian MAP ≅ SRM

- Maximum A Posteriori (MAP):

$$f = \text{argmax } P(f|D)$$
$$= \text{argmax } P(D|f)\,P(f)$$
$$= \text{argmin } \underbrace{-\log P(D|f)}_{\text{Negative log likelihood}} \quad \underbrace{-\log P(f)}_{\text{Negative log prior}}$$

Negative log likelihood = Empirical risk $R_{emp}[f]$

Negative log prior = Regularizer $\Omega[f]$

- Structural Risk Minimization (SRM):

$$f = \text{argmin } R_{emp}[f] + \Omega[f]$$

Example: Ridge regression a.k.a. Gaussian processes

- Structural Risk Minimization (SRM):

$$\mathbf{w} = \text{argmin}_{\mathbf{w}} \underbrace{\| \mathbf{y} - X \mathbf{w}^T \|^2}_{\text{Negative log likelihood}} + \underbrace{\lambda \| \mathbf{w} \|^2}_{\text{Negative log prior}}$$

- Maximum A Posteriori (MAP):

$$\mathbf{w} = \text{argmin}_{\mathbf{w}} -\log P(D|f) \quad -\log P(f)$$
$$= \text{argmax}_{\mathbf{w}} P(D|f)\, P(f)$$
$$= \text{argmax}_{\mathbf{w}} \exp(-\| \mathbf{y} - X \mathbf{w}^T \|^2) \exp(-\lambda \| \mathbf{w} \|^2)$$
$$= \text{argmax}_{\mathbf{w}} \exp(-\|\mathbf{y} - X \mathbf{w}^T\|^2) \exp(-\| \mathbf{w} \|^2 / 2\sigma^2)$$

Bayesian "learning"

- Remember the idea of ensembles:

Do not select a model, "vote" on all possible models according to their "performance".

Replace:

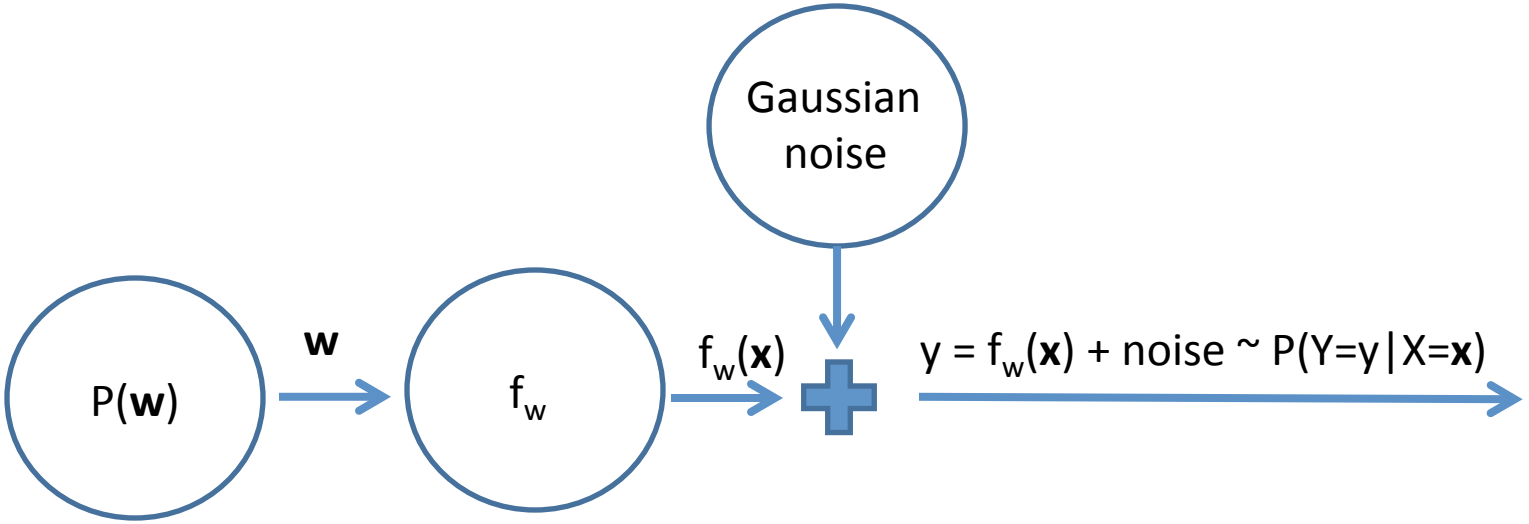
$$\mathbf{w} = \text{argmax}_{\mathbf{w}} \exp(-\|\mathbf{y} - X\,\mathbf{w}^T\|^2) \exp(-\|\,\mathbf{w}\,\|^2 / 2\sigma^2)$$

$$f(\mathbf{x}) = \mathbf{w}\,\mathbf{x}^T \qquad \sim P(\mathbf{w}\,|\,\text{data})$$

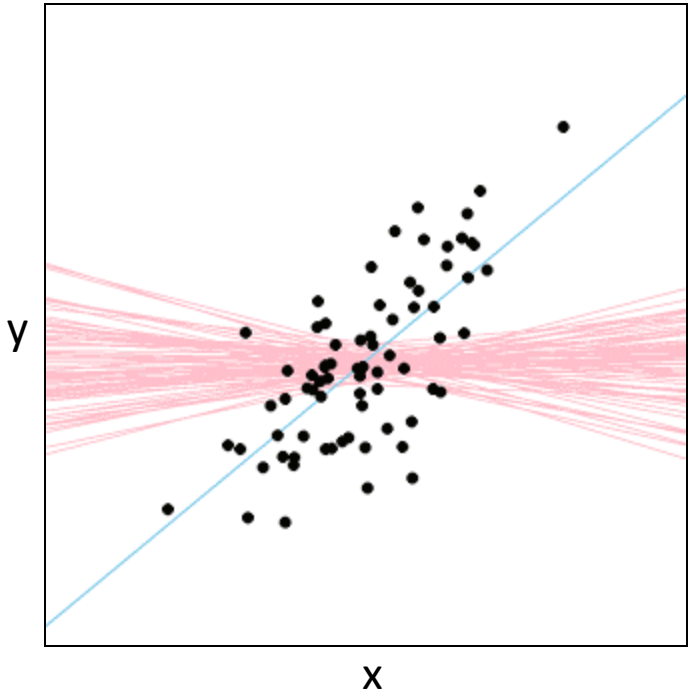
by

$$P(y\,|\,\mathbf{x}) = \int_w P(y\,|\,x, \mathbf{w})\, P(\mathbf{w}\,|\,\text{data})\, d\mathbf{w}$$

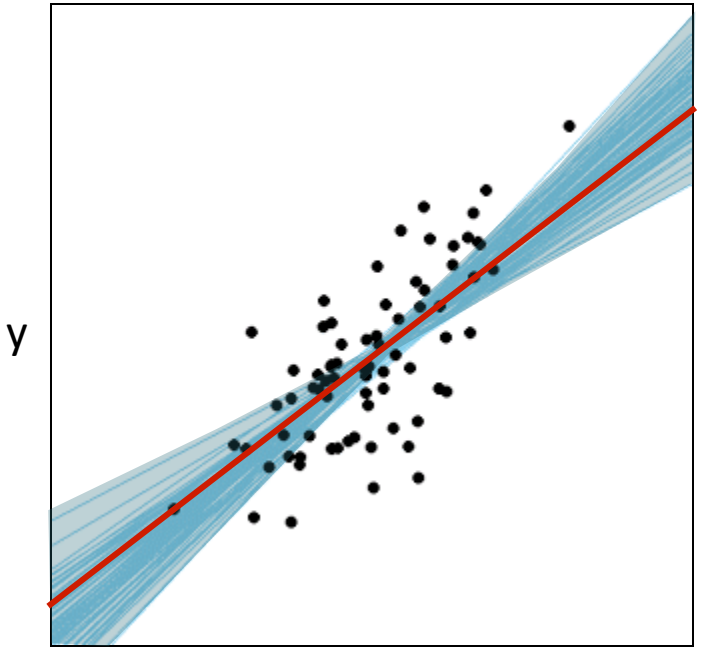
Generative model



Linear Gaussian Processes



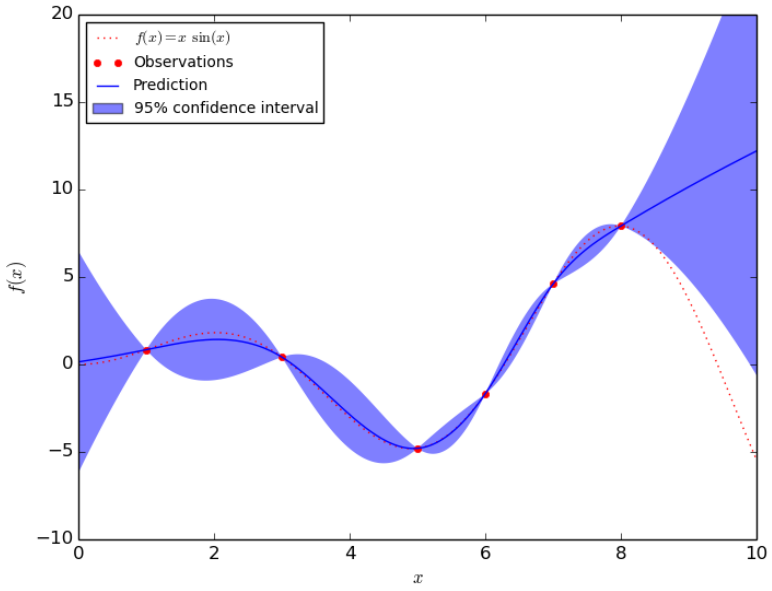
Blue: frequentist regression
Red: prior on **w**

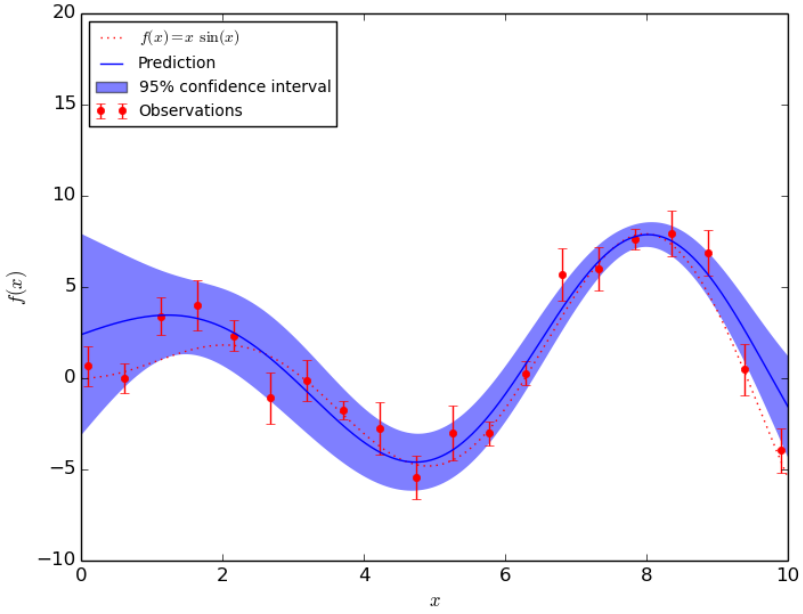


Blue: Bayesian regression

http://doingbayesiandataanalysis.blogspot.com/2013_12_01_archive.html

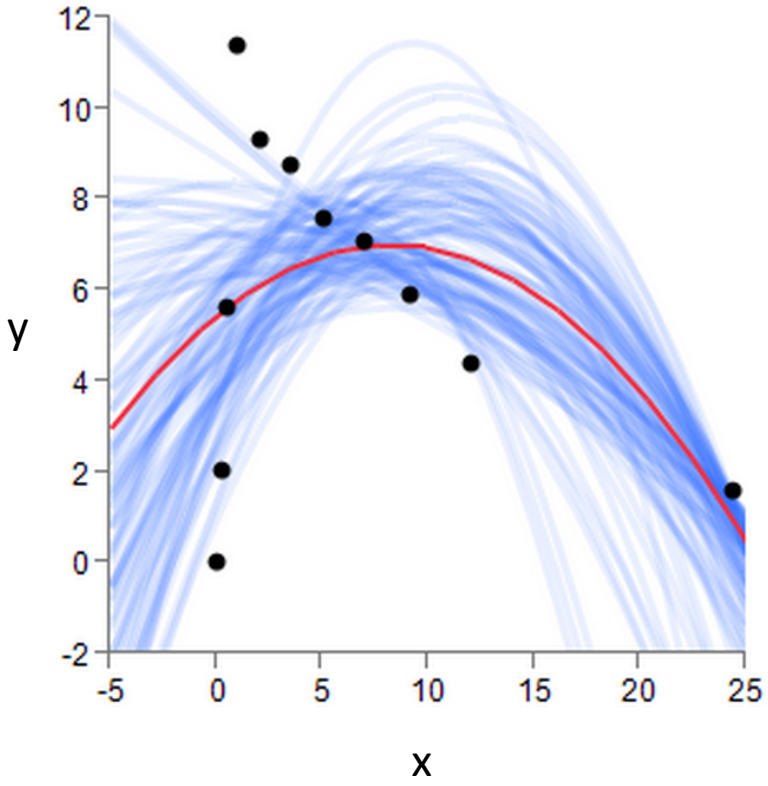
Kernel Gaussian Processes





http://scikit-learn.org/stable/auto_examples/gaussian_process/plot_gp_regression.html

Bootstrap



http://blogs.sas.com/content/jmp/2012/12/04/visualizing-the-variability-of-a-curve-using-bootstrapping/

Summary

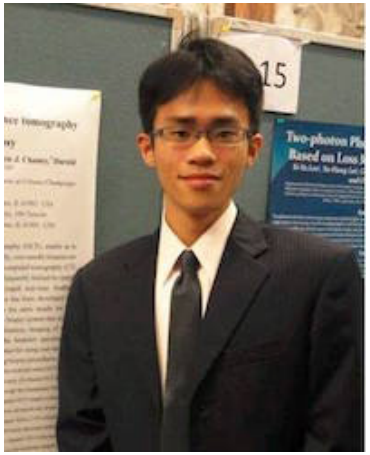
- **Kernel trick for generative models:** When d >> N the covariance matrix X^TX is too large, we can use XX^T instead in many algorithms. This lends itself to "kernelization":
  - Kernel PCA,
  - Kernel LDA,
  - Kernel ridge regression a.k.a. "Gaussian processes".
- **Parameter estimation:** From a data generating point of view, **Maximum Likelihood (ML)** makes sense as a parameter estimation paradigm:
  - The "classical" ways of estimating mean and covariance are ML estimators.
  - Regularization has also its equivalent in the data generating setting: the Maximum A Posteriori (MAP) framework.
- **Gaussian processes:** For regression, Gaussian processes allow you to estimate P(Y|X), so compute error bars of predictions.

Thanks

Alexei (Alyosha) Efros

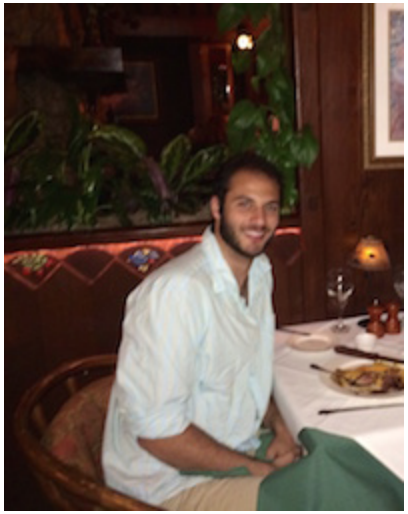
Nihar B. Shah

Brian Chu



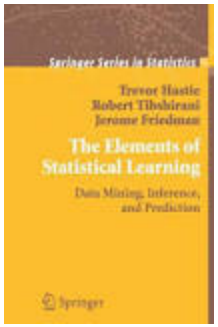
Weicheng Kuo

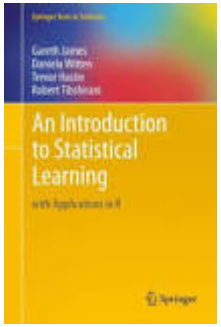
Deepak Pathak



Shaun Singh

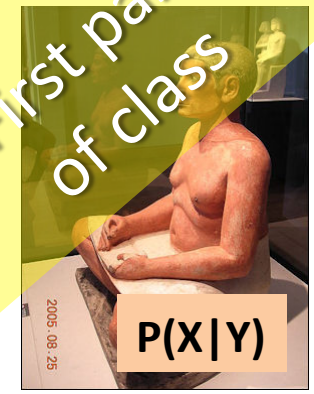
Faraz Tavakoli


Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Springer


An Introduction to Statistical Learning
Springer

Thanks

Generative models



Today

Algorithm:
Centroid
$\mathbf{w} = \Sigma_k y^k \mathbf{x}^k = X^T \mathbf{y}$
$y^k \in \{+1/N_1, -1/N_0\}$
$\mathbf{w} = \mu^{[1]} - \mu^{[0]}$

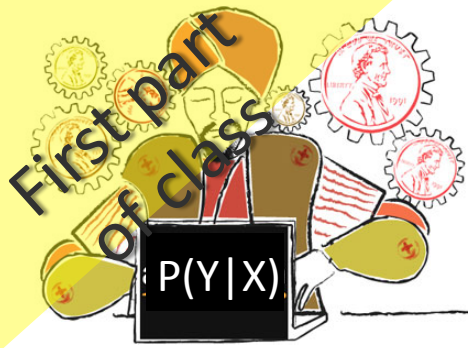
LDA

Gaussian processes

Naïve Bayes

Algorithm:
$\mathbf{w} = \Sigma^{-1}(\mu^{[1]} - \mu^{[0]})$
$\Sigma = X^T X + \lambda I, \quad \lambda > 0$
$\mathbf{w} = \Sigma^{-1} X^T \mathbf{y}$
$y^k \in \{+1/N_1, -1/N_0\}$

Mixture models



Discriminative models

$$f(\mathbf{x}) = \left(\sum_k \alpha_k \left[\Phi(\mathbf{x}^k) \right) \bullet \Phi(\mathbf{x}) \right]$$

$$\mathbf{w} \qquad k(\mathbf{x}^k, \mathbf{x}; \theta)$$

Algorithm:
Hebb's rule
$\mathbf{w} = \Sigma_{k=1:N}\, y^k \mathbf{x}^k$
$\mathbf{w} = X^T \mathbf{y}$

Hebb's method

- $\alpha_{k=} y_k$
- $k(\mathbf{x}^k, \mathbf{x}) = \mathbf{x}^k \bullet \mathbf{x}$
- No θ
- $f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x}$

"Perceptrons"

- α_k fitted to R_{train}
- $k(\mathbf{x}^k, \mathbf{x}) = \mathbf{x}^k \bullet \mathbf{x}$
- No θ
- $f(\mathbf{x}) = \mathbf{w} \bullet \mathbf{x}$

Algorithms:
SVM
Logistic regression
Ridge regression
$\mathbf{w} = X^+ \mathbf{y} = \Sigma^{-1} X^T \mathbf{y}$

Parzen windows

- $\alpha_{k=} y_k$
- Any $k(\mathbf{x}^k, \mathbf{x}; \theta)$
- θ fitted to R_{CV}
- $f(\mathbf{x}) = \Sigma_k\, \alpha_k\, k(\mathbf{x}^k, \mathbf{x}\,; \theta)$

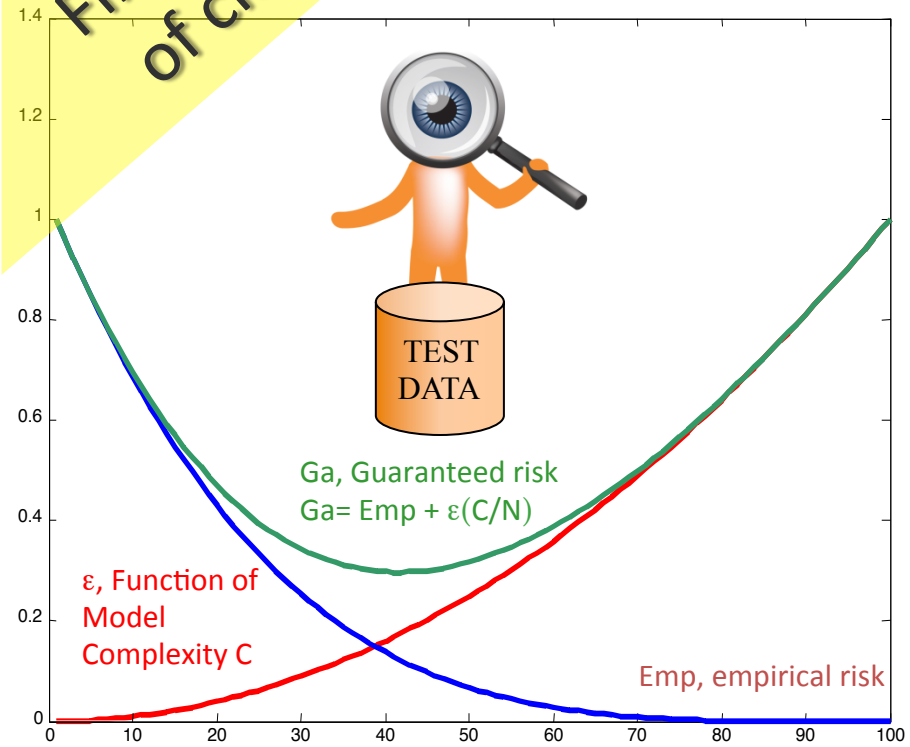
Kernel methods

- α_k fitted to R_{train}
- Any $k(\mathbf{x}^k, \mathbf{x}; \theta)$
- θ fitted to R_{CV}
- $f(\mathbf{x}) = \Sigma_k\, \alpha_k\, k(\mathbf{x}^k, \mathbf{x}\,; \theta)$

First part of class

Structural Risk Minimization

Vapnik, 1974



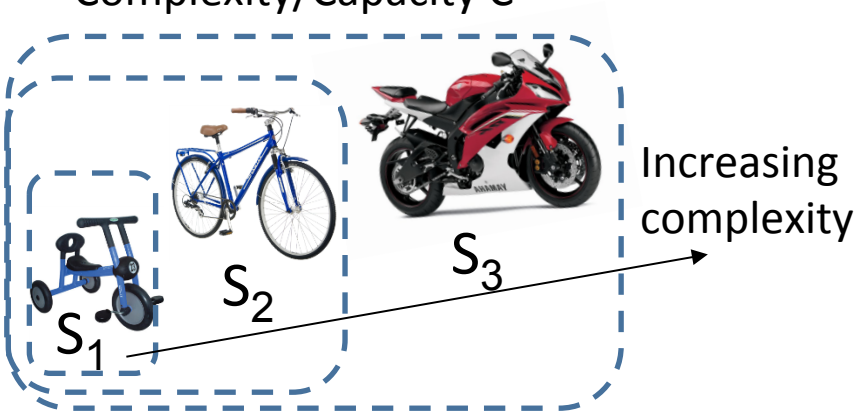
With *high* probability $(1-\delta)$, $0<\delta<<1$

$$R[f] \le \underbrace{R_{emp}[f] + \varepsilon(\delta, C/N)}_{R_{gua}[f]}$$

$$-\log P(D \mid f) \sim -\log P(D \mid f) - \log P(f)$$

Negative log likelihood = Empirical risk R[f]

Negative log prior = Regularizer $\Omega[f]$

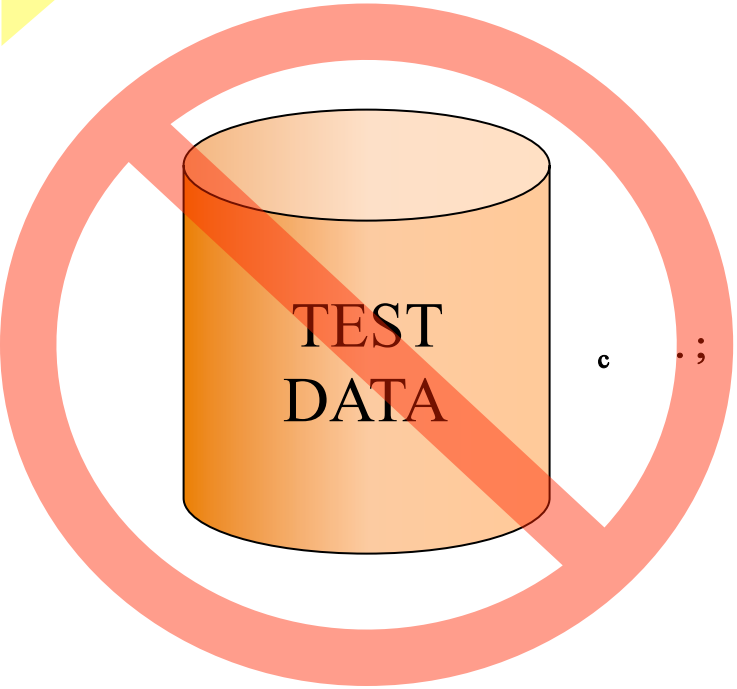


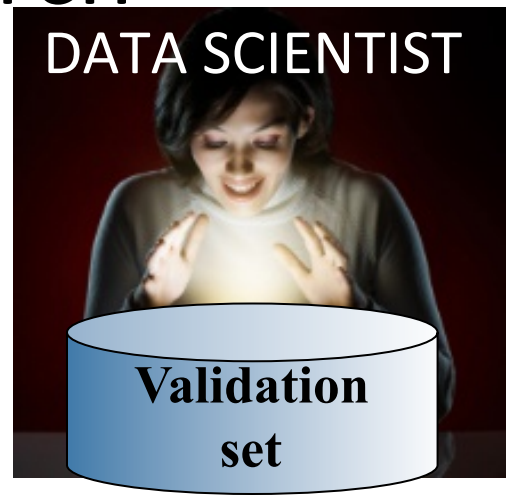
Make nested subsets of models, increasing complexity/capacity:

$S_1 \subset S_2 \subset \ldots S_N$

First part of class

Model selection / hyper-parameter search





Training
set

$$f^{**} = \text{argmin}_{\theta} R_2[f^*, D] \text{ , such that } f^* = \text{argmin}_{\alpha} R_1[f, D]$$

First part of class

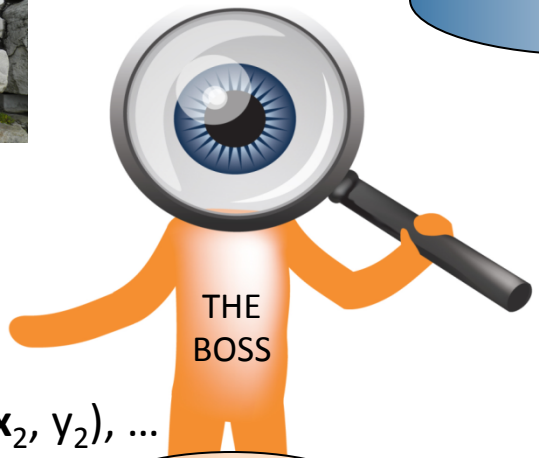
Your turn to play…

P(X, Y)

TRAINING DATA

Validation set

Training set



I.I.D. examples ($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), …

TEST DATA

THE ML "STAFF"

First part of class

Drive safely!

Keep C/N (or d_{eff}/N) under control

Come to my office hours…
Wed 2:30-4:30 Soda 329

Next time: Prof. Efros!