# Introduction to machine learning

Maksim Kretov

Lecture 2: Basics of machine learning

5vision, 2017

# Course information

**Course**

10 lectures + 2 seminars; February-May 2017.

**Schedule and up-to-date syllabus**

https://goo.gl/xExEuL

**Contact information and discussion**

Maksim Kretov (kretovmk@gmail.com)

Slack group: https://miptmlcourse.slack.com

to get an invite, send e-mail to kretovmk@gmail.com.

# Plan of the course

| | | | |
|---|---|---|---|
| <u>Math and basics of ML</u> | (1-2) | ← **Today** | *Theoretical tasks* |
| Some of ML methods | (3) | | |
| *Seminar on ML basics* | (4) | | |
| Basics of neural networks | (5) | ← **Start playing with NNs** | *Practical tasks* |
| Deep learning overview | (6) | | |
| Training deep networks | (7) | | |
| DL for Computer Vision | (8-9) | **Solving more complex ML tasks using NNs** | |
| DL for time series prediction | (10-11) | | |
| *Concluding seminar* | (12) | | |

# Plan for the lecture

A. Reminders
1. Math and probability theory
2. Previous lecture

B. Decision theory
1. Bayesian approach
2. Frequentist approach

C. Introduction to machine learning (continued)
1. Bias-variance tradeoff
2. Empirical risk minimization
3. Generalization ability

D. Model selection
1. Cross-validation
2. Bayesian approach

E. Homework

# A.1 Reminders: Math and probability theory

**Matrix calculus for $f: \mathrm{R}^n \to \mathrm{R}$**

Gradient: $\nabla_x f(x) = [\partial f(x)/\partial x_i]$;

Hessian: $\nabla_x^2 f(x) = \left[\partial^2 f(x)/\partial x_i \partial x_j\right]$;

for $g(x) = [g_i(x)]$:

Jacobian: $\nabla_x g(x) = \left[\partial g_i(x)/\partial x_j\right]$

**Quadratic and linear functions:**

$\nabla_x b^T x = b$

$\nabla_x x A^T x = 2Ax$ (if $A$ is symmetric)

$\nabla_x^2 x A^T x = 2A$ (if $A$ is symmetric)

***Exercise:*** *prove formulas for quadratic and linear functions.*

See detailed review in [1]

# A.1 Reminders: Math and probability theory

**Probability theory**

Joint distribution: $p(X = x, Y = y) \triangleq p(x, y)$ (for brevity)

Marginal distribution: $p(y) = \sum_x p(x, y)$

Conditional distribution: $p(x|y)$

Probability chain rule: $p(x, y) = p(x|y)p(y)$

Expectation: $\mathrm{E}[x] = \sum_x xp(x)$ Variance: $\mathrm{Var}[x] = \sum_x (x - \mathrm{E}[x])^2 p(x)$

**Bayes rule:** $p(x|y) = \frac{p(y|x)p(x)}{\sum_x p(y|x)p(x)}$

See detailed review in [1]

# A.1 Reminders: Math and probability theory

**Statistics**

Sample mean: $\hat{\mu} = \frac{1}{N}\sum_n x_n$ ***Exercise:** Prove it is unbiased estimator.*

Sample variance: $\hat{\sigma}^2 = \frac{1}{N-1}\sum_n (x_n - \hat{\mu})^2$ ***Exercise:** Prove it is unbiased estimator.*

Independent identically distributed: $p(x_1, .. x_N) = \prod_n p(x_n)$

(i.i.d. hypothesis)

**Information theory**

KL divergence: $\mathrm{KL}(p||q) = \sum_k p_k \ln \frac{p_k}{q_k}$

Entropy: $\mathrm{H}(p) = \sum_k p_k \ln p_k$

***Exercise:** Prove that KL divergence is non-negative.*

See detailed review in [1]

# A.2 Reminders: Previous lecture

## Supervised learning

Training set: $\boldsymbol{D} = \{(\mathbf{x}_{\boldsymbol{n}}, y_n), n = 1, .. N\}$ *(inputs and labels!)*

$Y$ are class ids => classification *(make partition of space)*

$Y \in \mathrm{R}$ => regression *(show how data are generated)*

$\mathbf{X}$ – $(N \times d)$ design matrix (features are d-dimensional)

**=> Predict $\boldsymbol{y}^*$ for new input $\mathbf{x}^*$**

## Unsupervised learning

Training set: $D = \{\mathbf{x}_{\boldsymbol{n}}, n = 1, .. N\}$ *(just inputs!)*

$\mathbf{X}$ – design matrix ( $N \times d$)

**=> Find compact description of data**

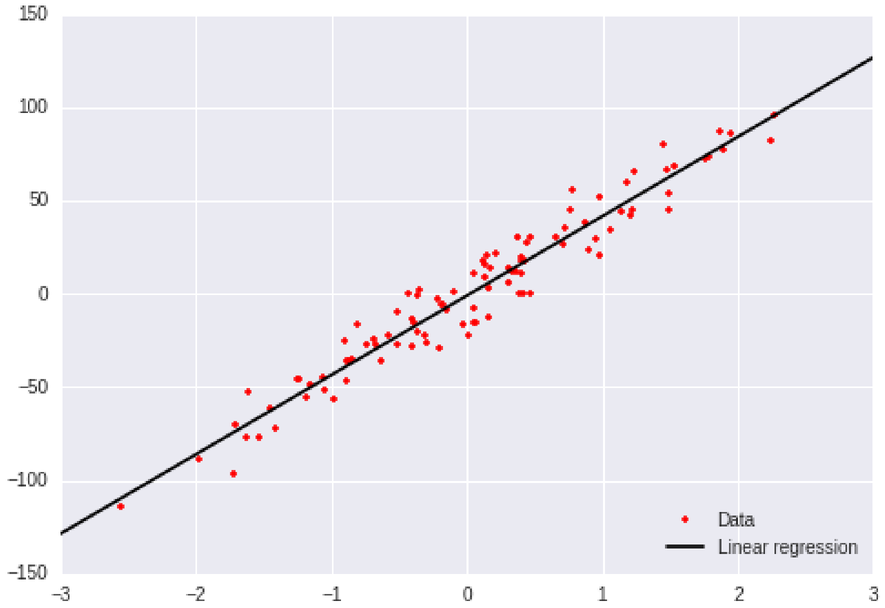
# A.2 Reminders: Previous lecture

## Deterministic view (Least Squares)

**Idea:** Convert ML task into optimization task by considering parameters $\theta$ of the model $f_\theta(\mathbf{x})$ as unknown but fixed.

1. Adopt parametric functional form $f_\theta(\mathbf{x})$
2. Choose loss function $L(y, \hat{y})$: $\mathbf{R}\mathrm{x}\mathbf{R} \rightarrow [0, \infty)$
3. Estimate values of $\theta$: $\theta_{opt} = \mathrm{argmin}_\theta \sum_n L_n\left(y, f_\theta(x_n)\right)$

**!! Parameters of predicting model**




Examples of loss function:

least squares (regression or classification)

cross-entropy (classification)

# A.2 Reminders: Previous lecture

## Probabilistic perspective (Maximum Likelihood Estimation)

**Idea:** Consider joint distribution $p(x, y|\theta)$ and treat $\theta$ as fixed but unknown parameters.

Given:

$$\boldsymbol{D} = \{(\mathbf{x}_{\boldsymbol{n}}, y_n), n = 1, .. N\}$$

Parametric distribution $p(\mathbf{x}, y|\theta)$

Select optimal parameters:

$$\theta_{opt} = \mathrm{argmax}_\theta p(\boldsymbol{D}|\theta)$$

**!! Parameters of probability distribution**

***Exercise:*** *Compare a) parameter estimation through minimization of KL divergence between empirical distribution and true distribution and b) MLE estimator.*

# A.2 Reminders: Previous lecture

## Probabilistic perspective (Bayesian Inference)

**Idea:** Consider joint distribution $p(x, y|\theta)$ and treat $\theta$ as random variables.

1. Infer posterior distribution $p(\theta|\boldsymbol{D})$:

$$p(\theta|\boldsymbol{D}) = \frac{p(\boldsymbol{D}|\theta)p(\theta)}{p(\boldsymbol{D})}$$

**!! Parameters of probability distribution**

$$p(\boldsymbol{D}) = \int p(\boldsymbol{D}|\theta)p(\theta)d\theta$$

2. Calculate distribution for new input $\mathbf{x}$:

$$p(\mathbf{x}, y|\boldsymbol{D}) = \int p(\mathbf{x}, y|\theta)p(\theta|\boldsymbol{D})d\theta$$

Point estimate: $\theta_{MAP} = \mathrm{argmax}_\theta p(\boldsymbol{D}|\theta)p(\theta)$

***Exercise:** prove formulas in linear regression example.*

**Example (Bayesian inference)**

Simplified linear regression.

$y = C + \varepsilon$

$p(\theta) = N(\theta_0, \sigma_0^2)$

$$p(\theta|\boldsymbol{D}) = \frac{p(\theta)}{p(\boldsymbol{D})}\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_\varepsilon}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\varepsilon^2}\right) =$$

$$= \frac{p(\theta)}{p(\boldsymbol{D})}\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_\varepsilon}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\varepsilon^2}\right)$$

$$\theta_N = \frac{N\sigma_0^2\bar{y}_N + \sigma_\varepsilon^2\theta_0}{N\sigma_0^2 + \sigma_\varepsilon^2} \quad \sigma_N^2 = \frac{\sigma_\varepsilon^2\sigma_0^2}{N\sigma_0^2 + \sigma_\varepsilon^2}$$

# A.2 Reminders: Previous lecture

## Bayesian Inference

Posterior $p(\theta|\boldsymbol{D})$ summarizes everything we know about $\theta$.

**Advantages:**

- Estimate of variance around the mean (measure of uncertainty of parameters)
- Less prone to overfitting (point estimate is overconfident)
- Point estimate may be at untypical point

**What is missing?**

We need to quantify our preferences given calculated probabilities, i.e. make actions.

**=> Bayesian decision theory**

# B.1 Decision theory: Bayesian approach

**Goal is to devise a decision procedure (policy):** $\delta : \boldsymbol{D} \to A$

$\rho(a|\mathbf{x}) \triangleq \mathrm{E}_{p(y|\mathbf{x},D)}[L(y, a\ )]$ Posterior expected loss

$\delta(\mathbf{x}) = \mathrm{argmin}_{a \in A}\rho(a|\mathbf{x})$ Bayes decision rule

Specifies optimal action for each possible input (rational behavior).

**Bayes estimators for common loss functions:**

**0-1 loss**

$$L(y, a) = \mathrm{I}(y \neq a) = \begin{cases} 0 \;\; if \;\; y = a \\ 1 \;\; if \;\; y \neq a \end{cases}$$

$\rho(a|\mathbf{x}) = 1 - p(y|\mathbf{x}, D)$

$y_*(\mathbf{x}) = \mathrm{argmax}_{y \in Y}p(y|\mathbf{x}, D)$

**Reject option (for risk averse domains)**

$$L(y = j, a = i) = \begin{cases} 0 \;\; if \; i = j \; and \; i, j \in \{1, .. C\} \\ \alpha_r \; if \; i = C + 1 \\ \alpha_s \; otherwise \end{cases}$$

# B.1 Decision theory: Frequentist approach

**Idea:** Avoid treating parameters of the model like random variables.

**Instead:** Consider sampling distribution (distribution that an estimator has when applied to multiple data sets sampled from the true but unknown distribution*).

$$R(\theta^*, \delta) \triangleq \mathrm{E}_{p(D'|\theta^*)}\left[L\big(\theta^*, \delta(D')\big)\right] = \int L\big(\theta^*, \delta(D')\big) p(D'|\theta^*) dD'$$

**!! Loss function depending on parameters**

Parameters – fixed, data – random.

**Problem: cannot be computed, $\theta^*$ is unknown.**

=> Let's consider general properties of estimators at first.

* Definition from [3]

# 10 minute break..

# C.1 Intro to ML: Bias-variance tradeoff

## Estimators

Point estimator or statistic: $\hat{\theta}_m = g(x_1, .. x_n)$

Example: sample mean $\hat{\mu} = \frac{1}{N} \sum_n x_n$

## Desirable properties of estimators

1. Consistent estimators

Estimator is consistent if it eventually recovers the true parameters that generated the data as the sample size goes to infinity: $\hat{\theta}(D) \rightarrow \theta^*$ as $|D| \rightarrow \infty$

2. Unbiased estimators

$$\text{bias}\left(\hat{\theta}(D)\right) = \mathrm{E}_{p(D|\theta^*)}\left[\hat{\theta}(D) - \theta^*\right]$$

3. Minimum variance estimators

# C.1 Intro to ML: Bias-variance tradeoff

## Bias-variance tradeoff

$$\bar{\theta} = \mathrm{E}_{p(D|\theta^*)}\left[\hat{\theta}\right]$$

$$\mathrm{E}_{p(D|\theta^*)}\left[\left(\hat{\theta} - \theta^*\right)^2\right] = \mathrm{E}\left[\left[\left(\hat{\theta} - \bar{\theta}\right) + (\bar{\theta} - \theta^*)\right]^2\right] = \mathrm{var}\left[\hat{\theta}\right] + \mathrm{bias}^2\left(\hat{\theta}\right)$$

Sometimes it is preferable to select biased estimator if it reduces variance.

If focus only on unbiased estimators, it introduces additional constraint in optimization task for MSE.

# C.2 Intro to ML: ERM

## Loss function

**Key idea**: Change loss function to depend on predictions rather than parameters.

Loss function $L(y, \hat{y})$, where $\hat{y} = f(\mathbf{x}, \theta)$- prediction label (value) for $\mathbf{x}$

So, ideal loss function for optimization:

$$f_* = \mathrm{argmin}_f J(f) = \mathrm{argmin}_f \int L(y, f(\mathbf{x}))p(y, \mathbf{x})dyd\mathbf{x}$$

Don't know $p(y, \mathbf{x})$, so use empirical distribution instead:

$$f_N = \mathrm{argmin}_f J_N(f) = \mathrm{argmin}_f \frac{1}{N}\sum_n L(y_n, f(\mathbf{x}_n))$$

# C.2 Intro to ML: ERM

## Empirical risk minimization approach

$$\theta^{opt} = \mathbf{argmin}_\theta \frac{1}{N}\sum_{n=1}^{N} L\big(y_n, f(\mathbf{x}_n, \theta)\big) + \lambda P(\theta)$$

$$f_* = \text{argmin}_f J(f)$$
$$f_F = \text{argmin}_{f \in F} J(f)$$
$$f_N = \text{argmin}_{f \in F} J_N(f)$$

Deviation in generalization error when using certain family of functions

Deviation due to optimizing empirical loss instead of the expected loss

$$\text{E}[J(f_N) - J(f_*)] = \text{E}[J(f_F) - J(f_*)] + \text{E}[J(f_N) - J(f_F)]$$

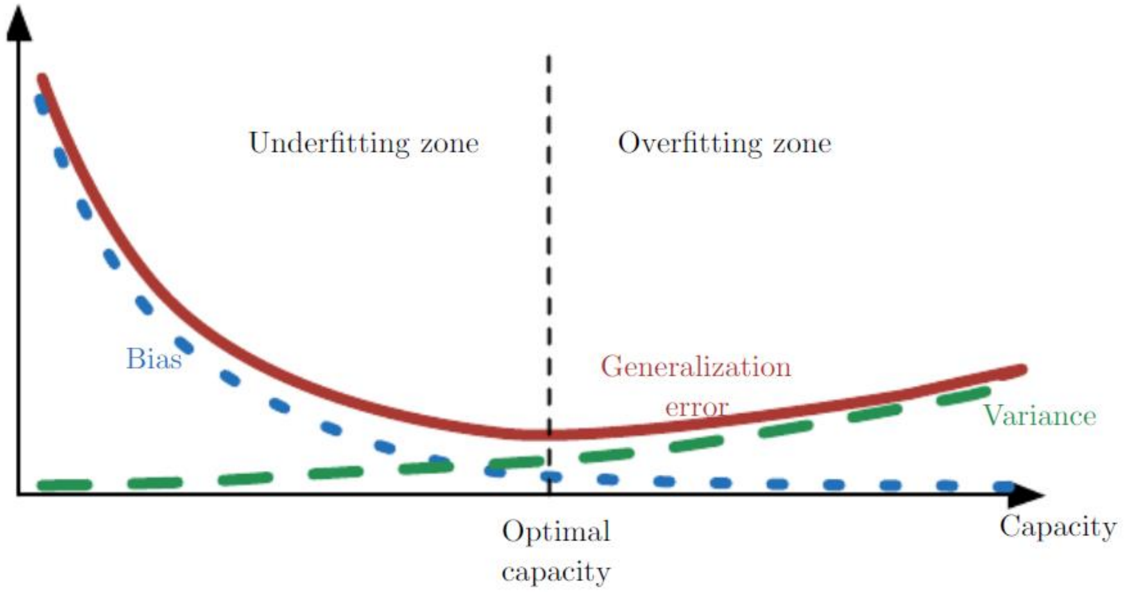
appr. error      est. error

**=> Tradeoff between accuracy and complexity:** non-flexible functions have big appr. error and complex functions are poorly fitted with fixed number of training points.

# C.3 Intro to ML: Generalization ability

## Generalization performance

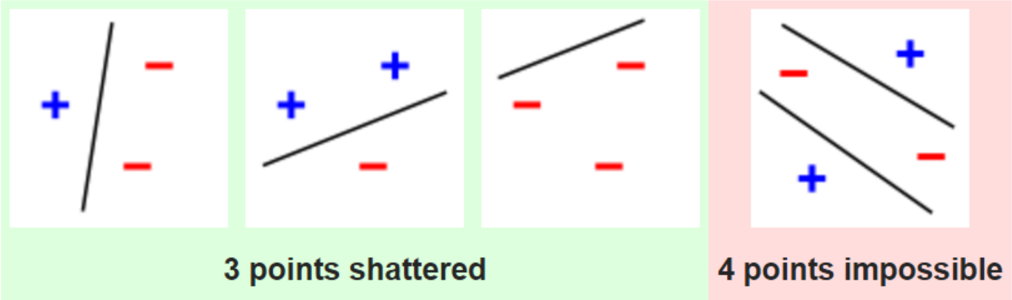
Average performance computed over different data sets which did not participate in training.




* Picture from [2]

# C.3 Intro to ML: Generalization ability

**VC dimension of model (capacity measure):** the largest possible value of ***m*** for which there exists a training set of ***m*** different *x* points that the classifier can label arbitrarily.




Statistical learning theory studies approximations to the generalization error.

$$J(f) = \int L(y, f(\mathbf{x}))p(y, \mathbf{x})dyd\mathbf{x} \qquad J_N(f) = \frac{1}{N}\sum_n L(y_n, f(\mathbf{x}_n))$$

$$J_N(f) \leq J(f) + O\left(\sqrt{\frac{VC(f)}{N}\ln\frac{N}{VC(f)} + \frac{1}{N}\ln\frac{1}{\delta}}\right) \qquad \text{with probability } 1 - \delta$$

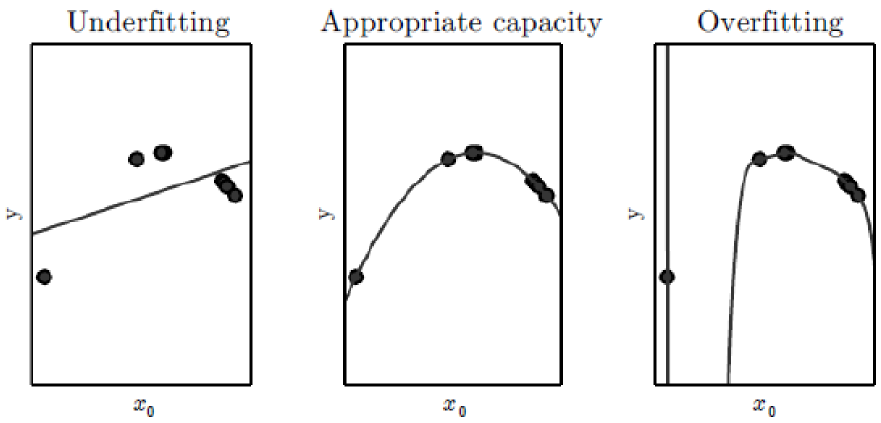
See [3] and references therein for details.

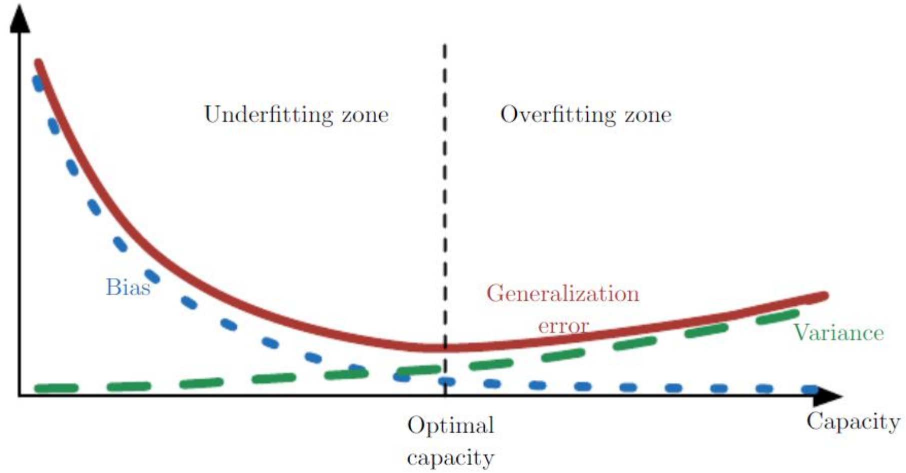
# D. Model selection

## What did we omit previously?

Model selection: $p(\theta|\boldsymbol{D}) = \frac{p(\boldsymbol{D}|\theta)p(\theta)}{p(\boldsymbol{D})} \rightarrow p(\theta|\boldsymbol{D}, m) = \frac{p(\boldsymbol{D}|\theta,m)p(\theta|m)}{p(\boldsymbol{D}|m)}$

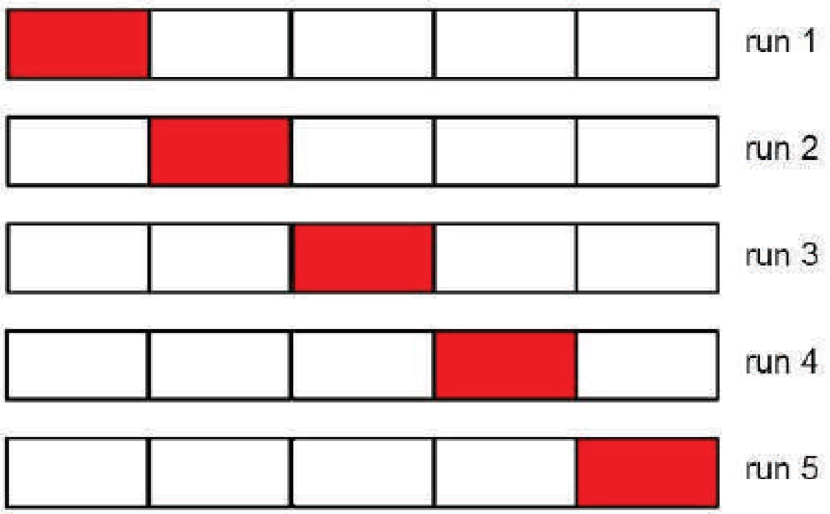
**Approaches:** Cross-validation, Bayesian model selection




* Picture from [2]

# D.1 Model selection: Cross-validation




## **Cross-validation**

Three datasets:

- training – fit model
- validation – select model
- test – test model

**Example of model selection:**

choose “right” degree of polynomial for approximation

If lack of training data – split into *K* folds. If set *K=N*, then leave-one out cross validation (LOOCV).

* Picture from [3]

# D.2 Model selection: Bayesian approach

**Compute posterior over models:**

$$p(m|D) = \frac{p(D|m)p(m)}{\sum_{m \epsilon M} p(D|m)p(m)}$$

Let's use uniform prior $p(m)$ over the models =>

Marginal likelihood
Integrated likelihood
Evidence

$$m_{opt} = \mathrm{argmax}_m p(D|m) = \int P(D|\theta)p(\theta|m)d\theta$$

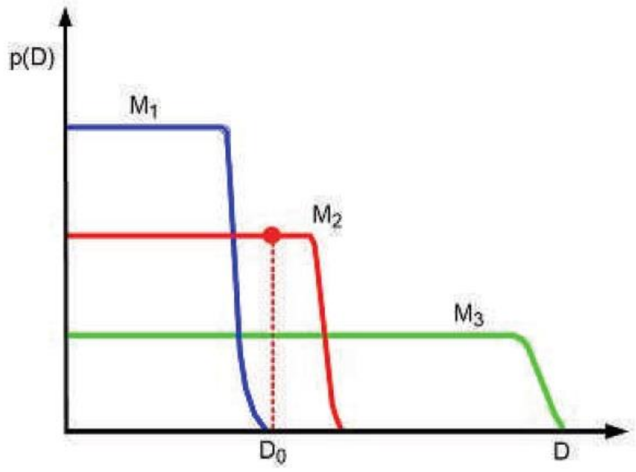
Models with more parameters do not necessarily have higher marginal likelihood (**Bayesian Occam's razor effect**).

# D.2 Model selection: Bayesian approach

## Interpretation #1

$$p(D|m) = p(y_1|m)p(y_2|y_1, m)p(y_3|y_{1:2}, m) \dots p(y_N|y_{1:N-1}, m)$$

=> Overfitting to early examples.




## Interpretation #2

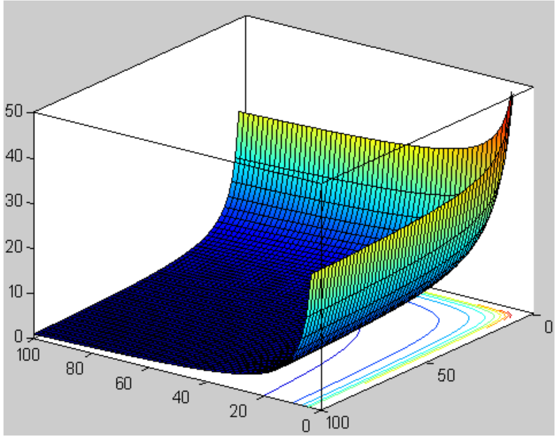
$\sum_{D'} p(D'|m) = 1$ (sum over all datasets)

=> For complex models probability must spread thinly.

***Exercise:** check out examples of calculation the "evidence" in [3], ch.5.3, read about Bayesian information criterion (BIC).*

* Pictures from [3]

# D.2 Model selection: Bayesian approach




## Example of model selection

Testing if coin is fair: $N$ coin tosses, $N_1$ heads and $N_0$ tails

Model $M_0$ : fair coin, $\theta = 0.5$

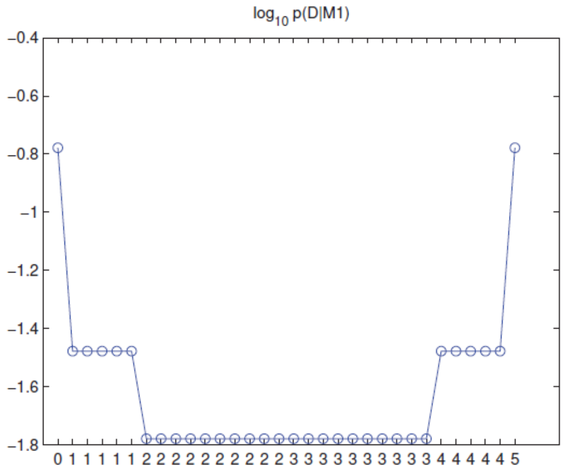
Model $M_1$: biased coin $\theta \in [0,1]$,

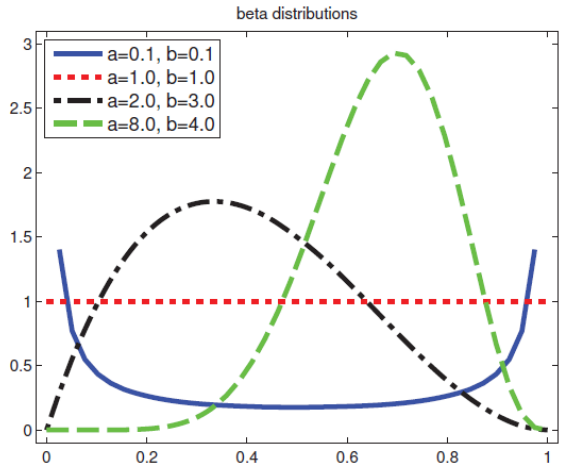
Beta prior on $\theta$: $p(\theta) = \text{Beta}(\theta|\alpha_1, \alpha_0)$

$$p(D|M_0) = 0.5^N$$

$$p(D|M_1) = \frac{B(\alpha_1 + N_1, \alpha_0 + N_0)}{B(\alpha_1, \alpha_0)}$$

For $N = 5$ and $\alpha_1 = \alpha_0 = 1$




* Pictures from [3]

# Next lecture

1. Representer theorem, kernel trick

2. Supervised learning examples: linear regression, SVM etc
3. Unsupervised learning examples

4. Why do we need deep learning?

# D. Homework

**For all:**

1. ML: [3] ch. 1, 2, 3, 5, 6 or [4] ch. 8, 12, 13 or [5] ch. 2, 3, 7.
2. Exercises from presentation.
3. Python (Jupyter, Numpy).

**Optional for enthusiasts:**

1. Start ML course.

# Refs


1. Thorough review of relevant math topics:

http://info.usherbrooke.ca/hlarochelle/ift725/review.pdf

2. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning.

3. Kevin P. Murphy, Machine Learning: A probabilistic perspective.

4. David Barber, Bayesian Reasoning and Machine Learning.

5. Sergios Theodoridis, Machine Learning: A Bayesian and optimization perspective.