Machine Learning

2024 (ML-2024)
Lecture 5. Bayesian approach

by Alexei Valerievich Kornaev, Dr. habil. in Eng. Sc.,
Researcher at the RC for AI, Assoc. Prof. of the Robotics and CV Master's Program, Innopolis University
Researcher at the RC for AI, National RC for Oncology n.a. NN Blohin
Professor at the Dept. of Mechatronics, Mechanics, and Robotics, Orel State University

Books

Handbook on Machine Learning by M. Artemyev et al., Yandex, 2022 (in Russian)
Understanding Deep Learning by Simon J.D. Prince, 2024
Practical Deep Learning / FastAI book by Jeremy Howard
Deep Learning by Ian Goodfellow and Yoshua Bengio and Aaron Courville, 2016.

Online platforms, courses, resources

Sirius online courses on ML (in Russian)
Stepik online courses (in Russian)
Hugging Face online courses
Coursera is unavailable so far

MIT Introduction to Deep Learning, MIT, 2024
Lecture Hall of the Faculty of Applied Mathematics and Informatics (in Russian)
Fast AI, courses, software, book by Jeremy Howard
Deep Learning, course by Semyon Kozlov (in Russian), 2019

3Blue1Brown, Animated Math
PyTorch Tutorial by Patrick Loeber, 2020

#someLinks

Read here: https://arxiv.org/, https://scholar.google.ru/
Collect the references here: https://mendeley.com/
Draw here: https://miro.com/app/dashboard/
Write the text here: https://www.overleaf.com/project
Write the code here: https://colab.research.google.com/
Collect the code here: https://github.com/
Find the journal here: https://journalfinder.elsevier.com/
Find the conference here: https://portal.core.edu.au/conf-ranks/?search=A

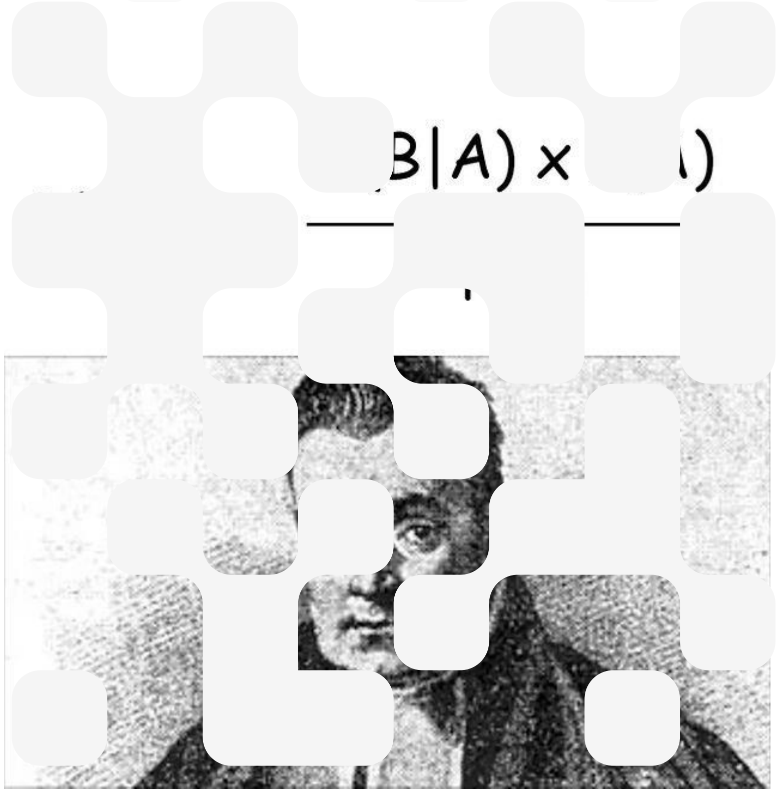
Agenda

I. Rules of probability and the Bayes' theorem

II. From discrete to continuous

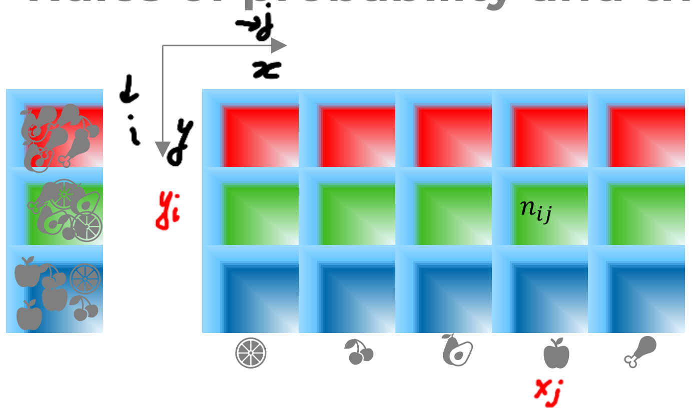
III. Frequentists vs Bayesians

IV. Bayesian framework



All models are wrong, but some are useful.
/George Box/

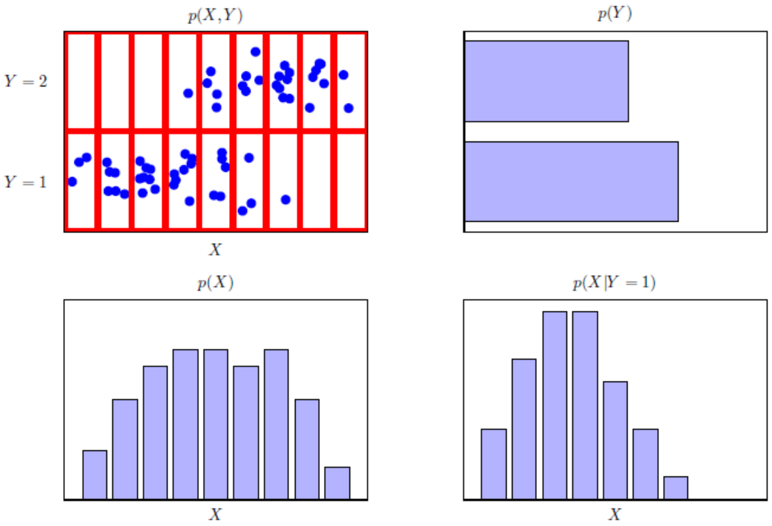
Rules of probability and the Bayes' theorem intuition



1. The *probability* of an event is the fraction of times that event occurs out of the total number of trails: $P(\ ) = \frac{\#}{\#}$, or $P(x_j) = \frac{\sum_i n_{ij}}{N} = \frac{c_j}{N}$, the limit that the total # of trails N goes to ∞ .
2. The probability that $x = x_j$ and $y = y_i$ is written $P(y_i, x_j) = \frac{n_{ij}}{N}$ and is called *joint probability.*
3. It can be seen from (2): $P(x_j) = \sum_i P(y_i, x_j)$ which is the *sum rule* of probability.
4. Consider $x = x_j$, then the fraction of such instances for which $y = y_i$ is written $P(y_i|x_j) = \frac{n_{ij}}{c_j}$ and is called *conditional probability*.
5. It can be seen from (4): $P(y_i, x_j) = \frac{n_{ij}}{N} = \frac{n_{ij}}{c_j}\frac{c_j}{N} = P(y_i|x_j)P(x_j)$ which is the *product rule*.
6. Taking the symmetry property $P(y_i, x_j) = P(x_j, y_i)$ into account and the rules (3),(5) the Bayes' theorem can be met: $P(y_i|x_j) = \frac{P(x_j|y_i)P(y_i)}{P(x_j)} = \frac{P(x_j|y_i)P(y_i)}{\sum_i P(y_i, x_j)} = \frac{P(x_j|y_i)P(y_i)}{\sum_i P(x_j|y_i)P(y_i)}$.

Pattern Recognition and Machine Learning by Ch. Bishop, 2006

Bayes' theorem intuition



The Bayes' theorem:

$$P(y_i|x_j) = \frac{P(x_j|y_i)P(y_i)}{\sum_i P(x_j|y_i)P(y_i)}.$$

Pattern Recognition and Machine Learning by Ch. Bishop, 2006

Bayes inference: a simple example

Steve is very **shy and withdrawn**, invariably helpful but with very little interest in people or in the world of reality. A **meek and tidy soul**, he has a need for order and structure, and a passion for detail.

Which of the following do you find more likely:
- Steve is a librarian
- Steve is a farmer

Bayes' theorem from 3Blue 1Brown: take the prior information into account

The Bayes' theorem:

$$P(y_i|x_j) = \frac{P(x_j|y_i)P(y_i)}{\sum_i P(x_j|y_i)P(y_i)}$$

$$P(Lib|Shy) = \frac{P(Shy|Lib)P(Lib)}{P(Shy|Lib)P(Lib) + P(Shy|Farm)P(Farm)} = \frac{4}{24} = \frac{1}{6} \approx 17\%$$

$$P(Shy|Lib) = \frac{4}{10}$$

$$P(Shy|Farm) = \frac{20}{200}$$

$$P(Lib) = \frac{10}{210}$$

$$P(Farm) = \frac{200}{210}$$

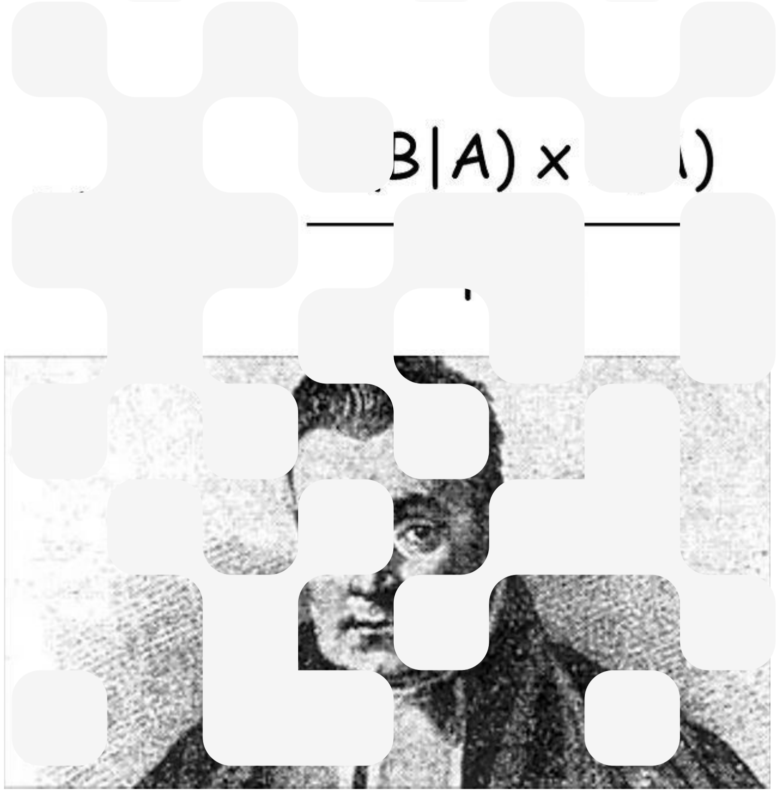
Agenda

I. Rules of probability and the Bayes' theorem

II. From discrete to continuous

III. Frequentists vs Bayesians

IV. Bayesian framework



All models are wrong, but some are useful.
/George Box/

From discrete to continuous: the probability density

The *probability density* must satisfy two conditions:

$$p(x) \geq 0,$$
$$\int_{-\infty}^{+\infty} p(x)dx = 1.$$

The *sum* and *product* rules take the following form, respectively:

$$p(x) = \int p(x, y)\, dy,$$

$$p(x, y) = p(y|x)p(y).$$

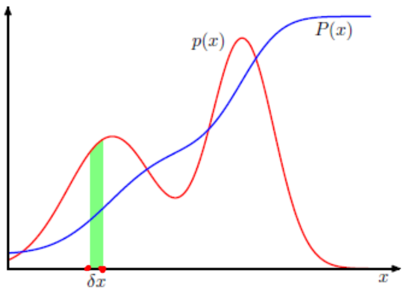
The *expectation* is the average value of some function $f(x)$ under the probability distribution:

$$E[f(x)] = \int p(x)f(x)\, d\cancel{y}.$$

The *variance* of the function is:

$$var[f(x)] = E[(f(x) - E[f(x)])^2].$$

Figure 1.12 The concept of probability for discrete variables can be extended to that of a probability density $p(x)$ over a continuous variable x and is such that the probability of x lying in the interval $(x, x + \delta x)$ is given by $p(x)\delta x$ for $\delta x \to 0$. The probability density can be expressed as the derivative of a cumulative distribution function $P(x)$.



[Pattern Recognition and Machine Learning by Ch. Bishop, 2006]

From discrete to continuous: the Gaussian distribution (probability density)

The *normal* or *Gaussian* distribution:

$$p(x) = N(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right),$$

$$\int_{-\infty}^{+\infty} p(x)dx = \int_{-\infty}^{+\infty} N(x|\mu,\sigma^2)dx = 1.$$

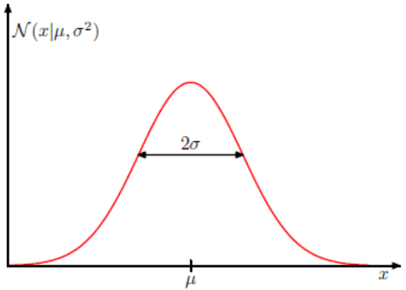
The *expectation* of $f(x) = x$ under the probability distribution:

$$E[x] = \int N(x|\mu,\sigma^2)xdx = \mu.$$

The *variance* of the function $f(x) = x$ is:

$$var[x] = E[x^2] - E[x]^2 = \sigma^2.$$

Figure 1.13 Plot of the univariate Gaussian showing the mean μ and the standard deviation σ.



[Pattern Recognition and Machine Learning by Ch. Bishop, 2006]

From discrete to continuous: the Bayes' theorem

The *sum* rule $p(x) = \int p(x,y)\,dy$, and its generalization:

$p(x_1,\ldots,x_m) = \int p(x_1,\ldots,x_m,\underbrace{x_{m+1},\ldots,x_k}) \underbrace{dx_{m+1}\ldots dx_k}.$

The *product* rule $p(x,y) = p(x|y)p(y)$, and its generalization:

$p(x_1,\ldots,x_m) = p(x_m|x_1,\ldots,x_{m-1})p(x_{m-1}|x_1,\ldots,x_{m-2})\ldots p(x_2|x_1)p(x_1).$

The *Bayes'* theorem:

$$p(y|x) = \frac{p(x|y)p(y)}{\int p(x,y)\,dy},$$

$$Posterior = \frac{Likelihood \cdot Prior}{Evidence}.$$

$P(x_j) = \sum_i P(y_i,x_j)$ is the *sum rule* of probability.

$P(y_i,x_j) = \frac{n_{ij}}{N} = \frac{n_{ij}}{c_j}\frac{c_j}{N} = P(y_i|x_j)P(x_j)$ is the *product rule.*

The *Bayes'* theorem: $P(y_i|x_j) = \frac{P(x_j|y_i)P(y_i)}{\sum_i P(x_j|y_i)P(y_i)}$.

product rule → sum rule

[Pattern Recognition and Machine Learning by Ch. Bishop, 2006]

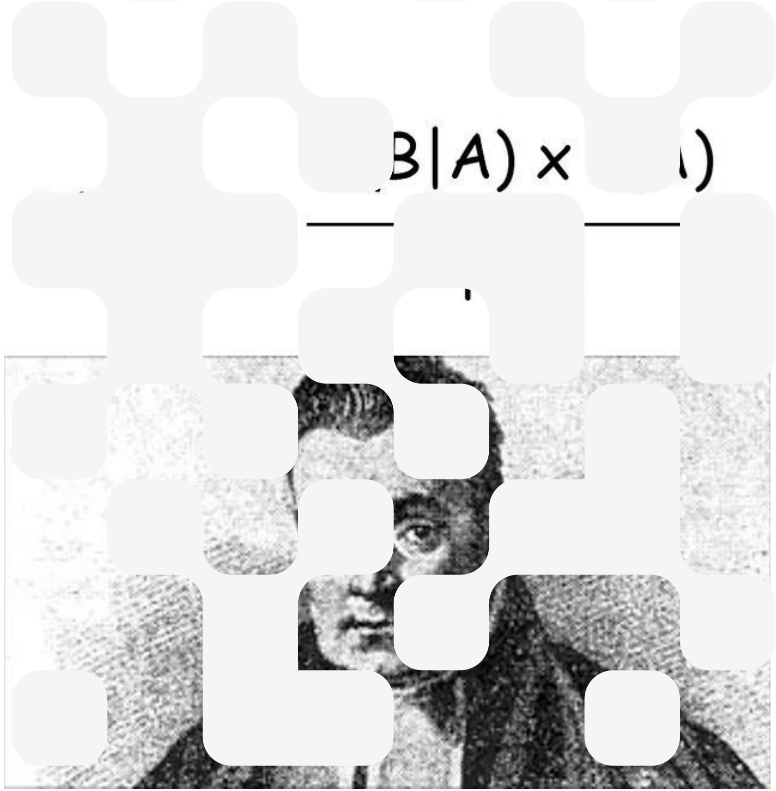
Agenda

I. Rules of probability and the Bayes' theorem

II. From discrete to continuous

III. Frequentists vs Bayesians

IV. Bayesian framework



All models are wrong, but some are useful.
/George Box/

Frequentist vs Bayesian frameworks

	95%	5%
	Frequentist	Bayesian
Randomness	Objective indefiniteness	Subjective ignorance
Inference	Random and Deterministic	Everything is random
Estimates	Maximal likelihood	Bayes theorem
Applicability	$n \gg size(\boldsymbol{\theta})$	$\forall n$

Advantages of a Bayesian framework are: regularization; latent variable modeling; extendibility; scalability.

[Нейробайесовский подход к задачам машинного обучения — Дмитрий Ветров] [Deep Bayes School]

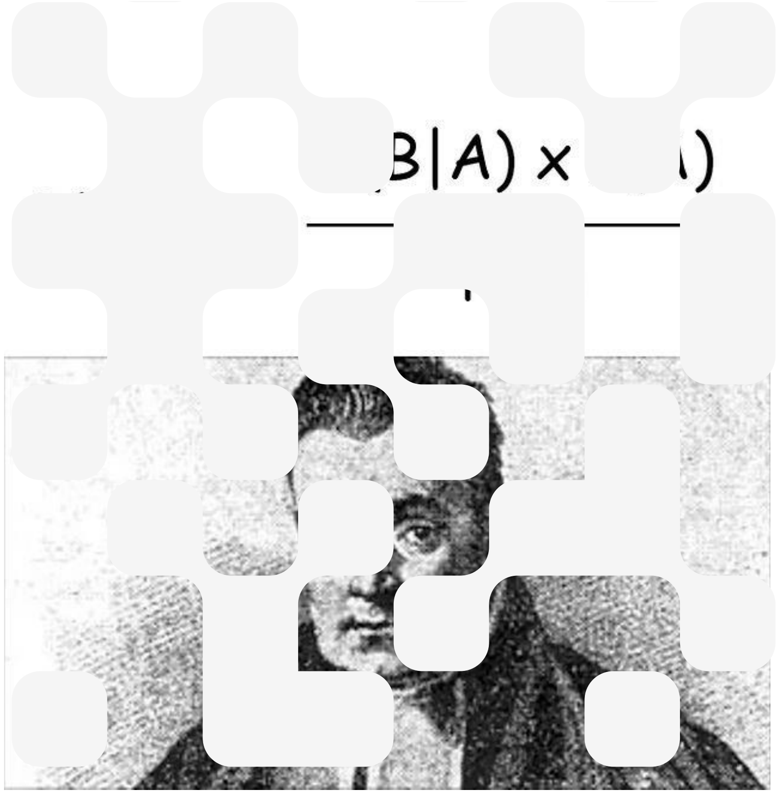
Agenda

I. Rules of probability and the Bayes' theorem

II. From discrete to continuous

III. Frequentists vs Bayesians

IV. Bayesian framework



All models are wrong, but some are useful.
/George Box/

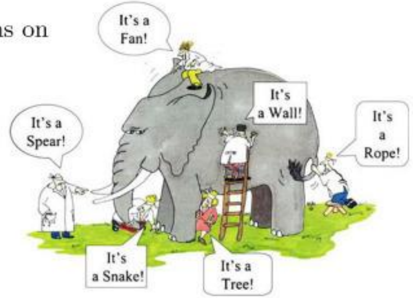
A Bayesian framework and inference example

Bayesian framework

- Encodes ignorance in terms of distributions
- Makes use of **Bayes Theorem**

$$\texttt{Posterior} = \frac{\texttt{Likelihood} \times \texttt{Prior}}{\texttt{Evidence}}, \quad p(\theta|X) = \frac{p(X|\theta)p(\theta)}{\int p(X|\theta)p(\theta)d\theta}$$

- Posteriors may serve as new priors, i.e. may combine multiple models!
- **BigData:** we can process data streams on an update-and-forget basis
- Support distributed processing



Bayesian inference

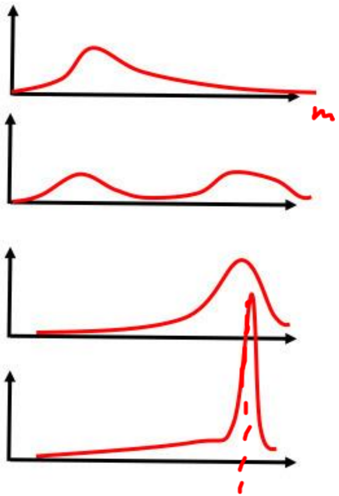
- Consider blind wisdomers who try to estimate the mass of an elephant using their tactile measurements.
- They start with common knowledge about animals typical masses $p(\theta)$
- The first wisdomer touches a tail

$$p(\theta|x_1) = \frac{p_1(x_1|\theta)p(\theta)}{\int p_1(x_1|\theta)p(\theta)d\theta}$$

- The second wisdomer touches a leg and uses $p(\theta|x_1)$ as **his new prior**

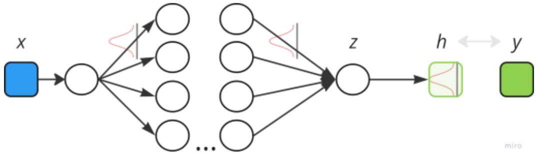
$$p(\theta|x_1, x_2) = \frac{p_2(x_2|\theta)p(\theta|x_1)}{\int p_2(x_2|\theta)p(\theta|x_1)d\theta}$$

- ...
- At the end they form sharp distribution $p(\theta|x_1, \ldots, x_m)$



[Нейробайесовский подход к задачам машинного обучения. Лекция Дмитрия Ветрова в Яндексе, 2017]

Bayesian framework intuition



Consider a probabilistic model $\boldsymbol{f} = \left[x^{(i)}, \boldsymbol{\phi}\right]$ parameterized with *weights distribution* $\boldsymbol{\phi}$ (e.g. normal prior distributioin) that maps each i-th input sample $x^{(i)}$ into the output distribution $z^{(i)}$ which then transforms into the hypothesis distribution $h^{(i)}$ (mean and standard deviation) with the mean that should be close to the label $y^{(i)}$.

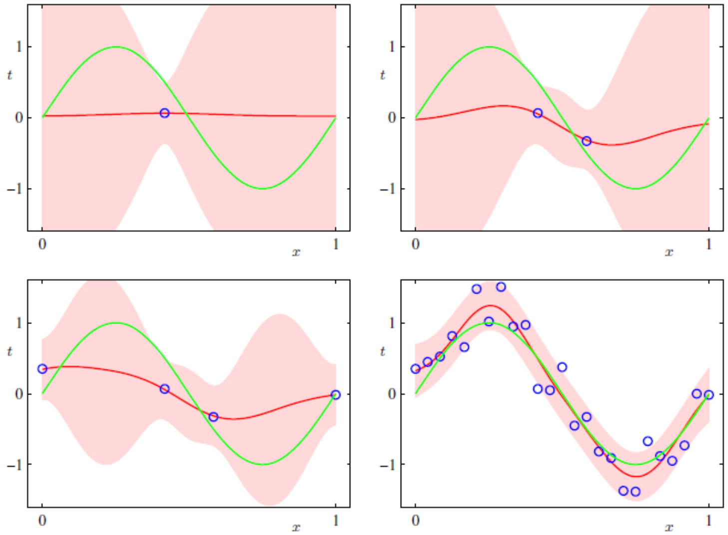


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis functions of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

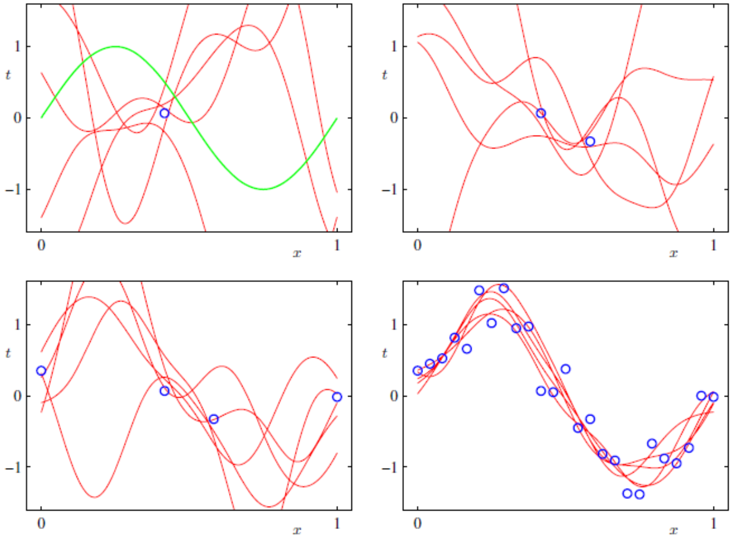


Figure 3.9 Plots of the function $y(x, \mathbf{w})$ using samples from the posterior distributions over $\mathbf{w}$ corresponding to the plots in Figure 3.8.

[Pattern Recognition and Machine Learning by Ch. Bishop, 2006]

With the little help from my AI assistant (DeepSeek V2.5 Coder)

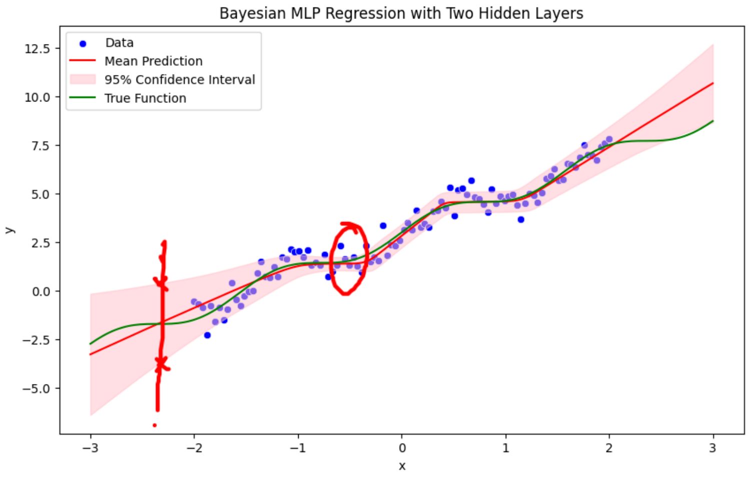
Dear model, would you please give me a code that demonstrates Bayesian approach to approximation? Use pytorch and seaborn, please

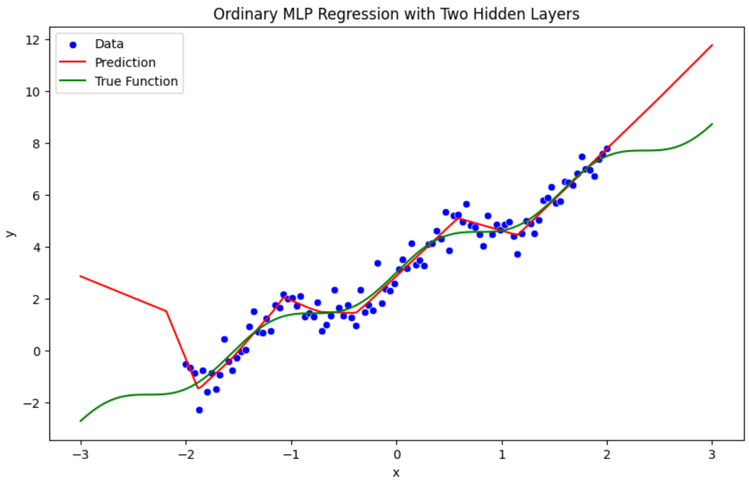
use MLP in your code, please.

make it with 2 hidden layers, please.

could you give me one more solution that operates with the same data, but using ordinary MLP of the same architecture?

*Then we fixed an error in code, and made more epochs (10K).





See the obtained code on GitHub

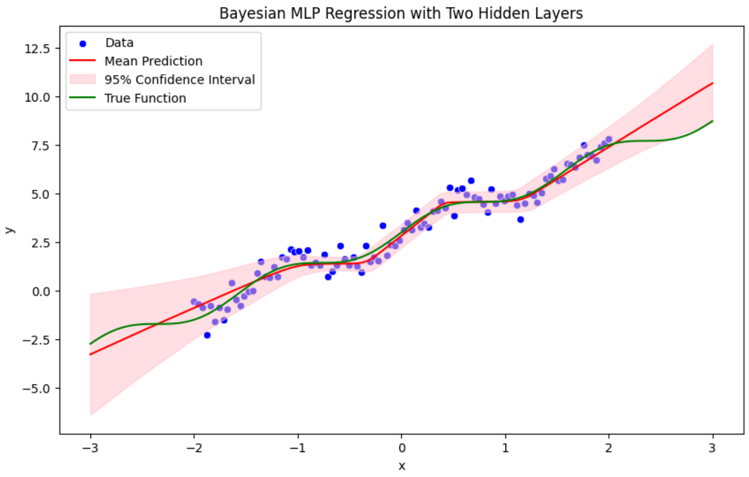
With the little help from my AI assistant (DeepSeek V2.5 Coder)

Dear model, would you please give me a code that demonstrates Bayesian approach to approximation? Use pytorch and seaborn, please

use MLP in your code, please.

make it with 2 hidden layers, please.

*Then I just decreased x limits -1<x<1, and made more epochs (10K).



See the obtained code on GitHub

Explanation:

1.Model Definition:

1. We define a Bayesian MLP model with two hidden layers. The weights and biases of each layer are treated as random variables with a prior distribution (Normal distribution in this case).
2. The variational parameters (mean and log-standard deviation) are learned during training.

2.Training:

1. We use the Evidence Lower Bound (ELBO) as the objective function, which is the sum of the negative log-likelihood and the KL divergence between the variational distribution and the prior.
2. The model is trained using gradient-based optimization (Adam optimizer).

3.Visualization:

1. After training, we sample multiple predictions from the model to estimate the mean and standard deviation of the predictions.
2. We plot the mean prediction along with the 95% confidence interval (2 standard deviations) and compare it with the true function and the data points.

Output:

The code will output a plot showing the data points, the true function, the mean prediction of the Bayesian MLP model with two hidden layers, and the 95% confidence interval around the mean prediction.

Thank you for your attention!

a.kornaev@innopolis.ru, @avkornaev