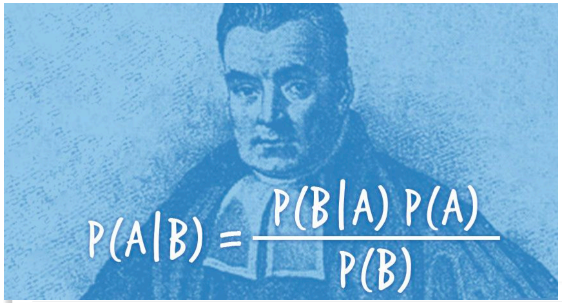
CS6190 Probabilistic Modeling

Spring 2024

Instructor: Shandian Zhe
zhe@cs.utah.edu
School of Computing

THE UNIVERSITY OF UTAH



Shandian Zhe: Probabilistic Machine Learning

Assistant Professor

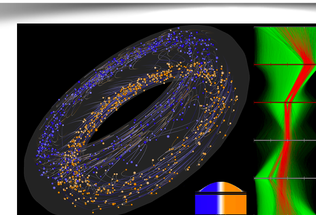
Research Topics:

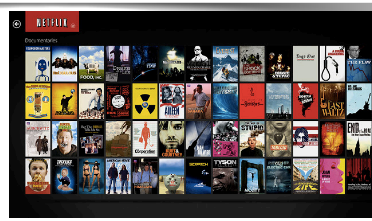
1. **Bayesian Deep Learning**
2. **Bayesian Nonparametrics**
3. **Physics-Informed Machine Learning**
4. **Physics-Informed Neural Networks**
5. **Probabilistic Graphical Models**
6. **Large-Scale Machine Learning**
7. **ML in simulation**
8. **…**

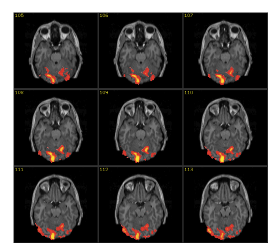
Applications:

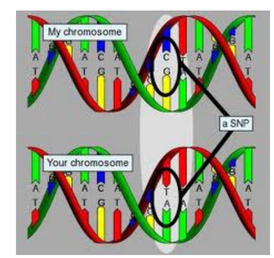
- **Physical Simulation**
- **Governing Eq. Discovery**
- **Collaborative Filtering**
- **Brain Imaging Data Analysis**

….









Outline

- What is probabilistic machine learning
- Why probabilistic/Bayesian machine learning
- Course requirements/policies (homework assignments, projects, etc.)
- Basic knowledge review

What is machine learning

"A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."

Tom Mitchell (1999)

Machine learning is the driving force of AI

Alpha-Go! A ML algorithm rather AI

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.

A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.

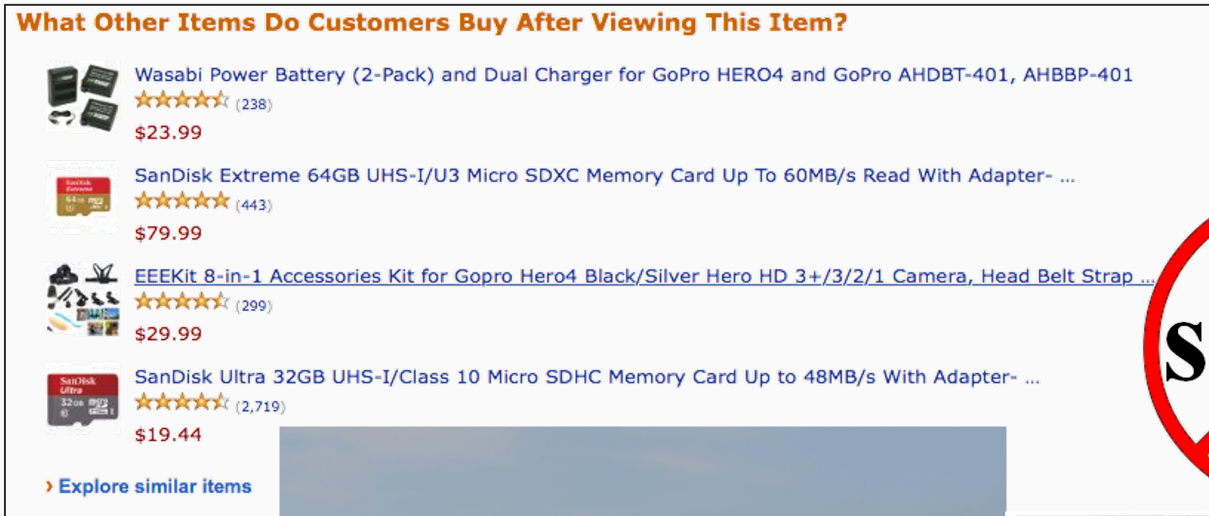
Teddy bears swimming at the Olympics 400m Butterfly event.

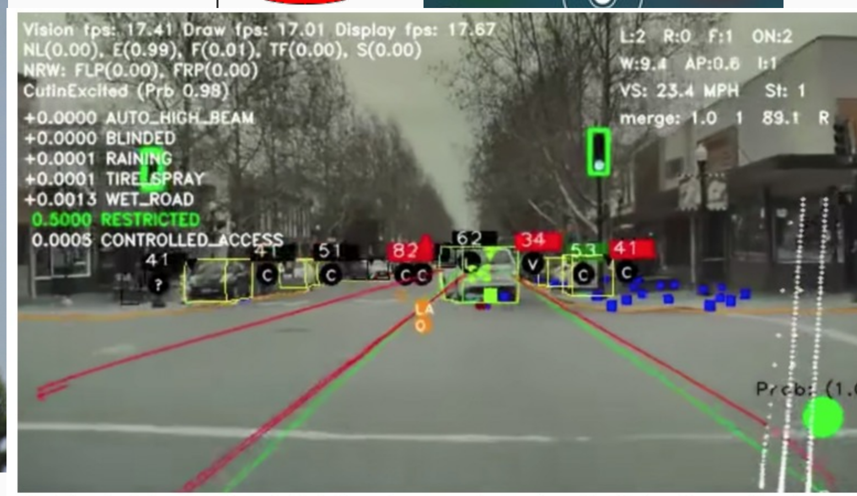
A cute corgi lives in a house made out of sushi.

A cute sloth holding a small treasure chest. A bright golden glow is coming from the chest.

Machine learning is everywhere!

And you are probably already using it





Machine learning is everywhere!

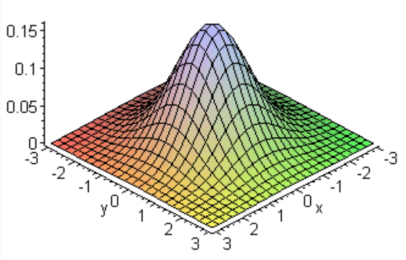
And you are probably already using it

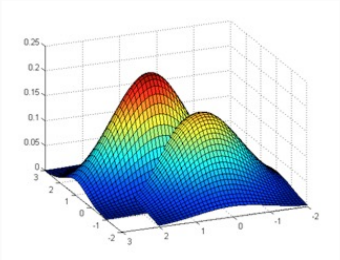
- Is an email spam?
- Find all the people in this photo
- If I like these three movies, what should I watch next?
- Based on your purchase history, you might be interested in…
- Will a stock price go up or down tomorrow? By how much?
- Handwriting recognition
- What are the best ads to place on this website?
- I would like to read that Dutch website in English
- Ok Google, Drive this car for me. And, fly this helicopter for me.
- Does this genetic marker correspond to Alzheimer’s disease?

What is probabilistic learning?

In a nutshell, probabilistic learning is branch of ML that uses **probabilistic (or Bayesian) principles** for model design and algorithm development.

Probabilistic Learning





Prior distribution

$$p(\boldsymbol{\theta})$$

Data likelihood

$$p(\mathbf{D}|\boldsymbol{\theta})$$

Posterior distribution

$$p(\boldsymbol{\theta}|\mathbf{D})$$

Bayes's Rule

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

Advantage

- Unified, principled mathematical framework

Priors + **Data** = **Posteriors**

$\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$ $\quad$ $\mathbf{D}|\boldsymbol{\theta} \sim p(\mathbf{D}|\boldsymbol{\theta})$ $\quad$ $p(\boldsymbol{\theta}|\mathbf{D})$

- Uncertainty reasoning

Asthma: 60%

Heart disease: 30%

Healthy: 10%

Raining: 70%

Sunny: 30%

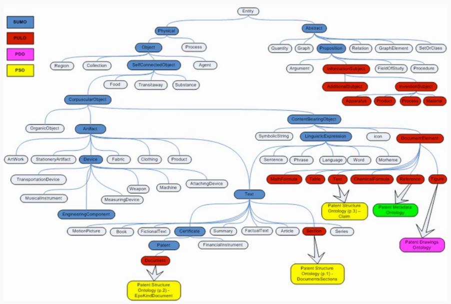
How important is the uncertainty?

Tesla death smash probe: Neither driver nor autopilot saw the truck

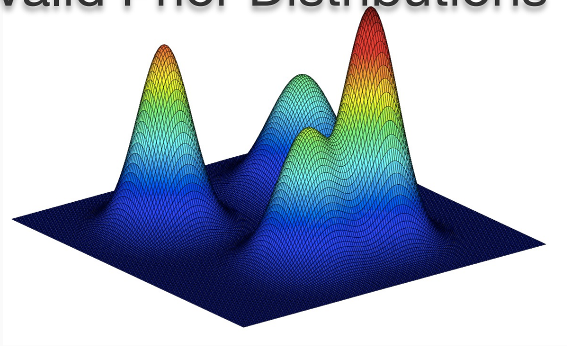
Challenges

- Modeling

Complex Knowledge/assumptions



Valid Prior Distributions



- Calculation

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

High dimensional integration

MCMC sampling

Variational approximations

Belief propagation

In this course

- We will cover both the classical and state-of-the-art approaches to deal with these challenges.

Overview of this course

Syllabus

Warning

1. This course is ***math intensive*** and requires ***a certain level of*** programming (with Matlab, R or Python). Python components may require TensorFlow and/or PyTorch. The coding workload is not heavy, but requests ***mathematical derivations and careful debugging***.

2. The workload is heavy (5-10 hours per-week)

How will you learn?

- Take classes to follow the math, understand the models and algorithms
- **Derive the math details by yourself!**
- Finish the homework assignments to deepen your understanding
- **Implement and debug the models and algorithms by yourself!**
- Course project to enlarge your vision and practice your capability

This course

Focuses on the mathematic foundations, modeling and algorithmic ideas in probabilistic learning

This course is not about

- Applying ML to specific tasks (e.g., image tagging and autonomous driving)
- Using specific ML tools/libraries, e.g., scikit-learn and PyTorch
- How to program and debug, e.g., with Python, R or Matlab

This course

is an advanced course for students who want to study ML in depth or quickly get to the frontier research of probabilistic learning

This course is not a preliminary course, e.g., entry-level introduction of statistics. That means,

The content can be hard for some ones

Don’t take this course if

- You are struggling with linear algebra, calculus or basic statistical concepts
- You are sick of mathematical symbols, derivations, proofs and calculations
- You do NOT feel good in programming and debugging

We assume that

- You are not scared of math, statistics, calculus and calculations; you are happy with them!
- You are comfortable with abstract symbols and matrices operations
- You can pick-up Matlab/Python/R quickly (even if you have never used them before)
- You enjoy debugging, step in, step out, print, etc.
- You can quickly learn how to use TensorFlow or PyTorch or Jax by following the documentation and searching for the online examples

and Most Important

- You have planed for enough efforts for this class (e.g., 5-10 hours per-week)

If you feel NOT right about any of these assumptions

- Seriously consider whether to take this course

We want you to succeed

- We do not want to make you feel tortured

Course information

- The course website contains all the detailed information
- The course website is linked to my homepage

My home page http://www.cs.utah.edu/~zhe/

Course website https://www.cs.utah.edu/~zhe/teach/cs6190.html

Basics Review

Note: this review is neither compressive nor in depth. Due to time limit, this review is just to point out key concepts and computational rules as the guidance. We list the references for you to check out details for future usage.

Matrix/Vector Derivative

- Standard notations
  - non-bold letters: scalars

$$a, b, x, y, B, D, G, \alpha, \gamma, \ldots$$

  - Bold small letters: vectors

$$\mathbf{a}, \mathbf{b}, \mathbf{x}, \mathbf{y}, \boldsymbol{\gamma}, \boldsymbol{\eta}, \ldots$$

  - Bold capital: matrices

$$\mathbf{A}, \mathbf{X}, \mathbf{Z}, \mathbf{\Gamma}, \ldots$$

Matrix/Vector Derivative

- scalar input, scalar output

$$y(x + \mathrm{d}x) = y(x) + a \cdot \mathrm{d}x + \text{(high-order terms)}$$

$$\frac{\partial y}{\partial x} = a \longleftrightarrow \mathrm{d}y = a \cdot \mathrm{d}x$$

- vector input, scalar output

$$\mathbf{x} = (x_1, \ldots, x_n)^\top, \quad \mathrm{d}\mathbf{x} = (\mathrm{d}x_1, \ldots, \mathrm{d}x_n)^\top$$

$$y(\mathbf{x} + \mathrm{d}\mathbf{x}) = y(\mathbf{x}) + \mathbf{a}\mathrm{d}\mathbf{x} + \text{(high-order terms)}$$

We use row-vector to represent gradient

$$\frac{\partial y}{\partial \mathbf{x}} = (\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_n}) = \mathbf{a} \quad \longleftrightarrow \quad \mathrm{d}y = \mathbf{a}\mathrm{d}\mathbf{x}$$

Matrix/Vector Derivative

- In general, vector input, vector output

$$\mathbf{y} = (y_1, \dots, y_m)^\top \qquad \mathbf{x} = (x_1, \dots, \mathbf{x}_n)^\top, \;\; \mathrm{d}\mathbf{x} = (\mathrm{d}x_1, \dots, \mathrm{d}x_n)^\top$$

$$\mathbf{y}(\mathbf{x} + \mathrm{d}\mathbf{x}) = \mathbf{y}(\mathbf{x}) + \mathbf{A}\mathrm{d}\mathbf{x} + (\text{high order terms})$$

What is this? What's size? $m \times n$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} = \mathbf{A} \qquad \longleftrightarrow \qquad \mathrm{d}\mathbf{y} = \mathbf{A}\mathrm{d}\mathbf{x}$$

Matrix/Vector Derivative

$$y(x + \mathrm{d}x) = y(x) + a \cdot \mathrm{d}x + (\text{high-order terms})$$

$$y(\mathbf{x} + \mathrm{d}\mathbf{x}) = y(\mathbf{x}) + \mathbf{a}\mathrm{d}\mathbf{x} + (\text{high-order terms})$$

$$\mathbf{y}(\mathbf{x} + \mathrm{d}\mathbf{x}) = \mathbf{y}(\mathbf{x}) + \mathbf{A}\mathrm{d}\mathbf{x} + (\text{high order terms})$$

In all the cases, {a, **a**, **A**} are derivatives. We define (partial) gradient as the derivative

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{A}$$

This is consistent with the definition of Jacobian. However, we need to be aware if output is scalar, the gradient is a row vector

Matrix/Vector Derivative

- What is the benefit of this notation? We can apply the chain-rule in a natural way

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \qquad \mathbf{x} = \mathbf{g}(\mathbf{z})$$

$$\mathbf{y} : m \times 1 \qquad \mathbf{x} : n \times 1 \qquad \mathbf{z} : q \times 1$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{z}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial \mathbf{z}}$$

$m \times q$

$m \times n$

$n \times q$

Matrix/Vector Derivative

- Some literature uses the notation of derivative transpose

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{A}^\top$$

The benefit is for scalar y, the gradient is a column vector. The cons is when doing the chain rule, you have to multiply from right to left. Why?

Matrix/Vector Derivative

- In whichever case, the key to derive/compute the derivative!

$$\mathbf{y}(\mathbf{x} + \mathrm{d}\mathbf{x}) = \mathbf{y}(\mathbf{x}) + \mathbf{A}\mathrm{d}\mathbf{x} + (\text{high order terms})$$

$$\Downarrow$$

$$\mathrm{d}\mathbf{y} = \mathbf{A}\mathrm{d}\mathbf{x}$$

- The general idea: recursively apply the chain rule to get the target derivative!

Matrix/Vector Derivative

- Take a scalar case as an example

$$y = 3x + \frac{1}{x^2}$$

$$\mathrm{d}y = \mathrm{d}(3x) + \mathrm{d}\left(\frac{1}{x^2}\right) = \ldots$$

Let’s do it together

Matrix/Vector Derivative

- How to apply chain rule for matrices/vectors

like scalar case, we have a set of basic rules in matrix world as well; just keep applying them recursively

$$
\begin{aligned}
\partial \mathbf{A} &= 0 \qquad (\mathbf{A} \text{ is a constant}) \\
\partial(\alpha \mathbf{X}) &= \alpha \partial \mathbf{X} \\
\partial(\mathbf{X}+\mathbf{Y}) &= \partial \mathbf{X} + \partial \mathbf{Y} \\
\partial(\mathrm{Tr}(\mathbf{X})) &= \mathrm{Tr}(\partial \mathbf{X}) \\
\partial(\mathbf{X}\mathbf{Y}) &= (\partial \mathbf{X})\mathbf{Y} + \mathbf{X}(\partial \mathbf{Y}) \\
\partial(\mathbf{X} \circ \mathbf{Y}) &= (\partial \mathbf{X}) \circ \mathbf{Y} + \mathbf{X} \circ (\partial \mathbf{Y}) \\
\partial(\mathbf{X} \otimes \mathbf{Y}) &= (\partial \mathbf{X}) \otimes \mathbf{Y} + \mathbf{X} \otimes (\partial \mathbf{Y}) \\
\partial(\mathbf{X}^{-1}) &= -\mathbf{X}^{-1}(\partial \mathbf{X})\mathbf{X}^{-1} \\
\partial(\det(\mathbf{X})) &= \det(\mathbf{X})\mathrm{Tr}(\mathbf{X}^{-1}\partial \mathbf{X}) \\
\partial(\ln(\det(\mathbf{X}))) &= \mathrm{Tr}(\mathbf{X}^{-1}\partial \mathbf{X}) \\
\partial \mathbf{X}^T &= (\partial \mathbf{X})^T
\end{aligned}
$$

Matrix/Vector Derivative

- Let's do several examples

$$y = (\mathbf{x} + \mathbf{b})^\top (\mathbf{x} + \mathbf{b})$$

$$y = \mathrm{tr}\left((\mathbf{I} + \mathbf{x}\mathbf{x}^\top)^{-1}\right)$$

Matrix/Vector Derivative

- Commonly used references

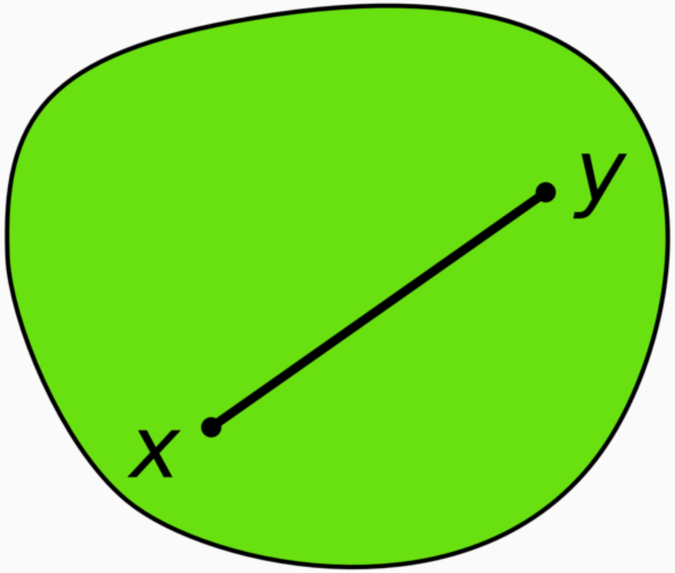
1. Old and New Matrix Algebra Useful for Statistics, By Tom Minka, 2001
2. Matrix Cookbook

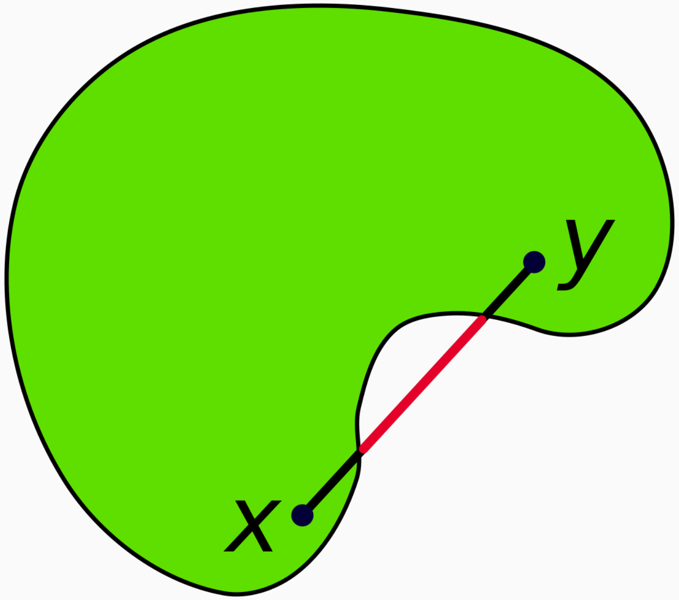
- Strongly suggest the tutorial made by our TM for more examples

https://www.youtube.com/watch?v=artvpNFSFgw

Basics Review

- Convex region/set

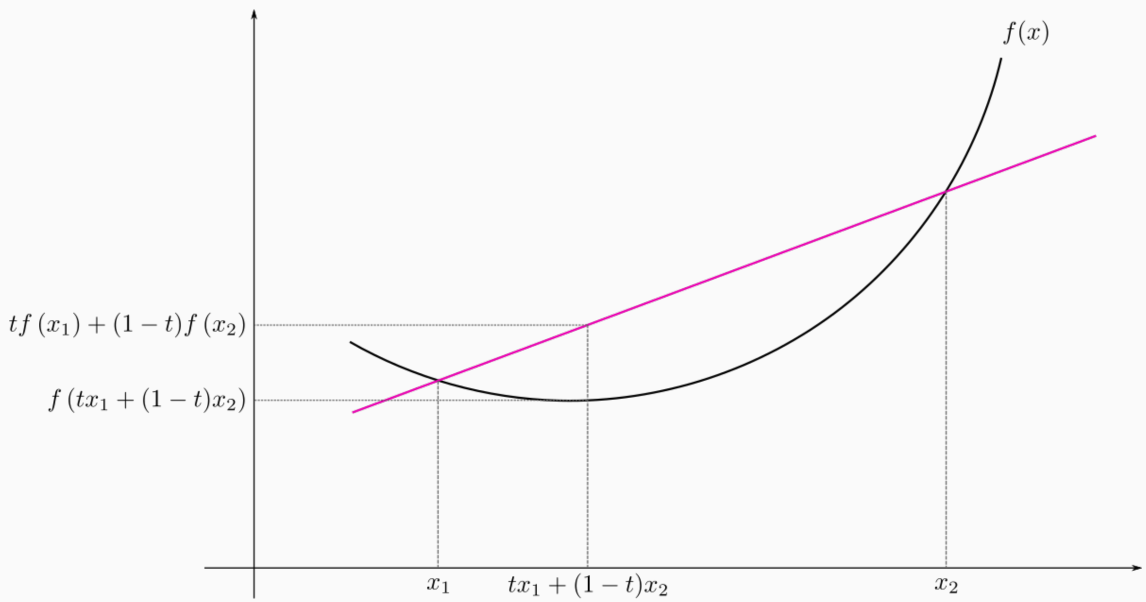




Basic Knowledge Review

- Convex function $f: X \longrightarrow R$
- The input domain X is a convex region/set

$$\forall x_1, x_2 \in X, \forall t \in [0,1]: \qquad f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2).$$



Basic Knowledge Review

- Examples of convex functions

Single variable

$$f(x) = e^x$$

$$f(x) = -\log(x)$$

multivariable

$$f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$$

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{x}$$

- How to determine a convex function?

When differentiable $\quad f(\mathbf{x}) \geq f(\mathbf{y}) + \nabla f(\mathbf{y})^\top (\mathbf{x} - \mathbf{y})$

When twice differentiable $\quad \nabla\nabla f(\mathbf{x}) \succeq 0$

Basic Knowledge Review

- Jensen's inequality (for convex function)

When X is random variable

$$f\big(\, E(X)\big) \le E(\, f(X)\,)$$

$$f\big(\, E(g(X))\big) \le E(\, f(g(X))\,)$$

Basic Knowledge Review

- Convex conjugate (Fenchel's duality)

for an arbitrary convex function f(·), there exists a duality function g(·)

$$f(x) = \max_{\lambda} \quad \lambda x - g(\lambda)$$
$$g(\lambda) = \max_{\mathbf{x}} \quad \lambda x - f(x)$$

Jensen's equality and convex conjugate plays the key role in approximate inference