10-701: Introduction to Machine Learning Lecture 5 – MLE & MAP

Henry Chai

1/31/24

Front Matter

- Announcements:
  - HW1 released 1/24, due 2/2 (Friday) at 11:59 PM
- Recommended Readings:
  - Mitchell, Estimating Probabilities

Recall: Recipe for Linear Regression

1. Define a model and model parameters
    1. Assume $y = \boldsymbol{w}^T \boldsymbol{x}$
    2. Parameters: $\boldsymbol{w} = [w_0, w_1, \dots, w_D]$

2. Write down an objective function
    1. Minimize the mean squared error

$$\ell_{\mathcal{D}}(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} \left(\boldsymbol{w}^T \boldsymbol{x}^{(n)} - y^{(n)} \right)^2$$

3. Optimize the objective w.r.t. the model parameters
    1. Solve in *closed form*: take partial derivatives, set to 0 and solve

Recall: Minimizing the Squared Error

$$\ell_{\mathcal{D}}(\boldsymbol{w}) = \frac{1}{N}\sum_{n=1}^{N}\left(\boldsymbol{w}^T\boldsymbol{x}^{(n)} - y^{(n)}\right)^2 = \frac{1}{N}\sum_{n=1}^{N}\left(\boldsymbol{x}^{(n)^T}\boldsymbol{w} - y^{(n)}\right)^2$$

$$= \frac{1}{N}\|X\boldsymbol{w} - \boldsymbol{y}\|_2^2 \text{ where } \|\boldsymbol{z}\|_2 = \sqrt{\sum_{d=1}^{D} z_d^2} = \sqrt{\boldsymbol{z}^T\boldsymbol{z}}$$

$$= \frac{1}{N}(X\boldsymbol{w} - \boldsymbol{y})^T(X\boldsymbol{w} - \boldsymbol{y})$$

$$= \frac{1}{N}(\boldsymbol{w}^T X^T X\boldsymbol{w} - 2\boldsymbol{w}^T X^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y})$$

$$\nabla_{\boldsymbol{w}}\ell_{\mathcal{D}}(\hat{\boldsymbol{w}}) = \frac{1}{N}(2X^T X\hat{\boldsymbol{w}} - 2X^T\boldsymbol{y}) = 0$$

$$\rightarrow X^T X\hat{\boldsymbol{w}} = X^T\boldsymbol{y}$$

$$\rightarrow \hat{\boldsymbol{w}} = (X^T X)^{-1}X^T\boldsymbol{y}$$

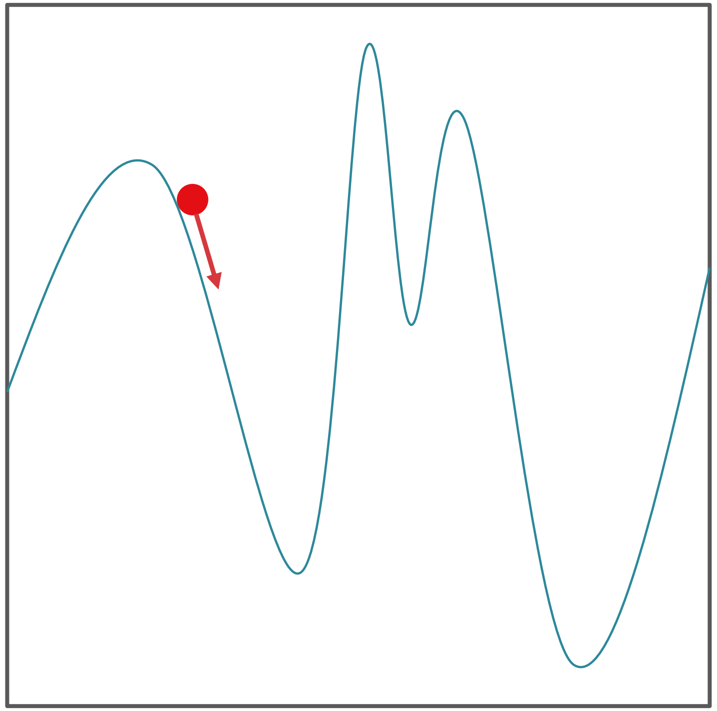
$$\hat{\boldsymbol{w}} = (X^T X)^{-1} X^T \boldsymbol{y}$$

Recall: Closed Form Solution

1. Is $X^T X$ invertible?
   - When $N \gg D + 1$, $X^T X$ is (almost always) full rank and therefore, invertible
   - If $X^T X$ is not invertible (occurs when one of the features is a linear combination of the others) then there are infinitely many solutions.
2. If so, how computationally expensive is inverting $X^T X$?
   - $X^T X \in \mathbb{R}^{D+1 \times D+1}$ so inverting $X^T X$ takes $O(D^3)$ time…
     - Computing $X^T X$ takes $O(ND^2)$ time
   - What alternative optimization method can we use to minimize the mean squared error?

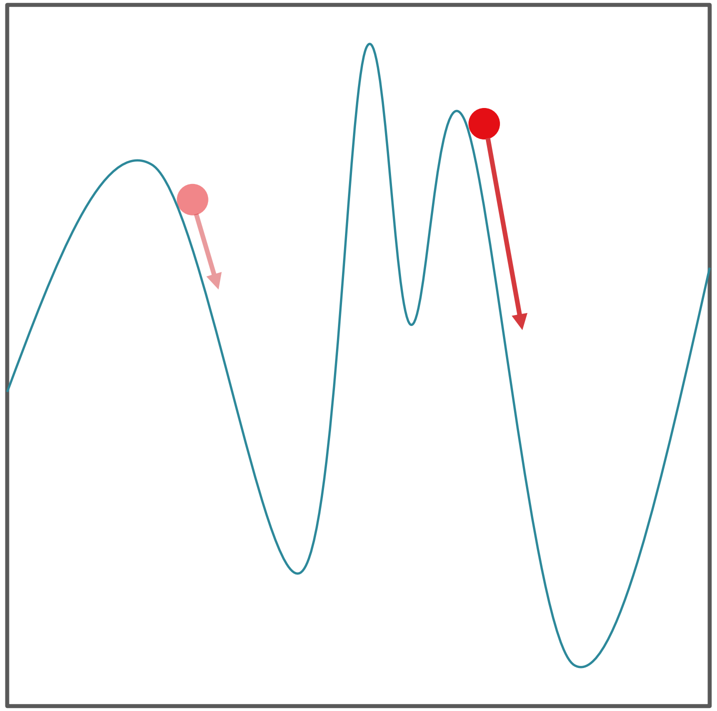
Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



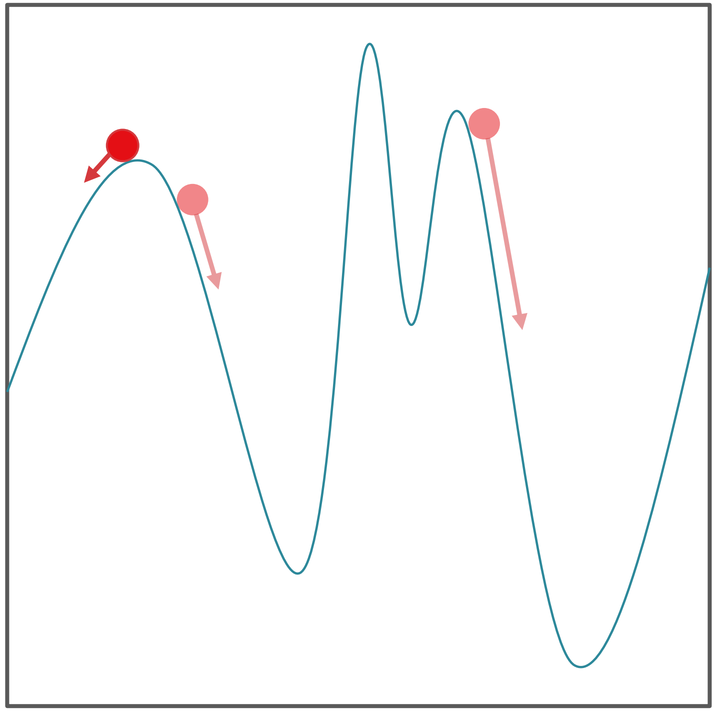
Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



Gradient Descent: Intuition

- An iterative method for minimizing functions
- Requires the gradient to exist everywhere



Gradient Descent

- Suppose the current weight vector is $\boldsymbol{w}^{(t)}$
- Move some distance, η, in the "most downhill" direction, $\widehat{\boldsymbol{v}}$:

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \eta\widehat{\boldsymbol{v}}$$

Gradient Descent: Step Direction

- Suppose the current weight vector is $\boldsymbol{w}^{(t)}$
- Move some distance, η, in the "most downhill" direction, $\widehat{\boldsymbol{v}}$:

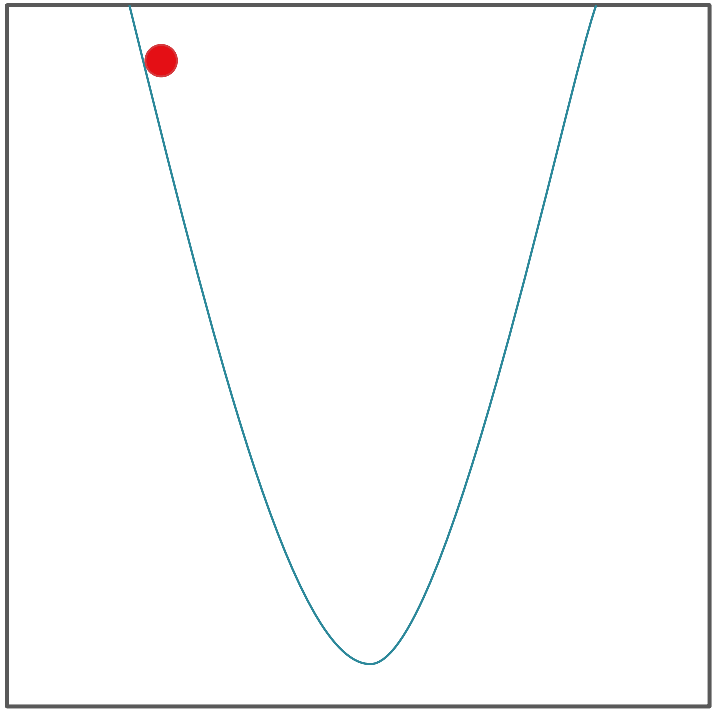
$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \eta\widehat{\boldsymbol{v}}$$

- The gradient points in the direction of steepest *increase* …
- … so $\widehat{\boldsymbol{v}}$ should point in the opposite direction:

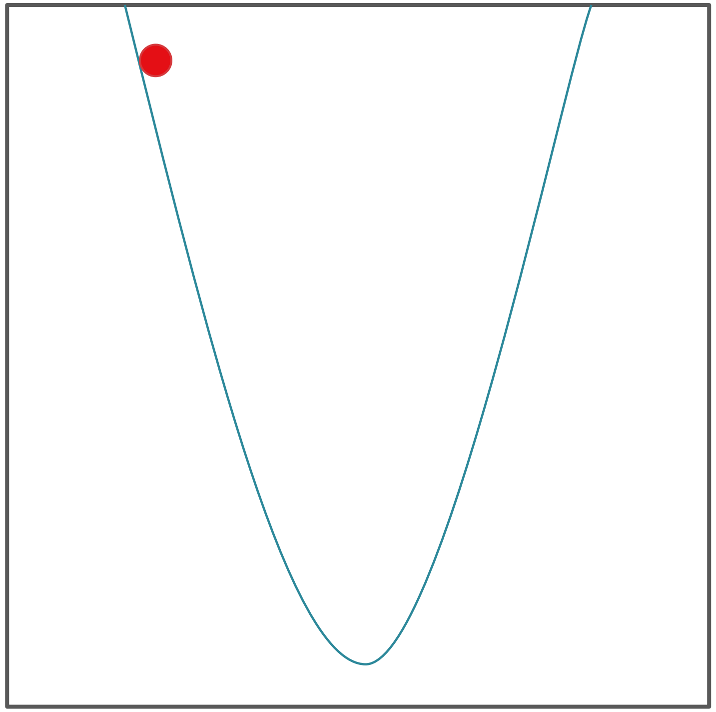
$$\widehat{\boldsymbol{v}}^{(t)} = -\frac{\nabla_{\boldsymbol{w}}\ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)}{\left\|\nabla_{\boldsymbol{w}}\ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|_2}$$

$$-\nabla_w \ell_{\mathcal{D}}(w^{(t)})$$

Gradient Descent: Step Size

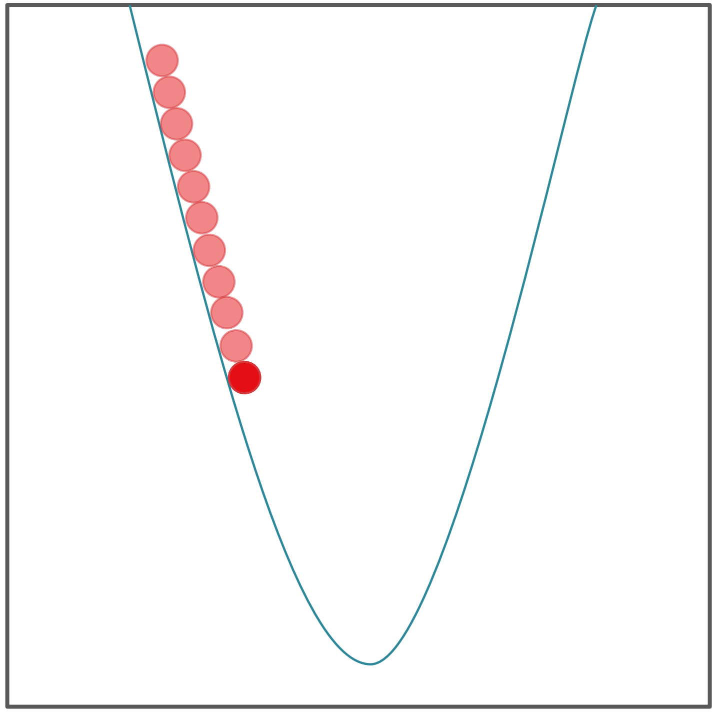


Small η

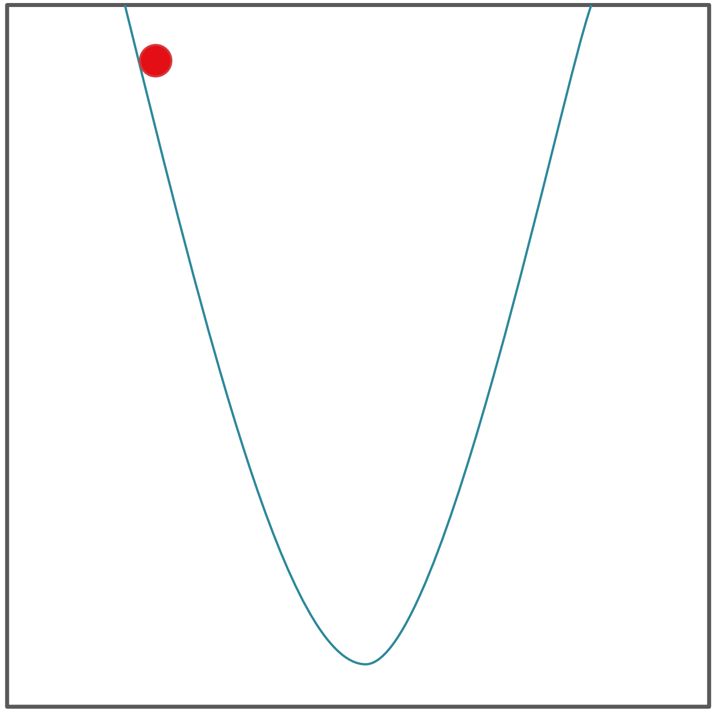


Large η

Gradient Descent: Step Size

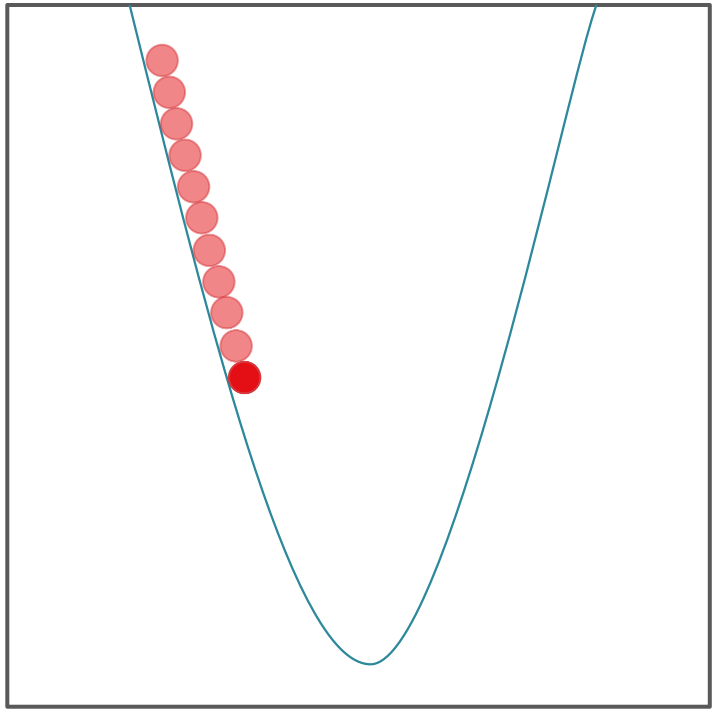


Small η

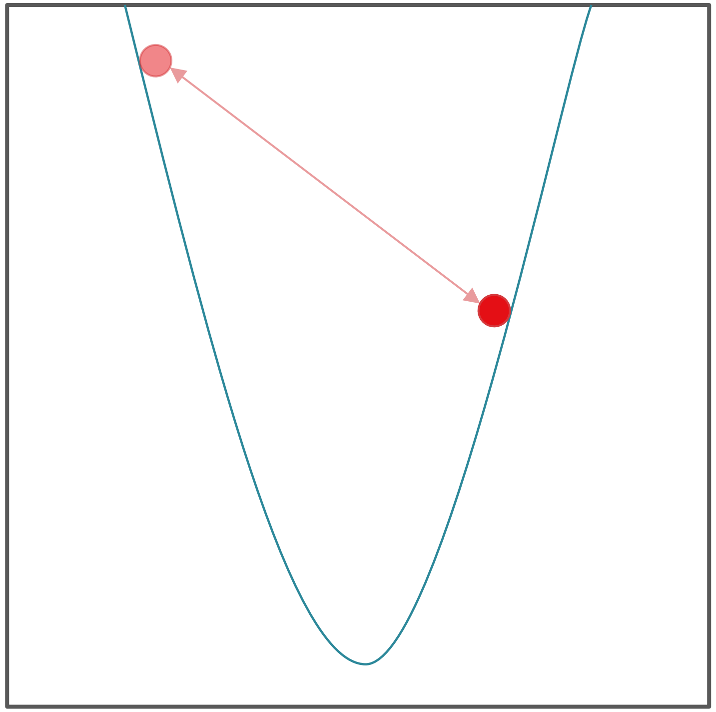


Large η

Gradient Descent: Step Size



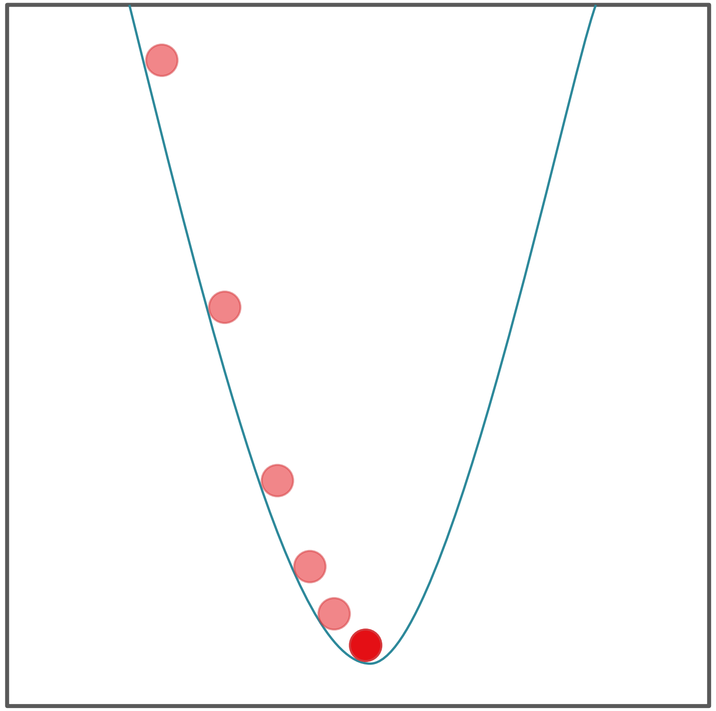
Small η



Large η

Gradient Descent: Step Size

- Use a variable $\eta^{(t)}$ instead of a fixed η!



- Set $\eta^{(t)} = \eta^{(0)} \left\| \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}} \left(\boldsymbol{w}^{(t)} \right) \right\|$
- $\left\| \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}} \left(\boldsymbol{w}^{(t)} \right) \right\|$ decreases as $\ell_{\mathcal{D}}$ approaches its minimum $\rightarrow \eta^{(t)}$ (hopefully) decreases over time

Gradient Descent

- $\hat{\boldsymbol{v}}^{(t)} = -\frac{\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)}{\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|}$
- $\eta^{(t)} = \eta^{(0)}\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\|$
- $\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \eta^{(t)}\hat{\boldsymbol{v}}^{(t)}$

$$= w^{(t)} + \left(-\frac{\nabla_w \ell_D(w^{(t)})}{\|\nabla_w \ell_D(w^{(t)})\|}\right)\eta^{(0)}\|\nabla_w \ell_D(w^{(t)})\|$$

$$= w^{(t)} - \eta \quad \nabla_w \ell_D(w^{(t)})$$

Gradient Descent

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta$

1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. Compute the gradient:
      $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right) = \frac{2}{N}\left(X^T X w - X^T y\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Gradient Descent

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta, \epsilon$

1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While $\left\|\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)\right\| > \epsilon$
   a. Compute the gradient:
      $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Gradient Descent

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta, T$

1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While $t < T$
   a. Compute the gradient:
      $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Why Gradient Descent for linear regression?

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, \eta, T$

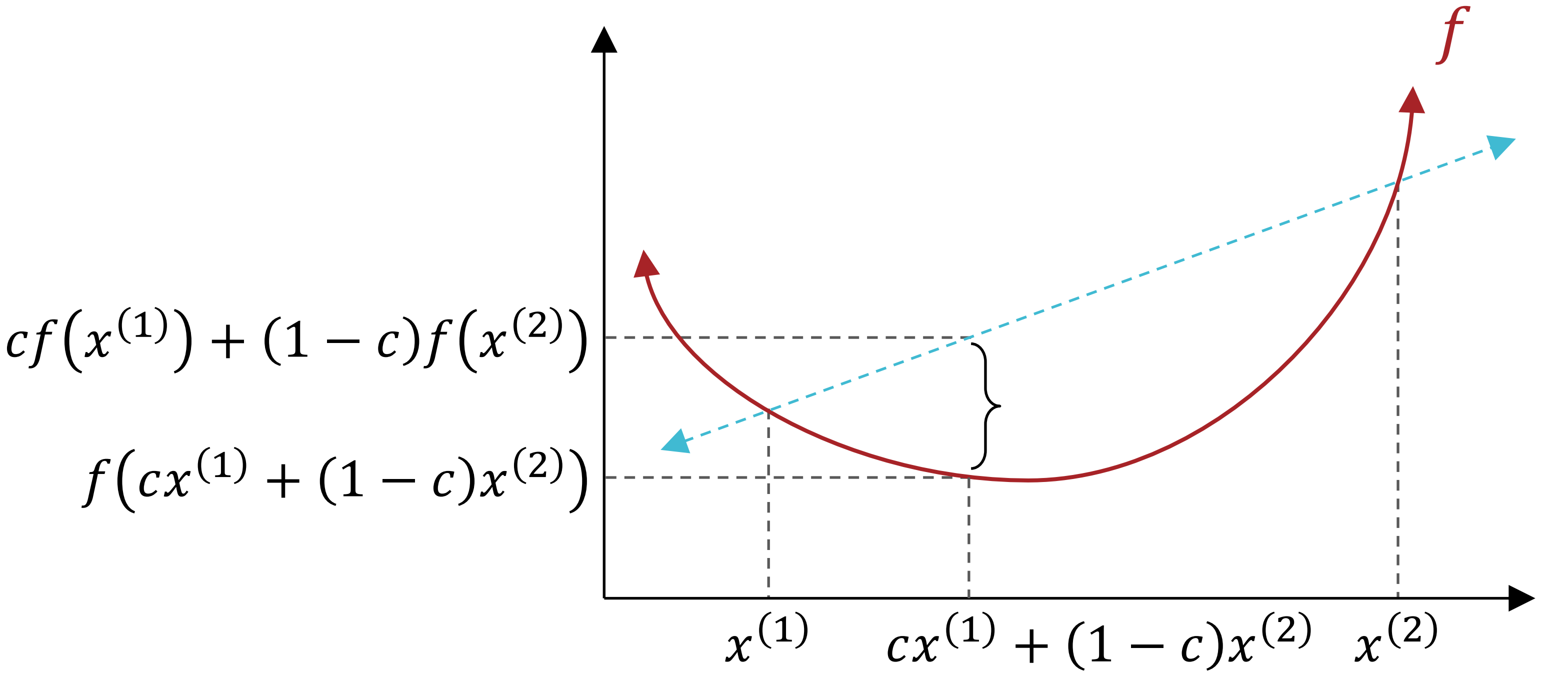
1. Initialize $\boldsymbol{w}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. Compute the gradient:
   $\nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right) = \frac{2}{N}\left(X^T X w - X^T y\right)$
   b. Update $\boldsymbol{w}$: $\boldsymbol{w}^{(t+1)} \leftarrow \boldsymbol{w}^{(t)} - \eta \nabla_{\boldsymbol{w}} \ell_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$
   c. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{w}^{(t)}$

Convexity

- A function $f: \mathbb{R}^D \rightarrow \mathbb{R}$ is convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 \leq c \leq 1$

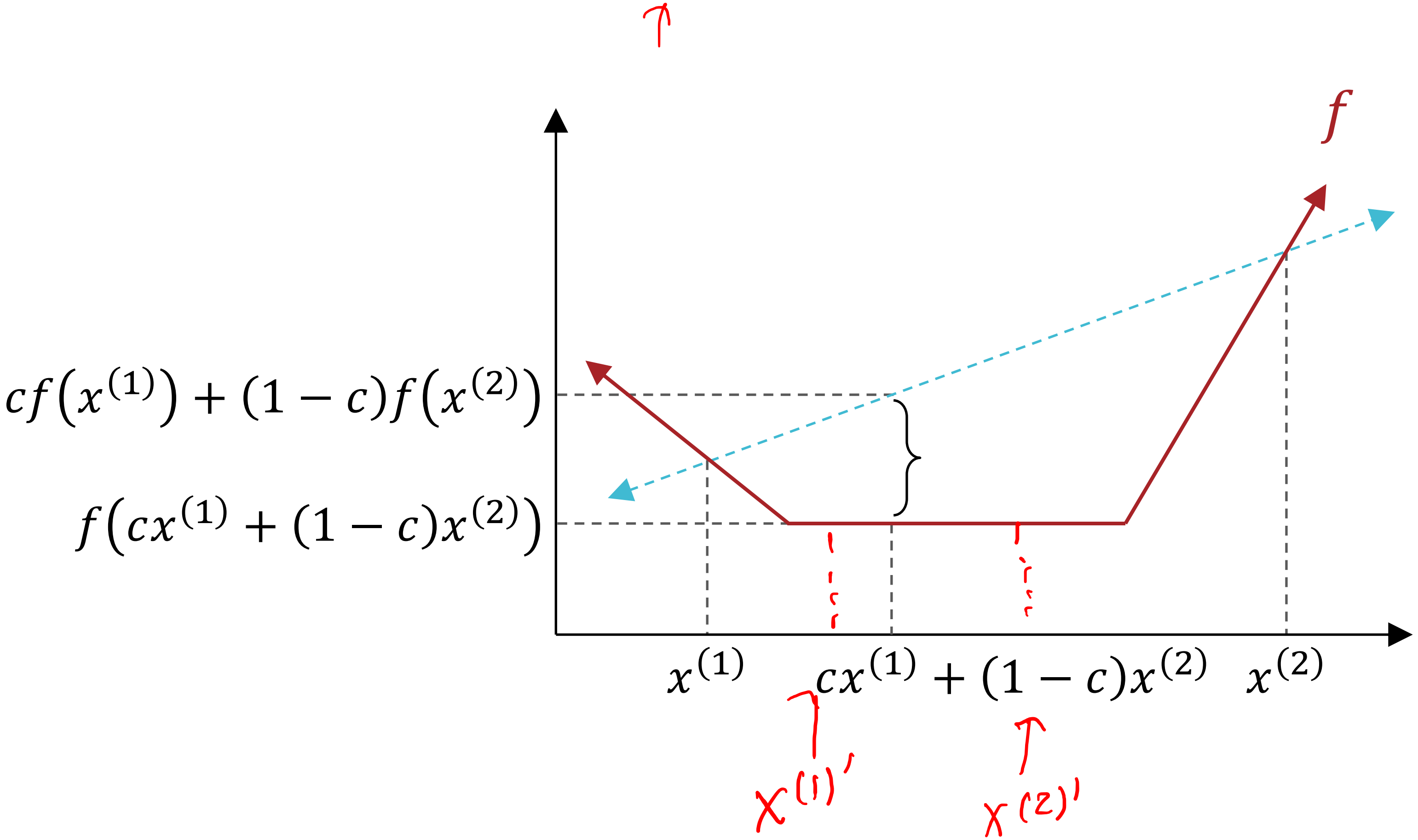
  $$f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) \leq cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$$



Convexity

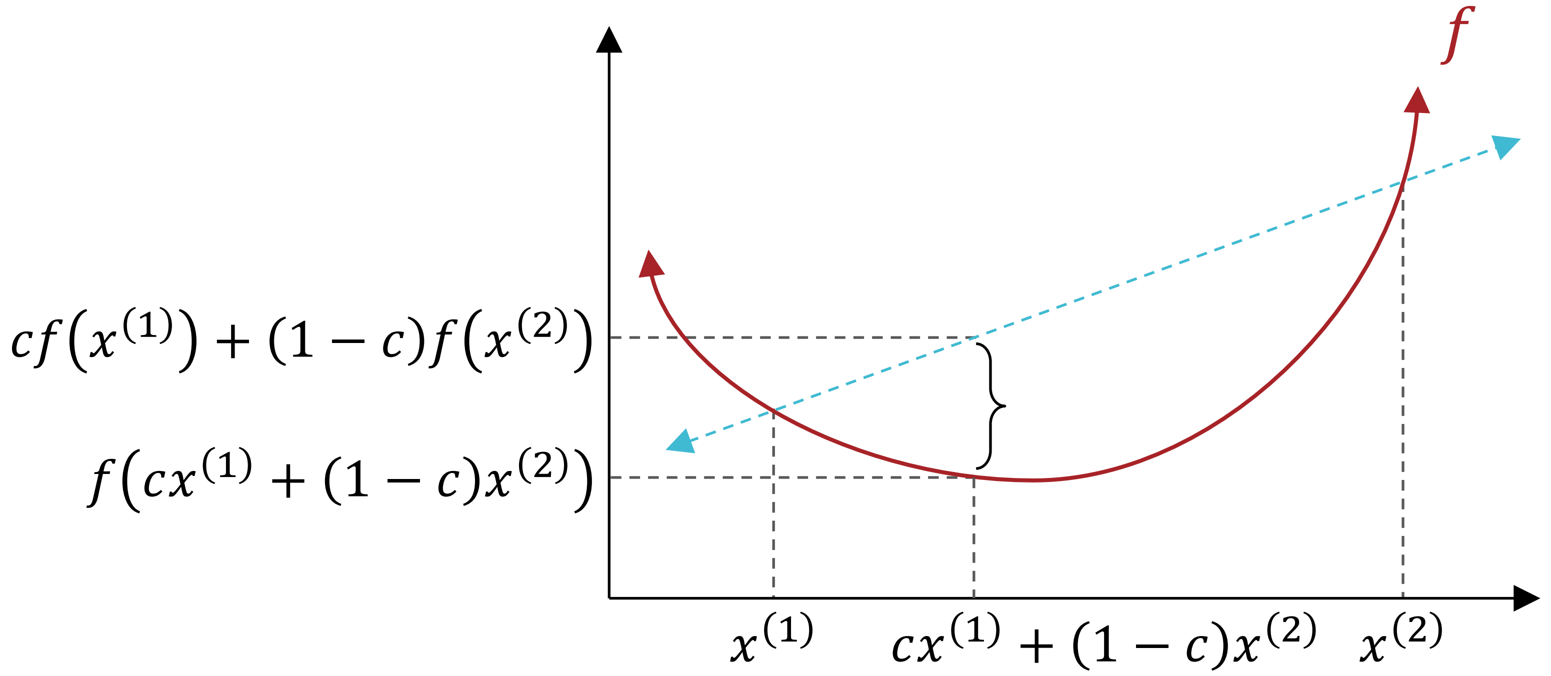
- A function $f : \mathbb{R}^D \rightarrow \mathbb{R}$ is convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 \leq c \leq 1$

$$f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) \leq cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$$

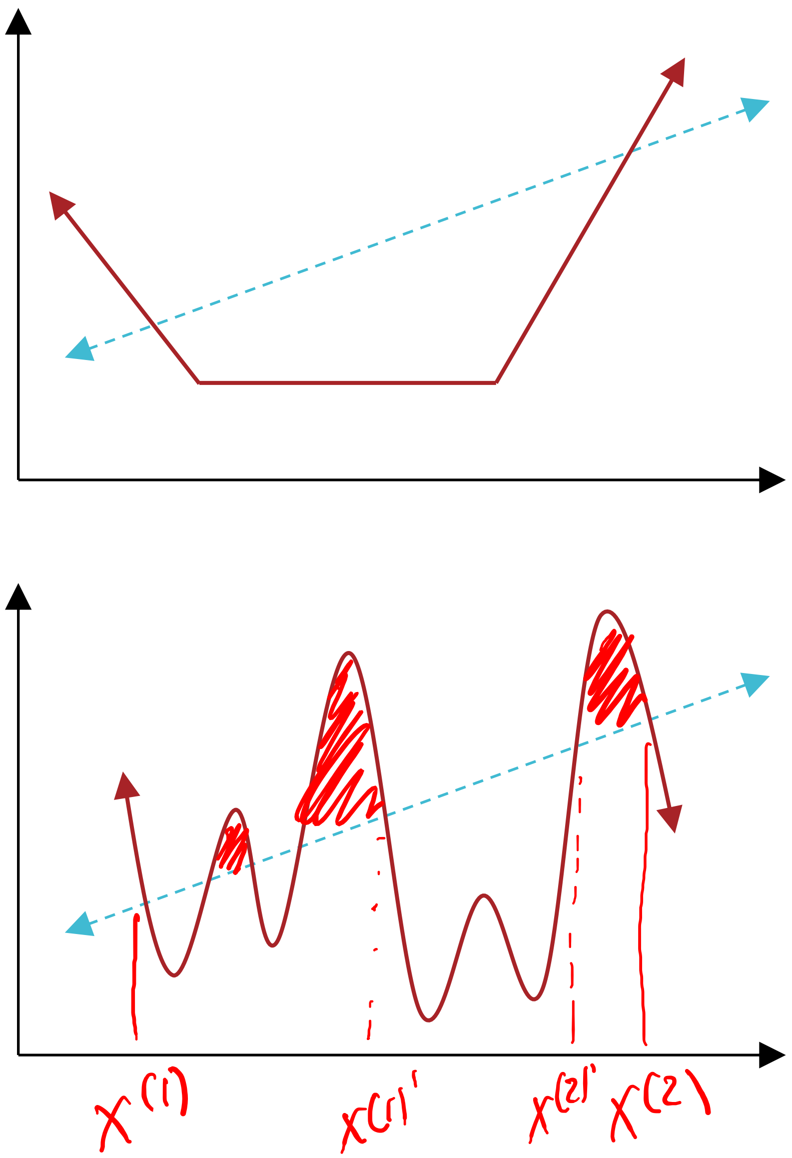


Convexity

- A function $f: \mathbb{R}^D \rightarrow \mathbb{R}$ is *strictly* convex if $\forall\, \boldsymbol{x}^{(1)} \in \mathbb{R}^D, \boldsymbol{x}^{(2)} \in \mathbb{R}^D$ and $0 < c < 1$
  $f\left(c\boldsymbol{x}^{(1)} + (1-c)\boldsymbol{x}^{(2)}\right) < cf\left(\boldsymbol{x}^{(1)}\right) + (1-c)f\left(\boldsymbol{x}^{(2)}\right)$



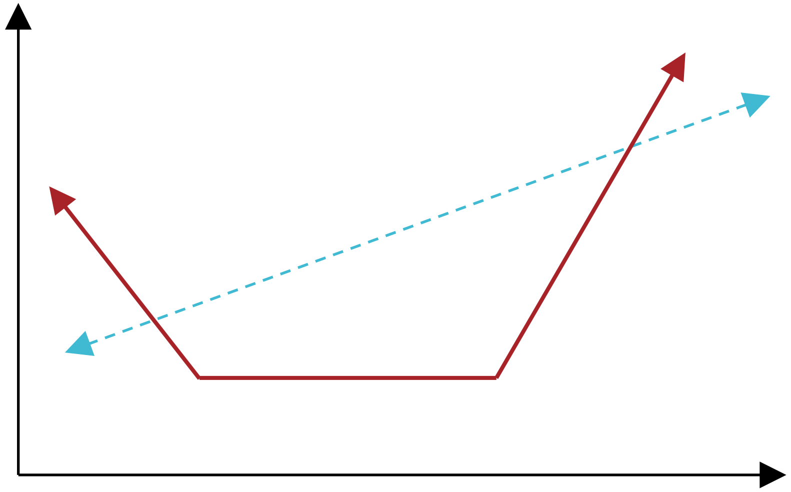
Convexity



Convex functions

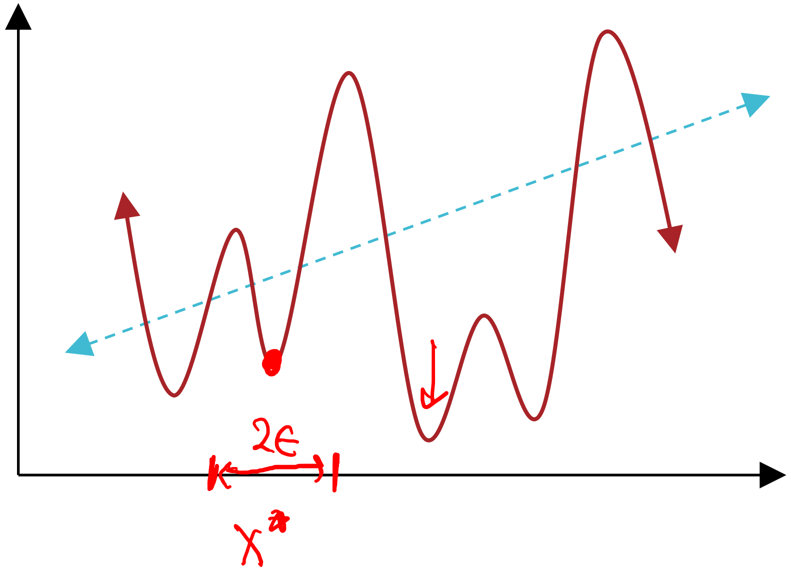
Non-convex functions

Convexity



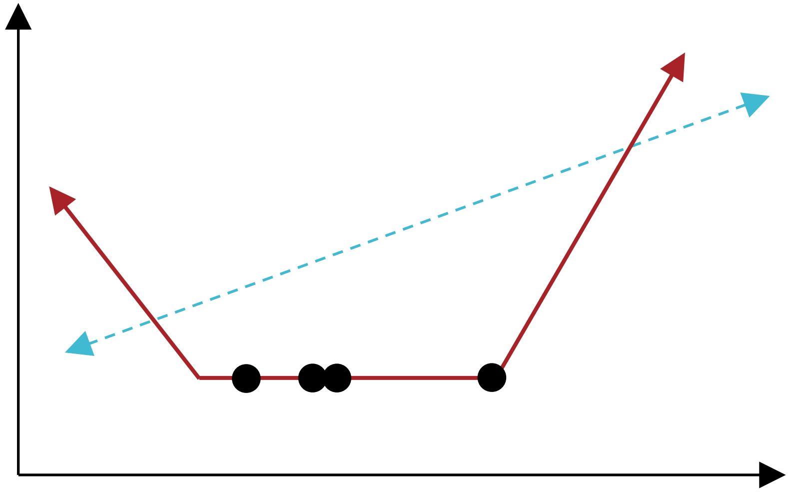
Given a function $f: \mathbb{R}^D \to \mathbb{R}$

- $\boldsymbol{x}^*$ is a *global* minimum iff $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \; \forall \, \boldsymbol{x} \in \mathbb{R}^D$



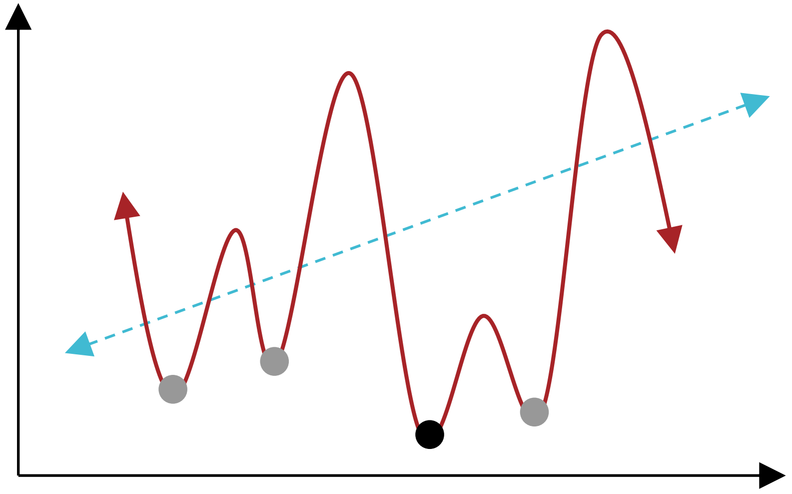
- $\boldsymbol{x}^*$ is a *local* minimum iff $\exists \, \epsilon$ s.t. $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \; \forall$ $\boldsymbol{x}$ s.t. $\|\boldsymbol{x} - \boldsymbol{x}^*\|_2 < \epsilon$

Convexity



Convex functions:

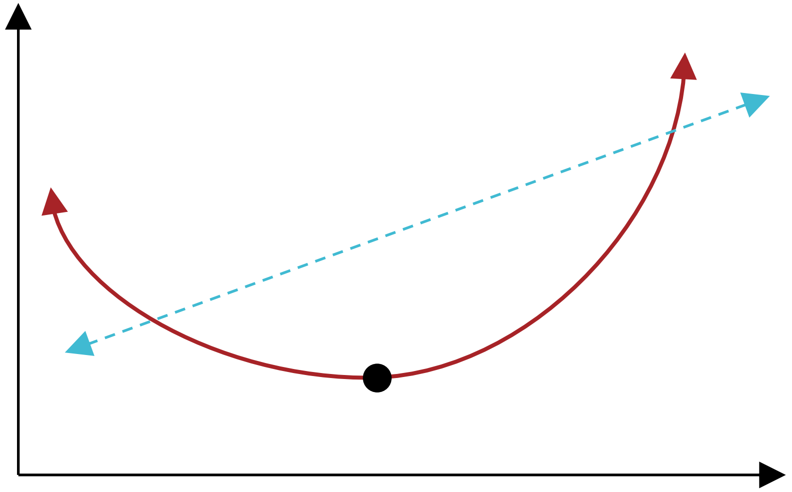
Each local minimum is a global minimum!



Non-convex functions:

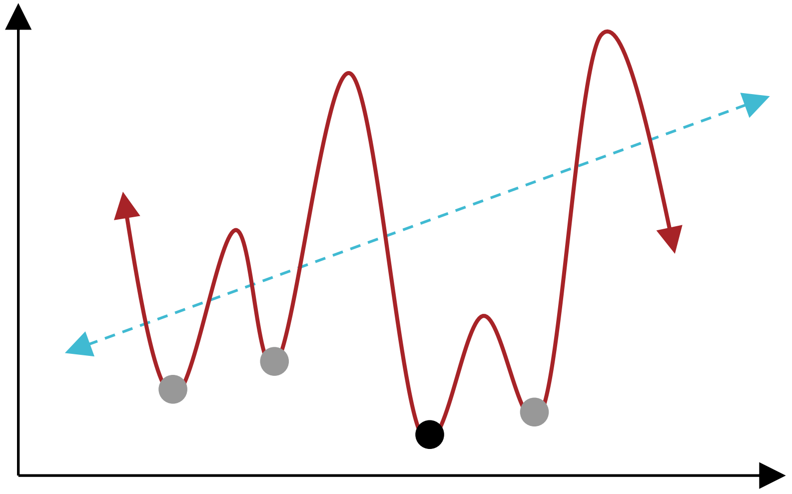
A local minimum may or may not be a global minimum…

Convexity



Strictly convex functions:

There exists a unique global minimum!

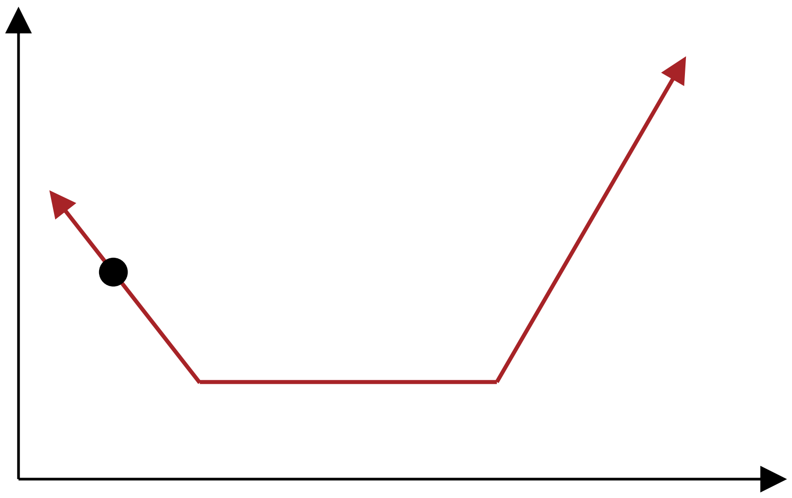


Non-convex functions:

A local minimum may or may not be a global minimum…

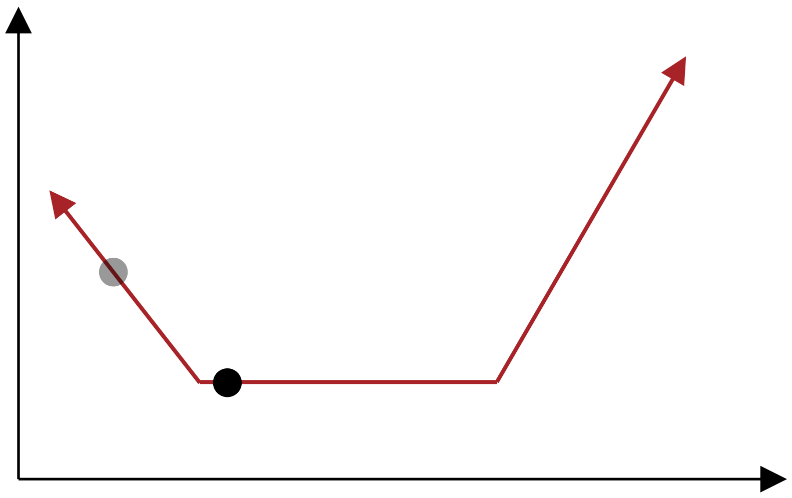
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



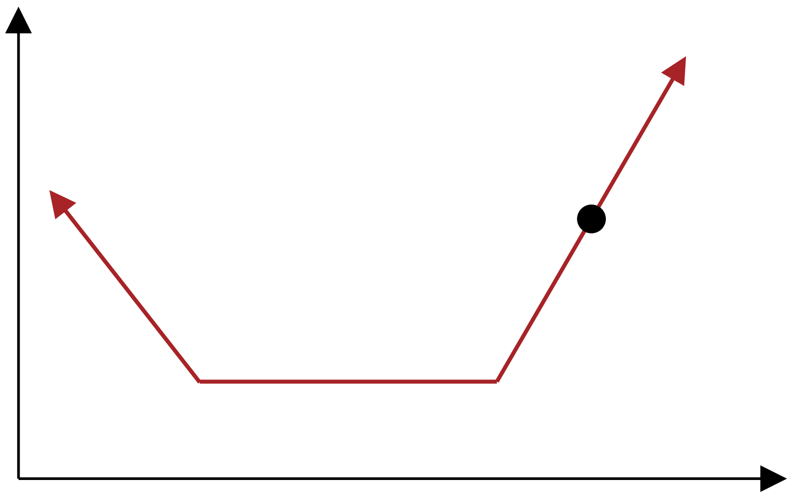
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



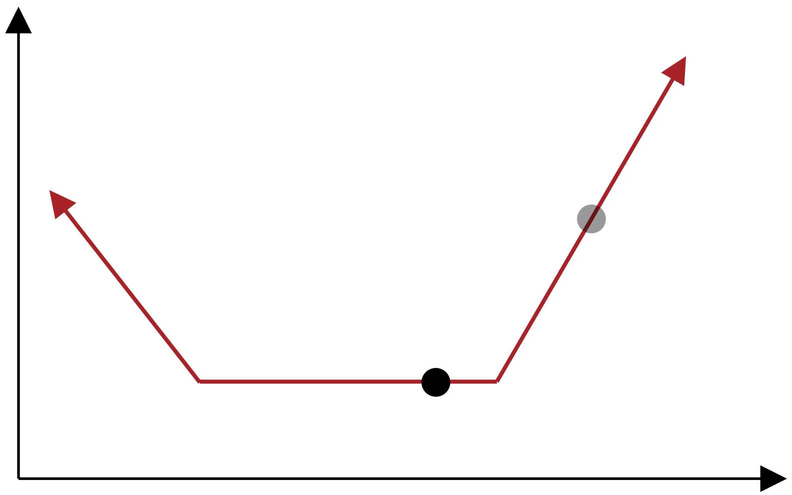
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



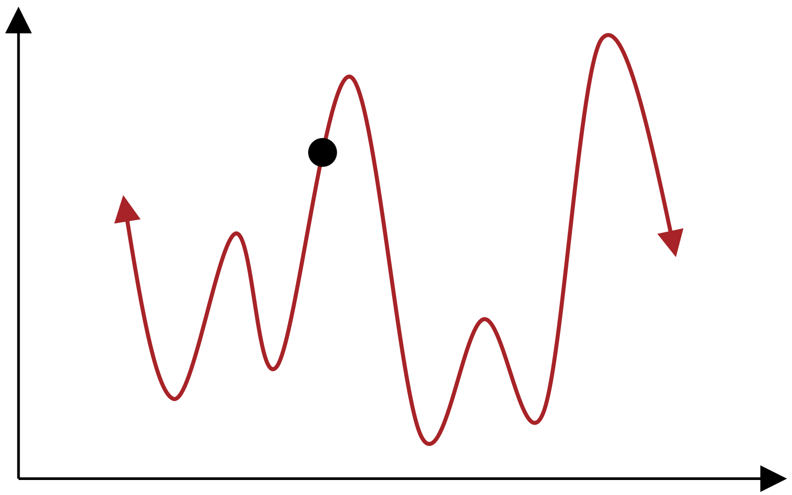
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



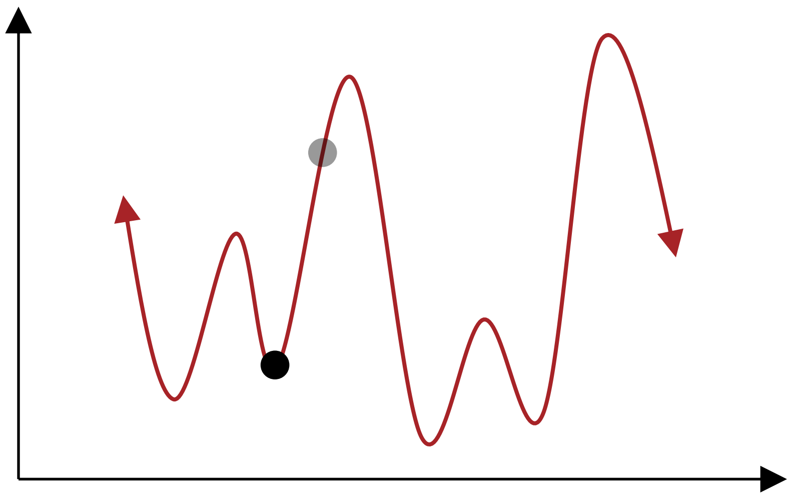
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



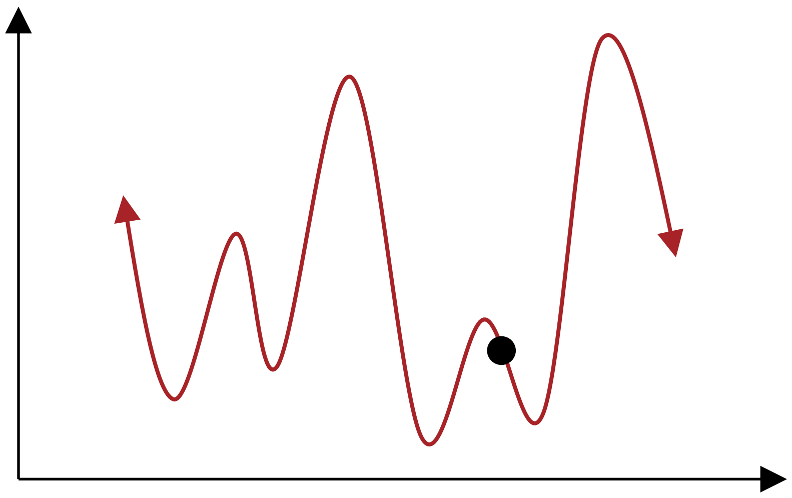
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



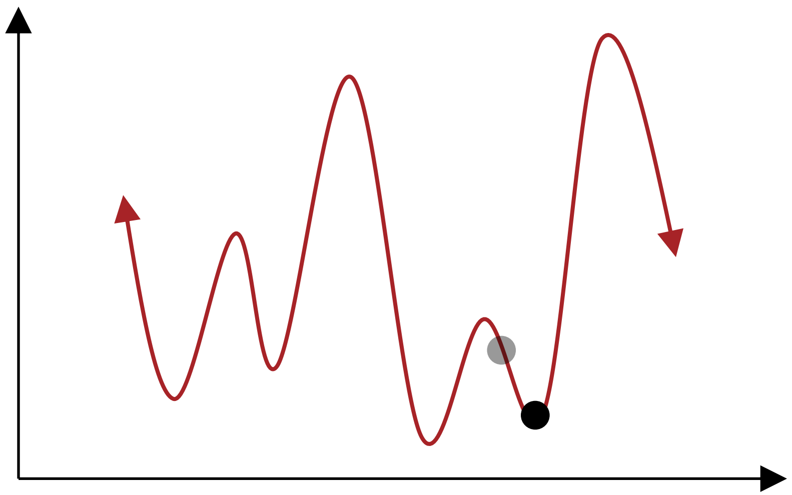
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



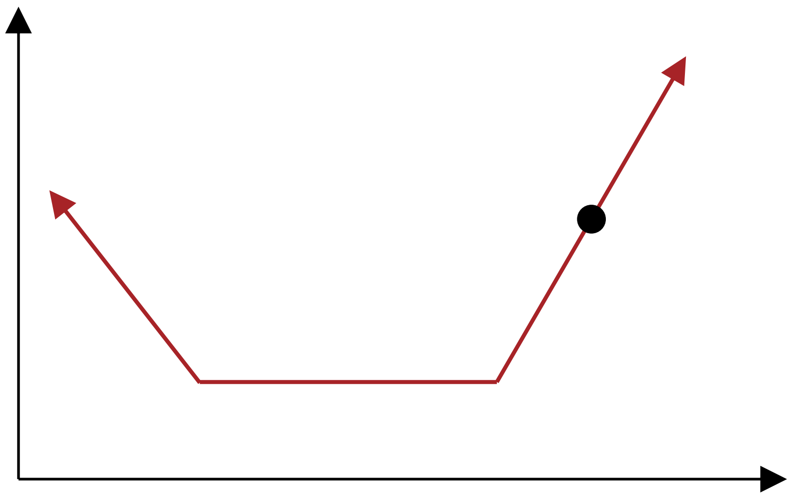
Gradient Descent & Convexity

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Not ideal if the objective function is non-convex…



The squared error for linear regression is convex (but not strictly convex)!

- Gradient descent is a local optimization algorithm – it will converge to a local minimum (if it converges)
  - Works great if the objective function is convex!



$H_{\boldsymbol{w}}\ell_{\mathcal{D}}(\boldsymbol{w}) = \frac{2}{N}X^T X$ is positive *semi*-definite

Key Takeaways

- Closed form solution for linear regression
  - Setting the gradient equal to 0 and solving for critical points
  - Potential issues: invertibility and computational costs
- Gradient descent
  - Effect of step size
  - Termination criteria
- Convexity vs. non-convexity
  - Strong vs. weak convexity
  - Implications for local, global and unique optima

Probabilistic Learning

- Previously:
  - (Unknown) Target function, $c^*: \mathcal{X} \rightarrow \mathcal{Y}$
  - Classifier, $h : \mathcal{X} \rightarrow \mathcal{Y}$
  - Goal: find a classifier, h, that best approximates c^*
- Now:
  - (Unknown) Target *distribution*, $y \sim p^*(Y|\boldsymbol{x})$
  - Distribution, $p(Y|\boldsymbol{x})$
  - Goal: find a distribution, p, that best approximates p^*

Likelihood

- Given N independent, identically distribution (iid) samples $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ of a random variable X
  - If X is discrete with probability mass function (pmf) $p(X|\theta)$, then the *likelihood* of $\mathcal{D}$ is

$$L(\theta) = \prod_{n=1}^{N} p\left(x^{(n)}|\theta\right)$$

  - If X is continuous with probability density function (pdf) $f(X|\theta)$, then the *likelihood* of $\mathcal{D}$ is

$$L(\theta) = \prod_{n=1}^{N} f\left(x^{(n)}|\theta\right)$$

Log-Likelihood

- Given N independent, identically distribution (iid) samples $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ of a random variable X
  - If X is discrete with probability mass function (pmf) $p(X|\theta)$, then the *log-likelihood* of $\mathcal{D}$ is

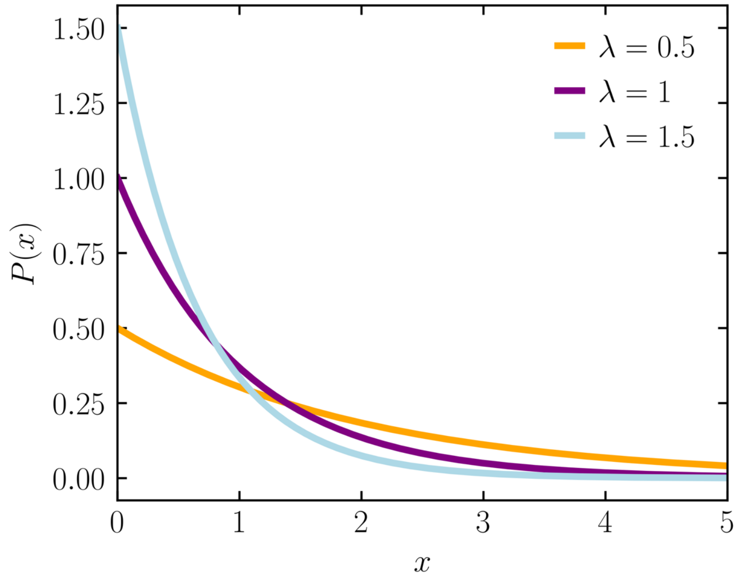
$$\ell(\theta) = \log \prod_{n=1}^{N} p\left(x^{(n)}|\theta\right) = \sum_{n=1}^{N} \log p\left(x^{(n)}|\theta\right)$$

  - If X is continuous with probability density function (pdf) $f(X|\theta)$, then the *log-likelihood* of $\mathcal{D}$ is

$$\ell(\theta) = \log \prod_{n=1}^{N} f\left(x^{(n)}|\theta\right) = \sum_{n=1}^{N} \log f\left(x^{(n)}|\theta\right)$$

Maximum Likelihood Estimation (MLE)

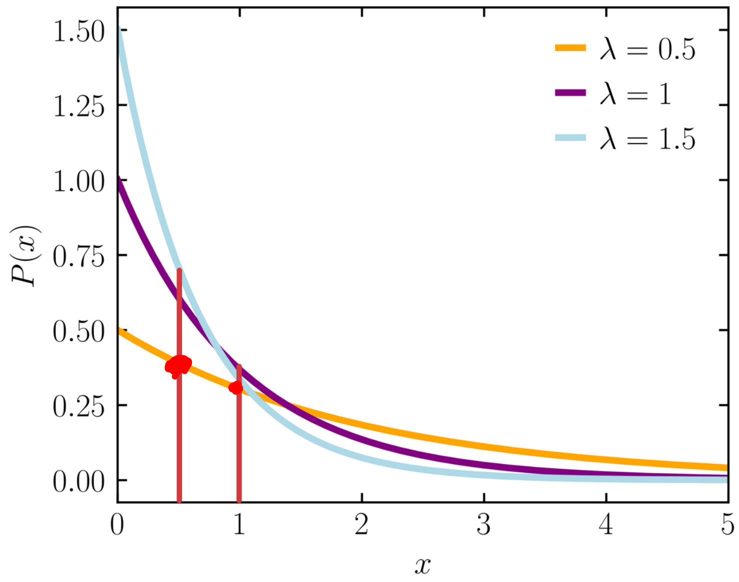
- Insight: every valid probability distribution has a finite amount of probability mass as it must sum/integrate to 1
- Idea: set the parameter(s) so that the likelihood of the samples is maximized
- Intuition: assign as much of the (finite) probability mass to the observed data *at the expense of unobserved data*
- Example: the exponential distribution



Source: https://en.wikipedia.org/wiki/Exponential_distribution#/media/File:Exponential_probability_density.svg

Maximum Likelihood Estimation (MLE)

- Insight: every valid probability distribution has a finite amount of probability mass as it must sum/integrate to 1
- Idea: set the parameter(s) so that the likelihood of the samples is maximized
- Intuition: assign as much of the (finite) probability mass to the observed data *at the expense of unobserved data*
- Example: the exponential distribution

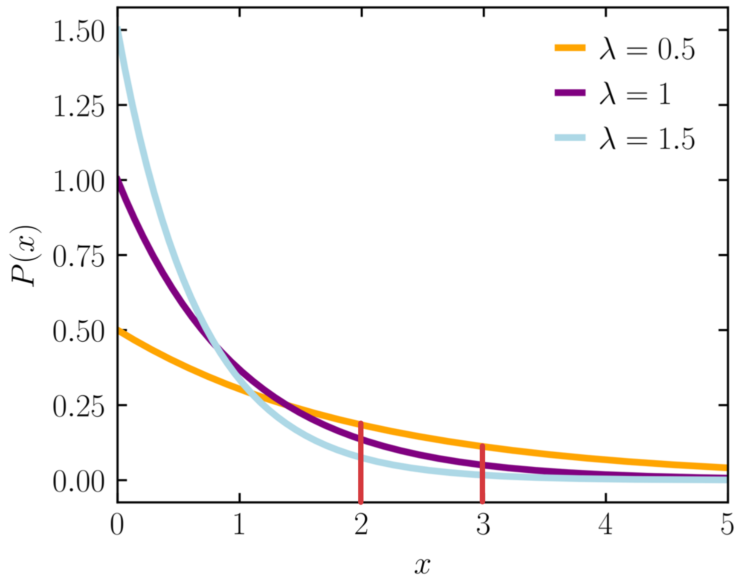


$\{x^{(1)} = 0.5, x^{(2)} = 1\}$

Source: https://en.wikipedia.org/wiki/Exponential_distribution#/media/File:Exponential_probability_density.svg

Maximum Likelihood Estimation (MLE)

- Insight: every valid probability distribution has a finite amount of probability mass as it must sum/integrate to 1
- Idea: set the parameter(s) so that the likelihood of the samples is maximized
- Intuition: assign as much of the (finite) probability mass to the observed data *at the expense of unobserved data*
- Example: the exponential distribution



$\{x^{(1)} = 2, x^{(2)} = 3\}$

Source: https://en.wikipedia.org/wiki/Exponential_distribution#/media/File:Exponential_probability_density.svg

Exponential Distribution MLE

- The pdf of the exponential distribution is

$$f(x|\lambda) = \lambda e^{-\lambda x}$$

- Given N **iid** (independent and identically distributed) samples $\{x^{(1)}, \ldots, x^{(N)}\}$, the likelihood is

$$L(\lambda) = \prod_{n=1}^{N} \lambda e^{-\lambda x^{(n)}}$$

Exponential Distribution MLE

- The pdf of the exponential distribution is

$$f(x|\lambda) = \lambda e^{-\lambda x}$$

- Given N **iid** (independent and identically distributed) samples $\{x^{(1)}, \ldots, x^{(N)}\}$, the log-likelihood is

$$\ell(\lambda) = \sum_{n=1}^{N} \log\left(\lambda e^{-\lambda x^{(n)}}\right)$$

$$= \sum_{n=1}^{N} \log(\lambda) + \left(-\lambda x^{(n)}\right)$$

$$= N \log \lambda - \lambda \sum_{n=1}^{N} x^{(n)}$$

$$\frac{\partial \ell}{\partial \lambda} = \frac{N}{\lambda} - \sum_{n=1}^{N} x^{(n)}$$

$$\Rightarrow \frac{N}{\hat{\lambda}} - \sum_{n=1}^{N} x^{(n)} = 0 \Rightarrow \frac{N}{\hat{\lambda}} = \sum_{n=1}^{N} x^{(n)} \Rightarrow \hat{\lambda} = \frac{N}{\sum_{n=1}^{N} x^{(n)}}$$

M(C)LE for Linear Regression

- If we assume a linear model with additive Gaussian noise

$$y = \boldsymbol{\omega}^T \boldsymbol{x} + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2) \rightarrow y \sim N(\boldsymbol{\omega}^T \boldsymbol{x}, \sigma^2) \ldots$$

then given $X = \begin{bmatrix} 1 & \boldsymbol{x}^{(1)^T} \\ 1 & \boldsymbol{x}^{(2)^T} \\ \vdots & \vdots \\ 1 & \boldsymbol{x}^{(N)^T} \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$, the MLE of $\boldsymbol{\omega}$ is

$$\hat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} \log P(\boldsymbol{y} | X, \boldsymbol{\omega})$$

$$\vdots$$

$$= (X^T X)^{-1} X^T \boldsymbol{y}$$

Bernoulli Distribution MLE

- A Bernoulli random variable takes value 1 with probability ϕ and value 0 with probability $1 - \phi$
- The pmf of the Bernoulli distribution is

$$p(x|\phi) = \phi^x (1 - \phi)^{1-x}$$

Coin Flipping MLE

- A Bernoulli random variable takes value 1 (or heads) with probability ϕ and value 0 (or tails) with probability $1 - \phi$
- The pmf of the Bernoulli distribution is

$$p(x|\phi) = \phi^x (1 - \phi)^{1-x}$$

- Given N iid samples $\{x^{(1)}, \dots, x^{(N)}\}$, the log-likelihood is

$$\ell(\phi) = \sum_{n=1}^{N} \log\left(p(x^{(n)} | \phi)\right)$$

$$= \sum_{n=1}^{N} \log\left(\phi^{x^{(n)}} (1-\phi)^{1-x^{(n)}}\right)$$

$$= \sum_{n=1}^{N} x^{(n)} \log(\phi) + (1-x^{(n)}) \log(1-\phi)$$

$$= N_1 \log \phi + N_0 \log(1-\phi)$$

where N_i = # of i's in D

Coin Flipping MLE

$$\ell(\phi) = N_1 \log \phi + N_0 \log(1-\phi)$$

- A Bernoulli random variable takes value 1 (or heads) with probability ϕ and value 0 (or tails) with probability $1-\phi$
- The pmf of the Bernoulli distribution is

$$p(x|\phi) = \phi^x (1-\phi)^{1-x}$$

- The partial derivative of the log-likelihood is

$$\frac{\partial \ell}{\partial \phi} = \frac{N_1}{\phi} - \frac{N_0}{1-\phi}$$

$$\Rightarrow \frac{N_1}{\hat{\phi}} - \frac{N_0}{1-\hat{\phi}} = 0$$

$$\Rightarrow \frac{N_1}{\hat{\phi}} = \frac{N_0}{1-\hat{\phi}} \Rightarrow N_1(1-\hat{\phi}) = N_0\hat{\phi}$$

$$\Rightarrow N_1 = \hat{\phi} N_1 + \hat{\phi} N_0 \Rightarrow \hat{\phi} = \frac{N_1}{N_1 + N_0}$$

Maximum a Posteriori (MAP) Estimation

- Insight: sometimes we have *prior* information we want to incorporate into parameter estimation
- Idea: use Bayes rule to reason about the *posterior* distribution over the parameters

MLE finds $\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \ L(\Theta) = P(D|\Theta)$

MAP finds

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \ P(\Theta|D)$$

$$= \underset{\Theta}{\operatorname{argmax}} \ \frac{P(D|\Theta)P(\Theta)}{P(D)}$$

$$= \underset{\Theta}{\operatorname{argmax}} \ P(D|\Theta)P(\Theta)$$

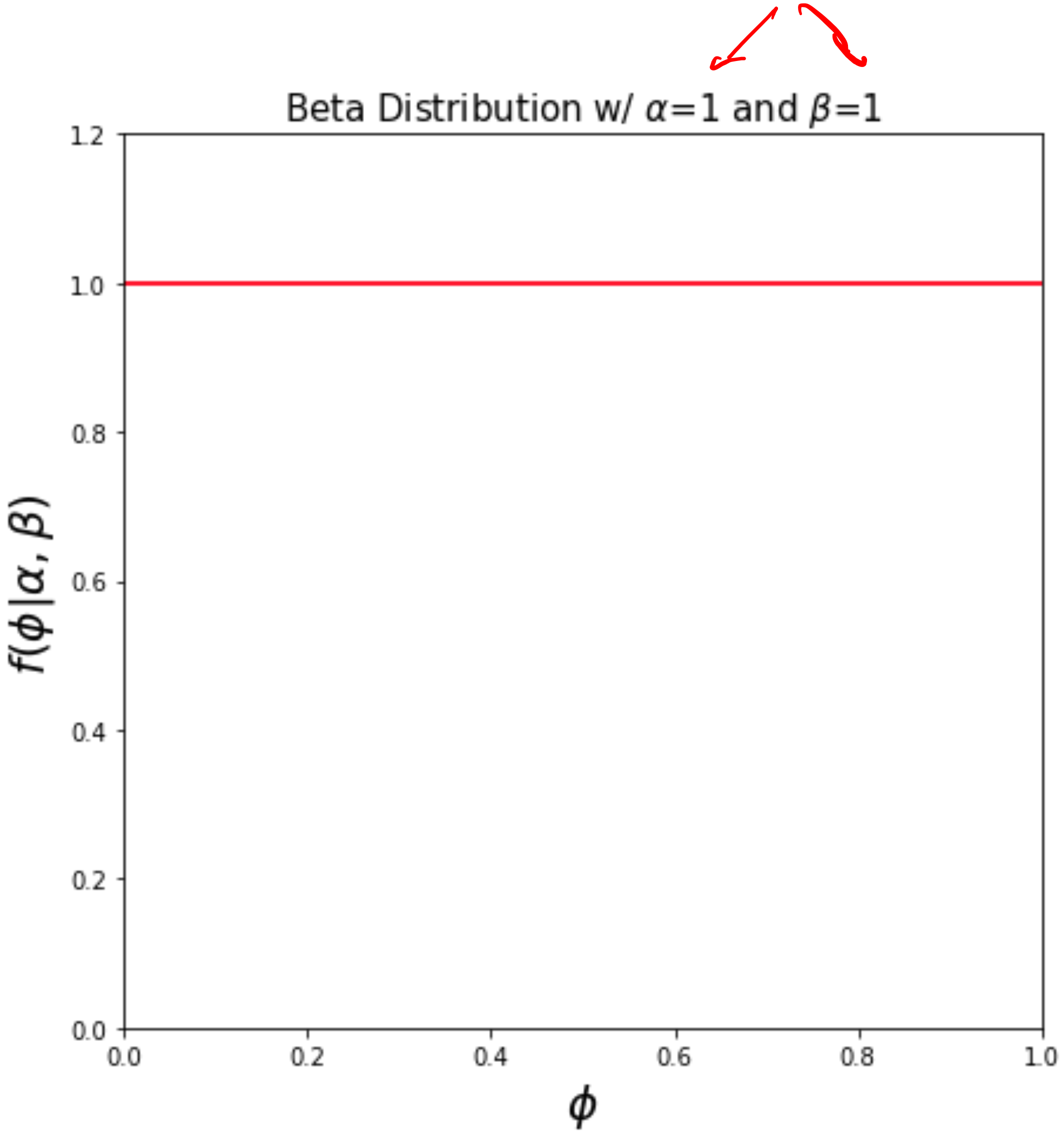
likelihood: $P(D|\Theta)$; prior: $P(\Theta)$

Coin Flipping MAP

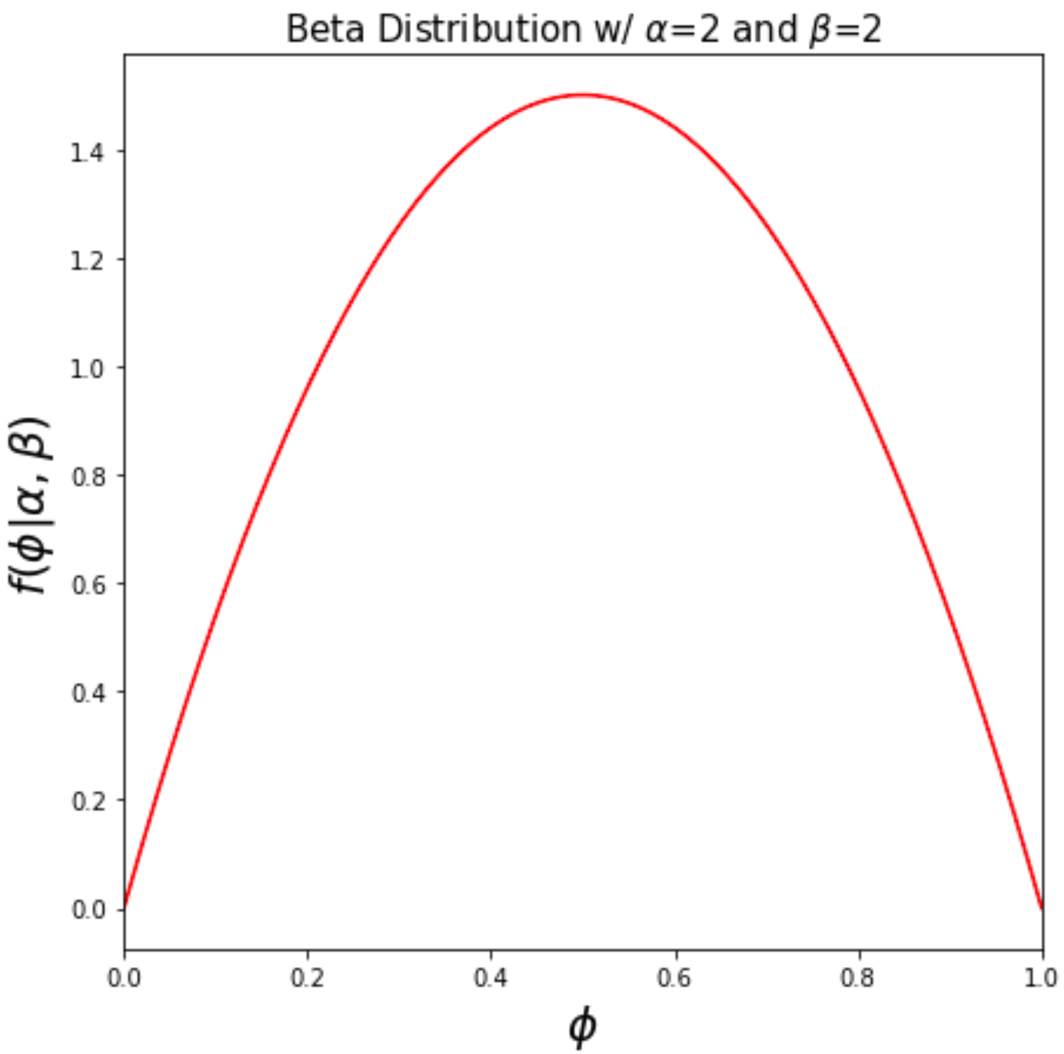
- A Bernoulli random variable takes value 1 (or heads) with probability ϕ and value 0 (or tails) with probability $1 - \phi$
- The pmf of the Bernoulli distribution is
$$p(x|\phi) = \phi^x(1-\phi)^{1-x}$$
- Assume a Beta prior over the parameter ϕ, which has pdf
$$f(\phi|\alpha,\beta) = \frac{\phi^{\alpha-1}(1-\phi)^{\beta-1}}{\mathrm{B}(\alpha,\beta)}$$

where $\mathrm{B}(\alpha,\beta) = \int_0^1 \phi^{\alpha-1}(1-\phi)^{\beta-1}d\phi$ is a normalizing constant to ensure the distribution integrates to 1

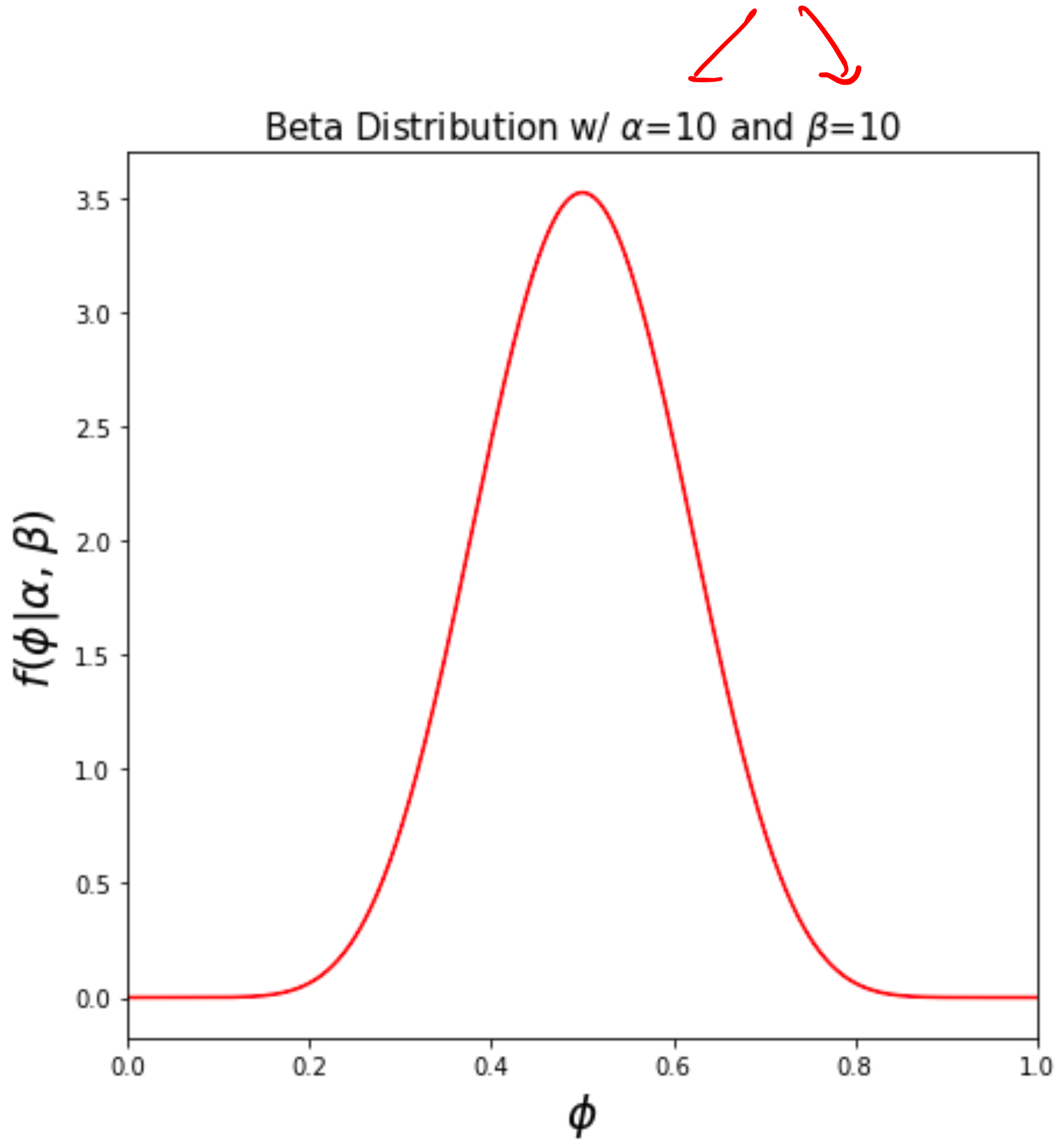
Beta Distribution



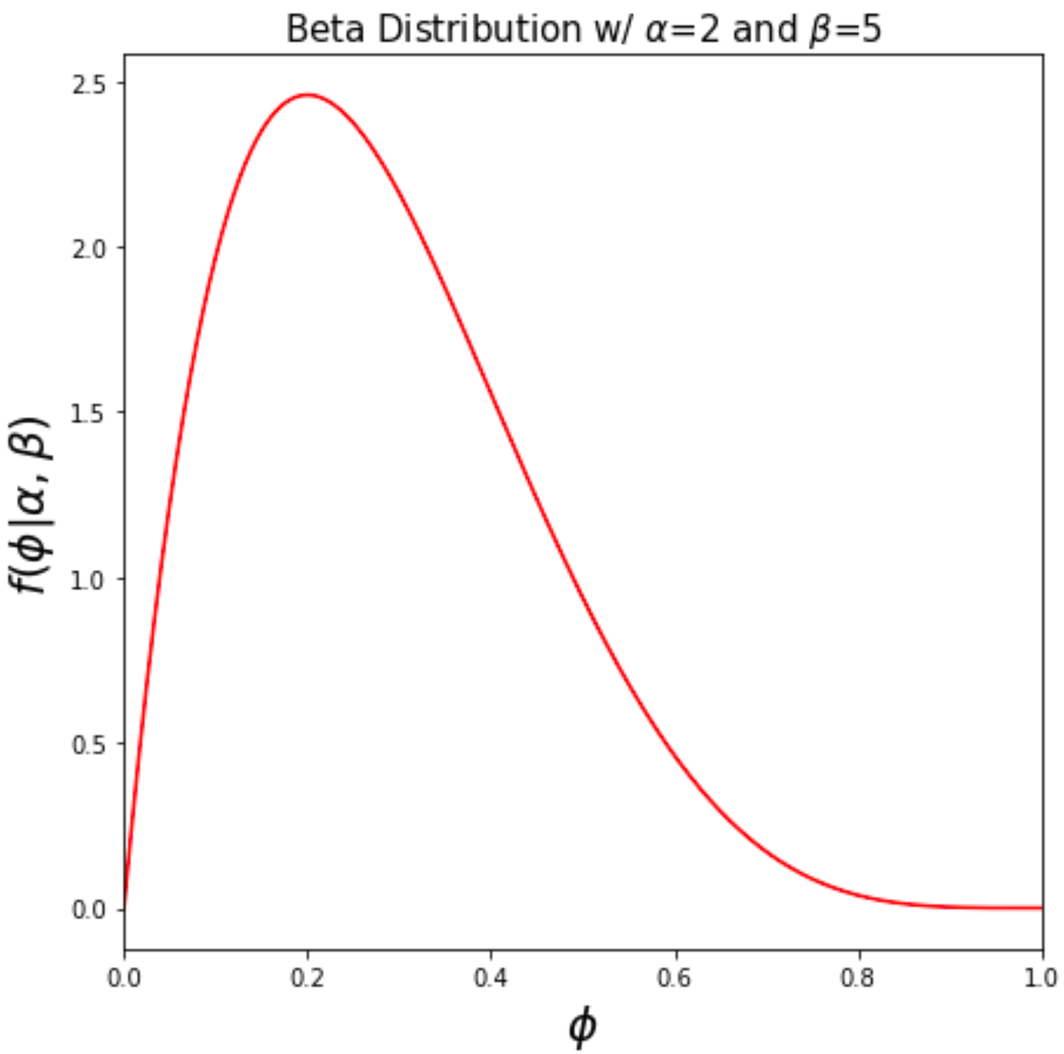
Beta Distribution



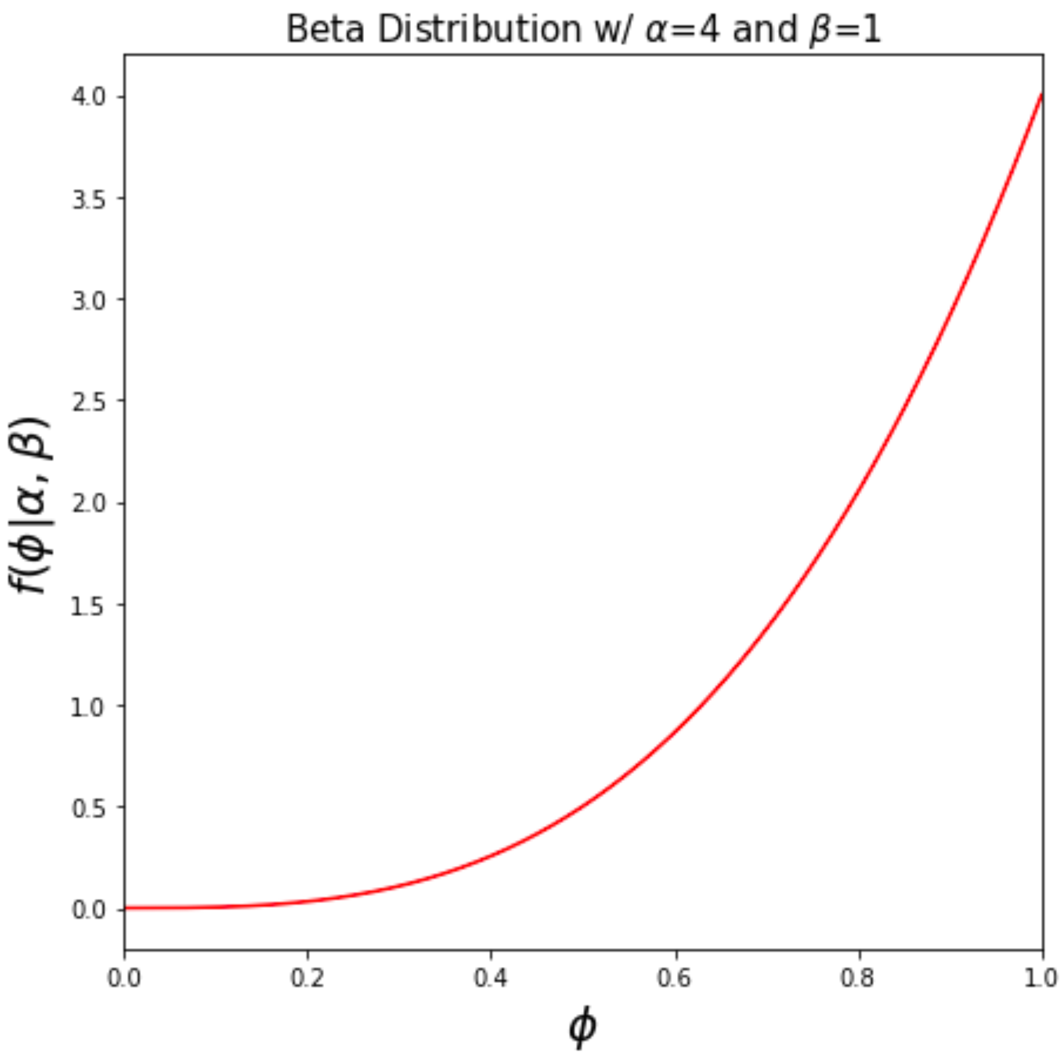
Beta Distribution



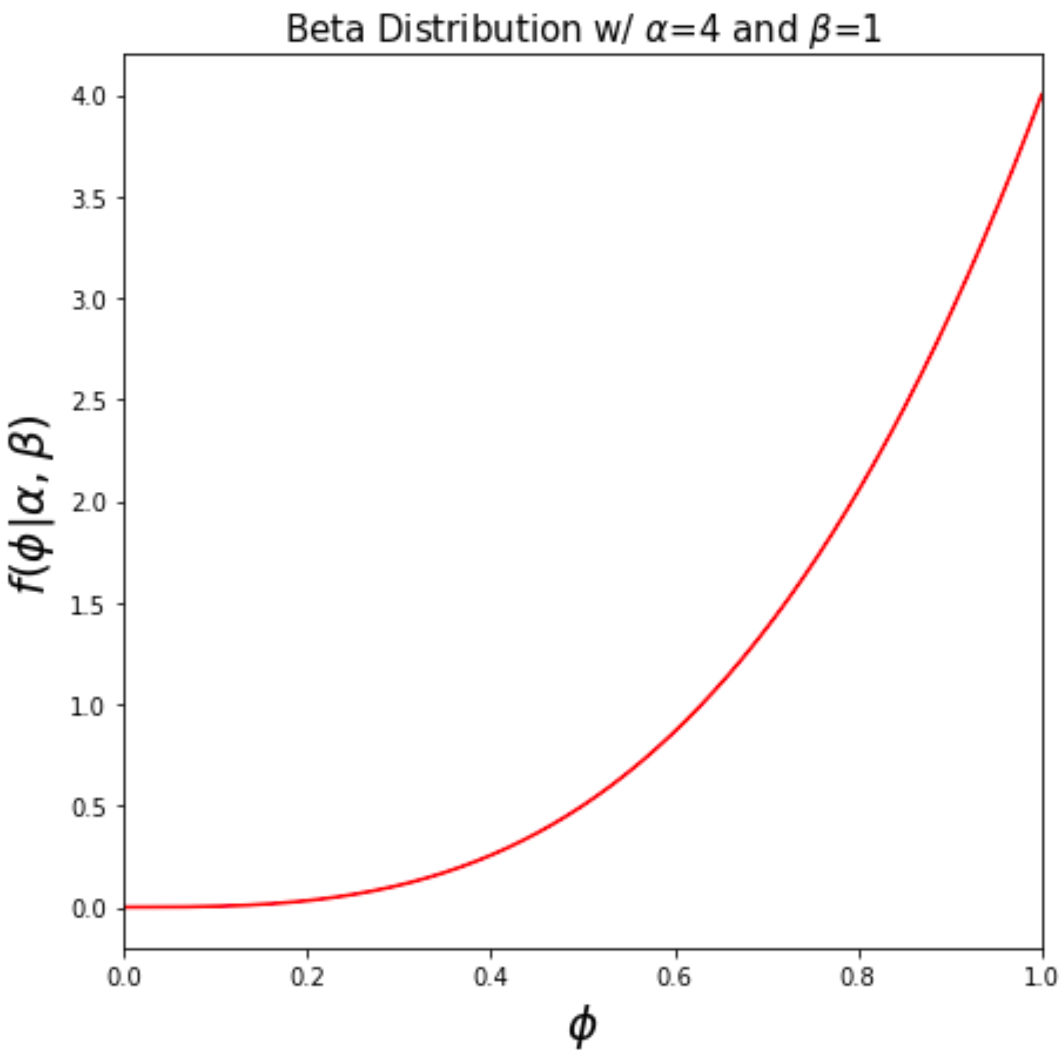
Beta Distribution



Beta Distribution



Okay, but why should we use this strange distribution as a prior?



Conjugate Priors

- For a given likelihood function $p(\mathcal{D}|\theta)$, a prior $p(\theta)$ is called a *conjugate prior* if the resulting posterior distribution $p(\theta|\mathcal{D})$ is in the same family as $p(\theta)$ i.e., $p(\theta|\mathcal{D})$ and $p(\theta)$ are the same type of random variable just with different parameters
  - We like conjugate priors because they are mathematically convenient
  - However, we do not **have** to use a conjugate prior if it doesn't align with our actual prior belief.

$$f(\phi|x,\alpha,\beta) = \frac{p(x|\phi)f(\phi|\alpha,\beta)}{p(x|\alpha,\beta)}$$

Example: Beta-Binomial Conjugacy

Example: Beta-Binomial Conjugacy

$$f(\phi|x,\alpha,\beta) = \frac{p(x|\phi)f(\phi|\alpha,\beta)}{p(x|\alpha,\beta)} = \frac{p(x|\phi)f(\phi|\alpha,\beta)}{\int p(x|\phi)f(\phi|\alpha,\beta)d\phi}$$

Beta-Binomial MAP

- Given N iid samples $\{x^{(1)}, \ldots, x^{(N)}\}$, the log-posterior is

$$\log(P(D|\theta)P(\theta)) = \log(P(D|\theta)) + \log(P(\theta))$$

$$= N_1 \log \phi + N_0 \log(1-\phi) + \log\left(\frac{\phi^{\alpha-1}(1-\phi)^{\beta-1}}{B(\alpha,\beta)}\right)$$

$$= N_1 \log \phi + N_0 \log(1-\phi) + (\alpha-1)\log\phi + (\beta-1)\log(1-\phi) - \log(B(\alpha,\beta))$$

Beta-Binomial MAP

- Given N iid samples $\{x^{(1)}, \dots, x^{(N)}\}$, the partial derivative of the log-posterior is

$$\frac{\partial}{\partial \phi}\left((N_1+\alpha-1)\log\phi + (N_0+\beta-1)\log(1-\phi) + \log(B(\alpha,\beta))\right)$$

$$\vdots$$

$$\phi_{MAP} = \frac{N_1+\alpha-1}{N_1+\alpha-1+N_0+\beta-1}$$

Coin Flipping MAP: Example

- Suppose $\mathcal{D}$ consists of ten 1's or heads ($N_1 = 10$) and two 0's or tails ($N_0 = 2$):

$$\phi_{MLE} = \frac{10}{10+2} = \frac{10}{12}$$

- Using a Beta prior with $\alpha = 2$ and $\beta = 5$, then

$\phi_{MAP} =$

Coin Flipping MAP: Example

- Suppose $\mathcal{D}$ consists of ten 1's or heads ($N_1 = 10$) and two 0's or tails ($N_0 = 2$):

$$\phi_{MLE} = \frac{10}{10+2} = \frac{10}{12}$$

- Using a Beta prior with $\alpha = 101$ and $\beta = 101$, then

Coin Flipping MAP: Example

- Suppose $\mathcal{D}$ consists of ten 1's or heads ($N_1 = 10$) and two 0's or tails ($N_0 = 2$):

$$\phi_{MLE} = \frac{10}{10+2} = \frac{10}{12}$$

- Using a Beta prior with $\alpha = 1$ and $\beta = 1$, then

$$\phi_{MAP} = \phi_{MLE} = \frac{N_1 + 1 - 1}{N_1 + 1 - 1 + N_0 + 1 - 1}$$

Key Takeaways

- Two ways of estimating the parameters of a probability distribution given samples of a random variable:
  - Maximum likelihood estimation – maximize the (log-)likelihood of the observations
  - Maximum a posteriori estimation – maximize the (log-)posterior of the parameters conditioned on the observations
    - Requires a prior distribution, drawn from background knowledge or domain expertise