Lecture #12: Logistic Regression and Gradient Descent

AM 207: Advanced Scientific Computing

Stochastic Methods for Data Analysis, Inference and Optimization

Fall, 2020

IACS

VE RI TAS

Outline

1. Logistic Regression
2. Gradient Descent
3. Convex Optimization

Logistic Regression

Coin-Toss Revisited: Modeling a Bernoulli Variable with Covariates

Let's revisit our model for coin-toss: we'd assumed that the outcomes $Y^{(n)}$ were independently and identically distributed as Bernoulli's, $Y^{(n)} \sim Ber(\theta)$. Today, we will re-examine the ***identical*** part of the modeling assumptions.

Realistically, the probability of $Y^{(n)} = 1$ depends on variables like force, angle, spin etc.

Let $\mathbf{X}^{(n)} \in \mathbb{R}^D$ be D number of such measurements of the n-th toss. We model the probability of $Y^{(n)} = 1$ as a function of these ***covariates*** $\mathbf{X}^{(n)}$:

$$Y^{(n)} \sim Ber\left(\mathrm{sigm}\left(f\left(\mathbf{X}^{(n)}; \mathbf{w}\right)\right)\right)$$

where $\mathbf{w}$ are the parameters of f and sigm is the sigmoid function $\mathrm{sigm}(z) = \frac{1}{1+e^{-z}}$.

Note: we need the sigmoid function to transform an arbitrary real number $f\left(\mathbf{X}^{(n)}; \mathbf{w}\right)$ into a probability (i.e. a number in $[0, 1]$).

The Logistic Regression Model

Given a set of N observations $(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})$. We assume the following model for the data.

$$Y^{(n)} \sim Ber\left(\text{sigm}\left(f\left(\mathbf{X}^{(n)}; \mathbf{w}\right)\right)\right).$$

This is called the ***logistic regression*** model.

Fitting this model on the data means ***inferring*** the parameters $\mathbf{w}$ that best aligns with the observations.

Once we have inferred the parameters $\mathbf{w}$, given a new set of covariates $\mathbf{x}^{\text{new}}$, we can **predict** the probability of $\mathbf{Y}^{\text{new}} = 1$ by computing

$$\text{sigm}\left(f\left(\mathbf{X}^{(n)}; \mathbf{w}\right)\right).$$

For now, we will assume that f is a linear function:

$$f\left(\mathbf{X}^{(n)}; \mathbf{w}\right) = \mathbf{w}^\top \mathbf{X}^{(n)}.$$

Interpreting a Logistic Regression Model

Suppose that you fit a logistic regression model to predict whether a loan application should be approved. Suppose that you have three covariates:

1. `x_1` representing gender: 0 for male, 1 for female
2. `x_2` for the income
3. `x_3` for the loan amount

Suppose that the parameters you found are:

$$p(y = 1|x_1, x_2, x_3) = \text{sigm}(-1 + 3x_1 + 1.5x_2 + 1.75x_3).$$

What are the parameters telling us about the most influential attribute for predicting loan approval? What does this say about our data?

Maximizing the Logistic Regression Log-likelihood

Given a set of N observations $(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})$. We want to find $\mathbf{w}_{\text{MLE}}$ that maximizes the log (joint) likelihood:

$$\begin{aligned}
\mathbf{w}_{\text{MLE}} =& \underset{\mathbf{w}}{\operatorname{argmax}}\ \ell(\mathbf{w}) \equiv \underset{\mathbf{w}}{\operatorname{argmin}}\ -\ell(\mathbf{w}) = \underset{\mathbf{w}}{\operatorname{argmin}} - \log \prod_{n=1}^{N} p(y^{(n)}|\mathbf{x}^{(n)}) \\
=& \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{n=1}^{N} -\log\left(\operatorname{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)})^{y^{(n)}} (1 - \operatorname{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)}))^{1-y^{(n)}} \right) \\
=& \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{n=1}^{N} -y^{(n)} \ \log \ \operatorname{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)}) \\
& - (1 - y^{(n)}) \log(1 - \operatorname{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)}))
\end{aligned}$$

Optimizing the likelihood requires us to find the stationary points of the gradient of $\ell(\mathbf{w})$:

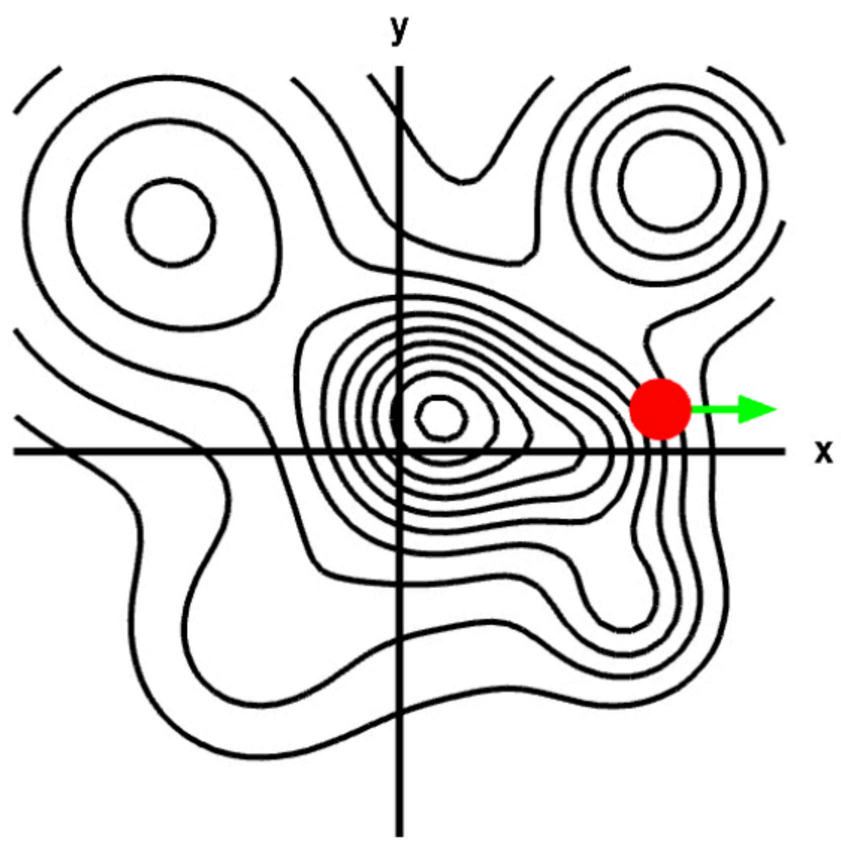
$$\nabla_{\mathbf{w}} \ell(\mathbf{w}) = - \sum_{n=1}^{N} \left(y^{(n)} - \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}^{(n)}}} \right) \mathbf{x}^{(n)} = \mathbf{0}$$

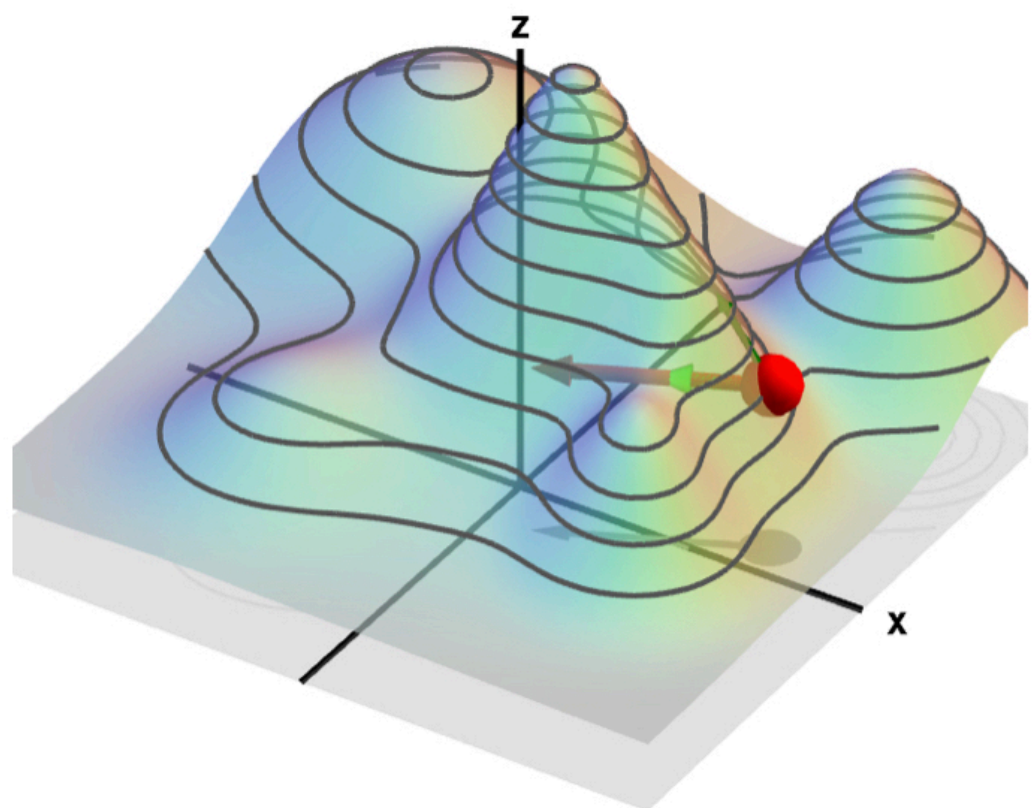
Can we solve for $\mathbf{w}$?

Gradient Descent

Gradient as Directional Information

The gradient is orthogonal to the level curve of f at x^* and hence, *when it is not zero*, points in the direction of the greatest instantaneous increase in f.





An Intuition for Gradient Descent

The intuition behind various flavours of gradient descent is as follows:

Gradient Descent: the Algorithm

1. start at random place: $\mathbf{w}^{(0)} \leftarrow \mathbf{random}$
2. until (stopping condition satisfied):

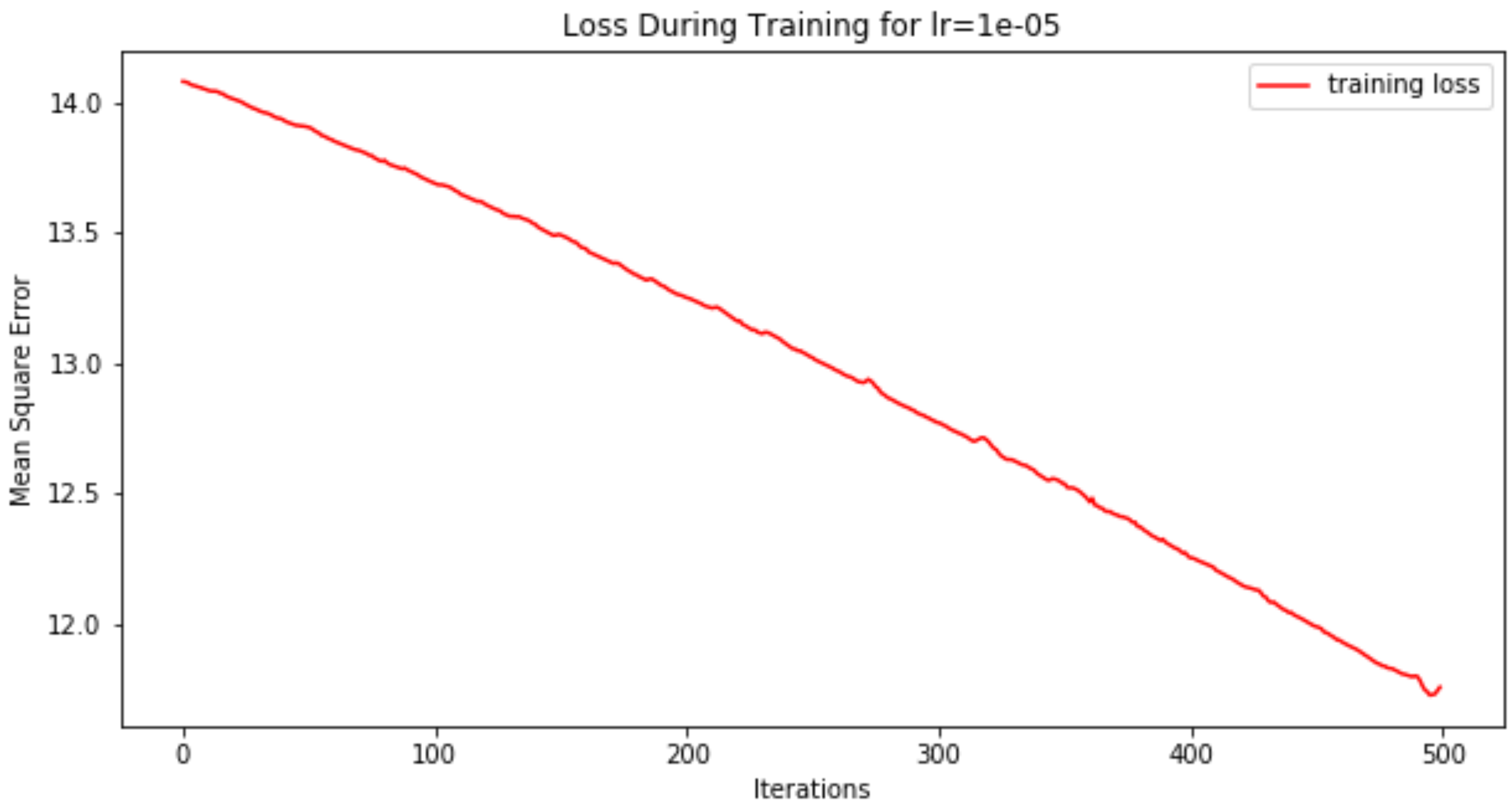
   a. compute gradient at $\mathbf{w}^{(t)}$: gradient ($\mathbf{w}^{(t)}$) = $\nabla_{\mathbf{w}}$ loss_function ($\mathbf{w}^{(t)}$)

   b. take a step in the negative gradient direction: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)}$ - η * gradient ($\mathbf{w}^{(t)}$)

Here η is called the ***learning rate***.

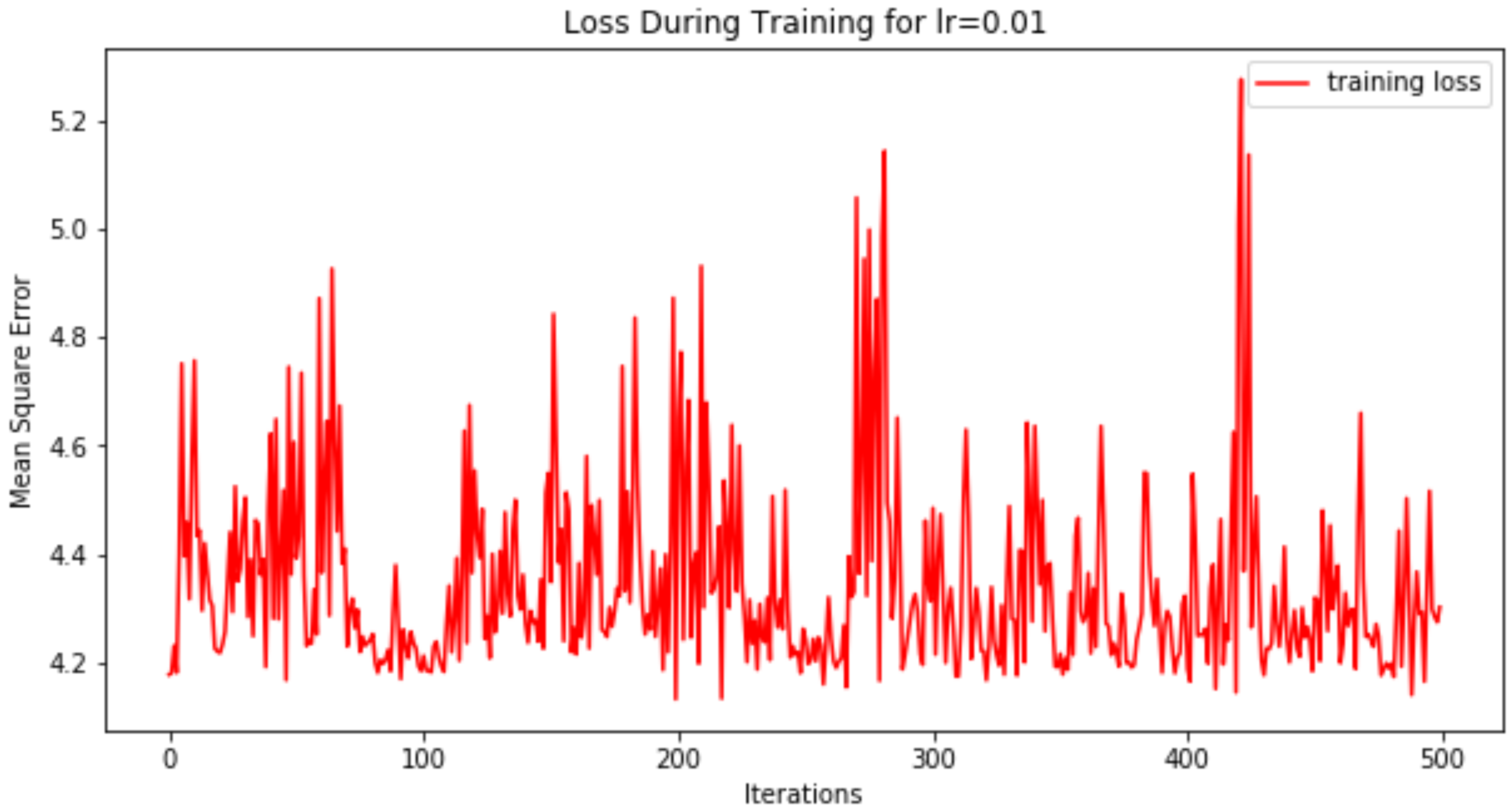
Diagnosing Design Choices with the Trajectory

If this is your objective function during training, what can you conclude about your step-size?

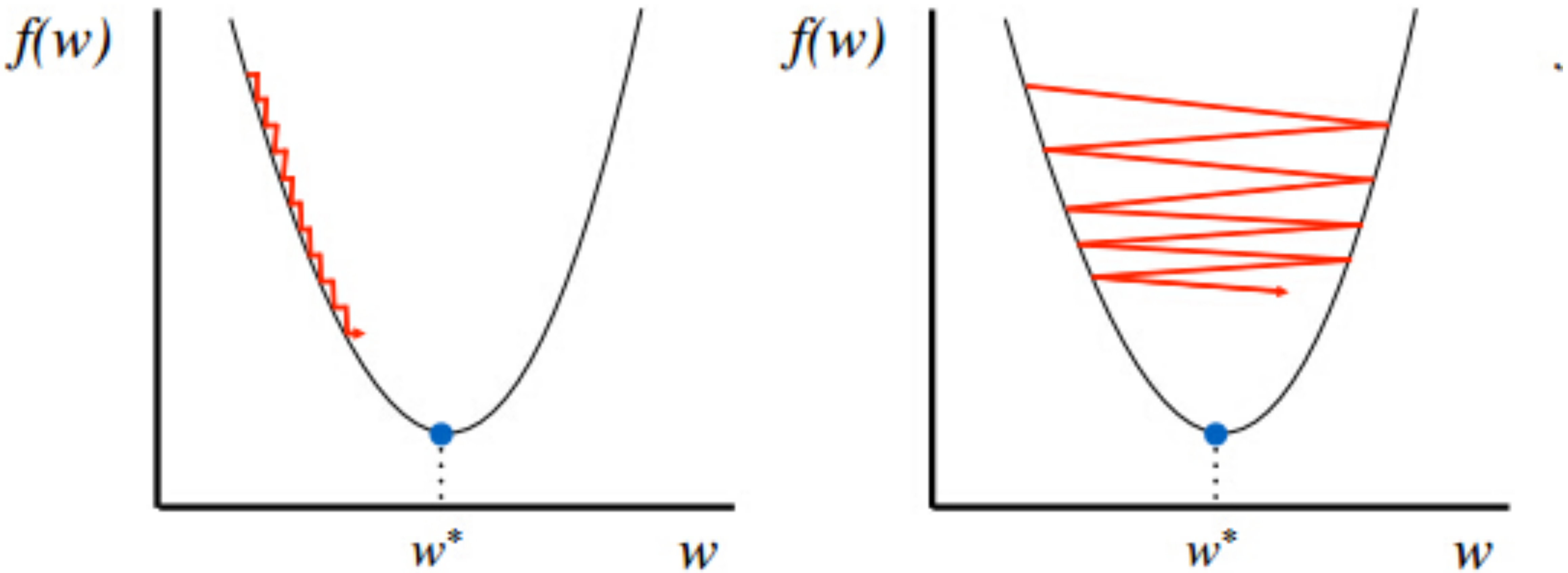


Diagnosing Issues with the Trajectory

If this is your objective function during training, what can you conclude about your step-size?



Gradient Descent: Step Size Matters

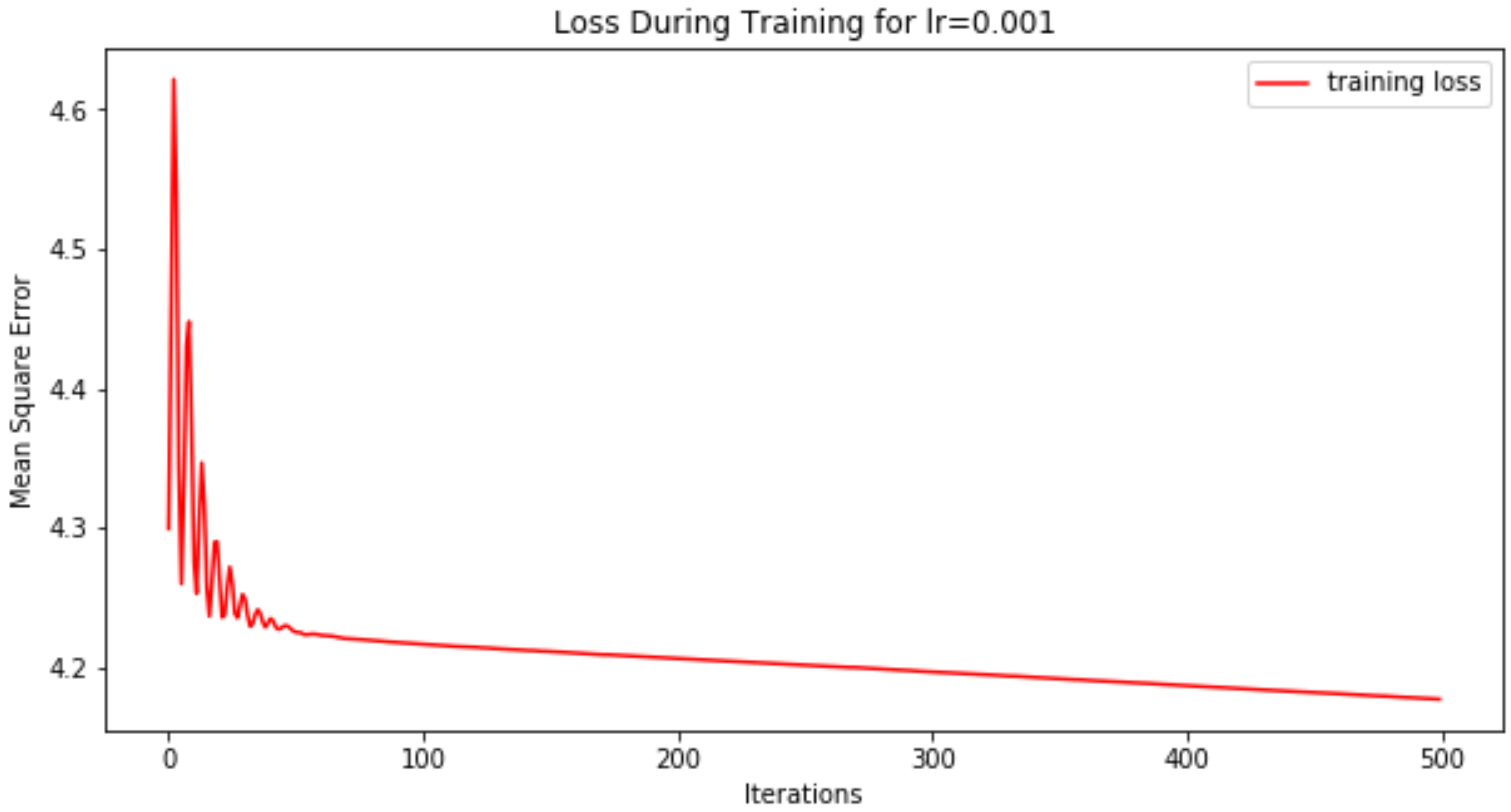


Too small: converge very slowly

Too big: overshoot and even diverge

Diagnosing Issues with the Trajectory

If this is your objective function during training, what can you conclude about your step-size?



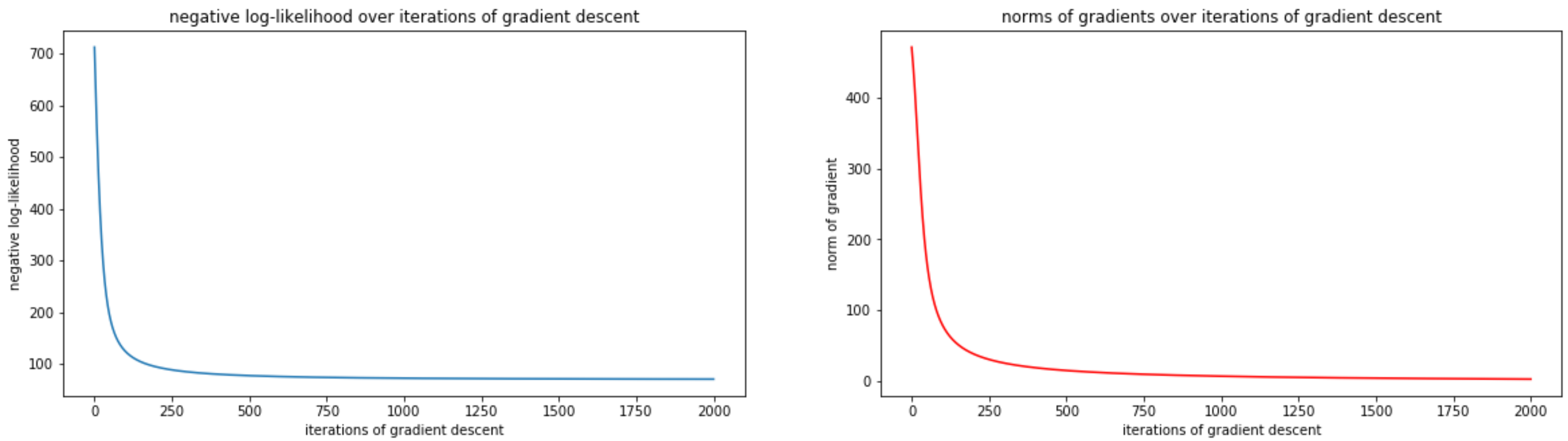
Gradient Descent for Logistic Regression

When diagnosing our gradient descent learning, we can:

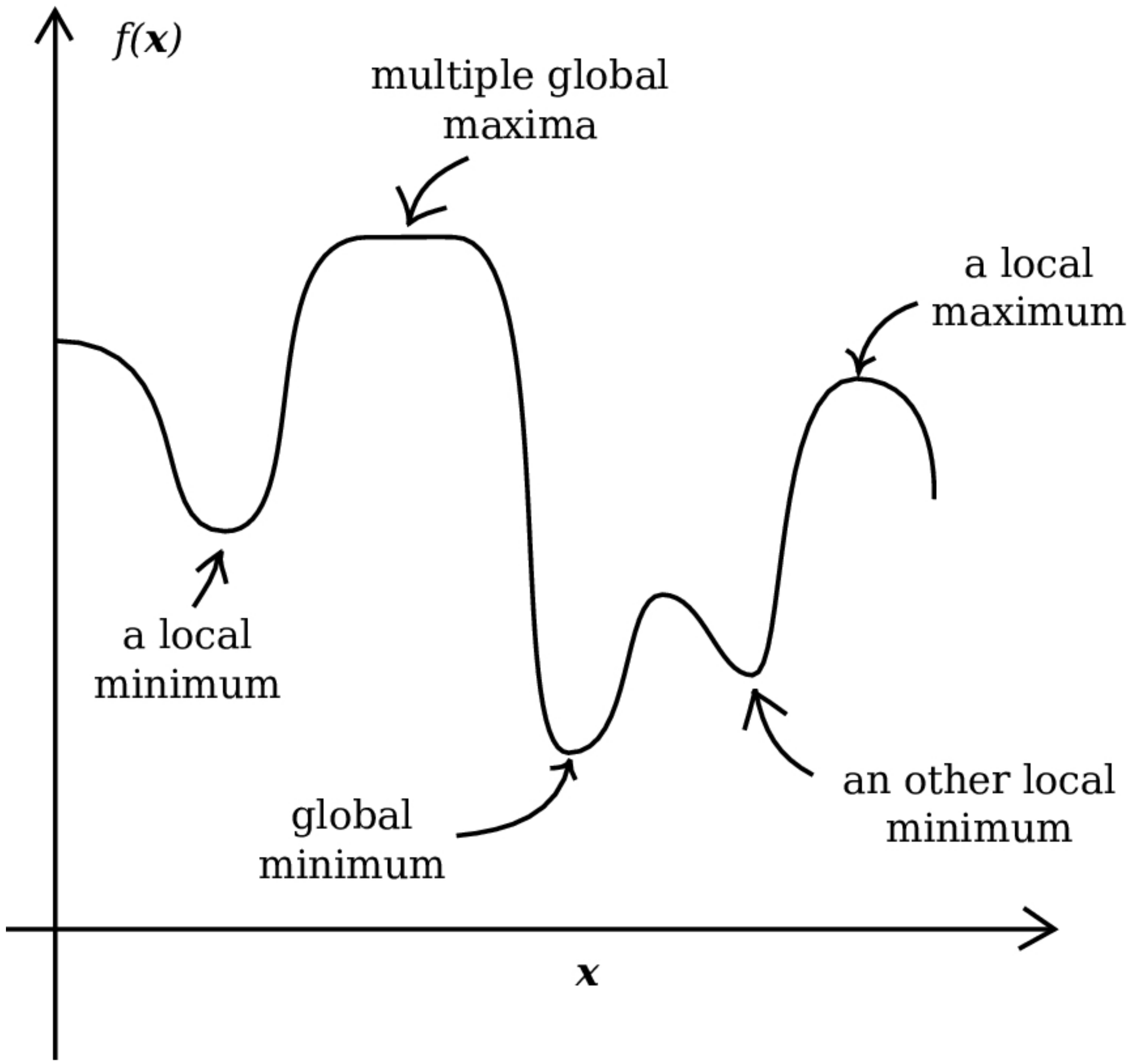
1. Visualize the log-likelihood. **What does this tell us?**
2. Visualize the norm of the gradients. **What does this tell us?**

What else should we visualize to check that our learned model aligns with the data?

In [3]:
```
fig, ax = plt.subplots(1, 2, figsize=(20, 5))
ax = plot_diagnostics(ax, nlls, grad_norms)
plt.show()
```



But Did We Optimize It?

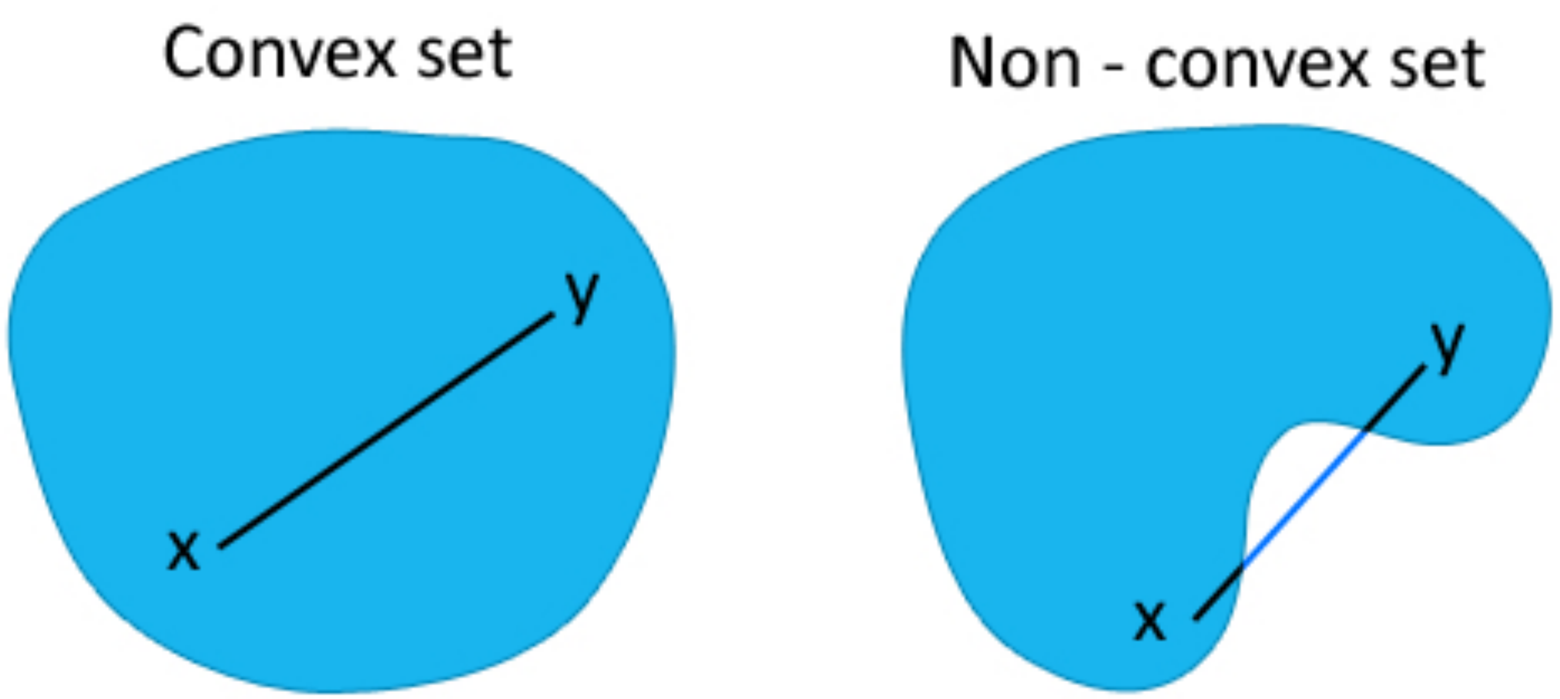


Convex Optimization

Convex Sets

A ***convex set*** $S \subset \mathbb{R}^D$ is a set that contains the line segment between any two points in S. Formally, if $x, y \in S$ then S contains all convex combinations of x and y:

$$tx + (1-t)y \in S, \qquad t \in [0, 1].$$

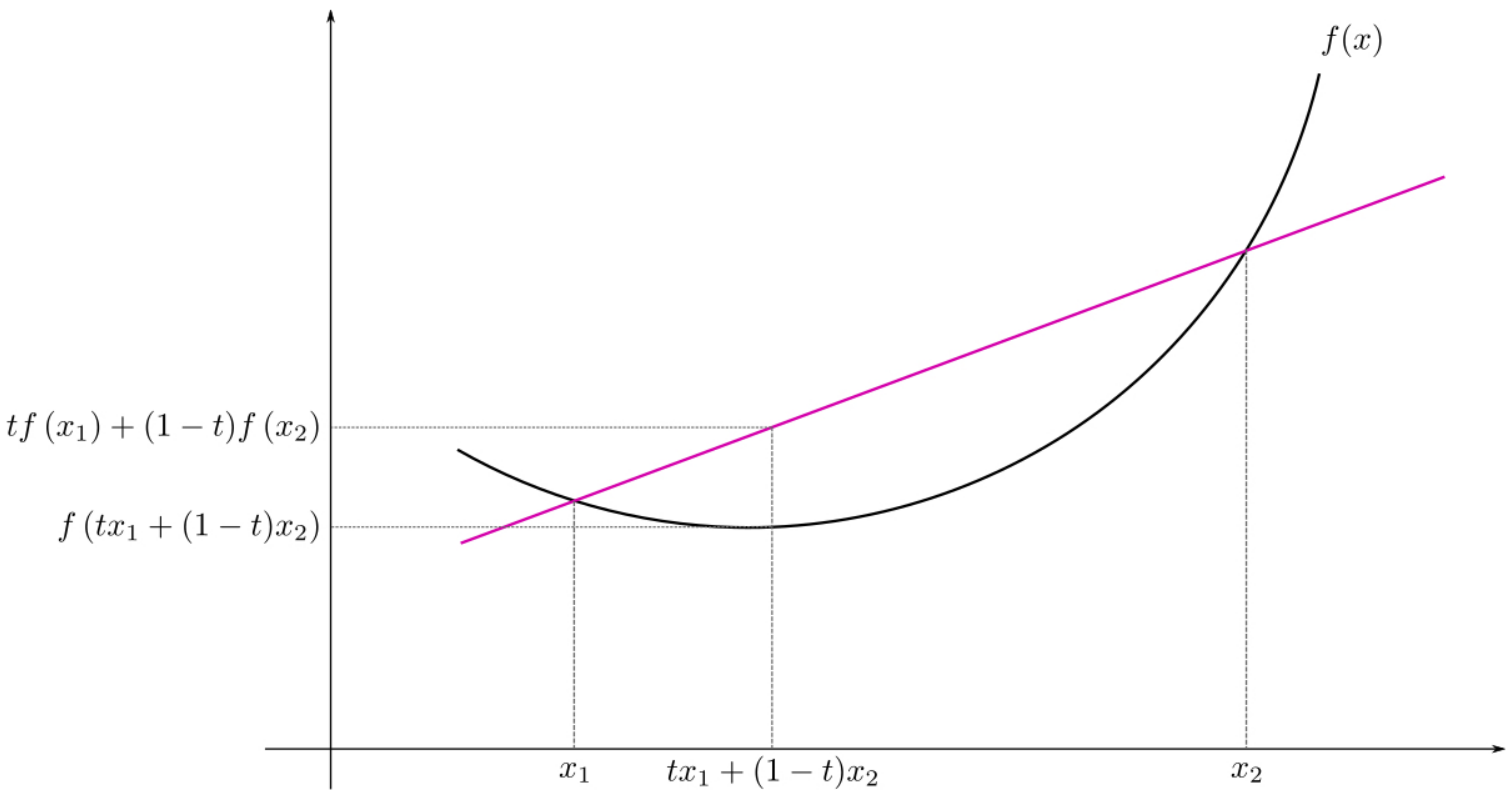


Convex Functions

A function f is a ***convex function*** if domain of f is a convex set, and the line segment between the points $(x,\ f(x))$ and $(y,\ f(y))$ lie above the graph of f. Formally, for any $x,\ y \in \mathrm{dom}(f)$, we have

$$\underbrace{f(tx + (1-t)y)}_{\text{height of graph of } f \text{ at a point between } x \text{ and } y} \leq \underbrace{tf(x) + (1-t)f(y)}_{\text{height of point on line segment between } (x, f(x)) \text{ and } (y, f(y))},$$

$$t \in [0, 1]$$

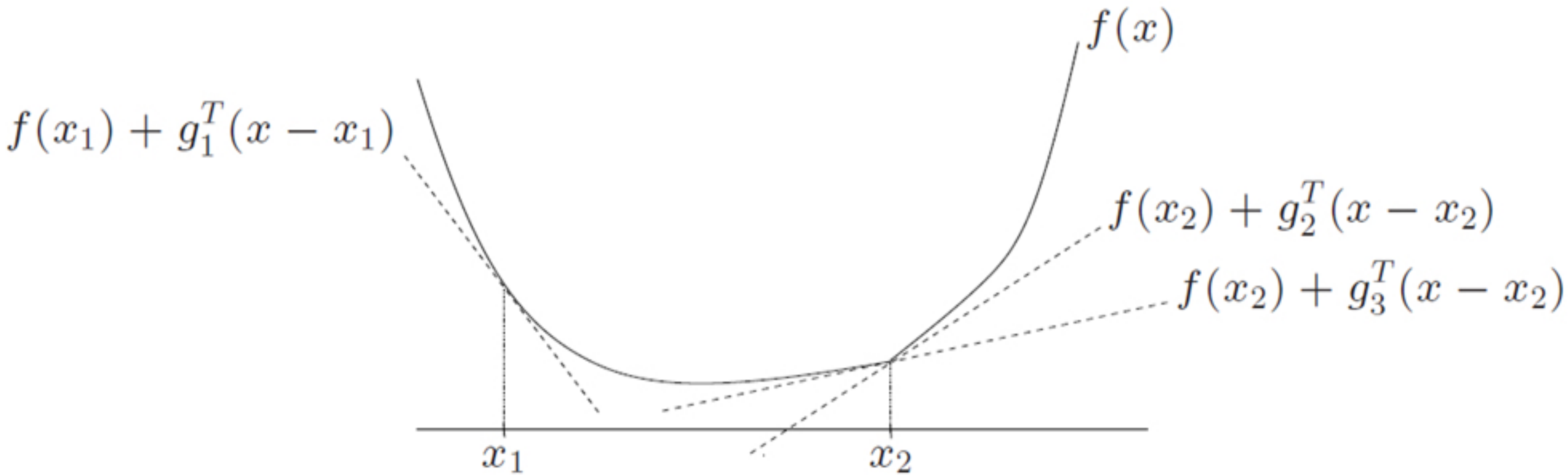


Convex Function: First Order Condition

How do we check that a function f is convex? If f is differentiable then f is convex if the graph of f lies above every tangent plane.

Theorem: If f is differentiable then f is convex if and only if for every $x \in \mathrm{dom}(f)$, we have

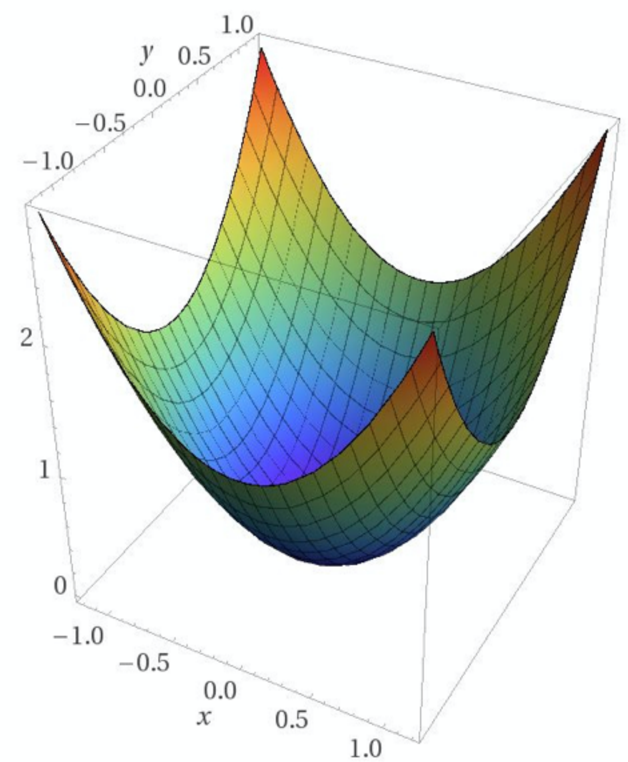
$$\underbrace{f(y)}_{\text{height of graph of } f \text{ over } y} \geq \underbrace{f(x) + \nabla f(x)^\top (y - x)}_{\text{height of plane tangent to } f \text{ at } x\text{, evaluated over } y}, \quad \forall y \in \mathrm{dom}(f)$$

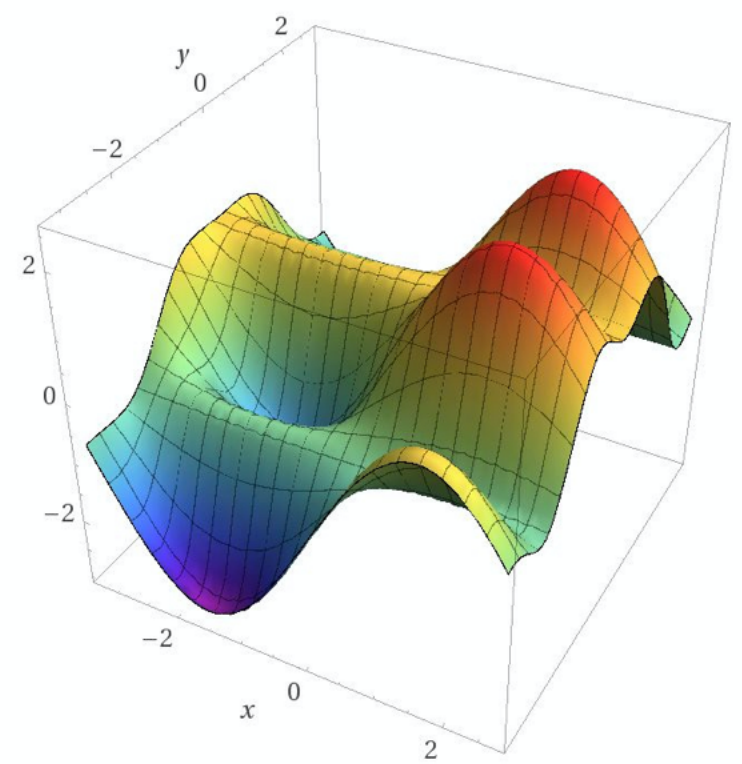


Convex Function: Second Order Condition

If f is twice-differentiable then f is convex if the "second derivative is positive".

Theorem: If f is twice-differentiable then f is convex if and only if the Hessian $\nabla^2 f(x)$ is positive semi-definite for every $x \in \mathrm{dom}(f)$.





Properties of Convex Functions

How to build complex convex functions from simple convex functions:

1. if $w_1, w_2 \geq 0$ and f_1, f_2 are convex, then $h = w_1 f_1 + w_2 f_2$ is convex

2. if f and g are convex, and g is univariate and non-decreasing then $h = g \circ f$ is convex

3. Log-sum-exp functions are convex: $f(x) = \log \sum_{k=1}^{K} e^x$

Note: there are many other convexity preserving operations on functions.

Convex Optimization

A ***convex optimization problem*** is an optimization of the following form:

$$
\begin{aligned}
\min \quad & f(x) && \text{(convex objective function)} \\
\text{subject to} \quad & h_i(x) \leq 0, i = 1, \ldots, i && \text{(convex inequality constraints)} \\
& a_j^\top x - b_j = 0, j = 1, \ldots, J && \text{(affine equality constraints)}
\end{aligned}
$$

The set of points that satisfy the constraints is called the ***feasible set***.

You can prove that the a convex optimization problem optimizes a convex objective function over a convex feasible set. But why should we care about convex optimization problems?

Theorem: Let f be a convex function defined over a convex feasible set Ω. Then if f has a local minimum at $x \in \Omega$ -- $f(y) \geq f(x)$ for y in a small neighbourhood of x -- then f has a global minimum at x.

Corollary: Let f be a differentiable convex function:

1. if f is unconstrained, then f has a **local minimum** and hence **global minimum** at x if $\nabla f(x) = 0$.
2. if f is constrained by equalities, then f has a global minimum at x if $\nabla J(x, \lambda) = 0$, where $J(x, \lambda)$ is the Lagrangian of the constrained optimization problem.

Convexity of the Logistic Regression Negative Log-Likelihood

But why do we care about convex optimization problems? Let's connect the theory of convex optimization to MLE inference for logistic regression. Recall that the negative log-likelihood of the logistic regression model is

$$
\begin{aligned}
-\ell(\mathbf{w}) &= -\sum_{n=1}^{N} y^{(n)}\ \log\ \mathrm{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)}) + (1 - y^{(n)}) \log(1 - \mathrm{sigm}(\mathbf{w}^\top \mathbf{x}^{(n)})) \\
&= \sum_{n=1}^{N} y^{(n)} \log(e^0 + e^{\mathbf{w}^\top \mathbf{x}^{(n)}}) + (1 - y^{(n)})(-\mathbf{w}^\top \mathbf{x}^{(n)})
\end{aligned}
$$

Proposition: The negative log-likelihood of logistic regression $-\ell(\mathbf{w})$ is convex.

What does this mean for gradient descent? If gradient descent finds that $\mathbf{w}^*$ is a stationary point of $-\nabla_{\mathbf{w}} \ell(\mathbf{w})$ then $-\ell(\mathbf{w})$ has a global minimum at $\mathbf{w}^*$. Hence, $\ell(\mathbf{w})$ is maximized at $\mathbf{w}^*$.

Proof of the Proposition: Note that

1. $-\mathbf{w}^\top \mathbf{x}^{(n)}$ and $(1 - y^{(n)})(-\mathbf{w}^\top \mathbf{x}^{(n)})$ are convex, since they are linear
2. $\log(e^0 + e^{\mathbf{w}^\top \mathbf{x}^{(n)}})$ is convex since it is the composition of a log-sum-exp function (which is convex) and a convex function $\mathbf{w}^\top \mathbf{x}^{(n)}$
3. $\sum_{n=1}^{N} y^{(n)} \log(e^0 + e^{\mathbf{w}^\top \mathbf{x}^{(n)}})$ is convex since it is a nonnegative linear combination of convex functions
4. $-\ell(\mathbf{w})$ is convex since it is the sum of two convex functions

But Does It Scale?

Gradient is such a simple algorithm that can be applied to **any optimization problem** for which you can compute the gradient of the objective function.

Question: Does this mean that maximum likelihood inference for statistical models is now an easy task (i.e. just use gradient descent)?

For every likelihood optimization problem, evaluating the gradient at a set of parameters $\mathbf{w}$ requires evaluating the likelihood of the entire dataset using $\mathbf{w}$:

$$\nabla_{\mathbf{w}} \ell(\mathbf{w}) = -\sum_{n=1}^{N} \left(y^{(n)} - \frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}^{(n)}}} \right) \mathbf{x}^{(n)} = \mathbf{0}$$

Imagine if the size of your dataset N is in the millions. Naively evaluating the gardient **just once** may take up to seconds or minutes, thus running gradient descent until convergence may be unachievable in practice!

Idea: Maybe we don't need to use the entire data set to evaluate the gradient during each step of gradient descent. Maybe we can approximate the gradient at $\mathbf{w}$ well enough with just a subset of the data.