STOCHASTIC GRADIENT DESCENT

GOALS FOR TODAY

1. Derive the gradients for the classification loss functions
2. Go over gradient descent
3. Choosing the optimal step size
4. Stochastic gradient descent

CLASSIFICATION

LOSS FUNCTION

Want to maximize the probability of all data being correctly classified

$$\Pr(\hat{Y} = y_1 \,|\, X = x_1) \Pr(\hat{Y} = y_2 \,|\, X = x_2) \ldots \Pr(\hat{Y} = y_m \,|\, X = x_m) = \prod_{i=1}^{m} \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

The product of all probabilities is in (0,1)

Maximum is reached when all probabilities are 1

$$l(w) \doteq - \prod_{i=1}^{m} \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

CLASSIFICATION

LOSS FUNCTION

Minimize a loss function:

$$l(w) \doteq -\prod_{i=1}^{m} \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

Minimize the negative product of probabilities

Equivalent to maximizing product of probabilities

CLASSIFICATION

LOSS FUNCTION

$$\nabla l(w) = \underbrace{-\left(\prod_{j=1}^{m} \Pr(\hat{Y} = y_j \,|\, X = x_j)\right)}_{(a)} \sum_{i=1}^{m} \frac{\partial}{\partial w} \ln \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

For any w, $(a) > 0$. We can interpret this as rescaling the gradient, but not changing the direction, i.e., it does not really matter.

CLASSIFICATION

LOSS FUNCTION

Could instead define a different loss function l' that measure the log-likelihood

$$l'(w) \doteq -\ln \prod_{i=1}^{m} \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

This is called the **negative log-likelihood** (NLL) and is a common loss function in machine learning

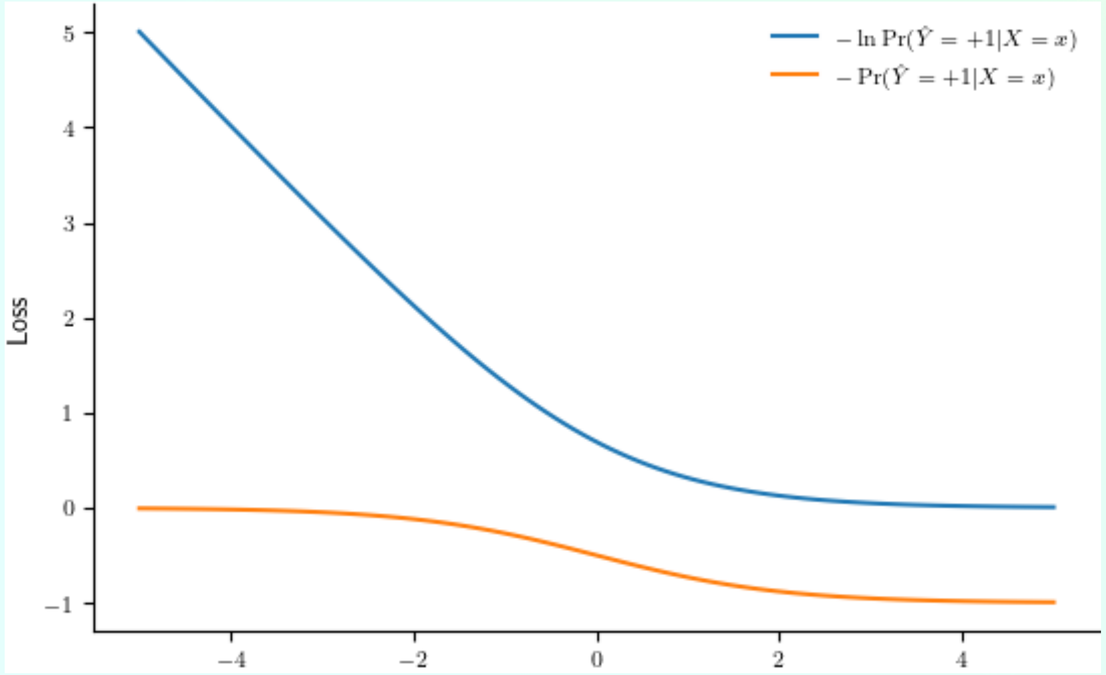
$$\nabla l'(w) = -\sum_{i=1}^{m} \frac{\partial}{\partial w} \ln \Pr(\hat{Y} = y_i \,|\, X = x_i)$$

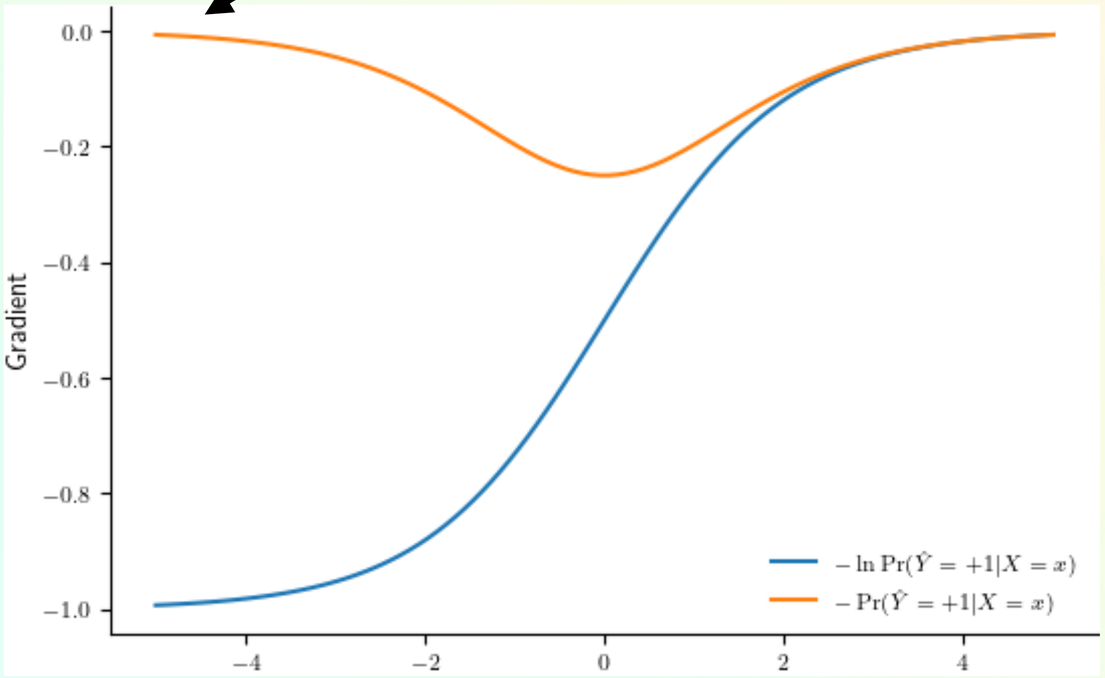
CLASSIFICATION

LOSS FUNCTION

A flat gradient (near zero) means parameters should not change much.

But predictions were really bad and we want them to change a lot





CLASSIFICATION

LOSS FUNCTION

$$\Pr(\hat{Y} = +1 \,|\, X = x) = f(x, w)$$

Expression for loss function that we can differentiate

$$\Pr(\hat{Y} = y_i \,|\, X = x_i) = y_i f(x_i, w) + (1 - y_i)\Big(1 - f(x_i, w)\Big)$$

Similarly, we have

$$\ln \Pr(\hat{Y} = y_i \,|\, X = x_i) = y_i \ln f(x_i, w) + (1 - y_i) \ln \Big(1 - f(x_i, w)\Big)$$

CLASSIFICATION

LOSS FUNCTION

Take-home questions:

1. $f(x, w) = \frac{1}{1 + e^{-w^\top x}}$, what is $\frac{\partial f(x, w)}{\partial w}$?
2. What is $\nabla l(w)$ for the NLL loss?

CLASSIFICATION

LOSS FUNCTION

$$\frac{\partial f(x,w)}{\partial w} = \frac{\partial}{\partial w}\frac{1}{1+e^{-w^\top x}} = -\frac{\frac{\partial 1}{\partial w}\left(1+e^{-w^\top x}\right) - \frac{\partial\left(1+e^{-w^\top x}\right)}{\partial w}}{\left(1+e^{-w^\top x}\right)^2} = -\frac{1}{\left(1+e^{-w^\top x}\right)^2}\frac{\partial e^{-w^\top x}}{\partial w}$$

$$= -\frac{1}{\left(1+e^{-w^\top x}\right)}\frac{1}{\left(1+e^{-w^\top x}\right)}\frac{\partial e^{-w^\top x}}{\partial w} = -f(x,w)^2\frac{\partial e^{-w^\top x}}{\partial w}$$

$$= -f(x,w)^2 e^{-w^\top x}\frac{\partial(-w^\top x)}{\partial w} = f(x,w)^2 e^{-w^\top x}\frac{\partial w^\top x}{\partial w} = f(x,w)^2 e^{-w^\top x} x$$

CLASSIFICATION

LOSS FUNCTION

NLL loss function for classification with $y_i \in \{0,1\}$

$$l(w) = -\sum_{i=1}^{m} y_i \ln\left(f(x_i, w)\right) + (1 - y_i)\ln\left(1 - f(x_i, w)\right)$$

$$\frac{\partial}{\partial w} \ln f(x, w) = ?$$

CLASSIFICATION

LOSS FUNCTION

$$\frac{\partial}{\partial w}\ln f(x,w) = \frac{1}{f(x,w)}\frac{\partial f(x,w)}{\partial w}$$

$$= \frac{1}{f(x,w)} f(x,w)^2 e^{-w^\top x} x = f(x,w) e^{-w^\top x} x$$

$$= \frac{1}{1+e^{-w^\top x}} e^{-w^\top x} x = \frac{e^{-w^\top x}}{\frac{e^{-w^\top x}}{e^{-w^\top x}} + e^{-w^\top x}} x$$

$$= \frac{1}{\frac{1}{e^{-w^\top x}} + 1} x = \frac{1}{e^{w^\top x} + 1} x = (1 - f(x,w))x$$

CLASSIFICATION

LOSS FUNCTION

$$\frac{\partial}{\partial w}\ln\left(1-f(x,w)\right) = \frac{1}{\left(1-f(x,w)\right)}\frac{\partial\left(1-f(x,w)\right)}{\partial w}$$

$$= -\frac{1}{\left(1-f(x,w)\right)}\frac{\partial f(x,w)}{\partial w} = -\frac{1}{\left(\frac{f(x,w)}{f(x,w)}-f(x,w)\right)}f(x,w)^2e^{-w^\top x}x$$

$$= -\frac{1}{f(x,w)\left(\frac{1}{f(x,w)}-1\right)}f(x,w)^2e^{-w^\top x}x = -\frac{1}{\left(\frac{1}{f(x,w)}-1\right)}f(x,w)e^{-w^\top x}x$$

$$= -\frac{1}{\left(\frac{1+e^{-w^\top x}}{1}-1\right)}f(x,w)e^{-w^\top x}x = -\frac{1}{e^{-w^\top x}}f(x,w)e^{-w^\top x}x = -f(x,w)x$$

CLASSIFICATION

LOSS FUNCTION

$$
\begin{aligned}
\nabla l(w) = & -\sum_{i=1}^{m} y_i \frac{\partial}{\partial w} \ln\left(f(x_i, w)\right) + (1 - y_i) \frac{\partial}{\partial w} \ln\left(1 - f(x_i, w)\right) \\
= & -\sum_{i=1}^{m} y_i (1 - f(x_i, w)) x_i - (1 - y_i) f(x_i, w) x_i \\
= & -\sum_{i=1}^{m} \left(y_i (1 - f(x_i, w)) - (1 - y_i) f(x_i, w)\right) x_i \\
= & -\sum_{i=1}^{m} \left(y_i - f(x_i, w)\right) x_i
\end{aligned}
$$

GRADIENT DESCENT

PROCESS

Initialize some w^0 (with linear function approximation $w^0 = 0$)

Do

$$w^{k+1} = w^k - \eta \nabla l(w^k)$$

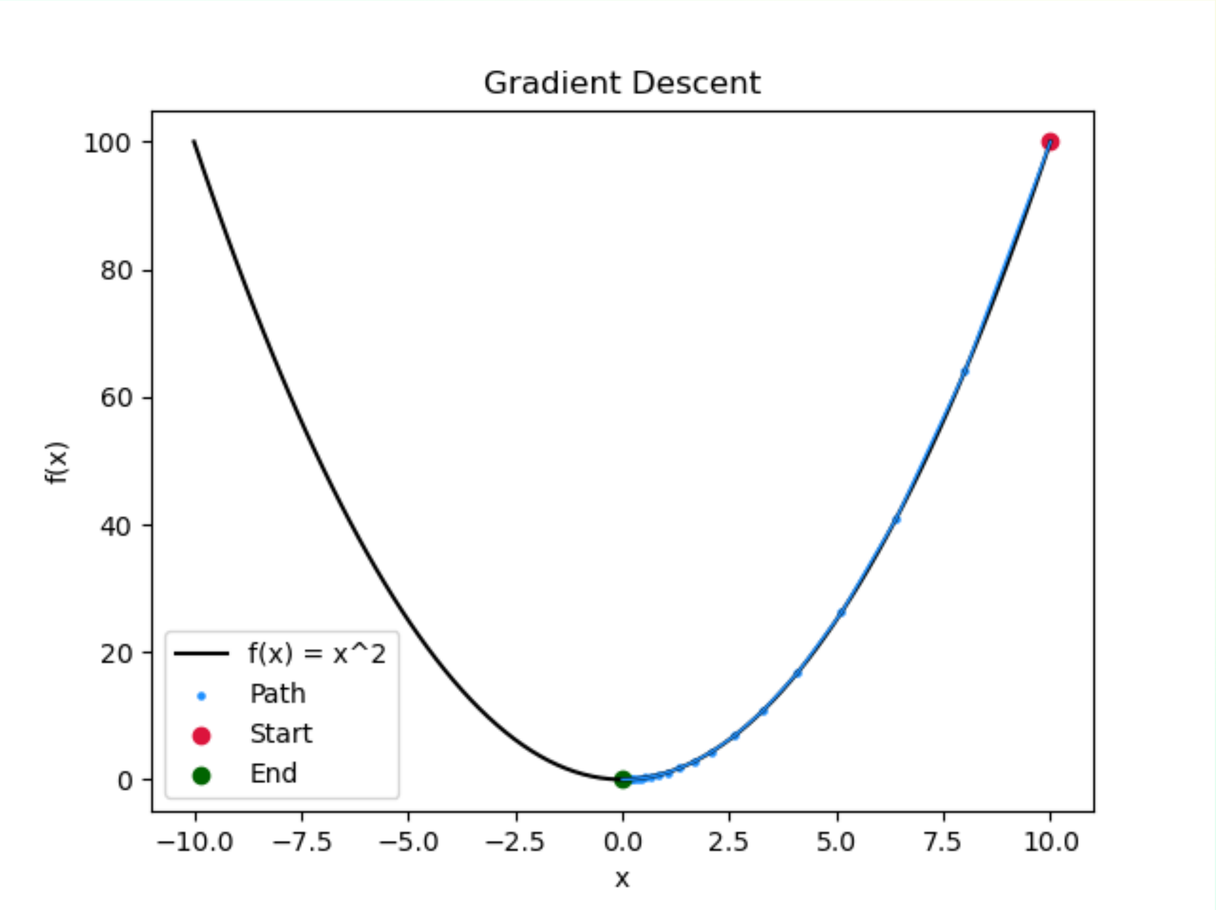
Until:

$$| l(w^{k+1}) - l(w^k) | < \epsilon_l \text{ or } \|w^{k+1} - w^k\| \leq \epsilon_w$$

Usually $\epsilon = 10^{-8}$

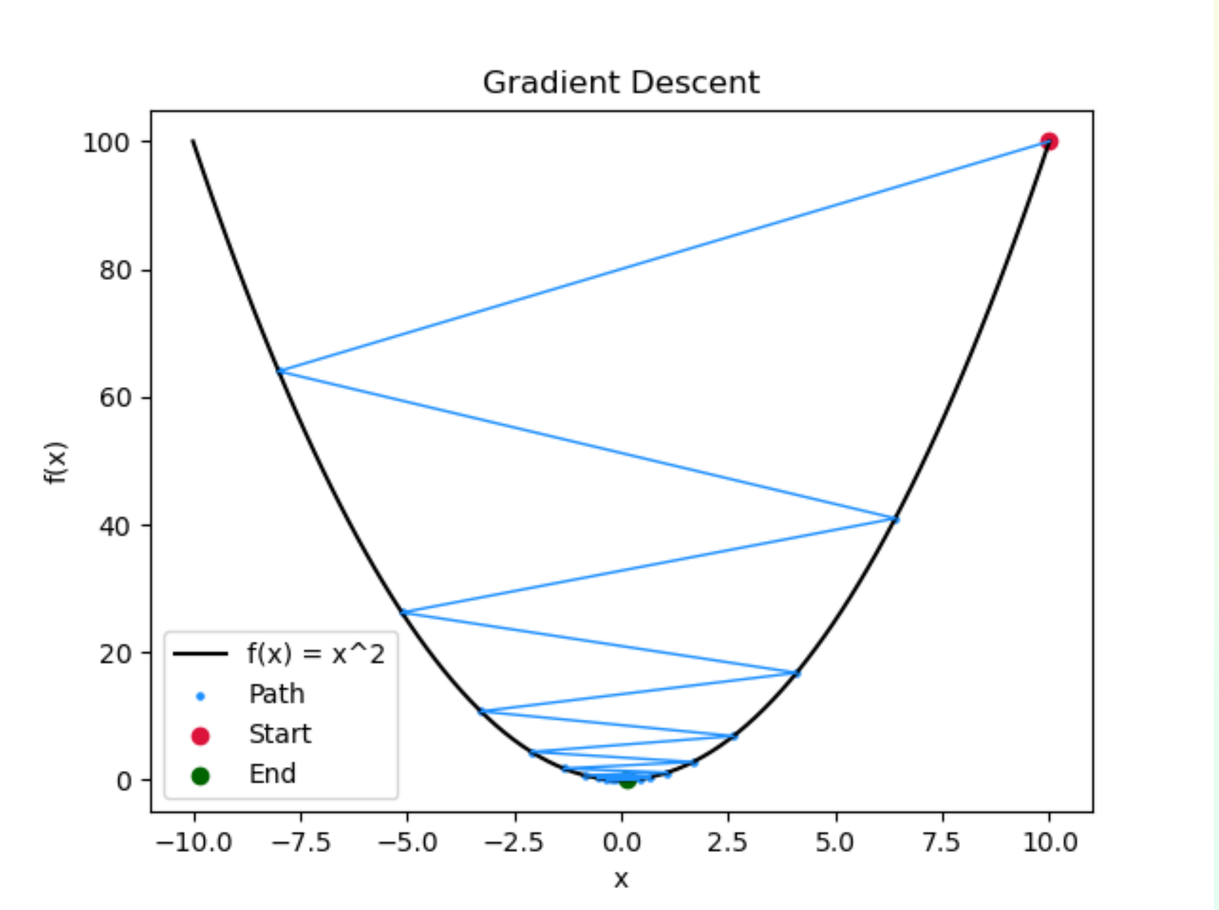
GRADIENT DESCENT

EXAMPLE



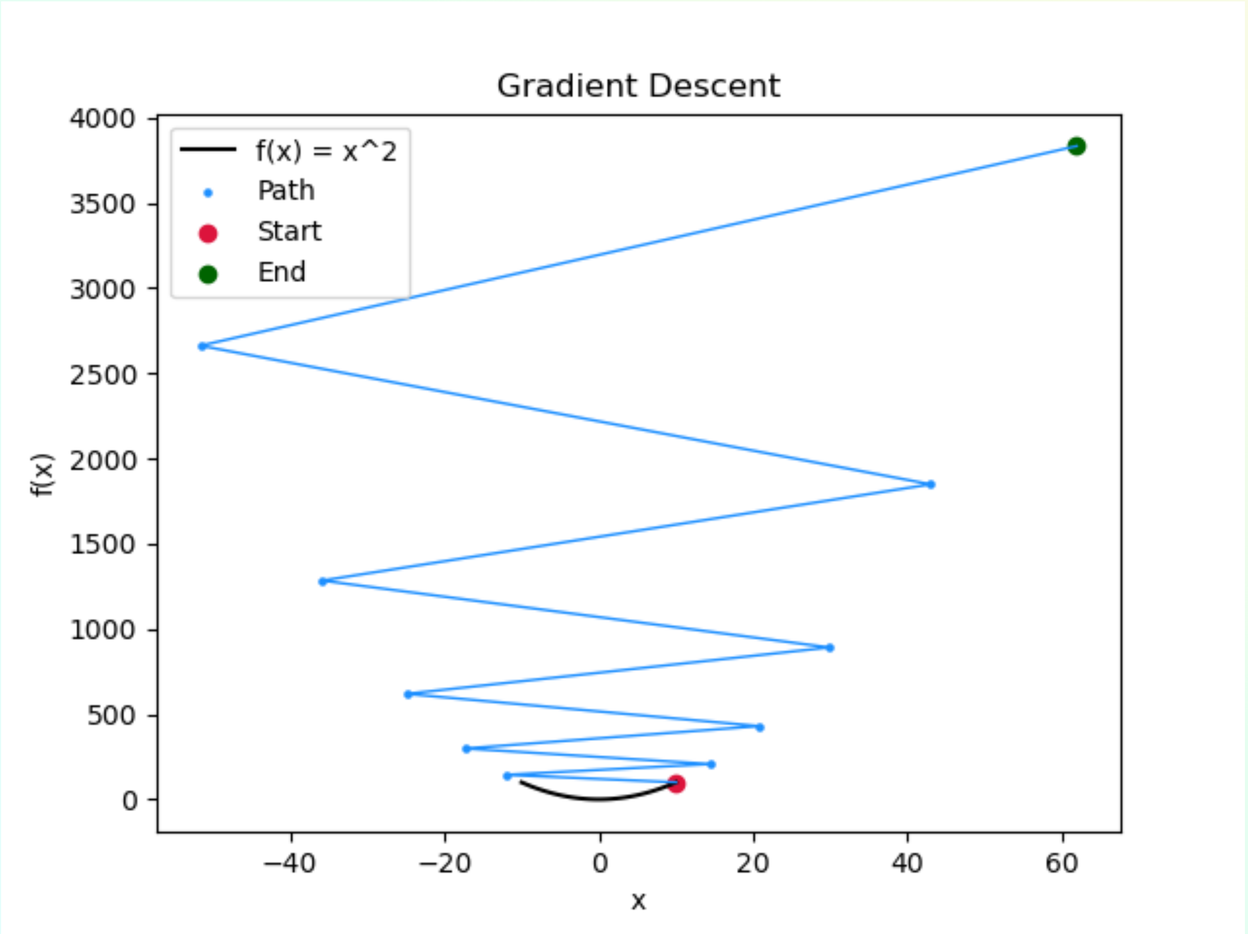
GRADIENT DESCENT

EXAMPLE — LARGE STEP SIZE



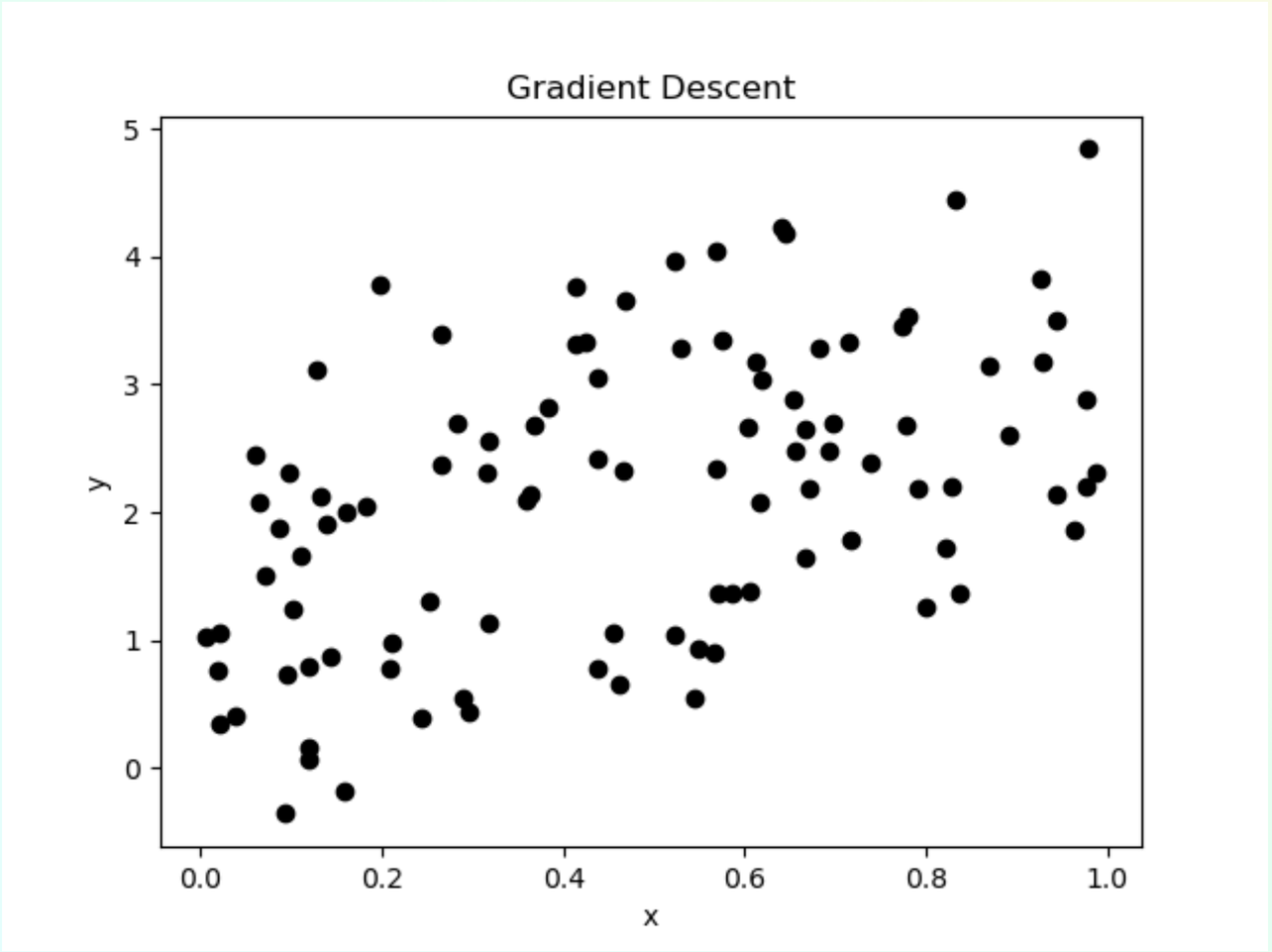
GRADIENT DESCENT

EXAMPLE — TOO LARGE OF STEP SIZE



GRADIENT DESCENT

EXAMPLE — 1D REGRESSION



GRADIENT DESCENT

EXAMPLE — 1D REGRESSION

$$\mathbf{x} = [1, x]$$

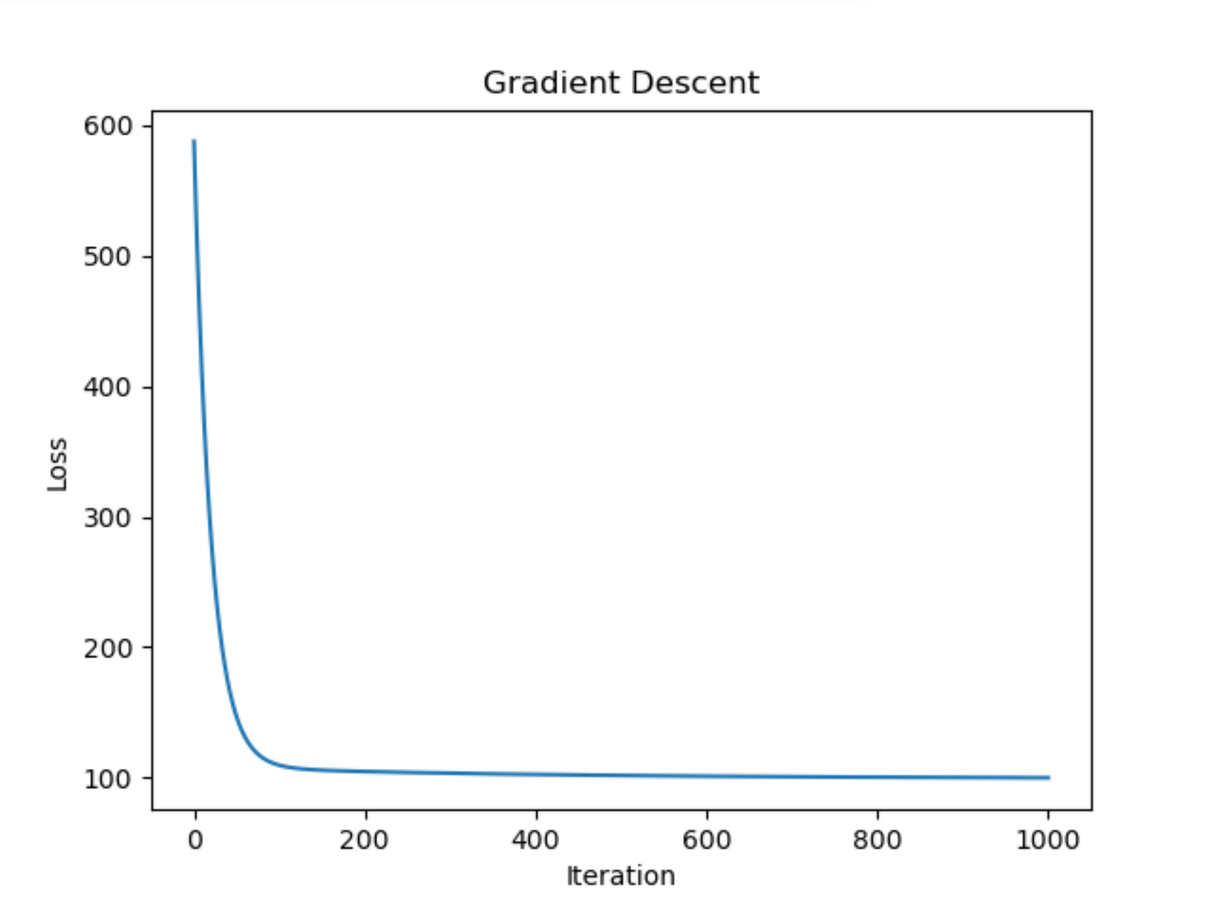
$$f(\mathbf{x}, w) = w^\top \mathbf{x}$$

$$l(w) = \frac{1}{m} \sum_{i=1}^{m} \left(f(\mathbf{x}_i, w) - y_i \right)^2$$

$$\nabla l(w) = \frac{1}{m} \sum_{i=1}^{m} \left(f(\mathbf{x}_i, w) - y_i \right) \mathbf{x}_i$$

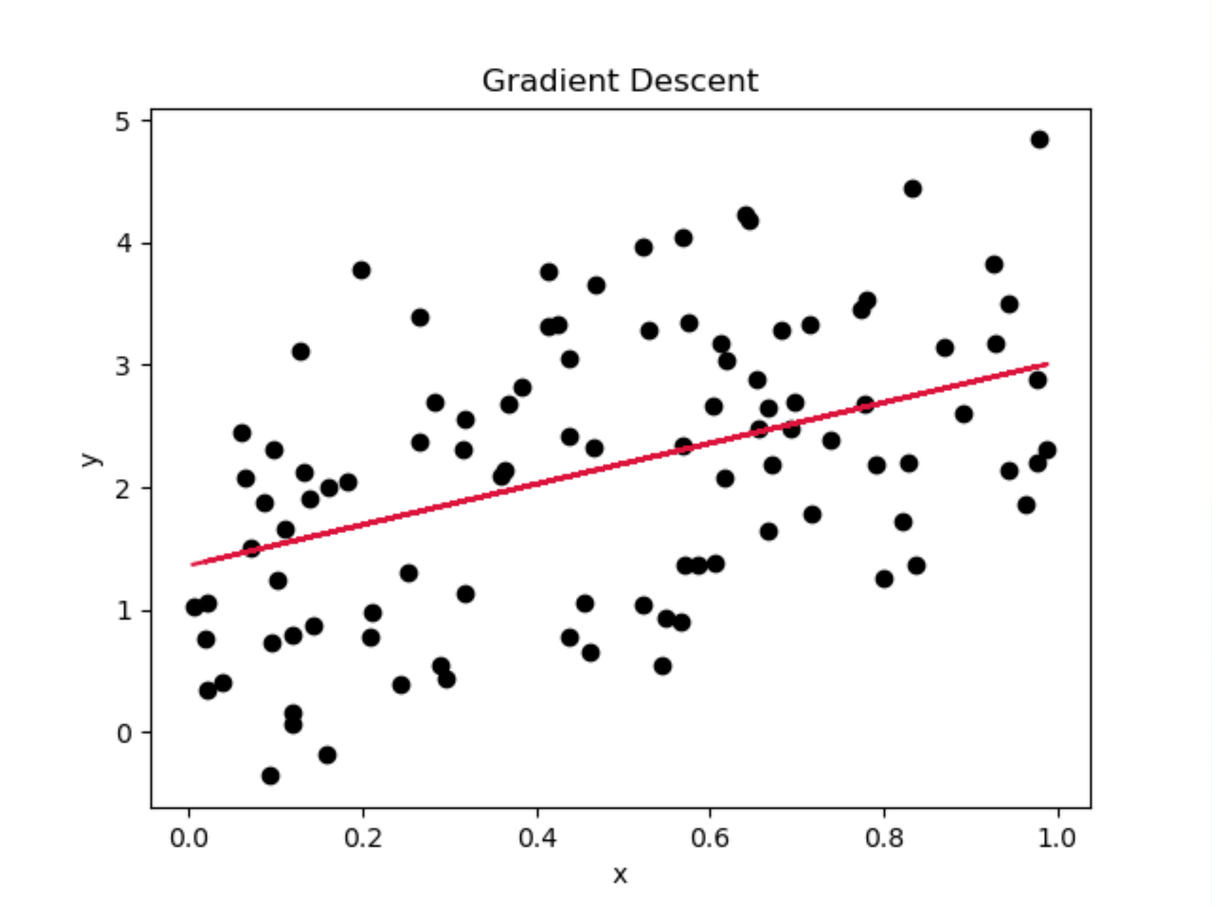
GRADIENT DESCENT

EXAMPLE — 1D REGRESSION



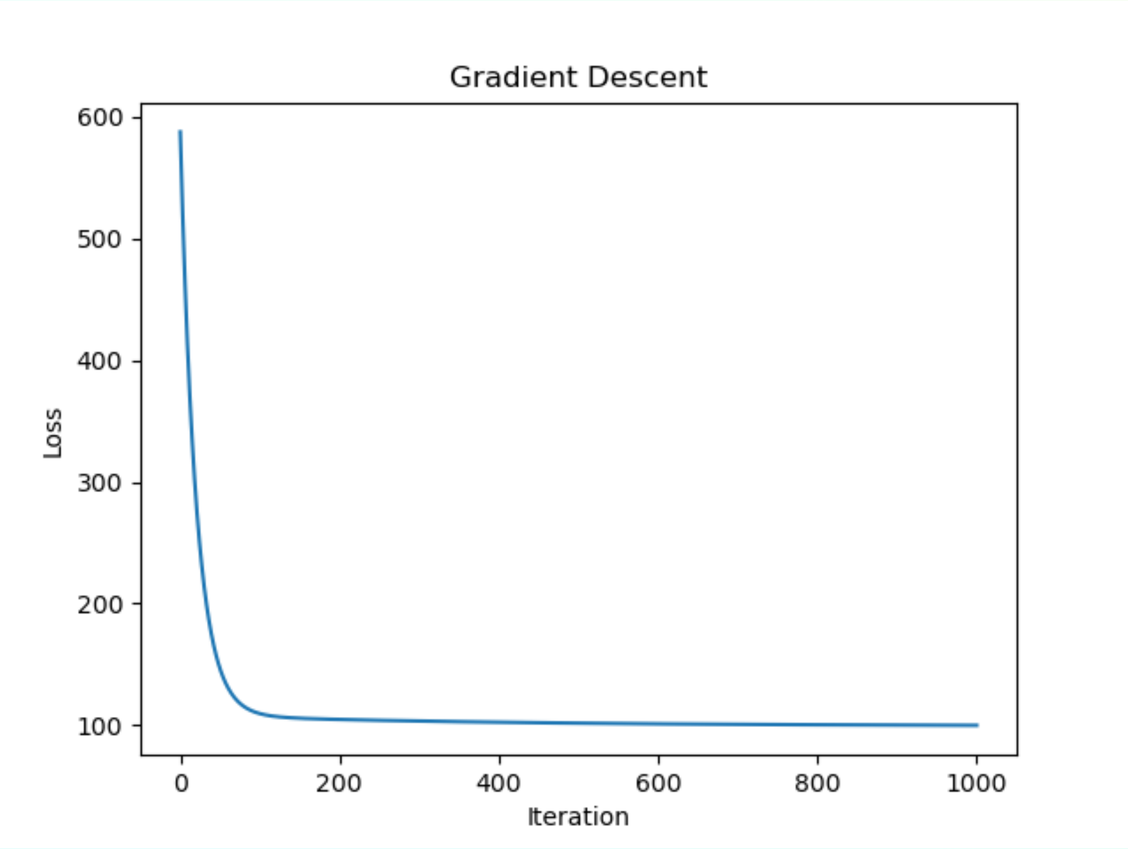
GRADIENT DESCENT

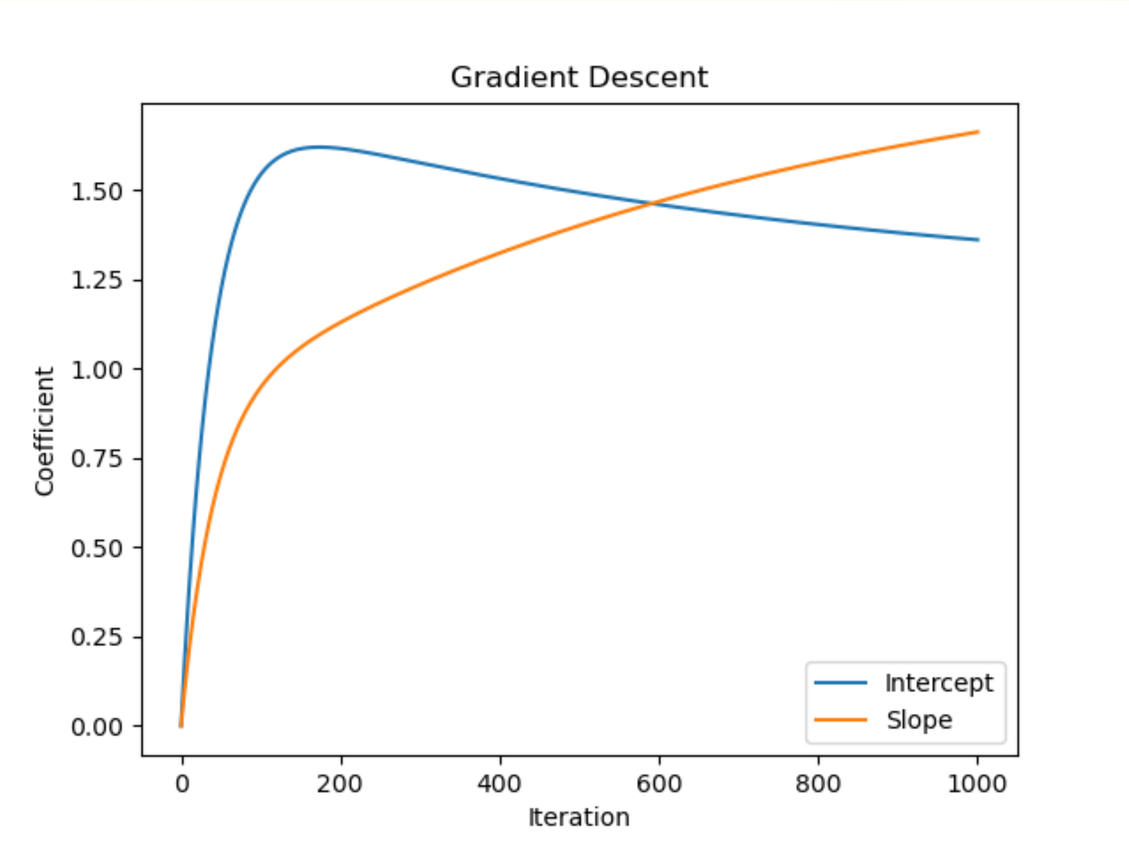
EXAMPLE — 1D REGRESSION



GRADIENT DESCENT

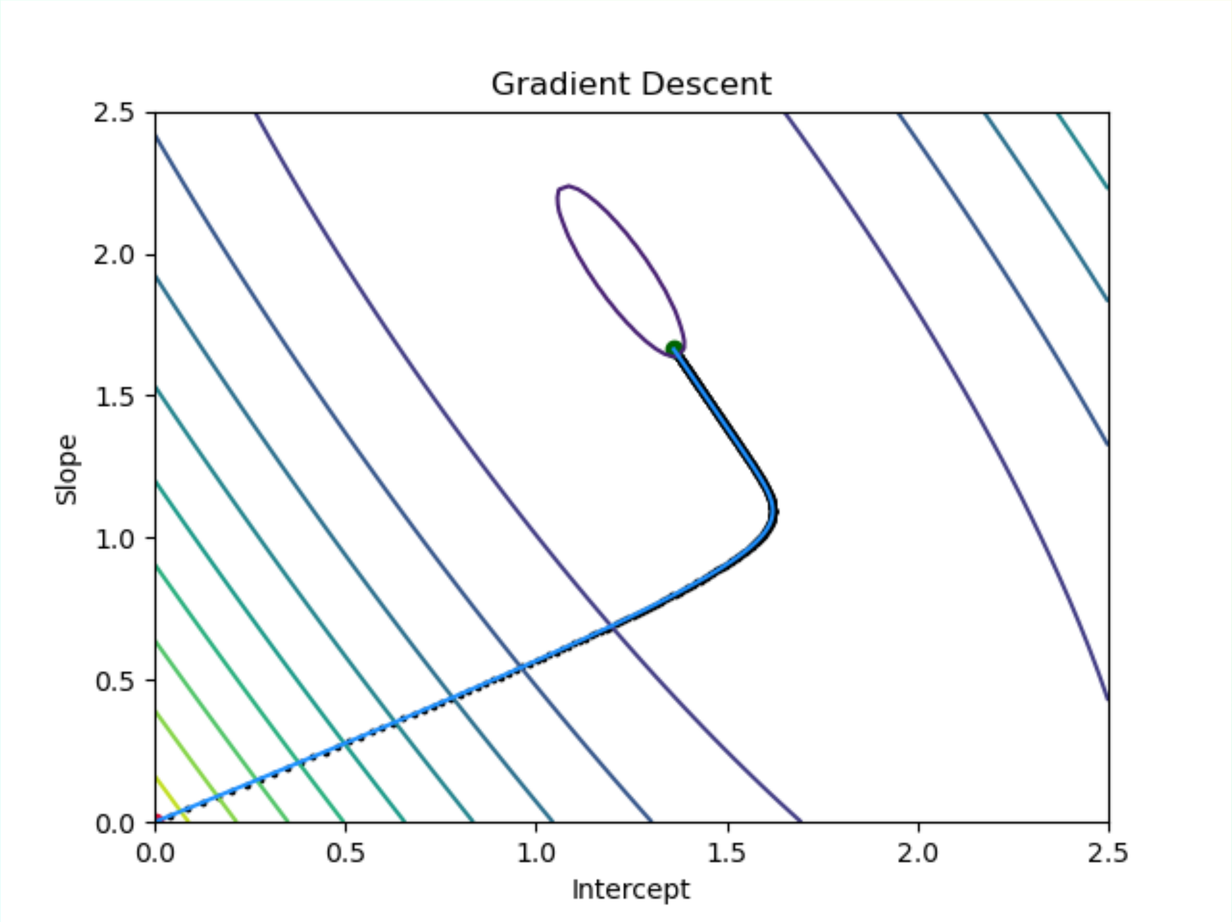
EXAMPLE — 1D REGRESSION





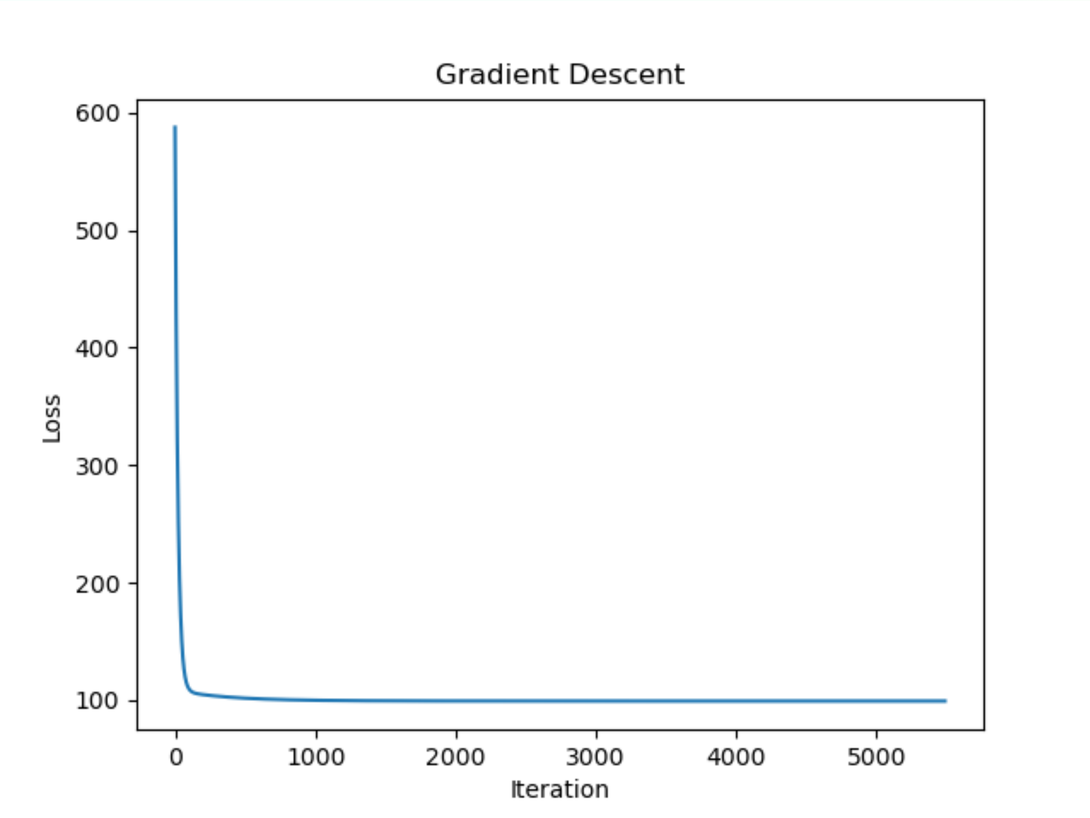
GRADIENT DESCENT

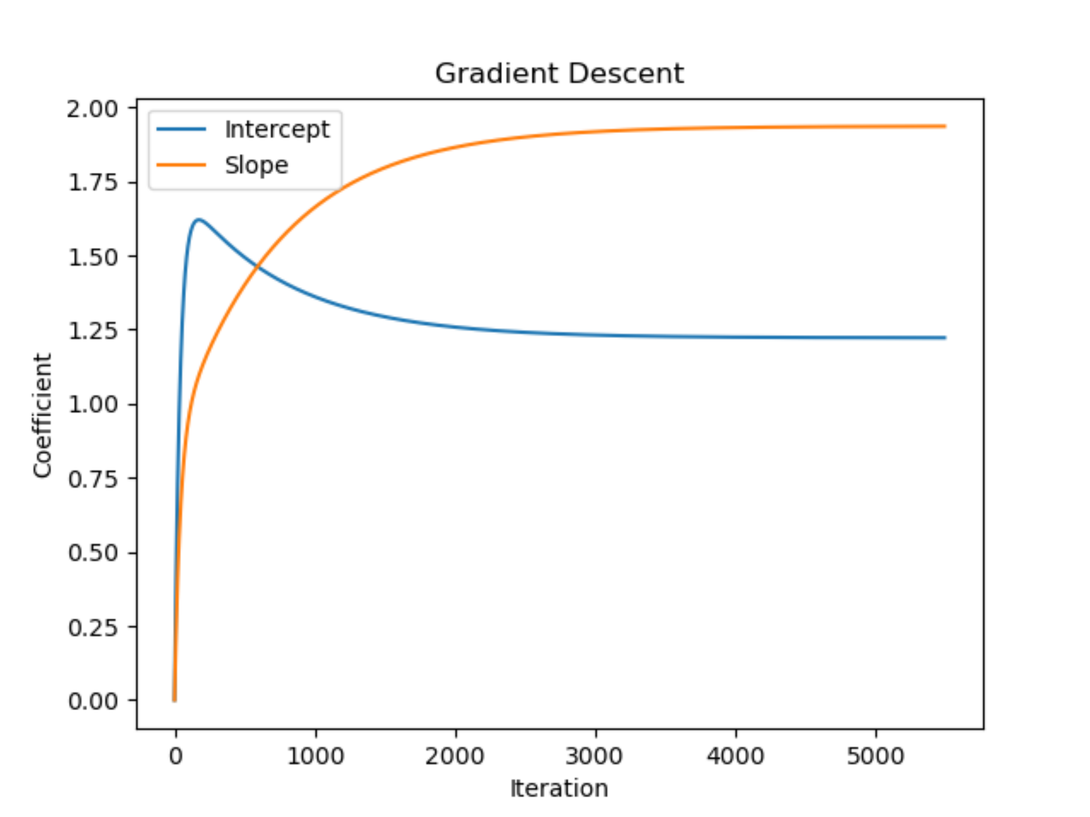
EXAMPLE — 1D REGRESSION



GRADIENT DESCENT

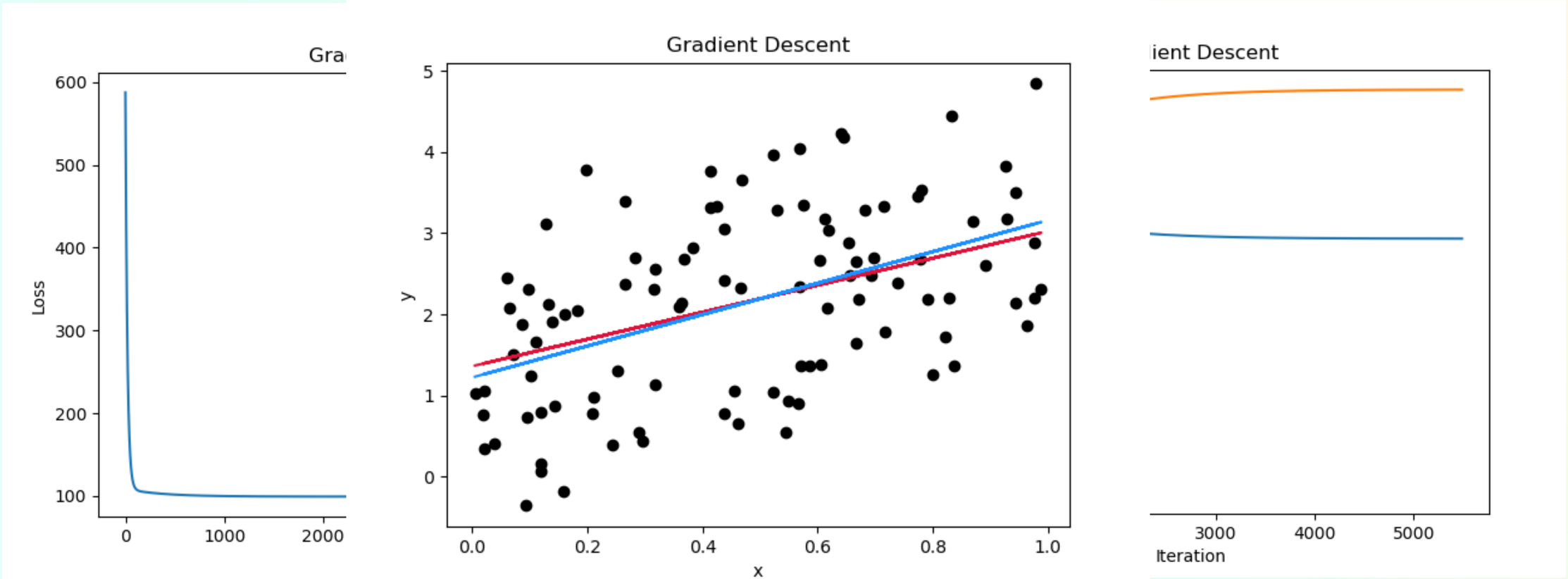
EXAMPLE — 1D REGRESSION





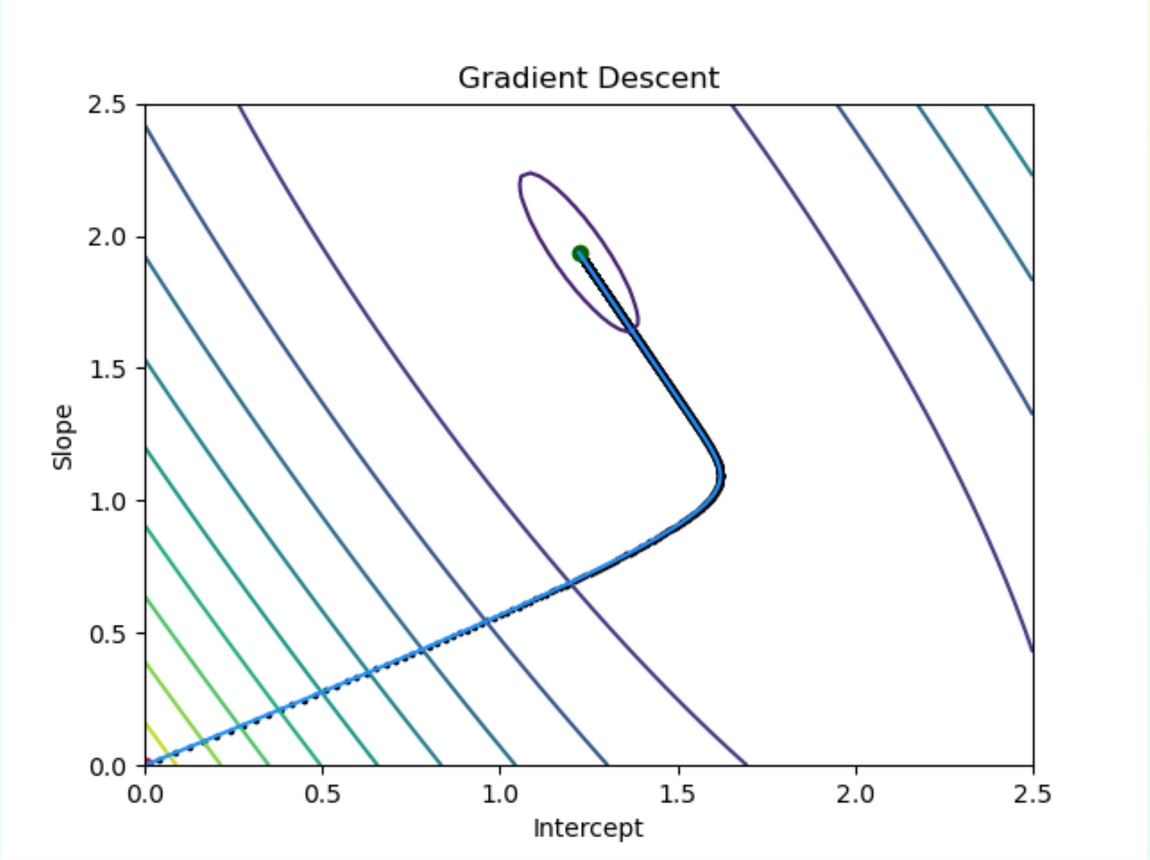
GRADIENT DESCENT

EXAMPLE — 1D REGRESSION



GRADIENT DESCENT

EXAMPLE — 1D REGRESSION



GRADIENT DESCENT

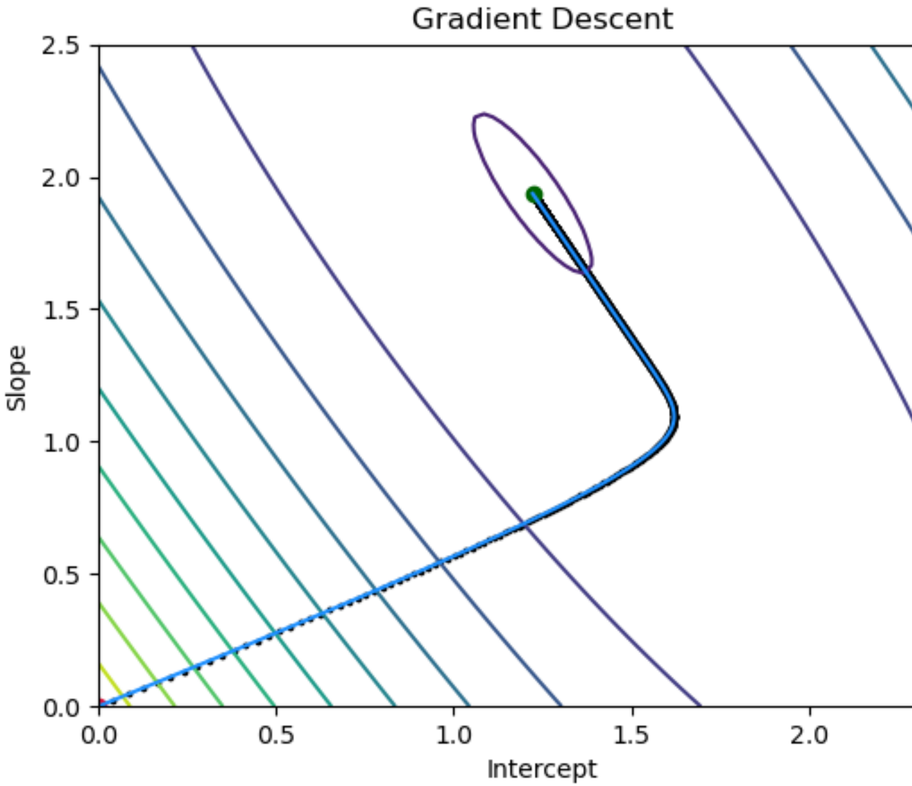
PROPERTIES

Quick initial learning

— gradient is large (steep part of the loss function)

Slow final learning

— gradient is small (flat part of the loss function)



GRADIENT DESCENT

PROPERTIES

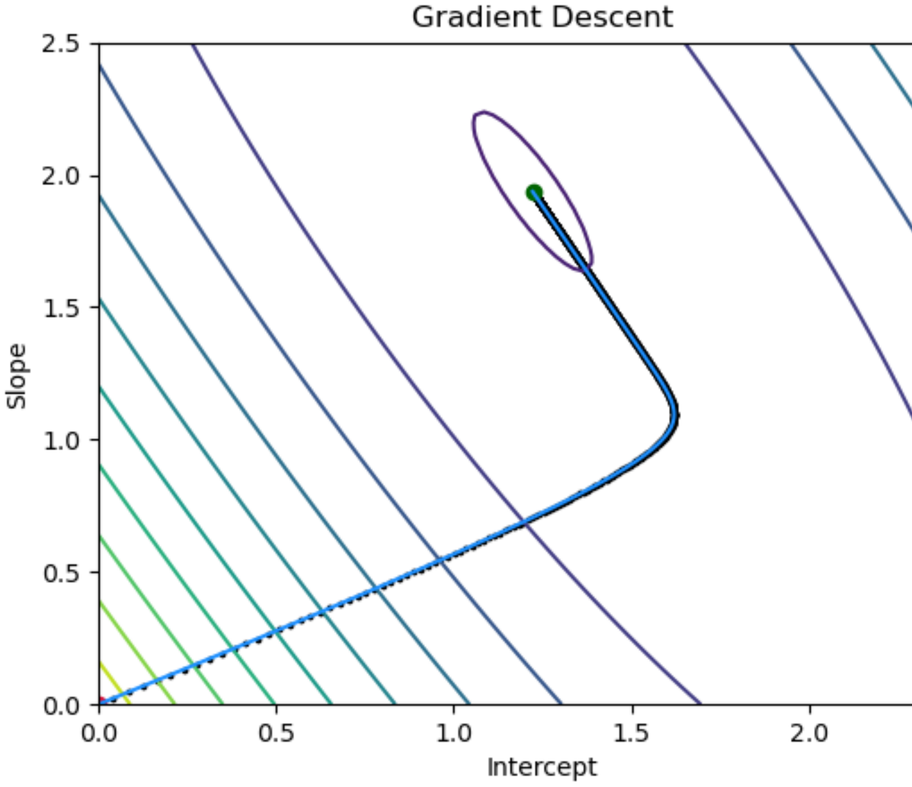
Quick initial learning

— gradient is large (steep part of the loss function)

Slow final learning

— gradient is small (flat part of the loss function)

How do we choose a good step size?



GRADIENT DESCENT

OPTIMAL STEP SIZE

Gradient descent likes smooth functions (do not change too quick)

The gradient is a linear approximation to the loss function (fitting a line to x^2)

If we step too far then the approximation is no longer good enough

$$\hat{l}(\Delta) = l(w) + \nabla l(w)^{\top} \Delta$$

Optimal Δ is one that is infinitely far along $\nabla l(w)$

Need to know how far to step

GRADIENT DESCENT

OPTIMAL STEP SIZE

Idea: create a better approximation:

$$\hat{l}(\Delta) = l(w) + \nabla l(w)^{\top}\Delta + \frac{1}{2}\Delta^{\top}\nabla^2 l(w)\Delta$$

$\nabla^2 l(w) \doteq \dfrac{\partial^2 l(w)}{\partial w \partial w} \in \mathbb{R}^{n \times n}$ — Hessian

$\Delta^{\top}\nabla^2 l(w)\Delta$

— models the curvature of $l(w)$, i.e., how fast $\nabla l(w)$ changes.

— creates a penalty for Δ becoming too large

GRADIENT DESCENT

OPTIMAL STEP SIZE

$$\arg\min_{\Delta} \hat{l}(\Delta) = \arg\min_{\Delta} \left[\nabla l(w)^{\top} \Delta + \frac{1}{2} \Delta^{\top} \nabla^2 l(w) \Delta \right]$$

$$\nabla \hat{l}(\Delta) = 0$$

Solve for Δ

GRADIENT DESCENT

OPTIMAL STEP SIZE

$$\nabla \hat{l}(\Delta) = \nabla l(w) + \nabla^2 l(w)\Delta$$

$$\begin{aligned}
\nabla l(w) + \nabla^2 l(w)\Delta &= 0 \\
\nabla^2 l(w)\Delta &= -\nabla l(w) \\
\nabla^2 l(w)^{-1}\nabla^2 l(w)\Delta &= -\nabla^2 l(w)^{-1}\nabla l(w) \\
\Delta &= -\nabla^2 l(w)^{-1}\nabla l(w)
\end{aligned}$$

GRADIENT DESCENT

NEWTONS METHOD

$$w^{k+1} = w^k - \nabla^2 l(w)^{-1} \nabla l(w)$$

Works really great near the optimum

Takes large steps when the gradient does not change a lot

$\nabla^2 l(w)$ must be invertible

Computationally expensive

GRADIENT DESCENT

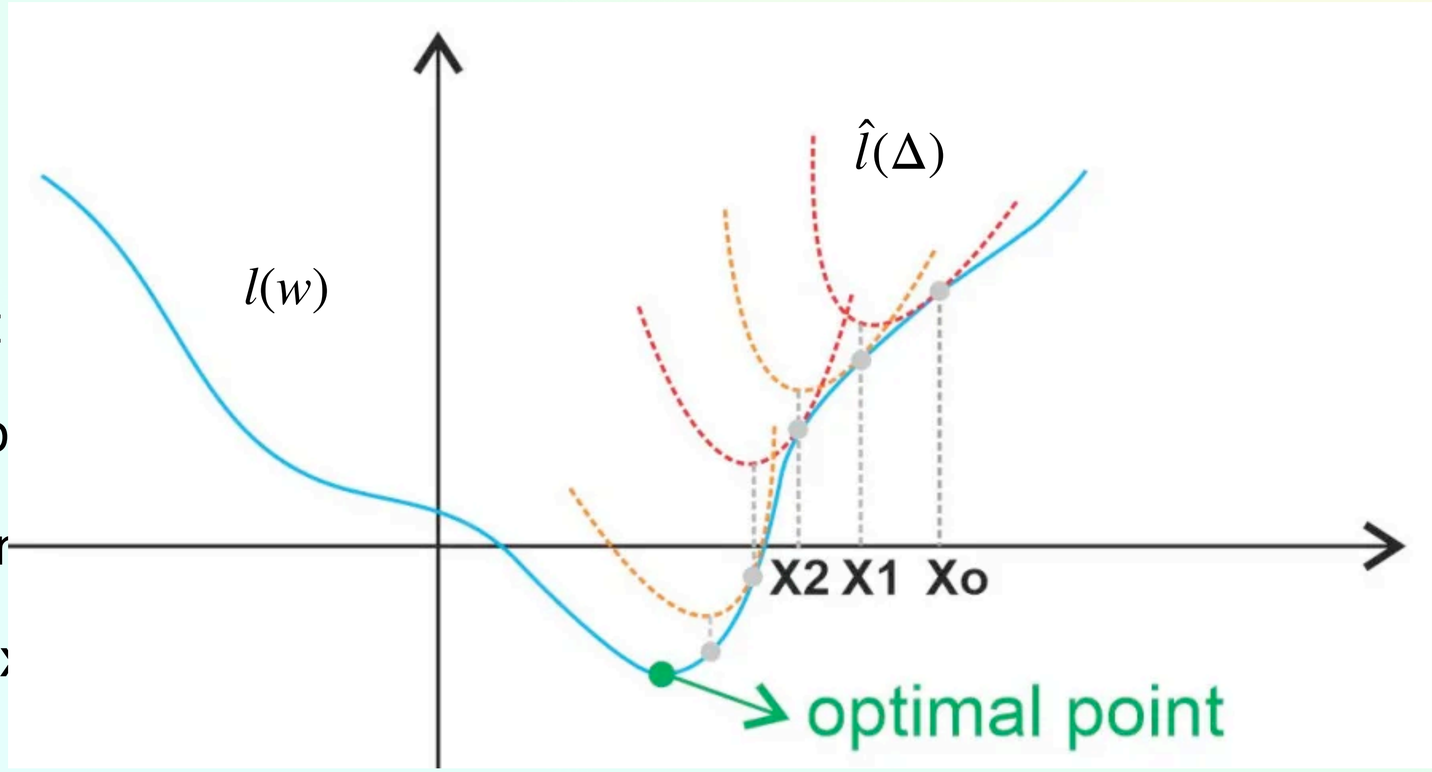
NEWTONS METHOD

Works really great

Takes large step

$\nabla^2 l(w)$ must be ir

Computationally ex



https://ardianumam.wordpress.com/2017/09/27/newtons-method-optimization-derivation-and-how-it-works/

GRADIENT DESCENT

OPTIMAL STEP SIZE

Optimal step size without computing $\nabla^2 l(w)$?

For some problems (linear regression)

$$\forall w, \nabla^2 l(w) = H$$

Then $\eta = \frac{2}{\lambda_1 + \lambda_n}$

Where λ_1 is the largest Eigen value of H and λ_n is the smallest

STOCHASTIC GRADIENT DESCENT

FASTER ITERATIONS

$$\nabla l(w) = \frac{1}{m} \sum_{i=1}^{m} (f(x_i, w) - y_i) x_i$$

What if m is very large (thousands, millions)

For every update we have to compute the gradient from all m points!

We need lots of gradient steps to find the minimum

STOCHASTIC GRADIENT DESCENT

FASTER ITERATIONS

$$\nabla l(w) = \frac{1}{m} \sum_{i=1}^{m} (f(x_i, w) - y_i) x_i$$

What if m is very large (thousands, millions)

For every update we have to compute the gradient from all m points!

We need lots of gradient steps to find the minimum

Idea: do not use all the data to update w

STOCHASTIC GRADIENT DESCENT

FASTER ITERATIONS

What if we compute the gradient from only $m - 1$ points? Is this direction good?

What if we compute using a single data point (x_i, y_i)?

STOCHASTIC GRADIENT DESCENT

FASTER ITERATIONS

What if we compute the gradient from only $m - 1$ points? Is this direction good?

What if we compute using a single data point (x_i, y_i)?

- Any single point may not point in the direction of the gradient
- On average the gradients of a single point will be the gradient

STOCHASTIC GRADIENT DESCENT

FASTER ITERATIONS

If $l(w) = \mathbf{E}\left[l_i(w)\right]$ then:

$$\nabla l(w) = \mathbf{E}\left[\nabla l_i(w)\right]$$

Randomly sample x_i, y_i from the data set D (with replacement)

$$l_i(w) = \left(f(X_i, w) - Y_i\right)^2$$

$$\mathbf{E}[l_i(w)] = \frac{1}{m}\sum_{i=1}^{m} l_i(w) = \frac{1}{m}\sum_{i=1}^{m}\left(f(x_i, w) - y_i\right)^2 = l(w)$$

STOCHASTIC GRADIENT DESCENT

PROCESS

Repeat forever

Sample $x_i, y_i \sim D$

Update parameters $w^{k+1} = w^k - \eta \nabla l_i(w)$

STOCHASTIC GRADIENT DESCENT

PROCESS

Repeat forever

Sample $x_i, y_i \sim D$

Update parameters $w^{k+1} = w^k - \eta \nabla l_i(w)$

Never converges to the optimal weights w^*

STOCHASTIC GRADIENT DESCENT

PROCESS

To find w^* two general approaches:

Decay η overtime: $\eta_k \propto \frac{1}{k}$,

any sequence of η_k is fine so long as $\sum_{k=1}^{\infty} \eta_k = \infty$ and $\sum_{k=1}^{\infty} \eta_k^2 < \infty$

STOCHASTIC GRADIENT DESCENT

PROCESS

To find w^* two general approaches:

1. Decay η overtime: $\eta_k \propto \frac{1}{k}$,

   any sequence of η_k is fine so long as $\sum_{k=1}^{\infty} \eta_k = \infty$ and $\sum_{k=1}^{\infty} \eta_k^2 < \infty$

   $\lim_{k \to \infty} w^k \to w^*$

2. Average w^k, Let $\bar{w}^k = \frac{1}{k} \sum_{j=1}^{k} w^k$

   For some $\eta > 0$, $\lim_{k \to \infty} \bar{w}^k = w^*$

STOCHASTIC GRADIENT DESCENT

PROCESS

2. Average w^k, Let $\bar{w}^k = \frac{1}{k}\sum_{j=1}^{k} w^k$

For some $\eta > 0$, $\lim_{k\to\infty} \bar{w}^k = w^*$

This one works because w^k will bounce around w^*

(Example in homework)

STOCHASTIC GRADIENT DESCENT

STEP SIZE

We often use a constant step size:

1. Decaying ones are too slow
2. We never learn forever
3. Seems to work well enough in practice

STOCHASTIC GRADIENT DESCENT

STEP SIZE

What step size should we use?

For regression:

Idea: largest step size that does not increase $l_i(w)$ after updating

STOCHASTIC GRADIENT DESCENT

STEP SIZE

$$w^{k+1} = w^k - \eta \nabla l_i(w^k)$$

$$l_i(w^{k+1}) = (f(x_i, w^{k+1}) - y_i)^2$$

Find η such that $l_i(w^{k+1})$ is minimized

Expand $l_i(w^{k+1})$ into a term that include η

Solve for η that makes the gradient of $\nabla l_i(w^{k+1}) = 0$

Take home Questions

NEXT CLASS

Next Class — ML Basics