On Linear Regression: Regularization, Probabilistic Interpretation, and Gradient Descent

Karl Stratos

June 19, 2018

Recall: Linear Regression

- Given $S = \left\{(\boldsymbol{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n}$, find model parameter $\boldsymbol{w} \in \mathbb{R}^d$ that minimizes the sum of squared errors:

$$J_S^{\mathrm{LS}}(\boldsymbol{w}) := \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)} \right)^2$$

- We will discuss three topics through linear regression.
    1. **Regularization** to prevent overfitting
    2. **Maximum likelihood estimation** (MLE) interpretation
    3. **Gradient descent** to estimate model parameter

- Far-reaching implications beyond linear regression

Overview

Regularization

A Probabilistic Interpretation of Linear Regression

Optimization by Local Search

Motivation

- The least squares solution is the *best* linear regressor on training data S (solved in closed-form):

$$\boldsymbol{w}_S^{\mathrm{LS}} := \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w})$$

Motivation

- The least squares solution is the *best* linear regressor on training data S (solved in closed-form):

$$\boldsymbol{w}_S^{\mathrm{LS}} := \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w})$$

- But we care nothing about how well we do on S! Rather, what we really care about is:

Can $\boldsymbol{w}_S^{\mathrm{LS}}$ handle a new $\boldsymbol{x}$ not already seen in S?

Motivation

- The least squares solution is the *best* linear regressor on training data S (solved in closed-form):

$$\boldsymbol{w}_S^{\mathrm{LS}} := \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w})$$

- But we care nothing about how well we do on S! Rather, what we really care about is:

Can $\boldsymbol{w}_S^{\mathrm{LS}}$ handle a new $\boldsymbol{x}$ not already seen in S?

- This is the heart of machine learning: thoery/applications of *generalization*.

A Model of Noisy Environment

- There is some "true" parameter $\boldsymbol{w}^* \in \mathbb{R}^d$.

A Model of Noisy Environment

- There is some “true” parameter $\boldsymbol{w}^* \in \mathbb{R}^d$.
- There is some input distribution $\boldsymbol{x} \sim \mathcal{D}$ and some **noise** distribution $\epsilon \sim \mathcal{E}$. Assume that $\mathbf{E}[\epsilon] = 0$ and $\mathrm{Var}(\epsilon) = \sigma^2$.

A Model of Noisy Environment

- There is some "true" parameter $\boldsymbol{w}^* \in \mathbb{R}^d$.
- There is some input distribution $\boldsymbol{x} \sim \mathcal{D}$ and some **noise** distribution $\epsilon \sim \mathcal{E}$. Assume that $\mathbf{E}[\epsilon] = 0$ and $\mathrm{Var}(\epsilon) = \sigma^2$.
- Each sample $(\boldsymbol{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ is generated by drawing $\boldsymbol{x} \sim \mathcal{D}$ and $\epsilon \sim \mathcal{E}$ and setting

$$y = \boldsymbol{w}^* \cdot \boldsymbol{x} + \epsilon$$

(Thus the training data S is a random variable.)

A Model of Noisy Environment

- There is some "true" parameter $\boldsymbol{w}^* \in \mathbb{R}^d$.
- There is some input distribution $\boldsymbol{x} \sim \mathcal{D}$ and some **noise** distribution $\epsilon \sim \mathcal{E}$. Assume that $\mathbf{E}\left[\epsilon\right] = 0$ and $\mathrm{Var}\left(\epsilon\right) = \sigma^2$.
- Each sample $(\boldsymbol{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ is generated by drawing $\boldsymbol{x} \sim \mathcal{D}$ and $\epsilon \sim \mathcal{E}$ and setting

$$y = \boldsymbol{w}^* \cdot \boldsymbol{x} + \epsilon$$

(Thus the training data S is a random variable.)

- Check that $\boldsymbol{w}_S^{\mathrm{LS}}$ is consistent/unbiased:

$$\mathbf{E}_S[\boldsymbol{w}_S^{\mathrm{LS}}] = \mathbf{E}_S\left[\boldsymbol{X}_S^+ \left(\boldsymbol{X}_S \boldsymbol{w}^* + \boldsymbol{\epsilon}\right)\right] = \boldsymbol{w}^*$$

Measuring the Future Performance

- We want $\boldsymbol{w}_S^{\mathrm{LS}}$ which is trained on S to incur small loss in expectation ("true/population error"):

$$\mathbf{E}_{S,\boldsymbol{x},\epsilon}\left[\left((\boldsymbol{w}^* \cdot \boldsymbol{x} + \epsilon) - \boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right)^2\right]$$

Measuring the Future Performance

- We want $\boldsymbol{w}_S^{\mathrm{LS}}$ which is trained on S to incur small loss in expectation ("true/population error"):

$$\mathbf{E}_{S,\boldsymbol{x},\epsilon}\left[\left((\boldsymbol{w}^* \cdot \boldsymbol{x} + \epsilon) - \boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right)^2\right]$$

- By the bias-variance decomposition of squared error, this is (omitting the expectation over $\boldsymbol{x}$):

$$\underbrace{\left(\boldsymbol{w}^* \cdot \boldsymbol{x} - \mathbf{E}_S[\boldsymbol{w}_S^{\mathrm{LS}}] \cdot \boldsymbol{x}\right)^2}_{\text{0 in this case}} + \mathrm{Var}_S\left(\boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right) + \underbrace{\sigma^2}_{\text{can't help}}$$

Measuring the Future Performance

- We want $\boldsymbol{w}_S^{\mathrm{LS}}$ which is trained on S to incur small loss in expectation ("true/population error"):

$$\mathbf{E}_{S,\boldsymbol{x},\epsilon}\left[\left((\boldsymbol{w}^* \cdot \boldsymbol{x} + \epsilon) - \boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right)^2\right]$$

- By the bias-variance decomposition of squared error, this is (omitting the expectation over $\boldsymbol{x}$):

$$\underbrace{\left(\boldsymbol{w}^* \cdot \boldsymbol{x} - \mathbf{E}_S[\boldsymbol{w}_S^{\mathrm{LS}}] \cdot \boldsymbol{x}\right)^2}_{\text{0 in this case}} + \mathrm{Var}_S\left(\boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right) + \underbrace{\sigma^2}_{\text{can't help}}$$

- The variance term can be large if parameter values are large.
  - $\left(\boldsymbol{w}_S^{\mathrm{LS}} \cdot \boldsymbol{x}\right)^2$ more sensitive to a perturbation of S

Ridge Regression

- "Shrink" the size of the estimator by penalizing its l_2 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} := \arg\min_{\boldsymbol{w} \in \mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda \, ||\boldsymbol{w}||_2^2$$

Ridge Regression

- "Shrink" the size of the estimator by penalizing its l_2 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} := \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda \left|\left|\boldsymbol{w}\right|\right|_2^2$$

- Closed-form solution given by (hence the name)

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} = (\boldsymbol{X}_S^\top \boldsymbol{X}_S + \lambda I_{d \times d})^{-1} \boldsymbol{X}_S^\top \boldsymbol{y}$$

Ridge Regression

- "Shrink" the size of the estimator by penalizing its l_2 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} := \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda \, ||\boldsymbol{w}||_2^2$$

- Closed-form solution given by (hence the name)

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} = (\boldsymbol{X}_S^\top \boldsymbol{X}_S + \lambda I_{d \times d})^{-1} \boldsymbol{X}_S^\top \boldsymbol{y}$$

- No longer unbiased: $\mathbf{E}_S[\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}}] \neq \boldsymbol{w}^*$ for $\lambda > 0$.

Ridge Regression

- "Shrink" the size of the estimator by penalizing its l_2 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} := \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda\,||\boldsymbol{w}||_2^2$$

- Closed-form solution given by (hence the name)

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}} = (\boldsymbol{X}_S^\top \boldsymbol{X}_S + \lambda I_{d\times d})^{-1}\boldsymbol{X}_S^\top \boldsymbol{y}$$

- No longer unbiased: $\mathbf{E}_S[\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}}] \neq \boldsymbol{w}^*$ for $\lambda > 0$.

- But the true error might be smaller!

$$\underbrace{\left(\boldsymbol{w}^* \cdot \boldsymbol{x} - \mathbf{E}_S[\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}}]\cdot \boldsymbol{x}\right)^2}_{\text{no longer 0}} + \underbrace{\mathrm{Var}_S\left(\boldsymbol{w}_{S,\lambda}^{\mathrm{LSR}}\cdot \boldsymbol{x}\right)}_{\text{smaller}} + \underbrace{\sigma^2}_{\text{can't help}}$$

Lasso Regression

- Another idea: penalize the l_1 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSL}} := \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda\,||\boldsymbol{w}||_1$$

Lasso Regression

- Another idea: penalize the l_1 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSL}} := \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda\,||\boldsymbol{w}||_1$$

- Still convex though not differentiable. Can be solved by existing convex optimization methods or subgradient descent.

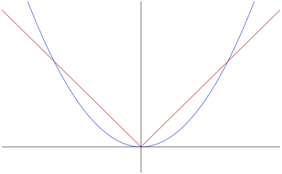
Lasso Regression

- Another idea: penalize the l_1 norm:

$$\boldsymbol{w}_{S,\lambda}^{\mathrm{LSL}} := \underset{\boldsymbol{w}\in\mathbb{R}^d}{\arg\min}\, J_S^{\mathrm{LS}}(\boldsymbol{w}) + \lambda\, ||\boldsymbol{w}||_1$$

- Still convex though not differentiable. Can be solved by existing convex optimization methods or subgradient descent.

- Solutions with zero entries are encouraged (hence the name).



(squared l_2 norm penalty vs l_1 norm penalty)

Summary on Regularization

- The l_2/l_1 regularized solutions can be framed as constrained solutions: for some $\alpha, \beta \in \mathbb{R}$

$$\boldsymbol{w}_{S,\lambda}^{\text{LSL}} := \underset{\boldsymbol{w} \in \mathbb{R}^d:\ ||w||_2 \leq \alpha}{\arg\min}\ J_S^{\text{LS}}(\boldsymbol{w})$$

$$\boldsymbol{w}_{S,\lambda}^{\text{LSR}} := \underset{\boldsymbol{w} \in \mathbb{R}^d:\ ||w||_1 \leq \beta}{\arg\min}\ J_S^{\text{LS}}(\boldsymbol{w})$$

- This is all to optimize the *expected future performance*.

- If we have infinite data, we don't need to worry about regularization.

General Definition of Overfitting

- **Population**: Input-output pairs (x, y) are distributed as $\mathcal{D}$.
- **Loss function**: A loss function $l(y, y') \in \mathbb{R}$ specifies the penalty on predicting y' when the correct answer is y.

General Definition of Overfitting

- **Population**: Input-output pairs (x, y) are distributed as $\mathcal{D}$.
- **Loss function**: A loss function $l(y, y') \in \mathbb{R}$ specifies the penalty on predicting y' when the correct answer is y.
- **Hypothesis class**: A class of functions $\mathcal{H}$ is chosen to model the input-output relationship (e.g., all hyperplanes).

General Definition of Overfitting

- **Population**: Input-output pairs (x, y) are distributed as $\mathcal{D}$.
- **Loss function**: A loss function $l(y, y') \in \mathbb{R}$ specifies the penalty on predicting y' when the correct answer is y.
- **Hypothesis class**: A class of functions $\mathcal{H}$ is chosen to model the input-output relationship (e.g., all hyperplanes).
- **Training data**: A fixed set of samples S is used to obtain your hypothesis

$$h_S = \arg\min_{h \in \mathcal{H}} \widehat{\mathbf{E}}_S \left[l(y, h_S(x)) \right] = \arg\min_{h \in \mathcal{H}} \frac{1}{|S|} \sum_{(x,y) \in S} l(y, h(x))$$

General Definition of Overfitting

- **Population**: Input-output pairs (x, y) are distributed as $\mathcal{D}$.
- **Loss function**: A loss function $l(y, y') \in \mathbb{R}$ specifies the penalty on predicting y' when the correct answer is y.
- **Hypothesis class**: A class of functions $\mathcal{H}$ is chosen to model the input-output relationship (e.g., all hyperplanes).
- **Training data**: A fixed set of samples S is used to obtain your hypothesis

$$h_S = \arg\min_{h \in \mathcal{H}} \widehat{\mathbf{E}}_S \left[l(y, h_S(x)) \right] = \arg\min_{h \in \mathcal{H}} \frac{1}{|S|} \sum_{(x,y) \in S} l(y, h(x))$$

- We say h_S **overfits** S if there is $h \in \mathcal{H}$ such that

$$\widehat{\mathbf{E}}_S \left[l(y, h_S(x)) \right] < \widehat{\mathbf{E}}_S \left[l(y, h(x)) \right]$$
$$\mathbf{E} \left[l(y, h_S(x)) \right] > \mathbf{E} \left[l(y, h(x)) \right]$$

Overview

Regularization

A Probabilistic Interpretation of Linear Regression

Optimization by Local Search

Linear Regression as MLE

- **Claim.** If $S = \left\{(\boldsymbol{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n}$ is generated by a particular probabilistic model, then the least squares solution is also the maximum likelihood solution under this model:

$$\boldsymbol{w}_S^{\mathrm{LS}} = \arg\max_{\boldsymbol{w} \in \mathbb{R}^d} \Pr\left(S | \boldsymbol{w}\right)$$

Linear Regression as MLE

- **Claim.** If $S = \left\{(\boldsymbol{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n}$ is generated by a particular probabilistic model, then the least squares solution is also the maximum likelihood solution under this model:

$$\boldsymbol{w}_S^{\mathrm{LS}} = \arg\max_{\boldsymbol{w} \in \mathbb{R}^d} \Pr\left(S | \boldsymbol{w}\right)$$

- Provides an alternative characterization of the method.

- This is a recurring theme: different approaches "converge" to the same thing.

The Probabilistic Model

- There is some input distribution $\boldsymbol{x} \sim \mathcal{D}$ (as before).
- The noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is (centered) **Gaussian**.
- For any $\boldsymbol{w} \in \mathbb{R}^d$, the output value is set to $y = \boldsymbol{w} \cdot \boldsymbol{x} + \epsilon$.

The Probabilistic Model

- There is some input distribution $\boldsymbol{x} \sim \mathcal{D}$ (as before).
- The noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is (centered) **Gaussian**.
- For any $\boldsymbol{w} \in \mathbb{R}^d$, the output value is set to $y = \boldsymbol{w} \cdot \boldsymbol{x} + \epsilon$.
- Thus

$$\begin{aligned}\Pr(\boldsymbol{x}, y | \boldsymbol{w}) &= \Pr(\boldsymbol{x}) \Pr(y | \boldsymbol{w}, \boldsymbol{x}) \\ &= \mathcal{D}(\boldsymbol{x}) \left(\frac{1}{\sqrt{2\pi}\sigma^2} \exp\left(\frac{-(y - \boldsymbol{w} \cdot \boldsymbol{x})^2}{2\sigma^2} \right) \right)\end{aligned}$$

MLE Coincides with Least Squares

Assuming each sample in $S = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$ is drawn iid,

$$\boldsymbol{w}_S^{\text{MLE}} := \arg\max_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i=1}^{n} \log \Pr\left(\boldsymbol{x}^{(i)}, y^{(i)} | \boldsymbol{w}\right)$$

MLE Coincides with Least Squares

Assuming each sample in $S = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$ is drawn iid,

$$\boldsymbol{w}_S^{\mathrm{MLE}} := \arg\max_{\boldsymbol{w} \in \mathbb{R}^d} \sum_{i=1}^{n} \log \Pr\left(\boldsymbol{x}^{(i)}, y^{(i)} | \boldsymbol{w}\right)$$

$$= \arg\min_{\boldsymbol{w} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)}\right)^2$$

MLE Coincides with Least Squares

Assuming each sample in $S = \left\{(\boldsymbol{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n}$ is drawn iid,

$$\begin{aligned}\boldsymbol{w}_S^{\mathrm{MLE}} &:= \arg\max_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i=1}^{n} \log \Pr\left(\boldsymbol{x}^{(i)}, y^{(i)} | \boldsymbol{w}\right) \\ &= \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w}\cdot\boldsymbol{x}^{(i)}\right)^2 = \boldsymbol{w}_S^{\mathrm{LS}}\end{aligned}$$

Overview

Regularization

A Probabilistic Interpretation of Linear Regression

Optimization by Local Search

Need for a Universal Optimization Technique

- In the case of linear regression, the optimal model on a training dataset is given in closed-form by $\boldsymbol{w}_{S}^{\mathrm{LS}} = X^{+}\boldsymbol{y}$.

Need for a Universal Optimization Technique

- In the case of linear regression, the optimal model on a training dataset is given in closed-form by $\boldsymbol{w}_S^{\mathrm{LS}} = X^+\boldsymbol{y}$.
- This **almost never happens** with a real world objective.

Need for a Universal Optimization Technique

- In the case of linear regression, the optimal model on a training dataset is given in closed-form by $\boldsymbol{w}_S^{\mathrm{LS}} = X^+\boldsymbol{y}$.

- This **almost never happens** with a real world objective.

We need a **general**, **efficient** optimization techinique that can be used for a wide class of models and objectives!

Local Search

Input: training objective* $J(\theta) \in \mathbb{R}$, number of iterations T
Output: parameter $\hat{\theta} \in \mathbb{R}^d$ such that $J(\hat{\theta})$ is small

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \ldots T-1$,
   2.1 Obtain $\Delta^t \in \mathbb{R}^n$ such that $J(\theta^t + \Delta^t) \leq J(\theta^t)$.
   2.2 Choose some "step size" $\eta^t \in \mathbb{R}$.
   2.3 Set $\theta^{t+1} = \theta^t + \eta^t \Delta^t$.
3. Return θ^T.

*Assumed to be differentiable in this lecture.

Local Search

Input: training objective* $J(\theta) \in \mathbb{R}$, number of iterations T
Output: parameter $\hat{\theta} \in \mathbb{R}^d$ such that $J(\hat{\theta})$ is small

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \ldots T - 1$,
   2.1 Obtain $\Delta^t \in \mathbb{R}^n$ such that $J(\theta^t + \Delta^t) \leq J(\theta^t)$.
   2.2 Choose some "step size" $\eta^t \in \mathbb{R}$.
   2.3 Set $\theta^{t+1} = \theta^t + \eta^t \Delta^t$.
3. Return θ^T.

What is a good Δ^t?

*Assumed to be differentiable in this lecture.

Gradient of the Objective at the Current Parameter

At $\theta^t \in \mathbb{R}^n$, the rate of increase (of the value of J) along a direction $u \in \mathbb{R}^n$ (i.e., $||u||_2 = 1$) is given by the **directional derivative**

$$\nabla_u J(\theta^t) := \lim_{\epsilon \to 0} \frac{J(\theta^t + \epsilon u) - J(\theta^t)}{\epsilon}$$

Gradient of the Objective at the Current Parameter

At $\theta^t \in \mathbb{R}^n$, the rate of increase (of the value of J) along a direction $u \in \mathbb{R}^n$ (i.e., $||u||_2 = 1$) is given by the **directional derivative**

$$\nabla_u J(\theta^t) := \lim_{\epsilon \to 0} \frac{J(\theta^t + \epsilon u) - J(\theta^t)}{\epsilon}$$

The **gradient** of J at θ^t is defined to be a vector $\nabla J(\theta^t)$ such that

$$\nabla_u J(\theta^t) = \nabla J(\theta^t) \cdot u \qquad\qquad \forall u \in \mathbb{R}^n$$

Gradient of the Objective at the Current Parameter

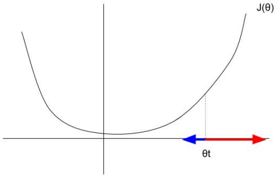
At $\theta^t \in \mathbb{R}^n$, the rate of increase (of the value of J) along a direction $u \in \mathbb{R}^n$ (i.e., $||u||_2 = 1$) is given by the **directional derivative**

$$\nabla_u J(\theta^t) := \lim_{\epsilon \to 0} \frac{J(\theta^t + \epsilon u) - J(\theta^t)}{\epsilon}$$

The **gradient** of J at θ^t is defined to be a vector $\nabla J(\theta^t)$ such that

$$\nabla_u J(\theta^t) = \nabla J(\theta^t) \cdot u \qquad\qquad \forall u \in \mathbb{R}^n$$

Therefore, the direction of the greatest rate of *decrease* is given by $-\nabla J(\theta^t) / \left|\left|\nabla J(\theta^t)\right|\right|_2$.



Gradient Descent

Input: training objective $J(\theta) \in \mathbb{R}$, number of iterations T
Output: parameter $\hat{\theta} \in \mathbb{R}^n$ such that $J(\hat{\theta})$ is small

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \ldots T-1$,

$$\theta^{t+1} = \theta^t - \eta^t \nabla J(\theta^t)$$

3. Return θ^T.

Gradient Descent

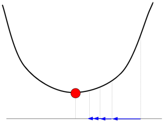
Input: training objective $J(\theta) \in \mathbb{R}$, number of iterations T
Output: parameter $\hat{\theta} \in \mathbb{R}^n$ such that $J(\hat{\theta})$ is small

1. Initialize θ^0 (e.g., randomly).
2. For $t = 0 \ldots T-1$,

$$\theta^{t+1} = \theta^t - \eta^t \nabla J(\theta^t)$$

3. Return θ^T.

When $J(\theta)$ is additionally *convex* (as in linear regression), gradient descent converges to an optimal solution (for appropriate step sizes).



Gradient Descent for Linear Regression

Input: training objective

$$J_S^{\mathrm{LS}}(\boldsymbol{w}) := \frac{1}{2} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)} \right)^2$$

number of iterations T

Output: parameter $\hat{\boldsymbol{w}} \in \mathbb{R}^n$ such that $J_S^{\mathrm{LS}}(\hat{\boldsymbol{w}}) \approx J_S^{\mathrm{LS}}(\boldsymbol{w}_S^{\mathrm{LS}})$

1. Initialize $\boldsymbol{w}^0$ (e.g., randomly).
2. For $t = 0 \ldots T - 1$,

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta^t \sum_{i=1}^{n} \boldsymbol{x}^{(i)} \cdot \left(y^{(i)} - \boldsymbol{w}^t \cdot \boldsymbol{x}^{(i)} \right)$$

3. Return $\boldsymbol{w}^T$.

Summary

- **Regularization** is an effort to prevent **overfitting** and optimize the true/population error.
- We can endow an alternative probabilistic interpretation of linear regression as MLE.
- **Gradient descent** is a local search algorithm that can be used to optimize *any* differentiable objective function.
  - A variant called "stochastic" gradient descent is the cornerstone of modern large-scale machine learning.