Overview of Machine Learning

in particular, Supervised Learning

Chris Cornwell

Mar 13, 2025

Outline

Machine Learning

Supervised learning

First look at Gradient Descent

Outline

Machine Learning

Supervised learning

First look at Gradient Descent

What is Machine Learning?

Definition by Tom Mitchell:

> *A "computer program" is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

What is Machine Learning?

Definition by Tom Mitchell:

> *A "computer program" is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

- The definition is intentionally general. Often, could think of E as "training" (updates to how program runs), based on observed data.

What is Machine Learning?

Definition by Tom Mitchell:

> *A "computer program" is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

- The definition is intentionally general. Often, could think of E as "training" (updates to how program runs), based on observed data.
- "computer program," for us, means a function implemented on a computer that produces output from given input. The output is how the program achieves the task T.

What is Machine Learning?

Definition by Tom Mitchell:

> *A "computer program" is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

- The definition is intentionally general. Often, could think of E as "training" (updates to how program runs), based on observed data.
- "computer program," for us, means a function implemented on a computer that produces output from given input. The output is how the program achieves the task T.
- The procedures discussed in class – linear regression and the Perceptron algorithm for half-space model – fit into this paradigm...*kind of*.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
    - Output of $\hat{y}$ on input x (potentially multiple variables).

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
   - Output of $\hat{y}$ on input x (potentially multiple variables).
   - T: fit observed points $\{(x_i, y_i)\}_{i=1}^{n}$ well with predictions $\{(x_i, \hat{y}_i)\}$, where $\hat{y}_i = mx_i + b$ for some m, b (an expectation of $(x, \hat{y})$ being good fit on *unobserved* data.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
    - Output of $\hat{y}$ on input x (potentially multiple variables).
    - T: fit observed points $\{(x_i, y_i)\}_{i=1}^{n}$ well with predictions $\{(x_i, \hat{y}_i)\}$, where $\hat{y}_i = mx_i + b$ for some m, b (an expectation of $(x, \hat{y})$ being good fit on *unobserved* data.
    - E: ??

      The data are used to get m and b, but you don't really "improve" with repeated use of data.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
   - Output of $\hat{y}$ on input x (potentially multiple variables).
   - T: fit observed points $\{(x_i, y_i)\}_{i=1}^{n}$ well with predictions $\{(x_i, \hat{y}_i)\}$, where $\hat{y}_i = mx_i + b$ for some m, b (an expectation of $(x, \hat{y})$ being good fit on *unobserved* data.
   - E: ??

     The data are used to get m and b, but you don't really "improve" with repeated use of data.

     Closed form for best choice of m, b, computing $(A^T A)^{-1} A^T \mathbf{y}$.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

1. Linear regression.
    - Output of $\hat{y}$ on input x (potentially multiple variables).
    - T: fit observed points $\{(x_i, y_i)\}_{i=1}^{n}$ well with predictions $\{(x_i, \hat{y}_i)\}$, where $\hat{y}_i = mx_i + b$ for some m, b (an expectation of $(x, \hat{y})$ being good fit on *unobserved* data.
    - E: ??

      The data are used to get m and b, but you don't really "improve" with repeated use of data.

      Closed form for best choice of m, b, computing $(A^T A)^{-1} A^T \mathbf{y}$.
    - P: Mean squared error.

Having closed form, result of simplicity of the form of $\hat{y}$

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
   - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
    - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).
    - T: predict labels correctly, using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label, $y = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)$

      ...hopefully works on *unobserved* data.

What is Machine Learning?

Definition by Tom Mitchell:

> *A computer program is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
    - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).
    - T: predict labels correctly, using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label, $y = \operatorname{sign}(\mathbf{w} \cdot \mathbf{x} + b)$
      ...hopefully works on *unobserved* data.
    - E: looking through observed data $X_i = (\mathbf{x}_i, 1)$, label y_i, and updating $W^{(t+1)} = W^{(t)} + y_i X_i$ when i found with $W^{(t)} \cdot (y_i X_i) \leq 0$.

Definition by Tom Mitchell:

> *A computer program is said to **learn** from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
   - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).
   - T: predict labels correctly, using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label, $y = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)$

     ...hopefully works on *unobserved* data.
   - E: looking through observed data $X_i = (\mathbf{x}_i, 1)$, label y_i, and updating $W^{(t+1)} = W^{(t)} + y_i X_i$ when i found with $W^{(t)} \cdot (y_i X_i) \leq 0$.
   - P: ??

     Whether its labels on all observed data are correct. But, only two results: *True* or *False*.

Definition by Tom Mitchell:

> *A computer program is said to* ***learn*** *from experience E, with respect to some task T and performance measure P if: its performance on T, as measured by P, improves with experience E.*

Examples:

2. The Perceptron algorithm.
    - Output of label ± 1 on input $\mathbf{x} \in \mathbb{R}^d$ (or something *turned into* $\mathbf{x} \in \mathbb{R}^d$).
    - T: predict labels correctly, using $W = (\mathbf{w}, b) \in \mathbb{R}^{d+1}$ to decide label, $y = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + b)$
      ...hopefully works on *unobserved* data.
    - E: looking through observed data $X_i = (\mathbf{x}_i, 1)$, label y_i, and updating $W^{(t+1)} = W^{(t)} + y_i X_i$ when i found with $W^{(t)} \cdot (y_i X_i) \leq 0$.
    - P: ??
      Whether its labels on all observed data are correct. But, only two results: *True* or *False*.

What are the general types of tasks in machine learning?

Supervised learning: the program learns from sample data that has labels. Goal: determine underlying function from sample data.

What are the general types of tasks in machine learning?

Supervised learning: the program learns from sample data that has labels. Goal: determine underlying function from sample data.

Examples.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Determine if a satellite image of ocean has floating trash.
- Try to auto-complete a sentence being typed.

What are the general types of tasks in machine learning?

Supervised learning: the program learns from sample data that has labels. Goal: determine underlying function from sample data.

Examples.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Determine if a satellite image of ocean has floating trash.
- Try to auto-complete a sentence being typed.

Unsupervised learning: there is sample data, but the data does not have any labels. Goal: discover something (a pattern, grouping, or some insight) about the data.

What are the general types of tasks in machine learning?

Supervised learning: the program learns from sample data that has labels. Goal: determine underlying function from sample data.

Examples.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Determine if a satellite image of ocean has floating trash.
- Try to auto-complete a sentence being typed.

Unsupervised learning: there is sample data, but the data does not have any labels. Goal: discover something (a pattern, grouping, or some insight) about the data.

Examples.

- Market segmentation.
- News feed (grouping similar news articles).
- Separate audio sources in a mixed signal.

Outline

The goal of supervised learning

Have an “input space” (which often is $\mathbb{R}^d$, or a subset of it, but could be a different space); and have an output space, or label space, Y.

The goal of supervised learning

Have an "input space" (which often is $\mathbb{R}^d$, or a subset of it, but could be a different space); and have an output space, or label space, Y.

- Given a sample $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in Y$, drawn from an (unknown) joint probability distribution $P_{X,Y} : \mathbb{R}^d \times Y \longrightarrow [0, \infty)$.

The goal of supervised learning

Have an "input space" (which often is $\mathbb{R}^d$, or a subset of it, but could be a different space); and have an output space, or label space, Y.

- Given a sample $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in Y$, drawn from an (unknown) joint probability distribution $P_{X,Y} : \mathbb{R}^d \times Y \longrightarrow [0, \infty)$.
- Goal: to learn, from $\mathcal{S}$, a function $f^* : \mathbb{R}^d \longrightarrow Y$ that "fits" (*approximates well*) the distribution $P_{X,Y}$.

The goal of supervised learning

Have an “input space” (which often is $\mathbb{R}^d$, or a subset of it, but could be a different space); and have an output space, or label space, Y.

- Given a sample $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in Y$, drawn from an (unknown) joint probability distribution $P_{X,Y} : \mathbb{R}^d \times Y \longrightarrow [0, \infty)$.
- Goal: to learn, from $\mathcal{S}$, a function $f^* : \mathbb{R}^d \longrightarrow Y$ that “fits” (*approximates well*) the distribution $P_{X,Y}$.
- You might not be able to have points on the graph of f^* be typically “very close” to samples from $P_{X,Y}$. However, ideally, for an $\mathbf{x} \in \mathbb{R}^d$ corresponding y-value on graph is near the expected value given $\mathbf{x}$.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

[1]Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.

[1] Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.
- To learn a function that fits well, you look for good parameters.

[1] Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.
- To learn a function that fits well, you look for good parameters.

How do we find good parameters?

Select a performance measure: **(empirical) loss function** $\mathcal{L}_\mathcal{S} : \Omega \longrightarrow \mathbb{R}$. In the empirical loss function, we use $\mathcal{S}$ in its definition.

[1]Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.
- To learn a function that fits well, you look for good parameters.

How do we find good parameters?

Select a performance measure: **(empirical) loss function** $\mathcal{L}_\mathcal{S} : \Omega \longrightarrow \mathbb{R}$. In the empirical loss function, we use $\mathcal{S}$ in its definition.

[1]Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.
- To learn a function that fits well, you look for good parameters.

How do we find good parameters?

Select a performance measure: **(empirical) loss function** $\mathcal{L}_{\mathcal{S}} : \Omega \to \mathbb{R}$. In the empirical loss function, we use $\mathcal{S}$ in its definition.

- Then, $\mathcal{L}_{\mathcal{S}}$ is used to determine how to make changes to parameters, ω, in order to decrease the value of $\mathcal{L}_{\mathcal{S}}$.

[1] Sometimes called a *hypothesis class*.

How to achieve the goal

Most often, we choose a *parameterized class* of functions[1], and we get f^* from that class.

- That is, there is a space of parameters Ω; an $\omega \in \Omega$ determines a function $f_\omega : \mathbb{R}^d \longrightarrow Y$, and the parameterized class is the set of all such functions f_ω.
- To learn a function that fits well, you look for good parameters.

How do we find good parameters?

Select a performance measure: **(empirical) loss function** $\mathcal{L}_\mathcal{S} : \Omega \longrightarrow \mathbb{R}$. In the empirical loss function, we use $\mathcal{S}$ in its definition.

- Then, $\mathcal{L}_\mathcal{S}$ is used to determine how to make changes to parameters, ω, in order to decrease the value of $\mathcal{L}_\mathcal{S}$.
- In an ideal situation, you converge to some ω^*, a minimizer of $\mathcal{L}_\mathcal{S}$, and set $f^* = f_{\omega^*}$.

[1]Sometimes called a *hypothesis class*.

For linear regression

Have sample data $\mathcal{S}$, with data points x_i in $\mathbb{R}$ (so, $d = 1$). The parameter space $\Omega = \mathbb{R}^2 = \{(m, b) \mid m \in \mathbb{R}, b \in \mathbb{R}\}$. For each $\omega = (m, b)$, we have

$$f_\omega(x) = mx + b.$$

Have sample data $\mathcal{S}$, with data points x_i in $\mathbb{R}$ (so, $d = 1$). The parameter space $\Omega = \mathbb{R}^2 = \{(m, b) \mid m \in \mathbb{R}, b \in \mathbb{R}\}$. For each $\omega = (m, b)$, we have

$$f_\omega(x) = mx + b.$$

Loss function: the MSE. That is, set

$$\mathcal{L}_{\mathcal{S}}(m, b) = \frac{1}{n}\sum_{i=1}^{n}(mx_i + b - y_i)^2.$$

Outline

Gradient descent with simple linear regression

For $\omega = (m, b)$, have $f_\omega(x) = mx + b$. Given sample data $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, note that the empirical loss function $\mathcal{L}_\mathcal{S}$ is a function of m and b (while $\mathcal{S}$ is *used* in its definition, the points $\mathbf{x}_i$ are not inputs to $\mathcal{L}_\mathcal{S}$).

Gradient descent with simple linear regression

For $\omega = (m, b)$, have $f_\omega(x) = mx + b$. Given sample data $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, note that the empirical loss function $\mathcal{L}_\mathcal{S}$ is a function of m and b (while $\mathcal{S}$ is *used* in its definition, the points $\mathbf{x}_i$ are not inputs to $\mathcal{L}_\mathcal{S}$).

Recall the definition $\mathcal{L}_\mathcal{S}(m, b) = \frac{1}{n}\sum_{i=1}^n (mx_i + b - y_i)^2$.

- The **gradient** of $\mathcal{L}_\mathcal{S}$ is the vector of partial derivatives: $\nabla\mathcal{L}_\mathcal{S} = \left(\frac{d}{dm}\mathcal{L}_\mathcal{S}, \frac{d}{db}\mathcal{L}_\mathcal{S}\right)$.

Gradient descent with simple linear regression

For $\omega = (m, b)$, have $f_\omega(x) = mx + b$. Given sample data $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, note that the empirical loss function $\mathcal{L}_\mathcal{S}$ is a function of m and b (while $\mathcal{S}$ is *used* in its definition, the points $\mathbf{x}_i$ are not inputs to $\mathcal{L}_\mathcal{S}$).

Recall the definition $\mathcal{L}_\mathcal{S}(m, b) = \frac{1}{n}\sum_{i=1}^n (mx_i + b - y_i)^2$.

- The **gradient** of $\mathcal{L}_\mathcal{S}$ is the vector of partial derivatives: $\nabla\mathcal{L}_\mathcal{S} = \left(\frac{d}{dm}\mathcal{L}_\mathcal{S}, \frac{d}{db}\mathcal{L}_\mathcal{S}\right)$.
- Get partial derivatives using the Chain rule:

$$\frac{d}{dm}\mathcal{L}_\mathcal{S} = \frac{2}{n}\sum_{i=1}^n (mx_i + b - y_i)x_i;$$

and

$$\frac{d}{db}\mathcal{L}_\mathcal{S} = \frac{2}{n}\sum_{i=1}^n (mx_i + b - y_i).$$

Aside: Recovering the normal equations

By utilizing the fact that a minimum of $\mathcal{L}_{\mathcal{S}}$ only occurs when $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ and $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$, we can recover the normal equations.

Aside: Recovering the normal equations

By utilizing the fact that a minimum of $\mathcal{L}_{\mathcal{S}}$ only occurs when $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ and $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$, we can recover the normal equations.

To simplify it (and still be able to generalize), say that we have $n = 3$ (in other words, $\mathcal{S}$ has just three points). Then, setting $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$, and $\bar{x}$ equal to the average of x_1, x_2, x_3,

Aside: Recovering the normal equations

By utilizing the fact that a minimum of $\mathcal{L}_{\mathcal{S}}$ only occurs when $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ and $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$, we can recover the normal equations.

To simplify it (and still be able to generalize), say that we have $n = 3$ (in other words, $\mathcal{S}$ has just three points). Then, setting $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$, and $\bar{x}$ equal to the average of x_1, x_2, x_3,

$$\begin{aligned}
\frac{d}{dm}\mathcal{L}_{\mathcal{S}} &= \frac{2}{3}\left((mx_1^2 + bx_1 - x_1y_1) + (mx_2^2 + bx_2 - x_2y_2) + (mx_3^2 + bx_3 - x_3y_3)\right) \\
&= \frac{2}{3}\left(m(x_1^2 + x_2^2 + x_3^2) + b(x_1 + x_2 + x_3) - (x_1y_1 + x_2y_2 + x_3y_3)\right) \\
&= \frac{2}{3}\left(m\mathbf{x}\cdot\mathbf{x} + b(3\bar{x}) - \mathbf{x}\cdot\mathbf{y}\right).
\end{aligned}$$

Aside: Recovering the normal equations

By utilizing the fact that a minimum of $\mathcal{L}_{\mathcal{S}}$ only occurs when $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ and $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$, we can recover the normal equations.

To simplify it (and still be able to generalize), say that we have $n = 3$ (in other words, $\mathcal{S}$ has just three points). Then, setting $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$, and $\bar{x}$ equal to the average of x_1, x_2, x_3,

$$\begin{aligned}\frac{d}{dm}\mathcal{L}_{\mathcal{S}} &= \frac{2}{3}\left((mx_1^2 + bx_1 - x_1y_1) + (mx_2^2 + bx_2 - x_2y_2) + (mx_3^2 + bx_3 - x_3y_3)\right) \\ &= \frac{2}{3}\left(m(x_1^2 + x_2^2 + x_3^2) + b(x_1 + x_2 + x_3) - (x_1y_1 + x_2y_2 + x_3y_3)\right) \\ &= \frac{2}{3}\left(m\mathbf{x}\cdot\mathbf{x} + b(3\bar{x}) - \mathbf{x}\cdot\mathbf{y}\right).\end{aligned}$$

And so, setting $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ amounts to the equation $m(\mathbf{x}\cdot\mathbf{x}) + b(3\bar{x}) = \mathbf{x}\cdot\mathbf{y}$.

Aside: Recovering the normal equations

By utilizing the fact that a minimum of $\mathcal{L}_{\mathcal{S}}$ only occurs when $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ and $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$, we can recover the normal equations.

To simplify it (and still be able to generalize), say that we have $n = 3$ (in other words, $\mathcal{S}$ has just three points). Then, setting $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$, and $\bar{x}$ equal to the average of x_1, x_2, x_3,

$$\begin{aligned}
\frac{d}{dm}\mathcal{L}_{\mathcal{S}} &= \frac{2}{3}\left((mx_1^2 + bx_1 - x_1y_1) + (mx_2^2 + bx_2 - x_2y_2) + (mx_3^2 + bx_3 - x_3y_3)\right) \\
&= \frac{2}{3}\left(m(x_1^2 + x_2^2 + x_3^2) + b(x_1 + x_2 + x_3) - (x_1y_1 + x_2y_2 + x_3y_3)\right) \\
&= \frac{2}{3}\left(m\mathbf{x}\cdot\mathbf{x} + b(3\bar{x}) - \mathbf{x}\cdot\mathbf{y}\right).
\end{aligned}$$

And so, setting $\frac{d}{dm}\mathcal{L}_{\mathcal{S}} = 0$ amounts to the equation $m(\mathbf{x}\cdot\mathbf{x}) + b(3\bar{x}) = \mathbf{x}\cdot\mathbf{y}$.

A similar computation will show that setting $\frac{d}{db}\mathcal{L}_{\mathcal{S}} = 0$ will give the equation $m(3\bar{x}) + b(3) = 3\bar{y}$. (With $\bar{y}$ being the average of y_1, y_2, y_3).

Aside: Recovering the normal equations

The computation above generalizes to imply that
$\nabla \mathcal{L}_{\mathcal{S}} = (\frac{d}{dm}\mathcal{L}_{\mathcal{S}}, \frac{d}{db}\mathcal{L}_{\mathcal{S}}) = (0,0)$ requires the equations

$$m(\mathbf{x} \cdot \mathbf{x}) + b(n\bar{x}) = \mathbf{x} \cdot \mathbf{y}$$
$$m(n\bar{x}) + b(n) = n\bar{y}.$$

Aside: Recovering the normal equations

The computation above generalizes to imply that $\nabla \mathcal{L}_{\mathcal{S}} = (\frac{d}{dm}\mathcal{L}_{\mathcal{S}}, \frac{d}{db}\mathcal{L}_{\mathcal{S}}) = (0, 0)$ requires the equations

$$m(\mathbf{x} \cdot \mathbf{x}) + b(n\bar{x}) = \mathbf{x} \cdot \mathbf{y}$$
$$m(n\bar{x}) + b(n) = n\bar{y}.$$

If you recall the entries in $A^T A$ and $A^T \mathbf{y}$ (where A is the matrix built in the simple linear regression procedure), these are precisely the normal equations.

Solving for m and b gives us $m = \frac{\mathbf{x} \cdot \mathbf{y} - n\bar{x}\bar{y}}{\mathbf{x} \cdot \mathbf{x} - n\bar{x}^2} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$, and $b = \bar{y} - m\bar{x}$.

Aside: Recovering the normal equations

The computation above generalizes to imply that $\nabla\mathcal{L}_{\mathcal{S}} = (\frac{d}{dm}\mathcal{L}_{\mathcal{S}}, \frac{d}{db}\mathcal{L}_{\mathcal{S}}) = (0,0)$ requires the equations

$$m(\mathbf{x}\cdot\mathbf{x}) + b(n\bar{x}) = \mathbf{x}\cdot\mathbf{y}$$
$$m(n\bar{x}) + b(n) = n\bar{y}.$$

If you recall the entries in A^TA and $A^T\mathbf{y}$ (where A is the matrix built in the simple linear regression procedure), these are precisely the normal equations.

Solving for m and b gives us $m = \frac{\mathbf{x}\cdot\mathbf{y}-n\bar{x}\bar{y}}{\mathbf{x}\cdot\mathbf{x}-n\bar{x}^2} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$, and $b = \bar{y} - m\bar{x}$.

- We are able to nicely represent the minimizer of $\mathcal{L}_{\mathcal{S}}$ precisely because of the linear nature of the class of functions $f_\omega(x) = mx + b$.

Returning to Gradient Descent

In anticipation that, in other settings, we not be able to nicely represent a minimizer of $\mathcal{L}_{\mathcal{S}}$, we consider another optimization approach.

Returning to Gradient Descent

In anticipation that, in other settings, we not be able to nicely represent a minimizer of $\mathcal{L}_{\mathcal{S}}$, we consider another optimization approach.

- Say that the (current) value of ω is (m_0, b_0). Then, recalling from Calculus III, the direction of *steepest descent*, that will produce the most rapid decrease in the value of $\mathcal{L}_{\mathcal{S}}$, is the direction of $-\nabla\mathcal{L}_{\mathcal{S}}(m_0, b_0)$.
- This indicates that we might be able to get closer to a minimizer by subtracting the gradient from (m_0, b_0) or, to make our step "small" perhaps, subtracting a small multiple of the gradient.

Returning to Gradient Descent

In anticipation that, in other settings, we not be able to nicely represent a minimizer of $\mathcal{L}_{\mathcal{S}}$, we consider another optimization approach.

- Say that the (current) value of ω is (m_0, b_0). Then, recalling from Calculus III, the direction of *steepest descent*, that will produce the most rapid decrease in the value of $\mathcal{L}_{\mathcal{S}}$, is the direction of $-\nabla\mathcal{L}_{\mathcal{S}}(m_0, b_0)$.
- This indicates that we might be able to get closer to a minimizer by subtracting the gradient from (m_0, b_0) or, to make our step "small" perhaps, subtracting a small multiple of the gradient.

Gradient descent: Choosing a constant $\eta > 0$ and given some current value of $\omega_i = (m_i, b_i)$, we attempt to get closer to the minimizer, ω^*, of the loss function by the update

$$\omega_{i+1} = \omega_i - \eta * \nabla\mathcal{L}_{\mathcal{S}}(m_i, b_i).$$

The constant η is called the **learning rate**.

Sources

The content of these slides has been combined from two references.

1. Notes taken from Machine Learning course, taught by Andrew Ng, Stanford U.
2. Notes from a lecture series on Deep Learning at Harvard, taught by Eli Grigsby.

Questions?