Overview of Machine Learning

with focus on Supervised Learning

Chris Cornwell

Mar 13, 2025

Outline

Outline

Machine Learning

Supervised learning

First look at Gradient Descent

What is Machine Learning?

- Concerned with designing, understanding algorithms which allow computer program to “learn.”

What is Machine Learning?

- Concerned with designing, understanding algorithms which allow computer program to "learn."
- Not as new as it seems, but rapid growth in last 2 decades.
  - field of study since 1950's;
  - related to much older statistical modeling.

What is Machine Learning?

- Concerned with designing, understanding algorithms which allow computer program to “learn.”
- Not as new as it seems, but rapid growth in last 2 decades.
  - field of study since 1950’s;
  - related to much older statistical modeling.
- Can be single machine learning model to perform a task (e.g., finding person in an image, speech-to-text, sentiment analysis)

What is Machine Learning?

- Concerned with designing, understanding algorithms which allow computer program to “learn.”
- Not as new as it seems, but rapid growth in last 2 decades.
    - field of study since 1950’s;
    - related to much older statistical modeling.
- Can be single machine learning model to perform a task (e.g., finding person in an image, speech-to-text, sentiment analysis)
- Or, many separate models combined together ⟵ makes AI function (e.g., self-driving cars, LLM’s or chatbots).

What is Machine Learning?

Very general, academic definition by Tom Mitchell:

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

What is Machine Learning?

Very general, academic definition by Tom Mitchell:

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

- Often, the experience *E* is called "training" (updates to how program runs); based on observed data.

What is Machine Learning?

Very general, academic definition by Tom Mitchell:

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

- Often, the experience E is called "training" (updates to how program runs); based on observed data.
- "computer program," for us ("learning algorithm"), will determine a function that produces output from given input (the data). The output is how the program achieves the task T.

What is Machine Learning?

Very general, academic definition by Tom Mitchell:

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

- Often, the experience *E* is called "training" (updates to how program runs); based on observed data.
- "computer program," for us ("learning algorithm"), will determine a function that produces output from given input (the data). The output is how the program achieves the task *T*.
- In class, we will discuss algorithms made for *regression* tasks, and others for *classification* tasks, that fit this paradigm.

What is Machine Learning?

Very general, academic definition by Tom Mitchell:

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

- Often, the experience *E* is called "training" (updates to how program runs); based on observed data.
- "computer program," for us ("learning algorithm"), will determine a function that produces output from given input (the data). The output is how the program achieves the task *T*.
- In class, we will discuss algorithms made for *regression* tasks, and others for *classification* tasks, that fit this paradigm.
- Performance measure *M*: for us, called a *cost function* or *loss function*.

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a <u>good</u> underlying function from sample data.

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Detect space debris, or trash on ocean surface.
- Auto-completion of typed sentence.

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Detect space debris, or trash on ocean surface.
- Auto-completion of typed sentence.

Unsupervised learning: algorithm uses sample data, but it is unlabeled. Goal: discover something (a pattern, grouping, or some insight) about the data based on its coordinates (features).

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Detect space debris, or trash on ocean surface.
- Auto-completion of typed sentence.

Unsupervised learning: algorithm uses sample data, but it is unlabeled. Goal: discover something (a pattern, grouping, or some insight) about the data based on its coordinates (features).

- Market segmentation.

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Detect space debris, or trash on ocean surface.
- Auto-completion of typed sentence.

Unsupervised learning: algorithm uses sample data, but it is unlabeled. Goal: discover something (a pattern, grouping, or some insight) about the data based on its coordinates (features).

- Market segmentation.
- News feed (grouping similar news articles).

General categories of tasks in machine learning?

Supervised learning: algorithm uses sample (input) data that have output "labels." Goal: determine a good underlying function from sample data.

- Housing price prediction
- Whether emails are phishing or not phishing.
- Detect space debris, or trash on ocean surface.
- Auto-completion of typed sentence.

Unsupervised learning: algorithm uses sample data, but it is unlabeled. Goal: discover something (a pattern, grouping, or some insight) about the data based on its coordinates (features).

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

Outline

Machine Learning

Supervised learning

First look at Gradient Descent

The goal of Supervised learning

- (Supervised $\iff$ labels)
- The sample data has labels – called **training data**. The goal is to "learn" a function from the training data that will do well labeling new data, not seen during learning process.
- "Doing well" is measured by a loss function (*M* from Mitchell's description).

Example Images

Example 1 - Supervised learning, Section 1.1 of textbook

Cat or Dog?

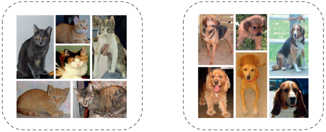

Figure 1.1 from textbook

- Training data. The images, each with label ’cat’ or ’dog’.
- Computer sees each image, pixels in 2D array with RGB value (a vector in $\mathbb{R}^3$) at each pixel.
- **Designing features.** “Cartoon image” of ML model’s function: computes N **features** from each image $\rightsquigarrow$ vectors (points) in $\mathbb{R}^N$ $\rightsquigarrow$ points with one label separated from those with other label (by graph of linear function).

Designing features, to easily separate data

Compute N **features** from each image $\rightsquigarrow$ vectors (points) in $\mathbb{R}^N$ $\rightsquigarrow$ points with one label separated from those with other label (by graph of linear function).

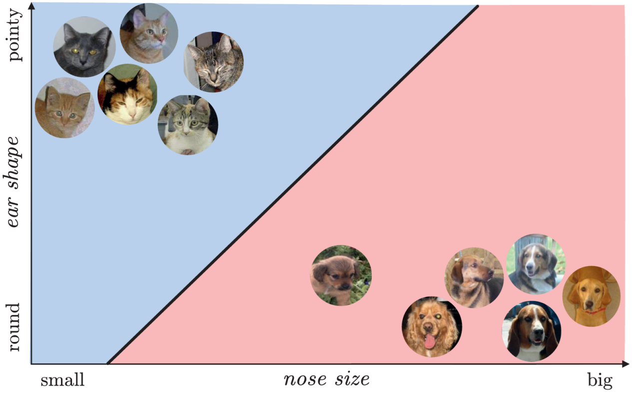

Figure 1.3 from textbook

Example 2 - Supervised learning, Section 1.2 of textbook

Predict share price

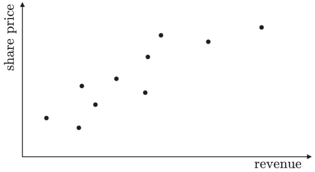

Figure 1.7, upper-left from textbook

- Training data. Revenues of ten corporations, each with label, the share price.
- Each revenue is simple number in $\mathbb{R}$. One "independent variable," call it x.
- **Designing features.** In this example, use feature (revenue) already at hand. Not always what you want to do when there are multiple independent variables (we'll see examples later, where you design

Example 2 - Predict response variable, share price

Revenue value: x. Find a function $f(x) = \hat{m}x + \hat{b}$, so that if y is share price for x and we set $\hat{y} = f(x)$ then y and $\hat{y}$ are close, on average. (Linear Regression)

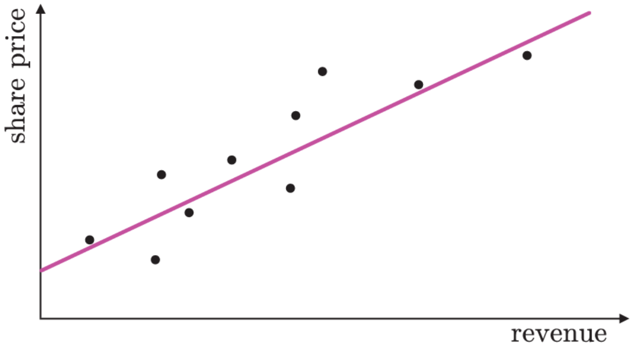

Figure 1.7, upper-right from textbook

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** $\mathcal{L}_\mathcal{S} : \Omega \to \mathbb{R}$. 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 $\mathcal{L}_\mathcal{S}$.
- 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

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

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

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

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

Machine Learning

Supervised learning

First look at Gradient Descent

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 $A^T A$ 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 $b = \bar{y} - m\bar{x}$.

- 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.

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?