Dive into Deep learning

2.4 ~ 2.7

Index

2.4. Calculus

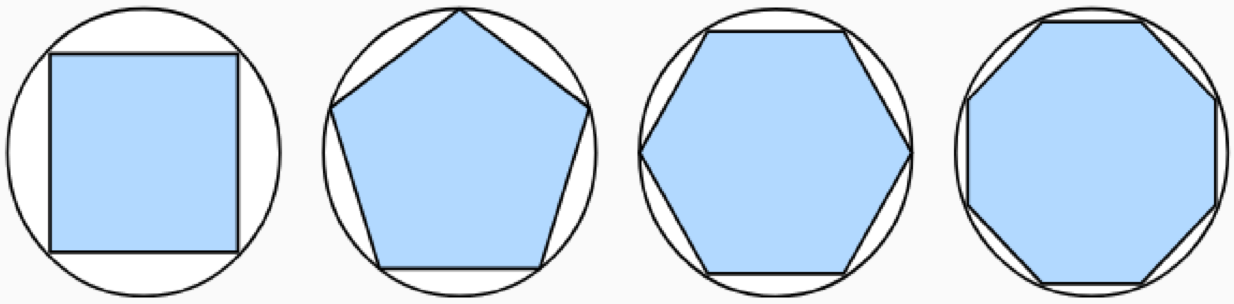


Fig. 2.4.1 Find the area of a circle with the method of exhaustion.

- Minimizing a loss function = score that answers the question “how bad is our model?”
- the task of fitting models

  i) optimization: the process of fitting our models to observed data

  ii) generalization: produce models whose validity extends beyond the exact set of data examples used to train them.

2.4.1. Derivatives and Differentiation

loss functions that are differentiable with respect to our model's parameters

Suppose that we have a function $f:\mathbb{R}\rightarrow\mathbb{R}$, whose input and output are both scalars. The *derivative* of f is defined as

$$f'(x)=\lim_{h\rightarrow 0}\frac{f(x+h)-f(x)}{h},$$

2.4.1. Derivatives and Differentiation

Let us familiarize ourselves with a few equivalent notations for derivatives. Given $y = f(x)$, where x and y are the independent variable and the dependent variable of the function f, respectively. The following expressions are equivalent:

$$f'(x) = y' = \frac{dy}{dx} = \frac{df}{dx} = \frac{d}{dx}f(x) = Df(x) = D_x f(x), \tag{2.4.2}$$

$$\frac{d}{dx}[Cf(x)] = C\frac{d}{dx}f(x), \tag{2.4.3}$$

the *sum rule*

$$\frac{d}{dx}[f(x) + g(x)] = \frac{d}{dx}f(x) + \frac{d}{dx}g(x), \tag{2.4.4}$$

the *product rule*

$$\frac{d}{dx}[f(x)g(x)] = f(x)\frac{d}{dx}[g(x)] + g(x)\frac{d}{dx}[f(x)], \tag{2.4.5}$$

and the *quotient rule*

$$\frac{d}{dx}\left[\frac{f(x)}{g(x)}\right] = \frac{g(x)\frac{d}{dx}[f(x)] - f(x)\frac{d}{dx}[g(x)]}{[g(x)]^2}. \tag{2.4.6}$$

2.4.2. Partial Derivatives

Let $y = f(x_1, x_2, \ldots, x_n)$ be a function with n variables. The *partial derivative* of y with respect to its i^{th} parameter x_i is

$$\frac{\partial y}{\partial x_i} = \lim_{h \rightarrow 0} \frac{f(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots, x_n) - f(x_1, \ldots, x_i, \ldots, x_n)}{h}. \tag{2.4.7}$$

To calculate $\frac{\partial y}{\partial x_i}$, we can simply treat $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ as constants and calculate the derivative of y with respect to x_i. For notation of partial derivatives, the following are equivalent:

$$\frac{\partial y}{\partial x_i} = \frac{\partial f}{\partial x_i} = f_{x_i} = f_i = D_i f = D_{x_i} f. \tag{2.4.8}$$

2.4.3. Gradients

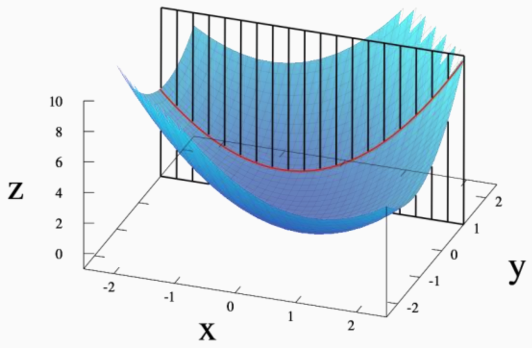
Suppose that the input of function $f:\mathbb{R}^n \rightarrow \mathbb{R}$ is an n-dimensional vector $\mathbf{x}=[x_1, x_2, \ldots, x_n]^\top$ and the output is a scalar. The gradient of the function $f(\mathbf{x})$ with respect to $\mathbf{x}$ is a vector of n partial derivatives:

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n}\right]^\top, \tag{2.4.9}$$

where $\nabla_{\mathbf{x}} f(\mathbf{x})$ is often replaced by $\nabla f(\mathbf{x})$ when there is no ambiguity.

Let $\mathbf{x}$ be an n-dimensional vector, the following rules are often used when differentiating multivariate functions:

- For all $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\nabla_{\mathbf{x}} \mathbf{A}\mathbf{x} = \mathbf{A}^\top$,
- For all $\mathbf{A} \in \mathbb{R}^{n \times m}$, $\nabla_{\mathbf{x}} \mathbf{x}^\top \mathbf{A} = \mathbf{A}$,
- For all $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\nabla_{\mathbf{x}} \mathbf{x}^\top \mathbf{A}\mathbf{x} = (\mathbf{A} + \mathbf{A}^\top)\mathbf{x}$,
- $\nabla_{\mathbf{x}} \|\mathbf{x}\|^2 = \nabla_{\mathbf{x}} \mathbf{x}^\top \mathbf{x} = 2\mathbf{x}$.



2.4.4. Chain Rule

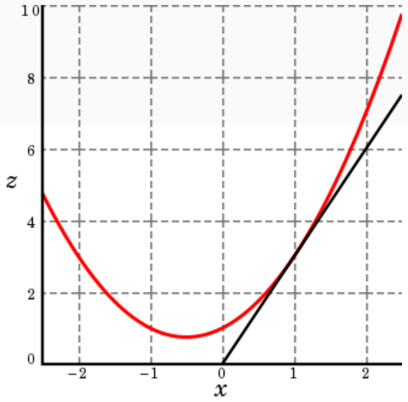
Let us first consider functions of a single variable. Suppose that functions $y = f(u)$ and $u = g(x)$ are both differentiable, then the chain rule states that

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}. \tag{2.4.10}$$

Now let us turn our attention to a more general scenario where functions have an arbitrary number of variables. Suppose that the differentiable function y has variables $u_1, u_2, \ldots, u_m$, where each differentiable function u_i has variables $x_1, x_2, \ldots, x_n$. Note that y is a function of $x_1, x_2, \ldots, x_n$. Then the chain rule gives

$$\frac{dy}{dx_i} = \frac{dy}{du_1}\frac{du_1}{dx_i} + \frac{dy}{du_2}\frac{du_2}{dx_i} + \cdots + \frac{dy}{du_m}\frac{du_m}{dx_i} \tag{2.4.11}$$

for any $i = 1, 2, \ldots, n$.



2.5. Automatic Differentiation

- 2.5.2. Backward for Non-Scalar Variables
  - Calling backward on a vector => Calculate the derivatives of the loss functions
  - To calculate the differentiation matrix (X)
  - the sum of the partial derivatives computed individually for each example in the batch. (O)

2.5. Automatic Differentiation

- 2.5.3. Detaching Computation
  - Detach y to return a new variable u that has the same value as y but discards any information about how y was computed in the computational graph.

MXNET PYTORCH TENSORFLOW

```
# Set `persistent=True` to run `t.gradient` more than once
with tf.GradientTape(persistent=True) as t:
    y = x * x
    u = tf.stop_gradient(y)
    z = u * x

x_grad = t.gradient(z, x)
x_grad == u
```

```
<tf.Tensor: shape=(4,), dtype=bool, numpy=array([ True,  True,  True,  True])>
```

Since the computation of `y` was recorded, we can subsequently invoke backpropagation on `y` to get the derivative of `y = x * x` with respect to `x`, which is `2 * x`.

2.6. Probability

Fig. 2.6.1 Images of varying resolutions (10×10, 20×20, 40×40, 80×80, and 160×160 pixels).

- it is easy for humans to recognize cats and dogs at the resolution of 160×160 pixels, it becomes challenging at 40×40 pixels and next to impossible at 10×10 pixels.
- Probability gives us a formal way of reasoning about our level of certainty

2.6. Probability

- 2.6.1. Basic Probability Theory
  Probability : individual count for that value / the total number.

Formally, *probability* can be thought of a function that maps a set to a real value. The probability of an event $\mathcal{A}$ in the given sample space $\mathcal{S}$, denoted as $P(\mathcal{A})$, satisfies the following properties:

- For any event $\mathcal{A}$, its probability is never negative, i.e., $P(\mathcal{A}) \geq 0$;
- Probability of the entire sample space is 1, i.e., $P(\mathcal{S}) = 1$;
- For any countable sequence of events $\mathcal{A}_1, \mathcal{A}_2, \ldots$ that are *mutually exclusive* ($\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ for all $i \neq j$), the probability that any happens is equal to the sum of their individual probabilities, i.e., $P(\bigcup_{i=1}^{\infty} \mathcal{A}_i) = \sum_{i=1}^{\infty} P(\mathcal{A}_i)$.

2.6. Probability

- **2.6.1.2. Random Variables**
  A random variable can be pretty much any quantity and is not deterministic.

- **2.6.2. Dealing with Multiple Random Variables**
  When we deal with multiple random variables, there are several quantities of interest

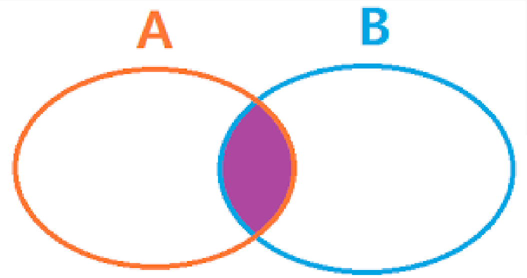
2.6. Probability

2.6.2.1. Joint Probability

The first is called the *joint probability* $P(A=a, B=b)$. Given any values a and b, the joint probability lets us answer, what is the probability that $A=a$ and $B=b$ simultaneously? Note that for any values a and b, $P(A=a, B=b) \leq P(A=a)$. This has to be the case, since for $A=a$ and $B=b$ to happen, $A=a$ has to happen *and* $B=b$ also has to happen (and vice versa). Thus, $A=a$ and $B=b$ cannot be more likely than $A=a$ or $B=b$ individually.

2.6.2.2. Conditional Probability

This brings us to an interesting ratio: $0 \leq \frac{P(A=a, B=b)}{P(A=a)} \leq 1$. We call this ratio a *conditional probability* and denote it by $P(B=b \mid A=a)$: it is the probability of $B=b$, provided that $A=a$ has occurred.



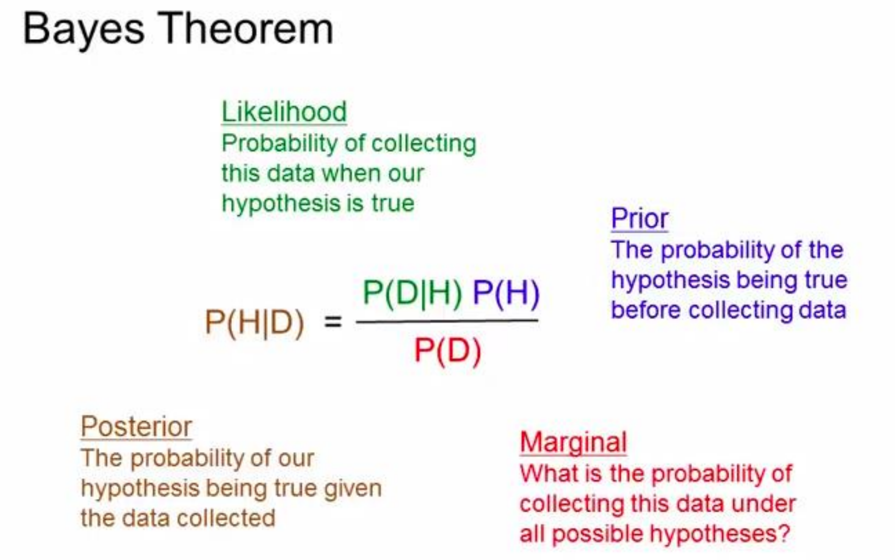
2.6. Probability

2.6.2.3. Bayes' theorem

Using the definition of conditional probabilities, we can derive one of the most useful and celebrated equations in statistics: *Bayes' theorem*. It goes as follows. By construction, we have the *multiplication rule* that $P(A, B) = P(B \mid A)P(A)$. By symmetry, this also holds for $P(A, B) = P(A \mid B)P(B)$. Assume that $P(B) > 0$. Solving for one of the conditional variables we get

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}. \tag{2.6.1}$$

Note that here we use the more compact notation where $P(A, B)$ is a *joint distribution* and $P(A \mid B)$ is a *conditional distribution*. Such distributions can be evaluated for particular values $A = a, B = b$.



2.6. Probability

2.6.2.4. Marginalization

- Probability value for one value :
  Probability of B amounts to accounting for A and aggregating the joint probabilities.

$$P(B) = \sum_{A} P(A, B), \tag{2.6.2}$$

which is also known as the *sum rule*. The probability or distribution as a result of marginalization is called a *marginal probability* or a *marginal distribution*.

2.6. Probability

- **2.6.2.5. Independence**
  Two random variables A and B being independent means:
  Occurrence of one event of A does not reveal any information about the occurrence of an event of B

random variable C if and only if $P(A, B \mid C) = P(A \mid C)P(B \mid C)$. This is expressed as $A \perp B \mid C$.

2.6. Probability

- **2.6.3. Expectation and Variance**

To summarize key characteristics of probability distributions, we need some measures. The *expectation* (or average) of the random variable X is denoted as

$$E[X] = \sum_{x} x P(X = x). \tag{2.6.9}$$

When the input of a function $f(x)$ is a random variable drawn from the distribution P with different values x, the expectation of $f(x)$ is computed as

$$E_{x \sim P}[f(x)] = \sum_x f(x) P(x). \tag{2.6.10}$$

In many cases we want to measure by how much the random variable X deviates from its expectation. This can be quantified by the variance

$$\mathrm{Var}[X] = E\left[(X - E[X])^2\right] = E[X^2] - E[X]^2. \tag{2.6.11}$$

Its square root is called the *standard deviation*. The variance of a function of a random variable measures by how much the function deviates from the expectation of the function, as different values x of the random variable are sampled from its distribution:

$$\mathrm{Var}[f(x)] = E\left[\left(f(x) - E[f(x)]\right)^2\right]. \tag{2.6.12}$$

2.7. Documentation

- In order to know which functions and classes can be called in a module, we invoke the `dir` function
- For more specific instructions on how to use a given function or class, we can invoke the `help` function

	MXNET	PYTORCH	TENSORFLOW
Functions and classes	from mxnet import np print(dir(np.random))	import torch print(dir(torch.distributions))	import tensorflow as tf print(dir(tf.random))
Finding the usage	help(np.ones)	help(torch.ones)	help(tf.ones)