Mathematic Foundation

Jiatong

Context

- Derivative, limit, Integral
- Optimization
- Probability
- Linear Algebra
- Some Basic Concepts of Machine Learning

Derivatives

- The changing rate of a function
- F’(x)

Slope

https://zybuluo.com/w568w/note/1081202

How to define slope for other functions

How to define slope for other functions (cont'd)

How to define slope for other functions (cont'd)

How to define slope for other functions (cont'd)

How to define slope for other functions (cont'd)

How to define slope for other functions (cont'd)

Limitation

$$\lim_{x \to a} f(x) = b$$

$$\lim_{x \to 1} x = ?$$

$$\lim_{x \to 3} x^2 = ?$$

$$\lim_{x \to 1} \frac{x^2 - 3x + 2}{x - 1} = ?$$

Limitation (cont'd)

- $\lim_{x \to 1} \frac{x^2 - 3x + 2}{x - 1}$

$= \lim_{x \to 1} \frac{(x-2)(x-1)}{(x-1)}$

$= \lim_{x \to 1} (x - 2) = -1$

$$\lim_{h \to 0} \frac{f(x+h)-f(x)}{x+h-x} = \lim_{h \to 0} \frac{f(x+h)-f(x)}{h}$$

Slope (defined with limitation

Comparison

	F(x)	F’(x)
Function value	Value of the function at x	Slope (limitation) of the function at x
Is Function	yes	Yes
derivative	F’(x)	F’’(x)
When function value is 0	No specific meaning	Is the x-axis value of the function’s local maximum (we call the points as critical points or stationary points)

Representation of Derivative

- What about the function with two variables
- Y = a + b

- Solution:
- $\frac{d}{da}Y$ and $\frac{d}{db}Y$

Partial Derivative

- When multivariate, we need to define partial derivatives
- Write as:
  - $\frac{\partial y}{\partial x}$

- F = ax + by
- $\frac{\partial F}{\partial x} = a$
- $\frac{\partial F}{\partial y} = b$

Gradient

- F = ax + by
- $\frac{\partial F}{\partial x} = a$
- $\frac{\partial F}{\partial y} = b$

- The gradient of the function is *(a, b)*

Some Basic Rules

- Scalar multiplication: $\partial_x[af(x)] = a[\partial_x f(x)]$
- Polynomials: $\partial_x[x^k] = kx^{k-1}$
- Function addition: $\partial_x[f(x) + g(x)] = [\partial_x f(x)] + [\partial_x g(x)]$
- Function multiplication: $\partial_x[f(x)g(x)] = f(x)[\partial_x g(x)] + [\partial_x f(x)]g(x)$
- Function division: $\partial_x \left[\frac{f(x)}{g(x)}\right] = \frac{[\partial_x f(x)]g(x) - f(x)[\partial_x g(x)]}{[g(x)]^2}$
- Function composition: $\partial_x[f(g(x))] = [\partial_x g(x)][\partial_x f](g(x))$
- Exponentiation: $\partial_x[e^x] = e^x$ and $\partial_x[a^x] = \log(a)e^x$
- Logarithms: $\partial_x[\log x] = \frac{1}{x}$

Integral – the opposite of derivative

Integral (Cont'd)

Want some more

- https://notendur.hi.is/vae11/%C3%9Eekking/principles_of_mathematical_analysis_walter_rudin.pdf
- 华东师范大学数学系，《数学分析》

Context

- Derivative, limit, Integral
- Optimization
- Probability
- Linear Algebra
- Some Basic Concepts of Machine Learning

Optimization (gradient-based)

- Objective function or criterion vs. cost function, loss function, or error function

- $x^* = argmin\, f(x)$

Gradient and Function

Critical Points

This local minimum performs nearly as well as the global one, so it is an acceptable halting point.
Ideally, we would like to arrive at the global minimum, but this might not be possible.
This local minimum performs poorly and should be avoided.
$f(x)$
x

Gradient Descent

Context

- Derivative, limit, Integral
- Optimization
- Probability
- Linear Algebra
- Some Basic Concepts of Machine Learning

Random Variable

A variable that can take on different values randomly

It may be discrete or continuous:

if discrete, it is one that has a finite or countably infinite number of states (may not be numerical numbers)

If continuous, it is associated with a real value

Probability Distribution

- A probability distribution is a description of how likely a random variable or set of random variables is to take on each of its possible states.

- The way we describe probability distributions depends on whether the variables are **discrete or continuous**

PMF and PDF

- Probability Mass Function (discrete) and Probability Density Function (continuous)

PMF

- Probability mass function is the probability distribution of a discrete random variable, and provides the possible values and their associated probabilities.
- It is the function p: $\mathbb{R} \rightarrow [0, 1]$ defined by
- $p_X(x_i) = P(X = x_i)$ for $-\infty < x < \infty$

- It follows:
  - $\sum_{p_X}(x_i) = 1$
  - $p_X(x_i) > 0$
  - $p_X(x) = 0$ for all other x

Uniform Distribution for Discrete Variables

- $P(x = x_i) = \frac{1}{k}$ (we assume there are k states)
- It is a PMF because it follows the three rules

Joint Probability Distribution

- A probability distribution over many variables is known as a joint probability distribution
- P(x=*x1*, y=*y1*) denotes the probability that x=*x*, y=*y* simultaneously.

Marginal Probability

- Sometimes we know the probability distribution over a set of variables and we want to know the probability distribution over just a subset of them. This is known as Marginal Probability

- For discrete random variables x and y and we know P(x, y), we can find P(x) with the sum rule:
- $\forall x \in x, P(\mathrm{x} = x) = \sum_y P(\mathrm{x} = x, \mathrm{y} = y)$

Conditional Probability

- In many cases, we are interested in the probability of some event, given that some other event has happened. This is called a conditional probability.
- We denote the conditional probability that y=*y* given x=*x* as P(y=*y* | x=*x*). And it is defined as:
- $P(\mathrm{y} = y|\mathrm{x} = x) = \frac{P(\mathrm{y}=y,\mathrm{x}=x)}{P(\mathrm{x}=x)}$ (constraints?)

Chain Rule of Conditional Probability

- Any joint probability distribution over many random variables may be decomposed into conditional distributions over only one variables:

- $P(x, y, z) = p(x|y, z)\, p(y|z) p(z)$

Independence

- Two random variables x and y are independent if their probability distribution can be expressed as a product of two factors, one involving only x and one only y

- $\forall x \in \mathrm{x}, y \in \mathrm{y},$
$$p(\mathrm{x} = x, \mathrm{y} = y) = p(\mathrm{x} = x)p(\mathrm{y} = y)$$

Conditional Independence

- Two random variables x and y are conditionally independent given a random variable z if the conditional probability distribution over x and y factorizes in this way for every value of z

- $\forall x \in \mathrm{x}, y \in \mathrm{y}, \mathrm{z} \in z,$

$$p(\mathrm{x} = x, \mathrm{y} = y | \mathrm{z} = z)$$

$$= p(\mathrm{x} = x | \mathrm{z} = z) p(\mathrm{y} = y | \mathrm{z} = z)$$

Expectation

- The expectation of some function f(x) with respect to a probability distribution P(x) is the average, or mean value that f takes on when x is drawn from P.
- For discrete variables this can be computed with a summation:
- $\mathbb{E}_{x\sim P}[f(x)] = \sum_x P(x)f(x)$

Variance and Covariance

- The variance gives a measure of how much the values of a function of a random variable x vary as we sample different values of x from its probability distribution
- $Var\big(f(x)\big) = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2]$
- The covariance gives some sense of how much two values are linearly related to each other, as well as the scale of these variables:
- $Cov\big(f(x), g(x)\big) = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])(g(x) - \mathbb{E}[g(x)])]$

Gaussian Distribution

- $N(x; \mu, \sigma^2) = \sqrt{\frac{1}{2\pi\sigma^2}} \exp(-\frac{1}{2\sigma^2}(x-\mu)^2)$

Multivariate Normal Distribution

- $N(x;\mu,\Sigma) = \sqrt{\frac{1}{(2\pi)^n \det(\Sigma)}} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$

- Σ is a positive definite symmetric matrix

Mixture of Distributions

- $P(x) = \sum_i P(c = i)P(x|c = i)$
- Gaussian Mixture model?

Bayes Rule

- We often find ourselves in a situation where we know P(y|x) and need to know p(x|y). Fortunately, if we also know P(x), we can compute the desired quantity using Bayes Rule, defined as:

- $P(x|y) = \frac{P(x)P(y|x)}{P(y)}$

How to measure information using Probability

- Principles:

- Likely events should have low information content, and in the extreme case, events that are guaranteed to happen should have no information content whatsoever.
- Less likely events should have higher information content.
- Independent events should have additive information.

Shannon Entropy (Information Content)

- $H(x) = \mathbb{E}_{x\sim P}[I(x)] = -\mathbb{E}_{x\sim P}[logP(x)]$
- When x is continuous, the Shannon entropy is known as the differential entropy

Kullback-Leibler Divergence (KL Divergence)

- If we have two separate probability distributions P(x) and Q(x) over the same random variable x, we can measure how different these two distributions are using the KL Divergence

- $D_{KL}(P||Q) = \mathbb{E}_{x\sim P}\left[log\frac{P(x)}{Q(x)}\right] = \mathbb{E}_{x\sim P}[logP(x) - \log Q(x)]$

KL Divergence (Cont'd)

- The KL Divergence is 0 if and only if P and Q are the same distribution in case of discrete variables, or equal "almost everywhere" in the case of continuous variables.
- KL Divergence is non-negative and measures the difference between two distributions
- KL Divergence is asymmetric:
  - $D(P||Q) \neq D(Q||P)$ for some P and Q

Cross Entropy

- It is an extension to KL divergence.
- $H(P, Q) = H(P) + D(P||Q)$

Context

- Derivative, limit, Integral
- Optimization
- Probability
- Linear Algebra
- Some Basic Concepts of Machine Learning

Last Part of Math → Linear Algebra

Scalars, Vector, Matrices, and Tensors

- Scalars: 0, 1, 0.4, 1.0
- Vectors: [0, 1], [1.4, ½]
- Matrix:
  - $\begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$
- Tensors: with more axes

Basic Function

- Transpose: $\begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 3 \end{bmatrix}$
- Add:
  - $\begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 8 & 6 \end{bmatrix}$
  - $\begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} + 1 = \begin{bmatrix} 2 & 3 \\ 5 & 4 \end{bmatrix}$
- Multiply
  - $\begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} * 2 = \begin{bmatrix} 2 & 4 \\ 8 & 6 \end{bmatrix}$

Multiplying Matrices and Vectors

- The matrix product of matrices A and B is a third matrix C. In order for this product to be defined, A must have the same number of columns as B has rows. If A has a shape of m*n, B is n*p, then C is m*p

- $C = AB$
- $C_{i,j} = \sum_k A_{i,k} B_{k,j}$

Some Rules for Matrices Multiplication

- $A(B+C) = AB + AC$
- $A(BC) = (AB)C$
- $(A\ B)^T = (B^T\ A^T)$

Linear Function

- $Ax=b$
- A is a known matrix with m row, n column
- b is a known vector with n row
- x is the variables to be solved

Identity Matrices

- An identity matrix is a matrix that does not change any vector when we multiply that vector by that matrix

- Formally, I_n is a n*n matrix and
- $\forall x \in \mathbb{R}^n, I_n x = x$

Inverse Matrix

- The Matrix Inverse of A is denoted as A^{-1} and
- $A^{-1} \cdot A = I_n$

Linear Dependence

- $\begin{cases} x + y = 10 \\ 2x + 2y = 20 \\ 3x + 3y = 30 \end{cases} \rightarrow \begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$
- The key linear components is (1, 1) (or (2, 2), (3, 3)
- A set of vectors is linearly dependent is that a vector can be represented by a linear combination of other vectors

Norms

- $\left|\left|x\right|\right|_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}}$ for $p \in \mathbb{R}, p \geq 1$

- f(x)=0 → x=0
- f(x+y)<=f(x) + f(y)
- f(ax) = af(x), a is a real number

Context

- Derivative, limit, Integral
- Optimization
- Probability
- Linear Algebra
- Some Basic Concepts of Machine Learning

Machine Learning Basics

1. Machine Learning basics
   - Why Machine Learning is needed?
   - Main Concepts: Generalization, Model Expressiveness, Overfitting
   - Formal Notation
   - Experiment Design

Write a Program* to Recognize the Digit 2

This is hard to do manually!

```
bool recognizeDigitAs2(int** imagePixels){...}
```

Write a Program* to Recognize the Digit 2

This is hard to do manually!

```
bool recognizeDigitAs2(int** imagePixels){...}
```

Machine Learning solution:

1. Assume you have a database (training data) of 2's and non-2's.
2. Automatically "learn" this function from data

A Machine Learning Solution

Training data are represented as pixel matrices:
Classifier is parameterized by weight matrix of same dimension.

Training procedure:

1. When observe "2", add 1 to corresponding matrix elements
2. When observe "non-2", subtract 1 to corresponding matrix elements

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

→

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0

→

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	-1	0	0	0
0	0	0	0	1	0	-1	0	0	0
0	0	0	1	0	0	-1	0	0	0
0	0	1	0	0	0	-1	0	0	0
0	1	1	1	1	1	-1	1	1	0
0	0	0	0	0	0	0	0	0	0

A Machine Learning Solution

Training data are represented as pixel matrices:
Classifier is parameterized by weight matrix of same dimension.

Training procedure:

1. When observe "2", add 1 to corresponding matrix elements
2. When observe "non-2", subtract 1 to corresponding matrix elements

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

→

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0

→

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	-1	0	0	0
0	0	0	0	1	0	-1	0	0	0
0	0	0	1	0	0	-1	0	0	0
0	0	1	0	0	0	-1	0	0	0
0	1	1	1	1	1	-1	1	1	0
0	0	0	0	0	0	0	0	0	0

Generalization $\neq$ Memorization

Key Issue in Machine Learning: Training data is limited

- If the classifier just memorizes the training data, it may perform poorly on new data
- "Generalization" is ability to extend accurate predictions to new data

E.g. consider shifted image:

Will this classifier generalize?

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	-1	0	0	0
0	0	0	0	1	0	-1	0	0	0
0	0	0	1	0	0	-1	0	0	0
0	0	1	0	0	0	-1	0	0	0
0	1	1	1	1	1	-1	1	1	0
0	0	0	0	0	0	0	0	0	0

One potential way to increase generalization ability:

- Discretize weight matrix with larger grids (fewer weights to train)

E.g. consider shifted image:

Now will this classifier generalize?

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	-1	0	0	0
0	0	0	0	1	0	-1	0	0	0
0	0	0	1	0	0	-1	0	0	0
0	0	1	0	0	0	-1	0	0	0
0	1	1	1	1	1	-1	1	1	0
0	0	0	0	0	0	0	0	0	0

→

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	-1	0	0	0
0	0	0	0	1	0	-1	0	0	0
0	0	0	1	0	0	-1	0	0	0
0	0	1	0	0	0	-1	0	0	0
0	1	1	1	1	1	-1	1	1	0
0	0	0	0	0	0	0	0	0	0

→

1	1	1	0	0
0	0	0	0	0
0	0	1	-1	0
0	1	0	-1	0
1	1	1	0	1

Model Expressiveness and Overfitting

- A model with more weight parameters may fit training data better
- But since training data is limited, expressive model stand the risk o
  overfitting to peculiarities of the data.

$$\text{Less Expressive Model} \iff \text{More Expressive Model}$$
$$\text{(fewer weights)} \qquad \text{(more weights)}$$

$$\text{Underfit training data} \iff \text{Overfit training data}$$

Model Expressiveness and Overfitting

Fitting the training data (blue points: x_n)
with a polynomial model: $f(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$
under squared error objective $\frac{1}{2}\sum_n (f(x_n) - t_n)^2$

Basic Problem Setup in Machine Learning

- Training Data: a set of $(x^{(m)}, y^{(m)})_{m=\{1,2,..M\}}$ pairs, where input $x^{(m)} \in R^d$ and output $y^{(m)} = \{0, 1\}$
  - e.g. x=vectorized image pixels, y=2 or non-2
- Goal: Learn function $f : x \rightarrow y$ to predicts correctly on new inputs x.
  - Step 1: Choose a function model family:
    - e.g. logistic regression, support vector machines, neural networks
  - Step 2: Optimize parameters w on the Training Data
    - e.g. minimize loss function $\min_w \sum_{m=1}^{M} (f_w(x^{(m)}) - y^{(m)})^2$

Experiment Design

- Training Data: used to learn function $f()$
- Development / Validation Data: used to evaluate how good $f()$ is, to make high-level model selection decisions, e.g.
  - Which machine learning model to deploy?
  - Tradeoff hyperparameter between regularizer and likelihood?
- Test Data: used to *really* evaluate how good $f()$ is
- IMPORTANT:
  - Don't train on test data
  - Don't do model selection on test data
  - Be careful in your experiment design, so that you aren't fooled by over-optimism when deploying a model in real life!

Bias-Variance Tradeoff

- When deploying a model $f()$, we're really interested in the expected error/loss on unseen data
- But we only have estimates based on dev loss
- This estimated loss can be viewed as bias + variance
  - bias: errors from simplifying assumptions in the model $f()$.
  - variance: A different sample of the training set may have resulted in very different $f()$.
  - Imagine throwing many darts: bias = closeness to bullseye, variance = dispersion of darts
- Generally, complex functions fit the data better, so have low bias bu more susceptible to high variance

K-Fold Cross-Validation

- Suppose we can't afford to hold out a dev set out of our training set.
- Divide data into K parts, e.g. K=5
  - Fold 1, 2, 3, 4 as training, Fold 5 as dev
  - Fold 2, 3, 4, 5 as training, Fold 1 as dev
  - Fold 3, 4, 5, 1 as training, Fold 2 as dev
  - Fold 4, 5, 1, 2 as training, Fold 3 as dev
  - Fold 5, 1, 2, 3 as training, Fold 4 as dev
- Run the same algorithm on all K cases, compute average dev loss.
- Select model based on avg. dev loss, then evaluate on test.
- How to pick K? Consider computation and Bias-Variance tradeoff

No Free Lunch Theorem

- No machine learning method is best for all datasets
- You must learn to choose an appropriate model family and optimization algorithm for your task
- Don't trust anyone who advertises a machine learning method that always wins.

Question and Comments