인공지능 수학 개론

(저자 김종락)

(임시적인) 목록

1. 선형대수와 인공지능
   1. 파이썬 소개: 수식 중심으로
   2. 선형대수 기초

2.1 실습

* 1. 행렬의 연산 및 역행렬

3.1 실습

* 1. 벡터와 공간, 행렬과 사상

4.1 실습

* 1. 고윳값, 고유벡터

5.1 실습

1. 미적분과 인공지능

6. 미분과 적분

6.1 실습

7. 편미분과 경사 하강법

7.1 실습

1. 확률과 통계

8. 조건부 확률과 베이즈 정리

8.1 실습

9. 상관분석과 분산 분석

9.1 실습

1. 머신러닝과 딥러닝

10. 머신러닝 소개

10.1 실습

11. 딥러닝 소개

11.1 실습

# Chapter 1. 파이썬 소개: 수식 중심으로

파이썬은 R과 함께 데이터 분석에서 널리 쓰이는 오픈 소스 언어이다. R은 통계적인 면에서 강점이 있고 파이썬은 좀 더 고급 데이터 분석을 할 때 유용한 장점이 있다. 여기에서는 파이썬 위주로 수학식을 실습하고자 한다.

파이썬 3이상의 버전을 쓰도록 한다. 이를 직접 자신의 노트북에 설치하여 주피터 노 트북(Jupyter Notebook)에서 작업을 하거나 설치가 어려운 사람은 구글 CoLab에서 로그인해서 진행하면 된다. CoLab의 주소는 [https://colab.research.google.com/](https://colab.research.google.com/%20) 이다. CoLab에는 Tensorflow도 설치되어 있어 직접 설치하는 수고를 피할 수 있다. 필자는 Jupyter Notebook을 주로 사용하였다. 명령어만 넣으면 다음 단계에 오류가 있는지 확인이 되기 때문에 실수를 줄일 수 있다. 또한 자신의 로컬 파일들을 읽고 저장하기 용의하다. 다만 용량이 큰 프로젝트를 할 경우 제약이 있다. 자신의 환경에 맞게 진행을 하면 된다.

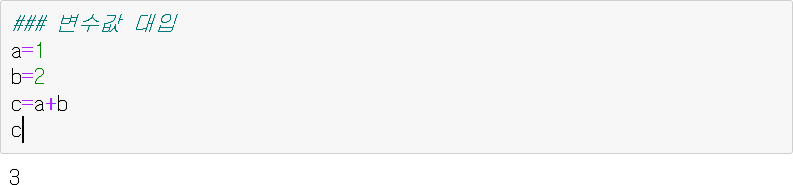
파이썬 설치는 <https://www.python.org/> 에 가서 다운받거나 다른 인터넷 사이트들이 있으니 생략하도록 한다. 대신 최신 파이썬(현재 3.10)이 항상 좋은 것이 아니다. 우리가 필요로하는 scipy, numpy, Tensorflow 등은 파이썬 3.6에서 무리없이 작동이 되기 때문이다. 대신 64비트용 설치 파일을 권장한다

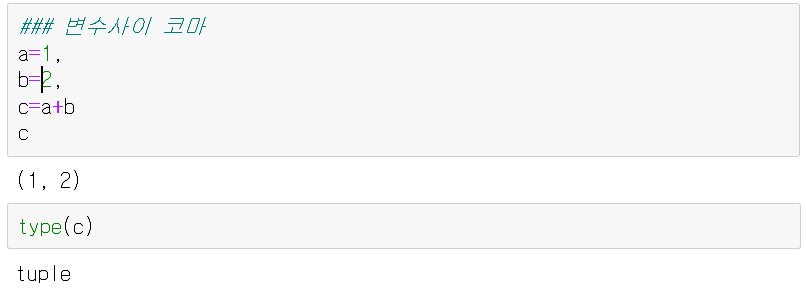
주피터 노트북으로 시작하기

텍스트이(가) 표시된 사진

자동 생성된 설명주피터 노트북을 성공적으로 설치한 후 New 페이지를 열면 아래와 같은 화면이 나온 다. 첫 파일명은 Untitled이며 확장자는 ipynb이다. 앞의 i는 interactive하다는 의미에서 나온 것이다. 아래 화면에서는 편이상 Ch1.1로 지정하였다.

이 빈칸에 1+1을 입력한 후 그냥 Enter를 치면 줄만 아래로 바뀔 분 결괏값이 나오 지는 않는다. 이 기능은 여러 개의 명령어를 한꺼번에 넣고 그 결과를 알고 싶을 때 유용하다. 하나 이상의 명령어를 실행시키는 간단한 방법은 Shift+Enter를 치는 것이 다. 그러면 아래와 같이 결괏값 2를 출력하고 새로운 입력값 라인이 형성된다.

수식 뒷 부분에 등호(=)는 넣지 않는다. 대신 변수를 지정할 경우는 =를 넣는다. 즉 a=1, b=2라고 하고 c=a+b라고 할 때, c의 값을 구하고자 할 경우 아래와 같이 대입 한다. 각 라인 끝에는 코마를 붙이지 않는다.

입력값 사이에 코마(,)를 넣을 경우 c의 값은 tuple이라는 type의 (1,2)가 나오니 주의해야 한다.

덧셈과 마찬가지로 뺄셈은 ‘–’ 기호로, 곱셈은 ‘\*’ 기호로, 나눗셈은 ‘/’ 기호를 이용 하여 자연스럽게 계산하면 된다.

곱셈 기호를 두 번 연속 즉 ‘\*\*’라고 쓸 경우 지수가 된다. 나눗셈의 경우 몫 (quotient)을 구하고 싶으면 ‘//’를 나머지(remainder)를 구하고 싶으면 ‘%’를 사용 한다.

참고로 논문을 Tex로 쓰면 지수는 $5^3$라고 표현한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

지금까지 파이썬의 명령어를 이용하여 사칙연산을 계산하는 방법을 배웠다. 이 정도 면 파이썬으로 전자계산기를 대체할 수 있다는 것을 알 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명print의 기능을 좀 더 유연하여 이용하여 결괏값을 좀 더 보기 좋게 만들 수 있다. print 안에는 문자열 또는 수식을 넣을 수 있는데, 문자열을 넣는 경우 “ ”를 두어야 한다. 수나 수식을 넣을 경우는 “ ”를 두지 않는다.

텍스트이(가) 표시된 사진

자동 생성된 설명예제로서, “googol은 10의 100제곱을 의미합니다.”를 인쇄하고 그 실제값을 출력값 으로 나타내보자.

자료구조

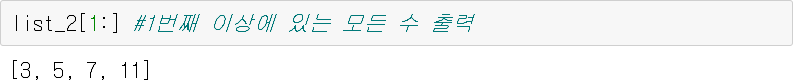
파이썬의 자료구조는 크게 리스트(list), 튜플(tuple), 딕셔너리(dictionary), 세트(set) 로 이루어져 있다.

* 리스트

리스트는 수학에서 집합과는 약간 다른 개념이다. 집합은 나열된 순서에 상관이 없으 나 리스트는 순서가 중요하다. 사용하는 형식은 ‘변수=[ ]’를 이용한다. 예를 들어, 1,2,3,4,5를 list\_1이라는 변수에 넣으려고 한다. 이때 순서를 고려하여 list\_1 = [1, 2, 3, 4, 5]라고 치면 된다. list\_1[0]은 1을 의미하고 list\_1[4]는 5를 의미한다. list\_1 뒤에 숫자를 추가하고 싶으면 list\_1.append(숫자)를 쓰면 된다. 만약 특정 위치에 특정 숫자를 넣고자 하면 list\_1.insert(위치, 숫자)라고 쓰면 된다. 리스트에 나열된 수들을 반대로 쓰고 싶으면 list\_1.reverse()라고 쓰면 된다. 이때 ()안은 빈 칸으로 둔다. 특정 숫자를 지우고 싶 으면 list\_1.remove(숫자)라고 쓰고 숫자들을 작은 수부터 큰 수로 정렬하고 싶으면 list\_1.sort()라고 쓰면 된다. 텍스트이(가) 표시된 사진

자동 생성된 설명

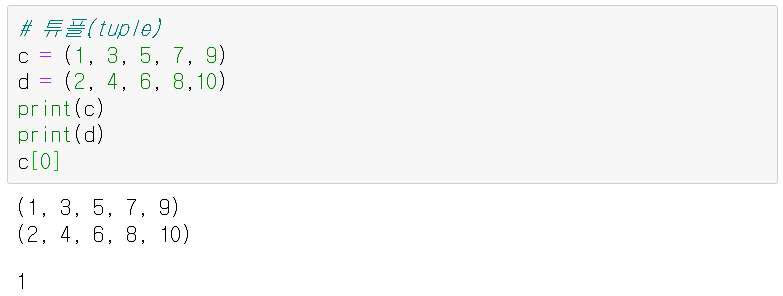
리스트에 있는 수들 중 일부를 불러내고 싶을 땐 ‘:’ 기호를 사용한다. 예를 들어 리 스트 list\_2 = [2,3,5,7,11]이라고 할 때 list\_2의 처음 3개의 수를 출력하고 싶으면 list\_2[0:3]이라고 치면 된다. 그러면 2,3,5가 출력된다. [0:3]의 의미는 0번째부터 3번 째 이전까지의 수를 의미한다. 즉 index 3에 해당하는 수는 출력하지 않는다.

어떤 특정 위치를 포함한 모든 수를 출력하고 싶을 경우에도 :을 사용할 수 있다.

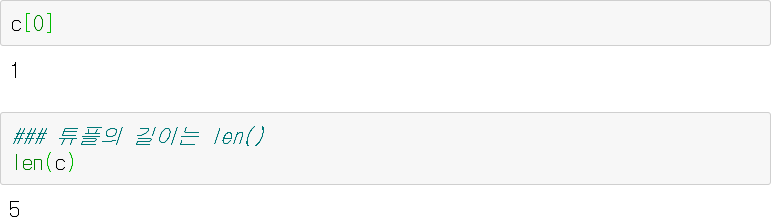
텍스트이(가) 표시된 사진

자동 생성된 설명

* 튜플

이제는 튜플(tuple)에 대해서 간단히 설명해 보자. 튜플은 리스트와 마찬가지로 수 또 는 문자들을 순서대로 나열한다. 튜플과 리스트의 가장 큰 차이점은 튜플의 값들은 변경이 되지 않는다는 것이다. 튜플이 수로 이루어졌을 경우 길이가 n인 벡터로 이해 하면 된다. 실제로 벡터를 구성하는 값들은 없애거나 추가할 수 없다. 두 개의 튜플 은 더할 수 있으나 벡터의 합과는 다소 다름에 주의해야 할 것이다. 또한 하나의 튜 플에 자연수를 곱할 수도 있다. 아래 예제들로 이를 설명해 보자.

c는 1,3,5,7,9로 이루어진 튜플이고 d는 2,4,6,8,10으로 이루어진 튜플이다. c의 첫 번째 원소를 알고 싶으면 ‘c[0]’ 라고 쓰면 된다. 이는 리스트와 같은 형식이니 새로 울 것은 없다. 튜플 c의 길이를 알고 싶으면 len(c)를 하면 된다.



두 개의 튜플을 더하면 어떨까? 이 경우는 첫 번째 튜플의 값들 뒤에 두 번째 튜플의 값들이 연결되어 새로운 튜플이 생성된다.

텍스트이(가) 표시된 사진

자동 생성된 설명위의 예제로 하면 c+d=(1,3,5,7,9,2,4,6,8,10)이 된다.

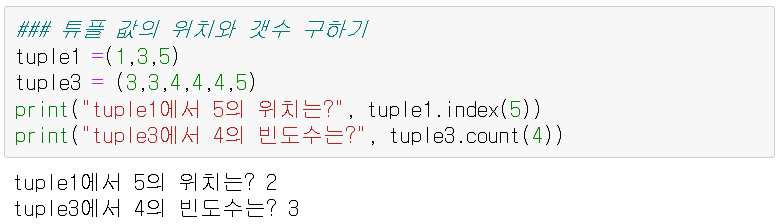
텍스트이(가) 표시된 사진

자동 생성된 설명따라서 튜플사이의 기호 ‘+’는 덧셈을 의미하는 것이 아님에 주의하도록 하자. 한편 하나의 튜플에 ‘\*’를 사용할 수도 있다. 예를 들어 c\*3라고 쓰면 이는 c+c+c를 의미 한다. 즉 c의 원소들이 3번씩 반복된다.

튜플은 숫자의 일반화이기 때문에 크기를 비교할 수 있다. 크기의 기준은 첫 번째 성 분들을 비교하여 더 큰 것이 크다. 만일 이것이 같으면 두 번째 성분들을 비교하면 된다. 여기에서 크기가 결정이 나면 나머지 성분들은 비교할 필요가 없다. 이런 것을 사전식 크기(dictionary order)라고 한다. 영어사전을 보면 단어들의 배열은 a로 시 작하는 단어들이 맨 먼저 오고 그런 단어들 사이에는 두 번째 알파벳이 가장 작은 것이 더 먼저 나오는 것과 같은 원리이다. (1,2) < (2,3) or (1,100) < (2,1) 또는 올림픽에서 금메달이 가장 많은 나라가 더 상위랭크가 되는 것도 좋은 예이다.

텍스트이(가) 표시된 사진

자동 생성된 설명예를 들어 tuple1=(1,3,5)라고 하고 tuple2=(1,4,3)라고 하면 어떤 것이 더 클까? 아니면 비교 불가능할까? 정답은 tuple2가 더 크다. 첫 번째 성분은 둘다 1이지만, 두 번째 성분은 3<4이므로 tuple2가 더 크다. 세 번째 성분은 5>3이지만 이미 2번째 성 분에서 크기가 결정되었으므로 고려하지 않아도 된다.

튜플의 값들은 변경이 되지 않지만 값의 위치라든지 값이 몇 개가 있는지를 세는 것 은 가능하다. 튜플 값의 위치는 ‘튜플.index(값)’의 명령어를 사용한다. tuple1.index(5)라고 하면 5가 있는 index를 구하라는 것이므로 2가 된다. tuple3=(3,3,4,4,4,5)라고 할 때, 4이 몇 번 나오는지 구하고자 할 경우는 ‘튜 플.count(값)’라는 명령어를 사용한다. 따라서 tuple3.count(4)은 3이 된다.

튜플과 리스트가 유사하다보니 서로 변환이 가능하다. 튜플을 리스트로 변환하는 명 령어는 ‘list((튜플변수명))’이다. 튜플명에 ‘( )’를 넣는 것에 주의하자. 마찬가지로 리스트 를 튜플로 변화하는 명령어는 ‘tuple((리스트변수명))’이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

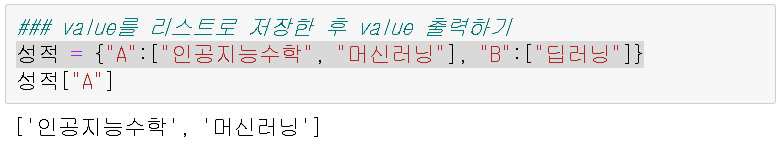
* 딕셔너리

딕셔너리(dictionary)는 key와 그에 대응하는 value가 하나의 쌍이 되고 이 쌍들을 나열한 것들의 모임이다. 딕셔너리는 리스트와 튜플과 구별하기 위하여 중괄호 ‘{ }’를 사용한다. 형식은 {key1:value1, key2:value2, ...}이다. key값은 숫자 또는 문자가 될 수 있다. 중복이 되지 말아야 한다. 좀 더 일반적으로는 key값은 튜플을 사용할 수도 있다. 대신 list는 key 값이 될 수 없으니 조심하도록 하자.

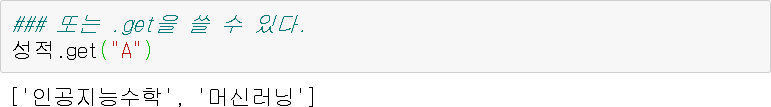
텍스트이(가) 표시된 사진

자동 생성된 설명예를 들어 AI과목이라는 딕셔너리에 인공지능수학, 머신러닝, 딥러닝을 넣고 싶고 각 각의 value를 1, 2, 3이라는 key값을 이용한다면 AI과목 = {1:“인공지능수학”, 2:“머 신러닝”, 3:“딥러닝”}라고 입력하면 된다. key 값 1을 이용하여 인공지능수학이라는 value를 출력할 수 있다.

value는 리스트가 될 수 있다. ‘성적’이라는 변수에 인공지능수학과 머신러닝을 A학 점, 딥러닝을 B학점이라고 할 때 이를 딕셔너리로 표현하면 다음과 같다.



get함수를 써서 value를 출력할 수 있다.



텍스트이(가) 표시된 사진

자동 생성된 설명특정 value뿐만 아니라 모든 value를 출력하고 싶으면 ‘변수.values()’라고 쓰고 모든 key를 출력하고 싶으면 ‘변수.keys()’라고 쓰면 된다.

텍스트이(가) 표시된 사진

자동 생성된 설명리스트와 마찬가지로 딕셔너리도 key:value 쌍을 추가하거나 삭제할 수 있다. 추가하 고자 할 경우는 ‘변수[새로운 key] = value’의 형태로 진행하면 된다. 삭제하고자 할 경우는 ‘del 변수[key]’를 하면 해당하는 key:value가 삭제된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

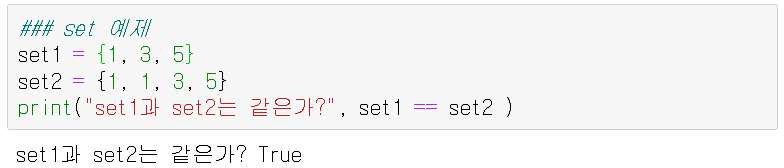
텍스트이(가) 표시된 사진

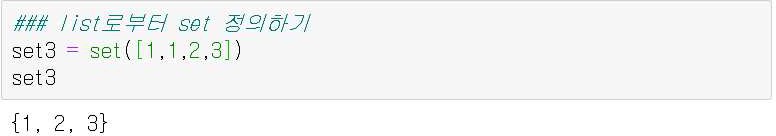
자동 생성된 설명또한 특정 key의 value만 수정할 수도 있다. 이런 경우 ‘변수(key) = 새 value’라고 쓰면 된다. 예를 들어 성적이라는 변수에서 B를 맞은 딥러닝을 프로그래밍으로 대체 할 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명마지막으로 딕셔너리 변수의 모든 key:value를 다 삭제하려면 ‘변수.clear()’형태를 사용하면 된다.

* 세트

세트(set)는 수학에서 말하는 집합을 의미한다. 따라서 리스트나 튜플과는 달리 세트 의 원소들 사이에는 순서가 없다. 또한 원소들이 중복되지 않는다. 두 집합사이에 교집합, 합집합, 차집합 등의 연산이 존재한다. 세트의 표시는 ‘{}’를 사용한다.

또한 리스트로부터 집합을 정의할 수 있다.

텍스트이(가) 표시된 사진

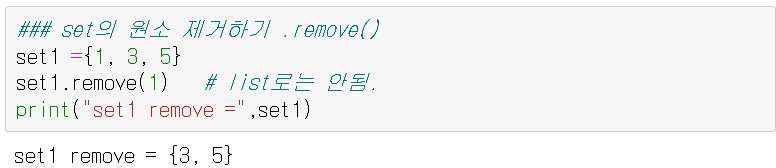
자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명교집합은 ‘변수1.intersection(변수2)’ 또는 ‘변수1 & 변수2’로 표현가능하고 합집합 은 ‘변수1.union(변수2)’ 또는 ‘변수1 | 변수 2’로 표현할 수 있다.

두 집합 A, B의 차집합은 A의 원소 중에 B에 있지 않는 것들의 모임으로 수학적으로 는 ‘A-B’로 표시한다. 실제로는 A-(A&B)에 해당한다. 또한 A, B의 합집합에서 교집 합을 뺀 집합은 (A | B)-(A&B)에 해당하고 ‘A ^ B’로 표시한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명리스트나 딕셔너리처럼 원소를 추가할 수 있는데 원소 하나를 추가할 경우는 .add() 라고 쓰고 여러 개의 원소를 추가할 경우는 .update()라고 쓴다. 원소 하나를 제거할 때는 .remove()라고 쓴다.

지금까지 1장에서는 파이썬의 기본 데이터 형태인 리스트, 튜플, 딕셔너리, 세트에 대 하여 간단히 살펴 보았다. 2장에서는 선형대수에 나오는 기본적인 수식들을 파이썬으 로 표현하는 것을 배우도록 할 것이다.

# Chapter 2. 선형대수 기초

선형대수는 대수학의 한 분야로 **하나 이상의 변수로 이루어진 일차식의 해(solution)를 다루는 수학분야**다. 예를 들어 는 변수가 두 개이고 식이 하나인 선형방정식이다. 이 해 집합을 그래프로 나타내면 직선이 나온다. 따라서 선형대수학은 기하학과 연관이 있다. 3개의 변수로 이루어진 선형 방정 식 의 경우 해 집합은 평면을 나타낸다. 일반적으로 n개의 변수로 이루어진 선형 방정식의 해 집합은 (n-1)-차원의 도형이 된다. 여러 개의 선형방정식을 연립선형방정식(a system of linear equations)라고 하고 선형대수는 연립선형방정식의 해를 어떻게 효율적으로 구할지 연구하는 분야이다.

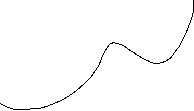
이제 반대로 어떤 점들(좌표들) 혹은 데이터들이 있다고 가정하자. 우선 아주 간단한 질문을 해보도록 하자. 이 주어진 점들이 어떤 연립선형방정식의 해 혹은 부분적인 해로 표현될 수 있을까?   
예를 들어 평면에 주어진 가 주어졌을 때 이 두 점을 지나는 직선은 혹은 이 된다. 따라서 이 두 점은 선형방정식의 부분해로 이해할 수 있다. 실제로 임의의 서로 다른 두 점은 유일한 선형 방정식을 만들어 낸다.

질문을 좀 더 확장하여 위의 두 점 외에 를 지나는 선형방정식은 존재할까? 는 을 만족하지 않으므로 그런 방정식은 존재하지 않는다. 여기에 두 가지의 접근 방법이 존재한다. 수학자는 일차 방정식에서 탈피하여 2차함수 즉, (단, 는 상수)을 생각하고 이 세 점이 이 2차 방정식의 해가 됨을 보일 것이다. 이제 데이터 사이언티스트의 관점에서 해결하도록 하자. 이 세 점들이 반드시 직선 위에 있지 않아도 된다는 가정을 하면 재미있는 현상이 벌어진다. 비록 직선 위에 있지 않더라도 **이 직선과 주어진 세 점의 거리를 최단으로 만드는 방법(즉, 실제값과 예상값과의 차이를 최소화 하고자 하는 방법)**이다. 이것이 더 현실적인 해법이다. 왜냐하면 이 방법은 주어진 점의 수가 증가한다고 하더라도 직선을 구하는 문제이기 때문에 고차원의 방정식의 해를 구하는 것보다 훨씬 쉽기 때문이다. 이 방법이 바로 **선형회귀 방법**이다. 선형회귀 방법은 머신러닝의 기초가 되고 딥러닝의 수식에도 등장한다. 이쯤 되면 왜 선형대수학을 공부해야 하는 것인지 충분이 설득이 되었으리라 믿는다.

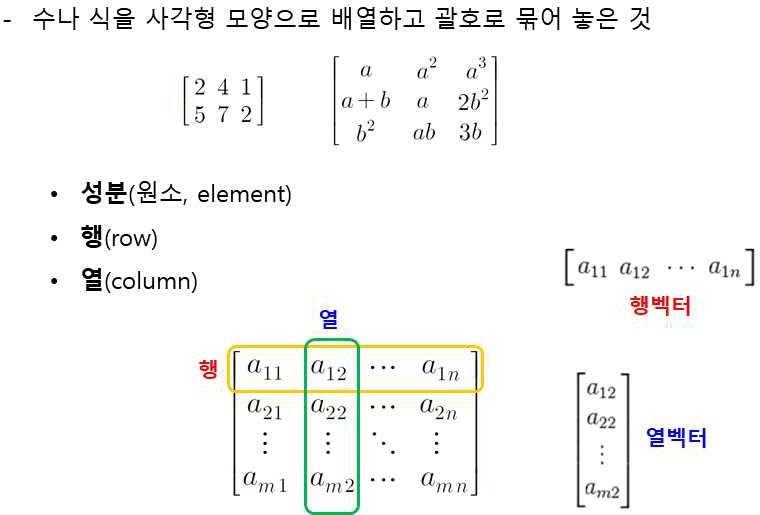
이번 장에서는 선형대수의 기본적이 개념들과 이를 파이썬으로 구현해 보고자 한다.

\*\*미분에서, 주어진 곡선 그래프에서 그 위의 한 점을 지나고 그래프에 접하는 직선의 방정식은 이다.   
즉 원래 곡선의 방정식인   
더 나아가 를 이차식으로 할 수 있을까?일반적으로 가 여러 번 미분 가능하면 를 차 다항식으로 approximation할 수 있다. 이것을 **테일러 전개** 라고 한다.

수학자와 공학자의 마인드 차이로 새로운 수학적 방법론이 탄생한 것이라고 생각할 수 있다.

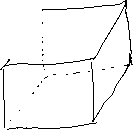
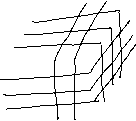


* **텍스트, 영수증이(가) 표시된 사진

  자동 생성된 설명선형방정식**  
  >> 선형방정식을 효율적으로 풀고 싶다면 어떻게 할까? 위의 변수들은 정말로 중요할까? 우리가 가장 관심을 갖고 있는 부분은 행렬로 표현된 식 중, 왼쪽 행렬 부분이다.  
  A=   
  >특히 행렬의 크기가 인 일 떄,   
  만약 의 판별식 이면, 를 만족하는 해집합 는 유일한 해를 갖는다. 일반적으로 가 인 경우에도 가 0이 아닐떄만 유일한 해집합이 존재한다.
* **행렬(Matrix)**

> 숫자 → 벡터(개의 숫자를 나열한 것. 리스트 혹은 튜플) →행렬(벡터를 여러 개 행으로 혹은 열로 나열한 것)

> 텐서(, 행렬을 여러 개 붙인 것)는 행렬을 일반화한 것이다.. 예를들어 3차원 텐서는 성분이 로 표현된다. 인공지능 딥러닝에서는 CNN을 기술할 때 적용된다. Convolutional Neural Network에서 사진을 많이 입력으로 하는데, 다양한 필터를 3차원 텐서의 한 layer로 이해할 수 있다.



텍스트, 영수증, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

* 는 자명함. 는 영어로 를 의미함
* 일 때 를 대칭행렬(라고 한다. 대칭행렬의 특징은 대각선을 기준으로 거울처럼 대칭이 된다.

텍스트, 영수증이(가) 표시된 사진

자동 생성된 설명

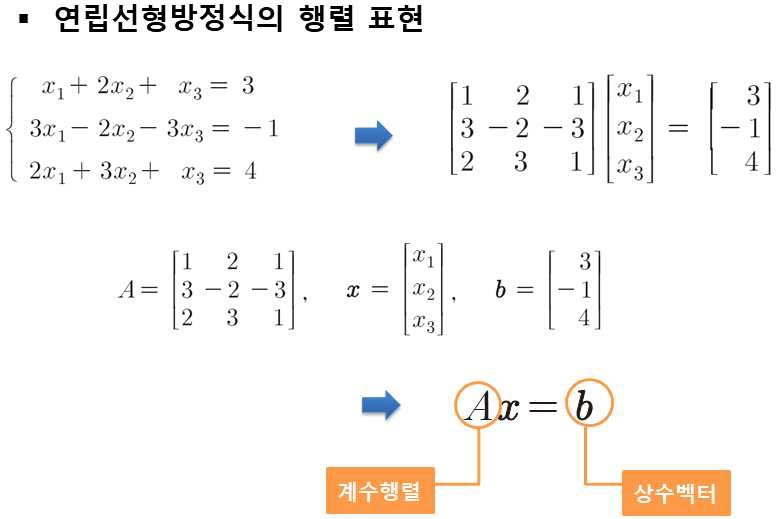
* 두 행렬의 합과 차는 두 행렬의 크기가 같아야 한다는 조건하에 성분끼리 더하거나 빼면 된다. 스칼라곱은 스칼라를 각 성분에 곱한다.
* (는 체()로써 덧셈과 곱셈이 자유로운 연산체계. 통상 를 실수집합 이라고 이해하면 됨.)  
  이 는 어떠한 대수적 구조를 갖는다고 할까? 이것은 **벡터공간**()을 이룬다. 두 개의 벡터를 더하거나 스칼라곱을 하듯이 같은 크기의 행렬들도 서로 더하거나 스칼라곱을 한다. 따라서 도 벡터공간이 된다.

테이블이(가) 표시된 사진

자동 생성된 설명

* 두 행렬의 곱이 자연스러운가?| 부자연스러운 것이 아니지만 이렇게 정의하면 새로운 연산체계가 생겨난다.
* 두 행렬의 성분끼리 곱하는 곱셈 연산도 존재한다. 실제로 이것은 파이썬에 이미 구현되어 있다. 데이터 분석에서는 이러한 연산도 의미가 있다.

* 행렬과 연립선형방정식의 관계



* 연립선형방정식을 행렬과 백터의 곱로 나타낼 수 있다. 이때을 **행렬의 식**이라고 부른다.
* 텍스트, 영수증이(가) 표시된 사진

  자동 생성된 설명연립선형방정식을 동치인 다른 연립선형방정식으로 변환할 수 있다. 마찬가지로 이에 대응하는 행렬방정식도 같이 변환된다.
* 연립선형방정식을 좀 더 간단하며 동치인 연립선형방정식으로 변화할 수 있다. 마찬가지로 이에 대응하는 Coefficient matrix도 reduce를 할 수 있다.

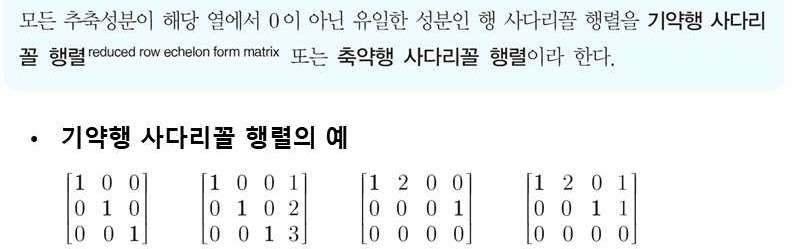
텍스트, 시계이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명● 예제

* 이제 행렬식에 나타나는 변수 벡터는 변하지 않는다.
* 행렬에서 각 행의 맨 왼쪽에 0이 아닌 성분을 피벗(pivot) 또는 추축성분(pivot entry)라고 한다.
* 테이블이(가) 표시된 사진

  자동 생성된 설명이중에서 row echelon form이 중요하다.
* 다른 책들에서는 pivot의 값이 1이 아니며 0이 아닌 수가 있을 경우에도 r.e.f.라고 부르기도 한다. 따라서 r.e.f.는 유일하지 않다.
* r.e.f.를 유일하게 만드는 형태를 r.r.e.f.(reduced row echelon form)이라고 한다
* ref를 더 심화한 행렬 형태가 reduced row echelon form(r.r.e.f)이다.



테이블이(가) 표시된 사진

자동 생성된 설명● 예제

* (a) 둘 다 아니다(pivot에 2가 있으므로)
* (b) r.r.e.f
* (c) r.r.e.f
* (d) r.e.f
* 왜 r.r.e.f.를 공부하는가? r.r.e.f.가 되면 해집합을 쉽게 구할 수 있다.
* r.r.e.f.만드는 과정을 가우스 소거법(Gauss elimination method)라고 한다. 계산속도를 줄일 수 있다는 장점이 있다.
* 벡터(Vector)텍스트이(가) 표시된 사진

  자동 생성된 설명
* 수 →벡터 → 행렬 → 텐서

* 텍스트, 시계이(가) 표시된 사진

  자동 생성된 설명예제
* 정방행렬 A를 k번 곱할 경우 이를 로 나타낸다.
* 실습(행렬과 벡터)

텍스트이(가) 표시된 사진

자동 생성된 설명(Reference 파이썬과 NumPy로 배우는 선형대수, 이정주(BJ 출판사))

* numpy를 이용하여 다양한 수식을 편리하게 구현한다.

텍스트이(가) 표시된 사진

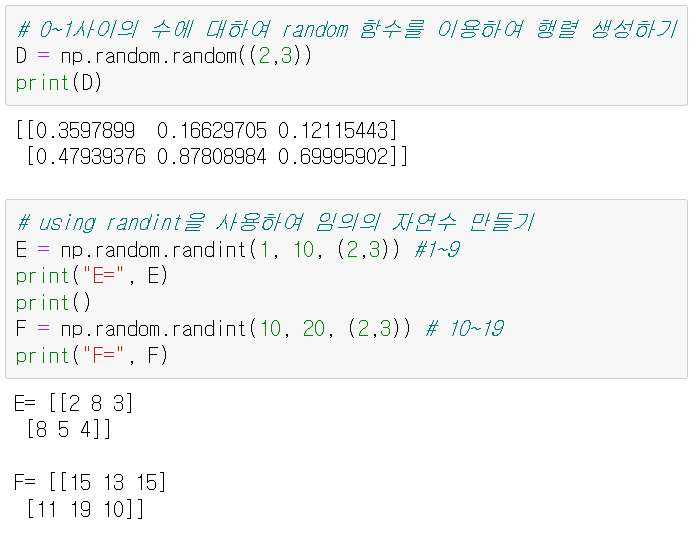
자동 생성된 설명

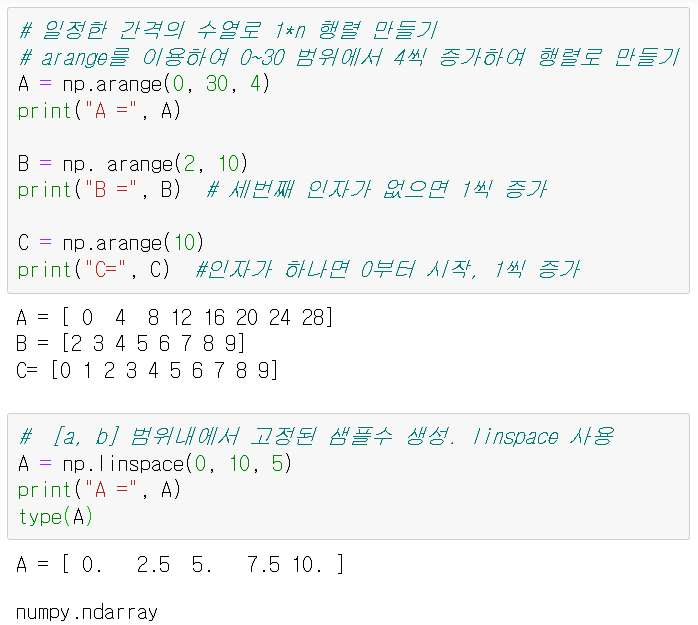
텍스트이(가) 표시된 사진

자동 생성된 설명

* eye(3)은 3차 단위행렬을 의미함.테이블이(가) 표시된 사진

  자동 생성된 설명
* ==



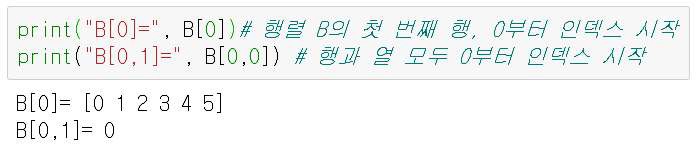


▶

텍스트이(가) 표시된 사진

자동 생성된 설명

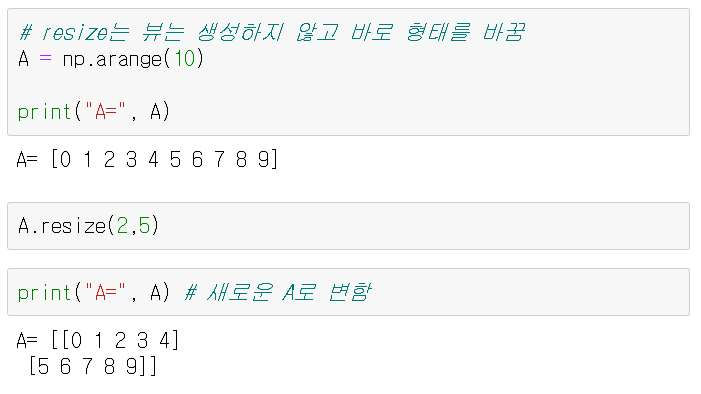
▶

* B[0,1]은 typo.

텍스트이(가) 표시된 사진

자동 생성된 설명

▶



▶

텍스트이(가) 표시된 사진

자동 생성된 설명

* vstack은 두 번쨰 행렬을 아래(vertically, column-wisely)방향으로 붙인다.

텍스트이(가) 표시된 사진

자동 생성된 설명

* hstack은 두 번째 행렬을 우측(horizontally, row-wisely)방향으로 붙인다.

텍스트이(가) 표시된 사진

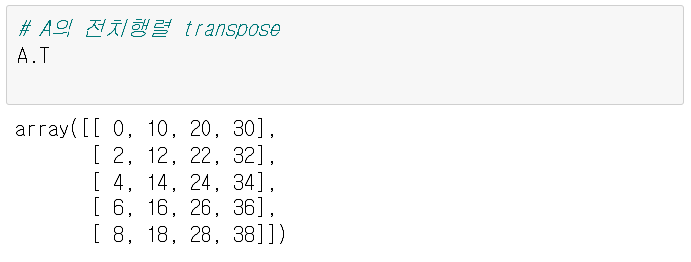
자동 생성된 설명

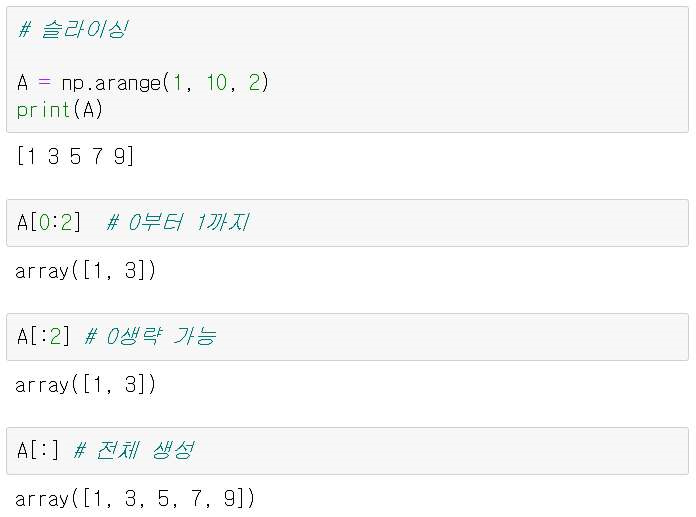
* concatenate((A,B), axis=0)는 .vstack((A,B))와 같다.
* concatenate((A,B), axis=1)는 .hstack((A,B))와 같다.

텍스트이(가) 표시된 사진

자동 생성된 설명

* 는 A.T로 구현한다.





텍스트이(가) 표시된 사진

자동 생성된 설명

▶

텍스트이(가) 표시된 사진

자동 생성된 설명

▶

텍스트이(가) 표시된 사진

자동 생성된 설명



텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

▶

# Chapter 3. 행렬의 연산 및 역행렬

연립선형방정식을 쉽게 푸는 방법은 없을까?

중학교 때 변수가 두 개인 일차방정식 두 개를 푸는 방법을 기억할 것이다. 가장 핵 심적인 방법은 변수 중의 하나를 소거하여 간단해진 식에서 정답을 구하는 방법이 었다. 이것을 소거법이라고 한다. 그런데 놀랍게도 변수가 3개 이상인 경우에도 이 방법이 가장 효율적이라는 것을 알 수 있다. 이것을 가우스-조던 소거법 (Gauss-Jordan elimination method)라고 한다.

이 방법을 정방행렬에 적용할 경우 역행렬을 효과적으로 구할 수 있다. 역행렬의 정의, 대각행렬, 삼각행렬 등에 대하여 알아본다. 또한 행렬이 만족하는 대수적 구조에 대하여 알아본다.

* 가우스-조단 소거법

텍스트이(가) 표시된 사진

자동 생성된 설명

(예제). 가우스-조르단 소거법으로 다음을 푸시오.

(정답)

(Exercise) 가우스 소거법으로 다음을 푸시오.

텍스트이(가) 표시된 사진

자동 생성된 설명

(정답)

* 행렬의 거듭제곱

텍스트, 시계이(가) 표시된 사진

자동 생성된 설명정방행렬 에 대하여 는 를 번 거듭제곱한 것이다.

▶ 증명?

▶ 의 의미는?

● 행렬 덧셈과 곱셈은 다음의 성질을 만족한다.

텍스트이(가) 표시된 사진

자동 생성된 설명크기가 같은 행렬 와 스칼라 에 대하여 다음이 만족된다.

▶ 이러한 성질을 만족하는 연산 체계를 무엇이라고 불리는가?

* 텍스트이(가) 표시된 사진

  자동 생성된 설명역행렬

▶ 역행렬은 실수 의 역원 과 같은 역할을 한다.

* 역행렬의 uniqueness

차 정방행렬 의 역행렬이 존재하면, 하나 뿐이다.

▶ (증명)

● 2차 정방행렬의 역행렬

텍스트, 시계이(가) 표시된 사진

자동 생성된 설명

▶ (문제) 가우스 소거법을 이용하여 증명하시오.

● 역행렬의 성질

텍스트이(가) 표시된 사진

자동 생성된 설명

차 정방행렬 가 가역이고 가 0아닌 스칼라 일 때 다음을 만족한다.

▶ (증명)

* 전치행렬(transpose of A)

라고 할 떄 를 A의 transpose라고 한다.

* 전치행렬의 성질텍스트이(가) 표시된 사진

  자동 생성된 설명

▶ (증명)

* 대칭행렬(symmetric matrix)과 반대칭행렬(skey-symmetric matrix)

▶ 대칭행렬의 특징은?

▶ 반대칭행렬의 특징은?

* 텍스트이(가) 표시된 사진

  자동 생성된 설명정방행렬의 표현 방법

▶ (증명)

● 대각행렬(diagonal matrix)

정방행렬 에 대하여 일 때 A를 대각행렬이라고 한다.

이 경우 간단히 로 표현한다.

▶ 같은 크기의 두 대각행렬의 곱은 대각행렬이다.

▶ 임의의 차 정방행렬 와 차 대각행렬 에 대하여 와 의 관계는?

* 대각합

정방행렬 에 대하여 대각성분에 있는 원소들의 합을 라고 한다.

▶ (예제)

* 대각합의 성질텍스트이(가) 표시된 사진

  자동 생성된 설명

▶ (증명)

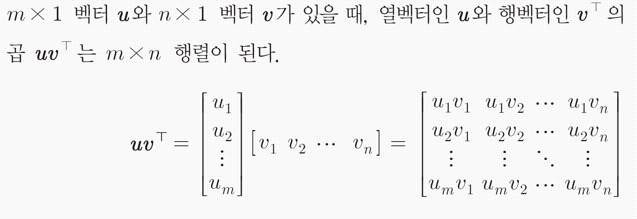
* 삼각행렬(triangular matrix)  
  - upper triangular matrix U  
  - lower triangular matrix L

▶ (예제)

* 텍스트이(가) 표시된 사진

  자동 생성된 설명(정리)

▶ (증명)

* 열벡터와 행벡터의 곱

▶ (예제)



이것을 (u tensor v)라고도 쓴다.

* 행렬 와 열벡터 에 대하여 를 쉽게 계산하는 방법은?

●

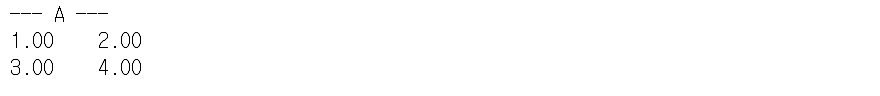
■ 실습

▶ 행렬을 출력하는 함수

텍스트이(가) 표시된 사진

자동 생성된 설명





텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명



텍스트이(가) 표시된 사진

자동 생성된 설명



# Chapter 4. 벡터와 공간, 행렬과 사상

벡터는 선형대수에서 행렬만큼 중요한 주제이다. 차원 벡터는 보통

로 표현된다. 우리가 알고 있는 실수는 1차원 벡터, 평면 위의 좌표 는 2차원 벡터, 공간 위의 좌표는 3차원 벡터로 생각할 수 있다. 하나의 데이터 혹은 샘플은 3차원 이상의 성분으로 이루어져 있다. 시간을 포함하여 4차원 위치를 나타낸다면, 4차원 벡터가 필요할 것이다.

이때 각 성분을 피쳐(feature)라고도 한다.

따라서 데이터 분석을 한다는 것은 여러 개의 벡터들을 하나의 공통 알고리즘으로 이해하고자 한다는 것이다.

벡터들로 이루어진 집합이 벡터의 합과 스칼라 곱에 닫혀있을 때 이 집합 을 **벡터 공간**이라고 한다. 두 개의 벡터 공간가 있을 때 이 두 개를 연결하는 함수를 **선형 변환**이라고 한다. 이 선형 변환이 행렬로 표현된다는 것은 중요한 사실이다.

따라서 행렬은 단순히 격자로 이루어진 배열이 아니고 함수로도 볼 수 있다. 왜냐하면, 행렬 에다가 차원 (열)벡터 를 곱하면 는 차원 벡터가 된다.

이 장에서는 여러 개념들을 소개하고 서로 어떻게 연결되는지 살펴본다.

* 벡터와 벡터공간
* 벡터의 합과 차

라고 할 때  
합은 차는 로 표현된다.

* 벡터의 스칼라 곱

가 스칼라이고 라고 할 때 스칼라 곱은 로 표현된다.

벡터의 차는 이므로 합과 스칼라곱만 있으면 된다.

* 벡터 공간

를 공집합이 아니라고 하고, 를 에 속하는 차원 벡터라고 하자.

가 스칼라라고 하자. 보통 스칼라는 실수집합 혹은 복소수 집합 를 의미하나, 일반적으로는 체()에서 성립한다.

텍스트이(가) 표시된 사진

자동 생성된 설명그러면 다음이 성립한다.



이것을 만족하는 집합을 벡터공간(vector space)라고 한다. 이 원소들을 벡터라고 한다.  
(1), (4), (5), (6)을 만족하는 것을 군(group)이라고 한다. (3)까지 만족할 때 이 군을 가환군(Abelian group)이라고 한다.

* 벡터 공간의 예
* 차원 식수 벡터들로 이루어진 집합
* 의 부분 공간=   
  의 부분집합으로써 그 자체가 벡터 공간을 이루는 집합
* (예) 에서 인 부분집합을 생각한다. 가 벡터공간인가? yes  
  은 와 동치, 즉 원점을 지나고 기울기가 -1인 직선을 의미한다. 따라서 의 원소는 형태이다.   
   이고,   
  . 또한 이므로 에 있다.  
  (6)번, 즉 역원이 존재해야하는데,   
  의 역원은   
  따라서 는 벡터공간이 된다.
* (예) 에서 인 부분집합을 생각한다. 가 벡터공간인가? no  
  보통 이 있느냐로 체크 가능, 왜냐하면 이 에 들어가 있지 않기 때문에.
* 실제로는 의 부분 공간은 , 원점을 지나는 직선들, 뿐이다.
* = 차수가 인 실수 계수를 갖는 다항식의 모임(벡터공간)  
  =   
     
     
  따라서 차원 이라고 한다
* M\_(m×n) (R) = 실수 위에서 정의된 크기가 m×n인 행렬들의 모임(벡터공간)  
  (두 행렬의 합과 스칼라 곱이 정의되어 있기 때문)  
  - 만약 행렬의 곱셈까지 생각한다면, M\_(m×n) (R)은 환(ring)이라는 수학적 구조가 된다. 자세한 내용은 생략.
* 부분 공간의 정의  
  벡터공간 의 부분집합 로서, 는 의 연산을 그대로 이용하면서 자신이 벡터공간이 될 때, 를 부분공간()이라고 한다  
  - 부분공간 테스트 ()  
  가 의 부분공간의 필요충분 조건은   
  (i)   
  (ii) (iii)
* 선형결합(linear combination)  
   이라고 할 때, 를 선형결합이라고 한다.
* 선형종속(linearly dependent)와 선형독립(linearly independent)  
  벡터공간 V의 부분집합 가 linearly independent라는 말은 을 성립하는 경우는 뿐이다.  
  그 이외의 경우를 linearly dependent 라고 한다. 즉, 을 만족하는 0이 아닌 가 존재한다.
* 예제  
  는 linearly independent하다.
* 생성 집합()   
  의 임의의 원소를 주어진 집합 의 선형 결합으로 표현 가능할 때, 가 를 (생성, )한다고 한다. 기호로는
* 기저()  
  벡터공간 의 부분집합 가 를 생성하고 동시에 가 할 때 를 의 기저라고 한다
* 표준기저()  
  의 표준기저는 이다. 직교 좌표계의 근본을 이루는 집합이 바로 표준기저였다.
* 정리  
  벡터공간 의 기저 가 주어져 있을 때, V의 임의의 원소는 기저들의 선형결합으로 유일하게 표현된다. 즉, 에 대하여,  
  (존재성)   
  그리고 이러한 들이 유일하다(유일성).  
  실제 증명에서 존재성은 가 한다는 성질을 이용하여 증명한다.  
  유일성은 의 성질을 이용하여 증명한다.
* 차원()  
  벡터공간 의 기저 의 개수를 벡터공간의 차원이라고 한다. 만약에  
   라면 (유한차원)이다. 그렇지 않다면 (무한차원) 라고 한다.  
  (질문) 무한차원의 벡터공간의 예제는 무엇인가?  
  복습: 벡터 공간 의   
  (왜냐하면 가 표준 기저가 되기 떄문이다.)  
  벡터 공간 라고 하면  
   이 기저를 이룬다. 따라서
* 벡터의 크기(norm)  
  . x의 크기 norm는 , 즉 의 크기는 원점 0과 사이의 거리를 의미한다.  
  텍스트이(가) 표시된 사진

  자동 생성된 설명



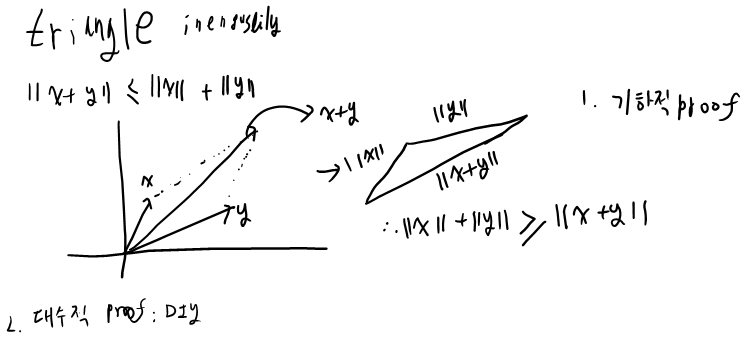
* 노름의 성질  
  텍스트이(가) 표시된 사진

  자동 생성된 설명  
  (3) 조건을 (삼각형 부등식)이라고 한다.
* 벡터의 내적  
  Let x=(x\_1,x\_2,…,x\_n ),y=(y\_!,y\_2,…,y\_n)∈R^n 두 벡터 x,y의 내적 inner x∙y=x\_1 y\_1+x\_2 y\_2+⋯+x\_n y\_n∈R로 정의한다. 이 내적을 usual inner product 또는 Euclidean inner product 라고도 한다.
* 내적의 성질  
  텍스트이(가) 표시된 사진

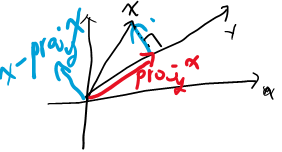
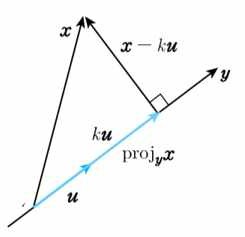
  자동 생성된 설명  
  (5)   
  (6)



* 내적과 사잇각  
    
  단, 여기서 는 가 이루는 각 에 대한 값
* 코시-슈바르츠 부등식   
  증명은 위의 내적과 사잇각의 식 이용.
* 삼각부등식 Triangle inequality



* 피타고라스의 정리 Pythagorean theorem  
  에 있는 임의의 두 벡터 \*가 직교하는 것(즉 ) 과 동치는  
     
  (증명)를 계산해 본다.
* 정사영 orthogonal projection  
  에 있는 임의의 두 벡터 에 대하여, 벡터 의 방향의 벡터를 의 위로의 정사영()이라고 한다. 영어로는 라고 읽고  
  라고 쓴다. 기학학적 의미는?
* 텍스트이(가) 표시된 사진

  자동 생성된 설명정사영을 dot product로 표현하면 다음과 같다.  
     
  (증명)  
    
  (단위벡터)라고 하자. 즉 는 방향의 단위벡터. 그러면 (단, 는 어떤 실수). 정사영의 정의에 의하여  
  이 된다. 관하여 풀면 가 된다. 따라서  
   가 된다.
* 예제  
  벡터 의 벡터 으로의 정사영을 구하시오.  
     
  - 정답은 (1,0)  
  만약에 이고 일 때
* (정의) 내적공간   
  실벡터공간 (즉, 실수 위에서 정의된 벡터공간)에 속하는 임의의 두 벡터 에 대하여 다음의 세가지 조건을 만족하는 함수를 와 의 내적이라고 하고 또는라고도 쓴다.  
  텍스트이(가) 표시된 사진

  자동 생성된 설명  
  (예제) 라고 할 때 에 대하여 내적공간을  
  으로 정의한다. 이것을 보통의 내적()또는 유클라디언내적()이라고 한다.
* (예제) 실수 위에서 정의된 행렬의 모임라고 할 때  
   에 대하여 내적공간을 아래와 같이 정의할 수 있다.  
  텍스트이(가) 표시된 사진

  자동 생성된 설명 이 내적의 의미는?  
  행렬 와 의 성분들끼리의 곱들의 합. 즉 이것은 행렬인 차원 벡터의 내적을 행렬로 자연스럽게 확장한 것이다.
* (예제) 구간 위에서 정의된 연속 함수 들의 집합은 벡터공간 을 이룬다. 이러한 에 대하여 아래와 같이 내적공간을 정의할 수 있다.  
  세가지 조건을 체크한다. 그중 (3) 조건은  
     
  이 일반적으로
* (정의) 벡터공간 가 주어지고, 임의의 에 대하여 실수 값 가 정의되고 다음을 만족할 때, 이 벡터공간을 노름공간 라고 한다  
  텍스트이(가) 표시된 사진

  자동 생성된 설명
* (예제) V를 내적이 정의된 실수 벡터 공간이라고 하고 라고 하면 는 노름공간이 된다.
* (예제) 그러나 이 라고 할 때 는 노름을 정의하지만 내적을 정의할수는 없다. 따라서 내적공간은 노름공간이 되나 그 역은 참이 아니다.
* **벡터와 미분**  
  이제 벡터와 미분을 연결해 보자. 즉 다변수함수 에 대하여 벡터에 관하여 미분을 할수가 있다.
* (정의) 다변수함수 에 대하여 벡터 에 대한 미분의  
  정의는 인 벡터이고 f의 **그래디언트** 라고 한다.  
  - 변수가 여러 개인 함수를 미분할 때 기호로는 ∂를 쓴다.
* (예제)  
  라고 할 때 를 구하시오
* (정의) 벡터함수 에 대하여 각 변수 에 대한 의 미분은 아래와 같다.  
  이다.  
  이라고 할 때  
   를 자코비안 행렬 라고 하고 라고 쓴다.
* (정의) 다변수 함수 에 대하여 벡터 에 대한 2차 미분 을 헤시안 행렬 라고 하고 라고 쓴다.  
  테이블이(가) 표시된 사진

  자동 생성된 설명
* (정의) 다변수함수 의 라플라시안 은  
   이다.
* (예제) 이라고 할 때 를 구하시오.  
  에 대하여 를 두 번 미분하면 가 된다. 에 대하여 를 두 번 미분하면 2가 된다. 따라서 가 된다.

# Chapter 5. 선형변환, 고윳값, 고유벡터

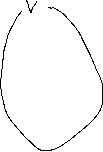
두 개의 벡터 공간 가 주어졌을 때, 이 공간들 사이의 관계를 어떻게 이해할 수 있을까? 두 벡터 공간이 **본질적으로 같다**는 것은 무슨 뜻일까? 또한 두 벡터 공간이 **유사하다는** 것은 무슨 뜻일까? 이러한 질문에 답하기 위해선 **선형변환**의 개념이 필요하다.  
이 관점은 ‘이면 이다’는 에 속하면 에 속한다. 등 수학에서 많이 접근하는 방법 중의 하나로서, 집합적으로는

**선형변환**은 하나의 벡터공간 에서 자신 또는 다른 벡터공간 가는 함수로서  
의 연산들(즉, 벡터합과 스칼리곱)이 그대로 보존되는 성질을 갖고있다. 따라서 의 원소들 사이의 연산 값이 에 의하여 이동된 원소(의 이미지)들 사이의 연산 값과 같다. 예를 들어, 거울은 우리의 이미지를 좌우가 바뀐 상태로 그대로 반영하고 있다. 망원경이나 현미경은 작은 물체를 크게 확대하지만 그 모양을 그대로 유지하고 있다. 이러한 성질을 수학적으로 설명할 수가 있다.

고윳값은 선형대수의 중에 특별한 성질을 갖고 있다. 주어진 선형변환 이 적당한 의 원소를 스칼라 곱으로 변환시키는 경우를 말한다. 즉 을 만족한다. 이때 나오는 를 고윳값이라고 하고 를 고유벡터라고 한다. 고윳값과 고유벡터는 선 형변환의 특징을 반영하는 중요한 개념들이다. 머신러닝에서 등장하는 주성분분석()은 고윳값의 크기가 큰 순서로 나열한 후 그에 대응하는 고유벡터들을 새로운 축으로 변환하여 기존 데이터를 읽어 들이는 방법이다. 이 방법은 차원의 저주(curse of dimension)을 부분적으로 해결한 것으로 볼 수 있다.

이 장에서는 여러 개념들을 소개하고 서로 어떻게 연결되는지 살펴본다.

* 선형변환
* (정의)  
  벡터공간 에서 벡터공간 로 가는 함수 가  
  (1) 모든 에 대하여   
  (2) 모든 와 스칼라 에 대하여   
  을 만족할 때 이 L을 선형변환(linear transformation, linear map)이라고 한다.  
  (그림)



* (예제) 이라고 하고 라고 할 때 은 선형변환이다.   
  (증명)  
  임의의 에 대하여  
  임의의 와 스칼라 에 대하여   
  따라서 은 선형변환이다
* (예제) 일반적으로 이라고 하고 (단,는 실수인 상수)라고 할 때  
  은 선형변환이다. 만약 이면 이고 이를 자명변환()이라고 한다.  
  (증명) Obvious!
* (예제) 일반적으로 이라고 하고 (단 는 상수이며 )일 때 은 선형변환이 아니다.  
  (증명) 을 선형 변환이라고 하자. 가 만족되어야 한다.  
     
     
  따라서   
   이어야 한다.  
  이는 모순이다 따라서 은 선형변환이 아니다.
* (질문) 선형변환은 어떻게 만들면 될까?  
   라고 하고 인 선형변환을 좀 더 다르게 표현해 보자.  
  따라서 스칼라로 표현된 선형변환을 대각행렬로 표현할 수 있었다.  
  여기에 영감을 받아 대각행렬대신 임의의 행렬로 대신해도 과연 선형변환이 될까?   
  정답은 된다!

* **(정리)** 벡터공간 이라고 하고 이라고 하자. 행렬 를 실수 위에서 정의된 행렬이라고 하자. 그러면 라고 정의된 함수가 선형변환이 된다. (단 여기서 는 열벡터로 이해한다.)  
  (증명)  
     
     
  .  
  놀라운 사실은 에서 로 가는 임의의 선형변환 은 행렬로 표현 가능하다는 것이다.
* (중요 정리) 만약 이 선형변환이라고 하면 를 만족하는   
  행렬 가 존재한다.  
  (증명)   
  임의의 벡터는 (단 는 표준기저)로 표현 할 수 있다.  
     
     
     
    
  (첫번쨰 등식은이기 때문이다. 즉  
     
     
     
  두번쨰 등식은 는 의 의 선형결합으로 표현가능하다.  
  세번쨰 등식은 으로 만들면 된다. 따라서 인 행렬 가 존재한다.  
    
  (정의) 위 정리에 나오는 를 선형변환 의 표준행렬()라고 한다.
* (예제)  
  - 회전변환 rotation  
   가 표준행렬이다.  
  - 반사변환 reflection  
  1. 축을 기준으로 반사하는 선형변환이 아래와 같을 때 표준행렬 를 구하시오.  
     
  2. y축을 기준으로 반사하는 선형변환과 그의 표준행렬을 B를 구하시오.  
     
     
     
  결론적으로 말하면 본질적으로 선형변환이라는 개념과 행렬이라는 개념이 동일한 것이다. 두 벡터공간 사이에 선형변환 이 존재하면   
  두 벡터공간은 서로 호모모픽(homomorphic)하다 라고 한다.   
  만약에 이면 와 가 동형(isomorphic)이다 라고 한다.   
    
  선형변환를 합성해도 선형변환이 된다.
* (정리) 벡터공간 에 대하여 두 개의 선형변환 에 대하여   
  그의 합성인 도 선형변환이 된다.  
  (증명) DIY

위 합성 선형변환을 행렬로 표현하면 어떤 것이 될까?

* (정리) 두 개의 선형변환 의 표준행렬을 각각 라고 할 때,  
  의 표준행렬은 이 된다.  
  (증명)

실수에서 곱셈의 항등원 1이 있듯이 항등원의 역할을 하는 선형변환이 있다.

* (정의) 벡터공간 V에 대하여 를 만족할 때,  
  를 항등선형변환(identity linear transformation)이라고 한다.   
  를 나타내는 표준행렬은 단위행렬 이다.

실수 a의 곱셈에 대한 역원 a^(-1)이 있듯이 역변환을 정의할 수 있다.

* (정의)  
  선형변환 과 에 대하여 를 만족할때 를 의 역변환()이라고 하고 이라고 쓴다.  
    
  만일 이고 선형변환의 역변환이 존재하고 표준행렬을 라고 하면  
  역변환 의 표준행렬은 이 된다. 선형변환 을 선형연산자()라고 한다.
* (정의) 선형연산자 가 주어졌을 때 모든 에 대하여  
  (여기서 ⋅은 내적을 의미 한다)을 만족할 때 L를 직교연산자(orhogonal linear operator)라고 한다.
* (정리) 선형연산자 가 주어졌을 때,  
  모든 에 대하여인 필요충분 조건은 모든 에 대하여  
  을 만족하는 것이다.  
  (Recall: )  
  (⟸)자면하다. 왜냐하면 라고 하면   
     
  (diy)

따라서 직교연산자를 노름보존 선형연산자(norm-preserving linear operator)라고 한다.

* 선형변환의 부분공간과 랭크
* (정의) A를 m×n행렬이라고 할 때 다음과 같이 정의한다.
* 열공간
* 행공간
* 영공간  
  (영공간은 를 패러티 체크 행렬로 갖는 오류정정부호(error-correcting code)라고도 한다.)  
  텍스트이(가) 표시된 사진

  자동 생성된 설명
* (정의) 를 행렬이라고 할 때 A의 계수를 라고 쓰고  
   로 계산된다.  
  는 의 부분공간이므로
* (예제)  
  라고 하자.   
  (a)   
     
  (b)   
     
  (c) 이므로 을 가우스-조르단 방법으로 푼다.   
  이므로 가 되는데, 이를 벡터형태로 나타내기 위하여 아래와 같이 표현한다.   
     
     
     
   (단 는 임의의 실수)이므로  
     
     
  (a)와 (b)에 의하면 . 이것은 일반적으로 같다.  
  (a)~(c)에 의하면 . 이것은 일반적으로 같다.
* (정리) 영공간의 정의역 R^n의 부분공간이다. (증명)  
  텍스트이(가) 표시된 사진

  자동 생성된 설명  
  따라서 영공간을 구하기 위해서는 Ax=0 의 해집합을 구하면 된다.
* (정의) 행렬 A의 영공간의 차원을 퇴화차수(nullity)라고 한다.  
  즉

* (정리) (**차원의 정리 또는 랭크 정리**)  
  행렬 에 대하여  
     
  즉, 이 성립한다.  
  (증명)  
  텍스트이(가) 표시된 사진

  자동 생성된 설명  
  의 개수(non-free variable의 개수)  
  의 개수(free variable의 개수)   
  따라서 =전체 의 개수

* 고윳값과 고유벡터
* (정의) 차 정방행렬 에 대하여 를 만족하는 가 존재할 때 를 의 고윳값()라고 하고 를 고유벡터()라고 한다.
* 고유벡터는 영벡터가 될 수 없으나 고윳값은 0이 될 수가 있다.
* 과 동치는 무엇일까?

따라서 행렬 의 고윳값을 모두 구할 수 있다.

* (정의) 차 정방행렬 A에 대하여 를   
  특성다항식 이라고 하고 을   
  A의 특성방정식 이라고 한다.
* 차 정방행렬 이 고윳값은 없을 수도 있고, 중복이 될 수도 있다. 최대 개를 갖을수 있다.
* 차 정방행렬 의 고윳값 중의 하나를 라고 하자. 그러면 이에 대응하는 고유 벡터들은 벡터공간을 이룬다. 이 벡터공간을 라고 하고 고유공간이라고 한다. 즉  
   즉 는 모든 고유벡터들과 영벡터를 합한 것이다.
* (정리) 차 정방행렬 의 고윳값은 최대 개다.   
  (증명)

* (정리) 차 정방행렬 이 가역행렬이고 이 고윳값이면 의 고윳값은이 된다.  
  (증명)  
  텍스트, 시계이(가) 표시된 사진

  자동 생성된 설명
* (정리) 차 정방행렬 에 대하여 개의 고윳값 이 존재할 때 다음이 성립한다.  
  (1)   
  (2)  
  (증명) (1) 을 대입하면 가 성립한다.   
  (2) (단,  
  이므로 가 성립한다.
* (정리)(케일리-해밀턴 정리 Cayley-Hamilton theorem) 차 정방행렬 의 특성방정식 일 때 대신에 를 대입하면 을 만족한다.  
  (증명) Omit.
* 주성분 분석(Principal Component Analysis, PCA)
* (정의) 차원의 데이터 에 대하여 평균벡터 (mean vector)는  
  로 정의된다. 공분산 행렬(covariance matrix)는  
  로서 행렬이다. 이면가 된다.
* (PCA) [참고, <응용이 보이는 선형대수학>]  
  텍스트이(가) 표시된 사진

  자동 생성된 설명
* 차원으로 근사할 때 발생하는 오차를 라고 하면 아래가 성립한다.  
  텍스트이(가) 표시된 사진

  자동 생성된 설명

한편 값은 아래와 같이 계산하면 이 된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

즉,

* 분산을 가장 크게 하는 기저벡터 를 찾는 문제를 풀어야 한다. 라그랑지 승수 (Lagrange multiplier)를 이용하여 다음의 문제를 해결한다.  
  텍스트이(가) 표시된 사진

  자동 생성된 설명
* 따라서 로부터 고윳값의 크기가 큰 순서대로 개의 고유벡터 를 선택한다. 보통 를 많이 선택한다.