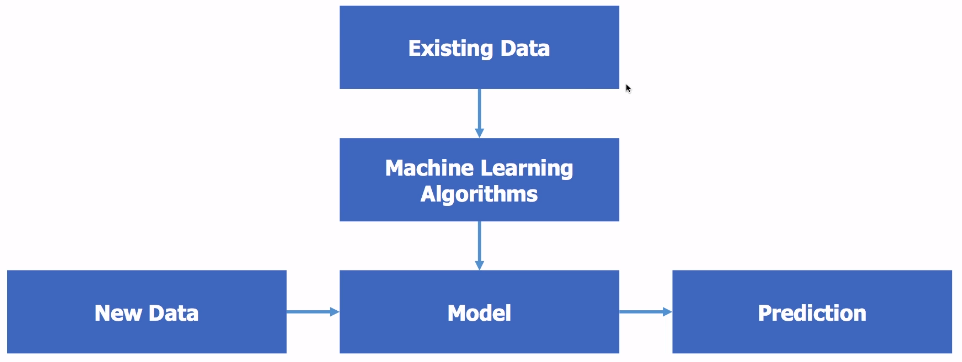
1. **First Week**

* **Big Data**
* 정해진 시간내에 한 대의 machine으로 처리가 어려운 크기의 데이터
* 고성능 머신 또는 다수의 machine을 사용해 목표대로 데이터를 처리할 수 있다.
* Ex) 실시간 대용량 SQL처리
* **Machine Learning**
* 데이터로부터 패턴을 학습하는 기법
* 데이터의 크기가 중요하지만 작다고 불가능한 것은 아니다.
* 알고리즘을 통해 모델을 추출
* Ex) 컴퓨터에서 손 글씨 이해시키기
* **Data Mining**
* Machine learning과 같은 의미 (데이터에서 의미 있는 규칙 찾기)
* 머신 러닝: 알고리즘
* 데이터 마이닝: 어플리케이션
* Ex) 스팸 필터, 상품 추천
* **Deep Learning**
* Neural Net 기반 machine learning 알고리즘
* 사진내 객체 인식, 번역 등에서 탁월한

|  |  |
| --- | --- |
| Big Data | Machine Learning |
| 데이터 처리 | 데이터 분석 |
| Data Engineering | Data Analysis |
| Hadoop, Spark,  NoSQL, BigTable | Navie, Bayes.  SVM, Logistics,  Regression |

* **Machine Learning Process**



* Model

예측을 위한 수학 공식, 함수

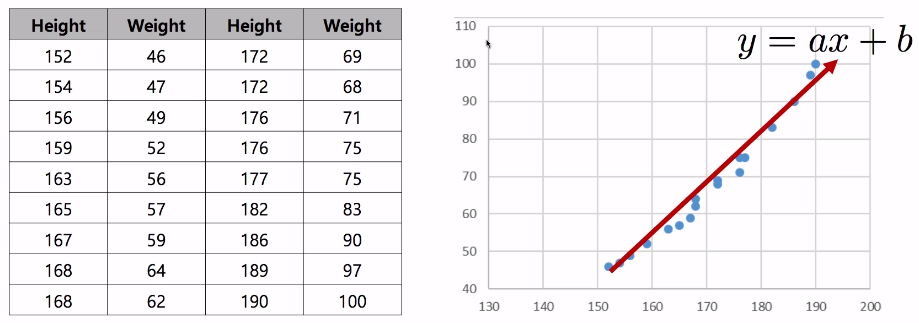
Ex) 1차 방정식, 확률분포, condition rule

* Algorithm

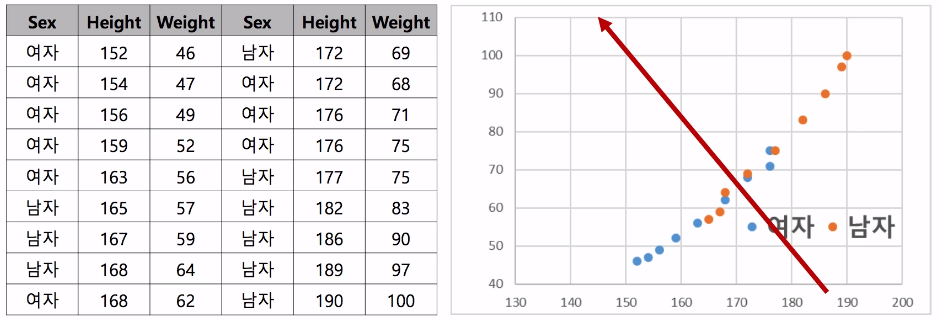
어떠한 문제를 풀기 위한 과정

Model을 생성하기 위한 (훈련) 과정

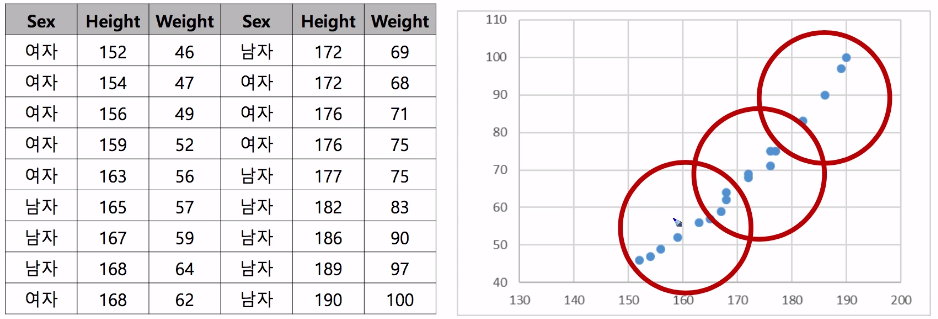
* **Regression**
* 회귀, 추세선을 긋는 것
* 데이터(X, Y)를 통해 X’의 결과 Y’을 예측(continuous Y)
* Ex) 주가 예측, 경제 성장률 예측, 영화 관람객 예측



* **Classification**
* 분류, 데이터의 유형을 나누는 것
* 기존 데이터를 바탕으로 데이터 유형을 나눠 보기
* 데이터(X, Y)를 통해 X’의 Y’을 분류(Discrete Y)
* Ex) 스팸메일 분류, 신문기사 분류



* **Clustering**
* 군집, 데이터를 모으는 것
* 사전 정보 없이 데이터 유형을 나누기
* 데이터(X)를 통해 X를 n개로 나누기
* Ex) 고객 집단 나누기



* **Supervised learning(지도학습)**

어떤 입력에 대해서 어떤 결과가 나와야 하는지 사전 지식을 갖고 있는 경우 해당 입력에 대해 특정 출력(label)이 나오도록 하는 규칙을 찾아낸다. 보통은 입력과 출력 쌍으로 구성되는 학습 데이터(training data)에 의해 입력으로부터 출력을 끌어내는 규칙(rule)을 발견하는 것을 학습의 목표로 하며, 흔히 회귀(regression) 방법이 여기에 해당한다.

* **Unsupervised learning(자율학습, 비지도 학습)**

입력은 있지만 정해진 출력이 없는 경우를 말하며, 순수하게 데이터를 갖고 있는 속성들을 이용해 그룹으로 나누는 경우를 말하며, 지도학습이 회귀방법을 사용하는 것과 달리, 군집화(clustering)에 해당된다. 가령 나이, 학력, 성별, 출생 지역 등의 조합에 따라 어떤 정당을 지지하는지 살펴보는 것도 자율 학습에 속한다. 일반적으로 시장 조사, 컴퓨터 클러스터링, 그림이나 동영상에 대한 auto-tagging 등에 사용된다.

* **Reinforcement learning(강화 학습)**

로봇의 학습 등에 사용할 수 있으며, 자신과 환경과의 상호 관계에 따라 자신의 행동을 개선해 나가는 학습법을 말한다. 가령 칭찬을 받은 행위는 더욱 많이 하고, 벌을 받을 만한 행위는 줄이는 것과 마찬가지로 적응성을 통해 학습을 강화해가는 것을 말한다. 물론 학습의 결과가 즉각적으로 나타나는 경우 효과적이라고 할 수 있다.

* **환경설정**
* 가상환경 설정

|  |
| --- |
| conda create -n ml\_scratch python=3.6  rm -rf ./anaconda/envs/ml\_scratch |

* 데이터 분석을 위한 python IDE jupyter notebook 설치

|  |
| --- |
| coda install jupyter |

* 데이터 분석 python 라이브러리 설치

|  |
| --- |
| conda install pandas |

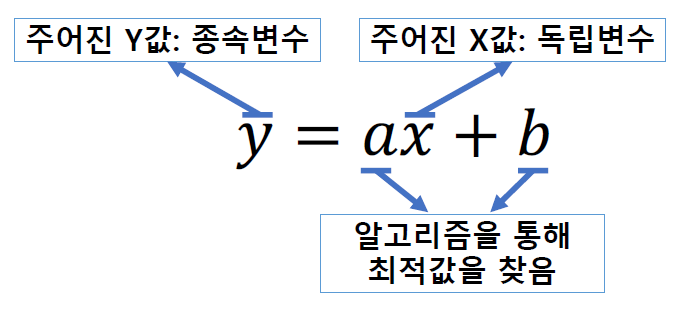
* 데이터 시각화를 위한 python package 설치

|  |
| --- |
| conda install matplotlib |

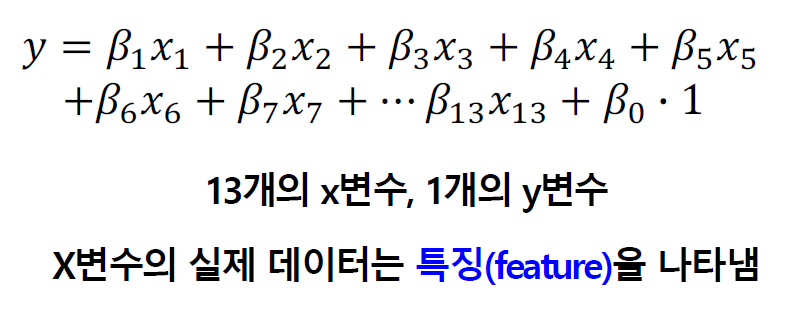
* **Jupyter Notebook**
* IPython 커널을 기반으로 한 대화형 python shell
* 일반적인 터미널 쉘 + 웹 기반 데이터 분석 Notebook 제공
* 미디어, 텍스트, 코드, 수식 등을 하나의 문서로 표현 가능
* 사실상의 데이터 분석 Interactive shell 표준
* Julia + Python + R

1. **Second Week**

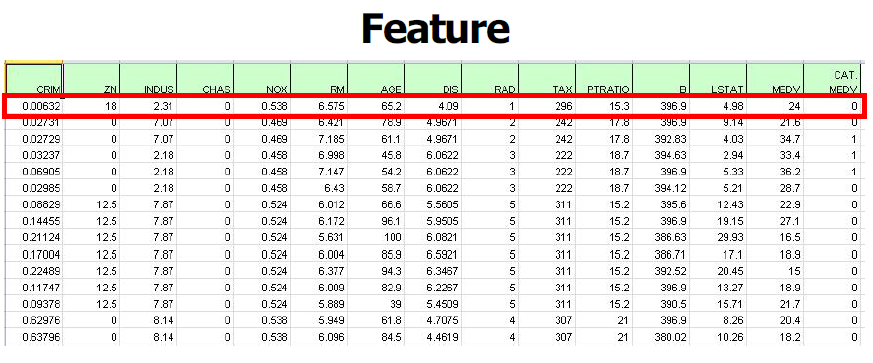
* **Key concept**
* Simple structure

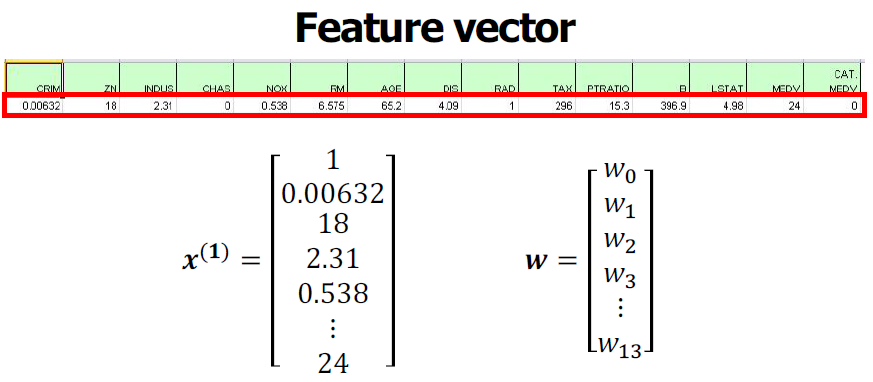


* Linear structure



* **Feature**
* 데이터의 특징을 나타내는 변수
* 일반적인 table에서 data를 표현할 때, column을 의미
* feature, independent variable, input variable 등 동일 의미로 사용
* 하나의 data instance (실제 데이터)는 feature vector로 표현: 테이블에서 data를 표현할 때, row를 의미
* **Feature vector**





* *Scalar:* italic

※ w: weight, i: index of data instance, j: index of feature

* **vector:** 소문자 bold
* **METRIX:** 대문자 bold

* **Course of dimensionality (차원의 저주)**
* 데이터의 feature가 증가할수록 데이터를 표현하는 공간이 증가한다.

1. Sparse vector increase. (값이 없는 feature가 늘어남)
2. Sample data size exponentially increase. (샘플데이터가 급속도록 늘어남)

* 데이터 분포나 모델 추정이 어려워진다.
* **Data type**

|  |  |
| --- | --- |
| **Continuous** | **Discrete** |
| 온도, 시험평균, 속도 | 성별, 우편주소, 등수 |
|  |  |

* Numeric types

단위(scale)가 존재하는 값으로 정량적인 측정이 가능한 data type

Ex) 온도, 속도, 거리

* Nominal types

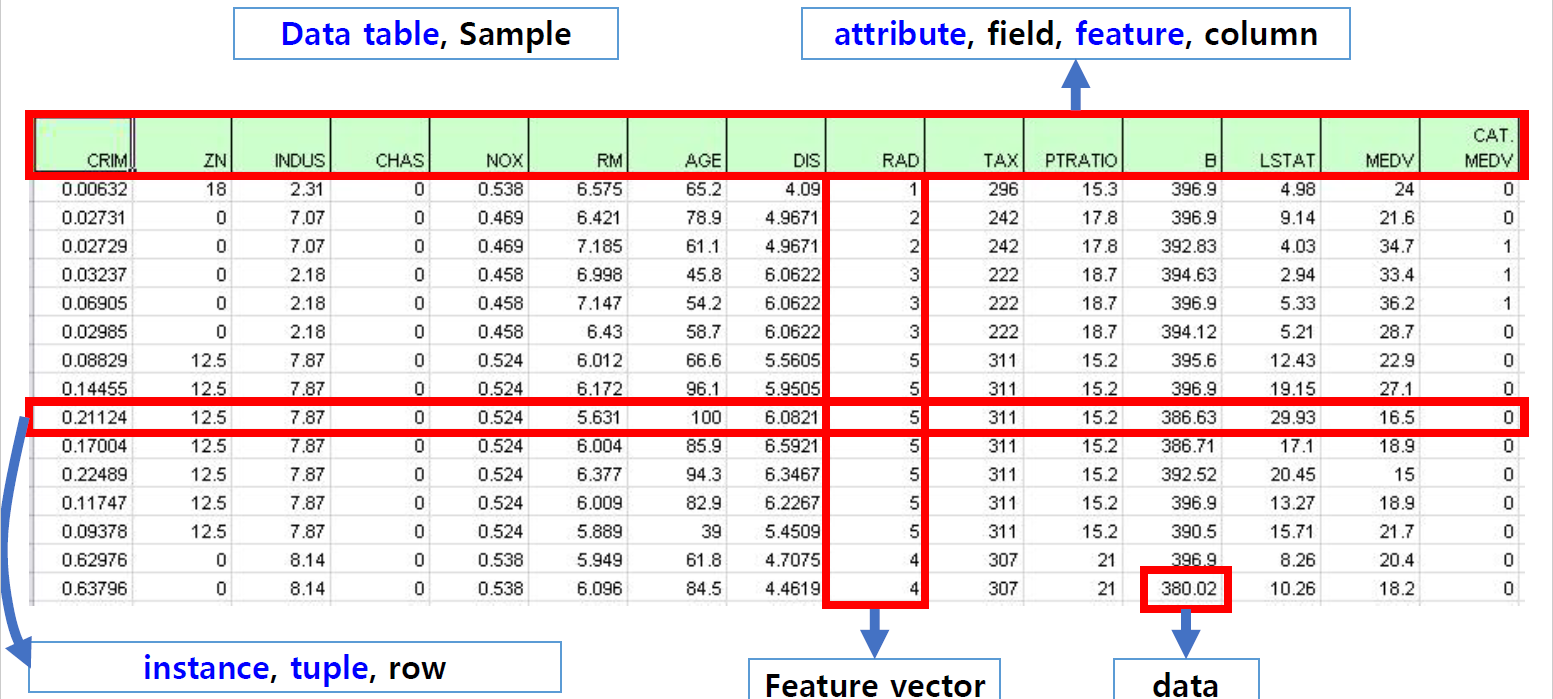
범주(category)로 분류가 가능한 data type

Ex) 학교명, 직업, 색깔

* Ordinal types

범주(category)로 분류가 가능하나 범주간의 순서가 있다. 하지만, scale이 없어 정량적인 측정이 불가능하다.

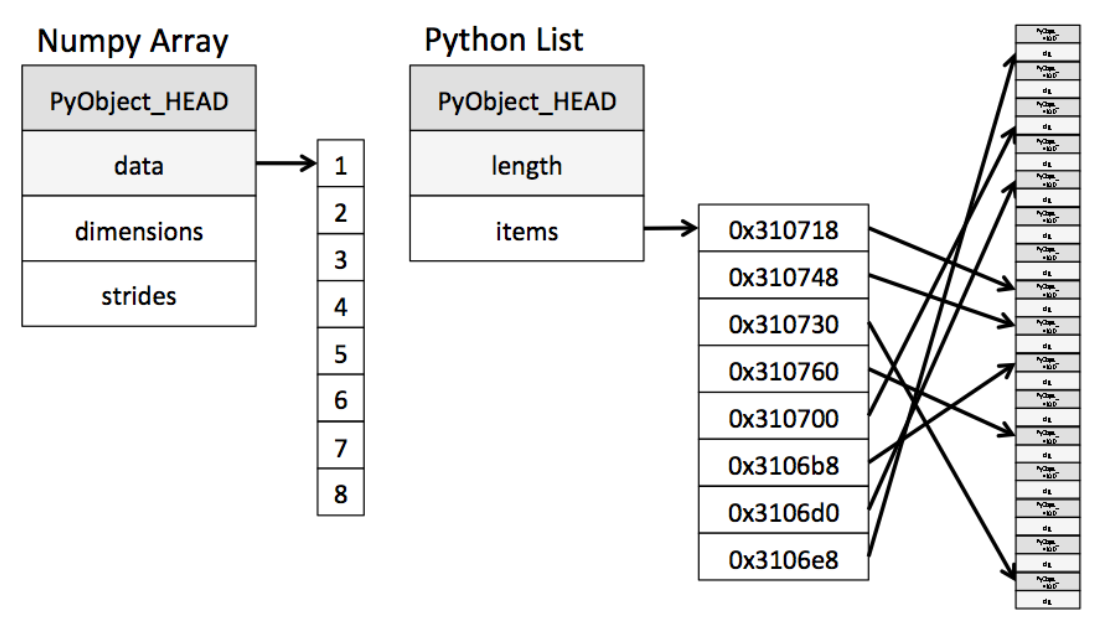
Ex) 음료수 병의 크기, 학점, 5점 척도 설문조사

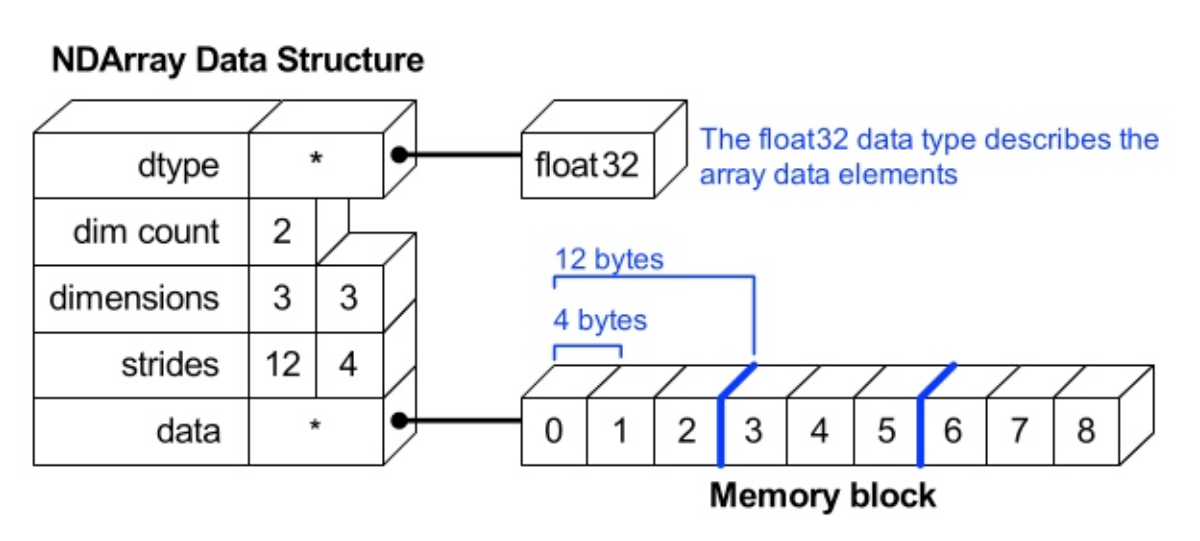


* **Pandas**
* 구조화된 데이터의 처리를 지원하는 Python 라이브러리
* 고성능 Array 계산 라이브러리인 Numpy와 통합하여, 강력한 “스프레드시트” 처리 기능을 제공
* 인덱싱, 연산용 함수, 전처리 함수 등을 제공함
* **Numpy**
* Numerical Python
* 파이선의 고성능 과학 계산용 기초 패키지
* Matrix와 Vector와 같은 array 연산의 표준
* 일반 list에 비해 빠르고, 메모리 효율적으로 사용
* 반복문 없이 데이터 배열에 대한 처리를 지원
* 선형대수와 관련된 다양한 기능을 제공
* C, C++, 포트란 등의 언어와 통합가능

1. **Third Week**

* **Numpy**
* Numerical Python
* 파이선의 고성능 과학 계산용 기초 패키지
* Matrix와 Vector와 같은 array 연산의 표준
* 일반 list에 비해 빠르고, 메모리 효율적으로 사용
* 반복문 없이 데이터 배열에 대한 처리를 지원
* 선형대수와 관련된 다양한 기능을 제공
* C, C++, 포트란 등의 언어와 통합가능
* **Numpy ndarray**
* 일반적으로 numpy는 np라는 alias를 이용해서 호출함
* np.array 함수를 활용하여 배열을 생성함 → ndarray
* 하나의 데이터 type만 배열에 저장 가능
* Dynamic typing not supported
* C의 Array를 사용하여 배열을 생성함





* **Handling shape**
* reshape(a, newshape, order)

|  |
| --- |
| np.array(matrix).reshape(4,2) |

* flatten()

|  |
| --- |
| np.array(tensor).flatten() |

* **indexing and slicing**
* indexing

|  |
| --- |
| test\_example = np.array([[1,2,3],[4,5,6]], int)  test\_example[row][col]  test\_example[row,col] |

* slicing

|  |
| --- |
| test\_example = np.array([[1,2,3,4,5],[6,7,8,9,10], int]  a[1:, ::2] # [6,8,10]  a[:, 1:3] # [[2,3],[7,8]] |

* **numpy creation functions**
* arange(start, stop, step, dtype)

범위를 지정하여, 값의 list를 생성하는 함수

|  |
| --- |
| np.arange(0,30,1).reshape(5,6) |

* ones, zeros and empty(shape, dtype, order)

지정한 shape과 같은 numpy.ndarry를 생성하는 함수

|  |
| --- |
| np.zeros(shape=(10,), dtype=np.int8)  np.ones((3,4))  np.empty(shape=(2,4)) |

* something\_like(a, dtype, order, subok)

|  |
| --- |
| test\_matrix = np.arange(30).reshape(5,6)  np.ones\_like(test\_matrix) |

* identity(n, dtype)

단위 행령을 생성하며 실수 연산을 위해 default dtype은 실수 계산을 위해 실수이다.

|  |
| --- |
| np.identity(3) |

* eye(N=row, M=col, k, dtype)

N\*M 행렬이면서 대각선이 1인 행렬을 생성한다.

k값은 대각선의 시작 index이고 양수는 행 기준이고 음수는 열 기준이다.

|  |
| --- |
| np.eye(3,5, k=-1) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |

* diag(v, k)

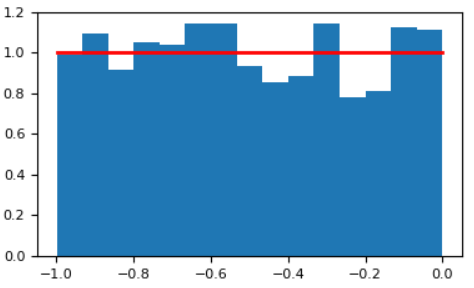
행렬에서 대각선(diagonal)의 값을 추출한다.

|  |
| --- |
| np.diag(matrix, k=1) |

* uniform(row, high, size)

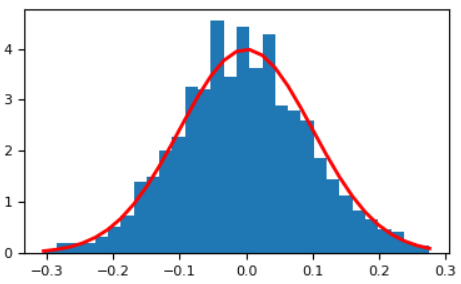
Draw samples from a uniform distribution

Samples are uniformly distributed over the half-open interval [low, high]



* normal(loc=mean, scale=sd, size)

Draw random samples from a Gaussian distribution

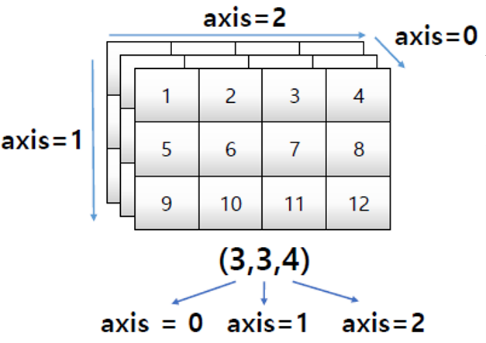


* **ndarray operation functions**
* sum, mean, std

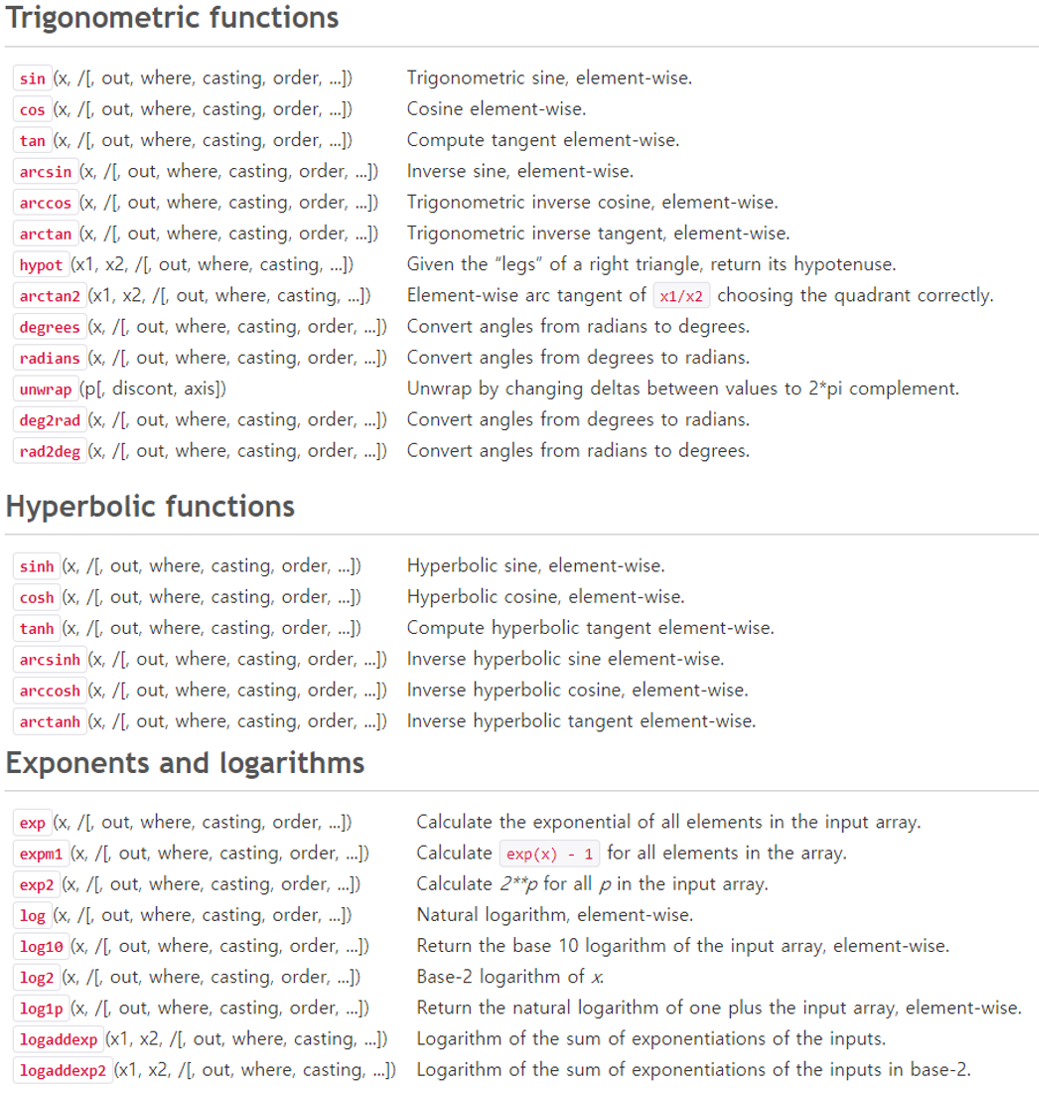
ndarray의 element들 간의 합을 구함, list의 sum 기능과 동일

* axis

operation function을 수행할 때, 기준이 되는 dimension 축



* Mathematical functions



* Concatenate
  + vstack(tup)

|  |
| --- |
| a = np.array(array([1,2,3])  b = np.array(array([1,2,3])  np.vstack((a,b)) |

* + hstack(tup)

|  |
| --- |
| a = np.array(array([1,2,3])  b = np.array(array([1,2,3])  np.hstack((a,b)) |

* + concatenate(tup, axis)

|  |
| --- |
| a = np.array(array([1,2] ,[3,4])  b = np.array(array([[5,6]])  np.concatenate((a,b.T), axis=1) |

1. **Fourth Week**