언론정보학부 광고홍보학과 손효지 20175074

Anaconda

-파이써만 설치할 수도 있지만 인공지능 구현을 위해 다른 패키지들도 설치할 필요가 있다

-인기있는 라이브러리가 모두 포함된 아나콘다(ananconda)를 설치

-아나콘다는 전세계 1100만명이 넘는 사용자가 오픈소스 배포판-업계표

-아나콘다: 데이터과해 및 기계학습을 위한 패키지들이 기본으로 포함

#아나콘다 설치 방법

1. www.anaconda.com/distribution/에 접속하여 화면 중간에 있는 파이썬 3.7 버전의 다운로드 버튼을 찾는다

2. 다운로드 버튼 아래에 64bit 32bit 버전중 본인의 PC의 운영체제에 맞는 버전을 클릭하면 된다

#아나콘다 Numpy 설정

Numpy는 일반적으로 많이 사용하는 모듈이기 때문에 기본으로 설치되어 있음

#아나콘다 내비게이터

박스에 체크가 되지 않았다면 체크하여 설치

#아나콘다 프롬프트 창

1번 명령문을 실행, pip가 최신버전이 아니면, 2번 명령문을 입력

Pip install Numpy -1

Python-m pip install –upgrade pip -2

Lesson 1 Numpy 기초문법

#Numpy 개요

#Numpy의 기초

#Numpy 라이브러리 개요

-Numpy: python을 위한 행렬 라이브러리

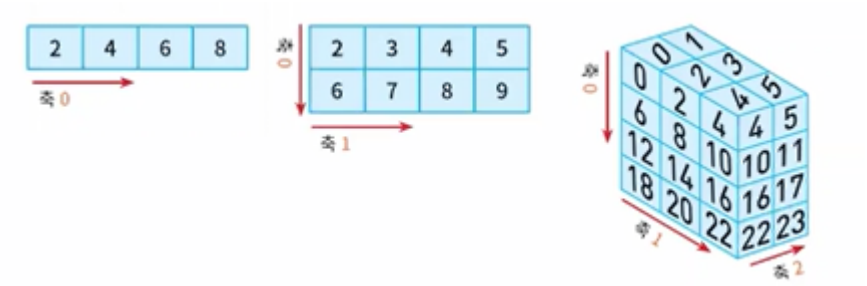
-과학적 계산을 위해 python에서 제작

-이산수학과 무작위 수 생성 등 수많은 작업을 할 수 있는 기능이 제공됨

-벡터와 행렬을 위한 특수한 배열 형식을 제공

-생성될 때 크기가 정해짐

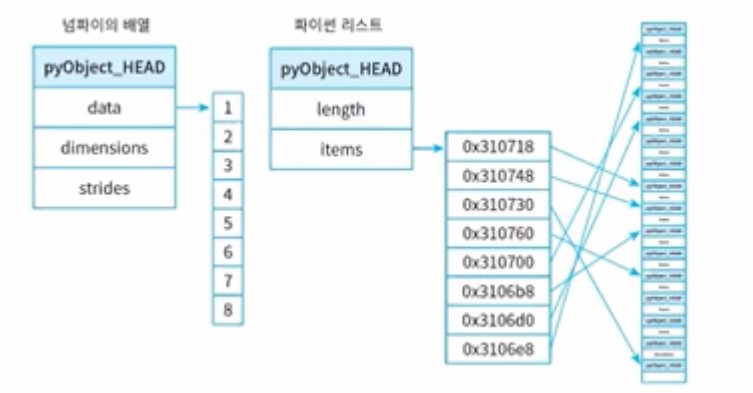
-텐서플로우와 잘 어울림



-기계 학습에서는 python의 기본 리스트로 충분하지 않다

-데이터를 처리할 때는 리스트와 리스트 간의 연산이 가능해야 하는데 python의 기본 리스트는 이것을 지원하지 않기 때문이다

-연산 속도도 중요하기 때문에 데이터 과학자들은 기본 리스트 대신에 Numpy를 선호한다



#1.Numpy 문법: 리스트 vs 넘파이 배열

```
In [2]: print("hello world")

hello world

In [4]: mid_scores=[10,20,30]
        final_scores=[70,80,90]
        total=[mid_scores+final_scores]
        print(total)

[[10, 20, 30, 70, 80, 90]]

In [5]: import numpy as np
        mid_scores=np.array([10,20,30])
        final_scores=np.array([70,80,90])
        total=mid_scores+final_scores
        print(total)

[ 80 100 120]
```

#2. Numpy 문법: Numpy 배열2

Mean()

-Nump에서 제공하는 배열을 이용하면 배열에 저장된 원소들의 평균값을 계산

-array()라는 메소드를 사용하여 1차원 배열 생성

-여기서 x라는 객체를 만들어 제공하는 mean()이라는 메소드를 이용하여 해당 값을 구할 수 있다

명령문	① ② ③	Numpy 모듈 가져오기 파이썬의 데이터형(예 : 리스트)을 numpy 형식으로 변환 Numpy에서 제공하는 기능수행(예 : 평균)
출력	① ② ③	import numpy as np x = np.array([1, 3, 5]) print(x.mean())

```
In [8]: import numpy as np
        x=np.array([1,3,5])
        print(x.mean())

        3.0
```

#3. Numpy 문법: Numpy 배열 3

Shape()

-shape은 해당 클래스에서 제공해주는 속성(attribute)

-1차원 배열로 3개의 원소가 있다는 결과가 나옴

명령문 print(x.shape) #(3,) 출력

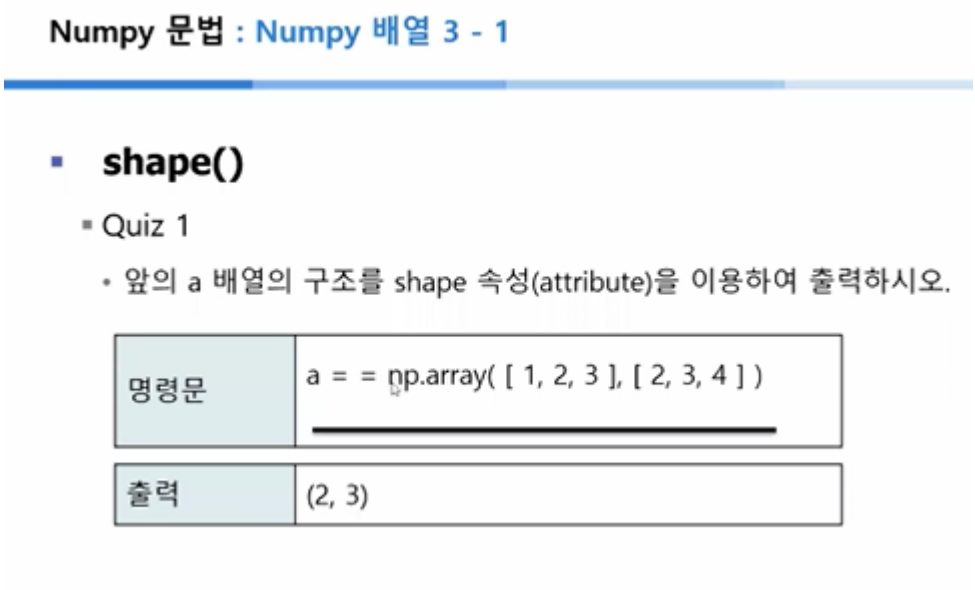
-numpy에서 지원하는 2차원 배열을 다음과 같이 만들면

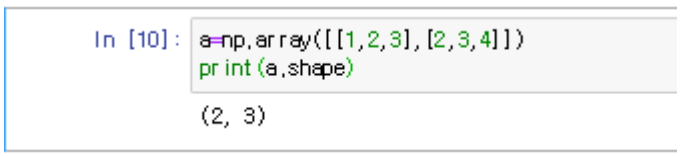
명령문 a=np.array([[1,2,3],[2,3,4]])

-a라는 배열은 2개의 원소를 가지고 있으며, 이들은 각각 3개의 원소를 가지고 있는 형태

-"numpy 형식의 배열 a는 3개의 원소로 구성된 2개의 원소가 있다"라고 표현

```
In [9]: import numpy as np
        x=np.array([1,3,5])
        print(x.mean())
        print(x.shape)

        3.0
        (3,)
```





#4. Numpy 문법: Numpy 배열 3-2

Reshape()

-Numpy 형식으로 배열의 원소를 입력할 때는 반드시 다음의 예시와 같이 리스트형식으로 입력

명령문 x=np.array([1,3,5])

x=np.array(1,3,5)

-reshape() 메소드를 추가하면 다음과 같이 Numpy의 2차원 배열을 원하는 모양으로 생성할 수 있다

명령문 x=np.array([1,3,5,7,9,11]).reshape(3,2)

Print(x)

출력 [[13]

[57]

[9 11]]

```
In [12]: x=np.array([1,3,5,7,9,11]).reshape(3,2)
         print(x)

[[ 1  3]
 [ 5  7]
 [ 9 11]]
```

#5. Numpy 문법: Numpy 배열 1

Zeros()

-괄호 안에 입력된 숫자만큼의 원소를 생성하고 0으로 초기화

-한 쌍의 괄호 [ ] 안에 숫자가 1개이면 벡터(1차원)를, 두 쌍의 괄호 안에 숫자가 2개이면 행렬([2차원])을 생성

명령문 import numpy as np

x=np.zeros([2,3])

print(x)

출력 ([[0,0,0]

[0,0,0]])

```
In [15]: x=np.zeros([2,3])
         print(x)

[[0. 0. 0.]
 [0. 0. 0.]]
```

#6. Numpy 문법: Numpy 배열 3-4

-2차원 배열에 1의 값들을 채움

명령문 y=np.ones([3,4])

Print(y)

출력 [[1,1,1,1]

[1,1,1,1]

[1,1,1,1]]

```
In [17]: x=np.ones([3,4])
         print(x)

[[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]
```

#7. Numpy 문법: Numpy 배열 3-4

ones()

-다음의 첫번째 print() 문은 2차원 배열의 첫번째 원소인 1차원 배열 출력

-두번째 print()문은 첫번째 1차원 배열에 저장된 원소들의 평균값(mean)을 실수 형태로 출력한다

-세번째 print()문은 x배열 전체의 원소들의 평균값을 구해 출력

-네번째 print()문은 x배열의 (모양)형식을 출력한다

출력 x=np.array([[1,3,5],[2,4,6]])

print(x[1])

print(x[1].mean())

print(x.mean())

print(x.shape)

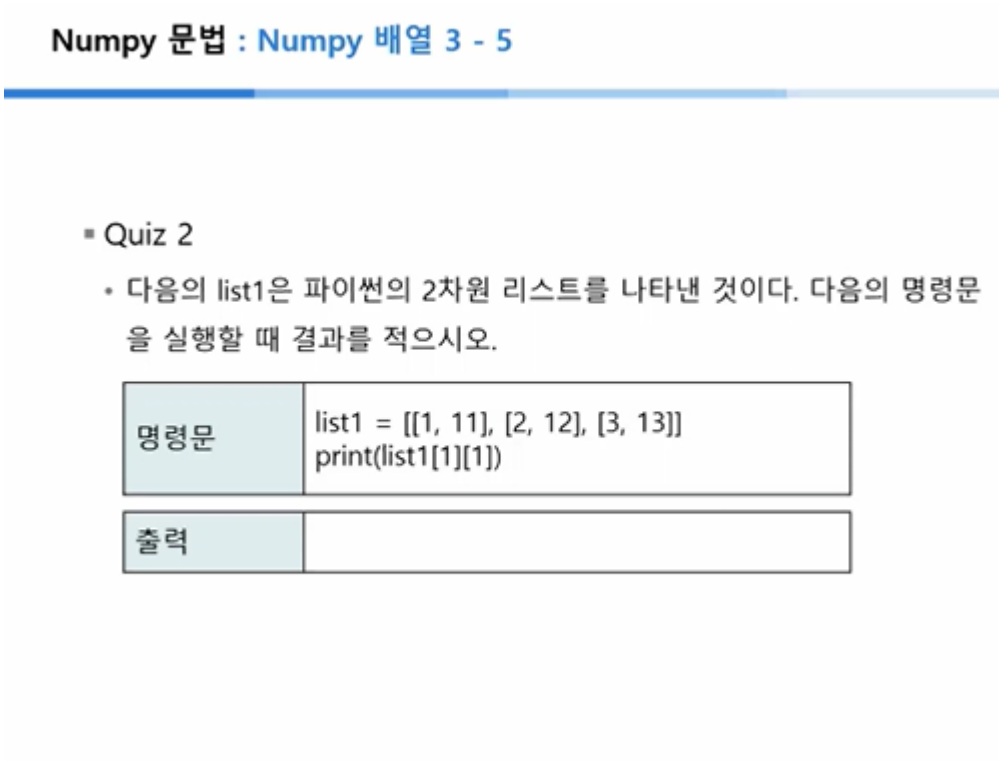
```
In [18]: x=np.array([[1,3,5],[2,4,6]])
         print(x[1])

[2 4 6]
```

```
In [19]: x=np.array([[1,3,5],[2,4,6]])
         print(x[0])

[1 3 5]
```

```
In [22]: x=np.array([[1,3,5],[2,4,6]])
         print(x[1])
         print(x[1].mean())

[2 4 6]
4.0
```

```
In [23]: x=np.array([[1,3,5],[2,4,6]])
         print(x[1])
         print(x[1].mean())
         print(x.mean())

[2 4 6]
4.0
3.5
```

```
In [24]: x=np.array([[1,3,5],[2,4,6]])
         print(x[1])
         print(x[1].mean())
         print(x.mean())
         print(x.shape)

         [2 4 6]
         4.0
         3.5
         (2, 3)
```



import numpy as np

x=np.array([[1,11],[2,12],[3,13]])

print(x[1][1])

[2,12]

답은 12