



주성분분석

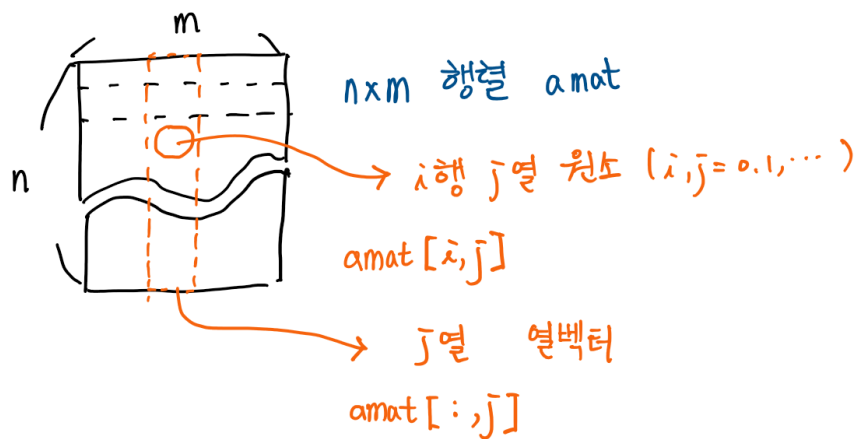
서울시립대학교 전종준

목차

- 행렬과 변환
- 주성분분석
- 행렬분해 (Singular Value Decomposition)와 데이터의 해석
- 1인가구 실습
- 반복문 연습

행렬과 변환

행렬의 원소와 열벡터



(코드) 행렬의 생성과 원소의 참조

```
import numpy as np
n = 3
m = 2
np.random.seed(1)
amat = np.random.uniform(size = (n,m))
amat[0,1]
amat[0,:]
amat[:,1]
```

행렬과 변환

행렬의 변환 1

$amat \times bvec = yvec$

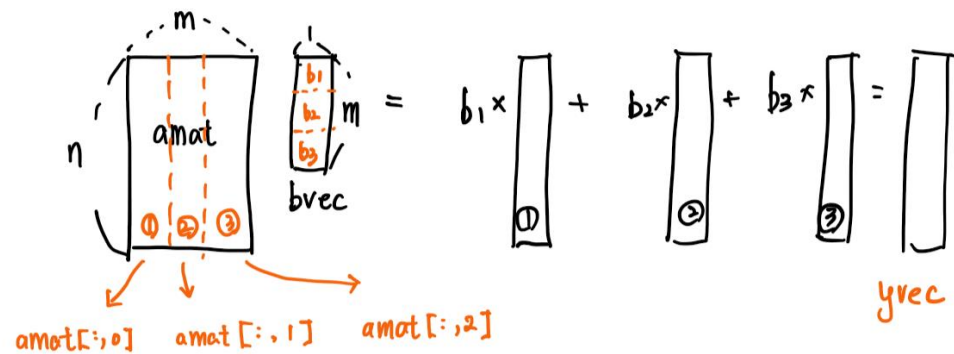
$amat : R^m \mapsto R^n$
($n \times m$ 행렬) $bvec$ (input) $yvec$ (output)

(코드) 행렬과 벡터의 곱

```
import numpy as np
n = 3
m = 2
np.random.seed(1)
amat = np.random.uniform(size = (n,m))
amat
bvec = np.array([0.1,0.5]).reshape(m,1)
bvec
amat@bvec
```

행렬과 변환

행렬의 변환 2

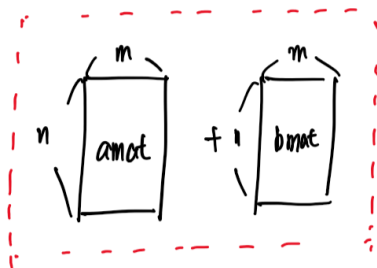


(코드) 행렬과 벡터의 곱

```
# columnwise operation
y1 = amat@bvec
y1.squeeze()
amat[:,0]*bvec[0]+amat[:,1]*bvec[1]
```

행렬과 변환

행렬의 덧셈



변환1 + 변환2
변환3

$$\underbrace{(A+B)}_C x = Ax + Bx$$

(코드) 행렬의 덧셈과 새로운 선형변환

행렬의 덧셈과 선형변환

```
np.random.seed(1)
```

```
amat_1 = np.random.uniform(size = (n,m))
```

```
amat_2 = np.random.uniform(size = (n,m))
```

```
amat_3 = amat_1 + amat_2
```

```
bvec = np.random.uniform(size = (m,1))
```

```
print( amat_1@bvec + amat_2@bvec, '\n' )
```

```
print (amat_3@bvec)
```

행렬과 변환

행렬의 곱셈과 변환

그림을 통한 이해!

$$\begin{array}{c}
 \begin{array}{|c|} \hline n \\ \hline \end{array} \begin{array}{|c|} \hline k \\ \hline \end{array} A \begin{array}{|c|} \hline k \\ \hline \end{array} \begin{array}{|c|} \hline m \\ \hline \end{array} B \begin{array}{|c|} \hline 1 \\ \hline \end{array} x = \begin{array}{|c|} \hline 1 \\ \hline \end{array} y \\
 (AB)x = y
 \end{array}$$

$$\cong A \cdot \underline{Bx} \\
 \in \mathbb{R}^k$$

$$x \xrightarrow{B} Bx \xrightarrow{A} A \cdot (Bx)$$

합성함수!

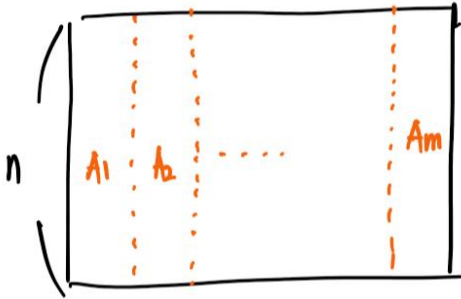
행렬과 변환

행렬의 열 벡터와 행렬의 이미지

- 행렬을 통해 만들어진 이미지(output)는 행렬 열벡터의 합으로 표현된다.

$$A = [A_1 : A_2 : \cdots : A_m]$$

$n \times m$ 행렬



$$x = (x_1, \cdots, x_m)^T$$

열벡터

$$x \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = x_1 A_1 + x_2 A_2 + \cdots + x_m A_m$$

행렬과 변환

행렬의 열 벡터와 행렬의 이미지

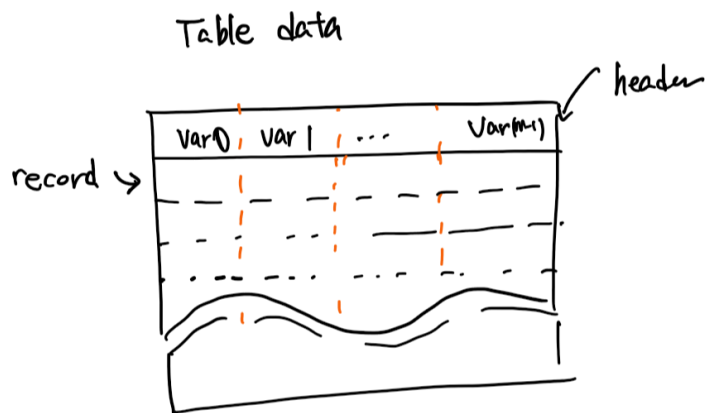
- 행렬을 통해 만들어진 이미지(output)는 행렬 열벡터의 합으로 표현된다.

코드

```
# columnwise operation
n = 10
p = 5
np.random.seed(1)
x = np.random.normal(size = (n,p))
betavec = np.random.uniform(size = (p,1))
y = x@betavec
z = np.zeros(n)
for i in range(p):
    z += x[:,i]*betavec[i]
z = z.reshape(-1,1)
np.concatenate((y,z), axis = 1)
```

데이터와 행렬

Tabular Data를 표시하는 전형적인 방법



그림

샘플 데이터를 불러보자

```
path = "C:/Users/jjjeo/Desktop/1인가구_시각화_교육용2/"  
data = pd.read_csv(path+'sample.csv', encoding='cp949')  
data.head()
```

BF_M3_AVG_CAL_USER	BF_M3_AVG_MSG_USER	HDAY_N_HOME_TOT_DURATION
59	16	54682
134	31	178399
47	14	408460
49	16	1269508
101	23	212467

코드에 대한 설명 파일을 불러보자

```
code_data = pd.read_csv(path + 'code_pre.csv',  
                          encoding = 'cp949')
```

코드에 대한 설명을 확인하자

```
code_data['속성명']
```

데이터와 행렬

차원 축약과 축약방법의 선택

- 지수화: 자료의 가중합을 통해 숫자의 생성

(자료의 평균에 대해 Normalization)

- 자료의 변동성을 가장 잘 나타내는 가중치를 제 1주성분이라 부름

- 제 1주성분과 내적이 0 (직교)면서 자료의 변동성을 잘 나타내는 가중치를 제 2주성분이라 부름

그림

0	최근 3개월 평균 문자량
1	최근 3개월 평균 통화대상자 수
2	최근 3개월 휴일 외출 건 수
3	복지로 app/web 사용 일수
4	휴일 집 근처 체류시간
5	평일 집 근처 체류시간
6	최근 3개월 평일 외출 건 수
7	최근 3개월 평일 이동거리
8	교육/학습 app/web 사용 일수
9	지하철이동일수 월별
10	생활 app/web 사용 일수

$$b_1^{(1)} \text{Var} 4 + b_2^{(1)} \text{Var} 5 + b_3^{(1)} \text{Var} 6 +$$

$$b_4^{(1)} \text{Var} 7 + b_5^{(1)} \text{Var} 9 = \text{지표 1}$$

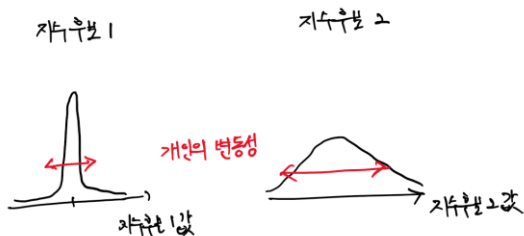
데이터와 행렬

주성분 벡터 찾기

- 표준화

- 집단 내에서 개인의 차이를 확인할 때 집단의 평균은 의미 없음
- 각 변수의 scale 효과를 조정

- 2개 이상의 주성분 벡터 찾기 (직교화 방법)



그림

데이터정규화

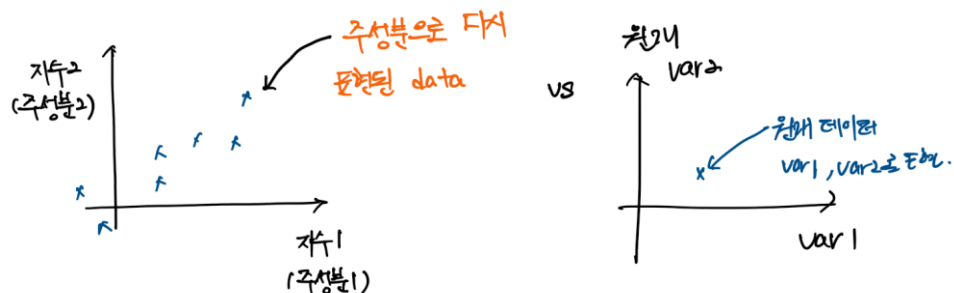
```
from sklearn.preprocessing import StandardScaler
x = StandardScaler().fit_transform(x)
x.mean(axis=0)
x.std(axis=0)
```

```
bvec = np.random.uniform(size = p)
bvec=bvec/np.linalg.norm(bvec)
(x[0,:]*bvec).sum()
y = x@bvec
plt.hist(y, bins = 200, range = (-10,10))
```

데이터와 행렬

주성분 벡터 찾기

- 표준화
 - 집단 내에서 개인의 차이를 확인할 때 집단의 평균은 의미 없음
 - 각 변수의 scale 효과를 조정
- 2개 이상의 주성분 벡터 찾기
(직교화 방법)



그림

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(x)
# 설명력
pca.explained_variance_ratio_
#
pca.singular_values_

pc_score = pca.fit_transform(x)
pc_score
import matplotlib.pyplot as plt
plt.plot(pc_score[:,0], pc_score[:,1], '.')
x.shape
```

데이터와 행렬

주성분 값을 이용한 데이터의 재표현

- 원본 데이터의 표현

- 주성분 벡터를 이용한 재표현

그림

```
# care about full_matrices = False
u, s, vh= np.linalg.svd(x, full_matrices=False)
u.shape
s.shape
vh.shape
(vh[0,:]*vh[1,:]).sum()

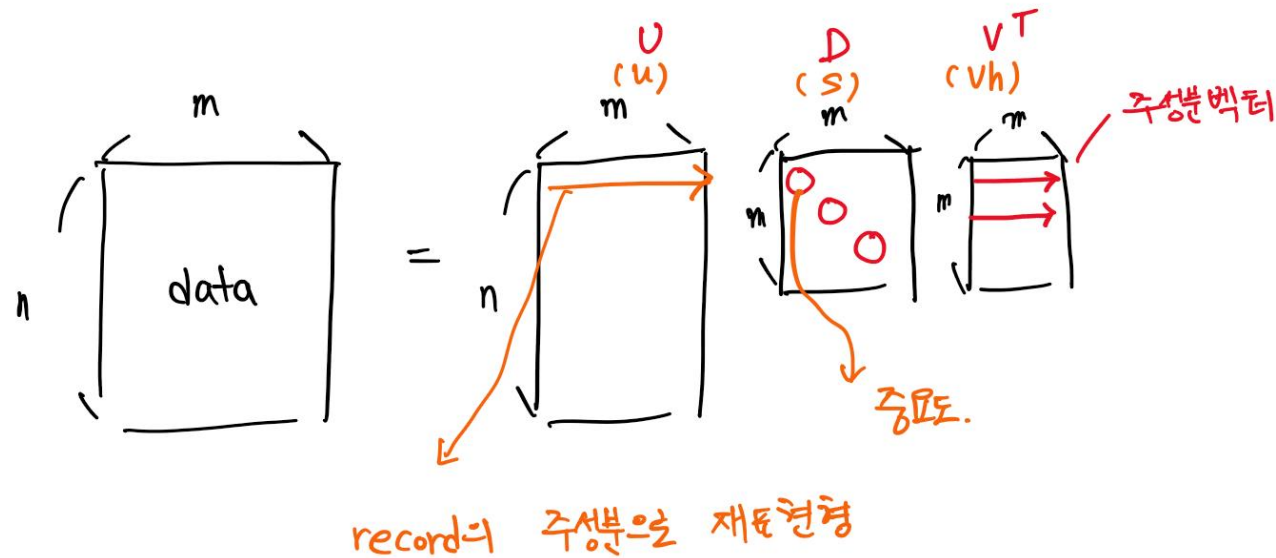
# 해석
features
vh[0,:]
vh[1,:]
# 0번 obs의 표현형
u[0,:]
# scaled
plt.plot(u[:,0],u[:,1], '.')
```

데이터와 행렬

Singular Value Decomposition (특이치분해)

그림

- 데이터의 재표현



실습 1

SVD (10분)

반복문

for 문 연습하기

그림

- for k in range(10):
- for k in list_a:
(여기서 list_a)는 list 변수
- for j, k in [(1,2), (10,20)]:
- [list_a[k] for k in [1,3,5,7]]

반복문

for 문 연습하기

그림

- list_a = ['Buddy', 'you're', 'a boy', 'make', 'a big noise']
- enumerate(list_a)
- zip(range(len(list_a)), list_a)
- for j,k in enumerate(list_a):

```
# for 문 연습하기
list_a = ['Buddy', 'you're', 'a boy', 'make', 'a big noise']

enumerate(list_a)

zip(range(len(list_a)), list_a)

for j,k in enumerate(list_a):
    print(j, k)

for j,k in zip(range(len(list_a)), list_a):
    print(j, k)
```

실습 2

APP 사용 데이터 지수 만들기 실습 (30분)