데이터 과학

L12.1: PCA Practice

Kookmin University

모듈 불러오기

- 사용할 모듈 import 하기

```
import torch
import requests
import matplotlib.pyplot as plt
```

Iris dataset

- 아이리스(붓꽃) 데이터
  - 붓꽃 종류별로 꽃받침과 꽃잎의 길이 및 너비를 측정한 데이터
  - https://archive.ics.uci.edu/ml/datasets/Iris

```
4.6,3.2,1.4,0.2,Iris-setosa
5.3,3.7,1.5,0.2,Iris-setosa
5.0,3.3,1.4,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
```

데이터 불러오기

```
iris_url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
r = requests.get(iris_url)
open('iris.data', 'wb').write(r.content)

vectors = []
answers = []
with open('iris.data', 'r') as f:
   for line in f:
       line = line.strip()
       if len(line) != 0:
           items = line.split(",")
           vectors.append([float(x) for x in items[:4]])
           answers.append(items[4])

species = {a: i for i, a in enumerate(set(answers))}
```

Tensor로 변환, 중심 옮기기

```
X = torch.FloatTensor(vectors)
Z = (X-torch.mean(X, axis=0))
```

findPC(): 주성분 찾기 함수

- 분산을 최대로하는 w벡터 찾기

```
def findPC(Z):
    w = torch.randn(Z.shape[1])
    w = w/(torch.dot(w,w))**0.5

    lr = 0.1

    for epoch in range(1001):
        w.requires_grad_(True)
        variance = torch.mean(torch.sum(Z * w, dim=1) ** 2)

        variance.backward()
        with torch.no_grad():
            w = w + lr * w.grad
            w = w/(torch.dot(w,w))**0.5

    return w, variance.item()
```

PCA()

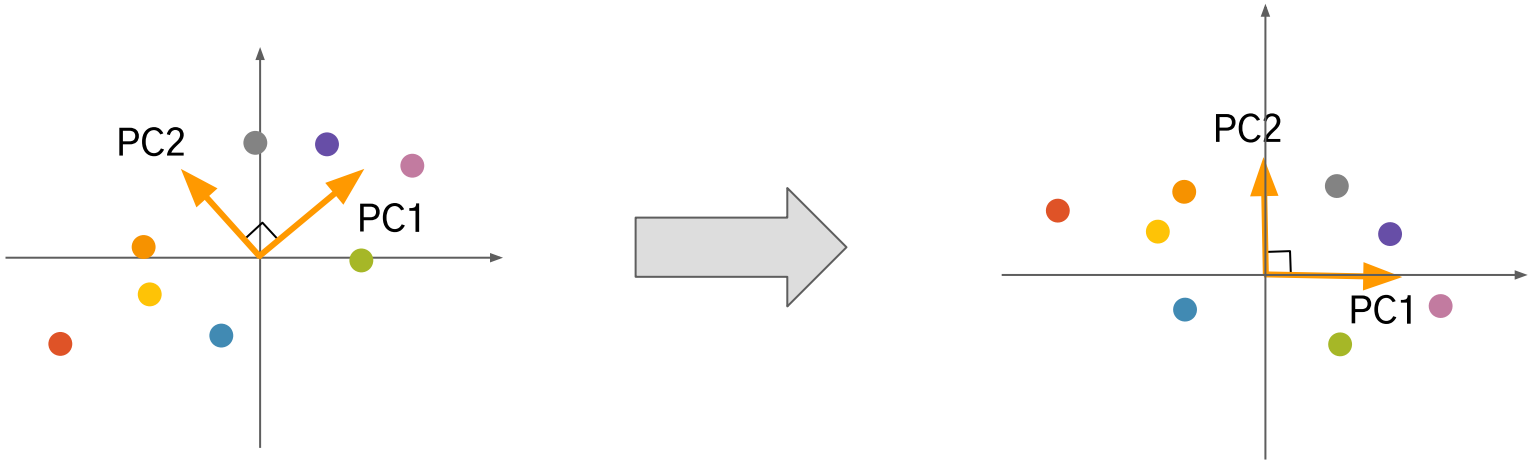
- 순서대로 주성분 찾기

```
def PCA(Z, n):
    W, V = [], []

    for _ in range(n):
        w, v = findPC(Z)
        W.append(w)
        V.append(v)
        Z = Z - (Z @ w.view(-1,1)) * w

    return W, V
```

transform()

- 각 축이 주성분이 되도록 기존 데이터를 변경

```
def transform(Z, W):
    return Z @ torch.Tensor(W).T
```



PCA 구해보기

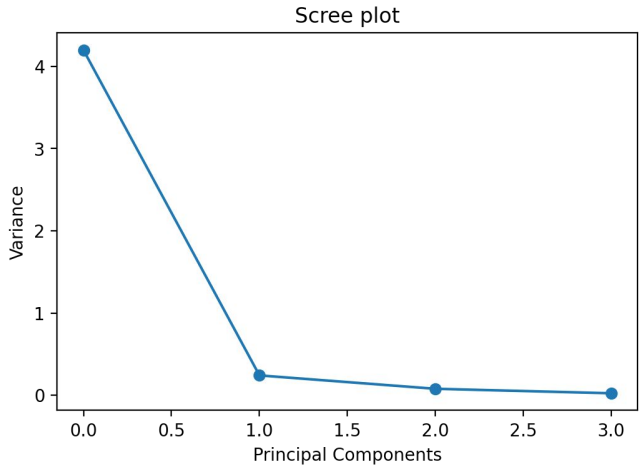
- PCA를 구하고, 데이터 변환

```
W, V = PCA(Z, 4)
K = transform(Z, W)
```

Scree plot 그리기

- 각 PC별 분산 값 확인

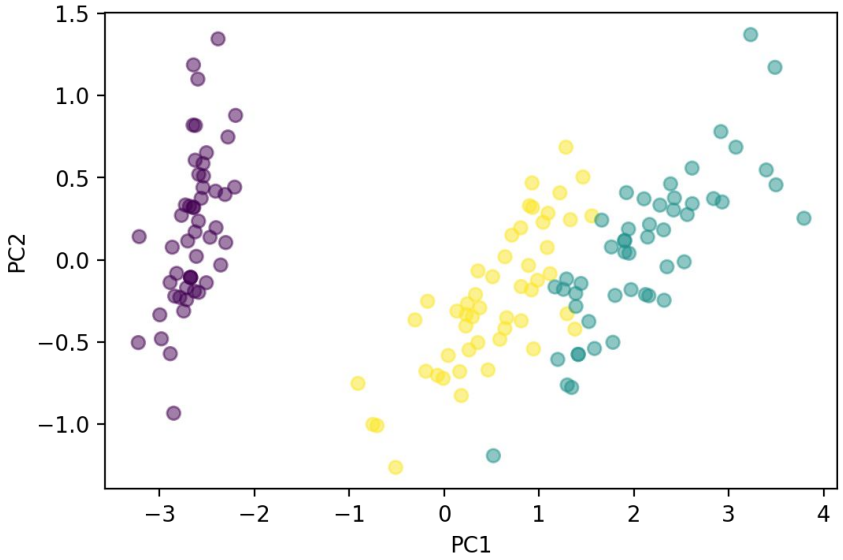
```
plt.title("Scree plot")
plt.xlabel("Principal Components")
plt.ylabel("Variance")
plt.plot(range(4), V, "o-")
plt.show()
```



변환된 그래프 그리기

- PC1과 PC2를 축으로 하는 그래프 그리기

```
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.scatter(K[:,0], K[:,1], c=[species[a] for a in answers], alpha=0.5)
plt.show()
```



Questions?