zero-base /

Life Changing Education

Chapter 22. Principal Component Analysis

PCA란?

zero-base /

PCA란

- 데이터 집합 내에 존재하는 각 데이터의 차이를 가장 잘 나타내 주는 요소를 찾아 내는 방법
- 통계 데이터 분석(주성분 찾기), 데이터 압축(차원감소), 노이즈 제거 등 다양한 분야에서 사용

zero-base /

간단한 PCA의 개념

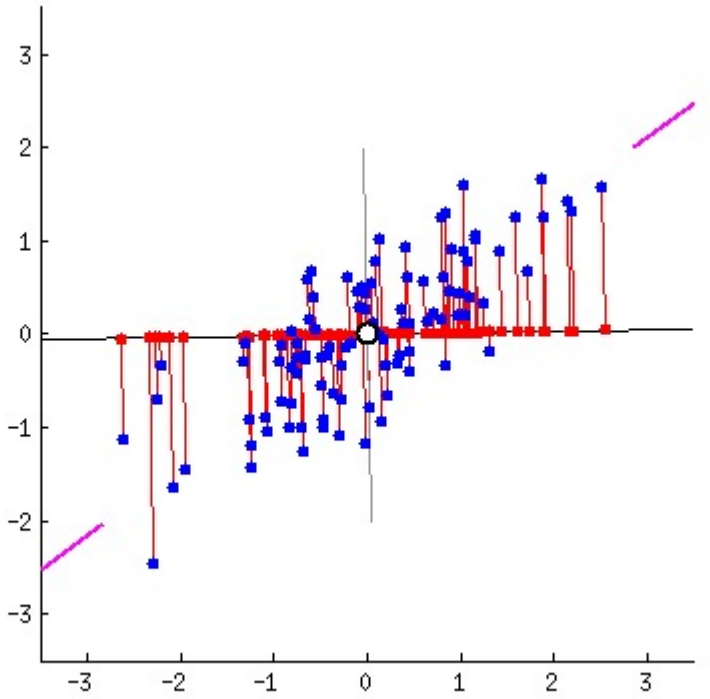
- **차원축소**(dimensionality reduction)와 **변수추출**(feature extraction) 기법으로 널리 쓰이고 있는 **주성분분석**(Principal Component Analysis)

- PCA는 데이터의 **분산**(variance)을 최대한 보존하면서 서로 직교하는 새 기저(축)를 찾아, 고차원 공간의 표본들을 선형 연관성이 없는 저차원 공간으로 변환하는 기법

- **변수추출**(Feature Extraction)은 기존 변수를 조합해 새로운 변수를 만드는 기법 (**변수선택**(Feature Selection)과 구분할 것)

zero-base /

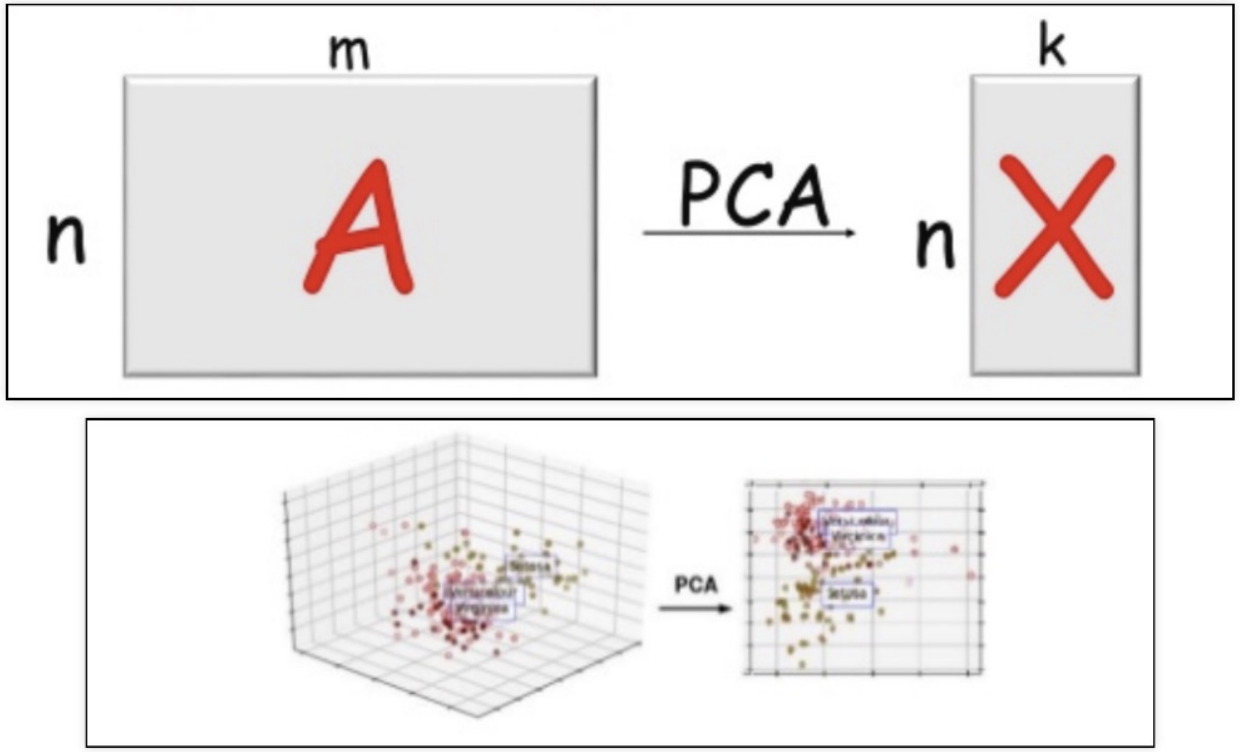
PCA



- 데이터를 어떤 벡터에 정사영시켜 차원을 낮출 수 있음

zero-base /

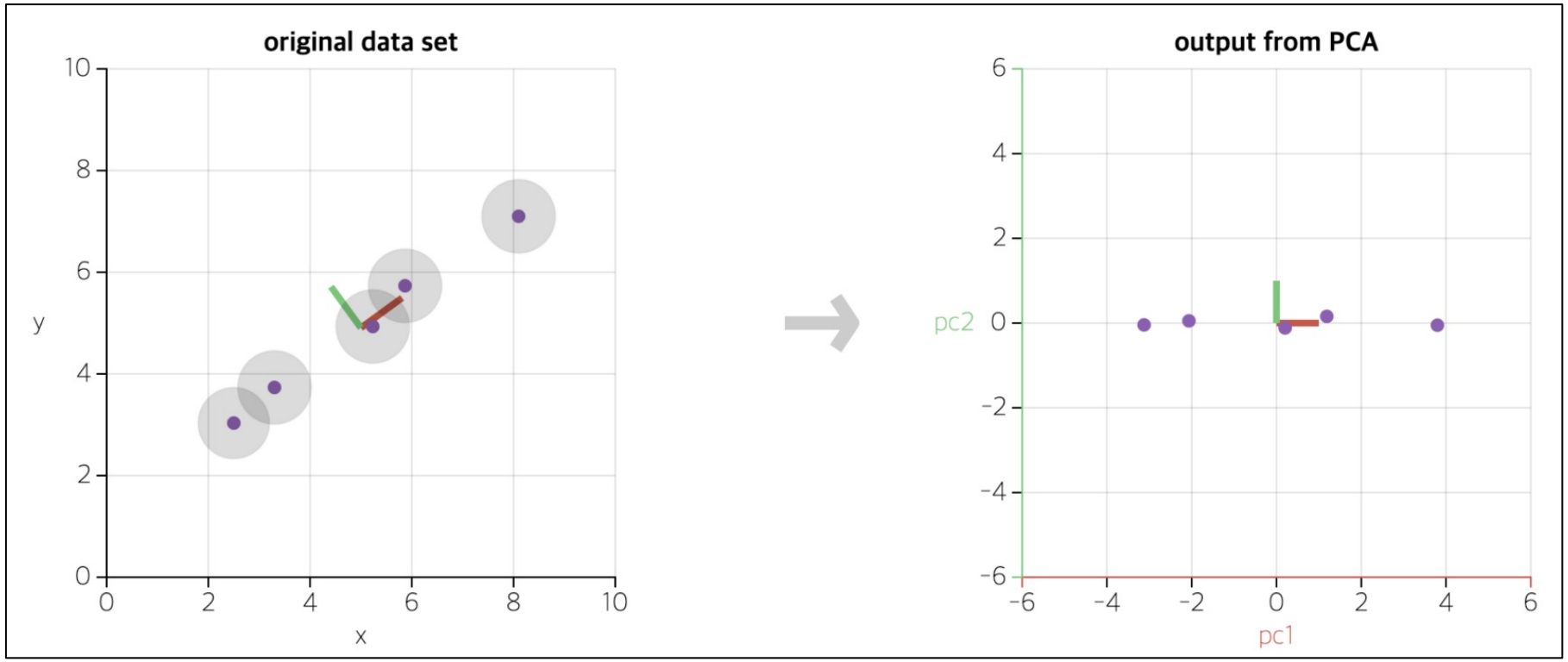
벡터를 이용해서 데이터를 다시 표현하기



- 데이터를 변환시켰을 때, 어떤 벡터를 선정하면 본래 데이터 구조를 가장 잘 유지할 수 있을까.

PCA란?

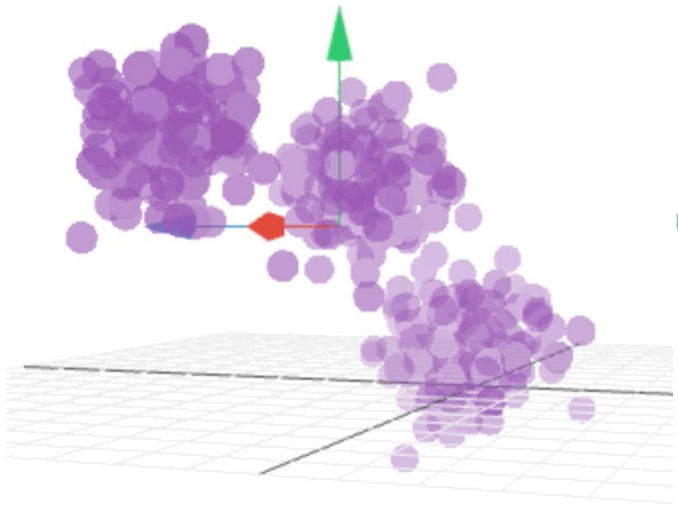
데이터를 새로운 축으로 표현하는 것

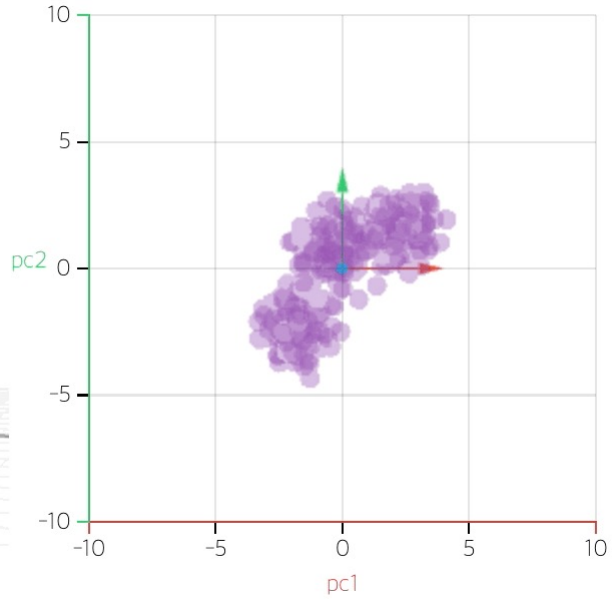


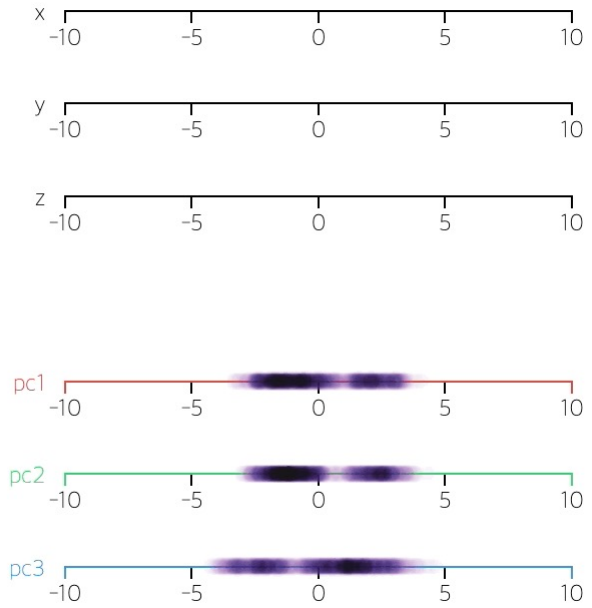
PCA란?

zero-base /

차원이 많은 경우 간단하게 표현해 볼 수 있다







sklearn을 이제 이용해보자

zero-base /

간단하게 데이터를 만들어보고

```
import seaborn as sns
sns.set_style('whitegrid')

rng = np.random.RandomState(13)
X = np.dot(rng.rand(2, 2), rng.randn(2, 200)).T
X.shape
```

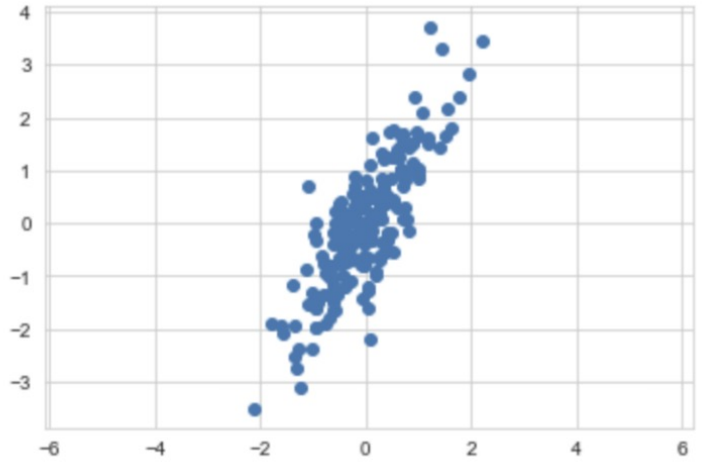
```
(200, 2)
```

- import numpy as np

zero-base /

이렇게 생긴 데이터

```
plt.scatter(X[:,0], X[:,1])
plt.axis('equal');
```



- import matplotlib.pyplot as plt

sklearn을 이제
이용해보자

fit~

```
pca = PCA(n_components=2, random_state=13)
pca.fit(X)
```

```
PCA(copy=True, iterated_power='auto', n_components=2, random_state=13,
    svd_solver='auto', tol=0.0, whiten=False)
```

zero-base /

벡터와 분산값

```
pca.components_
```

```
array([[ 0.47802511,  0.87834617],
       [-0.87834617,  0.47802511]])
```

```
pca.explained_variance_
```

```
array([1.82531406, 0.13209947])
```

zero-base /

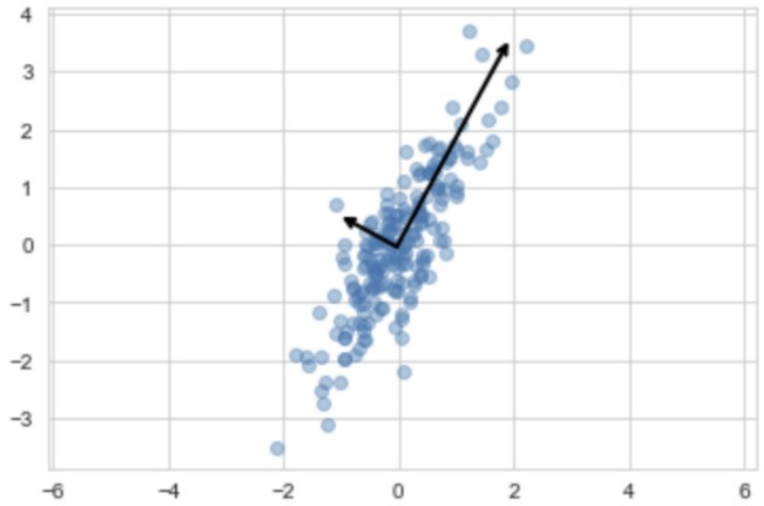
주성분 벡터를 그릴 준비를 하고

```
def draw_vector(v0, v1, ax=None):
    ax = ax or plt.gca()
    arrowprops = dict(arrowstyle='->',
                      linewidth=2, color='black',
                      shrinkA=0, shrinkB=0)
    ax.annotate('', v1, v0, arrowprops=arrowprops)
```

sklearn을 이제 이용해보자

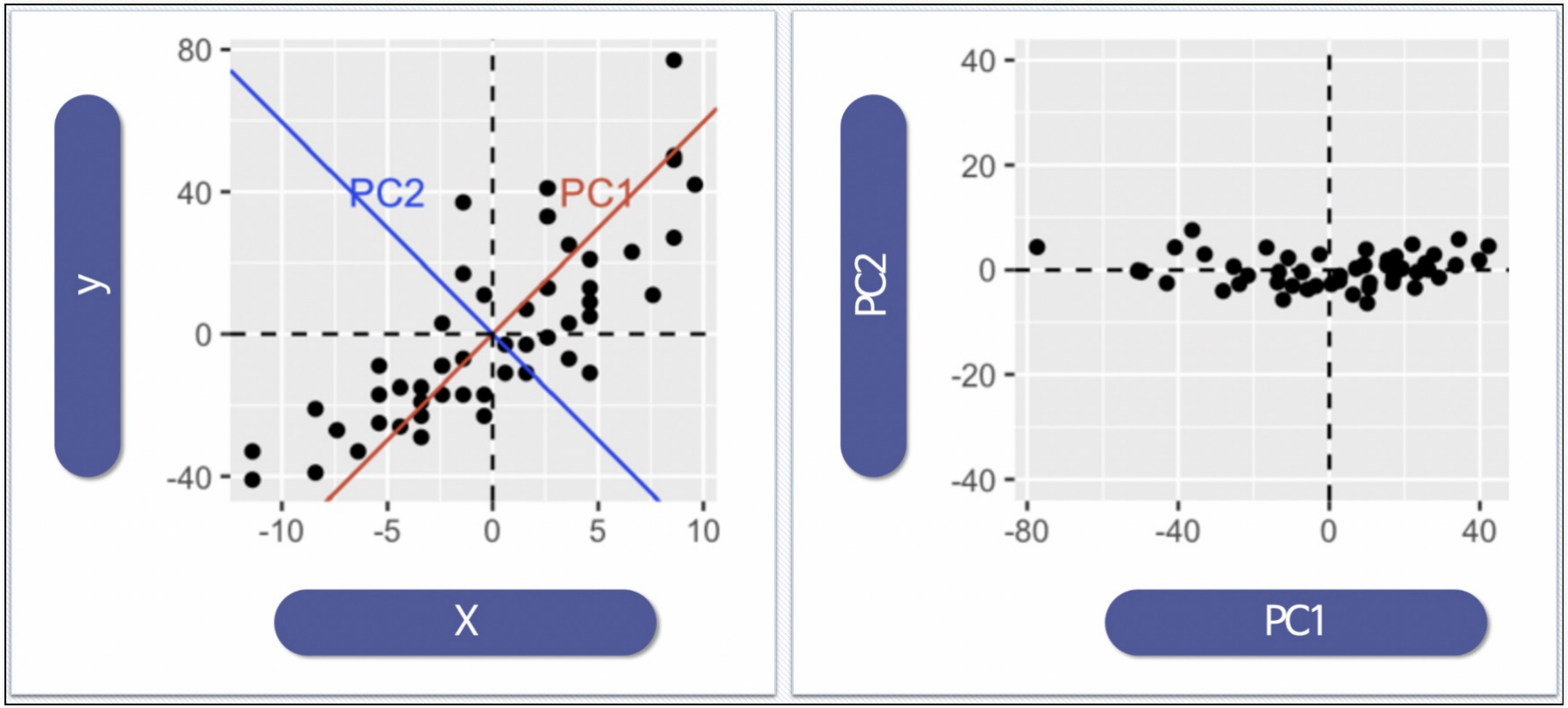
그려보자~

```
plt.scatter(X[:, 0], X[:, 1], alpha=0.4)
for length, vector in zip(pca.explained_variance_, pca.components_):
    v = vector * 3 * np.sqrt(length)
    draw_vector(pca.mean_, pca.mean_ + v)
plt.axis('equal')
plt.show()
```



zero-base /

데이터의 주성분을 찾은 다음 주축을 변경하는 것도 가능



zero-base /

이번에는 n_components를 1로 두고

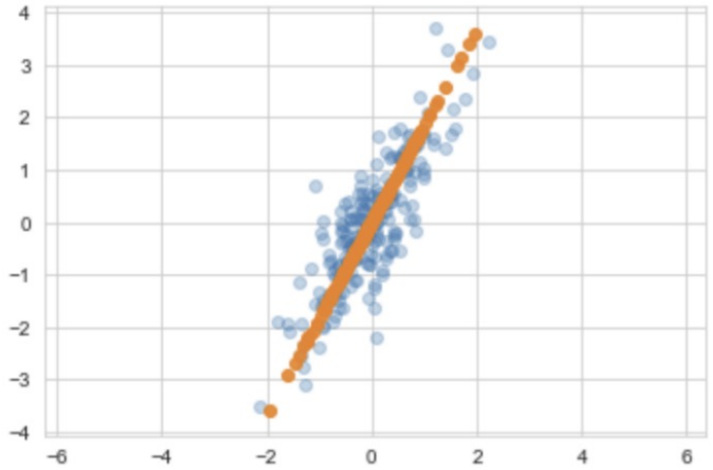
```
pca = PCA(n_components=1, random_state=13)
pca.fit(X)
X_pca = pca.transform(X)

print(pca.components_)
print(pca.explained_variance_)
```

```
[[0.47802511 0.87834617]]
[1.82531406]
```

zero-base /

어쩌면 linear regression과 같은 결과일지도~

```
X_new = pca.inverse_transform(X_pca)
plt.scatter(X[:, 0], X[:, 1], alpha=0.3)
plt.scatter(X_new[:, 0], X_new[:, 1], alpha=0.9)
plt.axis('equal')
plt.show()
```



실습 - Iris data

zero-base /

추억의 iris 데이터

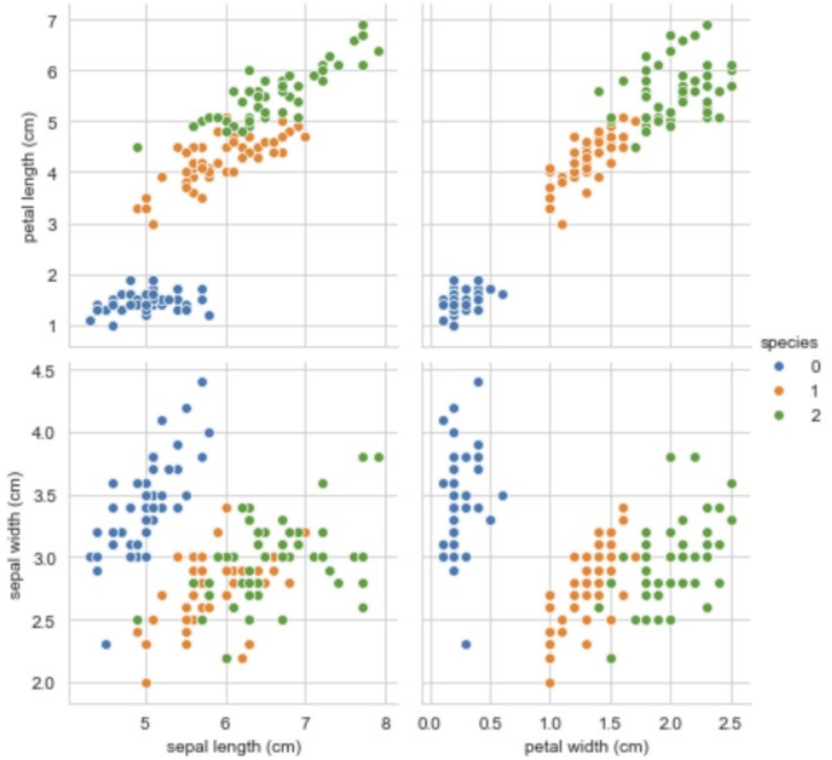
```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

iris_pd = pd.DataFrame(iris.data , columns=iris.feature_names)
iris_pd['species'] = iris.target
iris_pd.head(3)
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0

zero-base /

특성 4개를 한 번에 확인하기는 어렵다

```
sns.pairplot(iris_pd, hue='species', height=3,
             x_vars=['sepal length (cm)', 'petal width (cm)'],
             y_vars=['petal length (cm)', 'sepal width (cm)']);
```



zero-base /

일단 Scaler를 적용하고

```
from sklearn.preprocessing import StandardScaler

iris_ss = StandardScaler().fit_transform(iris.data)
iris_ss[:3]
```

```
array([[-0.90068117,  1.01900435, -1.34022653, -1.3154443 ],
       [-1.14301691, -0.13197948, -1.34022653, -1.3154443 ],
       [-1.38535265,  0.32841405, -1.39706395, -1.3154443 ]])
```

zero-base /

pca 결과를 return하는 함수도 하나 만들어 두고

```
from sklearn.decomposition import PCA

def get_pca_data(ss_data, n_components=2):
    pca = PCA(n_components=n_components)
    pca.fit(ss_data)

    return pca.transform(ss_data), pca
```

zero-base /

return값 확인해보고

```
iris_pca, pca = get_pca_data(iris_ss, 2)
iris_pca.shape
```

```
(150, 2)
```

```
pca.mean_
```

```
array([-1.69031455e-15, -1.84297022e-15, -1.69864123e-15, -1.40924309e-15])
```

```
pca.components_
```

```
array([[ 0.52106591, -0.26934744,  0.5804131 ,  0.56485654],
       [ 0.37741762,  0.92329566,  0.02449161,  0.06694199]])
```

zero-base /

pca가 적용된 결과

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

PCA
(2 components)

	principal component 1	princial component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

zero-base /

pca 결과를 pandas로 정리

```
def get_pd_from_pca(pca_data, cols=['pca_component_1','pca_component_2']):
    return pd.DataFrame(pca_data, columns=cols)
```

zero-base /

간단히 4개의 특성을 두 개의 특성으로 정리

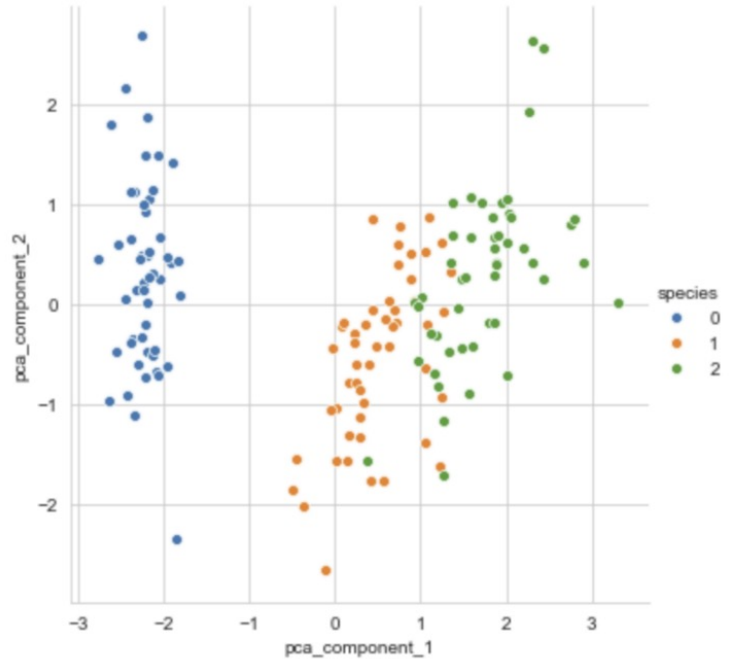
```
iris_pd_pca = get_pd_from_pca(iris_pca)
iris_pd_pca['species'] = iris.target
iris_pd_pca.head(3)
```

	pca_component_1	pca_component_2	species
0	-2.264703	0.480027	0
1	-2.080961	-0.674134	0
2	-2.364229	-0.341908	0

zero-base /

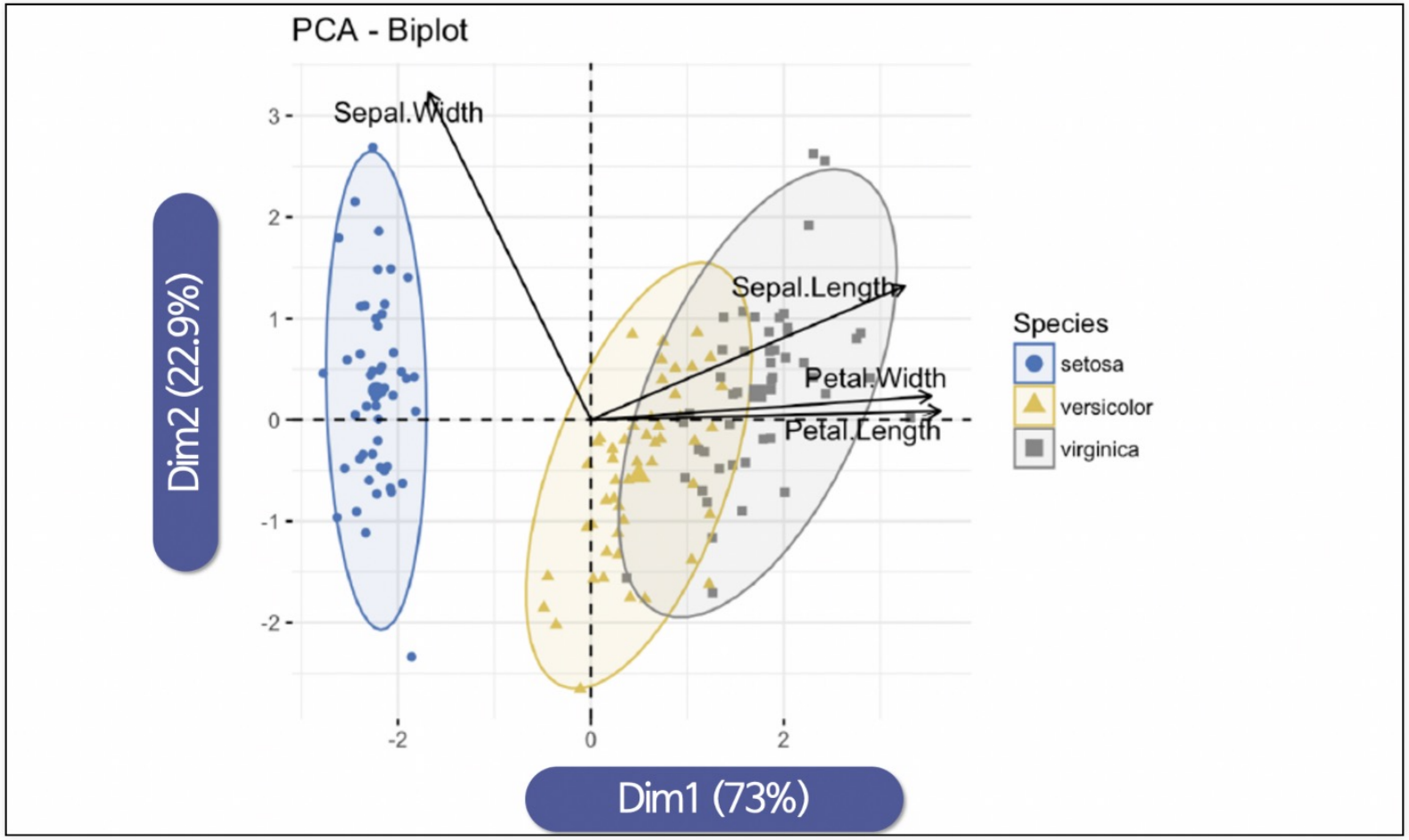
두 개의 특성을 그려보자

```
sns.pairplot(iris_pd_pca, hue='species', height=5,
             x_vars=['pca_component_1'], y_vars=['pca_component_2']);
```



zero-base /

주성분 분석의 위력



zero-base /

4개 특성을 모두 사용해서 randomforest에 적용하면

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

def rf_scores(X, y, cv=5):
    rf = RandomForestClassifier(random_state=13, n_estimators=100)
    scores_rf = cross_val_score(rf, X, y, scoring='accuracy', cv=cv)

    print('Score : ', np.mean(scores_rf))

rf_scores(iris_ss, iris.target)
```

```
Score :  0.96
```

zero-base /

이번에는 두 개의 특성만 적용했을 때

```
pca_X = iris_pd_pca[['pca_component_1', 'pca_component_2']]

rf_scores(pca_X, iris.target)
```

```
 Score :  0.9066666666666666
```

실습 - wine data

zero-base /

다시 와인데이터

```
wine_url = 'https://raw.githubusercontent.com/PinkWink/ML_tutorial'+\
                                    '/master/dataset/wine.csv'
wine = pd.read_csv(wine_url, sep=',', index_col=0)
wine.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	color
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	1
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	1
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	1
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	1

zero-base /

와인 색상 분류 (red/white)

```
wine_y = wine['color']
wine_X = wine.drop(['color'], axis=1)
wine_X.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

zero-base /

StandardScaler 적용

```
wine_ss = StandardScaler().fit_transform(wine_X)
wine_ss[:3]
```

```
array([[ 0.14247327,  2.18883292, -2.19283252, -0.7447781 ,  0.56995782,
        -1.10013986, -1.44635852,  1.03499282,  1.81308951,  0.19309677,
        -0.91546416, -0.93722961],
       [ 0.45103572,  3.28223494, -2.19283252, -0.59764007,  1.1979747 ,
        -0.31132009, -0.86246863,  0.70148631, -0.11507303,  0.99957862,
        -0.58006813, -0.93722961],
       [ 0.45103572,  2.55330026, -1.91755268, -0.66069923,  1.02669737,
        -0.87476278, -1.09248586,  0.76818761,  0.25811972,  0.79795816,
        -0.58006813, -0.93722961]])
```

zero-base /

두 개의 주성분으로 줄이는 건 데이터의 50%가 안되는구나

```
pca_wine, pca = get_pca_data(wine_ss, n_components=2)
pca_wine.shape
```

```
(6497, 2)
```

```
print_variance_ratio(pca)
```

```
variance_ratio:  [0.25346226 0.22082117]
sum of variance_ratio:  0.47428342743236135
```

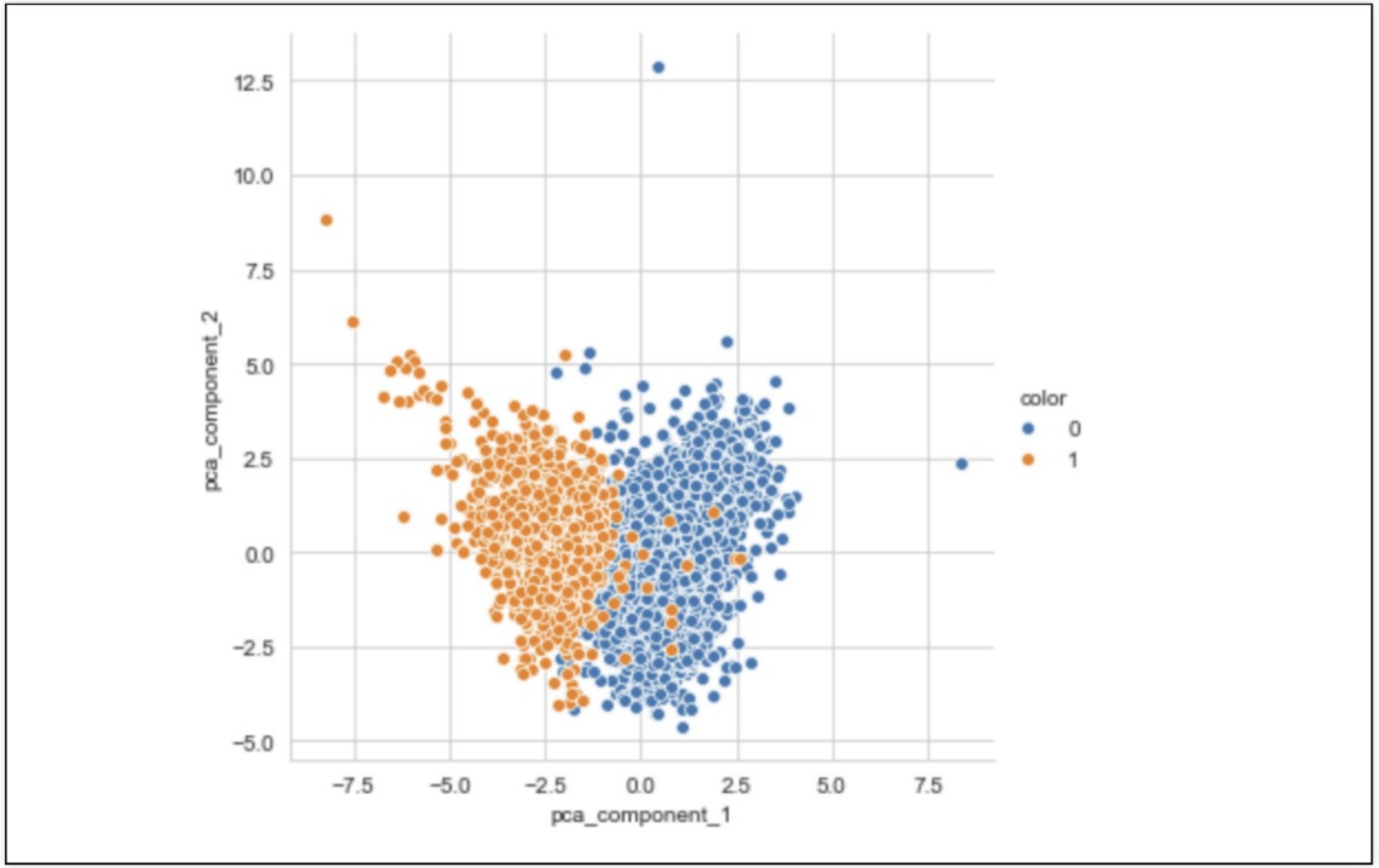
zero-base /

그래도 그려보자

```
pca_columns = ['pca_component_1','pca_component_2']
pca_wine_pd = pd.DataFrame(pca_wine, columns=pca_columns)
pca_wine_pd['color'] = wine_y.values

sns.pairplot(pca_wine_pd, hue='color', height=5,
             x_vars=['pca_component_1'], y_vars=['pca_component_2']);
```

zero-base /

괜찮은 듯 한데



zero-base /

Random Forest에 적용했을 때 원 데이터와 큰 차이가 없다

```
rf_scores(wine_ss, wine_y)
```

```
 Score :  0.9935352638124
```

```
pca_X = pca_wine_pd[['pca_component_1', 'pca_component_2']]
rf_scores(pca_X, wine_y)
```

```
 Score :  0.981067803635933
```

zero-base /

주 성분 3개로 표현해달라고 했더니, 98% 이상을 표현할 수 있다고

```
pca_wine, pca = get_pca_data(wine_ss, n_components=3)
print_variance_ratio(pca)

cols = ['pca_1', 'pca_2', 'pca_3']
pca_wine_pd = get_pd_from_pca(pca_wine, cols=cols)

pca_X = pca_wine_pd[cols]
rf_scores(pca_X, wine_y)
```

```
variance_ratio:  [0.25346226 0.22082117 0.13679223]
sum of variance_ratio:  0.6110756621838704
Score :  0.9832236631728548
```

zero-base /

주성분 3개로 표현한 것을 정리하고

```
pca_wine_plot = pca_X
pca_wine_plot['color'] = wine_y.values
pca_wine_plot.head()
```

	pca_1	pca_2	pca_3	color
0	-3.348438	0.568926	-2.727386	1
1	-3.228595	1.197335	-1.998904	1
2	-3.237468	0.952580	-1.746578	1
3	-1.672561	1.600583	2.856552	1
4	-3.348438	0.568926	-2.727386	1

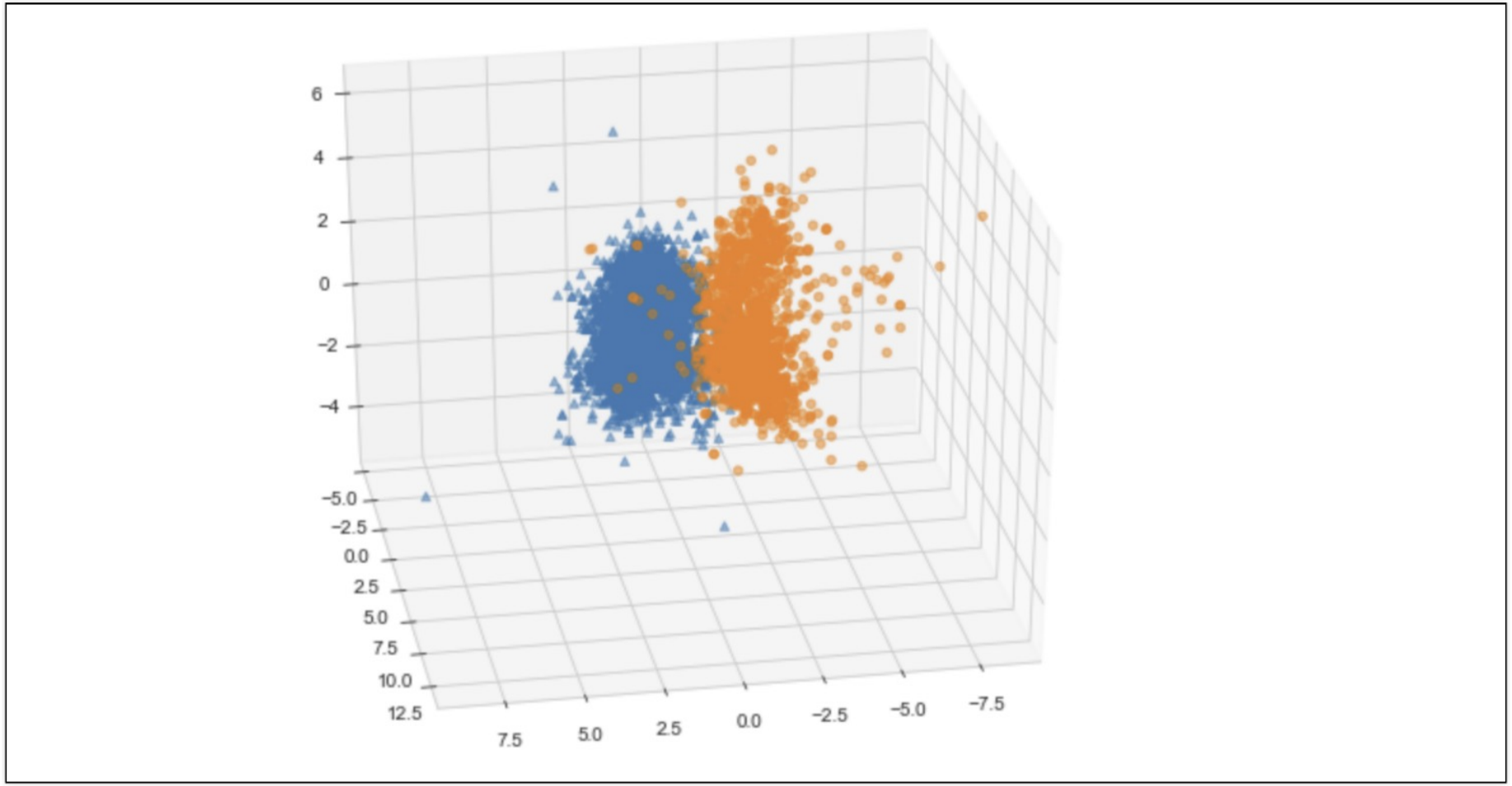
zero-base /

3D로 그려보자

```
from mpl_toolkits.mplot3d import Axes3D

markers=['^', 'o']

fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

for i, marker in enumerate(markers):
    x_axis_data = pca_wine_plot[pca_wine_plot['color']==i]['pca_1']
    y_axis_data = pca_wine_plot[pca_wine_plot['color']==i]['pca_2']
    z_axis_data = pca_wine_plot[pca_wine_plot['color']==i]['pca_3']

    ax.scatter(x_axis_data, y_axis_data, z_axis_data,
               s= 20, alpha=0.5, marker=marker)

ax.view_init(30, 80)
plt.show()
```

zero-base /

결과



zero-base /

조금 마음에 안 들면 plotly로~

```
import plotly.express as px

fig = px.scatter_3d(pca_wine_plot,
                    x='pca_1', y='pca_2', z='pca_3',
                    color='color', symbol='color',
                    opacity=0.4)
fig.update_layout(margin=dict(l=0, r=0, b=0, t=0))
fig.show()
```

zero-base /

결과