zero-base /

Life Changing Education

Chapter 26. Titanic data using PCA, kNN

Titanic

zero-base /

다시 타이타닉으로~

```
import pandas as pd

titanic_url = 'https://raw.githubusercontent.com/PinkWink/ML_tutorial'+\
                                        '/master/dataset/titanic.xls'
titanic = pd.read_excel(titanic_url)
titanic.head()
```

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	ho
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S	2	NaN	St
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NaN	Mc PQ Ch ON
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	Mc PQ Ch ON
			Allison,											Mc

zero-base /

이름 분리해서 title 만들고

```
import re

title = []
for idx, dataset in titanic.iterrows():
    title.append(re.search('\,\s\w+(\s\w+)?\.', dataset['name']).group()[2:-1])

titanic['title'] = title
titanic.head()
```

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	ho
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S	2	NaN	St
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NaN	Mo PQ Ch ON
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	Mo PQ Ch ON

zero-base /

귀족과 평민 등급을 구별하고

```
titanic['title'] = titanic['title'].replace('Mlle', 'Miss')
titanic['title'] = titanic['title'].replace('Ms', 'Miss')
titanic['title'] = titanic['title'].replace('Mme', 'Mrs')

Rare_f = ['Dona', 'Dr', 'Lady', 'the Countess']
Rare_m = ['Capt', 'Col', 'Don', 'Major', 'Rev', 'Sir', 'Jonkheer', 'Master']

for each in Rare_f:
    titanic['title'] = titanic['title'].replace(each, 'Rare_f')

for each in Rare_m:
    titanic['title'] = titanic['title'].replace(each, 'Rare_m')

titanic['title'].unique()
```

```
array(['Miss', 'Rare_m', 'Mr', 'Mrs', 'Rare_f'], dtype=object)
```

zero-base /

gender 컬럼 생성

```
from sklearn.preprocessing import LabelEncoder

le_sex = LabelEncoder()
le_sex.fit(titanic['sex'])
titanic['gender'] = le_sex.transform(titanic['sex'])

le_sex.classes_
```

```
array(['female', 'male'], dtype=object)
```

zero-base /

grade 컬럼 생성

```
le_grade = LabelEncoder()
le_grade.fit(titanic['title'])
titanic['grade'] = le_grade.transform(titanic['title'])

le_grade.classes_
```

```
array(['Miss', 'Mr', 'Mrs', 'Rare_f', 'Rare_m'], dtype=object)
```

zero-base /

현재까지 데이터 정리

```
titanic.head()
```

sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest	title	gender	grade
female	29.0000	0	0	24160	211.3375	B5	S	2	NaN	St Louis, MO	Miss	0	0
male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NaN	Montreal, PQ / Chesterville, ON	Rare_m	1	4
female	2.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON	Miss	0	0
male	30.0000	1	2	113781	151.5500	C22 C26	S	NaN	135.0	Montreal, PQ / Chesterville, ON	Mr	1	1

zero-base /

null이 아닌 데이터만

```
titanic = titanic[titanic['age'].notnull()]
titanic = titanic[titanic['fare'].notnull()]
titanic.info()
```

zero-base /

데이터 나누고

```
from sklearn.model_selection import train_test_split

X = titanic[['pclass','age','sibsp','parch','fare',
                            'gender', 'grade']].astype('float')
y = titanic['survived']

X_train, X_test, y_train, y_test = \
                    train_test_split(X, y, test_size=0.2, random_state=13)
```

zero-base /

pca 적용 준비

```
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
```

```
def get_pd_from_pca(pca_data, col_num):
    cols = ['pca_'+str(n) for n in range(col_num)]
    return pd.DataFrame(pca_data, columns=cols)
```

zero-base /

```
def get_pd_from_pca(pca_data, col_num):
    cols = ['pca_'+str(n) for n in range(col_num)]
    return pd.DataFrame(pca_data, columns=cols)
```

```
import numpy as np

def print_variance_ratio(pca, only_sum=False):
    if only_sum==False:
        print('variance_ratio: ', pca. explained_variance_ratio_)
    print('sum of variance_ratio: ', np.sum(pca.explained_variance_ratio_))
```

zero-base /

두 개의 축으로 변환

```
pca_data, pca = get_pca_data(X_train, n_components=2)
print_variance_ratio(pca)
```

```
variance_ratio:  [0.93577394 0.06326916]
sum of variance_ratio:  0.9990431009511265
```

zero-base /

그려보자

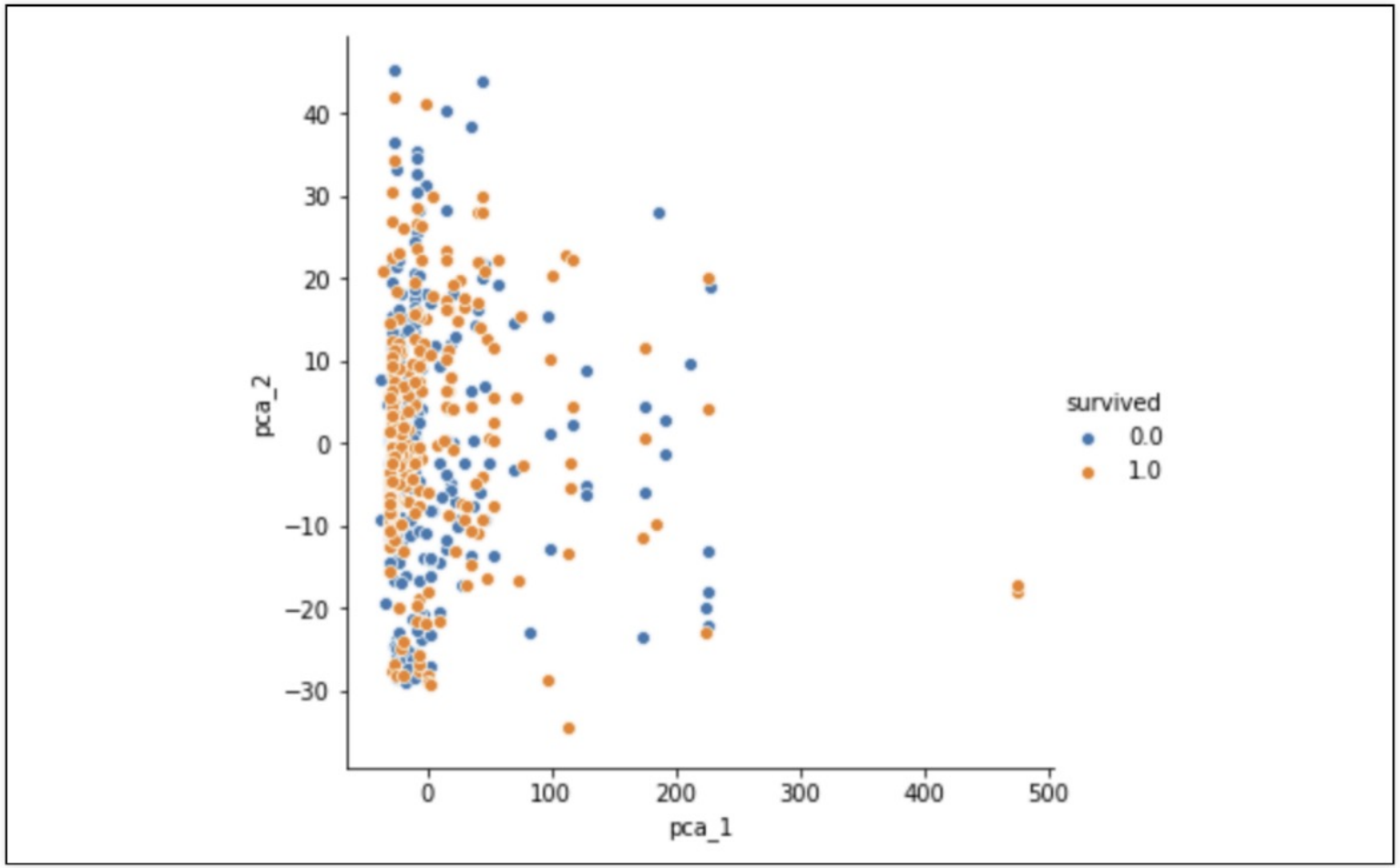
```
import seaborn as sns

pca_columns = ['pca_1','pca_2']
pca_pd = pd.DataFrame(pca_data, columns=pca_columns)
pca_pd['survived'] = y_train

sns.pairplot(pca_pd, hue='survived', height=5,
             x_vars=['pca_1'], y_vars=['pca_2']);
```

zero-base /

구분이 잘 될까



zero-base /

그럼 세 개로

```
pca_data, pca = get_pca_data(X_train, n_components=3)
print_variance_ratio(pca)
```

```
variance_ratio:  [9.35773938e-01 6.32691630e-02 4.00903990e-04]
sum of variance_ratio:  0.9994440049413542
```

zero-base /

그려서 확인해볼 준비

```
pca_pd = get_pd_from_pca(pca_data, 3)

pca_pd['survived'] = y_train.values
pca_pd.head()
```

	pca_0	pca_1	pca_2	survived
0	-28.763184	4.479379	-0.451531	0
1	41.587362	22.084594	0.011834	0
2	-19.598979	-10.999936	0.558167	0
3	-28.232483	-6.559632	-1.349217	1
4	-29.055717	-1.510811	-0.538886	0

zero-base /

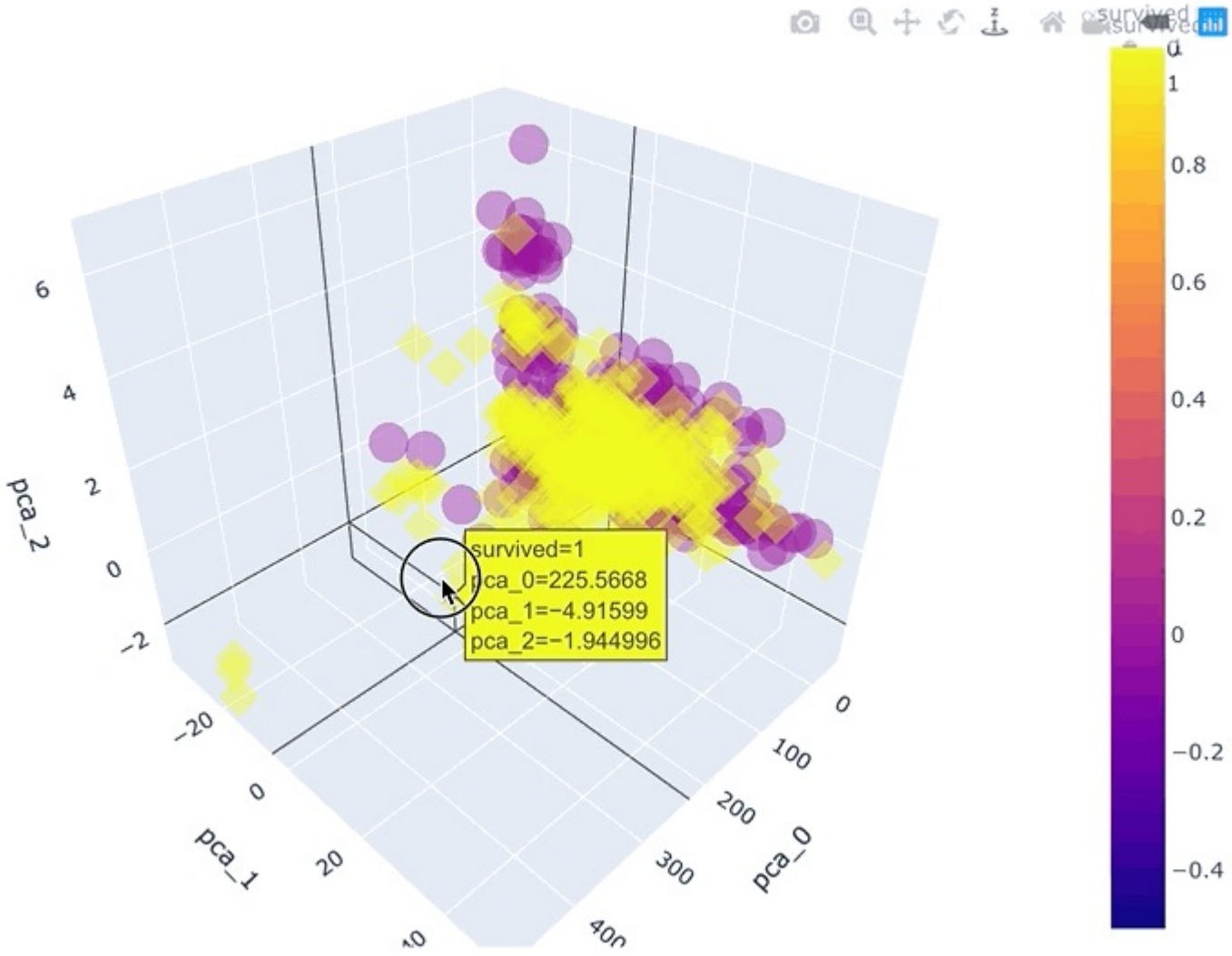
plotly.express로

```
import plotly.express as px

fig = px.scatter_3d(pca_pd,
                    x='pca_0', y='pca_1', z='pca_2',
                    color='survived', symbol='survived',
                    opacity=0.4)
fig.update_layout(margin=dict(l=0, r=0, b=0, t=0))
fig.show()
```

zero-base /

결과



zero-base /

pipe line 구축

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

estimators = [('scaler', StandardScaler()),
              ('pca', PCA(n_components=3)),
              ('clf', KNeighborsClassifier(n_neighbors=20))]

pipe = Pipeline(estimators)
pipe.fit(X_train, y_train)

pred = pipe.predict(X_test)
print (accuracy_score(y_test, pred))
```

```
0.7703349282296651
```

zero-base /

다시 확인

```
# ['pclass','age','sibsp','parch','fare','gender', 'grade']
# array(['Miss', 'Mr', 'Mrs', 'Rare_f', 'Rare_m'], dtype=object)

dicaprio = np.array([[3, 18, 0, 0, 5, 1, 1]])
print('Decaprio : ', pipe.predict_proba(dicaprio)[0,1])

winslet = np.array([[1, 16, 1, 1, 100, 0, 3]])
print('Winslet : ', pipe.predict_proba(winslet)[0,1])
```

```
Decaprio :  0.05
Winslet :  0.85
```