K-Nearest Neighbor 알고리즘을 활용한

유방암 진단 데이터 모델링

(데이터마이닝 개인과제_유혜원_2020170845)

과목명: 데이터마이닝(IMEN321)

산업경영공학부

2020170845 유혜원

Ⅰ. 서론

1. 문제 정의

데이터 분석, 인공지능 기법들이 발전함에 따라 다양한 분야에서 활용이 이루어지고 있다. 의료분야에서는 충분한 의학지식을 가지고 있는 숙련자가 진단을 내리는 기존의 방식에서 벗어나, 수많은 의료 데이터들을 활용해 컴퓨터가 데이터를 보고 진단을 내릴 수 있도록 하는 시도가 계속되고 있다. 이번 프로젝트에서는 유방암과 연관이 있는 데이터들을 이용하여 유방암 여부를 분류하는 모델을 구현해본다.

2. 데이터 설명

데이터는 scikit-learn패키지에 포함된 Breast cancer Wisconsin (diagnostic) dataset으로, 종양에 대한 데이터들로 이루어져 있다. 데이터의 Attribute는 종양의 Radius, Texture, Perimeter, Area Smoothness, Compactness, Concavity, Concave points, Symmetry, Fractal dimension의 평균, 표준편차, 최악(또는 최댓값)으로 이루어져 있다. 따라서 이 데이터는 총 30개의 독립변수와, 1(양성)과 0(음성) 값을 갖는 종속변수로 이루어져 있다. 관측치의 수는 569개이다.

데이터의 형태를 간단하게 살펴보면 다음과 같다.

```
(569, 31)
   mean radius  mean texture  mean perimeter  mean area  ...  worst concave points  worst symmetry  worst fractal dimension  target
0        17.99         10.38          122.80     1001.0  ...                0.2654          0.4601                  0.11890       0
1        20.57         17.77          132.90     1326.0  ...                0.1860          0.2750                  0.08902       0
2        19.69         21.25          130.00     1203.0  ...                0.2430          0.3613                  0.08758       0
3        11.42         20.38           77.58      386.1  ...                0.2575          0.6638                  0.17300       0
4        20.29         14.34          135.10     1297.0  ...                0.1625          0.2364                  0.07678       0

[5 rows x 31 columns]
```

Ⅱ. 본론

1. 데이터 전처리

1) 필요 패키지 불러오기

이번 분석에 사용된 패키지와 함수들은 다음과 같다.

```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn import datasets
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
```

2) 데이터 불러오기 및 데이터프레임 만들기

Scikit-learn 패키지에서 유방암 진단 데이터를 불러온다. 그리고 data값과 attribute 이름을 하나의 데이터프레임으로 구성한다. 이때, shape() 함수로 행과 열의 수를 확인하고, head() 함수로 전체 데이터의 형태를 확인한다.

```
11  load_df = datasets.load_breast_cancer() #데이터 불러오기
12
13  data = pd.DataFrame(load_df['data'])
14  feature=pd.DataFrame(load_df['feature_names'])
15  data.columns = feature[0]
16  target = pd.DataFrame(load_df['target'])
17  target.columns=['target']
18  df = pd.concat([data,target], axis=1)
19  print(df.shape)
20  print(df.head()) #데이터 형태 파악
```

3) 데이터 표준화 및 Target 할당

Scikit-learn패키지의 StandScalar()를 이용해 데이터 표준화를 진행하고, X_변수에 할당하였다. Y변수에는 target의 데이터 값을 할당하였다.

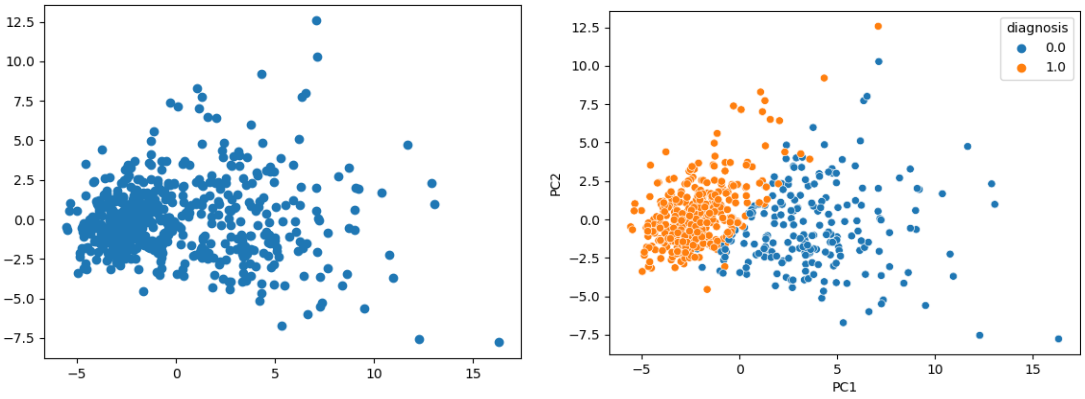
```
23  X_ = StandardScaler().fit_transform(data) #데이터 표준화
24  Y = target # y변수에 target 할당
```

2. 데이터 탐색

데이터의 분포를 확인해보기 위해 먼저 PCA 분석을 진행하고, 2개의 축을 기준으로 데이터를 그래프로 나타내었다. 또한, target 데이터의 값을 이용해, 진단 결과에 따라 구분하였다.

```
27  pca = PCA(n_components=2) #scatterplot을 그리기 위해 PCA분석을 n=2로 진행
28  pc = pca.fit_transform(X_)
29  plt.scatter(pc[:,0],pc[:,1]) #PCA 분석이 끝난 값을 좌표로 찍어보기
30  plt.show()
31
32  pc_y = np.c_[pc,Y]
33  df_ = pd.DataFrame(pc_y, columns=['PC1','PC2','diagnosis'])
34  sns.scatterplot(data=df_, x='PC1', y='PC2', hue='diagnosis')
35  plt.show() #진단 결과를 색상으로 표현한 그래프 그리기
```

위 코드를 실행시킨 결과 다음과 같은 그래프가 그려졌다.



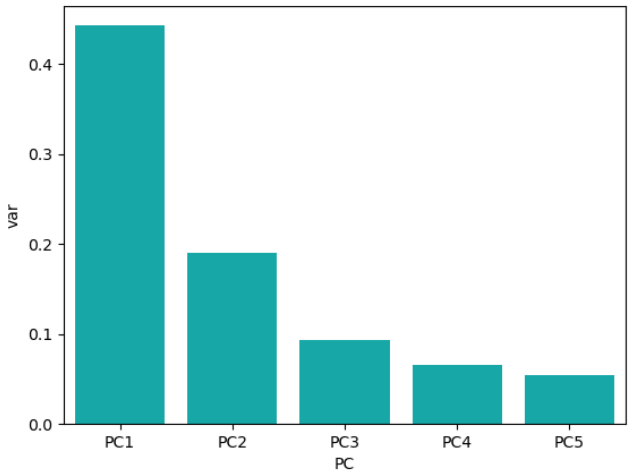
그래프를 확인해본 결과, KNN모델을 이용해 유방암 여부를 확인할 수 있을 것으로 예상해볼 수 있다.

3. 모델 구현

1) PCA분석의 적정레벨을 찾고 데이터 분할하기

PCA분석에서 각 축의 설명력을 구하고, 이를 Barplot으로 나타낸다. 70%~80%가량의 설명력을 갖는 레벨이 n=3임을 확인하고, n=3으로 설정한 뒤 PCA분석을 진행한다. 그리고 그 결과를 train_test_split()을 이용하여 랜덤하게 train set과 test set으로 분할한다. 이에 필요한 코드와 Barplot은 다음과 같다.

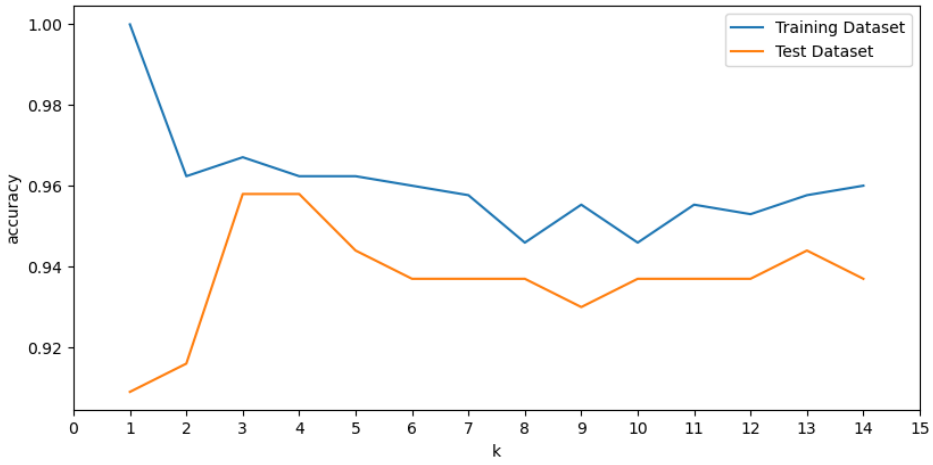
```
pca = PCA(n_components = 5)
pc = pca.fit_transform(X_)
df_var = pd.DataFrame({'var': pca.explained_variance_ratio_, 'PC': ['PC1','PC2','PC3','PC4','PC5']})
sns.barplot(x='PC', y='var', data = df_var, color = 'c')
plt.show() #5개의 축이 가지는 설명력을 barplot으로 나타냄


pca = PCA(n_components = 3)
pc = pca.fit_transform(X_)
X_train, X_test, Y_train, Y_test = train_test_split(pc,Y, stratify=Y, random_state= 30)
```



2) KNN모델링에 필요한 최적의 K값 구하기

앞서 분할한 train set과 test set을 이용하여 KNN모델링을 시작한다. 이 때 최적의 K값을 구하기 위해 K값이 1부터 14까지 변화할 때 모델의 accuracy가 어떻게 변화하는지 그래프로 그려보았다. 필요한 코드와 그래프는 다음과 같다.

```
53  #최적의 K를 찾기
54  train_acc = []
55  test_acc = []
56
57  for n in range(1,15):
58      KNN_model = KNeighborsClassifier(n_neighbors=n)
59      KNN_model.fit(X_train, Y_train)
60      train_acc.append(KNN_model.score(X_train,Y_train))
61      test_acc.append(KNN_model.score(X_test,Y_test))
62
63  plt.figure(figsize=(12,9))
64  plt.plot(range(1,15), train_acc, label = 'Training Dataset')
65  plt.plot(range(1,15), test_acc, label = 'Test Dataset')
66  plt.xlabel("k")
67  plt.ylabel("accuracy")
68  plt.xticks(np.arange(0,16, step=1))
69  plt.legend()
70
71  plt.show()
```



그래프를 확인해보면, K=3일 때 Test set의 accuracy가 높다는 것을 알 수 있다. 따라서 K=3으로 설정하고 KNN모델링을 구현한다.

3) K가 3일 때, KNN모델링 구현하기

K=3으로 설정하고 KNN모델링을 하고, score 함수를 활용해 모델의 정확도를 검증하는 코드는 다음과 같다.

```
73  # k가 3일 때 Knn modeling을 하고, score 확인하기
74  KNN_model = KNeighborsClassifier(n_neighbors=3)
75  KNN_model.fit(X_train,Y_train.values.ravel())
76  prediction = KNN_model.predict(X_test)
77
78  print(KNN_model.score(X_train,Y_train))
79  print(KNN_model.score(X_test, Y_test))
```

4. 성능 평가

앞서 수행한 성능 평가의 결과는 다음과 같았다.

```
0.9671361502347418
0.958041958041958
```

위의 값은 train set을 기준으로 정확도를 평가한 것이고, 아래의 값은 test set을 기준으로 정확도를 평가한 것이다. 각각 96.7%, 95.8%로 높은 정확도를 가지고 있음을 확인할 수 있다.

Ⅲ. 결론

PCA분석과 K-Nearest Neighbor Model을 활용하여 모델링한 결과 준수한 성능의 모델을 구성할 수 있었다. 즉, 주어진 attribute들을 이용해 유방암을 진단할 수 있음을 확인하였다. 이와 같이 의료 데이터를 이용해 질병을 진단함으로써 진단의 정 확도를 향상시키거나, 진단의 속도를 향상시킬 수 있다. 또한, 단순히 질병의 유무만을 진단하는 것이 아니라, 데이터 값에 따라 질병일 확률을 계산하고 확신할 수 없는 데이터는 숙련된 의사의 진단과 함께 고려된다면 질병의 진단을 더욱 정확하고 빠르게 할 수 있을 것으로 보인다.