**Machine Learning & Deep Learning**

**기말고사 대체 과제**

2015310553 Youseop Lee

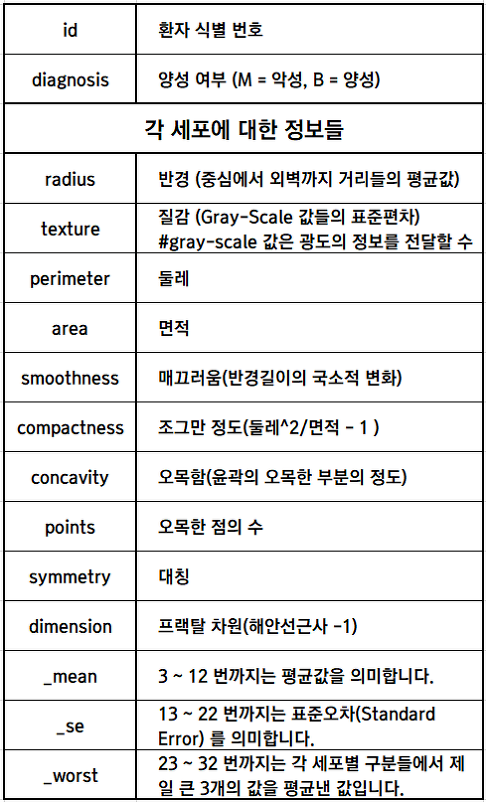
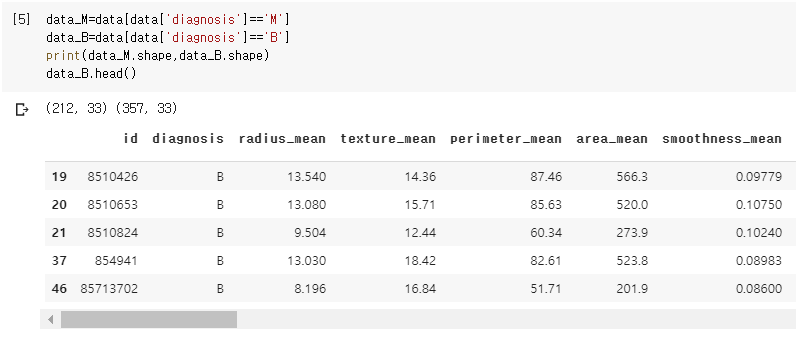
1. 해결하고자 하는 문제에 대한 설명

어머니 가슴에 유방암으로 진행될 가능성이 있는 종양이 여러 개 존재해, 몇 년 전 그중 일부를 제거하는 수술을 받으셨다. 악성으로 진행될 가능성이 높은 종양들을 제거했지만, 한편으로는 악성으로 번질 가능성이 있는 종양들이 아직 남아있기도 하다.

Kaggle에서 Wisconsin 대학교의 Breast Cancer Data를 토대로 머신러닝 알고리즘들을 활용해 종양이 악성으로 번질 가능성이 있는지 아닌지 예측하는 시스템을 만들고자 한다.

인간의 판단에 의존하지 않는 정확한 시스템을 만든다면 악성일 경우 제때 필요한 조치를 취할 수 있을 것이고, 양성이라면 환자의 심리상태를 안정시키는데 기여할 수 있을 것이다.

1. 해결하고자 하는 문제와 관련된 데이터에 대한 설명

Breast Cancer Wisconsin Data Set[[1]](#footnote-1) & Column explanation[[2]](#footnote-2)

Diagnosis가 B인 행과 M인 행으로 각각 분류한다. 위에서 볼 수 있듯이 M과 B의 데이터 개수가 불균형을 이뤄 두 데이터 모두 210개로 개수를 맞춰주었다.

Diagnosis’는 추후 Labeling을 위해 사용할 Column이고, 규칙성 없는 환자 ‘id’와 값이 들어있지 않은 ‘unnamed: 32’를 제거했다.

1. 문제를 해결하기 위해 적용하고자 하는 방법에 대한 설명 (왜 이 방법을 이용하여 주어진 문제를 해결할 수 있는가)

이 데이터를 사용해 종양을 구별해 내는데, SVM (Support Vector Machine)을 적용했다.

우리의 목적은 종양 별로 30개의 데이터를 이용해 이 종양이 양성인지 악성인지 가려내는 것이다. 두개의 카테고리로 이루어진 데이터를 지도 학습시키는 특징을 가지기에 효율적으로 구분해낼 수 있을 것이라 생각했다.

추가적으로 ANN(Artificial Neural Network)도 사용해 결과를 비교해볼 계획이다.

0개의 column으로 이루어진 데이터로, 이를 모두 활용하면 오히려 효율이 떨어질 수 있음을 감안해 양성과 악성을 보다 잘 구별할 수 있는 인자를 골라내서 사용했다. 각각 column의 P-value를 구해 값이 작은 하위 12개의 항목을 구했다.

12는 임의로 정한 값이며, [4]에서 이 값을 변수로 사용해 SVM과 ANN의 performance가 어떻게 변하는지에 대해 실험을 해 보았다.

Rank = len(P\_value\_Rank)

NoOfData=210

FeatureSelected\_B = np.zeros((Rank,NoOfData))

FeatureSelected\_M = np.zeros((Rank,NoOfData))

count=0

for i in P\_value\_Rank[0]:

  i=int(i)

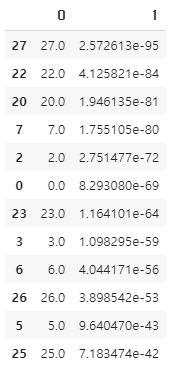
  FeatureSelected\_B[count,:]=data\_B\_sample.iloc[i,:].values

  FeatureSelected\_M[count,:]=data\_M\_sample.iloc[i,:].values

  count+=1

FeatureSelected=pd.DataFrame(np.concatenate([FeatureSelected\_B,FeatureSelected\_M],axis=1))

print(FeatureSelected.shape)

1. 위의 방법을 데이터에 적용하여 얻은 결과

NoOfData    = int(data\_M\_sample.shape[1])

NoOfFeature = int(data\_M\_sample.shape[0])

P\_value = np.zeros((NoOfFeature , 2))

for i in np.arange(NoOfFeature):

    T\_test       = np.array(sp.ttest\_ind

(data\_B\_sample.iloc[i,:] , data\_M\_sample.iloc[i,:]))

    P\_value[i,0] = i

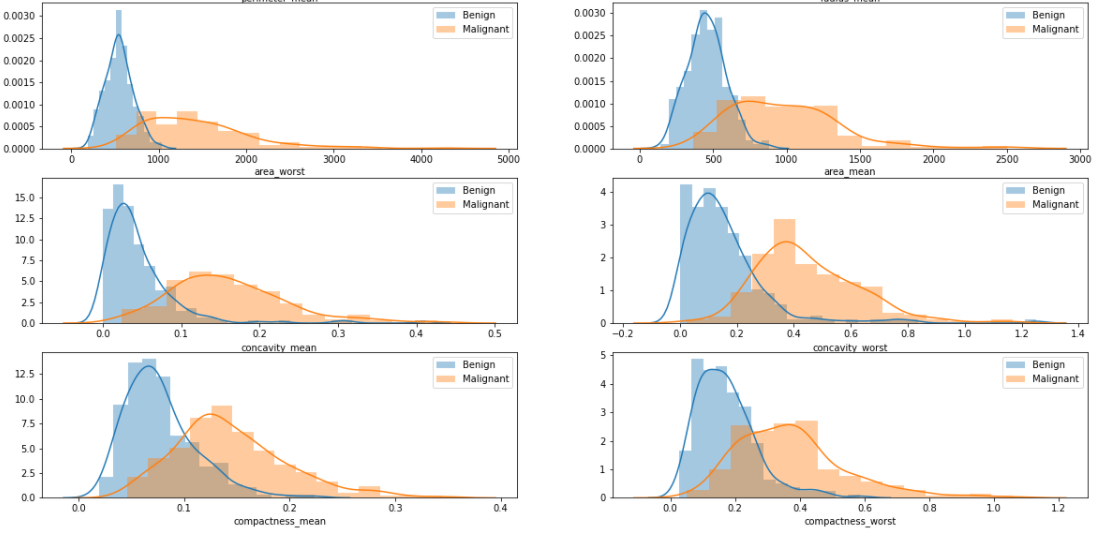
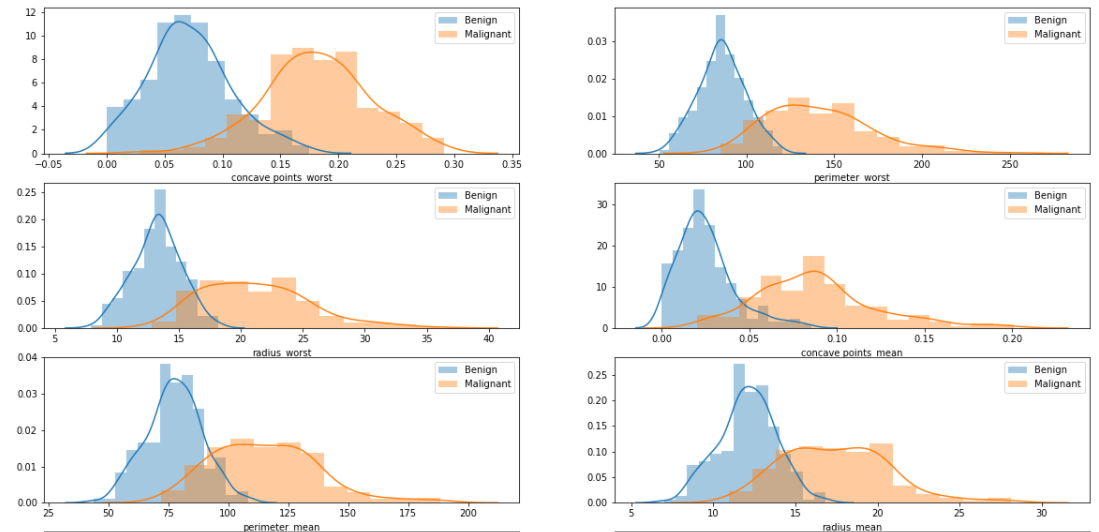
    P\_value[i,1] = T\_test[1]

P\_value      = pd.DataFrame(P\_value)

P\_value\_Rank = P\_value.sort\_values([1],ascending=True)

Ranking=12

P\_value\_Rank=P\_value\_Rank[ :Ranking ]



[P-value 하위 12개 그래프(B와 M을 가장 잘 구분해주는 요소 12개)]

data-M와 data-B에서 P-value로 뽑힌 항목들에 대한 data 재배열후 FeatureSelected생성 (0-209열 – B, 210-419열 - M)

FeatureSelected Data를 토대로 교차검증을 위해 K-Fold를 수행했다. 이번 실험에서는 5개의 Fold로 데이터를 나누었다.

[P-value : 12 values]

이 값들을 SVM모델에 넣어 Fold별로 예측 정확도를 계산해 보았다.

Add    = 0

Divide = 0

for i in range(Fold):

    c1 = 'Validation\_CurrentFold = Validation\_Fold%d'%(i+1)

    exec(c1)

    Predicted = np.array(SVM\_model\_Final.predict(Validation\_CurrentFold))

    c2 = 'svmscore\_Fold%d = metrics.accuracy\_score(ValidationFold\_Label , Predicted)'%(i+1)

    exec(c2)

    Add += metrics.accuracy\_score(ValidationFold\_Label , Predicted)

    Divide += 1

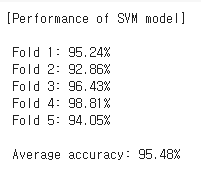
Avg\_accuracy = Add/Divide

print('[Performance of SVM model] \n')

print(' Fold 1: {:.2f}% \n Fold 2: {:.2f}% \n Fold 3: {:.2f}% \n Fold 4: {:.2f}% \n Fold 5: {:.2f}%'.

        format(svmscore\_Fold1\*100, svmscore\_Fold2\*100, svmscore\_Fold3\*100, svmscore\_Fold4\*100, svmscore\_Fold5\*100))

print('\n Average accuracy: {:.2f}%'.format(Avg\_accuracy\*100))



동일 데이터를 ANN을 이용해 학습시켜보았다.

learningRate=0.001, noOfNeuron=20, iteration=10000

for i in range(Fold):

    s1= 'Data = Training\_Fold%d'%(i+1)

    s2= 'Data\_Val = Validation\_Fold%d'%(i+1)

    exec(s1)

    exec(s2)

    s3 = 'Validation\_Acc\_Fold%d = sess.run(accuracy, feed\_dict={X : Data\_Val, Y: Label\_Val})'%(i+1)

    exec(s3)

Accuracy\_sum = 0

for j in range(Fold):

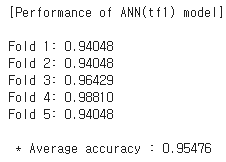
    s3 = 'Accuracy\_sum += Validation\_Acc\_Fold%d'%(j+1)

    exec(s3)

print('[Performance of ANN(tf1) model] \n')

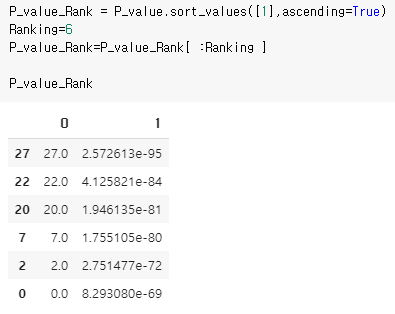
print('Fold 1: {:.5f} \nFold 2: {:.5f} \nFold 3: {:.5f} \nFold 4: {:.5f} \nFold 5: {:.5f} \n\n \* Average accuracy : {:.5f}'

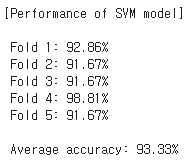
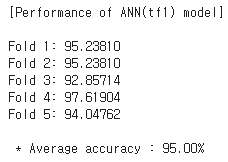
      .format(Validation\_Acc\_Fold1, Validation\_Acc\_Fold2, Validation\_Acc\_Fold3, Validation\_Acc\_Fold4, Validation\_Acc\_Fold5, Accuracy\_sum/Fold))



[P-value : 6 values]

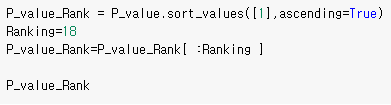
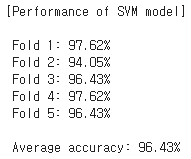
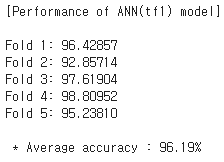
동일조건에서 특징 값 하위 6개를 추출한 후 다시 학습을 진행시켜 보았다.





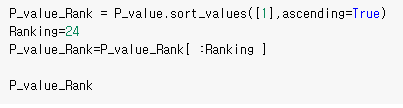
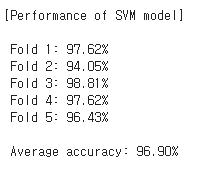
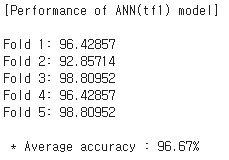
[P-value : 18 values]

동일조건에서 특징 값 하위 18개를 추출한 후 다시 학습을 진행시켜 보았다.

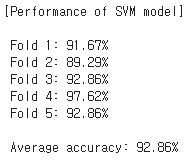
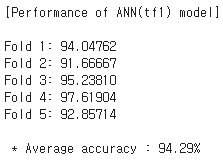


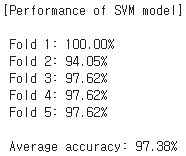
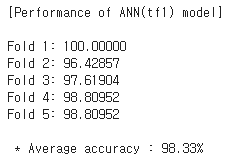
[P-value : 24 values]

동일조건에서 특징 값 하위 24개를 추출한 후 다시 학습을 진행시켜 보았다.



추가적으로 P-value를 4개, 30개 뽑은 경우에 대해서도 실험을 진행했다.

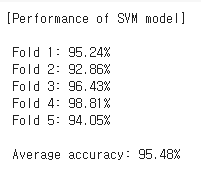
[P-value : 4 values]

[P-value : 30 values]

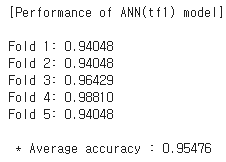
1. 결과에 대한 해석

Breast Cancer의 30개 특징 값들에 대해 P-value를 구한 후에 양성과 악성을 가장 잘 구별해 줄 수 있는 값 12개를 뽑아 SVM, ANN 분석을 진행해 보았다.

분석 결과 SVM모델은 각각의 Fold에 대해 95.24%, 92.86%, 96.43%, 98.81%, 94.05%의 정확도를 보였으며 평균 95.48%의 정확도로 악성과 양성 종양을 구별했다.



ANN모델은 learningRate=0.001, noOfNeuron=20, iteration=10000 로 설정 후 데이터 학습을 진행했다. 각각의 Fold에 대해서 94.05%, 94.05%, 96.43%, 98.81%, 94.05%로 평균 95.48%의 정확도로 악성과 양성 종양을 구별했다.



ANN과 SVM모델의 각각의 Fold에 대한 정확도는 달랐지만 평균 정확도는 소수 두번째 자리까지 거의 비슷한 성능을 보여주었다.

추가적으로 위의 실험에서는 P-value 하위 12개를 선별하여 데이터 학습을 시켰는데 변수의 종류를 변화시켰을 때 SVM, ANN모델의 정확도가 어떻게 변할지 알아보았다.

먼저 변수의 개수를 P-value기준 하위 6, 12, 18, 24개(4, 14, 30 추가)로 뽑아 각 모델의 테스트 정확도를 알아보았다.

아래의 표는 변수의 개수에 따른 SVM, ANN모델의 정확도를 나타낸 것이다.

|  |  |  |
| --- | --- | --- |
| #p-value | SVM(%) | ANN(%) |
| 4 | 92.86 | 94.29 |
| 6 | 93.33 | 95 |
| 12 | 95.48 | 95.48 |
| 14 | 97.14 | 97.38 |
| 18 | 96.43 | 96.19 |
| 24 | 96.9 | 96.67 |
| 30 | 97.38 | 98.33 |

아래의 그래프는 P-value의 개수(가로축)에 따른 SVM, ANN의 Performance%(세로축)를 나타낸 그래프이다.

SVM과 ANN 모두 변수의 개수가 늘어남에 따라 정확도가 높아지는 경향성을 보였다. 추가적으로 변수의 개수가 상대적으로 적을 경우 이번 데이터 에서는 SVM보다 ANN이 더 좋은 성능을 보였음을 확인할 수 있다. ANN의 iteration, learning rate, #ofnuron값에 따라 정확도가 변하며, 이 결과를 모든 데이터에 적용시키기엔 한계가 분명히 있지만,

‘’<https://github.com/youseop> >> github에서 파일을 확인하실 수 있습니다.’’

**Reference**

빅데이터와 인공지능을 이용한 시스템 강건설계 실습자료

위스콘신 대학교 Breast Cancer 데이터 from kaggle

<https://www.kaggle.com/uciml/breast-cancer-wisconsin-data>

1. [<https://www.kaggle.com/uciml/breast-cancer-wisconsin-data>] [↑](#footnote-ref-1)
2. [<https://gomguard.tistory.com/52>] [↑](#footnote-ref-2)