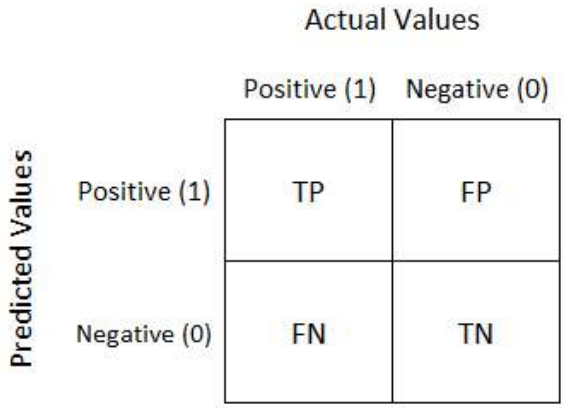
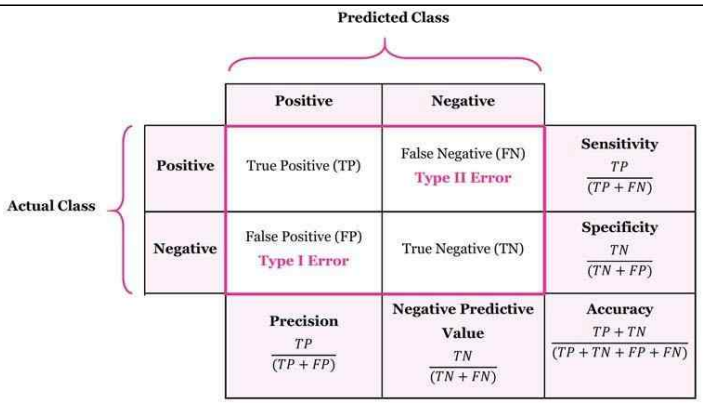
세종대학교

팀번호	팀번호 입력

2024-1학기 창의학기제 주간학습보고서 (6주차)

창의과제	세종대학교 집현캠퍼스를 개선시킨 웹서비스 개발				
이름	신찬영	학습기간	5월 10일 ~ 5월 12일		
학번	23012094	학습주차	6	학습시간	3
학과(전공)	인공지능	과목명	자기주도창의전공1	수강학점	3
※ 수강학점에 따른 회차별 학습시간 및 10주차 이상 학습 준수					
금주 학습목표	인공지능 모델을 평가하는데 많이 사용하는 Confusion matrix에 대해 알아보고 우리가 만든 인공지능 모델에 적용시켜본다.				
학습내용	 confusion matrix는 기본적으로 위의 형태를 띄고 각 영역마다 TP, FP, FN, TN으로 나뉜다. 위에서 아래까지의 합은 실제 값의 양성값(TP+FN)과 음성값(FP+TN)을 의미하고 왼쪽부터 오른쪽 까지의 합은 모델이 예측한 양성값(TP+FP)과 음성값(FN+TN)을 의미한다. TP:모델이 양성이라 예측한 것 중에서 실제로 맞은 개수 FP:모델이 양성이라 예측한 것 중에서 틀린 개수 FN:모델이 음성이라 예측한 것 중에서 틀린 개수 TN:모델이 음성이라 예측한 것 중에서 실제로 맞은 개수				



(이사진은 위의 사진과 다르게 Actual class(실제값)과 predicted class(예측값)의 위치가 다르다.)
TP, FP, TN, FN을 이용하여 Sensitivity(민감도), Specificity(특이도), Precision, Negative Predictive Value, Accuracy(정확도)를 구할 수 있다.

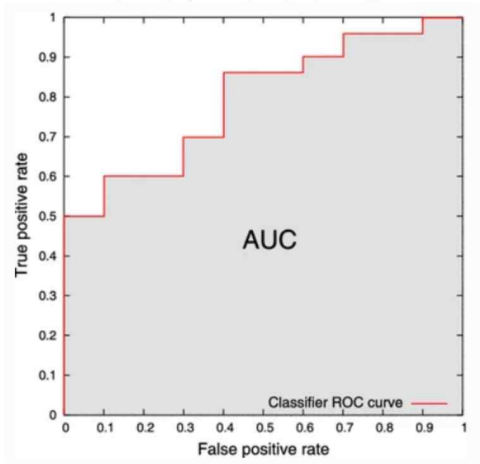
Sensitivity(민감도):모델이 양성이라 예측한 것 중에서 실제로 양성인 것의 비율
Specificity(특이도):모델이 음성이라 예측한 것 중에서 실제로 음성인 것의 비율
Precision:실제로 양성인 것중에서 모델이 맞힌 비율(양성 성능)
Negative Predictive Value:실제로 음성인 것 중에서 모델이 맞힌 비율(음성 성능)
Accuracy(정확도):전체 데이터에서 모델이 맞힌 비율

또한 위의 사진은 보면 type 1 error, type 2 error라는 것이 있다. type 1 error는 FP을 의미 하고 type 2 error는 FN을 의미한다. 이렇게 FP, FN으로 나누는 이유는 분류 문제에 따라 type 1 error, type 2 error 두 개 중 하나가 다른 하나보다 치명적이기 때문이다.

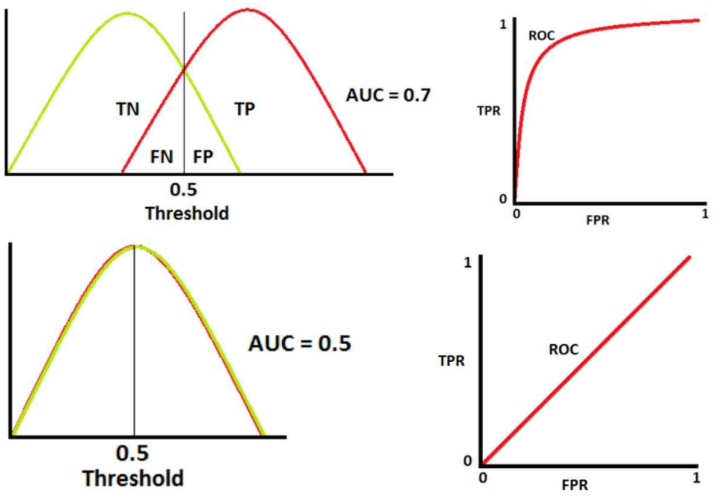
예를 들어 암환자 분류 문제에서는 암환자(양성)을 정상인(음성)으로 판단하는 것은 매우 치명적이기 때문에 type 2 error를 사용한다.

TP, FP, TN, FN을 이용해 Auraccy이외의 새로운 평가 기준을 만들 수 있다. 그것은 바로 f1-score이다. f1-score= 2*Precision*Sensitivity/(Precision+Sensitivity) 이렇게 구할 수 있다.

이 f1-score는 데이터가 불균형할 때 사용된다. 만약에 TP=0 FP=0 TN=20 FN=980 이라면 민감도=0이지만 Accuracy는 98%가 나온다. 즉 모델은 음성이라고만 말할 뿐인 이상한 모델인데 데이터의 불균형으로 인해 정확도가 98%나 되어서 좋은 모델처럼 보인다. 이 모델의 f1 score를 구하면 0으로 나온다. 즉 모델이 이상한 것을 알 수 있다.



위의 Sensitivity(민감도)와 Specificity(특이도)를 이용하여 ROC 커브라는 것을 만들 수 있따. ROC 커버는 임계값에 따른 모델의 분류 성능에 대한 측정 그래프이다. 위 그래프의 x축은 FPR로 1-Specificity(특이도)을 의미하고 y축은 TPR 민감도를 의미한다. 또한 아래 면적이 크면 클수록 모델의 성능이 좋은 것이다.

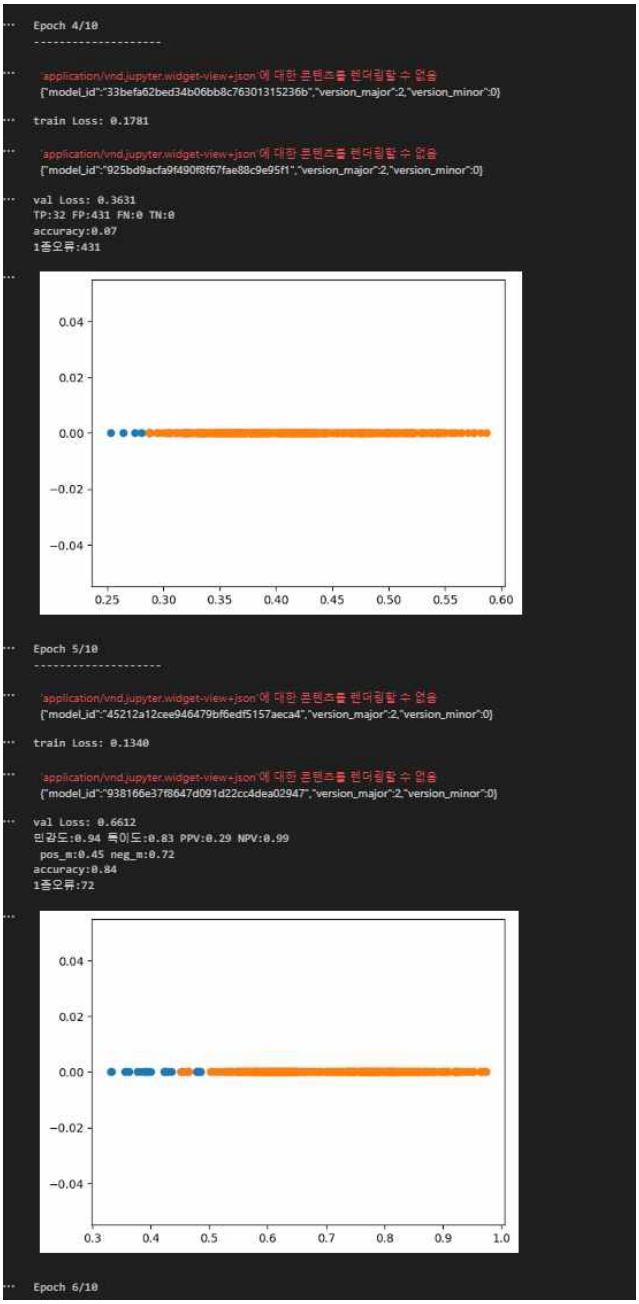


(선을 기준으로 왼쪽은 음성값 오른쪽은 양성값으로 예측한걸로 취급한다.)

이렇게 모델이 분류한 정도에 따라 ROC그래프가 달라지는 것을 볼 수 있다.

```python
def train_model(model, dataloader_dict, criterion, optimizer, num_epoch):
    since = time.time()

    for epoch in range(num_epoch):
        print('Epoch {}/{}'.format(epoch + 1, num_epoch))
        print('-'*20)
        TP=0
        TN=0
        FP=0
        FN=0
        for phase in ['train', 'val']:
            if phase == 'train':
                model.train()
            else:
                model.eval()

            epoch_loss = 0.0

            for inputs1,inputs2, labels in tqdm(dataloader_dict[phase]):
                inputs1 = inputs1.to(device)
                inputs2=inputs2.to(device)
                labels=labels.to(device)
                optimizer.zero_grad()


                with torch.set_grad_enabled(phase == 'train'):#학습 시에만 연산 기록을 추적
                    outputs1,outputs2 = model(inputs1,inputs2)

                    loss = criterion(outputs1,outputs2, labels)

                    epoch_loss += loss.item() * inputs1.size(0)

                for i in range(32):
                    if phase=='val':
                        try:
                            concatenated=torch.stack((inputs1[i].to('cpu'),inputs2[i].to('cpu')),dim=0)
                        except:
                            break
                        distance=F.pairwise_distance(outputs1[i],outputs2[i])
                        if(torch.tensor([1.], device='cuda:0')==labels[i]):
                            if(distance>0.5):
                                FN+=1
                            else:
                                TP+=1
                        else:
                            if(distance>0.75):
                                TN+=1
                            else:
                                FP+=1

                for i in range(16):
                    if phase=='val':
                        if epoch==num_epoch-1:
                            try:
                                concatenated=torch.stack((inputs1[i].to('cpu'),inputs2[i].to('cpu')),dim=0)
                            except:
                                break
                            distance=F.pairwise_distance(outputs1[i],outputs2[i])
                            img = torchvision.utils.make_grid(concatenated).permute(1, 2, 0) # CHW to HWC
                            plt.figure(figsize=(10, 5)) # 이미지 크기 설정
                            plt.imshow(img)
                            plt.text(10, -10, 'Dissimilarity: {:.2f}'.format(distance.item()), color='red')
                if phase == 'train':
                    loss.backward()
                    optimizer.step()
            epoch_loss = epoch_loss / len(dataloader_dict[phase].dataset)

            print('{} Loss: {:.4f}'.format(phase, epoch_loss))
            if(phase=='val'):
                try:
                    print("민감도:%.2f 특이도:%.2f PPV:%.2f NPV:%.2f"%(TP/(TP+FN),TN/(FP+TN),TP/(TP+FP),TN/(TN+FN)))
                except:
                    print("TP:%d FP:%d FN:%d TN:%d"%(TP,FP,FN,TN))

                print("accuracy:%.2f"%((TP+TN)/(TP+TN+FP+FN)))
                print("1종오류:%d"%(FP))
    time_elapsed = time.time() - since
    print('Training complete in {:.0f}m {:.0f}s'.format(
        time_elapsed // 60, time_elapsed % 60))
    return model
```

위의 사진의 코드는 모델을 학습하고 검증 데이터셋을 이용해 각 에포크마다 평가하는 코드

	이다. 모델을 평가하는 부분에서 모델이 예측한 distance(내얼굴과 others의 얼굴이 얼마나 유사한지에 대한 값 값이 낮을수록 유사하다)가 내가 설정한 값보다 작다면 모델이 양성으로 판단한 것으로 설정하고 아니라면 음성으로 판단한 것이라고 설정하여 그에 맞게 TP, FP, FN, TN을 정해 주었다.  마진 값에 의해 TP,FP등의 값이 변해서 더 효율적으로 모델을 평가하기 위해 그래프를 추가하였다.(파란색이 양성 값 주황색이 음성값이다.) 점의 x값은 모델이 예측한 distance이다 주황색점과 파란색점이 명확하게 나눠질수록 좋은 모델이다. (자세한 설명과 하이퍼 파라미터를 조정 과정은 이지민 학생의 주간 학습 보고서 7주차 또는 9주차에 작성될 예정이다.)
학습방법	기존 인공지능 수학 수강과목의 강의자료를 다시보고 더 깊은 이해를 위해 인터넷을 사용하였다.

세종대학교

학습성과 및 목표달성도	TP, FP, FN, TN에 대하여 확실히 이해하였고 실사용에 이 개념을 적용할 수 있게 되었다. 기존의 Accuracy는 양성 데이터가 많지 않아 정확하지 않았지만 이 것을 이용하여 모델을 정확하게 평가하여 더 정교하게 하이퍼파라미터를 조절할 수 있게 되었다.
참고자료 및 문헌	https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 https://encord.com/glossary/confusion-matrix/ 세종대 수강과목 인공지능 수학 1-심태용 교수님 3~4주차 수강과목 https://bioinformaticsandme.tistory.com/328
내주 계획	이 모델을 웹에 적용해야하니 html과 웹에대해 공부한다.

2024 년 5 월 7일

지도교수 (인)