Diabetes 예측 프로그램

1. 구현과정
   a. 데이터 전처리

      utils.py의 preprocess 함수를 import하여 결측값을 평균값으로 대체, 스케일 차이를 해결하기 위해 표준화를 진행한다.

   b. 신경망 설계

      입력층 – 64개 뉴런, ReLU 활성화 함수 사용(ReLU – 비선형 관계 학습가능, 기울기 소실 문제 방지)

      은닉층 – 32개 뉴런, ReLU 활성화 함수 사용

      출력층 – 1개 뉴런, Sigmoid 활성화 함수 사용(0과 1사이로 출력값 제한 => 확률로 평가 가능)

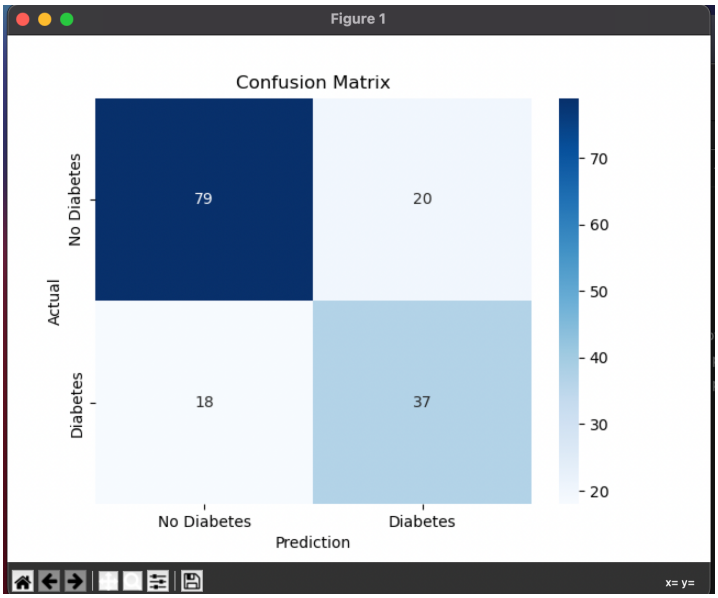
   c. 모델 학습

      epochs=100(100번 반복) 학습, 훈련 데이터의 20%를 검증 데이터로 활용

   d. 모델 평가

      train_accuracy, test_accuracy, Confusion Matrix, ROC curve로 모델 평가

2. Confusion Matrix(혼동 행렬)



TP – 실제 값이 Diabetes, 예측 값이 Diabetes

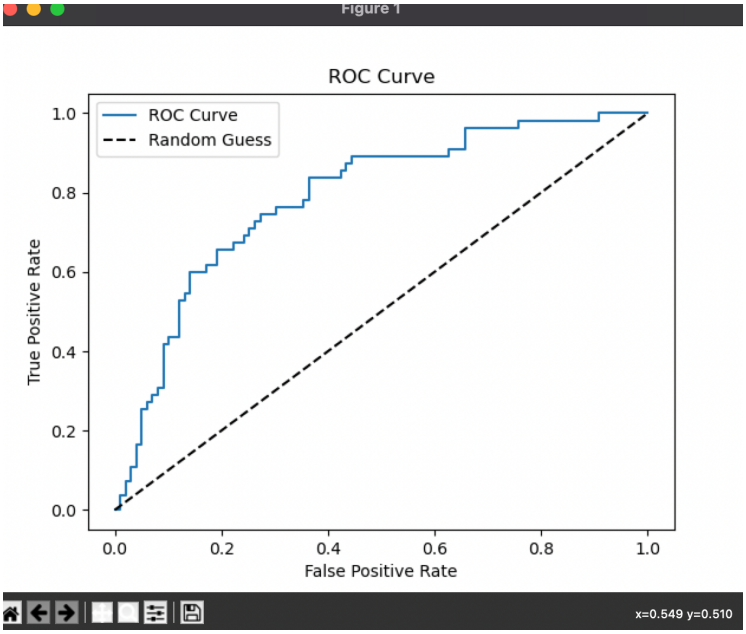
TN – 실제 값이 No Diabetes, 예측 값이 No Diabetes

FP – 실제 값이 No Diabetes, 예측 값이 Diabetes

FN – 실제 값이 Diabetes, 예측 값이 No Diabetes

때문에 TP, TN의 비율이 높아야 하고 FP, FN의 비율이 낮을 수록 좋은 모델이다. 위의 Figure 1을 보면, TP의 값은 37이고, TN의 값은 79이다. 또한 FP, FN의 값은 상대적으로 낮기 때문에 좋은 성능을 보여주고 있다고 평가할 수 있다.

3. ROC 곡선(Receiver Operating Characteristic Curve)



x축 – FPR 즉, 잘못 예측한 비율
y축 – TPR 즉, 올바르게 예측한 비율

따라서 FPR은 낮고, TPR은 높을 때 모델 성능이 더 우수하다. 때문에, 곡선이 왼쪽 위로 더욱 빠르게 상승하여 유지될 수록 모델 성능이 우수하다. 따라서 위 figure1의 성능 지표를 보면, Random Guess보다 ROC curve가 더욱 좋은 예측율을 보여주고 있음을 알 수 있다.

4. 발전 방향

더 많은 데이터 셋(Kaggle 등)을 확보하여 학습하고, Residual Networks, LSTM등 더 복잡한 모델을 활용하여 구현한다. 또한 Leaky ReLU 또는 ELU 등의 활성화 함수를 사용하여 성능을 개선할 수 있다.