딥러닝 모델의 교차검증 (Cross Validation)

2018년 11월 24일 토요일 오전 12:51

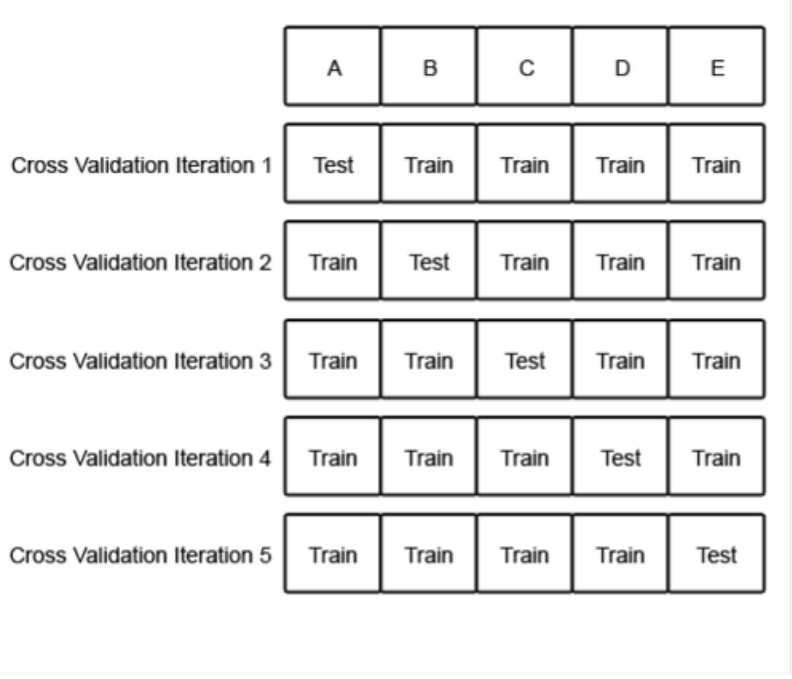
교차검증이란?

Keras로 딥러닝 모델을 만들고 이를 교차 검증 하는 방법을 포스팅하겠습니다. 우선 **교차 검증(Cross validation)** 이 무엇인지에 대해 설명이 필요할 것 같은데요. 교차 검증이란 통계학에서 모델을 "평가" 하는 방법입니다.

모델을 평가하기 위해 기본적인 방법은 바로 트레이닝셋과 테스트셋을 분리하여 트레이닝셋으로 모델의 계수를 추정하고 테스트셋으로 성능을 평가하는 것인데요.

교차검증은 이러한 기본적인 작업의 문제점을 보완하기 위해서 쓰입니다. 이 문제점이란 바로 **데이터셋의 크기가 작은 경우 테스트셋에 대한 성능 평가의 신뢰성이 떨어지게 된다**는 것입니다. 테스트셋을 어떻게 잡느냐에 따라 성능이 아주 상이하게 나온다면, 우연에 의한 효과로 인해 모델 평가 지표에 편향이 생기게 되겠죠.

이를 해결하기 위해 교차 검증은 **모든 데이터가 최소 한 번은 테스트셋으로 쓰이도록** 합니다. 아래의 그림을 보면, 데이터를 5개로 쪼개 매번 테스트셋을 바꿔나가는 것을 볼 수 있습니다. 첫 번째 Iteration에서는 BCDE를 트레이닝 셋으로, A를 테스트셋으로 설정한 후, 성능을 평가합니다. 두 번째 Iteration에서는 ACDE를 트셋으로, B를 테스트셋으로하여 성능을 평가합니다. 그러면 총 5개의 성능 평가지표가 생기게 되는데, 보통 이 값들을 평균을 내어 모델의 성능을 평가하게 됩니다. (아래 데이터는 모두 사실은 트레이닝 데이터입니다. Iteration이라는 상황안에서만 테스트셋이 되는 것입니다.)



(cf. 여기서 validation set은 없습니다. 성능 평가를 할 때, **validation set을 사용하냐 아니면 cross validation을 사용하냐**, 둘 중 하나를 선택하는 걸로 이해하시면 됩니다. 데이터의 수가 적을 때는 시간은 오래걸리지만 cross validation을 하는 것이 일반적입니다.)

교차검증을 통해 달성할 수 있는 성능 평가의 목적

모델 성능 평가는 보통 2개의 목적이 있습니다. 1. Unseen 데이터에 대한 성능을 예측하기 위해, 2. 더 좋은 모델을 선택하기 위해 (혹은 Hyperparameter Tuning) 입니다. 교차 검증은 1,2를 달성하기 위한 좋은 방법입니다.

파이썬에서 CV를 하는 방법은 여러가지가 있지만 scikit-learn이 많이 사용됩니다. 본 포스팅에서는 keras에서 모델을 만들고 이를 교차검증하여 성능을 평가하는 방법을 알아보겠습니다.

코드

```
# MLP for Pima Indians Dataset with 10-fold cross validation via sklearn
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
import numpy

# Function to create model, required for KerasClassifier
def create_model():
	# create model
	model = Sequential()
	model.add(Dense(4, input_dim=8, activation='relu'))
	model.add(Dense(4, activation='relu'))
	model.add(Dense(1, activation='sigmoid'))
	# Compile model
	model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
	return model

# fix random seed for reproducibility
seed = 7
numpy.random.seed(seed)

# load pima indians dataset
dataset = numpy.loadtxt("pima-indians-diabetes.data.csv", delimiter=",")

# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]

# create model
model = KerasClassifier(build_fn=create_model, epochs=150, batch_size=10, verbose=0)

kfold = KFold(n_splits=2, shuffle=True, random_state=seed)
results = cross_val_score(model, X, Y, cv=kfold)
```

설명

- create_model()을 통해 Keras Model 객체를 얻음
- KerasClassifier Wrapper를 통해 Keras Model을 Scikit learn에서 사용할 수 있는 객체로 변환을 한다.
- Scikit learn의 kFold를 통해 KFold의 Rule을 지정한다.
- cross_val_score을 통해 해당 모델의 cross validation score를 계산한다.

결과

array([0.65104168, 0.59114585])

0.621093765677 (평균)

위 예제 코드에서는 간단하게 보이기 위해 2 Fold cross validation score를 계산하였습니다. results에 다음과 같은 두 개의 성능 지표가 계산되며 이 때 지표는model을 만들 때 입력한 accuracy를 지표로합니다. accuracy는 예측한 것중에 맞은 것의 비율입니다.

출처: https://3months.tistory.com/321 [Deep Play]