**<1>**

코드를 살펴보면, 사이킷런의 fetch\_california\_housing() 함수를 사용해 캘리포니아 주택 데이터셋을 적재하였다.

그 후에 훈련 세트, 검증 세트, 테스트 세트로 나누고 모든 특성의 스케일을 변경하였다.

MLP구조를 이용하였고, 활성화 함수로 ReLU, 비용함수로 MSE, Optimizer는 SGD를 사용하였다.

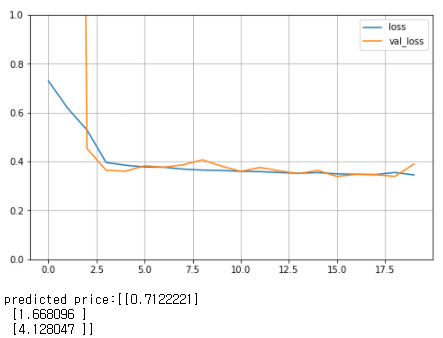
MLP는 여러층의 퍼셉트론으로 적어도 1개 이상의 은닉층을 보유한다.

ReLU는 기울기가 갑자기 변해서 경사 하강법이 엉뚱한 곳으로 튈 수 있다. 하지만 보통 잘 작동되고 계산 속도가 빠른 장점이 있어 기본 활성화 함수로 사용된다.

SGD는 keras에서 주로 사용하는 옵티마이저이다.

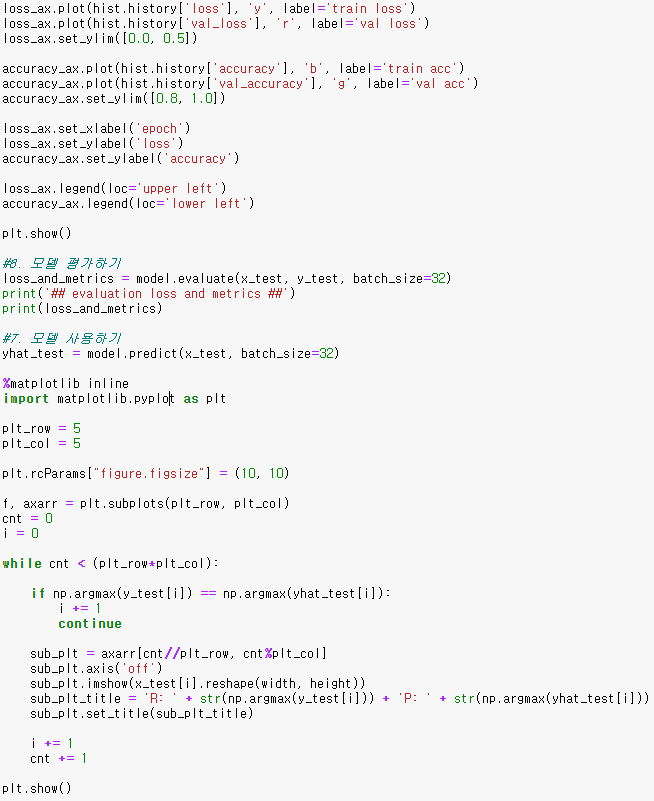
이후에 epoch = 20로 모델을 학습을 시키고, 모델에 대한 평가를 MSE로 진행한다.

훈련의 비용함수와 테스트의 비용함수를 epoch에 대하여 그래프로 나타내고, 최종적으로 예측된 주택 가격을 출력한다.



출력된 그래프를 보면, 훈련의 비용함수와 테스트의 비용함수가 비슷하다는 것을 확인하였다.

**<2>**

****

CNN은 3차원을 처리할 수 있기 때문에 정보 추출, 문장 분류, 얼굴 인식 등에 널리 사용되고, 특히 이미지 및 비디오 처리에 유용하다.

입력 데이터의 특징을 추출하여 특징들의 패턴 파악하는 구조이다.

코드를 살펴보면, 사이킷런에서 MNIST 데이터를 읽어들이고 처리하였다.

훈련 세트와 테스트 세트로 나누고, 훈련 세트와 검증 세트로 분리하고, 데이터셋을 전처리하였다.

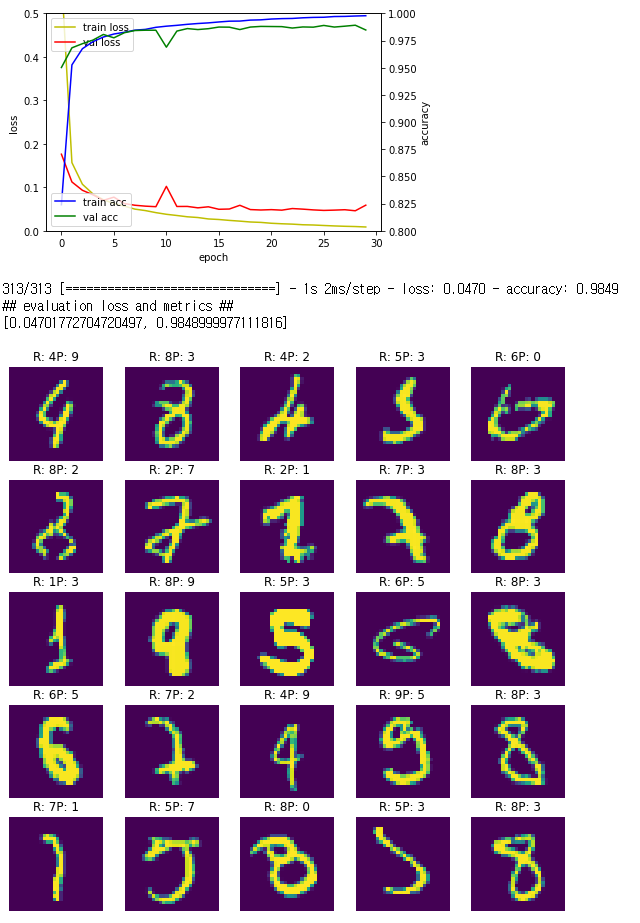
불러온 60000개의 훈련 세트를 훈련 세트 50000개와 검증 세트 10000개로 분리하였다.

Sequential, Conv2D, MaxPooling2D(활성화 함수: ReLU), Conv2D, MaxPooling2D(활성화 함수: ReLU), Flatten, Dense(활성화 함수: ReLU), Dense(활성화 함수: softmax)로 CNN 모델을 구성하였다.

이후 모델을 학습하는 과정에서 비용함수로 categorical\_corssentropy, optimizer로 SGD를 사용하였다.

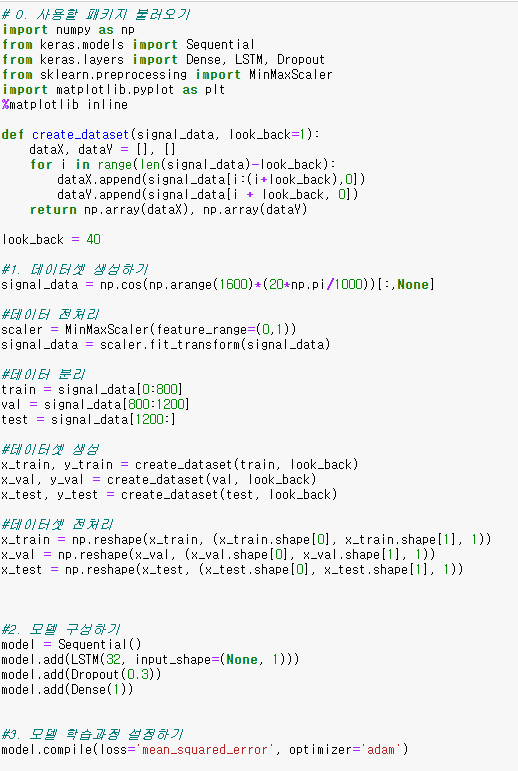
epoch=30으로 모델을 학습시켰고, 모델 평가는 loss\_and\_metrics로 진행하였다.

훈련에서의 비용함수와 테스트에서의 비용함수를 epoch에 대하여 그래프로 나타낸다.

마지막으로 숫자의 인식률을 표시하고, 오류 인식된 숫자를 그림으로 표시했다.

train loss, val loss와 train acc, val acc 그래프가 비슷한 형태임을 확인하였다.

**<3>**





RNN은 입력 데이터에 있는 순차 정보를 캡처한다.

여러 단계에서 매개 변수를 공유하므로 훈련 매개 변수가 감소하고 계산 비용도 감소한다.

반복적이고 순차적인 데이터에 효과적이다.

코드를 살펴보면, 데이터셋을 생성하고 전처리한 후에 데이터 세트를 훈련, 검증, 테스트 세트로 분리하였다. 이들에 대한 코사인 신호의 시계열을 생성하고, 데이터를 전처리한다.

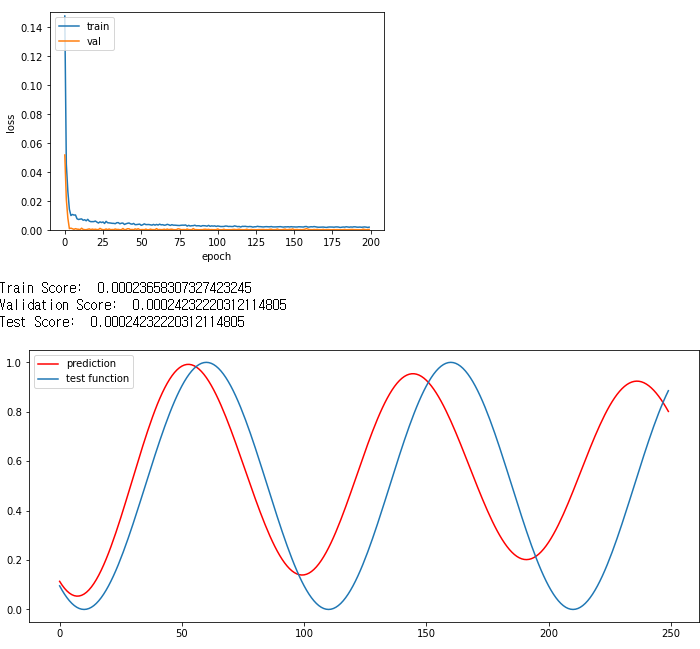
RNN의 단점인 장기기억이 불가능하다는 것과 그레디언트 소실, 폭주 문제를 해결한 LSTM을 모델 구성에 사용하였다.

모델 학습 과정을 설정하면서 비용함수는 MSE를 사용하고, optimizer는 고속 optimizer인 adam을 사용하였다..

학습 준비가 되면 epoch=200으로 학습시키고, 모델 평가를 진행한다.

훈련에서 비용 함수와 테스트에서 비용함수를 epoch에 대하여 그래프로 나타낸다.

테스트 시계열과 예측된 시계열을 비교하여 그래프로 그린다.



훈련 세트와 검증 세트의 그래프가 비슷한 형태이다. Train, Validation, Test score를 비교해봄으로서 검증하였다.

cosine label에 대한 LSTM의 예측 그래프를 확인할 수 있다. cosine 함수의 형태와 거의 비슷하므로 LSTM의 예측성능이 양호했음을 알 수 있다.