Lab-04-1 Multivariable Linear regression

Simple Linear Regression : $H(x) = Wx + b$

입력변수가 3개라면 $H(x) = W_1x_1 + W_2x_2 + W_3x_3 + b$

x의 길이가 훨씬 길다면 matmul()로 한 번에 계산한다.

ex) hypothesis = X_train.matmul(W) + b

Cost function은 기본 Simple Linear Regression과 같음.

☆ nn.Module을 상속해 모델 생성

nn.Linear(3, 1) 입력차원 3, 출력차원 1

hypothesis 계산은 forward()에서 함.

gradient 계산은 backward()로 함.

Hypothesis = model(X_train)

☆ F.mse_loss

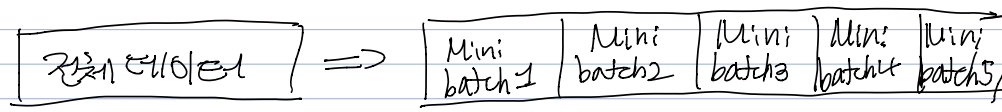
torch.nn.functional에서 제공하는 loss function을 사용하면 쉽게 다른 loss와 교체가 가능하다.

import torch.nn.functional as F

Cost = F.mse_loss(prediction, y_train)

Lab-04-2 Loading Data

엄청난 양의 데이터를 한 번에 학습시키면 너무 느리고 하드웨어 적으로 불가능하기 때문에 일부분의 데이터로만 학습함.



업데이트를 좀 더 빨리 할 수 있지만 전체 데이터를 쓰지 않아서 잘못된 방향으로 업데이트할 수 있음.

✱ from torch.utils.data import Dataset
- `__len__()` : 데이터 셋의 총 데이터 수
- `__getitem()` : 어떠한 인덱스 idx를 받았을 때, 그에 상응하는 입출력 데이터 반환

✱ from torch.utils.data import DataLoader
- batch_size = : 몇 개 mini batch의 크기, 통상적으로 2의 제곱수로 설정
- shuffle=True : Epoch 마다 데이터셋을 섞어서 데이터가 학습되는 순서를 바꾼다.
- enumerate(dataloader) : mini batch 인덱스와 데이터를 받음.