학번 / 이름	2171262 박소희
전공	IT융합공학부 사물인터넷트랙
주제	homework #2 보고서

1) 소스코드

```
import torch
import pandas as pd                          #csv 파일을 읽어오기 위해 import
from torch.utils.data import Dataset         #데이터셋 클래스
from torch.utils.data import DataLoader      #데이터 로더 클래스

class WineDataset(Dataset):
    #생성자에서 텐서의 모양 초기화
    def __init__(self, csv_path):
        self.csv = pd.read_csv(csv_path)
        self.csv = torch.FloatTensor(self.csv.values)
        #csv 파일의 값만 뽑아와서 텐서형으로 할당한다.
        self.t1, self.t2 = torch.split(self.csv, 11, dim=1)
        #dim=1 을 기준으로 1~11 열 data 와 12 열 data 를 분리
        self.t2 = self.t2.squeeze() #2 차원 -> 1 차원으로 차원 줄이기
        #print(self.t1.shape) #1599x11
        #print(self.t2.shape) #1599

    #객체 인덱스 접근 위함
    def __getitem__(self, idx):
        return self.t1[idx], self.t2[idx]

    #객체의 크기 반환
    def __len__(self):
        return len(self.t1) #1599

dataset = WineDataset('winequality-red-rev.csv')
dataloader = DataLoader(dataset, batch_size=8, shuffle=True)
#배치 크기 8, 한번에 8 개 데이터를 섞어서 가져온다
x_batch, y_batch = next(iter(dataloader))
#8 개 data 를 각각 x_batch y_batch 텐서에 쏙쏙 할당
print(x_batch.shape) #8,11
print(y_batch.shape) #8
```

2) 소스코드 설명

- **Pandas 라이브러리 설치**

  redwine엑셀 파일을 불러오기 위한 pandas 라이브러리를 설치하기 위해

  $ pip install pandas 명령어를 현재 내가 사용할 conda 가상환경 방에 입력하였다. 파이썬 파일에는 import pandas as pd로써 pd를 pandas 라이브러리로써 사용하겠다고 선언하였다.

- **dataset 객체 생성 및 __init__ 생성자 호출**

  dataset = WineDataset('winequality-red-rev.csv')를 통해 객체 dataset을 생성 및 초기화 하였다. 객체 생성시 자동으로 __init__ 생성자 함수가 호출되고 생성자 함수에서는,

  1. 엑셀파일의 경로를 파라미터로 받아서 read_csv()를 통해 csv파일의 값들을 읽고 값들을 self.csv변수에 저장한다.
  2. 우리가 원하는 것은 csv파일의 첫 행을 제외한 1599x12의 data들이기 때문에 torch.FloatTensor(self.csv.values)로써 값만 뽑아와서 텐서형으로 반환해준다.
  3. x_batch와 y_batch의 모양을 print했을 때 각각 8 x 11, 8의 사이즈가 도출되었다. 이것은 와인의 특징을 나타내는 11열까지의 값들과 와인의 품질을 결정하는 12열의 값들을 분리하여 x,y에 각각 할당하겠다는 것을 의미한다. 따라서 self.csv 텐서를 torch.split(self.csv, 11, dim=1)로써 1~11열까지와 12열의 값들을 분리하여 각각 t1, t2 텐서에 저장한다. t1사이즈 : 1599x11 / t2사이즈: 1599x1
  4. 3까지 진행했을 때는 t1텐서와 t2텐서 모두 2차원의 텐서 크기를 갖는다. 하지만 우리가 원하는 y_batch의 사이즈는 1차원이다. 따라서 본래 1599x1 사이즈였던 t2텐서를 squeeze()를 통해 1차원으로 축소한다.

- **DataLoader를 통해 8개 데이터 추출 (배치사이즈 8)**

  dataloader = DataLoader(dataset, batch_size=8, shuffle=True)를 통해 dataset 객체의 data들을 섞어서 8개의 텐서 data를 추출한다. 그 반환값을 dataloader 객체에 할당한다. 이때 dataset의 크기는 WineDataSet에 정의된 길이를 반환해주는 메소드 __len__을 통해 DataLoader가 판별 가능하다. Dataloader를 print해보면 dataloader 객체의 경로와 함께 참조값(주소)가 출력된다.

- **루프를 돌면서 x_batch, y_batch에 값 할당**

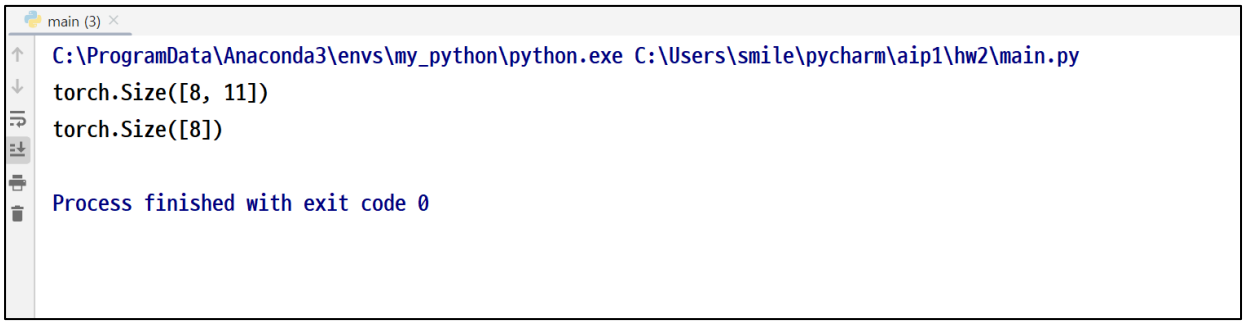
  x_batch, y_batch = next(iter(dataloader))의 의미는 반복문을 돌면서 iteration 속성을 가진 dataloader 객체의 다음 값이 있다면 x, y배치에 각각 2개의 텐서를 할당하겠다는 뜻이다. 여기서 batch 사이즈가 8이기 때문에 총 8번 반복하며 WineDataSet에 정의된 __getitem__ 메소드를 통해 각 텐서의 index에 접근하여 값 추출이 가능하다. __getitem__메소드는 t1, t2의 텐서를 각각 인덱싱해서 반환해주도록 구현하면 된다.

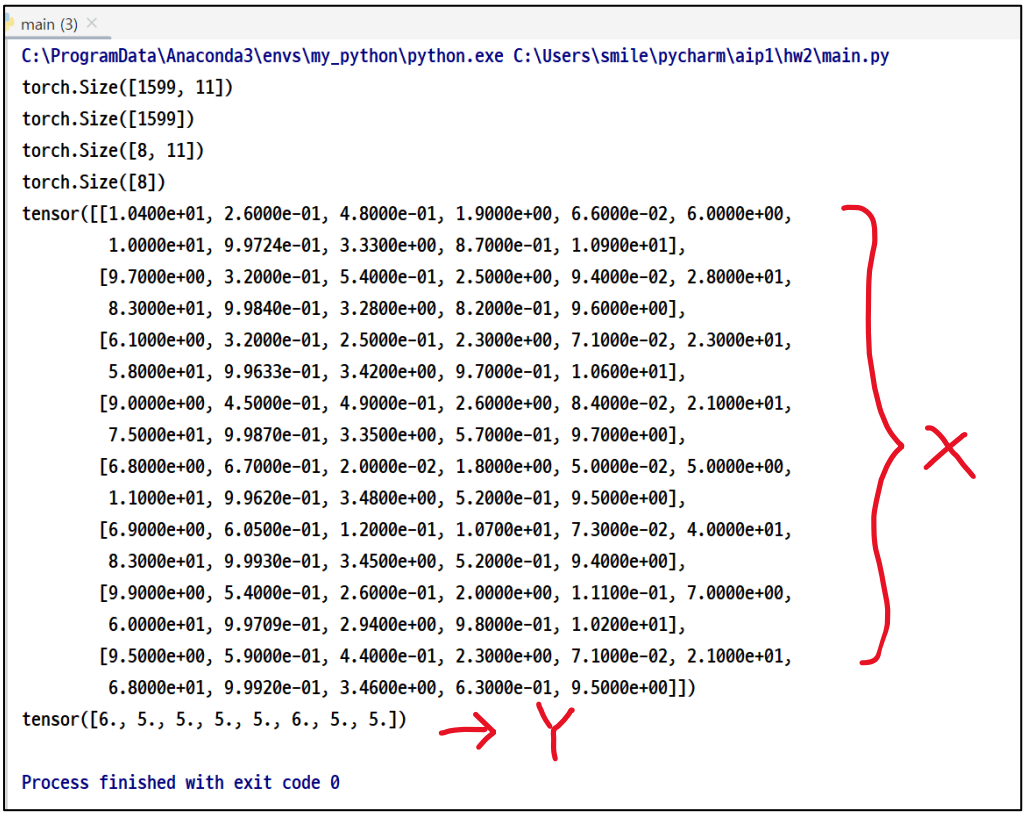
- **x_batch, y_batch shape 출력**

  이제 x_batch와 y_batch의 모양을 출력해보면 우리가 기대한 8x11, 8의 크기가 출력이 된다. 그 이유는 Dataloader를 통해 dataset의 값을 섞어서 8개의 data들을 x, y batch에 각각 반환해주도록 구현했기 때문이다. (batch_size = 8)

  x_batch, y_batch 값들을 print해보면 실행할 때마다 다른 값들이 출력된다.

3) 결과 캡처





Pandas 라이브러리를 사용하여 csv값을 읽고 텐서값들을 dataLoader 클래스를 이용하여 랜덤하게 배치 사이즈만큼 데이터를 추출해보는 실습을 해보았다.

DataSet과 DataLoader의 사용법을 익힐 수 있는 의미 있는 시간이었다.