딥러닝 모델 구현해 보기

학습 내용

- 타이타닉 데이터 셋을 활용한 딥러닝 모델 구현해 보기

- 첫번째 데이터 셋 : 자전거 공유 업체 시간대별 데이터
- **두번째 데이터 셋 : 타이타닉 데이터 셋(PyTorch)**

목차

01. 사전 환경 설치

목차로 이동하기

CPU 버전 PyTorch 설치

```
# 가상환경 만들기
conda create --name cpuDL python=3.10
conda activate cpuDL

# Jupyter Notebook
conda install -c conda-forge notebook jupyter

# 추가 설치
pip install pandas scikit-learn

# PyTorch 설치하기
# 01 Anaconda 사용 시
conda install pytorch torchvision torchaudio cpuonly -c pytorch

# 02 pip 사용 시
pip install torch torchvision torchaudio
```

GPU 버전 PyTorch 설치

```
# 가상환경 만들기
conda create --name gpuDL python=3.10

# Jupyter Notebook
```

```
conda install -c conda-forge notebook jupyter

# 추가 설치
pip install pandas scikit-learn

# PyTorch 설치하기
# 01 Anaconda 사용 시
conda install pytorch=2.1 torchvision torchaudio pytorch-
cuda=12.1 -c pytorch -c nvidia

# 02 pip 사용 시
pip3 install torch==2.1.0 torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
```

In [1]:
```
import torch

# PyTorch 버전 확인
print(torch.__version__)

# CUDA 사용 가능 여부 확인
print(torch.cuda.is_available())

# 사용 가능한 GPU 장치 수 확인
print(torch.cuda.device_count())
```

```
A module that was compiled using NumPy 1.x cannot be run in
NumPy 2.0.1 as it may crash. To support both 1.x and 2.x
versions of NumPy, modules must be compiled with NumPy 2.0.
Some module may need to rebuild instead e.g. with 'pybind11>=2.12'.

If you are a user of the module, the easiest solution will be to
downgrade to 'numpy<2' or try to upgrade the affected module.
We expect that some modules will need time to support NumPy 2.

Traceback (most recent call last):  File "<frozen runpy>", line 198, in _run_modu
le_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel_launc
her.py", line 18, in <module>
    app.launch_new_instance()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\traitlets\confi
g\application.py", line 1075, in launch_instance
    app.start()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\kerne
lapp.py", line 739, in start
    self.io_loop.start()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\tornado\platfor
m\asyncio.py", line 205, in start
    self.asyncio_loop.run_forever()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\asyncio\base_events.py", line
608, in run_forever
    self._run_once()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\asyncio\base_events.py", line
1936, in _run_once
    handle._run()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\asyncio\events.py", line 84,
in _run
    self._context.run(self._callback, *self._args)
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\kerne
lbase.py", line 545, in dispatch_queue
    await self.process_one()
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\kerne
lbase.py", line 534, in process_one
    await dispatch(*args)
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\kerne
lbase.py", line 437, in dispatch_shell
    await result
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\ipker
nel.py", line 362, in execute_request
    await super().execute_request(stream, ident, parent)
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\kerne
lbase.py", line 778, in execute_request
    reply_content = await reply_content
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\ipker
nel.py", line 449, in do_execute
    res = shell.run_cell(
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\ipykernel\zmqsh
ell.py", line 549, in run_cell
    return super().run_cell(*args, **kwargs)
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\in
teractiveshell.py", line 3075, in run_cell
    result = self._run_cell(
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\in
teractiveshell.py", line 3130, in _run_cell
    result = runner(coro)
```

```
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\async_helpers.py", line 128, in _pseudo_sync_runner
    coro.send(None)
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\interactiveshell.py", line 3334, in run_cell_async
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\interactiveshell.py", line 3517, in run_ast_nodes
    if await self.run_code(code, result, async_=asy):
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\IPython\core\interactiveshell.py", line 3577, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "C:\Users\daniel_wj\AppData\Local\Temp\ipykernel_30396\1229476210.py", line 1, in <module>
    import torch
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\__init__.py", line 1382, in <module>
    from .functional import *  # noqa: F403
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\functional.py", line 7, in <module>
    import torch.nn.functional as F
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\nn\__init__.py", line 1, in <module>
    from .modules import *  # noqa: F403
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\nn\modules\__init__.py", line 35, in <module>
    from .transformer import TransformerEncoder, TransformerDecoder, \
  File "C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\nn\modules\transformer.py", line 20, in <module>
    device: torch.device = torch.device(torch._C._get_default_device()),  # torch.device('cpu'),
C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\torch\nn\modules\transformer.py:20: UserWarning: Failed to initialize NumPy: _ARRAY_API not found (Triggered internally at C:\cb\pytorch_1000000000000\work\torch\csrc\utils\tensor_numpy.cpp:84.)
  device: torch.device = torch.device(torch._C._get_default_device()),  # torch.device('cpu'),
2.1.2
True
1
```

PyTorch 버전	권장 CUDA 버전
1.13.x	11.7
2.0.x	11.8
2.1.x	12.1

In [2]:
```
import torch

print(torch.__version__)
```

```
2.5.1
```

02. 라이브러리 및 데이터 불러오기

목차로 이동하기

In [2]:
```
import numpy as np
import pandas as pd
import torch
import torch.nn as nn
import torch.optim as optim
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
```

In [3]:
```
# 시드 고정
torch.manual_seed(42)
np.random.seed(42)

# 1. 데이터 준비
# 데이터 로드
data = pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/m

# 필요한 피처 선택
features = ['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare']
target = 'Survived'

# 성별 인코딩
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})

# 결측값 처리
imputer = SimpleImputer(strategy='median')
X = imputer.fit_transform(data[features])
y = data[target].values
```

In [4]:
```
# 스케일링
scaler = StandardScaler()
X = scaler.fit_transform(X)

# 데이터 분할
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# NumPy to PyTorch Tensor 변환
# X_train이라는 NumPy 배열을 PyTorch의 FloatTensor로 변환.
# FloatTensor는 32비트 부동 소수점 숫자로 구성된 텐서를 생성.
# 이 변환은 PyTorch 모델에서 데이터를 처리할 수 있도록 준비하는 단계
X_train = torch.FloatTensor(X_train)
X_test = torch.FloatTensor(X_test)
y_train = torch.FloatTensor(y_train).unsqueeze(1)
y_test = torch.FloatTensor(y_test).unsqueeze(1)
```

03. 신경망 모델 정의

목차로 이동하기

- 딥러닝의 이해를 위해 일부 특징(변수)만 지정하였음.
- 이미지를 사용할 때는 지정된 이미지 전체를 입력 데이터로 사용하는 경우가 대부분.

In [5]:
```
# 2. 신경망 모델 정의
model = nn.Sequential(
    nn.Linear(6, 16),
    nn.ReLU(),
    nn.Dropout(0.3),
    nn.Linear(16, 8),
    nn.ReLU(),
    nn.Dropout(0.3),
    nn.Linear(8, 1),
    nn.Sigmoid()
)

# 3. 모델 학습
# 손실 함수와 옵티마이저 정의
criterion = nn.BCELoss()  # 이진 분류 손실 함수
optimizer = optim.Adam(model.parameters(), lr=0.001)
```

In [6]:
```
# 학습 진행
epochs = 100
for epoch in range(epochs):
    # 순전파
    outputs = model(X_train)
    loss = criterion(outputs, y_train)

    # 역전파
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    # 20번마다 손실 출력
    if (epoch + 1) % 20 == 0:
        print(f'Epoch [{epoch+1}/{epochs}], Loss: {loss.item():.4f}')
```

```
Epoch [20/100], Loss: 0.7141
Epoch [40/100], Loss: 0.7019
Epoch [60/100], Loss: 0.6749
Epoch [80/100], Loss: 0.6433
Epoch [100/100], Loss: 0.5905
```

In [7]:
```
# 4. 모델 평가
model.eval()  # 평가 모드
with torch.no_grad():
    test_outputs = model(X_test)
    predicted = (test_outputs > 0.5).float()
    accuracy = (predicted == y_test).float().mean()
    print(f'Test Accuracy: {accuracy.item():.4f}')
```

```
Test Accuracy: 0.7877
```

04. 새로운 데이터로 예측

목차로 이동하기

In [8]:
```
# 예측을 위한 새로운 데이터
new_passengers = pd.DataFrame([
    [1, 'female', 29, 0, 0, 100],  # 1등급 여성
    [3, 'male', 45, 1, 0, 8],      # 3등급 남성
    [2, 'female', 8, 3, 1, 21]     # 2등급 여자아이
```

```
], columns=['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare'])

# 전처리
new_passengers['Sex'] = new_passengers['Sex'].map({'male': 0, 'female': 1})
new_scaled = scaler.transform(new_passengers)

# 예측
model.eval()
with torch.no_grad():
    predictions = model(torch.FloatTensor(new_scaled))

# 결과 출력
for i, (_, passenger) in enumerate(new_passengers.iterrows()):
    print(f"\n승객 {i+1}")
    print(f"정보: {passenger['Pclass']}등급, {'남성' if passenger['Sex'] == 0 el
    print(f"생존확률: {predictions[i].item():.2%}")
```

```
승객 1
정보: 1등급, 여성, 29세
생존확률: 52.06%

승객 2
정보: 3등급, 남성, 45세
생존확률: 40.67%

승객 3
정보: 2등급, 여성, 8세
생존확률: 46.83%
```

```
C:\Users\daniel_wj\anaconda3\envs\gpuDL\Lib\site-packages\sklearn\base.py:486: Us
erWarning: X has feature names, but StandardScaler was fitted without feature nam
es
  warnings.warn(
```

추가 실습

- 여러개의 특징을 선택 및 신경망의 뉴런 추가 등으로 성능을 개선시켜 보자.