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**

목차

01. 라이브러리 및 데이터 불러오기

목차로 이동하기

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import pandas as pd
import tensorflow as tf
```

In [2]:

```
import keras
from keras.models import Sequential
from keras.layers import Dense
```

In [3]:

```
print(keras.__version__)
```

```
2.9.0
```

In [4]:

```
train = pd.read_csv("./titanic/train.csv")
test = pd.read_csv("./titanic/test.csv")
print(train.shape, test.shape)
```

```
(891, 12) (418, 11)
```

In [5]:

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [6]:

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Pclass       418 non-null    int64  
 2   Name         418 non-null    object 
 3   Sex          418 non-null    object 
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64  
 6   Parch        418 non-null    int64  
 7   Ticket       418 non-null    object 
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object 
 10  Embarked     418 non-null    object 
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

02. 입력 및 출력 지정

목차로 이동하기

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

In [10]:

```
input_col = ['Pclass', 'SibSp', 'Parch']
labeled_col = ['Survived']
```

In [11]:

```
X = train[ input_col ]
y = train[ labeled_col ]
X_val = test[ input_col ]
```

In [12]:

```
seed = 0
np.random.seed(seed)
```

In [13]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                     random_state=0)
```

In [14]:

```
print(X_train.shape, X_test.shape)
print()
print(y_train.shape, y_test.shape)
```

```
(668, 3) (223, 3)

(668, 1) (223, 1)
```

03. 딥러닝 구축 및 학습시키기

목차로 이동하기

In [15]:

```
from keras.models import Sequential
from keras.layers import Dense
```

In [16]:

```
model = Sequential()
model.add(Dense(30, input_dim=3, activation='relu'))
model.add(Dense(15, activation='relu') )
model.add(Dense(1, activation='sigmoid'))
```

딥러닝 설정 및 학습

In [17]:

```
model.compile(loss = 'binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
model.fit(X_train, y_train, epochs=100, batch_size=10)
```

```
6946
Epoch 13/100
67/67 [==============================] - 0s 2ms/step - loss: 0.6046 - accuracy: 0.
6856
Epoch 14/100
67/67 [==============================] - 0s 2ms/step - loss: 0.6014 - accuracy: 0.
6826
Epoch 15/100
67/67 [==============================] - 0s 2ms/step - loss: 0.5990 - accuracy: 0.
6886
Epoch 16/100
67/67 [==============================] - 0s 2ms/step - loss: 0.5982 - accuracy: 0.
6901
Epoch 17/100
67/67 [==============================] - 0s 2ms/step - loss: 0.5961 - accuracy: 0.
6901
Epoch 18/100
67/67 [==============================] - 0s 2ms/step - loss: 0.5964 - accuracy: 0.
6901
```

모델 평가

In [18]:

```
model.evaluate(X_test, y_test)
```

```
7/7 [==============================] - 0s 1ms/step - loss: 0.5864 - accuracy: 0.7309
```

Out[18]:

```
[0.5864036083221436, 0.7309417128562927]
```

In [19]:

```
print("₩n Accuracy : %.4f" % (model.evaluate(X_test, y_test)[1]))
```

```
7/7 [==============================] - 0s 3ms/step - loss: 0.5864 - accuracy: 0.7309

 Accuracy : 0.7309
```

In [20]:

```
pred = model.predict(X_val)
```

```
14/14 [==============================] - 0s 1ms/step
```

In [21]:

```
sub = pd.read_csv("./titanic/gender_submission.csv")
sub.columns
```

Out[21]:

```
Index(['PassengerId', 'Survived'], dtype='object')
```

In [22]:

```
pred[:, 0] > 0.5
```

Out[22]:

```
array([False, False, False, False, False, False, False,  True, False,
       False, False,  True,  True, False,  True, False, False, False,
       False, False,  True,  True,  True,  True,  True, False,  True,
       False,  True, False, False, False, False, False,  True, False,
       False, False, False, False,  True,  True, False, False,  True,
       False,  True, False,  True,  True,  True, False,  True,  True,
       False, False, False, False, False,  True, False, False, False,
       False,  True, False, False,  True,  True, False, False, False,
       False,  True,  True,  True, False,  True, False, False, False,
        True,  True, False, False, False, False, False, False,  True,
       False, False,  True, False,  True, False,  True, False, False,
       False,  True, False, False, False, False, False, False, False,
       False, False, False, False,  True, False,  True, False, False,
       False,  True, False, False, False,  True, False, False,  True,
       False, False, False, False, False,  True, False, False, False,
       False, False, False, False, False, False,  True,  True, False,
        True, False,  True, False,  True,  True,  True, False, False,
        True, False, False,  True, False,  True,  True, False, False,
       False, False, False, False,  True, False,  True, False, False,
       False, False, False, False,  True, False,  True, False,  True,
       False,  True,  True, False,  True, False,  True, False, False,
       False, False,  True, False, False,  True, False,  True, False,
       False, False, False,  True,  True,  True, False,  True, False,
       False,  True, False, False, False, False, False, False,  True,
       False,  True,  True, False, False, False, False, False,  True,
        True, False, False, False, False, False,  True, False, False,
        True, False,  True, False,  True,  True,  True,  True,  True,
       False, False,  True, False,  True, False, False,  True, False,
        True, False, False, False, False, False, False, False, False,
       False,  True, False, False, False,  True, False, False, False,
        True, False,  True, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False,  True,
       False, False,  True, False, False,  True, False, False,  True,
       False,  True, False, False, False,  True, False, False,  True,
        True,  True,  True, False, False, False, False, False,  True,
       False,  True, False, False, False, False, False, False,  True,
        True, False,  True,  True, False, False,  True,  True, False,
       False, False,  True, False,  True, False, False, False, False,
       False,  True, False, False, False, False, False, False,  True,
       False, False,  True, False,  True,  True, False, False, False,
       False,  True, False, False,  True, False, False, False,  True,
       False, False,  True,  True, False,  True,  True, False, False,
        True, False, False, False, False, False, False,  True, False,
       False, False, False,  True,  True,  True, False, False,  True,
       False,  True, False, False,  True, False,  True,  True,  True,
       False, False,  True, False, False, False,  True, False, False,
        True, False, False, False])
```

In [23]:

```
sub['Survived'] = pred[:, 0] > 0.5
```

In [24]:

```
sub.loc[sub['Survived']==True, 'Survived'] = 1
sub.loc[sub['Survived']==False, 'Survived'] = 0
```

In [26]:

```
sub.to_csv("titanic_submit.csv", index=False)
```

추가 실습

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