딥러닝 모델 구현해 보기

학습 내용

- 자전거 공유 업체 시간대별 대여 대수의 데이터을 이용한 딥러닝 모델 만들기
- URL : https://www.kaggle.com/competitions/bike-sharing-demand (https://www.kaggle.com/competitions/bike-sharing-demand)

목차

01. 라이브러리 임포트

목차로 이동하기

In [1]:

```
import tensorflow as tf
import keras

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

In [2]:

```
print("tf version : {}".format(tf.__version__))
print("keras version : {}".format(keras.__version__))
print("numpy version : {}".format(np.__version__))
print("matplotlib version : {}".format(matplotlib.__version__))
print("pandas version : {}".format(pd.__version__))
```

```
tf version : 2.10.0
keras version : 2.10.0
numpy version : 1.21.5
matplotlib version : 3.5.1
pandas version : 1.4.2
```

02. 데이터 셋 로드 및 데이터 탐색

목차로 이동하기

In [3]:

```
## train 데이터 셋 , test 데이터 셋
## train 은 학습을 위한 입력 데이터 셋
## test 은 예측을 위한 새로운 데이터 셋(평가)
## parse_dates : datetime 컬럼을 시간형으로 불러올 수 있음
train = pd.read_csv("./bike/bike_mod_tr.csv", parse_dates=['datetime'])
test = pd.read_csv("./bike/bike_mod_test.csv", parse_dates=['datetime'])
```

데이터 탐색

In [4]:

```
train.columns
```

Out[4]:

```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp',
       'atemp', 'humidity', 'windspeed', 'casual', 'registered', 'count',
       'year', 'month', 'day', 'hour', 'minute', 'second', 'dayofweek'],
      dtype='object')
```

In [5]:

```
test.columns
```

Out[5]:

```
Index(['datetime', 'season', 'holiday', 'workingday', 'weather', 'temp',
       'atemp', 'humidity', 'windspeed', 'year', 'month', 'day', 'dayofweek',
       'hour', 'minute', 'second'],
      dtype='object')
```

In [6]:

```
print(train.info())
print()
print(test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 19 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   datetime    10886 non-null  datetime64[ns]
 1   season      10886 non-null  int64         
 2   holiday     10886 non-null  int64         
 3   workingday  10886 non-null  int64         
 4   weather     10886 non-null  int64         
 5   temp        10886 non-null  float64       
 6   atemp       10886 non-null  float64       
 7   humidity    10886 non-null  int64         
 8   windspeed   10886 non-null  float64       
 9   casual      10886 non-null  int64         
 10  registered  10886 non-null  int64         
 11  count       10886 non-null  int64         
 12  year        10886 non-null  int64         
 13  month       10886 non-null  int64         
 14  day         10886 non-null  int64         
 15  hour        10886 non-null  int64         
 16  minute      10886 non-null  int64         
 17  second      10886 non-null  int64         
 18  dayofweek   10886 non-null  int64         
dtypes: datetime64[ns](1), float64(3), int64(15)
memory usage: 1.6 MB
None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6493 entries, 0 to 6492
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   datetime    6493 non-null   datetime64[ns]
 1   season      6493 non-null   int64         
 2   holiday     6493 non-null   int64         
 3   workingday  6493 non-null   int64         
 4   weather     6493 non-null   int64         
 5   temp        6493 non-null   float64       
 6   atemp       6493 non-null   float64       
 7   humidity    6493 non-null   int64         
 8   windspeed   6493 non-null   float64       
 9   year        6493 non-null   int64         
 10  month       6493 non-null   int64         
 11  day         6493 non-null   int64         
 12  dayofweek   6493 non-null   int64         
 13  hour        6493 non-null   int64         
 14  minute      6493 non-null   int64         
 15  second      6493 non-null   int64         
dtypes: datetime64[ns](1), float64(3), int64(12)
memory usage: 811.8 KB
None
```

03. 입력(input)과 출력(output) 지정

목차로 이동하기

X : weather, temp (시간, 온도)

y : count - 자전거 시간대별 렌탈 대수

In [7]:

```
input_col = [ 'weather', 'temp']
labeled_col = ['count']
```

In [8]:

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

데이터 나누기

In [9]:

```
from sklearn.model_selection import train_test_split
```

In [10]:

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

In [11]:

```
print(X_train.shape)
print(X_test.shape)
```

```
(8164, 2)
(2722, 2)
```

In [12]:

```
### 난수 발생 패턴 결정 0
seed = 0
np.random.seed(seed)
```

04. 딥러닝 모델 만들기 및 학습

목차로 이동하기

- 케라스 라이브러리 중에서 Sequential 함수는 딥러닝의 구조를 한층 한층 쉽게 쌓아올릴 수 있다.
- Sequential() 함수 선언 후, 신경망의 층을 쌓기 위해 model.add() 함수를 사용한다
- input_dim 입력층 노드의 수
- activation - 활성화 함수 선언 (relu, sigmoid)
- Dense() 함수를 이용하여 각 층에 세부 내용을 설정해 준다.

In [13]:

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

In [14]:

```
model = Sequential()
model.add(Dense(32, input_dim=2, activation='relu'))
model.add(Dense(16, activation='relu') )
model.add(Dense(16, activation='relu') )
model.add(Dense(1))
```

In [15]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 32)                96

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 16)                272

 dense_3 (Dense)             (None, 1)                 17

=================================================================
Total params: 913
Trainable params: 913
Non-trainable params: 0
_________________________________________________________________
```

미니배치의 이해

- 이미지를 하나씩 학습시키는 것보다 여러 개를 한꺼번에 학습시키는 쪽이 효과가 좋다.
- 많은 메모리와 높은 컴퓨터 성능이 필요하므로 일반적으로 데이터를 적당한 크기로 잘라서 학습시킨다.
  - **미니배치**라고 한다.

In [16]:

```
## 학습의 조기 종료 함수 - EarlyStopping()
from keras.callbacks import EarlyStopping
early_stopping = EarlyStopping(patience = 10) # 조기종료 콜백함수 정의
```

딥러닝 compile과 모델 학습(최적화)

In [17]:

```
model.compile(loss = 'mean_squared_error', optimizer='rmsprop')
model.fit(X_train, y_train, epochs=100,
          validation_data=[X_test, y_test],
          batch_size=16,
          callbacks=[early_stopping])
```

```
511/511 [==============================] - 1s 2ms/step - loss: 27475.8105 - val_lo
ss: 27217.1191
Epoch 54/100
511/511 [==============================] - 1s 2ms/step - loss: 27471.6152 - val_lo
ss: 27293.3496
Epoch 55/100
511/511 [==============================] - 1s 2ms/step - loss: 27458.7949 - val_lo
ss: 27258.4531
Epoch 56/100
511/511 [==============================] - 1s 2ms/step - loss: 27432.8184 - val_lo
ss: 27233.8965
Epoch 57/100
511/511 [==============================] - 1s 2ms/step - loss: 27426.7031 - val_lo
ss: 27146.7656
Epoch 58/100
511/511 [==============================] - 1s 2ms/step - loss: 27437.7188 - val_lo
ss: 27213.7520

Epoch 59/100
511/511 [==============================] - 1s 2ms/step - loss: 27446.7227 - val_lo
```

모델 평가

In [18]:

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

```
86/86 [==============================] - 0s 1ms/step - loss: 27185.6953
```

Out[18]:

```
27185.6953125
```

예측 수행

In [19]:

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

```
203/203 [==============================] - 0s 1ms/step
```

In [20]:

```
sub = pd.read_csv("./bike/sampleSubmission.csv")
sub['count'] = pred

sub.loc[sub['count']<0, 'count'] = 0
```

In [21]:

```
sub.to_csv("./bike/nn_sub_2207.csv", index=False)
```

추가 실습

변수를 추가를 통해 성능을 향상시켜보자(5-10분) - epoch수도 증가

- (예) ['hour', 'temp', 'dayofweek', 'workingday', 'season', 'weather']
- (예) 200epoch, ['hour', 'temp', 'dayofweek', 'workingday', 'season', 'weather']
- (예) 300epoch, ['hour', 'temp', 'dayofweek', 'workingday', 'season', 'weather']
- input_col = ['hour', 'temp', 'weather', 'season', 'holiday', 'temp', 'workingday', 'windspeed'] 300epoch