RNN

In [1]:

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

from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
```

In [3]:

```
# 파일
df = pd.read_csv('전체데이터_병합.csv',encoding='cp949',parse_dates=['y_m'])
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1666 entries, 0 to 1665
Data columns (total 39 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   y_m                1666 non-null   datetime64[ns]
 1   city               1666 non-null   object
 2   location           1666 non-null   object
 3   area_cnt           1666 non-null   float64
 4   em_cnt             1666 non-null   int64
 5   em_g               1666 non-null   int64
 6   pay_amt            1666 non-null   int64
 7   제주도민_여             1666 non-null   float64
 8   외국인거주_여            1666 non-null   float64
 9   제주도민_남             1666 non-null   float64
 10  외국인거주_남            1666 non-null   float64
 11  제주도민_60이상          1666 non-null   float64
 12  제주도민_60미만          1666 non-null   float64
 13  total_pop          1666 non-null   float64
 14  패스트푸드_결제건수         1666 non-null   float64
 15  패스트푸드_결제금액         1666 non-null   float64
 16  간식_결제건수            1666 non-null   float64
 17  간식_결제금액            1666 non-null   float64
 18  농축수산물_결제건수         1666 non-null   float64
 19  농축수산물_결제금액         1666 non-null   float64
 20  마트/슈퍼마켓_결제건수       1666 non-null   float64
 21  마트/슈퍼마켓_결제금액       1666 non-null   float64
 22  식품_결제건수            1666 non-null   float64
 23  식품_결제금액            1666 non-null   float64
 24  배달_결제건수            1666 non-null   float64
 25  배달_결제금액            1666 non-null   float64
 26  식당_결제건수            1666 non-null   float64
 27  식당_결제금액            1666 non-null   float64
 28  풍속                 1666 non-null   float64
 29  기온                 1666 non-null   float64
 30  습도                 1666 non-null   float64
 31  강수                 1666 non-null   float64
 32  전국_누적확진자           1666 non-null   float64
 33  전국_월별확진자           1666 non-null   float64
 34  제주_누적확진자           1666 non-null   float64
 35  제주_월별확진자           1666 non-null   float64
 36  visit_pop_cnt      1666 non-null   float64
 37  visit_pop_cnt_lf   1666 non-null   float64
 38  visit_pop_cnt_sf   1666 non-null   float64
```

```
dtypes: datetime64[ns](1), float64(33), int64(3), object(2)
memory usage: 507.7+ KB
```

Out[3]:

	y_m	city	location	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_여	제주도민_남	...	기
0	2018-01-01	서귀포시	남원읍	52.0	9570	42437700	1270773	9306.0	200.0	9806.0	...	6.25658
1	2018-01-01	서귀포시	대륜동	38.0	21666	57612600	1676850	6637.0	95.0	6836.0	...	8.00430
2	2018-01-01	서귀포시	대정읍	89.0	10185	38885550	1164122	10725.0	677.0	10360.0	...	5.41787
3	2018-01-01	서귀포시	대천동	37.0	20280	53858550	1593709	6475.0	137.0	6685.0	...	8.00430
4	2018-01-01	서귀포시	동홍동	49.0	45936	118701000	3501286	11569.0	642.0	11124.0	...	5.77150

5 rows × 39 columns

In [4]:

```
df.isnull().sum()
```

Out[4]:

```
y_m                    0
city                   0
location               0
area_cnt               0
em_cnt                 0
em_g                   0
pay_amt                0
제주도민_여            0
외국인거주_여          0
제주도민_남            0
외국인거주_남          0
제주도민_60이상        0
제주도민_60미만        0
total_pop              0
패스트푸드_결제건수    0
패스트푸드_결제금액    0
간식_결제건수          0
간식_결제금액          0
농축수산물_결제건수    0
농축수산물_결제금액    0
마트/슈퍼마켓_결제건수 0
마트/슈퍼마켓_결제금액 0
```

```
식품_결제건수            0
식품_결제금액            0
배달_결제건수            0
배달_결제금액            0
식당_결제건수            0
식당_결제금액            0
풍속                  0
기온                  0
습도                  0
강수                  0
전국_누적확진자           0
전국_월별확진자           0
제주_누적확진자           0
제주_월별확진자           0
visit_pop_cnt       0
visit_pop_cnt_lf    0
visit_pop_cnt_sf    0
dtype: int64
```

In [5]:

```python
df = df.fillna(0)
```

In [6]:

```python
df['y_m'] = pd.to_datetime(df['y_m'],format='%Y%m')
df['year'] = df['y_m'].dt.year
df
```

Out[6]:

	y_m	city	location	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_여	제주도민_남	...	
0	2018-01-01	서귀포시	남원읍	52.0	9570	42437700	1270773	9306.0	200.0	9806.0	...	62
1	2018-01-01	서귀포시	대륜동	38.0	21666	57612600	1676850	6637.0	95.0	6836.0	...	66
2	2018-01-01	서귀포시	대정읍	89.0	10185	38885550	1164122	10725.0	677.0	10360.0	...	70
3	2018-01-01	서귀포시	대천동	37.0	20280	53858550	1593709	6475.0	137.0	6685.0	...	66
4	2018-01-01	서귀포시	동홍동	49.0	45936	118701000	3501286	11569.0	642.0	11124.0	...	69
...	...	...	...	...	...	...	...	...	...	...	...	
1661	2021-06-01	제주시	일도2동	87.0	84360	147438200	4402149	16569.0	200.0	16077.0	...	75
1662	2021-06-01	제주시	조천읍	141.0	27732	63927750	1911187	12422.0	242.0	13017.0	...	82

	y_m	city	location	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_여	제주도민_남	...	
1663	2021-06-01	제주시	한경면	71.0	8031	27060150	809898	4531.0	100.0	4627.0	...	93
1664	2021-06-01	제주시	한림읍	112.0	25653	82746990	2476292	10341.0	1140.0	10891.0	...	92
1665	2021-06-01	제주시	화북동	84.0	66088	110750050	3306029	12238.0	161.0	12062.0	...	75

1666 rows × 40 columns

In [7]:

```
# df_l = df['location'] == '노형동'
# dfl = df[df_l]
# dfl
```

In [8]:

```
# df_l = df['location'] == '건입동'
# dfl = df[df_l]
# dfl
```

In [9]:

```
# df_l = df['location'] == '연동'
# dfl = df[df_l]
# dfl
```

In [10]:

```
df_l = df['location'] == '이도2동'
dfl = df[df_l]
dfl.head()
```

Out[10]:

	y_m	city	location	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_여	제주도민_남	...	
33	2018-01-01	제주시	이도2동	132.0	99670	246651600	7371540	25593.0	344.0	24510.0	...	66.3
71	2018-02-01	제주시	이도2동	132.0	92603	230440750	6886558	25488.0	332.0	24417.0	...	60.7
109	2018-03-01	제주시	이도2동	132.0	103323	249276500	7448453	25421.0	341.0	24400.0	...	73.1

	y_m	city	location	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_여	제주도민_남	...	
147	2018-04-01	제주시	이도2동	132.0	102728	227249550	6788163	25420.0	342.0	24393.0	...	68.8
185	2018-05-01	제주시	이도2동	132.0	113606	234533700	7004233	25435.0	354.0	24448.0	...	75.8

5 rows × 40 columns

In [13]:

```
# 정규화
scaler = MinMaxScaler()
scale_cols =  dfl.drop(columns=['y_m', 'city','location'], axis=1)
scale_cols[:] = scaler.fit_transform(scale_cols[:])
# 유의변수 추출
scale_cols = scale_cols.drop(columns=['제주도민_60미만','total_pop','식당_결제금액','
scale_cols.head()
```

Out[13]:

	area_cnt	em_cnt	em_g	pay_amt	제주도민_여	외국인거주_남	제주도민_60이상	농축수산물_결제건수	마트/슈퍼마켓_결제금액	식품_제금
33	0.0	0.245119	0.530298	0.531984	1.000000	0.000000	0.000000	0.186319	0.599766	0.0033
71	0.0	0.140992	0.403869	0.405325	0.732824	0.015873	0.018366	0.190725	0.558672	0.1304
109	0.0	0.298944	0.550769	0.552070	0.562341	0.095238	0.035465	0.000000	0.528745	0.0819
147	0.0	0.290177	0.378981	0.379629	0.559796	0.126984	0.050032	0.097623	0.610611	0.0000
185	0.0	0.450456	0.435790	0.436058	0.597964	1.000000	0.089297	0.273507	0.852501	0.1526

In [14]:

```
# 데이터셋 분리(시계열)

TEST_SIZE =  36# 3년 데이터
WINDOW_SIZE = 6 # 6개월 데이터

test = scale_cols[:-TEST_SIZE]
train = scale_cols[-TEST_SIZE:]
```

In [15]:

```
# 훈련데이터와 테스트데이터 분리에 사용
def make_dataset(data, label, window_size=20):
    feature_list = []
    label_list = []
    for i in range(len(data) - window_size):
        feature_list.append(np.array(data.iloc[i:i+window_size]))
        label_list.append(np.array(label.iloc[i+window_size]))
    return np.array(feature_list), np.array(label_list)
```

In [16]:

```
# 훈련데이터와 테스트데이터 분리
feature_cols = scale_cols.columns
label_cols = ['em_g']

train_feature = train[feature_cols]
train_label = train[label_cols]

train_feature, train_label = make_dataset(train_feature, train_label, 20)

x_train, x_valid, y_train, y_valid = train_test_split(train_feature, train_label, tes
x_train.shape, x_valid.shape
```

Out[16]:

```
((12, 20, 17), (4, 20, 17))
```

In [17]:

```
test_feature = test[feature_cols]
test_label = test[label_cols]

test_feature.shape, test_label.shape
```

Out[17]:

```
((6, 17), (6, 1))
```

In [18]:

```
test_feature, test_label = make_dataset(test_feature, test_label, 4)
test_feature.shape, test_label.shape
```

Out[18]:

```
((2, 4, 17), (2, 1))
```

In [19]:

```
# tensorflow, keras
import tensorflow as tf
from tensorflow import keras
```

In [20]:

```
# keras 모형
from keras.models import Sequential
from keras.layers import Dense
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.layers import LSTM

model = Sequential()
model.add(LSTM(16,
               input_shape=(train_feature.shape[1], train_feature.shape[2]),
               activation='relu',
               return_sequences=False)
          )

model.add(Dense(1))
```

In [21]:

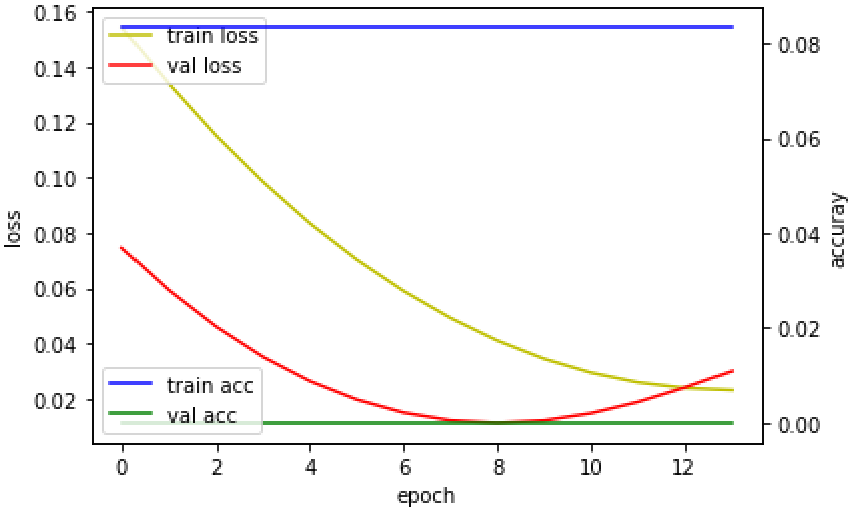
```
import os

model.compile(loss='mean_squared_error', optimizer='adam',metrics=["acc"]) # acc 안나
early_stop = EarlyStopping(monitor='val_loss', patience=5)

model_path = 'model'
filename = os.path.join(model_path, 'tmp_checkpoint.h5')
checkpoint = ModelCheckpoint(filename, monitor='val_loss', verbose=1, save_best_only=

history = model.fit(x_train, y_train,
                                    epochs=200,
                                    batch_size=16,
                                    validation_data=(x_valid, y_valid),
                                    callbacks=[early_stop, checkpoint])
```

```
Epoch 1/200
1/1 [==============================] - 2s 2s/step - loss: 0.1544 - acc: 0.0833 - val_l
oss: 0.0746 - val_acc: 0.0000e+00

Epoch 00001: val_loss improved from inf to 0.07465, saving model to model₩tmp_checkpoi
nt.h5
Epoch 2/200
1/1 [==============================] - 0s 44ms/step - loss: 0.1339 - acc: 0.0833 - val
_loss: 0.0593 - val_acc: 0.0000e+00

Epoch 00002: val_loss improved from 0.07465 to 0.05925, saving model to model₩tmp_chec
kpoint.h5
Epoch 3/200
1/1 [==============================] - 0s 48ms/step - loss: 0.1152 - acc: 0.0833 - val
_loss: 0.0462 - val_acc: 0.0000e+00

Epoch 00003: val_loss improved from 0.05925 to 0.04617, saving model to model₩tmp_chec
kpoint.h5
Epoch 4/200
1/1 [==============================] - 0s 48ms/step - loss: 0.0985 - acc: 0.0833 - val
_loss: 0.0352 - val_acc: 0.0000e+00

Epoch 00004: val_loss improved from 0.04617 to 0.03521, saving model to model₩tmp_chec
kpoint.h5
Epoch 5/200
1/1 [==============================] - 0s 45ms/step - loss: 0.0835 - acc: 0.0833 - val
_loss: 0.0265 - val_acc: 0.0000e+00

Epoch 00005: val_loss improved from 0.03521 to 0.02648, saving model to model₩tmp_chec
kpoint.h5
Epoch 6/200
1/1 [==============================] - 0s 48ms/step - loss: 0.0704 - acc: 0.0833 - val
_loss: 0.0198 - val_acc: 0.0000e+00

Epoch 00006: val_loss improved from 0.02648 to 0.01981, saving model to model₩tmp_chec
kpoint.h5
Epoch 7/200
1/1 [==============================] - 0s 52ms/step - loss: 0.0591 - acc: 0.0833 - val
_loss: 0.0151 - val_acc: 0.0000e+00

Epoch 00007: val_loss improved from 0.01981 to 0.01508, saving model to model₩tmp_chec
kpoint.h5
Epoch 8/200
1/1 [==============================] - 0s 45ms/step - loss: 0.0493 - acc: 0.0833 - val
_loss: 0.0123 - val_acc: 0.0000e+00

Epoch 00008: val_loss improved from 0.01508 to 0.01229, saving model to model₩tmp_chec
kpoint.h5
Epoch 9/200
1/1 [==============================] - 0s 53ms/step - loss: 0.0411 - acc: 0.0833 - val
_loss: 0.0114 - val_acc: 0.0000e+00

Epoch 00009: val_loss improved from 0.01229 to 0.01137, saving model to model₩tmp_chec
kpoint.h5
Epoch 10/200
1/1 [==============================] - 0s 50ms/step - loss: 0.0345 - acc: 0.0833 - val
_loss: 0.0123 - val_acc: 0.0000e+00

Epoch 00010: val_loss did not improve from 0.01137
Epoch 11/200
1/1 [==============================] - 0s 52ms/step - loss: 0.0295 - acc: 0.0833 - val
_loss: 0.0149 - val_acc: 0.0000e+00
```

```
Epoch 00011: val_loss did not improve from 0.01137
Epoch 12/200
1/1 [==============================] - 0s 58ms/step - loss: 0.0261 - acc: 0.0833 - val
_loss: 0.0189 - val_acc: 0.0000e+00

Epoch 00012: val_loss did not improve from 0.01137
Epoch 13/200
1/1 [==============================] - 0s 56ms/step - loss: 0.0240 - acc: 0.0833 - val
_loss: 0.0241 - val_acc: 0.0000e+00

Epoch 00013: val_loss did not improve from 0.01137
Epoch 14/200
1/1 [==============================] - 0s 56ms/step - loss: 0.0233 - acc: 0.0833 - val
_loss: 0.0301 - val_acc: 0.0000e+00

Epoch 00014: val_loss did not improve from 0.01137
```

In [23]:

```
fig, loss_ax = plt.subplots()

acc_ax = loss_ax.twinx()

loss_ax.plot(history.history['loss'], 'y', label='train loss')
loss_ax.plot(history.history['val_loss'], 'r', label='val loss')

acc_ax.plot(history.history['acc'], 'b', label='train acc')
acc_ax.plot(history.history['val_acc'], 'g', label='val acc')

loss_ax.set_xlabel('epoch')
loss_ax.set_ylabel('loss')
acc_ax.set_ylabel('accuray')
loss_ax.legend(loc='upper left')
acc_ax.legend(loc='lower left')

# plt.savefig('rnn_l.png')
```



In [24]:

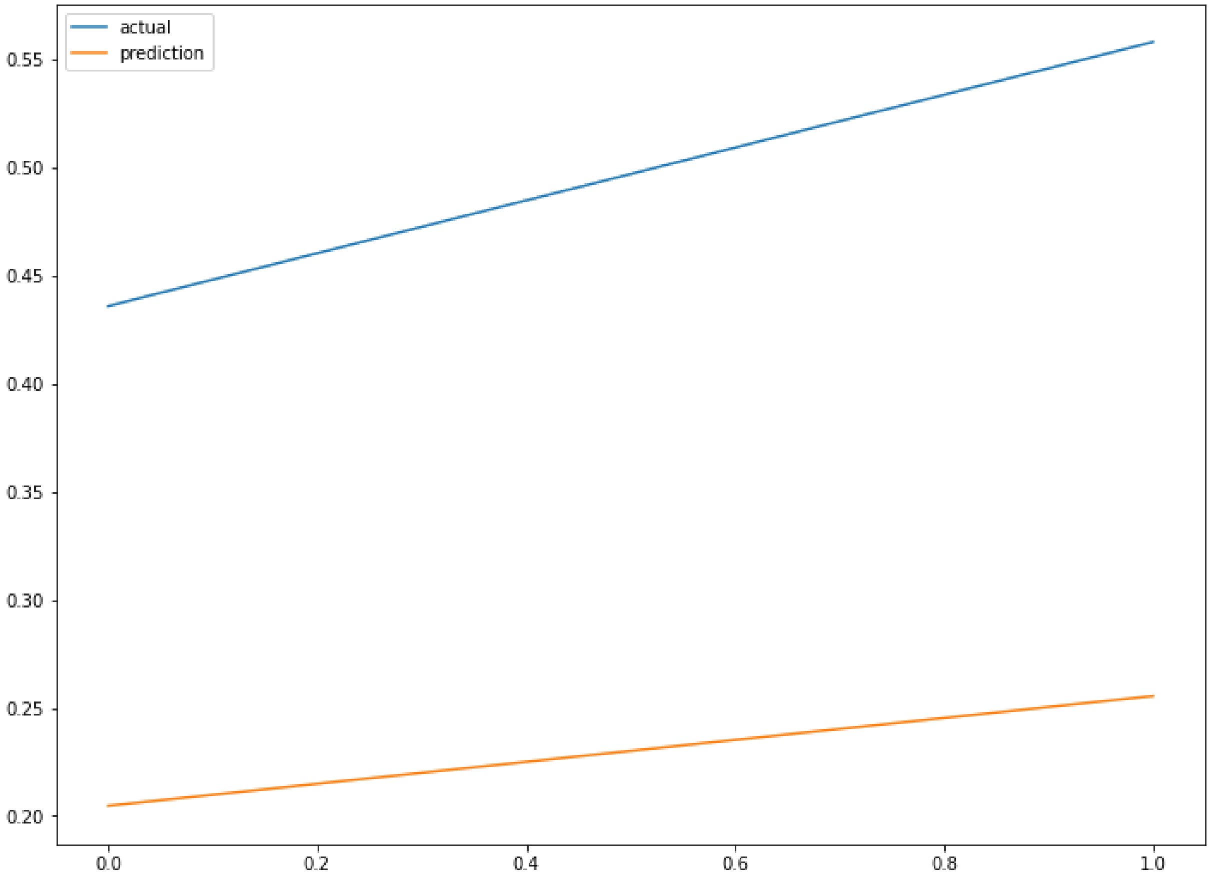
```
model.load_weights(filename)
pred = model.predict(test_feature)

pred.shape
```

```
WARNING:tensorflow:Model was constructed with shape (None, 20, 17) for input KerasTens
or(type_spec=TensorSpec(shape=(None, 20, 17), dtype=tf.float32, name='lstm_input'), na
me='lstm_input', description="created by layer 'lstm_input'"), but it was called on an
input with incompatible shape (None, 4, 17).
(2, 1)
```

Out[24]:

In [25]:

```
plt.figure(figsize=(12, 9))
plt.plot(test_label, label = 'actual')
plt.plot(pred, label = 'prediction')
plt.legend()
# plt.savefig('rnn_l_p.png')
```


actual
prediction