In [1]:

```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D, MaxPooling2D
from keras.optimizers import SGD
from keras.utils import np_utils
from scipy import misc
import glob
import matplotlib.pyplot as plt
from PIL import Image
import math
import pandas as pd
```

```
Using TensorFlow backend.
```

모델과 r2

In [2]:

```
seed = 7
np.random.seed(seed)
width = 1
height = 1

# R2
def r_squared(y_true, y_hat):
    ssr = 0
    sst = 0
    e = np.subtract(y_true, y_hat)
    y_mean = np.mean(y_true)
    for item in e:
        ssr += item**2
    for item in y_true:
        sst += (item - y_mean)**2
    r2 = 1 - ssr / sst
    return r2

#compile()함수 이용해서 모델 학습과정 설정하기
def compile_model(model):
    lrate = 0.01
    sgd = SGD(lr=lrate, momentum=0.9, decay=1e-6, nesterov=True)
    model.compile(loss='sparse_categorical_crossentropy', optimizer=sgd)
    return model

#model
def create_model():
    model = Sequential()

    model.add(Convolution2D(32, 3, 3,
                            border_mode='valid',
                            input_shape=(100, 100, 3)))
    model.add(Activation('relu'))
    model.add(Convolution2D(32, 3, 3))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))

    model.add(Convolution2D(64, 3, 3,
                            border_mode='valid'))
    model.add(Activation('relu'))
    model.add(Convolution2D(64, 3, 3))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))

    model.add(Flatten())
    model.add(Dense(256))
    model.add(Activation('relu'))
    model.add(Dropout(0.5))

    model.add(Dense(2))
    model.add(Activation('softmax'))

    return model
```

데이터추출

In [3]:

```
#데이터추출 ---- 여기가 문제 같음


def extract_data():
    file_name = 'merge_data_Daehan.csv' #해당파일 open
    infile = open(file_name, 'r')
    temp_buffer = []
    for line in infile:
        temp_buffer.append(line.strip('₩n')) # '₩n'기준으로 temp_buffer에 append
    temp_buffer = temp_buffer[8:] #실질적인 data 시작지점인 8부터 append 시작
    i = 0
    groups = []
    temp = []
    for item in temp_buffer:
        #if i != 390:
        if i != 1223: #갯수 확인하고 1223개씩 나누기
            temp.append(item)
            i += 1 # 1씩 추가하기
        else:
            groups.append(temp)
            temp = []
            i = 0
    groups.append(temp)
    infile.close() #해당파일 close
    return groups

#데이터분할
def split_data(data):
    groups = []
    for item in data:
        temp_buffer = []
        for string in item:
            number = string.split(',')  # ',' 기준으로 split
            temp_buffer.append(number)  # temp_buffer에 append
        groups.append(temp_buffer)
    return groups


#분할된 데이터 모으기
def load_sample_data():
    original_data = extract_data()
    splitted_data = split_data(original_data)
    useful_data = extract_useful_data(splitted_data)
    return useful_data


#필요정보취합
def extract_useful_data(data):
    groups = []
    for group in data:
        temp_buffer = []
        for item in group:
            temp = [item[2], item[3]]
            temp = [float(i) for i in temp]
            temp_buffer.append(temp)
        groups.append(temp_buffer)
    #print(len(groups)
    return groups
```

이미지학습

In [4]:

```
#픽셀값
def get_pixel_values():
    file_name = './figures_v2'
    pixels = []
    for filename in glob.glob(file_name + '₩*.png'):
        im = misc.imread(filename)
        pixels.append(im)
    return pixels

#이미지변환
def convert_image():
    file_name = './figures_v2'
    for filename in glob.glob(file_name + '₩*.png'):
        img = Image.open(filename)
        img = img.convert('RGB')
        img.save(filename)

#이미지 plot(그림그리기)
def plot_data(data):
    #t = np.arange(0, 29, 1)
    t = np.arange(0, 33, 1)# 파일 전체 range 맞춰 변환
    file_name_number = 0
    fig = plt.figure(frameon=False, figsize=(width, height))
    for group in data:
        print(len(group))
        #count = 30 # 파일 전체 갯수가 다르기 때문에 count 범위 수정
        count = 34
        while count <= (len(group)-5):
            high = []
            low = []
            for item in group[count-34:count]:
                high.append(item[0])
                low.append(item[1])
            file_name = r'₩fig_' + str(file_name_number)
            ax = plt.Axes(fig, [0., 0., 1., 1.])
            ax.set_axis_off()
            fig.add_axes(ax)
            ax.plot(t, high[0:-1], 'b', t, low[0:-1], 'g')
            fig.savefig('./figures_v2' + file_name, dpi=100)
            fig.clf()
#             file_name_number += 1 # 10칸씩 움직이니까 10으로 수정
#             count += 11
            file_name_number += 10 # 10칸씩 움직이니까 10으로 수정
            count += 10
    print('Created %d files!' % file_name_number)
```

수익률계산

In [5]:

```
#수익률계산
def find_returns(data):
    returns = []
    price1 = []
    price2 = []
    for group in data:
        count = 34 # 나눈 window 맞춰 count 수정 at plot data
        while count <= (len(group)-5):  #count로 나눠진 각각의 값들 (5개묶음이라 시작갯수 5개 제외)
            current_data = group[count-1] #현재데이터
            future_data = group[count+4] #미래데이터
            p1 = np.mean(current_data) #계산1
            p2 = np.mean(future_data) #계산2
            price1.append(p1) #각각의 결과값 pricen에 append
            price2.append(p2)

            #math.log(p2/p1)에서 예외처리
            if p1 <= 0 or p2 <= 0: #S1. 아예 ()안이 값이 말이 안되는 경우
                returns.append(0)
                count += 10
#             elif math.log(p2/p1)>= 2:
#                 #S2. log(p2/p1)이 2보다 커서 값이 의미가 없는 경우
#                 print(p1)
#                 print(p2)
#                 print('aaaa')
#                 returns.append(0)
#                 count += 10
            else:
                returns.append(math.log(p2/p1)) #S3. 그외 정상적인경우
                count += 10

#     print(returns)
#     print(price1)
#     print(price2)
    return returns
```

In [6]:

```
def main(): #실행
    print('load_sample_data()')
    data= load_sample_data()
    print('plot_data(data)')
    plot_data(data)
    convert_image()
    print('convert_image()')
    x = np.asarray(get_pixel_values())
    y = np.asarray(find_returns(data))
    x_train = x[0:92463] # train, test set 각각의 학습
    y_train = y[0:92463]
    x_test = x[92463:]
    y_test = y[92463:]
#     x_train = x[0:4340]
#     y_train = y[0:4340]
#     x_test = x[0:4340]
#     y_test = y[0:4340]

#    y_true = y_test
#    y_train = np_utils.to_categorical(y_train, 2)
#    y_test = np_utils.to_categorical(y_test, 2)

    x_train = x_train.astype('float32') #train,test set 타입변경
    x_test = x_test.astype('float32')
    x_train /= 255.0
    x_test /= 255.0

    model = create_model()
    model = compile_model(model)

    print('cnn')

    # Fit the model
    epochs = 10
    model.fit(x_train, y_train, validation_data=(x_test, y_test), #모델학습과정
              nb_epoch=epochs,
              shuffle=True, batch_size=100, verbose=1)
#    scores = model.evaluate(x_test, y_test, verbose=0)
#    print('Accuracy: %.2f%%' % (scores[1] * 100))
    classes = model.predict_classes(x_test, verbose=0)
    # classes = list(classes) #
    list_classes = list(classes)
    #y_test = list(y_test)
    list_y_test = list(y_test)
    r2 = r_squared(list_y_test, list_classes)
    print(r2)


if __name__ == '__main__':
    main()
```

```
load_sample_data()
plot_data(data)
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
```

```
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
1223
Created 925820 files!
convert_image()
```

```
C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:6: DeprecationW
arning: `imread` is deprecated!
`imread` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``imageio.imread`` instead.
```

```
WARNING:tensorflow:From C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩tensorflow₩py
thon₩framework₩op_def_library.py:263: colocate_with (from tensorflow.python.framewor
k.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩keras₩backend
₩tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) wit
h keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_pro
b`.
```

```
C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:32: UserWarnin
g: Update your `Conv2D` call to the Keras 2 API: `Conv2D(32, (3, 3), input_shape=(10
0, 100,..., padding="valid")`
C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:34: UserWarnin
g: Update your `Conv2D` call to the Keras 2 API: `Conv2D(32, (3, 3))`
C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:40: UserWarnin
g: Update your `Conv2D` call to the Keras 2 API: `Conv2D(64, (3, 3), padding="vali
d")`

cnn
WARNING:tensorflow:From C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩tensorflow₩py
thon₩ops₩math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is depreca
ted and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.

C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:42: UserWarnin
g: Update your `Conv2D` call to the Keras 2 API: `Conv2D(64, (3, 3))`
C:₩Users₩user₩.conda₩envs₩ml₩lib₩site-packages₩ipykernel_launcher.py:37: UserWarnin
g: The `nb_epoch` argument in `fit` has been renamed `epochs`.

Train on 92463 samples, validate on 119 samples
Epoch 1/10
92463/92463 [==============================] - 2804s 30ms/step - loss: 0.0014 - val_
loss: 1.1921e-07
Epoch 2/10
92463/92463 [==============================] - 2766s 30ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 3/10
92463/92463 [==============================] - 2733s 30ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 4/10
92463/92463 [==============================] - 2699s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 5/10
92463/92463 [==============================] - 2696s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 6/10
92463/92463 [==============================] - 2689s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 7/10
92463/92463 [==============================] - 2694s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 8/10
92463/92463 [==============================] - 2686s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 9/10
92463/92463 [==============================] - 2690s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
Epoch 10/10
92463/92463 [==============================] - 2708s 29ms/step - loss: 6.9740e-04 -
val_loss: 1.1921e-07
-0.002534318568449212

<Figure size 72x72 with 0 Axes>
```

In [ ]: