boost

April 11, 2020

[72]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import networkx as nx
import matplotlib.pyplot as plt
import math

from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten, LeakyReLU, ReLU
from keras.utils import np_utils,  to_categorical
from keras import optimizers
from sklearn.model_selection import train_test_split

import os
print(os.path)
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
<module 'ntpath' from 'C:\\Users\\swcan\\Anaconda3\\lib\\ntpath.py'>
```

[77]:
```
cut_off = .485;

def file_get(s):
    train_path = "~/Dropbox/dataScience/comp_2/Y_Train_" + s + ".csv"
    test0_path = "~/Dropbox/dataScience/comp_2/Y0_" + s + ".csv"
    test1_path = "~/Dropbox/dataScience/comp_2/Y1_" + s + ".csv"
    train_df = pd.read_csv(train_path,header=None);
    test_0_df = pd.read_csv(test0_path,header=None);
    test_1_df = pd.read_csv(test1_path,header=None);
    train_df = train_df.values;
    test_0_df = test_0_df.values;
    test_1_df = test_1_df.values;

    return train_df, test_0_df, test_1_df

def add_to_datum(train,datum):
```

```python
    sz = np.shape(datum);
    ind = 0;
    for i in range(sz[1]):
        j = datum[:,i];
        m = np.sum(j);
        if (m < 0.00001):
            ind = i;
            break;

    sz_train = np.shape(train);
    if (sz_train[1] < 2):
        #now lets get to it
        datum[:,ind] = train[:sz[0],0];
        datum[:,ind+1] = train[sz[0]:,0];
    else:
        datum[:,ind] = train[:sz[0],1];
        datum[:,ind+1] = train[sz[0]:,1];

    return datum

def add_to_test(t0,t1,test):

    sz = np.shape(test);
    ind = 0;
    for i in range(sz[1]):
        j = test[:,i];
        m = np.sum(j);
        if (m < 0.00001):
            ind = i;
            break;

    test[:,ind] = t0[:,0];
    test[:,ind+1] = t1[:,0];

    return test

def make_sub(Y,cut_off,file_name):

    #Ouput the model
    sub = pd.read_csv("sample_sub.csv");
    sub = sub.values
    #print(sub)
    for i in range(sz_test[0]):
        y = 0;
        if (Y[i,0] > cut_off):
            y = 1;
        sub[i,1] = y;
    sub = pd.DataFrame(sub);
    sub.to_csv(file_name,header=['edge','label'],index=None);
```

[41]:
```
#Provide the cross validation information I have

cross_valid = [[.918,.728], #KNN
               [.938,.782], #RF50
               [.939,.791], #NN
               [.931,.777], #RF150
               [.76,.74], #SVM
               [.75,.743]] #RBF

names = ["knn","rf50","nn","rf150","svm","rbf"];

sz_cross = np.shape(cross_valid);
```

[42]:
```
#KNN as our basis
train, test0, test1 = file_get("knn");

sz_train = np.shape(train);
sz_test = np.shape(test0);
print(sz_train,sz_test)
```

```
(29552, 1) (3168, 1)
```

[43]:
```
len_train = sz_train[0] / 2;
print(len_train)
len_train = int(len_train)
datum = np.zeros((len_train,int(sz_cross[0] * 2)))
datum_test = np.zeros((sz_test[0],int(sz_cross[0] * 2)))

datum = add_to_datum(train,datum);
print(datum)
datum_test = add_to_test(test0,test1,datum_test);
print(datum_test)
```

```
14776.0
[[0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [1. 1. 0. ... 0. 0. 0.]
 ...
 [1. 1. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]]
[[0. 0. 0. ... 0. 0. 0.]
 [1. 1. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 ...
 [1. 1. 0. ... 0. 0. 0.]
 [1. 1. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]]
```

[44]:
```
for i in range(len(names) - 1):
    train, test0, test1 = file_get(names[i+1]);
    datum = add_to_datum(train,datum);
    datum_test = add_to_test(test0,test1,datum_test);
print(datum)
print(datum_test)
```

```
[[0.         0.         0.10964031 ... 0.04431844 0.         0.        ]
 [0.         0.         0.5582     ... 1.         1.         1.        ]
 [1.         1.         0.9612     ... 1.         1.         1.        ]
 ...
 [1.         1.         0.8758     ... 1.         1.         1.        ]
 [0.         0.         0.11640456 ... 0.         0.         0.        ]
 [0.         0.         0.03920813 ... 0.07522474 0.         0.        ]]
[[0.         0.         0.70499774 ... 0.6833496  0.         0.        ]
 [1.         1.         0.1514     ... 0.         1.         1.        ]
 [0.         0.         0.6448     ... 0.89284357 0.         0.        ]
 ...
 [1.         1.         0.2146     ... 0.09972205 1.         1.        ]
 [1.         1.         0.1008     ... 0.11350026 1.         1.        ]
 [0.         0.         0.79719504 ... 0.34042849 0.         0.        ]]
```

[39]:
```
#Create our supervised vector
Y = np.zeros((len_train,1));
oof = pd.read_csv("train.csv",header=None);
oof = oof.values;
print(np.shape(oof))
Y = oof[:len_train,-1];
print(Y)
```

```
(29552, 294)
[0. 1. 1. ... 1. 0. 0.]
```

[63]:
```
#Now we have everything how we want it
#Lets do a Bagged/Boosted model first
Y_out = np.zeros((sz_test[0],1))

accs = np.zeros((sz_cross[0]*2,1));
j = 0;
for i in range(sz_cross[0]):
    accs[j] = cross_valid[i][0] *  cross_valid[i][1];
    j += 1;
    accs[j] = cross_valid[i][0] *  cross_valid[i][1];
    j += 1;

accs = accs.reshape(1,-1);
#print(accs)
tot_acc = np.sum(accs)
#print(tot_acc)

#Boosting and bagging
for i in range(sz_test[0]):
    #print(datum_test[i,:],accs)
    o = np.sum(datum_test[i,:] * accs) / tot_acc;
    Y_out[i,0] = o;

make_sub(Y_out,cut_off,"boost_sub.csv")
```

[73]:
```
#Stack the model
#Use a NN
X = datum.reshape(-1,int(sz_cross[0]  * 2),1)
datum_test = datum_test.reshape(-1,int(sz_cross[0] * 2),1);

print(np.shape(X))

X_train, X_test, y_train, y_test = train_test_split(X,Y,test_size = .25);
```

```
(14776, 12, 1)
```

[75]:
```
#create convolution neural network
model = Sequential()
model.add(Dense(256, input_shape=(sz_cross[0] * 2,1)))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.5))

model.add(Flatten())

model.add(Dense(64))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.33))

model.add(Dense(64))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.33))

model.add(Dense(32))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.33))

model.add(Dense(32))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.33))

model.add(Dense(16))
model.add(LeakyReLU(alpha=0.05))
model.add(Dropout(0.33))

model.add(Dense(1, activation='sigmoid'))

model.summary();
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_7 (Dense)              (None, 12, 256)           512
_________________________________________________________________
leaky_re_lu_6 (LeakyReLU)    (None, 12, 256)           0
_________________________________________________________________
dropout_6 (Dropout)          (None, 12, 256)           0
_________________________________________________________________
flatten_2 (Flatten)          (None, 3072)              0
_________________________________________________________________
dense_8 (Dense)              (None, 64)                196672
_________________________________________________________________
leaky_re_lu_7 (LeakyReLU)    (None, 64)                0
_________________________________________________________________
dropout_7 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_9 (Dense)              (None, 64)                4160
_________________________________________________________________
leaky_re_lu_8 (LeakyReLU)    (None, 64)                0
_________________________________________________________________
dropout_8 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_10 (Dense)             (None, 32)                2080
_________________________________________________________________
leaky_re_lu_9 (LeakyReLU)    (None, 32)                0
_________________________________________________________________
dropout_9 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_11 (Dense)             (None, 32)                1056
_________________________________________________________________
leaky_re_lu_10 (LeakyReLU)   (None, 32)                0
_________________________________________________________________
dropout_10 (Dropout)         (None, 32)                0
_________________________________________________________________
dense_12 (Dense)             (None, 16)                528
```

```
_________________________________________________________________
leaky_re_lu_11 (LeakyReLU)   (None, 16)                0
_________________________________________________________________
dropout_11 (Dropout)         (None, 16)                0
_________________________________________________________________
dense_13 (Dense)             (None, 1)                 17
=================================================================
Total params: 205,025
Trainable params: 205,025
Non-trainable params: 0
_________________________________________________________________
```

[78]:
```
#Compile model
sgd = optimizers.SGD(lr=.01);
model.compile(loss='MSE',
              optimizer='adam',
              metrics=['accuracy'])

history = model.fit(X_train,y_train, shuffle=True,
          batch_size=10,epochs=25,verbose=1,
          validation_data=(X_test,y_test))
```

```
WARNING:tensorflow:From C:\Users\swcan\Anaconda3\lib\site-
packages\keras\optimizers.py:790: The name tf.train.Optimizer is deprecated.
Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From C:\Users\swcan\Anaconda3\lib\site-
packages\keras\backend\tensorflow_backend.py:988: The name tf.assign_add is
deprecated. Please use tf.compat.v1.assign_add instead.

Train on 11082 samples, validate on 3694 samples
Epoch 1/25
11082/11082 [==============================] - 11s 1ms/step - loss: 0.0140 -
acc: 0.9839 - val_loss: 0.0014 - val_acc: 0.9984
Epoch 2/25
11082/11082 [==============================] - 7s 635us/step - loss: 0.0011 -
acc: 0.9988 - val_loss: 8.3054e-04 - val_acc: 0.9992
Epoch 3/25
11082/11082 [==============================] - 7s 632us/step - loss: 0.0011 -
acc: 0.9988 - val_loss: 8.2936e-04 - val_acc: 0.9992
Epoch 4/25
11082/11082 [==============================] - 7s 607us/step - loss: 0.0012 -
acc: 0.9987 - val_loss: 0.0023 - val_acc: 0.9976
Epoch 5/25
11082/11082 [==============================] - 7s 644us/step - loss: 0.0017 -
acc: 0.9982 - val_loss: 0.0025 - val_acc: 0.9976
Epoch 6/25
```

```
11082/11082 [==============================] - 7s 627us/step - loss: 9.0372e-04
- acc: 0.9990 - val_loss: 3.0035e-04 - val_acc: 0.9997
Epoch 7/25
11082/11082 [==============================] - 7s 648us/step - loss: 5.6867e-04
- acc: 0.9993 - val_loss: 5.4142e-04 - val_acc: 0.9995
Epoch 8/25
11082/11082 [==============================] - 7s 637us/step - loss: 9.8440e-04
- acc: 0.9989 - val_loss: 0.0042 - val_acc: 0.9954
Epoch 9/25
11082/11082 [==============================] - 7s 607us/step - loss: 0.0018 -
acc: 0.9982 - val_loss: 0.0010 - val_acc: 0.9986
Epoch 10/25
11082/11082 [==============================] - 7s 606us/step - loss: 7.3318e-04
- acc: 0.9992 - val_loss: 2.7270e-04 - val_acc: 0.9997
Epoch 11/25
11082/11082 [==============================] - 7s 638us/step - loss: 9.4724e-04
- acc: 0.9990 - val_loss: 2.8525e-04 - val_acc: 0.9997
Epoch 12/25
11082/11082 [==============================] - 7s 616us/step - loss: 7.3715e-04
- acc: 0.9993 - val_loss: 2.7073e-04 - val_acc: 0.9997
Epoch 13/25
11082/11082 [==============================] - 7s 593us/step - loss: 7.9582e-04
- acc: 0.9992 - val_loss: 8.0186e-04 - val_acc: 0.9992
Epoch 14/25
11082/11082 [==============================] - 7s 603us/step - loss: 0.0020 -
acc: 0.9978 - val_loss: 0.0024 - val_acc: 0.9976
Epoch 15/25
11082/11082 [==============================] - 7s 604us/step - loss: 0.0013 -
acc: 0.9986 - val_loss: 2.7073e-04 - val_acc: 0.9997
Epoch 16/25
11082/11082 [==============================] - 7s 593us/step - loss: 0.0013 -
acc: 0.9987 - val_loss: 5.4142e-04 - val_acc: 0.9995
Epoch 17/25
11082/11082 [==============================] - 7s 597us/step - loss: 0.0013 -
acc: 0.9986 - val_loss: 5.5688e-04 - val_acc: 0.9995
Epoch 18/25
11082/11082 [==============================] - 9s 838us/step - loss: 7.0528e-04
- acc: 0.9993 - val_loss: 5.1038e-04 - val_acc: 0.9995
Epoch 19/25
11082/11082 [==============================] - 10s 914us/step - loss: 5.4590e-04
- acc: 0.9995 - val_loss: 5.2589e-04 - val_acc: 0.9995
Epoch 20/25
11082/11082 [==============================] - 10s 922us/step - loss: 0.0013 -
acc: 0.9986 - val_loss: 0.0016 - val_acc: 0.9984
Epoch 21/25
11082/11082 [==============================] - 10s 927us/step - loss: 4.9601e-04
- acc: 0.9995 - val_loss: 8.1213e-04 - val_acc: 0.9992
Epoch 22/25
```

```
11082/11082 [==============================] - 10s 937us/step - loss: 0.0012 -
acc: 0.9987 - val_loss: 5.4142e-04 - val_acc: 0.9995
Epoch 23/25
11082/11082 [==============================] - 10s 920us/step - loss: 0.0027 -
acc: 0.9972 - val_loss: 0.0046 - val_acc: 0.9954
Epoch 24/25
11082/11082 [==============================] - 10s 927us/step - loss: 0.0012 -
acc: 0.9987 - val_loss: 8.1213e-04 - val_acc: 0.9992
Epoch 25/25
11082/11082 [==============================] - 10s 914us/step - loss: 0.0013 -
acc: 0.9985 - val_loss: 5.4142e-04 - val_acc: 0.9995
```

```
       ␣
 ↪---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call␣
 ↪last)

        <ipython-input-78-b20607b0f11a> in <module>
         11 Y_out = model.predict(datum_test)
         12
    ---> 13 Y_out = ( Y_out_0 + Y_out_1 ) / 2;
         14
         15 print(Y_out)


        NameError: name 'Y_out_0' is not defined
```

[79]:
```
Y_out = model.predict(datum_test)

make_sub(Y_out,cut_off,"stacked_sub.csv")

print(Y_out)
```

```
[[1.]
 [0.]
 [1.]
 ...
 [0.]
 [0.]
 [1.]]
```

[ ]:
```

```