

LR_Models_VGG16

March 29, 2019

1 Library Import

```
In [1]: import os
import pandas as pd
import numpy as np
import pickle
import time
# Machine Learning Algorithms
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
# Metrics
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.metrics import precision_recall_fscore_support
from sklearn.model_selection import validation_curve, learning_curve
from joblib import dump
```

2 Magnification Identification

```
In [2]: train_path="A:\\Projects\\Major Project\\Extracted CNN Features\\VGG16\\train"
test_path="A:\\Projects\\Major Project\\Extracted CNN Features\\VGG16\\test"
```

```
In [3]: # Training Paths
X_train=np.load(train_path+"\\data_cnn_VGG16_train.npy")
Y_train=np.load(train_path+"\\data_mag_VGG16_train.npy")
# Cancer class
cancerclass_train=np.load(train_path+"\\data_cancerclass_VGG16_train.npy")
# Cancer type
cancertype_train=np.load(train_path+"\\data_cancertype_VGG16_train.npy")
# Testing Paths
X_test=np.load(test_path+"\\data_cnn_VGG16_test.npy")
Y_test=np.load(test_path+"\\data_mag_VGG16_test.npy")
# Cancer class
cancerclass_test=np.load(test_path+"\\data_cancerclass_VGG16_test.npy")
```

```

# Cancer type
cancertype_test=np.load(test_path+"\\data_cancertype_VGG16_test.npy")

In [4]: param_grid={'C': [.001, .01, .1, 1, 10]}

In [5]: start_time=time.clock()
        gs1=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,n

        start_time = time.clock()
        #Training of Model
        gs1.fit(X_train,Y_train)
        print(time.clock() - start_time, "seconds")

        print(gs1.best_score_)
        print(gs1.best_params_)

c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
    """Entry point for launching an IPython kernel.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
    after removing the cwd from sys.path.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
    FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
    "this warning.", FutureWarning)

625.3853572480001 seconds
0.8333860759493671
{'C': 0.001}

c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
    import sys

In [6]: clf=gs1.best_estimator_
        clf.fit(X_train,Y_train)
        print(clf.score(X_test,Y_test))

c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
    FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
    "this warning.", FutureWarning)

0.9006329113924051

In [7]: dump(clf,'models/LR/LR_Models_VGG16_Magnification.joblib')

```

```
Out[7]: ['models/LR/LR_Models_VGG16_Magnification.joblib']
```

```
In [8]: clf2=LogisticRegression(C=.001)
        clf2.fit(X_train,Y_train)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
"this warning.", FutureWarning)
```

```
Out[8]: LogisticRegression(C=0.001, class_weight=None, dual=False, fit_intercept=True,
        intercept_scaling=1, max_iter=100, multi_class='warn',
        n_jobs=None, penalty='l2', random_state=None, solver='warn',
        tol=0.0001, verbose=0, warm_start=False)
```

```
In [9]: print(clf2.score(X_test,Y_test))
```

```
0.9006329113924051
```

```
In [10]: pred=clf2.predict(X_test)
```

```
In [11]: con=confusion_matrix(Y_test,pred)
```

```
In [12]: print(con)
```

```
[[371  26   0   0]
 [ 17 347  28   0]
 [   3  22 340  29]
 [   0   1  31 365]]
```

```
In [13]: precision_score(Y_test, pred, average='micro')
```

```
Out[13]: 0.9006329113924051
```

```
In [14]: recall_score(Y_test, pred, average='micro')
```

```
Out[14]: 0.9006329113924051
```

```
In [15]: f1_score(Y_test, pred, average='micro')
```

```
Out[15]: 0.9006329113924051
```

```
In [16]: precision_recall_fscore_support(Y_test,pred)
```

```
Out[16]: (array([0.9488491 , 0.87626263, 0.85213033, 0.92639594]),
         array([0.93450882, 0.88520408, 0.86294416, 0.91939547]),
         array([0.94162437, 0.88071066, 0.85750315, 0.92288243]),
         array([397, 392, 394, 397], dtype=int64))
```

3 CancerClass Identification

```
In [17]: Y_train_40=[]
        X_train_40=[]

        Y_train_100=[]
        X_train_100=[]

        Y_train_200=[]
        X_train_200=[]

        Y_train_400=[]
        X_train_400=[]

        for i in range(0,len(Y_train)):
            if(Y_train[i]==40):
                Y_train_40.append(cancerclass_train[i])
                X_train_40.append(X_train[i])
            if(Y_train[i]==100):
                Y_train_100.append(cancerclass_train[i])
                X_train_100.append(X_train[i])
            if(Y_train[i]==200):
                Y_train_200.append(cancerclass_train[i])
                X_train_200.append(X_train[i])
            if(Y_train[i]==400):
                Y_train_400.append(cancerclass_train[i])
                X_train_400.append(X_train[i])

In [18]: X_train_40=np.array(X_train_40)
        X_train_100=np.array(X_train_100)
        X_train_200=np.array(X_train_200)
        X_train_400=np.array(X_train_400)
        Y_train_40=np.array(Y_train_40)
        Y_train_100=np.array(Y_train_100)
        Y_train_200=np.array(Y_train_200)
        Y_train_400=np.array(Y_train_400)
        print(Y_train_40.size)
```

1596

```
In [19]: Y_test_40=[]
        X_test_40=[]

        Y_test_100=[]
        X_test_100=[]

        Y_test_200=[]
        X_test_200=[]
```

```

Y_test_400=[]
X_test_400=[]

for i in range(0,len(Y_test)):
    if(Y_test[i]==40):
        Y_test_40.append(cancerclass_test[i])
        X_test_40.append(X_test[i])
    if(Y_test[i]==100):
        Y_test_100.append(cancerclass_test[i])
        X_test_100.append(X_test[i])
    if(Y_test[i]==200):
        Y_test_200.append(cancerclass_test[i])
        X_test_200.append(X_test[i])
    if(Y_test[i]==400):
        Y_test_400.append(cancerclass_test[i])
        X_test_400.append(X_test[i])

```

```

In [20]: X_test_40=np.array(X_test_40)
        X_test_100=np.array(X_test_100)
        X_test_200=np.array(X_test_200)
        X_test_400=np.array(X_test_400)
        Y_test_40=np.array(Y_test_40)
        Y_test_100=np.array(Y_test_100)
        Y_test_200=np.array(Y_test_200)
        Y_test_400=np.array(Y_test_400)

```

4 CancerClass Magnification classification 40

```

In [21]: param_grid={'C': [.001, .01, .1, 1, 10]}
        gs1=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,

        start_time = time.clock()
        #Training of Model
        gs1.fit(X_train_40,Y_train_40)
        print(time.clock() - start_time, "seconds")

        print(gs1.best_score_)
        print(gs1.best_params_)

```

```

c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
after removing the cwd from sys.path.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)

```

```

23.18713202899994 seconds
0.8646616541353384

```

```
{'C': 0.01}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
import sys
```

```
In [22]: clf3=gs1.best_estimator_
         clf3.fit(X_train_40,Y_train_40)
         clf3.score(X_test_40,Y_test_40)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1181:
FutureWarning)
```

```
Out[22]: 0.7884130982367759
```

```
In [23]: dump(clf3,'models/LR/LR_Models_VGG16_Magnification_40.joblib')
```

```
Out[23]: ['models/LR/LR_Models_VGG16_Magnification_40.joblib']
```

```
In [24]: clf=LogisticRegression(C=.01)
         clf.fit(X_train_40,Y_train_40)
         clf.score(X_test_40,Y_test_40)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1181:
FutureWarning)
```

```
Out[24]: 0.7884130982367759
```

```
In [25]: pred=clf.predict(X_test_40)
```

```
In [26]: con=confusion_matrix(Y_test_40,pred)
```

```
In [27]: print(con)
```

```
[[139  61]
 [ 23 174]]
```

```
In [28]: precision_score(Y_test_40,pred)
```

```
Out[28]: 0.8580246913580247
```

```
In [29]: recall_score(Y_test_40,pred)
```

```
Out[29]: 0.695
```

```
In [30]: f1_score(Y_test_40,pred)
```

```
Out[30]: 0.7679558011049723
```

```
In [31]: precision_recall_fscore_support(Y_test_40,pred)
```

```
Out[31]: (array([0.85802469, 0.74042553]),
          array([0.695      , 0.88324873]),
          array([0.7679558 , 0.80555556]),
          array([200, 197], dtype=int64))
```

5 CancerClass Magnification classification 100

```
In [32]: gs2=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,
```

```
    start_time = time.clock()
    #Training of Model
    gs2.fit(X_train_100,Y_train_100)
    print(time.clock() - start_time, "seconds")

    print(gs2.best_score_)
    print(gs2.best_params_)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
  This is separate from the ipykernel package so we can avoid doing imports until
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1181:
  FutureWarning)
```

```
25.976905048999924 seconds
0.8755186721991701
{'C': 0.01}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
```

```
In [33]: c=gs2.best_estimator_
        c.fit(X_train_100,Y_train_100)
        c.score(X_test_100,Y_test_100)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1181:
  FutureWarning)
```

```
Out[33]: 0.8673469387755102
```

```
In [34]: dump(c,'models/LR/LR_Models_VGG16_Magnification_100.joblib')
```

```
Out[34]: ['models/LR/LR_Models_VGG16_Magnification_100.joblib']
```

6 CancerClass Magnification classification 200

```
In [35]: gs3=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,
```

```
    start_time = time.clock()
    #Training of Model
    gs3.fit(X_train_200,Y_train_200)
    print(time.clock() - start_time, "seconds")
```

```
print(gs3.best_score_)
print(gs3.best_params_)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
This is separate from the ipykernel package so we can avoid doing imports until
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
```

```
24.21296059700012 seconds
0.87569573283859
{'C': 0.001}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
```

```
In [36]: c=gs3.best_estimator_
         c.fit(X_train_200,Y_train_200)
         c.score(X_test_200,Y_test_200)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
```

```
Out[36]: 0.8121827411167513
```

```
In [37]: dump(c, 'models/LR/LR_Models_VGG16_Magnification_200.joblib')
```

```
Out[37]: ['models/LR/LR_Models_VGG16_Magnification_200.joblib']
```

7 CancerClass Magnification classification 400

```
In [38]: gs4=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,1
```

```
start_time = time.clock()
#Training of Model
gs4.fit(X_train_400,Y_train_400)
print(time.clock() - start_time, "seconds")

print(gs4.best_score_)
print(gs4.best_params_)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
This is separate from the ipykernel package so we can avoid doing imports until
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
```



```
21.026892133000047 seconds
0.8697183098591549
{'C': 0.001}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
```

```
In [39]: c=gs4.best_estimator_  
         c.fit(X_train_400,Y_train_400)  
         c.score(X_test_400,Y_test_400)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m  
FutureWarning)
```

```
Out[39]: 0.7909319899244333
```

```
In [40]: dump(c, 'models/LR/LR_Models_VGG16_Magnification_400.joblib')
```

```
Out[40]: ['models/LR/LR_Models_VGG16_Magnification_400.joblib']
```

7.1 Benign Sub-Classification Using Cancer Classification

```
In [41]: Y_train_1=[]  
         X_train_1=[]  
  
         for i in range(0,len(Y_train)):  
             if(cancerclass_train[i]==1):  
                 Y_train_1.append(cancertype_train[i])  
                 X_train_1.append(X_train[i])  
  
         X_train_1=np.array(X_train_1)  
         Y_train_1=np.array(Y_train_1)  
         print(Y_train_1.size)  
  
         Y_test_1=[]  
         X_test_1=[]  
  
         for i in range(0,len(Y_test)):  
             if(cancerclass_test[i]==1):  
                 Y_test_1.append(cancertype_test[i])  
                 X_test_1.append(X_test[i])  
  
         X_test_1=np.array(X_test_1)  
         Y_test_1=np.array(Y_test_1)
```

1683

```

In [42]: classes=[11,12,13,14]

In [ ]:

In [43]: from sklearn.utils.class_weight import compute_class_weight

In [44]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_1)

In [45]: print(class_weight)

[1.66964286 0.51752768 1.67629482 1.14645777]

In [46]: print(np.unique(Y_train_1))

[11 12 13 14]

In [47]: print(len(X_train_1))

1683

In [48]: print(len(Y_test_1))

792

In [49]: d = dict(enumerate(class_weight, 1))

In [50]: print(d)

{1: 1.6696428571428572, 2: 0.5175276752767528, 3: 1.6762948207171315, 4: 1.146457765667575}

In [51]: d1={1:11,2:12,3:13,4:14}

In [52]: d=dict((d1[key], value) for (key, value) in d.items())

In [53]: d

Out[53]: {11: 1.6696428571428572,
          12: 0.5175276752767528,
          13: 1.6762948207171315,
          14: 1.146457765667575}

In [54]: gs3=GridSearchCV(LogisticRegression(class_weight=d),param_grid=param_grid,scoring="acc

    start_time = time.clock()
    #Training of Model
    gs3.fit(X_train_1,Y_train_1)
    print(time.clock() - start_time, "seconds")

    print(gs3.best_score_)
    print(gs3.best_params_)

```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
This is separate from the ipykernel package so we can avoid doing imports until
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\model_se
DeprecationWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
"this warning.", FutureWarning)
```

```
93.51835731599999 seconds
0.6874628639334521
{'C': 0.01}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
```

```
In [55]: clf4=gs3.best_estimator_
         clf4.fit(X_train_1,Y_train_1)
         print(clf4.score(X_test_1,Y_test_1))
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
"this warning.", FutureWarning)
```

```
0.43434343434343436
```

```
In [56]: dump(clf4,'models/LR/LR_Models_VGG16_CancerType_Benign.joblib')
```

```
Out[56]: ['models/LR/LR_Models_VGG16_CancerType_Benign.joblib']
```

```
In [57]: pred=clf4.predict(X_test_1)
```

```
In [58]: precision_recall_fscore_support(Y_test_1,pred)
```

```
Out[58]: (array([0.67045455, 0.31974249, 0.57608696, 0.56849315]),
          array([0.30890052, 0.745      , 0.26368159, 0.415      ]),
          array([0.42293907, 0.44744745, 0.36177474, 0.47976879]),
          array([191, 200, 201, 200], dtype=int64))
```

```
In [59]: confusion_matrix(Y_test_1,pred)
```

```
Out[59]: array([[ 59,  86,  13,  33],
                [ 11, 149,  20,  20],
                [  6, 132,  53,  10],
                [ 12,  99,   6,  83]], dtype=int64)
```

7.2 Malignant Sub-Classification Using Cancer Classification

```
In [60]: Y_train_2=[]
        X_train_2=[]

        for i in range(0,len(Y_train)):
            if(cancerclass_train[i]==2):
                Y_train_2.append(cancertype_train[i])
                X_train_2.append(X_train[i])

        X_train_2=np.array(X_train_2)
        Y_train_2=np.array(Y_train_2)
        print(Y_train_2.size)

        Y_test_2=[]
        X_test_2=[]

        for i in range(0,len(Y_test)):
            if(cancerclass_test[i]==2):
                Y_test_2.append(cancertype_test[i])
                X_test_2.append(X_test[i])

        X_test_2=np.array(X_test_2)
        Y_test_2=np.array(Y_test_2)

4637

In [61]: classes=[21,22,23,24]

In [62]: from sklearn.utils.class_weight import compute_class_weight

In [63]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_2)

In [64]: print(class_weight)

[0.35669231 2.72764706 1.96150592 3.12466307]

In [65]: print(np.unique(Y_train_2))

[21 22 23 24]

In [66]: print(len(X_train_2))

4637

In [67]: print(len(Y_test_2))
```

788

```
In [68]: d = dict(enumerate(class_weight, 1))
```

```
In [69]: print(d)
```

```
{1: 0.3566923076923077, 2: 2.7276470588235293, 3: 1.9615059221658206, 4: 3.1246630727762805}
```

```
In [70]: d1={1:21,2:22,3:23,4:24}
```

```
In [71]: d=dict((d1[key], value) for (key, value) in d.items())
```

```
In [72]: d
```

```
Out[72]: {21: 0.3566923076923077,
          22: 2.7276470588235293,
          23: 1.9615059221658206,
          24: 3.1246630727762805}
```

```
In [73]: gs3=GridSearchCV(LogisticRegression(class_weight=d),param_grid=param_grid,scoring="acc
```

```
    start_time = time.clock()
    #Training of Model
    gs3.fit(X_train_2,Y_train_2)
    print(time.clock() - start_time, "seconds")

    print(gs3.best_score_)
    print(gs3.best_params_)
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
  This is separate from the ipykernel package so we can avoid doing imports until
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1182:
  FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1182:
  "this warning.", FutureWarning)
```

```
506.596508156 seconds
0.6911796420099202
{'C': 0.01}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
```

```
In [74]: clf4=gs3.best_estimator_
         clf4.fit(X_train_2,Y_train_2)
         print(clf4.score(X_test_2,Y_test_2))
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_m
"this warning.", FutureWarning)
```

```
0.383248730964467
```

```
In [75]: dump(clf4, 'models/LR/LR_Models_VGG16_CancerType_Malignant.joblib')
```

```
Out[75]: ['models/LR/LR_Models_VGG16_CancerType_Malignant.joblib']
```

```
In [76]: pred=clf4.predict(X_test_2)
```

```
In [77]: precision_recall_fscore_support(Y_test_2,pred)
```

```
Out[77]: (array([0.33047945, 0.63636364, 0.48039216, 0.53191489]),
          array([0.965      , 0.175      , 0.245      , 0.13297872]),
          array([0.49234694, 0.2745098 , 0.32450331, 0.21276596]),
          array([200, 200, 200, 188], dtype=int64))
```

```
In [78]: confusion_matrix(Y_test_2,pred)
```

```
Out[78]: array([[193,  1,  0,  6],
                [153, 35, 10,  2],
                [124, 13, 49, 14],
                [114,  6, 43, 25]], dtype=int64)
```