

# LR\_Models\_ResNet50

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\\ResNet50\\train"
test_path="A:\\Projects\\Major Project\\Extracted CNN Features\\ResNet50\\test"
```

```
In [3]: # Training Paths
X_train=np.load(train_path+"\\data_cnn_ResNet50_train.npy")
Y_train=np.load(train_path+"\\data_mag_ResNet50_train.npy")
# Cancer class
cancerclass_train=np.load(train_path+"\\data_cancerclass_ResNet50_train.npy")
# Cancer type
cancertype_train=np.load(train_path+"\\data_cancertype_ResNet50_train.npy")
# Testing Paths
X_test=np.load(test_path+"\\data_cnn_ResNet50_test.npy")
Y_test=np.load(test_path+"\\data_mag_ResNet50_test.npy")
# Cancer class
cancerclass_test=np.load(test_path+"\\data_cancerclass_ResNet50_test.npy")
```

```

# Cancer type
cancertype_test=np.load(test_path+"\\data_cancertype_ResNet50_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)

493.199970677 seconds
0.888132911392405
{'C': 0.01}

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)

0.9322784810126582

In [7]: dump(clf,'models/LR/LR_Models_ResNet50_Magnification.joblib')

```

```
Out[7]: ['models/LR/LR_Models_ResNet50_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))
```

```
0.9082278481012658
```

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

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

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

```
[[385  12   0   0]
 [ 18 357  17   0]
 [   6  23 334  31]
 [   0   2  36 359]]
```

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

```
Out[13]: 0.9082278481012658
```

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

```
Out[14]: 0.9082278481012658
```

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

```
Out[15]: 0.9082278481012658
```

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

```
Out[16]: (array([0.94132029, 0.90609137, 0.8630491 , 0.92051282]),
          array([0.9697733 , 0.91071429, 0.84771574, 0.90428212]),
          array([0.95533499, 0.90839695, 0.8553137 , 0.91232529]),
          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)

```

```

17.025031709000018 seconds
0.8765664160401002

```

```
{'C': 0.1}
```

```
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]: 0.8312342569269522
```

```
In [23]: dump(clf3,'models/LR/LR_Models_ResNet50_Magnification_40.joblib')
```

```
Out[23]: ['models/LR/LR_Models_ResNet50_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]: 0.8287153652392947
```

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

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

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

```
[[142  58]
 [ 10 187]]
```

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

```
Out[28]: 0.9342105263157895
```

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

```
Out[29]: 0.71
```

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

```
Out[30]: 0.8068181818181819
```

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

```
Out[31]: (array([0.93421053, 0.76326531]),
          array([0.71       , 0.94923858]),
          array([0.80681818, 0.84615385]),
          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)
```

```
20.07028837200005 seconds
0.8695909899229401
{'C': 0.1}
```

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

```
In [33]: c=LogisticRegression(C=.01)
    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]: 0.8520408163265306
```

```
In [34]: dump(c,'models/LR/LR_Models_ResNet50_Magnification_100.joblib')
```

```
Out[34]: ['models/LR/LR_Models_ResNet50_Magnification_100.joblib']
```

## 6 CancerClass Magnification classification 200

```
In [80]: 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)
```

```
16.266127203999986 seconds
0.891156462585034
{'C': 1}
```

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

```
In [81]: 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[81]: 0.8401015228426396
```

```
In [82]: dump(c, 'models/LR/LR_Models_ResNet50_Magnification_200.joblib')
```

```
Out[82]: ['models/LR/LR_Models_ResNet50_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)
```



```
17.722524070000077 seconds
0.893661971830986
{'C': 0.01}
```

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

```
In [39]: c=LogisticRegression(C=.001)
         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]: 0.801007556675063
```

```
In [40]: dump(c, 'models/LR/LR_Models_ResNet50_Magnification_400.joblib')
```

```
Out[40]: ['models/LR/LR_Models_ResNet50_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)
```

```
79.02528817500001 seconds
0.7540106951871658
{'C': 10}
```

```
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)
```

```
0.48358585858585856
```

```
In [56]: dump(clf4, 'models/LR/LR_Models_ResNet50_CancerType_Benign.joblib')
```

```
Out[56]: ['models/LR/LR_Models_ResNet50_CancerType_Benign.joblib']
```

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

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

```
Out[58]: (array([0.80851064, 0.33688699, 0.52380952, 0.75806452]),
          array([0.39790576, 0.79          , 0.27363184, 0.47          ]),
          array([0.53333333, 0.47234679, 0.35947712, 0.58024691]),
          array([191, 200, 201, 200], dtype=int64))
```

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

```
Out[59]: array([[ 76,  80,  23,  12],
                [ 10, 158,  20,  12],
                [  8, 132,  55,   6],
                [  0,  99,   7,  94]], 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:1186:
  FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_model\logistic.py:1186:
  "this warning.", FutureWarning)
```

```
342.09768086999986 seconds
0.7455251240025879
{'C': 0.1}
```

```
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)
```

```
0.4149746192893401
```

```
In [75]: dump(clf4, 'models/LR/LR_Models_ResNet50_CancerType_Malignant.joblib')
```

```
Out[75]: ['models/LR/LR_Models_ResNet50_CancerType_Malignant.joblib']
```

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

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

```
Out[77]: (array([0.35218978, 0.64788732, 0.52          , 0.52272727]),
          array([0.965          , 0.23          , 0.325          , 0.12234043]),
          array([0.51604278, 0.33948339, 0.4          , 0.19827586]),
          array([200, 200, 200, 188], dtype=int64))
```

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

```
Out[78]: array([[193,  5,  1,  1],
                [134, 46, 20,  0],
                [ 98, 17, 65, 20],
                [123,  3, 39, 23]], dtype=int64)
```