LR_Models_Xception

March 28, 2019

1 Library Import

```
In [6]: 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
        #Dumping Model Library
        from joblib import dump,load
```

2 Magnification Identification

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

```
In [3]: # Training Paths
        X_train=np.load(train_path+"\\data_cnn_Xception_train.npy")
        Y_train=np.load(train_path+"\\data_mag_Xception_train.npy")
        # Cancer class
        cancerclass_train=np.load(train_path+"\\data_cancerclass_Xception_train.npy")
        # Cancer type
        cancertype_train=np.load(train_path+"\\data_cancertype_Xception_train.npy")
        # Testing Paths
        X_test=np.load(test_path+"\\data_cnn_Xception_test.npy")
        Y_test=np.load(test_path+"\\data_mag_Xception_test.npy")
        # Cancer class
        cancerclass_test=np.load(test_path+"\\data_cancerclass_Xception_test.npy")
        # Cancer type
        cancertype_test=np.load(test_path+"\\data_cancertype_Xception_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_launche
  """Entry point for launching an IPython kernel.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launche
  after removing the cwd from sys.path.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_mo
  FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_mo
  "this warning.", FutureWarning)


426.280630189 seconds
0.8299050632911392
{'C': 0.1}


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


0.8759493670886076
```

```
In [7]: clf2=LogisticRegression(C=.001)
        clf2.fit(X_train,Y_train)

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

```
Out[7]: 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 [8]: print(clf2.score(X_test,Y_test))

0.7879746835443038
```

```
In [9]: pred=clf2.predict(X_test)

In [10]: con=confusion_matrix(Y_test,pred)

In [11]: print(con)

[[370  23   1   3]
 [ 82 277  24   9]
 [ 15  62 251  66]
 [  4   3  43 347]]
```

```
In [12]: precision_score(Y_test, pred, average='micro')

Out[12]: 0.7879746835443038

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

Out[13]: 0.7879746835443038

In [14]: f1_score(Y_test, pred, average='micro')

Out[14]: 0.7879746835443038

In [15]: precision_recall_fscore_support(Y_test,pred)

Out[15]: (array([0.78556263, 0.75890411, 0.78683386, 0.81647059]),
          array([0.93198992, 0.70663265, 0.63705584, 0.87405542]),
          array([0.85253456, 0.7318362 , 0.70406732, 0.84428224]),
          array([397, 392, 394, 397], dtype=int64))
```

3 CancerClass Identification

```
In [9]: 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 [10]: 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 [11]: 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 [12]: 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 [20]: 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_launche
  after removing the cwd from sys.path.
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_mo
  FutureWarning)


18.780343469 seconds
0.8677944862155389
```

```
{'C': 1}
```

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

```
In [21]: 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_m
  FutureWarning)
```

```
Out[21]: 0.8236775818639799
```

```
In [22]: 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_m
  FutureWarning)
```

```
Out[22]: 0.743073047858942
```

```
In [23]: pred=clf.predict(X_test_40)
```

```
In [24]: con=confusion_matrix(Y_test_40,pred)
```

```
In [25]: print(con)
```

```
[[109  91]
 [ 11 186]]
```

```
In [26]: precision_score(Y_test_40,pred)
```

```
Out[26]: 0.9083333333333333
```

```
In [27]: recall_score(Y_test_40,pred)
```

```
Out[27]: 0.545
```

```
In [28]: f1_score(Y_test_40,pred)
```

```
Out[28]: 0.68125
```

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

```
Out[29]: (array([0.90833333, 0.67148014]),
          array([0.545     , 0.94416244]),
          array([0.68125   , 0.78481013]),
          array([200, 197], dtype=int64))
```

5 CancerClass Magnification classification 100

```
In [30]: gs2=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,r

         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_launche
  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_mo
  FutureWarning)
```

```
20.942412992000015 seconds
0.8571428571428571
{'C': 0.1}
```

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

```
In [31]: 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_mo
  FutureWarning)
```

```
Out[31]: 0.7397959183673469
```

6 CancerClass Magnification classification 200

```
In [32]: gs3=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,r

         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_launche
  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_mo
  FutureWarning)
```

```
17.166663098000015 seconds
0.8713667285095856
{'C': 0.1}
```

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

```
In [33]: c=LogisticRegression(C=.001)
         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_mo
  FutureWarning)
```

```
Out[33]: 0.6294416243654822
```

7 CancerClass Magnification classification 400

```
In [34]: gs4=GridSearchCV(LogisticRegression(),param_grid=param_grid,scoring="accuracy",cv=10,r

         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_launche
  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_mo
  FutureWarning)
```

```
14.505052512999953 seconds
0.8570422535211267
{'C': 10}
```

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

```
In [35]: 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_mo
  FutureWarning)
```

```
Out[35]: 0.6120906801007556
```

7.1 Benign Sub-Classification Using Cancer Classification

```
In [20]: 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 [37]: classes=[11,12,13,14]

In [ ]:

In [38]: from sklearn.utils.class_weight import compute_class_weight

In [39]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_1
In [40]: print(class_weight)
```

```
[1.66964286 0.51752768 1.67629482 1.14645777]
```

```
In [41]: print(np.unique(Y_train_1))
```

```
[11 12 13 14]
```

```
In [42]: print(len(X_train_1))
```

```
1683
```

```
In [43]: print(len(Y_test_1))
```

```
792
```

```
In [44]: d = dict(enumerate(class_weight, 1))
```

```
In [45]: print(d)
```

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

```
In [46]: d1={1:11,2:12,3:13,4:14}
```

```
In [47]: d=dict((d1[key], value) for (key, value) in d.items())
```

```
In [48]: d
```

```
Out[48]: {11: 1.6696428571428572,
          12: 0.5175276752767528,
          13: 1.6762948207171315,
          14: 1.146457765667575}
```

```
In [49]: gs3=GridSearchCV(LogisticRegression(class_weight=d),param_grid=param_grid,scoring="ac

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

```
79.43079291100003 seconds
0.708853238265003
{'C': 1}
```

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

```
In [50]: 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_mo
  FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_mo
  "this warning.", FutureWarning)
```

```
0.5151515151515151
```

```
In [51]: pred=clf4.predict(X_test_1)
```

```
In [52]: precision_recall_fscore_support(Y_test_1,pred)
```

```
Out[52]: (array([0.74509804, 0.38059701, 0.51020408, 0.67894737]),
          array([0.39790576, 0.765     , 0.24875622, 0.645     ]),
          array([0.51877133, 0.50830565, 0.33444816, 0.66153846]),
          array([191, 200, 201, 200], dtype=int64))
```

```
In [53]: confusion_matrix(Y_test_1,pred)
```

```
Out[53]: array([[ 76,  60,  22,  33],
                [ 10, 153,  22,  15],
                [  9, 129,  50,  13],
                [  7,  60,   4, 129]], dtype=int64)
```

7.2 Malignant Sub-Classification Using Cancer Classification

```
In [21]: 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 [55]: classes=[21,22,23,24]
```

```
In [56]: from sklearn.utils.class_weight import compute_class_weight
```

```
In [57]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_
```

```
In [58]: print(class_weight)
```

```
[0.35669231 2.72764706 1.96150592 3.12466307]
```

```
In [59]: print(np.unique(Y_train_2))
```

```
[21 22 23 24]
```

```
In [60]: print(len(X_train_2))
```

```
4637
```

```
In [61]: print(len(Y_test_2))
```

```
788
```

```
In [62]: d = dict(enumerate(class_weight, 1))
```

```
In [63]: print(d)
```

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

```
In [64]: d1={1:21,2:22,3:23,4:24}
```

```
In [65]: d=dict((d1[key], value) for (key, value) in d.items())
```

```
In [66]: d
```

```
Out[66]: {21: 0.3566923076923077,
          22: 2.7276470588235293,
          23: 1.9615059221658206,
          24: 3.1246630727762805}
```

```
In [67]: gs3=GridSearchCV(LogisticRegression(class_weight=d),param_grid=param_grid,scoring="ac

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

```
311.47988350599996 seconds
0.7064912659046797
{'C': 1}
```

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

```
In [68]: 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_mo
  FutureWarning)
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\linear_mo
  "this warning.", FutureWarning)


0.39720812182741116
```

```
In [69]: pred=clf4.predict(X_test_2)
```

```
In [70]: precision_recall_fscore_support(Y_test_2,pred)
```

```
Out[70]: (array([0.31443299, 0.75409836, 0.52252252, 0.76470588]),
          array([0.915     , 0.23      , 0.29      , 0.13829787]),
          array([0.46803069, 0.35249042, 0.37299035, 0.23423423]),
          array([200, 200, 200, 188], dtype=int64))
```

```
In [71]: confusion_matrix(Y_test_2,pred)
```

```
Out[71]: array([[183,   5,   8,   4],
                [137,  46,  17,   0],
                [133,   5,  58,   4],
                [129,   5,  28,  26]], dtype=int64)
```

8 Dumping Models

```
In [5]: clf=LogisticRegression(C=0.1)
        clf.fit(X_train,Y_train)
        clf.score(X_test,Y_test)
```

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


Out[5]: 0.8759493670886076
```

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

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

```
In [ ]:
```

```
In [13]: clf=LogisticRegression(C=1)
         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_mo
  FutureWarning)
```

```
Out[13]: 0.8236775818639799

In [14]: dump(clf,'models/LR/LR_Models_Xception_Magnification_40.joblib')

Out[14]: ['models/LR/LR_Models_Xception_Magnification_40.joblib']

In [ ]:

In [15]: clf=LogisticRegression(C=0.1)
         clf.fit(X_train_100,Y_train_100)
         clf.score(X_test_100,Y_test_100)
```

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

```
Out[15]: 0.7959183673469388

In [16]: dump(clf,'models/LR/LR_Models_Xception_Magnification_100.joblib')

Out[16]: ['models/LR/LR_Models_Xception_Magnification_100.joblib']

In [ ]:

In [27]: clf=LogisticRegression(C=0.1)
         clf.fit(X_train_200,Y_train_200)
         clf.score(X_test_200,Y_test_200)
```

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

```
Out[27]: 0.7944162436548223

In [28]: dump(clf,'models/LR/LR_Models_Xception_Magnification_200.joblib')

Out[28]: ['models/LR/LR_Models_Xception_Magnification_200.joblib']

In [ ]:

In [18]: clf=LogisticRegression(C=10)
         clf.fit(X_train_400,Y_train_400)
         clf.score(X_test_400,Y_test_400)
```

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

```
Out[18]: 0.8312342569269522
```

```
In [19]: dump(clf,'models/LR/LR_Models_Xception_Magnification_400.joblib')
```

```
Out[19]: ['models/LR/LR_Models_Xception_Magnification_400.joblib']
```

```
In [ ]:
```

```
In [25]: clf=LogisticRegression(C=1)
         clf.fit(X_train_1,Y_train_1)
         clf.score(X_test_1,Y_test_1)
```

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

```
Out[25]: 0.5050505050505051
```

```
In [26]: dump(clf,'models/LR/LR_Models_Xception_CancerType_Benign.joblib')
```

```
Out[26]: ['models/LR/LR_Models_Xception_CancerType_Benign.joblib']
```

```
In [ ]:
```

```
In [22]: clf=LogisticRegression(C=1)
         clf.fit(X_train_2,Y_train_2)
         clf.score(X_test_2,Y_test_2)
```

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

```
Out[22]: 0.383248730964467
```

```
In [24]: dump(clf,'models/LR/LR_Models_Xception_CancerType_Malignant.joblib')
```

```
Out[24]: ['models/LR/LR_Models_Xception_CancerType_Malignant.joblib']
```

```
In [ ]:
```

```
In [ ]:
```