

SVM_Models_ResNet50

March 29, 2019

1 Import Library

```
In [1]: import os
import pandas as pd
import numpy as np
import pickle
import time
# Machine Learning Algorithms
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from sklearn.metrics import confusion_matrix
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import GridSearchCV
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
import matplotlib.pyplot as plt
from sklearn.metrics import precision_recall_fscore_support
from sklearn.svm import SVC
from joblib import dump, load
```

2 Loading Paths

```
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]: param_range = [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]

param_grid = [{'svc__C': param_range,
               'svc__kernel': ['linear']},
               {'svc__C': param_range,
               'svc__gamma': ['auto'],
               'svc__kernel': ['rbf']}]
pipe_svc = make_pipeline(SVC(random_state=1))

In [4]: # 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")

```

2.1 Magnification classification

```
In [5]: start_time = time.clock()
```

```

param_grid = [{'svc__C': param_range,
               'svc__kernel': ['linear']},
               {'svc__C': param_range,
               'svc__gamma': ['auto'],
               'svc__kernel': ['rbf']}]

```

```

gs = GridSearchCV(estimator=pipe_svc,
                  param_grid=param_grid,
                  scoring='accuracy',
                  cv=10,
                  n_jobs=-1)
gs = gs.fit(X_train, Y_train)
print(gs.best_score_)
print(gs.best_params_)
print(time.clock() - start_time, "seconds")

```

```

clf = gs.best_estimator_
clf.fit(X_train, Y_train)
print('Test accuracy: %.3f' % clf.score(X_test, Y_test))

```

```

c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
"""Entry point for launching an IPython kernel.

```

```

0.8931962025316456
{'svc__C': 0.001, 'svc__kernel': 'linear'}
4356.639901603 seconds

```

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

Test accuracy: 0.925

```
In [6]: clf = gs.best_estimator_  
        clf.fit(X_train, Y_train)  
        print('Test accuracy: %.3f' % clf.score(X_test, Y_test))  
        print(time.clock() - start_time, "seconds")
```

Test accuracy: 0.925
4436.813218459 seconds

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

```
In [7]: dump(clf, 'models/SVM/SVM_Models_ResNet50_Magnification.joblib')
```

```
Out[7]: ['models/SVM/SVM_Models_ResNet50_Magnification.joblib']
```

```
In [8]: pred=clf.predict(X_test)
```

```
In [9]: con=confusion_matrix(Y_test, pred)
```

```
In [10]: print(con)
```

```
[[387  10   0   0]  
 [  5 366  21   0]  
 [  2  28 350  14]  
 [  0   0  38 359]]
```

```
In [11]: precision_recall_fscore_support(Y_test, pred)
```

```
Out[11]: (array([0.9822335 , 0.90594059, 0.85574572, 0.96246649]),  
          array([0.97481108, 0.93367347, 0.88832487, 0.90428212]),  
          array([0.97850822, 0.91959799, 0.87173101, 0.93246753]),  
          array([397, 392, 394, 397], dtype=int64))
```

2.2 CancerClass Magnification Classification

```
In [12]: 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])

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)

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])

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)

```

1596

2.3 CancerClass Magnification Classification-40

```

In [13]: start_time = time.clock()
        pipe_svc = make_pipeline(SVC(random_state=1))

        param_range = [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]

        param_grid = [{'svc__C': param_range,
                        'svc__kernel': ['linear']},
                       {'svc__C': param_range,
                        'svc__gamma': ['auto'],
                        'svc__kernel': ['rbf']}]

        gs = GridSearchCV(estimator=pipe_svc,
                           param_grid=param_grid,
                           scoring='accuracy',
                           cv=10,
                           n_jobs=-1)
        gs = gs.fit(X_train_40, Y_train_40)
        print(gs.best_score_)
        print(gs.best_params_)
        print(time.clock() - start_time, "seconds")

        clf = gs.best_estimator_
        clf.fit(X_train_40, Y_train_40)
        print('Test accuracy: %.3f' % clf.score(X_test_40, Y_test_40))

```

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

```
"""Entry point for launching an IPython kernel.
```

```
0.87468671679198
```

```
{'svc__C': 0.01, 'svc__kernel': 'linear'}
```

```
207.40240035199986 seconds
```

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

```
Test accuracy: 0.864
```

```
In [14]: dump(clf, 'models/SVM/SVM_Models_ResNet50_Magnification_40.joblib')
```

```
Out[14]: ['models/SVM/SVM_Models_ResNet50_Magnification_40.joblib']
```

2.4 CancerClass Magnification Classification-100

```
In [15]: start_time = time.clock()
         pipe_svc = make_pipeline(SVC(random_state=1))
```

```
param_range = [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]
```

```
param_grid = [{'svc__C': param_range,
               'svc__kernel': ['linear']},
               {'svc__C': param_range,
               'svc__gamma': ['auto'],
               'svc__kernel': ['rbf']}]
```

```
gs = GridSearchCV(estimator=pipe_svc,
                  param_grid=param_grid,
                  scoring='accuracy',
                  cv=10,
                  n_jobs=-1)
```

```
gs = gs.fit(X_train_100, Y_train_100)
```

```
print(gs.best_score_)
```

```
print(gs.best_params_)
```

```
print(time.clock() - start_time, "seconds")
```

```
clf = gs.best_estimator_
```

```
clf.fit(X_train_100, Y_train_100)
```

```
print('Test accuracy: %.3f' % clf.score(X_test_100, Y_test_100))
```

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

```
"""Entry point for launching an IPython kernel.
```

```
0.8737403675163011
{'svc__C': 0.01, 'svc__kernel': 'linear'}
232.72287316100028 seconds
```

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

Test accuracy: 0.860

```
In [16]: print(gs.best_estimator_)
```

```
Pipeline(memory=None,
      steps=[('svc', SVC(C=0.01, cache_size=200, class_weight=None, coef0=0.0,
        decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
        kernel='linear', max_iter=-1, probability=False, random_state=1,
        shrinking=True, tol=0.001, verbose=False))])
```

```
In [17]: dump(clf, 'models/SVM/SVM_Models_ResNet50_Magnification_100.joblib')
```

```
Out[17]: ['models/SVM/SVM_Models_ResNet50_Magnification_100.joblib']
```

2.5 CancerClass Magnification Classification-200

```
In [18]: tart_time = time.clock()
         pipe_svc = make_pipeline(SVC(random_state=1))

         param_range = [0.001, 0.01, 0.1, 1.0, 10.0, 100.0]

         param_grid = [{'svc__C': param_range,
                        'svc__kernel': ['linear']},
                        {'svc__C': param_range,
                        'svc__gamma': ['auto'],
                        'svc__kernel': ['rbf']}]

         gs = GridSearchCV(estimator=pipe_svc,
                           param_grid=param_grid,
                           scoring='accuracy',
                           cv=10,
                           n_jobs=-1)
         gs = gs.fit(X_train_200, Y_train_200)
         print(gs.best_score_)
         print(gs.best_params_)
         print(time.clock() - start_time, "seconds")

         clf = gs.best_estimator_
         clf.fit(X_train_200, Y_train_200)
         print('Test accuracy: %.3f' % clf.score(X_test_200, Y_test_200))
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
    """Entry point for launching an IPython kernel.
```

```
0.8979591836734694
{'svc__C': 10.0, 'svc__gamma': 'auto', 'svc__kernel': 'rbf'}
434.18689976600035 seconds
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
    """Entry point for launching an IPython kernel.
```

```
Test accuracy: 0.843
```

```
In [19]: dump(clf, 'models/SVM/SVM_Models_ResNet50_Magnification_200.joblib')
```

```
Out[19]: ['models/SVM/SVM_Models_ResNet50_Magnification_200.joblib']
```

2.6 CancerClass Magnification Classification-400

```
In [20]: tart_time = time.clock()
         pipe_svc = make_pipeline(SVC(random_state=1))
```

```
gs = GridSearchCV(estimator=pipe_svc,
                  param_grid=param_grid,
                  scoring='accuracy',
                  cv=10,
                  n_jobs=-1)
gs = gs.fit(X_train_200, Y_train_200)
print(gs.best_score_)
print(gs.best_params_)
print(time.clock() - start_time, "seconds")
```

```
clf = gs.best_estimator_
clf.fit(X_train_200, Y_train_200)
print('Test accuracy: %.3f' % clf.score(X_test_400, Y_test_400))
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher.py:1:
    """Entry point for launching an IPython kernel.
```

```
0.8979591836734694
{'svc__C': 10.0, 'svc__gamma': 'auto', 'svc__kernel': 'rbf'}
635.6826773900002 seconds
```



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

Test accuracy: 0.811

```
In [21]: dump(clf, 'models/SVM/SVM_Models_ResNet50_Magnification_400.joblib')
```

```
Out[21]: ['models/SVM/SVM_Models_ResNet50_Magnification_400.joblib']
```

2.7 Benign Sub-Classification Using Cancer Classification

```
In [22]: 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 [ ]:
```

```
In [23]: classes=[11,12,13,14]
```

```
In [ ]:
```

```
In [24]: from sklearn.utils.class_weight import compute_class_weight
```

```
In [25]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_1)
```

```
In [26]: print(class_weight)
```

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

```
In [27]: print(np.unique(Y_train_1))
```

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

```
In [28]: print(len(X_train_1))
```

```
1683
```

```
In [29]: print(len(Y_test_1))
```

```
792
```

```
In [30]: d = dict(enumerate(class_weight, 1))
```

```
In [31]: print(d)
```

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

```
In [32]: d1={1:11,2:12,3:13,4:14}
```

```
In [33]: d=dict((d1[key], value) for (key, value) in d.items())
```

```
In [34]: d
```

```
Out[34]: {11: 1.6696428571428572,  
          12: 0.5175276752767528,  
          13: 1.6762948207171315,  
          14: 1.146457765667575}
```

```
In [ ]:
```

```
In [ ]:
```

```
In [35]: pipe_svc = make_pipeline(SVC(random_state=1,class_weight=d))  
        gs3=GridSearchCV(estimator=pipe_svc,  
                          param_grid=param_grid,  
                          scoring='accuracy',  
                          cv=10,  
                          n_jobs=-1)  
        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
import sys
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\model_se
DeprecationWarning)
```

```
478.025977797900034 seconds
0.8015448603683898
{'svc__C': 100.0, 'svc__gamma': 'auto', 'svc__kernel': 'rbf'}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
# Remove the CWD from sys.path while we load stuff.
```

```
In [36]: clf4=gs3.best_estimator_
         clf4.fit(X_train_1,Y_train_1)
         print(clf4.score(X_test_1,Y_test_1))
```

```
0.45075757575757575
```

```
In [37]: dump(clf4,'models/SVM/SVM_Models_ResNet50_CancerType_Benign.joblib')
```

```
Out[37]: ['models/SVM/SVM_Models_ResNet50_CancerType_Benign.joblib']
```

```
In [ ]:
```

2.8 Malignant Sub-Classification Using Cancer Classification

```
In [38]: 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 [39]: classes=[21,22,23,24]
```

```
In [40]: from sklearn.utils.class_weight import compute_class_weight
```

```
In [41]: class_weight=compute_class_weight(class_weight='balanced', classes=classes,y=Y_train_2)
```

```
In [42]: print(class_weight)
```

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

```
In [43]: print(np.unique(Y_train_2))
```

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

```
In [44]: print(len(X_train_2))
```

4637

```
In [45]: print(len(Y_test_2))
```

788

```
In [46]: d = dict(enumerate(class_weight, 1))
```

```
In [47]: print(d)
```

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

```
In [48]: d1={1:21,2:22,3:23,4:24}
```

```
In [49]: d=dict((d1[key], value) for (key, value) in d.items())
```

```
In [50]: d
```

```
Out[50]: {21: 0.3566923076923077,
          22: 2.7276470588235293,
          23: 1.9615059221658206,
          24: 3.1246630727762805}
```

```
In [51]: pipe_svc = make_pipeline(SVC(random_state=1,class_weight=d))
        gs3=GridSearchCV(estimator=pipe_svc,
                          param_grid=param_grid,
                          scoring='accuracy',
                          cv=10,
                          n_jobs=-1)
        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
import sys
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\sklearn\model_se
DeprecationWarning)
```

```
3620.7466244029993 seconds
```

```
0.787362518869959
```

```
{'svc__C': 100.0, 'svc__gamma': 'auto', 'svc__kernel': 'rbf'}
```

```
c:\users\karan gupta\appdata\local\programs\python\python37\lib\site-packages\ipykernel_launcher
# Remove the CWD from sys.path while we load stuff.
```

```
In [52]: clf4=gs3.best_estimator_
        clf4.fit(X_train_2,Y_train_2)
        print(clf4.score(X_test_2,Y_test_2))
```

```
0.3946700507614213
```

```
In [53]: pred=clf4.predict(X_test_2)
```

```
In [54]: precision_recall_fscore_support(Y_test_2,pred)
```

```
Out [54]: (array([0.33732877, 0.72222222, 0.51515152, 0.38888889]),
          array([0.985      , 0.195      , 0.34      , 0.03723404]),
          array([0.50255102, 0.30708661, 0.40963855, 0.06796117]),
          array([200, 200, 200, 188], dtype=int64))
```

```
In [55]: confusion_matrix(Y_test_2,pred)
```

```
Out [55]: array([[197,  2,  1,  0],
                [142, 39, 19,  0],
                [108, 13, 68, 11],
                [137,  0, 44,  7]], dtype=int64)
```

```
In [56]: dump(clf4, 'models/SVM/SVM_models_ResNet50_CancerType_Malignant.joblib')
```

```
Out[56]: ['models/SVM/SVM_models_ResNet50_CancerType_Malignant.joblib']
```

```
In [ ]:
```