Appendix-3: SVM with Kernel

Support Vector Machines (SVM) are a powerful classification algorithm that learns a decision boundary by maximizing the margin between classes. It can be extended to multiclass classification using one-vs-one or one-vs-rest schemes.

We used scikit-learn's SVM implementation with the RBF kernel with default hyperparameters in this notebook.

In [1]:

```
# import library dependencies
import numpy as np
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f
import joblib
```

Import Data

In [2]:

```
ROOT_PATH='../'
```

In [3]:

```
# function to open pickle file
def unpickle(file):
    import pickle
    with open(file, 'rb') as fo:
        dict = pickle.load(fo, encoding='bytes')
    return dict
```

In [4]:

```
# store each pickle files in individual batches
batch1 = unpickle(ROOT_PATH+"cifar-10-batches-py/data_batch_1")
batch2 = unpickle(ROOT_PATH+"cifar-10-batches-py/data_batch_2")
batch3 = unpickle(ROOT_PATH+"cifar-10-batches-py/data_batch_3")
batch4 = unpickle(ROOT_PATH+"cifar-10-batches-py/data_batch_4")
batch5 = unpickle(ROOT_PATH+"cifar-10-batches-py/data_batch_5")
test_batch = unpickle(ROOT_PATH+"cifar-10-batches-py/test_batch")
```

In [5]:

```
# function to create labels and images from data
def load_data0(btch):
    labels = btch[b'labels']
    imgs = btch[b'data'].reshape((-1, 32, 32, 3))

    res = []
    for ii in range(imgs.shape[0]):
        img = imgs[ii].copy()
        img = np.fliplr(np.rot90(np.transpose(img.flatten().reshape(3,32,32)
        res.append(img)
    imgs = np.stack(res)
    return labels, imgs
```

In [6]:

```
# function to load data into training and test set
def load_data():
    x_train_l = []
    y_train_l = []
    for ibatch in [batch1, batch2, batch3, batch4, batch5]:
        labels, imgs = load_data0(ibatch)
        x_train_l.append(imgs)
        y_train_l.extend(labels)
    x_train = np.vstack(x_train_l)
    y_train = np.vstack(y_train_l)

    x_test_l = []
    y_test_l = []
    labels, imgs = load_data0(test_batch)
    x_test_l.append(imgs)
    y_test_l.extend(labels)
    x_test = np.vstack(x_test_l)
    y_test = np.vstack(y_test_l)
    return (x_train, y_train), (x_test, y_test)
```

Preprocess Data

In [7]:

```
# create training and test set
(x_train, y_train), (x_test, y_test) = load_data()
```

In [8]:

```
print('x_train shape:', x_train.shape)
print('y_train shape:', y_train.shape)
print('x_test shape:', x_test.shape)
print('y_test shape:', y_test.shape)
```

```
x_train shape: (50000, 32, 32, 3)
y_train shape: (50000, 1)
x_test shape: (10000, 32, 32, 3)
y_test shape: (10000, 1)
```

In [9]:

```
print(x_train.shape[0], 'train samples (x)')
print(y_train.shape[0], 'train samples (y)')
```

```
50000 train samples (x)
50000 train samples (y)
```

In [10]:

```
print(x_test.shape[0], 'test samples (x)')
print(y_test.shape[0], 'test samples (y)')
```

```
10000 test samples (x)
10000 test samples (y)
```

In [11]:

```
# flatten the images and scale the pixel values to [0, 1]
x_train = x_train.reshape((x_train.shape[0], -1))
x_test = x_test.reshape((x_test.shape[0], -1))
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255
```

Define model and train

In [12]:

```
# define an SVM model with RBF kernel
svm_k = SVC(kernel='rbf', gamma=0.001, C=100, verbose=True)
```

In [13]:

```
svm_k.fit(x_train, y_train.ravel())
```

```
[LibSVM]..............................................*.....................
*
optimization finished, #iter = 67928
obj = -72635.574909, rho = 11.252609
nSV = 3543, nBSV = 284
........................................................*...................
.....*
optimization finished, #iter = 80197
obj = -142618.383454, rho = -5.890823
nSV = 4214, nBSV = 1055
...............................................*....................*
optimization finished, #iter = 67271
obj = -87028.135115, rho = -2.703447
nSV = 3474, nBSV = 495
...............................................*......................*
optimization finished, #iter = 69994
obj = -113765.818076, rho = -8.265132
nSV = 3589, nBSV = 772
.......................................*.................*
optimization finished, #iter = 56663
obj = -73170.855762, rho = -1.181562
nSV = 3098, nBSV = 426
..............................*...............*
optimization finished, #iter = 45319
obj = -62392.946127, rho = -4.595452
nSV = 2611, nBSV = 339
............................................*......................*
optimization finished, #iter = 66203
obj = -79612.182691, rho = 2.396883
nSV = 3370, nBSV = 395
.................................................................*..........
...................*
optimization finished, #iter = 94911
obj = -176695.640705, rho = -10.632566
nSV = 4714, nBSV = 1412
.....................................................*.....................*
optimization finished, #iter = 74642
obj = -87057.979248, rho = 10.045430
nSV = 3788, nBSV = 412
....................................*................*
optimization finished, #iter = 52767
obj = -48100.816261, rho = -14.466735
nSV = 2817, nBSV = 122
.......................................*....................*
optimization finished, #iter = 59420
obj = -52028.722466, rho = -11.682215
nSV = 3259, nBSV = 136
................................*...............*
optimization finished, #iter = 47467
obj = -45472.062614, rho = -18.087346
nSV = 2643, nBSV = 145
```

```
.................................*................*
optimization finished, #iter = 49865
obj = -41991.126764, rho = -10.317473
nSV = 2873, nBSV = 99
............................*..............*
optimization finished, #iter = 42426
obj = -37809.463301, rho = -14.317431
nSV = 2536, nBSV = 80
...................................*................*
optimization finished, #iter = 51982
obj = -43931.607458, rho = -8.232210
nSV = 3023, nBSV = 112
.................................................*......................*
optimization finished, #iter = 71101
obj = -78042.245466, rho = -17.526227
nSV = 3603, nBSV = 333
............................................................................
.........*........................................*
optimization finished, #iter = 125028
obj = -132675.472625, rho = 0.864516
nSV = 5962, nBSV = 534
............................................................................
.....*....................................*
optimization finished, #iter = 117461
obj = -213455.068092, rho = 5.134571
nSV = 5686, nBSV = 1651
............................................................................
.....*..................................*
optimization finished, #iter = 115339
obj = -322272.301884, rho = -1.989278
nSV = 6591, nBSV = 2918
............................................................................
*.....................................*
optimization finished, #iter = 113233
obj = -200487.521821, rho = 5.470717
nSV = 5486, nBSV = 1510
......................................................................*.....
.........................*
optimization finished, #iter = 100705
obj = -214485.672013, rho = 8.373796
nSV = 5336, nBSV = 1742
...................................................................*........
.....................*
optimization finished, #iter = 96893
obj = -144786.602701, rho = 11.385714
nSV = 4644, nBSV = 956
....................................*...................*
optimization finished, #iter = 55510
obj = -75380.036036, rho = 0.019196
nSV = 2854, nBSV = 486
......................................*.................*
optimization finished, #iter = 55648
obj = -57237.625518, rho = 13.587947
nSV = 3028, nBSV = 212
.....................................................................*......
...........................*
optimization finished, #iter = 102871
```

```
obj = -183729.995736, rho = -7.392411
nSV = 4978, nBSV = 1374
............................................................................
.....................................*......................................
......*
optimization finished, #iter = 157332
obj = -293682.724521, rho = 2.023147
nSV = 7619, nBSV = 2289
............................................................................
*.................................*
optimization finished, #iter = 109895
obj = -198430.880321, rho = 0.599597
nSV = 5519, nBSV = 1465
................................................................*...........
...................*
optimization finished, #iter = 94813
obj = -123348.543868, rho = 5.373340
nSV = 4605, nBSV = 740
......................................*.................*
optimization finished, #iter = 55240
obj = -63630.772111, rho = -3.141234
nSV = 2878, nBSV = 318
..............................................*....................*
optimization finished, #iter = 66695
obj = -63094.445601, rho = 11.497117
nSV = 3542, nBSV = 208
...................................................................*........
.........................*
optimization finished, #iter = 100071
obj = -177502.484257, rho = 8.563752
nSV = 4855, nBSV = 1310
...................................................................*........
......................*
optimization finished, #iter = 97197
obj = -212167.080573, rho = 12.941400
nSV = 5242, nBSV = 1818
.....................................................................*......
..........................*
optimization finished, #iter = 101288
obj = -162962.530224, rho = 14.365815
nSV = 4846, nBSV = 1116
.................................*................*
optimization finished, #iter = 49502
obj = -71431.602761, rho = 0.915468
nSV = 2660, nBSV = 438
...................................*.................*
optimization finished, #iter = 52060
obj = -53636.992306, rho = 16.798654
nSV = 2795, nBSV = 188
....................................................................*.......
........................*
optimization finished, #iter = 99845
obj = -160527.437301, rho = -0.573750
nSV = 4897, nBSV = 1118
...................................................................*........
........................*
optimization finished, #iter = 99004
```

```
obj = -136189.060714, rho = 4.541130
nSV = 4832, nBSV = 837
..............................*................*
optimization finished, #iter = 46804
obj = -51696.291496, rho = -2.014652
nSV = 2538, nBSV = 261
.....................................*...................*
optimization finished, #iter = 56154
obj = -50844.056441, rho = 10.573630
nSV = 3102, nBSV = 146
...........................................*......................*
optimization finished, #iter = 65091
obj = -87000.807635, rho = 5.746809
nSV = 3416, nBSV = 462
......................*..........*
optimization finished, #iter = 32157
obj = -39959.428293, rho = 0.706176
nSV = 1963, nBSV = 192
.............................*...............*
optimization finished, #iter = 44080
obj = -42792.384475, rho = 13.975580
nSV = 2635, nBSV = 123
..............................*.............*
optimization finished, #iter = 43252
obj = -44977.517580, rho = -5.966771
nSV = 2443, nBSV = 184
.............................................*.....................*
optimization finished, #iter = 66572
obj = -64601.279916, rho = 7.469977
nSV = 3601, nBSV = 221
................................................*.......................*
optimization finished, #iter = 71878
obj = -80756.525974, rho = 15.862760
nSV = 3597, nBSV = 372
Total nSV = 42569
```

Out[13]:

```
▾ SVC
SVC(C=100, gamma=0.001, verbose=True)
```

Save and load model

In [15]:

```python
# Save the model to a file
joblib.dump(svm_k, 'svm_with_kernel.sav')
```

Out[15]:

```
['svm_with_kernel.sav']
```

In [16]:

```python
# Load the saved model from a file
loaded_model = joblib.load('svm_with_kernel.sav')
```

In [17]:

```python
# predict the labels of the test set
y_pred = loaded_model.predict(x_test)
```

Evaluate the model

In [18]:

```
# compute the accuracy, precision, recall, and F1 score of the classifier
train_acc = accuracy_score(y_train.ravel(), loaded_model.predict(x_train))
test_acc = accuracy_score(y_test.ravel(), y_pred)
precision = precision_score(y_test.ravel(), y_pred, average='macro')
recall = recall_score(y_test.ravel(), y_pred, average='macro')
f1 = f1_score(y_test.ravel(), y_pred, average='macro')
```

In [19]:

```
# print accuracy, precision, recall, and F1 score
print('Train Accuracy:', train_acc)
print('Test Accuracy:', test_acc)
print('Precision: {:.3f}'.format(precision))
print('Recall: {:.3f}'.format(recall))
print('F1 Score: {:.3f}'.format(f1))
```

```
Train Accuracy: 0.93224
Test Accuracy: 0.539
Precision: 0.542
Recall: 0.539
F1 Score: 0.540
```