hw2_coding

March 23, 2023

1 SVM

We will be utilizing the Scikit Learn SVM package for this homework, as you had practiced before in Tutorial. You may find it helpful to review the Scikit Learn's SVM documentation (http://scikit-learn.org/stable/modules/svm.html).

We apply soft-margin SVM to handwritten digits from the processed US Postal Service Zip Code data set. The 2D data represents extracted features of intensity and symmetry for training and testing.

Hint: - Answers without corresponding experimentation to validate your findings will be assigned minimal partial credit. - Submitting the .ipynb file and the corresponding .pdf file.

[ ]:
```python
# import libraries
import numpy as np
from sklearn.svm import SVC
# calculating accuracy
from sklearn.metrics import accuracy_score
```

[ ]:
```python
### Loading data
trainData = np.loadtxt('features.train')
testData = np.loadtxt('features.test')
```

[ ]:
```python
# spliting data for features and labels
dataTrain = trainData[:,1:]
labelTrain = trainData[:,0]
dataTest = testData[:,1:]
labelTest = testData[:,0]
```

[ ]:
```python
def cout(kernel,totalSV,trainAccuracy,testAccuracy):
    # report your results
    print("Kernel: "+ str(kernel))
    print("Number of Support Vectors: "+ str(totalSV))
    print("Train Accuracy: "+ str(trainAccuracy))
    print("Test Accuracy: " + str(testAccuracy))
```

2 linear Kernel(20pts)

Consider the linear kernel $K(x_n, x_m) = x_n^T x_m$. Displaying the outpout by above funtion *cout*.

```
[ ]: """
     Three variables assigned   by yourself.
     predTrain : Predicted results on the training set
     predTest : Predicted results on the test set
     totalSV: The number of support vectors
     """

     kernel = "linear"
     ### YOUR CODE HERE - Create and  train the model SVM
     clf = SVC(kernel=kernel)
     clf.fit(dataTrain, labelTrain)
     ### YOUR CODE HERE - predict on the training set
     predTrain = clf.predict(dataTrain)
     ### YOUR CODE HERE - predict on the test set
     predTest = clf.predict(dataTest)
     ### YOUR CODE HERE - account the number of support vectors
     totalSV = clf.n_support_.sum()
     trainAccuracy = accuracy_score(labelTrain,predTrain)
     testAccuracy = accuracy_score(labelTest,predTest)
     cout(kernel,totalSV,trainAccuracy,testAccuracy)
```

```
Kernel: linear
Number of Support Vectors: 5850
Train Accuracy: 0.4169524070772185
Test Accuracy: 0.3976083707025411
```

3 polynomial kernel(20pts)

Consider the polynomial kernel $K(xn, xm) = (1+x_n^T x_m)^Q$, where Q is the degree of the polynomial. When C=0.01, which value of $Q \in \{2, 5\}$ results in the highest testAccuracy?

Answer: $Q = 2$

```
[ ]: kernel = "poly"

     for Q in [2,5]:
         print("#########"+str(Q)+"#############")
         ### YOUR CODE HERE - Create and  train the model SVM
         clf = SVC(kernel=kernel, degree=Q)
         clf.fit(dataTrain, labelTrain)
         ### YOUR CODE HERE - predict on the training set
         predTrain = clf.predict(dataTrain)
         ### YOUR CODE HERE - predict on the testing set
         predTest = clf.predict(dataTest)
```

```
    ### YOUR CODE HERE - account the number of support vectors
    totalSV = clf.n_support_.sum()
    trainAccuracy = accuracy_score(labelTrain,predTrain)
    testAccuracy = accuracy_score(labelTest,predTest)
    cout(kernel,totalSV,trainAccuracy,testAccuracy)
```

```
#########2#############
Kernel: poly
Number of Support Vectors: 5989
Train Accuracy: 0.3963790975174873
Test Accuracy: 0.38216243148978574
#########5#############
Kernel: poly
Number of Support Vectors: 6047
Train Accuracy: 0.35043203950075436
Test Accuracy: 0.34479322371699056
```

4 RBF Kernel(20pts)

Consider the radial basis function (RBF) kernel $K(xn, xm) = exp(-||x_n - x_m||^2)$ in the soft-margin SVM approach. Which value of $C \in \{0.01, 1, 100, 10^4\}$ results in the highest testAccuracy?

Answer: $C = 10000$

```
[ ]: kernel = "rbf"
     for C in [0.01,1,100,10**4]:
         print("#########"+str(C)+"#############")
         ### YOUR CODE HERE - Create and  train the model SVM
         clf = SVC(kernel=kernel, C=C)
         clf.fit(dataTrain, labelTrain)
         ### YOUR CODE HERE - predict on the training set
         predTrain = clf.predict(dataTrain)
         ### YOUR CODE HERE - predict on the testing set
         predTest = clf.predict(dataTest)
         ### YOUR CODE HERE - account the number of support vectors
         totalSV = clf.n_support_.sum()

         trainAccuracy = accuracy_score(labelTrain,predTrain)
         testAccuracy = accuracy_score(labelTest,predTest)
         cout(kernel,totalSV,trainAccuracy,testAccuracy)
```

```
#########0.01#############
Kernel: rbf
Number of Support Vectors: 6736
Train Accuracy: 0.3339733918529694
Test Accuracy: 0.3243647234678625
#########1#############
Kernel: rbf
```

```
Number of Support Vectors: 6001
Train Accuracy: 0.3996708270470443
Test Accuracy: 0.3786746387643249
#########100#############
Kernel: rbf
Number of Support Vectors: 5708
Train Accuracy: 0.4302564805925113
Test Accuracy: 0.3961136023916293
#########10000#############
Kernel: rbf
Number of Support Vectors: 5679
Train Accuracy: 0.4302564805925113
Test Accuracy: 0.3966118584952666
```