Exercise8_SVM

November 5, 2021

0.1 Exercise 8: SVM

0.1.1 Part 1:

In this part, we will apply SVM for classification of breast cancer using the Wisconsin's data. Read the description of the dataset, then answer the following questions. Whenever needed to split the data, use random_state=0 in train_test_split

1) Find the accuracy of SVM classifier with parameter C=0.1, and radial basis function kernel (rbf) of parameter Gamma =0.2.

[19]:
```
from sklearn.svm import SVC
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import cross_val_score
import numpy as np

CancerDataset=load_breast_cancer()

X_train, X_test, Y_train, Y_test = train_test_split(CancerDataset['data'],␣
↪CancerDataset['target'], random_state=0)

svmModel = SVC(kernel='rbf', gamma=0.2, C=0.1)
svmModel.fit(X_train, Y_train)

acc = svmModel.score(X_test, Y_test)
print('Accuracy is %.4f.' % acc)
```

```
Accuracy is 0.6294.
```

2) Repeat part (1) but scale the features with MinMaxScaler. Compare results of (1) and (2) and comment on the results.

[20]:
```
scaler = MinMaxScaler().fit(X_train)

X_train_t = scaler.transform(X_train)
X_test_t = scaler.transform(X_test)

svmModel.fit(X_train_t, Y_train)
```

```
acc = svmModel.score(X_test_t, Y_test)
print('Accuracy is %.4f.' % acc)
```

```
Accuracy is 0.9510.
```

(AP): The accuracy is much better when using the MinMaxScaler instead of the normal data.

3) Using scaled features, find best SVM classifier. Try values of the regularization C=[0.1, 1, 5] and RBF parameter Gamma = [0.1, 1, 5]. Use 5-fold cross validation to find the best parameters (using all possible combinations of these values for C and gamma).

[21]:
```
X_trainval_t = X_train_t
Y_trainval = Y_train

best_acc = 0
best_c = -1
best_gamma = -1

for c in [0.1, 1, 5]:
    for gamma in [0.1, 1, 5]:
        svmModel = SVC(kernel='rbf', gamma=gamma, C=c)
        scores = cross_val_score(svmModel, X_trainval_t, Y_trainval, cv=5)

        mean = scores.mean()

        if mean > best_acc:
            best_acc = mean
            best_c = c
            best_gamma = gamma

print(f'Best tuning parameters are c={best_c} and gamma={best_gamma}.')
print('Trainval accuracy with these tuning parameters = %.4f.' % best_acc)

BestModel = SVC(kernel='rbf', gamma=best_gamma, C=best_c)
BestModel.fit(X_trainval_t, Y_trainval)
print('Test accuracy with these tuning parameters = %.4f' % BestModel.
 ↪score(X_test_t, Y_test))
```

```
Best tuning parameters are c=1 and gamma=1.
Trainval accuracy with these tuning parameters = 0.9812.
Test accuracy with these tuning parameters = 0.9720
```

0.1.2 Part 2:

Coding not required. You can upload handwritten answer for this part if needed. Consider dataset below with just four samples, 2 features (X1, X2) and binary class label Y.

Observations:

X1=-1, X2=-1, Y=-1

X1=-1, X2= 1, Y= 1

X1= 1, X2=-1, Y=1

X1= 1, X2=1, Y=-1

1) Can a linear SVC perfectly slassify the data?
2) Suppose we modify the features to be (X1 , X1.X2), i.e we have interaction term instead of just X2. Can a linear SVC work well on the transformed features? Justify

(AP) I have uploaded my handwritten solutions in a separate file. 1. A linear SVC can not perfectly classify the data because it is not linearly separable, so there is no way to draw a line that separates it. 2. Yes, a linear SVC can now classify the data. The transformation has resulted in the data being linearly separable now.