Sri Hari Malla
Indian Institute of Technology Hyderabad
Fundamentals of Machine Learning
CS19BTECH11039

Report

Sri Hari Malla - CS19BTECH11039

October 6, 2021

1 4. SVM

Reading Data:

- Used pandas module to read the data. The data is directly read from internet with the given URL.
- Assigned the column names while reading with features and labels.
- Separated the labels with '1' and '5' from the entire data set into X_train,y_train,X_test and y_test.

1.1 4(a)

- Imported SVC from sklearn module.
- Imported accuracy_score from sklearn.metrics module.
- Created a classifier by explicitly mentioning the linear kernel.
- Trained the model using `.fit()` method on X_train and y_train.
- Predicted over the test set using `.predict()` method on X_test.
- Used `accuracy_score()` over y_test and predictedY to get the accuracy over the test set.
- Used `n_support_` to get the number of support vectors used.

1.1.1 4(a) Observations:

```
Accuracy score = 0.9787735849056604
Support vectors = [14 14]
```

1.2 4(b)

- For each of the case, the data points are abstracted with respect to given count.
- Trained the model using `.fit()` method on X_train and y_train.
- Now predicted over the test set using `.predict()` method on X_test.
- Used `accuracy_score()` over y_test and predictedY to get the accuracy over the test set.
- Used `n_support_` to get the number of support vectors used.

1.2.1 4(b) Observations:

```
case1 : First 50 data points
    Accuracy Score = 0.9811320754716981
    Number of support vectors : [1 1]
case2 : First 100 data points
    Accuracy Score = 0.9811320754716981
    Number of support vectors : [2 2]
case3 : First 200 data points
    Accuracy Score = 0.9811320754716981
    Number of support vectors : [4 4]
case4 : First 800 data points
    Accuracy Score = 0.9811320754716981
    Number of support vectors : [7 7]
```

1.3 4(c)

- Explicitly mentioned the kernel to be 'poly'
- Also set the `coef0` of kernel to 1.
- The C and q were taken in lists.
- Iterated using for loops and iterated all the possibilities.
- Training error is obtained by using `score()` on `fit()`
- Testing error is obtained by using `accuracy_score()`
- Used `n_support_` to get the number of support vectors used.

1.3.1 4(c) Observations:

```
C =  0.0001 Degree =  2
Training Error :  0.008968609865470878
C =  0.0001 Degree =  5
Training Error :  0.004484304932735439
C =  0.001 Degree =  2
Number of support vectors: [38 38]
C =  0.001 Degree =  5
Number of support vectors: [12 13]
C =  0.01 Degree =  2
Training Error :  0.004484304932735439
C =  0.01 Degree =  5
Training Error :  0.0038436899423446302
C =  1 Degree =  2
Testing Error :  0.018867924528301883
C =  1 Degree =  5
Testing Error :  0.021226415094339646

C = 0.0001, Training error is actually lower when degree is 5 than degree is 2.
So, it's False
C = 0.001, Number of support vectors are low when degree is 5 than degree is 2.
So, it's True
C = 0.01, Training error is actually higher when degree is 2 than degree is 5.
So, it's False
C = 1, Testing error is actually higher when degree is 5 than degree is 2.
So it's False
```

1.4 4(d)

- The Kernel is now set to 'rbf'
- Set the gamma of kernel to 1
- Took the values of C in a list and iterated over the list to get all possibilities.
- Training error is obtained by using `score()` on `fit()`
- Testing error is obtained by using `accuracy_score()`

1.4.1 4(d) Observations:

```
Training error for rbf kernel with C =  0.01  is  0.0038436899423446302
Testing error for rbf kernel with C =  0.01  is  0.02358490566037741
```

```
Training error for rbf kernel with C =  1.0  is  0.004484304932735439
Testing error for rbf kernel with C =  1.0  is  0.021226415094339646
Training error for rbf kernel with C =  100.0  is  0.0032030749519538215
Testing error for rbf kernel with C =  100.0  is  0.018867924528301883
Training error for rbf kernel with C =  10000.0  is  0.002562459961563124
Testing error for rbf kernel with C =  10000.0  is  0.02358490566037741
Training error for rbf kernel with C =  1000000.0  is  0.0006406149903908087
Testing error for rbf kernel with C =  1000000.0  is  0.02358490566037741

Testing error is minimum for C = 100
Training error is minimum for C = 1e6
```

2 5. SVM

Reading Data:

- I've used pandas module to read the data. The data is downloaded and is used from local storage.
- Features for training were read into train_data, labels for training were read into train_labels, features for validation set were read into valid_data and labels for testing were read into valid_labels.

2.1 5(a)

- Created a classifier by explicitly mentioning the linear kernel.
- Training error is obtained by using `score()` on `fit()`
- Testing error is obtained by using `accuracy_score()`
- Used `n_support_` to get the number of support vectors used.

2.1.1 5(a) Observations:

```
Training Error :  0.0
Testing Error :  0.02400000000000002
Support Vectors :  [542 542]
```

2.2 5(b)

2.2.1 RBF kernel

- The kernel was set to 'rbf'

- Gamma set to 0.001
- Training error is obtained by using `score()` on `fit()`
- Testing error is obtained by using `accuracy_score()`
- Used `n_support`

Observations:

```
Training error :  0.0
Testing error :  0.5
Number of support vectors :  [3000 3000]
```

2.2.2 Polynomial kernel

- The kernel was set to 'poly'
- Degree set to 2
- coef0 set to 1
- Training error is obtained by using `score()` on `fit()`
- Testing error is obtained by using `accuracy_score()`
- Used `n_support`

Observations:

```
Training error :  0.0
Testing error :  0.02100000000000002
Number of support vectors :  [817 938]
```

```
Final observations:
    Number of support vectors used is greater for RBF kernel.
    Training error is 0 for both.
    Testing error is least for polynomial kernel.
```

LaTeX generated document

THE END