HW4: SVM

Prof. Chia-Yu Lin

National Central University

2022 Fall

Prediction by the trained SVM model

- We have finished the training of SVM model in the class.
- We can use “test.csv” to do the test.

```
1  #取得測試資料的欄位
2  testset = testset[testset.columns[:449]]
```

SVM預測

```
1 svm.predict(testset)
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
       6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
       6, 6, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
       7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
       7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7])
```

- But Prof. doesn’t have the answer of test.csv. I cannot check the correctness of the prediction.
- So….
- We can randomly choose some data from training dataset to do the test.

HW4-1: Randomly choose data from training dataset to do the test (1/2)

- Read training data.
- Check the row of training data.
- Set the range of random function. Randomly generate 10 integers.

```
資料的形式:(1741, 450)
[1062  236 1097  906  716 1670  848  961  145  130]
```

HW4-1: Randomly choose data from training dataset to do the test (2/2)

- Use the result of random function be the index to get the row.

	col1	col2	col3	col4	col5	col6	col7	col8	col9	col10	...	col440	col441	col442	col443	col444	col445	col446	col447	col448	col449
1062	70.1	70.1	70.1	70.0	70.2	70.2	70.4	70.5	70.7	70.8	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
236	78.6	78.6	78.6	78.6	79.2	79.9	80.5	81.3	82.2	83.6	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1097	81.4	81.4	81.4	81.5	81.8	82.0	82.3	82.6	82.9	83.1	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
906	76.2	76.2	76.2	76.2	76.4	76.7	77.0	77.5	78.2	79.2	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
716	69.7	69.7	69.8	69.8	72.0	71.3	72.4	75.7	75.4	77.2	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1670	70.0	70.0	70.0	70.1	70.4	70.7	70.8	71.0	71.2	71.5	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
848	84.9	84.9	84.9	84.9	84.9	85.1	85.4	86.2	86.6	87.5	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
961	72.3	72.3	72.3	72.3	72.8	73.3	74.2	75.2	76.5	78.2	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
145	81.2	81.2	81.2	81.2	81.4	82.0	83.0	84.1	85.2	86.7	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
130	92.0	92.0	92.0	92.0	93.4	94.2	96.1	98.3	101.0	103.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

10 rows × 449 columns

SVM Prediction

- Prediction

```
1 predicted=svm.predict(dataset_fortest)
2 predicted
```

```
array([5, 1, 5, 4, 4, 7, 4, 4, 3, 0])
```

- Actual Answer

```
1 expected=label.values
2 expected
```

```
array([5, 1, 5, 4, 4, 7, 4, 4, 1, 0])
```

Accuracy

```
print(metrics.classification_report(expected,predicted))
print(metrics.confusion_matrix(expected, predicted))
accuracy = accuracy_score(expected, predicted)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
print('precision:',metrics.precision_score(expected, predicted,average='macro'))
print('recall:',metrics.recall_score(expected, predicted,average='macro'))
print('F1-score:',metrics.f1_score(label,predicted,average='macro'))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      0.50      0.67         2
           3       0.00      0.00      0.00         0
           4       1.00      1.00      1.00         4
           5       1.00      1.00      1.00         2
           7       1.00      1.00      1.00         1

    accuracy                           0.90        10
   macro avg       0.83      0.75      0.78        10
weighted avg       1.00      0.90      0.93        10

[[1 0 0 0 0 0]
 [0 1 1 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 4 0 0]
 [0 0 0 0 2 0]
 [0 0 0 0 0 1]]
Accuracy: 90.00%
precision: 0.8333333333333334
recall: 0.75
F1-score: 0.7777777777777777
```

HW4-2

- Repeat the above steps using random forest model.

HW Submission (1/2)

- According to the following description to finish “HW4-SVM.ipynb”
- Printscreen
  - Predicted result of SVM model
  - Actual answer
  - Accuracy of the SVM model

```
1 predicted=svm.predict(dataset_fortest)
2 predicted
```

```
array([5, 1, 5, 4, 4, 7, 4, 4, 3, 0])
```

```
1 expected=label.values
2 expected
```

```
array([5, 1, 5, 4, 4, 7, 4, 4, 1, 0])
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
1	1.00	0.50	0.67	2
3	0.00	0.00	0.00	0
4	1.00	1.00	1.00	4
5	1.00	1.00	1.00	2
7	1.00	1.00	1.00	1
accuracy			0.90	10
macro avg	0.83	0.75	0.78	10
weighted avg	1.00	0.90	0.93	10

```
[[1 0 0 0 0 0]
 [0 1 1 0 0 0]
 [0 0 0 0 0 0]
 [0 0 0 4 0 0]
 [0 0 0 0 2 0]
 [0 0 0 0 0 1]]
Accuracy: 90.00%
precision: 0.8333333333333334
recall: 0.75
F1-score: 0.7777777777777777
```

HW Submission (2/2)

- Same as random forest model. Printscreen the predicted result of random forest model, actual answer and accuracy of random forest model.
- Put the printscreens in the word document.
- Compress the word document and HW4-SVM.ipynb and upload.
- Deadline: 2022. Nov. 14