Evaluation

Ko, Youngjoong

Sungkyunkwan University

nlp.skku.edu, nlplab.skku.edu

Contents

- Introduction
- Evaluation Method
- Evaluation with Scikit-Learn
- Assignments

Introduction

❖ Introduction

- The assignment is to calculate various evaluation methods using classification results and labels. (See page 19 for specifics)

- In this PDF, we will explain various evaluation methods and how to use the functions of scikit-learn library for evaluation.

Evaluation Method

❖ Evaluation Methods

- There are several evaluation methods as below.
  - Confusion Matrix
  - Accuracy
  - Precision
  - Recall
  - F1-score
  - Macro Averaging
  - Micro Averaging

Evaluation Method

❖ Confusion Matrix

- Confusion matrix is a matrix of prediction and actual labels.

- Example of confusion matrix (ex. Classification for cat and non-cat)
  - True Positive (TP) : predict cat as cat
  - True Negative (TN) : predict non-cat as non-cat
  - False Positive (FP) : predict cat as non-cat
  - False Negative (FN) : predict non-cat as cat

Label \ Prediction	Predict as cat	Predict as non-cat
Actually cat	True Positive (TP)	False Negative (FN)
Actually non-cat	False Positive (FP)	True Negative (TN)

Evaluation Method

❖ Accuracy

- Accuracy is ratio of correctly predicted samples to the whole samples.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

Label \ Prediction	$\hat{T}$	$\hat{F}$
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

Evaluation Method

❖ Precision

- Precision is ratio of actual true samples to the samples predicted as true.

$$Precision = \frac{TP}{TP + FP}$$

Label \ Prediction	$\hat{T}$	$\hat{F}$
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

Evaluation Method

❖ Recall

- Recall is ratio of samples predicted as true to the actually true samples.

$$Recall = \frac{TP}{TP + FN}$$

Label \ Prediction	$\hat{T}$	$\hat{F}$
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

Evaluation Method

❖ F1-score

- F1-score is harmonic mean between precision and recall.

$$F1 - score = \frac{2 \times precision \times recall}{precision + recall}$$

Evaluation Method

❖ Macro Averaging

- To calculate Macro averaging scores, we first calculate precision, recall and F1-score for each label respectively and then divide each score with the number of labels. (see next page)

Label \ Prediction	$\hat{A}$	$\hat{B}$	$\hat{C}$
A	9	3	0
B	3	5	1
C	1	1	4

$\hat{T}$, $\hat{F}$: Prediction
T, F : Labels

A	$\hat{T}$	$\hat{F}$
T	9	3
F	4	11

B	$\hat{T}$	$\hat{F}$
T	5	4
F	4	14

C	$\hat{T}$	$\hat{F}$
T	4	2
F	1	20

Evaluation Method

❖ Macro Averaging

A	$\hat{T}$	$\hat{F}$
T	9	3
F	4	11

$Precision_A = 0.69$
$Recall_A = 0.75$
$F1 - score_A = 0.72$

B	$\hat{T}$	$\hat{F}$
T	5	4
F	4	14

$Precision_B = 0.55$
$Recall_B = 0.55$
$F1 - score_B = 0.55$

C	$\hat{T}$	$\hat{F}$
T	4	2
F	1	20

$Precision_C = 0.66$
$Recall_C = 0.80$
$F1 - score_C = 0.72$

$$Macro\ averaging\ Precision = \frac{0.69 + 0.55 + 0.66}{3} = 0.63$$

$$Macro\ averaging\ Recall = \frac{0.75 + 0.55 + 0.80}{3} = 0.70$$

$$Macro\ averaging\ F1 - score = \frac{0.72 + 0.55 + 0.72}{3} = 0.66$$

Evaluation Method

❖ Micro Averaging

- To calculate Micro-averaging scores, we first add up all confusion matrices into one confusion matrix, and then calculate precision, recall and F1-score using the confusion matrix. (see next page)

Label \ Prediction	$\hat{A}$	$\hat{B}$	$\hat{C}$
A	9	3	0
B	3	5	1
C	1	1	4

$\hat{T}$, $\hat{F}$: Prediction
T, F : Labels

A	$\hat{T}$	$\hat{F}$
T	9	3
F	4	11

B	$\hat{T}$	$\hat{F}$
T	5	4
F	4	14

C	$\hat{T}$	$\hat{F}$
T	4	2
F	1	20

Evaluation Method

❖ Micro Averaging

A	$\hat{T}$	$\hat{F}$
T	9	3
F	4	11

B	$\hat{T}$	$\hat{F}$
T	5	4
F	4	14

C	$\hat{T}$	$\hat{F}$
T	4	2
F	1	20

SUM

$Micro$	$\hat{T}$	$\hat{F}$
T	18	9
F	9	45

$$Micro\ averaging\ Precision = 0.67$$

$$Micro\ averaging\ Recall = 0.67$$

$$Micro\ averaging\ F1 - score = 0.67$$

Evaluation with Scikit-Learn

❖ Evaluation with Scikit-Learn

- 'sklearn.metrics' provides following evaluation python classes.
  - confusion_matrix
  - accuracy_score
  - precision_score
  - recall_score
  - f1_score

Evaluation with Scikit-Learn

❖ Examples of confusion_matrix, accuracy_score

```
 1 from sklearn.metrics import confusion_matrix, accuracy_score
 2
 3 # Label
 4 y_true = [1, 0, 0, 1, 1, 0, 0, 1, 1, 0]
 5 # Prediction
 6 y_pred = [1, 1, 1, 1, 1, 0, 0, 1, 0, 1]
 7
 8 # confusion matrix
 9 print(confusion_matrix(y_true, y_pred))
10 # Accuracy
11 print('Accuracy : ', accuracy_score(y_true, y_pred))
```

```
[[2 3]
 [1 4]]
Accuracy :  0.6
```

Label \ Prediction	0	1
0	2	3
1	1	4

```
 1 from sklearn.metrics import confusion_matrix, accuracy_score
 2
 3 # Label
 4 y_true = [2, 1, 0, 2, 0, 1, 1, 2, 2, 0, 0, 1, 2, 1, 0]
 5 # Prediction
 6 y_pred = [2, 1, 2, 2, 2, 1, 1, 2, 0, 0, 0, 2, 1, 0, 1]
 7
 8 # confusion matrix
 9 print(confusion_matrix(y_true, y_pred))
10 # Accuracy
11 print('Accuracy : ', accuracy_score(y_true, y_pred))
```

```
[[2 1 2]
 [1 3 1]
 [1 1 3]]
Accuracy :  0.5333333333333333
```

Label \ Prediction	0	1	2
0	2	1	2
1	1	3	1
2	1	1	3

Evaluation with Scikit-Learn

❖ precision_score, recall_score, f1_score

- Parameters
  - average : string (Default : ‘binary’)
    - Parameters used for multi-label classification
    - macro : use ‘macro averaging’
    - micro : use ‘micro averaging’

```
 1 from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score
 2
 3 # Label
 4 y_true = [2, 1, 0, 2, 0, 1, 1, 2, 2, 0, 0, 1, 2, 1, 0]
 5 # Prediction
 6 y_pred = [2, 1, 2, 2, 2, 1, 1, 2, 0, 0, 0, 2, 1, 0, 1]
 7
 8 # confusion matrix
 9 print(confusion_matrix(y_true, y_pred))
10 # Precision
11 print('Macro averaging Precision : ', precision_score(y_true, y_pred, average='macro'))
```

```
[[2 1 2]
 [1 3 1]
 [1 1 3]]
Macro averaging Precision :  0.5333333333333333
```

Evaluation with Scikit-Learn

❖ Examples of precision_score, recall_score, f1_score

```
 1 from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score
 2
 3 # Label
 4 y_true = [1, 0, 0, 1, 1, 0, 0, 1, 1, 0]
 5 # Prediction
 6 y_pred = [1, 1, 1, 1, 1, 0, 0, 1, 0, 1]
 7
 8 # confusion matrix
 9 print(confusion_matrix(y_true, y_pred))
10 # Precision
11 print('Precision : ', precision_score(y_true, y_pred))
12 # Recall
13 print('Recall : ', recall_score(y_true, y_pred))
14 # F1-score
15 print('F1-score : ', f1_score(y_true, y_pred))
```

```
[[2 3]
 [1 4]]
Precision :  0.5714285714285714
Recall :  0.8
F1-score :  0.6666666666666666
```

Evaluation with Scikit-Learn

❖ Examples of precision_score, recall_score, f1_score

```python
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score

# Label
y_true = [2, 1, 0, 2, 0, 1, 1, 2, 2, 0, 0, 1, 2, 1, 0]
# Prediction
y_pred = [2, 1, 2, 2, 2, 1, 1, 2, 0, 0, 0, 2, 1, 0, 1]

# confusion matrix
print(confusion_matrix(y_true, y_pred))
# Precision
print('Macro averaging Precision : ', precision_score(y_true, y_pred, average='macro'))
print('Micro averaging Precision : ', precision_score(y_true, y_pred, average='micro'))
# Recall
print('Macro averaging Recall : ', recall_score(y_true, y_pred, average='macro'))
print('Micro averaging Recall : ', recall_score(y_true, y_pred, average='micro'))
# F1-score
print('Macro averaging F1-score : ', f1_score(y_true, y_pred, average='macro'))
print('Micro averaging F1-score : ', f1_score(y_true, y_pred, average='micro'))
```

```
[[2 1 2]
 [1 3 1]
 [1 1 3]]
Macro averaging Precision :  0.5333333333333333
Micro averaging Precision :  0.5333333333333333
Macro averaging Recall :  0.5333333333333333
Micro averaging Recall :  0.5333333333333333
Macro averaging F1-score :  0.52996632996633
Micro averaging F1-score :  0.5333333333333333
```

Assignment

❖ Assignment

1) Use the ‘Evaluation.json’ file as an input.

   - See page 22 for input file format.

2) Construct ‘confusion matrix’, and calculate ‘accuracy’, ‘precision’, ‘recall’, and ‘F1-score’ with respect to ‘macro averaging’ and ‘micro averaging’.

3) You have to create .txt file containing results of above metrics in form of percentage (file I/O) by using your code.

   - See page 23 for output file format.

4) You are **allowed to use only ‘Scikit-Learn’** for calculating above metrics.

5) You cannot use any library except scikit-learn and json.

Assignment

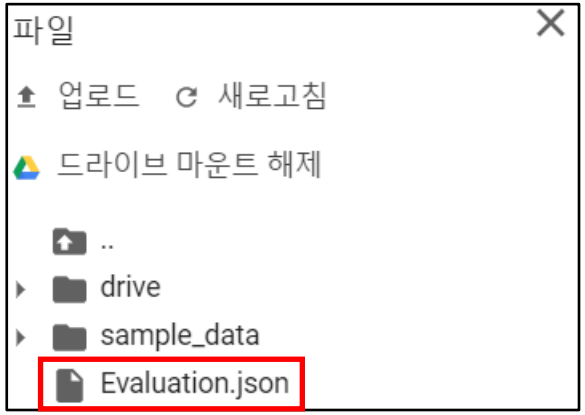
❖ Assignment

6) Teaching assistants have been checking out “Copy” to protect plagiarism by a copy detecting system. 0 scores will be assigned to all “Copy” results.

7) Round down to the fourth digit after the decimal point.
   Ex) 78.42159 → 78.4215 / 51.172426 → 51.1724

8) For this assignment, If there is any evidence that you upload your source code or download it from any public internet site (Stack Overflow, Github, etc.), we will regard it as copy and you will get 0 score.

9) Upload the input file to the same folder where the source code exists as shown on the right.



10) All paths in the code use a relative path as below.
   Ex)

```
with open('./Evaluation.json') as f:
```

Assignment

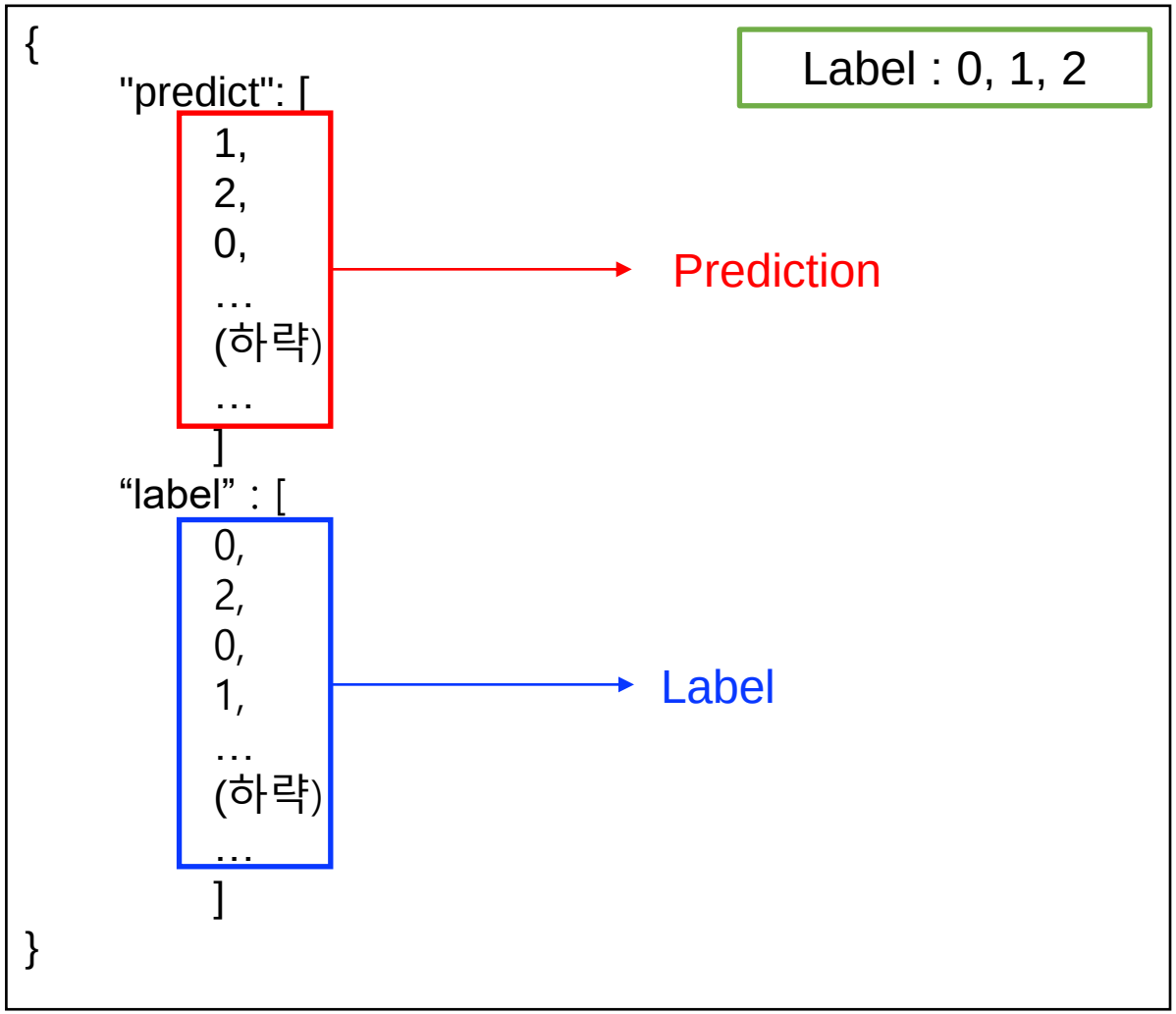
❖ Submission File (submit following files as a compressed .zip file)

1) Python code file (.py) (python version 3.x , not .ipynb file)
   - Format: “Student Number_Name_EVAL.py”.
     - Ex) “2020000000_홍길동_EVAL.py”
   - **You will get deduction score if you submit the code with any other format such as .ipynb.**
   - Should develop your code **on Colab**
   - You will get **deduction score if** you submit different outputs from different environments (not Colab).

2) TEXT file (.txt)
   - Format: “Student Number_Name_EVAL.txt”.
   - You will get **deduction score if you copy and paste the results from print() function.** This means that your code has to create .txt file by file I/O mechanism as below.

Ex)
```
fw = open('./Student-Number_NAME.txt', 'w', encodig='UTF-8')
fw.write(result)
fw.close()
```

JSON Input File



An Example of Output File

Student Number_Name_EVAL.txt

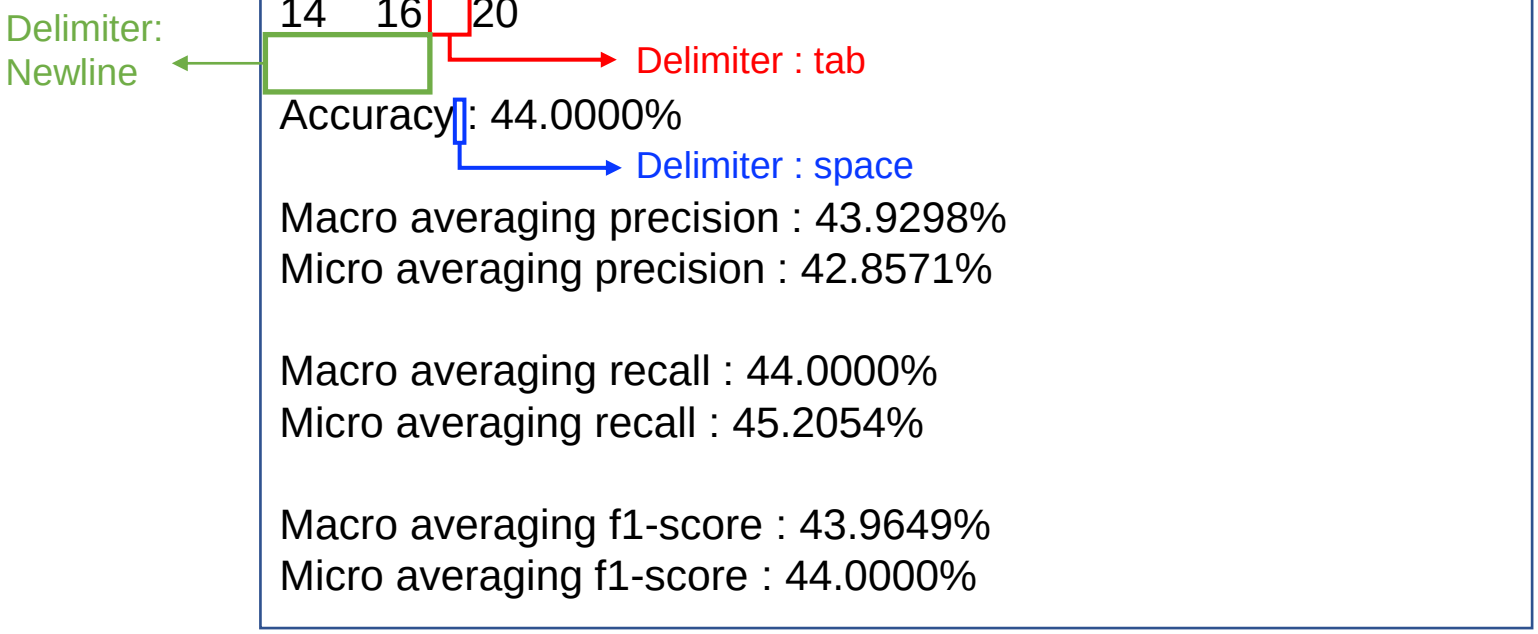
Confusion matrix
25 12 13
16 21 13
14 16 20

Accuracy: 44.0000%

Macro averaging precision : 43.9298%
Micro averaging precision : 42.8571%

Macro averaging recall : 44.0000%
Micro averaging recall : 45.2054%

Macro averaging f1-score : 43.9649%
Micro averaging f1-score : 44.0000%



- Above examples can be different with actual answers.
- Round down to the fourth digit after the decimal point.